Cat. No. W83-E1-5

OMRON

C20

Programmable Controller

SYSMAC
C-Series

USER'S MANUAL

This user's manual provides all the information you need to install, operate, and maintain the OMRON SYSMAC-C20 Programmable Controller, which can be used for a wide range of industrial applications.

The C20 is OMRON's response to the demand for a low-cost, versatile industrial control system that can be quickly installed and easily operated by technicians with little or no experience with programmable controllers.

Features

Improved cost effectiveness
All functions for a small-scale control system are integrated in a single housing. The C20 can be used in the place of conventional relay control panels and is ideally suited for economical control of manufacturing and other applications which require fewer input/output control points.

Slim profile for convenient mounting
The C20's compact design allows it to be installed in locations where space is at a premium. Hardly larger than a standard text book, the C20 can easily be mounted either horizontally or vertically near the machinery it operates.

Choice of memory chips
Either RAM or EPROM memory chips can be installed in the C20. RAM offers the convenience of a read/write memory for control programs that are under development or subject to modification; EPROM chips can be used for "permanent" storage for rarely changed programs or when multiple copies are needed for programs.

Optional optical-fiber communication
As with the other models of the SYSMAC C-Series programmable controllers, I/O linkage can be accomplished using a remote I/O master unit and optical-fiber cables. This state-of-the-art technology enables noise-free communications over long distances at ultra-high speeds.

Detachable terminal block
An easily removable terminal block greatly facilitates wiring the I/O lines and makes maintenance more convenient.

Upwardly compatible peripheral devices
The programming console, printer interface unit, and PROM writer can be shared with all programmable controllers in the SYSMAC C-Series. Moreover, with the addition of a peripheral interface, it is possible to program and operate the PC through a portable graphic programming console (either LCD or CRT) as well as a multisupport base unit.

Introduction

About this manual

This manual was designed, written, and illustrated to be highly useful to people at all levels of understanding and experience with programmable controllers, including the first-time user. It has also been organized and indexed to allow easy access to specific information.

The only special knowledge you need is an understanding of ladder diagrams using the familiar symbols adopted for conventional relay-based industrial systems.

Here's what you will find in the following chapters and appendixes:

Chapter 1
Programmable controllers in general and the C20 in particular are introduced. Each part of the C20 is illustrated and explained, and different system configurations are presented.

Chapter 2
What you need to know to install the C20 is explained, including important environmental, wiring, mounting, and power supply considerations.

Chapter 3
You're introduced to programming and the eight steps used to create a PC control program. This chapter focuses on the first three — assessing the controlled operation, assigning input/output points, and writing relay ladder diagrams.

Chapters 4 and 5
These two chapters cover using the programming console to write your application program and put it to work controlling your equipment or process through the C20.

Chapter 6
Examples are given which show you in simplified form some of the numerous control applications that can be realized by the C20.

Appendixes
The many peripheral devices available to expand the C20 system and provide valuable support services are introduced. Also covered are the basic and special instructions used in programming, as well as product specifications and other detailed technical information. A comprehensive index is included for easy reference to this manual.

Contents

OMRON

Chapter 1 Introduction to the C20

Chapter 2 Installing the C20

Chapter 3 Introduction to programming

OMRON

Contents

Chapter 4 Using the programming console

Chapter 5 Checking and running your program

Chapter 6 Application examples

Contents

Chapter 1 Introduction to the C20

Overview

This introductory chapter explains why programmable controllers have become such a valuable part of modern-factories. The SYSMAC-C20 is described, including its basic components and operating principles.

PC Basics

Programmable controllers, or PCs, evolved as industries sought economical ways to automate their production lines, particularly those involved in the manufacturing of equipment and other heavy industry products. The PCs took the place of relay-based control systems which were comparatively slower, less reliable, and which presented formidable wiring and maintenance requirements.

PCs operate by monitoring input signals from such sources as pushbuttons, sensors, and limit switches. When changes are detected in the signals, the controller system reacts, through user-programmed internal logic, to produce output signals. These signals operate the external loads of the controlled system, such as relays, motor controls, indicator lights, and alarms.

This type of control system eliminates much of the wiring and rewiring that was necessary with the conventional relay-based system. Instead, the programmed logic provides the "wiring network" which can be changed as required by simply reprogramming the PC. Thus the automated processes of a production line can be controlled and modified at will, for highly economical adaptability to a changing manufacturing environment.

A typical programmable controller has three basic components—an input/output section, a central processing unit (CPU), and a programming device.

Input/output section

This section consists of wiring and interfacing relays that connect the PC to the equipment being controlled.

Central processing unit

The CPU contains the control circuitry as well as the memory that stores the control plan that guides equipment operation. It is the heart of the PC and organizes all controller activity by scanning the control plan along with the status of the inputs and executes specified commands to specified outputs.

Programming device
This device is used to enter the control plan into the CPU's memory. An operator keys in the instructions used to sequentially control the application process. There are several programming methods; in the case of the SYSMAC C-Series, relay ladder diagram programming is used because plant personnel are familiar with that format and it provides for convenient record-keeping.

About the SYSMAC-C20

The C20 comes in two versions, a basic unit and an expandable unit.

The capabilities of the basic unit include up to 1194 program statements, 28 I/O points, and 136 internal auxiliary relays. The expandable unit is functionally identical to the basic unit except that it can be expanded using expansion I/O units to include up to 140 I/O points.

The C20 offers the flexibility of either RAM- or EPROM-based operation. When a RAM chip is installed as the memory, programs written by the user can be modified and rewritten. EPROM provides a semi-permanent storage for completed programs.

The programming console can be detached and is upwardly compatible with the full line of SYSMAC C-Series programmable controllers.

C20 Components

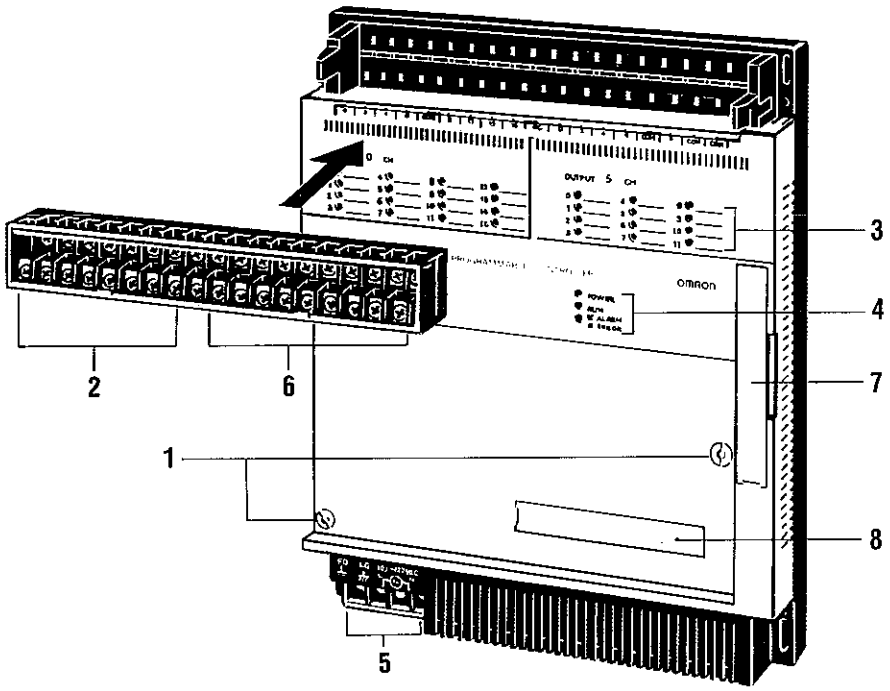
Due to its compact design, the C20 incorporates the detachable I/O terminals and microprocessor functions in a single housing called the CPU. The detachable programming console functions as the programming device. Additionally, various optional peripheral devices are available to support system expansion.

The two main components are explained and illustrated in the next several pages.

CPU

Central processing unit
The CPU is the main enclosure of the C20 incorporating the microprocessor and I/O terminals. The controlling program written using a programming device is stored and executed in the CPU.

CPU



1 Console screws
These stud screws are removed with a flat-bladed screwdriver to open the cover for initial setting and maintenance purposes.

2 Input terminals
These terminals are used to connect the wiring carrying input signals from the devices of the controlled system.

3 Input/output indicators
These LEDs show the state of the signals (by turning red when ON and dark when OFF) for each of the input and output terminals.

4 LED indicators (POWER, RUN, ERROR/ALARM)
These LEDs light up to show the state of the main power supply (ON/OFF), when the C20 is in the RUN mode, and if any error has occurred.

5 Power terminals
These terminals connect the AC or DC power source and are grounded at a resistance of less than 100Ω.

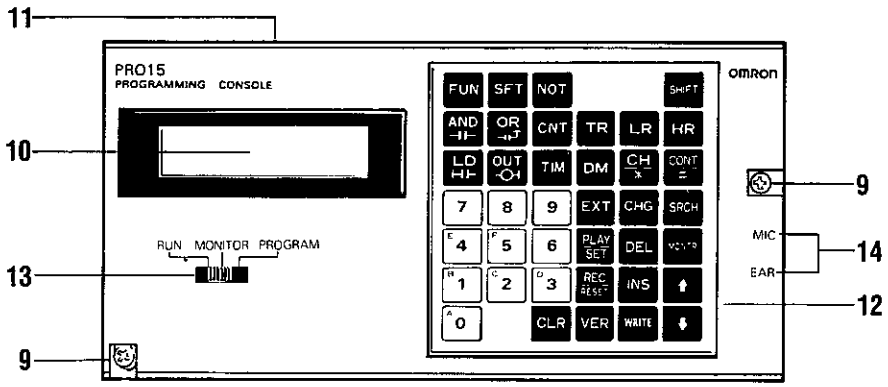
6 Output terminals
These terminals are used to connect the wiring carrying output signals from the CPU to the output devices.

7 Connector for expansion I/O units
This is available only on the expandable type CPU and is used to connect to expansion I/O units to increase the number of I/O points.

8 Connector for peripheral devices
This connects the programming console or another peripheral device.

Programming console

This is the standard programming device used with the C20. The control programs written by the programming console are stored and run in the CPU.



9 Mounting screws

These two screws secure the detachable programming console to the CPU.

10 LCD

This displays the program as it is being written and is used for checking and monitoring program operation. It also displays error messages.

11 Contrast control

The contrast control adjusts contrast level of display.

12 Keypad

The color-coded keypad is functionally divided into the following areas:

White	10 keys used to input program addresses, timing values, and other types of numeric entry
Red	One key used to clear the display
Yellow	12 keys used to provide editing functions while writing and correcting the control program
Gray	16 keys used to input instruction words used in the program

The function of each key is detailed in Chapter 4.

13 Mode selector switch

This three-position switch selects one of three operation modes of the PC: program, monitor, and run. These modes are explained in detail in Chapters 4, and 5.

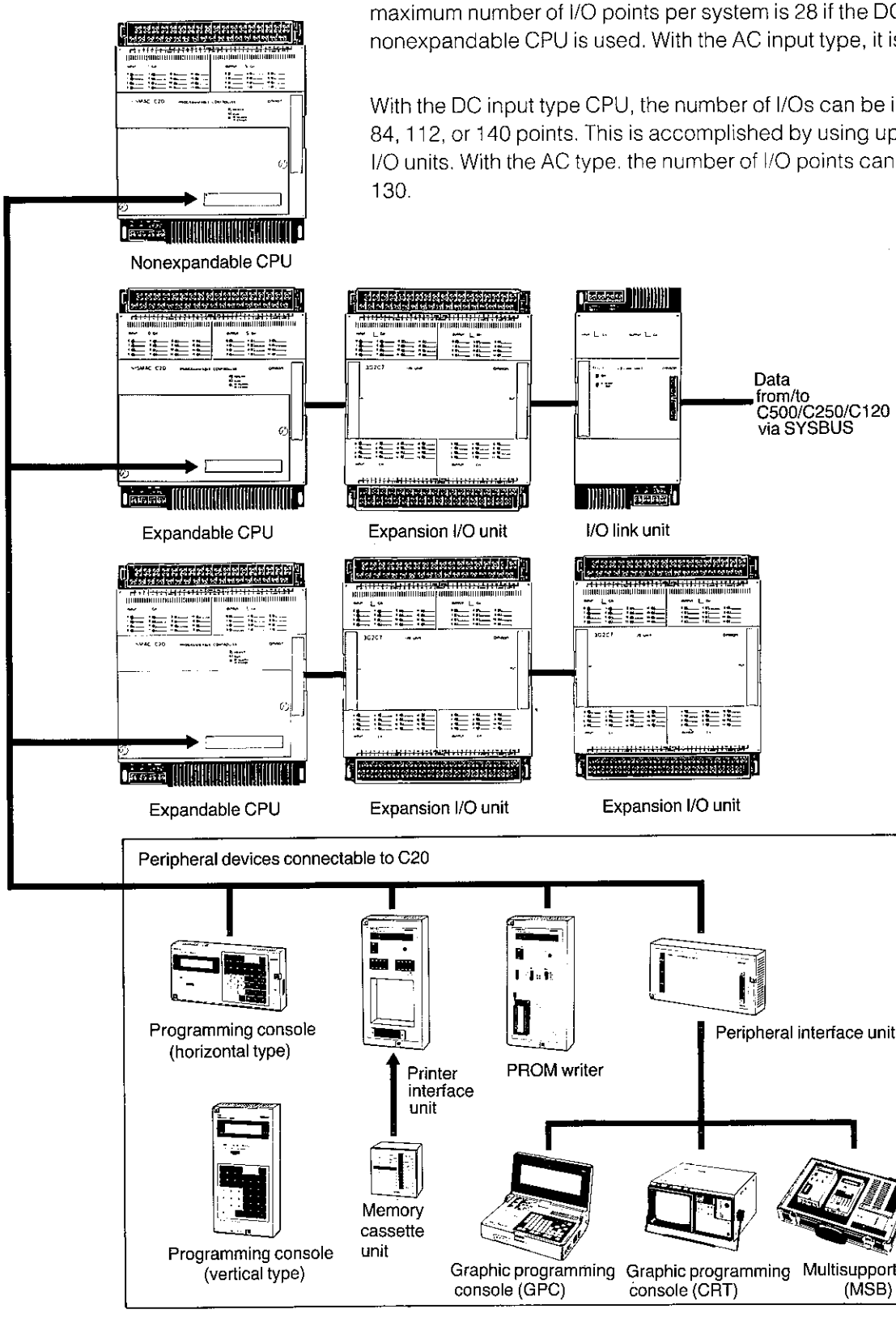
14 Jacks for connecting cassette tape recorder

Programs may be saved to a standard cassette tape recorder connected to the output (MIC) jack. Previously written programs can also be supplied to the CPU via the input (EAR) jack. These operations are explained in Chapter 5.

Basic system configuration

The figure below shows some of control systems employing the C20. The maximum number of I/O points per system is 28 if the DC input type nonexpandable CPU is used. With the AC input type, it is 26.

With the DC input type CPU, the number of I/Os can be increased to 56, 84, 112, or 140 points. This is accomplished by using up to two expansion I/O units. With the AC type, the number of I/O points can be 52, 78, 104, or 130.



Note: Although there are several additional peripheral devices available for other PCs in the SYSMAC C-Series, only those shown here can be used with the C20.

Chapter 2 Installing the C20

Overview

This chapter tells you how to install the C20. You may wish to skip this section and go first to Chapter 3, Introduction to programming, which provides important information on how to determine the configuration of your controlling system.

But first...

The following are some important points that should always be observed to ensure correct installation and operation of the C20.

Static electricity

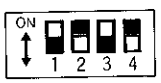
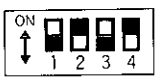
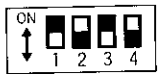
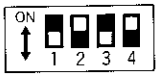
In low-humidity conditions, static electricity may cause problems with the C20's components. Therefore, before touching the C20, be sure to first touch a grounded metallic object to discharge any static electricity buildup.

Memory chips

The C20 offers a choice of using either a RAM or EPROM chip for the program memory, according to your application.

With the nonexpandable CPU, there are still 170 program addresses available as standard even when no RAM chip is mounted and by mounting a RAM chip, the number of addresses can be increased to 512. With the expandable CPU, the standard memory capacity is 1194 addresses.

The following table provides the necessary information as to the type of chip that can be used in the expandable and nonexpandable type CPUs as well as the DIP switch settings required for each type of chip.

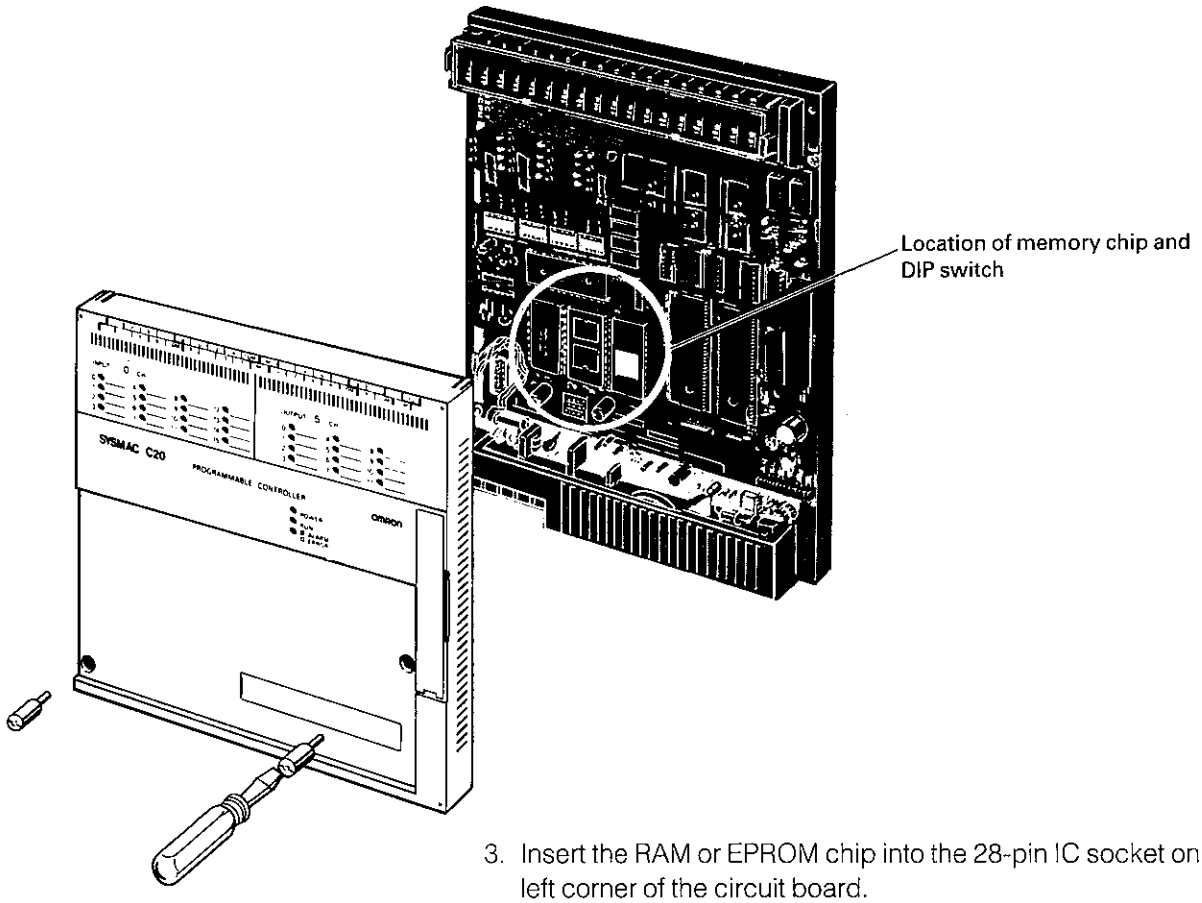
Nonexpandable CPU		Expandable CPU	
RAM	Use Type RAM-G	RAM	RAM not required
Memory capacity	512 addresses	Memory capacity	1,194 addresses
DIP switch setting		DIP switch setting	
			
EPROM	Use Type ROM-H	EPROM	Use Type ROM-H
Memory capacity	1,194 addresses	Memory capacity	1,194 addresses
DIP switch setting		DIP switch setting	
			

Installing memory chips

The procedure for installing memory chips is described below:

1. Use the information given in the table above to confirm that the type of chip you have is correct.
2. Remove the front panel of the CPU by loosening the mouting screws with a flat-bladed screwdriver. You'll be able to see the CPU's circuit board.

Caution: Before opening the CPU, always be sure that the power supply to the C20 has been turned OFF.



3. Insert the RAM or EPROM chip into the 28-pin IC socket on the lower left corner of the circuit board.
4. Set the DIP switch to the correct setting for the type memory chip you have installed. For this, refer to the table on the previous page.

Testing the environment

When you're ready to install the C20, first give attention to the environmental conditions under which it will normally operate. The PC is ruggedized for reliable use under tough conditions, but you still need to avoid using it in these areas:

Where the ambient temperature is below 0° or above 55°C.

Where abrupt temperature changes may cause condensation.

Where relative humidity is below 35% or above 85%.

Where corrosive or flammable gas may occur.

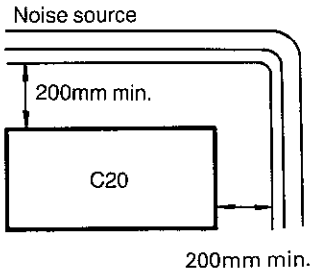
In extremely dusty conditions, or where subject to salt or iron particles.

Where subject to abnormal vibration or shock.

In direct sunlight.

Where it can be splattered with water, oil, or chemical substances.

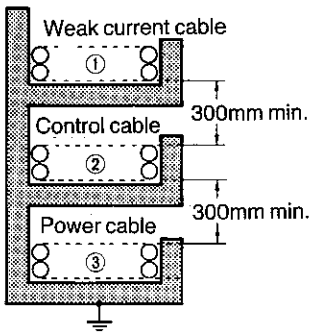
Dealing with problems of noise



One of the most important considerations when installing and wiring the C20 is the problem of noise. The C20 should be mounted at least 200mm from high-tension equipment or wiring.

Control panel wiring

Wherever possible, always use wiring ducts to contain and protect the wiring for the I/O units. I/O wires should not be placed in the same duct with a power line or other wiring.



Grounding (at a ground resistance of less than 100Ω)

① SYSMAC I/O circuit
② SYSMAC power circuit
General control circuit
③ Power circuit

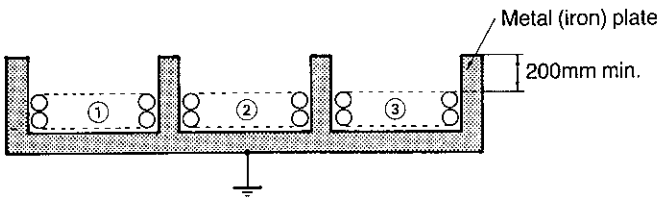
External wiring

Use standard wiring conduits for routing I/O and power lines to the C20. However, be sure to use separate conduits for the I/O wiring.

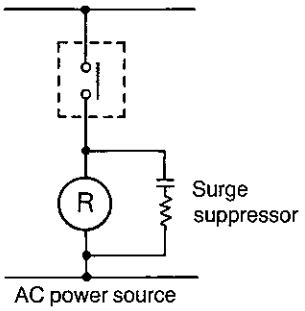
The following points are important when the controlled system requires the laying of 400V 10A max. or 220V 20A max. power lines.

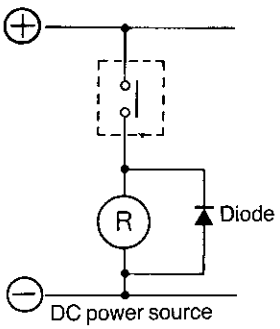
A minimum distance of 300mm must be provided between the I/O lines and power cables when their conduits are run parallel to each other.

If the cables must be placed in the same duct at the point of connection to the equipment, be sure to screen them with a grounded metal plate.



Grounding (at a ground resistance of less than 100Ω)





Noise-suppression measures

Be sure to take appropriate noise-suppression measures when any electrical device likely to produce noise is employed as a PC output load.

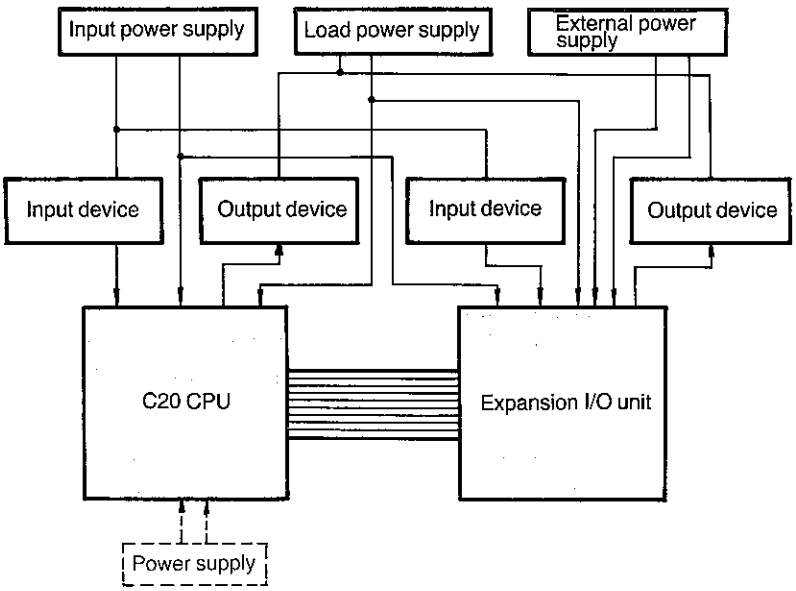
For example, electromagnetic relays and valves generating noise of more than 1,200V require noise suppression.

For AC-supplied noise sources, connect a surge suppressor in parallel with the coil of each device. For DC-operated noise sources, connect a diode in parallel with the coil of each device.

When mounting a CPU and I/O expansion unit in a control panel, be sure to completely ground the intermediate mounting plate. The mounting plate must be finished with high-conductivity plating to insure noise immunity.

Power supply

Where possible, use independent power sources for the input, the load, and for the CPU and I/O units.

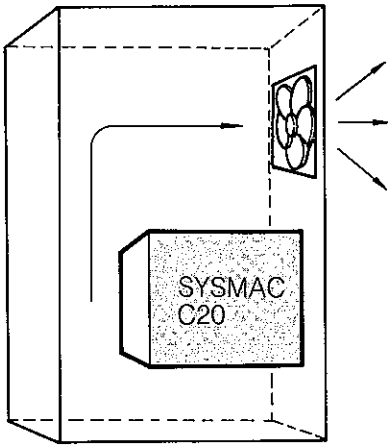


DC 24V/AC 100 to 120V/AC 200 to 240V

Grounding

Terminal LG is a noise filter neutral terminal which does not normally require grounding. When electrical noise is a problem, however, this terminal should be short circuited with terminal FG.

Control panel mounting



Some of the main considerations when mounting the PC in a control panel are accessibility for operation and maintenance and protection against heat.

These are some of the things you should consider:

Use M4×25 mounting screws.

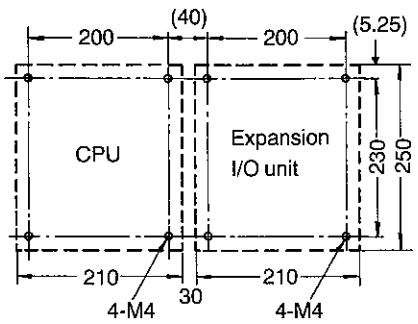
Provide the PC with adequate space for ventilation.

Avoid mounting the controller directly above any heat-generating source, such as a heater, transformer, or high-capacity resistor.

If the ambient temperature is above 55°C within the panel, you must install a fan for forced-air ventilation.

Avoid mounting the PC in a panel in which high-tension wiring or equipment is also installed.

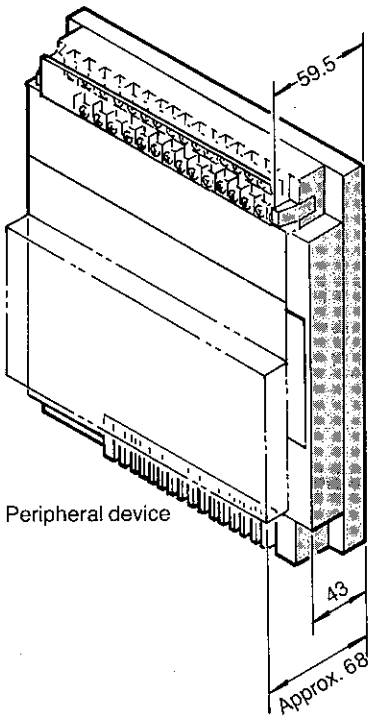
Use wire with a cross-sectional area of at least 2mm^2(AGW14) to prevent possible voltage drop. Use twisted pair cables for the wiring.



Mounting dimensions

When the CPU and I/O unit are mounted together in the control panel, a distance of about 30mm should be maintained between the two units. This means that the holes for the CPU mounting screws and those for the I/O unit mounting screws should be separated by about 40mm.

SYSMAC C20



Although the CPU is actually 59.5mm thick (measured at the I/O terminal block), be sure to leave enough room in front of the unit to allow for installation of optional peripheral devices, such as the programming console.

When the programming console is attached to the PC, the total depth increases to 68mm.

Power supply

Use a commercially available DC 24V, AC 100 to 120V or AC 200 to 240V power supply for the C20.

As the DC 24V power supply, Type 3G2A3-PS221 is available. Here are the electrical specifications.

Characteristics \ Type	Power supply unit
	3G2A3-PS221
Input voltage	AC 100/110/120V (AC 85 to 132V)/AC 200/220/240V (AC 170 to 264V), 50/60Hz±7Hz (AC 200V is the factory-set condition)
Output voltage	24V±5%
Output capacity	1.5A max. (36W)
Efficiency	70% min.
Power failure	Retains 10ms max.
Inrush current	5A max.
Fuse	2A, incorporated
Leakage current	1mA max. between FG terminal and ground
Weight	560g

Where possible, use independent power sources for the inputs, the load, and for the CPU and expansion I/O units.

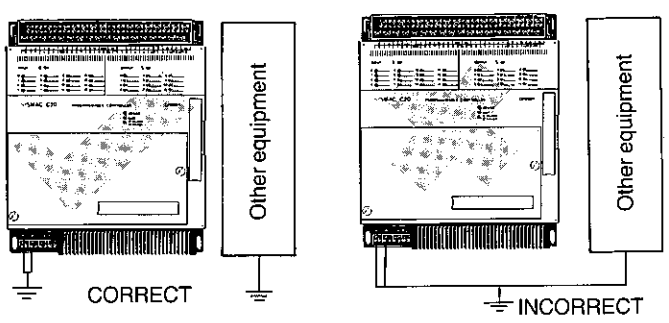
Grounding

Attach a dedicated grounding wire with a cross-sectional area of at least 2mm^2(AWG14) to terminal FG to lessen the possibility of electrical shock. Ground resistance must be less than 100Ω.

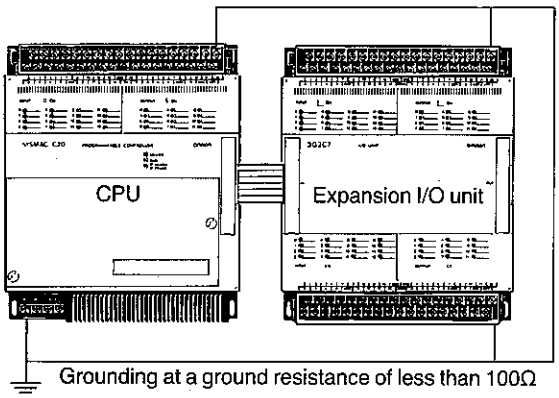
Do not use a grounding wire longer than 20m. Care must be taken because ground resistance is affected by the nature of the ground, water content, season, and the amount of time that has elapsed since the wire was laid underground.

In case a large noise occurs causing the C20 to malfunction, short-circuit terminals LG and FG for grounding at a resistance of 100Ω.

PC operation may be adversely affected if the grounding wire is shared with other equipment or if grounding is attempted by attaching the grounding terminal to the metal superstructure of a building.



When the I/O expansion unit is used with the CPU, the FG terminals on both devices are used for grounding.



Battery

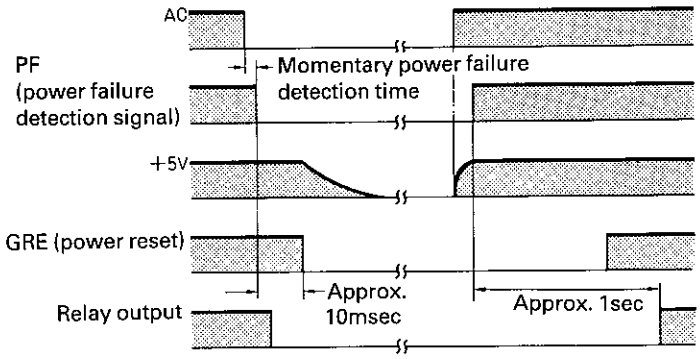
The CPU has a built-in battery to back up the memory in case of power failure. This backup battery is required regardless of whether a RAM or EPROM chip is used as the program memory. The battery's service life is about five years when the PC is used at a temperature of 25°C, and is somewhat shortened in temperatures higher than this or where humidity is unusually high. These factors should also be kept in mind when storing the PC.

When the battery is almost completely discharged, the ALARM/ERROR indicator on the CPU blinks. You then have one week to change the battery before memory loss occurs. A fresh battery must be installed within 3 minutes after the old one has been removed from its connector.

Protections against power failure

A power sequence circuit is incorporated in the PC to prevent malfunctioning because of a momentary power failure or a drop in the supply voltage.

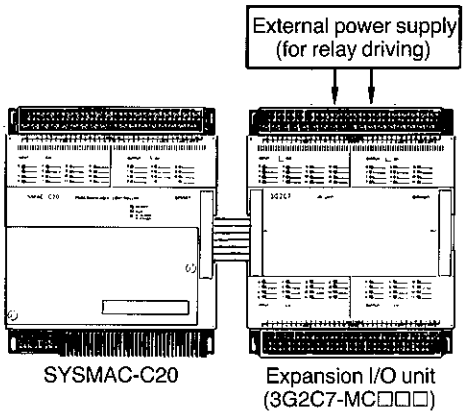
If the voltage drops below 85%, the PC stops operating and the external output relays are automatically turned off.



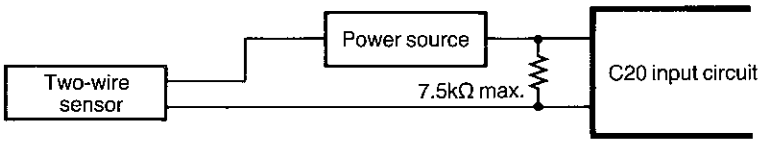
The PC ignores all momentary power failures when the interruption lasts no longer than 10msec. If the interruption is more than 10msec, but less than 20msec, it may or may not be detected. Failures longer than 20msec will cause the PC to stop operating; external output relays are automatically turned off.

Operation automatically resumes when the supply voltage is restored to more than 85% of the rated voltage.

When the expansion I/O unit (3G2C7-MC□□□) is used, an external power supply is required to drive the relays. This external power supply is not necessary when only the CPU is used.



When a two-wire sensor is used for the input, the input signal may be turned ON by a leakage current exceeding 3mA. To prevent this, connect a bleeder resistor of 7.5kΩ or less as shown.



Chapter 3 Introduction to programming

Overview

This chapter introduces the eight steps you'll take in creating the program the PC uses to carry out its control operations. You can skip this chapter if you're already familiar with PC programming using relay ladder diagrams. Instead, you may wish to go directly to Chapters 4 and 5 which focus on how the C20 programming console is used to install the program in the memory of the CPU.

Relay ladder diagram method

If you're an old hand at relay-based control systems, you'll find a lot that's familiar in the way a 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.

But even if the actual devices are gone, the symbols and other control concepts used to describe their operation still are used. These are the basis of the relay ladder diagram programming method. This chapter has been written on the assumption that you are already experienced with this relay symbology, or have access to information that can teach you about it.

Eight basic programming steps

In creating the control program for the PC, you'll follow these eight basic steps:

1. Determine what the controlled system must do and in what order.
2. Assign input and output devices, that is, designate the external devices that will send signals to and receive signals from the PC.
3. Draw a diagram, using relay ladder symbols. This represents, in the correct sequence, all of the required functions and their relationship.
4. Code the ladder symbols into a form that can be input to the CPU by the programming console.
5. Transfer these written instructions to the CPU via the programming console.
6. Edit the program.
7. Test the program for errors.
8. Save the completed program.

Steps 1 through 4 are the focus of this chapter.

Step 1
Assessing the controlled operation

This, of course, is a highly important part of setting up a PC-controlled system. The PC's flexibility allows a wide latitude in not just *what* operations can be controlled, but in *how* they will be controlled.

To apply a PC to any control task, the system requirements must first be determined. The major part of that assessment focuses on the input/output requirements.

Input/output requirements

The first thing that must be assessed is the number of I/Os 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 a PC output signal.

These should be totaled and it should be remembered that the C20, when not expanded, has 16 input and 12 output points.

In terms of voltages, all the signals to the C20 must be 24 V DC inputs. Because the programmable controller has three independent output blocks, three output voltages are available through the common terminals: 24 V DC and up to 250 V AC max.

Sequence, timing, and relationship assessment

Next you need to determine the sequence and timing at which these controls will occur. How each of the controlled devices relates to the others should be identified (such as a photoelectric switch to a motor), as well as what response should occur between them.

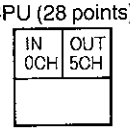
For instance, a photoelectric switch might be functionally tied to a motor by way of a counter within the PC. The motor would start when the PC receives an input from a switch and would be stopped by the signal output by the PC when the counter had received five input signals from the photoelectric switch.

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

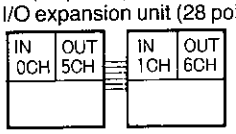
Having made this assessment, you now will be ready to go to Step 2 of programming—assigning the input/output devices or points.

Step 2
Assigning I/Os

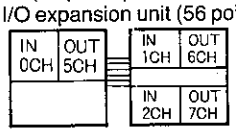
CPU (28 points)*



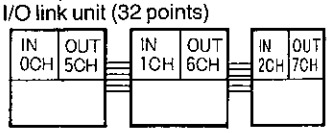
CPU (28 points)*
+ I/O expansion unit (28 points)*



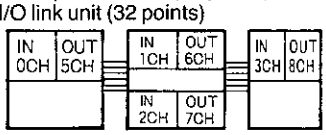
CPU (28 points)*
+ I/O expansion unit (56 points)*



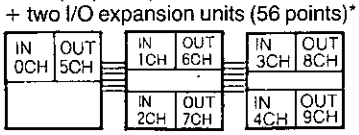
CPU (28 points)*
+ I/O expansion unit (28 points)*
+ I/O link unit (32 points)



CPU (28 points)*
+ I/O expansion unit (56 points)*
+ I/O link unit (32 points)



CPU (28 points)*
+ two I/O expansion units (56 points)*



*When the system is configured of AC input type units, the number of I/O points per system is decreased as follows:
28-point system → 26 points
56-point system → 52 points

Points and channels

The programmable controllers of the SYSMAC C-Series all use the concept of I/O channels to identify individual I/O terminals, or points. Each of these channels consists of 16 points.

The four-digit number used to identify an I/O point therefore can be broken down into the left-hand two digits, which identify the channel, and the right-hand two digits, which identify the point within the channel.

For example, "0000" identifies the first point of the first channel and "0104" identifies the fifth point of the second channel. In the C20, the first five channels (00 to 04) are used for input and the next five (05 to 09) for output.

The actual number of points used by the basic C20 is 16 (00 to 15) inputs and 12(00 to 11) outputs.

Remember that the C20 can be expanded by adding expansion I/O units. Up to four new channels can be added for a total of 80 (5×16) input and 60 (5×12) output points.

I/O channel assignment

The assignment of I/O channels is as shown on the left.

Note: The total number of I/O points, including those of an I/O link unit, cannot exceed 140.

Assigning internal auxiliary relays

In the output channel, the four points (12 to 15) that are not used to send output signals directly to output devices function as internal relays. If these internal relays are to be used, they must be assigned as well during Step 2. How they function is explained next.

These relays do not control external devices directly. Instead, they are used as data memory or data process areas in controlling other relays, timers and counters. Functionally, these internal auxiliary relays are equivalent to the internal relays used in relay control panel. This is known as "internal output."

Assigning numbers to timers and counters

The C20 can accommodate up to 48 timers or 48 counters, or combinations of timers and counters not exceeding 48. How these are used is explained in Chapter 4. They must also be assigned identifying numbers, in a range from 00 to 47. (Note that these are not input/output points, but rather a way for you to identify the timer or counter you want to use.)

There are two basic considerations when assigning counter and timer numbers.

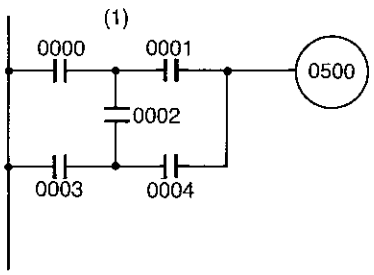
Do not give counters and timers the same number. For instance, there cannot be a Timer 01 and a Counter 01.

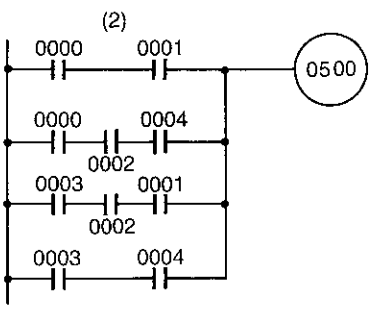
When you're finished assigning the input and output points, internal auxiliary relays, and timers/counters, you're ready to proceed to the next step.

Step 3 Writing a relay ladder diagram

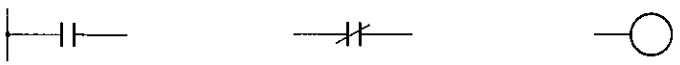
So far we know three basic things about our operation. We know which devices are to be controlled, we know how they relate to each other, and we know the sequence (or timing) at which the controlled tasks must take place.

