MITSUBISHI ELECTRIC

MELFA

Industrial Robots

Instruction Manual
(Movemaster Commands)

CR1/CR2
Controller

Art. no.: 160277
26 02 2002
Version C
BFP-A8056

MITSUBISHI ELECTRIC INDUSTRIAL AUTOMATION

⚠ Safety Precautions

Always read the following precautions and the separate "Safety Manual" before starting use of the robot to learn the required measures to be taken.

⚠ CAUTION	All teaching work must be carried out by an operator who has received special training. (This also applies to maintenance work with the power source turned ON.) → Enforcement of safety training
⚠ CAUTION	For teaching work, prepare a work plan related to the methods and procedures of operating the robot, and to the measures to be taken when an error occurs or when restarting. Carry out work following this plan. (This also applies to maintenance work with the power source turned ON.) → Preparation of work plan
⚠WARNING	Prepare a device that allows operation to be stopped immediately during teaching work. (This also applies to maintenance work with the power source turned ON.) → Setting of emergency stop switch
⚠ CAUTION	During teaching work, place a sign indicating that teaching work is in progress on the start switch, etc. (This also applies to maintenance work with the power source turned ON.) → Indication of teaching work in progress
⚠WARNING	Provide a fence or enclosure during operation to prevent contact of the operator and robot. → Installation of safety fence
⚠ CAUTION	Establish a set signaling method to the related operators for starting work, and follow this method. → Signaling of operation start
⚠ CAUTION	As a principle turn the power OFF during maintenance work. Place a sign indicating that maintenance work is in progress on the start switch, etc. → Indication of maintenance work in progress
⚠ CAUTION	Before starting work, inspect the robot, emergency stop switch and other related devices, etc., and confirm that there are no errors. → Inspection before starting work

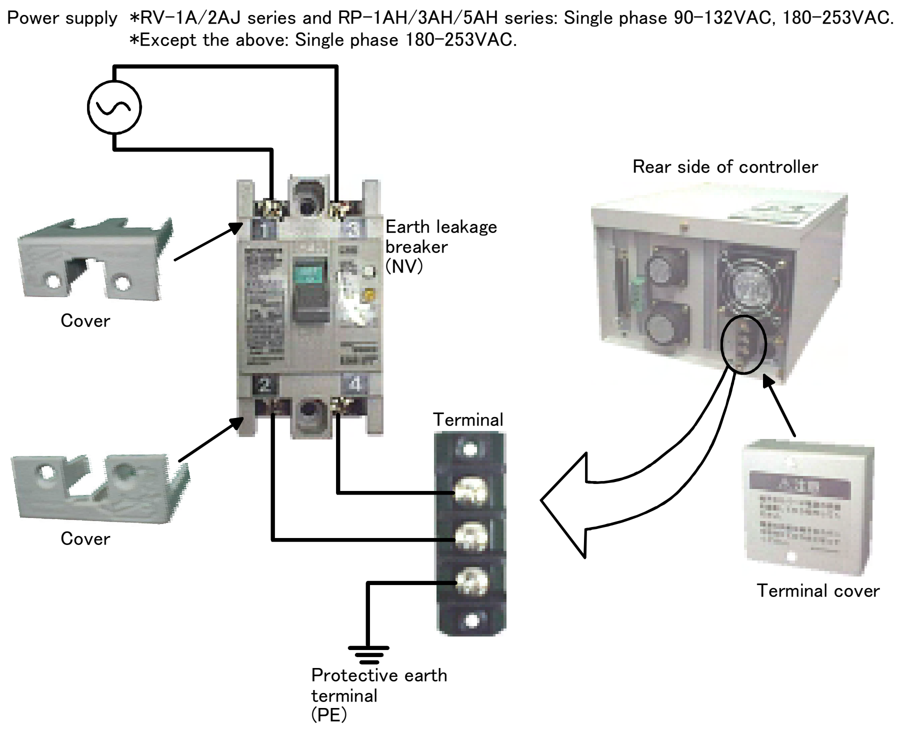
The points of the precautions given in the separate "Safety Manual" are given below.
Refer to the actual "Safety Manual" for details.

⚠ CAUTION	Use the robot within the environment given in the specifications. Failure to do so could lead to a drop or reliability or faults. (Temperature, humidity, atmosphere, noise environment, etc.)
⚠ CAUTION	Transport the robot with the designated transportation posture. Transporting the robot in a non-designated posture could lead to personal injuries or faults from dropping.
⚠ CAUTION	Always use the robot installed on a secure table. Use in an instable posture could lead to positional deviation and vibration.
⚠ CAUTION	Wire the cable as far away from noise sources as possible. If placed near a noise source, positional deviation or malfunction could occur.
⚠ CAUTION	Do not apply excessive force on the connector or excessively bend the cable. Failure to observe this could lead to contact defects or wire breakage.
⚠ CAUTION	Make sure that the workpiece weight, including the hand, does not exceed the rated load or tolerable torque. Exceeding these values could lead to alarms or faults.
⚠WARNING	Securely install the hand and tool, and securely grasp the workpiece. Failure to observe this could lead to personal injuries or damage if the object comes off or flies off during operation.
⚠WARNING	Securely ground the robot and controller. Failure to observe this could lead to malfunctioning by noise or to electric shock accidents.
⚠ CAUTION	Indicate the operation state during robot operation. Failure to indicate the state could lead to operators approaching the robot or to incorrect operation.
⚠WARNING	When carrying out teaching work in the robot's movement range, always secure the priority right for the robot control. Failure to observe this could lead to personal injuries or damage if the robot is started with external commands.
⚠ CAUTION	Keep the jog speed as low as possible, and always watch the robot. Failure to do so could lead to interference with the workpiece or peripheral devices.
⚠ CAUTION	After editing the program, always confirm the operation with step operation before starting automatic operation. Failure to do so could lead to interference with peripheral devices because of programming mistakes, etc.
⚠ CAUTION	Make sure that if the safety fence entrance door is opened during automatic operation, the door is locked or that the robot will automatically stop. Failure to do so could lead to personal injuries.
⚠ CAUTION	Never carry out modifications based on personal judgments, or use non-designated maintenance parts. Failure to observe this could lead to faults or failures.
⚠WARNING	When the robot arm has to be moved by hand from an external area, do not place hands or fingers in the openings. Failure to observe this could lead to hands or fingers catching depending on the posture.
⚠ CAUTION	Do not stop the robot or apply emergency stop by turning the robot controller's main power OFF. If the robot controller main power is turned OFF during automatic operation, the robot accuracy could be adversely affected.

C.Precautions for the basic configuration are shown below.(When CR1-571 is used for the controller.)

⚠ CAUTION

Provide an earth leakage breaker that packed together on the primary power supply of the controller as protection against electric leakage. Confirm the setting connector of the input power supply voltage of the controller, if the type which more than one power supply voltage can be used. Then connect the power supply.
Failure to do so could lead to electric shock accidents.


Power supply *RV-1A/2AJ series and RP-1AH/3AH/5AH series: Single phase 90-132VAC, 180-253VAC.
*Except the above: Single phase 180-253VAC.
Rear side of controller
Earth leakage breaker (NV)
Cover
Cover
Terminal
Terminal cover
Protective earth terminal (PE)

⚠WARNING

For using RH-5AH/10AH/15AH.
While pressing the brake releasing switch on the robot arm, beware of the arm which may drop with its own weight.
Dropping of the hand could lead to a collision with the peripheral equipment or catch the hands or fingers.

Revision history

Date of print	Specifications No.	Details of revisions
1999-11-02	BFP-A8056Z	First print.
1999-12-28	BFP-A8056Z-A	Error in writing correction.
2000-04-05	BFP-A8056Z-B	Error in writing correction.
2000-05-11	BFP-A8056	Formal style.
2000-12-11	BFP-A8056-A	Error in writing correction.
2002-1-18	BFP-A8056-B	The description which varies in the axis structure of the robot was added. (PD, MP, PR, MJ, DJ, TLM, PRN)] Error in writing correction.
2002-02-26	BFP-A8056-C	Error in writing correction.
2005-01-06	BFP-A8056-D	Change title.

■ Introduction

Thank you for purchasing the Mitsubishi industrial robot.
This instruction manual explains the commands and describes sample programs for the ″MOVEMASTER COMMANDS″.
Always read through this manual before starting use to ensure correct usage of the robot.
Note that there may be cases when these MOVEMASTER COMMANDS cannot be used because of the model. Refer to the standard specifications of the corresponding product and check whether these commands can be used. The information contained in this document has been written to be accurate as much as possible. Please interpret that items not described in this document ″cannot be performed.″.

Applicable model list. (Jan. 2002)

*RP-1AH/3AH/5AH series
*RV-1A/2AJ
*RV-2A/3AJ series
*RV-4A/5AJ series
*RH-5AH/10AH/15AH series

CONTENTS

1 Before starting use

This chapter explains the details and usage methods of the instruction manuals, the basic terminology and the safety precautions.

1.1 Using the instruction manuals

1.1.1 The details of each instruction manuals

The contents and purposes of the documents enclosed with this product are shown below. Use these documents according to the application.
For special specifications, a separate instruction manual describing the special section may be enclosed.

Safety Manual	Explains the common precautions and safety measures to be taken for robot handling, system design and manufacture to ensure safety of the operators involved with the robot.
Standard Specifications	Explains the product's standard specifications, factory-set special specifications, option configuration and maintenance parts, etc. Precautions for safety and technology, when incorporating the robot, are also explained.
Robot Arm Setup & Maintenance	Explains the procedures required to operate the robot arm (unpacking, transportation, installation, confirmation of operation), and the maintenance and inspection procedures.
Controller Setup, Basic Operation and Maintenance	Explains the procedures required to operate the controller (unpacking, transportation, installation, confirmation of operation), basic operation from creating the program to automatic operation, and the maintenance and inspection procedures.
Detailed Explanation of Functions and Operations	Explains details on the functions and operations such as each function and operation, commands used in the program, connection with the external input/output device, and parameters, etc.
Explanations of MOVEMASTER COMMANDS	Explains details on the MOVEMASTER commands used in the program. (For RV-1A/2AJ and RV-2A/3AJ series)
Troubleshooting	Explains the causes and remedies to be taken when an error occurs. Explanations are given for each error No.

1.1.2 Symbols used in instruction manual

The symbols and expressions shown in Table 1-1 are used throughout this User's Manual. Learn the meaning of these symbols before reading this instruction manual.

Table 1-1 : Symbols in instruction manual

Symbol	Meaning
⚠DANGER	Precaution indicating cases where there is a risk of operator fatality or serious injury if handling is mistaken. Always observe these precautions to safely use the robot.
⚠WARNING	Precaution indicating cases where the operator could be subject to fatalities or serious injuries if handling is mistaken. Always observe these precautions to safely use the robot.
⚠CAUTION	Precaution indicating cases where operator could be subject to injury or physical damage could occur if handling is mistaken. Always observe these precautions to safely use the robot.
[JOINT]	If a word is enclosed in brackets or a box in the text, this refers to a key on the teaching pendant.
[＋／ ＦＯＲＷＤ] ＋ [＋ Ｘ] （Ａ）　　　　（Ｂ）	This indicates to press the (B) key while holding down the (A) key. In this example, the [+/Forward] key is pressed while holding down the [+X/+Y] key.
[ＳＴＥＰ／ＭＯＶＥ] ＋([ＣＯＮＤ]→[ＲＰＬ ↓]) （Ａ）　　　　　（Ｂ）　　（Ｃ）	This indicates to hold down the (A) key, press and release the (B) key, and then press the (C) key. In this example, the [Step/Move] key is held down, the [Condition] key is pressed and released, and the [Replace ↓ key is pressed.
Ｔ ／ Ｂ	This indicates the teaching pendant.

1.2 Safety Precautions

Always read the following precautions and the separate "Safety Manual" before starting use of the robot to learn the required measures to be taken.

⚠ CAUTION All teaching work must be carried out by an operator who has received special training. (This also applies to maintenance work with the power source turned ON.)
→ Enforcement of safety training

⚠ CAUTION For teaching work, prepare a work plan related to the methods and procedures of operating the robot, and to the measures to be taken when an error occurs or when restarting. Carry out work following this plan. (This also applies to maintenance work with the power source turned ON.)
→ Preparation of work plan

⚠ WARNING Prepare a device that allows operation to be stopped immediately during teaching work. (This also applies to maintenance work with the power source turned ON.)
→ Setting of emergency stop switch

⚠ CAUTION During teaching work, place a sign indicating that teaching work is in progress on the start switch, etc. (This also applies to maintenance work with the power source turned ON.)
→ Indication of teaching work in progress

⚠ DANGER Provide a fence or enclosure during operation to prevent contact of the operator and robot.
→ Installation of safety fence

⚠ CAUTION Establish a set signaling method to the related operators for starting work, and follow this method.
→ Signaling of operation start

⚠ CAUTION As a principle turn the power OFF during maintenance work. Place a sign indicating that maintenance work is in progress on the start switch, etc.
→ Indication of maintenance work in progress

⚠ CAUTION Before starting work, inspect the robot, emergency stop switch and other related devices, etc., and confirm that there are no errors.
→ Inspection before starting work

1.2.1 Precautions given in the separate Safety Manual
The points of the precautions given in the separate ″Safety Manual″ are given below.
Refer to the actual ″Safety Manual″ for details.

⚠ CAUTION	Use the robot within the environment given in the specifications. Failure to do so could lead to a drop or reliability or faults. (Temperature, humidity, atmosphere, noise environment, etc.)
⚠ CAUTION	Transport the robot with the designated transportation posture. Transporting the robot in a non-designated posture could lead to personal injuries or faults from dropping.
⚠ CAUTION	Always use the robot installed on a secure table. Use in an instable posture could lead to positional deviation and vibration.
⚠ CAUTION	Wire the cable as far away from noise sources as possible. If placed near a noise source, positional deviation or malfunction could occur.
⚠ CAUTION	Do not apply excessive force on the connector or excessively bend the cable. Failure to observe this could lead to contact defects or wire breakage.
⚠ CAUTION	Make sure that the workpiece weight, including the hand, does not exceed the rated load or tolerable torque. Exceeding these values could lead to alarms or faults.
⚠ WARNING	Securely install the hand and tool, and securely grasp the workpiece. Failure to observe this could lead to personal injuries or damage if the object comes off or flies off during operation.
⚠ WARNING	Securely ground the robot and controller. Failure to observe this could lead to malfunctioning by noise or to electric shock accidents.
⚠ CAUTION	Indicate the operation state during robot operation. Failure to indicate the state could lead to operators approaching the robot or to incorrect operation.
⚠ WARNING	When carrying out teaching work in the robot's movement range, always secure the priority right for the robot control. Failure to observe this could lead to personal injuries or damage if the robot is started with external commands.
⚠ CAUTION	Keep the jog speed as low as possible, and always watch the robot. Failure to do so could lead to interference with the workpiece or peripheral devices.
⚠ CAUTION	After editing the program, always confirm the operation with step operation before starting automatic operation. Failure to do so could lead to interference with peripheral devices because of programming mistakes, etc.
⚠ CAUTION	Make sure that if the safety fence entrance door is opened during automatic operation, the door is locked or that the robot will automatically stop. Failure to do so could lead to personal injuries.
⚠ CAUTION	Never carry out modifications based on personal judgments, or use non-designated maintenance parts. Failure to observe this could lead to faults or failures.
⚠ WARNING	When the robot arm has to be moved by hand from an external area, do not place hands or fingers in the openings. Failure to observe this could lead to hands or fingers catching depending on the posture.
⚠ CAUTION	Do not stop the robot or apply emergency stop by turning the robot controller's main power OFF. If the robot controller main power is turned OFF during automatic operation, the robot accuracy could be adversely affected.

2 Explanation of Commands

2.1 Setting the parameters

The parameters must be set before using the MOVEMASTER commands.
Change the working language parameter: RLNG value to "0" as shown in Table 2-1, and then turn the controller power ON and OFF before starting use.
Refer to the separate manual "Detailed Explanations of Functions and Operations" for details on changing the parameters.

Table 2-1 : Parameter: RLNG setting

Setting value	Working language	Remarks
1	MELFA-BASIC Ⅳ	Default setting
0	MOVEMASTER COMMANDS	

Note) When the MOVEMASTER COMMANDS is selected, the multitask function cannot be used.

2.2 Setting of RS-232C

(1) Communication specification

When RS-232C is used, it is necessary to set setting the communication shown in Table 2-2 in the same value on the robot controller side and the personal computer side. When this is not the same, it is not possible to communicate normally.

Table 2-2 : Communication specification

Item	Set value
Baud rate	9600 BPS
Data bit length	8 bit
Parity bit	Even parity
Stop bit length	2 bit
Changing line code	CR

The set value shown in Table 2-2 can be changed with the value set when shipping by robot controller's "Communication setting" parameter.
Refer to separate manual "Detailed Explanations of Functions and Operations" for details.

(2) DTR control

To do the DTR control effectively, robot controller's "CDTR232" parameter is set in "1".

Table 2-3 : Parameter:CDTR232 setting

Set value	Content	Remarks
1	DTR control effective	
0	DTR control invalidity	Setting when shipping

Refer to separate manual "Detailed Explanations of Functions and Operations" for modification method of the parameter.
Refer to page 99, "Appendix 1 : RS-232C signal conductor's timing chart (DTR control)" , because the timing chart of the controll signal line is specified.

2.3 Overview of commands

This section gives an overview of a wealth of commands provided for the MOVEMASTER command method.Commands are classified mainly into six types to the following. You should use appropriate commands according to your application. (Refer page 91, "3Command List")

(1) Position and motion control commands

These commands are concerned with definition of position and coordinates as well as assignment of interpolation, speed, timer, tool, palette, etc.

(2) Program control commands

These commands are concerned with conditional branching, repetitive operation, interrupting of signals, starting and stopping, counter operation, etc.

(3) Hand control command

These commands are concerned with opening/closing action of hand. For the optional motor-operated hand, you can control gripping force by parameter setting.

(4) I/O control command

These commands are concerned with input/output control of external I/O. Both single and parallel bits can be handled enabling the logical operation in the internal register.

(5) Communication command (Using RS-232C)

These commands transfer internal information of the robot such as counters, positions, program list, input/output signal status, parameters to personal computer.

(6) Other command

These commands are concerned with setting of parameters, selecting of programs, resetting of errors, and describing of comments.

2.4 Explanation of command

All commands are explained in alphabetical order to the following

(1) A viewpoint of command explanation

This shows the meaning of title described in the explanation of each command.

1) * Symbol	:The command that has * mark can not be described in the program with line number. (If done, it will become error.) You should execute the commands directly by personal computer or teaching box.
2) 【 Function 】	:Shows the brief Function of command.
3) 【 Input Format 】	:Shows the argument of the command entry. < > indicates the command parameter.[ ] indicates the parameter that can be omitted.
4) 【Term】	:Shows the meaning and limited range of command argument.
5) 【 Explanation 】	:Explains detailed Function and precautions.
6) 【 Relating Parameters 】	:Shows the parameters that have relationship with the command.
7) 【 Sample program 】	:Gives a typical program using MOVEMASTER command and BASIC language.

(2) Line

Line consists of line number and a command sentence. The length of one line contains 120 characters in maximum.

(3) Position number

The range of position number is 1-999. When program is different from each other, positions from 1-900 are memorized as individual data even if they have the same number. On the other hand, positions from 901-999 are memorized as common.

Further, you can appoint position number indirectly by the value of counter.

```
An example  10 SC 21 ,10     :Sets value 10 in counter 21.
            20 MO @ 21       :Moves to position 10, the value of counter 21.
```

【 Relating Commands 】 CF,HE,MA,MC,MO,MR,MRA,MS,MT,MTS,PC,PL,PR,PX,SF

(4) Counter number

The range of counter number is 1−99. When program is different from each other, counters from 1−90 are memorized as individual data even if they have the same number. On the other hand, counters from 91−99 are memorized as common. Further, you can appoint counter number indirectly by the value of counter.

```
An example  10 SC 21 ,10      :Sets value10 in counter 21.
            20 IC @ 21        :Adds one to counter 10, the value of counter 21.
```

【 Relating Commands 】CL,CP,CR,DC,IC,OC,SC

(5) Character string numbers

By using a character string number, character strings (max. 120 alphanumeric or symbol characters) can be communicated to the external device over the serial channel.

The character string number can be specified from $1 to $99. If the program differs between $1 and $90, these will be registered as separate data even if the number is the same. On the other hand, $91 to $99 are common for all programs. Depending on the counter number value (details), the character string number cannot be specified indirectly.

```
An example  10 SC $1, "ABC"  :Character string "ABC" is set in character string  number 1
            20 CP $1         :The details of character string number 1 are set in the character string
                              register
            30 CL $2         :The details of the character string register are set in character string
                              number 2.
```

【 Relating commands 】CL, CP, CR, INP, LG, NE, EQ, SM, SC

(6) About the operation data and the comparison value

The operation data and the comparison value have the method of specifying the content of the method of directly specifying the numerical value (Specification by the decimal number and the hexadecimal number is possible) and the counter number.

When the numerical value is directly specified by the decimal number.When the numerical value is specifiedby the hexadecimal number, "&" is input to the head of the hexadecimal number.

When the content of the counter number is specified, "@" is input to the head of the counter number.

```
An example) 10 ADD 10        ;10 is added to internal register.
            20 ADD &000A     ;A(hexadecimal number) is added to internal register.
            30 SC 1, 10      ;10 is setted to counter number 1.
            40 ADD @1        ;The value of counter number 1(10) is added to internal register.
```

【Relating commands】: ADD, AN, DIV, EQ, LG, MUL, NE OR, SM, SUB, XO

(7) Direct execution

If you transfer command to the controller without line number by personal computer or teaching box, it will be executed instantaneously. (Not be programed.) Especially hold with care in case executing moving command. You can execute PRN and RS−232C communication command even if the program is running.

Note:Make the DTR control shown in the previous item "2.2 Setting of RS−232C" effective when you do direct execution with RS−232C.
A direct execution is to make the MODE changeover switch of the controller control panel AUTO (Ext.), and do it.

(8) Internal register

The data taken from external I/O are memorized in so called "internal register", then used for conditional branching, comparison, logical bit operation, counter setting, and etc. When you carry out conditional branching on the basis of counter value, you use this internal register too.

【 Relating Commands 】AN,CL,CP,DR,EQ,ID,INP,NE,OR,SM,TB,XO

(9) Character string register

The character string led in to the external device over the serial channel is stored in the memory called the "character string register". It is then used for making settings of the character string comparison or character string number to jump conditions.

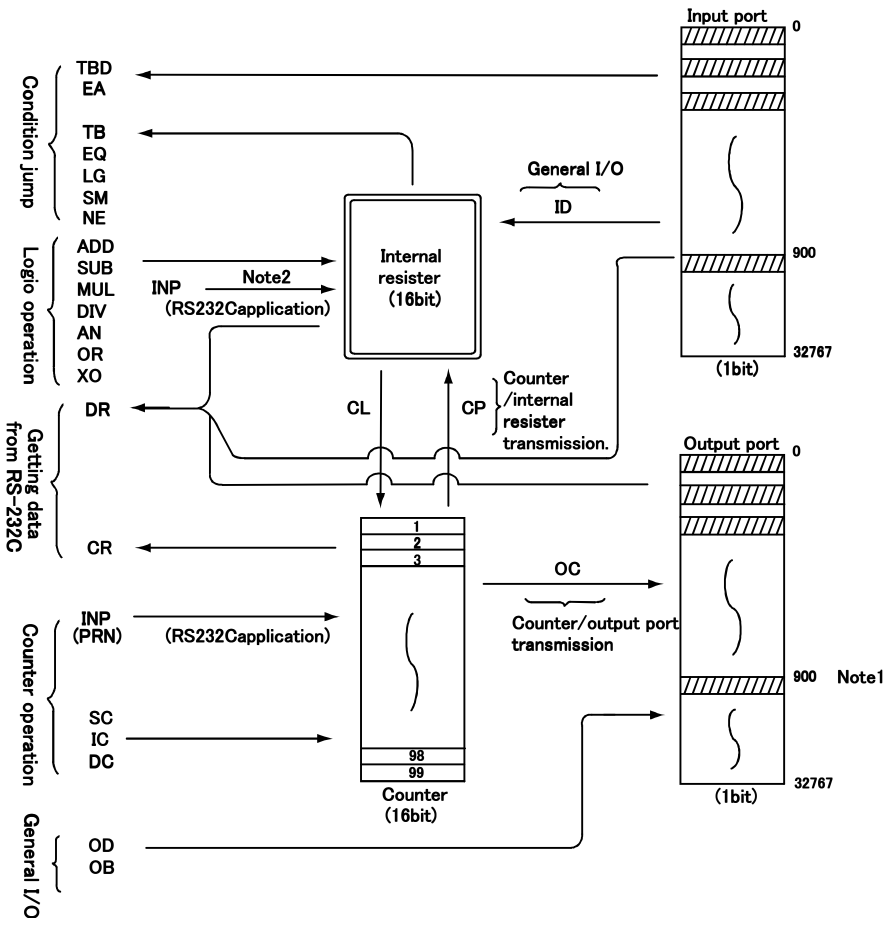
【 Relating commands】CL, CP, INP, LG, NE, EQ, SM

(10) Additional axis

When using the additional axis, specify the value of additional axis with J7(additional axis 1) or J8(additional axis 2) as for the joint coordinates, and specify with L1(additional axis 1) or L2(additional axis 2) as for the XYZ coordinates.

(11) Internal resister and related command. (Reference)

It shows it as follows about interior resister, counter, input port and flow of treatment of command concerned.

