SYSMAC
Programmable Controller
C500

OPERATION MANUAL

OMRON

Preface

The Omron C500 PC offers an effective way to automate many types of processes such as manufacturing, assembly, and packaging, to save time and money. The PC is fully equipped with facilities such as programming instructions and data areas, to control these processes directly, or remotely. Distributed control systems can be designed to allow centralized monitoring and supervision of several separate controlled systems. Monitoring and supervising can also be done through a host computer, connecting the controlled system to the data bank. Thus, it is possible to to have adjustments in system operation made automatically to compensate for requirement changes.

This manual describes the characteristics and abilities of the PC, as well as the aspects of operation and preparation that demand attention. Before attempting to operate the PC, thoroughly familiarize yourself with the information contained herein

Corrections Insert

Manual: SYSMAC Programmable Controller C500 Operation Manual

Catalog No.: W131-E1-1

Corrected Catalog No.: W131-E1-1A

Date: April 1995

This corrections insert lists corrections made to this manual since it was printed. Only corrections that affect the proper and efficient operation of the device are listed; minor changes, such as those in spelling and syntax, have been omitted.

Please mark your manuals so that the corrections are noted on the pages concerned, and then securely add any required pages from the *Corrections Insert* to the rear of the manual.

• Page 20

Add the following subsection.

1-1-1 Battery-free Operation

Operation without a battery is possible by placing the user program and I/O table data on PROM. Note the following precautions before attempting battery-free operation.

Data will be handled as described below when operating without a battery or with a discharged battery.

- HR area, T/C area (present values), and DM area value will not be preserved during power interruptions.
- SR area data will not be stable during power interruptions

To prevent problems that can occur in the above data, write the following instructions into the beginning of the user program to clear the load off control (SR 6015) and the data retention flag (SR 6012).

Address	Instruction	Operands
0000	LD	6115
0001	TCMP(95)	
		#6FFF
		60
		60
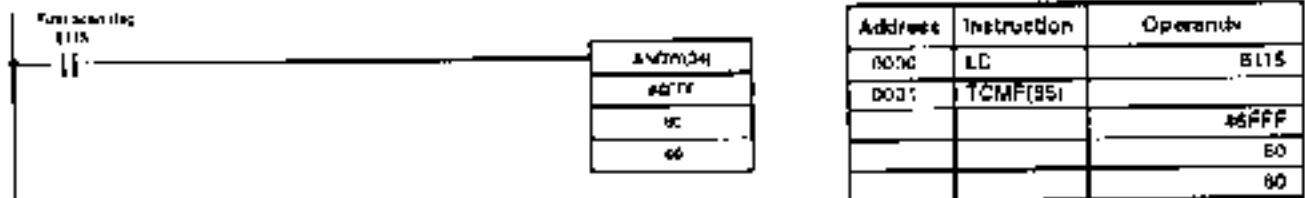
Use the following procedure to write the PROM.

1, 2, 3...
1. Write the user program to a RAM Memory Unit.
2. Register the I/O table using the procedure on page 29
3. Transfer the I/O table using the procedure on page 33. You can specify during the transfer procedure not to detect low battery voltage as an error.
4. Transfer the contents of the RAM to PROM using a PROM writer.

• Pages 83 and 86

Add the following note.

Note HR, T/C, and DM area data is backed up during power interruptions via the battery in the CPU. If this battery is missing or the voltage is low, HR, T/C, and DM area data may be lost during power interruptions.

1

• Page 184

Add the following information.

> FAL numbers are recorded in memory as they occur. The contents of the three FAL numbers in memory is as follows:
>
> 1st FAL number: First FAL number to be generated.
> 2nd FAL number: Smallest FAL number of all FAL numbers that have been generated larger than the first FAL number
> 3rd FAL number: Smallest FAL number of all FAL numbers that have been generated smaller than the first FAL number.

• Page 189

Add the following note.

> **Note** If the Intelligent I/O Unit is busy when WRIT (87) is executed, execution will not be carried out until the next scan when the Unit is not busy. Use a self-holding bit to ensure that the execution condition for WRIT (87) remains ON until it is executed.

• Page 192

Add the following note.

> **Note** If the read data is not ready or the Intelligent I/O Unit is busy when READ (88) is executed, execution will not be carried out until at least the next scan. Use a self-holding bit to ensure that the execution condition for READ (88) remains ON until it is executed.

• Pages 222 and 223

Add the following error to the end of the table. All columns not given below are blank (——) in the table and have been omitted here.

Item	Error display	Main cause of error	Corrections
Contradictory mode settings	MODE SETTING ERR	The Programming Console switch is not set for the host link when a Host Link Unit (3G2A5-LK101-(P)EV1/LK201-EV1 is set to the host link mode.	Set the Programming Console switch for the host link and then input the password.

• Page 229

Add the following CPU Backplane Power Supply to the table.

Name	Remarks		Model
CPU Backplane Power Supply	24 VDC	Output: 7 A at 5 VDC	C500-PS213-E

• Page 237

Correct the model number of the Peripheral Interface Unit to C200H-IP006.

Delete all of the GPC models except for the 3G2C5-GPC03-E.

• Page 237 and 239

Add the following information on the most current Support Software (programming software).

LSS   The most current versions of the LSS are the C500-SF711-EV3 (2D 5" disks) and the C500-SF312-EV3 (2HD 3.5" disks).

SSS   The most advanced Support Software offered by OMRON is the SYSMAC Support Software (SSS: C500-ZL3AT-E, 2HD 3.5" disks). The SSS runs on an IBM PC/AT or compatible and supports both C-series PCs and CVM1 PC.

TABLE OF CONTENTS

SECTION 3
I/O Assignments and Data Areas 69

SECTION 4
Programming Instructions 89

SECTION 5
Scan Time and I/O Response Time

SECTION 6
Error Messages and Troubleshooting

APPENDICES 229

INDEX 261

SECTION 1
Introduction to Programming

1-1
Relay Circuits:
The Roots of PC Logic

This chapter introduces the major steps involved in programming. If you are confident with relay ladder diagrams, you may skip over 1-3 Basic Programming Instructions and 1-4 Programming Techniques.

If you're an old hand at relay-based control systems, you'll find a lot that's familiar in the way the Programmable Controller (PC) works. This is because the circuits and internal logic of the PC take the place of the relays, timers, counters, and other formerly discrete devices. The actual operation of the machinery takes place as if those discrete devices were still in place, but with a great deal more flexibility and reliability.

Even though there aren't any actual discrete devices within the PC, the symbols and other control concepts used to describe their operation are still used. These are the basis of the relay ladder diagram programming method. This chapter has been written on the assumption that you are familiar with relay ladder diagrams.

Relay Terminology vs. PC Terminology

The terminology used throughout this manual is slightly different from relay terminology, but the concepts are the same. In fact the data areas are even named with relay terminology. Refer to Section 3 for details on the name and purpose of each area.

The following shows the relationship between the relay terminology you may be used to and the PC terminology used for Omron PCs.

Relay	PC
contact	input
coil	output
relay	bit

In other words, the inputs and outputs referred to in regard to programming are the bits referred to in regard to data areas. This is the same as a relay coil being an output, and the same relay's contacts being inputs (switches) for other devices. (Note that where the word "point" occurs, it refers to an actual point on a Unit attached to the PC.)

1-2
Basic Programming
Steps

To create a PC control program, follow these basic steps

1. Determine what the controlled system must do and in what order
2. Assign input and output devices to PC I/O bits. That is, designate the external devices that will send signals to and receive signals from the PC
3. Using relay ladder symbols, draw a diagram to represent the sequence of required operations and their inter-relationships.
4. If a Programming Console is used, code the ladder symbols into a list of mnemonic instructions so that the program can be input to the CPU.
5. Transfer these written instructions to the CPU via the Programming Console, the GPC, FIT, or from a host computer using LSS. (Refer to Appendix A for a brief explanation of these terms.)
6. Check for program errors.
7. Correct the errors by changing the program.
8. Execute the program and test it for execution errors.
9. Correct the execution errors by changing the program.

The remainder of Section 1 will focus on Steps 1 through 4.

1-2-1
Assessing the
Control Task

Assessing the control task is, of course, a highly important part of setting up a PC controlled system. The PC's flexibility allows a wide latitude in what operations can be controlled, and in how they can be controlled.

To apply the PC to a control task, first determine the system requirements.

Input/Output
Requirements

The first thing that must be assessed is the number of input and output points that your system will require. This is done by identifying each device that is to send an input signal to the PC or which is to receive an output signal from the PC. Each input or output point must then be assigned an I/O bit.

Keep in mind that the number of I/O bits available depends on the PC system configuration. (See 3-2 I/O and Internal Relay Area for more details.)

Sequence, Timing,
and Relationship
Assessment

Next, determine the sequence in which control operations are to occur, and the relative timing of the operations. Identify the physical relationships between the controlled devices as well as the kinds of responses that should occur between them.

2

For instance, a photoelectric switch might be functionally tied to a motor by way of a counter within the PC. When the PC receives an input from a switch, it starts the motor. The PC stops the motor when the counter has received five input signals from the photoelectric switch.

Each of the related tasks must be similarly determined, from the beginning of the controlled operation to the end.

Having made this assessment, you will be ready to go to step 2 of programming—assigning the input/output devices to I/O bits.

Input/Output Assignments

The PC uses the concept of I/O channels. An I/O channel consists of 16 bits.

The four digit number used to identify an I/O bit, also known as the address of the bit, can be broken down into two parts. The leftmost two digits identify the channel, and the rightmost two digits identify the bit within the channel. See the discussion on addressing conventions in 3-1 I/O Assignments and Data Areas.

Assigning Non-I/O IR Bits

Bits that are not used to directly send or receive signals to or from external devices function like the internal relays used in a relay control panel. They are used as data process areas in controlling other bits, timers, and counters. Assign these "internal relays" or work bits when you assign I/O bits during Step 2.

Assigning Numbers to Timers and Counters

Identify timers and counters with a number that ranges from 000 to 127. When assigning timer and counter numbers, be careful not to use the same timer/counter number for another timer/counter. For example, there cannot be a Timer 001 and a Counter 001.

When you're finished assigning the I/O bits, work bits, and timers/counters, proceed to the next step - drawing the ladder diagram.

1-2-2
<u>Drawing the Ladder Diagram</u>

Once you have determined which devices are to be controlled, how they relate to each other, and the sequence (or timing) at which the controlled tasks must take place, write a ladder diagram.

In the ladder diagram, use the four-digit addresses that you assigned to the I/O bits and work bits, as well as the three-digit numbers you gave to the timers and counters. You'll also use relay symbols such as the following.
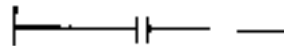
Relay ladder term: normally open contact
PC term: (normally open) input
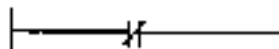
Relay ladder term: normally closed contact
PC term: (normally closed) input

Relay ladder term: coil
PC term: output

When you have finished writing your ladder diagram, the next step is to encode the diagram into a language that the PC can understand.

1-2-3
<u>Converting the Ladder Diagram Into Mnemonic Code</u>

If you are using a Graphic Programming Console, LSS, or FIT (See Appendix A), you can directly program the PC in ladder diagram logic. However, if you are using the Programming Console, you must convert the ladder diagram into mnemonic code. Mnemonic code consists of addresses, instructions, and data.

"Addresses" in this context refer to program addresses - locations in the PC's program memory where instructions and data are stored. Instructions tell the PC what to do using the operand data that follows each instruction. Each instruction is a step in the program, and address numbers provide a way to reference steps.

When programming, the addresses will automatically be displayed and do not have to be set unless for some reason a different location is desired for the program.

For example, let's consider a mini-program that ANDs two inputs

Here's the ladder diagram for this.
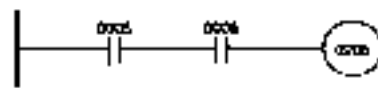
Address	Instruction	Data
0000	LD	0005
0001	AND	0006
0002	OUT	0205
0003	END (01)	—

First, indicate the beginning of the program with LD. In our ladder diagram, the bus bar represents LD. Thus, this instruction is always used when the logic line starts from the bus bar

Because the first input in the AND circuit must be stored as the data for LD, we write this down in the "data" column on our sheet. In our example, this data is 0005.

The next element of the ladder diagram is AND. The data for AND is the number assigned to the second input, in this case 0006. On our sheet we write this next to AND

Next, we need OUT to output the result of the AND'd inputs in our circuit. We write this instruction and designate the output to which we want this signal sent. We've chosen this to be 0205 and have written that as the next entry on the sheet. Any I/O bit used with OUT must correspond to an I/O Unit mounted to the PC.

Finally, we program END to tell the PC that the program is over.

1-3
Basic Programming
Instructions
(LD, OUT, AND, OR, NOT,
and END)

Except for END, each of these indispensable instructions has a corresponding
key on the Programming Console. To enter LD, OUT, AND, OR, or NOT,
simply press the appropriate key. END is programmed by pressing the FUN,
0, and 1 keys

LD and OUT

LD starts each logic line or block. When a logic line starts with an NO (nor-
mally open) input, use LD. Use OUT for outputs.

AND

This is used to serially connect two or more inputs

OR

This is used to connect two or more inputs in parallel.

NOT

This inverts its input; often used to form an NC (normally closed) input or
output. NOT can be used with LD, OUT, AND, or OR.
(NOT is also used when programming differentiated instructions. Refer to
Section 4-1-1.)

END

This Indicates the end of the program. If you forget to include END, the pro-
gram will not execute and the error message " NO END INST" will be dis-
played on the LCD of the Programming Console.

1-3-1
AND LD and OR LD

AND LD connects two blocks in series. In other words, AND LD logically ANDs
two blocks There is no limit to the number of blocks that can be connected
together in series with AND LDs.
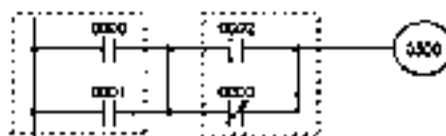
Address	Instruction	Data
0000	LD	0000
0001	OR	0001
0002	LD	0002
0003	OR NOT	0003
0004	AND LD	—

Key Sequence

The second LD is for the first input in the second block. The AND LD connects these two blocks in series. There are two ways to connect blocks in series.
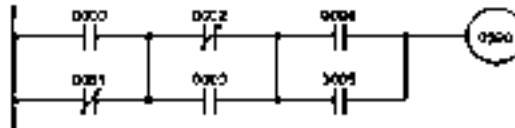
Coding Method #1

Address	Instruction	Data
0000	LD	0000
0001	OR NOT	0001
0002	LD NOT	0002
0003	OR	0003
0004	AND LD	—
0005	LD	0004
0006	OR	0005
0007	AND LD	—
	:	.
0014	OUT	0500

Coding Method #2

Address	Instruction	Data
0000	LD	0000
0001	OR NOT	0001
0002	LD NOT	0002
0003	OR	0003
0004	LD	0004
0005	OR	0005
	:	:
0011	AND LD	—
0012	AND LD	
	:	:
0014	OUT	0500

Using the first coding method, the number of AND LDs is unlimited. When AND LDs are used as in method 2, though, the total number of the LD and LD NOTs before the AND LDs must be eight or less. Therefore, if nine or more are required, use the first coding method.

OR LD

OR LD connects two blocks in parallel. In other words, OR LD logically ORs two blocks. There is no limit to the number of blocks that can be connected together in parallel with OR LDs.
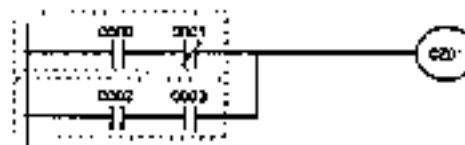
Address	Instruction	Data
0000	LD	0000
0001	AND NOT	0001
0002	LD	0002
0003	AND	0003
0004	OR LD	—
0005	OUT	0201

**Key Sequence**

The second LD is for the first input in the second block. The OR LD connects these two blocks in parallel. As in the case of AND LD, there are two ways for this to be done.
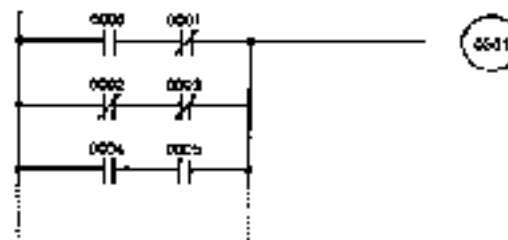
Coding Method #1

Address	Instruction	Data
0000	LD	0000
0001	AND NOT	0001
0002	LD NOT	0002
0003	AND NOT	0003
0004	OR LD	—
0005	LD	0004
0006	AND	0005
0007	OR LD	—
	:	:
0012	OUT	0501

Coding Method #2

Address	Instruction	Data
0000	LD	0000
0001	AND NOT	0001
0002	LD NOT	0002
0003	AND NOT	0003
0004	LD	0004
0005	AND	0005
	:	:
0013	OR LD	—
0014	OR LD	—
	:	:
0019	OUT	0501

Again, the same restriction applies as in the case of AND LD. Using the first coding method, the number of OR LDs is unlimited, but when OR LDs are used as in method 2. the total number of the LD and LD NOTs before the OR LDs must be eight or less. Therefore, if nine or more are required, use the first coding method.

1-4
Programming
Techniques

The number of inputs in series or parallel is unlimited. Therefore, use as many inputs as required to configure a clear circuit. Note that any inputs on bridging lines cannot be programmed and therefore must be eliminated.

The bridge circuit (1) shown below, for example, should be replaced with the second circuit (2).
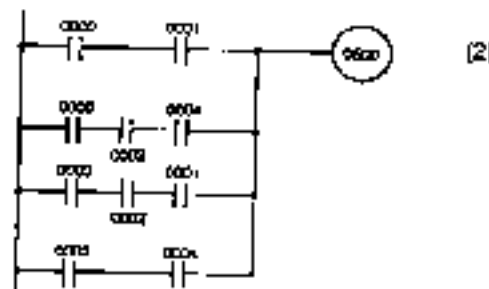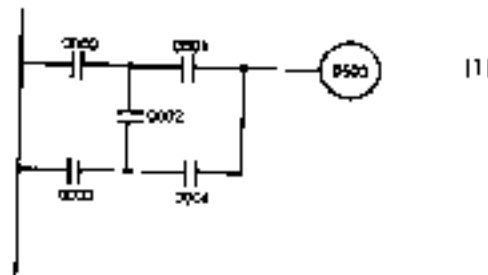
Since the number of times I/O bits, work bits, and timers/counters can be used as inputs is not limited, use them as many times as required to simplify your program. Often, complicated programs are the result of attempts to reduce the number of times a bit is used as an input.

Signals always flow from the left bus bar to the right, and the program is always scanned from the top to the bottom.

A logic line cannot start with an output. Use a normally ON SR bit if it is necessary to have an output always ON.
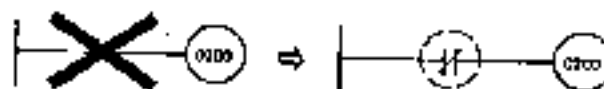
Each logic line starting from the left bus bar must end with an OUT, a timer/counter, or an instruction. The line cannot end with an input. Unlike the actual circuit diagram, the right bus bar need not be written into the ladder diagram.
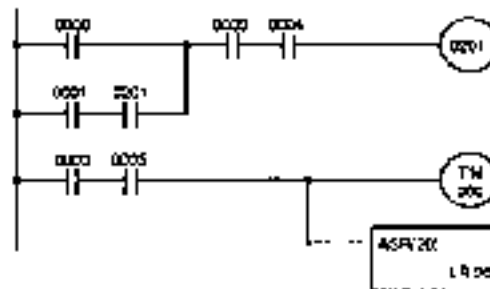
Since the timers/counters must be assigned numbers from a range of 000 to 127, a convenient way to make this assignment is to begin at one end of the range for timers and the other end for counters. This helps to prevent use of the same number for both a timer and a counter.

Timers/counters cannot directly produce an external output signal but must be programmed to an output with OUT.

The same bit number cannot be assigned as an output more than once. However, an output bit can be later used as an input as many times as desired.

The program is executed from the first program address to the first END. This feature can be used for test runs: divide the program into several blocks by inserting ENDs, then execute the program on a block-by-block basis. When the first block has been checked for correct execution, delete the first END. Continue this process until the program has been completely tested.
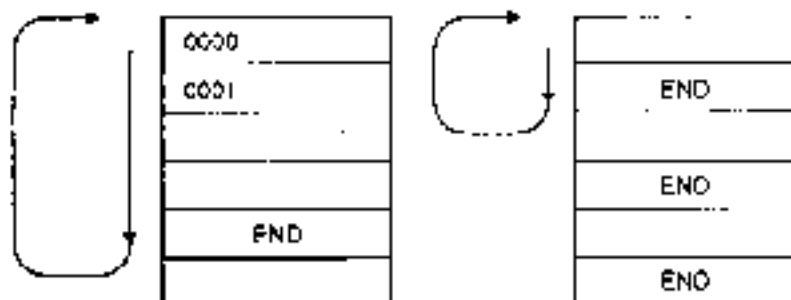
1-4-1
Program Organization
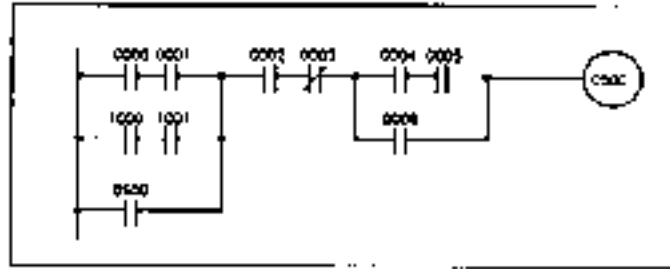
To organize the code for the circuit shown above
   (1) Divide the circuit into small blocks ([a] to [f]).
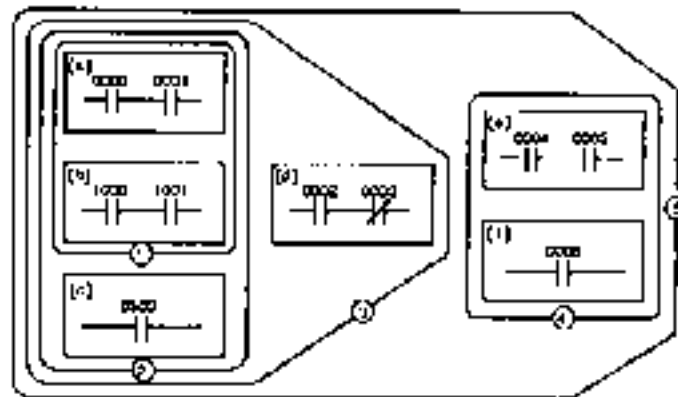
   (2) Program each block from top to bottom, then from left to
       right.
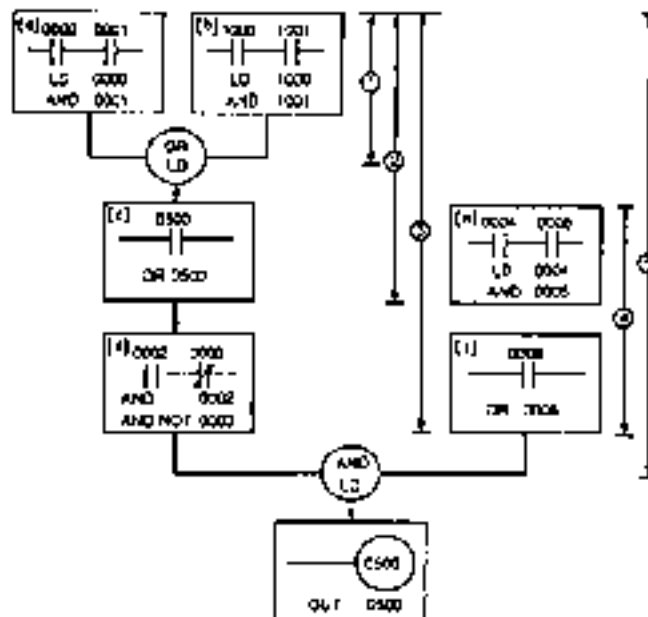
•Mnemonic Code

	Address	Instruction	Data
[a]	0000	LD	0000
	0001	AND	0001
[b]	0002	LD	1000
	0003	AND	1001
	0004	OR LD	—
[c]	0005	OR	0500
[d]	0006	AND	0002
	0007	AND NOT	0003
[e]	0008	LD	0004
	0009	AND	0005
[f]	0010	OR	0006
	0011	AND LD	—
	0012	OUT	0500

1-4-2
Programming Examples

(1) Parallel-Series Circuit

To program a parallel-series circuit, simply program the parallel circuit blocks first, and then the series circuit blocks. In the following example, first program block a, then block b
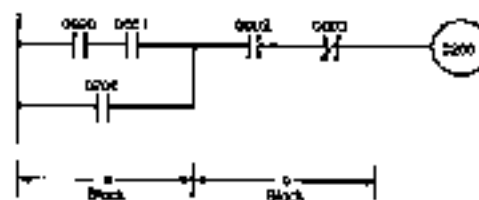
Address	Instruction	Data
0000	LD	0000
0001	AND	0001
0002	OR	0200
0003	AND	0002
0004	AND NOT	0003
0005	OUT	0200

Series-Parallel Circuit

To program a series-parallel circuit, divide the circuit into the series circuit blocks and parallel circuit blocks. Program each block and then combine the blocks into one circuit. In the following example, divide the circuit into blocks a and b, and program each block. Then combine blocks a and b with AND LD.
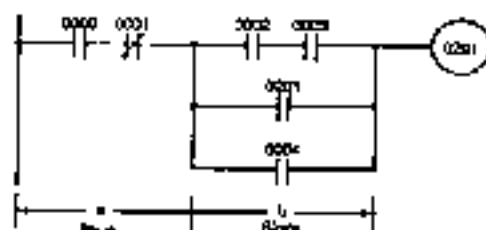
Address	Instruction	Data
0000	LD	0000
0001	AND NOT	0001
0002	LD	0002
0003	AND	0003
0004	OR	0201
0005	OR	0004
0006	AND LD	—
0007	OUT	0201

The parallel circuit block of the series-parallel circuit shown below can also be divided into two branches. In this case, program block a, and then blocks b1 and b2 in this order. Then combine blocks b1 and b2 with OR LD. Finally, combine block a and block b with AND LD.
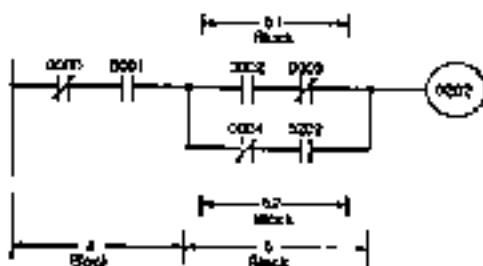
Address	Instruction	Data
0000	LD NOT	0000
0001	AND	0001
0002	LD	0002
0003	AND NOT	0003
0004	LD NOT	0004
0005	AND	0202
0006	OR LD	—
0007	AND LD	—
0008	OUT	0202

**Connecting Parallel
Circuits in Series**

To program two or more parallel circuit blocks in series, first divide the entire circuit into the parallel circuit blocks. Then subdivide each parallel circuit block into the individual blocks. Program each of the parallel circuit blocks, and then combine them in series.

In the following example, program block a1, and then block a2. Then combine both blocks with OR LD. In the same manner, program blocks b1 and b2, and combine them. Finally, combine the two parallel circuit blocks with AND LD.
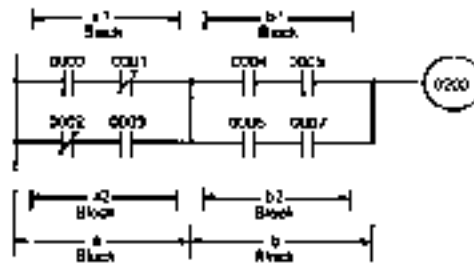
Address	Instruction	Data
0000	LD	0000
0001	AND NOT	0001
0002	LD NOT	0002
0003	AND	0003
0004	OR LD	—
0005	LD	0004
0006	AND	0005
0007	LD	0006
0008	AND	0007
0009	OR LD	—
0010	AND LD	—
0011	OUT	0200
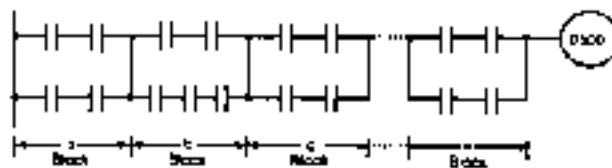
A series of blocks are programmed in the same way. That is,
a -> b -> (a • b) -> c -> (a • b • c) -> d ...

15

Complicated Circuits-A
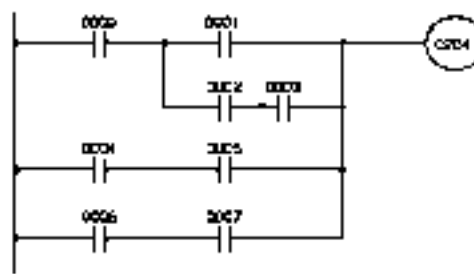
Address	Instruction	Data
0000	LD	0000
0001	LD	0001
0002	LD	0002
0003	AND	0003
0004	OR LD	—
0005	AND LD	—
0006	LD	0004
0007	AND	0005
0008	OR LD	—
0009	LD	0006
0010	AND	0007
0011	OR LD	—
0012	OUT	0204

The circuit below
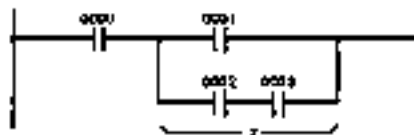
can be thought of as:
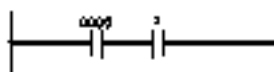
or as:
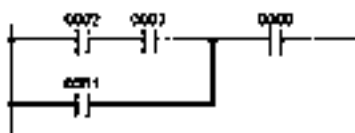
A complicated circuit can often be simplified by rewriting.

Complicated Circuits-B
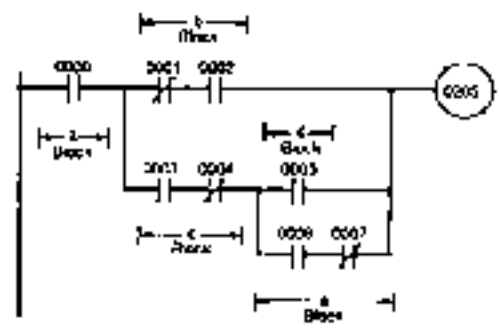
Address	Instruction	Data
0000	LD	0000
0001	LD NOT	0001
0002	AND	0002
0003	LD	0003
0004	AND NOT	0004
0005	LD	0005
0006	LD	0006
0007	AND NOT	0007
0008	OR LD	—
0009	AND LD	—
0010	OR LD	—
0011	AND LD	
0012	OUT	0205

The circuit above can be rewritten as.
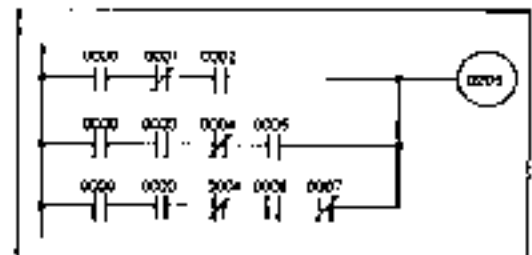
Complicated Circuits C
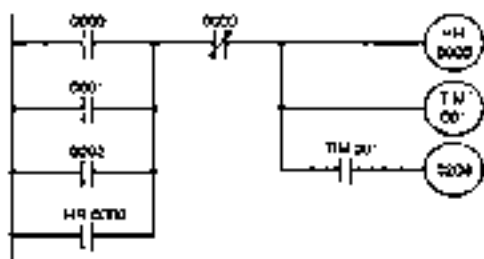
Address	Instruction	Data	
0000	LD		0000
0001	OR		0001
0002	OR		0002
0003	OR	HR	0000
0004	AND NOT		0000
0005	OUT	HR	0000
0006	TIM		001
		#	0100
0007	AND	TIM	001
0008	OUT		0200

Note: A bit in the Holding Relay (HR) area retains its status when a power failure occurs.

Problem Circuits to be Avoided
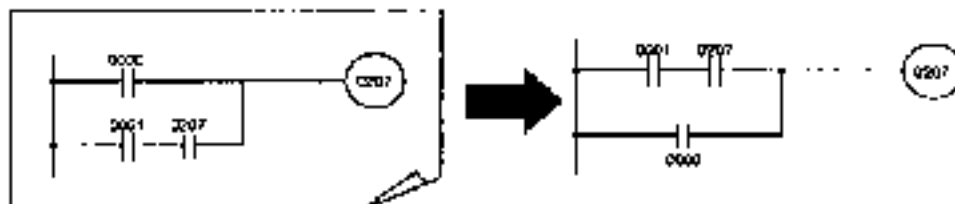
The circuit on the left requires one more program step (an OR LD) than the circuit on the right. By rewriting this code, scan time is saved and program memory space is used more efficiently.
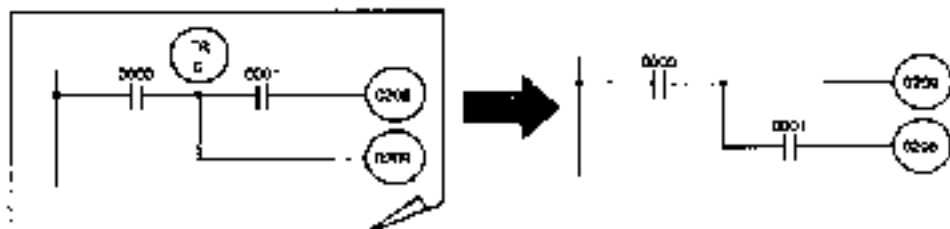
The circuit on the left requires an extra Temporary Relay Area (TR) bit and one more program step (LD) than the circuit on the right. Again, by rewriting this code, scan time is saved and program memory space is used more efficiently.
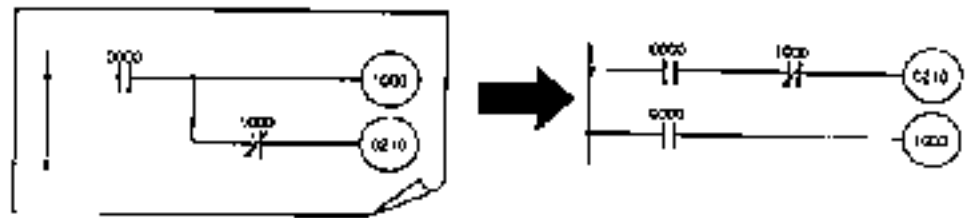
Output 0210 in the circuit on the left cannot be turned ON because of the order in which the PC executes the instructions. By rewriting this code, output 0210 can be turned ON
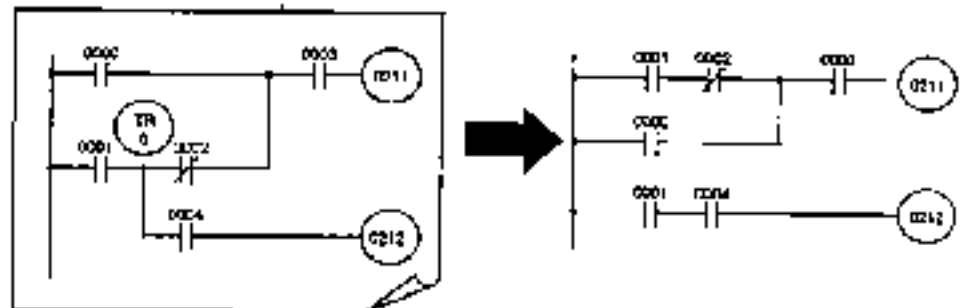
In a circuit with a branch like the one on the left where another input is included, either a TR bit must be used or the circuit must be rewritten as shown on the right.
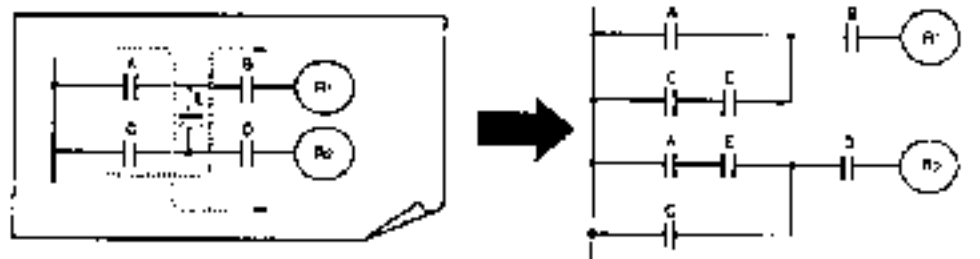
The circuit on the left cannot be programmed. In order to make the signals flow in the directions indicated by the dotted lines, rewrite the circuit as shown on the right.

SECTION 2
Using the Programming Console

This section focuses on how to use the Programming Console to prepare the system for programming, to enter program data, and to monitor system operations and program execution. If you are not using a Programming Console, you can skip this section.

Note: Any of the Programming Console operations described in this section can be cancelled at any time by pressing the CLR key. In some cases, the CLR key may need to be pressed 2 or 3 times.

2-1
The Programming Console

The Programming Console is the most commonly used programming device for the C500 PC. It is a compact device that is available either as a hand-held model or for direct mounting to the PC.

Ladder logic program instructions cannot be directly input through the Programming Console. There are, however, other programming devices available as listed in Appendix A.

Refer to each programming device Operation Manual for details about its operations.

2-1-1
The Keyboard

The keyboard of the Programming Console is functionally divided by key color into the following four areas:

White Numeric Keys

These ten keys are used to input numeric program data such as program addresses, input/output bit numbers and values, and timer/counter numbers and values.

The numeric keys are also used in combination with the function key (FUN) for entering instructions with function codes.

Red CLR Key

This key clears the display and cancels current Programming Console operations. It is also used when you key in the password at the beginning of programming operations.

Yellow Operation Keys

These yellow keys are used for writing and correcting programs. Detailed explanations of their functions are given later in this section.

Gray Instruction Keys

Except for the SHIFT key on the upper right, these gray keys are the ones you'll use to insert instructions into your program. The SHIFT key is similar to the shift key of a typewriter, and is used to obtain the second function of those keys that have two functions.

The remaining gray keys have mnemonic names. The functions of these keys are described below.

FUN Used to select and enter instructions with function codes. To enter an instruction with function code, press the FUN key and then the appropriate numerical value. Instructions and their function codes are listed in Appendix C

SFT Enters a shift register instruction.

NOT Inverts the instruction before it. Often used to form a normally closed input or output.

AND Enters a logical AND instruction.

OR Enters a logical OR instruction

CNT Enters counter instructions. After CNT, enter the counter and data

LD ⊣⊢ Enters load instructions.

OUT -O- Enters output instructions.

TIM Enters timer instructions. After TIM, enter the timer data.

TR Used to specify a TR bit.

LR Used to specify the LR area.

HR Used to specify the HR area.

DM Used to specify the DM area.

CH ± Used to specify a channel.

CONT # Used to search for a bit.

2-1-2
The Mode Switch

To select one of three operating modes — RUN, MONITOR, or PROGRAM — use the mode switch.

In RUN mode, programs are executed. When the PC is switched into this mode, it begins controlling equipment according to the program instructions written in its program memory.

Note: Do not leave the Programming Console connected to the PC by an extension cable when in RUN mode

MONITOR mode allows you to visually monitor in-progress program execution. For instance, if you want to check that a particular input bit is in the correct state at the right time, you can move to the program address (or step) that references that input bit. In MONITOR mode, I/O processing is handled in the same way as in RUN mode.

In PROGRAM mode, the PC does not execute programs. PROGRAM mode is for creating and changing programs, clearing program memory, and registering the I/O table.

Mode Changes   The following situations cause the PC mode to change:

(1) Peripherals not connected
When power is applied to the PC without a peripheral device connected, the PC is automatically set to RUN mode.

(2) Peripherals connected
If the Programming Console is connected to the PC when power is applied, the PC is set to the mode indicated by the Programming Console's mode selector. To be on the safe side, make sure that the PC is in PROGRAM mode when first applying power, in case there is an unknown program in program memory.

If a device such as a Peripheral Interface Unit, P-ROM Writer, or a Printer Interface Unit is attached to the PC when the power is turned on, the PC is automatically set to PROGRAM mode

Note: If the PC power supply is already turned on when any peripheral device is attached to the PC, the PC stays in the same mode it was in before the peripheral device was attached. The mode can be changed, though, if the Programming Console is attached, with the MODE selector on the front panel of the Programming Console. If it is necessary to have the PC in PROGRAM mode, (for the P-ROM Writer, Printer Interface Unit, etc.), be sure to select this mode before connecting the peripheral device, or alternatively, apply power to the PC after the peripheral device is connected.

2-1-3
The Display Message Switch

On the rear of the Programming Console case, on the righthand side of the external connector, there is a small switch for selecting either Japanese or English language messages for display on the console. It is factory set to OFF which causes English language messages to be displayed.

2-2
Preparation for Programming

The following sequence of operations will be performed before beginning actual program input and execution.

Set mode selector to PROGRAM mode.
Enter password.
Clear program memory.
Register the I/O table
Perform checks until all errors are eliminated.

Each of these operations is described in detail in the following subsections. Except for password entry, all of the other operations are regularly used Programming Console operations. All operations should be done in PRO-GRAM mode unless otherwise noted

2-2-1
Entering the Password

To gain access to the PC's programming functions, you must first enter the password. The password prevents unauthorized access to the program.

The PC prompts you for a password when power is first applied to the PC or after the Programming Console has been connected to the PC. To gain access to the system when the "Password!" message appears on the console, press the CLR and MONTR keys.

CLR | ⟨PROGRAM⟩
         | PASSWORD !

MONTR | ⟨PROGRAM⟩   BZ

Indicates the mode set by the mode selector switch.

Note: The Programming Console displays the current mode in angle brackets (⟨⟩). Be sure that the PC is in PROGRAM mode before you enter the password. Then, after you enter the password, you can change the mode to RUN or MONITOR with the mode selector.

2-2-2
Clear Memory

Using the clear operation it is possible to clear all or part of the IR, HR, DM and TC areas. Unless otherwise specified, the clear operation will clear all memory areas above provided that the Memory Unit attached to the PC is a RAM Unit or an EEP-ROM Unit and the write-enable switch is ON. If the write-enable switch is OFF, or the Memory Unit is a ROM Unit, program memory is **not** cleared

Before beginning to program or when installing a new program, clear all areas.

Key Sequence
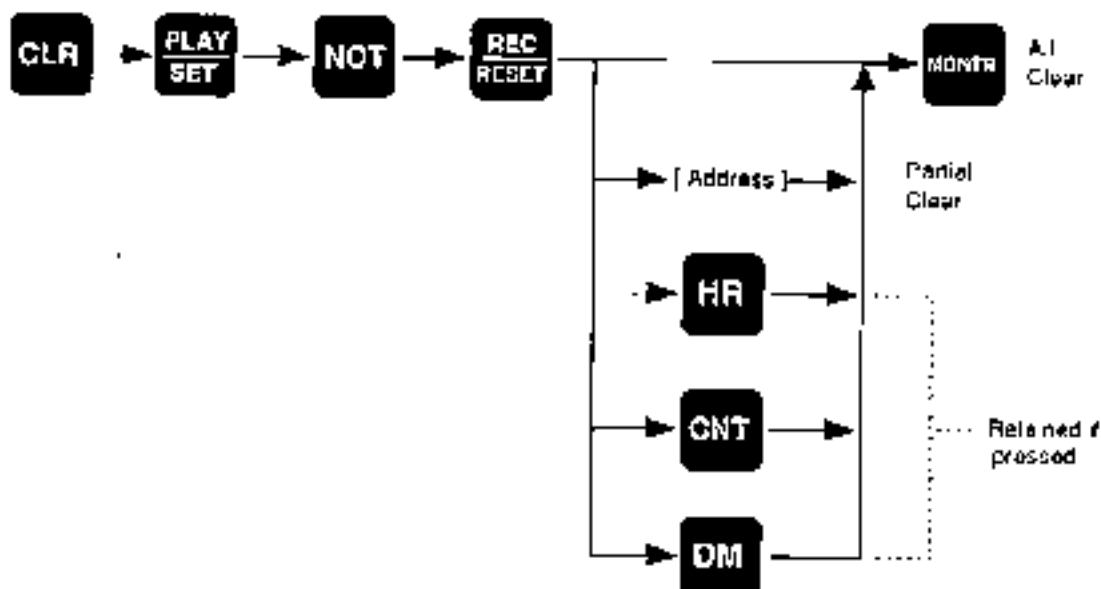
Initial Clear All
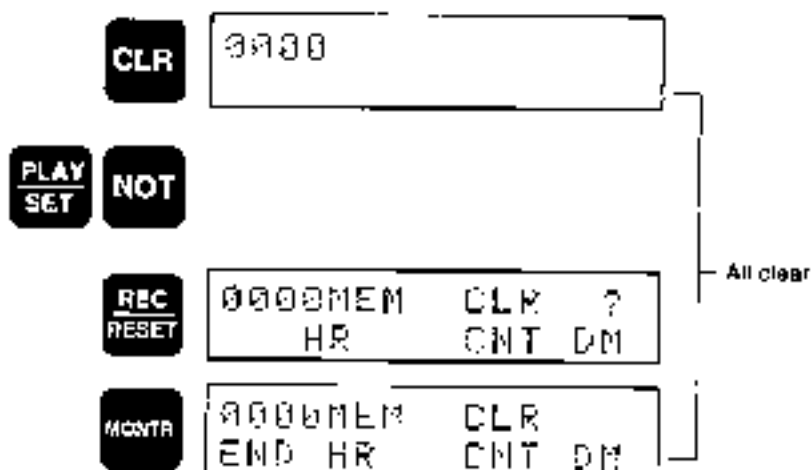
It is possible to retain the data in specified areas when clearing memory. To retain the data in any of HR, TC, or DM press the appropriate key after entering REC/RESET.

It is also possible to retain a portion of the program memory from the beginning to a specified address. After pressing the REC/RESET key, specify the last address to be retained.

For example, to leave the program data from 0000 to 0122 untouched, but to clear the addresses from 0123 to the end of program memory, key in the address 0123 after pressing the REC/RESET key.

Note: 1. A warning will sound when the memory clear operation is begun.
2. If a mistake is made during input, repeat the operation from CLR

Example:
Leaving the TC area
uncleared and retaining
memory up to address 0122

CLR	0000
PLAY SET	0000
NOT	0000
REC RESET	0000MEM CLR ? HR CNT DM
CNT	0000MEM CLR ? HR DM
1 2 3	0123MEM CLR ? HR DM
MONTR	0000MEM CLR END HR DM

27

2-2-3
Registering the I/O Table

The I/O Table Registration operation writes the types of I/O Units controlled by the PC and the rack locations of the I/O Units into the I/O table memory area of the CPU. It also clears all I/O bits. The I/O table must be registered before programming operations are begun. A new I/O table must also be registered whenever I/O Units are changed because the previous I/O table remains in memory.