Now we need to put this down in a form that is codeable—that is, in the form of a relay ladder diagram.





For this you'll use the four-digit numbers you just assigned to the input and output relays and internal auxiliary relays, as well as the two-digit numbers you gave to the timers and counters. You'll also use such relay symbols as the following. The complete list of symbols is shown in Appendix B.



Normally open contact | Normally closed contact | Output

In writing the ladder diagram, keep these things in mind.

Each logic line starting from the left bus bar must end with a relay coil, a timer/counter, or a special instruction. Unlike the actual circuit diagram, the right bus bar need not be written into the ladder diagram.

The number of contacts in series or parallel is not limited for use on a logic line within the programming capacity of the PC. Therefore, you can use as many contacts as you wish. If this feature of the C20 is effectively used, even a complicated circuit can frequently be replaced with a simpler one.

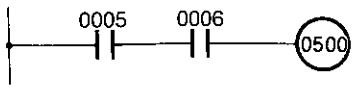
The bridge circuit (1) shown on the left, for example, could be replaced with the one shown below it (2).

Step 4
Turning the ladder diagram into PC code

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

This language consists of addresses, instructions, and data. The addresses are locations in the memory where instructions and data are stored. The instructions are used to tell the PC what to do using the data that follows each instruction.

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



Here's the ladder diagram for this.

To code this, we first need to specify the memory address from which this program starts. In this example we'll use 0000 as the starting address. To this address, an instruction that indicates the beginning of the program must be written. For the C20, the LD (LOAD) instruction serves this purpose. In our ladder diagram, the bus bar represents the LD instruction. Thus this instruction always is used any time the logic line starts from the bus bar.

Address	Instruction	Data
0000	LD	0005
0001	AND	0006
0002	OUT	0500
0003	END	

On our coding sheet, next to address 0000 we write the instruction "LD".

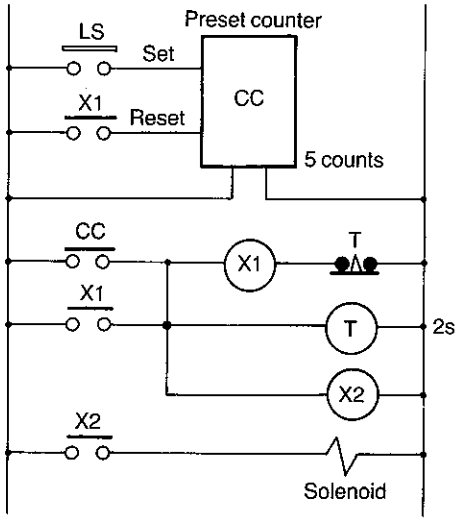
Because the first contact in the AND circuit must be stored as the data of the LD instruction, we write this down in the "data" column on our sheet. In our example, this data is 0005.

The next element of the ladder diagram is the AND instruction, which we assign to address 0001 in the program, as shown. The data for the AND instruction is the number assigned to the second contact, in this case 0006. On our sheet we write this next to the AND instruction.

Next we need an OUT (OUTPUT) instruction to output the result of the ANDed contacts in our circuit. We write this instruction in address 0002, and designate the output relay number to which we want this signal sent. We've chosen this relay to be 0500 and have written that as the next entry on the sheet.

We're finished, except that we always have to tell the PC when a program is over. So we assign an END statement to address 0003.

Sample program for coding practice



Now let's try a slightly more complicated program that shows how to relate Steps 1–4 to an actual application.

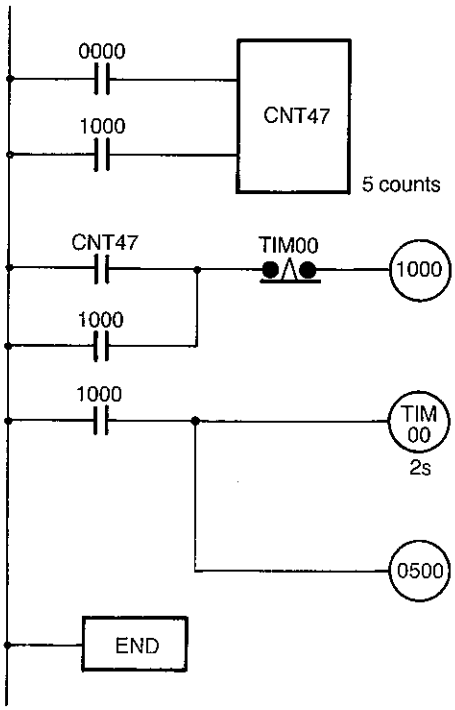
In the figure is a control circuit for a packaging machine. The circuit is used to detect and count the number of products being carried on an assembly line. When it counts five products, the circuit energizes a solenoid. The solenoid receives electrical current for a period of two seconds and is then shut off.

Using Step 1 we determine that there is an input device (a limit switch), an output device (a solenoid), a control relay (an internal auxiliary relay), a timer, and a counter.

To carry out Step 2, we have to assign numbers to the components constituting this circuit.

This is how our assignment looks:

Input device (limit switch)	:	Input relay No. 0000
Output to solenoid	:	Output relay No. 0500
Control relay (X1)	:	Internal auxiliary relay No. 1000
Counter CC	:	CNT47
Timer T	:	TIM00



The ladder diagram that represents this circuit using the numbers we assigned looks like the one at left.

Next we'll encode the ladder diagram as required for Step 3. These are the key things that require attention:

Be sure to use the assigned input number.

Since the timers and counters must be assigned numbers from a range of 0 to 47, a convenient way to make this assignment is to begin at one end of the range for timers and the other end for counters. That's why we gave Timer T the number 00, while Counter CC was assigned the number 47. This helps to prevent possible use of the same number for both a timer and a counter.

Timers and counters cannot directly produce an output but must be output by means of an output relay.

This relay contact can be reused as often as desired, but the coil cannot be reassigned.

The assigned output relay plays the role of the load in this example.

The right bus bar does not need to be written.

Each logic line must be ended with a relay coil, timer/counter, or special instruction.

Coding must be done from left to right and from top to bottom.

Based on those considerations, this is the coding chart we came up with:

Address	Instruction	Data
0000	LD	0000
0001	LD	1000
0002	CNT	47
		#0005
0003	LD	CNT47
0004	OR	1000
0005	AND.NOT	TIM00
0006	OUT	1000
0007	LD	1000
0008	TIM	00
		#0020
0009	OUT	0500
0010	END(FUN01)	

This completes the first four steps. In the next chapter we'll see how to actually operate the PC based on the programming code we have written.

Chapter 4 Using the programming console

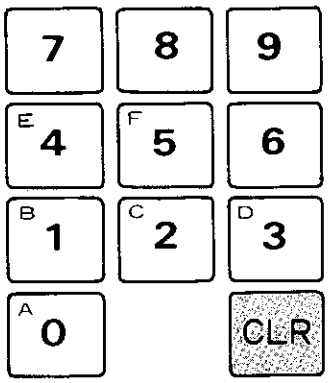
Overview

In the last chapter you learned how to perform Steps 1–4 of the eight basic programming steps. This chapter focuses on Step 5—using the programming console to enter the program code in the CPU.

The functions of the programming console are presented step-by-step as we enter the program created at the end of Chapter 3. The last part of this chapter explains how the programming console's editing functions work to make writing and modifying programs a lot more convenient.

The keyboard

In an earlier chapter the functions of the color-coded keyboard were briefly mentioned. Now let's take a closer look at each of these keys.

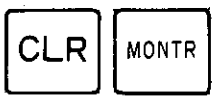


Numeric keys
These are the white keys numbered 0 to 9.

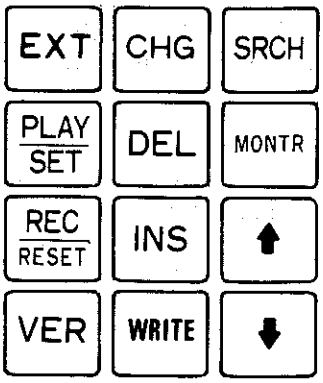
These keys are used to input numeric values used for program data. For instance, in our program in the last chapter, these keys would be used to input the input/output numbers and timer/counter numbers and values.

These keys are also used in combination with the function key (FUN) for special instructions. These instructions are explained in Appendix B.

CLR key
This red key is used to clear the display. It is also a key you'll use while keying in the "password," which is used to foil unauthorized access to the PC's program. Rather than using an actual password, though, you gain access using this two-keystroke entry:



When you do this, on the console display you'll see written either PROGRAM, MONITOR, or RUN. Pressing the CLR key again makes the word disappear and prepares the PC for the operation you have selected with the three-position mode switch.



Operation keys

These yellow keys are the ones you'll use to carry out the editing functions of the programming console. These functions will be explained in more detail later but at this point it's important that you know how three in particular are used.

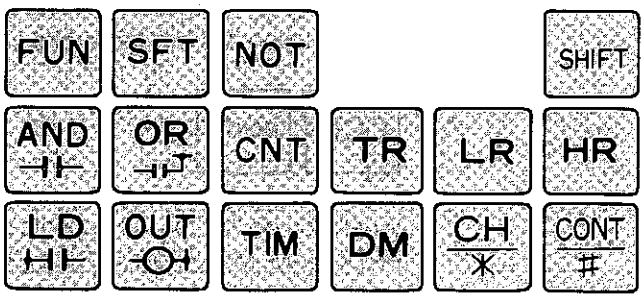
The first two are the arrow keys. When you want to move through your program a step at a time, you press the bottom (down) arrow key. The displayed address of the program will increment once for each press. To go the opposite direction, press the other (up) key. The program will decrement one step at a time until it reaches its beginning.

The arrow keys are normally used for moving only a small number of steps in the program. Later we'll show you several ways to move right to the program step you want.

The third key in the yellow group you should know about now is the WRITE key. During programming when you have written an instruction and its data, use this key to register the instruction in the PC memory at the address desired.

Instruction keys

Except for the SHIFT key on the upper right, these gray keys are the ones you'll use to place instructions in 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.

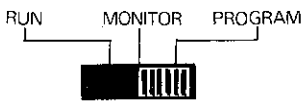


Each of the remaining gray keys has its function indicated by an abbreviation. These are what those abbreviations mean:

FUN	Selects a special function. Used to key in special instructions. These special instructions are realized by pressing FUN and then the appropriate numerical values. The special instructions are listed in Appendix B.
SFT	Enters SHIFT REGISTER instruction.
NOT	Forms NC contact.
AND	Enters AND instruction used for ANDing two contacts.
OR	Enters OR instruction used for ORing two contacts.
CNT	Enters counter instruction. Must be followed by counter data.
LD	Enters LOAD instruction used for loading a specified input.
OUT	Enters OUTPUT instruction for outputting to a specified output point.
TIM	Enters timer instruction. Must be followed by timer data.
TR	Enters temporary memory relay instruction.
LR*	Enters link relay instruction.
HR	Enters holding relay instruction.
DM*	Enters data memory instruction.
CH*	Specifies a channel.
CONT	Used to search for a contact.

* Although these functions are not available on the C20, these keys are provided to ensure programming console compatibility with other SYSMAC C-Series PCs.

Mode switch



Now let's turn to the three-position mode switch you will use to select the operating mode of the PC. As you see, there are three modes.

The RUN mode is the one you will use to begin PC operation. When you turn on this mode, the PC begins controlling the equipment using the program you have written into the PC memory.

The MONITOR mode allows you to visually monitor the operation in progress. For instance, if you want to check that a particular relay is in the correct state (either ON or OFF) at the proper time, you can move to the address (or step) that references that relay.

Using the PC in the RUN and MONITOR modes is explained in Chapter 5.

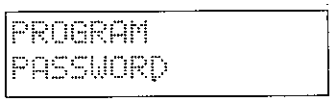
The PROGRAM mode, which is fully explained in this chapter, is used during the programming operation.

Console display

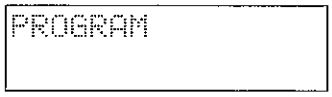
The easy-to-read display is your window into the workings of the PC. The display format changes depending on the mode that you have selected. Let's first examine the PROGRAM mode.

PROGRAM mode

If the mode selector switch is turned to PROGRAM, this is what you'll see when you apply power to the PC.



This is your electronic sentry that blocks unauthorized use of the PC. To gain access, an actual password isn't necessary—just this key sequence:



The PC responds with a beep each time you depress a key. This is what you now will see, informing you of the mode you are in.

To clear this, press

At this point you are ready to begin entering your program step-by-step using the code you wrote from your ladder diagram in Chapter 3. The display you see is the starting address of the memory. This location is used in connection with the quick-search editing functions.

Let's skip this for now because we won't need to use it until after we have input our program.

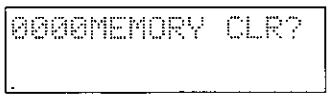
Although it is possible to overwrite an old program with a new one, this is not recommended because it may cause confusion and lead to program-writing errors.

If the programming console has been used before and an old program currently exists in the memory, carry out the procedure given next.

To erase existing memory

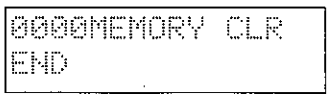
Caution: The following procedure will entirely and permanently erase any program that currently exists in the CPU memory.

Anytime you want to erase the memory or start over when inputting your program use this key sequence:



At this point, the display will be the one shown at left to allow you to reconsider. If you then press

MONTR



you have finished the memory clearing operation and the display will change to this.

Now if you press

CLR

you can begin writing your program.

Getting started, summarized

Putting all these steps together, here's a summary on what to do to begin programming the CPU:

1. Confirm whether a RAM chip is mounted and the DIP switches are set correctly.
2. Apply power to PC
3. Turn mode selector switch to PROGRAM
4. Do these key sequences:

CLR MONTR CLR

5. If you want to completely clear the old memory, press

PLAY/SET NOT REC/RESET MONTR CLR

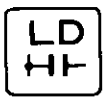
Example program

For practice, let's key in the program code we previously wrote. This is the coding chart from Chapter 3.

Address	Instruction	Data
0000	LD	0000
0001	LD	1000
0002	CNT	47
		#0005
0003	LD	CNT47
0004	OR	1000
0005	AND.NOT	TIM00
0006	OUT	1000
0007	LD	1000
0008	TIM	00
		#0020
0009	OUT	0500
0010	END(FUN01)	

Step 5
Entering the program in the CPU

The first information to be keyed in is the LD instruction. To do this, press



and you see this display.

```
0000
LD              0000
```

The first set of four 0s at left is the beginning address position, and is where the LD instruction will be stored. The second set of four 0s at right is a numeric value representing the input point. Currently the input point is 0000. Since this is the value you assigned to the input of Counter 47, you can leave it unchanged. Then, to WRITE this LD instruction to memory address 0000, press

and this display appears.

```
0001READ
NOP (00)
```

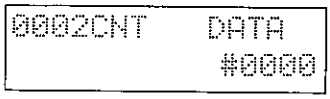
Here, 0001 is the address number; READ means you are reading the program; and NOP (00) means that no operation has yet been assigned to this address.

Next, enter

again to specify the reset input of the counter. READ disappears because you are now *writing* to address 0001 instead of only *reading* its contents. Now key in the the relay number and write it to the address with

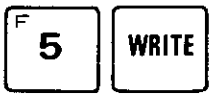
In the next address we want to input the counter. This is done by pressing

Now we have to specify the coil number of the counter we want, which is 47. Do this by keying in



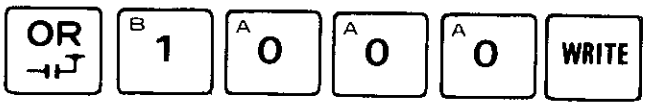
This second counter display at left

indicates the set value of the counter. Currently it is 0000. We want to change this to 5, the number of signals that must be input by the optical sensor. So key in



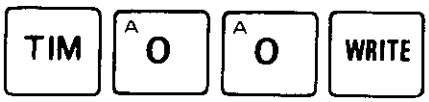
Now we have to enter the LD instruction to specify the contact of the counter that corresponds to the coil of Counter 47. Do this by pressing

For address 0004, key in



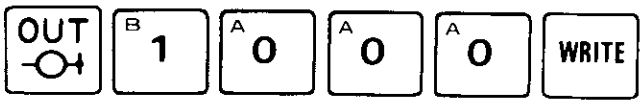
to enter the OR instruction and specify the internal auxiliary relay number. So that the timer acts to turn on the solenoid only 2 seconds, for address 0005 key in

This causes the contact of the timer instruction to act as an NC (normally closed) contact. Key in

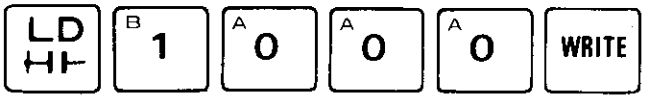


Now we have finished entering the timer contact. The timer coil for this contact must exist somewhere in the program for the timer instruction to be executed.

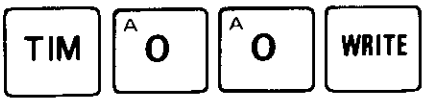
For address 0006 to designate the relay coil, key in



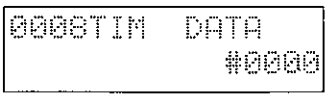
For address 0007, key in the contact corresponding to the internal auxiliary relay coil we have just entered.



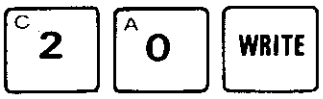
Remember that the coil and the time value for the contact keyed in for address 0005 must still be specified. This is done by keying in



for address 0008.



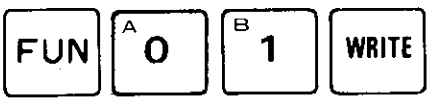
This message is displayed asking for the timer data. Since the data includes a decimal point before the last digit, to set a value of 2 seconds, we enter



The only thing that remains is to complete the circuit by specifying the output. Do this by entering

The output relay number for this output is 0500, which is automatically selected by the PC. (Remember that 05xx is the output channel designation, while 00xx is the input channel designation. The relay number is xx.)

The final instruction in our program is the END instruction, which tells the PC that the program is complete. Writing this requires the use of the FUN key and the value 01 which represents the END instruction. So key in



and you're finished.

Now check your program

To check whether the program has been correctly written, go back through it using the arrow keys to scan what you've done. If you see you need to make corrections, just overwrite the statements that are in error.

Deletion and insertion of instructions is explained next.

Deleting instructions

Let's return for a moment to Timer 00. For practice, let's delete it from the program using the following procedure.

First, go to address 0008. Then key in

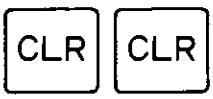
This eliminates the TIM instruction and moves the next instruction (LD, in this case) into the 0008 address location.

This function is useful when you wish to modify or correct an existing program.

Inserting instructions

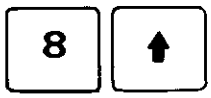
Inserting an instruction somewhere in a program is almost as easy. Let's assume that we now realize it was a mistake to eliminate Timer 00 from the program. To put it back into the program, we follow this six-step insert procedure.

1. First key in

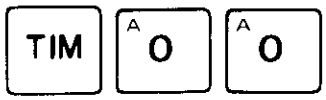


This takes you to the first address of the memory.

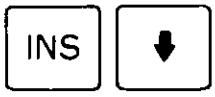
2. Then key in the number of the address where you wish to make the insertion.



3. Press

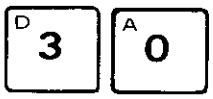


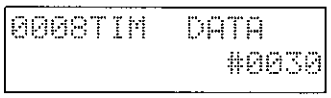
4. And then



to move the program down one step, reserving address 0008 for the TIM instruction.

5. Key in the TIM data



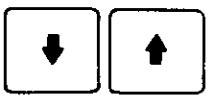


6. Then press

to complete the insertion.

Note: Steps 5 and 6 are necessary only to input the timer and counter data. These are not required for insertion of other types of instructions.

You can check that the insertion has been correctly done by moving the program using the arrow keys.



Spend some time becoming acquainted with these editing functions by inserting and deleting various instructions. In the next chapter you'll learn how to put these to use in debugging much larger programs.

Chapter 5 Checking and running your program

Overview

The C20 has a number of helpful features to help you debug your program and fine-tune the operation before and after it is under way.

This chapter covers the final three steps for programming. In Step 6 we explain how the quick-search editing functions make locating instructions quick and easy. Also explained is how to correct the program using the C20's debugging capabilities.

Once you're ready to begin a test run of the program, you'll learn in Step 7 how you can monitor the operation and make any necessary modifications based on the actual performance of the equipment.

Finally, Step 8 shows you how to use a standard cassette tape recorder to save and load your program.

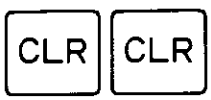
Step 6
Editing the program

Quick-search editing functions

The C20 programming console has three features that greatly facilitate program editing and debugging. One allows you to go directly to specific program addresses. Another allows you to hunt for specific instructions. The last lets you speedily check each contact. These editing features eliminate the need for you to travel through the program a single step at time—quite a tedious process when a large number of addresses are involved.

Going directly to a known address

This is a simple way of going straight to the address you want. Do this by first clearing the display by pressing



Then key in the number of the address you want.

Let's assume that we want to change the timer value from 2.0 seconds to 3.0 seconds. We first key in its address

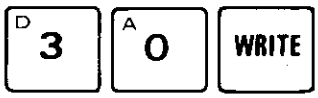
and follow that with

When you do this you will see that the display has gone directly to address 0008. Before you can change the timer value, however, you must first go to the timer's data by pressing

to move the program down one step. This is the display.

```
0008TIM  DATA
           #0020
```

At this point you can make the change to 3.0 seconds by keying in



Check the display. It should now look like this.

```
0008TIM  DATA
           #0030
```

When correcting or modifying a program, this method of address locating is particularly useful to go quickly to a new part of the program. Note, though, that you must always start at the first address in memory which is reached by pressing

```
0000
```

one or more times.

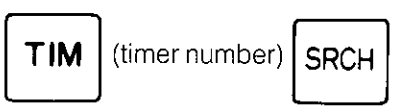
Searching for a specific instruction

This second editing technique gives you a convenient way to locate a particular instruction for which you do not have to know the program address. Assume, for example, that you wished to go to an unknown address containing Timer 00. To do this, press

several times until you see the first address.

```
0000
```

Then key in

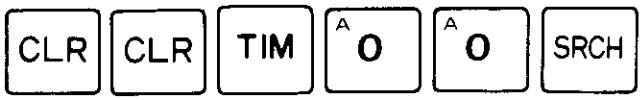


(timer number)

This takes you to address 0008 which contains Timer 00.

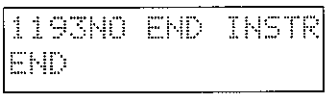
This editing feature has a wide range of uses. For instance, in a long program you may wish to determine whether you have inadvertently used the same timer or counter number more than once. This normally would be an error, since a particular timer or counter should be used only once in a program.

Let's imagine for a moment that our program is actually quite large and that there's a possibility that Timer 00 has been duplicated somewhere. We would search for the first reference to this timer by using the approach we just learned. Do this by keying in



which moves us again to address 0008. To search for any illegal timers, simply press

again, which tells the PC to scan the rest of the program for any other mention of a times with the number 00. As can see, there are no others in our program. (To test this further, go to any address, say 55, write in TIM00, and then repeat the search procedure.)



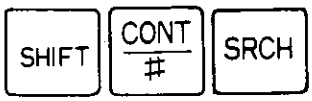
NO END INSTRUCTION message
In doing this you may have reached this display message.

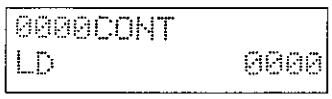
The number 1193 here represents the total number of address locations available. Normally when you see this message, it means that you forgot to put an END instruction at the end of your program.

This is one time when you can ignore the error message, though. The search function has carried you into the unused region of the memory (address locations 0011 to 1193). No new END statement is actually required here.

Searching relay contacts

The quick-search operation can be performed still another way. If you wish to locate specific contacts, this can be accomplished easily in any of the three modes with this simple operation. Start from the first address of memory. Then press



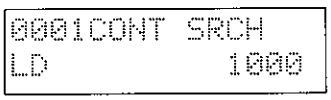


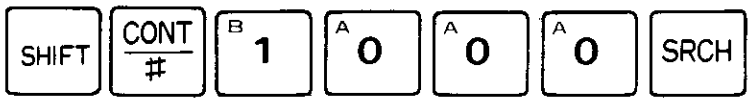
If you still have our sample program in the CPU memory, this should be the display. Then, to go to the next contact with the same number, again press

The display now changes to the next address containing an instruction with the same contact number (the four digits at the lower right of the display). This continues until the end of the program is reached.

If you want to search for a contact having a different contact number, follow the same steps as before. But this time, enter the numeric value of the contact you want to locate.

Using our sample program, if you enter 1000 and press the search key, the display will show the LD instruction in address 0001. This is how you would do it:



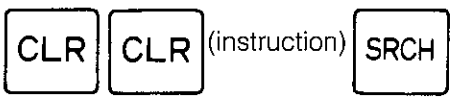


Search function summary

The C20 gives you three convenient ways to search through a long program for specific addresses, instructions, or contacts. The key entries for these are summarized here:

Address search

Specific instruction search



Specific contact search

Step 7
Testing for errors

The C20's debugging features can be used to catch many types of programming errors. For this you use the FUN and MONTR keys in any of the three modes.

To see if the program you entered in the PC has a programming error, press


```
0000ERR CHK
OK
```

This is the display if no error is found.

If an error is found in the program, the corresponding error message number will be displayed. Refer to Appendix E, Maintenance and troubleshooting, and carry out the corrective action listed there.

If more than one error exists, continue pressing

to display each error message one at a time.

There are basically two levels of errors—fatal and non-fatal. A fatal error, such as a memory error, prevents the PC from operating. A non-fatal error, such as a battery failure, allows the operation to proceed but still must be corrected. In case both types of errors have occurred, the error messages for the fatal error take precedence over the other and must be corrected first.

Status check

At times you may want to scan the status (whether ON or OFF) of each relay contact before you start the control operation. This can be an effective troubleshooting technique and is available in both the RUN and MONITOR modes.

You would normally perform the status check after you have carried out the debugging step above. To begin, go to the first address and press

You'll see the address, the instruction, and the word OFF or ON displayed in the upper right hand corner of the LCD. OFF or ON indicates the current state of the relay contact. Continue pressing the DOWN arrow key to check the status of each of the relays that follow.

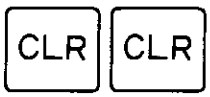
Forced set/reset

During the execution of a program, this operation is used to force set or reset (for one scan time) the operating status of each I/O relay, internal auxiliary relay, holding relay, timer, or counter. This operation is only meaningful while the PC is in the MONITOR mode.

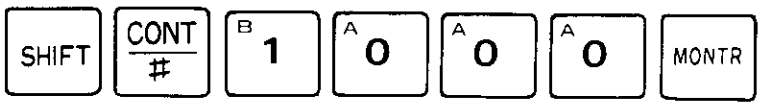
The most common use of this function is during a trial run of the controlled system. For instance, if a particular task (such as illuminating a heat lamp for drying) would normally take 30 minutes but you simply wish to test the contact, use this function to force reset the contact after a few moments of lamp operation. The program could then continue to the next task in the control sequence without delay.

Forced relay set

To do a forced set of a relay contact, first place the PC in the MONITOR mode and press



Now specify the contact that you wish to force set. In this case, Relay 1000 is to be forced set. Press



The display shows the present status of the relay which in this case is OFF

Now, to force set the relay (to ON), press

This force sets the relay from OFF to ON.

Forced relay reset

To force reset a relay to the OFF state, press

This turns the currently ON relay to OFF.

Forced timer set

To force set a timer, put the PC in MONITOR mode and press

TIM MONTR

The display shows the first timer, Timer 00, and its preset time if the timer is not currently running. Then, to clear the set time for one scan time, press

PLAY/SET MONTR

Due to the short cycle period, however, you will not be able to see this clearing actually take place. If the timer is currently operating, depressing

MONTR

displays the current value of the timer. If the timer has completely gone through its timing period, you will see this value.

Depressing

REC/RESET

restarts the timing operation from the set time.

Note: Special auxiliary relays 1808 to 1907 cannot be forcibly set or reset. These relays serve as flags that are internally "raised" and "lowered" (set and reset) to enable monitoring of PC operation. If an attempt is made to set or reset one of these relays, you'll hear a beep and no key input will be accepted by the programming console.

Changing set value of timer or counter

It's also possible to change the set value of a timer/counter during program execution. This is used for such purposes as slowing down or speeding up an assembly line operation without having to stop the program.

This operation can only be done in MONITOR mode. To reset the value of a timer, first key in

CLR CLR TIM (timer number) SRCH

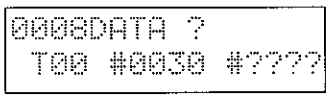
In our sample program, the timer number is 00.

Now press

to move to the timer data display.

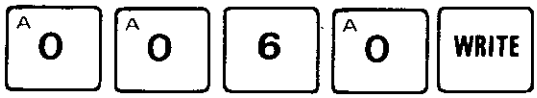
The timer currently is set for 3.0 seconds. To change this to 6.0 seconds, press

and the PC display will query you about the new data.



0008DATA ? informs you that the operation to be performed is the setting of a value for the instruction address 0008; T00 is the number of the timer; #0030 is the timer's present set time; and #???? asks what you want the new set time to be.

In response, key in your new set time and write it to the PC memory.



Follow the same procedure to change the set value of a counter.

Input/output monitor

While the PC automatically controls the equipment, you can keep a watchful eye over the operation using the PC's monitoring capabilities.

You may, for instance, have a particular unit of equipment that you want to check for correct operation. Or you may want to keep a running check on the value of a down counter.

The C20 allows you to easily check the status of any device connected to it by this procedure:

Place the PC in the MONITOR mode during operation.

Locate the device you wish to check. We could do this for Counter 47 in our practice program, for instance, by keying in

CLR CLR CNT E 4 7 SRCH

which gives this display.

```
0002SRCH    OFF
CNT          47
```

Press

MONTR

and the display changes to this.

```
C47
0005
```

If the PC were currently operating, we would be able to watch the counter as it is decremented. The same thing is also possible with timers.

Rapid check of counter/timer values

For a fast way to visually scan the set value of all timers and counters in a program, go to an address where a timer or counter is located. Then press

MONTR

and the set value of the timer/counter is displayed as a four-digit number.

```
 T00
0000
```

Then, each time you press

↓ or ↑

you will be able to see the set time of the next timer/counter.

Rapid check of relay status

You can check the status of relay contacts using the same technique. To do this, go to the address of a relay you want to check. Then press

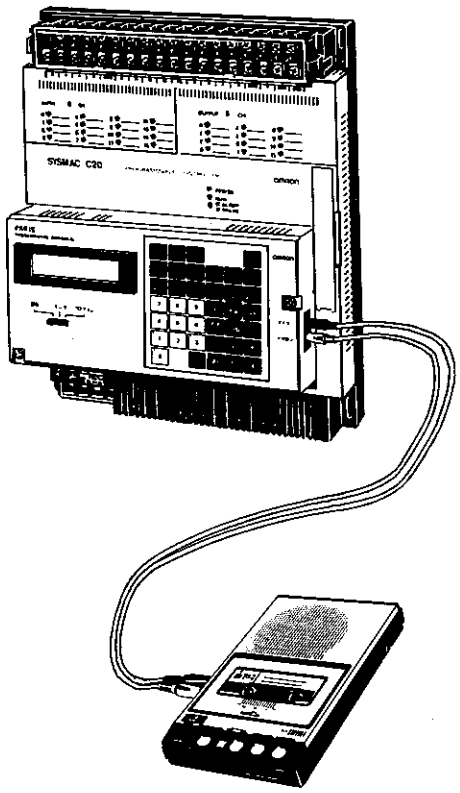
MONTR

to display the current status (whether ON or OFF) of the relay. Then, to move to the next relay, press either

```
1000
 OFF
```

↓ or ↑

Step 8
Saving your program to cassette tape



At this point it's assumed that you have completed program debugging and the trial run of the equipment and are satisfied with the operation. Now it's time to save the program to a cassette tape.

For this, you can use any reliable cassette tape recorder.

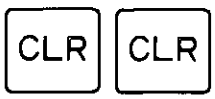
This provides a way to save your programs for later use. The same tape can also be used to program other C20s that control identical operations. Multiple copies of the tape can be made using a conventional tape copier, at either normal or high speed.

Store only one program to a tape (or side of a tape). The reason for this is that there is no way to identify individual programs if more than one have been stored on the same side of a cassette. The only requirement is that the tape be at least 7 minutes long. Either a standard or microcassette tape can be used.

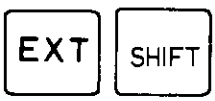
First you'll learn how to save the program you have in the PC's memory. Then you'll see how to load it back into the PC and verify that it was loaded correctly.

To save the program, carry out this procedure:

1. Plug one end of a Type SCY-PLG01 cable into the MIC jack of the PC and the other into the MIC jack of the tape recorder.*
2. Plug a second cable into the EAR jacks of both devices.**
3. Turn the volume and tone controls of the tape recorder to their maximum levels.
4. Switch the PC to the PROGRAM mode.
5. Press



6. Press



A message is displayed to ask you to start the save operation.

* On some tape recorders this may correspond to the LINE-IN jack.
** On some tape recorders this may correspond to the LINE-OUT jack.

7. Press the button or buttons on the tape recorder that begin the recording.
8. Then within 5 seconds press

REC
RESET

on the programming console keyboard. A blinking rectangle appears in the right corner of the display. This means the program is being saved to the tape.

9. Wait about 7 minutes.

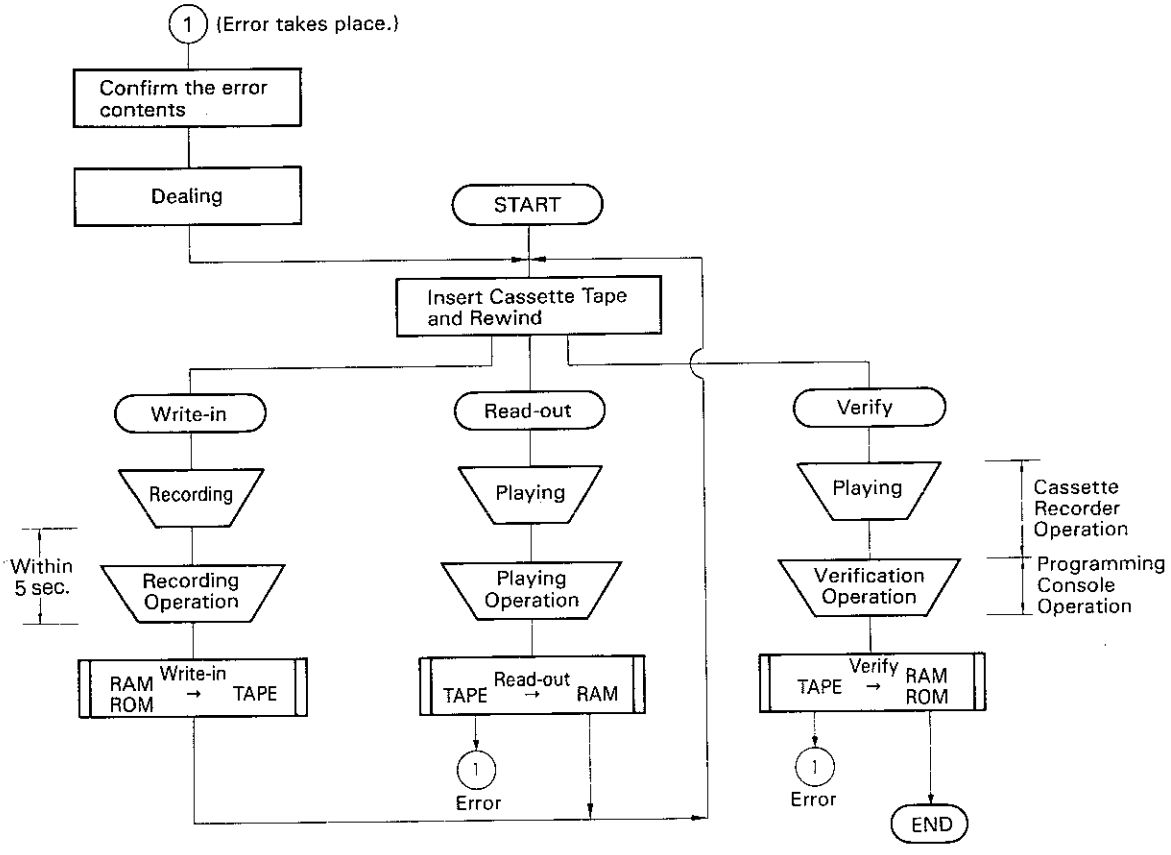
The program is being saved from the first program address to the end of the RAM/EPROM memory area. The PC display increments as each address goes into the cassette recorder. During this period you can halt the save operation by pressing

CLR

```
1100 RECORD END
```

10. When the program (plus vacant memory) has been completely recorded, the save operation stops with this message.

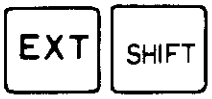
Refer to this flowchart for the correct procedure when saving your program to the cassette recorder.



Loading and verifying the program

Before loading a program into the C20, determine that the tape is correctly positioned. The taped program should begin to load into the PC within 5 seconds after the loading operation has been initiated. You can determine this by listening for the sound that marks the beginning of the program. Then rewind the tape slightly so a few seconds of blank tape precedes the beginning of the program. Then follow these steps:

1. Press



2. Press PLAY on the tape recorder to start loading the tape into the PC.
3. Press


```
0000MT PLAY
```

on the PC keyboard. When you do, this message appears and you'll see a blinking rectangle on the upper right portion of the display.

4. Wait about 7 minutes for the program to be completely loaded.
5. When the loading is complete, a message appears.

```
1100PLAY END
```

Verifying the program

It's always a good practice to check that the program was loaded correctly. This is done by performing the following procedure:

1. Rewind the tape until you reach the beginning. Provide about 5 seconds of blank tape leader before the taped program segment begins.
2. Press

3. Turn on the PLAY button of the tape recorder.
4. On the C20 keyboard, press


```
0000MT VER
```

When you do, a message appears and a blinking rectangle indicates that program verification is taking place.

5. Wait about 7 minutes. A message will appear when verification has been successful.

```
1100VER END
```

6. If a program loading error has occurred, one of the messages described in Appendix E will be displayed. In this case, repeat the tape load operation.

Chapter 6 Application examples

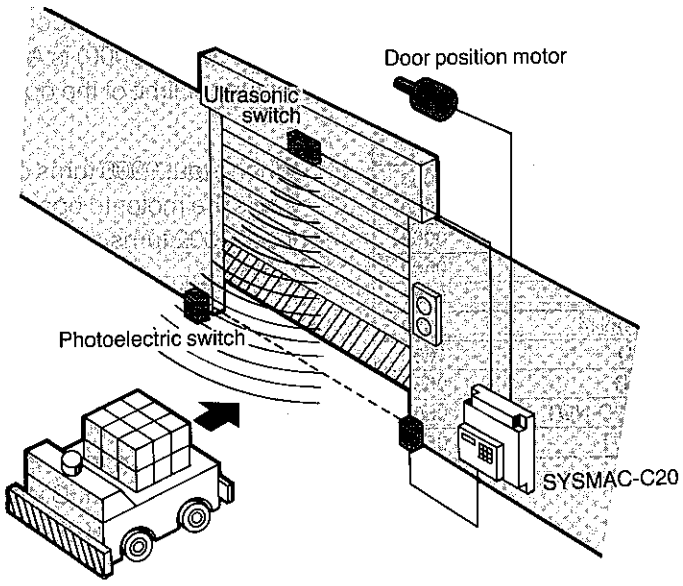
Overview

As a member of the SYSMAC C-Series programmable controllers, the C20 is intended for economical control of operations that have a relatively restricted number of automated tasks. Nevertheless, the number and types of applications possible for equipment and process control using the C20 are practically limitless—ranging from automatic door operation to many types of assembly line applications.

This chapter presents examples of some of those applications. In each case an explanation of the application program, its ladder diagram, and the coding chart are given. These programs are intentionally simplified and are meant for instructional purposes only.

Automatic control of warehouse door

In this example, the programmable controller is used to open and close an automatic door to a warehouse to allow an approaching object (automobile or other vehicle) to enter or leave. The C20 makes it possible for two different sensing systems to be used to control the door operation.



Explanation

As input devices that send control input signals to the C20, an ultrasonic switch and a photoelectric switch are employed.

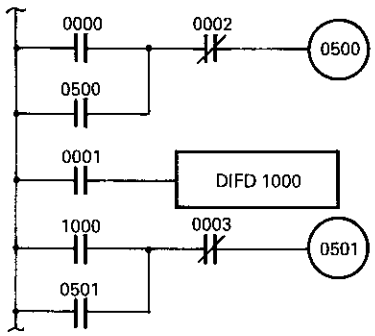
The ultrasonic switch emits an ultrasonic wave. When there is an object (vehicle) in the way of this ultrasonic wave, the wave reflects back to the ultrasonic switch which detects the object.

The photoelectric switch used in this example consists of two elements: a light source and a receiver. The light source emits a light beam that is constantly received by the receiver. If a vehicle or other object interrupts the light beam, the photoelectric switch detects it.

In response to the input signals sent from these switches, the C20 issues control outputs that drive the motor of the door for opening and closing. The C20 also receives inputs from two additional switches: one detects the upper limit of the door movement and the other detects the lower limit.

The following is part of a ladder diagram that might be used to perform these coordinated tasks.