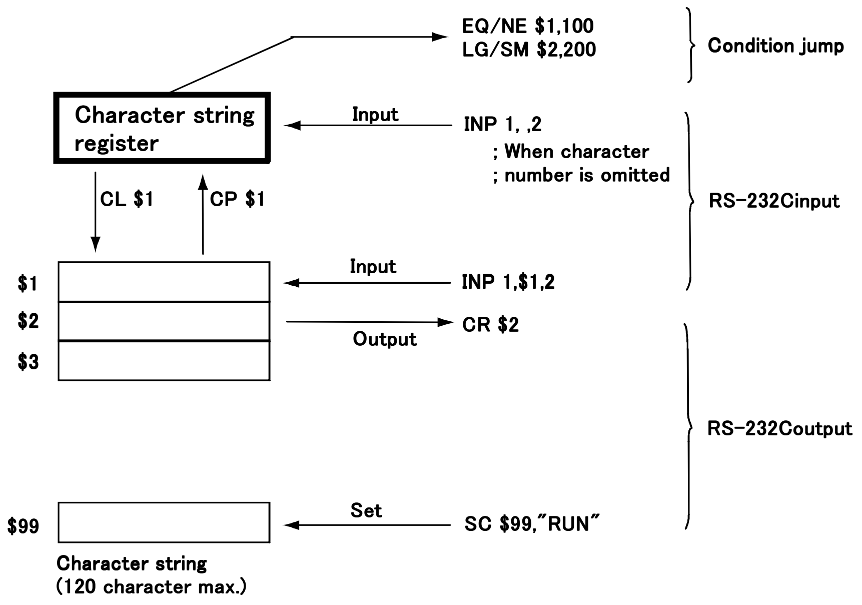


Note 1) Address 900 is the hand input/output.

Note 2) If the counter number is omitted, it will be set in the internal resister.

(12) Character string register and related commands (Reference)

The flow of the character string register, character string and related command process is shown below.



2.5 Explanation of each command

Hereafter, each command (instruction) is explained in alphabetical order.

ADD (Add)

【 Function 】
Adds the operation data to the value of the internal register, and stores it in the internal register.

【 Input Format 】

```
ADD   <Operation data>
```

【 Term 】

<Operation data>	Describes the data to be operated as a numeric value or counter No. with @.
	−32768 ≦ numeric value (decimal) ≦ 32767
	&8000 ≦ numeric value (hexadecimal) ≦ &7FFF
	@1 ≦ Counter No. ≦ @99

【 Explanation 】
(1)Designate the operation datasetting as a numeric value or counter No.
When designating with a numeric value, use a decimal or hexadecimal value. When using a hexadecimal, add a ″&″ to the head of the operation data.
When setting with a counter No. will be used as the operation data.
(2)The operation results are stored in the internal register, so operation, comparison and reading, etc., of the operation results can be carried out with the related commands.
(Refer to SUB, MUL, DIV. EQ, NE, LG, SM, CL, DR, OR, AN, XO commands)

【 Sample program 】(Movemaster command)

```
10 CP1                               ; Stores counter No.1 value in internal register
20 ADD @2                            ; Counter No.2 to the contents of multiplies internal register value.
30 CL3                               ; Sets internal register value in counter No.3
                                       (Counter No.3 = counter No.1 + counter No.2)
40 CP1                               ; Stores counter No.1 value in internal register
50 ADD 15                            ; Add 15 to the contents of multiplies internal register value.
60 CL4                               ; Sets internal register value in counter No.4
                                       (Counter No.4 = counter No.1 + 15)
```

AN (And)

【 Function 】
ANDs the specified value with the internal register, then stores the result to the internal register.

【 Input Format 】

```
AN   <operation data>
```

【 Term 】

<operation data>	Specify the data to be operated or counter No. with @. −32768 ≦ operation data (decimal) ≦ 32767 &8000 ≦ operation data (hexadecimal) ≦ &7FFF @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) Specify the data to be operated in decimal or hexadecimal.
Any hexadecimal value must be headed by "&".
(2) The operation result is stored into the internal register and can be changed, compared or read by relevant commands. (See the EQ, NE, LG, SM, CL, DR, OR commands)
(3) Execution of the AN command after the input commands (ID) allows receiving of only the required bits of the input data fetched from the external device.

【 Sample program 】 (Movemaster command)

```
10 ID                        ; Fetches data from external input port.
20 AN &000F                  ; Receives four lower bits only
30 CL 12                     ; Loads above data into counter 12
40 EQ 8,200                  ; Jumps to line 200 if above data is equal to 8.
50 ED                        ; Ends program.
200 MO 99                    ; Moves to position 99.
```

CF (Change figure)

【 Function 】
Changes the attitude data of the robot at the specified position.

【 Input Format 】
(1) 6-axis type

```
CF  <position number> [, [<R/L>] [, [<A/B>] [, [<N/F>]]]]
```

(2) 5-axis type

```
CF  <position number> [, [<R/L>] [, [<A/B>]]]
```

(3) 4-axis type

```
CF  <position number> [, [<R/L>]]
```

【 Term 】

<Position number>	Specify position number changing attitude data in integer value. 1 ≦ position number ≦ 999
<R/L>	Appoint structure flag of the robot to Right or Left. R : Right (Default) L : Left
<A/B>	Appoint structure flag of the robot to Above or Below. (5-axis and 6-axis type only) A : Above (Default) B : Below
<N/F>	Appoint structure flag of the robot to Non flip or Flip. (6-axis type only) N : Nonflip (Default) F : Flip

【 Explanation 】
(1) Changes the attitude data of the robot at the specified position.
(2) Does not effect the coordinate data of the specified position and the open/close state of the hand.
(3) Even if the position is reachable before executing this command, there is case becoming not reachable in the waist and shoulder joint after execution.
(4) You can confirm the attitude of the specified position by PR command.

【 Sample program 】(Movemaster command)
(1)6-axis type

```
10 PD 1,530,0,470,10,135,-10,R,A,N  ; Defines the position 1. (Right、Above、Non Flip)
20 MO 1                             ; Moves to position 1.
30 CF 1,R,A,F                       ; Changes the attitude of position 1. (Right、Above、Flip)
40 MO 1                             ; Defines the position 1.
```

(2)5-axis type

```
10 PD 1,-280,-50,970,-10,-20,L,A    ; Defines the position 1. (Left、Above)
20 MO 1                             ; Moves to position 1.
30 CF 1,R,A                         ; Changes the attitude of position 1. (Right、Above)
40 MO 1                             ; Defines the position 1.
```

(3)4-axis type

```
10 PD 1,200,0,100,,,0,R             ; Defines the position 1. (Right)
20 MO 1                             ; Moves to position 1.
30 CF 1,L                           ; Changes the attitude of position 1. (Left)
40 MO 1                             ; Defines the position 1.
```

CL (Counter Load)

【 Function 】
The internal register value is set in the counter with the specified number.
The character string register details are set in the character string with the specified value.

【 Input Format 】

```
CL   <counter number/character string number>
```

【 Term 】

<Counter number>	Specify counter No. in numeric value or counter No. with @. 1 ≦ counter No. ≦ 99 @1 ≦ counter No. ≦ @99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99

【 Explanation 】
<When counter number is specified>
(1) Sets the data fetched from the input port to the specified counter. Hence, the CL commands must be executed after the input command. (See ID)
(2) Since the data from the input port is treated as signed, the data set to the counter is signed. (Take value between −32768 and 32767.)
(3) Used to specify the number of job sequences and the counter value at palletizing from an external device, such as a programmable controller. At this case, any of the logical operation commands (see AN,OR,XO) may be used as necessary.
(4) Execution of the CL command after the CP command allows the counter data to be transferred.
(5) The counter value can be changed, compared or read by the relevant commands.
(See IC, DC, SC, CP, CR commands.)

<When character string number is specified>
(1) The data lead in from the character string register is set in the specified character string. Thus, this command must be executed after the character string register setting command (refer to CP, INP commands) is executed.
(2) This is used to set and confirm the work details or work results from an external device such as a sequencer. Use the compare command are required at this time. (Refer to EQ, LG, NE, SM commands.)
(3) The character string register value is set in the specified character string by this command, so if this command is executed after the CP command, the data can be copied between character strings.
(4) Operation, comparison and reading of the character string are possible by using the related commands.
(See CP, CR, EQ, INP, LG, NE, SC, SM commands.)

【 Sample program 】(Movemaster command)

```
10 ID                  ; Fetches data from external input port.
20 CL 25               ; Sets above data to counter 25.
30 CP 11               ; Sets data of counter 11 to internal register.
40 CL 21               ; Sets data of internal register to counter 21.
50 SC $5,″ABC″         ; Set character string ″ABC″ in character string number 5
60 CP $5               ; Set details of character string number 5 in the character string register
70 CL $10              ; Set the details of the character string register in character string number 10
```

CP (Compare Counter)

【 Function 】

The value of the specified counter is set in the internal register. The details of the specified character string are set in the character string register.

【 Input Format 】

```
CP   <counter number/character string number>
```

【 Term 】

<Counter number>	Specify counter No. in numeric value or counter No. with @. 1 ≦ counter No. ≦ 99 @1 ≦ counter No. ≦ @99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99

【 Explanation 】

<When counter number is specified>

(1) To be executed before a conditional jump command (see EQ,NE,LG,SM commands) is executed if the value in the specified counter is used for the jump. Conditional branching command carries out jump on the basis of value of internal register set by the CP command.

(2) Even if the value of the specified counter has changed after execution of the CP command, the value of internal register remains intact. Accordingly when you carry out conditional branching by value of counter, need to carry out this command after the counter value has changed.

(3) The input instruction (see ID) uses the same internal register, meaning that the old contents of the internal register are lost when any input command is executed.

(4) The contents of the counter can be changed or read by the relevant commands.
(See SC, IC, INP, DC, CR, CL, AN, OR, XO, EQ, LG, NE, SM commands.)

<When character string number is specified>

(1) When jumping the conditions due to the details of the specified character string, this command must be executed before the conditions jump command (refer to EQ, NE, LG, SM commands). The conditions jump command will execute comparison jumping based on the value of the character string register set by the CP command.

(2) Even if the details of the specified character string change after this command is executed, the character string register value will not be affected. Thus, when executing condition jump according to the details of the character string, this command must be executed after the details of the character string change.

(3) Operation, comparison and reading of the character string are possible by using the related commands.
(Refer to CP, CR, EQ, INP, LG, NE, SC, SM commands.)

【 Sample program 】 (Movemaster command)

```
100 IC 21                 ; Add 1 to the contents of counter 21
110 CP 21                 ; Sets the value of counter 21 to the internal register.
120 EQ 255,500            ; If the contents of the internal register equal 255, the program jumps to line number 500.
130 GT 100                ; The program jumps to line number 100.
500 SC 21,0               ; Sets value 0 to counter 21.
600 SC $5,″OK″            ; Set character string ″OK″ in character string number 5
610 CP $5                 ; Set details of character string number 5 in the character string register
620 EQ $10,800            ; If the contents of the character string register equal  character string number 10, the
                            program jumps to line number 800.
    :
800 GT 100                ; The program jumps to line number 100.
```

CR (Counter Read)

【 Function 】
The details of the specified counter or character string are read out. (Using RS-232-C)

【 Input Format 】

```
CR  <counter number/character string number>
```

【 Term 】

<Counter number>	Specify counter No. in numeric value or counter No. with @. 1 ≦ counter No. ≦ 99 @1 ≦ counter No. ≦ @99
<Character string number>	Specify character string number in numerical value which "$" is added to the head. $1 ≦ character string number ≦ $99

【 Explanation 】
(1) Outputs the contents of the specified counter from RS-232C port.
(2) The output format is in ASCII coded decimal.
(3) Because the terminator of the output data is carriage return (Hex. 0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. "LINE INPUT #" statement is equivalent to this in BASIC.
(4) If an undefined counter is read, the initial value of 0 is returned. (+0 is returned in the case of "0")
(5) If a non-set character string is read, the final code's carriage return (CR: hexadecimal 0D) will be returned.
(6) The contents of the counter are battery backed after the power is switched off.

【 Sample program 】 (N88BASIC)
<When counter number is specified>

```
10 OPEN" COM1;E83" AS #1               ; Opens the RS-232C communication file by the BASIC.
20 INPUT "Counter number";N            ; Enter the counter number.
30 INPUT "Counter data "; D            ; Enter the counter data.
40 PRINT #1, "SC "+STR$(N)+","+STR$(D)  ; Specified data is input into the counter.
50 PRINT #1,"CR"+STR$(N                ; Transfers the data to the personal computer.
60 LINE INPUT #1,A$                    ; Saves the received data to A$.
70 PRINT  A$                           ; Displays the data to the screen.
80 ED                                  ; Program ends.

RUN                                    ; Run the BASIC program.
Counter number  1
Counter data 100
+100
```

<When character string number is specified>

```
10 OPEN" COM1;E83" AS #1               ; Opens the RS-232C communication file by the BASIC.
20 INPUT "Character string number";N   ; Enter the character string number.
30 INPUT "Character string data ";J$   ; Enter the character string data.
40 PRINT #1, "SC $"+STR$(N)+","+CHR$(&H22)+J$+CHR$(&H22)
                                       ; Specified data is input into the counter.
50 PRINT #1,"CR $"+STR$(N)             ; Transfers the data to the personal computer.
60 LINE INPUT #1,A$                    ; Saves the received data to A$.
70 PRINT A$                            ; Displays the data to the screen.
80 ED                                  ; Program ends.

RUN                                    ; Run the BASIC program.
Character string number  $1
Character string data ABC
ABC
```

DA (Disable Act)

【 Function 】
Disables an interrupt of the specified bit through the external input port.

【 Input Format 】
```
DA   <input bit number>
```

【 Term 】

<Input bit number>	Specify the bit number to be disabled. 0 ≦ input bit number ≦ 32767 0 ～ 8999 : Input signal interrupt (0–299: general input, 900–903: hand input) 9003 ～ 32767 : Input signal interrupt 9002 : Communication interrupt

【 Explanation 】
(1) Clears the interruptible state of the bit defined by the interrupt enable commands (see the EA commands).
(2) After the DA command has been executed, no interrupt is executed by the specified bit during program execution. Note that execution of the DA command dose not affect the interruptible states of the other bits.
(3) To inhibit repeated interrupts by a level signal, the DA command must be executed at the beginning line to which the program jumps after the interrupt has taken place.

【 Sample program 】
See EA commands.

DC (Decreement counter)

【 Function 】
Subtracts 1 from the value in the specified counter.

【 Input Format 】
```
DC   <counter number>
```

【 Term 】

<Counter number>	Specify counter No. in numeric value or counter No. with @. 1 ≦ counter No. ≦ 99 @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) Error occurs if the counter value becomes smaller than −32768.
(2) Used to count the number of workpieces and job sequences and to set the number of grid points in pallet.
(3) The contents of the counter can be changed, compared or read by the relevant command.
(See SC, IC, CP, CR, CL, AN, OR, XO commands.)

【 Sample program 】 (Movemaster command)
```
10 SC 21,15          ;Sets value 15 to the counter 21.
20 DC 21             ;Subtracts 1 from the value in the counter 21.
```

DIV (Div)

【 Function 】
Divides the value of the internal register by the operation data, and stores it in the internal register.

【 Input Format 】

```
DIV   <operation data>
```

【 Term 】

<Operation data>	Describes the data to be operated as a numeric value or counter No. with @. −32768 ≦ Numeric value (decimal) ≦ 32767 &8000 ≦ Numeric value (hexadecimal) ≦ &7FFF @1 ≦ Counter ≦ @99

【 Explanation 】
(1) Designate the operation data setting as a numeric value or counter No.
When designating with a numeric value, use a decimal or hexadecimal value. When using a hexadecimal, add a ″&″ to the head of the operation data.
When setting with a counter No. add a ″@″ to the head of the counter No.
The contents of the set counter No. will be used as the operation data.
(2) The operation results are stored in the internal register, so operation, comparison and reading, etc., of the operation results can be carried out will the related commands.
(Refer to ADD, SUB, MUL, EQ, NE, LG, SM, CL, DR, OR commands)
(3)When result is a real number, the decimal part is rounded down.

【 Sample program 】 (Movemaster command)

```
10  CP 1                  ; Stores counter No.1 value in internal register.
20  DIV @2                ; Divides internal register value by counter No.2 value
30  CL 3                  ; Sets internal register value in counter No.3
                           (Counter No.3 = counter No.1 / counter No.2)
40  CP 1                  ; Stores counter No.1 value in internal register.
50  DIV 5                 ; Divides internal register value by 5
60  CL 4                  ; Sets internal register value in counter No.4
                           (Counter No.4 = counter No.1/5)
```

DJ (Draw Joint)

【 Function 】
Rotates the specified joint by the specified angle from the current position. (Joint interpolation)

【 Input Format 】

```
DJ  <joint number>, <turning angle>
```

【 Term 】

<Joint number>	Specify joint number that you want to move. 1) 6-axis type 1:J1 axis 2:J2 axis 3:J3 axis 4:J4 axis 5:J5 axis 6:J6 axis 7:J7 axis(additional axis 1) 8:J8 axis(additional axis 2) 2) 5-axis type 1:J1 axis 2:J2 axis 3:J3 axis 5:J5 axis 6:J6 axis 7:J7 axis(additional axis 1) 8:J8 axis(additional axis 2) 3) 4-axis type 1:J1 axis 2:J2 axis 3:J3 axis 4:J4 axis 7:J7 axis(additional axis 1) 8:J8 axis(additional axis 2)
<Rotating angle>	Specify the amount of joint that you want to move.

【 Explanation 】
(1)The least increment of each axis is 0.001 (mm or degree) for the RP-1AH/3AH/5AH Series and the RH-5AH/10AH/15AH Series, the other series is 0.01 (mm or degree). (e.g. specify 20.001 for 20.001 mm for RH-5AH) (e.g. specify 20.01 for 20.01 mm for RV-1A)
(2) The open/close state of the hand does not change before and after the movement. Error occurs before the joint motion if any turning angle entry exceeds the robot's operational space.

【 Sample program 】 (Movemaster command)

```
10 MO 1                 ; Moves to position 1.
20 DJ 1,10              ; Turns the J1 axis 10 degrees in the positive direction.
```

DL* (Delete Line)

【 Function 】
Deletes commands of the specified line or step in the program.

【 Input Format 】
```
DL  <line number (a)> [, [<line number (b)>  [, [<step(a)> [,[<step(b)>]]]]]]
```

【 Term 】

<Line number(a)>	Specify the top line number that you want to delete in the program.
<Line number(b)>	Specify the last line number that you want to delete in the program.
<step(a)>	Specify the top step number that you want to delete in the program.
<step(b)>	Specify the last step number that you want to delete in the program.

1 ≦ line number (a), (b) step (a), (b) ≦ 32767
Line number (a) ≦ line number (b), step (a) ≦ step (b)

【 Explanation 】
(1) Deletes commands from step (a) to step (b), then deletes commands from line (a) to line (b). (Step (b) and line (b) are included too.)
(2) When you omitted line (b), only line (a) is deleted.
(3) When you omitted step (b), only step (a) is deleted.
(4) An error will occur if the size relation of (a) and (b) is incorrect.

【 Sample program 】 (Movemaster command)
```
100 MO 10              ; Moves to position 10.
110 MO 12              ; Moves to position 12.
120 MO 15              ; Moves to position 15.
130 MO 17              ; Moves to position 17.
140 MO 20              ; Moves to position 20.
DL  110                ; Deletes line number 110.
DL  120,140            ; Deletes line number from 120 to 140.
```

DP (Decrement Position)

【 Function 】
Moves the robot to a predefined position with a position number smaller than the current one. (Joint interpolation)

【 Input Format 】
```
DP
```

【 Explanation 】
(1) Moves the robot to a predefined position with a position number smaller than, and closest to, the current one. (See command IP)
(2) Error takes place if there is no predefined position which is smaller in position number than the current position.
(3) Even if an error occurs, current position number is still maintained.

【 Sample program 】 (Movemaster command)
```
100  MO 3              ; Moves to position 3.
110  MO 4              ; Moves to position 4.
120  MO 5              ; Moves to position 5.
130  DP                ; Moves to position 4.
```

DR (Data Read)

【 Function 】
Reads the values of the internal register, hand check state, and general output state. (Using RS-232C)

【 Input Format 】

```
DR   [<output bit number>]
```

【 Term 】

<Output bit number>	0 ≦ output bit number ≦ 32767 (0 for default)

【 Explanation 】
(1) Outputs the values of the internal register, hand check state, and general output state through the RS-232C. Allows the external input data and hand open/close state to be read when executed after the input command (ID).
(2) Hand check state read by the DR command corresponds to the current state.
(3) When you specify the output bit number, you can read the general output state of 16 bits-width beginning from the specified bit number.
(4) The output format is in ASCII coded hexadecimal, which is headed by ″&H″ and delimited by ″,″ (Hex. 2C).
The output format: Value of hand check state and the internal register, general output
The first column next to ″& H″ responds to the hand check input and the following 4 columns responds to the value of internal register, and the remaining 4 columns headed by ″&H″ responds to the current general output.
(5) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.

【 Sample program 】(N88BASIC)

```
10 OPEN ″COM1;E83″AS#1 ; Opens the RS-232C communication file by the personal computer.
20 PRINT #1,″ID″       ; Sets the value of input port to the internal register.
30 PRINT #1,″DR″       ; Reads the value of internal register and hand check input.
40 LINE INPUT #1,A$    ; Saves the data to A$.
50 PRINT ″Data is″;A$  ; Displays the value of A$.
60 ED                  ; Ends

RUN                    ; Run the BASIC program.
Data is  &H10FB2, &H30BA ; Displays data.
                       ; Value of hand check input      : 1 (Hexadecimal)
                       ; Value of the internal register : 0FB2 (Hexadecimal)
                       ; Value of general output        : 30BA (Hexadecimal)
```

DS (Draw Straight)

【 Function 】
Moves the end of the hand to a position away from the current position by the distance specified in X,Y and Z directions. (Linear interpolation)

【 Input Format 】

```
DS  [<travel distance in X>], [<travel distance in Y>], [<travel distance in Z>]
```

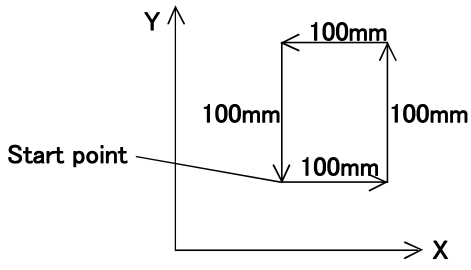
【 Term 】

<Travel distance in X>	Specify the amount that you want to move in X direction from the current position.
<Travel distance in Y>	Specify the amount that you want to move in Y direction from the current position.
<Travel distance in Z>	Specify the amount that you want to move in Z direction from the current position. (Zero travelling for default of each axis.)

【 Explanation 】
(1) The least input increment for the distance is 0.01mm (e.g., specify 20.01 for 20.01mm)
(2) The attitude of the hand, including the open/close status of the gripper, remains the same before and after the movement.
(3) Error occurs before or during movement if the destination or travel path exceeds the operating space of the robot. Especially the roll joint tends to exceed the operating space since it is controlled to remain the same orientation during the movement.
(4) Moving speed during linear interpolation is decided by the SP or SD commands. (Hand tip at constant speed.)
(5) The hand tip is decided by the tool length at that time.

【 Sample program 】 (Movemaster command)

```
10 DS 100,0,0            ; Moves to X axis direction by 100 mm.
20 DS 0,100,0            ; Moves to Y axis direction by 100 mm.
30 DS −100,0,0           ; Moves to X axis direction by −100 mm.
40 DS 0,−100,0           ; Moves to Y axis direction by −100 mm.
```



* In the above example, the hand tip moves through the four corners of a square by linear interpolation and returns the start point finally.

DW (Draw)

【 Function 】
Moves the end of the hand to a position away from the current position by the distance specified in X, Y and Z directions. (Joint interpolation)

【 Input Format 】

```
DW   [<travel distance in X>], [<travel distance in Y>], [<travel distance in Z>]
```

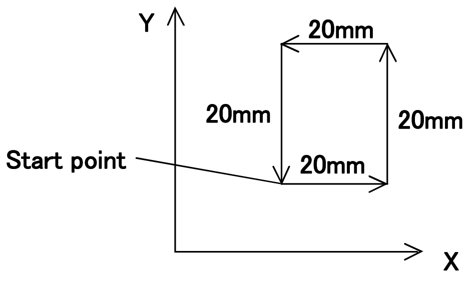
【 Term 】

<Travel distance in X>	Specify the amount that you want to move in X direction from the current position.
<Travel distance in Y>	Specify the amount that you want to move in Y direction from the current position.
<Travel distance in Z>	Specify the amount that you want to move in Z direction from the current position. (Zero travelling for default of each axis.)

【 Explanation 】
(1) The least input increment for the distance is 0.01mm (e.g., specify 20.01 for 20.01mm)
(2) The attitude of the hand, including the open/close status of the gripper, remains the same before and after the movement. Error occurs before the movement if the destination exceeds the operating space of the robot.
(3) The moving path draws a curve in the case of large travelling path because of the linear interpolation.
(4) The hand tip is decided by the tool length at that time. (See TL command.)

【 Sample program 】 (Movemaster command)

```
10 DW 200,0,0           ; Moves to X axis direction by 200 mm
20 DW 0,200,0           ; Moves to Y axis direction by 200 mm
30 DW -200,0,0          ; Moves to X axis direction by -200 mm
40 DW 0,-200,0          ; Moves to Y axis direction by -200 mm
```



* In the above example, the hand tip moves through the four corners of a square by joint interpolation and returns the start point finally.

EA (Enable Act)

【 Function 】
Enables the interrupt motion by the specified bit of the external input signal.