I/O Table Registration can be performed only in PROGRAM mode.

The I/O verification error message, "I/O VER ERR," will appear when starting programming operations or after I/O Units have been changed. This error is cleared by registering a new I/O table

Initial I/O Table Registration
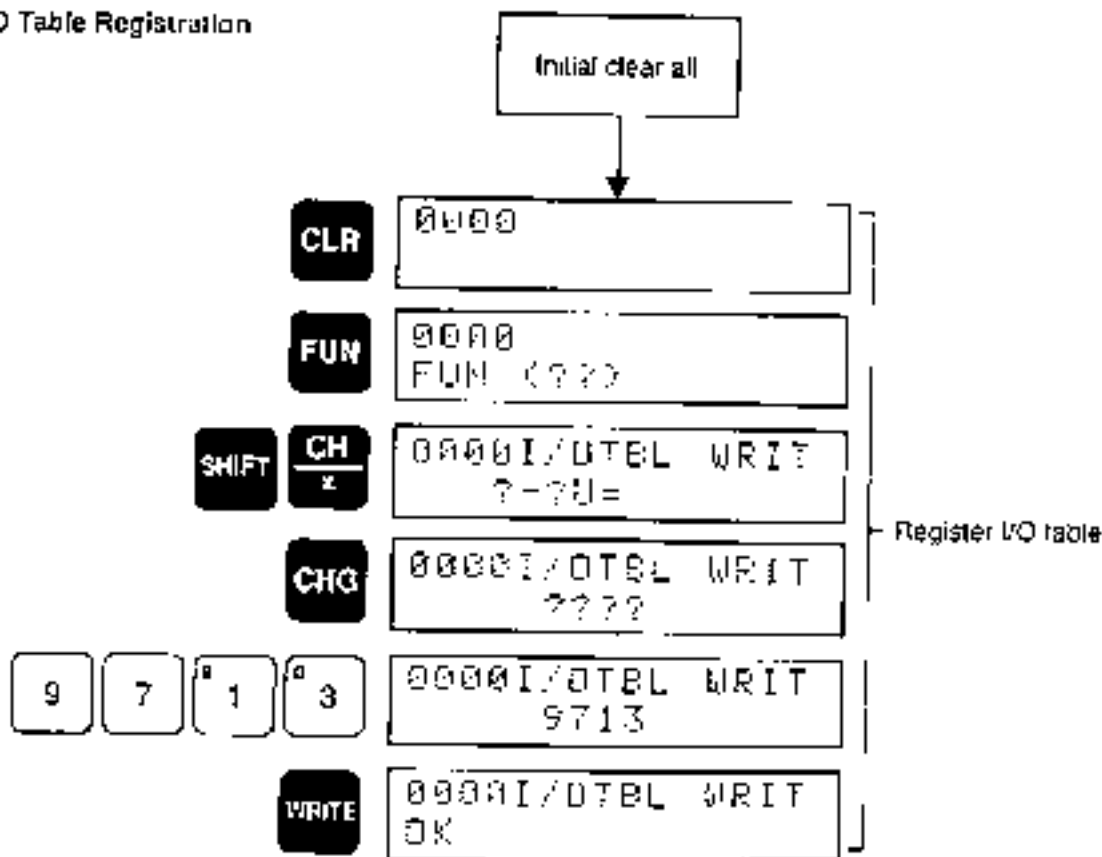
2-2-4
Verifying the I/O Table

The I/O Table Verification operation is used to check the I/O table registered in memory to see if it matches the actual sequence of I/O Units mounted. The first inconsistency discovered will be displayed as shown below. Every subsequent pressing of the VER key displays the next inconsistency.

Key Sequence


```
CLR      0000
```

```
FUN      0000
         FUN (??)
```

```
SHIFT  CH/x    0000I/OTBL?
                 ?-/U=
```

```
VER      0000I/OTBL  VER
         OK                    (No errors)
```

```
VER      0000I/OTBL  VER
         0-1U=0***  I***       (An error occurred)
```

— Actual I/O arrangement.
— Registered I/O table.
— I/O slot no.
— Rack no.
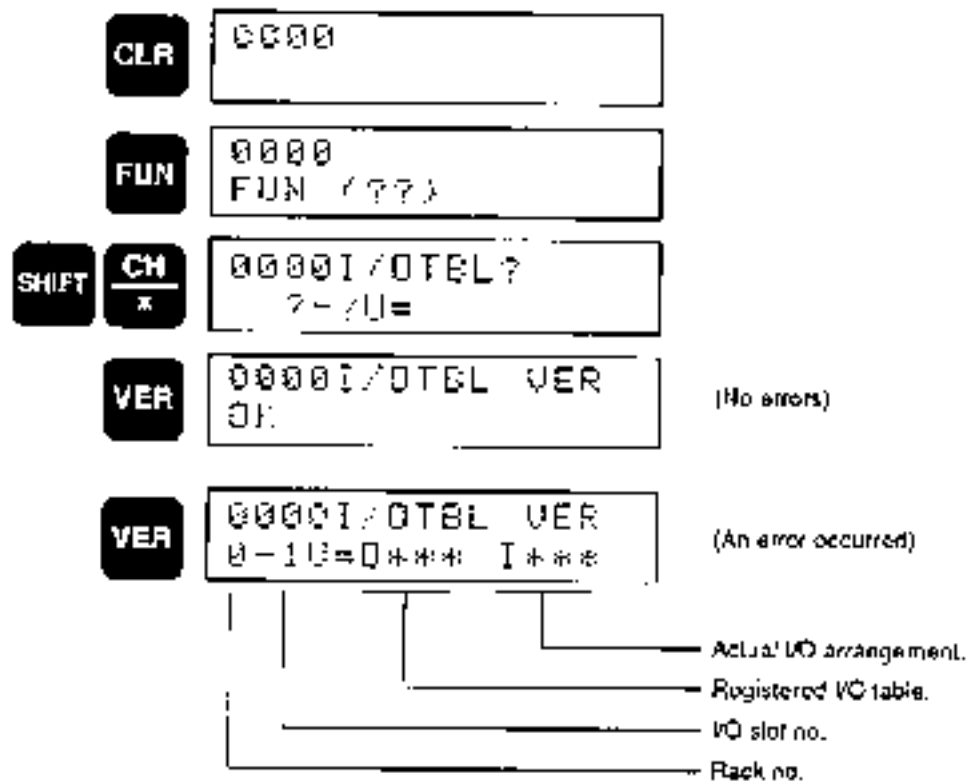
Meaning of Displays

Optical Transmitting I/O Unit no. Error
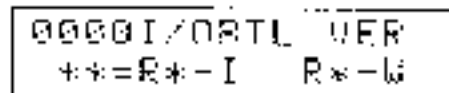
```
0000I/ORTL  VER
**=R*-I    R*-U
```
— Duplication

Remote I/O Error

```
0000I/OBTL  VER
*-*U=****   RMT*
```
— This is a Remote I/O Unit that has not been registered.

2-2-5
<u>Reading the I/O Table</u>

The I/O Table Read operation is used to access the I/O table that is currently registered in the CPU memory.

Example of I/O Unit Mounting

Key Sequence

CLR
```
0000
```

FUN
```
0000
FUN (??)
```
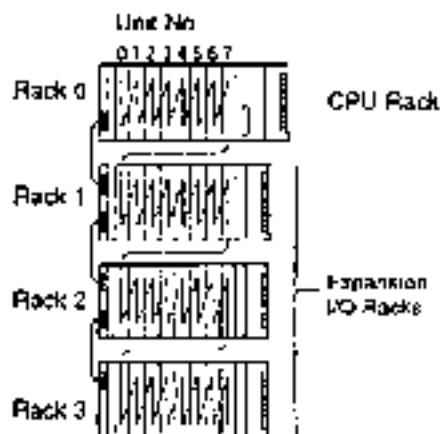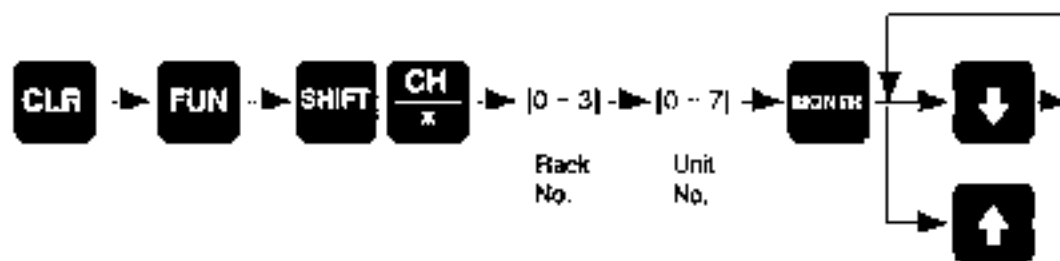
SHIFT **CH/#**
```
0000I/OTBL  ?
  ?-?U=
```

[▲ D]
```
0000I/OTBL  ?
  0-?U=
```

[▼ 5]
```
0000I/OTBL  READ
  0-5U=
```

MONTR
```
0000I/OTBL  ?
  0-5U=I***    05
```

↑
```
0000I/OTBL  READ
  0-4U=I***    04
```

↓
```
0000I/OTBL  READ
  0-5U I***    05
```

Meaning of Displays

•I/O Unit Designations for Displays

No. of Points	Input Unit	Output Unit
16	I***	O***
32	II**	OO**
64	IIII	OOOO

•I/O Units, Special I/O Units,
 I/O Link Units

```
00001/OTBL  READ
  *-*U=:::::    ***
```

——————— I/O channel no.

——————— I/O Unit type: I (input)
           or O (output).

——————— Unit no

——————— Rack no.

•Remote I/O Master Unit

```
00001/OTBL  READ
  1-4U=RMT*
```

——————— Master no. (0-3)

•Remote I/O Slave Units

```
00001/OTBL  READ
R::::-*U=*****    ***
```

——————— I/O channel no.

——————— I/O Unit type: I (input) or
           O (output)

——————— I/O slot no.

——————— Remote I/O Slave Unit no. (0-1).

——————— Remote I/O Master Unit no. (0-3).

——————— Indicates a Remote I/O Unit.
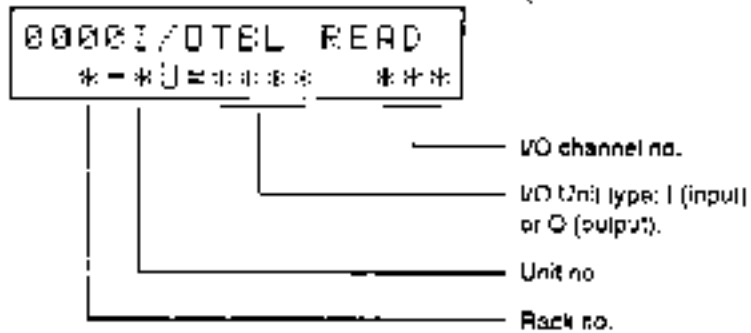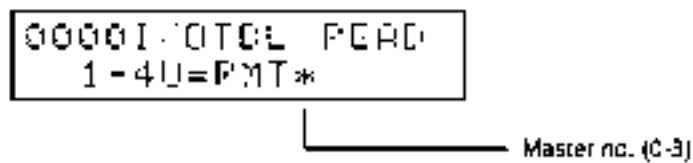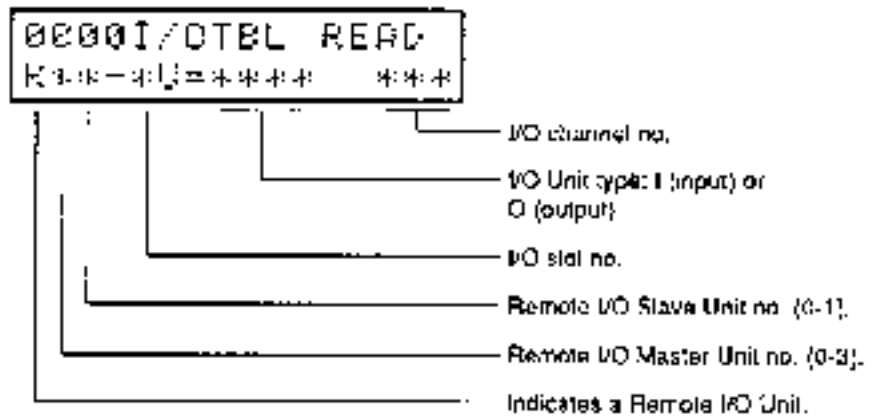
•Optical Transmitting I/O
 Units, I/O Link Units, and
 Remote Terminals

```
00001/OTBL  READ
  ***HU-R***
```

——————— I/O Unit type: I (input)
           or O (output).

——————— Remote I/O Master no. (0-3).

——————— Channel (H,L).

——————— I/O Channel no. (0-31).

32
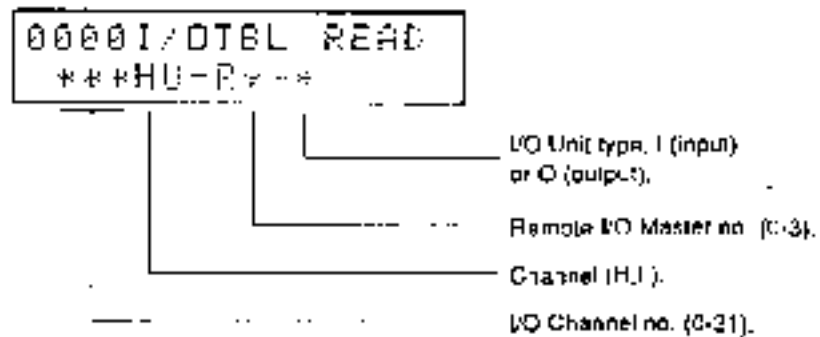
2-2-6
Transferring the I/O Table

The I/O Table Transfer operation transfers a copy of the I/O table to RAM program memory to allow the user program and I/O table to be written together into EP-ROM.

Note: When power is applied to a PC which has a copy of an I/O table stored in its program memory, the I/O table of the CPU will be overwritten. Changes made in the I/O table do not affect the copy of the I/O table in program memory; I/O Table Transfer must be repeated to change the copy in program memory.

The I/O Table Transfer operation will not work in the following cases:
1. When the memory unit is not RAM.
2. If there is less than 0.2 KB remaining in program memory.
3. If the P-ROM Writer back-panel DIP switch is not set for the C500 (all four pins OFF).

This operation can be done only in PROGRAM mode.

Key Sequence

Example

CLR

```
0000
```

FUN

```
0000
FUN (???)
```

SHIFT **CH/×**

```
0000I/OTBL
  ?-?1=
```

SHIFT **EXT**

```
0000I/OTBL→   UM
          ????
```

9 **7** **1** **3**

```
0000I/OTBL→   UM
          9713
```

WRITE

```
0000I/OTBL→   UM
OK
```

The following indicates the I/O table cannot be transferred

WRITE

```
0000I/OTBL→   UM
DISABLED
```
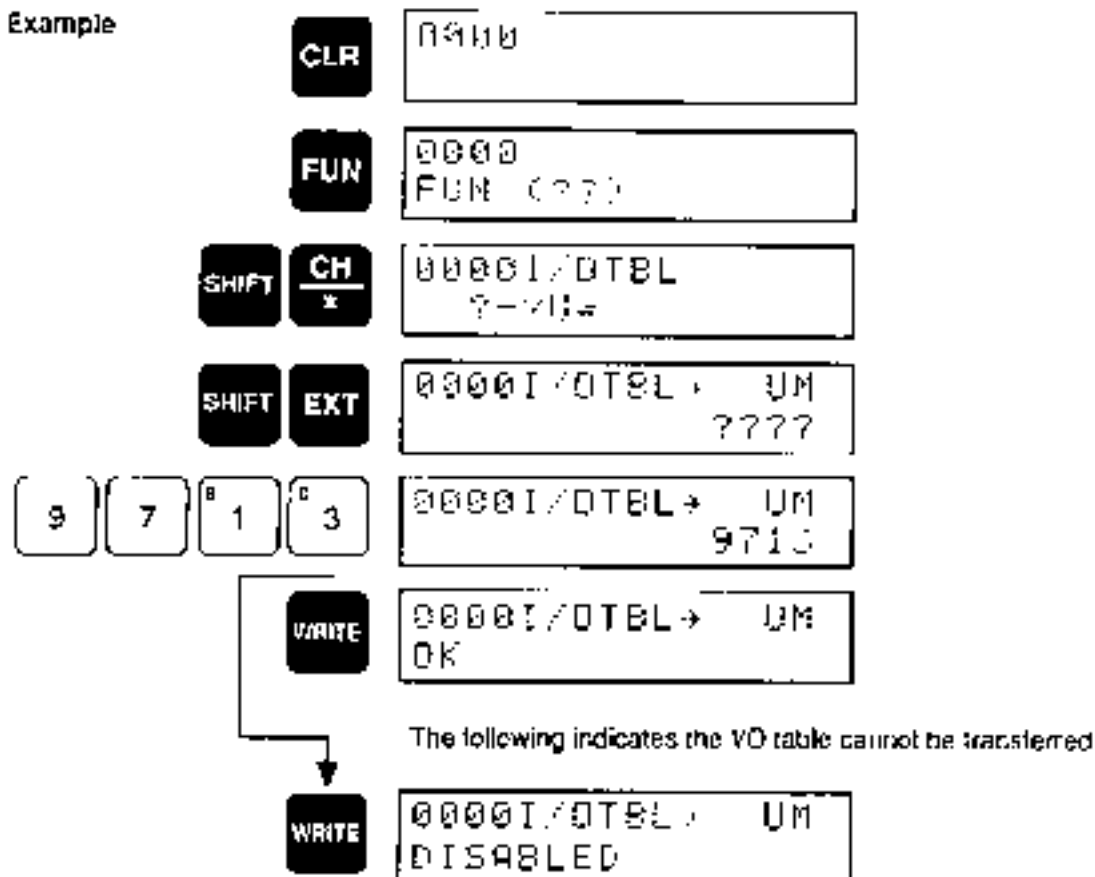
The I/O Table Transfer operation can be performed by inputting 9712 instead
of 9713. If it is, the diagnostic check for the battery will not be carried out and
the contents of the HR, TC, and DM areas will not be preserved.

2-3
Programming Operations

The Programming Console operations described in this section can be cancelled by pressing the **CLR** key

2-3-1
Setting a Program Address

To write, insert, read, or delete program instructions, you must first specify the address at which to read or make changes.

Leading 0s of the address expression need not be keyed in. That is, when specifying an address such as 0053 you need to enter 53 only

After specifying the address, press the down-arrow key once to display the contents of the address.

Key Sequence

CLR ➞ [Address]

Example

CLR | 0000 |

| a 1 | 2 | 3 | c 4 | | 1234 |

⬇ | 1234 |
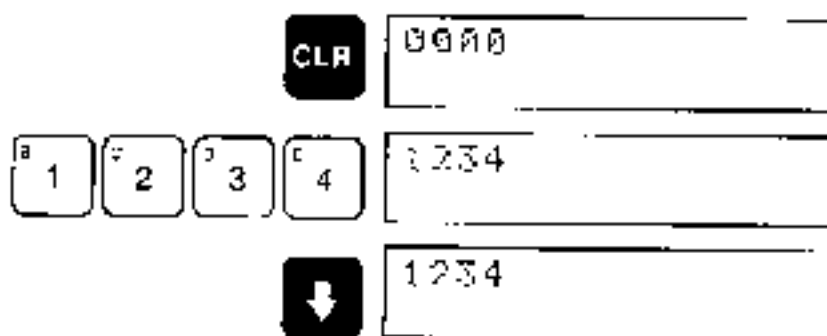
2-3-2
Program Read

To read out program data from the program memory, specify the address from which to read, then press the down-arrow key.

The down-arrow key must be pressed once to display the contents of a specified address. Then after the specified address contents are displayed, the up and down-arrow keys serve as decremental and incremental data-read keys, respectively. That is, the up-arrow key will display the contents of (current address - 1) and the down arrow key, the contents of (current address + 1)

Key Sequence
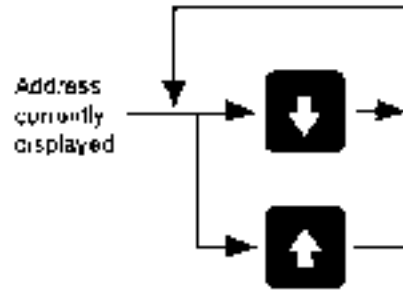
Example

Reading a segment of program code such as the following would result in Programming Console displays that show the program address, instruction, and data for each step that is read.

Address	Instruction	Data
0000	LD	0000
0001	AND	0001
0002	TIM	000
		# 0123
0003	LD	0100

2-3-3
Instruction Search

To search for specific instructions in program memory, first either set a specific address (see 2-3-1 Setting a Program Address) or read through the program (see 2-3-2 Program Read) to the address from which the instruction is to be searched for. Then, specify the particular instruction you wish to search for and press the SRCH key.

This operation can be performed in RUN, PROGRAM or MONITOR mode. While the second LD is being searched for in the example below, the message

```
0200SRCH·G
LD              0000
```

is displayed.

If the SRCH key is pressed continuously, all the addresses having the specified instruction are successively displayed until either END or the last program memory address is encountered.

To search for the set value of a timer or a counter, first search for TIM or CNT and then use the down-arrow key to access the set value. Note that pressing any key other than SRCH terminates the search operation.

Key Sequence

(To search for the set value of
a multiword instruction)

37
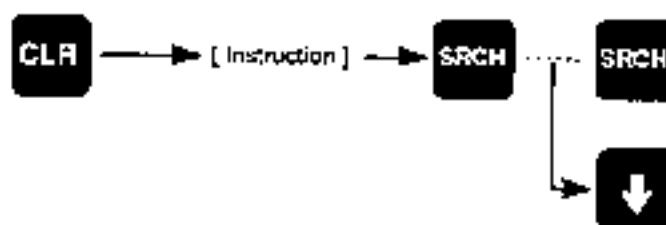
Example
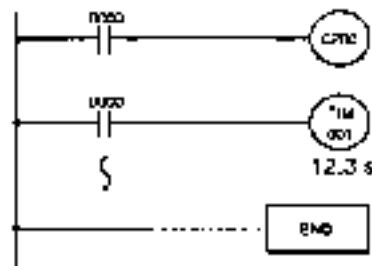
Address	Instruction	Data
0000	LD	0000
0001	OUT	0200
0002	LD	0000
0003	TIM	001
		# 0123
	:	:
6000	END[01]	—

| CLR | 0000 |

| LD HR | 0000
LD   0000 |

| SRCH | 0200SRCH
LD   0000 |

| SRCH | 0202SRCH
LD   0000 |

| SRCH | 6000SRCH
END (01)(06.4KW) |

(Multiword instruction search)

| CLR | 0000 |

| 1 0 0 | 0100 |

| TIM 1 | 0100
TIM   001 |

| SRCH | 0203SRCH
TIM   001 |

| ↓ | 0203 TIM DATA
#0123 |

2-3-4
Bit Search

The bit search operation is very similar to the monitor operations described in 2-4 Monitor Operations. To monitor or search, the bit is first specified and then the operation (either monitor or search). To specify an IR, SR, HR, or LR bit (see Section 3 I/O Assignments and Data Areas) or timer/counter within the program, use the SHIFT and CONT/# keys. Then to search, press SRCH.

Key Sequence

Example

In the preceding example, the **CONT/#** key is used in combination with the SHIFT key to specify an IR bit. Then, using the SRCH key, the rest of the program is searched for instructions that use IR bit 0005. When the PC is in the process of searching for another instruction that uses IR bit 0005, the display appears as:

```
0000SRCH^G
CONT        0005
```

2-3-5
Instruction Insert

This operation is used to change a program by inserting an instruction. Instructions cannot be inserted into a program during RUN or MONITOR mode

Key Sequence

Locate position in program then enter:

[Instruction] ⟶ **INS** ⟶ ⬇

Example
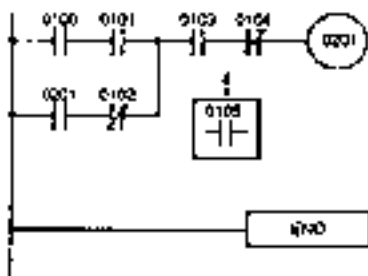
Before Insertion

Address	Instruction	Data
0000	LD	0100
0001	AND	0101
0002	LD	0201
0003	AND NOT	0102
0004	OR LD	—
0005	AND	0103
0006	AND NOT	0104
0007	OUT	0201
0008	END(01)	—

After Insertion

Address	Instruction	Data
0000	LD	0100
0001	AND	0101
0002	LD	0201
0003	AND NOT	0102
0004	OR LD	—
0005	AND	0103
0006	AND	0105
0007	AND NOT	0104
0008	OUT	0201
0009	END(01)	—

CLR | 0000

OUT
-O- | 0000
OUT 0000

[2] [0] [1] | 0000
OUT 0201

SRCH | 0207SRCH
OUT 0201

Find the address before which you want to insert.

[↑] | 0206READ
AND NOT 0104

AND
-| |- | 0206
AND 0000

[1] [0] [5] | 0206
AND 0105

Insert the instruction.

INS | 0206INSERT?
AND 0105

[↓] | 0207INSERT END
AND NOT 0104

[↑] | 0206READ
AND 0105

2-3-6
<u>Instruction Delete</u>
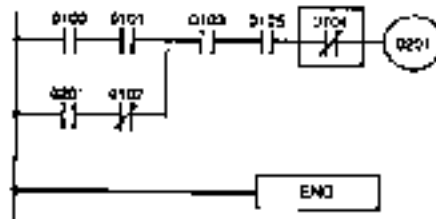
This operation is used to change a program by deleting an instruction.

Instructions cannot be deleted from a program when in RUN or Monitor mode.

Key Sequence

Instruction
currently ⟶ **DEL** ⟶ ⬆
displayed

Example

Before Deletion

Address	Instruction	Data
0000	LD	0100
0001	AND	0101
0002	LD	0201
0003	AND NOT	0102
0004	OR LD	—
0005	AND	0103
0006	AND	0105
0007	AND NOT	0104
0008	OUT	0201
0009	END(01)	—

After Deletion

Address	Instruction	Data
0000	LD	0100
0001	AND	0101
0002	LD	0201
0003	AND NOT	0102
0004	OR LD	—
0005	AND	0103
0006	AND	0105
0007	OUT	0201
0008	END(01)	—

Key	Display
CLR	`0000`
OUT -O+	`0A00` `OUT          0000`
2 0 1	`0000` `OUT          0201`
SRCH	`0200SRCH` `OUT          0201`
↑	`0207READ` `ANDNOT       0104`
DEL	`0207DELETE?` `ANDNOT       0104`
↑	`0207DELETE  END` `OUT          0201`
↑	`0206READ` `AND          0105`

Find the instruction which you want to delete.

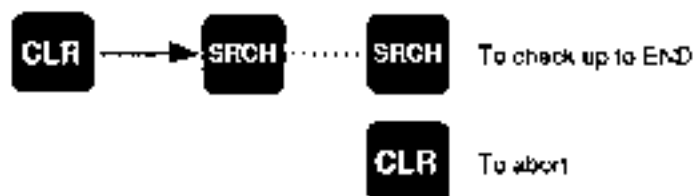Make sure that this is the instruction to be deleted.

When you delete an instruction, you must first read it before deleting it. The actual deletion is accomplished by pressing the up-arrow key.

The program addresses following the deleted instruction are automatically decremented. So, after a deletion, the next address will be displayed and can be easily deleted. In this way, a number of instructions can be quickly deleted by repeatedly pressing the up-arrow key.

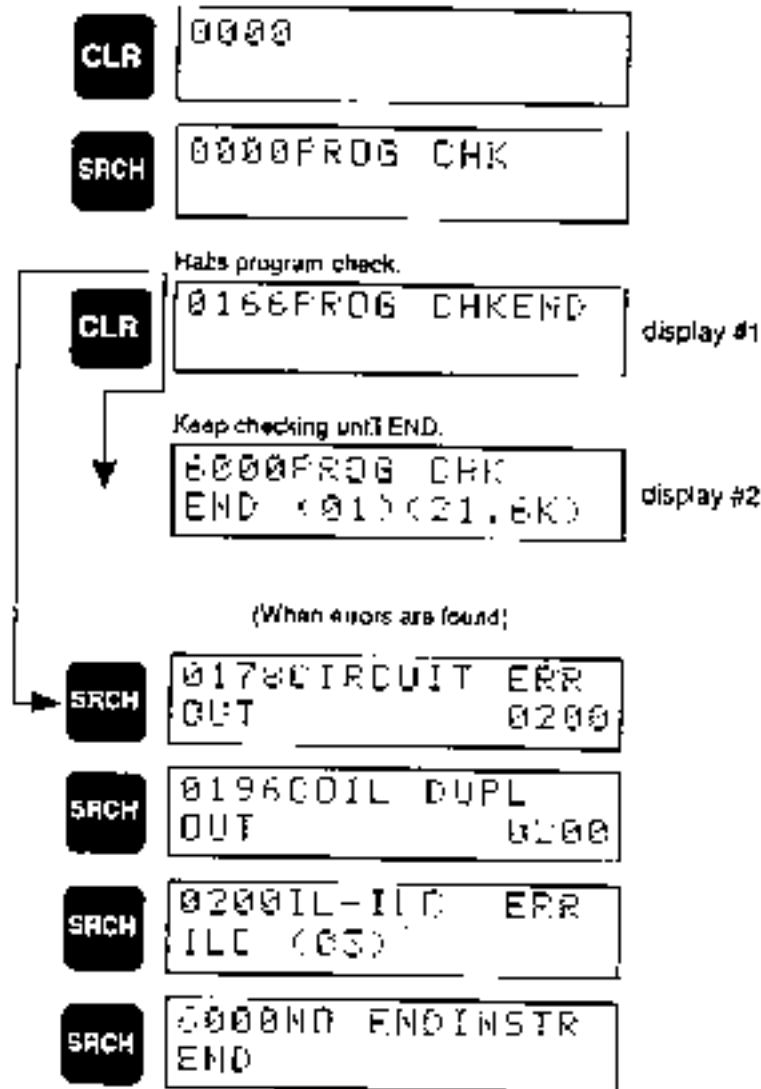Be careful not to inadvertently delete instructions.

43

2-3-7
<u>Program Check</u>

This operation does a syntax check on a program. When a program has been changed in any way, it should first be checked for programming errors before execution. A program can be checked only in PROGRAM mode.

Key Sequence

To check up to END

To abort

Note: Refer to 6-5 Program Errors for the error displays and their meanings.

Example

CLR | `0000`

SRCH | `0000PROG CHK`

Halts program check.

CLR | `0166PROG CHKEND` — display #1

Keep checking until END.

`0000PROG CHK`
`END  <01><21.6K>` — display #2

(When errors are found)

SRCH | `0178CIRCUIT ERR`
`OUT        0200`

SRCH | `0196COIL DUPL`
`OUT        0200`

SRCH | `0200IL-ILC  ERR`
`ILC <03>`

SRCH | `0000ND ENDINSTR`
`END`

Pressing the CLR key during a program check cancels the check (display #1).

Use the SRCH key to display the addresses of offending instructions and the data they contain. To successively check the program up to END, hold down the SRCH key (display #2).

2-3-8
Scan Time Read

To display the current scan time, enter CLR then MONTR. See Section 5 Scan Time and I/O Response Time for details

Note that the time displayed by this operation is an average scan time. The differences in displayed values depend on when the MONTR key is pressed.

This operation is available only when the PC is in RUN or MONITOR mode. Also, the scan time will not be displayed if the program is stopped.

Example

CLR	0000

MONTR	0000SCAN TIME 054.1MS

MONTR	0000SCAN TIME 053.9MS

2-3-9
Error Message Read

When an error occurs during program execution, it can be displayed for identification by pressing CLR, FUN, and then MONTR. If an error message is displayed, press the MONTR key to clear the error. Sometimes a beeper will sound and the error cannot be cleared. If this happens, take the appropriate corrective action (see Section 6 Error Messages and Troubleshooting) to eliminate the error.

When several errors occur, the respective error messages can be displayed by pressing the MONTR key. The sequence in which error messages are displayed depends on the priority levels of the errors. The following highest priority errors cause the CPU to halt.

> MEMORY ERR
> NO END INST
> I/O BUS ERR
> I/O SET ERR (Rack number)
> I/O UNIT ERR
> SYS FAIL FALS (Number)

The next group of errors do not stop the CPU.

> SYS FAIL FAL (Number)
> DPL ERR
> REMOTE I/O ERR (Remote I/O Unit no.)
> I/O VER ERR
> SCAN TIME OVER

Key Sequence

CLR → FUN → MONTR ------- MONTR

2-4
Monitor Operations

The monitor operation allows you to monitor bits, channels, or timers/counters and it supplements the functions provided by the Program Read, Instruction Search, and Bit Search operations described in previous subsections. In all cases, monitoring involves specifying the bit, channel or timer/counter to be monitored and then pressing the MONTR key.

The monitor operation described in this section may be performed in RUN, MONITOR, or PROGRAM mode and can be cancelled by pressing the CLR key.

2-4-1
General Status Monitoring

To monitor the status of an IR, SR, HR, or LR bit or channel, or a DM channel, specify the desired data, then press the MONTR key. Likewise, to monitor the set value of a timer or a counter, specify the desired timer/counter then press the MONTR key.
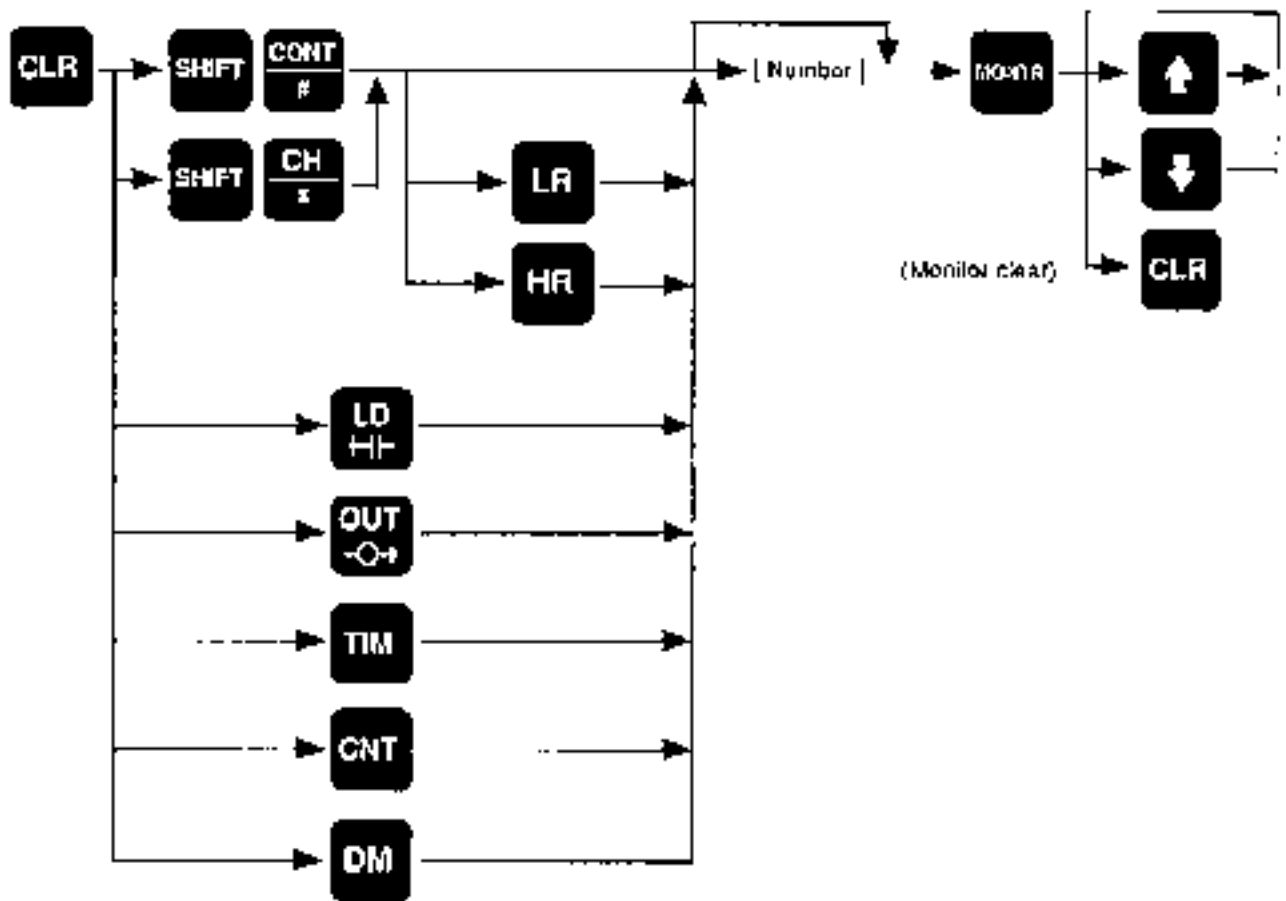
Using this operation you can simultaneously monitor the status of up to six values (any of channels, timers/counters, or bits). Of these 6 values, 3 are displayed at any one time. The 3 values selected for display may be changed at will and information is lost only if a 7th value is selected for monitoring, in which case the first value selected is lost and the 7th is read. This monitoring process can be continued for as long as desired.

The monitor value displayed in the leftmost corner can be force-set or, if it is channel data or a present value, can be changed.

Bit monitor displays indicate the ON/OFF status of the bit in question. Channel monitor displays show the binary (see 2-4-3), or hexadecimal contents of the specified channel. Timer/counter monitor displays show the BCD present value and a "T" in the lower left-hand corner when the present value (PV) has become 0.

By pressing the up-arrow and down-arrow keys, the display bit number or channel number is automatically incremented or decremented

Key Sequence

CLR → SHIFT CONT/# ⟶ [Number] → MONTR → ↑ →

SHIFT CH/＊ → LR →

→ HR →

(Monitor clear)   → CLR

LD/HR →

OUT/-○→ →

TIM →

CNT →

DM →

→ ↓

Examples

Program Read then Monitor

| 1 | 0 | 0 |

```
0100
```

↓

```
0100READ
TIM        000
```

MONTR

```
 T000
 1234
```

↓

```
 T001
▲0000
```

Indicates a time-out.

CLR

```
0100                    ┐ Monitor Clear
TIM         001
```

Bit Monitor

CLR

```
0000
```

LD / 1

```
0000
LD        0001
```

MONTR

```
0001
   ON
```

CLR

```
0000
CONT      0001
```

Channel Monitor

CLR

```
0000
```

SHIFT / CH

```
0000
CHANNEL      00
```

LR / 1

```
0000
CHANNEL  LR   01
```

MONTR

```
 L01
 FFFF
```

↑

```
 L00
 0000
```

Multiple Monitoring

Key Sequence

Single-item
monitor clear

All monitor
clear
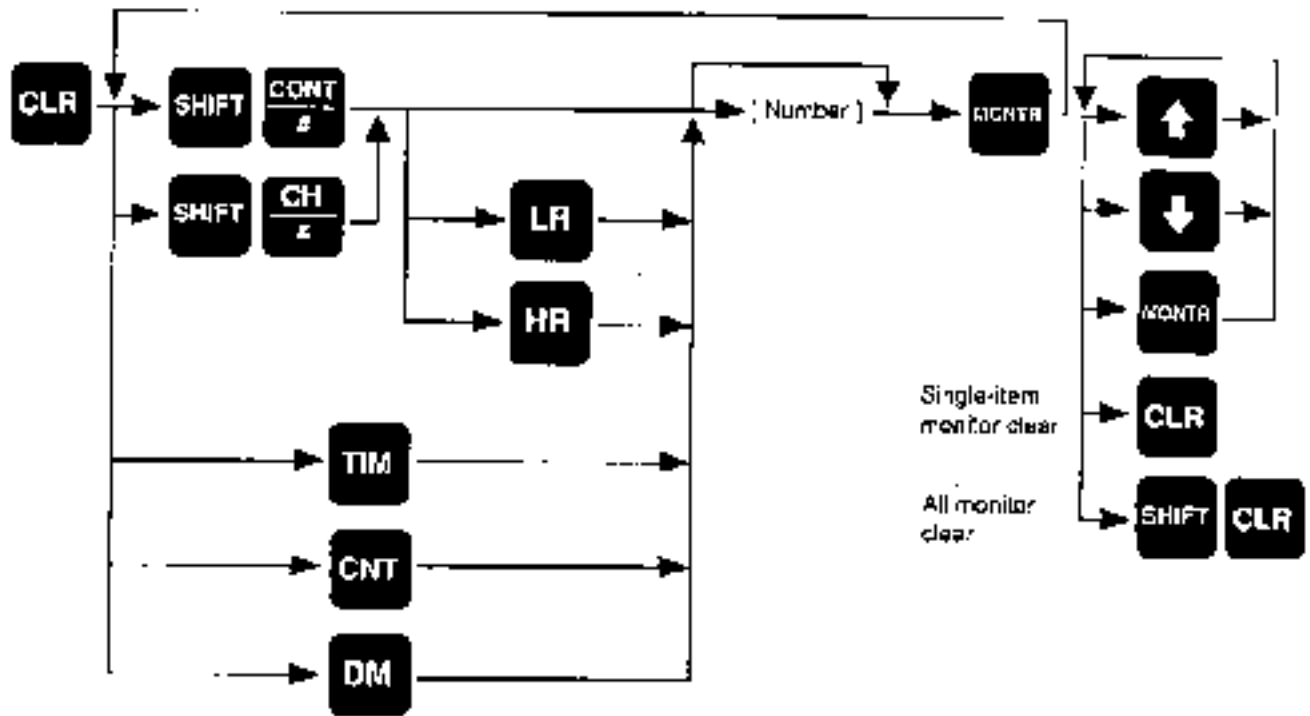
As shown in the next example, the first bit's status gets shifted to the right when other bits are monitored. If more than three bits are monitored, the bit monitored first gets shifted off the display but is still stored in the internal register.

Example

Key	Display
CLR	0000
TIM	0000 TIM　　　　　　000
MONTR	T000 0100
SHIFT **CONT #**	0000　T000 　　　　010A
1	0001　1000
MONTR	0001　T000 　OFF　0100
DM	D000　0001　T000 　　　　OFF　0100
MONTR	D000　0001　T000 10FF　　OFF　0100
MONTR	T000　D000　0001 0100　10FF　　OFF
CLR	D000　0001 10FF　　OFF	Clears the right side of the previous display.
CLR	0001 　OFF	
CLR	0000 CONT　　　　　　0001	
SHIFT **CLR**	0000 CHANNEL　DM　　000	Clears all three monitors.

2-4-2
Displaying a Single
Channel in Binary

You can specify that the contents of a monitored channel be displayed in binary by pressing the SHIFT and MONTR keys after the channel number has been input. Channels can be successively monitored by using the up-arrow and down-arrow keys to increment and decrement the display channel number. To clear the binary display press the CLR key.

Key Sequence

Example

CLR
```
0000
```
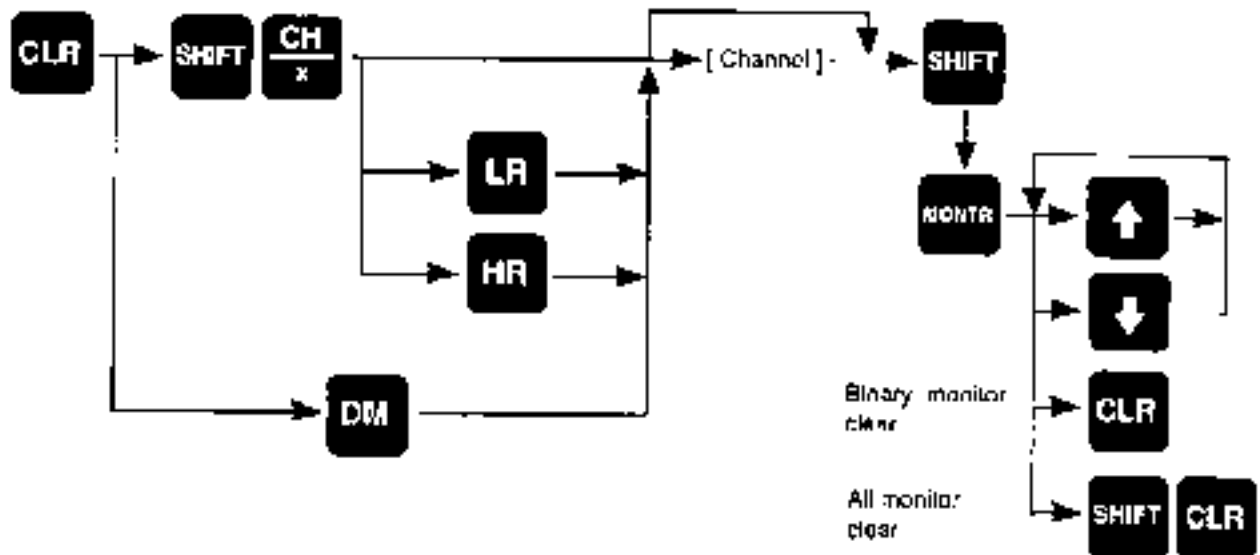
SHIFT **CH/***
```
0000
CHANNEL            00
```

SHIFT **MONTR**
```
        c  00  MONTR
00000000000001111
```

↓
```
        c  01  MONTR
00000101010101100
```

CLR
```
0000
CHANNEL            01
```

CLR
```
0000
```

DM
```
0000
CHANNEL    DM   000
```

MONTR
```
D000
FFFF
```

SHIFT **MONTR**
```
     D000    MONTR
1111111111111111
```

CLR
```
D000
FFFF
```

SHIFT **CLR** **CLR**
```
0000
CHANNEL    DM   000
```

2-5
Data Modification Operations

These operations are used to change channel data, to assign new set values and present values to timers and counters, and to set/reset bits. Except for the Hex <-> ASCII change operation, data modification operations can be done only in PROGRAM and MONITOR modes.

2-5-1
Force Set/Reset

This operation force sets and resets IR, SR, HR, and LR bits. It can also be used to force set/reset timers and counters.

Key Sequence

Bit or Timer/Counter currently displayed.

Programming Example
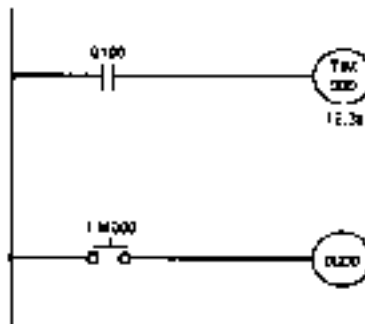
Address	Instruction	Data	
0000	LD		0100
0001	TIM		000
		#	0123
0002	LD	TIM	000
0003	OUT		0000

Examples: ...
... ...
... ...