Ladder diagram



Coding chart

Address	Instruction	Data
0200	LD	0000
0201	OR	0500
0202	AND-NOT	0002
0203	OUT	0500
0204	LD	0001
0205	FUN14	1000
0206	LD	1000
0207	OR	0501
0208	AND-NOT	0003
0209	OUT	0501

I/O assignment

Input	Relay
Ultrasonic switch	0000
Photoelectric switch	0001
Switch detecting upper limit of door movement	0002
Switch detecting lower limit of door movement	0003
Output	
To raise door	0500
To lower door	0501

Operation

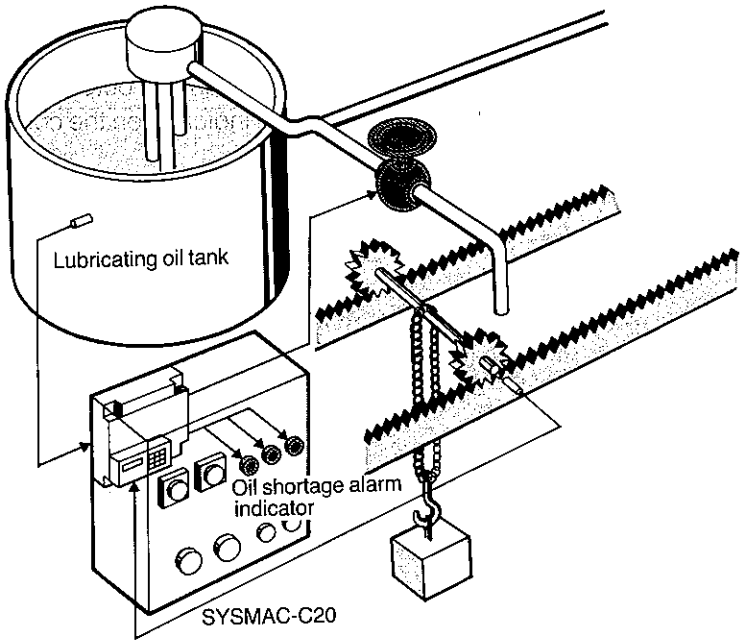
As we see in the ladder diagram, the input signal from the ultrasonic switch (input 0000) is ANDed with a limit switch (input 0002) that detects the upper limit of the door movement. This switch is normally closed.

When input 0000 turns on, the C20 outputs a signal (output 0500) that starts the motor to open the door. When the door moves to its upper limit, input 0002 turns on. This stops the output 0500 signal, and that stops the motor.

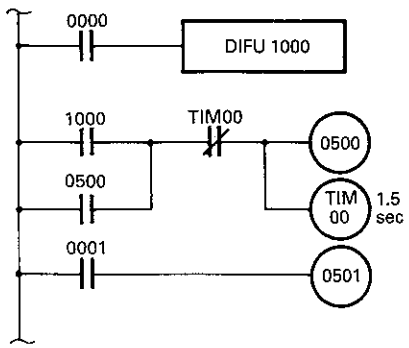
At the same time, the input signal from the photoelectric switch (input 0001) is connected to a differentiation-down circuit. (This circuit is programmed by the DIFD instruction, described in Appendix B, Instruction words.) This connection causes Internal Auxiliary Relay 1000 assigned to the DIFD instruction to turn on for only one program scan time at the falling edge of input 0001. Internal Auxiliary Relay 1000 is ANDed with a second limit switch that detects the lower limit of the door movement. Consequently, when Internal Auxiliary Relay 1000 turns on the AND condition is satisfied for one scan time. During this period, output 501 turns on. This then turns on the motor which lowers the door.

Automatic lubricating oil supplier

In this example, the C20 is used to control the supply of lubricating oil to the gears and bearings of an assembly line. This automatic lubrication reduces friction and wear on the parts, thus improving the efficiency of the assembly line and decreasing the amount of energy consumed by high-power motors.



Ladder diagram



Explanation

When the object to be lubricated reaches a predetermined point, the lubricating oil is applied. In the above illustration, a gear is supplied with oil. A sensor detects the gear when it comes to a specific position. In response to the input signal from this sensor, the C20 outputs a signal that opens an electromagnetic valve. This valve is opened for a set period of time, supplying a predetermined quantity of oil.

The level of the lubricating oil in the tank is monitored by a sensor that serves as an input device to the C20. If the oil level in the tank falls below a specific level, the sensor detects it and inputs a signal to the C20. In response, the C20 outputs a signal that illuminates an alarm indicator lamp on the C20's control panel.

I/O assignment

Input	Relay
Position detection Lower-limit oil level	0000 0001
Output	
Electromagnetic valve for oil supply Oil shortage alarm indicator	0500 0501

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	DIFU (13)	1000
0202	LD	1000
0203	OR	0500
0204	AND-NOT	TIM00
0205	OUT	0500
0206	TIM	00
		#0015
00207	LD	0001
0208	OUT	0501

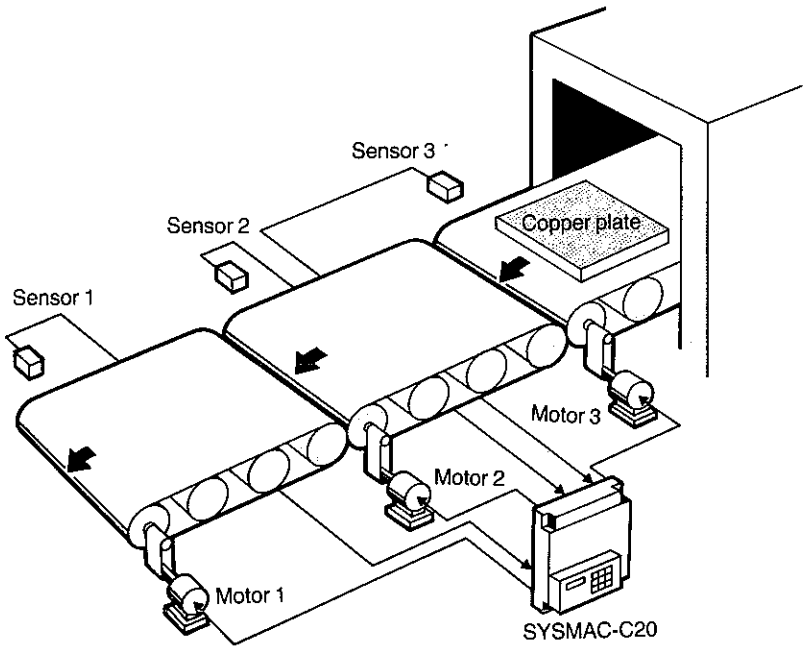
Operation

The signal from the sensor that detects the position of the part to be oiled is connected to a differentiation-up circuit so that the Internal Auxiliary Relay 1000 turns on for one scan time at the leading edge of the input signal (input 0000). When this internal auxiliary relay turns on, output 0500 turns on, causing the electromagnetic valve to open, supplying oil to the gear. The valve stays open for 1.5 seconds as governed by Timer 00.

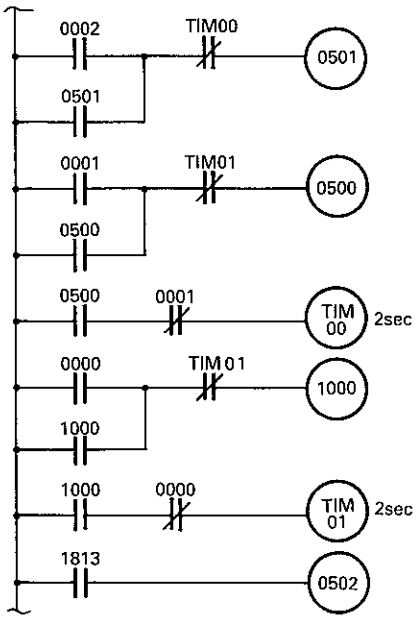
When the oil in the tank is almost depleted, a sensor (assigned as input 0001) turns on and output signal 0501 is issued by the C20 to illuminate an alarm indicator on the control panel:

Conveyor belt motor control

The C20 is used to start and stop motors of a segmented conveyor belt in this application. This permits only those belt sections actually carrying an object to be moving, while those not carrying an object remain stationary, thus conserving power.



Ladder diagram



Coding chart

Address	Instruction	Data
0200	LD	0002
0201	OR	0501
0202	AND-NOT	TIM00
0203	OUT	0501
0204	LD	0001
0205	OR	0500
0206	AND-NOT	TIM01
0207	OUT	0500
0208	LD	0500
0209	AND-NOT	0001
0210	TIM	00
		#0020
0211	LD	0000
0212	OR	1000
0213	AND-NOT	TIM01
0214	OUT	1000
0215	LD	1000
0216	AND-NOT	0000
0217	TIM	01
		#0020
0218	LD	1813
0219	OUT	0502

Explanation

In this example, metal plates are being transported. The position of a plate is detected by a proximity switch located next to each belt segment. As long as the metal plate is within the detecting range of the switch, the programmable controller issues a control output that causes the motor of the next conveyor belt to work. When the metal plate moves out of the detection range of the proximity switch, the timer is activated and when the set time has elapsed, the motor of the first conveyor belt stops.

I/O assignment

Input	Relay
Sensor 1 Sensor 2 Sensor 3	0000 0001 0002
Output	
Motor 1 Motor 2 Motor 3	0500 0501 0502

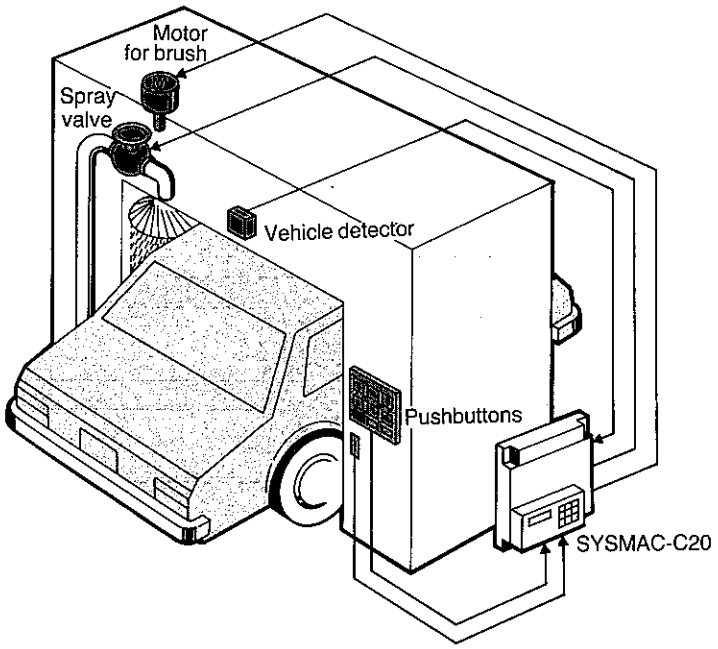
Operation

Since the metal plate is detected by sensors 3, 2, and 1 in that sequence, the first input signal in the ladder diagram comes from sensor 3 (input 0002). When input 0002 turns on, output 0501, corresponding to motor 2, is issued by the C20.

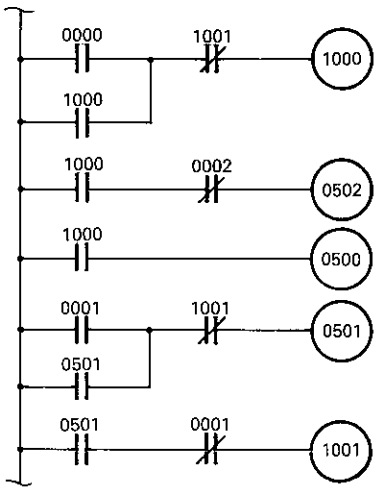
When input 0001 turns on, motor 1 is started by output 0500. The operation of motor 2 only lasts 2 seconds, the time required for the metal plate to travel to the next conveyor belt. Thus the motors operate only when they are actually needed.

Automatic car washing machine

The C20 is used to control a car washing machine in this example.



Ladder diagram



Explanation

As input devices, a vehicle-detecting device and pushbutton switches are used. In response to the signals from these devices, the C20 opens a valve for the spray and starts the motor for the revolving brush.

I/O assignment

Input	Relay
Start button	0000
Vehicle detector	0001
Condition at which washing machine stops	0002
Output	
Spray valve	0500
Brush motor	0501
Movement of washing machine	0502

Coding Chart

Address	Instruction	Data
0000	LD	0000
0001	OR	1000
0002	AND-NOT	1001
0003	OUT	1000
0004	LD	1000
0005	AND-NOT	0002
0006	OUT	0502
0007	LD	1000
0008	OUT	0500
0009	LD	0001
0010	OR	0501
0011	AND-NOT	1001
0012	OUT	0501
0013	LD	0501
0014	AND-NOT	0001
0015	OUT	1001

Operation

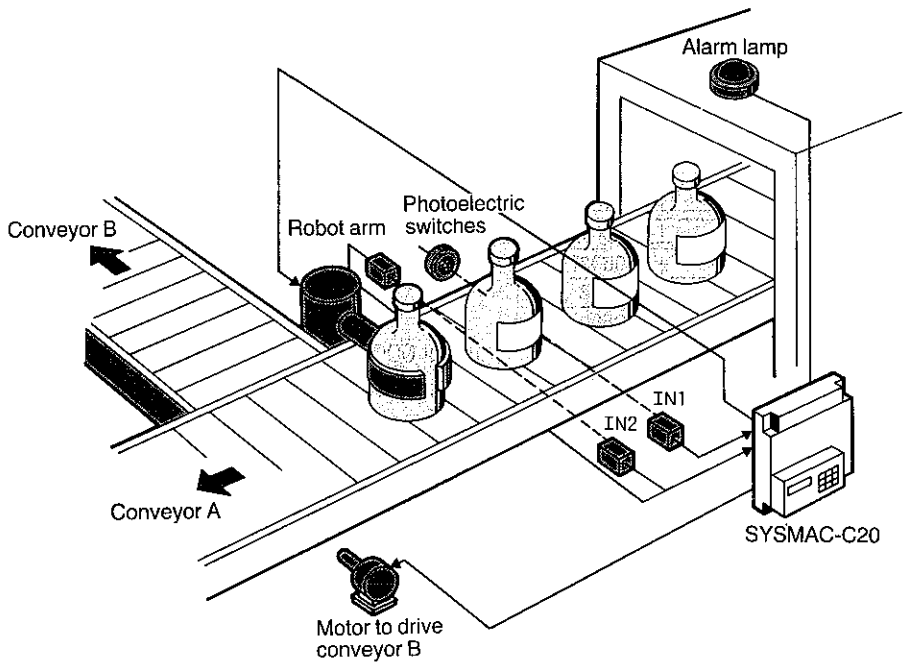
When the start button (input 0000) is depressed, the washing machine begins operating (i.e., the machine starts moving and at the same time, the spray valve is opened).

The washing process continues until Internal Auxiliary Relay 1001, which can also serve as a reset input, is opened. Input 0000 and Internal Auxiliary Relay 1001 are ANDed and the result is internally output to Internal Auxiliary Relay 1000. This result is then ANDed with input 0002 (the condition under which the washing machine is stopped). As long as the condition of this AND circuit is satisfied, output 502 is turned on, causing the washing apparatus to travel the entire length of the vehicle.

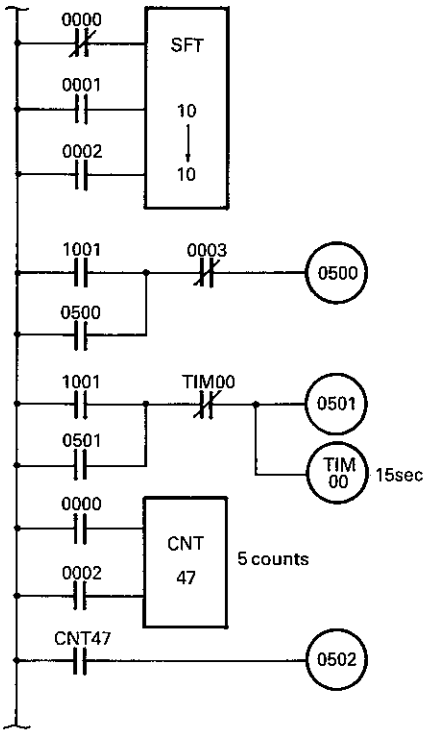
Internal Auxiliary Relay 1000 output is also directly connected to output 0500 that causes the spray valve to open. Output 501 is issued to start operating the brush when the vehicle detector sends a signal to the C20 (input 0001) (unless Internal Auxiliary Relay 1001 is open).

Bottle Label Detection

This application features a system in which bottles being moved on a conveyor belt are checked by photoelectric switches for labels.



Ladder diagram



Coding chart

Address	Instruction	Data
0200	LD-NOT	0000
0201	LD	0001
0202	LD	0002
0203	FUN10	10
		10
0204	LD	1001
0205	OR	0500
0206	AND-NOT	0003
0207	OUT	0500
0208	LD	1001
0209	OR	0501
0210	AND-NOT	TIM00
0211	OUT	0501
0212	TIM	00
		#0150
0213	LD	0000
0214	LD	0002
0215	CNT	47
		#0005
0216	LD	CNT47
0217	OUT	0502

Explanation

If a defective product (a bottle without a label) is detected, the C20 directs a robot arm to remove the bottle and place it on another conveyor line. The C20 also counts the number of defective products. If their number reaches a preset value, the C20 causes an alarm lamp to go on.

I/O assignment

Input	Relay
Label missing detection	0000
Bottle detection	0001
Stop	0002
Robot arm's original position detection	0003
Output	
Robot arm	0500
Conveyor B motor	0501
Alarm lamp	0502

Operation

This application makes use of the shift register (SFT) instruction. This instruction must be programmed in the order of data input (0000), clock input (0001), and reset input (0002). Moreover, the instruction data must be specified in channel units.

In this example, channel 10 is specified as the data. When a defective product (a bottle bearing no label) is detected, input 0000 is turned on and the state is stored in point 1000 of channel 10. At the next input signal of 0001, the state of point 1000 of channel 10 is shifted to 1001, indicating detection of a defective product. In response, output 0500 is issued unless input 0003 is applied. When output 0500 is issued, the robot arm removes the defective product on conveyor A and places it on conveyor B.

Input 0003 is turned on when the arm of the robot, after removing the defective bottle, returns to its original position. When relay 1001 is turned on, output 0501 is also issued, starting conveyor B.

Inputs 0000 and 0002 also serve as the count input and reset input, respectively, of the counter. When the counter counts five count inputs (five defective bottles), the counter turns on output 0502, which in turn illuminates the alarm lamp.

Appendix A System expansion and peripherals

Overview

The C20 has a full line of peripherals to expand the system and provide a number of valuable support services. These are the peripheral devices available for the C20:

Expansion I/O Unit

I/O Link Unit

SYSBUS System for optical-fiber data transmission

Link Adapter

Peripheral Interface Unit

Printer Interface Unit

PROM Writer

Graphic Programming Console (GPC/CRT)

Multisupport Base

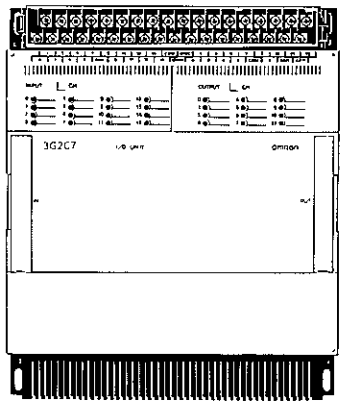
This appendix presents an abbreviated introduction to these C20-related products, many of which are upwardly compatible with other members of the C-Series line of programmable controllers.

Expansion I/O unit

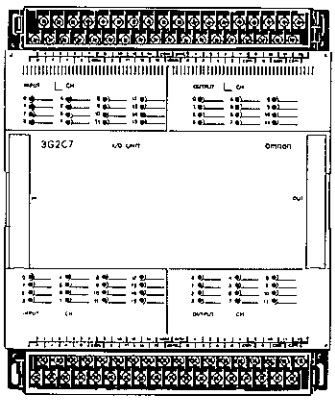
The basic C20 comes with 28 I/O points when the DC input type CPU is used. The number of I/O points available from the AC input type CPU is 26. If your controlled system so requires, you can increase the number of I/O points of an expandable C20 by connecting either one or two expansion DC input type I/O units to the DC input type CPU for a maximum of 140 I/O points. The maximum number of I/O points is 130 when the system is configured of AC input type units.

The expansion I/O unit can be classified by two factors: the input type (DC or AC) and the number of I/O points. The DC input type comes with 28 or 56 I/O points, whereas the AC input type offers 26 to 52 I/O points.

Type 3G2C7-MC223/227



Type 3G2C7-MC224/228



Explanation

When an expansion I/O unit is connected to the CPU, the input/output channels of the expansion I/O unit and the CPU are automatically designated.

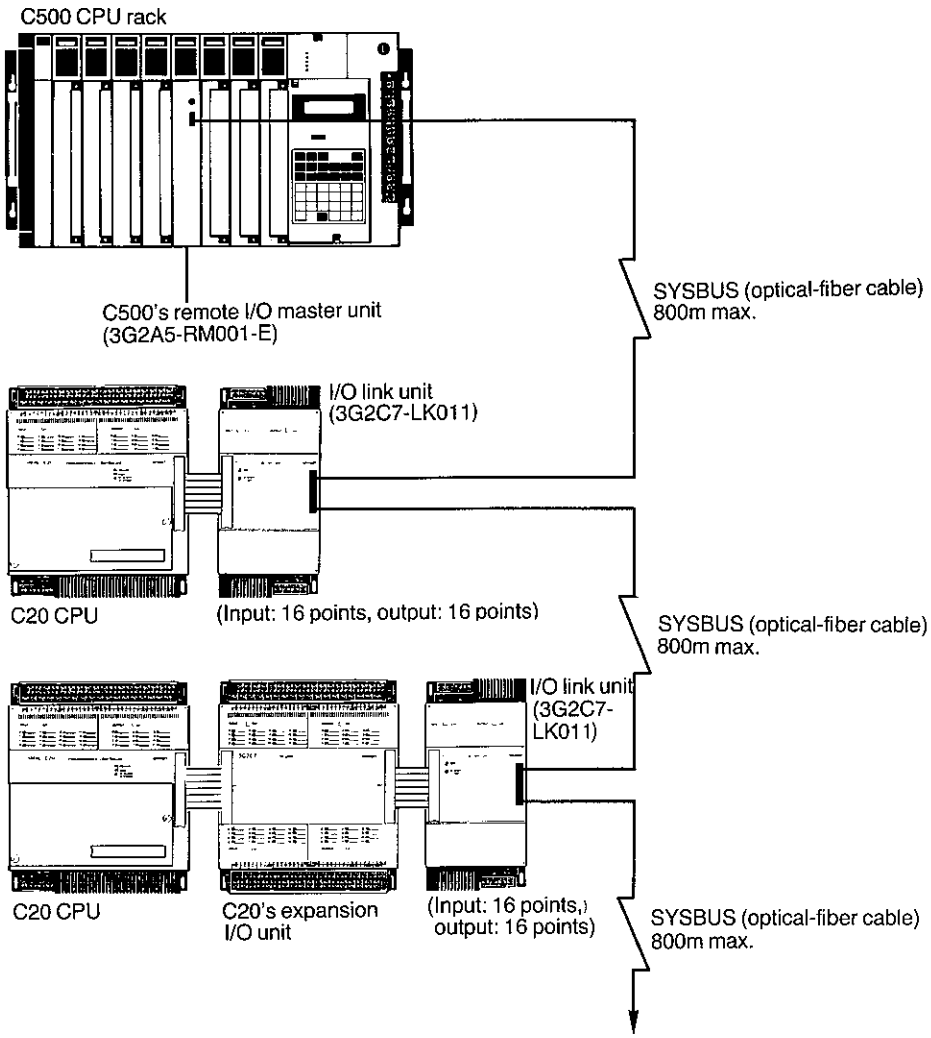
Since the CPU's input channel is fixed to channel 00 and its output channel is fixed to channel 05, the new input and output channels provided by the expansion I/O unit are numbered beginning from the channel next to the CPU's input/output channel. That is, when a 28- or 26-point expansion I/O unit (with one input and one output channel) is connected to the CPU, the new input channel is assigned as channel 01 and the new output channel is assigned as channel 06.

Likewise, if a 56- or 52-point expansion I/O unit (with two input and two output channels) is connected, the new input channels are assigned as channels 01 and 02 and the new output channels are assigned as channels 06 and 07.

In addition to the load power supply, a separate power supply is required to drive the C20's output relays.

I/O link unit

System configuration example



The I/O link unit makes possible state-of-the-art optical-fiber communication within a SYSBUS system employing a higher-level PC (such as the C120 or C500),one or more C20s, and other C-Series PCs.

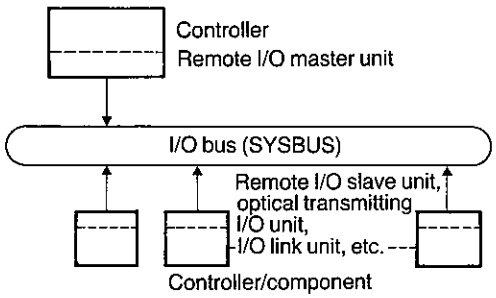
Remote data communication between the C20s and other devices is controlled by a remote I/O master unit. The controlled equipment can be located as far as 800 meters from the I/O master unit.

The network of control equipment is tied together through the SYSBUS system, in which a pair of optical-fiber cables transmits high-speed data over long distances with high immunity to electrical noise.

The I/O link unit comes in one type: 3G2C7-LK011

Since the remote I/O master unit serves to control the optical transmitting I/O unit as well as the I/O link unit, it can be used to configure a system in which both types of units are used together with a remote I/O slave unit. This means a distributed system using a number of SYSMAC PCs is possible.

SYSBUS system



Applications

For interlocking pieces of equipment or stations, high-speed data transmission/reception, improving line noise immunity, reducing wiring requirements.

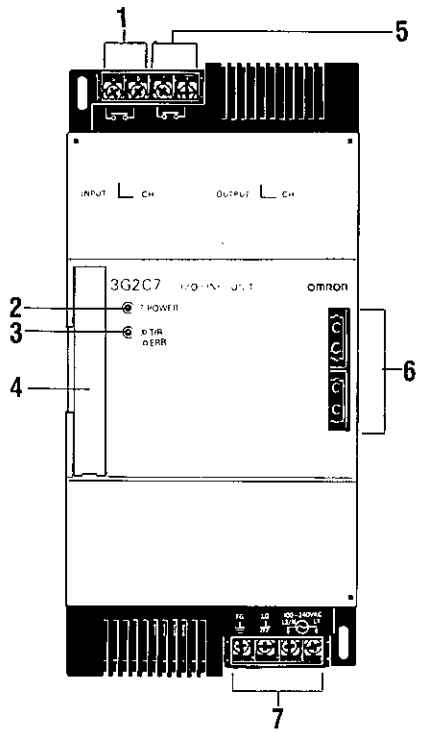
Features

Long-distance, high-speed data communication realized by an optical transmission system with excellent noise immunity. Because only a single pair of optical-fiber cables is used as the communication line, the amount of wiring can be significantly reduced.

Specifications

Communication system	Bidirectional, half-duplex
Communication format	1:N
Transmission speed	187.5k bps
Transmission system	Time-division multiplexing
Synchronization system	Start-stop synchronization
Transmission control process	Exclusive process (cyclical control system)
Number of stations possible	64 max. (with a transmission capacity of 8 points each)
Transmission lines	Double-core optical-fiber cable (crystal polymer, core diameter: 250μm)

I/O Link unit



1 RUN OUT terminals

Used to monitor the operating status of the I/O link unit. Output signals are issued from these terminals when power is applied to the C20 CPU and to the I/O link unit with the PC connected to the remote I/O master unit. The PC must be in RUN or MONITOR mode.

2 LED power indicator

Turns on when power is applied to the I/O link unit.

3 LED indicator for transmission/error indication

Blinks during normal transmission. A steady glow indicates a transmission or bus error has occurred.

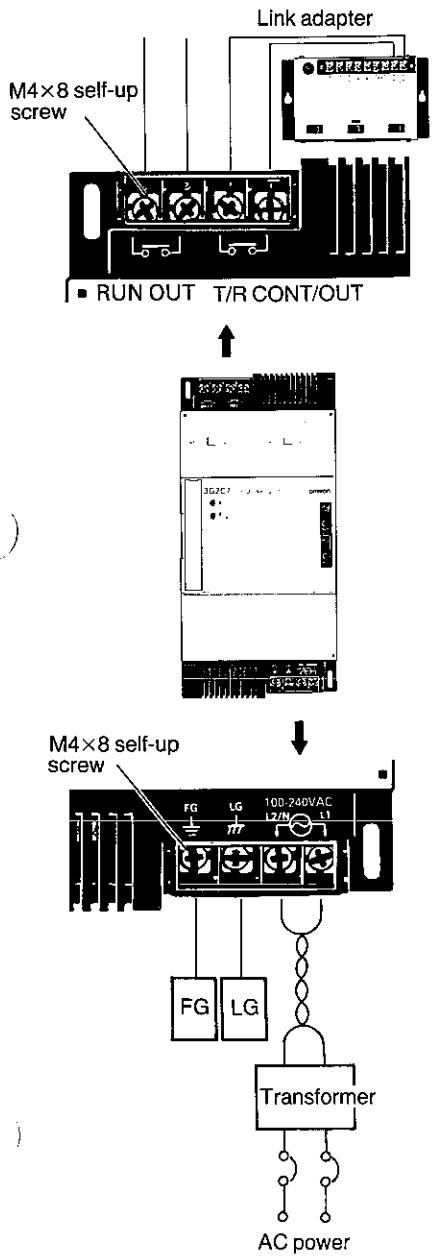
4 Connector
Connects the CPU or expansion I/O unit of the C20 to the I/O link unit.

5 T/R CONT OUT terminals
Outputs a repeater signal in a system incorporating SYSBUS. A link adapter is also used. The repeater output turns on when power is applied to the CPU of the C20 and I/O link unit.

6 SYSBUS connector
Connects an optical-fiber cable to the I/O link unit.

7 Power terminals
For applying AC power. The FG and LG terminals must be grounded at a ground resistance of less than 100Ω.

I/O link unit wiring



The following points should be kept in mind when wiring the I/O link unit.

Be sure to perform grounding at a ground resistance of less than 100Ω to safeguard against electric shock.

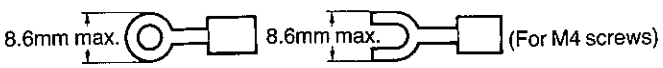
The LG terminal is a noise filter neutral terminal. Normally, grounding is not required. However, if serious noise problems exist and may cause the I/O link unit to malfunction, ground this terminal at a resistance of less than 100Ω.

Use twisted pair wires having a cross-sectional conductor area of at least $2mm^2$ (AWG14).

For normal power line noise, the C20's internal noise-suppressing circuit is sufficient. However, supplying power through a transformer with a voltage ratio of 1:1 helps to greatly reduce equipment-to-ground noise. Installation of such a transformer is recommended.

Use an AC power supply with a supply voltage within the rated operating voltage range (AC 85 to 264V).

Use solderless terminals.



For details on the optical-fiber cable, refer to the user's manual for the SYSMAC-C Series optical-fiber cable.

Channel and end station setting

For the I/O link unit to operate, it must be designated as one of the input/output channels of the PC in which the remote I/O master unit is installed. This channel setting is carried out using the DIP switch in the I/O link unit.

A maximum of 16 I/O link units can be connected to a remote I/O master unit. However, the combined number of I/O points of the connected I/O link units cannot exceed the I/O point limit of the PC in which the remote I/O master unit is installed. (If the C500 is used, the maximum number of I/O points is 512; for the C120, the maximum is 256).

In addition to the I/O link units, remote I/O slave units and optical transmitting I/O units can also be connected to the remote I/O master unit.

Note: The remote I/O slave unit and optical transmitting I/O units are peripherals used with the higher-level SYSMAC-C Series PCs, and cannot be used alone with the C20. See the user manuals of these PCs for information on those devices.

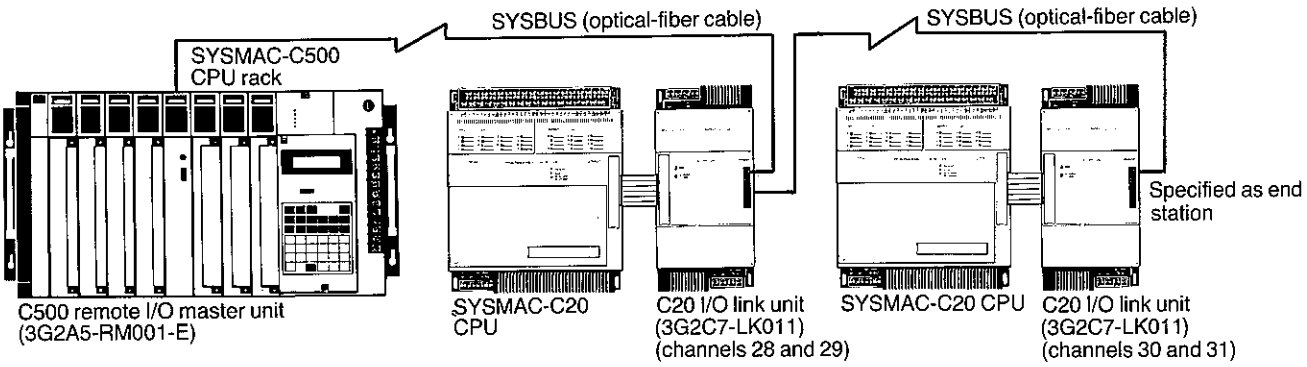
The network of interrelated components is governed by PC in which the remote I/O master unit is installed using a system of channel designation, which is explained next.

The channels of the input and output units connected to the PC (either the CPU alone, or the CPU with one or two expansion I/O units) are automatically assigned starting from channel 00. You must assign the channel number of each of the I/O link units and optical transmitting I/O units, however. When doing this, you must make sure that no channel designations overlap—the same channel should not be assigned to two different devices.

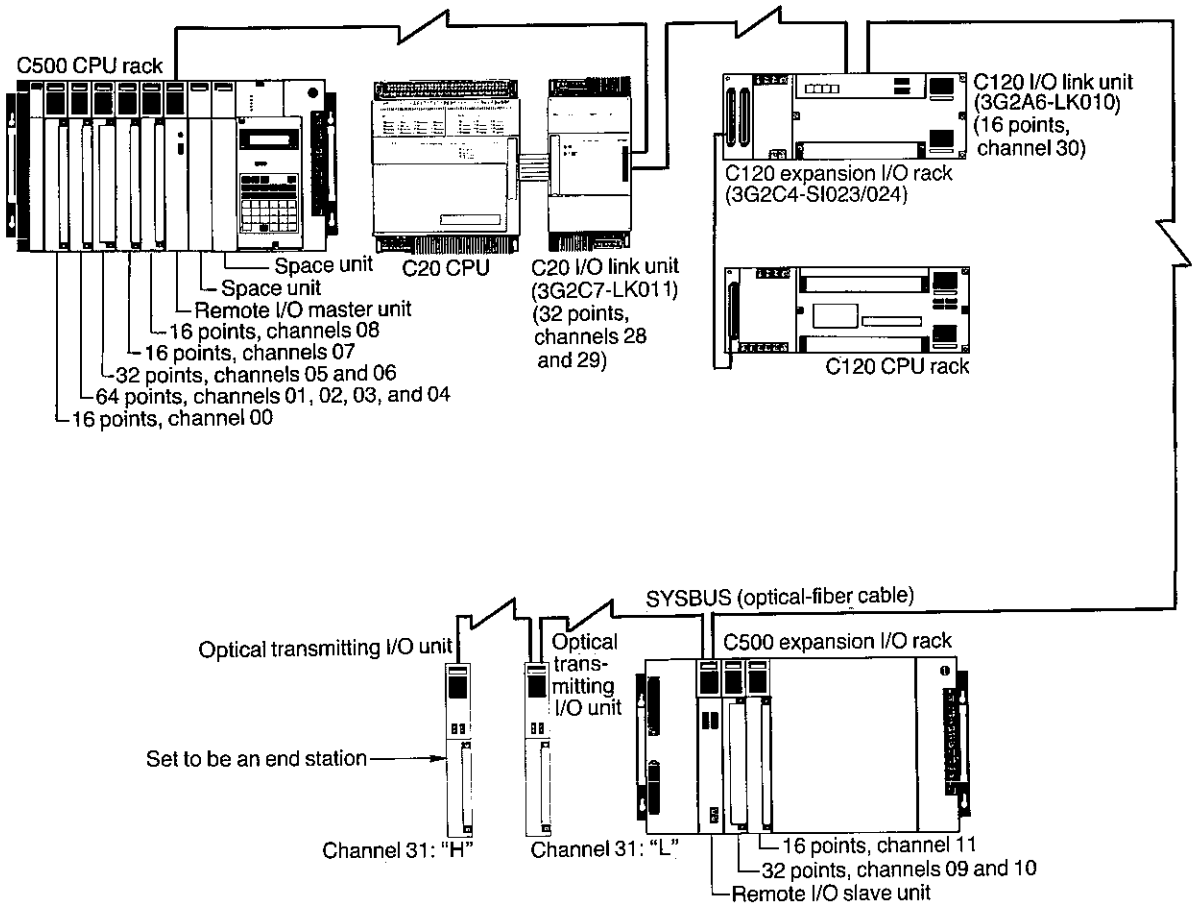
There's a technique you can use to keep this from happening. When assigning the channels for those devices, start with the highest channel number—channel 31—and assign the following channels decrementing for each one.

Of the I/O link units connected to the remote I/O master unit, the I/O link unit which is physically the furthest from the remote I/O master unit serves as the end station. When you assign channel numbers and must designate the "end station" keep in mind that this is a *physical* description and is not related to the size of the channel number. Channel 31 may or may not actually be the end station.

System example 1



System example 2



Note: **On using optical transmitting I/O unit**

The optical transmitting I/O units are used for remote control of equipment when eight or fewer input or output points are required. The C20 communicates with these optical transmitting I/O units through the remote I/O master unit installed in the higher-level PC (C120, C250, or C500).

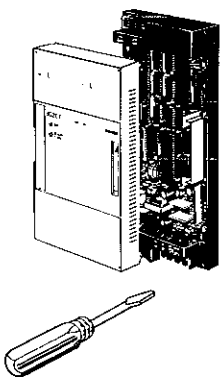
Each optical transmitting I/O unit has eight input or output points. This means that when a channel is to be formed, two of the units must be used together, with one serving as the "higher" half of the channel and the other serving as the "lower" half. Both halves must be assigned as either input or output. You cannot mix input points and output points in creating one channel.

For example, when you have specified a pair of optical transmitting I/O units as the higher and lower halves of channel 31, you should not specify the higher half as an input channel and the lower half as output. Instead, create a second channel using one or more other optical transmitting I/O units. Thus, channel 31 would be used for input and channel 30 would be used for output.

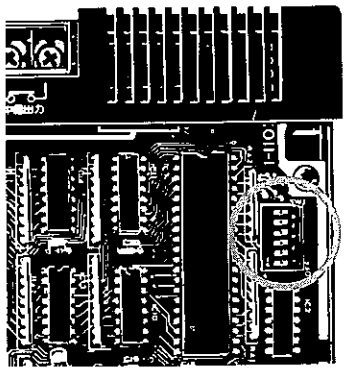
DIP switch setting

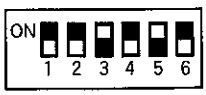
Use this procedure for setting the DIP switch when designating channels and the end station.

1. First, refer to the I/O assignments of the PC or PCs connected to the remote I/O master unit. This is so you can determine the last input/output channel number. This procedure is explained in a moment.
2. Decide what the channel number of the I/O link unit should be. When doing this, be careful that no I/O link unit channel number overlaps the last assigned channel number of the PC or exceeds the allowable number of I/O points of the PC. If a channel number higher than the maximum number of I/O points is assigned to an I/O link unit, an error occurs.



3. Next check whether the I/O link unit connected to the C20 is at the end of the SYSBUS (that is, the unit physically furthest from the remote I/O master). If so, this I/O link unit must be specified as the end station. Check that no other I/O link unit connected to the SYSBUS is already specified as the end station.
4. Turn off the power to the I/O link unit.
5. Confirm that the LED power indicator on the I/O link unit is not lighted. Then open the front cover of the I/O link unit with a flat-bladed screwdriver.





DIP switch setting example (channel 5)

6. Locate the six-pin DIP switch on the upper right side of the printed circuit board inside the unit.

These pins, represented by numbers 1 to 5, are used to specify a channel. Each pin has a respective bit value (16, 8, 4, 2, and 1), permitting you to specify the channel you want by combining the values you need. For instance, if you wanted to set channel 5, you would turn DIP switch pins 3 and 5 ON. (Pin 3 has a value of 4, and pin 5 has a value of 1; thus, 4 + 1 = 5.) Similarly, if you wished to set channel 31, you would turn all the pins ON (16 + 8 + 4 + 2 + 1 = 31).

The remaining pin, pin 6, is turned ON only when the unit is the end station.

7. After you have carried out the DIP switch setting, reattach the cover.
8. Connect the I/O link unit to the C20 with a cable. Also connect the optical-fiber cable for the SYSBUS to the I/O link unit.

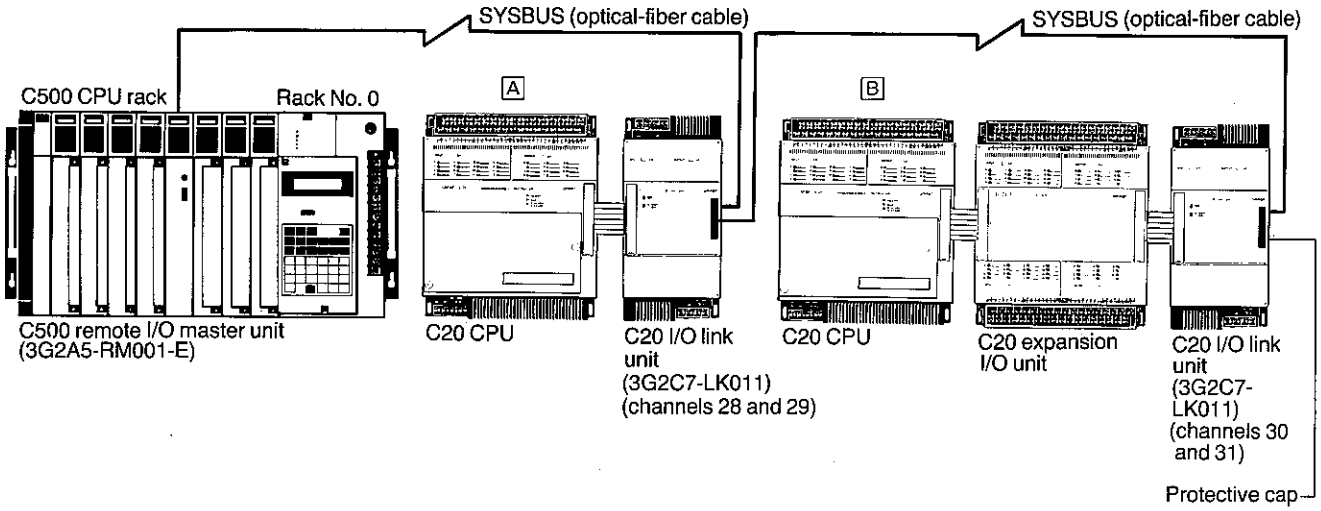
The I/O link unit has two optical-fiber cable connectors. Either connector can be connected to the remote I/O master unit. When the I/O link unit is specified as the end station, be sure to cover the unused connector with a protective cover to prevent the I/O link unit from malfunctioning due to external light.