【 Input Format 】

```
EA    [<+/->] <input bit number>, <line number> [, [<branching approach>]]
```

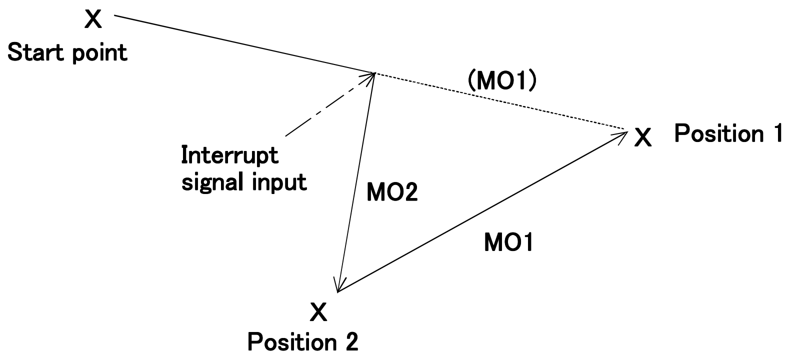
【 Term 】

<+/->	+ : If the <input bit number> of the external input port turns ON, the program jumps to the <line number>. (Default) – : If the <input bit number> of the external input port turns OFF, the program jumps to the <line number>.
<Input bit number>	Specify the bit number of external input signal that you want to assign for interrupt signal 0 ≦ input bit number ≦ 32767 0 ~ 8999 : Input signal interrupt (0–299: general input, 900–903: hand input) 9003 ~ 32767 : Input signal interrupt ・Enables the interrupt motion by the external input signal. ・The EA command allows eight input bits to be registered as interrupt signals concurrently. ・When the interrupt signal is input, the robot is decelerated to a stop and the program jumps to the <line number>. ・Once the interrupt has taken place, the robot remains stopping until the DA command is executed at the beginning line to which the program jumps or the interrupt signal turns off. ・The signal input except for the specified bit does not occur any interrupt. 9002 : Communication interrupt ・When data received through the RS-232C port, executes the program from the specified line number.
<Line number>	Specify the line number to which the program jumps by the interrupt signal. 1 ≦ line number ≦ 32767
<Branching approach>	Specify a jump or subroutine calling. 0:Jump (See "GT" command) (Default) 1:Subroutine calling (See "GS" command)

【 Explanation 】
(1) Causes an interrupt to be executed by an external signal while the program is running. When the interrupt signal is input after execution of the EA command, the robot is decelerated to a stop and the program jumps to the specified line number.
(2) The EA command allows 8 signals, giving the priority to the larger number, to be registered as the interrupt signal at the same time.
(3) Once this command has been executed, the effective condition is maintained till the interrupt disable command (DA), the end command (ED), or the reset command (RS) is carried out.
(4) If the specified line number has no command, error will occur in the program execution.

【 Sample program 】(Movemaster command)

```
100 EA +5,500              ; Sets that the program jumps to the line 500 in case of bit 5
                             turning ON.
110 MO 1                   ; Moves to position 1.
120 ED                     ; Ends program.
500 DA 5                   ; Interrupt disable
510 MO 2                   ; Moves to position 2.
520 GT 110                 ; Jumps to line number 110.
```



＊In the above example, line 100 causes an interrupt to be enabled and line 110 moves the robot to position 1. When the specified signal is input during this motion, the robot is decelerated to a stop and then the program jumps to line 500, where the interrupt is disabled. Line 510 moves the robot to position 2 and line 520 causes the program to jump to line 110. The robot moves to position 1 again.

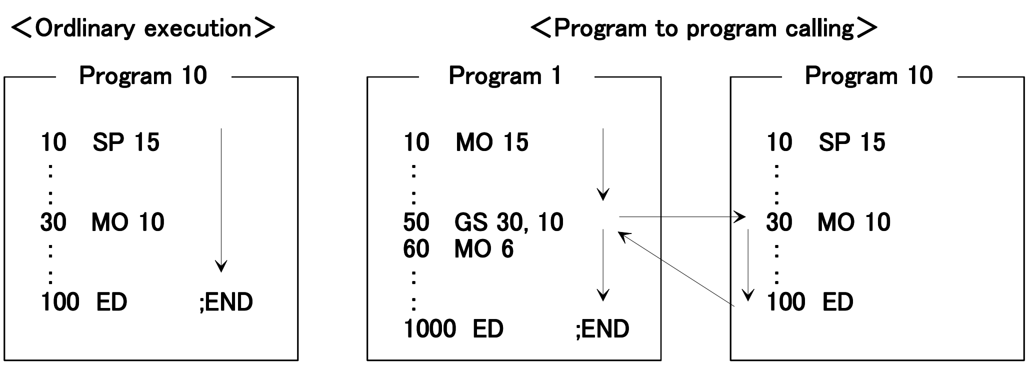
ED (End)

【 Function 】
Ends the program.

【 Input Format 】

ED

【 Explanation 】
(1) Marks the end of a program. The program ends when the ED command is executed. In the case of a program to program subroutine, however, returns to the calling program .
(2) Required at the end of a program unless the program commands are directly executed from the personal computer.



```
Program 10
10   SP 15
:
:
30   MO 10
:
:
100  ED        ;END
```

```
Program 1
10   MO 15
:
:
50   GS 30, 10
60   MO 6
:
:
1000  ED       ;END
```

```
Program 10
10   SP 15
:
:
30   MO 10
:
:
100  ED
```

* Program can call other program from inside the program using GS command.
The arrow shows the sequential order of program execution .
In the above example of <Program to program calling>, the program is executed to line 50 then jumps to the program 10. After executing to line 30 of program 10, the program returns line 60 of program 1. Program ends when the ED command of program 1 is executed.

【 Sample program 】(Movemaster command)

```
100 SP 3                ; Set speed 3
110 MO 3                ; Moves to position 3.
120 MO 5                ; Moves to position 5.
130 ED                  ; Ends the program.
```

EQ (Equal)

【 Function 】
This compares the value of the internal register with a specified value. If they are the same, the program will jump to the specified line number. The character string register and the details of the specified character string are compared, and if the values are the same, the program will jump to the specified line number.

【 Input Format 】

```
EQ   <compared value>, <branching line number>
```

【 Term 】

<Compared value>	Specify the value that the internal register compares contents with or counter No. with @. −32768 ≦ compared value (decimal) ≦ 32767 & 8000 ≦ compared value (hexadecimal) ≦ &7FFF @1 ≦ counter number ≦ @99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99
<Branching line number>	Specify the line number to which the program jumps when the comparison result is equal. 1 ≦ branching line number ≦ 32767

【 Explanation 】
<When counter number is specified>
(1) Causes a jump to occur conditionally in accordance with the external input data or the internal counter value.
(2) If the internal register value equals to the compared value (i.e., when the condition is met), the program jumps to the specified line. Otherwise (i.e., when the condition is not met), the program continues in sequence.
(3) A value can be loaded into the internal register by executing the input command (See ID) for the external input data or by executing the counter set command (See CP) for the counter data. Accordingly when you carry out conditional branching, need to execute either of the above commands beforehand.
(4) The compared value may be defined either in decimal or hexadecimal. A hexadecimal value must be headed by ″&″.
(5) Error occurs at a jump if the specified line number does not exist.

<When character string number is specified>
(1) The conditions will jump depending on the details of the character string register.
(2) If the details of the character string register are equal to the details of the character string (when the conditions are established), the program will jump to the specified line number. If the details are not equal (when conditions are not established), the next line will be executed.
(3) By executing an INP command, the data input from an external device will be set in the character string register. The details of the character string number will be set by executing a CP command. Thus, when executing condition jumping, one of these commands must be executed first.
(4) If the specified line number is not registered, an error will occur during jumping.

【 Sample program 】(Movemaster command)

```
100 ID                  ; Fetches data from external input port.
110 EQ 100,130          ; Jumps to line 130 if the data equals 100.
120 ED                  ; Ends the program if above condition is not met.
130 SC $5,″OK″          ; Set character string ″OK″ in character string number 5
140 CP $5               ; Set details of character string number 5 in the character string register
150 EQ  $10,200         ; Jumps to line 200 if the data equals character string number 10.
                          :
200 MO 7                ; Moves to  position 7.
```

ER* (Error Read)

【 Function 】
Reads the current error status and error history contents. (Using RS-232C)

【 Input Format 】

```
ER  [<error history number>]
```

【 Term 】

<Error history number>	Specify the number of error history. 1 ≦ error history number ≦ 128 (If omitted, shows the current error)

【 Explanation 】
(1) Outputs the error condition of the robot from the RS-232C port.
(2) Specify the error history number on the basis of the current error (0). When you specified 1, outputs the error information former than the current error by 1.
(3) The ER command outputs the former error history in the ASCII format as below.
The output format: Error history number, error number, year, month, day, hour, minute,second.
(4) If the error history number is omitted, the ER command outputs the current error with following ASCII code.
0: No error
1: High level error
2: Low level error
3: Warning
(5) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.
(6) Useful to transfer a sequence of data from the personal computer to the robot while checking for an error.

【 Sample program 1 】(N88BASIC)

```
10   OPEN ” COM1 : E83” AS #1 ; Opens the RS-232C communication file from the personal computer by BASIC.
20   PRINT#1,” MO1”           ; Moves to position 1.
30   GOSUB 100                ; Calls the subroutine line 100 in BASIC.
40   PRINT #1,” MO2”          ; Moves to  position 2.
50   GOSUB 100                ; Calls the subroutine line 100 in BASIC.
60   ED                       ; Ends
100  PRINT #1,” ER”           ; Reads the current error.
110  LINE INPUT #1,A$         ; Saves the received data to A$.
120  IF A$=” 0” THEN RETURN   ; If there is no error, returns subroutine.
130  PRINT ” ″Error level is ″; A$ ; Displays the data on the personal computer screen.
140  ED                       ; Ends

RUN                           ; Run the BASIC program.
Error level is  2             ; Displays error level.
                               (2: Operation error)
```

【 Sample program 2 】(N88BASIC)

```
10  OPEN ” COM1:E83” AS#1      ; Opens the RS-232C communication file from the personal computer by BASIC.
20  INPUT ″History number is ″; N ; Enters the error history number by the personal computer.
30  PRINT #1,″ER″+STR$(N)      ; Reads the error information.
40  LINE INPUT #1,A$           ; Saves the received data to A$.
50  PRINT ″Error information ″;A$ ; Displays the data on the personal computer screen.
60  ED                         ; Ends

RUN                            ; Run the BASIC program.
History number is  1
Error information  1,3800,93,10,21,11,34,20
```

GC (Grip Close)

【 Function 】
Close the grip of hand.

【 Input Format 】

```
G C   [<hand number>]
```

【 Term 】

<Hand number>	Specify the hand number opening grip.	
	0: Hand 1 (Default)	4: Hand 5
	1: Hand 2	5: Hand 6
	2: Hand 3	6: Hand 7
	3: Hand 4	7: Hand 8

【 Explanation 】
(1) This command drives the solenoid valve and closes the hand (or picks up the workpiece).
The general-purpose output signal can also be controlled with the OB command to close the hand. (Refer to the OB command.)
(2) When closing the hand and gripping the workpiece, it may take some time for the movement to settle and the hand to move. Thus, in some cases, a TI command must be provided before and after this command to establish a delay timer.

【 Relating Parameters 】
The output signal No. for solenoid driving and the status of the hand output signal when the power is turned ON can be set with parameters for each hand No.
As a default, hands 1 to 4 are set in the output signal Nos. 900 to 907. When using hands 5 to 8, the parameters must be set.
Refer to the separate "Instruction Manual/Detailed Explanation of Functions and Operations" for details on changing the parameters.
The outlines of the related parameters are given in Table 2-4.

Table 2-4 : Hand related parameters

Parameter name	Setting details
HANDTYPE	Identification of single/double solenoid and setting of output signal No.
HANDINIT	Setting of pneumatic hand output at power ON

【 Sample program 】(Movemaster command)

```
10 MO 10,O              ; Moves to  position 10 with hand opened.
20 TI 5                 ; Sets 0.5 second timer.
30 GC                   ; Closes hand to grasp workpiece.
40 TI 5                 ; Sets 0.5 second timer.
50 MO 15,C              ; Moves to position 15 with hand closed.
```

GF (Grip Flag)

【 Function 】
Defines the open/close state of the grip of the hand (used with the PD command).

【 Input Format 】
```
G F   <switch>
```

【 Term 】

<switch>	Specify open or close state of the hand in 0 or 1. 0: Open 1: Close

【 Explanation 】
(1) Defines the open or close state of the hand grip used with the PD command which defines the coordinates of the specified position. The PD command takes precedence if the hand open/close state has been specified by the PD command.

【 Sample program 】(Movemaster command)
```
10 GF 0                    ; Sets the grip flag to open.
20 PD 10,50,320,70,50,40,30,R  ; Defines position 10 with hand opened. (The example of the coordinate value is 6-axis
                               type.)
```

GO (Grip Open)

【 Function 】
Opens the grip of the hand.

【 Input Format 】

```
ＧＯ　[<hand number>]
```

【 Term 】

<Hand number>	Specify the hand number opening grip.	
	0: Hand 1 (Default)	4: Hand 5
	1: Hand 2	5: Hand 6
	2: Hand 3	6: Hand 7
	3: Hand 4	7: Hand 8

【 Explanation 】
(1) This command drives the solenoid valve and opens the hand (or releases the workpiece).
The general-purpose output signal can also be controlled with the OB command to open the hand. (Refer to the OB command.)
(2) When opening the hand and releasing the workpiece, it may take some time for the movement to settle and the hand to move. Thus, in some cases, a TI command must be provided before and after this command to establish a delay timer.

【 Relating Parameters 】
The output signal No. for solenoid driving and the status of the hand output signal when the power is turned ON can be set with parameters for each hand No.
As a default, hands 1 to 4 are set in the output signal Nos. 900 to 907. When using hands 5 to 8, the parameters must be set.
Refer to the separate "Instruction Manual/Detailed Explanation of Functions and Operations" for details on changing the parameters.
The outlines of the related parameters are given in Table 2-5.

Table 2-5 : Hand related parameters

Parameter name	Setting details
HANDTYPE	Identification of single/double solenoid and setting of output signal No.
HANDINIT	Setting of pneumatic hand output at power ON

【 Sample program 】(Movemaster command)

```
10 MO 10,C          ; Moves to position 10 with hand closed.
20 TI  5            ; Sets 0.5 second timer.
30 GO               ; Opens hand to release workpiece.
40 TI 5             ; Sets 0.5 second timer.
50 MO 15,O          ; Moves to position 15 with hand opened.
```

GP (Grip Pressure)

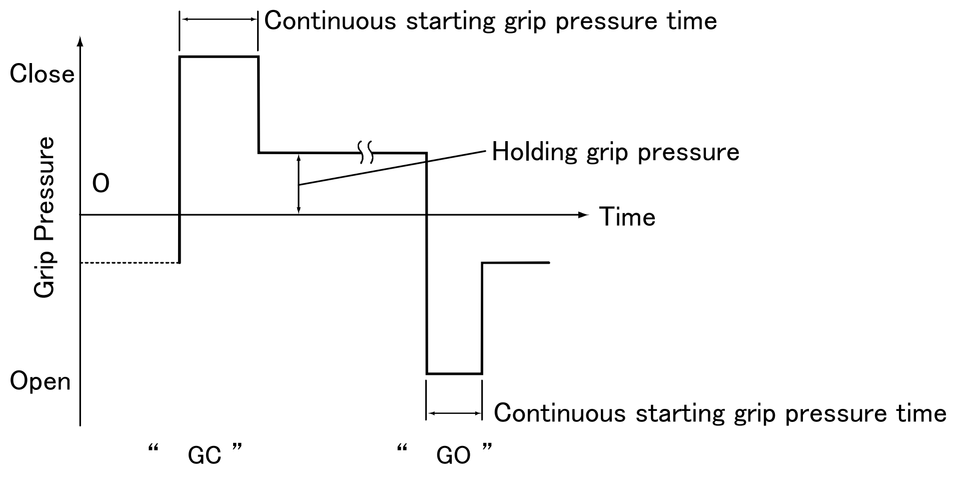
【 Function 】
Sets the continuous starting grip pressure time for opening and closing the hand.

【 Input Format 】

```
GP  <Starting grip pressure> <Holding grip pressure>, <Continuous starting grip pressure time>
```

【 Term 】

<Starting grip pressure>	This is invalid when using the pneumatic hand. 0 ≦ Starting grip pressure ≦ 63(3.5kg)
<Holding grip pressure>	This is invalid when using the pneumatic hand. 0 ≦ Holding grip pressure ≦ 63(3.5kg)
<Continuous starting grip pressure time>	Describe the time to continue to the starting grip pressure as an integer. 0 ≦ Continuous starting grip pressure time ≦ 99 (9.9s) The unit is 0.1 seconds.


Continuous starting grip pressure time
Close
Grip Pressure
0
Holding grip pressure
Time
Open
Continuous starting grip pressure time
" GC "
" GO "

【 Explanation 】
(1) The continuous starting grip pressure time is the set value multiplied by 0.1s (max. 9.9s). Set the optimum value according to the target to be gripped. The set value is valid until it is reset, and is set commonly for all programs.
(2) The default state when the power is turned ON is ″GP 63, 63, 3″, and the continuous starting grip pressure time is 0.3s.
(3) When using the pneumatic hand, the starting grip pressure and holding grip pressure values are invalid.
(4) The robot movement will stop during the continuous starting grip pressure time.

【 Sample program 】 (Movemaster command)

```
10 GP 63, 63, 10          ; Set the continuous starting grip pressure time to 1s.
20 GC                     ; Close the hand with the above settings.
```

GS (Go Sub)

【 Function 】
Carries out subroutine beginning with the specified line number.

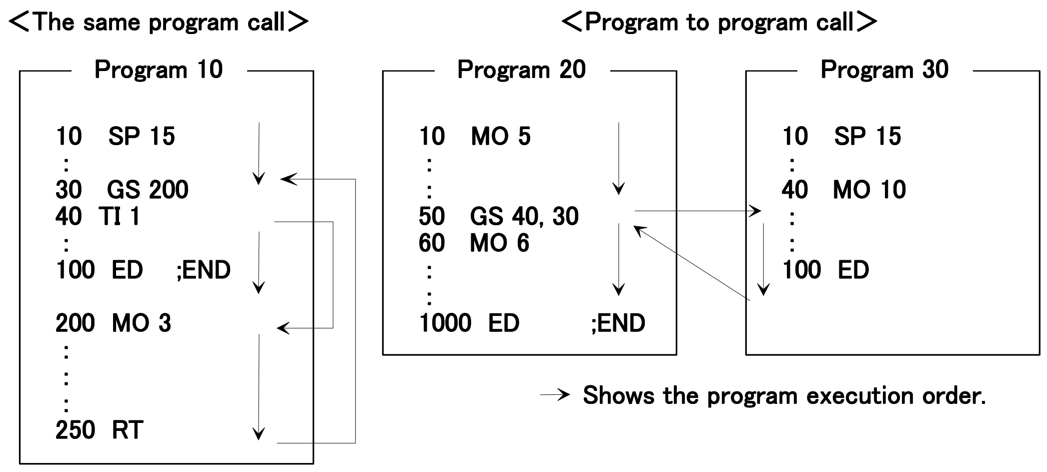
【 Input Format 】

```
GS    [<line number>] [, [<program name>]]
```

【 Term 】

<Line number>	Specify line number of subroutine in integer value.	
	1 ≦ line number ≦ 32767	
<Program name>	Specify program name of subroutine in integer value or characters. (Less than 8 characters)	
	1 ≦ program name ≦ 8 (characters)	
	Possible letter used	: Digit (0–9) Character (A – Z)
	Impossible letter used	: (＊ + , . / : ; [¥] ' ″ ! @ #)
	Special specification	: When you specified only numeric value, the program name is handled as number. Need to enclose program name with ″ ″ in the case of character used.

【 Explanation 】
(1) Allows the program to jump to the specified line of the specified program and execute subroutine. The program returns to the main program after executing the subroutine. When you specified program number, returns to the main program by ED command and when you specified only line number, returns by RT command.
(2) Use the RT command to terminate the subroutine existing in the same program. Use the ED command to terminate the subroutine existing in other program.
(3) If the specified line or the specified program does not exist, error occurs at the time of GS execution.
(4) When you omitted line number, executes the specified program from the top line.
(5) When you omitted line number and program name, nothing occurs .
(6) To call subroutines in other subroutines is called ″nesting″. Up to 9 nesting levels are possible.



```
Program 10
10   SP 15
:
30   GS 200
40  TI 1
:
100  ED    ;END

200  MO 3
:
:
:
250  RT
```

```
Program 20
10   MO 5
:
:
50   GS 40, 30
60   MO 6
:
:
1000  ED        ;END
```

```
Program 30
10   SP 15
:
40   MO 10
:
:
100  ED
```

→ Shows the program execution order.

＊In the above example of <The same program call>, executes the program from line 10 to 30, then calls the subroutine of line 200. When the RT command is executed in the subroutine, the program returns to the main program and continues from line 40. The program ends when the ED command is executed.

＊Program can call other program from inside the program using GS command. In the above example of <Program to program call>, executes the program from line 20 to 50, then calls the program 30. Executes the program from line 40 to 100 and returns to the main program, i.e., line 60 of program 20. The program ends when the ED command is executed.

【 Sample program 】 (Movemaster command)
```
10 GS 100                ; Carry out subroutine beginning with line number 100.
  :                      ;
90 ED                    ; Ends program.
100 MO 11                ; Moves to position 11.  ┐
110 MO 12                ; Moves to position 12.  ├─Subroutine
120 MO 13                ; Moves to position 13.  │
130 RT                   ; Ends subroutin         ┘
```

GT (Go To)

【 Function 】
Jumps to the specified line number unconditionally.

【 Input Format 】

G T　<line number>

【 Term 】

<Line number>	Specify the line number to which the program jumps. 1 ≦ line number ≦ 32767

【 Explanation 】
(1) Causes the program to jump to the specified line number.
(2) If the specified line number does not exist, error occurs at the time of GT execution.

【 Sample program 】 (Movemaster command)
```
10 MO 1                  ; Moves to position 1.
20 GT 100                ; Jumps to line 100 unconditionally.
 :
100 MO 12                ; Moves to position 12.
110 MO 15                ; Moves to position 15.
 :
```

HE (Here)

【 Function 】
Defines the current coordinates as the specified position.

【 Input Format 】

```
H E   <Position number>
```

【 Term 】

<Position number>	Specify the position number to be registered. 0 ≦ position number ≦ 999 Registers the current position to the user-defined origin in case of zero.

【 Explanation 】
(1) The current position coordinates (XYZ system) are calculated based on the current tool length (refer to the TL command). In the default state, the coordinates are based on the hand installation surface.
(2) If a single number is assigned to two different positions, the one defined last takes precedence with the former cleared.
(3) The open/close position of the hand and the structure flag data are also stored as the position data.
(4) Error occurs if the HE command is executed before the origin setting.
(5) When you specified zero position number, current position data in joint coordinates are defined to user-defined origin parameter USERORG.
(6)Use TI command before HE command when you register coordinate value under the condition that a robot stops completely.

【 Sample program 】 (Movemaster command)

```
10 MO 10                ; Moves to position 10.
20 DW 10,0,0            ; Moves to +X direction by 10 mm
30 HE 11                ; Defines above location as position 11.
```

HLT (Halt)

【 Function 】
Interrupts the motion of the robot and the operation of the program

【 Input Format 】

```
HLT
```

【 Explanation 】
(1) Interrupts the operation of the program and decelerates the robot to a stop. (It becomes the same condition that the external stop signal is input or the STOP switch of the controller front panel is pushed.)
(2) To restart the program, push the START switch, input the starting signaling, or execute the RN command. Program restarts from the next line of HLT command.
(3) If the HLT command is directly executed from the personal computer during program running, the program is interrupted and the robot stops with deceleration.
(4) The robot does not stop by the HLT command, however, during the execution of the direct motion command.

【 Sample program 】 (Movemaster command)

```
10 MO 1                 ; Moves to position 1.
20 HLT                  ; Stops
30 MO 2                 ; Moves to position 2.
40 ED                   ; Ends program.
```

The program restarts with START switch from line 30.

HO (Home)

【 Function 】
Defines the current location and the attitude as origin point.

【 Input Format 】
```
HO   [<origin setting approach>]
```

【 Term 】

<Origin setting approach>	Specify the method to set origin in integer value. 0: Mechanical stopper origin 1: Jig origin 2: User-defined origin

【 Explanation 】
(1) Establishes the reference position for origin setting.
(2) If you have replaced the robot or changed the combination of robot and controller, you must carry out origin setting again using this command.
Teaching pendant can be used for origin setting. Refer to separate manual ″Detailed explanations of function and operations″ for operation method.
(3)Change the HOE parameter in the authorized state. Afterwards, this instruction is executed directly.
Complete the orogin setting, set HOE parameter in the prohibition state. The program cannot be execute like the authorized state.

【 Relating Parameters 】
Permits the origin setting from the command (HO).

Parameter name	HOE	: Origin setting permission parameter 0: Does not permit the use of HO command. (Default) 1: Permits the use of HO command.

【 Sample program 】(N88BASIC)
```
10 OPEN” COM1:E83” AS #1   ;Opens the  RS-232C communication file from the personal computer in BASIC.
20 PRINT #1,” HO”          ;Executes the ″HO″ command from the personal computer.
30 ED                      ;Ends

RUN                        ;Run the BASIC program.
```

IC (Increment Counter)

【 Function 】
Adds 1 to the value of the specified counter.

【 Input Format 】
```
IC   <Counter number>
```

【 Term 】

<Counter number>	Specify counter No. in numeric value or counter No. with @. 1 ≦ counter No. ≦ 99 @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) Error occurs if the counter value exceeds 32767.
(2) Used to count the number of workpieces and job sequence and to set the number of grid point in the pallet.
(3) The contents of the counter can be changed, compared, or read by the relevant command.
(See SC, DC, CP, CR, CL, AN, OR, XO commands.)

【 Sample program 】(Movemaster command)
```
10 SC 21,15          ;Sets value 15 to counter 21.
20 IC 21             ;Add 1 to the contents of counter 21.
```

ID (Input Direct)

【 Function 】
Fetches data unconditionally from the external input and hand check input.

【 Input Format 】

```
ID    [<input bit number>]
```

【 Term 】

<Input bit number>	Specify the bit number of input port in integer value. Fetches data of 16 bits width including the specified bit. 0 ≦ input bit number ≦ 32767 (0 for default)

【 Explanation 】
(1) Fetches signals from the external equipment, e.g., programmable controller, unconditionally. The data from the hand check input can be fetched by specifying the 900th number to the input bit number.
(2) The fetched data is loaded into the internal register and is subsequently used for comparison, bit test, etc. (See EQ, NE, LG, SM, TB commands.)
(3)This is a dedicated input/output parameter, and can be input with the bit assigned as a dedicated input. In the default state, the input signal Nos. 0 to 5 are assigned as dedicated inputs.