Addresses will be counted down and consecutive bits set/reset as long as the PLAY/SET or REC/RESET key is held down. Channels 61 through 63 cannot be reset. Attempts to do so will activate a beeper.

Key	Display
CLR	0000
SHIFT **CONT #**	0000 SET　　　　　0000
1 **0** **0**	0000 SET　　　　　0100
MONTR	0100 　OFF
PLAY SET	0100 ■　ON
REC RESET	0100 ▣　OFF
TIM	T000　0100 　　　　OFF
MONTR	T000　0100 0123　　OFF
PLAY SET	T000　0100 ■0000　　OFF
	T000　0100 0123　　OFF
MONTR	T000　0100 ＿0000　　ON
REC RESET (hold)	T000　0100 ■0123　　ON
	T000　0100 0122　　ON

Counting

2-5-2
PV Change 1
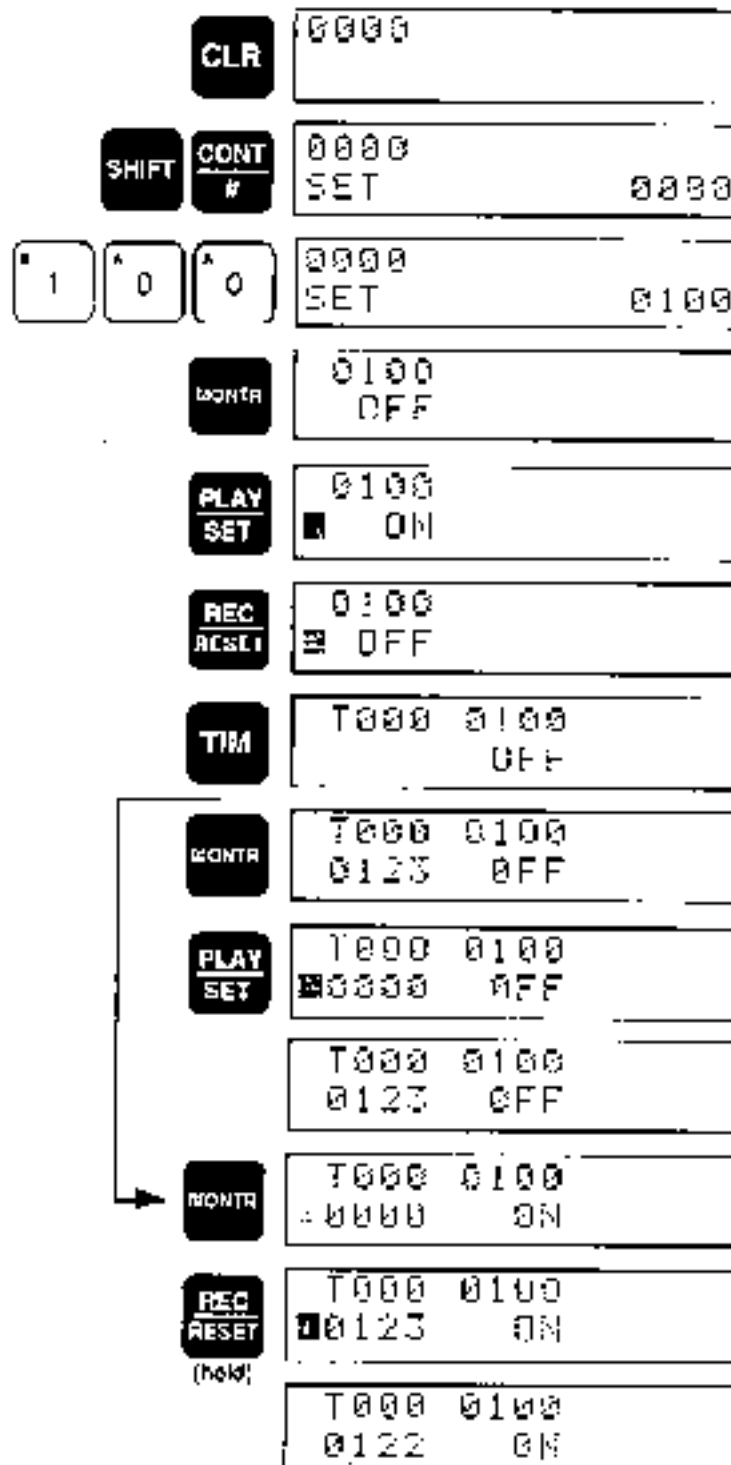
This operation changes the BCD present value of timers and counters. It also can be used to assign a new 4-digit decimal or hexadecimal value to an IR, HR, LR, or DM channel.

Key Sequence

Channel or Timer/Counter currently displayed. ⟶ **CHG** ⟶ (Data) ⟶ **WRITE**

Example

(This example is in MONITOR mode.)

CLR `0000`

TIM `0000`
`TIM          000`

MONTR `T000`
`0122`
———— Timing

CHG `PRES VAL ?`
`T000 0119 ????`
———— Timing   Change PV

`2` `0` `0` `PRES VAL?`
`T000 0100 0200`
———— Timing

WRITE `T000`
`0199`
———— Timing

57
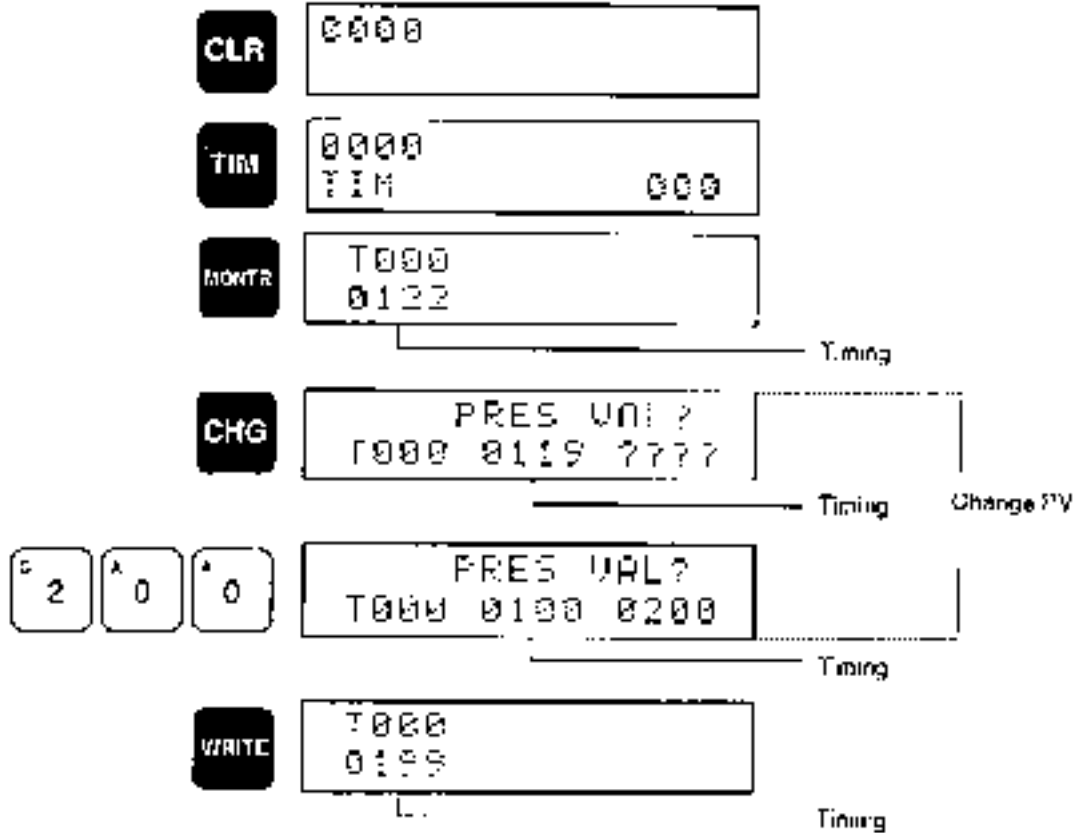
2-5-3
PV Change 2

This operation assigns a new 16 digit binary value to an IR, HR, LR, or DM channel.

Key Sequence
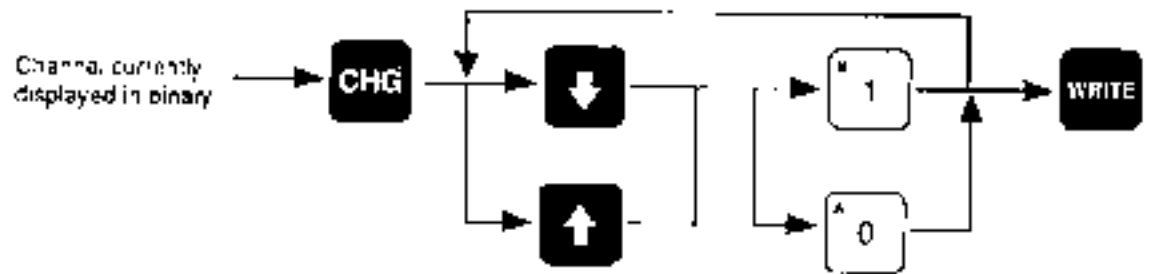
Example

CLR	`0000`	

| **SHIFT** **CH/x** | `0000`
`CHANNEL        00` | |

| **1** | `0000`
`CHANNEL        01` | |

| **SHIFT** **MONTR** | `       c 01  MONTR`
`0000:0101010101` | |

| **CHG** | `       c 01   CHG?`
`█00001010101010101` | |

| **1** | `       c 01   CHG?`
`1█0001010101010101` | |

| **0** | `       c 01   CHG?`
`10█001010101010101` | |

| **↓** | `       c 01   CHG?`
`100█01010101010101` | |

| **↓** | `       c 01   CHG?`
`1000█1010101010101` | |

| **↑** | `       c 01   CHG?`
`100█01010101010101` | |

| **↑** | `       c 01   CHG?`
`10█001010101010101` | |

| **WRITE** | `       c 01   CHG?`
`100001010101010101` | |

   `— IR bit 0115`   `— IR bit 0100`

The blinking square which can be shifted to the left with the up arrow key and to the right with the down-arrow key, indicates the position of the bit that can be changed. After positioning to the desired bit, a 0 or a 1 can then be entered as the new bit value. After a bit value has been changed, the blinking square will appear at the next position to the right of the changed bit.

2-5-4

Timer/Counter
SV Change 1

This operation changes the set value of a timer or counter while the program is being executed. This operation must be done in either MONITOR or PROGRAM mode.

Key Sequence

Timer/Counter
currently displayed. ──────▶ [↓] ──────▶ CHG ──────▶ [SV] ──────▶ WRITE

Example

CLR	0000

TIM	0000 TIM 000

SRCH	0201SRCH TIM 000

↓	0201 TIM DATA #0123

CHG	0201 TIM DATA T000 #0123 #????

1 2 4	0201 TIM DATA T000 #0123 #0124

SHIFT CH/×	0201 DATA? T000 #0123 c???

1 0	0201 DATA? T000 #0123 c 10

WRITE	0201 TIM DATA 10
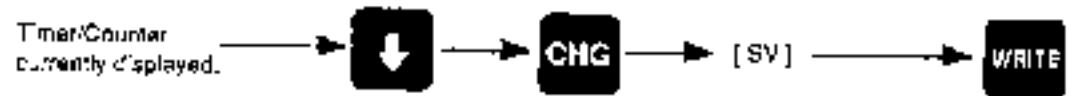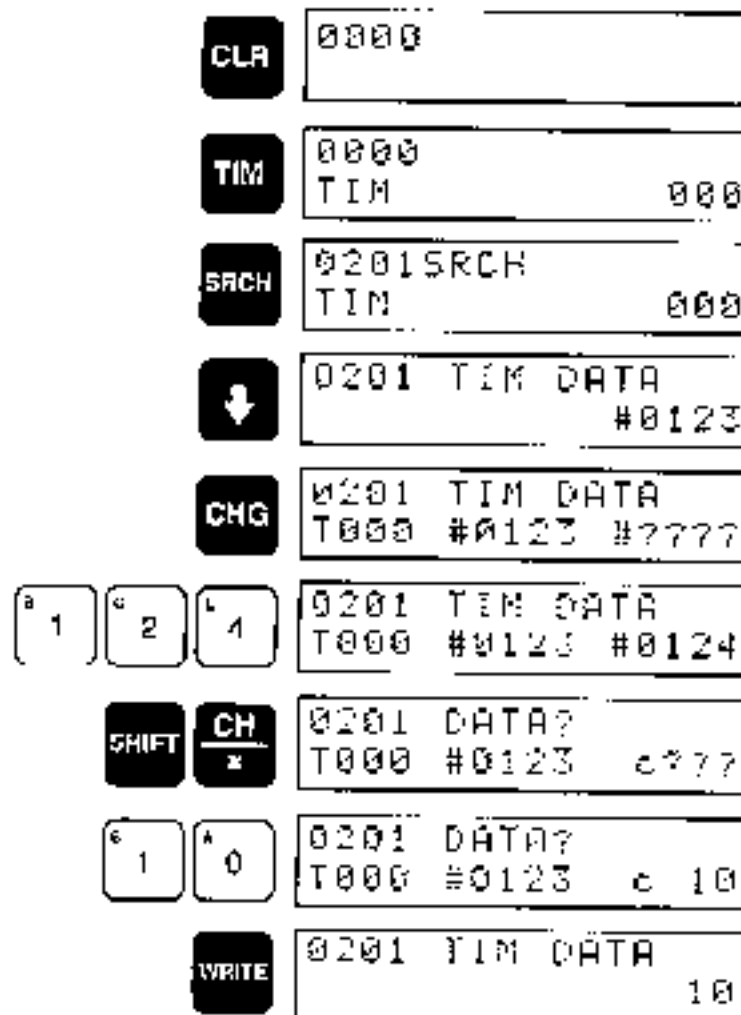
2-6
Cassette Tape Operations

PC programs (from user program memory-UM) or UM data may be backed-up on a standard commercially available cassette tape recorder. Any kind of magnetic tape of adequate length will suffice. (Note: To save an 8-Kword program, the tape must be 15 minutes long.) Always allow about 5 seconds of blank tape leader before the taped data begins. Store only one program on a single side of a tape; there is no way to identify separate programs stored on the same side of the tape.

Use patch cords to connect the cassette recorder earphone (or LINE OUT) jack to the Programming Console EAR jack and the cassette recorder microphone (or LINE-IN) jack to the Programming Console MIC jack. Set the cassette recorder volume and tone controls to maximum levels.

Note: For all operations, saving, loading, and verifying:
    The PC must be in the PROGRAM mode.
    While the operation is in progress, the cursor blinks and the block count is incremented on the display.
    Operation may be halted at any time by pressing the CLR key.

2-6-1
<u>Saving a Program</u>
<u>to Tape</u>

This operation copies program data from UM onto the cassette tape.

The procedure is as follows:

Press the **EXT** key.

Select a file number for the data that is to be saved.

Start cassette tape recording.

After about 5 seconds, press the **SHIFT** and **REC/RESET** keys.

Program saving continues until END is reached. At that time the program size in Kwords is displayed.

Key Sequence

Wait for about 5 seconds

(Cancel with the CLR key.)

Example

CLR

```
5000
```

EXT

```
0000MT
FILE  NO.00000060
```
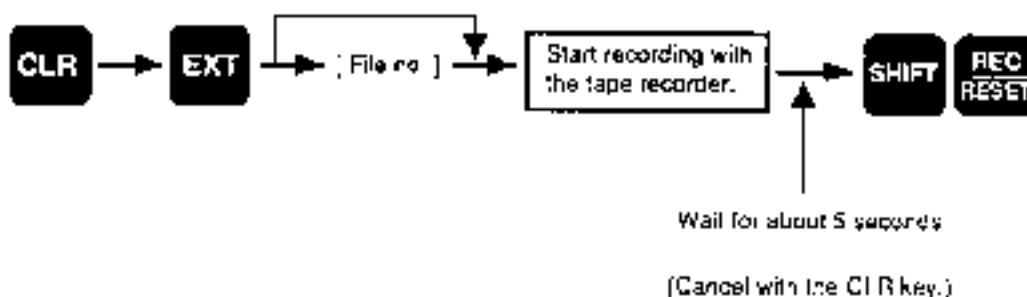
[B 1] **[C 2]**

```
0000MT
FILE  NO.00000012
```

Start recording with the tape recorder.

Let it run for about 5 seconds.

SHIFT **REC RESET**

```
0000MT    RECORD  █
FILE  NO.00000012
```
Blinking

(Recording in progress)

```
0075MT    RECORD  █
FILE  NO.00000012
```

(When it comes to END)

```
0145MT    RECORD  █
END  (01)(01.0KW)
```

Stop recording with the CLR key.

CLR

```
0145MT    DISCONT
END  (01)(01.0KW)
```

(Saved up to END.)

```
6000RECORD    END
END  (01)(24.0KW)
```

└─ Final address

2-6-2
Restoring Program Data

This operation restores program data from a cassette tape and writes it to user program memory (UM).

The procedure is as follows:

Press the **EXT** key.

Specify the number of the file to be restored.
Start playing the cassette tape.

Within 5 seconds. press the **SHIFT** and **PLAY/SET** keys to restore data.

Program restoration continues until END is reached. at which time the program size in Kwords is displayed
To restore program data recorded on two sides of a tape or on two or more tapes, begin restoring from the lowest address.
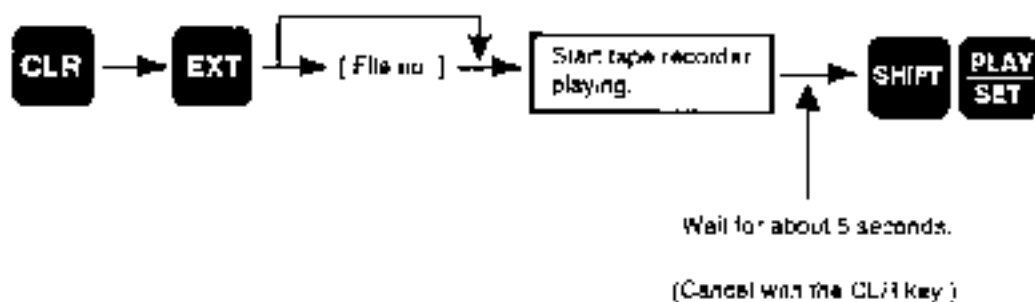
Key Sequence

CLR → **EXT** → [File no] → | Start tape recorder playing. | → **SHIFT** **PLAY/SET**

Wait for about 5 seconds.

(Cancel with the CLR key.)

Example

[CLR] `0000`

[EXT] `0000MT`
`FILE NO.00000000`

[1] [2] `0000MT`
`FILE NO.00000012`

Start the tape recorder playing

Within 5 seconds...

[SHIFT] [PLAY SET] `0000MT    PLAY  ■`   Blinking
`FILE NO.00000012`

(Restoring in progress)

`0075MT    PLAY  ■`
`FILE NO.00000012`

(When it comes to END)

`0145MT    RECORD ■`
`END (01)(01.0KW)`

Stop restoring with the CLR key.

[CLR] `0145MT    DISCONTD`
`END (01)(01.0KW)`

(Restored up to END.)

`0000RECORD   END`
`END (01)(24.0KW)`

— Final address

2-6-3
<u>Verifying Program Data</u>

This operation verifies that the contents of user program memory (UM) and the cassette tape program data match.

The procedure is as follows:

Press the **EXT** key

Specify the number of the file to be verified.

· Start playing the cassette tape

Within 5 seconds, press the **VER** key to verify data

Program verification continues until END is reached, at which time the program size in Kwords is displayed.

To verify program data recorded on two sides of a tape or on two or more tapes, begin verifying from the lowest address.
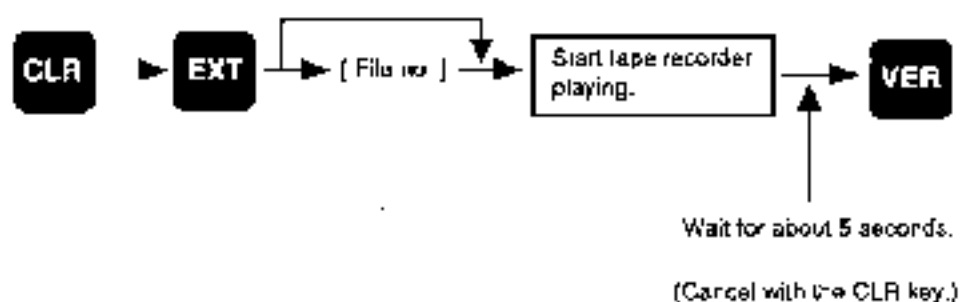
Key Sequence

CLR ▶ **EXT** ─▶ [File no] ─▶ | Start tape recorder playing. | ─▶ **VER**

Wait for about 5 seconds.

(Cancel with the CLR key.)

Example

| CLR | `0000` |

| EXT | `0000MT`
`FILE NO.00000000` |

| 1 | 2 | `0000MT`
`FILE NO.00000012` |

Start the tape recorder playing

Within 5 seconds ...

| VER | `0000MT    PLAY  ▓`
`FILE NO.00000012` | Blinking |

(Verification in progress)

`0075MT    PLAY  ▓`
`FILE NO.00000012`

(When it comes to END)

`0145MT    RECORD ▓`
`END  (01)(0.10KW)`

Stop verification with the **CLR** key.

| CLR | `0145MT    DISCONT0`
`END  (01)(0.10KW)` |

(Verified up to END.)

`6000RECORD    END`
`END  (01)(24.0KW)`

— Final address

SECTION 3
I/O Assignments and Data Areas

3-1
Overview

This section explains how I/O bits are used to identify individual I/O terminals and discusses the functions of the various types of data areas in the PC.

I/O Channels

The PC operates by monitoring input signals from such sources as push-buttons, sensors, and limit switches. Then, according to the program in its memory, the PC reacts to the inputs by outputting signals to external loads such as relays, motor controls, indicator lights, and alarms.

I/O channels are used to identify the input/output bits that correspond to the external terminal points through which the PC interacts with physical devices. Each channel consists of 16 bits. The I/O bits are assigned addresses as follows.

Addressing Conventions

Channel numbers are two-digit expressions and bit numbers are also two-digits. Thus four digits are used to address a particular I/O bit. Examples of I/O channel/bit addresses are shown below.

I/O Channel # +	Bit # (0 - 15)		—> I/O address
Channel 16, bit 3:	16	03	—> 1603
Channel 3, bit 15	03	15	—> 0315

Like I/O channel addresses, data area locations are also referenced by specifying a channel number and a bit number.

When the data is read as a four digit decimal or hexadecimal number, each digit represents a set of four bits in the channel (16 bits in all). Therefore, the rightmost digit of the decimal or hexadecimal number represents the rightmost four bits (3 to 0) of the channel.
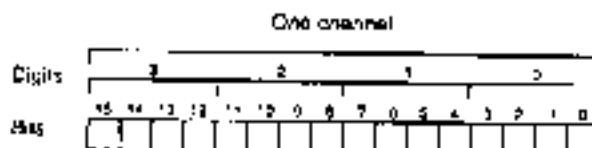
If, for example, the ON/OFF status of the rightmost four bits is 0101 in binary, the corresponding digit would be 5 in decimal or hexadecimal. In the case where the ON/OFF status is 1111 in binary, the hexadecimal number would be F (decimal 15).

69

Types of Data Areas   I/O channel bits are part of the I/O and Internal Relay (IR) Area . Bits that are not used for actual input or output operations constitute the remaining part of IR and are referred to as "work" bits. These work bits do not control external devices directly, rather they are used as data processing areas to control other bits, timers and counters

Timers and counters are found in the Timer/Counter (TC) area. The Special Relay (SR) area is used for system clocks, flags, and status information. There is also a Link Relay (LR) area for inter PC communication in systems that employ PC Link Units.

The function of the Holding Relay (HR) area is to store data and to retain the data values when the power to the PC is turned off. Data Memory (DM) is also used for internal data storage and manipulation and its values are also retained when power is off, but, unlike the HR area, it is only accessible in channel units. TR bits are used for temporary storage.

The programs that control the PC and all of its input and output operations are stored in the User Program Memory (UM). The capacity of the program memory depends on the type of RAM or ROM mounted to the CPU.

The following table shows the bits allocated within the PC.

Area	Bit Address Range
I/O	0000 to 3115
Work Bits	3200 to 6002
SR	6003 to 6307
TR	TR 0 to TR 7
HR	HR 0000 to HR 3115
LR	LR 0000 to LR 3115
I/C	000 to 127 (channels)
DM	DM 000 to DM 511 (channels)

Note: IR bits not used for I/O and also bits which are not used in other areas can be used as work bits

3-2
I/O and Internal Relay Area - IR

The I/O and Internal Relay Area (IR) is used for both I/O and internal data storage and manipulation. The channels available for I/O are as indicated in the table below.

Note that the actual number of IR channels that can be used as I/O channels is determined by the model of the CPU and the hardware configuration of the PC system.

I/O Channels

Channel Number											
00Ch	01Ch	02Ch	03Ch	04Ch	05Ch	~	26Ch	27Ch	28Ch	29Ch	30Ch	31Ch
00	00	00	00	00	00		00	00	00	00	00	00
01	01	01	01	01	01		01	01	01	01	01	01
02	02	02	02	02	02		02	02	02	02	02	02
03	03	03	03	03	03		03	03	03	03	03	03
04	04	04	04	04	04		04	04	04	04	04	04
05	05	05	05	05	05		05	05	05	05	05	05
06	06	06	06	06	06		06	06	06	08	06	06
07	07	07	07	07	07		07	07	07	07	07	07
08	08	08	08	08	09		08	08	08	08	08	08
09	09	09	09	09	09		09	09	09	09	09	09
10	10	10	10	10	10		10	10	10	10	10	10
11	11	11	11	11	11		11	11	11	11	11	11
12	12	12	12	12	12		12	12	12	12	12	12
13	13	13	13	13	13		13	13	13	13	13	13
14	14	14	14	14	14		14	14	14	14	14	14
15	15	15	15	15	15		15	15	15	15	15	15

Work-bit Channels

Work Bits															
32Ch	33Ch	34Ch	35Ch	36Ch	37Ch	38Ch	39Ch	40Ch	41Ch	42Ch	43Ch	44Ch	45Ch	46Ch	47Ch
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01
02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02
03	03	03	03	03	03	03	03	03	03	03	03	03	03	03	03
04	04	04	04	04	04	04	04	04	04	04	04	04	04	04	04
05	05	05	05	05	05	05	05	05	05	05	05	05	05	05	05
06	06	06	06	06	06	06	06	06	06	06	06	06	06	06	06
07	07	07	07	07	07	07	07	07	07	07	07	07	07	07	07
08	08	08	08	08	08	08	08	08	08	08	08	08	08	08	08
09	09	09	09	09	09	09	09	09	09	09	09	09	09	09	09
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
48Ch	49Ch	50Ch	51Ch	52Ch	53Ch	54Ch	55Ch	56Ch	57Ch	58Ch	59Ch	60Ch	61Ch	62Ch	63Ch
00	00	00	00	00	00	00	00	00	00	00	00	00			
01	01	01	01	01	01	01	01	01	01	01	01	01			
02	02	02	02	02	02	02	02	02	02	02	02	02			
03	03	03	03	03	03	03	03	03	03	03	03				
04	04	04	04	04	04	04	04	04	04	04	04				
05	05	05	05	05	05	05	05	05	05	05	05				
06	06	06	06	06	06	06	06	06	06	06	06				
07	07	07	07	07	07	07	07	07	07	07	07				
08	08	08	08	08	08	08	08	08	08	08	08				
09	09	09	09	09	09	09	09	09	09	09	09				
10	10	10	10	10	10	10	10	10	10	10	10				
11	11	11	11	11	11	11	11	11	11	11	11				
12	12	12	12	12	12	12	12	12	12	12	12				
13	13	13	13	13	13	13	13	13	13	13	13				
14	14	14	14	14	14	14	14	14	14	14	14				
15	15	15	15	15	15	15	15	15	15	15	15				

Note: Channels 58 and 59 may not be available as work-bit channels if they
      are required for PC Link Unit flags (channel 58) or Remote I/O Unit flags
      (channel 59).

Input Bit Usage

Input bits can directly input external signals to the PC. In programming, they can be used in any order and as often as necessary. They cannot be used in output instructions.
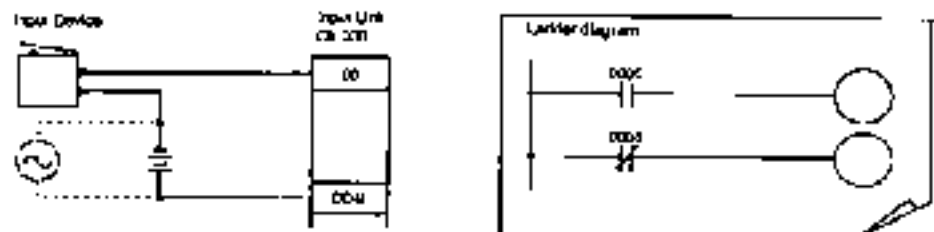
Output Bit Usage

Output bits are used to output program execution results. In programming they can be used in any order and as often as necessary
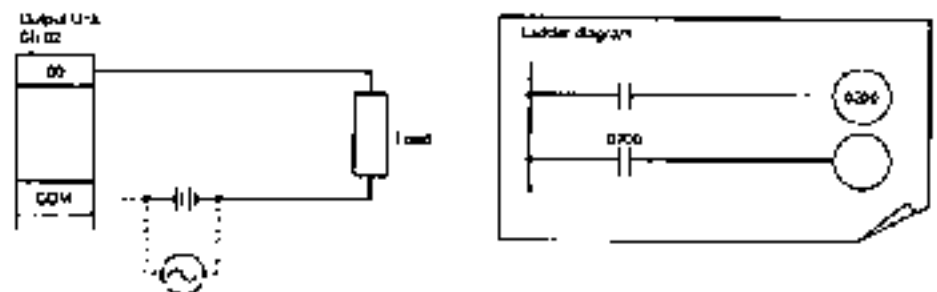
I/O Unit Mounting Location

When mounting I/O Units to the PC Racks, any type of I/O Unit can be mounted in any order. I/O channel numbers will be assigned serially according to the mounting order of the I/O Units. The mounting order of the I/O Units must then be registered using the I/O Table Register operation (see 2-2-3 Registering the I/O Table). The registered I/O table can then be checked with the I/O Table Read or I/O Table Verify operations. Note that vacant slots are not automatically registered. (Space may be reserved using a Dummy I/O Unit (see below)).

The I/O channel numbers are automatically assigned in sequence to the I/O Units mounted to the Racks. The top leftmost position is the starting point (i.e. C000, channel 00, bit 00) and bit numbers are assigned top to bottom, left to right.

I/O Channel Assignments

Starting position for I/O bit assignments

CPU Rack

Expansion I/O Rack
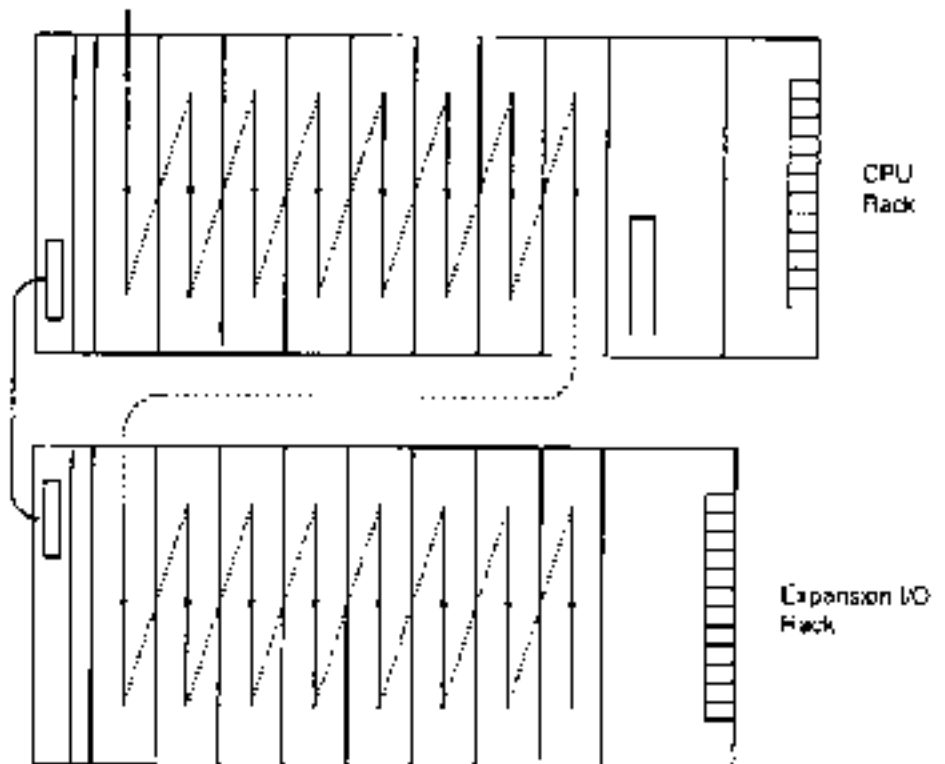
Slot Reservation

If an I/O Unit is later mounted to an unreserved vacant slot, the I/O Unit locations will disagree with the registered table and will cause an I/O verification error to occur. If an unplanned I/O Unit is required, change the programmed channel numbers for the I/O Units to the right of the added I/O Unit and register the table again.

Likewise, if a mounted I/O Unit is replaced with an I/O Unit with a different number of points, the channel numbers assigned to the I/O Units already mounted to the right of the new I/O Unit will need to be reassigned. The same is also true when a mounted I/O Unit is removed from the Rack, resulting in a vacancy.

The channel numbers will not be changed, however, if an I/O Unit is replaced with another Unit having the same number of points.

Space can be reserved for future addition of an I/O Unit(s) with a Dummy I/O Unit.

3-3
Special Relay Area - SR

The SR area is used for monitoring system operation, generating clock pulses, and signalling errors. The SR area addresses range from 6003 to 6307.

The following table lists the functions of SR area flags and bits. Unless otherwise stated, flags are OFF until the specified condition arises, then they are turned ON by the system. Restart bits are usually OFF, but when the user turns one ON then OFF again, it will restart the particular link.

Bit	Function
6003	SYSNET error flag
6004	SYSNET run flag
6008	CPU-mounting Host Link error flag
6009	CPU-mounting Host Link restart bit
6010	PC Link level 1 restart bit
6011	Power failure flag
6012	Date retention flag
6013	Rack-mounting Host Link restart bit
6014	PC Link level 0 restart bit
6015	Load-off control (Shuts off output loads when ON)
6100 to 6107	FAL No. output area: an 8-bit FAL code is output here by FAL, FALS, or the system when a failure occurs.
6108	Battery alarm flag
6109	Scan time error flag
6110	I/O verification error flag
6111	Rack-mounting Host Link Unit error flag
6112	Remote I/O error flag
6113	Normally ON flag
6114	Normally OFF flag
6115	First scan flag (ON for 1 scan only)
6200 to 6207	PC Link level 0, Units 0 to 7 Run flags
6208 to 6215	PC Link level 0, Units 0 to 7 Error flags
6300	0.1-second clock pulse
6301	0.2-second clock pulse
6302	1.0-second clock pulse
6303	Error (ER) flag
6304	Carry (CY) flag
6305	Greater than (GR) flag
6306	Equals (EQ) flag
6307	Less than (LE) flag

Note: Channel 58, which is normally part of the work-bit area, may be used for PC Link Unit flags if many PC Link Units are required. The bit usage in this case is as below.

Bit	Function
5800 to 5807	PC Link level 1, Units 8 to 15 Run flags
5808 to 5815	PC Link level 0, Units 8 to 15 Error flags

3-3-1
Data Retention Flag

When the data retention flag, bit 6012, is ON, the current operating status of I/O bits, work bits, and link bits is retained. This flag is effective, though, only when the PC is operating in MONITOR or RUN mode.

Having the data retention flag OFF clears the status data when RUN mode starts. This flag is normally OFF but it can be turned ON with OUT

3-3-2
Load Off Control

When the load-off flag, bit 6015, is ON, all output to the Output Units is inhibited and the OUT INHB indicator on the front panel of the CPU lights.

When the load-off flag is OFF, Output Units are refreshed normally. The load-off flag is normally OFF but it can be turned ON with OUT.

When a power failure occurs, the load-off flag retains the status it had before the power failure.

3-3-3
FAL Number Output Area

Bit numbers 6100 to 6107.

FAL or FALS execution outputs a 2 digit BCD FAL code (for error diagnosis) to this 8-bit area. The system also outputs a FAL number to this area when an alarm output occurs, such as one caused by battery failure.

This area can be reset by executing FAL 00 (See 4-10-1 FAL(06)) or through a Failure Read Programming Console operation.

3-3-4
Battery Alarm Flag

When the battery alarm flag, bit 6108, is ON it indicates that the supply voltage of the CPU backup battery has dropped. The warning indicator lamp on the front panel of the CPU is lit.

3-3-5
Scan Time Error Flag

When the scan time exceeds 100 ms, the scan time error flag, bit 6109, turns ON and the warning indicator lamp on the front panel of the CPU lights.

Unless the scan time exceeds the maximum limit (see 4-10-2 Set Watchdog Timer), system execution continues but timing may become inaccurate

3-3-6
I/O Verification Error Flag

The I/O verification error flag, bit 6110, turns ON when the number of I/O Units mounted on the CPU Rack and Expansion I/O Rack disagrees with the I/O table registered.

3-3-7
First Scan Flag

The first scan flag, bit 6115, turns ON when program execution starts and turns OFF after one scan.

3-3-8
Instruction Execution Error Flag, ER

Attempting to execute an instruction with incorrect data turns the ER flag, bit 6303, ON Common causes of an instruction error are non-BCD operand data when BCD data is required, or an indirectly addressed DM channel that is non-existent. When the ER flag is ON, the current instruction is not executed.

3-3-9
Arithmetic Operation Flags

Carry Flag. CY

The CY flag, bit 6304, turns ON when there is a carry in the result of an arithmetic operation, or when a rotate or shift instruction moves a "1" into CY. This flag is set and cleared by STC and CLC, respectively. If necessary use CLC before any instruction using CY. (See 4-8-3 Set and Clear Carry.)

Equal Flag, EQ

The EQ flag, bit 6306, turns ON when the result of CMP (compare) shows two operands to be equal, or when the result of an arithmetic operation is zero.

Greater Than Flag, GR

The GR flag, bit 6305, turns ON when the result of CMP (compare) shows the second of two 4-digit operands to be greater than the first.

Less Than Flag, LE

The LE flag, bit 6307, turns ON when the result of CMP (compare) shows the second of two 4-digit operands to be less than the first.

3-3-10
Clock Pulses

Pulse width	0.1 s	0.2 s	1.0 s
Flag	6300	6301	6302

Generates 0.1-s clock pulse

Generates 0.2-s clock pulse

Generates 1.0-s clock pulse
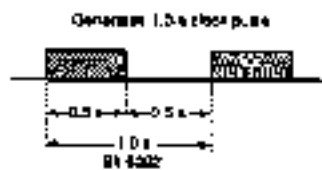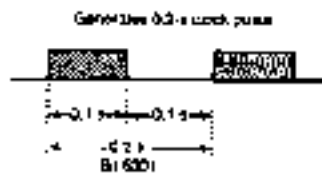
Each clock bit is ON for the first half of the rated pulse time, then OFF for the latter half. In other words, each clock pulse generator flag has a duty factor of 1 to 1.

Note. Since the 0.1 second and 0.2 second clock pulses have ON times of 50 and 100 ms, respectively, if the scan time is too long, the CPU will not be able to accurately read the pulse.

3-3-11
Special I/O Flags

Use of the following SR flags and bits depends on the particular configuration of your PC system. These flags and bits are used when components such as PC Link Units, Remote I/O Units, SYSNET Link Units, or Host Link Units are contained within the PC system. For additional information, consult the System Manual for the particular Units involved.

The following bits can be employed as work bits when the special type of Unit associated with them is not connected to the system.

·PC Link I/O Error Flags

When PC Link Units are used in the system, channels 58 and 62 may be used to monitor the operating status of up to 16 PC Link Units.

Channel	PC Link Units
58	Nos. 08 to 15
62	Nos. 00 to 07

For each channel, bits 00 to 07 are ON when the Units are in RUN mode and bits 08 to 15 are ON when an error occurs in the corresponding Unit.

For example if the contents of Ch 58 are 02FF as below, then it means that Units 8 to 15 (link level 1) are in RUN mode, and Unit 5 (link level 1) has an error.

Ch 58

0000	0010	1111	1111
0	2	F	F

·PC Link Restart Flags

Link Level	Restart Flag
# 0	6014
# 1	6010

Turn these flags ON then OFF to restart the PC Link System.

·Remote I/O Error Flag

Indicates an error in a Remote I/O Unit (Bit 6112)

•SYSNET Error and Run
Flags

Flag	Bit
Error Flag	6003
Run Flag	6004

These flags indicate the state of the SYSNET system.

•Host Link Error Flags

A Host Link error flag turns ON if an error occurs in a Host Link Unit on the PC. The PC has two Host Link Unit error flags. One to indicate an error in a Rack-mounting Host Link Unit (Bit 6111), and the other to indicate an error in a CPU-mounting Host Link Unit (Bit 6008).

•Host Link Restart Flags

Turn a Host Link restart flag ON, and then OFF again to restart a Host Link Unit that has an error. The PC has two types of Host Link restart flags. One restarts a Rack-mounting Host Link Unit (Bit 6013), and the other restarts a CPU-mounting Host Link Unit (Bit 6009)

3-4
Holding Relay Area - HR

The HR area is used to store and manipulate various kinds of data. Its addresses range from 0000 to 3115. HR area channels retain their data when the system operating mode changes and also during power failure.

To access bits in this area, prefix the address number with "HR" (e.g., HR 0101 for bit 01 in HR channel 01) by pressing the HR key on the Programming Console.

See 4-2-5 Latching Relay for an example of how to use HR channels as "latching relays" that hold their data when a power failure occurs.

					Channel No./Bit No.						
HR Ch00	HR Ch01	HR Ch02	HR Ch03	HR Ch04	HR Ch05	HR Ch06	HR Ch07	~	HR Ch29	HR Ch30	HR Ch31
00	00	00	00	00	00	00	00		00	00	00
01	01	01	01	01	01	01	01		01	01	01
02	02	02	02	02	02	02	02		02	02	02
03	03	03	03	03	03	03	03		03	03	03
04	04	04	04	04	04	04	04		04	04	04
05	05	05	05	05	05	05	05		05	05	05
06	06	06	06	06	06	06	06		06	06	06
07	07	07	07	07	07	07	07		07	07	07
08	08	08	08	08	08	08	08	~	08	08	08
09	09	09	09	09	09	09	09		09	09	09
10	10	10	10	10	10	10	10		10	10	10
11	11	11	11	11	11	11	11		11	11	11
12	12	12	12	12	12	12	12		12	12	12
13	13	13	13	13	13	13	13		13	13	13
14	14	14	14	14	14	14	14		14	14	14
15	15	15	15	15	15	15	15		15	15	15

3-5
Temporary Relay Area - TR

The TR area consists of eight bits and is used for storing data at program branching points. This is useful for programs that have many output branching points but where IL and ILC (branch instructions) cannot be used. See 4-2-3 Temporary Relay.

To access bits in this area, prefix the address number with "TR" (i.e., TR 00 to TR 07) by pressing the TR key on the Programming Console

A TR number within a given section (i.e. a single branch from the left-hand bus bar of a ladder diagram) of the program must not be duplicated. However, the same TR number can be used again in different program sections. Unlike the IR and SR areas, TR bits can only be used in conjunction with LD and OUT.

3-6
Link Relay Area - LR

The LR area ranges from 0000 to 3115. In a system employing PC Link Units, part of the LR area is devoted to system data communications. (Refer to the PC Link Systems manual for details.) The part of the LR area that is not required by the PC Link Units can be used for internal data storage and manipulation, in the same manner as the IR area.

LR area data is NOT retained when the power fails, when the program mode changes, or when it is reset by an IL-ILC bypass (see Interlock under 4-2-2).

To access bits in this area, prefix the address number with "LR" (e.g., LR 0101 for bit 01 in LR channel 01) by pressing the LR key on the Programming Console.

					Channel No./Bit No.						
LR Ch00	LR Ch01	LR Ch02	LR Ch03	LR Ch04	LR Ch05	LR Ch06	LR Ch07	~	LR Ch29	LR Ch30	LR Ch31
00	00	00	00	00	00	00	00		00	00	00
01	01	01	01	01	01	01	01		01	01	01
02	02	02	02	02	02	02	02		02	02	02
03	03	03	03	03	03	03	03		03	03	03
04	04	04	04	04	04	04	04		04	04	04
05	05	05	05	05	05	05	05		05	05	05
06	06	06	06	06	06	06	06		06	06	06
07	07	07	07	07	07	07	07		07	07	07
08	08	08	08	08	08	08	08	~	08	08	08
09	09	09	09	09	09	09	09		09	09	09
10	10	10	10	10	10	10	10		10	10	10
11	11	11	11	11	11	11	11		11	11	11
12	12	12	12	12	12	12	12		12	12	12
13	13	13	13	13	13	13	13		13	13	13
14	14	14	14	14	14	14	14		14	14	14
15	15	15	15	15	15	15	15		15	15	15

3-7
Timer/Counter Area - TC

The TC area, which ranges from 000 to 127, is a single data area in which timer and counter data is stored for use by TIM, TIMH (FUN 15), CNT, and CNTR (FUN 12). This area is accessible in channel units only, which serve as the storage area for the set-value (SV) and the present-value (PV) of the timer/counter (TIM/CNT). TIM/CNT numbers are three digits. To specify a timer or a counter, prefix the three-digit TC number with "TIM" or CNT" (e.g., TIM 001 or CNT 126) or, for TIMH and CNTR, prefix the number with the appropriate FUN code.

Once a given TC number has been specified, that same number cannot be used again for any other timer or counter. For example, if TIM 010 has been specified in a program, a subsequent use of CNT 010 will generate an error.

The timer/counter area retains the SV of both timers and counters during power failure, but the PV of the timers is not retained. The PV of the counters is retained.

3-8
Data Memory Area - DM

The DM area is used for internal data storage and manipulation, and is accessible only in 16-bit channel units. The DM area retains data during power failure. DM area channels are numbered 000 through 511.

This area cannot be used by instructions with bit size operands, such as LD, OUT, AND, and OR.