9. Apply power to the system. The power can be applied to each unit in the system in any order. However, when a link adapter is employed, a power application sequence must be followed. The details of this are described later in the discussion on the link adapter.
10. If you should get a message on the display of the remote I/O master unit that no end station has been designated, refer to the I/O table by using the programming console.

You can start operating the system as soon as power is applied after you have performed the necessary settings. Refer to Appendix E, Maintenance and inspection, should a problem occur.

System example 3

In this system, the remote I/O master unit is connected to a C500 so that this PC can transmit and receive data to/from the C20 at a remote location.



Explanation

In this example, channels 0 to 27 have been automatically assigned to the C500, which has a remote I/O master unit installed.

To establish communication between the nearest C20 (via its I/O link unit) and the C500, channels 28 and 29 are manually assigned. This assignment is from the perspective of the C500.

This nearest C20 is "C20 (A)" in the figure.

For "C20 (B)", channels 30 and 31 are manually assigned. This channel designation is also from the perspective of the C500.

The distinction about "perspective" has been made for this reason: when you assign the channels of the C20s, the assigned *numbers* will be different from those of the higher-level PC—but the *channels* themselves are the same.

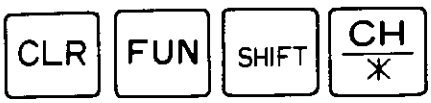
In the example, from the C20's perspective, channels 01 (input) and 06 (output) are assigned to the I/O link unit connected to the C20 (A). These channels are the same as channels 28 and 29 of the C500. To the I/O link unit connected to C20 (B), channels 03 (input) and 08 (output) are assigned. These are the same as channels 30 and 31 of the C500.

Generating the I/O table

When you find it necessary to check the I/O assignments of the C500 (or other higher-level PC) and the C20, first attach the programming console to the CPU of the C500. Then set the mode selector switch on the programming console to the PROGRAM position and perform the key operations as shown below.

Note: Before beginning, the C500, C20, and I/O link unit must be properly connected and all supplied with power.

To generate the I/O table, key in

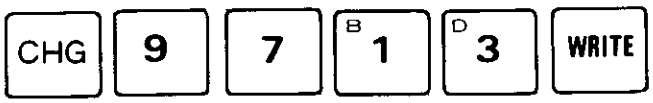


in that sequence.

```
0000I/OTBL WRITE
           ????
```

When the SHIFT and CH/* keys have been depressed, a message appears indicating the start of I/O table generation. Now key in

```
0000I/OTBL WRITE
           9713
```



You should now start to see the I/O table being generated. When it is completed, you can check its contents by doing the following. First, key in

```
0000I/OTBL WRITE
OK
```

Next key in


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

and a message is displayed. You can now make your first channel assignment inquiry.

For example, to check what kind of I/O unit is connected to the fourth slot on the C500 specified as No. 0 rack, key in

```
0000I/OTBL?
  0-4U=
```

This causes the I/O unit connected at the 04 position to be displayed.

```
0000I/OTBL READ
  0-4U=RMT0
```

In this case, the remote I/O master unit (RMT0) is connected to the fourth slot of No. 0 rack.

Beginning at that setting, by pressing

you can sequentially monitor all the channels connected to the remote I/O master unit.

```
0000I/OTBL READ
  28LU=R0-0
```

This message indicates that channel 28, specified as an output channel, is connected to the remote I/O master unit. Another press of the key reveals that channel 29 is also connected and is specified as an input channel.

```
0000I/OTBL READ
  28HU=R0-0
```

```
0000I/OTBL READ
  30LU=R0-0
```

Channel 30 is specified as an output channel.

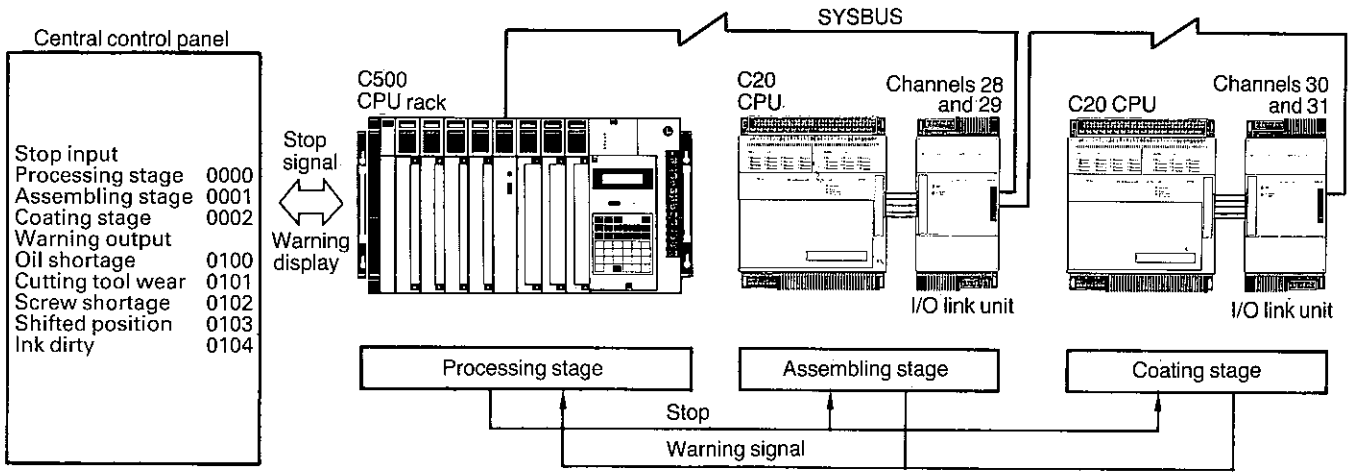
```
0000I/OTBL READ
  30HU=R0-0
```

Channel 31 is specified as an input channel.

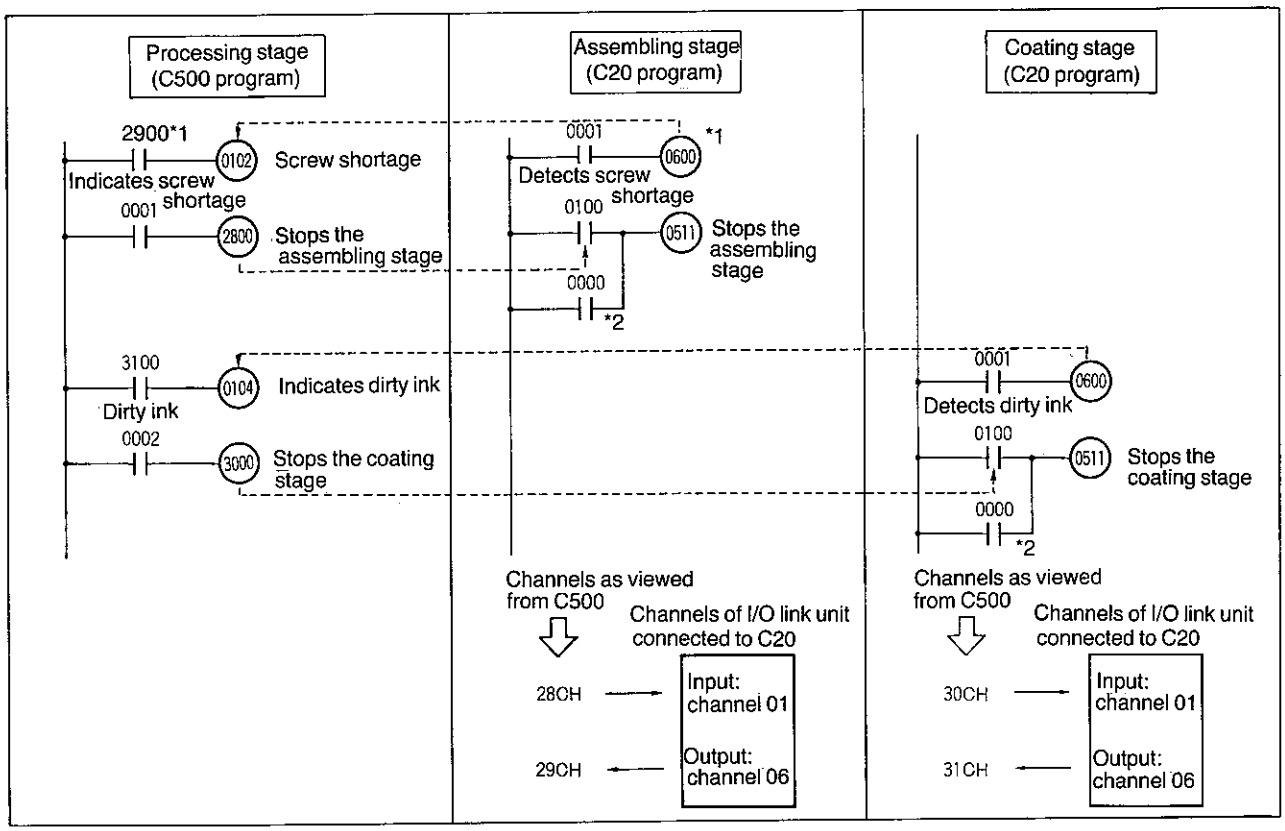
Application example

A control system employing I/O link units is shown below. This system controls three manufacturing stages: processing, assembling, and coating objects.

The dual role of the I/O link units is to transmit a specific quantity of information (such as stop and warning signals) and to interlock equipment and stations in the system. If a warning signal is generated during one of the stages, it is received by the central control panel and displayed. The control panel reacts with a return stop signal which shuts off the operation.



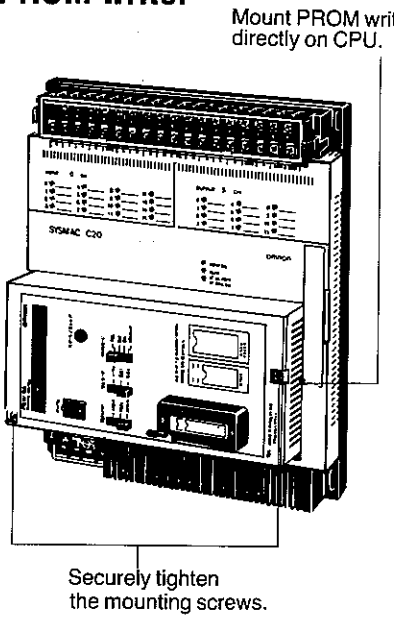
This ladder diagram shows part of the circuit used for this control system.



*1. When the screw supply is depleted during the assembling step, output relay 0600 of the I/O link unit is turned ON. Viewed from the C500, this output relay is input relay 2900 of the C500, which is connected to the remote I/O master unit. (Since the I/O link unit is using channels 28 and 29, the I/O link unit must be set as channel 28 with its DIP switch.) Therefore, the signal indicating the screw shortage that is output from relay 0600 of the I/O link unit is input to relay 2900 in the processing stage. In response, relay 0102 on the central control panel is turned ON, issuing the alarm signal.

*2. By connecting the RUN output of the I/O link unit to input relay 0000 of the C20 in the coating stage, the other two stages (processing and assembling) can be stopped in case a failure occurs in the I/O link unit in the coating stage.

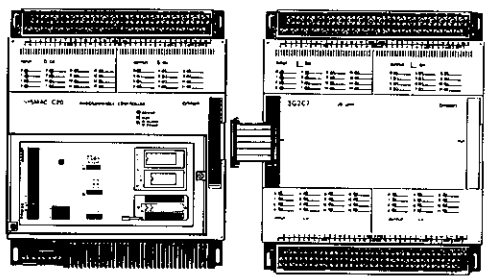
PROM writer



The PROM writer provides C20 users a means of storing their programs on microchip, as opposed to cassette tape. This way the program can be "permanently" saved or easily duplicated for use elsewhere. The EPROM (erasable programmable read-only memory) chip can be written and then later erased by ultraviolet light.

The PROM writer has functions to write the contents of the CPU memory to an EPROM chip and vice versa, and also to verify the contents of the EPROM against the CPU memory. However, these functions of the PROM writer work in "normal write mode" but not "high-speed write mode".

The PROM writer can be directly mounted on the C20. It is important that when using this unit mounted on the C20, the I/O cable connecting the CPU and expansion I/O unit not be connected. Otherwise, the PROM writer may not operate. As the EPROM chip, use Type ROM-H (#2764 or the equivalent).



The PROM writer may not operate when mounted on the C20 if the CPU is connected by cable to an expansion I/O unit.

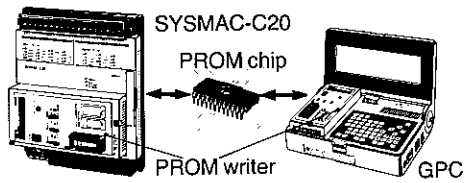
Graphic programming console

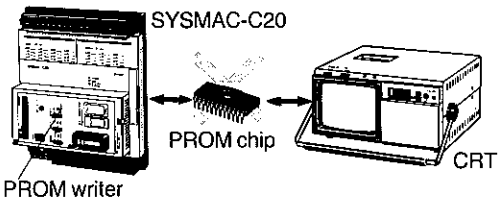
The graphic programming console is a quick, visual way to write programs using ladder diagrams and instruction words. The user presses instruction keys and other control keys on the keyboard. In response, the symbols appear on a screen on the portable unit.

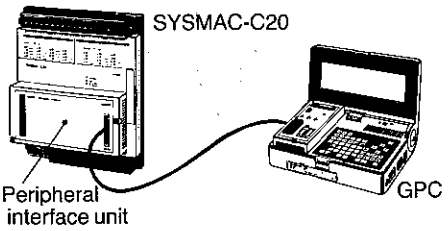
The graphic programming console is available in two types. One employs an LCD (liquid crystal display) for the screen whereas the other uses a CRT (cathode ray tube). The CRT type is referred to as "CRT" while the LCD type is referred to as "GPC".

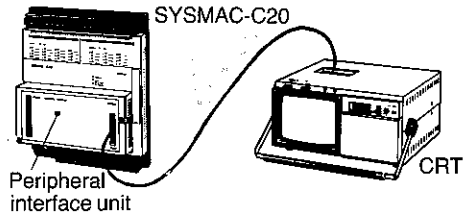
Both the CRT and GPC store programs onto cassette tape and load programs into the CPU memory using the peripheral interface unit.

Caution: PROMs that have been written by the PROM writer used with the C20 cannot be directly read by a PROM writer connected to the graphic programming console, nor can the reverse operation be performed. In either case the message "MEMORY ERR" is displayed. The peripheral interface unit is required to transfer programs between these devices.

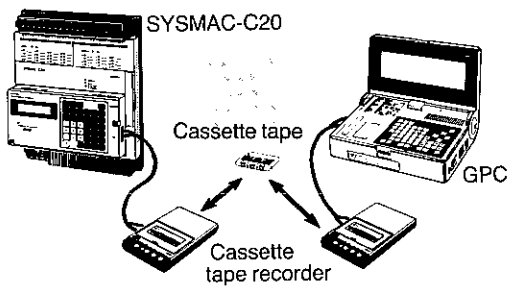


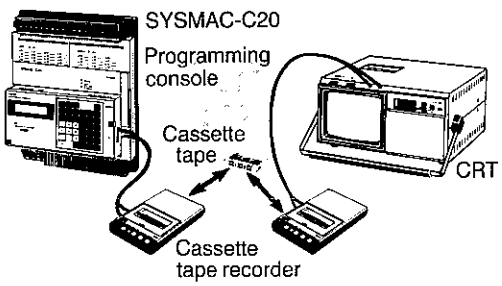


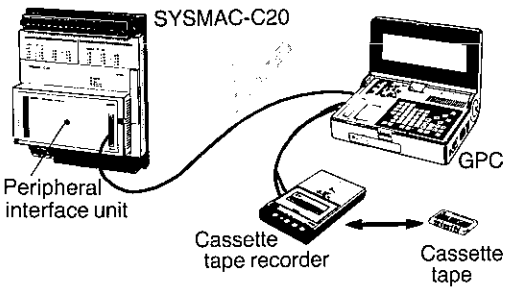


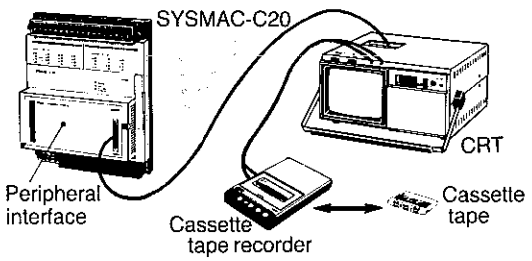


The peripheral interface unit is also required for programs recorded on a cassette tape recorder. Programs that have been saved to cassette tape connected to the programming console of the C20 cannot be directly read by a cassette tape player connected to the graphic programming console, nor can the reverse operation be performed. For this reason, always use the peripheral interface unit to transfer programs between these devices.



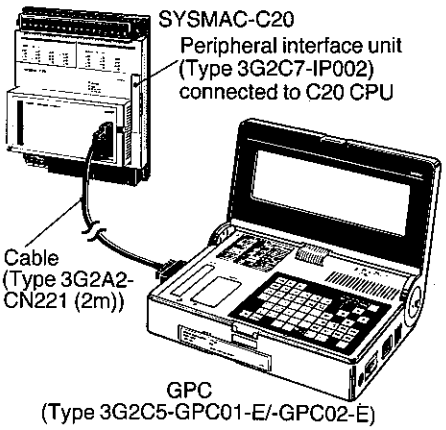






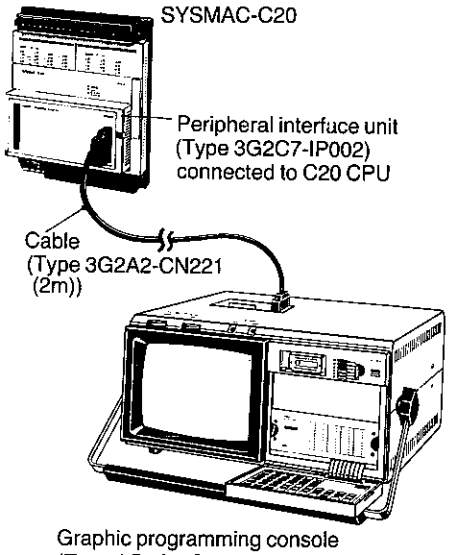
The features of the respective consoles are explained next.

GPC



The GPC allows programs to be written in both ladder diagram and mnemonic form. The program can be written on the GPC without the GPC having to be connected to the PC; once completed, the program can be transferred to the PC's internal memory. For storage purposes, the program can also be transferred to a cassette tape. During operation of the PC, the GPC can be connected to the PC to monitor the PC's operating status. A program written on the GPC can be printed out if a printer is connected to the GPC via the printer interface unit. Moreover, if the floppy disk unit is connected, high-speed transfer of the program to a floppy disk can be achieved.

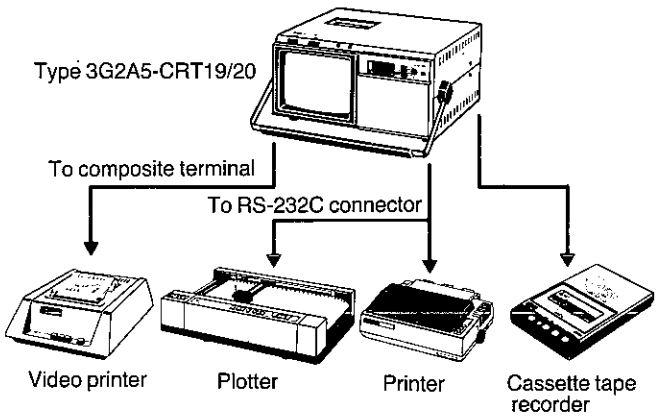
CRT



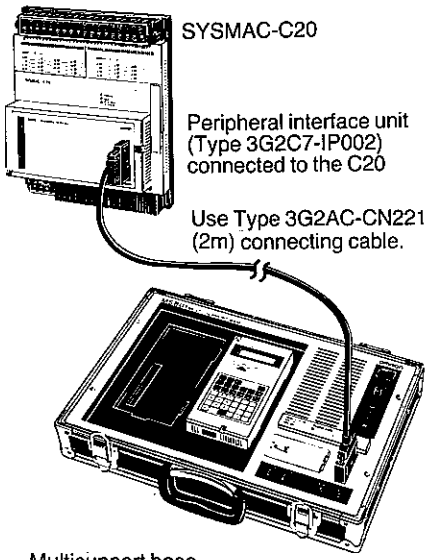
The CRT features two registration functions: label registration and user program registration. A "label" is a four-digit alphanumeric code assigned to a desired relay number making it possible for the relay numbers to be mnemonically used. With the user program registration function, a repeatedly used program can be registered and then easily called from memory whenever it is needed.

One of the other advantages of the CRT is that more peripheral devices can be connected to the CRT than is possible with the GPC.

Specifically, as shown, an X-Y plotter and video printer can be connected in addition to a printer and cassette tape recorder. For details, refer to the User's Manual for Type 3G2A5-CRT19/20 Graphic Programming Console.



Multisupport base

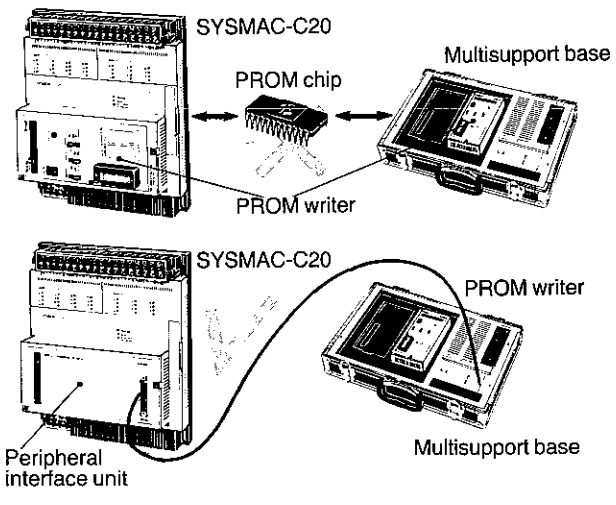


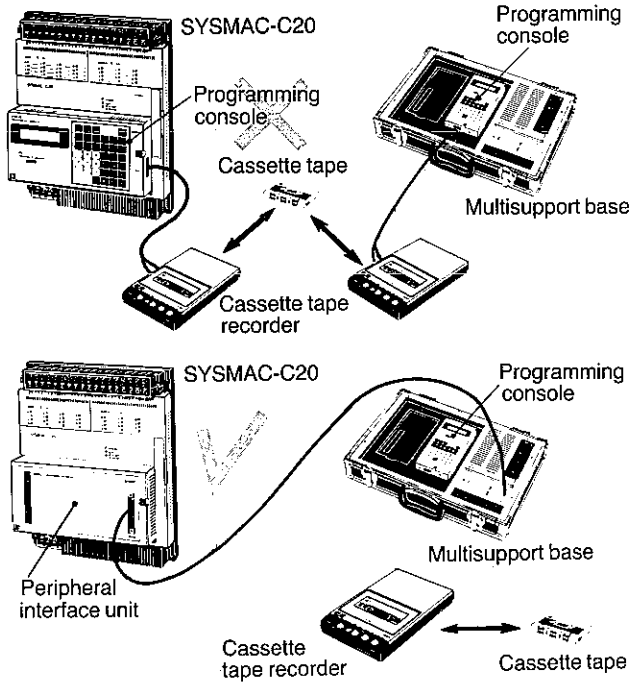
Multisupport base
(Type 3G2A5-MSB01-E/-MSB02-E)

The multisupport base serves as a programming device for C series programmable controllers. The programming console is attached to the multisupport base and used in the same way as the C20. The multisupport base also has some functions not available on the C20. The multisupport base must be connected to the C20 with a dedicated peripheral interface unit (Type 3G2C7-IP002). Note that the other peripheral interface units (Type 3G2A5-IP001-E/-IP003-E) cannot be used with the C20

Caution: PROMs that have been written by a PROM writer used with the C20 cannot be directly read by a PROM writer connected to the multisupport base, nor can the reverse operation be performed. If either attempt is made, the message "MEMORY ERR" is displayed. The peripheral interface unit is required to transfer programs between these devices.

The peripheral interface unit is also required for programs recorded on a cassette tape recorder. Programs that have been saved to a cassette tape connected to the programming console of the C20 cannot be directly read by a cassette tape player connected to the graphic programming console, nor can the reverse operation be performed.





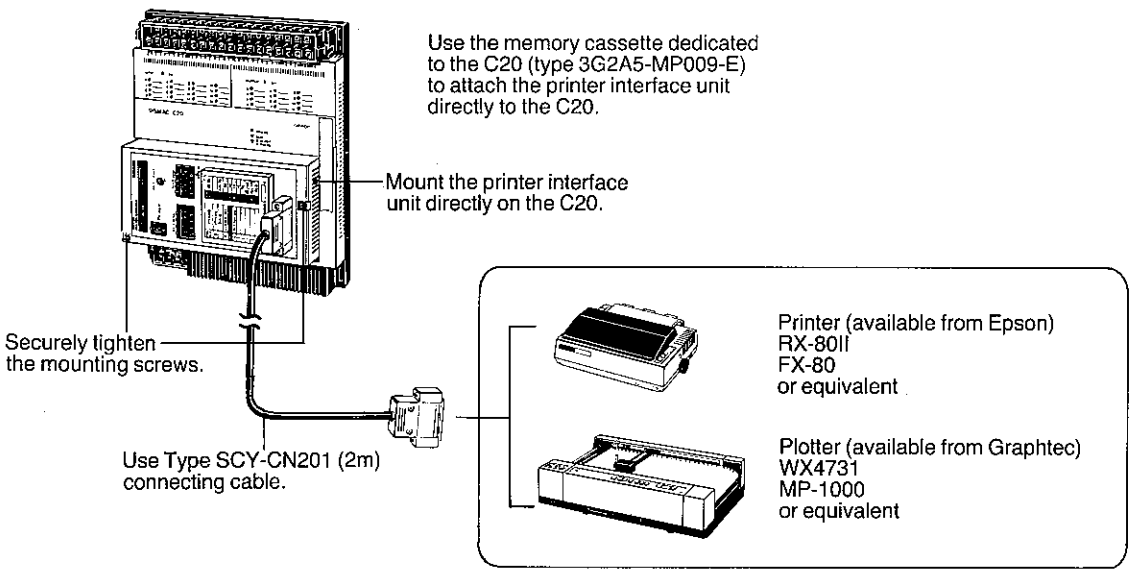
When the multisupport base is connected to a programmable controller in the SYSMAC-C series, the connected PC must be specified by the selector switch on the multisupport base. When the multisupport base is connected to the C20, the setting of this selector switch is the same as when specifying the C250.

The multisupport base can be connected to various peripheral devices in the SYSMAC-C series.

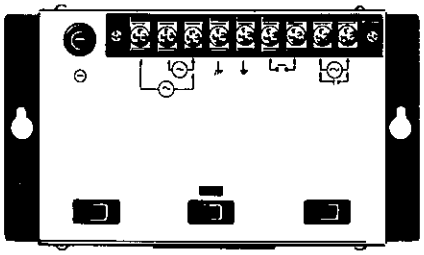
Printer interface unit

The printer interface unit interfaces the C20 with printers so that ladder diagrams, mnemonic lists, cross-reference lists, data memory contents, and other information can be printed out for a written record. This can be helpful as a diagnostic tool in correcting or modifying programs. To connect the printer interface unit to a printer, interface board #8145 available from Epson is required.

*The printer interface unit can also be used with the multisupport base or graphic programming console. In this case, however, the memory cassette of the printer interface unit must be Type 3G2A5-MP002-E.*When the printer interface unit is mounted to the C20 to print out programs, you can stop the printer in one of two ways. Either the printing can be halted at any desired point, or the printer can be made to pause between each printed page. A button on the printer is used to select between the two. When the button is pressed, the stop indicator on the printer interface unit blinks at intervals of 0.1 second if the printing is to be immediately stopped and at intervals of 0.4 second if printing the current page only is specified.



Link adapter



When optical fiber transmission lines are employed in the PC system, the link adapter (Type 3G2A9-AL002-E) may be used to prevent malfunctioning of the optical transmission path due to a power failure in one of the units of the system.

Appendix B Instruction words

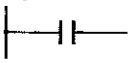
Overview

This appendix describes the instructions used by the C20. The information includes the name, key input sequence, symbol, and function of each instruction along with a ladder diagram and coding chart.

LOAD (LD)

(address) LD (relay no.)

Symbol



Function

Starts operation of each logic line; forms substrings

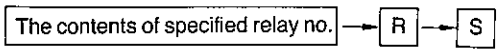
Remarks

When a logic line starts with an NO contact, this instruction indicates that a new string or sub-string of relay logic is being started at the specified program address.

Use this instruction for each logic line that starts with an NO contact.

To form substrings, combine this and other instructions such as AND or OR.

The LD instruction causes the contents (ON or OFF state) of the specified relay number to be stored in the result register. It also causes the existing contents of the result register to be transferred to the stack register.



Contents of data	LD	LD-NOT AND, AND-NOT OR, OR-NOT	OUT
I/O relay, internal auxiliary relay	0000 to 1907		0500 to 1807
Holding relay	HR 000 to 915		
Timer/counter	TIM/CNT00 to 47		
Temporary memory relay	TR 0 to 7	–	TR 0 to 7

OUTPUT (OUT)

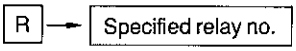
OUT (relay no.)

Symbol

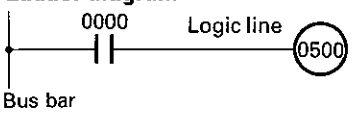
Function

Outputs result of a logical operation to a specified relay or shift register

Remarks
This instruction is used to output the result register contents (the result of a logical operation) to as specified relay. The specified relay may be either an output relay, internal auxiliary relay, holding relay (this is also referred to as "retentive relay") or temporary memory relay.



Ladder diagram



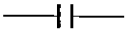
Coding chart

Address	Instruction	Data
0000	LD	0000
0001	OUT	0500

AND

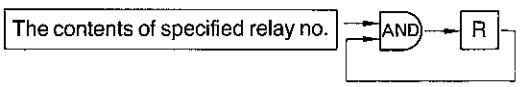
(relay no.)

Symbol

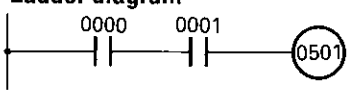


Function
Performs logical AND of register result (i.e., connects two or more contacts in series) with the address portion of AND instruction

Remarks
This instruction ANDs the result of a logical operation previously saved to the result register with the specified relay. This result of the logical AND operation is then stored in the result register.



Ladder diagram



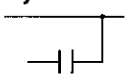
Coding chart

Address	Instruction	Data
0003	LD	0000
0004	AND	0001
0005	OUT	0501

OR

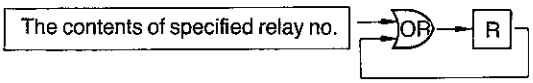
(relay no.)

Symbol

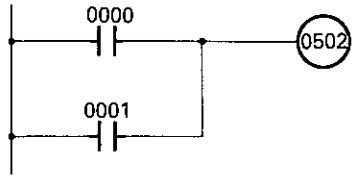


Function
Performs logical OR between a specified relay (i.e., connects two or more contacts in parallel) and the contents of the result register

Remarks
This instruction performs the logical OR of the contents of the result register with the specified relay. The result of the logical OR operation is then stored in the result register.



Ladder diagram



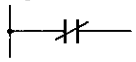
Coding chart

Address	Instruction	Data
0006	LD	0000
0007	OR	0001
0008	OUT	0502

LOAD-NOT (LD-NOT)

(address) LD NOT (relay no.)

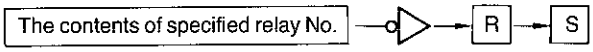
Symbol



Function
Starts operation of each logic line with an NC contact

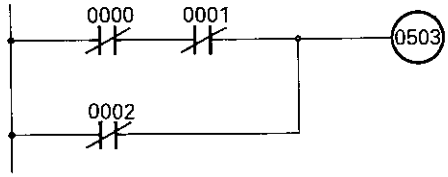
Remarks
When a logic line starts with an NC contact, this instruction is used in the place of the LD instruction.

The LD-NOT instruction inverts the contents of the specified relay number and stores it in the result register. Like the LD instruction, this instruction causes the existing contents of the result register to be transferred to the stack register.



To form substrings, combine this and other instructions such as AND or OR.

Ladder diagram



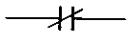
Coding chart

Address	Instruction	Data
0009	LD-NOT	0000
0010	AND-NOT	0001
0011	OR-NOT	0002
0012	OUT	0503

AND-NOT

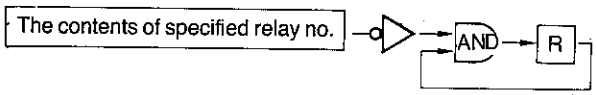
AND NOT (relay no.)

Symbol



Function
Connects NC contacts in series

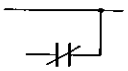
Remarks
This instruction inverts the contents of a specified relay and then carries out a logical AND operation with the contents of the result register. The result is then stored in the result register.



OR-NOT

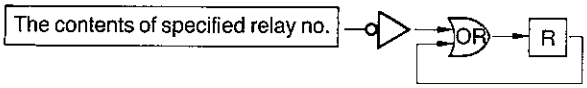
OR NOT (relay no.)

Symbol



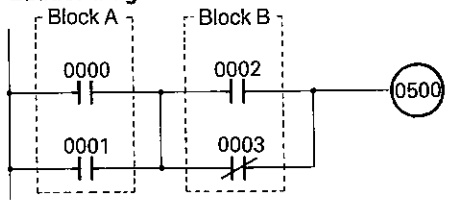
Function
Connects NC contacts in parallel

Remarks
This instruction inverts the contents of a specified relay and then carries out a logical OR operation with the contents of the result register. The result is then stored in the result register.



AND-LOAD (AND-LD)

Ladder diagram

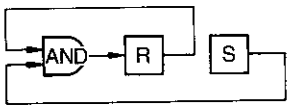


Function

Connects two blocks (A and B in this example) in series

Remarks

This instruction is used for interblock AND operation between two blocks. There is no limit to the number of blocks that can be connected in series by the AND-LD instruction.



Coding chart

Address	Instruction	Data
0200	LD	0000
0201	OR	0001
0202	LD	0002
0203	OR-NOT	0003
0204	AND-LD	–
0205	OUT	0500

Explanation

The LD0000 and OR0001 instructions cause the result of the logical OR operation in block A to be stored in the result register.

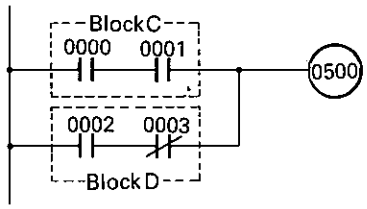
The LD0002 instruction in block B causes the result of the operation in block A to be transferred to the stack register and the result of the logical operation between the LD0002 and OR-NOT0003 instructions in block B to be stored in the result register.

The AND-LD instruction causes a logical AND operation to be performed between the result register and the stack register. The result of the logical AND operation is then stored in the result register.

OR-LOAD (OR-LD)

OR LD

Ladder diagram

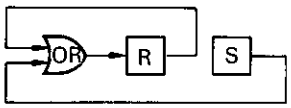


Function

Connects two blocks (C and D in this example) in parallel

Remarks

This instruction is used for interblock OR operation between two or more blocks. There is no limit to the number of blocks that can be connected in parallel by the OR-LD instruction.



Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND	0001
0202	LD	0002
0203	AND-NOT	0003
0204	OR-LD	–
0205	OUT	0500

Explanation

The LD0000 and AND0001 instructions cause the result of the logical AND operation in block C to be stored in the result register.

The LD0002 instruction in block D causes the result of the operation in block C to be transferred to the stack register and the result of the logical operation between the LD0002 and AND-NOT0003 instructions in block D to be stored in the result register.

The OR-LD instruction causes a logical OR operation to be performed between the result register and the stack register. The result of the logical OR operation is then stored in the result register.

TIMER (TIM)

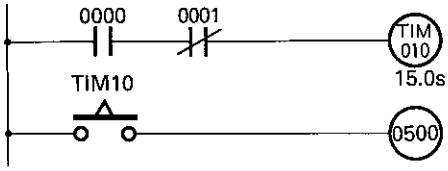
TIM (timer no.) (set value)

Symbol

Function

Performs ON-delay timer operation

Ladder diagram



Remarks

This instruction can be used as an ON-delay timer in the same manner as a relay circuit. The set time can be between 0 and 999.9 seconds. Timer numbers can be set from 00 to 47. Do not give timers and counters the same number.

The timer starts when the content of the result register is logical 1 and resets when the content of the register is logical 0.

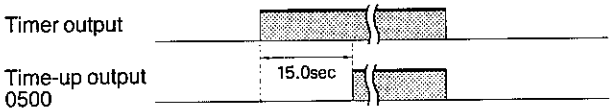
A time-up contact designates the timer number itself. Both NO and NC contacts can be used in the number required.

The timer works by decrementing, producing an output when the present value (the time remaining) becomes 0000. When input to the timer is turned off, the timer is reset and the present value returns to the preset time. The timer output is transmitted externally through an output relay as shown in the circuit example.

Coding chart

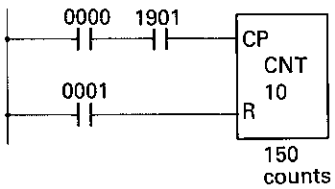
Address	Instruction	Data
0200	LD	0000
0201	AND-NOT	0001
0202	TIM	10*
		# 0150**
0203	LD	TIM 10
0204	OUT	0500

*Timer number
**Set time value



The set value for the TIM/CNT instruction cannot be set on a channel basis.

Ladder diagram



Memory retentive timer

If a power failure occurs, the timer is reset and the present value returns to the preset value. Therefore, if it is necessary that the current value of the timer be retained in the event of a power failure, a memory retentive timer circuit must be used for the program. The example circuit is configured by combining the PC's clock and the counter instruction with Special Auxiliary Relay 1901 used to generate a 0.2sec pulse.

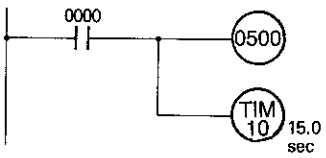
Coding chart

Instruction	Data
LD	0000
AND	1901
LD	0001
CNT	10
	# 0150

Consecutive OUT instruction and TIM instruction

Using this circuit, a LD instruction produces an output and simultaneously turns on a timer.

Ladder diagram



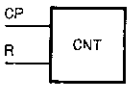
Coding chart

Instruction	Data
LD	0000
OUT	0500
TIM	10
	# 0150

COUNTER (CNT)

CNT (counter no.) (set value)

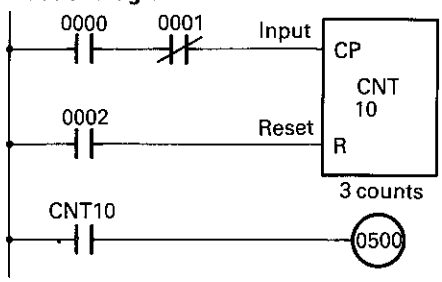
Symbol



Function

Performs down-counter operation

Ladder diagram



Remarks

This instruction can be used as a preset counter in the same manner as a relay circuit. The preset value can be between 0 and 9999 counts. Counter numbers can be assigned from 00 to 47. Do not give counters and timers the same number.

The counter resets when the contents of the result register are logical 1 and starts counting when the contents of the register are logical 0. The count input is provided from the stack register.

A count-up contact designates the counter number. NO and NC contacts can be used in combination, and in any quantity.

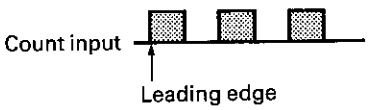
The counter decrements, producing an output when the count value becomes 0000. The present value of the counter returns to the preset value when a reset input is applied. The counter output is transmitted to an external device through an output relay as shown in the ladder diagram.

After the preset count is finished, any additional counts are ignored.

The program for a counter must be entered in the order of a count input circuit, a reset input circuit, and a counter coil.

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND-NOT	0001
0202	LD	0002
0203	CNT	10
		# 0003
0204	LD	CNT 10
0205	OUT	0500



At the leading edge (from OFF to ON) of a count input signal, the counter decrements by a value of 1.

When a count input and a reset input are simultaneously applied, the reset input takes precedence. No counting will be performed after that even if the reset input is removed.

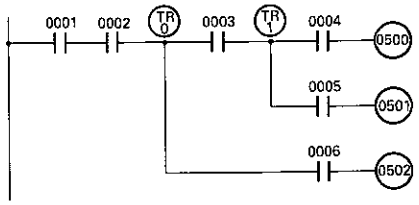
If a power failure occurs, the current count value (the number of counts remaining) is retained in memory and the counter is not reset.

The set value for the TIM/CNT instruction cannot be set on a channel basis.

TEMPORARY MEMORY RELAY (TR)

OUT TR (relay number) or LD TR (relay number)

Ladder diagram



Function

Serves as temporary memory relay

Remarks

To program a temporary memory relay, the TR instruction must be used with an OUT or LD instruction.