【 Sample program 】(Movemaster command)

```
100 ID                  ; Fetches the input data into the internal resister for comparison.
110 EQ 100,130          ; If the input data equals 100, then jumps to line number 130.
120 ED                  ; Else ends program.
130 MO 1                ; Moves to position 1.
140 ID 100              ; Fetches the input data into the internal resister for  comparison.(Input signals 100 to
                          115.)
150 TB +0,180           ; If the input bit 100 is ON, then jumps to line 180.
160 TB +5,200           ; If the input bit 105 is ON, then jumps to line 200.
170 ED                  ; Else ends program.
180 MO 2                ; Moves to position 2.
190 ED                  ; Ends program.
200 MO 3                ; Moves to position 3.
210 ED                  ; Ends program.
```

NP (Input)

【 Function 】
The specified counter value, the coordinate value of the position number or the data of the specified character string is received according to the PRN command. (Using RS-232-C)

【 Input Format 】

```
INP   <channel number>, <counter number/position number/character string number>
      [, [<contents selection>]]
```

【 Term 】

<Channel number>	Specify the channel number opened by the OPN command. 0 ≦ channel number ≦ 2
<Counter number>	Specify counter number. 1 ≦ counter number ≦ 99
<Position number>	Specify position number. 1 ≦ position number ≦ 999
<Character string number>	Specify character string number in numerical value which $" is added to the head. $1 ≦ character string number ≦ $99
<Contents selection>	Select either counter or position or character string number corresponding to <Counter number/ position number/character string number>. 0: Counter number (Default) 1: Position number 2: character string number

【 Explanation 】
(1) This command receives the specified counter value, the coordinate value of the position number or the data of the specified character string is received according to the PRN command through the RS-232-C port.
(2) The OPN command must be executed first to open the RS-232-C channel.
(3) If the counter number is omitted, the data will be read into the internal register. If the character string number is omitted, the data will be read into the character string register. If the position number is omitted, an error will occur during execution.
(4) The data is sent from an external device such as a personal computer using the PRM command. The robot program will stop while the data is being red.
(5) The PRM command can be executed before the INP command while the program is running. In that case, the sent PRM command data will be registered once, and then will be led into the specified counter, position or character string when the INP command is executed.A max. of 256 characters can be registered in the robot. If the PRM command is executed in succession and the number of registered characters exceeds 256 characters, the robot will be set to the "L" level based on the RS-232-C ER (DRT) and RS (RTS) signal lines (DR (DSR) and CS (CTS) signal lines on the personal computer side). Temporarily stop the data transmission from the personal computer during this time.
(6) If there is an error in the data sent by the PRN command, an error will occur when the INP command is executed.

【 Sample program 】 (Movemaster command)

```
10 OPN 2,1          ; Opens the RS-232C port.
20 INP 2,1,0        ; Reads the data of counter 1 from the RS-232C port.
30 INP 2,5,1        ; Reads the data of position 5 from the RS-232C port.
40 IC 1             ; Adds 1 to the contents of counter 1.
50 MO 5             ; Moves to position 5.
60 OPN 1,1          ; Opens the RS-232C port.
70 INP 1,$10,2      ; Reads the data of character string 10 from the RS-232C port.
```

IP (Increment Position)

【 Function 】
Moves the robot to a predefined position with a position number greater than the current one. (Joint interpolation)

【 Input Format 】

```
IP
```

【 Explanation 】
(1) Moves the robot to a predefined position with a position number greater than, and closest to, the current one. (See the DP command.)
(2) Error occurs if there is no predefined position which is greater in position number than the current position.
(3) Even if an error occurs, the current position number still remains unchanged.

【 Sample program 】(Movemaster command)

```
10 MO 5                  ; Moves to position 5.
20 MO 4                  ; Moves to position 4.
30 MO 3                  ; Moves to position 3.
40 IP                    ; Moves to position 4.
50 IP                    ; Moves to position 5.
```

LG (If Larger)

【 Function 】
This compares the value of the internal register with a specified value. If larger, the program will jump. The character string register and the numbers of characters in a specified character string are compared. If the character string register is larger, the program will jump.

【 Input Format 】

```
L G   <Compared value/Character string number>, <Branching line number>
```

【 Term 】

<Compared value>	Specify the value compared with the internal register. −32768 ≦ Compared value (decimal) ≦ 32767 & 8000 ≦ Compared value (hexadecimal) ≦ & 7FFF @1 ≦ counta number ≦ @99
<Character string number>	Specify character string number in numerical value which "$" is added to the head. $1 ≦ character string number ≦ $99
<Branching line number>	Specify the line number to which the program jumps when the value of the internal register is larger than compared value. 1 ≦ branching line number ≦ 32767

【 Explanation 】
<When compared value is specified> >
(1) Causes a jump to occur conditionally in accordance with the external input data or the internal counter value.
(2) If the internal register value is larger than the compared value (i.e., when the condition is met), the program jumps to the specified line. Otherwise (i.e., when the condition is not met), the program continues in sequence.
(3) A value can be loaded into the internal register by executing the input command (See ID) for the external input data or by executing the counter set command (See CP) for the counter data. Accordingly when you carry out conditional branching, need to execute either of the above commands beforehand.
(4) The compared value may be defined either in decimal or hexadecimal. A hexadecimal value must be headed by "&".

<When character string number is specified>
(1) The conditions will jump depending on the data input from an external source or the number of characters in a specified character string.
(2) If the number of characters in the character string register is larger than the number of characters in a specified character string (when the conditions are established), the program will jump to the specified line number. If the number is smaller (when conditions are not established), the next line will be executed. If the specified line number is not registered, an error will occur when jumping.
(3) By executing an INP command, the data input from an external device will be set in the character string register. The details of the character string number will be set by executing a CP command. Thus, when executing condition jumping, one of these commands must be executed first.

【 Sample program 】(Movemaster command)

```
100 ID               ; Fetches the data from the external input port.
110 LG 100,130       ; If the input data is larger than 100, jumps to line 130.
120 ED               ; Else program ends.
130 MO 1             ; Moves to position 1.
140 OPN 1,1          ; Opens the RS-232C port.
150 INP  1, ,2       ; Reads the data of character string register from the RS-232C port.
160 LG  $5,200       ; Jumps to line 200 if the data length large than character string number 5.
 :
200 ED               ; Ends program.
```

LR* (Line Read)

【 Function 】
Reads the program of the specified line number. (Using RS-232C)

【 Input Format 】

```
L R   [<line number>]
```

【 Term 】

< Line number >	Specify the line number to be read. 0 ≦ line number ≦ 32767 (If omitted, reads the current line number stopping)

【 Explanation 】
(1) Outputs the program of the specified line number (or the current stopping line number) from the RS-232C port.
(2) The output format is ASCII coded as follows;
・If you specify the line number,..Program content is read.
・If you omit the line number (or specify zero)........................Current stopping line number is read.
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. "LINE INPUT #" statement is equivalent to this in BASIC.
(4) The hexadecimal 0D is read out when the specified line has not been defined.
(5) If an error takes place, you can confirm the line number in which the error occurs by executing the LR command without line number.

【 Sample program 】(N88BASIC)

```
10 OPEN" COM1:E83" AS#1   ; Opens the RS-232C communication file from the personal computer in BASIC.
20 INPUT "Start line ";   ; Enter the top line number that you want to read.
30 INPUT "End line";      ; Enter the last line number that you want to read.
40 FOR I=S TO E STEP 10   ; Repeatedly
50 PRINT #1," LR" +STR$(I) ; Transmit "LR" + "line number"  to the controller side.
60 LINE INPUT #1,A$       ; Saves the received data to A $.
70 IF A$="" THEN 90       ; If there is no data, jumps to line 90.
80 PRINT I ; : PRINT A$   ; Displays the data on the personal computer  screen.
90 NEXT                   ; Repeats and jumps to line 40.
100 ED

RUN                       ; Run the BASIC program.
```

MA (Move Approach)

【 Function 】
Moves the hand tip to the added position. (Linear interpolation)

【 Input Format 】

```
M A    <position number(a)>, <position number(b)> [, [<O/C>]]
```

【 Term 】

<Position number (a)>	Specify the position number to be added. (The reference position)
<Position number (b)>	Specify the position number to add. (The increments position) 1 ≦ position number (a) (b) ≦ 999
<O/C>	Specify open or close state of the hand. O: Hand open C: Hand close

【 Explanation 】
(1) Moves to the added position, i.e., the coordinates of positions (a) and (b) are added to make the destination, although positions (a) and (b) remain unchanged after executing the MA command. (See the SF command.)
(2) If the open/close state of the hand has been specified, the robot moves after executing the hand control command. If it has not been specified, the hand state in position (a) remains valid.
(3) If the calculating results exceed the robot's operational space, error occurs before the robot moves.
(4) Error also takes place if positions (a) and (b) have not been defined.
(5) The position of the hand tip is decided by the tool length currently established.

【 Sample program 】(Movemaster command)
(1)6–axis type

```
10 HE 1                  ; Sets the current coordinates to position 1.
20 PD 5,0,0,30,0,0,0     ; Defines the Z coordinate of position 5 as 30 mm.
30 MA 1,5,O              ; Moves to the position that only Z direction added to coordinate value of position 1 by 30
                           mm with the hand opened.
```

(2)5–axis type

```
10 HE 1                  ; Sets the current coordinates to position 1.
20 PD 5,0,0,30,0,0       ; Defines the Z coordinate of position 5 as 30 mm.
30 MA 1,5,O              ; Moves to the position that only Z direction added to coordinate value of position 1 by 30
                           mm with the hand opened.
```

(3)4–axis type

```
10 HE 1                  ; Sets the current coordinates to position 1.
20 PD 5,0,0,30,,,0       ; Defines the Z coordinate of position 5 as 30 mm.
30 MA 1,5,O              ; Moves to the position that only Z direction added to coordinate value of position 1 by 30
                           mm with the hand opened.
```

* Coordinates values of position 1 and position 5 do not change.

MC (Move Continuous)

【 Function 】
Moves the robot continuously through the predefined intermediate points between two specified position numbers. (Linear interpolation)

【 Input Format 】

```
M C <position number(a)>, <position number(b)> [, [<O/C>]]
```

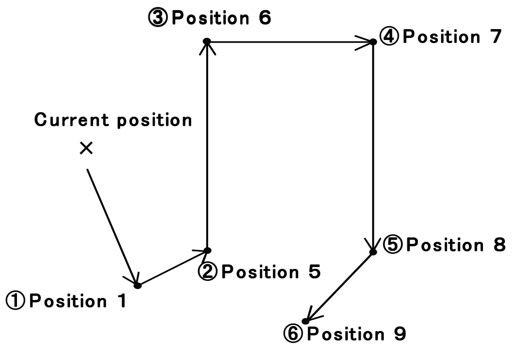
【 Term 】

<Position number (a)>	Specify the top position number moving continuous.
<Position number (b)>	Specify the last position number moving continuous. 1 ≦ position number (a) (b) ≦ 999 | Position number (a) − position number (b)| ≦ 99
<O/C>	Specify open or close state of the hand. (If omitted, the hand data of each position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) Moves the robot along the series of positions via (a) to (b) without acceleration and deceleration. (Linear interpolation)
(2) Depending on whether position number of (a) is greater than that of (b), or vice versa, the robot moves through the intermediate points in descending or ascending order. The robot decelerates to a stop as it reaches the end position.
(3) When the hand open/close setting has been done, hand control is executed before the movement.
(4) Since the robot does not accelerate or decelerate during motion, error may occur when the path involves a great change in direction of any of the joints at high speed.
(5) The speed of travel during linear interpolation is determined by the SP or SD command. (Hand tip at constant speed)
(6) Error occurs if specified positions (a) and (b) have not been defined or if the difference between the position numbers (a) and (b) exceeds 99.
(7) Error also takes place during movement if the movement path goes beyond the robot's operational space.

【 Sample program 】 (Movemaster command)

```
10 SP 10           ; Sets speed to 10.
20 MO 1            ; Moves to position 1 in joint interpolation.
30 MC 5,9          ; Moves continuously from position 5 to 9 in linear interpolation.
```



MJ (Move Joint)

【 Function 】
Turns each joint the specified angle from the current position. (Joint interpolation)

【 Input Format 】
(1)6-axis type

```
MJ   [ J1 ] [ , [ J2 ] [ , [ J3 ] [ , [ J4 ] [ , [ J5 ] [ , [ J6 ] [ , [ J7] [ , [ J8 ] ] ] ] ] ] ] ]
```

(2)5-axis type

```
MJ   [ J1 ] [ , [ J2 ] [ , [ J3 ] [ , [ J5 ] [ , [ J6 ] [ , [ J7] [ , [ J8 ] ] ] ] ] ] ]
```

(3)4-axis type

```
MJ   [ J1 ] [ , [ J2 ] [ , [ J3 ] [ , [ J4 ] [ , [ J7] [ , [ J8 ] ] ] ] ] ]
```

【 Term 】

<Each joint angle>	Specify relative amount of each joint turning from the current position.

【 Explanation 】
(1)The least increment of each axis is 0.001 (mm or degree) for the RP-1AH/3AH/5AH Series and the RH-5AH/10AH/15AH Series, the other series is 0.01 (mm or degree). (e.g. specify 20.001 for 20.001 mm for RH-5AH) (e.g. specify 20.01 for 20.01 mm for RV-1A)
(2) The open/close state of the hand does not change before and after the movement. Error occurs before the joint motion if any turning angle entry exceeds the robot's operational space.
(3) The default turning angle is 0.
(4) Refer to the "Joint jog operation" of separate manual "ROBOT ARM SETUP & MAINTENANCE" for the sign of each joint.

【 Sample program 】(Movemaster command)
(1)6-axis type

```
10 MJ 90,0,0,0,0,0          ; Turns the J1-axis + 90 degrees.
20 MJ 0,-30,0,0,0,0         ; Turns the J2-axis - 30 degrees.
30 MJ 0,0,0,20,0,0,10       ; Turns the J4-axis + 20 degrees, moves the J7-axis (additional axis 1) + 10 mm.
```

(2)5-axis type

```
10 MJ 90,0,0,0,0            ; Turns the J1-axis + 90 degrees.
20 MJ 0,-30,0,0,0           ; Turns the J2-axis - 30 degrees.
30 MJ 0,0,0,20,30           ; Turns the J5-axis + 20 degrees, the J6-axis + 30 degrees.
40 MJ 90,0,0,0,0,0,10       ; Turns the J1-axis + 90 degrees, moves the J8-axis (additional axis 2) + 10 mm.
```

(3)4-axis type

```
10 MJ 90,0,0,0              ; Turns the J1-axis + 90 degrees.
20 MJ 0,0,0,20              ; Turns the J4-axis + 20 degrees.
30 MJ 0,0,0,0,10,20         ; Moves the J7-axis (additional axis 1) + 10 mm, the J8-axis (additional axis 2) + 20 mm.
```

MO (Move)

【 Function 】
Moves the hand tip to the specified position. (Joint interpolation)

【 Input Format 】

```
MO  <position number> [, [<O/C>]]
```

【 Term 】

<Position number>	Specify the destination position number in integer value. 1 ≦ position number ≦ 999
<O/C>	Specify open or close state of the hand. (If omitted, the hand state of the position is valid) O: Hand open C: Hand close

【 Explanation 】
(1) Moves the tip of hand to the coordinates of the specified position by joint interpolation.The hand tip is decided by the tool length currently established.
(2) If open/close state of the hand1 has been specified, the robot moves after executing the hand1 control command. If it has not been specified, the definition of the specified position is executed.
Please describe ″GO 1″(Hand2 open) or ″GC 1″(Hand2 close) command before this command when you execute the control of hand2 before moving.
(3) Error takes place if the specified position has not been predefined or the movement exceeds the robot's operational space.

【 Sample program 】(Movemaster command)

```
10 SP 10                ; Sets speed to 10.
20 MO 20,C              ; Moves to position 20 with hand closed.
30 MO 30,O              ; Moves to position 30 with hand opened.
```

MP (Move Position)

【 Function 】
Moves the tip of hand to a position whose coordinates (position and angle) have been specified. (Joint interpolation)

【 Input Format 】
(1)6–axis type

```
MP  [ X ] [, [ Y ] [ , [ Z ] [ , [ A ] [ , [ B ] [ , [ C ] [ , [ L1 ] [ , [ L2 ] ] ] ] ] ] ] ]
    [ , [< R/L >]  [ , [< A/B >]  [ , [< N/F >] ] ] ]
```

(2)5–axis type

```
MP  [ X ], [ [ Y ] [ , [ Z ] [ , [ A ] [ , [ B ] [ ,  , [ L1 ] [ , [ L2 ] ] ] ] ] ] ] ] [ , [< R/L >]  [ , [< A/B >] ] ]
```

(3)4–axis type

```
MP  [ X ], [ [ Y ] [ , [ Z ] [ , , , [ C ] [ , [ L1 ] [ , [ L2 ] ] ] ] ] ] ] [ , [< R/L >]  [ , [< A/B >] ] ]
```

【 Term 】

<X, Y, Z coordinate>	Specify the position in XYZ coordinates (mm) of the robot. (Zero for default)
<A, B, C turning angle>	Specify the posture axes of the robot. A–axis is rotation angle around the X axis. B–axis is rotation angle around the Y axis. C–axis is rotation angle around the Z axis. (Zero for default) 6–axis type: The A, B, and C axes are valid. 5–axis type: The A and B axes are valid. Specify the comma in the axis which doesn't exist 4–axis type: Only the C axis is valid. Specify the comma in the axis which doesn't exist.
<L1, 2>	The coordinate value of the additional axis is designated.
<R/L>	Specify the structure flag of the robot. (Right or Left) R: Right (Default) L: Left
<A/B>	Specify the structure flag of the robot. (Above or Below)(5–axis and 6–axis type.) A: Above (Default) B: Below
<N/F>	Specify the structure flag of the robot. (Nonflip or Flip) (6–axis type only.) N: Non flip (Default) F: Flip

【 Explanation 】
(1) The least increment of the coordinate value is 0.01 mm or 0.01 degree.
(2) If the structure flag has not been specified, Right and Above flag is selected.
(3) If the specified value exceeds the robot's operational space, error occurs at the execution of the MO command.
(4) The open or close state of the hand remains the same before and after the movement.
(5) The position of hand tip is decided by the tool length currently established.

【 Sample program 】 (Movemaster command)
(1)6–axis type

```
10 MP 400,0,300,0,0,0          ; Moves to the specified coordinates. (X=400, Y=0, Z=300, A=0, B=0, C=0)
20 MP 400,0,350,90,0,90,R,A,F  ; Moves to the specified coordinates. (Structure flags are also specified.)
                                (X=400, Y=0, Z=350, A=90, B=0, C=90, Right, Above, Flip)
30 MP 400,0,350,90,0,90,20,R,A,F; Moves to the specified coordinates. (Structure flags are also specified.)
                                (X=400, Y=0, Z=350, A=90, B=0, C=90, L1=20, Right, Above, Flip)
```

(2)5–axis type

```
10 MP 250,0,300,0,180          ; Moves to the specified coordinates. (X=250, Y=0, Z=300, A=0, B=180)
20 MP 400,0,500,0,70,R,B       ; Moves to the specified coordinates. (Structure flags  are also specified.)
                               (X=400, Y=0, Z=500, A=0, B=70, Right, Below)
30 MP 400,0,500,0,70,,20,R,B   ; Moves to the specified coordinates. (Structure flags  are also specified.)
                               (X=400, Y=0, Z=500, A=0, B=70, L1=20, Right, Below)
```

(3)4–axis type

```
10 MP 200,0,100,,,0            ; Moves to the specified coordinates. (X=200, Y=0, Z=100, C=0)
20 MP 200,0,100,,,0,L          ; Moves to the specified coordinates. (Structure flags are also specified.)
                                (X=200, Y=0, Z=100, C=0, Left)
30 MP 200,0,100,,,0,10,20,L    ; Moves to the specified coordinates. (Structure flags are also specified.)
                                (X=200, Y=0, Z=100, C=0, L1=10, L2=20, Left)
```

MR (Move R)

【 Function 】
Moves the tip of hand through the predefined intermediate positions in circular interpolation.

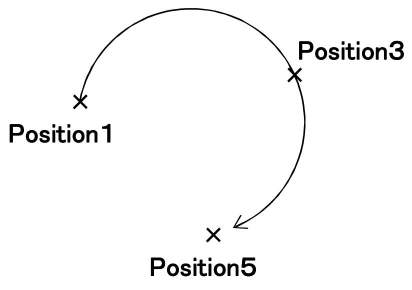
【 Input Format 】
```
MR    <Position number(a)> ,  <Position number(b)>,  <Position number(c)>
      [,  [< O/C >]]
```

【 Term 】

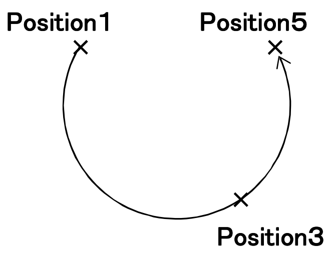
<Position number>	Specify the positions on the circle. 1 ≦ position number ≦ 999
<O/C>	Specify open or close state of the hand. (If omitted, the hand state of the position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) Moves the tip of hand through specified positions from (a) via (b) to (c) drawing an arc.
(2) The moving speed of circular interpolation is decided by the SP or SD command. (The tip of hand at constant speed.) Since the locus accuracy depends on the speed of circulari nterpolation, set the moving speed lower if you need high accuracy.
(3) The open or close state of the hand does not change before and after the movement.
(4) If the starting position (a) is different from the current position, the robot moves to the starting position by linear interpolation.
(5) If the circular interpolation is interrupted by the stop signal and restarted by the start signal, the robot moves the remaining arc. If the tip of hand is kept away from the stopping position by JOG operation in the above case, the robot moves to the stopping position by joint interpolation then moves the remaining arc.
(6) Error takes place if the specified position has not been predefined or exceeds the robot's operational space. The robot moves by linear interpolation if three positions (a), (b) and (c) are located on a straight line or if two of three positions are the same.
(7) If the moving direction of each joint changes greatly at the beginning of circular interpolation, error may occur. Set speed lower or set timer at the beginning in this case.
(8) The drawing direction and the locus of the arc depend on the order of the specified positions.
<In the case of MR 1, 3, 5>

< In the case of MR 1, 3, 5>



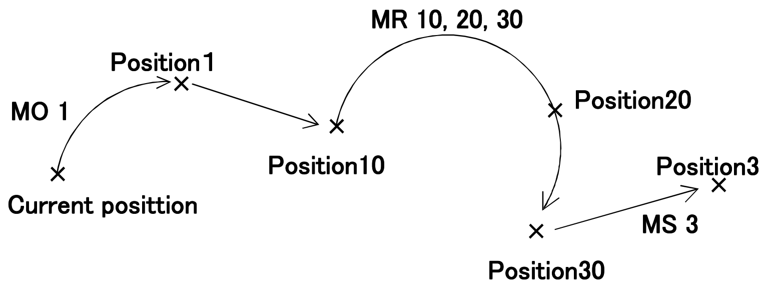
(Example 1)



(Example 2)

【 Sample program 】 (Movemaster command)

```
10 SP 8                 ; Set speed to 8.
20 MO 1                 ; Moves to position 1.
30 MR 10,20,30          ; Moves to position 10 by linear interpolation.
                          Moves the arc determined by position 10, 20, 30 by circularinterpolation.
40 MS 3                 ; Moves to position 3 by linear interpolation.
50 ED                   ; Ends program.
```



MRA (Move RA)

【 Function 】
Moves to the specified position in circular interpolation.

【 Input Format 】

```
ＭＲＡ　<position number > [, [<O/C>]]
```

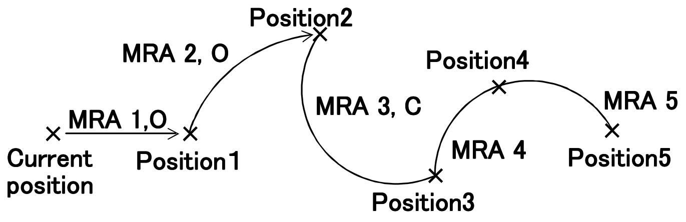
【 Term 】

<Position number>	Specify the destination position. 1 ≦ position number ≦ 999
<O/C>	Specify open or close state of the hand. (If omitted, the hand state of the position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) Moves the tip of hand on the arc which is defined by the former and the latter positions of the MRA commands. The tip of hand is decided by the tool length currently established.
(2) If the open or close state of the hand has been specified, the robot moves after executing the hand control.
(3) Error takes place if the specified position has not been predefined.
(4) If the MRA command does not continue more than three, it becomes similar to the MC command. The command except for the movement command (MO, MS, etc), however, can be executed between the MRA command.
(5) If the execution of the MRA command is interrupted and the tip of hand is kept away from the stopping position by JOG operation, the robot moves, when restarted, to the stopping position by linear interpolation then moves the remaining arc.