Indirect Addressing

Normally, when data is specified for an instruction, the instruction operation is performed directly on that data. For example, suppose a Compare with IR 05 as the first operand, and DM 010 as the second operand, is in the program. When this instruction is executed the data in IR 05 is compared with that in DM 010.

It is possible, however to use indirect addresses as operands for the instructions. So, if *DM 100 is specified as the data for a programming instruction, the contents of DM 100 specifies another DM channel at which the actual operand data is to be found.

If, for example, the content of DM 100 is 0324, then *DM 100 is DM 324 and the data that the program instruction actually uses is the content of DM 324.

3-9
Program Memory - UM

Program instructions are stored in program memory. The amount of program memory available depends on the type of Memory Unit attached to the PC. Memory Units come in different types - RAM and ROM memory packs - and for each type there are different sizes. (Refer to the Hardware Manual for details.)

To store instructions in program memory, input the instructions through the Programming Console, or download programming data from a GPC, FIT, floppy disk, cassette tape, or host computer.

SECTION 4
Programming Instructions

4-1
Overview

The C500 PC has a large selection of programming instructions that allows for easy programming of complicated control processes. The programming instructions described in this section are divided into categories by operation Each instruction's explanation includes the ladder diagram symbol and the mnemonic code for the instruction. Examples of how to use instructions are also provided.

Instruction Data and Flags

For each instruction the Data Areas and Flags subsections list the data areas that can be specified and the error flags that are applicable. Refer to Section 2 I/O Assignments and Data Areas for the size of each data area in the PC and the address of each flag The following abbreviations are used:

Data Areas

IR.	I/O and Internal Relay Area
SR:	Special Relay Area
HR:	Holding Relay Area
LR·	Link Relay Area
TC·	Timer/Counter Area
DM:	Data Memory Area
*DM:	Indirectly Addressed Data Memory Area
#:	Constants (see note below)

Note: The value that can be specified for a given constant depends on the particular instruction that uses it. When the constant is to specify a data area channel, it must correspond to a channel address within the data area. In cases where the constant is the data to be contained within a channel, it may be hexadecimal or decimal, as required by the instruction.

Flags

ER·	Error flag
CY:	Carry flag
EO:	Equals flag
GR:	Greater than flag
LE·	Less than flag

ER is the flag most often used for monitoring an instruction's execution. When an ER flag goes ON, it indicates that an error occurred in attempting to execute the current instruction. The Flags subsection for each instruction lists possible reasons for the ER flag being ON.

Unless otherwise noted, an instruction is not executed when the ER flag is ON.

4-2
Basic Instructions

4-2-1
LD, OUT, AND, OR, NOT, and END

These six basic instructions are indispensable in almost any program. All have a corresponding key on the Programming Console, which you press to enter the instruction, except for END which is programmed by pressing the FUN, 0, and 1 keys. Refer to 1-3 Basic Programming Instructions for details about how to use and enter these instructions.

Instruction	Operation
LD	Starts each logic line or block.
OUT	Indicates an output bit
AND	Performs a logical AND operation on two inputs.
OR	Performs a logical OR operation on two inputs.
NOT	Inverts whatever is below it; often used to form an NC (normally closed) input. NOT can be used with LD, OUT, AND, or OR.
END(01)	Indicates the end of the program.

4-2-2
Interlock - IL(02) and
ILC(03)

IL(interlock) is always used in conjunction with ILC (interlock clear). If the IL condition is OFF (i e, the bit just before the IL branch is OFF), the section of program between IL and ILC is not executed. If the IL condition is OFF the states of outputs in the section of program between IL and ILC is as below:

OFF	Output bits
Reset	Timers
Unchanged	Counters, shift registers, latches

Key Sequence

To input IL, enter
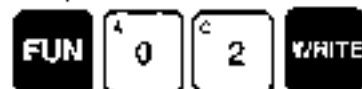
To input ILC, enter
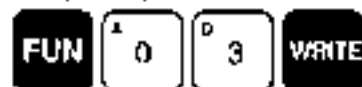
**Example Circuit:
Ladder Diagram and
Mnemonic Code**
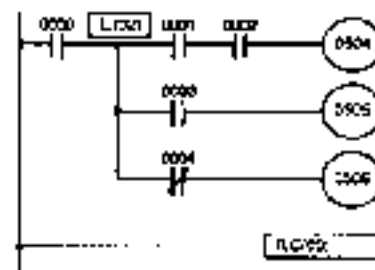
Address	Instruction	Data
0000	LD	0000
0001	IL(02)	
0002	LD	0001
0003	AND	0002
0004	OUT	0504
0005	LD	0003
0006	OUT	0505
0007	LD NOT	0005
0008	OUT	0506
0009	ILC(03)	

If 0000 (IL condition) is ON, the program between IL and ILC is executed normally.

IL-IL-ILC
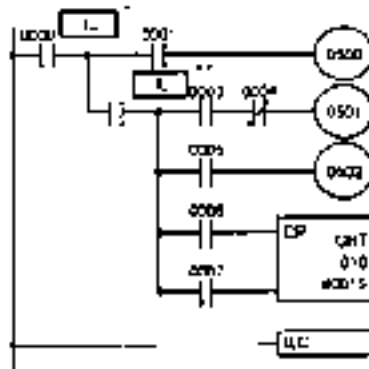
When the first IL condition * is OFF, outputs 0500, 0501, and 0502 are all OFF, and the counter CNT 010 retains its present count value.

When the first IL condition * is ON and the second IL condition ** goes OFF, the state of output 0500 matches that of bit 0000, outputs 0501 and 0502 turn OFF, and the counter retains its present value.

When both IL conditions are ON at the same time, the program executes as if they were not there.

As shown above, more than one IL can be used with a single ILC. Although this causes an IL-ILC error message to occur when the program check is performed, execution proceeds normally. However, all of the ILs before the ILC are cleared. IL/ILC nesting (e.g., IL-IL-ILC-ILC) is not allowed.

4-2-3
<u>Temporary Relay - TR</u>

A TR bit can be used as a temporary work bit for branching to more than one output bit. TR is used when a ladder diagram cannot be programmed with IL or ILC

TR bits can be used several times, but they cannot be duplicated within the same block.

TR can only be input via the TR key as LD or OUT data. Since the Programming Console is the only device which allows TR key input, you can program TR only when using the Programming Console.
It is not possible to monitor TR with either the Programming Console or other devices.

Key Sequence

To output to a TR bit, enter

| OUT -O- | | TR | [N] | WRITE |

To input from a TR bit, enter

| LD -| |- | | TR | [N] | WRITE |

Example Circuit:
Ladder Diagram and
Mnemonic Code
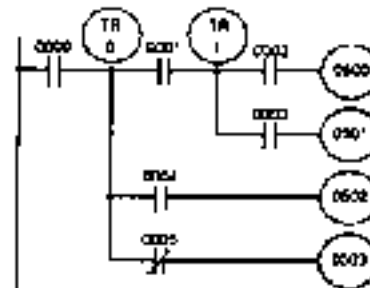
Address	Instruction	Data	
0000	LD		0000
0001	OUT	TR	0
0002	AND		0001
0003	OUT	TR	1
0004	AND		0002
0005	OUT		0500
0006	LD	TR	1
0007	AND		0003
0008	OUT		0501
0009	LD	TR	0
0010	AND		0004
0011	OUT		0502
0012	LD	TR	0
0013	AND NOT		0005
0014	OUT		0503

How to use TR Bits

Eight TR bits are provided: TR 0 to TR 7. These can be used an unlimited number of times in a program, provided they are not duplicated within the same block.

The next example shows a case where TR is needed and another case where TR is not needed.
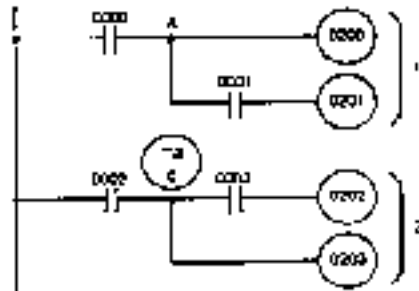
Case 1:   TR is not needed here because output 0201 and AND0001 can be connected with output 0200 through Point A.

Case 2:   Here, because the data at the branch point and that at output 0202 is not necessarily the same, TR is needed.

Comparison of TR and
IL/ILC

Use of IL/ILC in place of TR results in a smaller program because IL/ILC does not require the extra addresses needed by LD TR.

4-2-4
Jump and Jump End - JMP(04)/JME(05)

JMP and JML, are used to branch program control to the first instruction after JME when the JMP condition (i.e., the state of the JMP input) is OFF

N, the jump number, serves to distinguish JMP/JME pairs when there are multiple jumps in a program A jump number can be any two digit number between 00 and 99.

JMP 00, however, is a special case When the instructions between JMP 00 and JME 00 are skipped, they are still processed but not executed Thus, processing time is required. On the other hand, instructions between JMP and JME with jump numbers other than "00" require no processing time at all and are entirely skipped over.

JMP 00-JME 00 can be programmed any number of times but non-zero jump numbers must be used only once in the program.

Key Sequence

To input JMP, enter
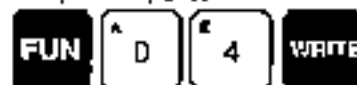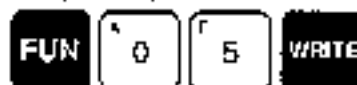
| FUN | D | 4 | WRITE |

To input JME, enter

| FUN | 0 | 5 | WRITE |

**Example Circuit:
Ladder Diagram and
Mnemonic Code**
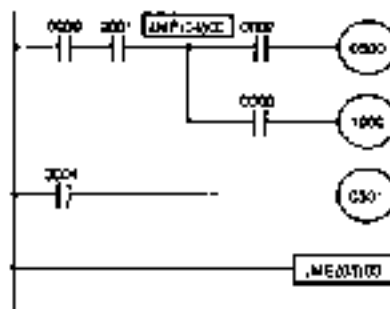
Address	Instruction	Data
0000	LD	0000
0001	AND	0001
0002	JMP(04)	00
0003	LD	0002
0004	OUT	0500
0005	LD	0003
0006	OUT	1000
0007	LD	0004
0008	OUT	0501
0009	JME(05)	00

In the preceding program, 0000 and 0001 serve as the JMP condition. When this condition is ON, the program between JMP and JME is executed normally. However, if the JMP condition is OFF (in this case, if either or both 0000 and 0001 are OFF), the program between JMP and JME is not executed, but the states of all bits (and timers) are retained.

JMP-JMP-JME

More than one JMP 00 can be used with the same JME 00. This causes a JMP-JME error message to be generated when the program check is performed, but the program executes normally
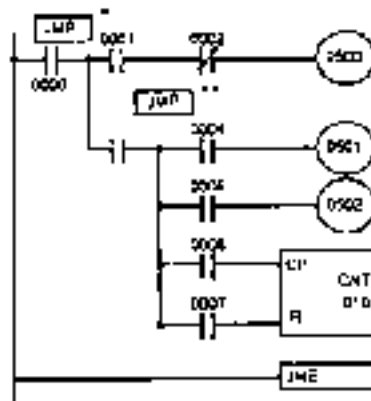
Multiple JMPs

When the first JMP condition * is OFF, outputs 0500, 0501, and 0502, and the counter all retain their states.

When the first JMP condition * is ON and the second JMP condition ** is OFF, the ON/OFF state of output 0500 depends on the states of 0001 and 0002, and outputs 0501 and 0502 and the counter retain their states.

When both JMP conditions are ON at the same time, the program executes as if neither JMP were there.

JMP vs Interlock

Since the states of I/O bits, timers, etc., are retained when JMP/JME branching occurs, JMP/JME is used to control devices that require a sustained output (e.g., pneumatics and hydraulics).

On the other hand, IL/ILC branching is used to control devices that do not require a sustained output, such as electronic instrumentation.

4-2-5
Latching Relay - KEEP(11)

KEEP is used as a latch. It maintains an ON or OFF state until one of its two inputs sets or resets it.

Bits that may be used as latches are those from the IR, HR, and LR data areas. If a HR bit is used as a latch, the latched data is retained even during a power failure.

When programming a latch, first load the set input, and then the reset input, before entering the FUN code.

Key Sequence

To input KEEP, enter
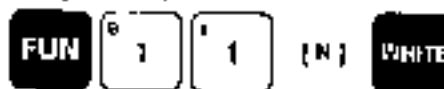
Example Circuit:
Ladder diagram and
Mnemonic Code
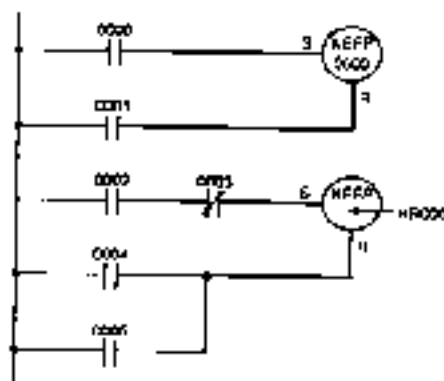
Address	Instruction	Data
0000	LD	0000
0001	LD	0001
0002	KEEP(11)	0500
0003	LD	0002
0004	AND NOT	0003
0005	LD	0004
0006	OR	0005

Data Areas

IR, HR, LR
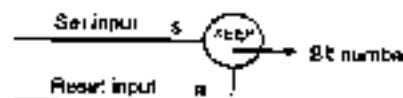
When the Set input is ON, the latch is maintained in an ON state until a reset signal turns it OFF. Reset has a higher priority and takes precedence when both inputs are ON.

97
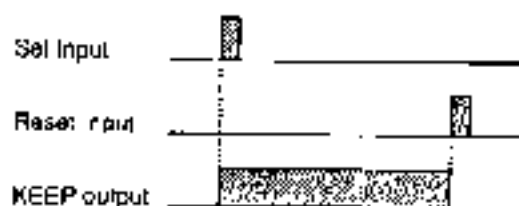
To program a self-latching circuit with KEEP, change the circuit below as shown.
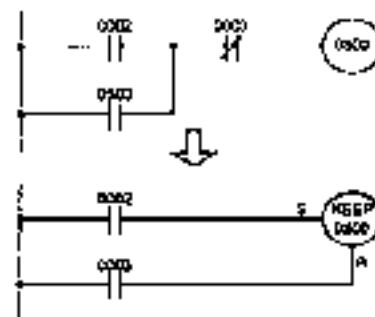
If the self-latching circuit above is used within an IL/ILC block and the IL condition turns OFF, the KEEP state will be retained whereas with the uppermost circuit, output 0500 will go OFF.
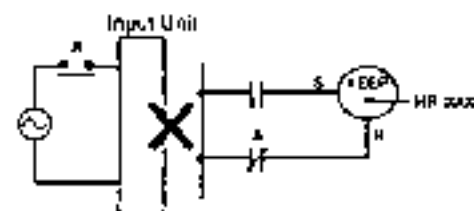
Note   Exercise caution when using a KEEP reset line that is taken from a bit directly connected to external instrumentation. In particular, a sudden drop in AC power that is insufficient to cause the PC's DC power to go off will reset the latch.

Application Example
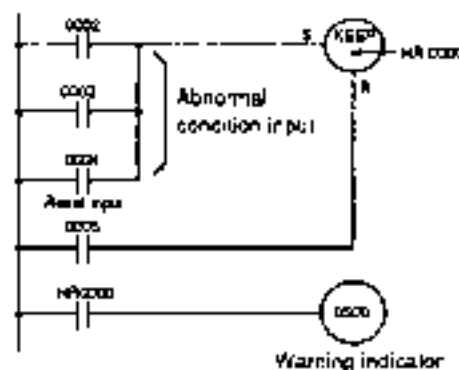
4-2-6
Differentiation - DIFU(13)
and DIFD(14)

DIFU and DIFD turn an output ON for a single scan. These instructions are used when single-scan execution of a particular instruction is desired.

DIFU turns its output ON when it detects an OFF->ON transition in its input signal.

DIFD turns its output ON when it detects an ON->OFF transition in its input signal.

Key Sequence

To input DIFU, enter

To input DIFD, enter

Example Circuit:
Ladder Diagram and
Mnemonic Code
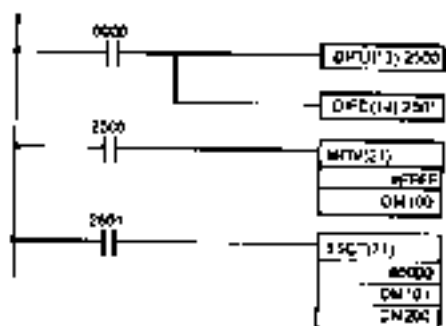
Address	Instruction	Data	
0000	LD		0000
0001	DIFU(13)		2500
0002	DIFD(14)		2501
0003	LD		2500
0004	MOV(21)	—	—
		#	FFFF
		DM	100
0005	LD		2501
0006	BSET(71)	—	—
		#	0000
		DM	101
		DM	200

In the preceding example, an OFF->ON transition in input 0000 turns IR 2500 ON and executes MOV for one scan.
An ON->OFF transition in input 0000 turns IR 2501 ON for one scan and executes BSET for one scan.

Timing

Input 0000

Leading-edge differentiation output 2800

Trailing-edge differentiation output 2801

1 scan   1 scan
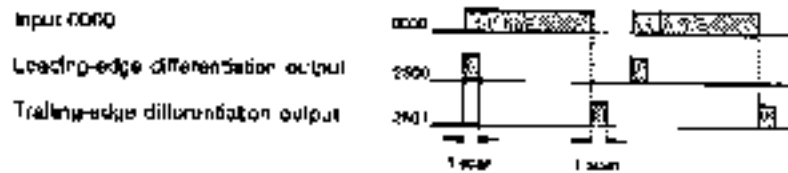
Note. A maximum of 128 pairs of DIFUs and DIFDs can be used in a given program. If 128 is exceeded, the error message "DIF OVER" is displayed on the LCD of the Programming Console and the 129th DIFU or DIFD, and any following, are treated as NOPs (no operation instructions).

Data Areas

IR, HR, LR

4-3
Timers and Counters

TIM is a decrementing ON delay timer instruction which requires a timer number and a set value (SV) as data. When the specified SV has elapsed, the timer output turns ON

CNT is a decrementing counter instruction and CNTH is a reversible counter instruction. Both require a counter number, SV, input signals, and a reset input.

The timer/counter number refers to an actual address in the TC area. The numbers must not be duplicated. Refer to 3-7 Timer/Counter Area, for the range of timer/counter numbers available.

4-3-1
Timer - TIM

TIM measures in units of 0.1 second, and the set value (SV) can range from 0 to 999.9 seconds., with an accuracy of ±0.1 seconds.

Key Sequence

To set a timer, enter

[TIM] [N] [WRITE]

then the set value.
To assign a constant SV, enter

[#] [WRITE]
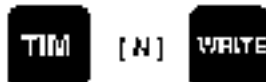
To externally set the timer value
(e.g., for a variable timer), enter
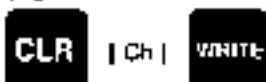
[CLR] [Ch] [WRITE]

[SHIFT] [CH/*] [Ch] [WRITE]

Example Circuit:
Ladder Diagram and
Mnemonic Code

Set value at 15 s

External timer setting via Ch 05
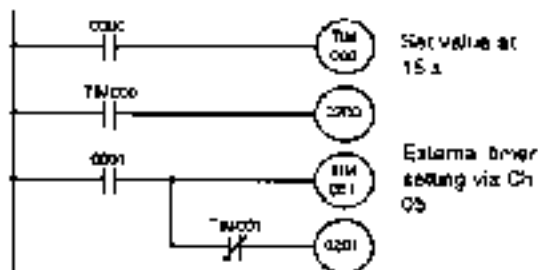
Address	Instruction	Data	
0000	LD		0000
0001	TIM		000
		#	0150
0002	LD	TIM	000
0003	OUT		0200
0004	LD		0001
0005	TIM		001
			05
0006	AND NOT	TIM	001
0007	OUT		0201

Timing Action of TIM 000

Timer input 000

ON-time output 000

15.0s

Timing Action of TIM 001

During RUN or MONITOR modes, a timer can be set according to the contents of a channel. In the second example above, Ch 05 is connected to an external device which sets the timer value, thus providing a variable timer.

Note: Timers inside an IL/ILC loop are reset when the IL condition goes OFF. A power failure will also reset timers.

As a countermeasure against having a timer reset when there is a power failure, use a counter and an internal clock for a timer, as shown below. This kind of timer preserves the present value of the timer.

6302 : (SR area) 1.0 s clock pulse
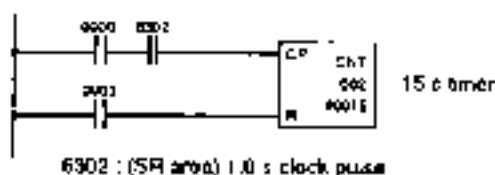
The externally set time value must be a BCD value, from 0 to 9999, or an error will occur. Although program execution will continue if a non-BCD value is used, timing accuracy cannot be expected.

Data Areas

IR, HR, LR, #

Flags

ER ———— The contents of the SV channel are not BCD.

4-3-2
High-Speed Timer - TIMH(15)

The high speed timer works the same as TIM except that TIMH measures in units of 0.01 second and its set value (SV) can range from 0.00 to 99.99 seconds with an accuracy of ±0.01 seconds.

Note: Timers inside an IL/ILC loop are reset when the IL condition goes OFF. A power failure will also reset timers.

Key Sequence

To set a high-speed timer, enter

[FUN] [1] [5] [N] [WRITE]

then the set value.
To assign a constant SV, enter

(#) [WRITE]

To externally set the high-speed timer value (e.g., for a variable timer).
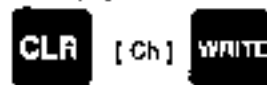
[CLR] [Ch] [WRITE]

[SHIFT] [CH ÷] [Ch] [WRITE]

Example Circuit:
Ladder Diagram and
Mnemonic Code

Assign a 1.5 s SV

External timer setting via Ch 06
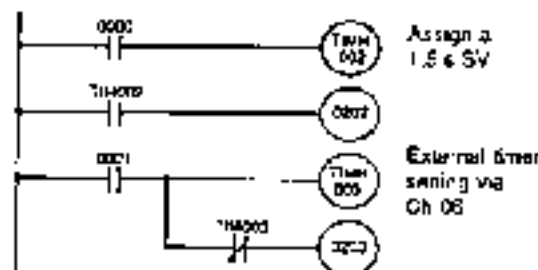
Address	Instruction	Data	
0000	LD		0000
0001	TIMH(15)		002
		#	0150
0002	LD	TIM	002
0003	OUT		0202
0004	LD		0001
0005	TIMH(15)		003
			06
0006	AND NOT	TIM	003
0007	OUT		0203

104

TIMH 002 Operation
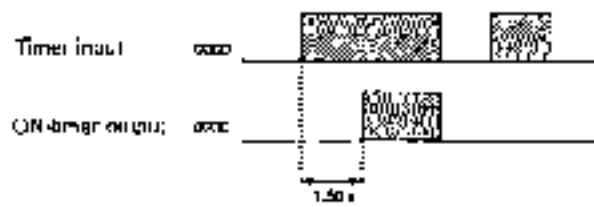
Timer input   00000

ON-timer output;   00000

1.50 s

Data Areas

IR, HR, LR

Flags

ER   The contents of the SV channel are not BCD
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been
exceeded.)

4-3-3
Counter - CNT

CNT is a preset decrementing counter. That is, it decrements its present count value (PV) when the count input signal goes from OFF to ON.

You must provide a count input, a reset input, SV, and a counter number to use CNT. The counter number refers to an actual address in the TC area. Since this area is also shared by timers, the numbers must not be duplicated. Refer to 3-7 Timer/Counter Area - TC for the range of timer/counter numbers available.

The counter set value can range from 0000 to 9999. It must be a BCD value.

CNT retains its PV within an IL/ILC loop when the IL condition is OFF.

Key Sequence

To set a counter, enter

CNT [N] WRITE

then the set value.
To assign a constant SV, enter

[#] WRITE

To externally set the counter value
(e.g., for a variable counter), enter

CLR [Ch] WRITE

or

SHIFT CH/* [Ch] WRITE

Example Circuit:
Ladder Diagram and
Mnemonic Code
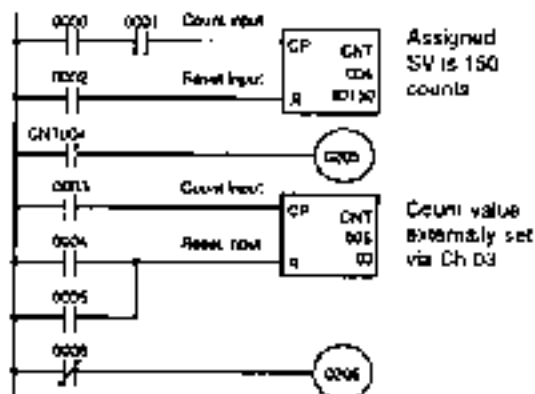
Address	Instruction	Data	
0000	LD		0000
0001	AND		0001
0002	LD		0002
0003	CNT		004
		#	0150
0004	LD	CNT	004
0005	OUT		0205
0006	LD		0003
0007	LD		0004
0008	OR		0005
0009	CNT		005
			03
0010	LD NOT		0006
0011	OUT		0205

Assigned
SV is 150
counts

Count value
externally set
via Ch 03

CNT 004 Operation

Counter input

Reset input

ON-counter output
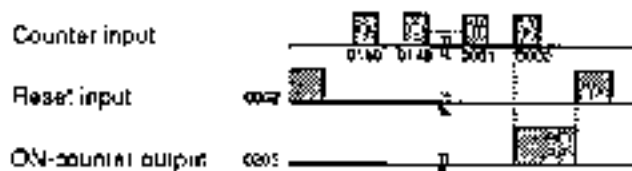
As a decrementing counter, CNT 004 produces an ON output when the PV becomes 0000. Once ON, the clock output stays ON until the reset signal turns ON. When the reset signal goes from OFF to ON, the PV is returned to the SV. While the reset signal is ON, count input signals are not accepted.

CNT 005 Operation

A counter can be externally set by specifying a channel through which the counter value in the program will be input from an external device. In the second example in the ladder diagram above, the contents of Ch 03 determine the set value of the counter in RUN and MONITOR mode. Thus, an externally connected device controls the variable setting of CNT 005 through Ch 03.

Data Areas

IR, HR, LR, #

Flags

ER – The contents of the SV channel are not BCD.

4-3-4
Reversible Counter -
CNTR(12)

The CNTR is a reversible, up-down circular counter. It increases or decreases the present value (PV) by one whenever the increment or decrement input signal, respectively, goes from OFF to ON. When both the increment and decrement input signals are ON at the same time, no counting is done.

Besides the increment and decrement input signals, you must also provide a reset input, SV, and a counter number, as with CNT. The counter number refers to an actual address in the TC area. Since this area is also shared by timers, the numbers must not be duplicated. Refer to 3-7 Timer/Counter Area - TC for the range of timer/counter numbers available.

When the reset signal is ON, the PV becomes 0000 and input signals are not accepted. When decremented from 0000, the present value (PV) is set to SV.

The counter set value can range from 0000 to 9999. It must be a BCD value.

CNTR retains its PV within an IL/ILC loop when the IL condition is OFF.

Key Sequence

To set the reversible counter, enter

FUN [1] [2] [N] **WRITE**

then enter the set value.
To assign a constant SV, enter

[#] **WRITE**

To externally set the counter value
(e.g., for a variable counter), enter

CLR [Ch] **WRITE**

or

SHIFT **CH/#** [Ch] **WRITE**

108

Example Circuit:
Ladder Diagram and
Mnemonic Code
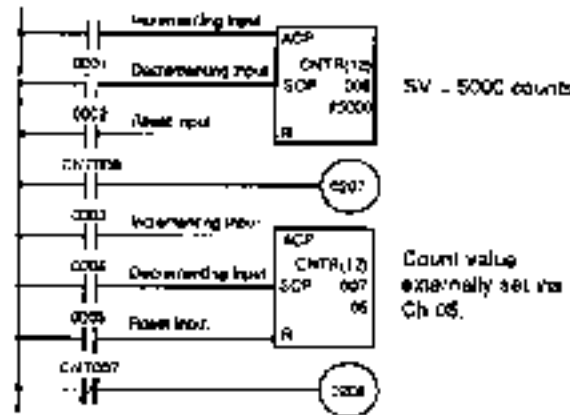
Address	Instruction	Data	
0000	LD		0000
0001	IC		0001
0002	LD		0002
0003	CNTR(12)		006
		#	5000
0004	LD	CNT	006
0005	OUT		0207
0006	LD		0003
0007	LD		0004
0008	LD		0005
0009	CNTR(12)		007
			05
0010	LD NOT	CNT	007
0011	OUT		0208

Counter Operation

Incremental input

Decremental Input

Counter output

Data Areas

IR, HR, LR, #

Flags

EH ——————— Externally set SV is not BCD.

4-3-5
Timer and Counter
Application Examples

Extended Timers

Extended timers can be formed by connecting TIMs in series. The following example shows two TIMs connected in series to form a 30 minute timer.
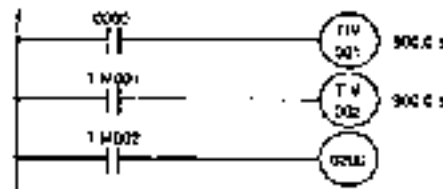
Address	Instruction	Data	
0000	LD		0000
0001	TIM	T	001
		#	9000
0002	LD	TIM	001
0003	TIM		002
		#	9000
0004	LD	TIM	002
0005	OUT		0200

TIMs can also be used in combination with CNT to form an extended timer. In the next example, TIM 001 generates a pulse every five seconds and CNT 002 counts the pulses. The total timing interval is found by:

$$\Delta T = (Timer\ SV + Scan\ Time) \times Number\ of\ Counts.$$

So, this is an example of a 500-second timer.
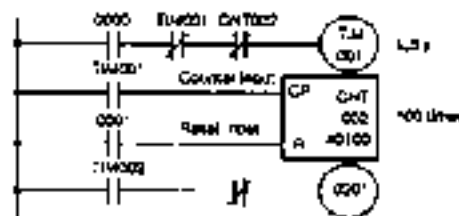
Address	Instruction	Data	
0000	LD		0000
0001	AND NOT	TIM	001
0002	AND NOT	CNT	002
0003	TIM		001
		#	0050
0004	LD	TIM	001
0005	LD		0001
0006	CNT		002
		#	0100
0007	LD	CNT	002
0008	OUT		0201

Another way to make an extended timer is by using one of the internal clock pulses (see 3-3 Special Relay Area) and a counter. In the following example, the PC's internal 1-second clock pulse is combined with CNT to form a 700-second timer. If the system reset flag (SR 6115), which turns ON for one scan when the PC starts operating, is ORd (i.e., connected in parallel) to the reset input of the counter, the counting operation starts from the set value when power is applied to the PC.
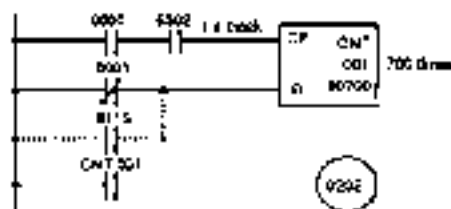
Address	Instruction	Data	
0000	LD		0000
0001	AND		6302
0002	LD NOT		0001
0003	CNT		001
		#	0700
0004	LD	CNT	001
0005	OUT		0202

Cumulative Counters

An extended counter that counts higher than 9,999 can be made by programming counters in series. The following is an example of a 20,000-count counter.
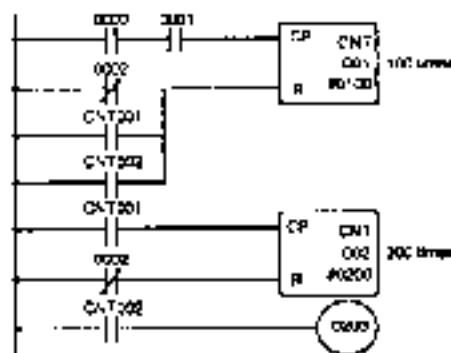
Address	Instruction	Data	
0000	LD		0000
0001	AND		0001
0002	LD NOT		0002
0003	OR	CNT	001
0004	OR	CNT	002
0005	CNT		001
		#	0100
0006	LD	CNT	001
0007	LD NOT		0002
0008	CNT		002
		#	0200
0009	LD	CNT	002
0010	OUT		0203

ON/OFF Delay Circuit
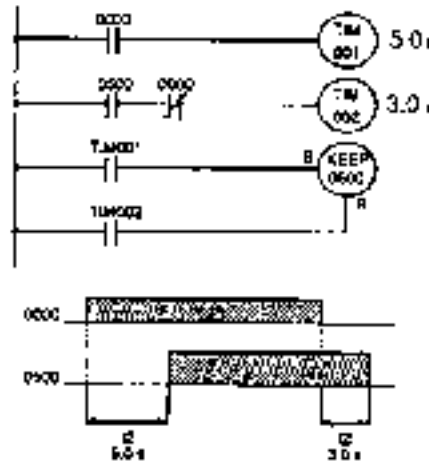
Address	Instruction		Data
0000	LD		0000
0001	TIM		001
		#	0050
0002	LD		0500
0003	AND NOT		0000
0004	TIM		002
		#	0030
0005	LD	TIM	001
0006	LD	TIM	002
	KEEP(11)		0500

One Shot Timer

The one shot timer shown below outputs a pulse for 1.5 seconds (the value of TIM 001's SV) after its input turns ON. (The duration of the ON input signal must be greater than the scan time.)
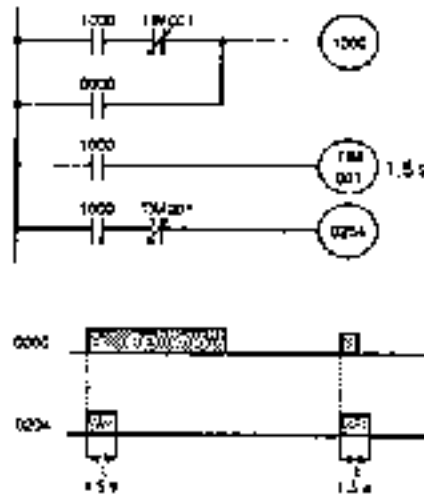
Address	Instruction		Data
0000	LD		1000
0001	AND NOT	TIM	001
0002	OR		0000
0003	OUT		1000
0004	LD		1000
0005	TIM		001
		#	0015
0006	LD		1000
0007	AND NOT	TIM	001
0008	OUT		0204
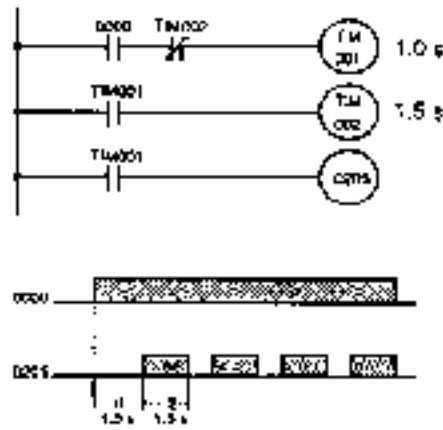
112

Flicker Circuit

Address	Instruction	Data	
0000	LD		0000
0001	AND NOT	TIM	C02
0002	TIM	.	001
		#	0010
0003	LD	TIM	001
0004	TIM		002
		#	0015
0005	LD	TIM	001
0006	OUT		0205

4-4
Data Shifting

The Instructions described in this chapter all shift data, but in differing amounts and directions. Each of the shift instructions is programmed with a function code.

To input these instructions through the Programming Console, you must press the FUN key followed by the appropriate function code. Data for the operands also has to be entered where required. Refer to the key sequences for each instruction.
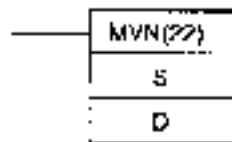
4-4-1
Shift Register - SFT(10)

SFT shifts the data by 1 bit in the specified channels. Both a beginning channel (B) and an end channel (E) must be specified as data.
E must be less than or equal to B, and B and E must be in the same data area.

Ladder Symbol and Key Sequence

Data input — IN
Shift pulse input — SP   SFT(10)   B
Reset signal input — R   E

B : Beginning channel
E : End channel

To left-shift data, enter
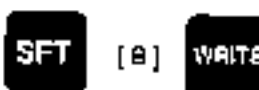
FUN 1 0 [B] WRITE

[E] WRITE

or

SFT [B] WRITE

[E] WRITE

**Example Circuit:
Ladder Diagram and
Mnemonic Code**
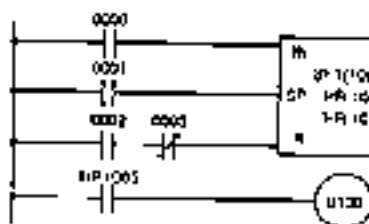
Address	Instruction	Data	
0000	LD		0000
0001	LD		0001
0002	LD		0002
0003	AND NOT		0003
0004	SFT(10)	HR	10
		HR	10
0005	LD	HR	1005
0006	OUT		0100

In the above example, HR 10 acts as a 16-bit shift register. An ON shift pulse signal shifts the data in HR 10 one bit to the left. This moves HR 1000 to 1014 into HR 1001 to 1015 and the bit content of HR 1015 is lost. Output 0100 turns ON when the bit content of HR 1005 is 1.

HR 1015	HR 1014	...	HR 1001	HR 1000
1/0	1/0		1/0	1/0

Lost data ←   ← Data Input 1/0

Timing

A reset input sets all bits to 0. The data are shifted at the leading edge of the clock input. If the HR area is used, data is retained during power failure. The content of bit 15 of channel E is lost each time the data is shifted.

Shifting More Than 16 Bits at a Time

The next example shows a 48-bit shift register which uses a 1-second clock pulse as the shift pulse signal. The contents of input 0005 are shifted across channels 28 to 30.

Data Areas

IR, HR, LR

4-4-2
Reversible Shift Register - SFTR(84)

SFTR shifts data in a specified channel or series of channels to either the left or right. A beginning and end channel, B and E, must be specified, even for shifting data within a single channel. B must be less than or equal to E, and B and E must be in the same data area. Also, a control channel which contains the reset input, shift pulse input, and data input must be provided.

Ladder Symbol and Key Sequence

```
            SFTR(84)
              C
              B
              E
```
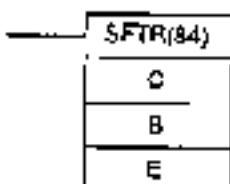
C: Control channel
B: Beginning channel
E: End channel

For SFTR, enter

[FUN] [8] [E 4] [WRITE]

[C] [WRITE]

[B] [WRITE]

[E] [WRITE]

Control Channel Data

```
| 15 | 14 | 13 | 12 |    -    | 00 |
```

Direction control
1 (ON): Shift to the left
0 (OFF): Shift to the right

Data input (IN)

Shift pulse input (SP)
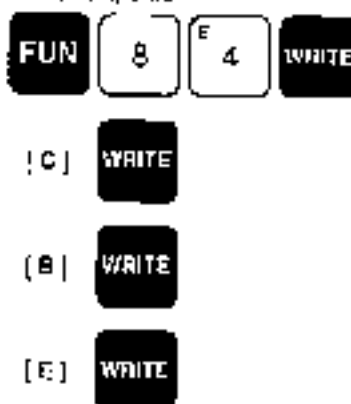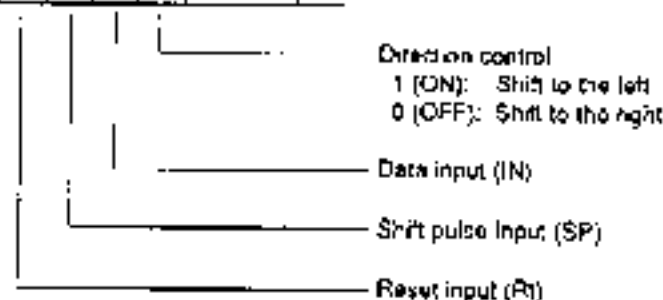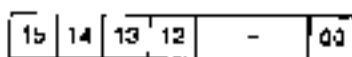
Reset input (R)

Control Channel Operation

When the reset input is applied to SFTR, (i.e. when bit 15 of the control channel is turned ON), all the bits of the control channel and the carry flag are cleared to 0, disabling SFTR from accepting any input.

117

When the data is being shifted to the left (from bit 00 toward bit 15), (bit 12 of the control channel is ON), the content of bit 13 of the control channel (data input) is transferred to bit 00 of channel B on the leading edge of the shift pulse (bit 14 of the control channel). As a result, all the data in channel B is shifted toward bit 15. At the same time, bit 15 is transferred to the carry flag.

When the data is being shifted to the right (from bit 15 to bit 00), (bit 12 of the control channel is OFF), the data input is transferred to bit 15 of channel E. As a result, all the data is shifted 1 bit toward bit 00. At the same time, the content of bit 00 of channel R is shifted to the carry flag.

**Example Circuit:
Ladder Diagram and
Mnemonic Code**
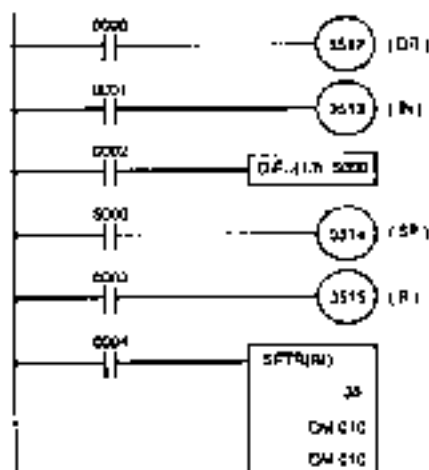
Address	Instruction	Data	
0000	LD		0000
0001	OUT		3512
0002	LD		0001
0003	OUT		3513
0004	LD		0002
0005	DIFU(13)		5000
0006	LD		5000
0007	OUT		3514
0008	LD		0003
0009	OUT		3515
0010	LD		0004
0011	SFTR(84)		—
			35
		DM	010
		DM	010

Data Areas | IH, HR, LH, DM, *DM |

Flags   ER ———— The B and E channels are in different areas, or B > E
                                      Indirectly addressed DM channel is non-existent.
                                      (DM data is not BCD, or the DM area boundary has been
   exceeded.)
                             CY ---- Receives the data of bit 00 or bit 15, depending on the direction
                                      of the shift.

4-4-3
Arithmetic Shift Left -
ASL(25)

ASL shifts a single channel of data one bit to the left, with carry (CY).

Ladder Symbol and
Key Sequence

```
————| ASL(25) |
     |   Ch    |
```
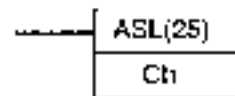
Ch: Channel whose data
    is to be shifted

For ASL, enter
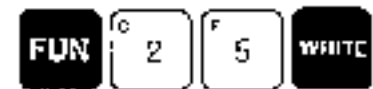
FUN [²c 2] [ᶠ 5] **WRITE**

[Ch] **WRITE**

Example Circuit:
Ladder Diagram and
 Mnemonic Code

```
| 0000                    ASL(25)
|——| |————————————————————| DM 010 |
```
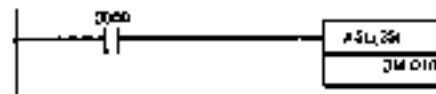
Address	Instruction	Data
0000	LD	0000
0001	ASL(25)	—
	DM	010

Example

Contents of Ch Before Shift

CY | Bit 15 ... Bit 00
□ | [1 0 0 1 1 1 0 0 0 1 0 1 0 0 1 1]

Contents of Ch After Shift

CY | Bit 15 ... Bit 00
[1] | [0 0 1 1 1 1 0 0 0 1 0 1 0 0 1 0]

Data Areas

| IR, HR, LR, OM, *DM |

Flags

EH ———— Indirectly addressed DM channel is non-existent.
         (DM data is not BCD, or the DM area boundary has been
   exceeded.)

CY ———— Receives the data of bit 15.

EQ ———— ON when the contents of the data channel are 0000; otherwise
         OFF.

Arithmetic Shift Right -
ASR(26)

ASR shifts a single channel of data one bit to the right, with carry (CY). Zero is always shifted into bit 15. In all other respects it is the same as ASL.

Ladder Symbol

```
┌─────────────┐
│  ASR(26)    │
│             │
│     Ch      │
└─────────────┘
```
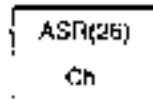
Ch: Channel to be shifted

When a zero or a one is shifted into bit 15, all bits are shifted to the right and bit 00 is shifted into CY.

Example

4-4-4
Rotate Left - ROL(27)

ROL rotates a single channel of data one bit to the left, with carry (CY). ROL shifts CY into bit 00 of the specified channel, and bit 15 into CY.
Use STC(41) or CLC(41) to force-set or force-reset the contents of CY before doing a rotate operation.

Ladder Symbol and Key Sequence

```
┌──────────┐
│ ROL(27)  │
├──────────┤
│   Ch     │
└──────────┘
```
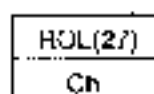
Ch: Channel to be rotated

For ROL, enter

[FUN] [2] [7] [WRITE]
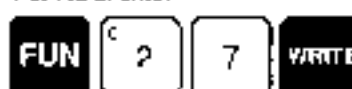
[Ch] [WRITE]

Channel data movement is as below.
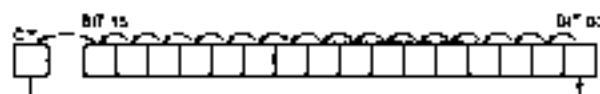
Examlpe Circuit: Ladder Diagram and Mnemonic Code
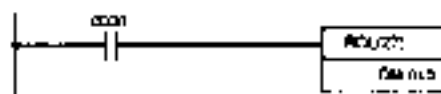
Address	Instruction	Data	
0000	LD		0000
0001	ROL(27)		—
		DM	010

Example
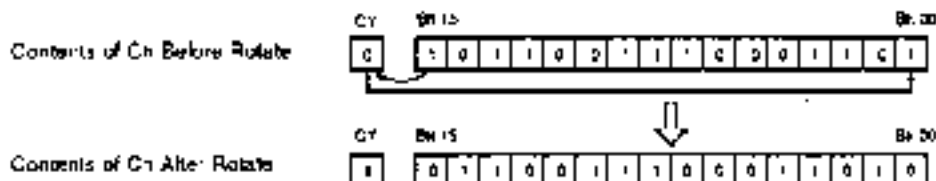
Contents of Ch Before Rotate

Contents of Ch After Rotate

Data Areas

```
IR, HR, LR, DM, *DM
```

Flags

ER ———— Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

CY ———— Receives the data of bit 15.

EQ ———— ON when contents of the data channel are 0000; otherwise, OFF.