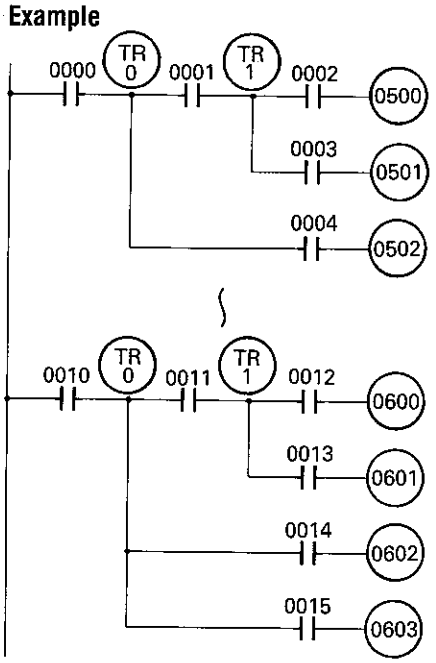
The TR instruction is used when a ladder diagram cannot be programmed with the interlock instructions. In an output branching circuit consisting of multiple blocks, temporary memory relays are used at each point. These temporary relays cannot be used in duplicate within the same block but can be used within different blocks.

Relay numbers can be from 0 to 7.

Coding chart

Address	Instruction	Data
0200	LD	0001
0201	AND	0002
0202	OUT	TR 0
0203	AND	0003
0204	OUT	TR 1
0205	AND	0004
0206	OUT	0500
0207	LD	TR 1
0208	AND	0005
0209	OUT	0501
0210	LD	TR 0
0211	AND	0006
0212	OUT	0502

Example



Special Instructions

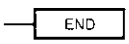
In addition to the instructions with their own keys, additional special instructions are supplied in the C20 that are available using the FUN (function) key. To write one of these instructions into a program, press

FUN (numeric key) (numeric key) WRITE

END

FUN A 0 B 1

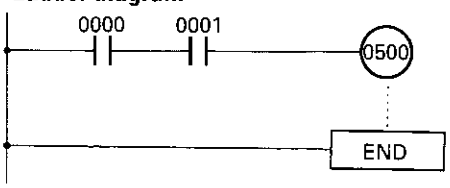
Symbol



Function

Indicates the end of the program

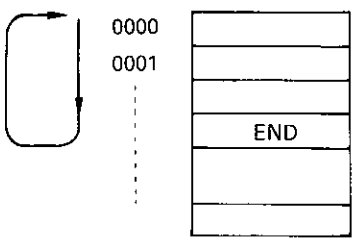
Ladder diagram



Remarks

This is always the last instruction written into a program and is used to indicate the program's end. An error message (NO END INST) is displayed when attempting to run or monitor the program if this instruction is missing.

The CPU scans the program data from address 0000 to the address with the END instruction in the sequence shown.



This instruction can be used to perform test runs of each completed circuit. To do this, place the END instruction at the end of each circuit in the series. After testing the circuit, delete the END instruction and test the next circuit in a similar manner.

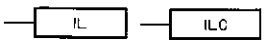
Coding chart

Address	Instruction	Data
0000	LD	0000
0001	AND	0001
0002	OUT	0500
0700	END(01)	–

INTERLOCK (IL) and INTERLOCK CLEAR (ILC)

FUN A 0 C 2 FUN A 0 D 3

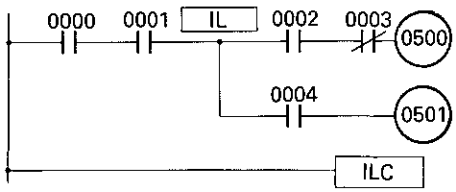
Symbol

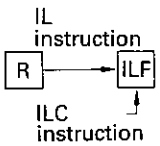


Function

Branches circuit to OUT instructions

Ladder diagram





Remarks

These two instructions are used in pairs when a circuit is branched to more than one OUT instruction. If the IL and ILC instructions are not used in pairs, an error results during program check.

All relay coils between these two instructions are set or reset depending on the current status of the relays.

When the interlock flip-flop (ILF) in the CPU is 0, the contents of the result register are fixed to 0. Consequently, as long as the ILF is set to 1, the output relay remains OFF

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND	0001
0202	IL (02)	–
0203	LD	0002
0204	AND-NOT	0003
0205	OUT	0500
0206	LD	0004
0207	OUT	0501
0208	ILC (03)	–

The IL instruction causes the contents of the result register to be transferred to the ILF. Therefore, the ILF is set to 0 if the contents of the result register are 0. The ILC instruction causes the ILF to be set to 1, regardless of the contents of the result register.

When the IL condition is OFF (for instance, when 0000 or 0001 is OFF in the above example), the state of each relay between the IL and ILC instruction is:

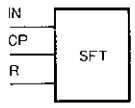
Output relay, internal auxiliary relay	OFF
Timer	Reset
Counter, shift register, holding relay	Holds present state

When the IL condition is ON, the state of each relay is the same as that in an ordinary relay circuit when the IL/ILC instructions are not used.

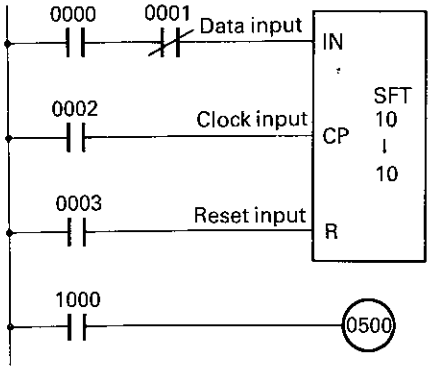
SHIFT REGISTER (SFT)

FUN | B 1 | A 0 (start channel no.) (end channel no.)

Symbol



Ladder diagram



Function

Acts as a serial input shift register

Remarks

The shift register must be programmed in the order of data input, clock input, reset input, and an SFT instruction (from the start channel to the end channel).

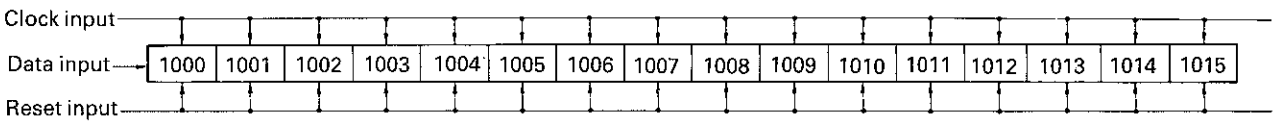
Each SFT instruction must be specified in units of 16 bits. In the circuit example, 16 bits from 1000 to 1015 are transferred.

The 16-bit contents of the shift register can be output bit by bit using the relay numbers of the specified channel.

When a reset input is applied to the shift register, all 16 bits are reset together.

The data are shifted at the leading edge of the input clock.

If the holding relay area is used, the data are retained during power failure until a clock or reset input is applied.



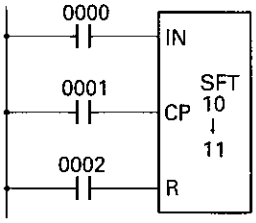
Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND-NOT	0001
0202	LD	0002
0203	LD	0003
0204	SFT (10)	10
		10
0205	LD	1000
0206	OUT	0500

Contents of data (start channel and end channel nos.)

Output relay, internal auxiliary relay	05 to 17
Holding relay	HR0 to 9

Ladder diagram



When more than 16-bit shift register is needed

If a shift register should need to exceed 16 bits, a shift register circuit can be configured by combining two or more stages of 16-bit shift registers.

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	LD	0001
0202	LD	0002
0203	SFT (10)	10
		11

The circuit configuration shown above is of a 32-bit shift register from 1000 to 1115.

The data for the SFT instruction must be input with the upper-stage SFT data which are less than or equal to the lower-stage SFT data and are within the same relay area.

LATCHING RELAY (KEEP)

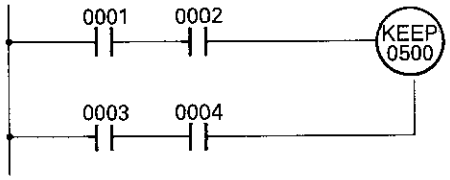
FUN | B 1 | B 1 (relay no.)

Symbol

Function

Acts as latching relay

Ladder diagram



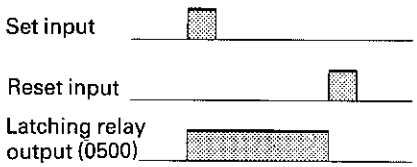
Remarks

This instruction can be used to create a latching relay which is used in the same manner as a relay circuit.

The latching relay operates when the content of the result register is logical 0 and the content of the stack register is logical 1. The relay releases when the content of the result register is logical 1.

The latching relay program must be entered in the order of a set input circuit, a reset input circuit, and a latching relay coil.

When both a set input and a reset input are applied at the same time, the reset input takes precedence.



If the holding relay is used as a latching relay, data in the memory are retained during power failure until a set or reset input is applied.

Coding chart

Address	Instruction	Data
0200	LD	0001
0201	AND	0002
0202	LD	0003
0203	AND	0004
0204	KEEP (11)	0500

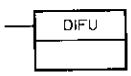
Contents of data

Output relay, internal auxiliary relay	0500 to 1807
Holding relay	HR000 to 915

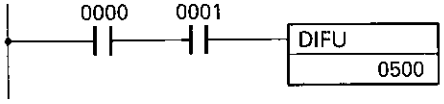
DIFFERENTIATION UP (DIFU)

FUN B 1 D 3 (relay no.)

Symbol



Ladder diagram



Function

Causes relay to operate at leading edge of the input for one scan time

Remarks

This instruction is used to output the differentiation of an input condition to a specified relay for one scan time.

The instruction must be set so that the output is issued for one scan time at the leading edge of the result register, that is, at the point when the register's level turns from 0 to 1.

When used with the DIFFERENTIATION DOWN (DIFD) instruction, the maximum number of DIFU and DIFD instructions that can be programmed together is 48.

The differentiation instructions perform their operations in response to changes in input after the PC starts operating.

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND	0001
0202	DIFU (13)	0500

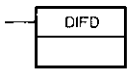
Contents of data

Output relay	0500 to 1807
Holding relay	HR000 to 915

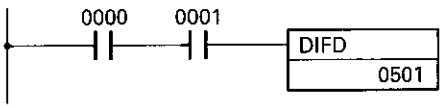
DIFFERIENTIATION DOWN (DIFD)

FUN B 1 E 4 (relay no.)

Symbol



Ladder diagram



Function

Causes relay to operate at the trailing edge of input for one scan time

Remarks

This instruction is used to output the differentiation of an input condition to a specified relay for one scan time.

The instruction must be set so that the output is issued for one scan time at the trailing edge of the result register, that is, at the point when the register's level turns from 1 to 0.

When used with the DIFFERENTIATION UP (DIFU) instruction, the maximum number of DIFD and DIFU instructions that can be programmed together is 48.

The differentiation instructions perform their operations in response to changes in input after the PC starts operating.

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND	0001
0203	DIFD (14)	0501

Contents of data

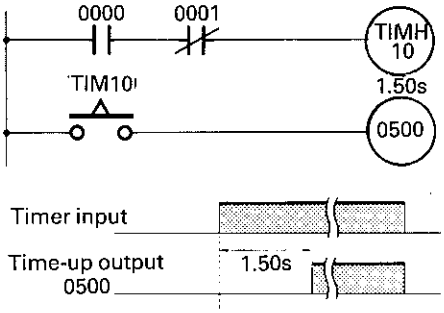
Output relay	0500 to 1807
Holding relay	HR000 to 915

HIGH SPEED TIMER (TIMH)

FUN | B 1 | F 5 (timer no.) (set value)

Symbol

Ladder diagram



Function

Performs high-speed timer operation

Remarks

This instruction can be used as a high-speed ON-delay timer in the same manner as a relay circuit. The set time can be between 00.00sec to 99.99sec, in increments of 0.01sec. Timer relay numbers can be set from 00 to 47. Do not give timers and counters the same number.

The operating conditions and contents of the operation are the same as the timer instruction.

If the scan time exceeds 10msec, the timing operation may be inaccurate.

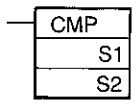
Coding chart

Address	Instruction	Data
0200	LD	0000
0201	AND-NOT	0001
0202	TIMH (15)	10
		# 0150
0204	LD	TIM 10
0205	OUT	0500

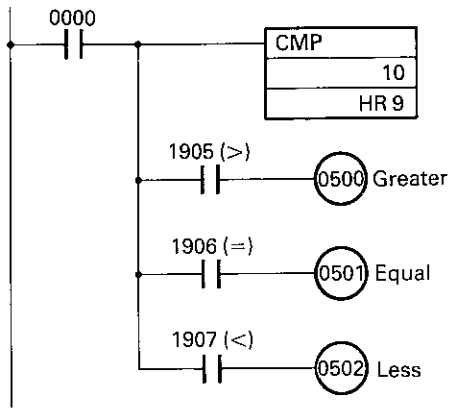
COMPARE (CMP)

FUN | C 2 | A 0 (S1) (S2)

Symbol



Ladder diagram



Function

Compares a channel data or a four-digit constant against another channel data

Remarks

The CMP instruction is used to compare a 16-bit channel data or a hexadecimal four-digit (16-bit binary) constant (S1) against another 16-bit channel data (S2).

As a result of this comparison:

1905 turns ON if the S1 is greater than the S2
1906 turns ON if the S1 is equal to the S2
1907 turns ON if the S1 is less than the S2

Coding chart

Address	Instruction	Data	
0200	LD		0000
0201	OUT	TR	0
0202	CMP (20)		–
0202			10
0202		HR	9
0203	LD	TR	0
0204	AND		1905
0205	OUT		0500
0206	LD	TR	0
0207	AND		1906
0208	OUT		0501
0209	LD	TR	0
0210	AND		1907
0211	OUT		0502

Contents of data (S1, S2)

Input/output relay, internal auxiliary relay	00 to 19
Holding relay	HR0 to 9
Timer/counter	TIM/CNT00 to 47
Constant	#0000 to FFFF

Note: When the result register is logical 1, a CMP instruction is executed at each scanning. To execute it only one time, program a differentiation circuit for the input.

Nothing is executed when the content of the result register is logical 0. Therefore, the compare result area of special auxiliary relays 1905 to 1907 holds the previous status, and on execution of an END instruction, all of these relays are cleared to 0. When the contents of the result register are logical 1, the CMP instruction is executed.

In the circuit example, when the program is executed, the 16-bit data of 10CH (1000 to 1015) are compared with the 16-bit data of HR9CH (HR900 to 915), and the result is output to the result area of special auxiliary relays 1905 to 1907.

Input/output relay 10CH

2^0	1000	$\times 16^0$
2^1	1001	
2^2	1002	
2^3	1003	
2^0	1004	$\times 16^1$
2^1	1005	
2^2	1006	
2^3	1007	
2^0	1008	$\times 16^2$
2^1	1009	
2^2	1010	
2^3	1011	
2^0	1012	$\times 16^3$
2^1	1013	
2^2	1014	
2^3	1015	

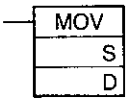
Result of comparison

	1905	1906	1907
10CH>HR9CH	1	0	0
10CH=HR9CH	0	1	0
10CH<HR9CH	0	0	1

A constant is compared with the four-digit hexadecimal (binary 16-bit) data.

MOVE (MOV)

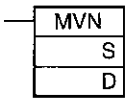
Symbol



FUN C 2 B 1 (S) (D)

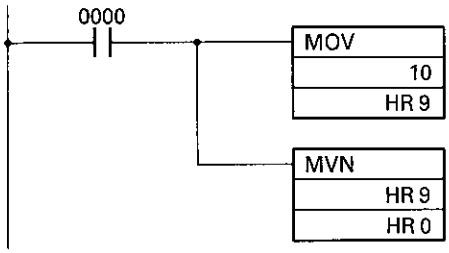
MOVE NOT (MVN)

Symbol



FUN C 2 C 2 (S) (D)

Ladder diagram



Function

MOV instruction transfers a channel data or a four-digit constant (16 bits) (S) to a specified channel (D)

MVN instruction inverts a channel data or a four-digit constant and transfers it to a specified channel

Remarks

The MOV instruction is used to transfer 16-bit channel data or a hexadecimal four-digit (16-bit binary) constant to a specified channel. The MVN inverts the channel data and then performs the same function as the MOV instruction.

Coding chart

Address	Instruction	Data
0200	LD	0000
0201	MOV (21)	–
		10
		HR 9
0202	MVN (22)	–
		HR 9
		HR 0

Contents of data

	S	D
Input/output relay, internal auxiliary relay	00 to 19	05 to 17
Holding relay	HR 0 to 9	
Timer/counter	TIM/CNT 00 to 47	–
Constant	#0000 to FFFF	–

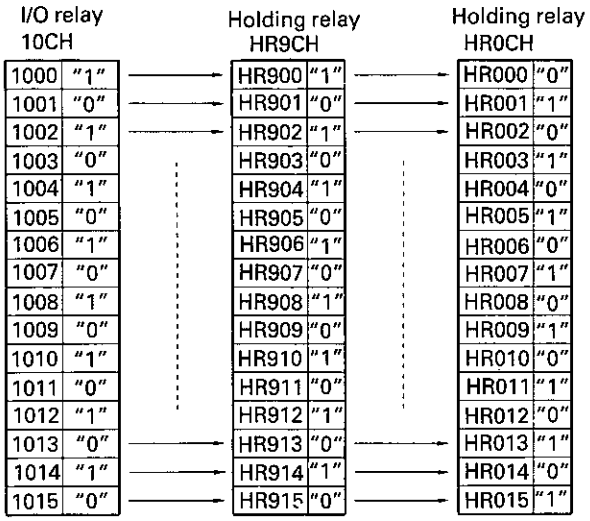
Note: If the transferred data are all 0, special auxiliary relay 1906 turns ON.

When the result register is logical 1, a MOV or MVN instruction is executed at each scanning. To execute it only once, program a differentiating circuit for the input.

Explanation

Nothing is executed when the content of the result register is logical 0. When the content of the result register is logical 1, transfer of data or of inverted data is performed.

When the program of the circuit example is performed, the 16-bit data of 10CH (1000 to 1015) is transferred to HR9CH (HR900 to HR915), then inverted to be further transferred to HR0CH (HR000 to HR015)



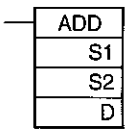
The result of the transfer will be:
HR9CH = 0 → 1906 = 1
(HR0) ≠ 0 → 1906 = 0

In constant designation, hexadecimal four-digit (binary 16-bit) data is either transferred or inverted and then transferred.

ADD

FUN | D 3 | A 0 (S1) (S2) (D)

Symbol



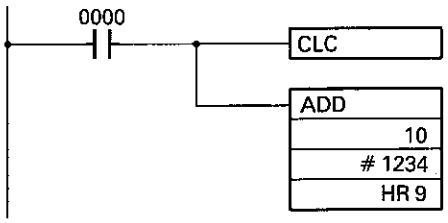
Function

Performs BCD addition of a channel data (S1) or four-digit constant to a specified channel data (S2) and then outputs the result to a specified channel (D)

Remarks

This instruction is used to do addition between two specified four-digit BCD data.

Ladder diagram



Coding chart

Address	Instruction	Data
0200	LD	0000
0201	CLC	–
0202	ADD (30)	–
		10
		# 1234
		HR 9

Contents of data

	S1, S2	D
Input/output relay, internal auxiliary relay	00 to 19	05 to 17
Holding relay	HR 0 to 9	
Timer/counter	TIM/CNT00 to 47	–
Constant	#0000 to 9999	–

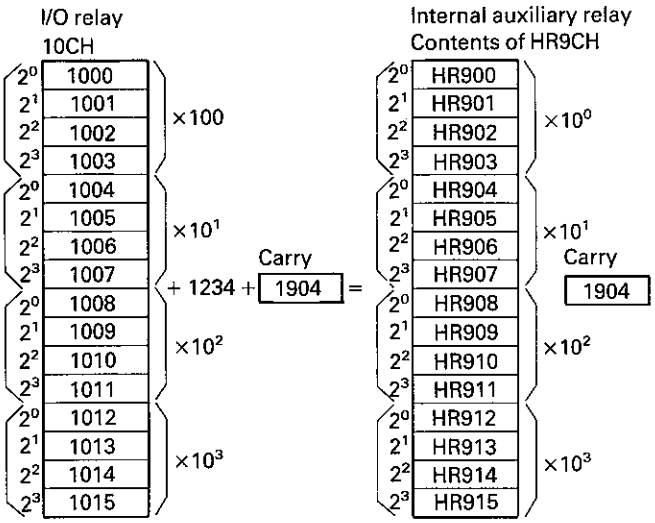
Notes: Before an ADD instruction is executed, a CLC instruction must be programmed to clear the carry flag (1904).

When the result register is logical 1, a BCD addition is executed at each scanning. To execute it only one time, program a differentiating circuit for the input.

Explanation

Nothing is executed when the content of the result register is logical 0. When the content of the result register is logical 1, addition of a four-digit BCD data to a four-digit BCD data including a carry (1904) is executed. If the result of the addition is 0000, 1906 is turned ON, and if there is a carry in the result, 1904 is turned ON.

In the circuit program, the 16-bit contents of 10CH (1000 to 1015) are added in units of four BCD digits to the 16-bit contents of the four-digit constant 1234, including a carry (1904), and the result of the addition is output to the 16-bit locations of HR9 CH (HR900 to HR915). If there is a carry in the result, 1904 is turned ON, and if the result of the addition is 0000, 1906 is turned ON.

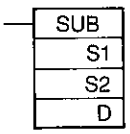


Before executing an ADD instruction, the carry register (1904) must always be cleared using a CLEAR CARRY instruction. This is not required in multistage addition.

SUBTRACT (SUB)

FUN | D 3 | B 1 (S1) (S2) (D)

Symbol



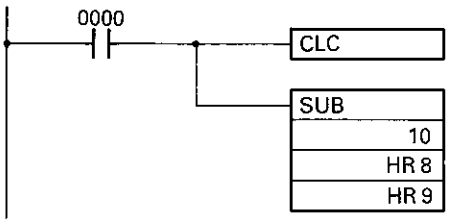
Function

Performs BCD subtraction of a channel data or four-digit constant (S2) from a specified channel data (S1) and then outputs the result to a specified channel (D)

Remarks

The SUB instruction is used to execute BCD subtraction between two specified four-digit data.

Ladder diagram



Coding chart

Address	Instruction	Data
0200	LD	0000
0201	CLC	–
0202	SUB (31)	–
		10
		HR 8
		HR 9

Contents of data

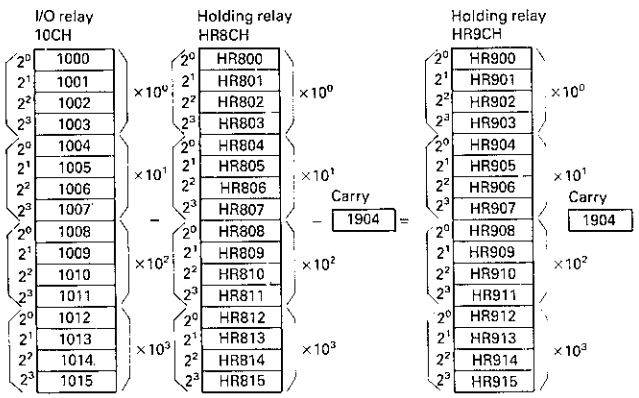
	S1, S2	D
Input/output relay, internal auxiliary relay	00 to 19	05 to 17
Holding relay	HR 0 to 9	
Timer/counter	TIM/CNT00 to 47	–
Constant	#0000 to 9999	–

Notes: Before executing a SUB instruction, a CLC instruction must be programmed to clear the carry flag (1904).
When the result register is logical 1, a BCD subtraction is executed with each scan. To execute it only once, program a differentiating circuit for the input.

Explanation

Nothing is executed when the content of the result register is logical 0. When the content of the result register is logical 1, four-digit BCD subtraction, including a carry (1904) is executed. If the result of the addition is 0000, 1906 is turned ON, and if there is a carry in the result, 1904 is turned ON.

In the circuit program, the 16-bit contents of HR8CH (HR800 to HR815), including a carry (1904), are subtracted in units of four BCD digits from the 16-bit contents of 10CH (1000 to 1015), and the result of the subtraction is output to the 16-bit locations of HR9CH (HR900 to HR915). If there is a carry in the result, 1904 is turned ON, and if the result of the subtraction is 0000, 1906 is turned ON.



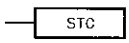
Before executing a SUB instruction, the carry register (1904) must always be cleared using a CLEAR CARRY (CLC) instruction. This can be omitted in multistage subtraction.

The CPU checks whether the data for BCD subtraction are in four BCD digits. If not, an error occurs, special auxiliary relay 1903 turns ON, and the program does not operate.

SET CARRY (STC)

FUN E 4 A 0

Symbol



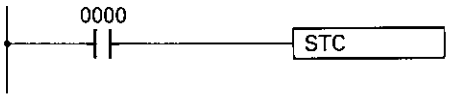
Function

This instruction sets the carry flag (CY) to 1

Remarks

Nothing is executed when the content of the result register is logical 0.

Ladder diagram



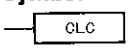
Coding chart

Address	Instruction	Data
0200	LD	0000
0201	STC (40)	–

CLEAR CARRY (CLC)

FUN E 4 B 1

Symbol



Function

Clears the carry flag (CY) to 0

Remarks

Nothing is executed when the content of the result register is logical 0.

Ladder diagram

Coding chart

Address	Instruction	Data
0202	LD	0001
0203	CLC (41)	–

Appendix C Scan time

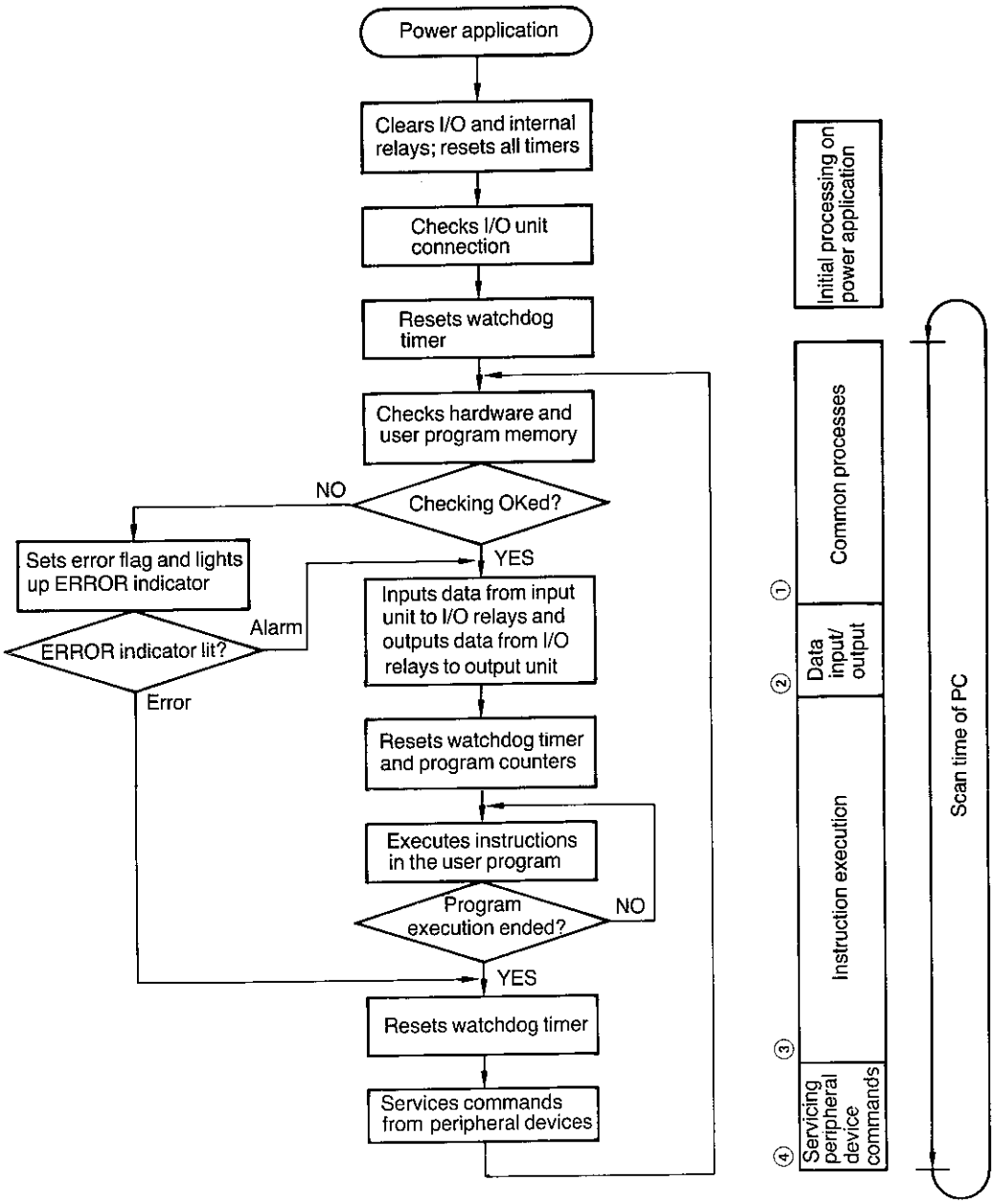
Overview

When a programmable controller operates, that is, when it executes its program to control an external system, a series of operations are performed inside the PC. These internal operations can be broadly classified into four categories:

Common processes such as resetting the watchdog timer and checking the user program memory
Data input and output
Instruction execution
Peripheral device command servicing

Scan time

The total time required for a programmable controller to perform all of these internal operations is called *scan time*. The following flowchart and diagram illustrate these internal operations.



Scan time

The required scan time differs depending on the configuration of the system, the number of I/O units, the number of special instructions, and whether peripheral devices are connected.

The following table shows the time required for each internal operation.

	Process	Execution time and variables
①	Resetting of watchdog timer Checking of user program memory Checking of I/O bus	1.07ms (fixed)
②	Reading data input from input unit to I/O relays Writing data (result of instruction execution) to output unit from I/O relays	1.04ms+0.33ms×N where, N = 1 when a 28-point I/O unit or I/O link unit is used or N = 2 when a 56-point I/O unit is used
③	Execution of instructions in the user program	Total time for executing instructions. The time required for execution differs depending on the instruction. (Refer to the section below, List of instruction execution times.)
④	Servicing of commands input from peripheral devices such as programming console, graphic programming console, etc.	1.1ms

The scan time can be obtained by adding 1, 2, 3, and 4 in this table. An adequately short period of a scan time is important to ensure an efficiently error-free operation. This is particularly important when using the high-speed timer, which malfunctions if the scan time exceeds 10ms.

List of instruction execution times

Basic instructions

Instruction	Execution time*
LOAD	12
LOAD-NOT	12
AND	11.5
AND-NOT	11.5
OR	11.5
OR-NOT	11.5
AND-LOAD	4
OR-LOAD	4
OUT	17.5
TIMER	80
COUNTER	76

* In microseconds

Special instructions

The *channel* includes the input/output relay, internal auxiliary relay, and holding relay areas.

Instruction	Execution time*	
NOP	2	
END	1.1	
INTERLOCK	4	
INTERLOCK CLEAR	3	
SHIFT REGISTER	102	(when one channel is shifted)
	248	(when 13 channels are shifted)
LATCHING RELAY	20	
DIFFERENTIATION UP	59	
DIFFERENTIATION DOWN	57	
HIGH-SPEED TIMER	81	
COMPARE	105	(comparison of a constant with a channel)
	195	(comparison of a timer/counter with a channel)
MOVE	89	(transfer of a constant to a channel)
	178	(transfer of a timer/counter to a channel)
MOVE NOT	90	(transfer of a constant to a channel)
	179	(transfer of a timer/counter to a channel)
ADD	210	(addition of two channels and then transfer of the result to a channel)
	334	(addition of a timer/counter and a constant and then transfer of the result to a channel)
SUBTRACT	217	(addition of two channels and then transfer of the result to a channel)
	340	(addition of a timer/counter and a constant and then transfer of the result to a channel)
SET CARRY	16	
RESET CARRY	16	

* In microseconds

Scan time calculation examples

When calculating the total scan time required by the PC to perform its internal operations, the system configuration must be taken into consideration. This means factoring into the calculation such things as the number of input/output units and special instructions, and whether peripheral devices are employed.

For example, let's see how to calculate the scan time when the PC is used by itself and with no special instructions. For the sake of simplicity, it is assumed the user program consists of 512 addresses and only uses the LD and OUT instructions.

Of the four parameters for the calculation (the times required for common processes, data input/output, instruction execution, and servicing peripheral device commands), how long it takes to perform the common processes and service peripheral device commands are fixed at 1.07ms and 1.1ms, respectively.

This leaves only the remaining two parameters to be calculated. The time required for data input/output can be determined by using the following equation where the variable N can be 0, 1, or 2 depending on whether an I/O unit is connected to the PC and which I/O unit it is.

Data input/output time = 1.04ms + 0.33ms × N

Since in this example we assume that the C20 is used by itself, the value of constant N equals 0. Thus,

1.04ms + (0.33ms × 0) = 1.04ms

Next, the time required for executing all the instructions in the program must be known. To do this, obtain the average execution time and then multiply that time by the number of addresses in the program.

Because only the LD and OUT instructions are used in this example, the average instruction execution time can be obtained by adding the execution times for the two types of instructions and then dividing the sum by two. To find the execution times for these two instructions, refer to "List of instruction execution times" on the preceding pages. We see from this list that the execution time of the LD instruction is 12μs and that of the OUT instruction is 17.5μs. Accordingly, the average execution time is:

(12μs + 17.5μs)/2 = 14.75μs

Now it is possible to obtain the total instruction execution time by multiplying 14.75μs by 512, the number of program addresses.

Total execution time = 512 addresses × 14.75μs = 7.55ms

Adding the four parameters provides the total scan time:

1.07ms + 1.04ms + 7.55ms + 1.1ms = 10.76ms

Notice that if no peripheral device is connected to the PC, the fourth parameter in this equation is 0ms.

Next, let's consider the scan time for a system that has a 56-point expansion I/O unit connected and the user program consists of 1194 addresses. Again, it is assumed that only the LD and OUT instructions are used in the program.

As before, there are two known factors: common process time and peripheral device command servicing time, which are 1.07ms and 1.1ms, respectively. The variable N of the equation for calculating the data input/output time is 2 because a 56-point expansion I/O unit is used. Therefore, the data input/output time is:

1.04ms + (0.33ms × 2) = 1.7ms

Because the program has 1194 addresses and uses only the LD and OUT instructions, the total instruction execution time is:

1194 addresses × 14.75μs = 17.61ms

Therefore, the scan time is:

1.07ms + 1.7ms + 17.61ms + 1.1ms = 21.48ms

Input/output response time of PC

Response time means the time from when the PC receives an input signal until it outputs a control signal. This response time varies. If the PC receives an input signal immediately after one scan time has finished and if the input ON-delay time of the PC is minimum, the response time is minimized. If those two factors exist, the response time can be obtained by simply adding together the minimum input ON-delay, output ON-delay, and scan times of the PC.

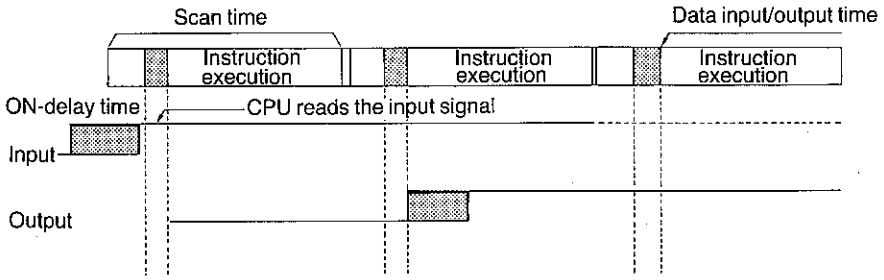
However, if the PC receives the input signal when one scan time has just started and the input ON-delay time is the maximum value, the response time is longer. This is because the CPU must wait until the end of the preceding scan time before the input signal can be accepted. Therefore, the maximum response time is the sum of the maximum input ON-delay, output ON-delay, and two scan times.

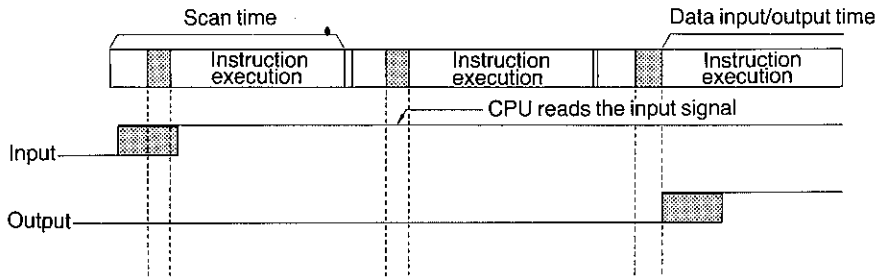
The timing charts of the minimum and maximum response times required for the PC to execute the following program are shown next.

Response time calculation example

Where the input ON-delay time is 10 (minimum) to 20ms (maximum), the scan time is 10ms, and the output ON-delay time is 15ms, the minimum and maximum response times are:

Minimum response time = 10ms + 10ms + 15ms = 35ms
Maximum response time = 20ms + (10ms × 2) + 15ms = 55ms





Appendix D Assignment of I/O channel and relay numbers

I/O channel assignment

The input channel of the CPU is fixed at channel 00 while the output channel is fixed at channel 05.

When an expansion I/O unit, I/O link unit, or both are connected to the CPU, the I/O channel of the connected units are automatically assigned and registered in the CPU.

For example, when a 28-point expansion I/O unit is connected, input channel 01 is automatically assigned to the expansion I/O unit and registered in the CPU. As the output channel number of the expansion unit, 06 is automatically assigned and registered.

Relay assignment

The input/output signals (devices) are connected to the input/output terminals on the CPU. Because the CPU uses the numbers assigned to the input/output terminals when executing the program, assignment and management of the input/output terminal numbers are required and must be correctly performed.

The numbers assigned to each relay are listed in the following tables. Use these to keep a record of which relays have been used.

Assignment of I/O channel and relay numbers

Relay numbers

Name	No. of points	Relay number									
Input relay	80	0000 to 0415									
		00CH		01CH		02CH		03CH		04CH	
		00	08	00	08	00	08	00	08	00	08
		01	09	01	09	01	09	01	09	01	09
		02	10	02	10	02	10	02	10	02	10
		03	11	03	11	03	11	03	11	03	11
		04	12	04	12	04	12	04	12	04	12
		05	13	05	13	05	13	05	13	05	13
		06	14	06	14	06	14	06	14	06	14
		07	15	07	15	07	15	07	15	07	15
Output relay	60	0500 to 0915									
		05CH		06CH		07CH		08CH		09CH	
		00	08	00	08	00	08	00	08	00	08
		01	09	01	09	01	09	01	09	01	09
		02	10	02	10	02	10	02	10	02	10
		03	11	03	11	03	11	03	11	03	11
		04	12	04	12	04	12	04	12	04	12
		05	13	05	13	05	13	05	13	05	13
		06	14	06	14	06	14	06	14	06	14
		07	15	07	15	07	15	07	15	07	15
Internal auxiliary relay	136	1000 to 1807									
		10CH		11CH		12CH		13CH		14CH	
		00	08	00	08	00	08	00	08	00	08
		01	09	01	09	01	09	01	09	01	09
		02	10	02	10	02	10	02	10	02	10
		03	11	03	11	03	11	03	11	03	11
		04	12	04	12	04	12	04	12	04	12
		05	13	05	13	05	13	05	13	05	13
		06	14	06	14	06	14	06	14	06	14
		07	15	07	15	07	15	07	15	07	15
		15CH		16CH		17CH		18CH			
		00	08	00	08	00	08	00			
		01	09	01	09	01	09	01			
		02	10	02	10	02	10	02			
		03	11	03	11	03	11	03			
		04	12	04	12	04	12	04			
		05	13	05	13	05	13	05			
		06	14	06	14	06	14	06			
		07	15	07	15	07	15	07			
Holding relay (retentive relay)	160	HR000 to 915									
		00CH		01CH		02CH		03CH		04CH	
		00	08	00	08	00	08	00	08	00	08
		01	09	01	09	01	09	01	09	01	09
		02	10	02	10	02	10	02	10	02	10
		03	11	03	11	03	11	03	11	03	11
		04	12	04	12	04	12	04	12	04	12
		05	13	05	13	05	13	05	13	05	13
		06	14	06	14	06	14	06	14	06	14
		07	15	07	15	07	15	07	15	07	15
		05CH		06CH		07CH		08CH		09CH	
		00	08	00	08	00	08	00	08	00	08
		01	09	01	09	01	09	01	09	01	09
		02	10	02	10	02	10	02	10	02	10
		03	11	03	11	03	11	03	11	03	11
		04	12	04	12	04	12	04	12	04	12
		05	13	05	13	05	13	05	13	05	13
		06	14	06	14	06	14	06	14	06	14
		07	15	07	15	07	15	07	15	07	15

Timer/counter numbers

Name	No. of points	Timer/counter number						
		TIM/CNT00 to 47						
Timer/counter	48	00	08	16	24	32	40	
		01	09	17	25	33	41	
		02	10	18	26	34	42	
		03	11	19	27	35	43	
		04	12	20	28	36	44	
		05	13	21	29	37	45	
		06	14	22	30	38	46	
		07	15	23	31	39	47	

Input relays

The CPU has 16 input relay points (one input channel). The number of input points can be increased to a maximum of 80 by the addition of expansion I/O units. Because one channel equals 16 points, this means that a maximum of five channels (from channel 00 to 04) are available.

The data from the SYSBUS are received by the input channel relays assigned to the I/O link unit.

Output relays

As with the input relays, the CPU has one output channel consisting of 16 relay points. However, of these 16 points, numbers 12 to 15 are internal auxiliary relays used to carry out CPU internal processes. For this reason, the number of output relays the CPU actually possesses is 12.

When expansion I/O units are connected to the CPU, a maximum of 60 output relays are available for a total of five channels, 05 to 09. The statuses of all the output channel relays assigned to the I/O link unit are transferred to the SYSBUS when an I/O link unit is employed.

Internal auxiliary relays

The CPU has 136 internal auxiliary relays (No. 1000 to 1807) that constitute channels 10 to 18.

Holding relays (retentive relays)

The CPU has 160 holding relays (No. HR000 to 915) that constitute holding relay channels 0 to 9. The holding relay retains data during power failure.