【 Sample program 】(Movemaster command)

```
10 MRA 1,O          ; Defines the arc with positions 1, 2, 3.
                      Moves to position 1 by linear interpolation.
20 MRA  2,O         ; Moves to position 2 by circular interpolation.
30 MRA  3,C         ; Moves to position 3 by circular interpolation.
40 TI  3            ; Timer 0.3 second.
50 MRA  4           ; Moves to position 4 by circular interpolation.
60 MRA 5            ; Moves to position 5 by circular interpolation.
70 ED               ; Ends program.
```



MS (Move Staight)

【 Function 】
Moves the tip of hand to the specified position. (Linear interpolation)

【 Input Format 】

```
MS   <position number> [, [<O/C>]]
```

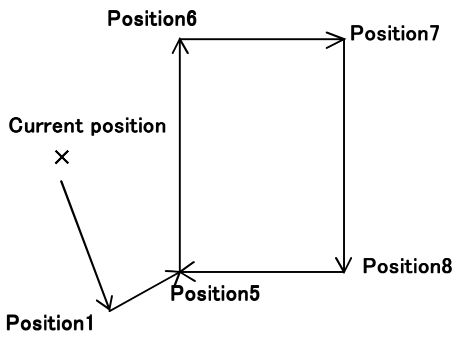
【 Term 】

<Position number>	Specify the destination position number in integer value. 1 ≦ position number ≦ 999
<O/C>	Specify open or close state of hand. (If omitted, the hand state of the position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) Moves the tip of hand to the specified position by linear interpolation.The tip of the hand is decided by the tool length currently established. (See the TL command.)
(2) Error occurs before or during movement if the destination or movement path goes beyond the robot's operational space.
(3) If open/close state of the hand1 has been specified, the robot moves after executing the hand1 control command. If it has not been specified, the definition of the specified position is executed.
Please describe "GO 1"(Hand2 open) or "GC 1"(Hand2 close) command before this command when you execute the control of hand2 before moving.
(4) The moving speed is decided by the SP or SD commands. (The tip of hand at constant speed.)
The default is SP 30.
(5) Use the MC command to move continuously through several positions by linear interpolation.

【 Sample program 】(Movemaster command)

```
10 SP 15          ; Sets speed to 15.
20 MO 1           ; Moves to position 1 by joint interpolation.
30 MS 5           ; Moves to position 5 by linear interpolation.
40 MS 6           ; Moves to position 6 by linear interpolation.
50 MS 7           ; Moves to position 7 by linear interpolation.
60 MS 8           ; Moves to position 8 by linear interpolation.
70 MS 5           ; Moves to position 5 by linear interpolation.
```



MT (Move Tool)

【 Function 】
Moves the tip of hand to a position away from the specified position by the distance as specified in the tool direction. (Joint interpolation))

【 Input Format 】

```
M T   <position number>, [<travel distance>] [, [<O/C>]]
```

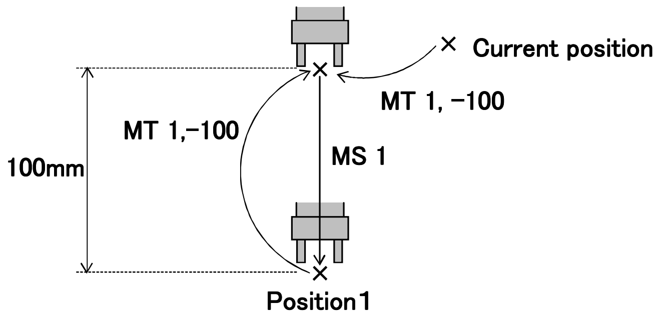
【 Term 】

<Position number>	Specify the destination position number in integer value. 1 ≦ position number ≦ 999
<Travel distance>	Specify the distance in tool direction from the specified position to the destination point. (Zero for default)
<O/C>	Specify open or close state of the hand. (If omitted, the hand state of the position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) The least increment of the distance is 0.01 mm.
(2) When the distance is positive, the tip of hand advances in the tool direction. When the distance is negative, the tip of hand retracts in the tool direction.
(3) If open/close state of the hand1 has been specified, the robot moves after executing the hand1 control command. If it has not been specified, the definition of the specified position is executed.
Please describe "GO 1"(Hand2 open) or "GC 1"(Hand2 close) command before this command when you execute the control of hand2 before moving.
(4) Error occurs when the MT command is executed if the specified position has not been predefined or if the destination exceeds the robot's operational space.

【 Sample program 】(Movemaster command)

```
10 MT 1,-100          ; Moves to the point away from the position 1 by 100 mm.
20 MS 1               ; Moves to position 1.
30 MT 1,-100          ; Moves to the point away from the position 1 by 100 mm.
```



MTS (Move Straight)

【 Function 】
Moves the tip of hand to a position away from the specified position by the distance as specified in the tool direction. (Linear interpolation)

【 Input Format 】

```
M T S   <position number>, [<travel distance>] [, [<O/C>]]
```

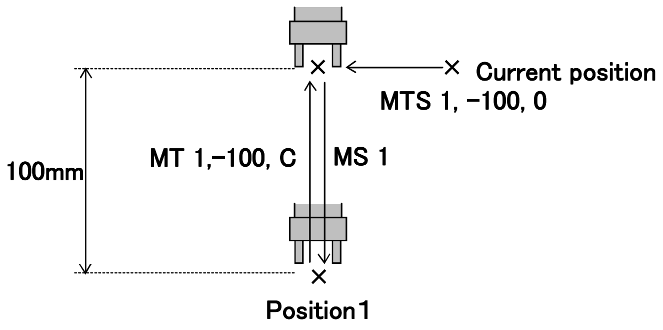
【 Term 】

<Position number>	Specify the destination position number in integer value. 1 ≦ position number ≦ 999
<Travel distance>	Specify the distance in tool direction from the specified position to the destination point. (Zero for default)
<O/C>	Specify open or close state of the hand. (If omitted, the hand state of the position is valid.) O: Hand open C: Hand close

【 Explanation 】
(1) The least increment of the distance is 0.01 mm.
(2) When the distance is positive, the tip of hand advances in the tool direction. When the distance is negative, the tip of hand retracts in the tool direction.
(3) If open/close state of the hand1 has been specified, the robot moves after executing the hand1 control command. If it has not been specified, the definition of the specified position is executed.
Please describe "GO 1"(Hand2 open) or "GC 1"(Hand2 close) command before this command when you execute the control of hand2 before moving.
(4) Error occurs when the MT command is executed if the specified position has not been predefined or if the destination exceeds the robot's operational space.

【 Sample program 】 (Movemaster command)

10 MTS 1,-100,O	; Moves to the point away from the position 1 by 100 mm with hand opened. Moves straight.
20 MS 1	; Moves to position 1.
30 MTS 1,-100,C	; Moves to the point away from the position 1 by 100 mm with hand closed. Moves straight.



MUL (Mul)

【 Function 】
Multiplies the value of the internal register and the operation data, and stores it in the internal register.

【 Input Format 】

```
ＭＵＬ　<operation data>
```

【 Term 】

<Operation data>	Describes the data to be operated as a numeric value or counter No. with @. −32768 ≦ numeric value (decimal) ≦ 32767 &8000 ≦ numeric value (hexadecimal) ≦ &7FFF @1 ≦ Counter No. ≦ @99

【 Explanation 】
(1) Designate the operation data setting as numeric value or counter No.
When designating with a numeric value, use a decimal or hexadecimal value. When using a hexadecimal, add a "&" to the head of the operation data.
When setting with a counter No. add a "@" to the head of the counter No.
The contents of the set counter No. will be used as the operation data.
(2) The operation results are stored in the internal register, so operation, comparison and reading, etc of the operation results can be carried out with the related commands.
(Refer to ADD, SUB, DIV, EQ, NE, LG, SM, CL, DR, OR commands)

【 Sample program 】 (Movemaster command)

```
10 CP 1         ; Stores counter No.1 value in internal register
20 MUL @2       ; Multiplies internal register value by counter No.2 value
30 CL 3         ; Sets internal register value in counter No.3
                 (Counter No.3 = counter No.1 * counter No.2)
40  CP 1        ; Stores counter No.1 value in internal register
50  MUL 5       ; Multiplies internal register value by 5
60  CL 4        ; Sets internal register value in counter No.4
                 (Counter No.4 = counter No.1 * 5)
```

N＊(Number)

【 Function 】
Select the specified program.

【 Input Format 】

```
N    <program name>
```

【 Term 】

<Program name>	Specify the robot program name. (Less than 8 characters) Possible letter used: Digit (0–9) Character (A – Z) Impossible letter used: ( ) ＊+ , . / : ; = ? [¥] ' ″ ! @ # Special specification :When you specified only numeric value, is handled for number. Need to enclose program name with ″ ″ in the case of character used.

【 Explanation 】
(1) Select the specified program. The program selected here becomes an object of the implementation, modification and operation. The program selected once remains till other program number is selected afresh. (Even if the power turns OFF, the program number remains unchanged.)
(2) You can confirm the current program number using the QN command from the personal computer. (See the QN command.)
(3) The following name is identified as the same.
An example:Handled as the same. 1,01,001,00000001 (Only numeric value)
Handled as not the same.1,1 A, A0001 (Includes alphabetic characters)
(4) The letters that controller can indicate to the LED are 0–9, A ～ Z (simplifie

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1:E83” AS #1  ; Opens the RS-232C communication file in BASIC.
20 PRINT #1,” N10”         ; Selects the program 10.
30 PRINT #1,” 10 MO 1”     ; Implementation of program. (Line 10)
40 PRINT #1,” 20 MS 2”     ; Implementation of program. (Line 20)
50 PRINT #1,” 30 ED”       ; Implementation of program. (Line 30)
60 ED                      ; Ends
```

NE (If Not Equal)

【 Function 】
This compares the value of the internal register with a specified value. If not equal, the program will jump. The character string register and details of a specified character string are compared. If not equal, the program will jump.

【 Input Format 】

```
NE  <compared value/character string number>, <branching line number>
```

【 Term 】

<Compared value>	Specify the value that the internal register compares contents with or counter No. with @. −32768 ≦ compared value (decimal) ≦ 32767 & 8000 ≦ compared value (hexadecimal) ≦ &7FFF @1 ≦ counter No. ≦ @99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99
<Branching line number>	Specify the line number to which the program jumps when the comparison result is not equal. 1 ≦ branching line pair ≦ 32767

【 Explanation 】
<When compared value is specified>
(1) Causes a jump to occur conditionally in accordance with the external input data or the internal counter value.
(2) If the internal register value does not equal to the compared value (i.e. when the condition is met), the program jumps to the specified line. Otherwise (i.e. when the condition is not met), the program continues in sequence.
(3) A value can be loaded into the internal register by executing the input command (See ID) for the external input data or by executing the counter set command (See CP) for the counter data. Accordingly when you carry out conditional branching, need to execute either of the above commands beforehand.
(4) The compared value may be defined either in decimal or hexadecimal. A hexadecimal value must be headed by ″&″.

<When character string number is specified>
(1) The conditions will jump depending on the data input from an external source or the contents of characters in a specified character string.
(2) If the contents of characters in the character string register is not equal to the details of a specified character string (when the conditions are established), the program will jump to the specified line number. If not (when conditions are not established), the next line will be executed.
(3) By executing an INP command, the data input from an external device will be set in the character string register. The details of the character string number will be set by executing a CP command. Thus, when executing condition jumping, one of these commands must be executed first.
(4) If the specified line number is not registered, an error will occur when jumping.

【 Sample program 】(Movemaster command)

```
10 ID                  ; Fetches data from external input port.
20 NE 80,100           ; Jumps to line 100 if the data does not equal 80.
30 ED                  ; Ends the program if above condition is not met.
100 MO 7               ; Moves to position 7.
110 OPN 1,1            ; Opens the RS-232C port.
120 INP 1, ,2          ; Reads the data of character string register from the RS-232C port.
130 NE  $2,200         ; Jumps to line 200 if the data not equals character string  number 2.
  :
200 ED                 ; Ends program.
```

NT (Nest)

【 Function 】
Carry out origin setting. (The robot moves to the user-defined origin.)

【 Input Format 】

```
NT
```

【 Explanation 】
(1) The moving sequence (Parameter UNG) of each joint is fixed by the parameter.
(2) If the arm can interfere with the object surrounding the robot, use the teaching box to move it to a safe location before origin setting.
(3) You can change the moving sequence and the attitude of the origin setting by the parameter.
(4) The origin setting order parameter UNG is set to 1,1,1,1,1,1,1,1 as the default. All axes simultaneously. (No. 1 axis to No. 8 axis from left. The numerical value shows the operation order.) The values are the movement order. When using NT, set the movement order before using.
The set values of the operation order are from 1 to 6. Does not operate even if other numerical values are set.
(5)The origin parameter to specify by customer is ″USERORG″

【 Sample program 】 (Movemaster command)

```
10 NT                    ; Executes origin setting.
20 MO 1                  ; Moves to position 1.
30 ED                    ; Ends program.
```

NW * (New)

【 Function 】
Deletes the specified program and position data.

【 Input Format 】

```
NW
```

【 Explanation 】
(1) Deletes all positions, counters, and character strings of the selecting program. Common positions (901-999), common counters (91-99), and common character strings ($91-$99) however, are not deleted.
(2) Origin setting, internal register, tool length, speed setting, pallet setting, and hand setting remain unchanged even if the NW command is executed.
(3) The NW command can not be executed in the program with line number. (Only direct execution is possible.)

【 Sample program 】 (N88BASIC)

```
10 OPEN ” COM1:E83” AS #1  ; Opens the RS-232C communication file from the personal computer in BASIC.
30 PRINT #1,” NW”          ; Transmit command ″NW″
60 ED                      ; Ends program.
```

NX (Next)

【 Function 】
Specifies the range of a loop in a program executed by the RC command.

【 Input Format 】

```
NX
```

【 Explanation 】
(1) Used in combination with the RC command to specify the range of a loop in a program executed by the RC command.
(2) Error occurs if there is no corresponding ″RC″ command specified.

【 Sample program 】
See the RC command.

OB (Output Bit)

【 Function 】
The output state of the bit designated as the external output signal is set.

【 Input Format 】

```
OB   [<+/->] <Bit number>
```

【 Term 】

<+/->	Set ON or OFF state of the specified bit. + : Bit ON - : Bit OFF
<Bit number>	Specify the bit number of external output. 0 ≦ bit number ≦ 32767

【 Explanation 】
(1) Set "+" to switch on the specified bit and "-" to switch off the specified bit.
(2) All bits other than the specified one are not affected by this command. The output state of the specified bit is retained until a new setting is made by the OB or OD command.
(3) This is a dedicated input/output parameter, and cannot be output to a bit assigned as a dedicated output. If output, an error will occur. As the default, the output signal Nos. 0 to 3 are assigned as dedicated outputs. Refer to the separate "Instruction Manual/Detailed Explanation of Functions and Operations" for details for details on the dedicated input/output parameters.
(4) For the pneumatic hand, the open/close state of hand 1 to 8 can be set using the OB command. Refer to Table 2-6 for parameter HANDTYP D900, D902 and following, and output.
This cannot be set for the motorized hand.

Table 2-6 : Opening/closing pneumatic hand with OB command.

Hand	Open/Close	GR1	GR2	GR3	GR4
		Output bit 900	Output bit 901	Output bit 902	Output bit 903
Hand 1	Open (GO 0)	ON	OFF	—	—
	Close (GC 0)	OFF	ON	—	—
Hand 2	Open (GO 1)	—	—	ON	OFF
	Close (GC 1)	—	—	OFF	ON

Notice : GR1-GR4 shows connector number of hand output cable in the robot arm.

【 Sample program 】(Movemaster command)

```
10 OD &FFFF          ; Turns the bits (0-15) of external output into ON entirely.
20 OB -10            ; Turns only bit 10 to OFF.
30 ED                ; Ends program.
```

OC (Output Counter)

【 Function 】
The designated Nos. counter value is unconditionally output to the external output signal.

【 Input Format 】

```
O C   <counter number> [, [<output bit>] [, [<bit width>]]]
```

【 Term 】

<Counter number>	Specify the counter number to be output. 1 ≦ counter number ≦ 99
<Output bit number>	Specify the reference bit number of output data 0 ≦ bit number ≦ 32767 (0 for default)
<Bit width>	Specify bit width of output data. 1 ≦ bit width ≦ 16 (16 for default)

【 Explanation 】
(1) The designated counter value is unconditionally output to the external output signal in its original state. The output data retains after that.
(2) Even if the OC command is executed, the value of the specified counter and the internal register remain intact.
(3) You can specify the range of output signal by setting the bit width of the OC command.

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1:E83” AS #1  ; Opens the RS-232C communication file from the personal computer in BASIC.
20 PRINT #1,” SC 5,&0008”   ; Set value 8 to counter 5
30 PRINT #1,” OC 5,8”       ; The counter 5 value is unconditionally output to external output signal bit 8
40 ED                       ; Ends program.
```

[Caution] Outputs signal bit 0 to 3 have dedicated signals assigned.

OD (Output Direct)

【 Function 】
Outputs the specified data unconditionally through the output port.

【 Input Format 】
```
OD   <output data> [, [<output bit number>] [, [<bit width>]]]
```

【 Term 】

<Output data>	Specified output data. −32768 ≦ output data (decimal) ≦ 32767 & 8000 ≦ output data (hexadecimal) ≦ & 7FFF
<Output bit number>	Specify the reference bit number of output data. 0 ≦ bit number ≦ 32767 (0 for default)
<Bit width>	Specify the bit width of output data. 1 ≦ bit width ≦ 16 (16 for default)

【 Explanation 】
(1) Outputs a signal (parallel data) unconditionally through the output port to external equipment such as a programmable controller. The output data retains after that.
(2) output data is defined either in decimal or hexadecimal. The data defined in hexadecimal must be headed by "&".
(3) For information on connections, refer to the separate manual "Standard Specifications".
(4) You can specify the range of output signal by setting the bit width of the OD command.
(5) This is a dedicated input/output parameter, and cannot be output to a bit assigned as a dedicated output. If output, an error will occur. As the default, the output signal Nos. 0 to 3 are assigned as dedicated outputs. Refer to the separate "Instruction Manual/Detailed Explanation of Functions and Operations" for details for details on the dedicated input/output parameters.

【 Sample program 】(Movemaster command)
```
10 OD &FFFF             ; Sets the output port of 16 bits width from bit 0 to ON.
30 OD  &FFFF,10,15      ; Sets the output port of 15 bits width from bit 10 to ON.
40 ED                   ; Ends program.
```

OG (Origin)

【 Function 】
Moves to the user−defined origin. (Joint interpolation)

【 Input Format 】
```
OG
```

【 Explanation 】
(1) Moves to the user−defined origin specified by the parameter USERORG by joint interpolation.
(2) The attitude of the robot, after executing the OG command, is the same as the attitude after executing the NT command. The parameter, which defines the moving sequence of origin setting, does not effect the OG command.

【 Sample program 】(Movemaster command)
```
10 NT                   ; Executes the origin setting.
20 MO 2                 ; Moves to position 2.
30 OG                   ; Moves to origin.
40 ED                   ; Ends program.
```

OPN (Open)

【 Function 】

Opens communication channel and specify input/output device.

【 Input Format 】

```
OPN   <channel number>, <device number>
```

【 Term 】

<Channel number>	Specify input/output channel number. 0 ≦ channel number ≦ 7
<Device number>	Specify input/output device number. 1: Standard RS-232C

【 Explanation 】

(1) The corresponding relation of the channel number and input/output devices is specified, and the channel is opened.
However, multiple <channel numbers> cannot be set in the same <input/output device number>.

(2) The counter, position and character string data can be read in with the INP command.

【 Sample program 】

```
10 OPN 1,1                ; The standard RS-232-C is opened.
                           (Channel number 1 is assigned.)
20  INP 1,1,1             ; The position data is read from RS-232-C.
30 INP 1,1,0              ; The counter data is read from RS-232-C.
40 INP 1,$1,2             ; The character string data is read from RS-232-C.
```

OR (Or)

【 Function 】
ORs the specified data and the internal register data.

【 Input Format 】

```
OR   <operation data>
```

【 Term 】

<Operation data >	Specify the data to be operated or counter No. with @. −32768 ≦ operation data (decimal) ≦ 32767 & 8000 ≦ operation data (hexadecimal) ≦ &7FFF @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) Specify the data to be operated in decimal or hexadecimal. Any hexadecimal value must be headed by ″&″.
(2) The operation result is stored into the internal register and can be changed, compared or read by relevant commands.
(See the ADD, SUB, MUL, DIV, EQ, NE, LG, SM, CL, DR, AN, XO commands)
(3) Execution of the OR command after the input commands (ID)allows to be set to the required bits of the input data fetched from the external device.

【 Sample program 】 (Movemaster command)

```
10 ID                ; Fetches data from external input port.
20 OR &FFF0          ; Sets 1 to all bits except lower order 4 bits.
30 EQ &FFFF,100      ; If the above data are all bit 1, jumps to line 100.
40 ED                ; Ends program.
  :
100 MO 10            ; Moves to position 10.
```

OVR (Override)

【 Function 】
Specify program override.

【 Input Format 】

```
OVR   <specified override>
```

【 Term 】

<Specified override>	Specify override value. (%) 1 ≦ specified override ≦ 200

【 Explanation 】
(1) Specifies the ratio of working speed of the robot.
(2) The OVR command is effective for every interpolation mode, i.e. joint interpolation, linear interpolation and circular interpolation.
(3) The actual speed in the program eventually becomes the following.
Joint interpolation speed = operation paner (T/B) x OVR command setting value x SP command setting value.
Linear interpolation speed = operation paner (T/B) x OVR command setting value x SP or SD command setting value.
Here, the playback override can be specified by means of the starting display of teaching box or the external input signal. The override specified by the OVR command is called program override.
(4) The initial value of program override is 100 %.
(5) The override value once specified in the program is effective till new value is set or the program ends.
(6) Error occurs at the execution of the OVR command if the value 0 is set to the specified override.
(7) As the acceleration and deceleration distance required for movement are preset, when the specified speed and acceleration/deceleration are set, if the movement distance is small, the set speed may not be reached.

【 Sample program 】 (Movemaster command)

```
10 SP 30                 ; Sets working speed 30 (100 %).
20 OVR 80                ; Sets override 80 %.
3 MO 2                   ; Moves to position 2.
40 ED                    ; Ends program.
```

＊If the operation panel (T/B) is specified to 50 % in the above example, the actual override is as follows;
Joint interpolation speed = 50 x 80 x 100 (%) = 40 (%)
The robot moves to position 2 with the speed of 40 % of maximum value.

PA (Pallet Assign)

【 Function 】
Defines the number of grid points in the column and row directions for the specified pallet.

【 Input Format 】
```
PA   <pallet number>, <number of column grid points>, <number of row grid points>
```

【 Term 】

<Pallet number>	Specify number of pallet using. 1 ≦ pallet number ≦ 9
<Number of column grid points >	Set grid points of column of pallet. 1 ≦ number of column grid points ≦ 32767
<Number of row grid points>	Set grid points of row of pallet. 1 ≦ number of row grid points ≦ 32767

【 Explanation 】
(1) The PA command must be executed before the pallet calculation command (see the PT command) is executed.
(2) The number of grid points is equivalent to that of the actual workpieces arranged on the pallet. For example, with a pallet holding 15 workpieces (3x5), the numbers of column and row grid points are 3 and 5, respectively.
(3) The column and row directions are decided by the directions of the terminating positions, respectively. (See the PT command)

【 Sample program 】(Movemaster command)
```
10 PA   5,20,30          ; Defines the pallet 5 as the pallet holding 20 x 30 grid points.
20 SC   51,15            ; Sets value 15 to counter 51. (column points)
30 SC   52,25            ; Sets value 25 to counter 52. (row points)
40 PT   5                ; Sets the calculated coordinates value of grid point to position 5.
50 MO   5                ; Moves to position 5. (The grid position)
60 ED                    ; Ends program.
```

PC * (Position Clear)

【 Function 】
Clears the data of the specified position(s)

【 Input Format 】
```
PC   <position number (a)> [, [<position number (b)>]]
```

【 Term 】

<Position number>	Specify position number deleting. 1 ≦ position number (a), (b) ≦ 999 Position number (a) ≦ position number (b)

【 Explanation 】
(1) Deletes all position data between positions (a) and (b). (Position (b) included)
(2) If the position number (a) is greater than the position number (b), error occur.

【 Sample program 】(N88BASIC)
```
10 OPEN ” 1:E83” AS #1  ; Opens the RS-232C communication file from the personal computer in BASIC.
20 PRINT #1,” MO 10”    ; Moves to position 10.
30 PRINT #1,” MO 11”    ; Moves to position 11.
40 PRINT #1,” MO 12”    ; Moves to position 12.
50 PRINT #1,” PC 11”    ; Delete the position 11.
60 PRINT 31,” DP”       ; Moves to position 10.
70 ED                   ; Ends program.
```

PD (Position Define) P D

【 Function 】

Defines the coordinates (location and angle) of the specified position.

【 Input Format 】

(1)6-axis type

```
P D  <Position number>,[[X] [, [Y] [, [Z] [, [A] [, [B] [, [C] [, [L 1] [, [L 2]
     [, [<R/L>]  [, [<A/B>]  [, [<N/F>] ] ] ] ] ] ] ] ] ] ] ] ] [, <O/C>]
```

(2)5-axis type

```
P D  <Position number>, [[X] [, [Y] [, [Z] [, [A] [, [B] [,, [L 1] [, [L 2]
     [, [<R/L>]  [, [<A/B>]  ] ] ] ] ] ] ] ] ] ] [, <O/C>]
```

(3)4-axis type

```
P D  <Position number>, [[X] [, [Y] [, [Z] [,,, [C] [, [L 1] [, [L 2]
     [, [<R/L>] ] ] ] ] ] ] ] [, <O/C>]
```

【 Term 】

<Position number>	Specify position number defining. 1 ≦ position number ≦ 999
<X, Y, Z coordinate>	Specify the location (mm) in XYZ coordinates of the robot. (0 for default)
<A, B, C turning angle>	Specify the posture axes of the robot. A-axis is rotation angle around the X axis. B-axis is rotation angle around the Y axis. C-axis is rotation angle around the Z axis. (Zero for default) 6-axis type: The A, B, and C axes are valid. 5-axis type: The A and B axes are valid. Specify the comma in the axis which doesn't exist 4-axis type: Only the C axis is valid. Specify the comma in the axis which doesn't exist.
<L1, 2>	The coordinate value of the additional axis is designated. (0 for default)
<R/L>	Specify the structure flag of the robot. (Right or Left) R: Right (Default) L: Left
<A/B>	Specify the structure flag of the robot. (Above or Below) (5-axis and 6-axis type.) A: Above (Default) B: Below
<N/F>	Specify the structure flag of the robot. (Nonflip or Flip) (6-axis type only.) N: Non flip (Default) F: Flip
<O/C>	Specify open or close state of hand 1. O: Hand 1 open C: Hand 1 close

【 Explanation 】

(1) The least increment of the coordinate value is 0.01 mm or 0.01 degree (e.g. specify 20.01 for 20.01 mm).