Rotate Right - ROR(28)

ROR rotates a single channel of data one bit to the right, including carry (CY).
ROR shifts CY into bit 15 of the specified channel, and bit 00 into CY.
Use STC(41) or CLC(41) to force-set or force reset the contents of CY before doing a rotate operation.

Ladder Symbol

```
        ┌─────────┐
 ───────┤ ROR(28) │
        ├─────────┤
        │   Ch    │
        └─────────┘
```
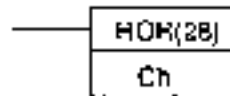
Ch: Channel to be rotated

Channel data movement is as below:

Example

Contents of Ch Before Rotate

Contents of Ch After Rotate
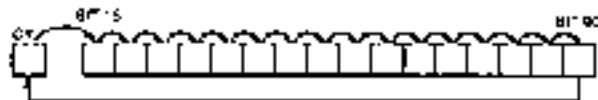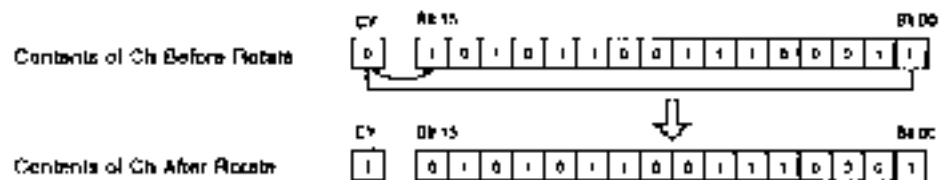
Data Areas and Flags

These are identical to ROL except that CY receives the data of bit 00 instead of the data of bit 15.

123

4-4-5

__One Digit Shift Left - SLD(74)__ SLD left shifts data between the beginning and end channels by one digit (four bits). When channels of data are shifted with SLD, zero is written into the first digit of the beginning channel, and the leftmost digit of the end channel is lost.

Ladder Symbol and Key Sequence

```
        ┌─────────────┐
────────│  SLD(74)    │
        ├─────────────┤
        │      B      │
        ├─────────────┤
        │      E      │
        └─────────────┘
```
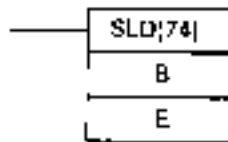
B. Beginning channel
E End channel

For SLD, enter

FUN **7** **4** **WRITE**
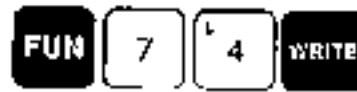
[B] **WRITE**

[E] **WRITE**

B and E must be in the same data area, and E must be greater than or equal to B.

Example Circuit: Ladder Diagram and Mnemonic Code

Address	Instruction	Data	
0000	LD		0000
0001	SLD(74)		—
		DM	010
		DM	011

Example

B and E before SLD

B and E after SLD
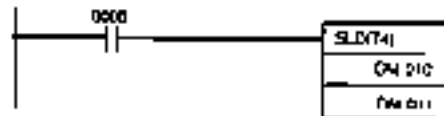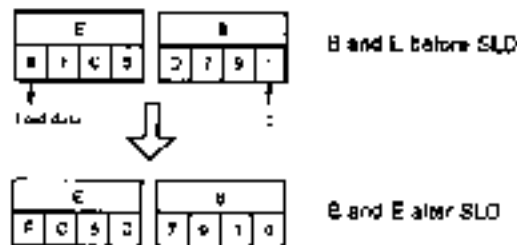
One Digit Shift Right -
SRD(75)

SRD shifts data between the beginning and end channels by one digit (four bits) to the right. Zero is written into the leftmost digit of the beginning channel and the rightmost digit of the end channel is lost.

Ladder Symbol

```
        ┌─────────────┐
   ─────┤   SRD(75)   │
        ├─────────────┤
        │      B      │
        ├─────────────┤
        │      E      │
        └─────────────┘
```
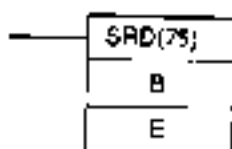
B: Beginning channel
E: End channel

B and E must be in the same data area, and E must be greater than or equal to B.
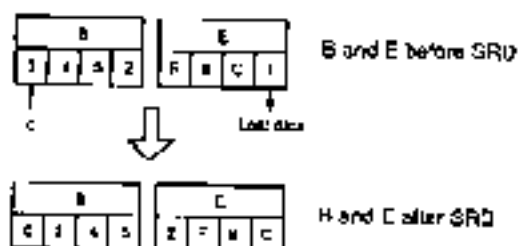
Example

B and E before SRD

B and E after SRD

Data Areas

```
IR, HR, LR, DM, *DM
```

Flags

ER ────── The B and E channels are in different areas, or B > E.
Indirectly addressed DM channel is non-existent
(DM data is not BCD, or the DM area boundary has been
exceeded.)

Note: If a power failure occurs during a shift operation across more than 50 channels, the shift operation may not be completed.

4-4-6
Word Shift - WSFT(16)

WSFT left shifts data between the beginning and end channels in channel units. Zeros are written into the beginning channel and the contents of the end channel are lost.

Ladder Symbol and Key Sequence


```
┌─── WSFT(16) ┐
│      B      │
│      E      │
└─────────────┘
```
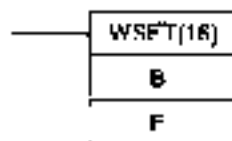
B: Beginning channel
E: End channel

For WSFT, enter
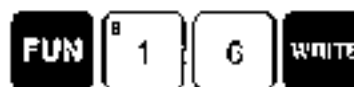
| FUN | 1 | 6 | WRITE |

| B | WRITE |

| E | WRITE |

Example Circuit: Ladder Diagram and Mnemonic Code
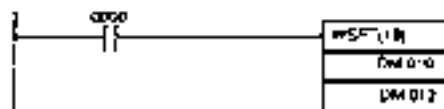
Address	Instruction	Data	
CC00	LD		0000
CC01	WSFT(16)		—
		DM	010
		DM	011

Example
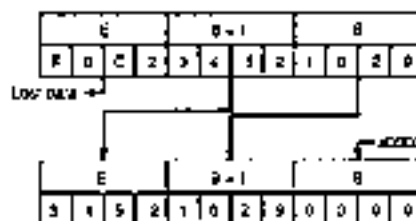
Data Areas

| IR, HR, LR, DM, *DM |

Flags

LR | The B and E channels are in different areas, or B > E.
Indirectly addressed DM channel is non existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

4-5
Data Movement

This section describes the instructions used for moving data between data areas. Data movement is essential for utilizing all of the internal data areas of the PC. Communication in linked systems also requires data movement.

Each instruction is programmed with a function code. To input these instructions through the Programming Console, press the FUN key followed by the appropriate function code. Data for the operands also has to be entered where required. Refer to the key sequences for each instruction

4-5-1

Move - MOV(21)
Move Not - MVN(22)

MOV transfers source data (either the data in a specified channel or a four-digit hexadecimal constant) to a destination channel.

MVN inverts the source data and then transfers it to the destination channel.

Ladder Symbols and Key Sequences
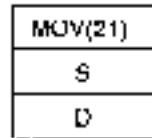
MOV(21)
S
D

S: Source data
D: Destination channel

MVN(22)
S
D

S: Source data
D: Destination channel

For MOV, enter
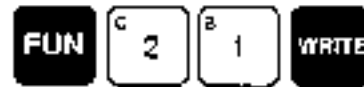
FUN **2** **1** **WRITE**
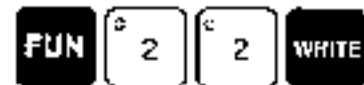
[S] **WRITE**

[D] **WRITE**

For MVN, enter

FUN **2** **2** **WRITE**

[S] **WRITE**

[D] **WRITE**

Example Circuit: Ladder Diagram and Mnemonic Code
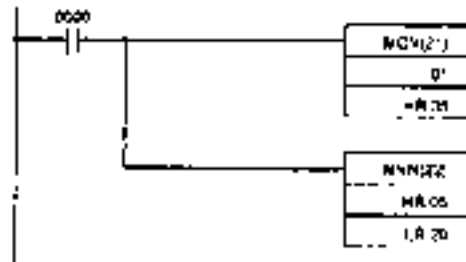
Address	Instruction	Data	
0000	LD		0000
0001	MOV(21)		—
			01
		HR	05
0002	MVN(22)		—
		HR	05
		LR	20

In the example circuit above, when input 0000 turns ON, MOV transfers the contents of channel 01 to HR 05, and MVN inverts the contents of HR 05 and transfers them into LR 20.
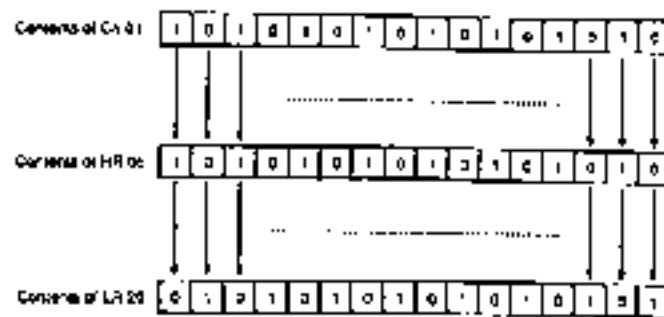
Data Areas

Source	Destination
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Timers and counters cannot be designated as destinations of MOV(21) or MVN(22). However you can change a timer's PV or a counter's PV by program control with BSET(71).

Flags

ER——— Indirectly addressed DM channel is non-existent. (DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ON when source and destination channel contents are 0000; otherwise OFF.

4-5-2
Block Set - BSET(71)

BSET copies a channel of data or a constant number to several consecutive channels.

Both a beginning channel (B) and an end channel (E) must be specified as data. B must be less than or equal to E, and B and E must be in the same data area.

BSET can be used to change timer/counter data. (This cannot be done with MOV(21) or MVN(22).)

Ladder Symbol and Key Sequence

```
──┤ BSET(71) │
   │   S     │
   │   B     │
   │   E     │
```
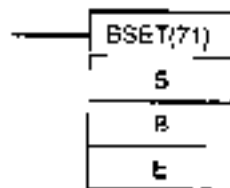
S: Source data
B: Beginning channel
E: End channel

For BSET, enter
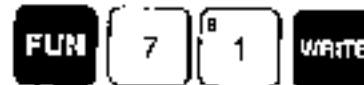
FUN [7] [1] **WRITE**

then the source and destination channel numbers.

[S] **WRITE**

[B] **WRITE**

[E] **WRITE**

Example Circuit: Ladder Diagram and Mnemonic Code
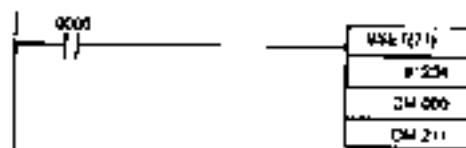
Address	Instruction	Data	
0000	LD		0000
0001	BSET(71)		—
		#	1234
		DM	000
		DM	211

Example
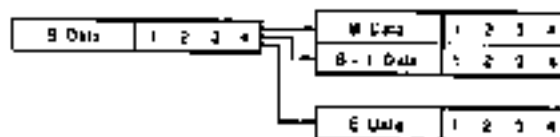
Data Areas

S	B and E
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER — The B and E channels are in different areas, or B > E.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

130

4-5-3
Block Transfer - XFER(70)

XFER moves the contents of several consecutive source channels to consecutive destination channels.

Ladder Symbol and Key Sequence

```
        ┌──────────────┐
────────┤   XFER(70)   │
        ├──────────────┤
        │      N       │
        ├──────────────┤
        │      S       │
        ├──────────────┤
        │      D       │
        └──────────────┘
```
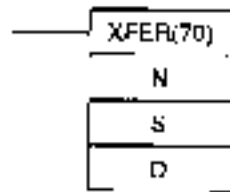
N: Number of channels to move
S: Source beginning channel
D: Destination beginning channel

For XFER, enter

FUN **7** **0** **WRITE**
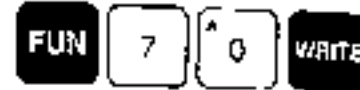
then the number of channels and the source beginning and destination beginning channel numbers.

[N] **WRITE**

[S] **WRITE**

[S] **WRITE**

N, the number of channels to be transferred, must be a 4-digit BCD number. Both the source and destination channels may be in the same data area, but their respective block areas must not overlap.

Example Circuit: Ladder Diagram and Mnemonic Code
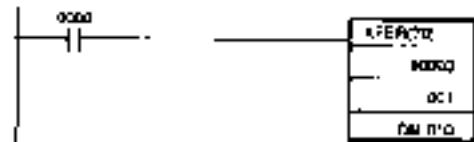
Address	Instruction	Data	
0000	LD		0000
0001	XFER(70)		—
		#	0003
			001
		DM	010

Example

These 3 are the channels to be transferred
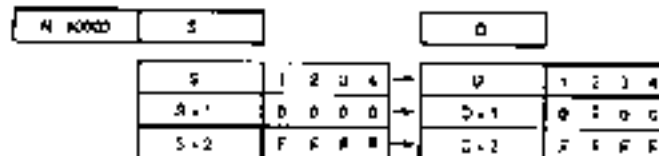
Data Areas

N	S and D
#	IR, LR, HR, TC, DM, *DM

Flag

LR   -   The specified number of channels to transfer is not BCD, or when added to the beginning channel number, the specified area is exceeded.

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

4-5-4
Data Exchange -
XCHG(73)

XCHG exchanges the data of two different channels. If you want to exchange the data between 2 blocks whose size is greater than 1 channel, use another data area as an intermediate buffer and transfer data to that area with XFER(70).

Ladder Symbol and Key Sequence

```
          XCHG(73)
            E1
            E2
```
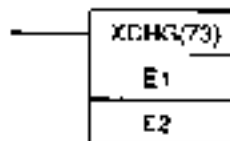
E1: Exchange channel 1
E2: Exchange channel 2

For XCHG, enter

[FUN] [7] [3] [WRITE]
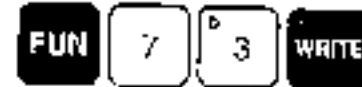
then the exchange channel numbers.
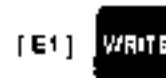
[E1] [WRITE]
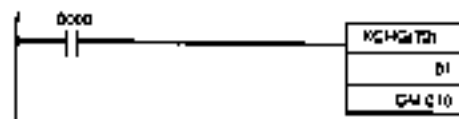
[E2] [WRITE]

Example Circuit: Ladder Diagram and Mnemonic Code

Address	Instruction	Data
0000	LD	0000
0001	XCHG(73)	—
		DI
		DM 010

Example

Data Areas

IR, HR, LR, TC, DM, *DM

Flag

ER ———— Indirectly addressed DM channel is non-existent
(DM data is not BCD, or the DM area boundary has been exceeded.)

4-5-5

Single Channel Distribution - DIST(80)

DIST moves one channel of data to a destination channel whose address is given by a destination base channel + an offset. That is, the offset is added to the destination base channel to determine the actual destination channel.

The offset data must be a 4-digit BCD value. Also, the final destination address must be in the same data area as the destination base channel.

Ladder Symbol and Key Sequence

DIST(80)
S
DBs
Of
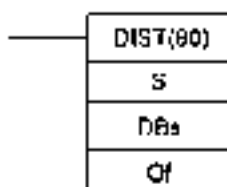
S: Source data
DBs: Destination base channel
Of: Offset data

For DIST, enter

FUN **8** **0** **WRITE**
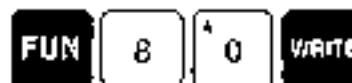
then the source and destination base channel numbers and the offset.

[S] **WRITE**

[DBs] **WRITE**

[Of] **WRITE**

Example Circuit: Ladder Diagram and Mnemonic Code

| 0000 |
| DIST(80) |
| 01 |
| DM 000 |
| 35 |

Address	Instruction	Data	
0000	LD		0000
0001	DIST(80)		
			01
		DM	000
			35
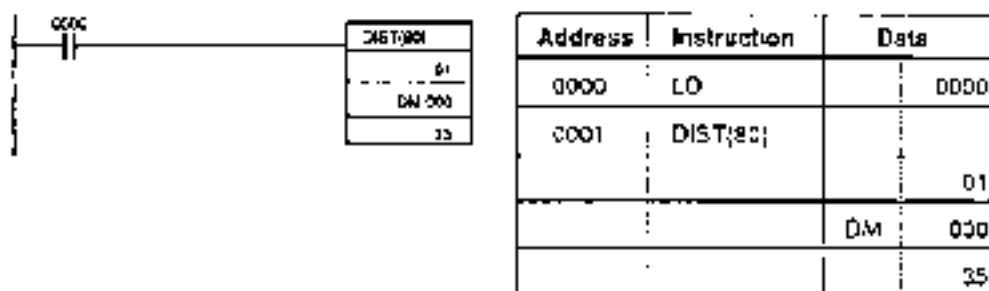
In the example above, if the content of channel 35 is 5 then the destination channel is DM 005.

Data Areas

S	DB
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, TC, DM, *DM

Of
IR, HR, LR, TC, DM, *DM, #

Flags   ER ———— The specified offset data is not BCD, or when added to the
destination base channel, the specified area is exceeded.
Indirectly addressed DM channel is non-existent
(DM data is not BCD, or the DM area boundary has been
exceeded.)

                             EQ ———— ON when source channel contents are 0000; otherwise OFF

4-5-6
Data Collection-
COLL(81)

COLL extracts data from the source channel and writes it to a destination channel. The address of the actual source channel is determined by adding the offset to the source channel.

The offset data must be a 4-digit BCD value. Also, the final source channel must be in the same data area as the source base channel.

Ladder Symbol and Key Sequence

```
┌─────────────┐
│  COLL(81)   │
├─────────────┤
│    SBs      │
├─────────────┤
│    Of       │
├─────────────┤
│    D        │
└─────────────┘
```
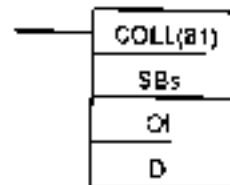
SBs: Source base channel
Of: Offset data
D: Destination channel

For COLL, enter

[FUN] [8] [1] [WRITE]
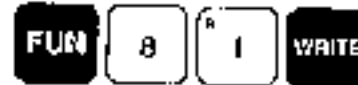
then the source base and destination channel numbers and the offset.
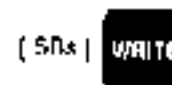
[SBs] [WRITE]

[Of] [WRITE]

[D] [WRITE]

Example Circuit: Ladder Diagram and Mnemonic Code
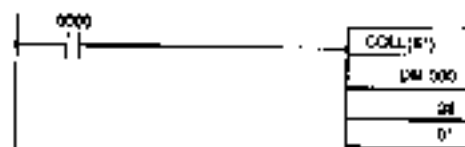
Address	Instruction	Data	
0000	LD		0000
0001	COLL(81)		—
			01
		DM	000
			35

In the example above, if the content of channel 35 is 5 then the source channel is DM 005.

Data Areas

SBs	Of
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, TC, DM, *DM, #

D
IR, HR, LR, TC, DM, *DM

Flags

ER────── The specified offset data is not BCD, or when added to the
source beginning channel, the specified area is exceeded.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been
exceeded.)

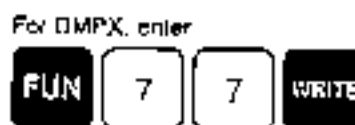EQ ──── · ON when source channel contents are 0000.

4-5-7
Move Bit - MOVB(82)

MOVB transfers the specified bit to another specified bit. The specifications for the source and destination bits are made in a control channel (or input as a constant) in BCD.

Ladder Symbol and Key Sequence

MOVB(82)
S
C
D
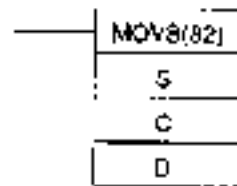
S: Source data
C: Control data
D: Destination channel

For MOVB, enter
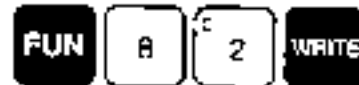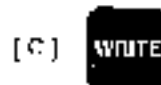
FUN | 8 | 2 | WRITE

then the source data and control data and destination channel number.

[S] WRITE

[C] WRITE

[D] WRITE

Control Data

Bit 15   Bit 00

Source bit specification (BCD - 0 to 15)
Destination bit specification (BCD - 0 to 15)

Example Circuit: Ladder Diagram and Mnemonic Code

0000

MOVB(82)
DM 000
00
15

Address	Instruction	Data	
0000	LD		0000
0001	MOVB(82)		—
		DM	000
			25
			15

Example

Control Data

| 0 | 0 | 0 | | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

    1   2   0   2

Source channel or data

Destination channel

Data Areas

S
IR, SR, HR, LR, DM, *DM, #
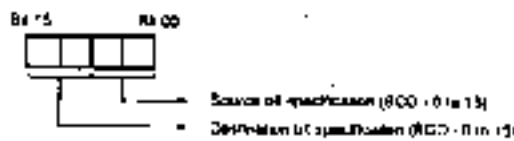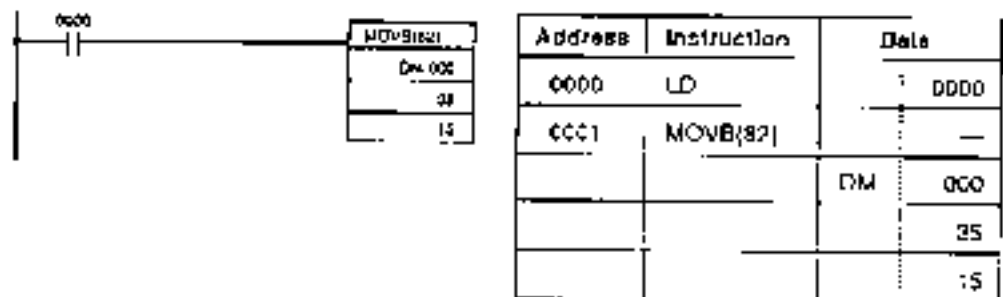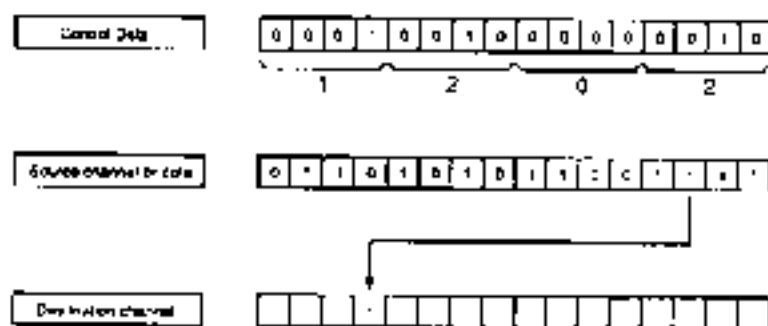
C
IR, HR, LR, TC, DM, *DM, #

D
IR, HR, LR, TC, DM, *DM

Note: S and C can be constants within the range 0000 to FFFF.

Flag

ER ———— Control data is not BCD, or is specifying a non-existent bit (i.e., bit specification must be 00 to 15)

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

4-5-8
Move Digit - MOVD(83)

MOVD moves the hexadecimal contents of the specified 4-bit source digit to the specified destination digit. Up to four digits can be transferred at one time with this instruction.

Ladder Symbol and Key Sequence

MOVD(83)
S
C
D

S: Source data
C: Control data (0,1,2,3)
D: Destination channel

For MOVD, enter
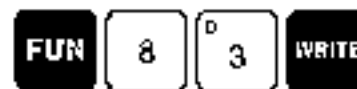
FUN **8** **3**(D) **WRITE**

then the source data and control data and destination channel number.

[S] **WRITE**

[C] **WRITE**

[D] **WRITE**

Control Data

Bit 15   Bit 00

Specifies which source digit (0 to 3) to move first
Number of digits to be moved (0 to 3)
0,1 digit (4 bits)
1,2 digits (8 bits)
2,3 digits (12 bits)
3,4 digits (16 bits)
Specifies the first destination digit (0 to 3)
Not used
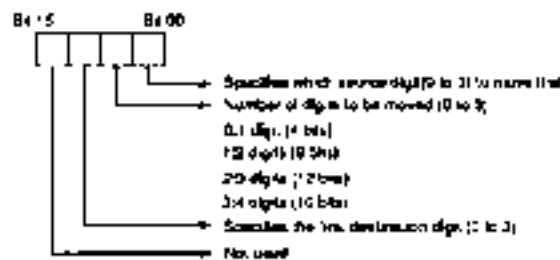
Note: If four digits are to be moved, and the first destination digit is "2", the data are transferred to destination digits 2, 3, 0, and then 1.

Example Circuit: Ladder Diagram and Mnemonic Code

0000

MOVD(83)
CH 000
35
CH 003
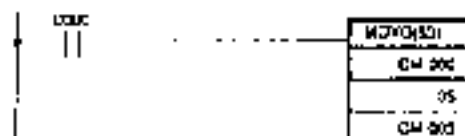
Address	Instruction	Data	
0000	LD		0000
0001	MOVD(83)		—
		DM	000
			35
		DM	003

Example
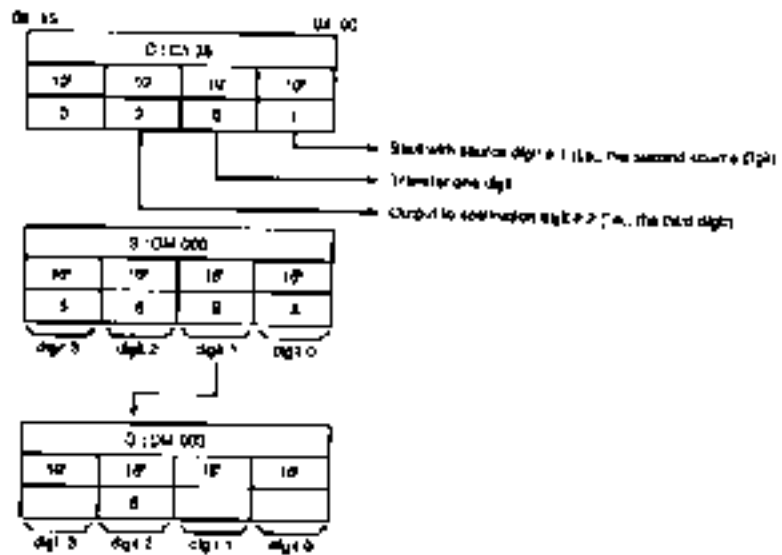
Examples of Multiple Digit Transfer

Data Areas

S	C
IR, SR, HR, LR, TC, DM, *DM , #	IR, HR, LR, TC, DM, *DM, #

D
IR, HR, LR, TC, DM, *DM
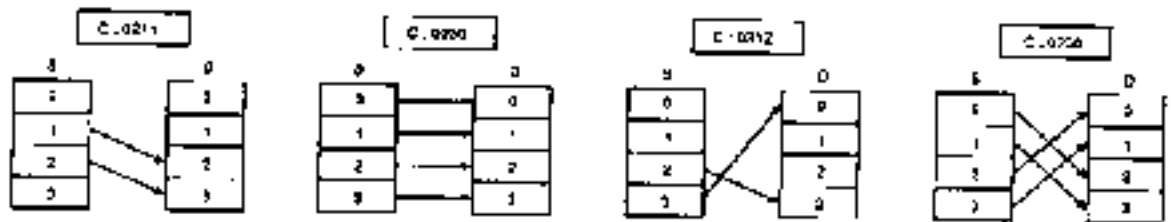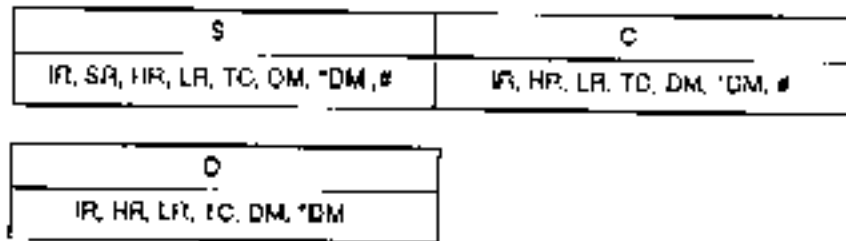
Flag

ER —— Control data is specifying a digit other than 0, 1, 2, or 3.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

4-6
Data Comparison Instructions

This section describes the instructions used for comparing data. Direct comparisons for equality are possible as well as range checks.

Each data comparison instruction is programmed with a function code. To input these instructions through the Programming Console, press the FUN key followed by the appropriate function code.

4-6-1
Compare - CMP(20)

CMP compares two sets of 4-digit hexadecimal data and outputs the results to the GR, EQ, and LE flags in the SR area. (Refer to 3-3-9.)

Ladder Symbol and Key Sequence

```
      CMP(20)
        C1
        C2
```

For CMP, enter

FUN **2** **0** **WRITE**
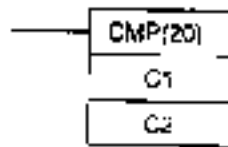
C1: Compare data 1
C2: Compare data 2
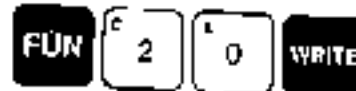
Example Circuit:
Ladder Diagram and
Mnemonic Code
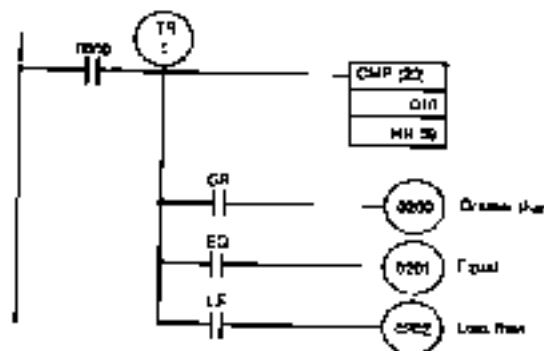
Address	Instruction	Data	
0000	LD		0000
0001	OUT	TR	0
0002	CMP(20)		
			C10
		KH	09
0003	AND		6305
0004	OUT		0200
0005	LD	TR	0
0006	AND		6306
0007	OUT		0201
0008	LD	TR	0
0009	AND		6307
0010	OUT		0202

Data Areas

IR, SR, HR, LR, TC, DM, *DM, #

Flags   ER | Indirectly addressed DM channel is non-existent.
                                               | (DM data is not BCD, or the DM area boundary has been
                                               | exceeded.)

                                       EQ ————| See table below
                                       LE ————| See table below.
                                       GR ————| See table below.

	C1>C2	C1=C2	C1<C2
EQ	OFF	ON	OFF
LE	OFF	OFF	ON
GR	OFF	OFF	OFF

Note: Instructions immediately preceding a compare may affect the EQ,
      LE, and GR flags.

4-6-2
Table Compare-
TCMP(85)

TCMP compares a 4-digit value with values in a table of 16 channels. If the compare value equals a value in the table, TCMP sets the corresponding bit in the result channel.

Ladder Symbol and Key Sequence

```
——————| TCMP(85) |
        |    CD    |
        |    TB    |
        |    R     |
```
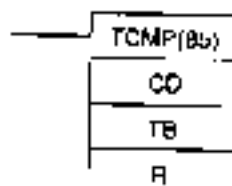
CD: Compare data
TB: First channel of the compare table
R: Result channel

For TCMP, enter

[FUN] [8] [5] [WRITE]

then the compare data and compare table beginning channel and result channel.

[CD] [WRITE]

[TB] [WRITE]

[R] [WRITE]

If, for example, the contents of channel TB equals CD, bit 00 of the result channel is set to 1. If not, bit 00 will be cleared to 0.

Example
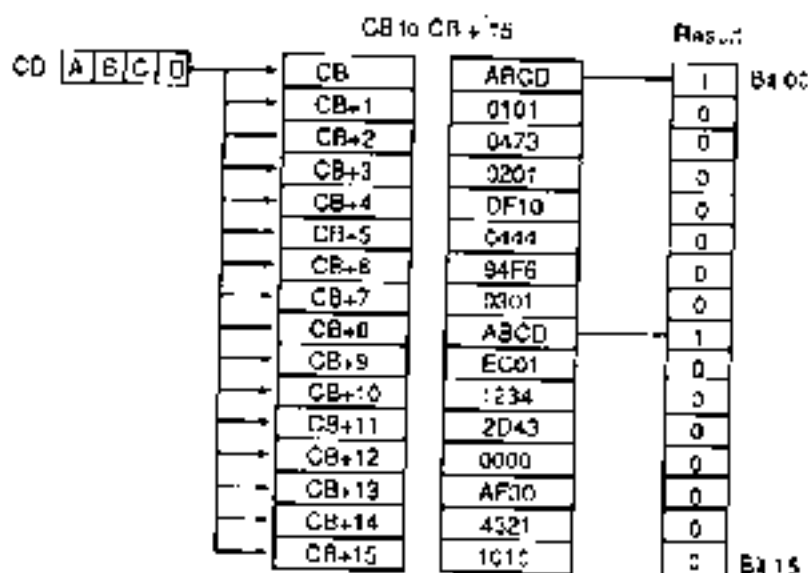
CD		CB to CB + 15		Result	
A B C D	CB	ABCD		1	Bit 00
	CB+1	0101		0	
	CB+2	0473		0	
	CB+3	0201		0	
	CB+4	DF10		0	
	CB+5	C444		0	
	CB+6	94F6		0	
	CB+7	3301		0	
	CB+8	ABCD		1	
	CB+9	EC01		0	
	CB+10	1234		0	
	CB+11	2D43		0	
	CB+12	0000		0	
	CB+13	AF00		0	
	CB+14	4321		0	
	CB+15	1010		0	Bit 15

Example Circuit:
Ladder Diagram and
Mnemonic Code

Address	Instruction	Data	
0000	LD	:	0000
0001	TCMP(85)	:	
		:	01
		DM :	000
		:	35
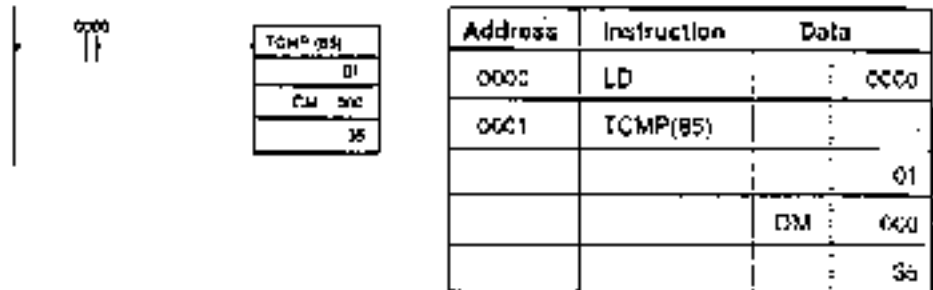
In the example above, when input 0000 turns ON, the contents of channel 01
are compared with each of the channels from DM 000 to DM 015. Where the
two sets of data are equal, TCMP sets the corresponding bit of channel 35.
When the data does not match a channel in the table, a "0" is written to the
corresponding bit in channel 35.

Data Areas

CD	TB and R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, TC, DM, *DM

Flags

ER ———— The compare table (i.e., TB through TB + 15) exceeds the
data area.
Indirectly addressed DM channel is non-existent
(DM data is not BCD, or the DM area boundary has been
exceeded.)

4-7
Data Conversion Instructions

The conversion instructions convert channel data that is in one format into another format and output the converted data to specified output channel(s).

Each data conversion instruction is programmed with a function code. To input these instructions through the Programming Console, press the FUN key followed by the appropriate function code.

4-7-1
<u>BCD to Binary - BIN(23)</u>

BIN converts four-digit, decimal data in one channel into 16-bit binary data, and outputs the converted data to a result channel.

Example

Source channel
3	4	5	2
x 10³	x 10²	x 10¹	x 10⁰

Result channel
3	5	7	6
x 2⁰	x 2¹⁰	x 2⁵	x 2¹⁰

Ladder Symbol and Key Sequence

```
        ┌──────────┐
────────┤  BIN(23) │
        ├──────────┤
        │    S     │
        ├──────────┤
        │    R     │
        └──────────┘
```

S: Source channel
R: Result channel

For BIN, enter
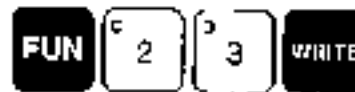
FUN **2** **3** **WRITE**

Enter the source and destination channel numbers.

[S] **WRITE**

[R] **WRITE**

**Example Circuit:
Ladder Diagram and
Mnemonic Code**

```
  0000
───┤├──────────┌──────────┐
                │ BIN(23)  │
                ├──────────┤
                │    10    │
                ├──────────┤
                │  HR 20   │
                └──────────┘
```
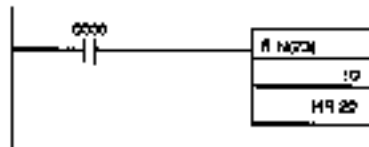
Address	Instruction	Data	
0000	LD		0000
0001	BIN(23)		—
			10
		HR	20

In the example above, when input 0000 turns ON, the BCD contents of channel 10 are converted into binary format and are then output to HR 20.

Data Areas

S	R
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, DM, *DM

Flags

ER ——— The contents of the source channel are not BCD
Indirectly addressed DM channel is non-existent
(DM data is not BCD, or the DM area boundary has been
exceaded.)

EQ · ON when the contents of the result channel are 0000.

4-7-2

Binary to BCD - BCD(24)

The BCD (binary-to-BCD conversion) instruction converts 16-bit binary data in one channel into four-digit, decimal data, and outputs the converted data to another channel

Example
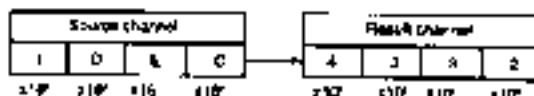
Ladder Symbol and Key Sequence
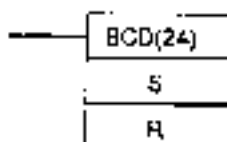
S: Source channel
R: Result channel

For BCD, enter
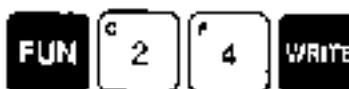
then the source and destination channel numbers.
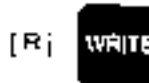
[S] WRITE

[R] WRITE

Example Circuit:
Ladder Diagram and
Mnemonic Code
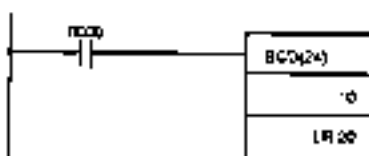
Address	Instruction	Data	
0000	LD		0000
0001	BCD(24)		—
			10
		LR	20

In the example above, when input 0000 turns ON, the binary contents of channel 10 are converted into BCD format and are then output to LR 20.

Data Areas

S	R
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, DM, *DM

Flags

ER ————| Result channel overflow (i.e. contents of R > 9999).
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ————| ON when the contents of the result channel are 0000.

Note: When the contents of the source channel exceed "270F", the converted result exceeds "9999", and the instruction is not executed. When the instruction is not executed, R remains unchanged.

4-7-3
4-to-16 Decoder - MLPX[76]

MLPX converts up to four, 4-bit hexadecimal values in the source channel into decimal numbers from 0 to 15 and then in the destination channel(s) turns ON the bit(s) that corresponds to the converted values. If more than one source digit is specified, then a single bit will be set to 1 in each of the corresponding destination channels.

Ladder Symbol and Key Sequence

```
┌──────────────┐
│   MLPX(76)    │
├──────────────┤
│      S        │
├──────────────┤
│      Di       │
├──────────────┤
│      RB       │
└──────────────┘
```
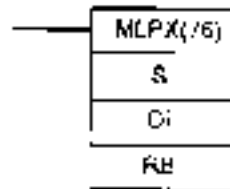
S: Source channel
Di: Digit designator
RB: Result beginning channel

For MLPX, enter

[FUN] [7] [6] [WRITE]
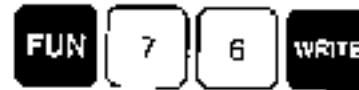
then the source channel, digit designator and result channel.
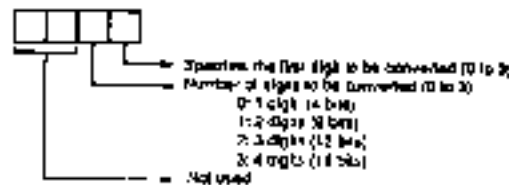
[S] [WRITE]

[Di] [WRITE]

[RB] [WRITE]

Digit Designator

Specifies the first digit to be converted (0 to 3)
Number of digits to be converted (0 to 3)
 0: 1 digit (4 bits)
 1: 2 digits (8 bits)
 2: 3 digits (12 bits)
 3: 4 digits (16 bits)
Not used

Example Circuit: Ladder Diagram and Mnemonic Code
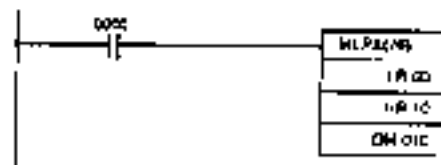
Address	Instruction	Data	
0000	LD		0000
0001	MLPX(76)		—
		LR	00
		HR	*0
		DM	D10

Decoding Multiple Digits

When decoding multiple digits, the first digit to be converted, as specified by Di, will be the first digit in the final output. For example, if the number of digits specified is "3", and the first digit is "2", the data are transferred from digits 2, 3, 0, and then 1.

DI : 0010   Di : 0030

Di : 0031   Di : 0023

Data Areas

S	Di
IR, SR, HR, LR, TC, DM, *DM,	IR, HR, LR, TC, DM, *DM, #

RD
IR, HR, LR, DM, *DM,
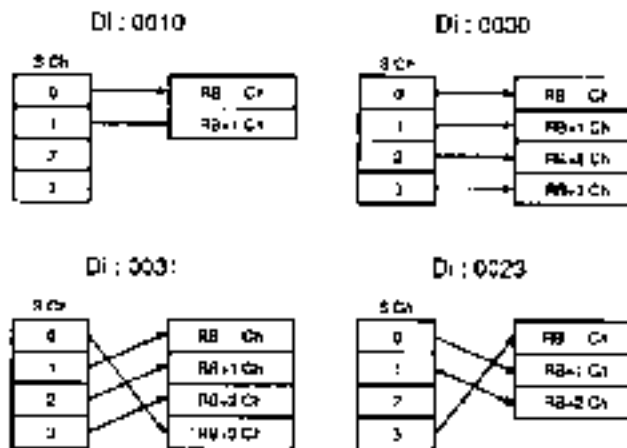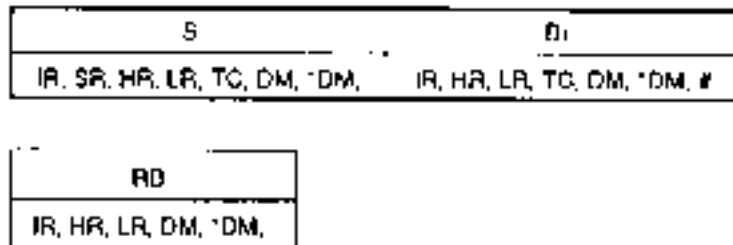
Note: Be sure that (RB + n), the specified destination plus the number of digits you are decoding, does not exceed the data area

Flags

ER ——— Incorrect digit designator, or RB+1 to RB+3 exceed the data area.

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

4-7-4
16-to-4 Encoder - DMPX(77)

DMPX determines the position of the highest ON bit in the specified source channel, encodes it into single-digit binary data, then transfers the result to the specified digit in the destination channel. Up to four digits (from four consecutive source channels) may be encoded and the digits written to the result channel.

Ladder Symbol and Key Sequence

```
┌─────────────┐
│  DMPX(77)   │
├─────────────┤
│     SB      │
├─────────────┤
│     R       │
├─────────────┤
│     Di      │
└─────────────┘
```
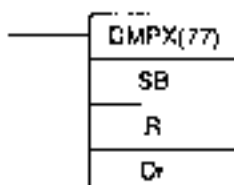
SB: Source beginning channel
R: Result channel
Di: Digit designator

For DMPX, enter

FUN **7** **7** **WRITE**

then the source, destination and digit designator data.

[SB] **WRITE**

[R] **WRITE**

[DI] **WRITE**

Digit Designator

Specifies the first digit to be converted (0 to 3)
Number of digits to be converted (0 to 3)
0: 1 digit (4 max)
1: 2 digits (8 max)
2: 3 digits (12 bits)
3: 4 digits (16 bits)
Not used
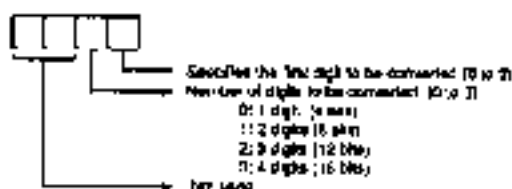
Example Circuit: Ladder Diagram and Mnemonic Code
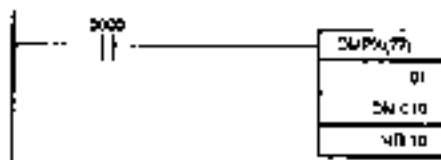
Address	Instruction	Data	
0000	LD		0000
0001	DMPX(77)		—
			01
		DM	010
		HR	10

Indicates the digit (1) in the destination channel is the first to be converted. In this example, the highest bit set to 1 is bit 09, therefore digit in the destination channel is encoded to 9. Note that the other digits of the result channel are not changed.

Encodes one digit only.
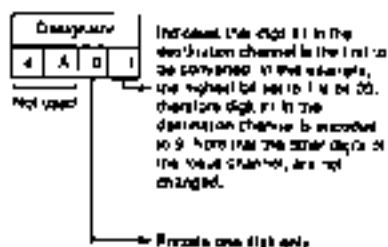
Encoding Multiple Digits

When encoding multiple digits, the first channel to be converted, as specified by Di, will be the first digit in the final output. For example, if the number of digits specified is "3", and the first digit is "2", the data are transferred to digits 2, 3, 0, and then 1.
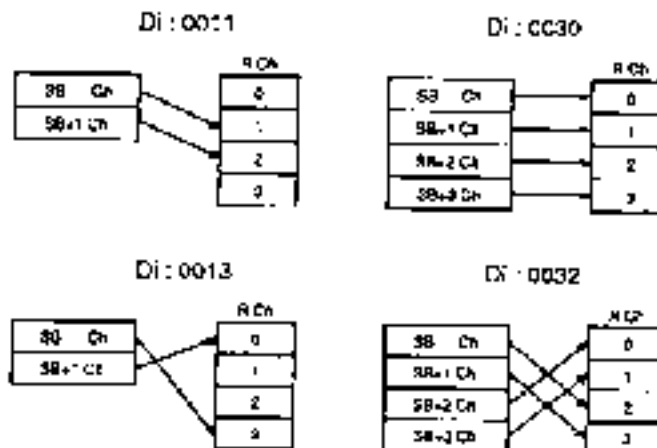
Data Areas

S	R
IR, SR, HR, LR, TC, DM, *DM,	IR, HR, LR, TC, DM, *DM,

DI
IR, HR, LR, DM, *DM, #

Note: Be sure that SB + n, the specified source channel plus the number of values to be encoded, does not exceed the data area.