Timers/counters

The CPU has 48 points of timers/counters, TIM/CNT00 to 47, that can be used for either timers or counters. Timers and counters cannot be assigned the same number.

Temporary memory relays

The CPU has eight temporary memory relay points (TR0 to 7).

Special auxiliary relays

The CPU has 16 special auxiliary relays, some of which operate or release according to internal conditions controlled by the hardware irrespective of the statuses of the I/O devices. Each of these special auxiliary relays functions as follows:

Relay 1808
This relay operates when a battery failure occurs. An alarm signal indicating a battery failure can be output to an external device by programming a circuit incorporating the contact of this relay.

Relay 1809
Normally OFF

Relay 1810
Normally OFF

Relay 1811
Normally OFF

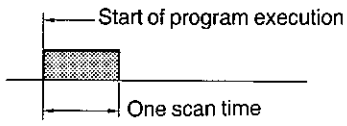
Relay 1812
Normally OFF

Relay 1813
Normally ON.

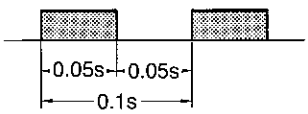
Relay 1814
Normally OFF

Relay 1815
This relay is turned ON for one scan time at the start of program execution.

A *scan time* is the time required for the PC to execute the user program once starting from address 0000 to the program's end instruction.

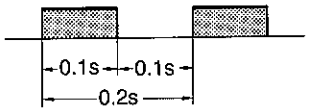


Relay 1900
This relay is used to generate a 0.1s clock. When used in conjunction with a counter, it functions as a timer that can retain its present value during power failure.

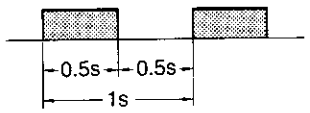


Note: The ON time of a 0.1s clock is 50ms. If a longer time is required for program execution, the CPU may fail to read the clock.

Relay 1901
This relay is used to generate a 0.2s clock. When used in conjunction with a counter, it functions as a long-time timer that can retain its present value during power failure.



Relay 1902
This relay is used to generate a 1s clock. When used in conjunction with a counter, it functions as a long-time timer that can retain its present value during power failure. The relay output can be also used as a flicker signal.



Relay 1903
This relay turns ON when the result of an arithmetic operation is not output in BCD form.

Relay 1904
This relay serves as a carry flag and operates or releases according to the result of an arithmetic operation. It can be forcibly turned ON by the SET CARRY (STC) instruction and turned OFF by the CLEAR CARRY (CLC) instruction.

Relay 1905
This relay turns ON if the result of the COMPARE (CMP) instruction is ">" (more than).

Relay 1906
This relay turns ON if the result of the compare operation is "=" (equal to). It may also turn ON if the result of an arithmetic operation is 0.

Relay 1907
This relay turns ON if the result of the compare operation is "<" (less than).

Relays TR0 to 7
These are temporary memory relays and may not necessarily be assigned in sequence. The same coil number of these relays must not be used in duplicate within the same block of a program. However, the same coil number can be used in a different block. When using a temporary memory relay, the letters "TR" must be prefixed to the relay number (e.g., TR0).

Relation between special auxiliary relays and instructions

Relay No.		Special auxiliary relays				
		1907 (<)	1906 (=)	1905 (>)	1904 (CY)	1903 (ER)
Executed instruction	Operation of relay FUN No.	Turns ON if the result when Compare instruction is executed is less than.	Turns ON if the result when Compare instruction is executed is equal or 0000.	Turns ON if the result when Compare instruction is executed is more than.	Turns ON if a carry is generated as a result of an arithmetic operation	Turns ON if data subject to BCD operation are not in BCD.
TIM						
TIMH	15					
CNT						
CMP	20	↕	↕	↕		
MOV	21		↕			
MVN	22		↕			
ADD	30		↕		↕	↕
SUB	31		↕		↕	↕
STC	40				"1"	
CLC	41				"0"	
END	01	"0"	"0"	"0"	"0"	"0"

Legend: ↕ – Change
Vacant – No change

Note:
1. When special auxiliary relay 1903 is turned ON, the ADD and SUB instructions are treated as NOP and the statuses of the other relays remain unchanged.
2. The statuses of these special auxiliary relays are not changed by executing instructions other than those listed in the above table.

Appendix E Maintenance and troubleshooting

Overview

Regular inspections and appropriate maintenance of the PC are essential to ensure the full life of your PC and trouble-free operation of your controlled system.

Safety measures to protect the system and minimize system downtime in the event of PC failure must also be taken. This chapter covers inspection and troubleshooting procedures for the C20.

Inspection items are presented in table form in the first part of this chapter. This is followed by a brief review of standard maintenance procedures. The troubleshooting guide presents the actions that should be taken in the case of each type of failure in easy-to-follow flowchart form. Finally, a list of the error messages displayed by the programming console is given, along with the corrective action.

Inspection

Inspection items

Semiconductor elements are employed as main components of the C20. Semiconductors are, however, subject to deterioration under severe environmental conditions and should be inspected periodically. The standard inspection cycle is 6 months to 1 year. The frequency of inspection should be increased if required.

If the C20 is found to be outside the criteria shown in the following table, corrections should be made so that the criteria are met.

Power supply	
CPU supply voltage fluctuation (measured at power terminal block using voltmeter)	C20 rated at AC 100 to 120V: AC 85 to 132V C20 rated at AC 200 to 240V: AC 170 to 264V C20 rated at DC 24V: DC 20.4 to 26.4V
Expansion I/O unit supply voltage fluctuation (measured at power terminal block using voltmeter)	Must conform to specifications of each expansion I/O unit
I/O link unit	AC 85 to 264V
Link adapter	Link adapter rated at AC 100 to 120V: AC 85 to 132V Link adapter rated at AC 200 to 240V: AC 170 to 264V Link adapter rated at AC/DC 12 to 24V: AC/DC 10.2 to 26.4V
Battery	DC 10.2 to 13.2V
Environmental conditions	
Ambient temperature in control panel	0 to +55°C
Humidity	35 to 85% RH without condensation
Dust	Must be relatively dust-free

Mounting

CPU and expansion I/O unit firmly secured?	Mounting screws must not be loose
Expansion I/O unit cable securely inserted in connector?	The cable must not be loose
Screws for external wiring firmly secured?	Screws must not be loose
Any break in external wiring cables	Must be free from visible abnormality
Battery service life	5 years

Optical-fiber cable

Bending radius of the optical-fiber cable	15mm min.

Caution: Be sure that the power is off before replacing the expansion I/O or other unit.

Remember

If a defective unit is discovered and replaced, confirm the operation of the replacement unit.

When returning a defective unit to OMRON, enclose a written description of the problem.

If the problem is a poor cable contact, wipe the connector pins with a clean all-cotton cloth moistened with industrial alcohol. Make certain that there is no cloth debris remaining on the cable before plugging it in.

Tools and testing devices for maintenance

Screwdrivers (phillips and flat-bladed)

VOM or digital voltmeter

Industrial alcohol and all-cotton cloth

Tools and testing devices for troubleshooting

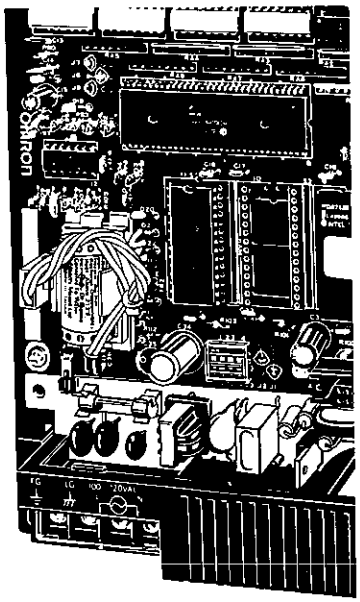
Oscilloscope

Pen-recording oscilloscope

Thermometer, hygrometer

Maintenance parts

To ensure continuous operation in the event of failure, it is recommended to keep at least one spare expansion I/O unit on hand.



Consummables

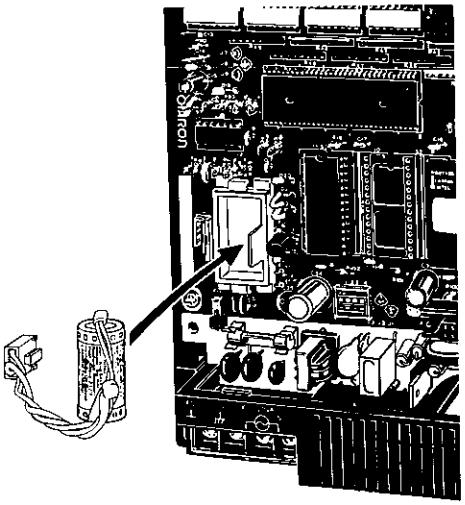
Fuse

For C20 and I/O link unit: 250V 1A (at supply voltage of AC 100 to 120/200 to 240V)/125V 3A (at supply voltage of DC 24V), 5.2 dia.×20

For link adapter: 250V 0.1A, 6.35dia.×31.8

Replacing the fuse

1) Turn off the power
2) Remove the C20's front cover
3) Replace the old fuse using a flat-bladed screwdriver
4) Reattach front cover



Battery

Type 3G2A9-BAT08

The service life of the battery is 5 years. When the battery has been fully discharged, the ERROR (ALARM) indicator blinks. If this happens, replace the battery with a new one within a week. The date when the battery must first be replaced is written on the side panel of the C20. For example, if it says:

FIRST REPLACEMENT 89/12

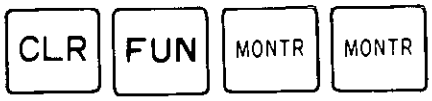
This means you should replace the battery by Dec. 1989

Replacing the battery

1) Turn off the power
   (If the power is not on, turn it on once and then off again after waiting 10 seconds)
2) Remove the C20's front cover
3) Remove the old battery together with the connector to replace it.

Caution: This procedure must be accomplished within 5 minutes of turning the power off.

4) Reattach the front cover
5) Attach the programming console; clear the "BATT LOW" message, observing the following procedure.

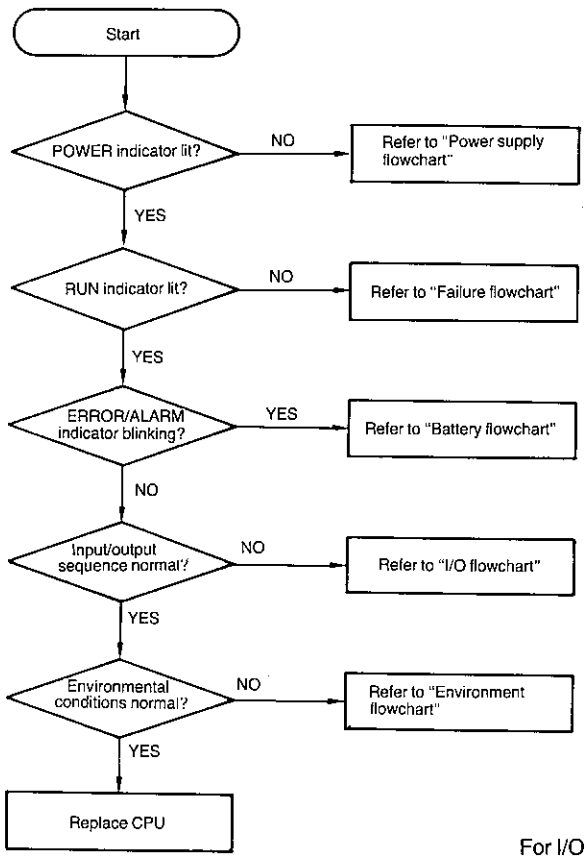


Troubleshooting

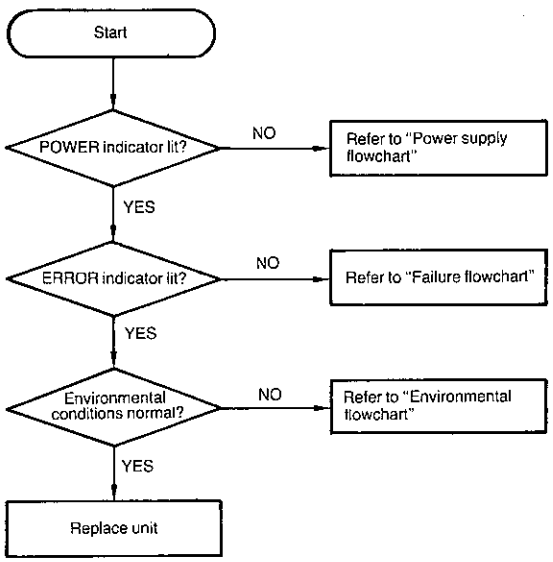
The following flowcharts will enable you to determine the source of any abnormality occurring in the PC or I/O link unit.

General flowcharts

For PC

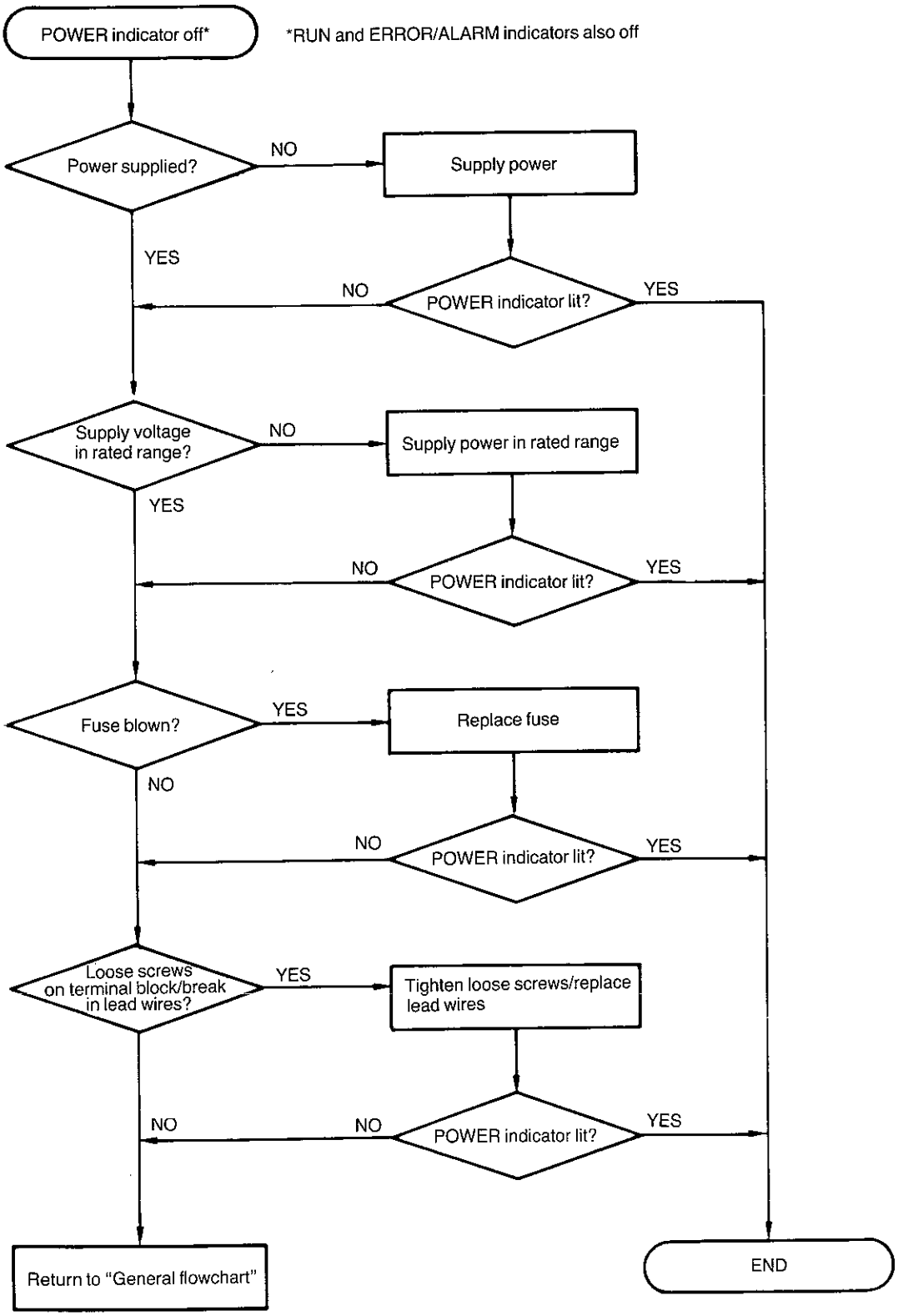


For I/O link unit



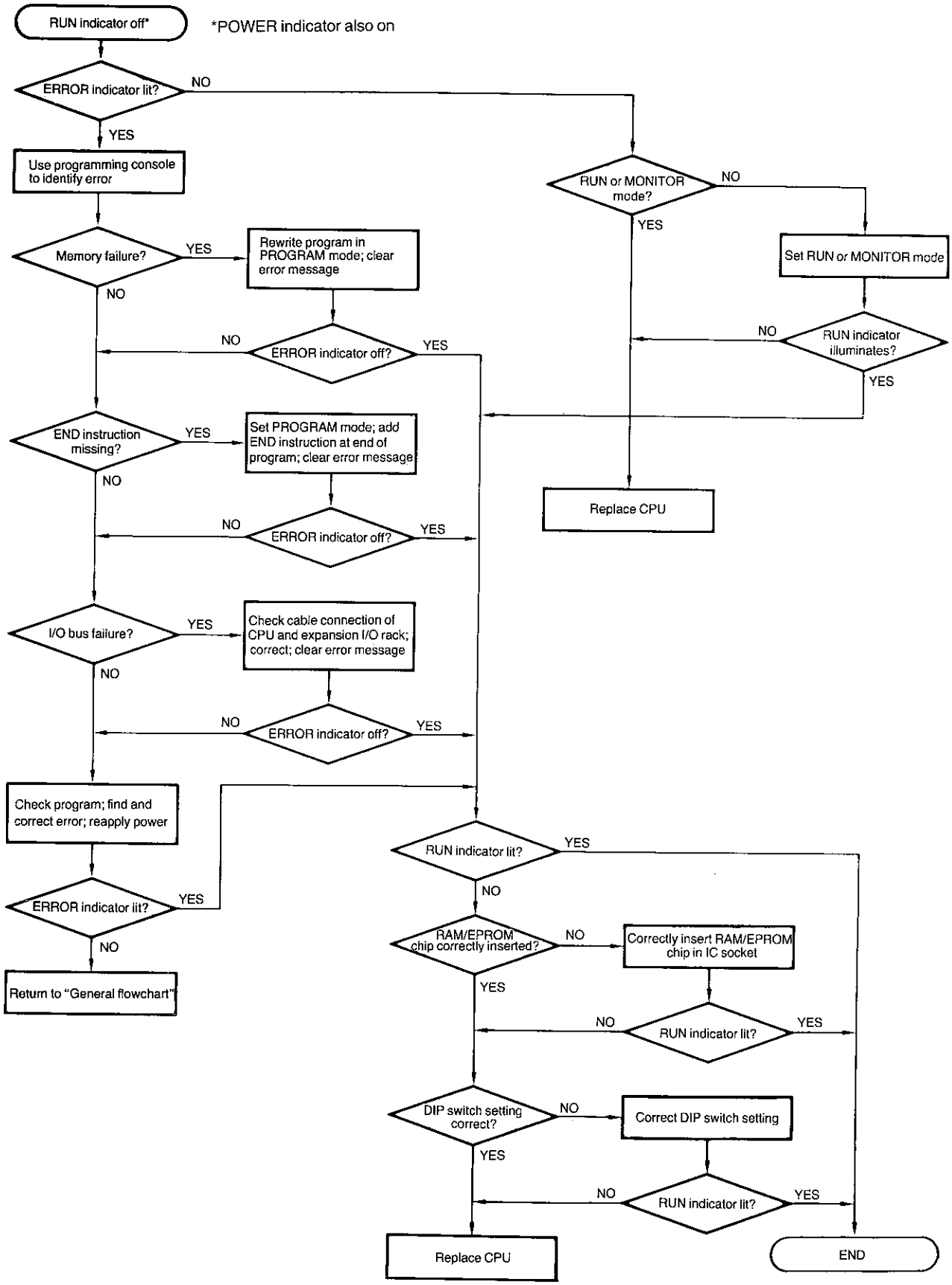
Power supply flowchart

For PC and I/O link unit



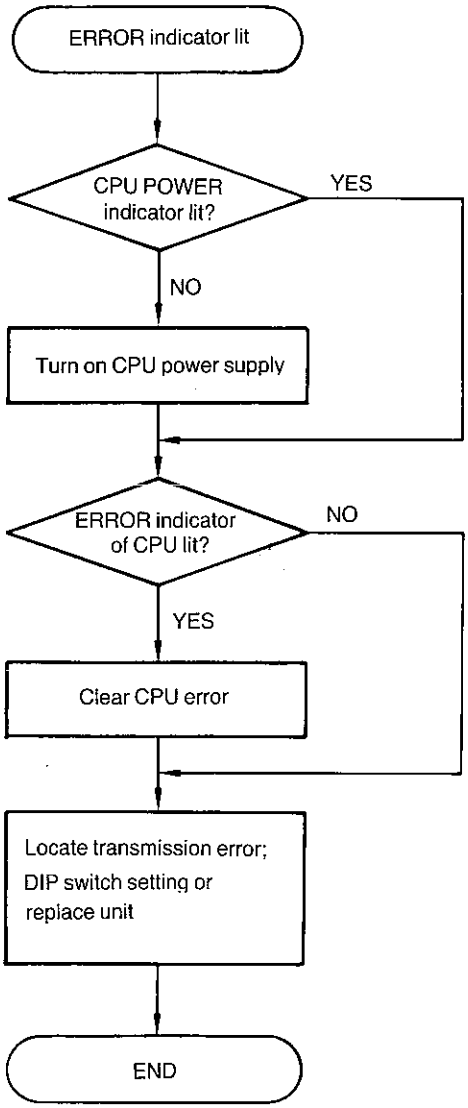
Failure flowchart

For PC



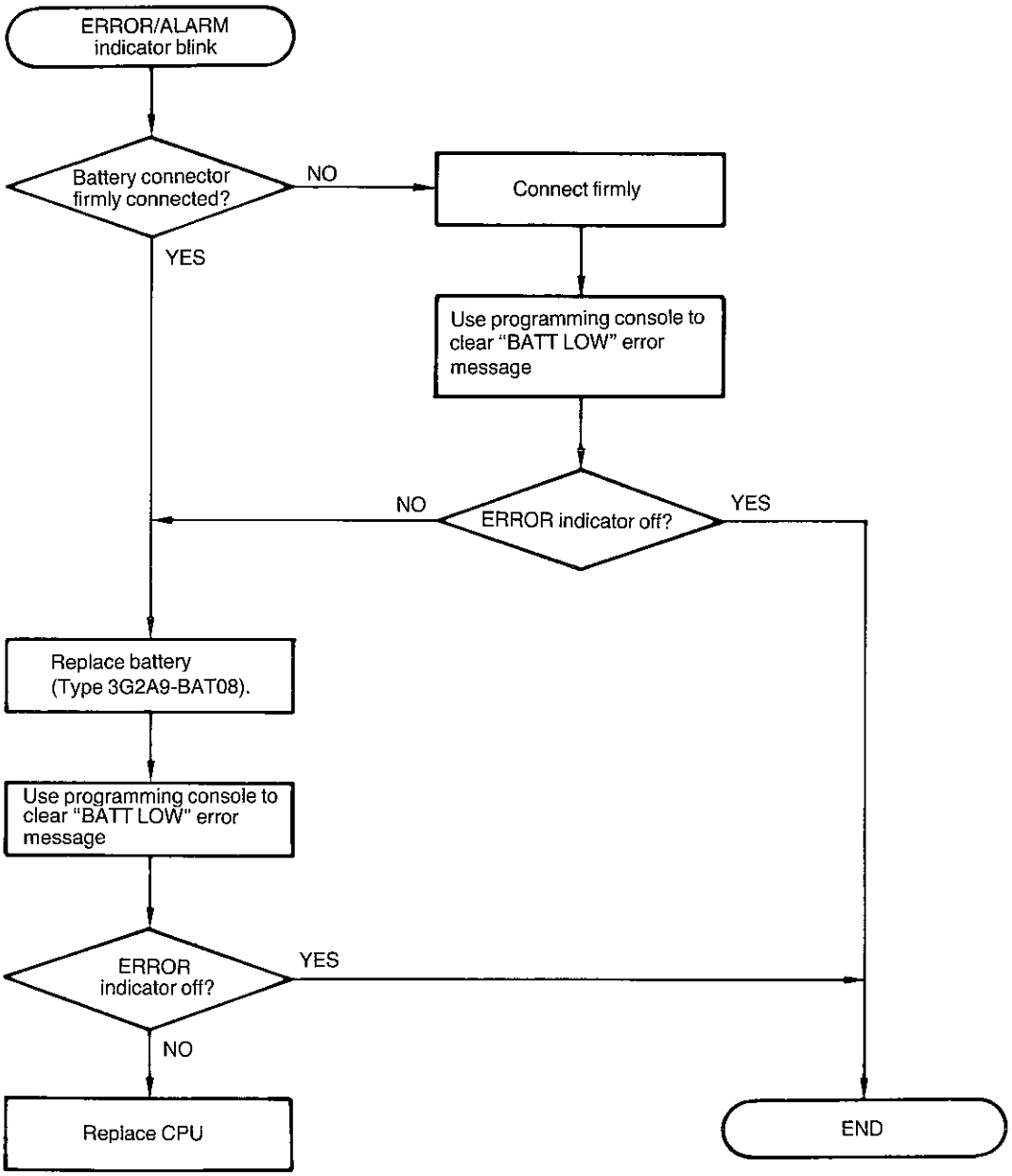
Failure flowchart

For I/O link unit



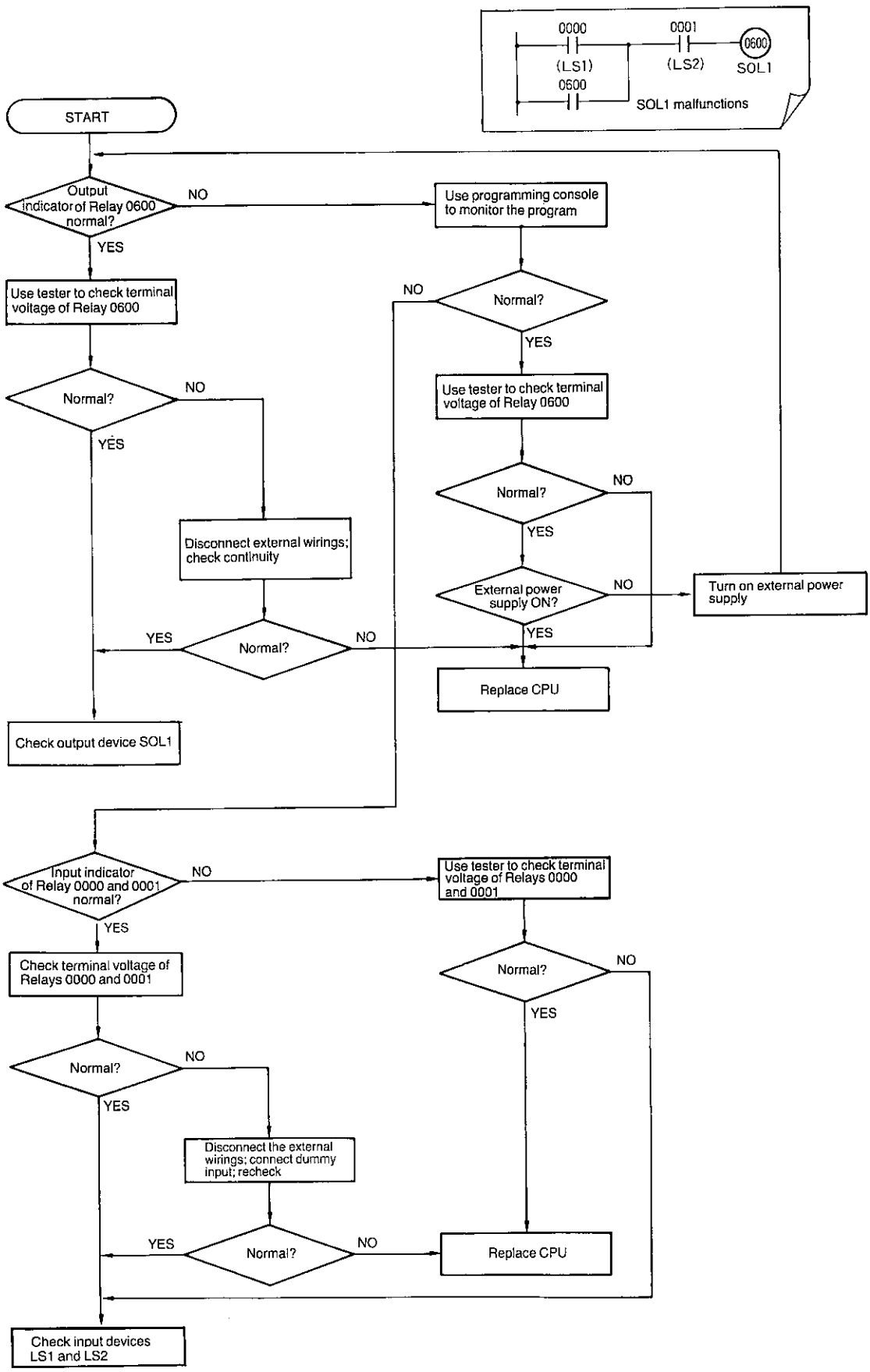
Battery flowchart

For PC



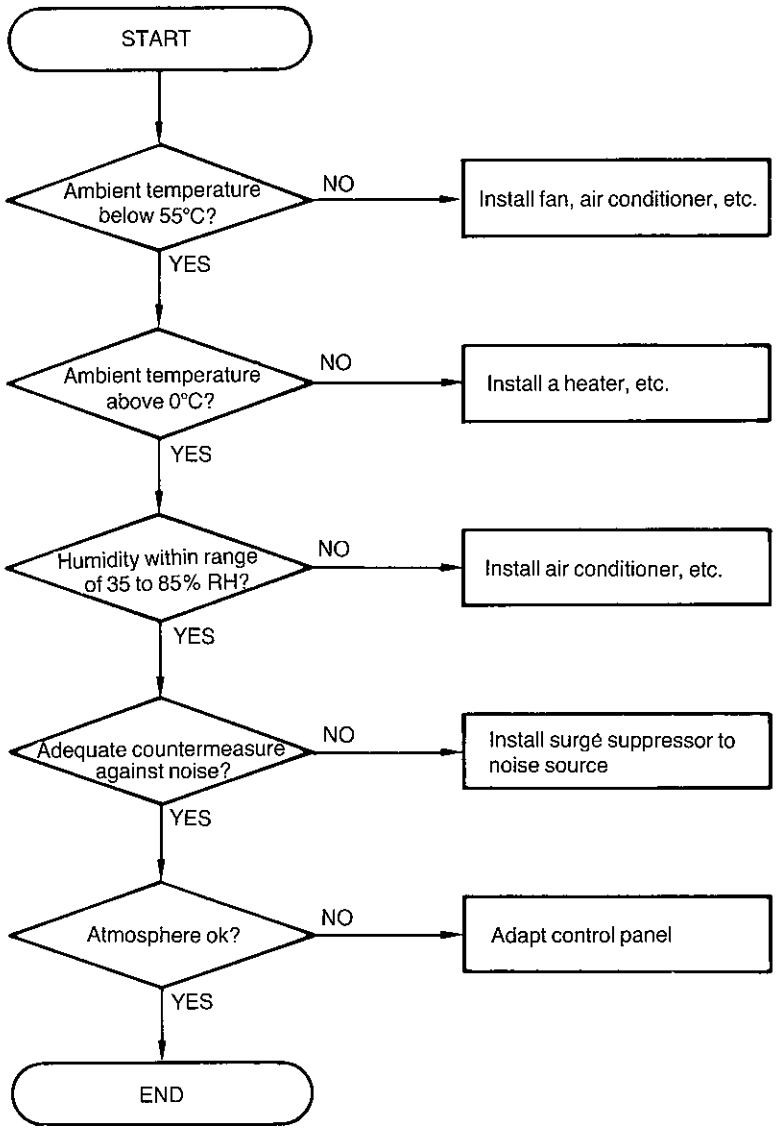
I/O flowchart

For PC



Environment flowchart

For PC and I/O link unit



List of self-diagnostic functions

	LED on CPU rack			Message on programming console display	Item	Remedy
	POWER	RUN	ALARM ERROR			
a.	•	•	•	—	Power failure	Turn on power.
	☼	•	☼	—	CPU failure	Set PROGRAM mode and perform power reset. Program may require correction.
				MEMORY ERR	Memory failure	Check RAM/ROM for correct mounting; debug program. After corrective action, perform failure reset operation.
				NO END INSTR	END instruction missing error	No END instruction at end of program. Write in END instruction and then perform failure reset operation.
				I/O BUS ERR	I/O bus failure	Check bus line connecting the C20 to I/O units. Check cable for disconnection or breakage. When I/O link unit is used, check channel setting and optical-fiber cable for breakage. After corrective action, perform failure reset operation.
b.	☼	☼	☼	BATT LOW	Battery failure	Check battery for proper insertion into the socket. Replace if fully discharged. After corrective action, perform failure reset operation.

☼: denotes that the indicator illuminates.
•: denotes that the indicator goes off.
☼: denotes that the indicator blinks.

a. "Fatal error" i.e., one that causes the CPU to stop operating
b. "Nonfatal error" one that does not cause the CPU to stop operating

CPU failure

The "watchdog timer" is a timer that is reset each time the CPU executes the END instruction. This timer is used to check the execution time of the CPU when it operates in the RUN or MONITOR mode. When the set time of the watchdog timer is up, this indicates a failure.

Memory failure

Sum check of the memory is performed at fixed intervals to check the memory contents and existence of any abnormal instruction.

END instruction missing error

Checks whether the END instruction is written at the end of the program.

I/O bus failure

The I/O cable is checked for breakage. The hardware of the I/O control section of the PC is also checked.

Battery failure

Checks the supply voltage of the memory backup battery. By forming a program as shown below using special auxiliary relay 1808, an alarm signal can be output to an external device when there is a drop in the battery supply voltage.

List of error messages

In addition to the messages described on the previous page, the following error messages may also be displayed on the programming console by keying in

Error message	Description
****ADR OVER	An address exceeding the last address of the user memory has been set. Set a different address.
****REPL ROM	An EPROM chip is mounted as the user program. Replace the EPROM with a RAM chip and then perform the intended write operation.
****SETDATA ERR	Hexadecimal data has been input as a constant where a decimal number is required. Input the data as a decimal number.
****I/O NO ERR	Data exceeding the input range is input. Check the limit of data each instruction can use and repeat data input.
MEMORY ERR	There is an abnormality in the user program memory. Check whether a RAM or EPROM chip is mounted in the CPU as the program memory. A wrong instruction exists in the user program. Check and correct the program. After the corrective action, clear the error message (refer to step 7, Testing for errors in Chapter 5).
I/O BUS ERR *	A failure has occurred in the bus connecting the CPU and expansion I/O unit. Check the bus. Also check the expansion I/O unit for disconnection before power application.
BATT LOW	Check whether the battery is correctly installed in the battery socket. The service life of the battery has expired. Replace the battery.
****NO END INSTR	The END instruction is missing. Write END instruction at end of program. Clear the error message (refer to step 7, Testing for errors in Chapter 5).
****PROG OVER	Program exceeds the memory capacity. Check the program.
☆☆☆☆VER ERR	The contents of the cassette tape and user program do not agree. Check the contents of the cassette tape and user program.
☆☆☆☆MT ERR	An error exists in the cassette tape. Replace the tape with a new one.

Appendix F Specifications

Overview

This appendix describes the specifications for the CPU, the I/O link unit, and the link adapter. Specifications for other peripheral devices are provided in the respective user's manuals.

Available types

Classification	Specifications							Weight	Type name
	Power supply	Input		No. of inputs	Output	No. of outputs	Expansion function		
CPU	DC 24V	DC 24V	NPN	16	Contact	12	NO	2kg max.	3G2C7-CPU73-E
							YES		3G2C7-CPU74-E
			PNP				NO		3G2C7-CPU83-E
							YES		3G2C7-CPU84-E
			NPN		Transistor		YES		3G2C7-CPU76-E
	AC 100 to 120V		NPN		Contact		NO		3G2C7-CPU13-E
							YES		3G2C7-CPU14-E
			PNP				NO		3G2C7-CPU23-E
							YES		3G2C7-CPU24-E
		AC 100 to 120V		14	Contact		YES		3G2C7-CPU34-E
					Triac		YES		3G2C7-CPU38-E
	AC 200 to 240V	DC 24V	NPN	16	Contact		NO		3G2C7-CPU43-E
							YES		3G2C7-CPU44-E
			PNP				NO		3G2C7-CPU53-E
							YES		3G2C7-CPU54-E
Expansion I/O unit	DC 24V for output driving	DC 24V	NPN	16	Contact	12			3G2C7-MC223
				32		24			3G2C7-MC224
			PNP	16		12			3G2C7-MC227
				32		24			3G2C7-MC228
			NPN	16	Transistor	12			3G2C7-MD211
				32		24			3G2C7-MD212
		AC 100 to 120V		14	Contact	12			3G2C7-MC22B
				28		24			3G2C7-MC22C
				14	Triac	12			3G2C7-MA221
				28		24			3G2C7-MA222
I/O link unit	AC 100 to 240V	No. of inputs: 16			No. of outputs: 16			1kg max.	3G2C7-LK011
I/O connecting cable	Cable length: 5cm							200g max.	3G2C7-CN501
	Cable length: 32cm								3G2C7-CN311
Mounting kit	For sort of piggyback arrangement of CPU and expansion I/O unit with 3G2C7-CN311								3G2C7-PAT02
Optional products RAM*	For nonexpandable CPU (3G2C7-CPU13-E/-CPU43-E/-CPU23-E/-CPU53-E/-CPU83-E)							50g max.	RAM-G
Optional products EPROM	For CPU							50gmax.	ROM-H

* The RAM chip is mounted in the expandable type CPU (i.e., Type 3G2C7-CPU14-E/-CPU44-E/-CPU74-E/-CPU24-E/-CPU54-E/-CPU84-E) as the factory-set condition for shipment.

Programming console (horizontal type)	For SYSMAC-C series programmable controllers	3G2A6-PR015-E
Programming console (vertical type)	For SYSMAC-C series programmable controllers	3G2A5-PR013-E
Programming console connecting cable	50cm, for C20 only	3G2C7-CN512
	1m, for C20 only	3G2C7-CN122
Cassette tape recorder connecting cable	1.5m	SCY-PLG01
PROM writer	For SYSMAC-C series programmable controllers	3G2A5-PRW04-E
Printer interface unit	For SYSMAC-C series programmable controllers (Printer/X-Y plotter connecting interface and memory cassette are separately available.)	3G2A5-PRT01-E
Memory cassette for printer interface unit	Exclusive for SYSMAC-C20	3G2A5-MP009-E
Printer connecting cable	2m (can be also used for X-Y plotter connection)	SCY-CN201
Peripheral interface unit	For connecting GPC (CRT type/LCD type) and MSB, exclusively for SYSMAC-C20	3G2C7-IP002
Peripheral interface unit connecting cable	For connecting GPC (CRT type/LCD type) and MSB	3G2A2-CN221
Graphic programming console (LCD type)	AC 110/120V (Memory pack is separately available.)	3G2C5-GPC01-E
Graphic programming console (LCD type)	AC 220/240V (Memory pack is separately available.)	3G2C5-GPC02-E
Memory pack for GPC (LCD type)	For SYSMAC-C series programmable controllers (C20, C120, C250, C500)	3G2C5-MP301-E
Graphic programming console (CRT type)	AC 110/120V	3G2A5-CRT19
Graphic programming console (CRT type)	AC 220/240V	3G2A5-CRT20
Multisupport base	AC 110V. Two devices can be connected. Memory cassette is separately available.	3G2A5-MSB01-E
	AC 220V. Two devices can be connected. Memory cassette is separately available.	3G2A5-MSB02-E
Memory cassette for multisupport base	For SYSMAC-C series programmable controllers	3G2A5-MP001-E
Battery set	For SYSMAC-C series programmable controllers	3G2A9-BAT08
RAM	For nonexpandable type CPU	RAM-G (See NOTE 2.)
	For expandable type CPU	(See NOTE 3.)
EPROM	For nonexpandable type CPU	ROM-H
	For expandable type CPU	ROM-H
Link adapter	Repeater for fiber optics (AC 110/220V, AC/DC 12 to 24V)	3G2A9-AL002-E
Fiber optics cable (for indoor use)	Cable length: 0.1m, with connector	3G5A2-OF011
	Cable length: 1m, with connector	3G5A2-OF101
	Cable length: 2m, with connector	3G5A2-OP201
	Cable length: 3m, with connector	3G5A2-OF301
	Cable length: 5m, with connector	3G5A2-OF501
	Cable length: 10m, with connector	3G5A2-OF111
	Cable length: 20m, with connector	3G5A2-OF211
	Cable length: 30m, with connector	3G5A2-OF311
	Cable length: 40m, with connector	3G5A2-OF411
	Cable length: 50m, with connector	3G5A2-OF511

NOTE:
1. The RAM and ROM for the CPU are separately available. The battery set is provided to the CPU.
2. The RAM-G and RAM-F are not compatible with each other.
   Programming of up to 170 addresses can be performed without the RAM-G.
3. The RAM chip is mounted in the expandable type CPU as the factory-set condition for shipment.
4. Although both the vertical type and horizontal type programming consoles can be used with the SYSMAC-C20, the use of the horizontal type is recommended in consideration of operability.