(2) Error does not occur even if the specified coordinates exceed the robot's operational space. The PD command, combined with the SF and the MA command, can define the amount for moving.

【 Sample program 】 (Movemaster command)

(1)6-axis type

```
10 PD 10,50,320,70,50,40,30,R,A,N,O   ; Defines the location and angle of position 10. (X=50, Y=320, Z=70, A=50,
                                        B=40, C=30, Right, Above, Nonflip, Hand1 open)
40 MO 10                              ; Moves to position 10.
50 ED                                 ; Ends program.
```

(2)5-axis type

```
10 PD 10,50,320,70,40,30,R,A,O        ; Defines the location and angle of position 10. (X=50, Y=320, Z=70, A=40,
                                        B=30, Right, Above, Hand1 open)
20 PD 11,50,320,70,40,30,,10,R,A,O    ; Defines the location and angle of position 11. (X=50, Y=320, Z=70, A=40,
                                        B=30, L1=10, Right, Above, Hand1 open)
30 MO 10                              ; Moves to position 10.
40 MO 11                              ; Moves to position 11.
50 ED                                 ; Ends program.
```

(3)4-axis type

```
10 PD 10,50,320,70,,,30,R,O           ; Defines the location and angle of position 10. (X=50, Y=320, Z=70, C=30,
                                        Right, Hand1 open)
```

```
20 PD 11,50,320,70,,,30,20,25,R,O      ; Defines the location and angle of position 11. (X=50, Y=320, Z=70, C=30,
                                         L1=20, L2=25, Right, Hand1 open)
30 MO 10                               ; Moves to position 10.
40 MO 11                               ; Moves to position 11.
50 ED                                  ; Ends program.
```

PL (Position Road)

【 Function 】
Replaces position (a) by position (b).

【 Input Format 】

```
PL   <position number(a)>,  <position number(b)>
```

【 Term 】

<Position number (a) >	Specify the position number or counter No. with @. (Destination) 1 ≦ position number (a) ≦ 999 @1 ≦ counter No. ≦ @99
<Position number (b)>	Specify the position number or counter No. with @. (Source) 1 ≦ position number (b) ≦ 999 @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) After executed, the PL command causes the coordinates of position (a) to be equivalent to those of position (b) and the old coordinates of position (a) to be cleared.
(2) After executed, the PL command also assigns the hand state at position (b) to that at position (a).
(3) Error occurs if position (b) is not defined.
(4) A new position is created if the position (a) is not defined.

【 Sample program 】 (Movemaster command)

```
10 HE 2              ; Sets the current coordinates and hand state to position 2.
20 PL 3,2            ; Replaces the coordinates of position 3 by position 2.
30 ED                ; Ends program.
```

PR (Position Read)

【 Function 】
Reads the coordinates of the specified position and the open/close state of the hand. (Using RS-232C)

【 Input Format 】

```
PR   [<position number>]
```

【 Term 】

<Position number>	Specify the position number that you want to read. 0 ≦ position number ≦ 999 (If omitted, the current position number is valid.)

【 Explanation 】
(1) Outputs the coordinates of the specified position and the open/close state of the hand throgh the RS-232C port. If the position number is omitted or equals 0, only the current position number is output .
(2) The data is ASCII coded as follows; The least increment is 0.01 mm or 0.01 degree.
Output format
6-axis type : X, Y, Z coordinate value, A, B, C turning angle degree, R/L, A/B, N/F, O/C
5-axis type : X, Y, Z coordinate value, A, B turning angle degree, R/L, A/B, O/C
4-axis type : X, Y, Z coordinate value,,, C turning angle degree, R/L, O/C
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.
(4) If you read the position that is not used in the program and not defined yet, ″zero″ is output with each coordinate.
6-axis type : 0,0,0,0,0,0
5-axis type : 0,0,0,0,0
4-axis type : 0,0,0,,,0
If you read the position that is already used in the program but not defined yet, ″0.00″ is output with each coordinate.
6-axis type : 0.00,0.00,0.00,0.00,0.00,0.00
5-axis type : 0.00,0.00,0.00,0.00,0.00
4-axis type : 0.00,0.00,0.00,,,0.00
(5) If you specify ″zero″ to the position number or omit it, you can identify the current position number when an error occurs in executing moving command.

【 Sample program 】(N88BASIC)
(1) 6-axis type

```
10 OPEN ” COM1:E83” AS #1      ; Opens the RS-232C communication file from the personal computer in BASIC.
20 INPUT ″Position number is″;P ; Inputs the position number that you want to read
30 PRINT #1,” PR” +STR$(P)     ; Transmits the PR command followed by the position number.
40 LINE INPUT #1,A$            ; Saves the received data to A$.
50 PRINT A$                    ; Displays the contents to the screen.
60 ED                          ; Ends program.

RUN                            : Run the program.
Position number is ? 15        ; Enter the position number.
+10.00,+380.00,+300.00,-70.00,+50.00,+40.00,R,A,N,C ; Outputs the contents of the position.
```

(2)5-axis type

```
10 OPEN ” COM1:E83” AS #1      ; Opens the RS-232C communication file from the personal computer in BASIC.
20 INPUT ″Position number is″;P ; Inputs the position number that you want to read
30 PRINT #1,” PR” +STR$(P)     ; Transmits the PR command followed by the position number.
40 LINE INPUT #1,A$            ; Saves the received data to A$.
50 PRINT A$                    ; Displays the contents to the screen.
60 ED                          ; Ends program.

RUN                            ; Run the program.
Position number is ? 15        ; Enter the position number.
+10.00,+380.00,+300.00,+50.00,+40.00,R,A,C ; Outputs the contents of the position.
```

(3)4−axis type

```
10 OPEN ” COM1:E83” AS #1       ; Opens the RS−232C communication file from the personal computer in BASIC.
20 INPUT ″Position number is″;P ; Inputs the position number that you want to read
30 PRINT #1,” PR” +STR$(P)      ; Transmits the PR command followed by the position number.
40 LINE INPUT #1,A$             ; Saves the received data to A$.
50 PRINT A$                     ; Displays the contents to the screen.
60 ED                           ; Ends program.

RUN                             ; Run the program.
Position number is ?  15        ; Enter the position number.
+10.00,+380.00,+300.00,,,40.00,R,C ; Outputs the contents of the position.
```

PRN ∗ (Print)

【 Function 】
The counter number setting value, position number coordinate value or the character string number data is transmitted from the personal computer in regard to the INP command. (Using RS-232-C)

【 Input Format 】

```
PRN  <counter value> | <position coordinates>| "<character string data>"
```

【 Term 】

<Counter value>	Specify the counter value setting to a counter. -32768 ≦ Counter value (decimal) ≦ 32767 & 8000 ≦ Counter value (hexadecimal) ≦ & 7FFF
<Position coordinates>	Specify the coordinates value setting to a position. Specify the following coordinates similar to the PD command. (See the PD command) (1) 6-axis type <X, Y, Z coordinates>, <A, B, C turning angle>, <additional axis 1, additional axis 2>, <R/L>, <A/B>, <N/F>, <O/C> (2) 5-axis type <X, Y, Z coordinates>, <A, B turning angle>,, <additional axis 1, additional axis 2>, <R/L>, <A/B>, <O/C> (3) 4-axis type <X, Y, Z coordinates>,,,<C turning angle>, <R/L>, <O/C>
<Character string data>	Specify the character string to be set. 120 characters or less including the line number and the command. Usable characters: numerals (0 to 9), alphabetic characters(A to Z),symbols (!@#, etc.)

【 Explanation 】
(1) Transmits the setting value of counter, the coordinates value of position or the character strings from the personal computer through the RS-232C port responding to the INP command in the program.
(2) The robot becomes wait condition in the INP command till the date is transmitted from personal computer by executing the PRN command.
(3) You can execute the PRN command prior to the INP command during the execution of the program.
(4) When transmitting character string data, enclose the character string in double quotations (").

【 Sample program 】(N88BASIC)
<Personal computer program>

```
10 OPEN " COM1:E83" AS #1     ; Opens the RS-232C communication file from the personal computer in BASIC.
20 INPUT "Counter data is" ;J ; Enter the setting value of counter from the personal computer.
30 PRINT #1,"PRN" +STR$(J)    ; Transmits the setting value of counter.
40 PRINT #1,"PRN 100,0,0,0,0,0" ; Transmits the coordinates value of position.
50 ED                         ; Ends program.
```

<Robot program>

```
10 OPN 2,1                    ; Opens the RS-232C port.
20 INP 2,1,0                  ; Reads the data from RS-232C port to counter 1.
30 INP 2,5,1                  ; Reads the data from RS-232C port to position 5.
40 INP 2,$3,2                 ; Reads the data from RS-232C port to character string 3.
50 IC 1                       ; Add 1 to counter 1.
60 MO 5                       ; Moves to position 5.
```

PT (Pallet)

【 Function 】
Calculates the coordinates of a grid point on the specified pallet and sets the coordinates value to the specified position.

【 Input Format 】

```
PT  <pallet number>
```

【 Term 】

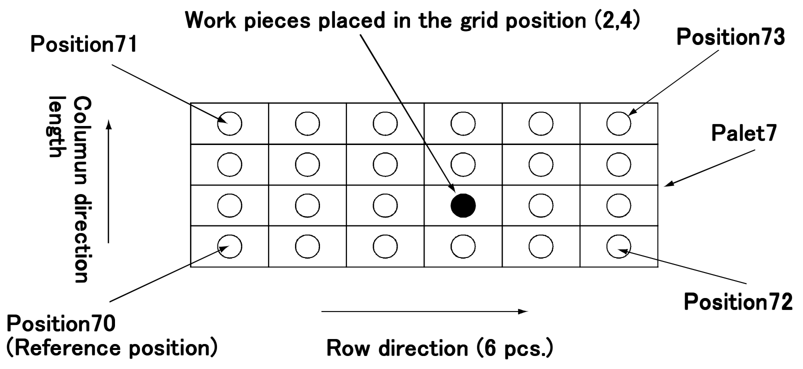
<Pallet number>	Specify the number of pallet using.
	1 ≦ pallet number ≦ 9

【 Explanation 】
(1) Calculates the coordinates of a grid point on the specified pallet and sets the coordinates to the position which number is corresponding to the specified pallet number. Before the PT command is executed, the pallet definition command (PA) must be executed for the pallet to be used. After the PT command has been executed, the position data previously defined for the target position is cleared.
(2) In order for the PT command to be executed, the pallet positions (grid points at four corners of the pallet) must be properly defined which identify a particular pallet and the pallet counters (column and row) be properly set that specify a particular grid point on the pallet. If the pallet positions and pallet counters are properly defined, therefore, execution of the PT command allows the coordinates of a grid point to be defined as the position number equivalent to the pallet number. The following is a listing of a combination of pallet positions and counters corresponding to each pallet number.

Pallet number	1	2	3	4	5	6	7	8	9
Pallet reference position	10	20	30	40	50	60	70	80	90
Pallet column terminating position	11	21	31	41	51	61	71	81	91
Pallet row terminating position	12	22	32	42	52	62	72	82	92
Pallet cornet position opposite ro reference	13	23	33	43	53	63	73	83	93
Pallet column counter	11	21	31	41	51	61	71	81	91
Pallet row counter	12	22	32	42	52	62	72	82	92
Pallet grid position	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

(3) Error occurs if the pallet position have not been defined and the pallet counters have not been set or have been set to have values exceeding those defined by the PA command. Error does not occur, however, even when the coordinates obtained for the grid point exceed the robot's operational space.
(4) The open or close state of the hand at the target grid point is the same as that in the pallet reference position.
(5) When executing the PT command, the tool length of the hand to be used must be properly defined by the TL command. The robot must be taught through the pallet positions (four corners) using the predefined correct tool length.
(6) Pay attention when you use pallet nine because the counters (91,92) of pallet nine are common counters among programs and other program may use these counters.

【 Sample program 】 (Movemaster command)
Suppose there is a pallet on which a total of 24 workpieces are arranged, 4 in the column direction and 6 in the row direction. Then have the system compute the coordinates of the workpiece placed in the grid position (2,4), i.e. the second grid in the column direction and the fourth grid in the row direction, and get the robot hand to reach that position. Assume that pallet 7 is used.



＊You must teach the positions at four corners (70,71,72,73) in advance.

```
10 TL 200          ; Sets the tool length equivalent to the hand using.
20 PA 7,4,6        ; Sets the pallet number and the grid points of column and row.
30 SC 71,2         ; Sets the number of grid point in column direction.
40 SC 72,4         ; Sets the number of grid point in row direction.
50 PT 7            ; Allows the coordinates of the target grid point to be set to position 7.
60 MO 7            ; Moves to position 7.
70 ED              ; Ends program.
```

PW (Pulse Wait)

【 Function 】
Waits for in-position of servomotor about all joints till it becomes within the specified value.

【 Input Format 】

```
PW  <positioning pulse>
```

【 Term 】

<Positioning pulse >	Specify the judgment pulse number of in-position. $1 \leqq$ positioning pulse $\leqq 10000$

【 Explanation 】
(1) Waits for in-position of servomotor about all joints till it becomes within the specified value.
(2) If you need the positioning of high accuracy when chucking a workpiece at a position, set small value for positioning pulse. If you need the positioning of low accuracy, set large value. If you set small value to the positioning pulse, the robot waits for positioning having the same effect as the TI command is executed.
(3) If you set small value to the positioning pulse when the handling workpiece is relatively heavy or the robot is moving at high speed, it may take a longer time to position than usual.
(4) Initial value of positioning pulse is 10000 pulse.
(5) When the setting value exceeds the above limit, error occurs.

【 Sample program 】 (Movemaster command)

```
10 PW 100          ; Waits for the positioning pulse becoming within 10 pulses.
20 MO 1            ; Moves to position 1.
30 GC              ; Closes hand.
40 ED              ; Ends program.
```

PX (Position Exchange)

【 Function 】
Exchanges the coordinates of the specified position for those of another specified position.

【 Input Format 】

```
P X   <position number(a)>,  <position number(b)>
```

【 Term 】

<Position number>	Specify the position number exchanging. 1 ≦ position number (a), (b) ≦ 999

【 Explanation 】
(1) After the PX command is executed, the coordinates of position (a) are exchanged for those of position (b)
(2) The open or close state of the hand at position (a) is also exchanged for that at position (b).
(3) Error occurs if positions (a) and (b) have not been predefined.

【 Sample program 】(Movemaster command)

```
10 HE 2                 ; Define the current position to position 2
20 MJ 20,30,10,0,0,0    ; Drives each joint by the specified amount.
30 GO                   ; Opens the hand.
40 HE 3                 ; Define the current position to position 3.
50 PX 2,3               ; Exchanges the coordinates value of position 2 for those of position 3.
60 ED                   ; Ends program.
```

QN (Question Number)

【 Function 】
Reads the program name or the program information.

【 Input Format 】

```
QN    [<program name>]
```

【 Term 】

<Program name>	Specify the robot program name to be read. (Max. 8 characters) Possible letter used:Digit (0–9) Character (A – Z) Impossible letter used: ( ) * + , . / : ; = ? [¥] ' ″ ! @ # Special specification: When you specified only numeric value, the program name is handled for number. Need to enclose program name with ″ ″ in the case of character used.

【 Explanation 】
(1) Outputs the selected program name or the selected program information through the RS–232C port. If you omitted the program name, the selected program name is turned over and if you specified it, the information about the program is turned over.
(2) The output format is ASCII coded as follows;
・The program number format: ″N″ followed by ″Program name″
・The program information format: Used number of steps, used number of positions, used number of counters
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1:E83” AS #1          ; Opens the RS–232C communication file from the personal computer in
                                     BASIC.
20 PRINT #1,” QN”                   ; Transmits the QN command.
30 LINE INPUT #1,A$                 ; Saves the received data to A$.
40 PRINT ″Selected program is ″ : A $ ; Displays the contents of A$ to the screen.
60 ED                               ; Ends.

RUN                                 ; Run the program.
Selected program is  N10            ; Outputs the program name.
```

RC (Repeat Cycle)

【 Function 】
Repeats the loop specified by the NX command the specified number of times.

【 Input Format 】

```
RC  <number of repeated cycles>
```

【 Term 】

<Number of repeated cycles>	Specify the number of times repeating. 1 ≦ number of repeated cycles (decimal) ≦ 32767 & 0001 ≦ number of repeated cycles (hexadecimal) ≦ & 7FFF

【 Explanation 】
(1) Used with the NX command to cause the loop specified by the NX command to be executed the specified number of times and causes the line number following NX to be subsequently executed.
(2) To incorporate another loop (between RC and NX) into the existing loop is called ″nesting″. Up to 9 nesting levels are possible.

【 Sample program 】 (Movemaster command)

```
10 MO 1                  ; Moves to position 1.
20 RC 3                  ; Repeats loop delimited by NX three times
30  MO 2                 ; Moves to position 2.
40  MO 3                 ; Moves to position 3.
50  MO 4                 ; Moves to position 4.
60 NX                    ; Delimits the loop.
70 MO 5                  ; Moves to position 5.
80 ED                    ; Ends program.
```

RN * (Run)

【 Function 】
Executes the specified part of commands in a program.

【 Input Format 】

```
RN   [<start line number>] [, [<end line number>] [, program name]]]
```

【 Term 】

<Start line number>	Specify the line number beginning. 1 ≦ start line number ≦ 32767 (The top line for default)
<End line number>	Specify the line number ending. 1 ≦ end line number ≦ 32767 (The last line or ED command line for default)
<Program name>	Specify the program name. (Max. 8 characters) Possible letter used:Digit (0-9) Character (A - Z) Impossible letter used : ( ) * + , . / : ; = ? [¥] ' " ! @ # Special specification : When you specified only numeric value, the program name is handled for number. Need to enclose program name with " " in the case of character used.

【 Explanation 】
(1) Runs the program starting with the specified starting line and ending with the line one ahead the specified ending line.
(2) If the program is to continue, restart with the ending line.
(3) If the teaching box is connected, the line number being executed is shown on its display.
(4) You can select the program by appointing the <program name>. In this case, the specified program becomes the target program and the program starts from the specified line.
(5) If you describe the RN command with line number in a program, no operation is executed.
(6) When the RN command is executed, the contents of the counter, character string, position, and output signal and hand status remain intact (not initialized).
(7) The following name is identified as the same.
An example:
・Handled as the same. 1,01,001,00000001 (Only numeric value)
・Handled as not the same. 1,1 A, A0001 (Includes alphabetic characters)
(8) The letters that controller can indicate to the LED are 0-9, A ~ Z (simplified).

【 Sample program 】(N88BASIC)

```
10 OPEN" COM1:E83" AS #1     ; Opens the RS-232C communication file from the personal  computer in BASIC.
20 PRINT #1," RN 100, ,2"    ; Executes the program 2 from line 100.
30 ED                        ; Ends program.
```

⚠CAUTION The execution mode of the program is a continuous operation mode.
Does repeat execute from the head line after the final line is executed, when the stop line number is omitted.

◇◆◇ Method of making to one cycle operation ◇◆◇
Push the END switch of controller operation panel or turn on "CYCLE" of a dedicated input signal.Stop the program being executed at the last line or END statement.
One cycle operation is possible by even making the operation format of the parameter:SLT "CYC". In this case, even when the same again program is execute because the program enters the state of the unselection after completing the execution, the program select is needed.

RS ＊(Reset)

【 Function 】
Resets the program and error condition.

【 Input Format 】

```
RS   [<reset number>]
```

【 Term 】

<Reset number>	Specify the contents of reset. Reset contents 0: Cancels error and resets program. (Default) 1: Makes all counters undefined condition. 2: Resets the battery timer. 3: Deletes all programs and all positions. (The same as the NW command.) 4: Resets the origin setting condition.

【 Explanation 】
(1) Resets error condition in error mode, switching servo from OFF to ON and causing the program to return to its beginning.
(2) If any of the axes has exceeded its software limit, the error cannot be reset.
(3) The outputs remain unchanged by resetting any errors.

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1;E83 AS #1 ; Opens the RS-232C communication file from personal computer in BASIC.
20 PRINT #1,” MO 1000”     ; Error occurs because of the wrong value.
30 PRINT #1,” RS”          ; Cancels the error.
40 ED                      ; Ends program.
```

RT (Return)

【 Function 】
Completes a subroutine and returns to the main program.

【 Input Format 】

```
RT   [<line number>]
```

【 Term 】

<Line number>	Specify the line number to jump. 1 ≦ line number ≦ 32767 (If omitted, returns to the next line of the GS command.)

【 Explanation 】
(1) Completes the subroutine called by the ″GS″ command and returns to the main program.
(2) Error occurs if the corresponding ″GS″ command is not specified.
(3) If you specify the line number, the program jumps to the specified line number after returning to the main routine.

【 Sample program 】
See the GS command.

SC (Set Counter)

【 Function 】
A specified value is set in the specified counter or character string.

【 Input Format 】

```
SC  <counter number/character string number>, [<counter set value/character string set value>]
```

【 Term 】

<Counter number>	Specify the number of counter setting. 1 ≦ counter number ≦ 99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99
<Counter set value>	Specify the value of counter setting.(0 for default) −32768 ≦ set value (decimal) ≦ 32767 & 8000 ≦ set value (hexadecimal) ≦ & 7FFF
<Character string set value>	Specify the character string to be set. Usable characters : numerals (0 to 9), alphabetic characters(A to Z) Unusable characters : ″!@# Number of characters : Within 120 characters including line number and SC command.

【 Explanation 】
<When counter number is specified>
(1) All counters are factory−set to zero.
(2) Used to count the number of workpieces and job sequences and to set the number of grid points in the pallet.
(3) The contents of the counter can be changed, compared or read by the relevant command. (Refer to the IC, INP, DC, CP, CR, CL, AN, OR, XO commands.)
(4) The counter set value remains unchanged when the RS 0 or ED command is executed.
(5) The contents of the counter are battery backed after the power is switched off.

<When a character string number is specified>
(1) Enclose the set character string with double quotations (″″).
Example) When setting the character string ABC, set ″ABC″.
(2) If the set character string is omitted, the details of the character string number will be blank. Thus, the details of the character string number can be deleted.
(3) Operation, comparison and reading of the character string are possible with the related commands. (Refer to the CP, CR, CL, EQ, NE, LG, SM, INP commands.)
(4) The value of the set character string will not change even if the RS 0 or ED command is executed. The value will be held by the battery even when the power is turned OFF.

【 Sample program 】(Movemaster command)

```
10 SC   21,10            ; Set value 10 to counter 21.
20 IC   21               ; Add 1 to counter 21.
30 CP   21               ; Set value of counter 21 to the internal register.
40 DR                    ; Outputs the value of the internal register through RS-232C port.
50 SC   $5,″OK″          ; Set character string ″OK″ in character string number 5
60 CP   $5               ; Set details of character string number 5 in the character string register
70 EQ   $10,200          ; Jumps to line 200 if the data equals character string number 10.
```

SD (Speed Define)

【 Function 】
Defines the moving velocity, time constant, acceleration/deceleration time, and continuous path setting.

【 Input Format 】
```
SD   <moving speed> [, <time constant>
        [, <acceleration rate>, <deceleration rate> [, <CNT setting>]]]
```

【 Term 】

<Moving speed>	Set moving speed at linear or circular interpolation. 0.01 ≦ moving speed ≦ 650.00 (mm/sec)
<Time constant>	1 ≦ time constant ≦ 300 (millisecond (Current value when omitted.)) Returns to default value with 0.
<Acceleration rate>	The command time for the acceleration to the maximum speed is set. Unit: % (Current value when omitted.) (Recommended range: 1 to 100)
<Deceleration rate>	The command time for the deceleration to the maximum speed is set. Unit: % (Current value when omitted.) (Recommended range: 1 to 100)
<CNT setting>	Specify the enable or disable state of continuous path mode. 0: Disable1: Enable

【 Explanation 】
(1) The minimum setting unit for the movement speed is 0.01mm/second or 0.01 degrees/second. (Example: For 20.05mm/sec., designate as 20.05.) The minimum setting unit for the time constant and acceleration/deceleration rate is 1 millisecond and 1% respectively. By using this command, the hand end movement speed for linear and circular interpolation can be set in more detail than the SP command. (Refer to the SP command.)
(2) Allows the moving speed (or angular speed) of the tip of hand for linear or circular interpolation to be defined in smaller increments than the SP command.
(3) Setting a large value to the time constant makes the robot operation slower and smoother.
(4) The standard acceleration/deceleration time is determined for each robot being used. Set as a rate in respect to this time.
The system default value is 100, 100. The acceleration rate changed by executing this command is reset to the system default value when the program is reset and when the ED statement is executed.
For the smooth movement when CNT is valid, the orbit path will differ according to the acceleration and movement speed. To carry out smooth movement at a constant speed, set the acceleration and deceleration to the same value. CNT is invalid in the default state.
(5) During linear or circular interpolation, a certain moving speed of the SD command may cause error in excess of the maximum speed of the corresponding joint. In this case, set the speed to a lower value.
(6) The CNT setting is invalid in the default state, so when validated, the acceleration/deceleration between the continuous movement commands will be omitted. (Path movement) However, acceleration/deceleration will be carried out when starting and stopping, and when there is a timer or input signal standby, etc., between the movement commands.
(7) By enabling the CNT setting, the robot moves continuously without acceleration and deceleration until the SD or SP command disables the CNT setting. (Path motion) However, the robot accelerates and decelerates at a starting and at a stopping point as well as when a timer or an input command is executed during the path motion.
(8) The time constant, acceleration/deceleration rate, and CNT setting of this command are valid during joint interpolation movement in addition to during linear and circular interpolation movement.