Flags

ER ─── Incorrect digit designator data, or SB+1 to SB+3 cross over a data area boundary.

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

Contents of the source input channels (SB, SB+1, SB+2, and SB+3) are 0000.

4-7-5
7-Segment Decoder -
SDEC(78)

SDEC converts 4 bits of 16-bit data into 8-bit data for 7-segment display.

Ladder Symbol and Key Sequence

```
┌─────────────┐
│  SDEC(78)   │
├─────────────┤
│      S      │
├─────────────┤
│     Di      │
├─────────────┤
│     DB      │
└─────────────┘
```
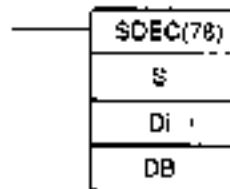
S: Source channel
Di: Digit designator
DB: Destination beginning channel

For SDEC, enter
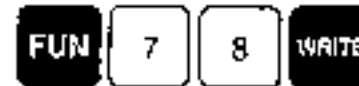
[FUN] [7] [8] [WRITE]

Then the source, destination and digit-designator data.

[S] [WRITE]

[Di] [WRITE]

[DB] [WRITE]

Digit Designator

Specifies the first digit to be converted (0 to 3)
Number of digits to be converted: (0 to 3)
  0: 1 digit (4 bits)
  1: 2 digits (8 bits)
  2: 3 digits (12 bits)
  3: 4 digits (16 bits)
Destination designator
  "0": Lower 8 bits
  "1": Higher 8 bits
Not used

Example Circuit: Ladder Diagram and Mnemonic Code

```
  0000
──┤ ├──────────────┐
                    │  SDEC(78)
                    │  DM 010
                    │  HR 05
                    │  20
```
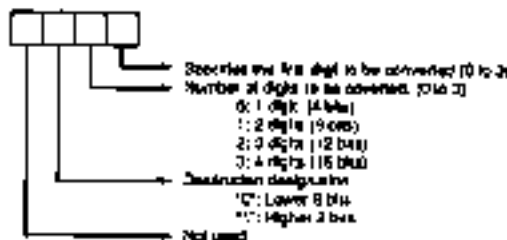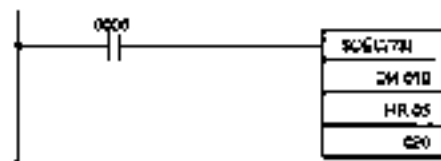
Address	Instruction	Data	
0000	LD		0000
0001	SDEC(78)		—
		DM	010
		HR	05
			20

Example
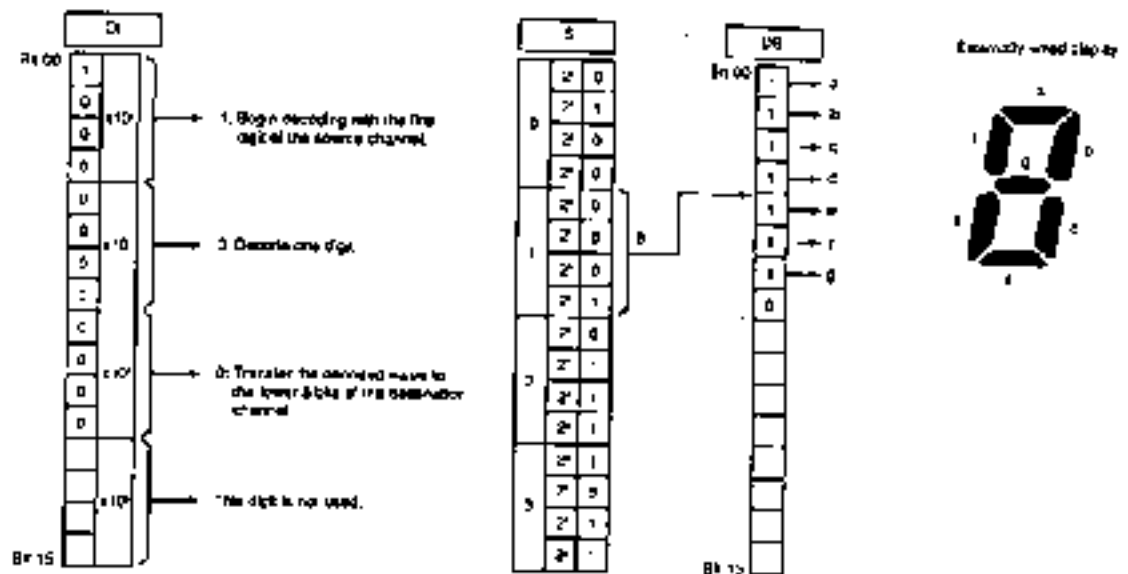
Encoding Multiple Digits

When encoding multiple digits, the output area available is three channels long, ending at channel DB+2. The encoded data is written sequentially into the channels beginning with the upper or lower half of DB depending on the designated information.

Di : 0011

Data Areas

S	Di
IR, SR, HR, LR, TC, DM, *DM,	IR, HR, LR, TC, DM, *DM, #

DB
IR, HR, LR, DM, *DM
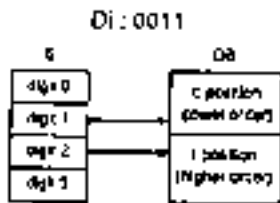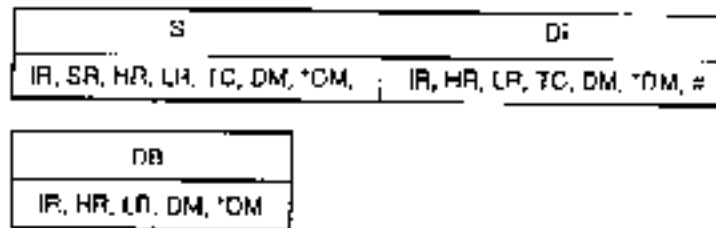
Note: Be sure that (DB + n), the specified destination channel plus the number of digits you are decoding, does not cross a data area boundary.

Flags

ER ——— Incorrect digit designator, or DB+1 to DB+3 exceed the data area

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

157

4-8
BCD Calculation
Instructions

The BCD calculation instructions - INC, DEC, ADD, SUB, MUL, DIV, FDIV, and ROOT - all perform arithmetic operations on BCD data.

For INC and DEC the input and output channels are the same. That is, the contents of their input channels are overwritten with the instruction results.

STC and CLC, instructions which set and clear the carry flag, are included in this group because most of the BCD operations make use of the carry flag in their results. Binary arithmetic and shift operations also use the carry flag.

Note: The addition and subtraction instructions use CY in the calculation as well as in the result. Be sure to clear CY if it is not required in the calculation.

Each BCD calculation instruction is programmed with a function code. To input these instructions through the Programming Console, you must press the FUN key followed by the appropriate function code.

4-8-1
Increment - INC(38)

INC increments 4-digit BCD data by one, without carry (CY).

Ladder Symbol and Key Sequence

```
    ┌─────────┐
────┤ INC(38) │
    ├─────────┤
    │   Ch    │
    └─────────┘
```
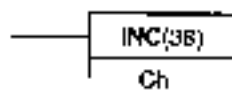
Ch.: Data channel

For INC, enter

FUN [3] [8] **WRITE**

WRITE | Ch | **WRITE**

Example Circuit: Ladder Diagram and Mnemonic Code

```
   0000
────┤├──────────────────┐
                   ┌─────────┐
                   │ INC(38) │
                   │  DM 010 │
                   └─────────┘
```
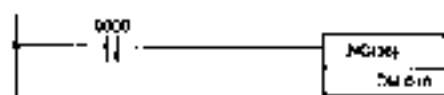
Address	Instruction	Data	
0000	LD		0000
0001	INC(38)		—
		DM	010

INC Operation

When input 0000 turns ON, 1 is added to the contents of DM 010. The incremented result is then output to DM 010. Until input 0000 goes OFF, the contents of DM 010 will be incremented on every scan.

Data Areas

```
IR, HR, LR, DM, *DM
```

Flags

ER ──────── The data to be incremented is not BCD.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ──────── ON when the incremented result is 0.

4-8-2
Decrement - DEC(39)

DEC decrements 4-digit BCD data by 1, without carry. DEC works the same way as INC except that it decrements the value instead of incrementing it

Ladder Symbol and
Key Sequence

```
     ┌─────────────┐
─────┤   DEC(39)   │
     ├─────────────┤
     │     Ch      │
     └─────────────┘
```
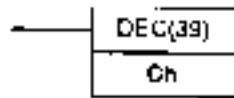
Ch: Data channel

For DEC, enter
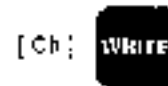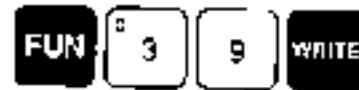
[FUN] [3] [9] [WRITE]

[Ch ; [WRITE]

Example Circuit:
Ladder Diagram and
Mnemonic Code

```
  0000
───┤ ├─────────────────┐
                  ┌──────────┐
                  │ DEC(39)  │
                  ├──────────┤
                  │  DM 010  │
                  └──────────┘
```
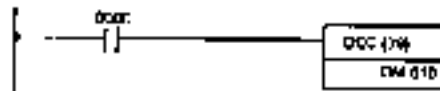
Address	Instruction	Data	
0000	LD		0000
0001	DEC(39)		—
		DM	010

DEC Operation

When input 0000 turns ON, 1 is subtracted from the contents of DM 010. The decremented result is then output to DM 010. Until input 0000 goes OFF, the contents of DM 010 will be decremented on every scan.

Data Areas

```
IR, HR, LR, DM, *DM
```

Flags

ER ──────┤ The data to be decremented is not BCD.
   │ Indirectly addressed DM channel is non-existent.
          │ (DM data is not BCD, or the DM area boundary has been
          │ exceeded.)
EQ ──────┤ ON when the decremented result is 0.

4-8-3
Set Carry - STC(40)
Clear Carry - CLC(41)

STC(40) sets the carry flag, CY (i.e., assigns a "1" to CY).
CLC(41) clears the carry flag, CY (i.e., assigns a "0" to CY).

Ladder Symbols and
Key Sequence

```
——| STC(40) |
```

```
——| CLC(41) |
```

For STC, enter

FUN [4] [0] **WRITE**

For CLC, enter

FUN [4] [1] **WRITE**

The carry flag is affected by the following instructions:

Instruction	FUN	Meaning of Carry Flag Value	
		1	0
ADD	30	There was an overflow in the result of an addition operation.	No overflow occurred.
SUB	31	Subtraction result < 0 (Output as 10's complement)	Subtraction result > 0
ASL	25	Before shifting, bit 15 was ON.	Before shifting, bit 15 was OFF.
ROL	27		
ASR	26	Before shifting, bit 00 was ON.	Before shifting, bit 00 was OFF
ROR	28		
SFT	84	If right-shifting: bit 00 was ON	Bit 00 was OFF
		If left-shifting: bit 15 was ON	Bit 15 was OFF.
STC	40	STC execution	——
CLC	41	——	CLC execution
END	01	——	END execution.

Note: CY is included in the calculations for addition, subtraction and shift instructions. If CY is not required, you should execute CLC before any addition, subtraction, or shift operation. Simply connect CLC to the same input as the instruction it precedes.

4-8-4
BCD Add - ADD(30)

ADD totals two 4-digit BCD values and outputs the result to the specified channel.

Ladder Symbol and Key Sequence

```
--[ ADD(30) ]
   [   Au   ]
   [   Ad   ]
   [   R    ]
```
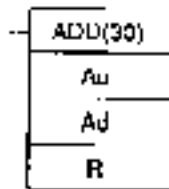
Au: Augend
Ad: Addend
R: Result channel

For ADD, enter
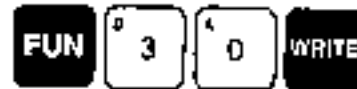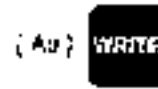
[FUN] [3] [0] [WRITE]

then the two operands and the output channel.

(Au) [WRITE]

[Ad] [WRITE]

[R) [WRITE]

Example Circuit: Ladder Diagram and Mnemonic Code
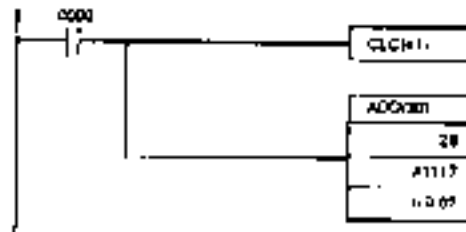
Address	Instruction	Data	
0000	LD		0000
0001	CLC(41)		—
0002	ADD(30)		—
			28
		#	1117
		HR	07

In the example above, when input 0000 turns ON, the BCD data of channel 28 (i.e, 2800 - 2815) and the constant value 1117 are added with carry, and the 4-digit result is output to HR 07. Add will set the carry flag if there is an overflow in the resulting sum.

153 + 1117 = 1,270
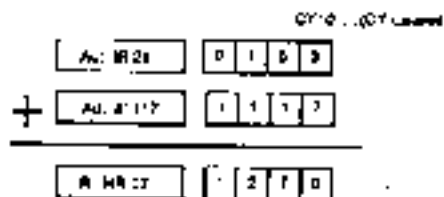
CY: 0.... [The resulting sum did not overflow 4 digits]

Data Areas

Au and Ad	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER —————| One or both of the channels to be added are not BCD.
         | Indirectly addressed DM channel is non-existent.
         | (DM data is not BCD, or the DM area boundary has been
 | exceeded.)

CY —————| Indicates a carry in the result.

EQ —————| ON when the resulting sum is 0.

4-8-5
BCD Subtract - SUB(31)

SUB subtracts one 4-digit BCD value from another, with carry, and outputs the result to the specified result channel

Ladder Symbol and Key Sequence

```
        SUB(31)
        Mi
        Su
        R
```
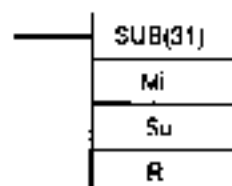
Mi: Minuend
Su: Subtrahend
R: Result channel

For SUB, enter

[FUN] [3] [1] [WRITE]
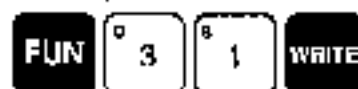
then the two values to be added and the output channel

[M] [WRITE]

[Su] [WRITE]

[R] [WRITE]

SUB Input Data and Output Result

Data	SUB Result	CY	EQ
Mi>Su	R = Mi - Su	0	0
Mi=Su	R = 0	0	1
Mi<Su	R = Mi + (10000-Su)	1	0

Example Circuit: Ladder Diagram and Mnemonic Code
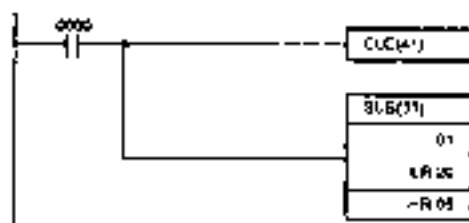
Address	Instruction	Data	
0000	LD		0000
0001	CLC(41)		—
0002	SUB(31)		—
			01
		LR	20
		HR	05

In the example above, when input 0000 turns ON, the 4-digit BCD data in LR 20 (i.e., LR 2000 to LR 2015) is subtracted from the 4-digit BCD data in channel 01, with carry, and the 4-digit result is output to HR 05.

6,106 - 5,811 = 295

| Mi: #R 001 | 6 | 1 | 0 | 6 |

| Su: LR 20 | 5 | 8 | 1 | 1 |

- CY

| R: HR 05 | 0 | 2 | 9 | 5 |

Data Areas

6-8-A
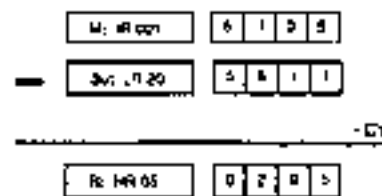
Mi and Su	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM, *

**Flags**

ER — — | The contents of the Mi and/or the Su channels are not BCD.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

CY— ——| ON when the result is negative (i.e., Mi < Su).

EQ ——| ON when the result is 0

Note: Be sure to clear the carry flag with CLC If it is not required before executing SUB, and check the state of CY after doing a subtraction with SUB If CY is turned ON as a result of SUB, the negative data is output as its 10's complement To convert the output result to its absolute value, subtract the value in the result channel from the constant 0.

165

4-8-6

BCD Multiply - MUL(32)

MUL multiplies two 4-digit BCD values together and outputs the result to the specified channels. Two channels are required for the output result.

Ladder Symbol and Key Sequence
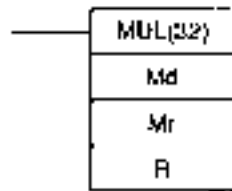
Md: Multiplicand
Mr: Multiplier
R: Result beginning channel number

For MUL, enter
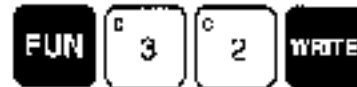
then the operands and the output beginning channel no.
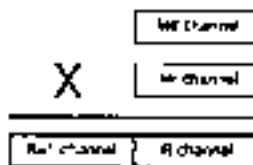
| Md) WRITE

[Mr] WRITE

[R] WRITE

Example Circuit: Ladder Diagram and Mnemonic Code

Address	Instruction	Data	
0000	LD		0000
0001	MUL (32)		—
			37
		DM	005
		HR	07

In the example above, when input 0000 turns ON, the BCD data in channel 37 is multiplied with the BCD data in DM 005, and the 8-digit result is output to HR 07 and HR 08.

Example

$3{,}356 \times 25 = 83{,}900$

	Md : R002			
	3	3	5	6

	Mr : DM 205			
X	0	0	2	5

R+1 : HR08				R : HR07			
0	0	0	8	3	9	0	0

Data Areas

Md and Mr	R
IR, SR, HR, LR, TC, DM, *DM , #	IR, HR, LR, DM, *DM

Flags

ER ———— The contents of the Md and/or the Mr channels are not BCD.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ———— ON when the result is 0.

4-8-7
BCD Divide - DIV(33)

DIV divides a 4-digit BCD number by another and outputs the result to the specified channel. Two channels, one for the quotient and one for the remainder, are required for the result.

Ladder Symbol and Key Sequence
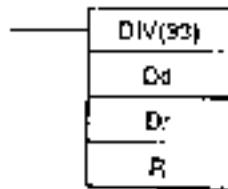
DIV(33)
Dd
Dr
R

Dd: Dividend
Dr: Divisor
R: Result channel

For DIV, enter
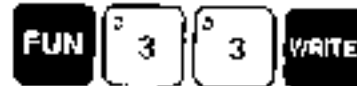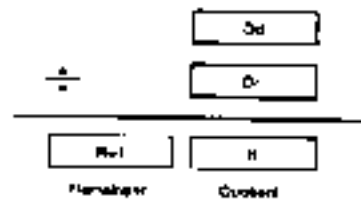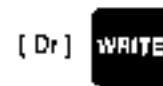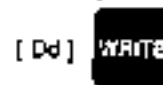
[FUN] [3] [3] [WRITE]

then the operands and the output beginning channel,

[Dd] [WRITE]

[Dr] [WRITE]

[R] [WRITE]

Example Circuit: Ladder Diagram and Mnemonic Code
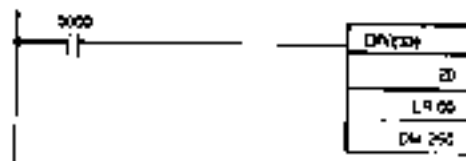
Address	Instruction	Data	
0000	LD		0000
0001	DIV(33)		—
			20
		LR	00
		DM	250

Example

3,452 ÷ 3 = 1,150   Remainder: 2

168
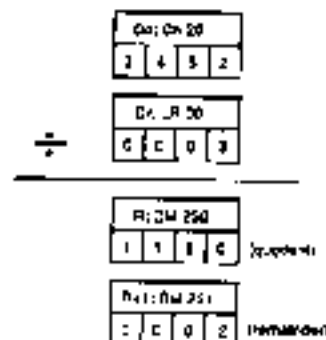
Data Areas

Dd and Dr	R
IR, SR, HR, LR, TC, DM, *DM , #	IR, HR, LR, DM, *DM

Flags

ER ————— The contents of the Dd and/or the Dr channels are not BCD.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been
exceeded.)

EQ ————— ON when the result is 0.

4-8-8
Floating Point Divide - FDIV(79)

FDIV divides a floating point value by another and outputs a floating point result. The dividend, divisor and resulting quotient each require two channels (8 digits).

To represent the floating point values, the rightmost seven digits are used for the mantissa and the leftmost digit is used for the exponent

Ladder Symbol and Key Sequence

```
        FDIV(79)
          Dd
          Dr
          R
```
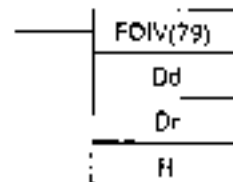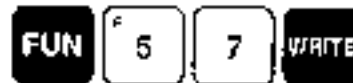
Dd: Dividend beginning channel no.
Dr: Divisor beginning channel no.
R: Result beginning channel no.

For FDIV, enter

[FUN] [5] [7] [WRITE]

then the operands and the output beginning channel no.

[Dd] [WRITE]

[Dr] [WRITE]

[R] [WRITE]

Exponent Digit
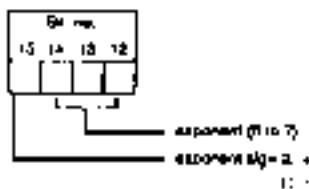
The valid range for both the input and result data is 0.0000001×10^{-7} to $0.9999999 \times 10^{+7}$. The resulting quotient is truncated to 7 digits.

Example Circuit:
Ladder Diagram and
Mnemonic Code
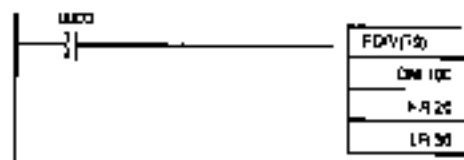
Address	Instruction	Data	
0000	LD		0000
0001	FDIV(79)		—
		DM	100
		HR	20
		LR	30

Example

$0.567 \times 10^{-2} \div 0.1234567 \times 10^{-3} = 0.4592703 \times 10^{2}$

0.5670000×10^{-2}

$\div$

0.1234567×10^{-3}

0.4592703×10^{2}

exponent sign: A : (1010) 10^{-2}
B : (1011) 10^{-3}
2 : (0010) 10^{2}
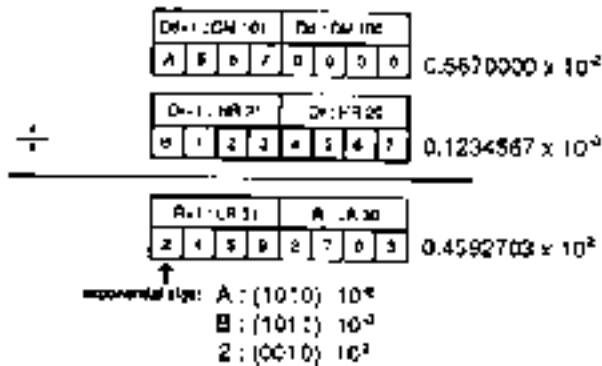
Data Areas

Dd and Dr	R
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, DM, *DM

Flags

ER ——— The divisor, Dr, is 0.
Contents of Dd, Dd +1, Dr, or Dr + 1 channels are not BCD.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ——— ON when the resulting quotient (channels R and R+1) is 0.

4-8-9

Square Root - ROOT(72)

ROOT computes the square root of an 8-digit BCD value and outputs the truncated 4-digit integer result to the specified channel.

Ladder Symbol and Key Sequence

```
┌─── ROOT(72)
│       Sq
│       R
```
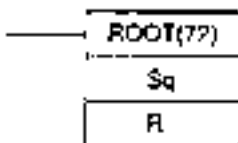
Sq: Source data beginning channel no.
R: Result channel

For ROOT, enter:

[FUN] [7] [2] [WRITE]
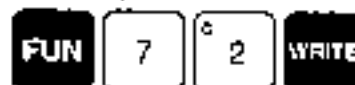
then the input beginning channel number and the output channel number
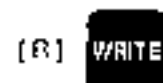
[Sq] [WRITE]

[R] [WRITE]

Example Circuit: Ladder Diagram and Mnemonic Code

```
0000
─┤├──────────── ROOT(72)
                   DM 000
                   01
```
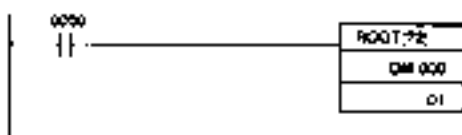
Address	Instruction	Data	
0000	LD		0000
0001	ROOT(72)		—
		DM	000
			01

Example

$$\sqrt{63,250,561} = 7,953 \quad (\text{approximately})$$
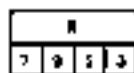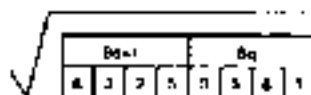
7953 0221...
Decimal part is truncated

Data Areas

Sq	R
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, DM, *DM

Flags

ER ────── Indirectly addressed DM channel is non-existent.
           (DM data is not BCD, or the DM area boundary has been exceeded.)
           The source data is not BCD.

EQ ────── ON when the result is 0.

4-9
Logic Instructions

The logic instructions - COM, ANDW, ORW, XORW, and XNRW   perform logical operations on channel data

Each logic instruction is programmed with a function code. To input these instructions through the Programming Console, press the FUN key followed by the appropriate function code.

4-9-1
Complement - COM(29)

COM inverts single channel data. That is, COM clears all ON bits and sets all OFF bits in the channel specified.

Ladder Symbol and Key Sequence

```
COM(29);
```
```
Ch
```
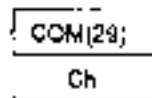
Ch : Channel whose contents are to be inverted.

For COM, enter

FUN **2** **9** **WRITE**
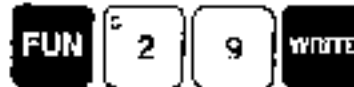
then the channel whose contents are to be inverted.

[R] **WRITE**

Example Circuit: Ladder Diagram and Mnemonic Code

Address	Instruction	Data
0000	LD	0000
0001	COM(29)	—
		HR 00
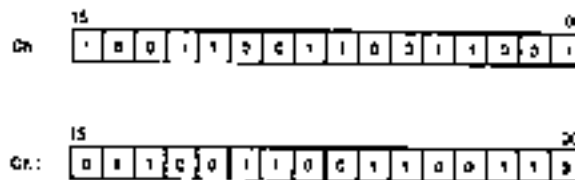
Example

Data Areas
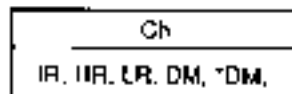
Ch
IR, HR, LR, DM, *DM,

Flags

ER ————| Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ————| ON when all bits of the result are 0

4-9-2
Logical AND - ANDW(34)

ANDW logically AND's two 16-bit data values and outputs the result to the specified channel.

Ladder Symbol and Key Sequence

```
        ANDW(34)
           I1
           I2
           R
```
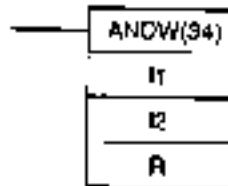
I1 : Input 1
I2: Input 2
R: Result channel no.

For AND, enter

[FUN] [3] [4] [WRITE]

then the two operands and the output channel.

[I1] [WRITE]

[I2] [WRITE]

[R] [WRITE]
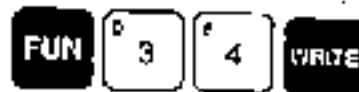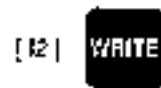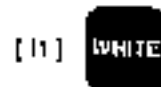
ANDW sets the corresponding bit in the output channel only when the corresponding bits in both inputs are 1.

I1	I2		R
1	1	→	1
1	0	→	0
0	1	→	0
0	0	→	0

**Example Circuit;
Ladder Diagram and
Mnemonic Code**
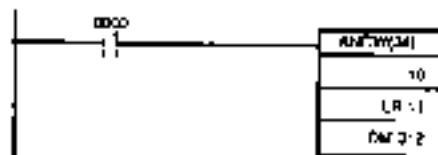
Address	Instruction	Data	
0000	LD		0000
0001	ANDW(34)		—
			10
		LR	1
		DM	012

Example

I Ch: `1 2 0 1 1 0 0 1 1 0 0 1 1 0 0 1`

R Ch: `0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1`

R Ch: `0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1`

Data Areas

I1 and I2	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER   Indirectly addressed DM channel is non-existent.
           (DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ——— ON when all bits of the result are 0.

4-9-3
Logical OR - ORW(35)

ORW logically OR's two 16 bit values and outputs the result to the specified channel.

Ladder Symbol and Key Sequence

ORW(35)
I1
I2
R
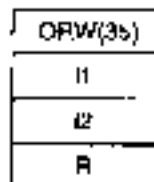
I1: Input 1
I2: Input 2
R: Result channel

For ORW, enter

FUN   3   5   WRITE

then the two operands and the output channel

[I1] WRITE

[I2] WRITE

[R] WRITE
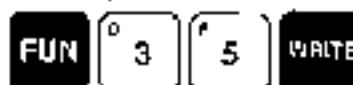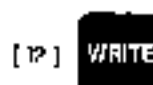
A bit in the output channel will be 1 if one or both of the corresponding bits in the input data are 1

I1	I2		R
1	1	$\rightarrow$	1
1	0	$\rightarrow$	1
0	1	$\rightarrow$	1
0	0	$\rightarrow$	0

Example Circuit: Ladder Diagram and Mnemonic Code

ORW(35)
01
LR 1
Dm 012

Address	Instruction	Data	
0000	LD		0000
0001	ORW(35)		—
			01
		LR	1
		DM	012

177

Example

I₁ Ch:

16															00
1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

I₂ Ch:

15															00
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

R Ch:

15															00
1	1	0	1	1	0	1	1	1	0	1	1	1	1	0	1

Data Areas

I1 and I2	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER ———— Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ———— ON when all bits of the result are 0.

4-9-4
Exclusive OR -
XORW(36)

XORW exclusively OR's two 16-bit data values and outputs the result to the specified channel.

Ladder Symbol and Key Sequence

```
┌─ XORW(36) ─┐
│            │
│     I1     │
│            │
│     I2     │
│            │
│     R      │
└────────────┘
```
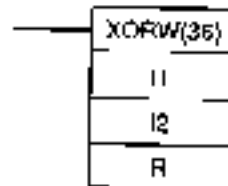
I1: Input 1
I2: Input 2
R: Result channel

For XORW, enter
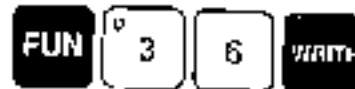
[FUN] [3] [6] [WRITE]

then the two operands and the output channel.

[I1] [WRITE]

[I2] [WRITE]

[R] [WRITE]

A bit in the output channel will be 1 only when the corresponding bits in the input channels differ.

I1	I2		R
1	1	→	0
0	1	→	1
1	0	→	1
0	0	→	0

Example Circuit: Ladder Diagram and Mnemonic Code

```
 0000
──┤├──────────── ┌─ XORW(36) ─┐
                  │     10     │
                  │    LR 11   │
                  │   DM 012   │
```
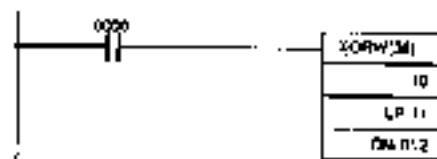
Address	Instruction	Data	
0000	LD		0000
0001	XORW(36)		—
			10
		LR	11
		DM	012

Example

I Ch.	15 ... 00
I2 Ch.	15 ... 00
R Ch.	15 ... 00

Data Areas

I1 and I2	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER ——— Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ——— ON when all bits of the result are 0

4-9-5
Exclusive NOR - XNRW(37)   XNRW exclusively NOR's two 16-bit data values and outputs the result to the specified channel.

Ladder Symbol and Key Sequence

XNRW(37)
I1
I2
R
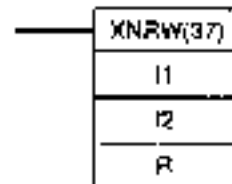
I1: Input 1
I2: Input 2
R: Result channel

For XNRW, enter:
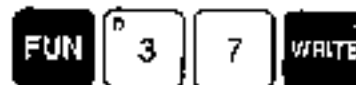
[FUN] [3] [7] [WRITE]

then the two operands and the output channel.

[I1] [WRITE]

[I2] [WRITE]

[R] [WRITE]

A bit in the output channel will be 1 only when the corresponding bits in the input channels are the same.

I1	I2		R
1	1	→	1
0	1	→	0
1	0	→	0
0	0	→	1

Example Circuit: Ladder Diagram and Mnemonic Code

```
  0000
───┤├──────────────┐
                 │ XNRW(37) │
                 │    10    │
                 │   LR 11  │
                 │  DM 012  │
```
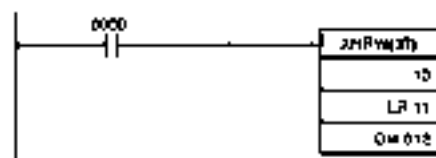
Address	Instruction	Data	
0000	LD		000
0001	XNRW(37)		—
			10
		LR	11
		DM	012

Example

Data Areas

I1 and I2	R
IR, SR, HR, LR, TC, DM, *DM, #	IR, HR, LR, DM, *DM

Flags

ER ——— Indirectly addressed DM channel is non existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

EQ ——— ON when all bits of the result are 0.

4-10
Special Instructions

This section details the operation of the four special purpose instructions: FAL, FALS, WDT, and IOPF.

4-10-1

Failure Alarm - FAL(06)
Severe Failure Alarm -
FALS(07)

FAL and FALS are diagnostic instructions which output an 8 bit BCD error code to the FAL output area. The error codes, FAL numbers 00 to 99, are arbitrarily assigned. FAL 00 sets the FAL area to 00 and is used for clearing other FAL codes. Note that there are also FAL and FALS calls made by the system. The error codes output by these calls are all greater than 99.

Ladder Symbols and Key Sequence

—[FAL(06) N1]

N1: FAL No.

—[FALS(07) N2]

N2: FAL No.

For FAL, enter

[FUN] [0] [6] [N1] [WRITE]

For FALS, enter

[FUN] [0] [7] [N2] [WRITE]
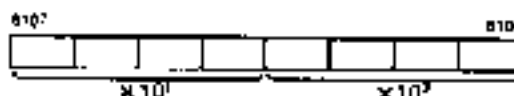
FAL Output Areas

FAL lights the warning indicator lamp on the front panel of the CPU but program execution continues. FALS, however, lights the error indicator lamp and stops the CPU, suspending program execution.

Resetting FAL Output

A maximum of three FAL error codes can be retained. To reset the FAL area execute FAL 00. Each time FAL 00 is executed another FAL error retained in memory is output to the FAL area.

Note that other alarms and failure indications, such as battery error and I/O errors, are also output to the FAL area.

Resetting FALS output

To reset the FALS output, remove the cause of the FALS error and then perform Error/Message Read (See 2-3-9) through the Programming Console.

**Example Circuit:
Ladder Diagram and
Mnemonic Code**
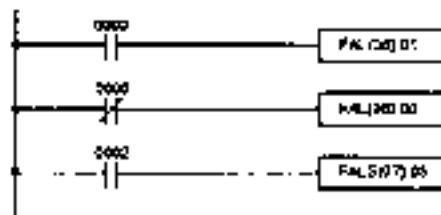
Address	Instruction	Data	
0000	LD		0000
0001	FAL(06)		01
0002	LD	NOT	0000
0003	FAL(06)		00
0004	LD		0002
0005	FALS(07)		03

In the above example, when input 0000 turns ON because of some error
condition, a FAL 01 code is output to the FAL area and the warning lamp on
the front panel of the CPU lights. When input 0000 turns OFF, FAL 00 is
executed and the FAL area is cleared.

Data Areas

N1	N2
00 to 99	01 to 99

4-10-2
Set Watchdog Timer -
WDT(94)

WDT changes the value of the watchdog timer (automatically set by the system to 130 ms). When the scan time exceeds the value of the watchdog timer, the system generates a FALS 9F and the CPU stops.

The value of the watchdog timer can be increased or decreased in units of 100ms. The number of 100-ms units is specified by T. T must be in the range $0 \leq T \leq 53$.

T	Set Value of Watchdog Timer
00	130 ms
01	130 to 230 ms
02	230 to 330 ms
03	330 to 430 ms
:	:
T	$130 + 100(T-1)$ to $130 + 100T$

Warning

If the scan time exceeds 6,500 ms, a FALS 9F will be generated and the system will halt.

**Ladder Symbol and
Key Sequence**
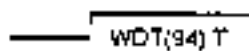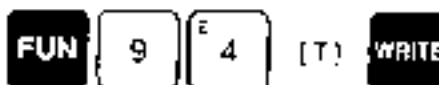
———— WDT(94) T ————

For WDT, enter

FUN **9** **4** [T] **WRITE**

**Example Circuit:
Ladder Diagram and
Mnemonic Code**
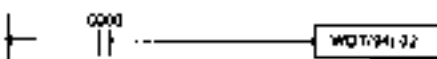
├──┤├──────────────────[WDT(94) 02]

Address	Instruction	Data
0000	LD	0000
0001	WDT(94)	02

In the above example, when input 0000 turns ON, WDT sets the value of the watchdog timer to the range 230 to 330 ms.

Special Instructions   Section 4-10

4-10-3
I/O Refresh - IORF(97)

IORF refreshes the specified I/O channels. Channels are refreshed as whole units.

I/O channels are normally refreshed once every scan (an END refresh), but with IORF a selected group of channels can be refreshed at any time during a scan.

Ladder Symbol and Key Sequence

```
        ┌──────────┐
────────│ IORF(97) │
        │    B     │
  - -   │          │
        │    E     │
        └──────────┘
```
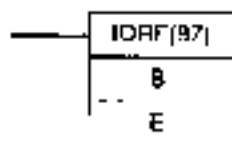
B: Beginning channel
E: End channel

For IORF, enter

[FUN] [9] [7] [WRITE]
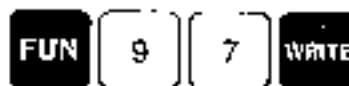
and then the beginning and end channel numbers.

[B] [WRITE]

[E] [WRITE]

B and E must be channels in the I/O section of the IR area, and E must be greater than B.

Example Circuit: Ladder Diagram and Mnemonic Code
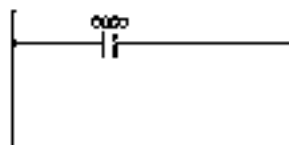
```
    0000
│────┤├────────────┐ IORF (97) │
                   │    10     │
                   │    14     │
```

Address	Instruction	Data
0000	LD	0000
0001	IORF(97)	—
		10
		14

Time Required to Execute IORF

T_{IORF} = 120 µs (fixed) + 15 µs × (no. of channels refreshed.)

IORF is available only for refreshing I/O Units that are directly connected to the CPU or to Expansion I/O Racks. IORF cannot be used to refresh Remote I/O Slave Units or Optical Transmitting I/O Units.

Data Areas

B and E
IR 00 to IR 31

4-11
Intelligent I/O
Instructions

The intelligent I/O instructions are used for input/output operations with Intelligent I/O Units, such as an ASCII Unit.

4-11-1
**Intelligent I/O Write -
WRIT(87)**

WRIT transfers channel data through a dedicated I/O channel and sequentially writes the data to the memory area of an Intelligent I/O Unit.

**Ladder Symbol and
Key Sequence**

```
 ┌─ WRIT(87)
─┤│
 │     N
 │     S
 └     D
```
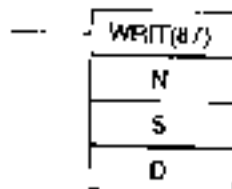
N: Number of channels to transfer
S: Source beginning channel
D: Destination I/O channel

For WRIT, enter
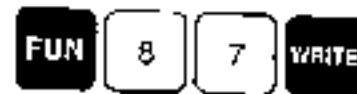
FUN **8** **7** **WRITE**

[N] **WRITE**

[S] **WRITE**

[D] **WRITE**

**Example Circuit:
Ladder Diagram and
Mnemonic Code**

```
 0000
─┤ ├──────────    WRIT(87)
                      01
                    DM 010
                      04
```

Address	Instruction	Data	
0000	LD		0000
0001	WRIT(87)		—
			01
		DM	010
			04
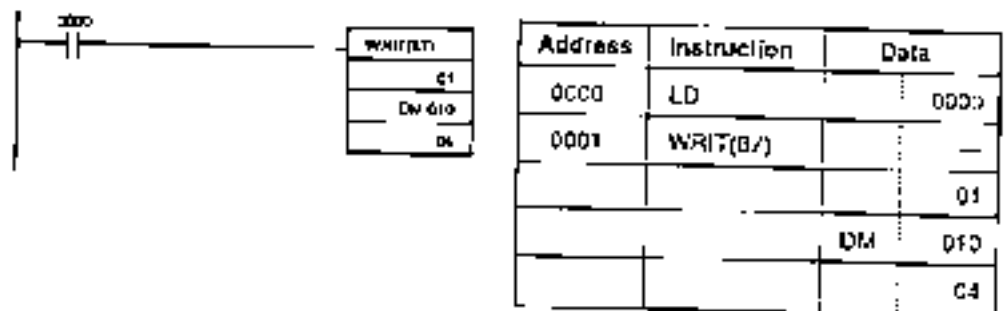
When input 0000 turns ON, the contents of channels DM 010 through 014 are transferred to channel 04 which is assigned to an Intelligent I/O Unit.

Example
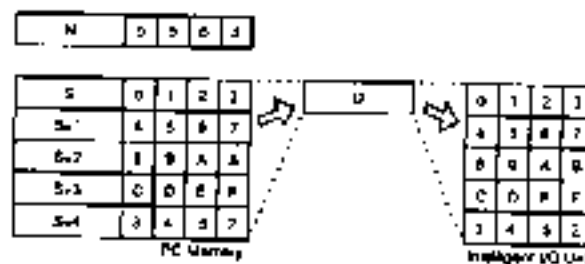
Application Example

Products are carried along a conveyor to a scale where they are weighed. When 100 products are weighed, the weight data is transferred to the Intelligent I/O Unit.
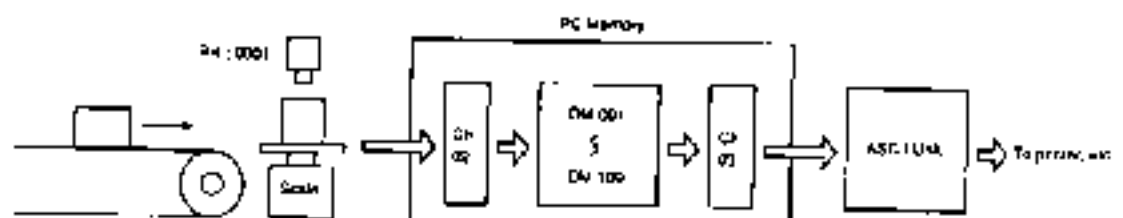
·Ladder Diagram and
 Mnemonic Code
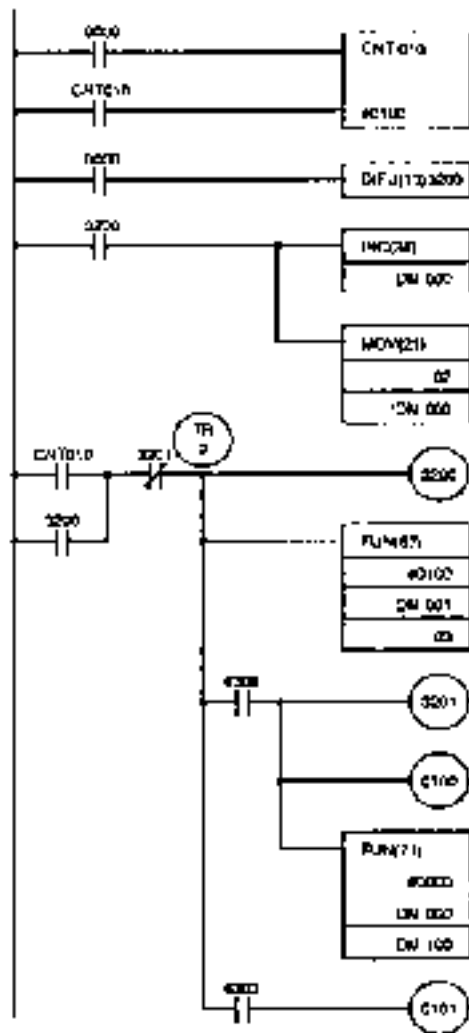
Address	Instruction	Data	
0000	LD		0000
0001	LD	CNT	010
0002	CNT		010
		#	0100
0003	LD		0000
0004	DIFU(13)		3200
0005	LD		3200
0006	IXC(38)		—
		DM	000
0007	MOV(21)		— —
			002
		*DM	000
0008	LD	CNT	010
0009	OR		3200
0010	AND.NOT		3201
0011	OUT	TR	0
0012	OUT		3200
0013	WRIT(87)		—
----- ----		#	0100
		DM	001
			003
0014	AND		6306
0015	OUT		3201
0016	OUT		0100
0017	BSET(71)		—
		#	0000
		DM	000
		DM	100
0018	LD	TR	0
0019	AND		6303
0020	OUT		0101

Data Areas

N	S
IR, HR, LR, TC, DM, *DM, #	IR, HR, LR, TC, DM, *DM

D
IR (I/O channel only)

Flags

ER ——— Destination is not an Intelligent I/O Unit channel.

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

The specified data range exceeds a data area boundary

EQ · OFF while WRIT is in progress; ON when WRIT completes.

Note: If an Intelligent I/O Unit is busy and unable to receive data, the writing will take place during the next scan. To make sure that WRIT execution has completed, check the EQ flag.

4-11-2
Intelligent I/O Read -
READ(88)

READ reads data from the memory area of an Intelligent I/O Unit and trans-
fers it through a dedicated I/O channel to the destination channels.

**Ladder Symbol and
Key Sequence**

```
┌─────────────┐
│  READ(88)   │
├─────────────┤
│      N      │
├─────────────┤
│      S      │
├─────────────┤
│      D      │
└─────────────┘
```
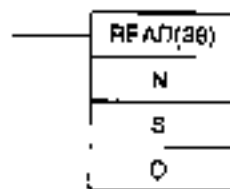
N: Number of channels to transfer
S: Source I/O channel
D: Destination beginning channel

For READ, enter
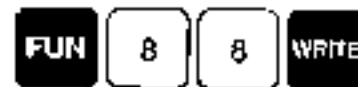
[FUN] [8] [8] [WRITE]

[N] [WRITE]

[S] [WRITE]

[D] [WRITE]

**Example Circuit:
Ladder Diagram and
Mnemonic Code**

Address	Instruction	Data	
0000	LD		0000
0001	READ(88)		—
			01
			03
		DM	010

When input 0000 turns ON, the contents of the memory area of the Intelligent
I/O Unit are sequentially read to I/O channel 03 and then transferred to
channels DM 010 through 014.