Ratings

Supply voltage	AC 100 to 120V AC 200 to 240V DC 24V
Operating voltage range	AC 85 to 132V AC 170 to 264V DC 20.4 to 26.4V
Power consumption	25 VA max. 20 W max.
Insulation resistance	5MΩ min. at DC 500V (between external terminal and outer casing)
Dieletric strength	AC 1,500V 50/60Hz for 1 minute (between external terminal and outer casing)
Noise immunity	1,000Vp-p Rise time: 1ns; Pulse width: 100ns to 1μs
Vibration	16.7Hz, 3mm double amplitude (in X, Y, and Z directions, respectively for 30 minutes)
Shock	10G (in X, Y, and Z directions, respectively 3 times)
Ambient temperature	Operating: 0° to +55°C Storage: −20° to +65°C
Humidity	35 to 85% RH (without condensation)
Atmosphere	Must be free from corrosive gases
Degree of protection	IP30

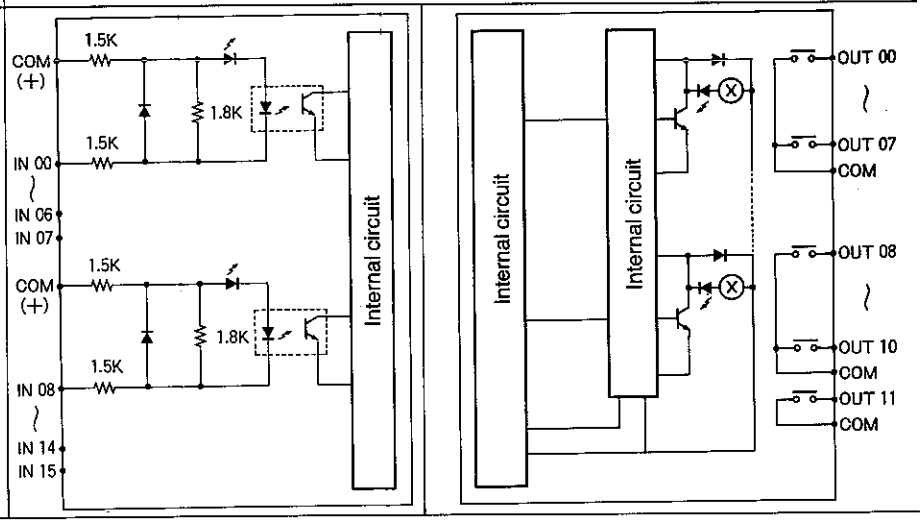
Characteristics

Control system	Stored program system
Main control element	MPU, CMOS, LS-TTL
Programming system	Ladder diagram
Instruction word length	6 bytes/address
Number of instructions	27
Execution time	Average: 10μs/address
Programming capacity	Nonexpandable CPU RAM 512 addresses EPROM 1194 addresses Expandable CPU RAM 1194 addresses EPROM 1194 addresses
Number of input/output relays	Nonexpandable CPU Input: 16 points for DC input type/14 points for AC input type, output: 12 points Expandable CPU Input: 80 points max. for DC input type/70 points max. for AC input type, output: 60 points max. (with two 56-point expansion I/O unit)

Number of internal auxiliary relays	136 points (Relays 1000 to 1807)
Number of holding relays	160 points (Relays HR000 to 915)
Number of timers/counters	48 points (Timers/Counters TIM(H)/CNT00 to 47) Timer: 0.1 to 999.9s High-speed timer:0.01 to 99.99s Counter: 1 to 9999 counts
Number of temporary memory relays	8 points (Relays TR0 to TR7)
Number of auxiliary relays	16 points (Relays 1808 to 1907) including: Battery failure flag (1808) Normally OFF relays (1809, 1810, 1811, 1812, and 1814) Normally ON relay (1813) Initial one scan time ON relay (1815) For generation of 0.1s clock (1900) For generation of 0.2s clock (1901) For generation of 1s clock (1902)
Memory protection against power failure	Status of holding relays and present value of counters before power failure retained in memory
Battery	Service life of built-in battery approximately 5 years at ambient temperature of 25°C. Battery life shortened if ambient temperature exceeds 25°C. Replace battery within one week after ALARM/ERROR indicator illuminates. Battery must be replaced within approximately 5 minutes after the power is turned off.
Diagnostic functions	CPU failure (watchdog timer) Battery failure Memory error I/O bus failure PROGRAM check (program execution time only) END instruction missing Instruction error

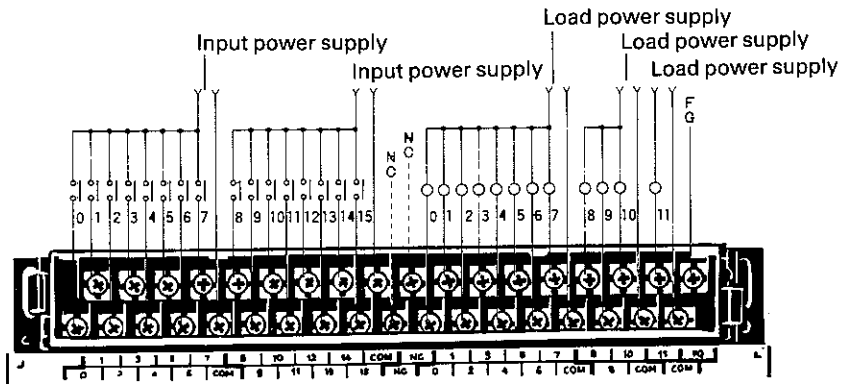
Input/Output specifications

CPU

Type	3G2C7-CPU13-E/-CPU14-E/-CPU43-E/-CPU44-E/-CPU73-E/-CPU74-E	
	DC input (NPN)	Contact output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	15ms max.
OFF delay time	20ms max.	15ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1) 8/4/2A/ common 14A/unit
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	—
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24V, with socket)
Power supply	DC 24V (CPU73-E/CPU74-E), AC 100 to 120V (CPU13-E/CPU14-E), AC 200 to 240V (CPU43-E/CPU44-E)	
Circuit configuration		

Terminal Signal name (I/O No.) / Terminal Signal name (I/O No.)

	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
	IN1 (0001)	IN3 (0003)	IN5 (0005)	IN7 (0007)	IN8 (0008)	IN10 (0010)	IN12 (0012)	IN14 (0014)	COM	NC	OUT1 (0501)	OUT3 (0503)	OUT5 (0505)	OUT7 (0507)	OUT8 (0508)	OUT10 (0510)	OUT11 (0511)	FG
	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
	IN0 (0000)	IN2 (0002)	IN4 (0004)	IN6 (0006)	COM	IN9 (0009)	IN11 (0011)	IN13 (0013)	IN15 (0015)	NC	OUT0 (0500)	OUT2 (0502)	OUT4 (0504)	OUT6 (0506)	COM	OUT9 (0509)	COM	COM

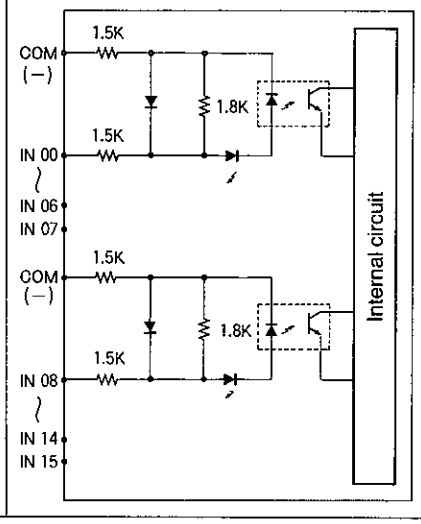


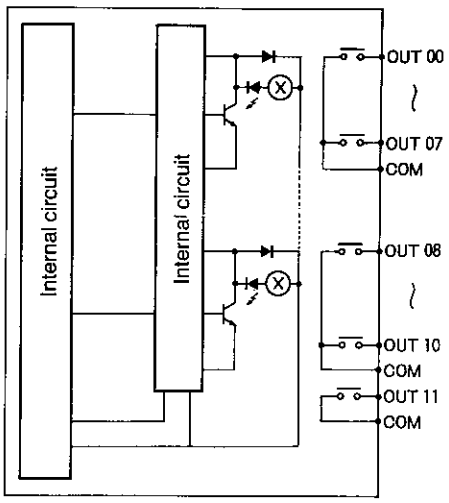
Note:
The input channel is fixed on channel 0 and the output channel on channel 5. The terminal block can be removed.

CPU

Type	3G2C7-CPU23-E/-CPU24-E/-CPU53-E/-CPU54-E/-CPU83-E/-CPU84-E	
	DC input (PNP)	Contact output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	15ms max.
OFF delay time	20ms max.	15ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1) 8/4/2A/ common 14A/unit
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	—
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24V, with socket)
Power supply	DC 24V (CPU83-E/CPU84-E), AC 100 to 120V (CPU23-E/CPU24-E), AC 200 to 240V (CPU53-E/CPU54-E)	

Circuit configuration



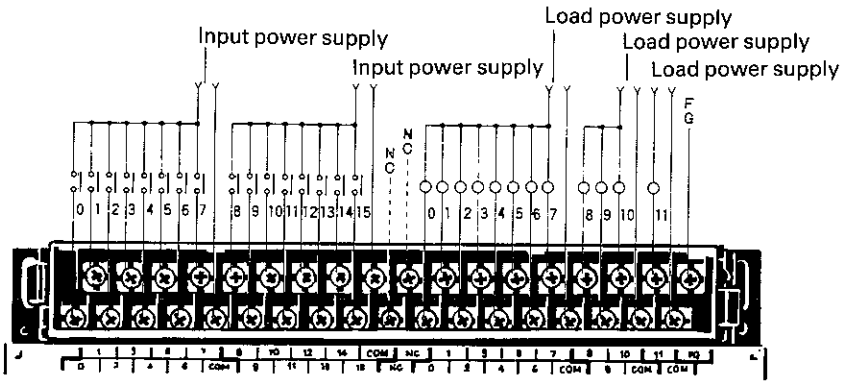


Terminal Signal name (I/O No.)

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
IN1 (0001)	IN3 (0003)	IN5 (0005)	IN7 (0007)	IN8 (0008)	IN10 (0010)	IN12 (0012)	IN14 (0014)	COM	NC	OUT1 (0501)	OUT3 (0503)	OUT5 (0505)	OUT7 (0507)	OUT8 (0508)	OUT10 (0510)	OUT11 (0511)	FG

Terminal Signal name (I/O No.)

A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
IN0 (0000)	IN2 (0002)	IN4 (0004)	IN6 (0006)	COM	IN9 (0009)	IN11 (0011)	IN13 (0013)	IN15 (0015)	NC	OUT0 (0500)	OUT2 (0502)	OUT4 (0504)	OUT6 (0506)	COM	OUT9 (0509)	COM	COM

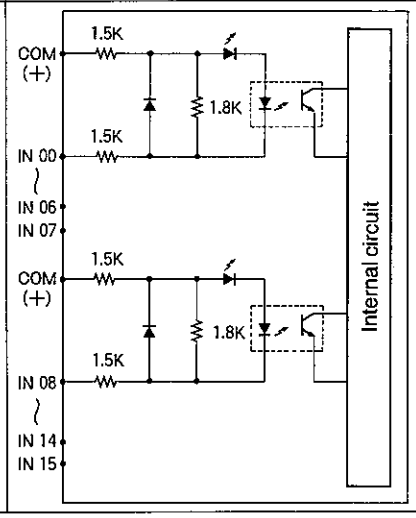


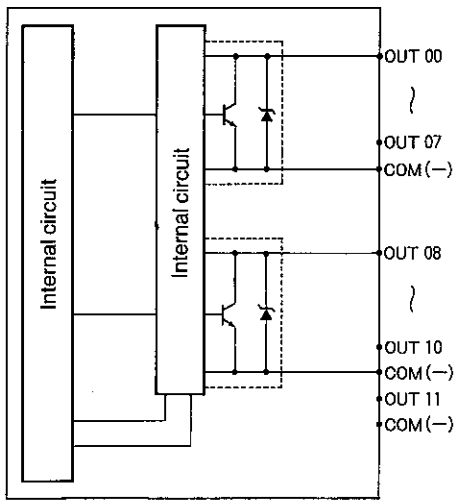
Note:
The input channel is fixed on channel 0 and the output channel on channel 5. The terminal block can be removed.

CPU

Type	3G2C7-CPU76-E	
	DC input (NPN)	Transistor output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	1.5ms max.
OFF delay time	20ms max.	1.5ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	DC 5 to 24V 0.5A max.
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	100μA max.
Saturation voltage	—	1.5V max.
External power supply	—	—
Service life (of relay)	—	(G3SD-YO1P-PC DC 24V)
Power supply	DC 24V	

Circuit configuration



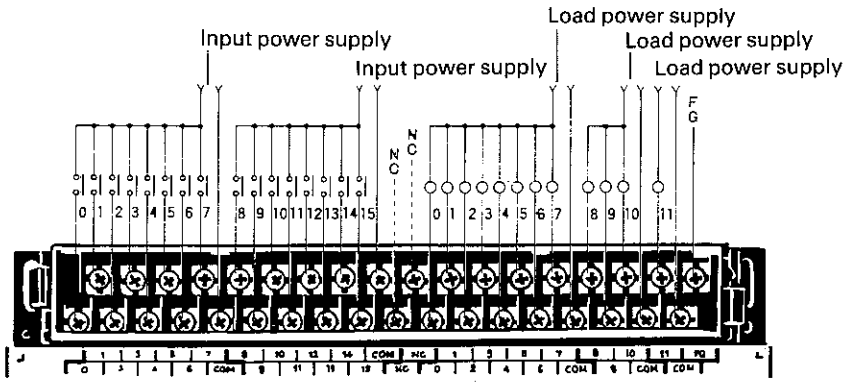


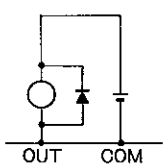
Terminal Signal name (I/O No.)

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
IN1 (0001)	IN3 (0003)	IN5 (0005)	IN7 (0007)	IN8 (0008)	IN10 (0010)	IN12 (0012)	IN14 (0014)	COM	NC	OUT1 (0501)	OUT3 (0503)	OUT5 (0505)	OUT7 (0507)	OUT8 (0508)	OUT10 (0510)	OUT11 (0511)	FG

Terminal Signal name (I/O No.)

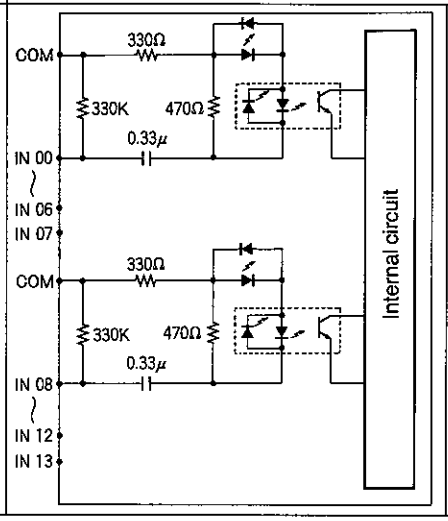
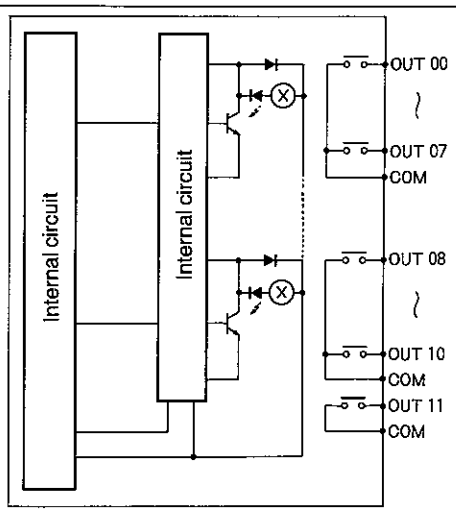
A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
IN0 (0000)	IN2 (0002)	IN4 (0004)	IN6 (0006)	COM	IN9 (0009)	IN11 (0011)	IN13 (0013)	IN15 (0015)	NC	OUT0 (0500)	OUT2 (0502)	OUT4 (0504)	OUT6 (0506)	COM	OUT9 (0509)	COM	COM



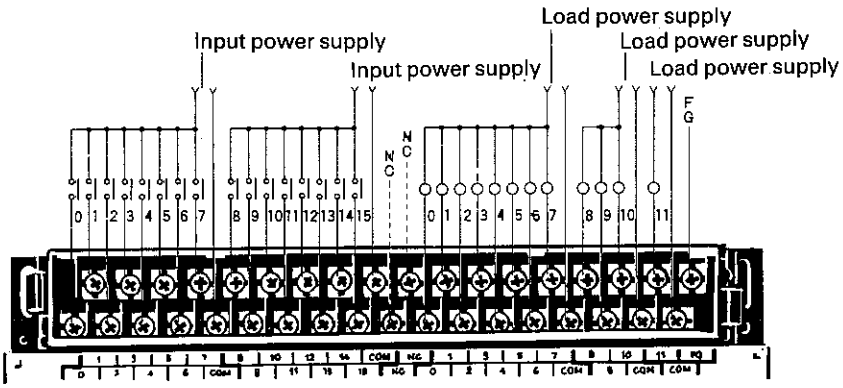


Note:
The input channel is fixed on channel 0 and the output channel on channel 5.
The terminal block can be removed.
When connecting an inductive load, connect a diode (1A, 100V min.) in parallel with the load.

CPU

Type	3G2C7-CPU34-E	
	AC 100 to 120V input	Contact output
Input voltage	AC 85 to 132V	—
Input impedance	9.7kΩ (50Hz)/8kΩ (60Hz)	—
Input current	10mA typ. (AC 100V)	—
ON delay time	35ms max.	15ms max.
OFF delay time	55ms max.	15ms max.
Number of circuits	14 points (8/6 points/common)	12 points (8/3/1 points/common)
ON voltage	AC 60V max.	—
OFF voltage	AC 20V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1) 8/4/2A/ common 14A/unit
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	—
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24V, with socket)
Power supply	AC 100 to 120V	
Circuit configuration		

Terminal	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0001)	IN3 (0003)	IN5 (0005)	IN7 (0007)	IN8 (0008)	IN10 (0010)	IN12 (0012)		COM	NC	OUT1 (0501)	OUT3 (0503)	OUT5 (0505)	OUT7 (0507)	OUT8 (0508)	OUT10 (0510)	OUT11 (0511)	FG
Terminal	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0000)	IN2 (0002)	IN4 (0004)	IN6 (0006)	COM	IN9 (0009)	IN11 (0011)	IN13 (0113)		NC	OUT0 (0500)	OUT2 (0502)	OUT4 (0504)	OUT6 (0506)	COM	OUT9 (0509)	COM	COM

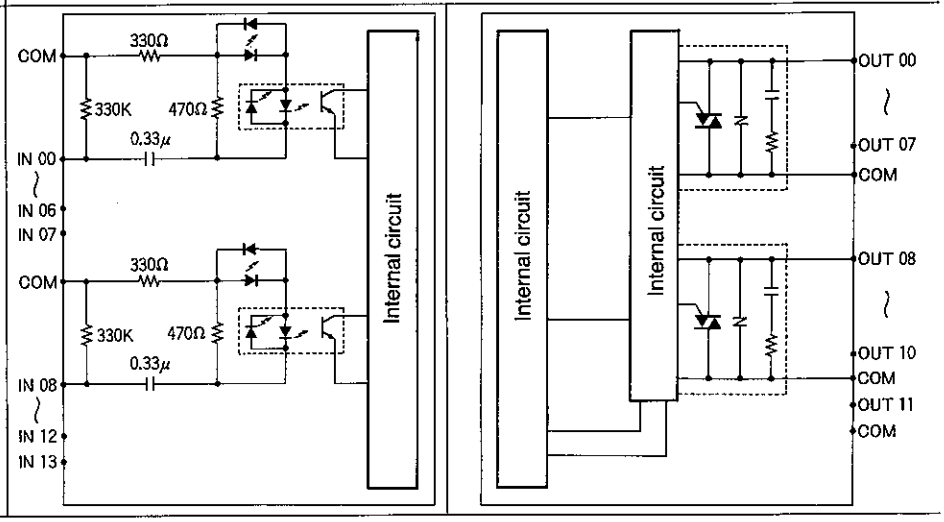


Note:
The input channel is fixed on channel 0 and the output channel on channel 5. The terminal block can be removed.

CPU

Type	3G2C7-CPU38-E	
	AC 100 to 120V input	Triac output
Input voltage	AC 85 to 132V	—
Input impedance	9.7kΩ (50Hz)/8kΩ (60Hz)	—
Input current	10mA typ. (AC 100V)	—
ON delay time	35ms max.	1.5ms max.
OFF delay time	55ms max.	1/2 of load frequency max.
Number of circuits	14 points (8/6 points/common)	12 points (8/3/1 points/common)
ON voltage	AC 60V max.	—
OFF voltage	AC 20V min.	—
Max. switching capacity	—	AC 85 to 250V 0.2A max.
Min. switching capacity	—	10mA (AC 100V)/20mA (AC 200V) min.
Leakage current	—	2mA (AC 100V)/5mA (AC 200V) max.
Saturation voltage	—	1.6V max.
External power supply	—	—
Service life (of relay)	—	(G3S-201PL-PC DC 24V)
Power supply	AC 100 to 120V	

Circuit configuration

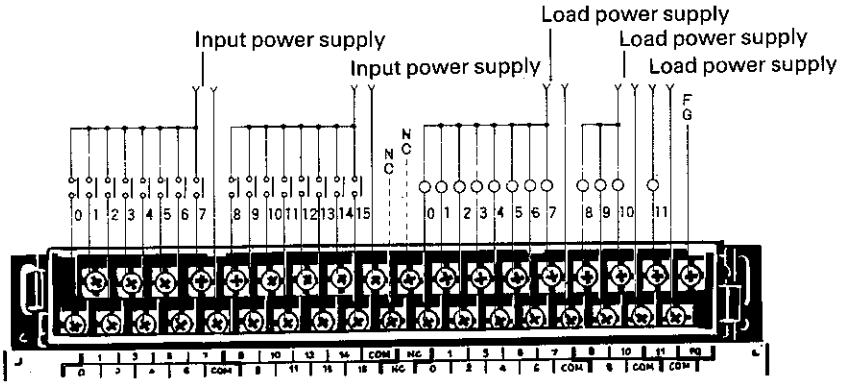


Terminal Signal name (I/O No.)

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
IN1 (0001)	IN3 (0003)	IN5 (0005)	IN7 (0007)	IN8 (0008)	IN10 (0010)	IN12 (0012)		COM	NC	OUT1 (0501)	OUT3 (0503)	OUT5 (0505)	OUT7 (0507)	OUT8 (0508)	OUT10 (0510)	OUT11 (0511)	FG

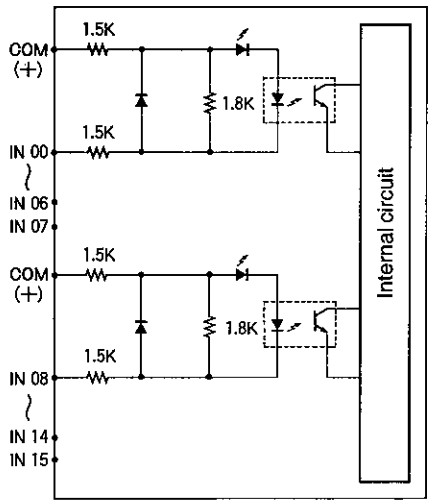
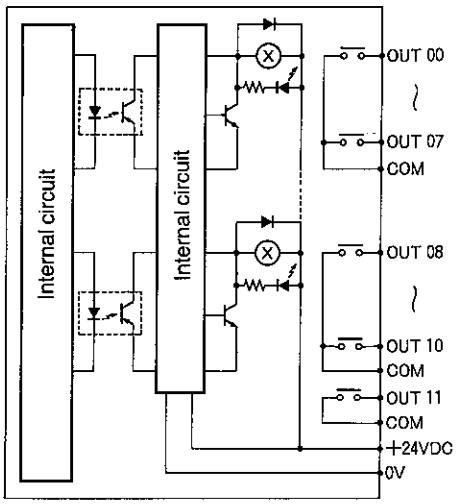
Terminal Signal name (I/O No.)

A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
IN0 (0000)	IN2 (0002)	IN4 (0004)	IN6 (0006)	COM	IN9 (0009)	IN11 (0011)	IN13 (0013)		NC	OUT0 (0500)	OUT2 (0502)	OUT4 (0504)	OUT6 (0506)	COM	OUT9 (0509)	COM	COM

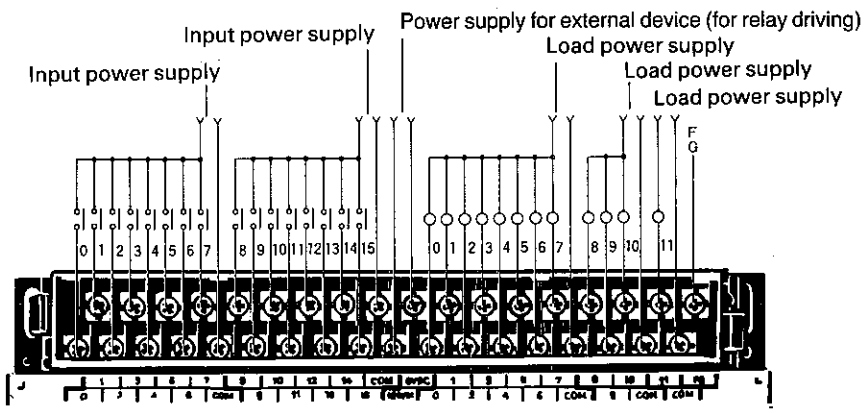


Note:
The input channel is fixed on channel 0 and the output channel on channel 5. The terminal block can be removed.

Expansion I/O unit

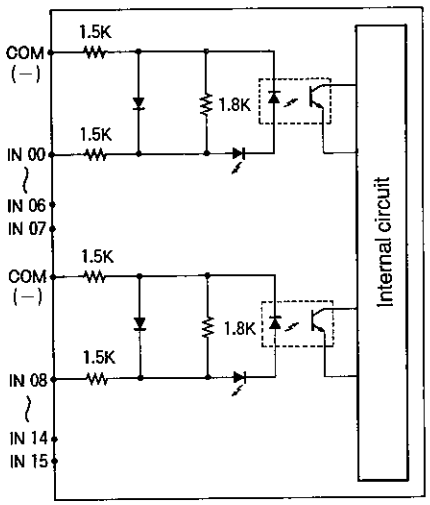
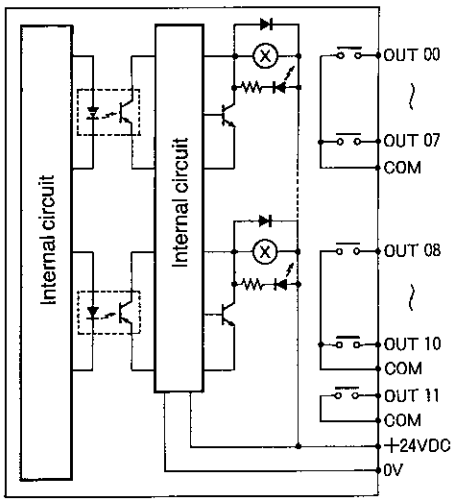
Type	3G2C7-MC223/-MC224	
	DC input (NPN)	Contact output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	15ms max.
OFF delay time	20ms max.	15ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1) 8/4/2A/ common 14A/unit
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	15mA/point, 180mA/unit
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24V, with socket)
Circuit configuration		

Terminal																		
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)	IN14 (0114)	COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
Terminal	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)	IN15 (0115)	+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM

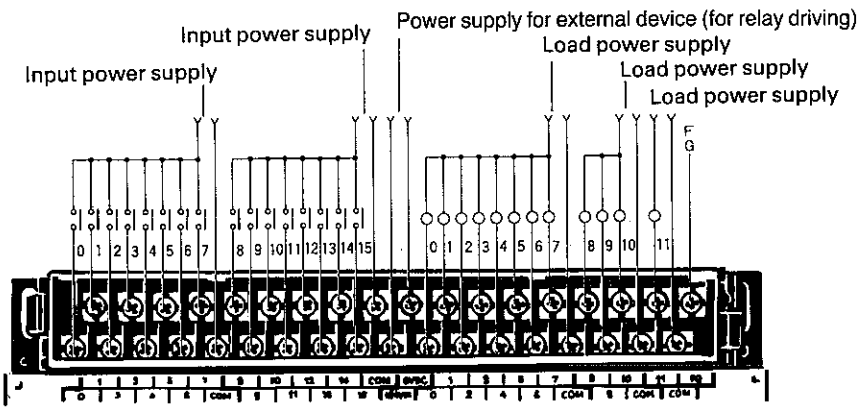


Note:
The input and output channels of MC223 (with 16 input points/12 output points) are fixed on channels 1 and 6. Those of MC224 (with 32 input points/24 output points) are fixed on channels 1 and 2 (input), and 6 and 7 (output).

Expansion I/O unit

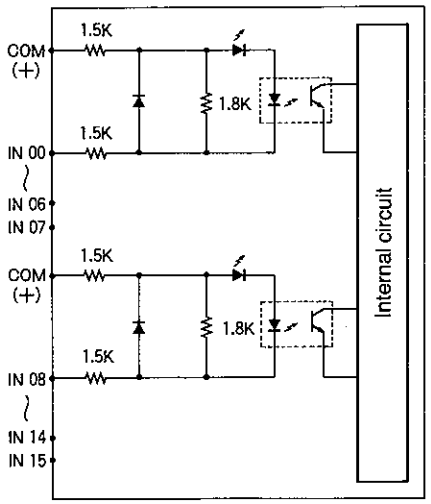
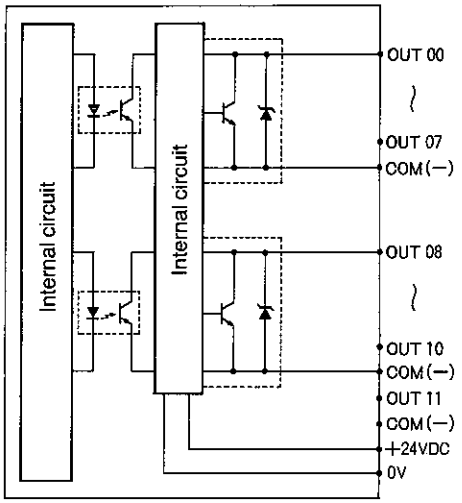
Type	3G2C7-MC227/-MC228	
	DC input (PNP)	Contact output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	15ms max.
OFF delay time	20ms max.	15ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1) 8/4/2A/ common 14A/unit
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	15mA/point, 180mA/unit
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24V, with socket)
Circuit configuration		

Terminal Signal name (I/O No.)	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)	IN14 (0114)	COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
Terminal Signal name (I/O No.)	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)	IN15 (0115)	+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM



Note:
The input and output channels of MC227 (with 16 input points/12 output points) are fixed on channels 1 and 6. Those of MC228 (with 32 input points/24 output points) are fixed on channels 1 and 2 (input), and 6 and 7 (output).

Expansion I/O unit

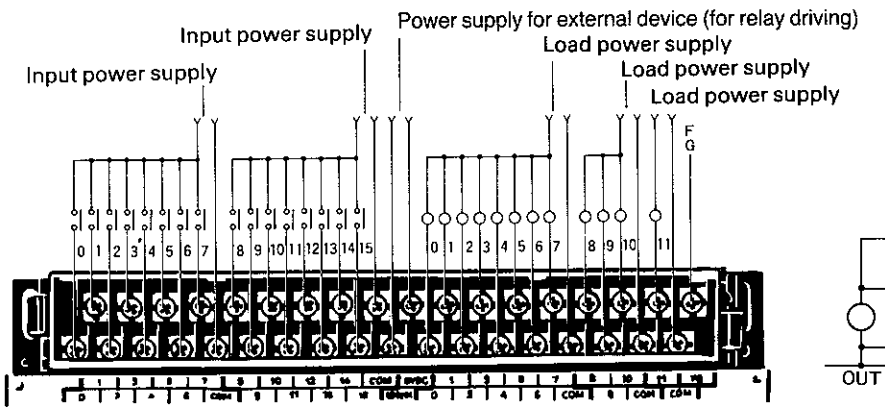
Type	3G2C7-MD211	
	DC input (NPN)	Transistor output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	1.5ms max.
OFF delay time	20ms max.	1.5ms max.
Number of circuits	16 points (8 points/common)	12 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	DC 5 to 24V 0.5A max.
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	100μA max.
Saturation voltage	—	1.5V max.
External power supply	—	15mA/point, 180mA/unit
Service life (of relay)	—	(G3SD-Z01P-PC DC 24V)
Circuit configuration		

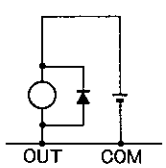
Terminal Signal name (I/O No.)

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)	IN14 (0114)	COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG

Terminal Signal name (I/O No.)

A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)	IN15 (0115)	+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM



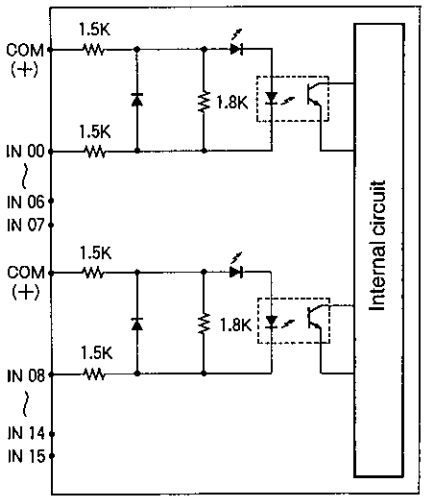
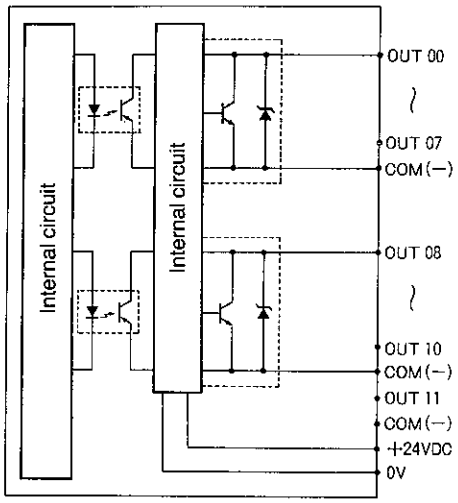
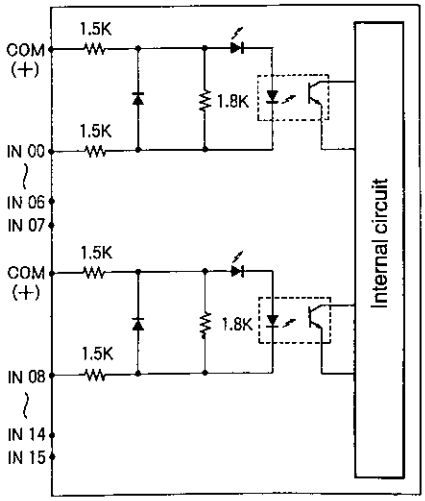
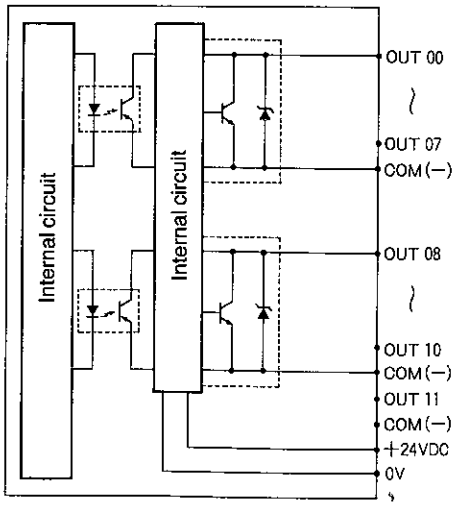


Note:
The input and output channels of MD211 (with 16 input points/12 output points) are fixed on channels 1 and 6.

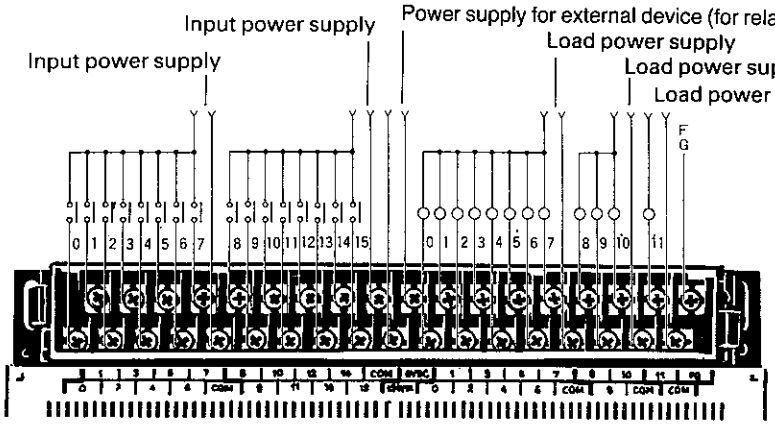
When connecting an inductive load, connect a diode (1A, 100V min.) in parallel with the load.

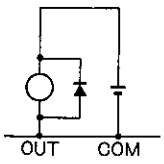
* Detachable terminal block

Expansion I/O unit

Type	3G2C7-MD212	
	DC input (NPN)	Transistor output
Input voltage	DC 24V +10%, −15%	—
Input impedance	3kΩ	—
Input current	7mA typ. (DC 24V)	—
ON delay time	20ms max.	1.5ms max.
OFF delay time	20ms max.	1.5ms max.
Number of circuits	32 points (8 points/common)	24 points (8/3/1 points/common)
ON voltage	DC 16V max.	—
OFF voltage	DC 5V min.	—
Max. switching capacity	—	DC 5 to 24V 0.5A max.
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	100μA max.
Saturation voltage	—	1.5V max.
External power supply	—	15mA/point, 360mA/unit
Service life (of relay)	—	(G3D-YO1P-PC DC 24V)
Circuit configuration		
		
	16 points×2=32 points	12 points×2=24 points

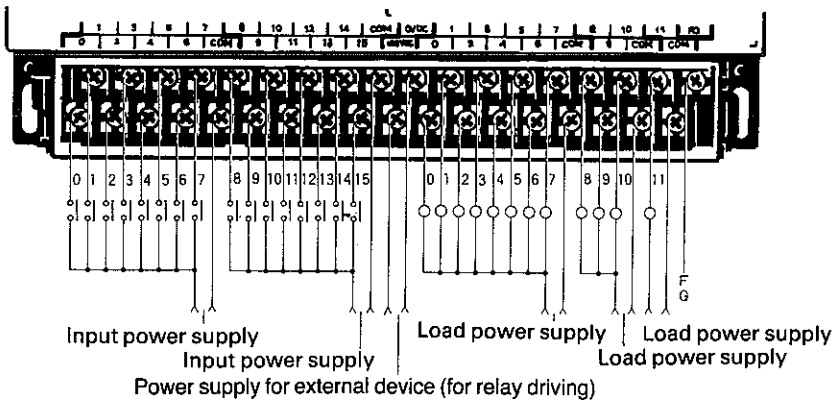
Terminal	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)	IN14 (0114)	COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
Terminal	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)	IN15 (0115)	+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM





Note:
Those of MD212 (with 32 input points/24 output points) are fixed on channels 1 and 2 (input), and 6 and 7 (output). When connecting an inductive load, connect a diode (1A, 100V) in parallel with the load.

* Detachable terminal block

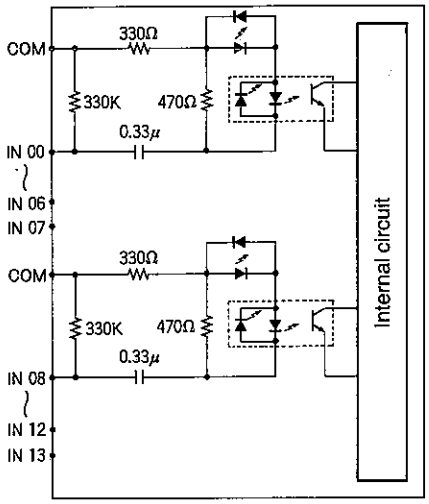


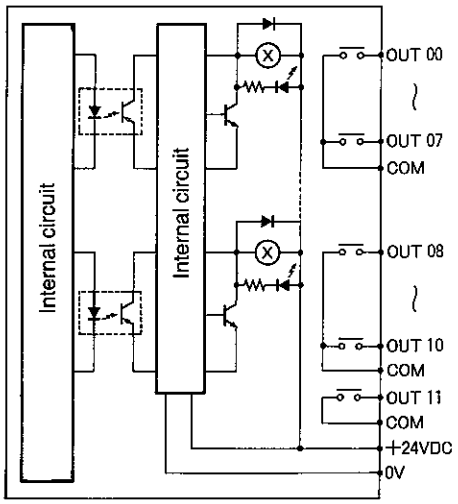
Terminal	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Signal name (I/O No.)	IN1 (0201)	IN3 (0203)	IN5 (0205)	IN7 (0207)	IN8 (0208)	IN10 (0210)	IN12 (0212)	IN14 (0214)	COM	0V	OUT1 (0701)	OUT3 (0703)	OUT5 (0705)	OUT7 (0707)	OUT8 (0708)	OUT10 (0710)	OUT11 (0711)	FG
Terminal	B17	B16	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Signal name (I/O No.)	IN0 (0200)	IN2 (0202)	IN4 (0204)	IN6 (0206)	COM	IN9 (0209)	IN11 (0211)	IN13 (0213)	IN15 (0215)	+24V DC	OUT0 (0700)	OUT2 (0702)	OUT4 (0704)	OUT6 (0706)	COM	OUT9 (0709)	COM	COM

Expansion I/O unit

Type	3G2C7-MC22B/-MC22C	
	AC 100 to 120V input	Contact output
Input voltage	AC 85 to 132V	—
Input impedance	9.7kΩ	—
Input current	10mA typ. (AC 100V)	—
ON delay time	35ms max.	15ms max.
OFF delay time	55ms max.	15ms max.
Number of circuits	14 points (8/6 points/common)	12 points (8/3/1 points/common)
ON voltage	AC 60V max.	—
OFF voltage	AC 20V min.	—
Max. switching capacity	—	AC 250V/DC 24V 2A max. (p.f.=1)
Min. switching capacity	—	DC 5V 10mA min.
Leakage current	—	—
Saturation voltage	—	—
External power supply	—	15mA/point, 180mA/unit
Service life (of relay)	—	Electrically: 300,000 operations Mechanically: 50,000,000 operations (G6B-1114P-US-M DC 24 V; with sokect)

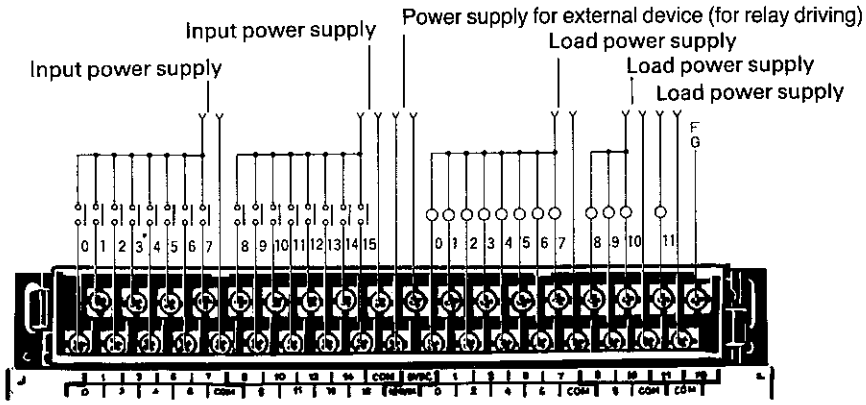
Circuit configuration





Terminal

	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)		COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
Terminal	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)		+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM

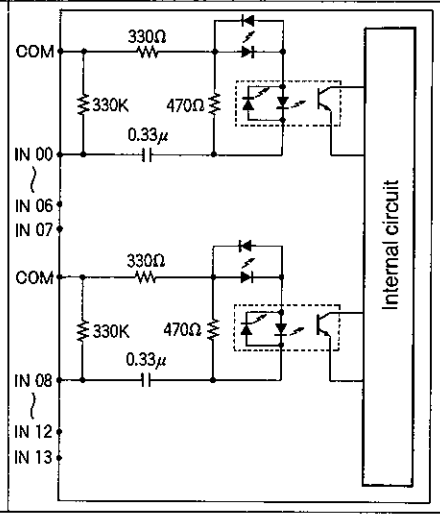


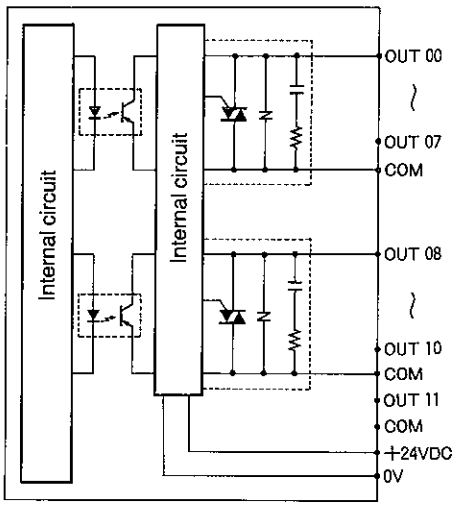
Note:
The input and output channels of MC22B (with 14 input points/12 output points) are fixed on channels 1 and 6. Those of MC22C (with 28 input points/24 output points) are fixed on channels 1 and 2 (input), and 6 and 7 (output).