【 Sample program 】(Movemaster command)
```
10 SP 15          ; Set the moving speed to 15.
20 MS 1           ; Moves to position 1 by linear interpolation.  (SP 15)
30 SD 100         ; Set the moving speed to 100mm/sec.
40 MS 2           ; Moves to position 2 by linear interpolation.  (100 mm/sec)
50 MO 3           ; Moves to position 3 by joint interpolation.   (SP 15)
60 MS 4           ; Moves to position 4 by linear interpolation.  (100 mm/sec)
70 ED             ; Ends program.
```

SF (Shift)

【 Function 】
Adds each coordinate value of position (b) to each coordinate value of position (a) and defines it again as a new position.

【 Input Format 】

```
SF   <position number(a)>,  <position number(b)>
```

【 Term 】

<Position number>	Specify the position number. 1 ≦ position number (a), (b) ≦ 999

【 Explanation 】
(1) The hand open or close state of position (a), as well as the structural flag (R/L, A/B, N/F), is not affected by the SF command.
(2) Error occurs if the position (a) and/or (b) have not been predefined.
(3) Does not effect any robot motion.
(4) The coordinate value of position (b) is not affected by the SF command.

【 Sample program 】(Movemaster command)

```
10 PD 20,0,0,20,0,0,0    ; Set the location and the attitude of position 20.
20 HE 10                 ; Set the current position to position 10.
30 SF 10,20              ; The position 10 is shifted only Z-coordinate 20 mm of position 20.
40 MO 10                 ; Moves to position 10.
50 ED                    ; Ends program.
```

SM (If Smaller)

【 Function 】
This compares the value of the internal register with a specified value. If smaller, the program will jump. The character string register and the numbers of characters in a specified character string are compared. If the character string register is smaller, the program will jump.

【 Input Format 】

```
SM   <compared value/character string number>, <branching line number>
```

【 Term 】

<Compared value>	Specify the value compared with the internal register or counter No. with @. −32768 ≦ Compared value (decimal) ≦ 32767 & 8000 ≦ Compared value (hexadecimal) ≦ & 7FFF @1 ≦ counter No. ≦ @99
<Character string number>	Specify character string number in numerical value which ″$″ is added to the head. $1 ≦ character string number ≦ $99
<Branching line number>	Specify the line number to which the program jumps. 1 ≦ branching line number ≦ 32767

【 Explanation 】
<When counter number is specified>
(1) Causes a jump to occur conditionally in accordance with the external input data or the internal counter value.
(2) If the internal register value is smaller than the compared value (i.e., when the condition is met), the program jumps to the specified line. Otherwise (i.e., when the condition is not met), the program continues in sequence. Error occurs at a jump if the specified line number does not exist.
(3) A value can be loaded into the internal register by executing the input command (See ID) for the external input data or by executing the counter set command (See CP) for the counter data. Accordingly when you carry out conditional branching, need to execute either of the above commands beforehand.
(4) The compared value may be defined either in decimal or hexadecimal. A hexadecimal value must be headed by ″&″.

<When character string number is specified>
(1) The conditions will jump depending on the data input from an external source or the number of characters in a specified character string.
(2) If the number of characters in the character string register is smaller than the number of characters in a specified character string (when the conditions are established), the program will jump to the specified line number. If the number is larger (when conditions are not established), the next line will be executed. If the specified line number is not registered, an error will occur when jumping.
(3) By executing an INP command, the data input from an external device will be set in the character string register. The details of the character string number will be set by executing a CP command. Thus, when executing condition jumping, one of these commands must be executed first.

【 Sample program 】(Movemaster command)

```
10 ID                    ; Fetches the data from the external input port.
20 SM 10,100             ; If the input data is smaller than 10, jumps to line 100.
30 MS 1                  ; Moves to  position 1 by linear interpolation.
40 ED                    ; Ends program.
100 MO 10                ; Moves to position 10.
140 OPN 1,1              ; Opens the RS-232C port.
150 INP 1, ,2            ; Reads the data of character string register from the RS-232C port.
160 SM   $5,200          ; Jumps to line 200 if the data smaller than character string number 10.
   :
200 MO 2                 ; Moves to position 2.
```

SP (Speed)

【 Function 】
Sets the operating speed, acceleration or deceleration time and the continuous path setting.

【 Input Format 】

```
SP   <speed level> [, [<H/L>] [, <CNT setting>]]
```

【 Term 】

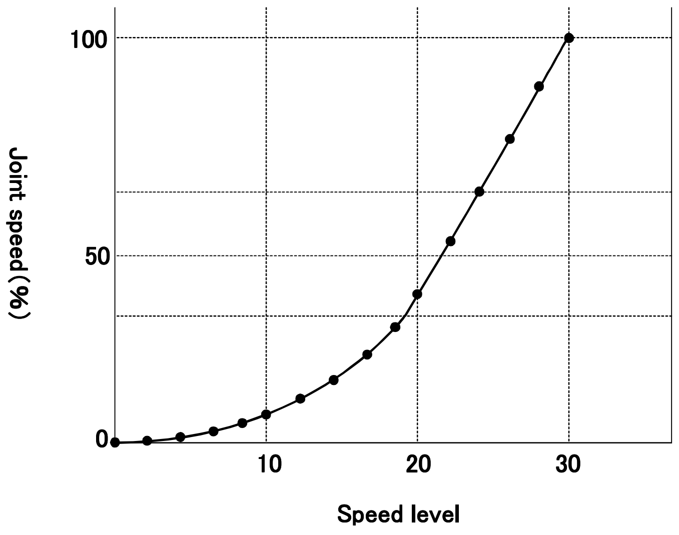
<Speed level>	Set moving speed. 0 ≦ speed level ≦ 30
<H/L>	Set acceleration/deceleration level. (Default is H.) H: High acceleration/deceleration (Max. 100%) L: Low acceleration/deceleration (Max. 50%)
<CNT setting>	Specify the enable/disable state of the continuous path mode. (Default is disable.) 0: Disable 1: Enable

【 Explanation 】
(1) Sets the operating speed in 31 steps and acceleration/deceleration time upon start and stop in 2 levels.
(2) The speed level is predetermined as a ratio to the maximum rpm of each joint for joint interpolation and as a ratio to the maximum speed of the tip of hand for linear interpolation. The initial value is 30.
(3) The acceleration/deceleration time may be selected from among H or L. The acceleration time is the maximum time for the robot to reach to the maximum speed. Therefore, when the moving speed does not reach to the maximum speed, the actual acceleration time becomes smaller than the specified value.
(4) The acceleration/deceleration command time can be designated in the two stages H and L. In either case, the rate is in respect to the maximum acceleration/deceleration speed for joint, linear and circular interpolation, and signify the acceleration during startup. (This also applies for deceleration.) Thus, if the movement speed does not reach the maximum speed, the acceleration/deceleration time will be smaller than the set value.
(5) For linear interpolation, the tip of hand, determined by the tool command, is moved at constant speed. In this case, error may result from any of the joints exceeding its maximum speed. If the motion of the position angle (6-axis type:A,B,C angle, 5-axis type:A,B angle) is greater than the motion of the distance (X, Y, Z), the robot moves in reference to the position angular speed.
The SD command allows the speed to be defined in smaller increments.
(6) CNT is invalid as the default setting.
(7) By enabling the CNT setting, the robot moves continuously without acceleration and deceleration until the SD or SP command disables the CNT setting. (Path motion) However, the robot accelerates and decelerates at a starting and at a stopping point as well as when a timer or a input command is executed during the path motion.
(8) The speed and acceleration/deceleration set once is valid until set again. If H/L is omitted, the previous setting value will continue be vaild. (If the SD command is executed, that setting value will be valid.)
(9) The setting returns to the default value when the ED command is executed or the program is reset.

The relation between the speed level and the moving speed.

SP	Joint interpolation (%)	SP	Joint interpolation (%)
0	0.1	16	19.0
1	0.4	17	22.2
2	0.6	18	25.9
3	0.8	19	29.8
4	1.1	20	34.2
5	1.5	21	40.7
6	2.0	22	47.3
7	2.7	23	53.9
8	3.7	24	60.5
9	4.9	25	67.1
10	6.5	26	73.7
11	8.2	27	80.2
12	9.7	28	86.8
13	11.6	29	93.4
14	13.7	30	100.0
15	16.2		

Notice.) The linear speed is the rate for the maximum linear speed X joint interpolation movement. The speed during linear interpolation movement is the posture angle reference when the posture angle (A, B, C rotation for RV−1A) movement range =B3 distance (X, Y, Z) movement amount.(The angular speed in degree/second is equivalent to the distance speed in mm/second divided by the value 2.12.)



⚠CAUTION CNT is invaild in the default state. If the motion speed, acceleration or deceleration is changed in validate CNT state, the robot motion locus will change, so take care to prevent collisions with the peripheral devices. If the robot might collide with the peripheral devices, invalidate CNT or shorten the acceleration/deceleration time.

【 Sample program 】(Movemaster command)

```
10 SP   8                    ; Sets the moving speed to 8.
20 MO   5                    ; Moves to position 5 by joint interpolation.
30 SP   10                   ; Sets the moving speed to 10.
40 MS   7                    ; Moves to position 7 by linear interpolation.
50 ED                        ; Ends program.
```

STR ＊(Step Read)

【 Function 】
Reads the contents of the specified step number, or the stopping step number. (Using RS-232C)

【 Input Format 】

```
STR   [<step number>]
```

【 Term 】

<Step number>	Specify the step number reading. 0 ≦ line number ≦ 32767

【 Explanation 】
(1) Outputs the contents of the specified step number, or the stopping step number through the RS-232C port.
(2) The output format is the line No. and program contents, and all are output with ASCII codes.
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.
(4) If you specify the undefined step number, the hexadecimal 0D is returned over.
(5) If the step number is not specified or zero is specified, the current stopping line number is read.
(6) In the above case, the command STR allows you to confirm the step number by a personal computer when error occurs

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1:E83” AS #1        ; Opens the RS-232C communication file from the personal computer in BASIC.
20 INPUT ″Reading step is ″ : J $  ; Enters the step number that you want to read.
30 PRINT #1,” STR” +J$           ; Transmits ″STR″ + ″step number″ to the controller.
40 LINE INPUT #1,A$              ; Saves the received data to A$.
50 PRINT A$                      ; Displays the data on the screen.
60 ED                            ; Ends program.

RUN                              ; Run the BASIC program.
Reading step is ? 2              ; Enters the step number.
10 MO 1                          ; Outputs the contents of the step.
```

SUB (Subtract)

【 Function 】
Subtracts the operation data from the value of the internal register, and stores it in the internal register.

【 Input Format 】

```
SUB   <operation data>
```

【 Term 】

<Operation data>	Describes the data to be operated as a numeric value or counter No. with @. −32768 ≦ numeric value (decimal) ≦ 32767 &8000 ≦ numeric value (hexadecimal) ≦ &7FFF @1 ≦ counter No. ≦ @99

【 Explanation 】
(1) Designate the operation data setting as a numeric value or counter No.
When designating with a numeric value, use a decimal or hexadecimal value, When using a hexadecimal, add a ″&″ to the head of the operation data.
When seting with a counter No. add a ″@″ to the head of the counter No.
The contents of the set counter No. will be used as the operation data.
(2) The operation results are stored in the internal register, so operation, comparison and reading, etc., of the operation results can be carried out with the related commands.
(Refer to ADD,MUL,DIV,EQ,NE,LG,SM,CL,DR,AN,OR,XO commands)

【 Sample program 】(Movemaster command)

```
10 CP 1                  ; Stores counter No.1 value in internal register
20 SUB @2                ; Subtracts counter No.2 value from internal register value
30 CL 3                  ; Sets internal register value in counter No.3
                          (Counter No.3 = counter No.1 − counter No.2)
40 CP 1                  ; Stores counter No.1 value in internal register
50 SUB 15                ; Subtracts 15 from internal register value
60 CL 4                  ; Sets internal register value in counter No.4
                          (Counter No.4 = counter No.1 − 15)
```

TB (Test Bit)

【 Function 】
Causes a jump to occur in accordance with the specified bit value in the internal register.

【 Input Format 】
```
TB   [<+/->] <bit number>, <branching line number>
```

【 Term 】

<+/->	Set the condition that compares bit. +: The bit is ON. (Default) -: The bit is OFF.
<Bit number>	Specify the bit number of the internal register. 0 ≦ bit number ≦ 15
<Branching line number>	Specify the line number to which the program jump. 1 ≦ branching line number ≦ 32767

【 Explanation 】
(1) Causes a jump to occur conditionally in accordance with the external input data or the internal counter value.
(2) The program jumps to the specified line number if the specified bit in the internal register is on or off (i.e., when the condition is met). Otherwise (i.e., when the condition is not met), the program continues in sequence.
(3) A value can be loaded into the internal register by executing the input command (see ID) for the external input data or by executing the compare counter command (see CP) for the counter data. It is therefore necessary to execute either of the above commands beforehand so that a conditional jump can occur.
(4) Error occurs at a jump if the specified line number is not predefined

【 Sample program 】(Movemaster command)
```
10 ID                       ; Fetches data from the external input port.
20 TB +1,100                ; If the bit number 1 of input data is ON, then jumps to line number 100.
30 MS 1                     ; Moves to position 1 by linear interpolation.
40 ED                       ; Ends program.
   :
100 MO 10                   ; Moves to position 10.
```

TBD (Test Bit Dirct)

【 Function 】
Causes a jump to occur in accordance with the specified bit value in the external input.

【 Input Format 】

```
TBD    [<+/->] <input bit number>, <branching line number>
```

【 Term 】

<+/->	Set the condition that compares bit. +: The bit is ON (Default) −: The bit is OFF
<Input bit number>	Specify the bit number of general input. 0 ≦ input bit number ≦ 32767
<Branching line number>	Specify the line number to which the program jumps. 1 ≦ branching line number ≦ 32767

【 Explanation 】
(1) Causes a jump to occur conditionally in accordance with the external input data directly.
(2) The program jumps to the specified line number if the specified bit in the external input is on or off (i.e., when the condition is met). Otherwise (i.e., when the condition is not met), the program continues in sequence
(3) It is not necessary to execute the input command (ID) beforehand, and the internal register remains intact after the execution of TBD command.
(4) Error occurs if the specified line number is not predefined.

【 Sample program 】(Movemaster command)

```
10 TBD +19,100          ; If the bit 19 of the external input is ON, then jumps to line number  100.
20 MS 1                 ; Moves to position 1 by linear interpolation.
30 ED                   ; Ends program.
   :
100 MO 10               ; Moves to position 10.
```

TI (Timer)

【 Function 】
Halts the motion for the specified length of time.

【 Input Format 】

```
TI    <timer counter>
```

【 Term 】

<Timer counter>	Set the period of timer. 0 ≦ timer counter ≦ 32767(0.1 sec)

【 Explanation 】
(1) Causes the robot to halt its motion for the specified counter value x 0.1 second. (Max. 3276.7 seconds)
(2) Used to introduce a time delay before and after the hand is opened or closed for gripping a workpiece.

【 Sample program 】(Movemaster command)

```
10 MO 1,O               ; Moves to position 1.
20 TI 5                 ; Wait for 0.5 second.
30 GC                   ; Closes hand.
40 TI 10                ; Wait for 1.0 second.
50 MO 2                 ; Moves to position 2.
60 ED                   ; Ends program.
```

TL (Tool)

【 Function 】
Establishes the distance between the hand mounting surface and the tip of hand.

【 Input Format 】

```
TL   [<tool length>]
```

【 Term 】

<Tool length>	Set the distance from the hand mounting surface to the tip of hand. 0 ≦ tool length ≦ 300.00 (mm) (0 for default)

【 Explanation 】
(1) The least input increment of the tool length is 0.01 mm (e.g., specify 200.05 for 200.05 mm).
The default value is 0mm.
(2) Once established, the tool length remains valid until a new value is set (battery backed when the power is switched off). When the tool length has been changed, the current position is also changed accordingly, which, however, does not involve any robotic motion. (Initial tool length is 123 mm.)
(3) Since the point defined by the TL command is the basis for calculation of the current position, XYZ jogging and commands involving the XYZ coordinates system, the accurate tool length must be established according to the tool being used.
(4) Whenever a program is to be run, the same tool length as that established during teaching must be set at the beginning of the program.
(5) The current value can be read from the RS-232C by executing the WT command.

⚠CAUTION When changing the tool length, accurately input the value so that the setting value is not mistaken.

⚠CAUTION If the original teaching point is moved to afer changing the tool length, the robot movement posture will change. Take special care to prevent interference with the periphery.

⚠CAUTION Even if the tool length is input accurately, the required precision may not be achieved due to the actual dimension precision and the installation posture precision, etc.,.

【 Sample program 】 (Movemaster command)

```
10 TL 120          ; Sets the tool length to 120 mm.
20 HE 1            ; Define the current position to position 1.
30 TL 100          ; Changes the tool length to 100 mm.
40 MO 1            ; Moves to position 1 advancing 20 mm in the tool direction.
50 ED              ; Ends program.
```

TLM (Tool Matrix)

【 Function 】
The tool conversion is carried out.

【 Input Format 】

```
T L M   [ X ] [ , [ Y ] [ , [ Z ] [ , [ A ] [ , [ B ] [ , [ C ] ] ] ] ] ]
```

【 Term 】

<tool length X>	Describe the X axis direction tool length at the tool coordinate system. Unit : mm
<tool length Y>	Describe the Y axis direction tool length at the tool coordinate system. Unit : mm
<tool length Z>	Describe the Z axis direction tool length at the tool coordinate system. Unit : mm
<A rotation>	Describe the X axis rotation angle at the tool coordinate system. Unit : degrees
<B rotation>	Describe the Y axis rotation angle at the tool coordinate system. Unit : degrees
<C rotation>	Describe the Z axis rotation angle at the tool coordinate system. Unit : degrees

【 Explanation 】
(1) The minimum setting unit for the tool length is either 0.01mm or 0.01 degrees.
(2) The set tool length will keep the previous until reset. This value is also held by the battery when the power is turned OFF.
(3) When the tool length is changed, the current position will change accordingly. However, the robot will not operate according to this.
(4) The point on the tool coordinate system operation command will be the control target for the position data operation, cartesian system operation command and XYZ jog operation.
Thus, set an accurate value that matches the tool (hand, etc.) being used.
(5) The current setting value can be read from RS-232C by executing the WTM command.
(6) All axes are valid for 6-axis type. Only Z axis is valid for 5-axis type. The X, Y, Z and C axes are valid for 4-axis type.

⚠CAUTION When changing the tool length, accurately input the value so that the setting value is not mistaken.

⚠CAUTION If the original teaching point is moved to afer changing the tool length, the robot movement posture will change. Take special care to prevent interference with the periphery.

⚠CAUTION Even if the tool length is input accurately, the required precision may not be achieved due to the actual dimension precision and the installation posture precision, etc.,.

【 Sample program 】 (Movemaster command)

```
10 TLM  0,0,100,0,0,0       ; Change tool length to Z = 100mm
20 HE  1                    ; Register current position in position 1
20 TLM  -50,0,100,0,0,0     ; Change tool length to X = -50mm
30 MS 1                     ; Move to position1 (Move to a position +50mm away from the current position in the tool
                              coordinate system's X axis direction.)
```

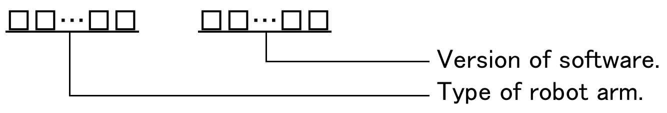
VR (Version Read)

【 Function 】
Reads the software version of the system ROM. (Using RS-232C)

【 Input Format 】

```
VR
```

【 Explanation 】
(1) Outputs the software version of the system ROM mounted in the controller through the RS-232C port.
(2) The output format is ASCII coded. (Example: ″RV-2AJ Ver. D1″)
The denotation of the software version is the following.



(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.

【 Sample program 】 (N88BASIC)

```
10 OPEN” COM1:E83” AS #1           ; Opens the RS-232C communication file from the personal computer in
                                     BASIC.
20 PRINT #1,” VR”                  ; Transmits the VR command.
30 LINE INPUT #1,A$                ; Saves the received data to A$.
50 PRINT ″Software version is ″ : A $; ; Displays the data on the screen.
60 ED                              ; Ends program.

RUN                                ; Run the program.
Software version is  RV-2AJ  Ver. D1 ; Outputs the version name.
```

WH (Where)

【 Function 】
Reads the coordinates of the current position and the open or close state of the hand. (Using RS-232C)

【 Input Format 】

```
WH
```

【 Explanation 】
(1) Causes the coordinates of the current position of the tip of hand, as determined by the tool length (see the TL, TLM command), and the hand open or close state to be output through the RS-232C port.
(2) The output format is ASCII coded as indicated below. The least output increment is 0.01 mm or 0.01 degree. (e.g., 20.01 for 20.01 mm)
Output format:
6-axis type : X, Y, Z coordinate value, A, B, C turning angle, additional axis 1, additional axis 2, R/L, A/B, N/F, O/C
5-axis type : X, Y, Z coordinate value, A, B turning angle,, additional axis 1, additional axis 2, R/L, A/B, O/C
4-axis type : X, Y, Z coordinate value,,, C turning angle, additional axis 1, additional axis 2, R/L, O/C
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.

【 Sample program 】(N88BASIC)
(1) 6-axis type

```
10 OPEN ” COM1:E83” AS #1              ; Opens the RS-232C communication file from the personal computer in
                                         BASIC.
20 PRINT #1,” WH”                      ; Transmits the command ″WH″.
40 LINE INPUT #1,A$                    ; Saves the received data to A$.
50 PRINT ″Current coordinates are ″ : A $ ; Displays the contents of A$ on the screen.
60 ED                                  ; Ends.

RUN                                    ; Run the program.
Current coordinates are  +10.00,       ; Displays the value of the current position.
+380.00,300.00,-70.00,50.00,+40.00, R, A, N, C
```

(2) 5-axis type

```
10 OPEN ” COM1:E83” AS #1              ; Opens the RS-232C communication file from the personal computer in
                                         BASIC.
20 PRINT #1,” WH”                      ; Transmits the command ″WH″.
40 LINE INPUT #1,A$                    ; Saves the received data to A$.
50 PRINT ″Current coordinates are ″ : A $ ; Displays the contents of A$ on the screen.
60 ED                                  ; Ends.

RUN                                    ; Run the program.
Current coordinates are  +10.00,       ; Displays the value of the current position.
+380.00,300.00,+50.00,+40.00, R, A, C
```

(3) 4-axis type

```
10 OPEN ” COM1:E83” AS #1              ; Opens the RS-232C communication file from the personal computer in
                                         BASIC.
20 PRINT #1,” WH”                      ; Transmits the command ″WH″.
40 LINE INPUT #1,A$                    ; Saves the received data to A$.
50 PRINT ″Current coordinates are ″ : A $ ; Displays the contents of A$ on the screen.
60 ED                                  ; Ends.

RUN                                    ; Run the program.
Current coordinates are  +10.00,       ; Displays the value of the current position.
+380.00,300.00,,,+40.00, R, C
```

WT (What Tool)

【 Function 】
Reads the tool length currently being established. (Using RS-232C)

【 Input Format 】

```
WT
```

【 Explanation 】
(1) Causes the tool length currently being established (by the TL command) to be output through the RS-232C port.
(2) The data is output in ASCII coded decimal. The least output increment is 0.01 mm (e.g., 105.07 is displayed for 105.07 mm).
(3) Because the terminator of the output data is carriage return (Hex.0D), it is necessary to handle serial data strings up to hexadecimal 0D in receiving a message by a personal computer. ″LINE INPUT #″ statement is equivalent to this in BASIC.
(4) All robotic motions are based on the established tool length. If a wrong tool length has been defined, the robot may interfere with a surrounding object. When the tool length is unknown, therefore, check the tool length using the WT command before starting the robot.

【 Sample program 】(N88BASIC)

```
10 OPEN ” COM1:E83” AS #1   ; Opens the RS-232C communication file from the personal computer in BASIC.
20 PRINT #1,” WT”           ; Transmits the command ″WT″.
40 LINE INPUT #1,A$         ; Saves the received data to A $.
50 PRINT ” TOOL=” ; A$      ; Displays the contents of A$ on the screen.
60 ED                       ; Ends.