Example
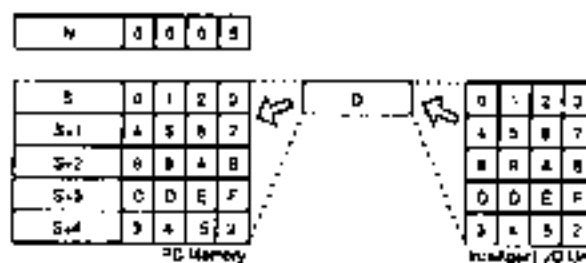
If the data cannot be sent or the Intelligent I/O Unit is busy, the reading will
take place during the next scan

To make sure that READ execution has completed, check the EQ flag

Data Areas

N	S
IR, HR, LR, TC, DM, *DM, #	IR (I/O channel only)

D
IR, HR, LR, TC, DM, *DM

Flags

ER ———— Source is not an Intelligent I/O Unit channel.

Indirectly addressed DM channel is non-existent.

(DM data is not BCD, or the DM area boundary has been exceeded.)

The specified data range exceeds a data area boundary.

EQ ———— OFF while READ is in progress; ON when READ completes.

4-12
SYSNET Instructions

The SYSNET instructions are used for communicating with devices linked to the PC through SYSNET.

4-12-1
Send - SEND(90)

SEND sends data to a device linked through SYSNET.

Ladder Symbol and Key Sequence

```
SEND(90)
   S
   D
   C
```
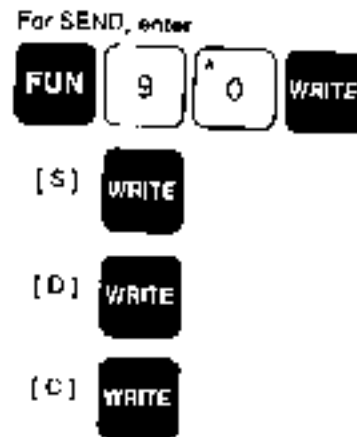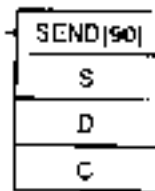
S: Source beginning channel
(on the sending PC)
D: Destination beginning channel
(on node number N)
C: Control data (3 channels)

For SEND, enter

[FUN] [9] [0] [WRITE]

[S] [WRITE]

[D] [WRITE]

[C] [WRITE]

Control Data

Channel	Bits 15 to 8	Bits 7 to 0
C	Number of channels (0 to 1000) (4-digit hexadecimal)	
C+1	Destination NSB/NSU (0,1)	Network number 0 to 255 (2-digit hexadecimal)
C+2	Destination Board Number (0,1,2)	Destination Node No. 0 to 127 (2 digit hexadecimal)

NSB: Network Service Board
NSU: Network Service Unit

NSB, NSU and the board number are all normally set to 0. Refer to the SYSNET Link System manual for details.

SEND takes the specified number of channels of data, starting with the source beginning channel S, sends them to node N, and writes the data to the destination channels beginning at channel D.

When the node number is set to 0, data is sent to all linked PC and personal computer nodes.

Example

195

Example Circuit:
Ladder Diagram and
Mnemonic Code

	SEND(90)
	01
	LR 20
	DM 010

Address	Instruction	Data	
0000	LD		0000
0001	SEND(90)		—
			01
		LR	20
		DM	010
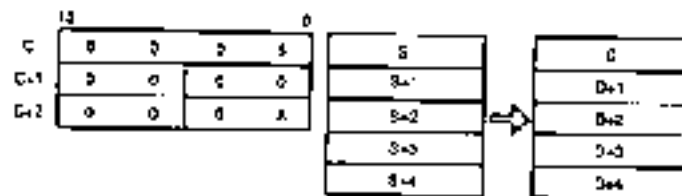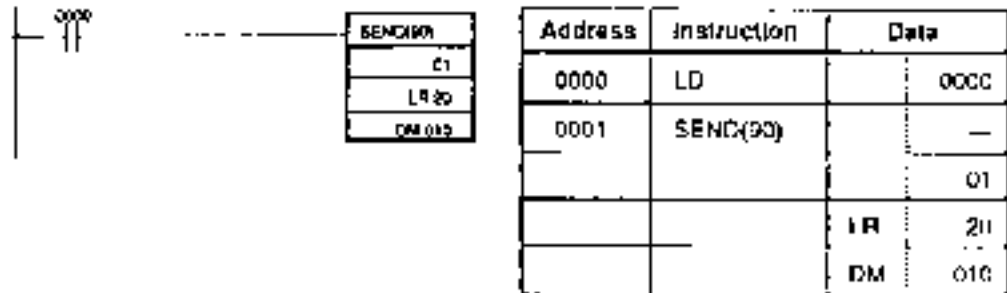
When input 0000 turns ON, data from channels 01 to 05 is sent and written to channels LR 20 through 24 of Node No. 10.

The data sent is that which is present when SEND is executed.

Use the SYSNET Error and Run Flags (see Section 3 3 11) to check whether or not the send operation finished.

Data Areas

S	D and C
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, TC, DM, *DM

Flags

ER ———— The specified node number > 127.
Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)
The data sent overflows the data area boundaries.
There is no SYSNET Link Unit.

4-12-2
Receive - RECV(98)

RECV receives data from a device linked through SYSNET.

**Ladder Symbol and
Key Sequence**

```
┌─────────────┐
│   RECV(98)  │
├─────────────┤
│      S      │
├─────────────┤
│      D      │
├─────────────┤
│      C      │
└─────────────┘
```
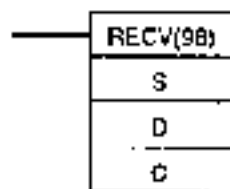
S: Source beginning channel
   (on node number N)
D: Destination beginning channel
   (on the receiving PC)
C: Control data (3 channels)

For RECV, enter

[FUN] [9] [8] [WRITE]

[S] [WRITE]

[D] [WRITE]

[C] [WRITE]

The specified number of channels of data sent from node N, beginning with the source channel S, are written to the requesting PC's destination channels beginning at D.

Control Data

Channel	Bits 15 to 8	Bits 7 to 0
C	Number of channels (0 to 1000) (4-digit hexadecimal)	
C+1	Destination NSB/NSU (0,1)	Network number 0 to 255 (2-digit hexadecimal)
C+2	Destination Board Number (0,1,2)	Destination Node No. 0 to 127 (2-digit hexadecimal)

NSB: Network Service Board
NSU: Network Service Unit

NSB, NSU and the board number are all normally set to 0. Refer to the SYSNET Link System Manual for details.

Example

	15			0				
C	0	0	0	5		S	⇒	D
C+1	0	0	0	0		S+1		D+1
C+2	0	0	0	1		S+2		D+2
						S+3		D+3
						S+4		D+4

Example Circuit:
Ladder Diagram and
Mnemonic Code

0000

		RECV(98)
	01	
	LR 20	
	DM 610	

Address	Instruction	Data	
0000	LD		0000
0001	RECV(98)		—
			01
		LR	20
		DM	610
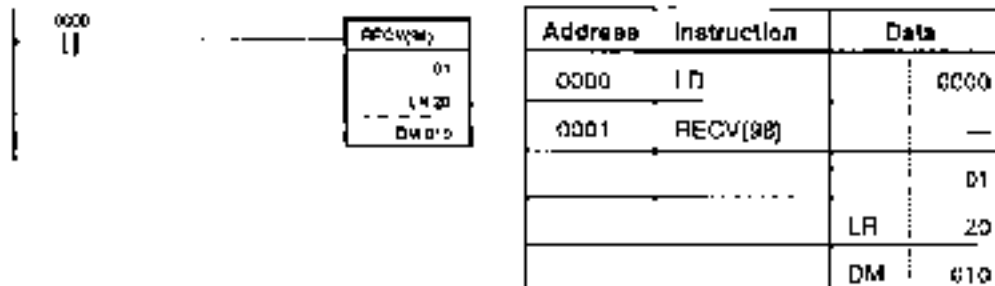
When input 0000 turns ON, channels LR 20 to 24 on the requesting PC receive the data from channels 01 to 05 which are assigned to node number 10.

Use the SYSNET Error and Run Flags (see Section 3-3-11) to check whether or not the receive operation finished

Data Areas

S	D and C
IR, SR, HR, LR, TC, DM, *DM	IR, HR, LR, TC, DM, *DM

Flags

ER ——— The specified node number > 127.

Indirectly addressed DM channel is non-existent.
(DM data is not BCD, or the DM area boundary has been exceeded.)
The data sent overflows the data area boundaries.
There is no SYSNET Link Unit.

4-12-3
About SYSNET Send and Receive Operations

SYSNET send and receive operations are based on command/response processing. That is, the transmission does not complete until the requesting node acknowledges a response from the target node. Note that the SYSNET Run Flag is not set (to 1) until the first END after the transmission is completed. Refer to the SYSNET Link Unit manual for details about command/response operations.

A SYSNET send or receive instruction is executed only once, however multiple send/receive instructions are permitted. To coordinate the error-free execution of SYSNET send and receive instructions, use the SR dedicated input control flags as described in the following table

SR Flag	Functions
RUN Flag (SR 6004)	0 during SEND/RECV execution (including command/response processing)
	0 following normal completion of SEND/RECV (i.e, after reception of response signal)
ERROR Flag (SR 6003)	1 after an unsuccessful SEND/RECV attempt. Error status is maintained until the next SEND/RECV occurs
	Error types: Timeout Error (command/response time > 1 sec) SEND/RECV Data Error

Timing
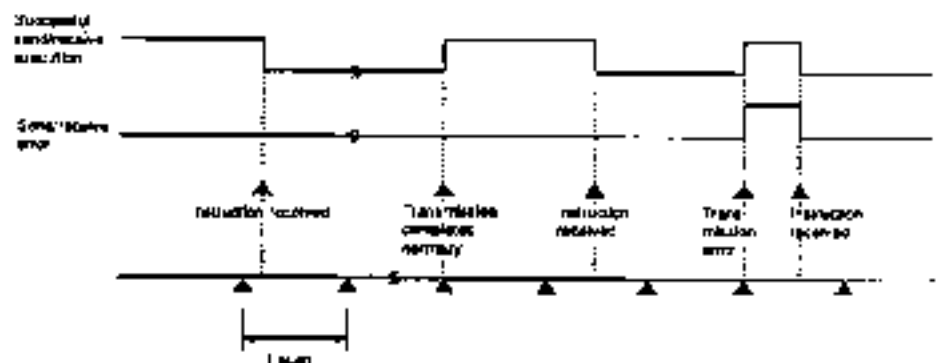
SYSNET Programming Example.
Multiple SEND/RECV

To guarantee successful SEND/RECV operations, your program must have exclusive control of the SYSNET RUN and ERROR Flags.
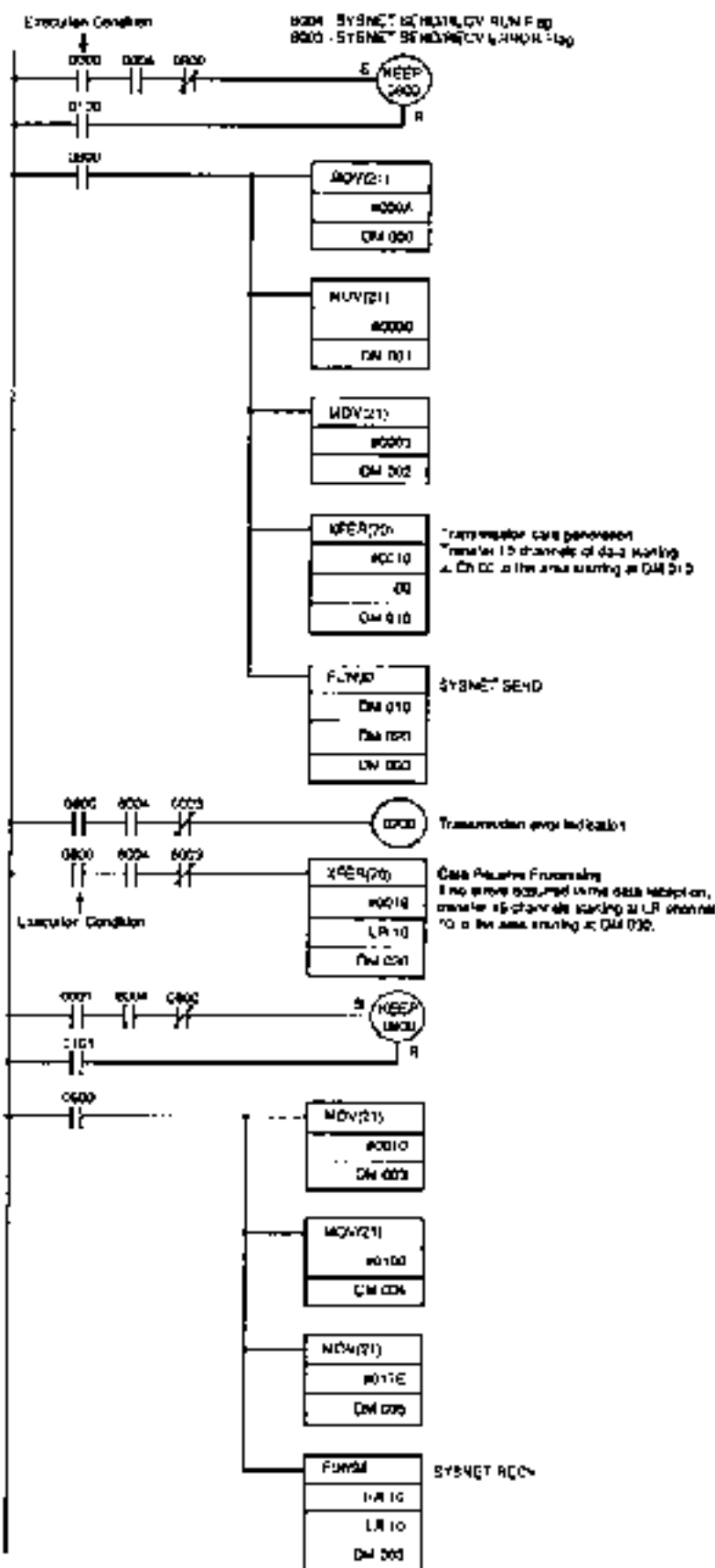
SYSNET SEND Controls
Transfer 10 channels of data, starting from DM 010 of
the requesting PC, to node No. 3 starting at DM 020

DM 000	0	0	0	A	Number of channels of data to transfer
DM 001	0	0	0	0	Destination – h50
DM 002	0	0	0	1	Destination Node Number = 3

SYSNET RECV Controls
Starting at LR 10 of the requesting PC, 16 channels of data
from Node No. 126, starting from HR 10, will be received

DM 003	0	0	1	0	Number of channels of data to transfer
DM 004	0	1	0	0	Destination – h3u
DM 005	0	1	7	E	Destination Board Number = 1 Destination Node Number = 3

SECTION 5
Scan Time and I/O Response Time

One of the most important factors when designing a PC-based control system is timing. How long does it take the PC to execute all the instructions in the program? How long does it take the PC to produce a control output in response to an input signal? For accurate system operations, these values must be known.

Although the scan time of the PC can be automatically calculated and monitored by the Programming Console, it is important to understand the concept of timing when designing and programming a control system.

The purpose of this section is to explain what scan time and I/O response time are, and to show how to calculate these quantities. Instruction execution times are listed in 5-3.

5-1
Scan Time and System Reliability

When the PC executes the program in its memory, a series of operations are performed inside the PC. These internal operations can be broadly classified into four categories:

1. Overseeing processes, such as watchdog timer resetting and diagnostic operations

2. Data input and output

3. Instruction execution

4. Peripheral device command servicing

Scan time is the total time required for the PC to perform all of the above operations. The duration of the scan time differs depending on the configuration of the system, the number of I/O points, the programming instructions used, and whether or not peripheral devices are connected.

The average, maximum, and minimum scan times can be displayed on the Programming Console display (refer to 2-3-8).

Internal Operations
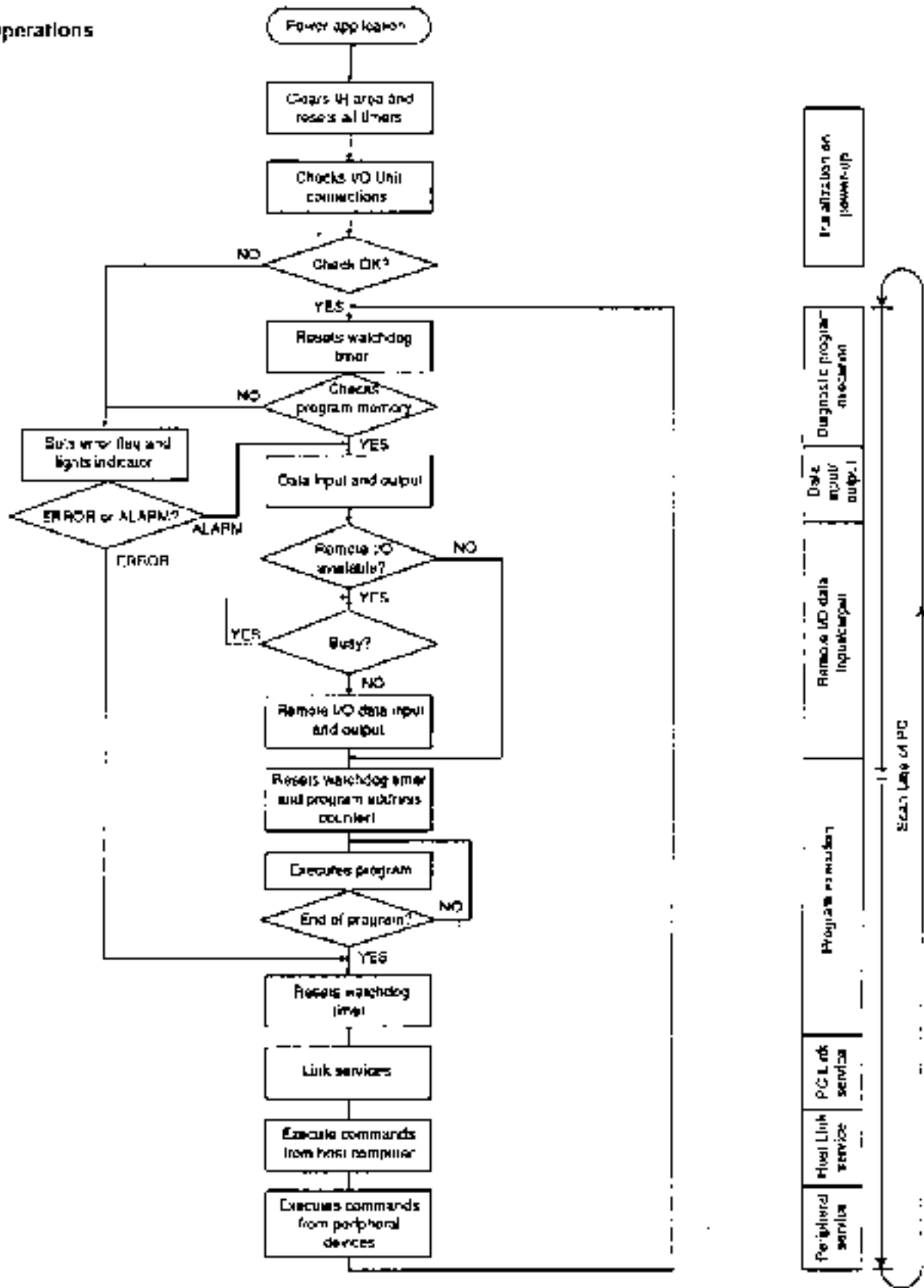
¹ The program address counter is used by the PC to control execution of the program. It must be reset to zero so that the program will be executed from the beginning.

Within the PC, the watchdog timer measures the scan time and compares it to the set value of the watchdog timer. If the scan time exceeds the set value of the watchdog timer, a FALS 9F error is generated and the CPU stops. (Refer to 4-10-2 Set Watchdog Timer.)

Even if the scan time does not exceed the set value of the watchdog timer, a large scan time can adversely affect the accuracy of system operations as shown in the following table.

Scan Time (ms)	Possible Adverse Affects
> 10	High-speed timer TIMH malfunction
> 20	0.02-second clock pulse malfunction
> 100	0.1-second clock pulse malfunction
> 200	0.2-second clock pulse malfunction
> 6500	FALS 9F generated and the system halts

The flowchart opposite illustrates the sequence of internal operations for the PC and the following table lists the time required for each internal operation.

		Process	Execution Time
Overseeing processes	(1)	• Resets watchdog timer • Checks I/O bus • Checks scan time • Checks program memory	1.4 ms (fixed)
I/O Unit refreshing	(2)	• Reads input data and updates IR input bits • Writes IR output data to Output Units	0 to 0.64 ms 20 μs / channel (16 points)
Remote I/O processing	(3)	• Reads Remote Input Unit data to IR area • Writes data from IR area to Remote Output Units	0 to 4.64 ms 1 ms / CPU-rack-mounted Master + 20 μs / channel on the Slave Rack(s).
Instruction execution	(4)	Executes user program instructions	Total of instruction execution times. Differs depending on size of program and instructions used. (see 5-3 Instruction Execution Times)
PC Link servicing	(5)	Transfers the data of the LR area to the PC Link Units.	1.2 ms (0 ms when no PC Link Units are used).
Host Link servicing	(6)	Processes host computer commands	1 to 5 ms [(1) + (2) + (3) + (4) + (5)] × 0.05 ms (0 ms if no Host Link Unit is mounted)
Peripheral device servicing	(7)	Processes peripheral device commands (e.g. from Programming Console/GPC)	1 to 5 ms [(1) + (2) + (3) + (4) + (5)] × 0.05 ms. (0 ms if no peripheral device is connected)

The scan time can be obtained by adding (1) through (7) in this table.

5-2
Calculating Scan Time

The system configuration must be taken into consideration when calculating the total scan time. This means taking into account such things as the number of I/O Units, the programming instructions used, and whether or not peripheral devices are employed. This subsection shows scan time calculation examples. To simplify the examples, the instructions used in the programs have been assumed to be all either LD or OUT. The average execution time for the instructions is then 4 µs and there is one instruction per program address

5-2-1
When Only I/O Units Are Used

Conditions

I/O Units.   Eight 16-point Units + eight 32-point Units
Program: 6,000 addresses
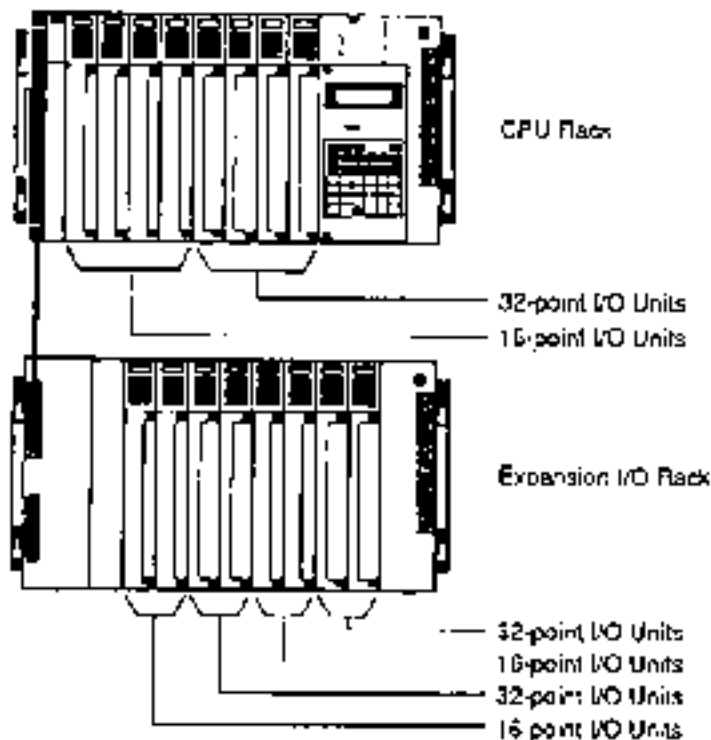
CPU Rack

32-point I/O Units
16-point I/O Units

Expansion I/O Rack

32-point I/O Units
16-point I/O Units
32-point I/O Units
16-point I/O Units

Calculation Example

	Process	Calculation	Process Time
(1)	Overseeing processes		1.4 ms
(2)	I/O refresh	[(16 pts × 8 + 32 pts × 8) + 16] × 20 μs	0.48 ms
(3)	Execution	4 μs × 6,000 addresses	24 ms
(4)	Peripheral device service	(1.4 ms + 0.48 + 24 ms) × 0.05	1.3 ms

Scan time	(1) + (2) + (4) + (7)	Approx. 27.2 ms

5-2-2
When I/O Units, Remote I/O Units, Host Link Units, and PC Link Units Are Used

Conditions

I/O Units.	Four 16-point Units + one 32-point Unit
Remote I/O Master:	One
Host Link Unit:	One
PC Link Unit:	One
Program.	6000 addresses
Remote I/O Slave:	Mounted with four 16-point I/O Units and four 32-point I/O Units
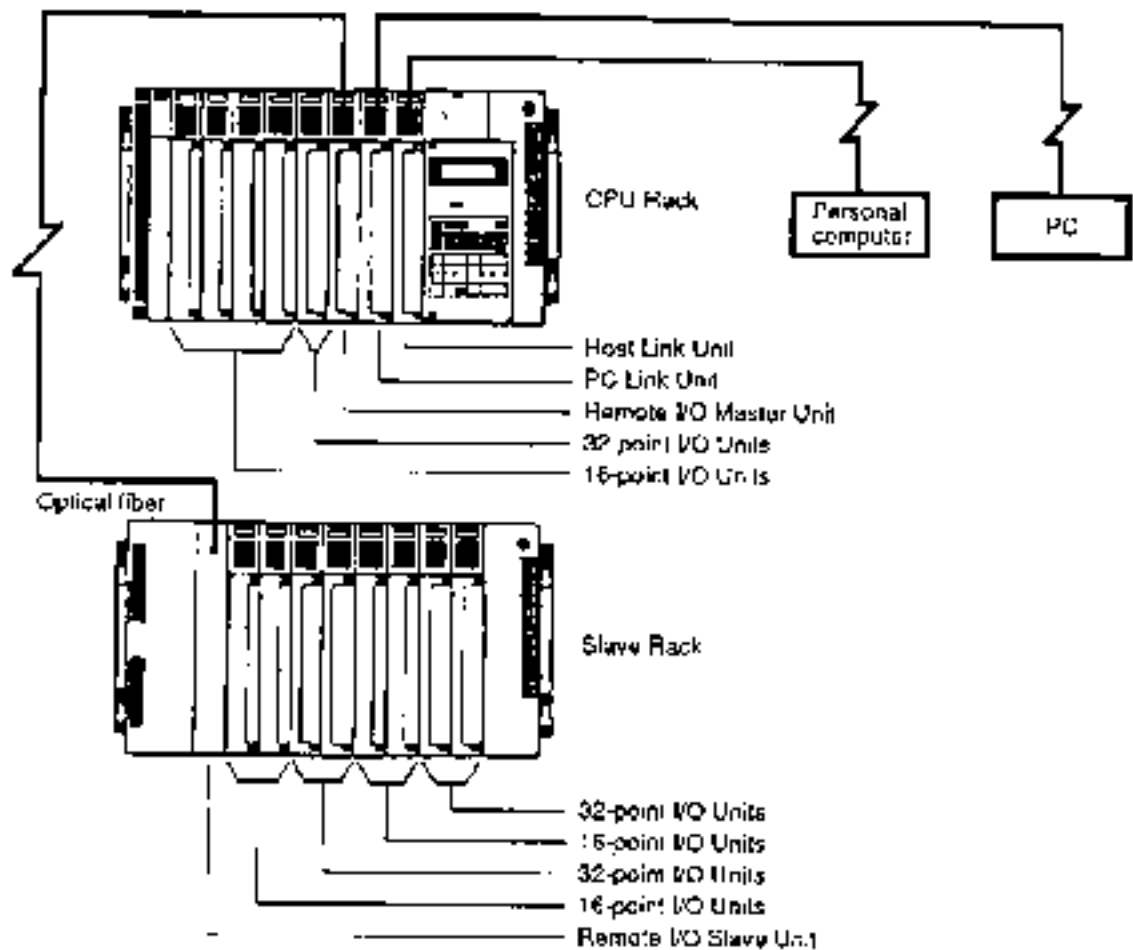
CPU Rack

Personal computer

PC

Host Link Unit
PC Link Unit
Remote I/O Master Unit
32-point I/O Units
16-point I/O Units

Optical fiber

Slave Rack

32-point I/O Units
16-point I/O Units
32-point I/O Units
16-point I/O Units
Remote I/O Slave Unit

Calculation Example

	Process	Calculation	Process Time
(1)	Overseeing processes		1.4 ms
(2)	I/O refresh	[(16 pts x 4 + 32 pts x 1) + 16] x 20 μs	0.12 ms
(3)	Remote I/O refresh	1 ms + [(16 pts x 4 + 32 x 4] + 16] x 20 μs	1.24 ms
(4)	Execution	4 μs x 6,000 addresses	24 ms
(5)	PC link service		1.2
(6)	Host link link service	(1.4 ms + 0.12 ms + 1.22 ms + 24 ms) x 0.05	1.33 ms
(7)	Peripheral device service	(1.4 ms + 0.12 + 1.24 ms + 24 ms) x 0.05	1.33 ms

Scan time	(1) + (2) + (3) + (4) + (5) + (6) + (7)	30.6 ms

5-3
Instruction
Execution Times

This sub-section lists the execution times for all instructions that are available for the C500 PC. The conditions which may affect the execution time of a given instruction are described briefly where relevant.

Also shown are non-execution times for all of the instructions. Non-execution time is the time required for the PC to process an instruction when it is not executed. Some non-execution times vary according to the reason the instruction was not executed. If a shift instruction, for example, is within an IL-ILC block or a JMP-JME block and is not executed, its non-execution time will differ from when its reset is ON and it doesn't execute.

Execution times are expressed in units of μs except where noted.

FUN No.	Instruction	No. of words	Execution time (μs)	Conditions	Non-execution time (μs)		
—	LD	4	4				
—	LD NOT	4	4				
—	AND	3	3				
—	AND NOT	3	3				
—	OR	3	3				
—	OR NOT	3	3				
—	AND LD	5	3.5				
—	OR LD	5	3.5				
—	OUT	6	6				
—	OUT NOT	6	9				
					R	IL	JMP
—	TIM	8	83	With a constant or *DM	113	106	55
—	CNT	8	78	With a constant or *DM	110	29	29
00	NOP	3	2				
01	END	3	63			—	
					H	IL	JMP
02	IL	4	22		17	14	14
03	ILC	4	17		14		
04	JMP	4	22		17	14	14
05	JME	4	17		14		
06	FAL	6	126		14	19	19
07	FALS	5	103		14	19	19
10	SFT	9	104	When shifting 1 Ch by 1 Ch	397	17	17
				When shifting 1 Ch by 61 Chs		95	
11	KEEP	6	6		—		
12	CNTR	8	92	With a constant	100	19	29
				With *DM	160	19	29

FUN No.	Instruction	No. of words	Execution time [μs]	Conditions	Non-execution time (μs)		
					R	IL	JMP
13	DIFU	6	43		43	29	34
14	DIFD	6	43		43	39	43
15	TIMH	8	83	With a constant	113	106	55
			3,896	With Ch data			
16	WSFT	8	157	When shifting 1 Ch by 1 Ch	21		
			3,896	When shifting *DM by 510 Chs			
20	CMP	8	85	When comparing a constant to a Ch	21		
			198	When comparing two *DM Chs			
21	MOV	8	88	When transferring a constant to a Ch	21		
			200	When transferring *DM to *DM			
22	MVN	8	91	When transferring a constant to a Ch	21		
			203	When transferring *DM to *DM			
23	BIN	8	148	When transferring a Ch to a Ch	21		
			250	When transferring *DM to *DM			
24	BCD	8	165	When transferring a Ch to a Ch	21		
			278	When transferring *DM to *DM			
25	ASL	7	85	When shifting a Ch	22		
			142	When shifting *DM			
26	ASR	7	86	When shifting a Ch	22		
			142	When shifting *DM			
27	ROL	7	90	When rotating a Ch	22		
			147	When rotating *DM			
28	ROR	7	88	When rotating a Ch	22		
			145	When rotating *DM			
29	COM	7	91	When inverting a Ch	22		
			138	When inverting *DM			
30	ADD	10	149	Constant + Ch ⟶ Ch	21		
			317	*DM + *DM ⟶ *DM			
31	SUB	10	155	Constant + Ch ⟶ Ch	21		
			323	*DM - *DM ⟶ *DM			
32	MUL	10	459	Constant x Ch ⟶ Ch	21		
			630	*DM x *DM ⟶ Ch			
33	DIV	10	834	Ch + constant ⟶ Ch	21		
			934	*DM + *DM ⟶ *DM			
34	ANDW	10	102	Constant AND Ch ⟶ Ch	21		
			270	*DM AND *DM ⟶ *DM			
35	ORW	10	102	Constant OR Ch ⟶ Ch	21		
			270	*DM OR *DM ⟶ *DM			
36	XORW	10	102	Constant XOR Ch ⟶ Ch	21		
			270	*DM XOR *DM ⟶ *DM			

FUN No.	Instruction	No. of words	Execution time (µs)	Conditions	Non-execution time (µs)
37	XNRW	10	102	Constant XNR Ch →Ch	21
			270	*DM XNR *DM ⟶ *DM	
38	INC	7	102	When incrementing a Ch	22
			158	When incrementing *DM	
39	DEC	7	102	When decrementing a Ch	22
			158	When decrementing *DM	
40	STC	4	25		12
41	CLC	4	25		24
70	XFER	10	278	When transferring 1 Ch	21
			4,070	When transferring 511 Chs by *DM	
71	BSET	10	183	When setting constant to Ch	21
			2,133	When setting *DM to 511 DM Chs	
72	ROOT	10	333	When outputing Ch data to a Ch	21
			1,068	When outputting 99999999 in *DM to *DM	
73	XCHG	10	142	Ch ⟷ Ch	21
			259	*DM ⟷ *DM	
74	SLD	10	180	When shifting 1 Ch	21
			1,645	When shifting 1,000 DM Chs by *DM	
75	SRD	10	178	When shifting 1 Ch	21
			1,645	When shifting 50 DM Chs by *DM	
76	MLPX	10	137	When decoding Ch data to a Ch	21
			264	When decoding *DM data to *DM	
77	DMPX	10	151	When encoding Ch data to a Ch	21
			300	When encoding *DM data to *DM	
78	SDEC	10	133	When decoding Ch data to a Ch	21
			260	When decoding *DM to *DM	
87	WRIT	8	0.36 ms	When writing 1 Ch	21
			3.64 ms	When writing 255 *DM Chs	
88	READ	8	0.38 ms	When reading 1 Ch	21
			3.65 ms	When reading 255 *DM Chs	
90	SEND	4	0.25 ms	When sending *DM data	21
			0.41 ms	When sending data other than *DM	
98	RECV	4	0.25 ms	When receiving data to an area other than *DM	21
			0.41 ms	When receiving data to *DM	

5-4
I/O Response Time

Response time is the time it takes for the PC to output a control signal after it has received an input signal. How long it takes to respond depends on factors such as the system configuration and when the CPU receives the input signal relative to the start of a scan. The response time for a single PC is discussed in this section. For response times for configurations involving the systems below, refer to the appropriate systems manual as indicated.

PC to Remote I/O (SYSBUS) system. Remote I/O Systems manual
PC Link system: PC Link Systems manual
Host Link system (SYSWAY);   Host Link Systems manual

Response Time for a Single PC
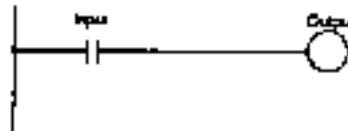
-Minimum I/O Response Time

The PC responds most quickly when it receives an input signal just prior to the input refresh phase of the scan. To find the response time for this case, add the input ON-delay and output ON-delay to the scan time of the PC.
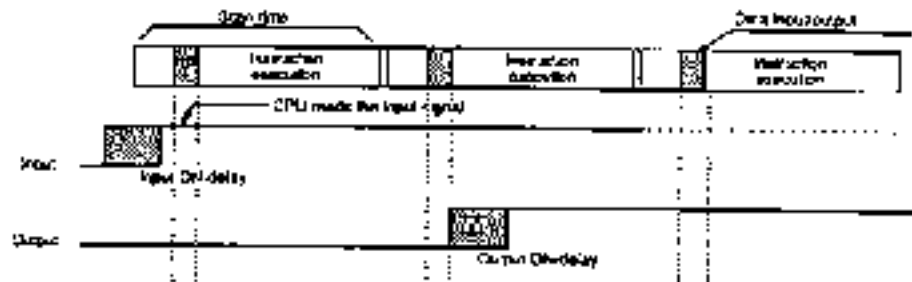
I/O response time = Input ON-delay + Scan time + Output ON-delay

-Maximum I/O Response Time

The PC takes longest to respond when it receives the input signal just after the input refresh phase of the scan. In this case the CPU does not recognize the input signal until the beginning of the next scan. Therefore, the maximum response time is the sum of the input ON-delay, the output ON-delay, and two scan times.

-Maximum I/O Response Time
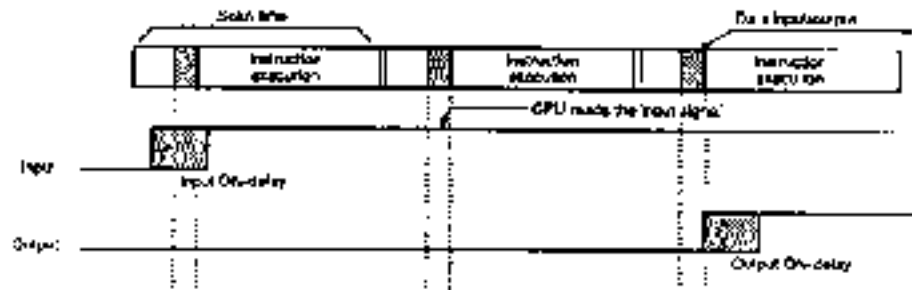
I/O response time = Input ON-delay + (Scan time x 2) + Output ON-delay

-Calculation Example

Input ON-delay	1.5 ms
Output ON-delay	15 ms
Scan time	20 ms

Minimum I/O response time = 1.5 ms + 20 ms + 15 ms = 36.5 ms
Maximum I/O response time = 1.5 ms + (20 ms x 2) + 15 ms = 56.5 ms

SECTION 6

Error Messages and Troubleshooting

The C500 PC has self-diagnostic functions to identify many types of abnormal system conditions. These functions minimize downtime and enable quick, smooth error correction

The error light on the front panel of the Programming Console indicates hardware errors such as CPU, I/O Unit and Remote I/O Unit malfunctions. The warning light indicates such things as scan time overrun, battery error or user-defined errors.

In addition, the Programming Console acts as a monitor by displaying explicit error messages and FAL numbers.

This section lists all the error messages displayed on the LCD of the Programming Console

6-1
Programmed Alarms and Error Messages

Use the diagnostic instructions, FAL(06) and FALS(07), to alert the system when certain arbitrarily defined error conditions arise. Note that it is entirely up to the user to decide the conditions under which a FAL or FALS is executed. Refer to 4-10-1 Failure Alarm for details about how to use these diagnostic instructions in your program.

When "FAL n" is executed, the value of the FAL number "n" is stored as a 2-digit BCD code in the SR area (See 3-3-3 FAL Number Output Area). FAL also lights up the warning indicator on the front panel of the CPU
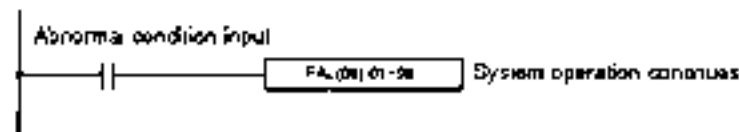
When FALS(07) is executed, the error indicator on the front panel of the CPU lights up and system operations are halted.

215
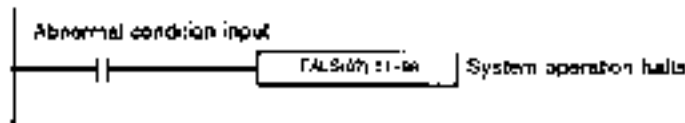
To resume system operations, determine the cause of the error, make correc
tions, then clear the error. Note that to clear an FALS error, the system must
be in PROGRAM mode.

FAL numbers 01 to 99 are arbitrarily assigned failure codes. FAL 00 is re
served for clearing other FAL codes present in the system (see 4-10-1 Fail-
ure Alarm)
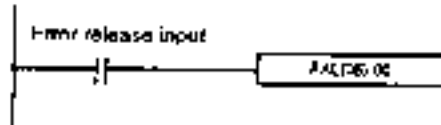
6-2
Reading and Clearing Errors and Messages

To display an error or a message on the Programming Console, press the CLR, FUN, and MONTR keys.

Then to display multiple errors or messages, press the MONTR key again. (Note that if the system is in PROGRAM mode, pressing the MONTR key clears the error message.) Continue pressing the MONTR key, taking note of the errors or messages, until all have been cleared and the message "ERR CHK OK" is displayed. Then proceed to correct each of the errors.

It is not possible to clear an error or a message while in RUN or MONITOR mode; the PC must be in PROGRAM mode.

The beeper will sound if the system cannot clear an error or a message for some reason. If this situation arises, display and clear the error or the message again.

The asterisks in the error messages on the following page indicate numerals in the actual display.

CLA	⌐⌐⌐⌐
FUN	0000 FUN (??)
MONTR	0000ERR CHK OK
MONTR	MEMORY ERR
MONTR	NO END INST
	I/O BUS ERR
	I/O SET ERR
	I/O UNIT OVER
	SYS FAIL FALS***

System operation halts (Highest priority errors)

	SYS FAIL FAL. ##
	DPL ERR
	RMTE I/O ERR *——— Remote I/O Unit No.
	I/O VER ERR
	SCAN TIME OVER
MONTR	0000ERR CHK OK

Non-abortive error (2nd highest priority)

When all errors have been cleared
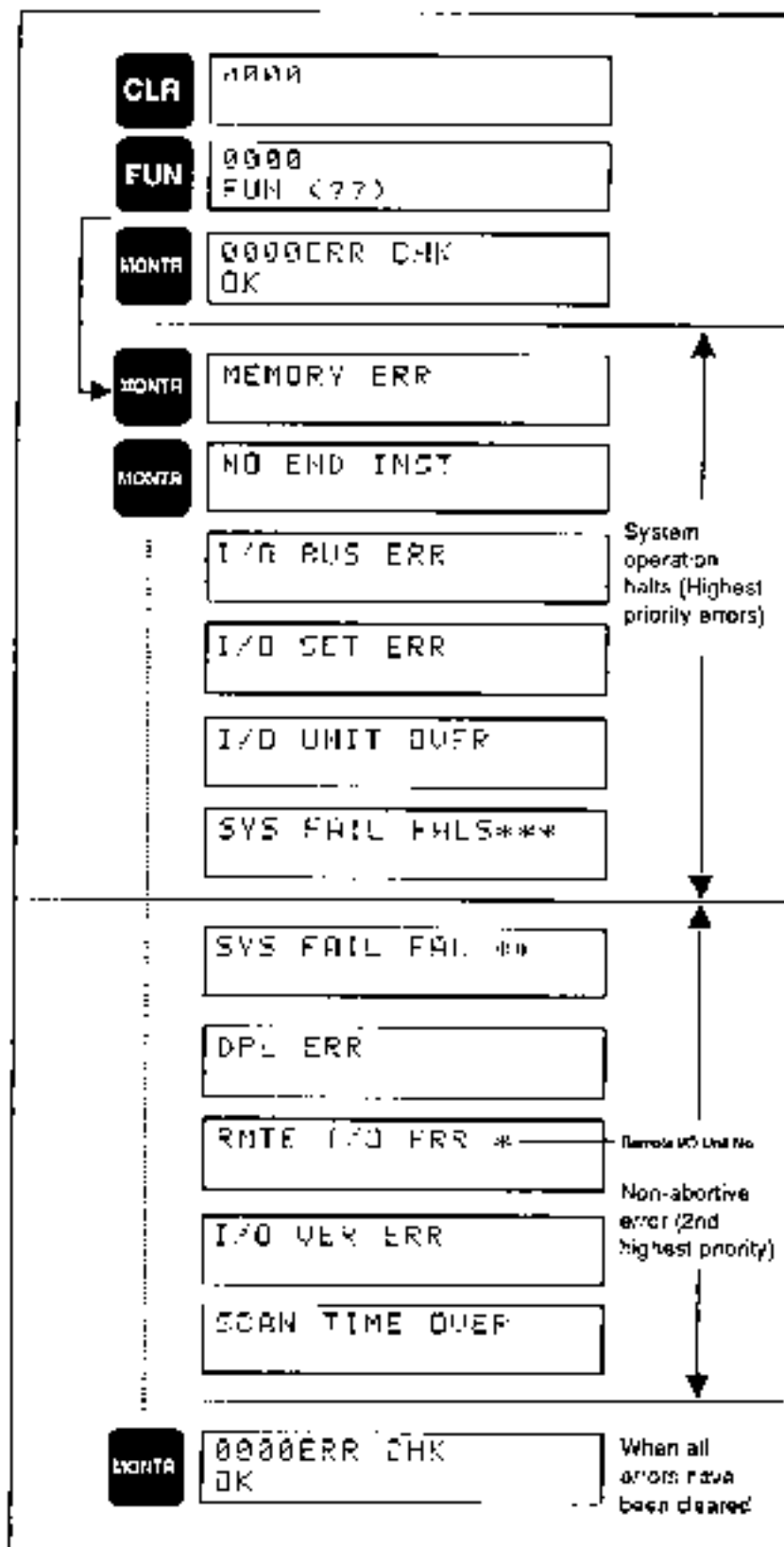
6-3
System Errors

Error State	Item	Message	CPU Unit LED States				
			Power	Mid-run	Error	Warning	Load Off
Not Operating	Waiting for start input		⌀	●	—	—	—
	Waiting for Remote I/O Units	CPU WAIT'G	⌀	●	—	—	—
	Power Interruption		●	●	●	●	●
Fatal	CPU error		⌀	●	⌀	—	—
	Memory error	MEMORY ERR	⌀	●	⌀	—	
	No END instruction	NO END INST	⌀	●	⌀	—	—
	I/O Bus error	I/O BUS ERR	⌀	●	⌀	...	—
	I/O table overflow	I/O UNIT OVER	⌀	●	⌀	—	—
	Invalid I/O table	I/O SET ERR	⌀	●	⌀	—	—
	System error	SYS FAIL FALS***	⌀	●	⌀	—	—

⌀ means that the LED is lit, ● that the LED is not lit.
— means that the LED being lit or not makes no difference.