Expansion I/O unit

Type	3G2C7-MA221	
	AC 100 to 120V input	Triac output
Input voltage	AC 85 to 132V	—
Input impedance	9.7kΩ (50Hz)/8kΩ (60Hz)	—
Input current	10mA typ. (AC 100V)	—
ON delay time	35ms max.	1.5ms max.
OFF delay time	55ms max.	1/2 of load frequency max.
Number of circuits	14 points (8/6 points/common)	12 points (8/3/1 points/common)
ON voltage	AC 60V max.	—
OFF voltage	AC 20V min.	—
Max. switching capacity	—	AC 85 to 250V/0.2A max. (p.f.=1)
Min. switching capacity	—	10mA (AC 100V)/20mA (AC 200V) min.
Leakage current	—	2mA (AC 100V)/5mA (AC 200V) max.
Saturation voltage	—	1.6V max.
External power supply	—	20mA/point, 240mA/unit
Service life (of relay)	—	(G3S-201PL-PC DC 24V)

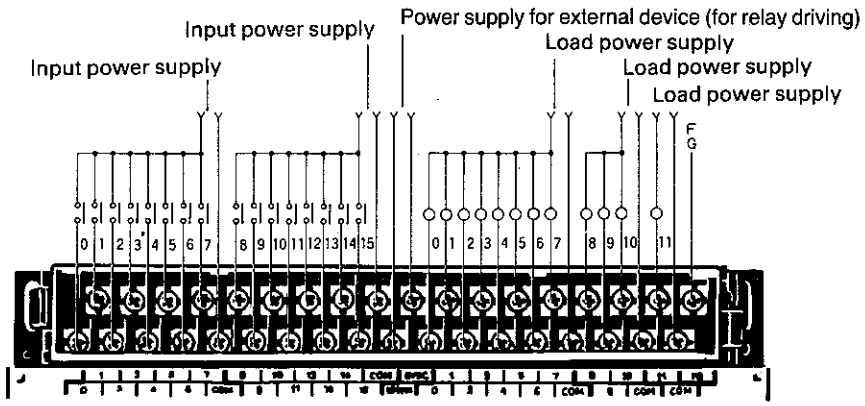
Circuit configuration





Terminal Signal name (I/O No.) / Terminal Signal name (I/O No.)

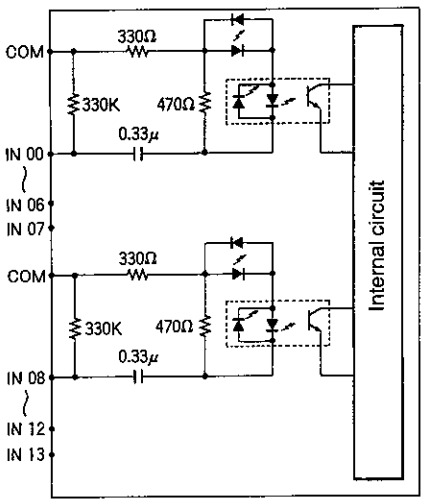
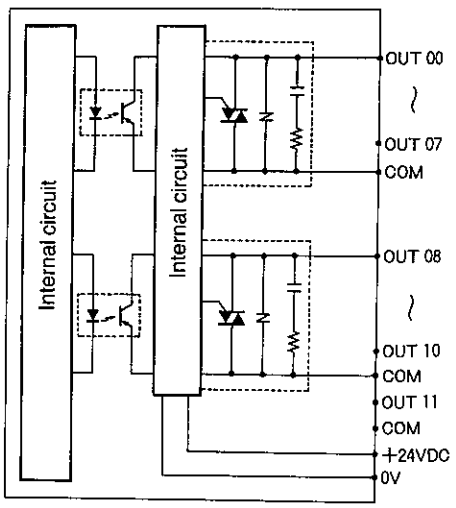
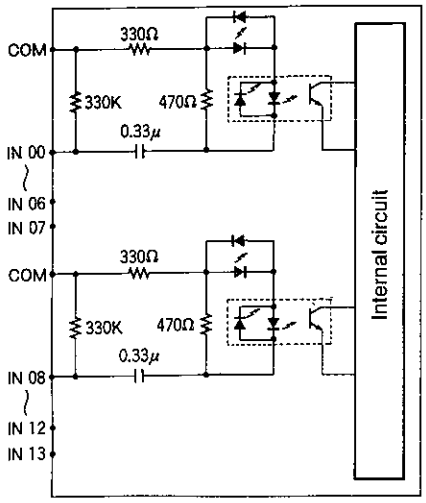
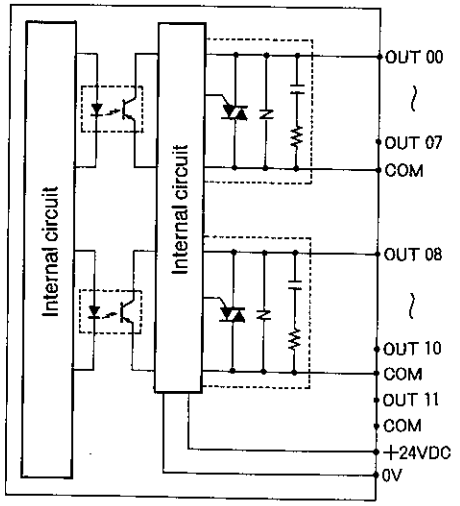
	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)		COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)		+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM



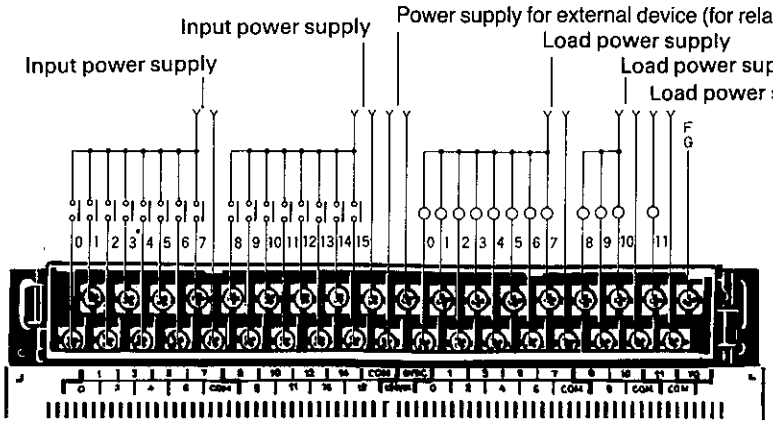
Note:
The input and output channels of MA221 (with 14 input points/12 output points) are fixed on channels 1 and 6.

* Detachable terminal block

Expansion I/O unit

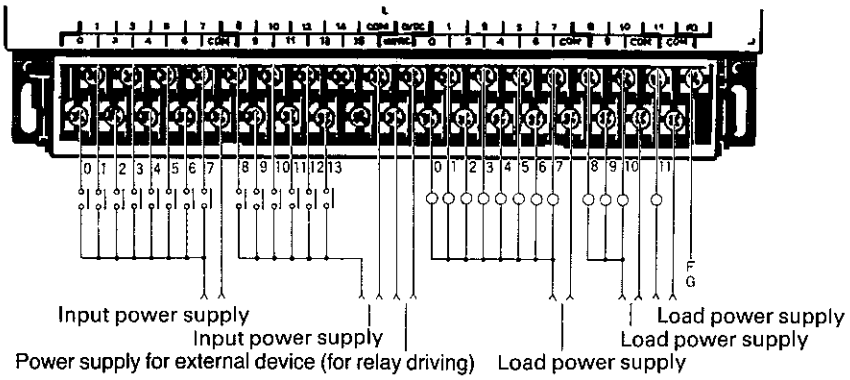
Type	3G2C7-MA222	
	AC 100 to 120V input	Triac output
Input voltage	AC 85 to 132V	—
Input impedance	9.7kΩ (50Hz)/8kΩ (60Hz)	—
Input current	10mA typ. (AC 100V)	—
ON delay time	35ms max.	1.5ms max.
OFF delay time	55ms max.	1/2 of load frequency max.
Number of circuits	28 points (8/6 points/common)	24 points (8/3/1 points/common)
ON voltage	AC 60V max.	—
OFF voltage	AC 20V min.	—
Max. switching capacity	—	AC 85 to 250V/0.2A max. (p.f.=1)
Min. switching capacity	—	10mA (AC 100V)/20mA (AC 200V) min.
Leakage current	—	2mA (AC 100V)/5mA (AC 200V) max.
Saturation voltage	—	1.6V max.
External power supply	—	20mA/point, 480mA/unit
Service life (of relay)	—	(G3S-201PL-PC DC 24V)
Circuit configuration		
		
	14 points×2=28 points	12 points×2=24 points

Terminal	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
Signal name (I/O No.)	IN1 (0101)	IN3 (0103)	IN5 (0105)	IN7 (0107)	IN8 (0108)	IN10 (0110)	IN12 (0112)		COM	0V	OUT1 (0601)	OUT3 (0603)	OUT5 (0605)	OUT7 (0607)	OUT8 (0608)	OUT10 (0610)	OUT11 (0611)	FG
Terminal	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17
Signal name (I/O No.)	IN0 (0100)	IN2 (0102)	IN4 (0104)	IN6 (0106)	COM	IN9 (0109)	IN11 (0111)	IN13 (0113)		+24V DC	OUT0 (0600)	OUT2 (0602)	OUT4 (0604)	OUT6 (0606)	COM	OUT9 (0609)	COM	COM



Note:
Those of MA222 (with 28 input points/24 output points) are fixed on channels 1 and 2 (input), and 6 and 7 (output).

* Detachabale terminal block



Terminal	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Signal name (I/O No.)	IN1 (0201)	IN3 (0203)	IN5 (0205)	IN7 (0207)	IN8 (0208)	IN10 (0210)	IN12 (0212)		COM	0V	OUT1 (0701)	OUT3 (0703)	OUT5 (0705)	OUT7 (0707)	OUT8 (0708)	OUT10 (0710)	OUT11 (0711)	FG
Terminal	B17	B16	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Signal name (I/O No.)	IN0 (0200)	IN2 (0202)	IN4 (0204)	IN6 (0206)	COM	IN9 (0209)	IN11 (0211)	IN13 (0213)		+24V DC	OUT0 (0700)	OUT2 (0702)	OUT4 (0704)	OUT6 (0706)	COM	OUT9 (0709)	COM	COM

I/O link unit

Ratings

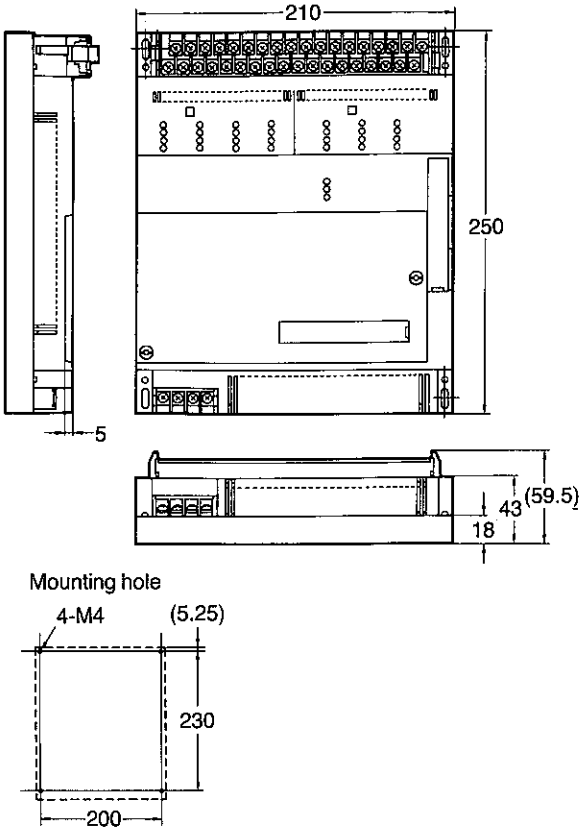
Supply voltage	AC 100 to 120 V
Operating voltage range	AC 85 to 132 V
Power consumption	15 VA max.
Insulation resistance	5MΩ min. at DC 500V (between external terminal and outer casing)
Dielectric strength	AC 1,500V 50/60Hz for 1 minute (between external terminal and outer casing)
Noise immunitiy	1,000Vp-p Rise time: 1ns; Pulse width: 100ns to 1μs
Vibration	16.7Hz, 3mm double amplitude (in X, Y, and Z directions, respectively for 30 minutes)
Shock	10G (in X, Y, and Z directions, respectively 3 times)
Ambient temperature	Operating: 0° to +55°C Storage: −20° to +65°C
Humidity	35 to 85% RH (without condensation)
Atmosphere	Must be free from corrosive gases
Degree of protection	IP30

Characteristics

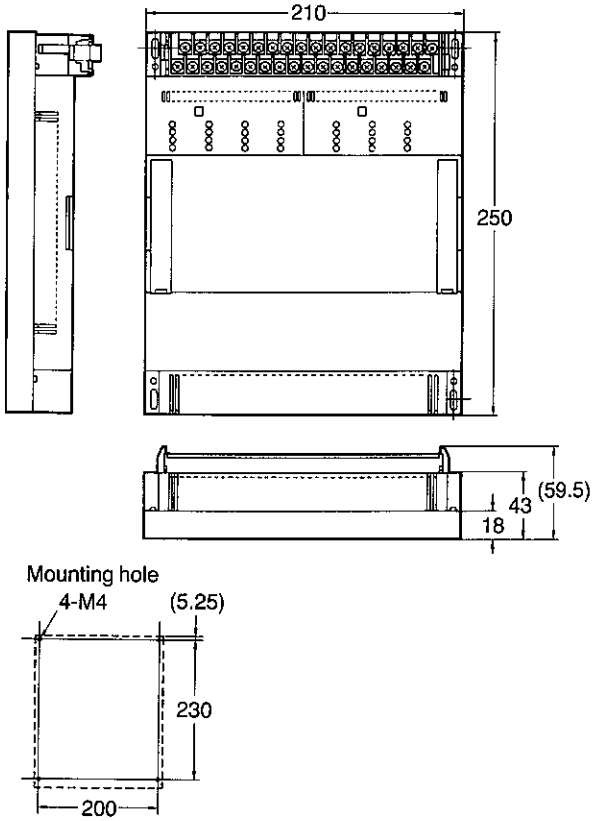
Transmission system	Time-division multiplexing cyclic system
Communication method	Half-duplex
Transmission speed	187.5k bps
Transmission delay	4ms/32 points
Transmission lines	Polymer-clad quartz core optical-fiber
Line distance	800m max.
Number of I/O points	32 (16 input and 16 output points)
External output	RUN Contact output, 2A max., SPST-NO (Model G6B) REPEATER Contact output, SPST-NO, (Model G6B), for repeater signal
Weight	1kg max.

Dimensions

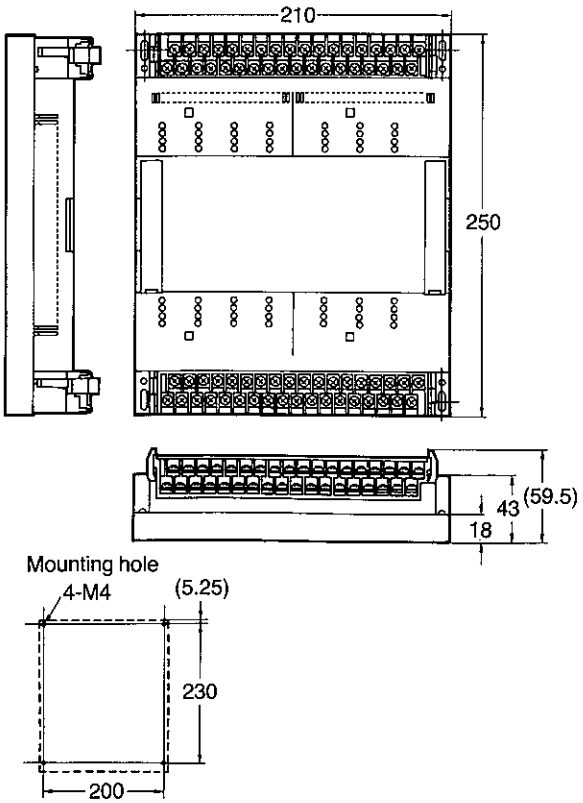
CPU



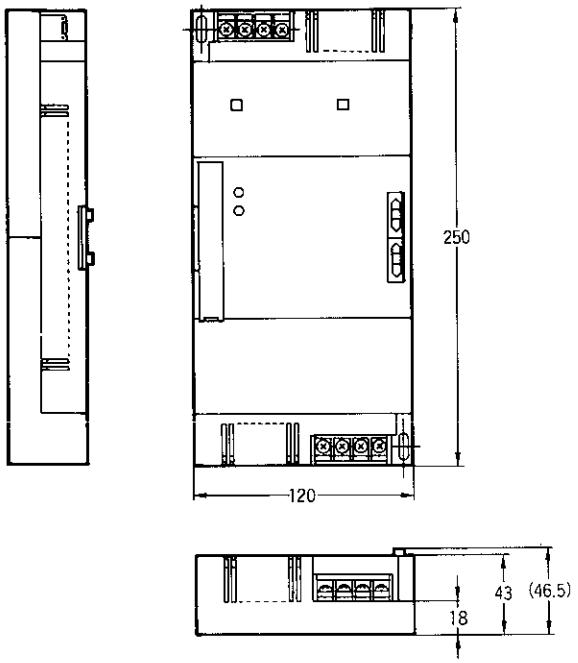
28-point expansion I/O unit

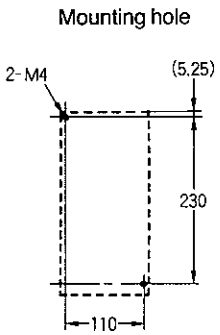


56-point expansion I/O unit

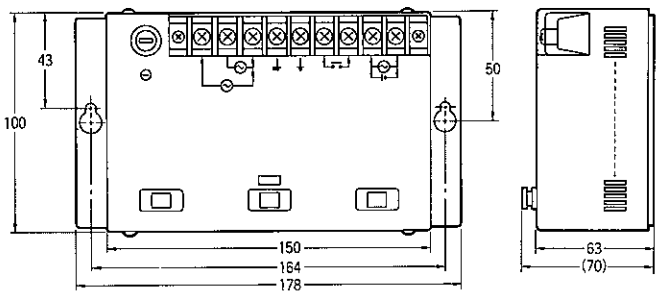


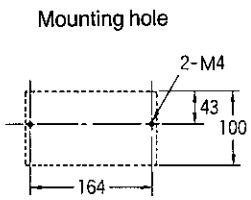
I/O link unit
(Type 3G2C7-LK011)



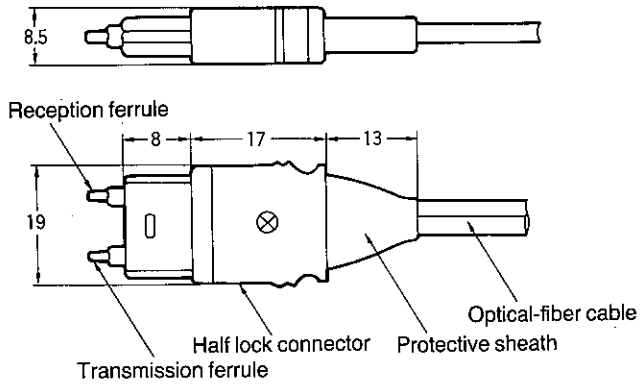


Link adapter
(Type 3G2A9-AL002-E)





Optical-fiber cable connector



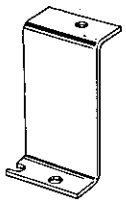
Appendix G Mounting kit

Overview

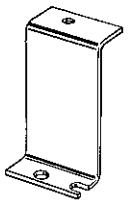
To save mounting space at the installation site, Type 3G2C7-PAT02 mounting kit is optionally available. This kit includes four mounting screws, two pairs of mounting brackets, and one mounting plate.

Screw A (M4x5, x4)

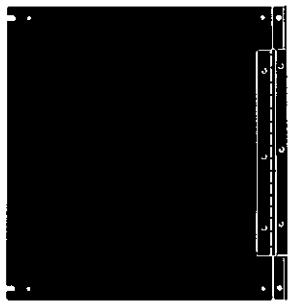
Bracket B (x2)



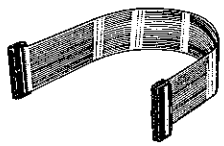
Bracket C (x2)



Mounting plate (x1)

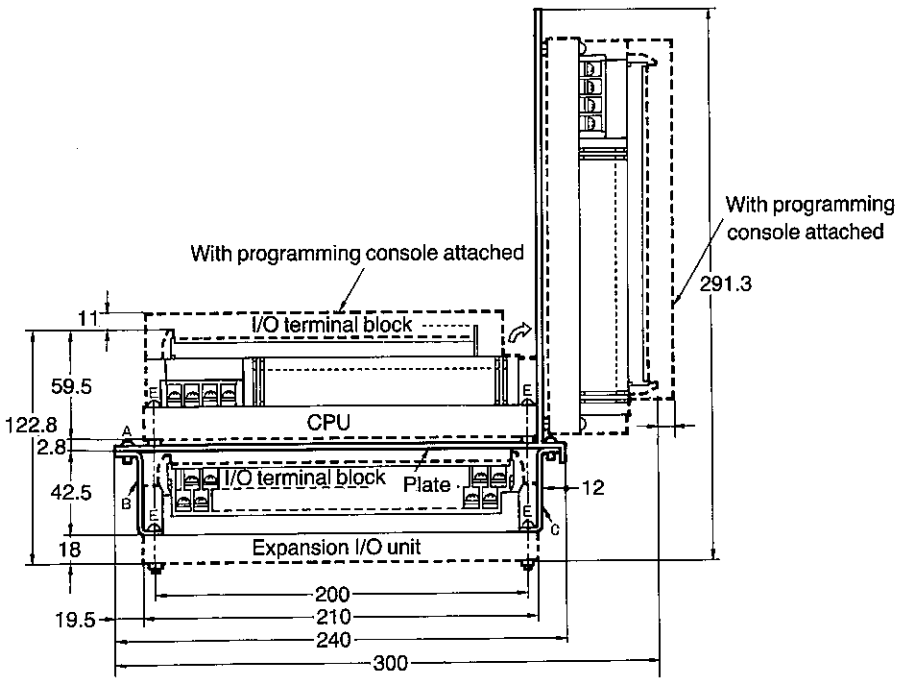


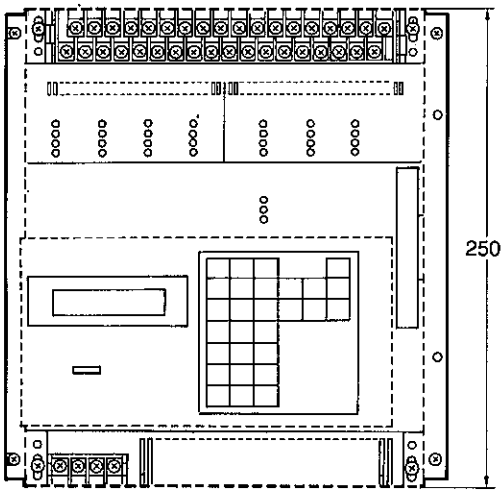
I/O connecting cable (3G2C7-CN311, 32cm) (option)

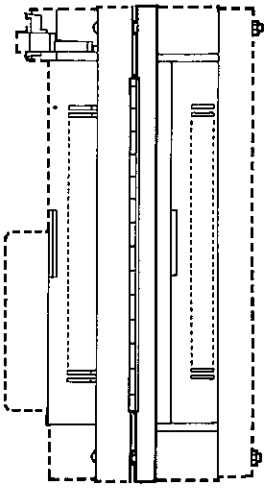


With this mounting kit, the CPU and one expansion I/O unit can be mounted in a sort of piggyback fashion, with the CPU on top and the expansion I/O unit on the bottom, saving mounting space. The CPU can be flipped over to the right, allowing you access to the indicators on the expansion I/O unit for monitoring and maintenance purposes.

Dimensions



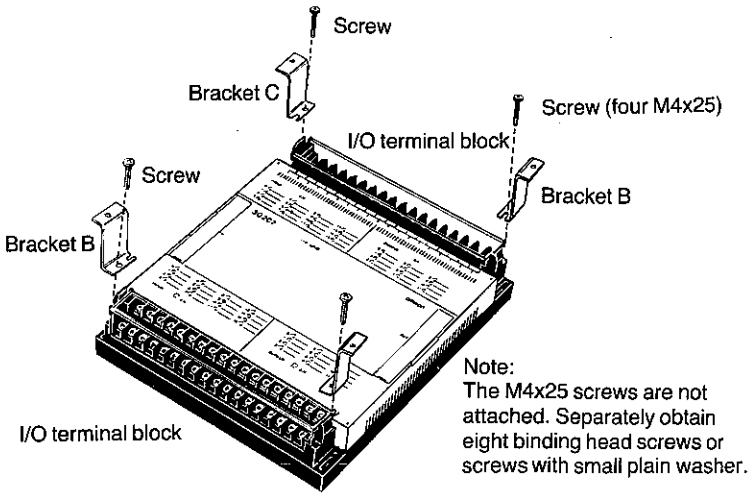




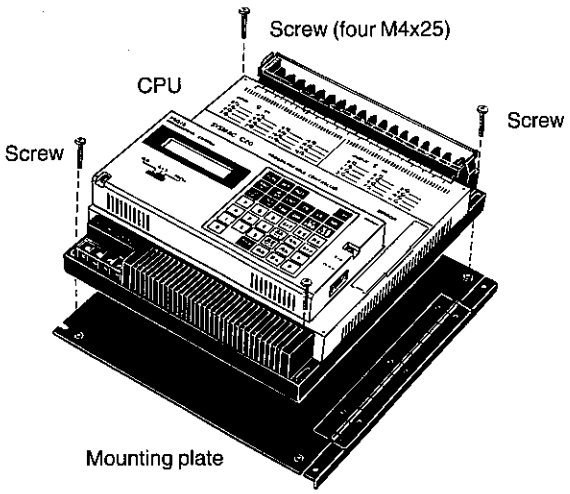
Mounting method

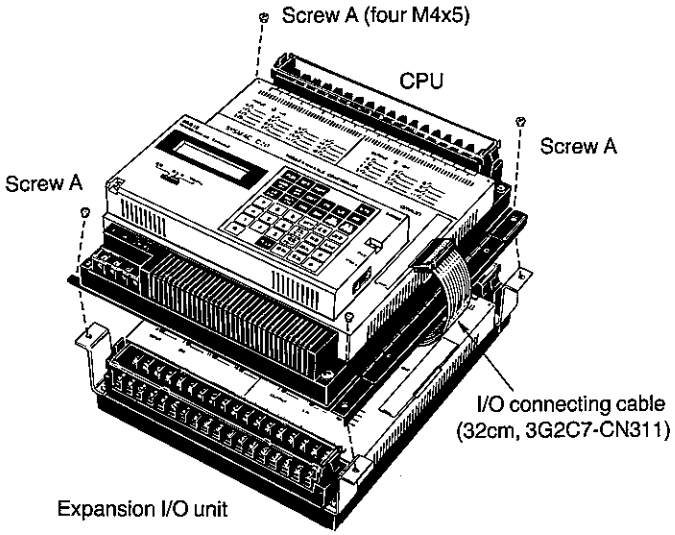
The mounting of the CPU and expansion I/O unit using this mounting kit can be performed in these three steps.

First, attach brackets B and C (two of each) to the expansion I/O unit and fix the brackets with four M4x25 screws.



Second, mount the CPU on the mounting plate with four M4x25 screws. Note that at this time the hinge of the mounting plate must face to the right. No nut is required to tighten the screws.

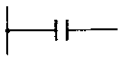
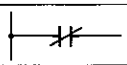
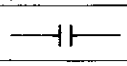
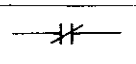
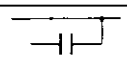
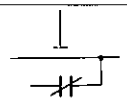
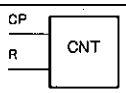




Finally, connect one end of the I/O connecting cable (Type 3G2C7-CN311, 32cm) to the left of the expansion I/O unit and connect the other end of the cable to the CPU. Fix the mounting plate with the CPU on it to the four mouting brackets with four M4x5 screws.

Appendix H List of instructions

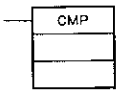
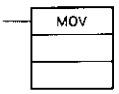
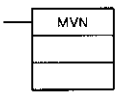
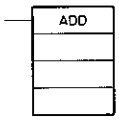
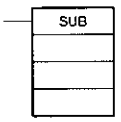
Basic instruction

Instruction	Symbol	Mnemonic	Operand	Function	Data
LOAD		LD	Relay No.	Logical start operation	Relay No. Input/output relays 0000 to 0915 Internal auxiliary relays 1000 to 1907 Holding relays HR000 to 915 Timers TIM00 to 47 Counters CNT00 to 47 Temporary memory relays TR0 to 7 (Temporary memory relays can only be used with the LD instruction.)
LOAD NOT		LD NOT	Relay No.	Logical NOT start operation	
AND		AND	Relay No.	Logical AND operation	
AND NOT		AND NOT	Relay No.	Logical AND NOT operation	
OR		OR	Relay No.	Logical OR operation	
OR NOT		OR NOT	Relay No.	Logical OR NOT operation	
AND LOAD		AND LD		Logical AND operation with the previous condition	
OR LOAD		OR LD		Logical OR operation with the previous condition	
OUT		OUT	Relay No.	Outputs the result of a logical operation to the specified output relay, internal auxiliary relay, latching relay, or shift register.	Relay No. Output relay 0500-0915 Internal auxiliary relays 1000 to 1807 Holding relays HR000 to 915 Temporary memory TR0 to TR7
TIMER		TIM	Timer No. Set value	ON-delay timer operation Set time: 0 to 999.9sec	Timers/counters 00 to 47 Set value Constant # 0000 to 9999
COUNTER		CNT	Counter No. Set count value	Down counter operation Set value: 0 to 9999	

Applied instructions

FUN No.	Instruction	Symbol	Mnemonic	Operand	Function	Data
00	NO FUNC-TION		FUN 0 0		Use this instruction when an instruction is added in the future. This instruction is also used for minute adjustment of scan time.	
01	END	—[END]	FUN 0 1		End of a program	
02	INTER-LOCK	—[IL]	FUN 0 2		Causes all the relay coils between this instruction and the ILC instruction to be reset or not reset according to the result immediately before this instruction.	
03	INTER-LOCK CLEAR	—[ILC]	FUN 0 3		Clears the IL instruction.	
10	SHIFT REGIS-TER	IN, CP, R, SFT	FUN 1 0	Start CH-No. CH-No. End	Shift register operation 15 0 15 0 End CH ... Start CH ← IN	Channel Numbers Output relay : 05 to 09CH Internal auxiliary relay : 10 to 17CH Holding relay : 0 to 9CH Start CH ≦ END CH Start and end channel can be same channel.
11	LATCH-ING RELAY	SET (KEEP), RESET	FUN 1 1	Relay No.	Latching relay operation	Relay No. Input output relay 0500 to 0915 Internal auxiliary relay 1000 to 1807 Holding relay HR000 to 915
13	DIFFER-ENTIA-TION UP	—[DIFU]	FUN 1 3	Relay No.	Causes a specified relay to operate for one scan time at the leading edge of the result of a logical arithmetic operation.	
14	DIFFER-ENTIA-TION DOWN	—[DIFD]	FUN 1 4	Relay No.	Causes a specified relay to operate for one scan time at the trailing edge of the result of a logical arithmetic operation.	
15	HIGH-SPEED TIMER	—(TIMH)	FUN 1 5	Timer No. Set value	Performs a high-speed on-delay (down type) timer operation. Set time: 00.00 to 99.99 sec	Timers/counters 00 to 47 Set value Constant # 0000 to 9999

List of instructions

FUN No.	Instruction	Symbol	Mnemonic	Operand	Function	Data
20	COM-PARE		FUN 2 0	S1 S2	Compares a channel data or a 4-digit constant against another channel data. $S_1 \gtreqless S_2$	S , S1 , S2 Input/output relays 00CH to 09CH Internal auxiliary relays 10CH to 19CH Holding relays HR0 to 9 Timers TIM00 to 47 Counters CNT00 to 47 Constant # 0000 to FFFF D Output relay 05 to 09CH Internal auxiliary relay 10 to 17CH Holding relay HR0 to 9CH
21	MOVE		FUN 2 1	S D	Transfers a channel data or a 4-digit constant (16 bits) to a specified channel. S → D	
22	MOVE NOT		FUN 2 2	S D	Inverts a channel data or a 4-digit constant and transfers it to a specified channel. $\overline{S} \rightarrow D$	
30	ADD		FUN 3 0	S1 S2 D	Performs BCD addition of a channel data or a 4-digit constant to a specified channel data. $S_1 + S_2 + CY = D, CY$	S1 S2 Input /output relay 00 to 19CH Holding relay HR0 to 9CH TIM/CNT00 to 47 Constant # 0000 to 9999 D Output relay 05 to 09CH Internal auxiliary relay 10 to 17CH Holding relay HR0 to 9CH
31	SUB-TRACT		FUN 3 1	S1 S2 D	Performs BCD subtraction of a channel data or a 4-digit constant from a specified channel data. $S_1 - S_2 - CY = D, CY$	
40	SET CARRY (STC)		FUN 4 0		Sets the carry (CY) to "1". 1 → CY	
41	CLEAR CARRY (CLC)		FUN 4 1		Clears the carry (CY) to "0". 0 → CY	

Index

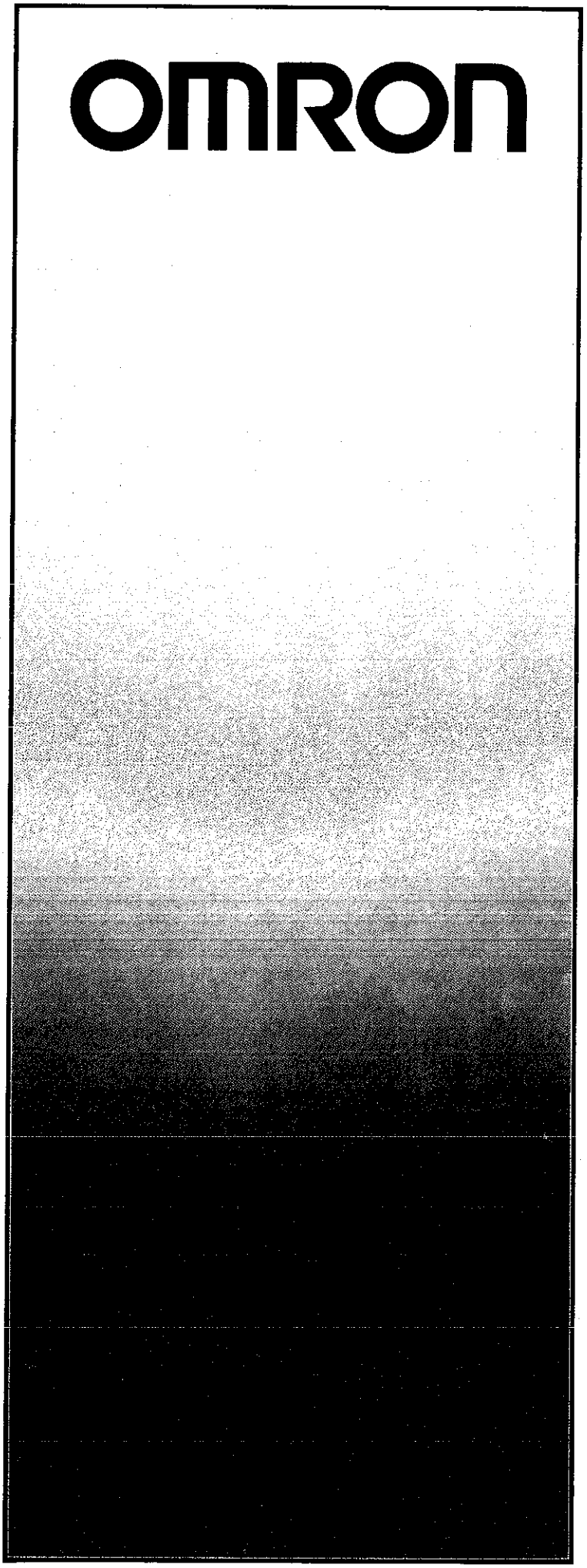
Index

I/O Assignment Table

Relay Nos		Description of signal	Relay NO		Description of signal
CH	00		CH	00	
	01			01	
	02			02	
	03			03	
	04			04	
	05			05	
	06			06	
	07			07	
	08			08	
	09			09	
	10			10	
	11			11	
	12			12	
	13			13	
	14			14	
	15			15	
CH	00		CH	00	
	01			01	
	02			02	
	03			03	
	04			04	
	05			05	
	06			06	
	07			07	
	08			08	
	09			09	
	10			10	
	11			11	
	12			12	
	13			13	
	14			14	
	15			15	

CODING SHEET

Program address	OP code	Relay No. (Data)	Remarks	Program address	OP code	Relay No. (Data)	Remarks
0				50			
1				51			
2				52			
3				53			
4				54			
5				55			
6				56			
7				57			
8				58			
9				59			
10				60			
11				61			
12				62			
13				63			
14				64			
15				65			
16				66			
17				67			
18				68			
19				69			
20				70			
21				71			
22				72			
23				73			
24				74			
25				75			
26				76			
27				77			
28				78			
29				79			
30				80			
31				81			
32				82			
33				83			
34				84			
35				85			
36				86			
37				87			
38				88			
39				89			
40				90			
41				91			
42				92			
43				93			
44				94			
45				95			
46				96			
47				97			
48				98			
49				99			



OMRON TATEISI ELECTRONICS CO.
Control Components H.Q.
9th Fl., Osaka Center Bldg. 4-68, Kitakyutaro, Higashiku,
Osaka 541 Japan
Phone: 06-282-2703 Fax: 06-282-2754
Telex: 522-2484 OMRONO J

OMRON ELECTRONICS INC.
1 East Commerce Drive, Schaumburg,
IL 60173, U.S.A.
Phone: (312) 843-7900 Fax: 312-843-8568
TWX: 910-291-0426 OMRONELEC SHBU

OMRON ELECTRONICS G.m.b.H.
Oberrather Strasse 6, D-4000 Düsseldorf 30
West Germany
Phone: (0211) 65020 Fax: (0211) 6502107 Telex: 8581890

OMRON ELECTRONICS (H.K.) LTD.
Unit 1605-6, Silvercord Tower 2,
30, Canton Road, Tsimshatsui,
Kowloon, Hong Kong
Phone: 3-7233827 (PBX) Fax: 3-7231475
Telex: 41092 OMRON HX

OMRON SINGAPORE (PTE.) LTD.
1298 Lorong 1, Toa Payoh #02-01, Singapore 1231
Phone: 2556988 Fax: 65-250-8245 Telex: RS23403

OMRON TAIWAN ELECTRONICS INC.
6th Fl., China Trust Bldg., N o.122 Tunhua North Road
Taipei, Taiwan R.O.C.
Phone: (02) 715-3331 Fax: (02) 712-6712

OMRON ELECTRONICS PTY. LTD.
Private Box No. 12.
Frenchs Forest Post Office
Frenchs Forest, N. S. W. 2086, Australia
Phone: (02) 975-1511 Fax: (02) 975-1518

Authorized distributor:

Printed in Japan
288-3M