Mid-run output	SR Area	Failure Code	Main cause of Error	Correction
	—		When the start input of CPU Power Unit is OFF.	Short-circuit the start input terminal of the CPU Power Unit
OFF	—		Remote I/O Unit power off, Terminator not set	Check the power unit supply and the Terminator setting.
OFF	—		When power has been cut off for at least 10 ms	Check voltage source and power lines. Try to power-up again.
OFF	··		Watchdog timer 130 ms or more.	• In PROGRAM mode, re-power-up the system • Check the user program again.
OFF	—	F1	• Memory Unit is incorrectly mounted or missing. • A memory parity error occurred • Improper Instruction	• Do a program check and fix the error. • Make sure that the Memory Unit is mounted correctly. • Check that the battery is inserted properly. • Clear error after fixing.
OFF	—	F0	END is not written at the end of the program.	Write END in the final address of the program
OFF	—	(Note)	A failure is in the bus line between the CPU rack and the Expansion I/O Racks.	Check the number of points with I/O Table Read, and use I/O Table Register to match the registered table to the actual table. • Clear error after fixing.
OFF	—	L1	I/O limitations exceeded	• Check I/O table with I/O Table Read. • Reassign I/O channels and then generate the new table with I/O Table Register
OFF	—	E0	When I/O Units have been replaced and the registered I/O table does not agree with the I/O Units actually mounted to the PC.	• Check I/O table with I/O Table Verify. • Reassign I/O channels and then generate the new table with I/O Table Register.
OFF	—	01 to 99 9F	FALS has been executed by the system	• The scan time may be too long. Check the program • Also possible when FUN07 is executed with 120 to 130 ms scan time.

Note: 6-3 indicates the rack number.

Error State	Item	Error display	CPU Unit LED States				
			Power	Mid-run	Error	Warning	Load Off
Non-fatal	System error	```SYS FAIL FAL**```	⊠	⊠	●	⊠	—
	Scan time overrun	```SCAN TIME OVER```	⊠	⊠	●	⊠	—
	I/O table verification error	```I/O VER ERR```	⊠	⊠	●	⊠	—
	Remote I/O error	```RMTE I/O ERR``` Remote I/O Unit #	⊠	⊠	●	⊠	—
	Battery error	```BATT LOW```	⊠	⊠	●	⊠	—
	Host Link error	—	⊠	⊠	●	—	—
	PC Link error	—	⊠	⊠	●	—	—
*	Load cut-off error	—	⊠	⊠	●	—	⊠

*   All I/O Unit outputs cut off
⊠   means that the LED is lit, ● that the LED is not lit.
---   means that the LED being lit or not makes no difference.

Mid-run output	SR Area	Failure Code	Main cause of Error	Correction
ON		01 to 99	FAL has been executed by the program.	Check the program.
ON	6109 ON	F8	Watchdog timer limit has been exceeded (100 - 130 ms)	Program scan time is too long. See Section 5.
ON	6110 ON	E7	The registered I/O table does not agree with the actual I/O Units.	Check the I/O Unit connections with I/O Table Verify, and set the I/O Units properly. Then use I/O Table Register to match the registered table to the actual table
ON	6112 ON	(Note)	A failure has occurred in the transmission line between a Master and Slave	Check the transmission line between the Master and Slave.
ON	6108 ON	F7	Battery is bad or is not installed properly.	Check battery connections, or replace battery.
ON	see Ref. #1 below		An error between the Host Link Units.	Refer to the Host Link Systems operation manual.
ON			An error occurred in the PC Link Unit	Refer to the PC Link Systems operation manual.
ON	6015 ON		When the load off flag is ON (SR 6015)	

Note: 6-3 indicates the Remote Master Unit number.

Ref. # 1: For Rack-mounting Units, 6111 ON
For CPU-mounting Units, 6006 ON

6-4
Program Input
Errors

Error Message	Correction
******REPL ROM	A EP-ROM Unit is mounted for the program memory. Replace it with a RAM or EEP-ROM Unit, and then write the data. The mounted RAM Unit is write protected. Turn the write enable on
******PROG OVER	The program cannot be written because the instruction in the last address is not NOP or END. Check the program and clear all addresses after END.
******ADDR OVER	The specified address exceeds the program memory area boundary. Correct the program
000001/OTBL WRIT DISABLED	The I/O table cannot be registered. Check the number of Remote I/O Units connected, duplicated use of Optical Transmitting I/O Units, the absence of a Terminator for a Remote I/O system, or too many I/O Units. Check all the I/O Units.
******SETDATA ERR	What should be a decimal constant has been entered as a hexadecimal number or, a constant exceeding the predetermined range has been entered. This generates a FALS 00. Correct the constant.
******I/O NO. ERR	An attempt has been made to enter I/O data which exceeds the predetermined range. Check the range and correct the data.

6-5
Program Errors

Error Message	Correction
****** ?????	The program has been destroyed. Write the program into memory again
******CIRCUIT ERR	The number of logical starts (LDs) does not match up with the number of logical operations (OR LDs and AND LDs). Correct the program.
******OPERAND ERR	The variable operand data specified is incorrect. Check the operand data range for each instruction.
******COIL DUPL	More than one instruction is trying to use the same output number. Check all output number assignments.
******NO END INST	There is no END at the end of the program. Write an END as the final program instruction
******LOCN ERR	The instruction currently displayed is in the wrong area.

Error Message	Correction
＊＊＊＊＋IL-ILC ERR	IL and ILC are not used in pairs. Correct the program.
＊＊＊＋JMP-JME ERR	JMP 00 and JME 00 are not used in pairs. Correct the program.
＊＋＊＊＊JME UNDEFD	The corresponding JME for a given JMP does not exist. Correct the program.
＊＊＊＊＊JMP UNDEFD	The corresponding JMP for a given JME does not exist. Correct the program.
＊＊＊＊＊DUPL	The number of the currently displayed instruction has already been programmed. Correct the program.
＊＊＊＊＊SBN-RET ERR	Incorrect usage of the displayed instruction (SBN or RET). Incorrect SBN usage is caused by more than one SBN having the same subroutine number. Correct the program.
＊＊＊＊＋SBN UNDEFD	The subroutine called by SBS does not exist. Correct the program.
＊＊＊＊＋SBS UNDEFD	A defined subroutine is not called by the main program. When this message is displayed because of interrupt routine definition, there is no problem. In all other cases, correct the program.
＊＊＊＊＊DIFOVER＊＊	More than 128 DIFU/DIFD's are programmed. Correct the program.

Cassette Tape Usage Errors

Error Message	Cause	Correction
00000 ERR＊＊＊＊＊＊＊＊ FILE NO.＊＊＊＊＊＊＊＊	The cassette file number and the file number specified by the user do not agree.	Make sure the file number is entered correctly, then try the tape operation again.
＊＊＊＊＊＊ VER ERR	The contents of the cassette tape do not match the contents of the program memory	Check the contents of the cassette tape and that of program memory.
＊＊＊＊＊ TAPE ERR	The cassette tape has an error.	Replace the tape with another.

6-6
Troubleshooting

CPU

Symptom	Possible Cause	Correction
Power Supply does not turn on.	Voltage selector terminal setting error.	Connect the voltage selector terminal correctly.
	Fuse is blown.	Replace Fuse.
Fuse blows repeatedly.	Voltage selector terminal setting error.	Connect the voltage selector terminal correctly.
	Circuit board is short-circuited, or burnt.	Replace CPU rack Power Supply Unit, or Backplane.
Run indicator does not light.	Start input is OFF.	Turn the start input ON.
	Programming error.	Correct the program.
	Power line is defective.	Replace CPU Power Supply Unit
"Run Output" does not turn on.	Power circuit is defective.	Replace CPU Power Supply Unit.
I/Os following a particular I/O number do not operate.	I/O bus is defective.	Replace Backplane
Abnormal I/Os of Expansion I/O Rack are in units of 8.	I/O Connecting Cable is defective. (Cable wiring is broken.)	Replace the I/O Connecting Cable.
	I/O bus is defective.	Replace Backplane.
One I/O turns ON erroneously.	I/O bus is defective.	Replace Backplane.
All I/Os of a particular I/O Unit do not operate.	I/O Bus is defective	Replace Backplane.

Input Unit

Symptom	Possible Cause	Correction
All Inputs do not turn ON. (Operation indicators do not light.)	External input voltage is not supplied.	Supply power.
	External input voltage is low	Raise supply voltage.
	Terminal screws are loose.	Tighten terminal screws.
	Faulty contact of terminal block connector.	Replace terminal block connector.
All Inputs do not turn ON. (Operation indicators are lit.)	Input circuit is defective.	Replace defective Input Unit.
All Inputs do not turn OFF.	Input circuit is defective.	Replace defective Input Unit.
One input bit does not turn ON.	Input device is defective.	Replace Input device.
	Input wiring is broken.	Check and replace input wiring.
	Terminal screws are loose.	Tighten terminal screws
	Faulty contact of terminal block connector.	Replace terminal block connector.
	Input ON time is too short.	Adjust external input device.
	Input circuit is defective.	Replace defective Input Unit
	Input number is incorrectly programmed as OUT.	Correct the program
One input bit does not turn OFF.	Input circuit is defective.	Replace defective Unit.
	Input number is incorrectly programmed as OUT.	Correct the program.
Inputs turn ON and OFF irregularly.	External input voltage is low.	Raise external voltage
	Malfunction due to noise.	Countermeasures against noise Install surge suppressor. Install insulating transformer Wire with shielded cable
	Terminal screws are loose.	Tighten terminal screws.

Input Unit (cont.)

Symptom	Possible Cause	Correction
Inputs turn ON and OFF irregularly	Faulty contact of terminal block connector.	Replace terminal block connector
	Common terminal screws are loose.	Tighten common terminal screws.
Abnormal input numbers are in units of 8 bits.	Faulty contact of terminal block connector	Replace terminal block connector.
	Data bus is faulty.	Replace defective Unit.
	CPU is defective.	Replace CPU.
Input operation indicator does not light.	LED indicator is defective.	Replace defective Unit

Output Unit

Symptom	Possible Cause	Correction
All Outputs do not turn ON.	Power for load is not supplied.	Supply power.
		Raise supply voltage.
	Terminal screws are loose.	Tighten terminal screws.
	Faulty contact of terminal block connector.	Replace terminal block connector.
	Fuse is blown.	Replace fuse.
	Faulty contact of I/O bus connector.	Replace defective Unit.
	Output circuit is defective.	Replace defective Unit.
All Outputs do not turn OFF.	Output circuit is defective.	Replace defective Unit.
One output does not turn ON (Operation indicator does not light.)	Output ON time is too short	Correct the program.
	OUT numbers are programmed in duplicate.	Correct the program.
	Output circuit is defective.	Replace defective Unit
One Output does not turn OFF (Operating indicator is lit.)	Output device is defective.	Replace output device.
	Output wiring is broken.	Check output wiring.
	Terminal screws are loose.	Tighten terminal screws
	Faulty contact of terminal block connector.	Replace terminal block connector.
	Output relay is defective.	Replace defective relay
	Output circuit is defective.	Replace defective Unit.
One output does not turn OFF (Operation indicator does not light.)	Output relay is defective	Replace defective relay.
	Leakage current, or residual voltage.	Replace external load or add dummy resistor.
One output does not turn OFF. (Operation indicator is lit.)	The OUT number has been programmed in duplicate.	Correct the program
	Output circuit is defective.	Replace defective Unit.

Output Unit (cont.)

Symptom	Possible Cause	Correction
Outputs turn ON and OFF irregularly.	Supply voltage for external load is low.	Raise external supply voltage.
	OUT numbers are programmed in duplicate.	Correct the program.
	Malfunction due to noise.	Countermeasures against noise Install surge suppressor. Install insulating transformer. Wire with shielded cable.
	Terminal screws are loose.	Tighten terminal screws.
	Faulty contact of terminal block connector.	Replace terminal block connector.
Abnormal output points are units of 8.	Common terminal screws are loose.	Tighten common terminal screws.
	Faulty contact of terminal block connector.	Replace terminal block connector.
	Fuse is blown	Replace fuse.
	Data bus is faulty.	
	CPU is defective.	Replace CPU.
Output operation indicator does not light.	LED indicator is defective	Replace defective Unit.

APPENDIX A
Standard Models

CPUs and Associated Units

Name	Remarks		Model
CPU Backplane	8 I/O slots	3 Link slots	3G2A5-BC081
		5 Link slots	C500 BC082
	5 I/O slots	3 Link slots	3G2A5-BC051
		5 Link slots	C500 BC052
CPU			3G2C3-CPU11-EV1
RAM Unit	4.4K bytes		3G2A5-MR431
	5.6K bytes		3G2A5-MR831
ROM Unit	5.6K words		3G2A5-MP031
	32K words max.		3G2C5 MP34
PP-ROM chip	27128, 128 bits		ROM-I
	2764, 64 bits		ROM-H
CPU Backplane Power Supply	100 to 120/200 to 240 VAC(selectable)	Output: 7 A 5 VDC	3G2A5-PS221 E
		Output: 12 A 5 VDC	3G2A5-PS223 F
	24 VDC	Output: 7 A 5VDC	3G2A5-PS21 -E
I/O Control Unit			3G2A5-II10

Expansion I/O Backplane

Name	Remarks		Model
Backplane	8 slots		3G2A5-BIC81
	5 slots		3G2A5-BIC51
Power Supply	100 to 120/200 to 240 VAC (selectable)		3G2A5-PS222-E
	24 VDC		3G2A5-PS212-E
I/O Interface Unit			3G2A5-II002
I/O Connecting Cable	Horizontal type	13 cm	3G2A5-CN111
		50 cm	3G2A5-CN511
		1 m	3G2A5-CN121
	Vertical type	30 cm	3G2A5-CN312
		50 cm	3G2A5-CN512
		80 cm	3G2A5-CN812
		1 m	3G2A5-CN122
		2 m	3G2A5-CN222

I/O Units

Name		Remarks			Model
Input Unit	DC	16 mA 5 to 12 VDC		16 pts	3G2A5-ID112
		10 mA 12 to 24 VDC		16 pts	3G2A5-ID213
		10 mA 12 to 24 VDC	ON response time: 15 ms max.	32 pts	3G2A5-ID215
			ON response time: 1.5 ms max.	32 pts	3G2A5-ID218
		10 mA 12 to 24 VDC		32 pts	C500-ID218CN
		7 mA 12 VDC, static		64 pts	C500-ID114
		10 mA 24 VDC, dynamic		64 pts	3G2A5-ID212
		7 mA 24 VDC, static		64 pts	3G2A5-ID219
	AC	10 mA 100 to 120 VAC		16 pts	3G2A5-IA121
		10 mA 200 to 240 VAC		16 pts	3G2A5-IA222
		10 mA 100 to 120 VAC		32 pts	3G2A5-IA122
		10 mA 200 to 240 VAC		32 pts	C500-IA223
	AC/DC	10 mA 12 to 24 VAC/DC		16 pts	3G2A5-IM211
		10 mA 12 to 24 VAC/DC		32 pts	3G2A5-IM212
	TTL	3.5 mA 5 VDC		32 pts	C500-ID501CN
Output Unit	Contact	2 A 250 VAC/24 VDC		16 pts	3G2A5-OC221
		2 A 250 VAC/24 VDC (sep. commons)		16 pts	3G2A5-OC223
		2 A 250 VAC/24 VDC		32 pts	3G2A5-OC224
	Transistor	1 A 12 to 24 VDC		16 pts	C500-OD217
		1 A 12 to 48 VDC		16 pts	3G2A5-OD411
		50 mA 24 VDC (sep. commons)		16 pts	3G2A5-OD215
		0.3 A 12 to 24 VDC		32 pts	C500-OD218
		0.3 A 12 to 48 VDC		32 pts	3G2A5-OD412
		0.3 A 12 to 24 VDC, PNP output		32 pts	3G2A5-OD212
		0.3 A 12 to 48 VDC connect to I/O		32 pts	C500-OC415CN
		0.1 A 24 VDC, dynamic		64 pts	3G2A5-OD211
		0.1 A 24 VDC, static		64 pts	3G2A5-OD213
	Triac	1 A 132 VAC max.		16 pts	3G2A5-OA121
		1 A 250 VAC max.		16 pts	3G2A5-OA222
		1 A 250 VAC max.		24 pts	3G2A5-OA223
		1 A 250 VAC max.		32 pts	C500-OA225
	TTL	35 mA 5 VDC		32 pts	C500-OD501CN
DC Input/Transistor Output Unit		12 to 24 VDC	Input: 10 mA	16 pts each	C500-MD211CN
			Output: 0.3 A		
Dummy I/O Unit		No. of I/O points can be set			3G2A5-DUM01

Special I/O Units

Name	Remarks		Model
A/D Conversion Input	4 to 20 mA +1 to +5 V	2 pts	3G2A5-AD001
	0 to +10 V	2 pts	3G2A5-AD002
	0 to -5 V	2 pts	3G2A5-AD003
	-10 to +10 V	2 pts	3G2A5-AD004
	-5 to +5 V	2 pts	3G2A5-AD005
	4 to 20 mA +1 to +5 V	4 pts	3G2A5-AD006
	0 to +10 V	4 pts	3G2A5-AD007
D/A Conversion Output	4 to 20 mA +1 to +5 V	2 pts	3G2A5-DA001
	0 to +10 V	2 pts	3G2A5-DA002
	0 to +5 V	2 pts	3G2A5-DA003
	-10 to +10 V	2 pts	3G2A5-DA004
	-5 to +5 V	2 pts	3G2A5-DA005
High-Speed Counter	6 BCD digits, 50K cps	1 pt	3G2A5-CT001
	6 BCD digits, 50K cps	1 pt	C500-CT012
PID			3G2A5-PID01-E
Position Control	1-axis, for stepping/servo motor		3G2A5-NC103-E
	1-axis, for stepping motor only		3G2A5-NC111-EV1
	2-axis, for servomotor		C500-NC221-E
	Stepping Motor Driver	Phase current: 0.5 to 2 A	3G2A5-SMD21
		Phase current: 0.6 to 4 A	3G2A5-SMD41
	Encoder Adapter		3G2A5-AE001
	Teaching Box		3G2A5-TU001-E
			C500-TU002-E
ASCII Unit	Adapter Box		3G2A5-IF101-E
	Power Supply		3G2A5-PS103-E
	RAM		3G2A5-ASC01
	RAM + EEP-ROM		3C2A5-ASC02

Link Units and Remote I/O Units

Name		Remarks		Model
Host Link	Rack-mounting	APF		3G2A5-LK101-PEV1
		PCF		3G2A5-LK101-EV1
		RS-232C/RS-422		3G2A5-LK201-EV1
		APF		C500-LK103-P
		PCF		C500-LK103
		RS-232C/RS-422		C500-LK203
	CPU-mounting	APF		3G2A6-LK101-PEV1
		PCF		3G2A6-LK101-EV1
		RS-232C		3G2A6-LK201-EV1
		RS-422		3G2A6-LK202-EV1
PC Link		Links up to 8 PCs		C500-LK009-V1
SYSNET Link		General-purpose		C500-SNT31-V2
Optical Remote I/O Master		APF		3G2A5-RM001-PEV1
		PCF		3G2A5-RM001-EV1
Optical Remote I/O Slave		APF	w/1 optical connector	3G2A5-RT001-PEV1
			w/2 optical connectors	3G2A5-RT002-PEV1
		PCF	w/1 optical connector	3G2A5-RT001-EV1
			w/2 optical connectors	3G2A5-RT002-EV1
Optical I/O Link		APF		3G2A5-LK010-PE
		PCF		3G2A5-LK010-E
Wired Remote I/O Master				C500-RM201
Wired Remote I/O Slave				C500-RT201
I/O Block		AC Input	Specify 100 VAC or 200 VAC	G71C-IA16
		DC Input	Specify 12 VDC or 24 VDC.	G71C-ID16
		Output	Specify 12 VDC or 24 VDC.	G71C-OC16

Link Units and Remote I/O Units

(continued)

Name	Remarks				Model
Optical Transmitting I/O	DC input	No-voltage contact, 100 VAC	8 pts	APF	3G5A2-ID001-PE
				PCF	3G5A2-ID001-E
	AC/DC input	12 to 24 VAC/DC 100 VAC	8 pts	APF	3G5A2-IM211-PE
				PCF	3G5A2-IM211-E
	AC input	100 VAC 100 VAC	8 pts	APF	3G5A2-IA121-PE
				PCI	3G5A2-IA121-E
		200 VAC 100 VAC	8 pts	APF	3G5A2-IA221-PE
				PCF	3G5A2-IA221-E
	Contact output	2 A 250 VAC 24 VDC 100/230 VAC	8 pts	APF	3G5A2-OC221-PE
				PCF	3G5A2-OC221-E
	Triac output	100/200 VAC 100/200 VAC	8 pts	APF	3G5A2-OA222-PE
				PCF	3G5A2-OA222-E
	Transistor output	0.3 A 12 to 48 VDC 100/200 VAC	8 pts	APF	3G5A2-OD411-PE
				PCF	3G5A2-OD411-E

Optical Fiber Cables

All Plastic Optical Fiber Cable (APF)

This cable can be used (in lengths of up to 20 m) with Units having the suffix "-P" in their model numbers. It cannot be used with Units without the suffix "-P". The connector (3G2A9-PF002) must be assembled by the user. When connecting Units of differing types (i.e. model numbers with and without the suffix "-P") always use plastic-clad optical fiber cable (PCF)

Name	Remarks	Model
Plastic optical fiber cable	Cable only (optical connectors not provided) Order in units of 5 m for cable less than 100 m. Else order in units of 200 m or 500 m.	3G5A2-PF002
Optical connector A	2 pcs (brown) for plastic optical fiber 10 m long max.	3G5A2-CO001
Optical connector B	2 pcs (black) for plastic optical fiber 8 to 20 m long	3G5A2-CO002
Plastic optical fiber cable	1 m, with optical connector A provided at both ends	3G5A2-PF101

Plastic-Clad Optical Fiber Cable (PCF)

This cable can be used (in lengths of up to 200 m) with Units having the suffix "-P" in their model numbers. It can also be used with Units without the suffix "-P" in lengths of up to 800 m.

Name	Remarks		Model
Optical fiber cable (for indoors)	0.1 m, w/connector	Ambient temperature; -10° to 70°C	3G5A2-OF001
	1 m, w/connector		3G5A2-OF101
	2 m, w/connector		3G5A2-OF201
	3 m, w/connector		3G5A2-OF301
	5 m, w/connector		3G5A2-OF501
	10 m, w/connector		3G5A2-OF111
	20 m, w/connector		3G5A2-OF211
	30 m, w/connector		3G5A2-OF311
	40 m, w/connector		3G5A2-OF411
	50 m, w/connector		3G5A2-OF511
Optical fiber cable (for indoors or outdoors)	1 to 500 m (Order in units of 1 m)	Ambient temperature; -10° to 70°C	3G5A2-OF002
	501 to 800 m; (Order in units of 1 m)	Ambient temperature; 0° to 55°C (Must not be subjected to direct sunlight)	

Note: The optical fiber cable is not easily connected. Cut the cable a little longer than required.

— length L (m) —
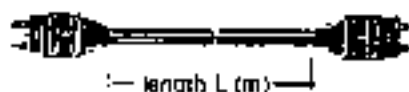
•Crystal Optical Fiber Cable

Crystal optical fiber cable is not available from Omron. Order it directly from the manufacturer.

SYSBUS

Name	Remarks	Model
Link Adaptor	RS-422, 3 pcs	3G2A9-AL001
	Optical (APF/PCF) 3 pcs	3G2A9-AL002-PE
	Optical (PCF) 3 pcs	3G2A9-AL002-E
	Optical (APF/PCF), RS-422, RS-232C, 1 pc each	3G2A9-AL004-PE
	Optical (PCF), RS-422, RS-232C, 1 pc each	3G2A9-AL004-E
	Optical (APF/PCF), Optical (AGF), 1 pc each	3G2A9-AL005-PE
	Optical (PCF), Optical (AGF), 1 pc each	3G2A9-AL005-E
	Optical (APF/PCF) 1 pc, Optical (AGF) 2 pcs	3G2A9-AL006-PE
	Optical (PCF) 1 pc, Optical (AGF) 2 pcs	3G2A9-AL006-E
Repeater	APF/PCF	3G5A2-RPT01-PE
	PCF	3G5A2-RPT01-E

Peripheral Devices

Name	Remarks		Model
Programming Console	Vertical		3G2A5-PRO23-E
Programming Console Connecting Cable	For connecting Programming Console, GPC, or FIT. (Only use CN221 (2 m) for Programming Console.)	2 m	3G2A2-CN221
		5 m	C500-CN523
		10 m	C500-CN131
		20 m	C500-CN231
		30 m	C500-CN331
		40 m	C500-CN431
		50 m	C500-CN531
Programming Console Adapter	For extending Programming Console. Connecting cable is separate.		3G2A5-AP001-E
Programming Console Base			3G2A5-BP001
Handheld Programming console			C200H-PRO27-E
Programming Console Adapter	Required for Handheld Programming Console		C500-AP003
Connecting Cable		2 m	C200H-CN222
		4 m	C200H-CN422
P-ROM Writer			3G2A5-PRW05-EV1
Printer Interface Unit	Memory pack is separate.		3G2A5-PRT01-E
Printer Connecting Cable	2 m, for connecting printer/plotter		SCY-CN201
Memory Pack (for printer interface)			3G2A5-MP102-EV1
Floppy Disk Interface unit			3G2C5-FDI03
Peripheral Interface unit	Connecting cable is separate		3G2A5-IP004-E
FIT	CPU and System Disk Set		FIT10-SET11-E
Graphic Programming Console	110 VAC		3G2C5-GPC01-E
	220 VAC		3G2C5-GPC02-E
	110 VAC		3G2C5-GPC03-E
	220 VAC		3G2C5-GPC04-E
GPC Memory Pack	w/comments		C500-MP303-EV1
CRT Interface Unit	For connecting GPC to CRT		3G2A5-GO101-E
Cassette recorder connecting cable	1m		SCYPOR-PLG01

Optional Products

Table	Name	Remarks	Model
B-4SC	Battery		3G2A5-BAT08
	I/O terminal block cover	For 38-pin block, special type	3G2A5-COV11
		For 38-pin block, standard type	3G2A5-COV12
		For 16-pin block, standard type	3G2A5-COV13
	Connector cover	For I/O connector protection	3G2A5-COV01
		For Link connector protection	3G2A5-COV02
		For IOC/IOIF connector protection	3G2A5-COV03
	Space Unit	For I/O Control Unit space	3G2A5-SP001
		For I/O Unit space	3G2A5-SP002

Commercially Available

Products

Name	Remarks		Model
Data recorder	100 VAC	PC-DR311	NEC
	100 VAC	MR 33DR	Sanyo
Printer	Interface board #8143 or #8145 is necessary.	SX 80T RX-80II FX-80 VX-80K VX-130K	Epson
		WX4731-1-01 FP5301R-1-01	Graphtec
X-Y plotter	Paper exchangeable	MP1000-1-01 MP2000-51	
Floppy disk drive	5-inch, double-sided, double-density, double-track (2DD)	PC9831-4W	NEC
	3.5-inch, double-sided, double-density, double-track (2DD)	PC9831-UW1 PC9831-UW2 PC9831-VW2	

Programming Devices

•Graphic Programming
Console (GPC)

The GPC allows you to perform all the functions of the Programming Console as well as many additional ones. PC programs can be written in ladder diagram as well as mnemonic form. As the program is written it is displayed on the liquid crystal display making modifications quick and easy. Syntax checks may be performed on the programs before they are downloaded to the PC. Many other functions are available depending on the Memory Pack used with the GPC

•FIT

FIT is an Omron computer with specially designed (FIT) software which will allow you to perform all of the functions that are available on the GPC. Moreover many of the functions that normally require the use of special peripheral units may be performed without them. With FIT, programs can be written (in ladder or mnemonic form), debugged, saved to floppy disks, or printed out.

•LSS

LSS is software that is designed for use with the IBM/AT (and some other computers) which will allow you to perform all the functions available on the FIT.

APPENDIX B
Programming Console Operations

Operation	Mode			Key Sequence
	Run	Mon.	Prog.	
Data All Clear	NO	NO	YES	
I/O Table Register	NO	NO	YES	
I/O Table Verify	YES	YES	YES	
I/O Table Read	YES	YES	YES	
I/O Table Transfer	NO	NO	YES	
Error Message Read	YES	YES	YES	
Setting Address	YES	YES	YES	

Operation	Mode			Key Sequence
	Run	Mon.	Prog.	
Program Read	YES	YES	YES	
Search	YES	YES	YES	
Instruction Insert	NO	NO	YES	
Instruction Delete	NO	NO	YES	
Program Check	NO	NO	YES	
Scan Time Read	YES	YES	NO	

Operation	Mode			Key Sequence
	Run	Mon.	Prog.	
General Monitor	YES	YES	YES	
Channel Monitor (Binary)	YES	YES	YES	
Forced Set/Reset	NO	YES	YES	
PV Change 1	NO	YES	YES	
PV Change 2	NO	YES	YES	

Operation	Mode			Key Sequence
	Run	Mon.	Prog.	
SV Change 1	NO	YES	NO	
Cassette Tape Write	NO	NO	YES	
Cassette Tape Read	NO	NO	YES	
Cassette Tape Verify	NO	NO	YES	

Instruction	Symbol	Mnemonic		Operand
Load	—⊣ ⊢—	LD	B	B: IR SR HR TR TC
Load Not	—⊣/⊢—	LD NOT	B	
And	—⊣ ⊢—	AND	B	
And Not	—⊣/⊢—	AND NOT	B	
Or	—⊣ ⊢—	OR	B	
Or Not	—⊣/⊢—	OR NOT	B	
And Load		AND LD	..	—
Or Load		OR LD		
Out	—(B)—	OUT	B	B: IR HR TR LR
Out Not	—(∅)—	OUT NOT	B	
Timer	—(TIM)—	TIM	N	N: TC
		—	SV	SV: IR SR HR LR #
Counter	CP CNT / R N SV	CNT	N	
		—	SV	

IR	SR	HR	LR	TC	DM	#
0000 to 6002	6003 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic	Function	Operand
00		NOP(00) —		
01	—[END(01)]	END(01) —	Ends the program	—
02	—[IL (02)]	IL(02) —	Causes program steps to be ignored and outputs cleared and timers reset depending on the result immediately before this instruction.	
03	—[ILC(03)]	ILC(03) —	Clears IL	
04/05	—[JMP(04)] —[JMP(05)] —[JMP(04)] —[JMP(05)]	JMP(04) — JME(05) — JMP(04) N JML(05) N	Causes all the program steps between this instruction and JMF to be ignored, or executed, according to the result immediately before this instruction.	N: 00 to 99
06	—[FAL(06)] —[FAL(06)]	FAL(06) N FAL(06) 00	Indicates an error that does not stop the CPU. Clears FAL area.	N: 01 to 99 —
07	—[FALS(07)]	FALS(07) N	Indicates an error that stops the CPU.	N: 01 to 99
10	I P——[SFT (10)] R	SFT(10) B — E	Shifts data in bit units. 15 0 15 0 E ≈ B —IN	B/E: IR HR LR
11	S:(KEEP B)	KEEP(11) B	Causes data bit to become latching.	B: IR HR LR
12	I DI——[CNTR N SV] RI	CNTR(12) N — SV	UP-DOWN (reversible) counter operation.	N: SV: 1C IR HR LR #

Code	Symbol	Mnemonic		Function	Operands	
13	DIFU(13) P	DIFU(13)	B	Causes the following instruction to operate for one scan time at the leading edge of the input signal.	B: IR HR LR	
14	DIFD(14) B	DIFD(14)	B	Causes the following instruction to operate for one scan time at the trailing edge of the input signal.	B: IR HR LR	
15	TI (TIMH)	TIMH(15)	N	High-speed, ON-delay timer operation. Set value: 0.01 to 99.99 s	N: TC	SV: IR SR HR LR #
			SV			
16	WSFT(16) B E	WSFT(16) — —	B E	Shifts data in Ch units.	B/E: IR HR LR DM *DM	
20	CMP(20) C1 C2	CMP(20) — —	C1 C2	Compares one channel's data, or a 4-digit constant, against another channel's data. Note: C1 and C2 cannot both be constants.	C1/C2: IR SR HR LR TC DM *DM #	
21	MOV(21) S D	MOV(21) — —	S D	Transfers channel data, or a 4-digit constant, to a specified channel.	S: IR SR HR LR TC DM *DM #	D: IR HR LR DM *DM
22	MVN(22) S D	MVN(22) — —	S D	Inverts channel data, or a 4-digit constant, and transfers it to a specified channel.		
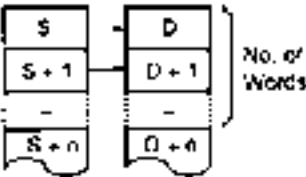
IR	SR	HR	LR	TC	DM	#
0000 to 6007	6000 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic		Function	Operands	
23	BIN(23) / S / R	BIN(23) / — / —	— / S / R	Converts BCD data into binary data. S (BCD) x10⁰, x10¹, x10², x10³ → R (BIN) x16⁰, x16¹, x16², x16³	S: IR SR HR LR TC DM *DM	R: IR HR LR DM *DM
24	BCD(24) / S / R	BCD(24) / — / —	— / S / R	Converts binary data into BCD data. S (BIN) x16⁰, x16¹, x16², x16³ → R (BCD) x10⁰, x10¹, x10², x10³	S: IR SR HR LR TC DM *DM	R: IR HR LR DM *DM
25	ASL(25) / Ch	ASL(25) / —	— / Ch	Shifts Ch data left.	Ch: IR HR LR DM *DM	
26	ASR(26) / Ch	ASR(26) / —	— / Ch	Shifts Ch data right.	Ch: IR HR LR DM *DM	
27	ROL(27) / Ch	ROL(27) / —	— / Ch	Rotates Ch left, with carry.	Ch: IR HR LR DM *DM	
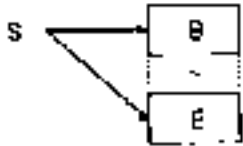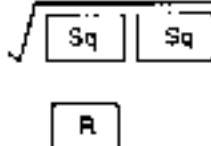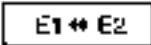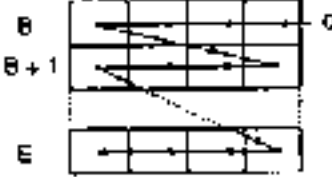
Code	Symbol	Mnemonic	Function	Operand
28	ROR(28) / Ch	ROR(28) — / — Ch		Ch: IR HR LR DM *DM
29	COM(29) / Ch	COM(29) — / — Ch	Inverts Ch data $\overline{Ch} \to Ch$	Ch: IR HR LR DM *DM
30	ADD(30) / Au / Ad / R	ADD(30) — / — Au / — Ad / — R	Performs BCD addition of one channel's data, or a 4-digit constant, and another channel's data. $Au + Ad + \boxed{CY} \to R \boxed{CY}$	Au/Ad: IR SR HR LR TC DM *DM # R: IR HR LR DM *DM
31	SUB(31) / Mi / Su / R	SUB(31) — / — Mi / — Su / — R	Performs BCD subtraction of one channel's data, or a 4-digit constant, and another channel's data. $Mi - Su - \boxed{CY} \to R \boxed{CY}$	Mi/Su: IR SR HR LR TC DM *DM # R: IR HR LR DM *DM
32	MUL(32) / Md / Mr / R	MUL(32) — / — Md / — Mr / — R	Performs BCD multiplication of one channel's data, or a 4-digit constant, and another channel's data. $Md \times Mr \to \boxed{R+1} \quad \boxed{R}$	Md/Mr: IR SR HR LR TC DM *DM # R: IR HR LR DM *DM

IR	SR	HR	LR	TC	DM	#
0000 to 5002	5003 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic		Function	Operands	
33	DIV(33) Dd Dr R	DIV(33) — — —	— Dd Dr R	Performs BCD division of one channel's data, or a 4-digit constant, and another channel's data. $\boxed{R+1}$ $\boxed{R}$	Dd/Dr: IR SR HR LR TC DM *DM #	R: IR HR LR DM *DM
34	ANDW(34) I1 I2 R	ANDW(34) — — —	— I1 I2 R	Performs a logical AND operation between two channel's data. I1 ∧ I2 → R	I1/I2: IR SR HR LR TC DM *DM #	R: IR HR LR DM *DM
35	ORW(35) I1 I2 R	ORW(35) — — —	— I1 I2 R	Performs a logical OR operation between two channel's data. I1 ∨ I2 → R	I1/I2: IR SR HR LR TC DM *DM #	R: IR HR LR DM *DM
36	XORW(36) I1 I2 R	XORW(36) — — —	— I1 I2 R	Performs a logical exclusive OR operation between two channel's data.	I1/I2: IR SR HR LR TC DM *DM #	R: IR HR LR DM *DM
37	XNRW(37) I1 I2 R	XNRW(37) — — —	— I1 I2 R	Performs a logical exclusive OR NOT operation between two channel's data.	I1/I2: IR SR HR LR TC DM *DM #	R: IR HR LR DM *DM
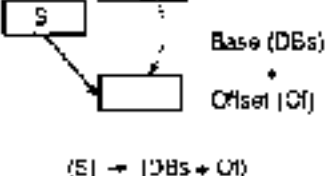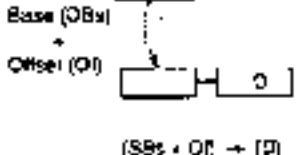
Code	Symbol	Mnemonic	Function	Operands
38	INC(38) / Ch	INC(38) — / — Ch	Increments a channel's data by 1.	Ch: IR HR LR DM *DM
39	DEC(39) / Ch	DEC(39) — / — Ch	Decrements a channel's data by 1.	Ch: IR HR LR DM *DM
40	STC(40)	STC(40) —	Sets the carry flag (Ch) to "1". 1→ Ch	
41	CLC(41)	CLC(41) —	Resets the carry flag (Ch) to "0". 0 → Ch	
70	XFER(70) / N / S / D	XFER(70) — / — N / — S / — D	Moves the contents of several consecutive source channels to several consecutive destination channels.	N: SR HR LR TC DM *DM # / S: IR SR HR LR TC DM *DM / D: IR HR LR TC DM *DM

For code 70, the function diagram shows: S → D, S+1 → D+1, ... S+n → D+n, No. of Words.

IR	SR	HR	LR	TC	DM	#
0000 to 6002	6003 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic	Function	Operands
71	BSET(71) / S / B / E	BSET(71) — / — S / — B / — E	Sets the same data to the specified consecutive channels S: Number of channels	S: IR SR HR LR TC DM *DM B/E: IR HR LR TC DM *DM
72	ROOT(72) / Sq / R	ROOT(72) — / — Sq / — R	Computes 8-digit BCD square root	Sq: IR HR LR TC DM *DM R: IR LR HR DM *DM
73	XCHG(73) / E1 / E2	XCHG(73) — / — E1 / — E2	Exchanges data between channels. E1 ↔ E2	E1/E2: IR HR LR DM *DM
74	SLD(74) / B / E	SLD(74) — / — B / — E	Shifts Ch data left in digit units (4 bits).	B/E: IR HR LR DM *DM

252

Code	Symbol	Mnemonic	Function	Operands
75	SRD(75) / S / L	SRD(75) — / — B / — E	Shifts Ch data right in digit units (4 bits).	B/E: IR HR LR TC DM *DM
76	MLPX(76) / S / DI / RB	MLPX(76) — / — S / — Di / — RB	Decodes 1 digit (4 bit) data into a bit position.	S: IR SR HR LR TC DM *DM Di/RB: IR HR LR TC DM *DM
77	DMPX(77) / SB / R / Di	DMPX(77) — / — SB / — R / — Di	Encodes the position of the highest bit that is ON into 1-digit (4-bit) data.	SB: IR SR HR LR TC DM *DM R/Di: IR HR LR TC DM *DM
78	SDEC(78) / S / Di / DB	SDEC(78) — / — S / — Di / — DB	Decodes 1-digit (4 bits) of Ch data into data for 7-segment display.	S: IR SR HR LR TC DM *DM Di/DB: IR HR LR TC DM *DM

IR	SR	HR	LR	TC	DM	#
0000 to 6002	6003 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic	Function	Operands
79	FDIV(79) Dd Dr R	FDIV(79) — — Dd Dr R	Performs floating point division between two 7-digit BCD data Dd + 1 Dd ÷ Dr + 1 Dr R + 1 R	Dd/Dr: IR HR LR TC DM *DM R: IR HR LR TC DM *DM
80	DIST(80) S DBs Of	DIST(80) — — S — DBs — Of	Transfers 16-bit data to a channel with address given by base plus offset. S → Base (DBs) + Offset (Of) (S) → (DBs + Of)	S: IR SR HR LR TC DM *DM # DBs/Of: IR HR LR TC DM *DM
81	COLL(81) SBs Of D	COLL(81) — — SBs — Of D	Extracts 16-bit data from a channel with address given by base plus offset and transfers the data to the specified channel. Base (DBs) + Offset (Of) → D (SBs + Of) → (D)	SBs: IR SR HR LR TC DM *DM # D/D: IR HR LR TC DM *DM

Code	Symbol	Mnemonic	Function	Operands
82	MOVB(82) S C D	MOVB(82) — — S — C — D	Transfers a specified bit in Ch S to a specified bit position in Ch D. S D	S: IR SR HR LR TC DM *DM # C: IR HR LR TC DM *DM # D: IR HR LR TC DM *DM
83	MOVD(83) S C D	MOVD(83) — — S — C — D	Transfers Ch data to the specified position in digit (4 bit) units. 15 0 S D	S: IR SR HR LR TC DM *DM # C: SR HR LR TC DM *DM # D: IR HR LR TC DM *DM
84	SFTR(84) C B E	SFTR(84) — — C — B — E	Shifts the specified data one bit to the left or right, with carry. 15 D2 0 15 D1 E B CY IN 15 D2 0 15 D1 E B CY	C/B/E: IR HR LR DM *DM #
85	TCMP(85) CD CB R	TCMP(85) — — CD — CB — R	Compares 16-bit data with a table consisting of 16-channel data CD → CB 0 CB+1 r : 0 CB+13 r CB+14 0 CB+15 1 1: agreement 0: disagreement	CD: IR SR HR LR TC DM *DM # CB: IR HR LR TC DM *DM # R: IR HR LR TC DM *DM

IR	SR	HR	LR	TC	DM	#
0000 to 6002	6003 to 6307	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

Code	Symbol	Mnemonic	Function	Operands	
87	WRIT(87) / N / S / D	WRIT(87) — / — N / — S / — D	Writes N Chs of data to an Intelligent I/O Unit through D.	N/S: IR SR HR LR TC DM *DM	D: I/O channel only
88	READ(88) / N / S / D	READ(88) — / — N / — S / — D	Reads N Chs of Data from an Intelligent I/O Unit through S.	N: IR HR LR TC DM *DM	S: I/O channel only D: IR HR LR TC DM *DM
90	SEND(90) / S / D / C	SEND(90) — / — S / — D / — C	Transfers data of C channels from Ch S to Ch D of node no. N and those that follow.	S: IR SR HR LR TC DM *DM	D/C: IR HR LR TC DM *DM
94	WDT(94)	WDT(94) N	Refreshes the watchdog timer the specified number of times.	N: 0 to 99	

Code	Symbol	Mnemonic	Function	Operands

97 — IORF(97) / B / E   IORF(97) — / — B / — E   Refreshes the specified I/O channels.   B/E: 00 to 31

98 — RECV(98) / S / D / C   RECV(98) — / — S / — D / — C   Transfers data of G channels from Ch S of node no. N to Ch D and those that follow.

Node No. | Source

S / S + 1 / S + n - 1

D / D + 1 / D + n - 1

Operands:
S: IR, SR, HR, LR, TC, DM, *DM
D/C: IR, HR, LR, TC, DM, *DM

IR	SR	HR	LR	TC	DM	#
0000 to 6002	6003 to 6107	0000 to 3115	0000 to 3115	000 to 127	000 to 511	0000 to 9999 or 0000 to FFFF

APPENDIX D
Inspection

Maintenance

Regular inspections and appropriate maintenance of the control devices are essential to ensure the full life of your PC, and the trouble-free operation of your controlled system. Safety measures to protect the system and to minimize system downtime in the event of a failure must also be taken

Inspection Items

If any of the following items are found to be outside the criteria shown in the tables, the necessary corrections should be made so that the criteria are met.

Power Supply

Supply voltage fluctuation	Power supply rated at 100 to 120VAC: 85 to 132VAC Power supply rated at 200 to 240VAC: 170 to 264VAC Power supply rated at 24 VDC: 20.4 to 26.4VDC (measured at power terminal block using voltmeter)
I/O supply voltage fluctuation	Must conform to I/O specifications
Battery	10.2 to 13.2VDC
Battery life	5 years (at 25°C)

Environmental Conditions

Ambient temperature in control panel	0°C to 55°C
Humidity	35 to 85% RH without condensation. Must be relatively dust-free

Mounting

All Racks and I/O Units firmly secured?	Mounting screws must not be loose.
Expansion I/O cables securely inserted in connectors?	Cables must not be loose.
Terminals for external wiring firmly secured?	Screws must not be loose
Any breakage in external wiring?	Must be free from visible abnormality.

Semiconductor elements are employed as main components of the PC. Because they are subject to deterioration under severe environmental conditions, these components should be inspected periodically. The standard inspection cycle is 6 months to 1 year.

When a Possibly-Defective Unit is Found –

Check first for a poor contact. Remove the Unit, and wipe the connector pins with a clean cotton cloth moistened with industrial alcohol. Make certain that there is no cloth debris remaining on the connector, and remount the Unit.

If the Unit Itself Is
Defective -

Turn the power off before replacing the Unit, and confirm the operation of the
new Unit

When returning a defective Unit to Omron, enclose a written description of the
problem.

Maintenance Tools
and Testers

Screwdrivers (phillips and standard)
VOM or digital voltmeter
Industrial alcohol and cotton cloth

Troubleshooting Tools
and Testers

Oscilloscope
Pen-recording oscilloscope
Thermometer
Hygrometer

Note: To ensure continuous operation in the event of failure, it is recom-
        mended to keep at least one spare I/O Unit on hand.

Index