RUN                         ; Run the program.
TOOL=105.7                  ; Outputs the tool length.
```

WTM (What Tool Matrix)

【 Function 】
The currently set tool length (X, Y, Z, A, B, C) is read out using RS-232C.

【 Input Format 】

```
WTM
```

【 Explanation 】
(1) This command outputs the current tool length set with the TLM command (refer to the TLM command) from the RS-232C port.
(2) The output format is a decimal ASCII format. The minimum output unit is 0.01mm.
(Example : When the data is 105.7mm, 105.07 will display.)
(3) The data end code (terminator) is the carriage return (CR : hexadecimal OD), so when receiving the data with a personal computer, etc., the series of data up to the hexadecimal OD must be handled as data train. With the general BASIC language, ″LINE INPUT#″ is equivalent to this.
(4) The robot movement is all carried out according to the tool length set at that time. If an incorrect tool length is set, the robot could interfere with oeripheral objects. Thus, if the tool length is unknown, confirm the tool lemgth with this command before operating the robot.

【 Sample program 】(N88BASIC)

```
10 OPEN ″COM1:E83″ AS #1 ; Open RS-232C with personal computer BASIC.
20 PRINT #1,″WTM″        ; Transmit command ″WTM″.
40 LINE INPUT #1,A$      ; Store received data in A$.
50 PRINT ″TLM=″ ;A$      ; Output contents of A$ on screen.
60 ED                    ;  End

RUN                      ; Execute program.
TLM=50.00,30.00,100.00,0,0,0 ; Output (display) tool lemgth.
```

XO (Exclusive Or)

【 Function 】
EXCLUSIVE ORs the specified data and the internal register data.

【 Input Format 】

```
XO   <operation data>
```

【 Term 】

<Operation data>	Specify the data to be operated or counter No. with @. −32768 ≦ operation data (decimal) ≦ 32767 & 8000 ≦ operation data (hexadecimal) ≦ & 7FFF @1 ≦ counta number ≦ @99

【 Explanation 】
(1) Specify the data to be operated in decimal or hexadecimal. Any hexadecimal value must be headed by ″&″.
(2) The operation result is stored into the internal register and can be changed, compared or read by relevant commands. (See the EQ, NE, LG, SM, CL, DR, AN, OR commands)
(3) Execution of the XO command after the input command (ID) allows the required bits of the input data fetched from the external device to be flipped to their opposite settings.

【 Sample program 】 (Movemaster command)

```
10 ID                  ; Fetches data from the external input port.
20 AN &000F            ; Fetches only lower 4 bits.
30 XO &000F            ; Flips data of 4 lower bits to their opposite settings.
40 CL 21               ; Sets above data to counter 21.
50 EQ 10,200           ; If the above data equals 10, then jumps to line 200.
60 ED                  ; Ends program.
   :
200 MO 99              ; Moves to position 99.
```

’ (Comment)

【 Function 】
Allows the programmer to write a comment.

【 Input Format 】

```
’  [<string consisting of up to 120 alphanumeric characters  including line number and ’ (apostrophe)>]
```

【 Explanation 】
(1) You can describe up to 120 alphanumeric characters including line number and ’ (apostrophe).
(2) A name or date can be attached to the created program, and an explanation can be attached to the end of the subroutine identification or command.
(3) The system ignores comments as it processes its commands.
(4) If the number of characters exceeds the limit, the whole excess is ignored.

【 Sample program 】 (Movemaster command)

```
10 ’*****************************   ;
20 ’ Sample Program                 ;
30 ’      Date:2000-04-05           ;
40 ’ Programmed by Arai Takasi      ; Specify the contents of program, the date of implementation, and thename
                                      of programmer, etc.
50 ’*****************************   ;
60 MO 1                             ; Move to standby point
```

3 Command List

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Program	Note	Page
■ Position motion control commands						
1	Change figure	For 6-axis robot CF a[,[R/L] [,[A/B] [,[N/F]]]] For 5-axis robot CF a[,[R/L][,[A/B]]] For 4-axis robot CF a[,[R/L]]	Changes the pose of the robot at position (a).	○	1 ≦ a ≦ 999 @1 ≦ a ≦ @99 (Designate indirect position No.) R : Right, L : Left. A : Above, B : Below. N : Non flip, F : Flip.	2-11
2	Draw joint	For 6-axis robot DJ a,b	Moves the joint (a) by the specifed amount (b). (Unit:Degree)	○	1 ≦ a ≦ 8 Joint interpolation.	2-17
3	Decrement position	DP	Moves to the previous position in number from current position.	○	joint interpolation.	2-18
4	Draw straight	DS [x],[y],[z]	Moves by the specified distance from current position.	○	linear interpolation.	2-20
5	Draw	DW [x],[y],[z]	Moves by the specified distance from current position.	○	joint interpolation.	2-21
6	Here	HE a	Memorizes current position as the position number (a).	○	0 ≦ a ≦ 999 @1 ≦ a ≦ @99 (Designate indirect position No.) (In the case of 0, the robot memorizes the data as the user defined origin.)	2-33
7	Home	HO [a]	Memorizes current position as the origin.	○	a=0: Mechanical stopper method a=1: Calibration jig method a=2: User defined origine method	2-34
8	Increment position	IP	Moves to the next position in number from current position.	○	joint interpolation.	2-37
9	Move approach	MA a1,a2[,[O/C]]	Moves to a position away from position (a1) by the distance of position (a2).	○	1 ≦ a1,a2 ≦ 999 @1 ≦ a1,a2 ≦ @99 (Designate indirect position No.) O : opening a hand. C : closing a hand.	2-40
10	Move continuous	MC a1,a2[,[O/C]]	Moves from the position (a1) to the position (a2) with linear interpolation.	○	1 ≦ a1,a2 ≦ 999 | a1-a2 | ≦ 99 @1 ≦ a1,a2 ≦ @99 (Designate indirect position No.) O : opening a hand. C : closing a hand.	2-41
11	Move joint	For 6-axis robot MJ [J1],[J2],[J3],[J4],[J5],[J6] [J7],[J8] For 5-axis robot MJ [J1],[J2],[J3],[J5],[J6][J7],[J8] For 4-axis robot MJ [J1],[J2],[J3],[J4],[J7],[J8]	Turns the joint by the specified angle from the current position. (Unit:Degree)	○	joint interpolation.	2-42

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
12	Move	MO a [,[O/C]]	Moves to the position (a) .	○	1 ≦ a ≦ 999 @1 ≦ a ≦ @99 (Designate indirect position No.) O : opening a hand. C : closing a hand.	2-43
13	Move position	For 6-axis robot MP [X],[[Y][,[Z][,[A][,[B][,[C][,[L1][,[L2]]]]]]]][,[<R/L>][,[<A/B>][,[<N/F>]]]]	Moves to the position (X,Y,Z,A,B,C,L1,L2).	○	Joint interpolation. R : Right, L : Left. A : A bove, B : Below. N : Non Flip, F : Flip.	2-44
		For 5-axis robot MP [X],[[Y][,[Z][,[A][,[B][,,[L1][,[L2]]]]]]][,[<R/L>][,[<A/B>]]]	Moves to the position (X,Y,Z,A,B,L1,L2).			
		For 4-axis robot MP [X],[[Y][,[Z][,,,[C][,[L1][,[L2]]]]]][,[<R/L>]]	Moves to the position (X,Y,Z,C,L1,L2).			
14	Move R	MR a1,a2,a3 [,[O/C]]	Moves on the arc that position (a1)(a2)(a3) determine with circular interpolation.	○	1 ≦ a1,a2,a3 ≦ 999 @ 1 ≦ a1,a2,a3 ≦@ 99 (Designate indirect position No.) O :Opening a hand C :Closing a hand	2-45
15	Move RA	MRA a [,[O/C]]	The robot moves on the arc that the previous and the next MRA determine with circular interpola-tion.	○	1 ≦ a ≦ 999 @ 1 ≦ a ≦@ 99 (Designate indirect position No.) O :Opening a hand C :Closing a hand	2-47
16	Move straight	MS a [,[O/C]]	Moves to the position (a) with linear interpolation.	○	1 ≦ a ≦ 999 @ 1 ≦ a ≦@ 99 (Designate indirect position No.) O :Opening a hand C :Closing a hand	2-48
17	Move tool	MT a,[b][,[O/C]]	Moves to a position away from position (a) by the distance (b) in the tool direction.(Unit:mm) Joint interpolation.	○	1 ≦ a ≦ 999 @ 1 ≦ a ≦@ 99 (Designate indirect position No.) O :Opening a hand C :Closing a hand	2-49
18	Move tool straight	MTS a,[b][,[O/C]]	Moves to a position away from position (a) by the distance (b) in the tool direction.(Unit:mm) Linear interpolation.	○	1 ≦ a ≦ 999 @ 1 ≦ a ≦@ 99 (Designate indirect position No.) O :Opening a hand C :Closing a hand	2-50
19	Nest	NT	Moves to the user specified origin position.	○		2-54
20	Origin	OG	Moves to the user specified origin position. Joint interpolation.	○	joint interpolation.	2-57
21	Override	OVR a	Establishes program over-ride. (Unit:mm)	○	1 ≦ a ≦ 100	2-60
22	Pllet hemp inn.	PA i,j,k	It establishes length direction of pallet of number appointing and lat-tice point number of width direction.	○	1 ≦ i ≦ 9 1 ≦ j,k ≦ 32767	2-61

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
23	Position clear	PC a1[,a2]]	Clears position data from (a1) to (a2).	×	a1 ≦ a2 @1 ≦ a1,a2 ≦@ 99 (Designate indirect position No.)	2-61
24	Position define	For 6-axis robot PD a,[X],[[Y][,[Z] [,[A][,[B][,[C] [,[L1][,[L2]]]]]]]] [,[<R/L>][,[<A/B>] [,[<N/F>]]]][,<O/C>]	It substitutes (X,Y,Z,A,B,C,L1,L2) to position (a).	○	1 ≦ a ≦ 999 @ 1 ≦ a ≦@ 99 (Designate indirect position No.) R:Right, L:Left A:Above, B:Below N:Nonflip, F:Flip O:hand open, C:hand close	2-62
		For 5-axis robot PD a,[X],[[Y][,[Z] [,[A][,[B][,,[L1] [,[L2]]]]]]][,[<R/L>] [,[<A/B>]]]][,<O/C>]	It substitutes (X,Y,Z,A,B,L1,L2) to position (a).			
		For 4-axis robot PD a,[X],[[Y][,[Z] [,,,[C][,[L1][,[L2]]]]]] [,[<R/L>]][,<O/C>]	It substitutes (X,Y,Z,C,L1,L2) to position (a).			
25	Pallet	PT a	Calculates the coordinates of a grid point on the pallet (a) and sets the coordinates value to the position (a).	○	1 ≦ a ≦ 99	2-67
26	Pulse wait	PW a	It waits for in position till all axis ring inward pulse (a) appointing.	○	1 ≦ a ≦ 10000	2-68
27	Speed define.	SD a [,b[,c,d[,e]]]	It defines speed (a), time const (b), acceleration rate (c), deceleration rate (d) and CNT setting (e) in linear interpolation and arc interpolation.	○	0.1 ≦ a ≦ 650.0 1 ≦ b ≦ 300 0 ≦ c,b ≦ 100 E = 0(disable),1(enable)	2-75
28	Speed	SP a [,[H/L] [,b]]	It defines speed (a), acceleration/ deceleration, CONT setting (b).	○	0 ≦ a ≦ 30 H:Quick motion, L:Smooth motion. b = 0(disable),1(enable)	2-78
29	Timer	TI a	Only clock time (a) appointing stops motion. (0.1 sec unit of measure)	○	0 ≦ a ≦ 32767	2-83
30	Tool	TL [a]	It establishes die length (a) to hand nose from hand installation department.	○	Unit of measure is mm. 0 ≦ a ≦ 300.0	2-84
31	Tool matrix	TLM X,Y,Z,A,B,C	The tool is replaced. All axes are valid for 6-axis type. Only Z axis is valid for 5-axis type. The X, Y, Z and C axes are valid for 4-axis type.	○	Unit of measure is mm,degree.	2-85
■ Program control commands						
32	Counter Load	CL a	Loads internal register value/ strings to the specified counter (a).	○	1 ≦ a ≦ 99 (Counter No.) @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.)	2-12

〇：Usable, ×：Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
33	Compare counter	CP a	Loads value/strings in counter (a) to the internal register.	〇	1 ≦ a ≦ 99(Counter No.) @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.)	2-13
34	Disable Act	DA a	Disables interrupt by the bit (a) of external input signal.	〇	0 ≦ a ≦ 32767	2-15
35	Decrement Line	DL [a1][,[a2] [,[b1][,[b2]]]]	Deletes program lines from (a1) to (a2),or steps from (b1) to (b2).	〇	a1 ≦ a2 b1 ≦ b2 1 ≦ a1,a2,b1,b2 ≦ 32767	2-18
36	Enable Act	EA [+/−] a,b[,[c]]	Enables interrupt by the bit (a) of external input signal, specifies line number (b) to which the program jumps when interrupt occurs, and specifies jump mode (c).	〇	＋:Bit on, ー:Bit off 0 ≦ a ≦ 32767 1 ≦ b ≦ 32767 c = 0 or 1 0:Jump, 1:Call subroutine	2-22
37	End	ED	Ends the program.	〇		2-24
38	If Equal	EQ a, b	Causes a jump to line number (b) if internal register value/strings equals (a) .	〇	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.) 1 ≦ b ≦ 32767	2-25
39	Go Sub	GS [a][,[b]]	Call subroutine which starts from line number (a) of program (b).	〇	1 ≦ a ≦ 32767	2-31
40	Go To	GT a	Causes a jump to line number (a).	〇	1 ≦ a ≦ 32767	2-32
41	Halt	HLT	Halts the program.	〇		2-33
42	If Larger	LG a ,c	Causes a jump to line number (c)if internal register value/strings is greater than (a) .	〇	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.) 1 ≦ b ≦ 32767	2-38
43	If Not Equal	NE a,c	Causes a jump to line number (c)if internal register value/strings does not equal (a) .	〇	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.) 1 ≦ b ≦ 32767	2-53
44	New	NW	Deletes all lines and positions of the selected program.	×		2-54
45	Next	NX	Specifies the range of a loop in a program executed by command RC.	〇		2-54

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
46	Repeat Cycle	RC a	Repeats the loop specified by command NX (a) times.	○	1 ≦ a ≦ 32767	2-71
47	Run	RN [a1][,a2[,b]]]	Executes program (b) from line (a1) to (a2). (a2 not included)	○	1 ≦ a1,a2 ≦ 32767	2-72
48	Return	RT [a]	Completes subroutine activated by command GS and returns to main program.	○	1 ≦ a ≦ 32767 a is the destination.	2-73
49	If Smaller	SM a, c	Causes a jump to line number (c) if the internal register value is smaller than (a) .	○	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.) 1 ≦ b ≦ 32767	2-77
■ Hand control commands						
50	Grip Close	GC[a]	Closes hand grip.	○	a= 0: The 1st hand. 1: The 2nd hand. 0(Hand 1) ≦ a ≦ 7(Hand 8)	2-27
51	Grip Flag	GF a	Defines the open/close state of hand grip used in conjunction with command PD.	○	a= 0: Open 1: Close	2-28
52	Grip Open	GO[a]	Opens hand grip.	○	a= 0: The 1st hand. 1: The 2nd hand. 0(Hand 1) ≦ a ≦ 7(Hand 8)	2-29
53	Grip Pressure	GP a1, a2, a3	Sets the gripping force (a1) and time (a3) and gripping force retention (a2) for the motorized hand.	○	0 ≦ a1,a2 ≦ 63 0 ≦ a3 ≦ 99 (0.1s unit)	2-30
■ I/O control commands						
54	Input Direct	ID [a]	Gets signal from external input without condition.	○	0 ≦ a ≦ 32767	2-35
55	Output Bit	OB [+/−] a	Sets the output state of bit (a) of external output signal.	○	0 ≦ a ≦ 32767 +: Bit on, −:Bit off.	2-55
56	Output Counter	OC a [,[a1][,[a2]]]	Outputs the counter value (a) to external output signal without condition. (a1: starting bit, a2: bit length)	○	1 ≦ a ≦ 99 @1 ≦ a ≦ @99 (Designate indirect counter No.) 0 ≦ a1 ≦ 32767 1 ≦ a2 ≦ 16	2-56
57	Output Direct	OD a (or &b) [,[a1][,[a2]]]	Outputs data (a) to external output signal without condition. (a1: starting bit, a2: bit length)	○	−32768 ≦ a ≦ +32767 &8000 ≦ a ≦ &7FFF 0 ≦ a1 ≦ 32767 1 ≦ a 2 ≦ 16	2-57
58	Test Bit	TB [+/−] a,b	Causes a jump to line number (b) in accordance with the state of bit (a) of internal register value.	○	0 ≦ a ≦ 15 +: Bit on, −:Bit off. 1 ≦ b ≦ 32767	2-82
59	Test Bit Direct	TBD [+/−] a,b	Causes a jump to line number (b) in accordance with the state of bit (a) of external input signal.	○	+: Bit on, −:Bit off. 0 ≦ a ≦ 32767 1 ≦ b ≦ 32767	2-83
■ Operation, subsiture, exchange commands						

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Program	Note	Page
60	Add	ADD a	The operation value (a) is added to the internal register value.	○	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-9
61	Sub	SUB a	The operation value (a) is subtracted from the internal register value.	○	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-81
62	Mull	MUL a	The operation value (a) is multiplied with the internal register value.	○	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-51
63	Div	DIV a	The internal register value is divided by the operation value (a).	○	−32768 ≦ a ≦ 32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-16
64	Increment count	IC a	The counter value (a) is incremented by 1.	○	1 ≦ a ≦ 99 @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-34
65	Decrement count	DC a	The counter value (a) is decremeted by 1.	○	1 ≦ a ≦ 99 @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-15
66	AND	AN a	The logical AND of the internal register and value (a) is obtained.	○	−32768 ≦ a ≦ +32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-10
67	OR	OR a	The logical OR of the internal register and value (a) is obtained.	○	−32768 ≦ a ≦ +32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-59
68	Exclusive OR	XO a	The exclusive OR of the internal register and value (a) is obtained.	○	−32768 ≦ a ≦ +32767 &8000 ≦ a ≦ &7FFF @1 ≦ a ≦ @99 (Designate indirect counter No.)	2-89
69	Set Counter	SC a,[b]	The value (b) or character string (b) is set in the counter (a).	○	1 ≦ a ≦ 99((Counter No.) $1 ≦ a ≦ $99(Character string No) −32768 ≦ b ≦ 32767 &8000 ≦ b ≦ &7FFF b:Character string Within 120 characters incliding line NO. and "SC".	2-74
70	Position load	PL a1,a2	The coordinate value of position No. (a2) is substituted for the position No. (a1).	○	1 ≦ a1,a2 ≦ 999 @1 ≦ a1,a2 ≦ @99 (Designate indirect position No.)	2-63
71	Shift	SF a1,a2	The coordinate value of position No. (a1) is shifted by the coordinate value of position No. (a2), and is re-registered.	○	1 ≦ a1,a2 ≦ 999 @1 ≦ a1,a2 ≦ @99 (Designate indirect position No.)	2-76

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
72	Position exchange	PX a1,a2	The coordinate values of the position No. (a1) and position No. (a2) are exchanged.	○	1 ≦ a1,a2 ≦ 999 @1 ≦ a1,a2 ≦ @99 (Designate indirect position No.)	2-69
■ RS-232C communication commands						
73	Counter reed	CR a	Reads contents of counter (a).	○	1 ≦ a ≦ 99 (Counter No.) @1 ≦ a ≦ @99 (Designate indirect counter No.) $1 ≦ a ≦ $99 (character string No.)	2-14
74	Data reed	DR [a]	Reads the values of the internal register, hand check state, and general output state. (a):Output bit No.	○	0 ≦ a ≦ 32767	2-19
75	Error reed	ER [a]	Reads error number. (a):Not specify Current error no read (a):Specify Error history read	×	1 ≦ a ≦ 128	2-26
76	Line reed	LR [a]	Reads program of line number (a).	×	0 ≦ a ≦ 32767	2-39
77	Position reed	PR [a]	Reads each coordinate value of position number (a).	○	0 ≦ a ≦ 999 @1 ≦ a ≦ @99 (Designate indirect position No.)	2-64
78	Question Number	QN [a]	Reads information of program number (a) or selected program number.	○	a is program name. Number of steps, number of position, number of counter are turned over program information.	2-70
79	Step Reed	STR [a]	Reads program of step number (a).	×	0 ≦ a ≦ 32767	2-80
80	Version Reed	VR	Reads version name of system ROM.	○		2-86
81	Where	WH	Reads each coordinate value of current position.	○		2-87
82	What Tool	WT	Reads current a tool length.	○		2-88
83	What Tool Matrix	WTM	Reads out current tool matrix. (X,Y,Z,A,B,C).	○		2-88
■ Other commands						
84	Input	INP a,[b],[,[c]]	Reads the value of counter (b) or position (b) or character strings (b) from channel (a).	○	0 ≦ a ≦ 7 1 ≦ b ≦ 99 1 ≦ b ≦ 999 $1 ≦ b ≦ $99 C=0 : Counter, 1 : Position, 2 : Character string No.	2-36
85	Number	N a	Selects program (a).	×	a:Program name.	2-52
86	Open	OPN a,b	Opens I/O (b) for channel (a).	○	0 ≦ a ≦ 7 b = 1 : Standard RS-232C	2-58

○ : Usable, × : Not usable

NO	Name	Input Format	Function	Pro-gram	Note	Page
87	Print	PRN a,b or c	Sends out the contents of counter (a) or position (b) or character string (c) from a personal computer in conjunction with commnand INP.	×	a:Counter value −32768 ≦ a ≦ +32767 &8000 ≦ a ≦ &7FFF b:Coordinates value For 6−axis robot (X, Y, Z, A, B,C,R/L, A/B,N/F,O/C) For 5−axis robot (X,Y,Z,A,B,R/L,A/B, O/C) c:Character string Within 120 characters incliding line NO. and ″SC″.	2-66
88	Reset	RS [a]	Resets the state of specified by (a).	×	a =0:Resets an error and program line number. 1: Resets all counter data. 2: Resets battery timer. 3: Equal NW command. 4: Resets the origin setting.	2-73
89	Comment	'	Describes a comment	○		2-89

4 Appendix

Appendix 1 : RS-232C signal conductor's timing chart (DTR control)

The standards for the RS-232C interface are essentially based on such items as electrical specifications, connector types, and pin numbers. The way of using various signal lines and communications protocols are so varied depending on the type of equipment. Accordingly, even though you have made the correct signal lines connected to the personal computer, you still may not be able to operate with it. Accordingly, in regards to making this connection, you should develop a full understanding of the functions of the signal lines that you will use between the controller and the personal computer before you make the connections. Please note that all data transmission between the controller and personal computer will be done in ASCII code.

(1) The timing of data transmissions from the personal computer to the robot.

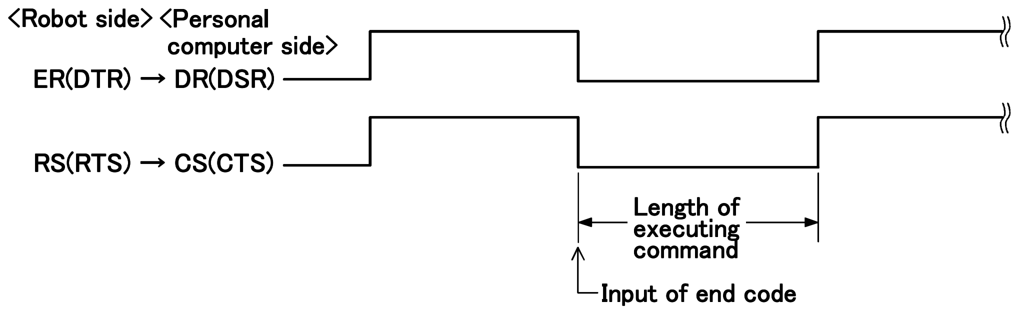
■ Robot side

ER (DTR), and RS (RTS) should both be at the "H" level and be waiting for data input. When the end code is input, ER (DTR) and RS (RTS) both change to level L, and execute the command received.Once the command has been executed, both ER (DTR) and RS (RTS) return to level H.

The end code is hexadecimal "OD" (CR:carriage return), and or hexadecimal "OD" + "OA" (LF: line feed).

■ Personal computer side

When DR (DSR) is at level H, you can transmit characters. When DR (DSR) is at level L, if you transmit characters, an error will result on the robot side.



<Robot side> <Personal computer side>
ER(DTR) → DR(DSR)
RS(RTS) → CS(CTS)
Length of executing command
Input of end code

Fig.4-1 : Timing of data transmissions (from personal computer to robot)

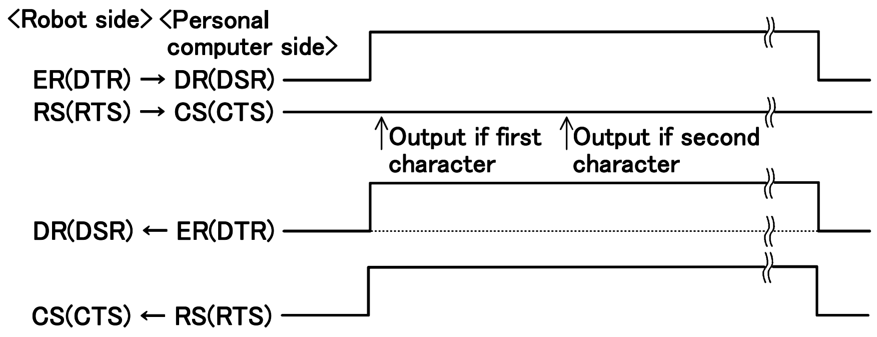
(2) The timing of data transmissions from the robot to the personal computer

■ Personal computer side

The request signal is sent after the ER(DTR) and RS (RTS) are both at level H, and the personal computer waits for a transmission from the robot side.

■ Robot side

After changing ER (DTR) to level H, you can begin to send data transmissions, and when the end code is transmitted, ER (DTR)'s level is changed back to level L.



<Robot side> <Personal computer side>
ER(DTR) → DR(DSR)
RS(RTS) → CS(CTS)
Output if first character
Output if second character
DR(DSR) ← ER(DTR)
CS(CTS) ← RS(RTS)

Fig.4-2 : Timing of data transmissions(from robot to personal computer)

◇◆◇ Supplementation ◇◆◇

１． When the personal computer transmits data to the robot, depending on the personal computer, the personal computer may ignore the ON/OFF signal line for the CS or DR, and transmit data continuously, resulting in an error on the robot side. In this case, make an adjustment on the personal computer, such as by using a timer to send data.

２． When the RS-232C read command is used repeatedly at the program on the robot side, if the processing speed of the personal computer is slow, a communications error can result on the side of the personal computer. (Line buffer overflow error). In this case, use a timer or some other technique so that you can expand the execution interval of the RS-232C read commands.

３. The robot cannot receive new commands when it is executing direct commands, such as when it moves by the MO command. Transmit a new command only after it finishes a command completely.
(By sending an ER command, you can confirm the existence of an alarm and the fact that a transmission has been completed.)

４． If the wrong command is transmitted by the RS-232C while the program is in progress, an error will be generated. In this case, reset the robot.

Appendix 2 : Sample program

This section explains how to make a program with Movemaster command method.

Table 4-1 : Shows effective programming procedure.

NO	Item	Operation description
1	Work plan	1) Assume main flow of work. 2) Define the work of the robot.
2	Drawing of flow- chart	1) Divide the whole work into separate jobs. 2) If there are conditional branchings in the work, divide the work into different programs or divide one program into different blocks. 3) Appoint position number at each work position. 4) Assign input/output signals. 5) Based on the above procedure, draw flowchart.
3	Programming	Based on the above flowchart, make the program.

(1) Pick and place operation

1) Work description

The robot transfers a workpiece from one place to another place.

2) Defined position

Position No.	Position description	Teaching way
Position 1	Picking position	By actual teaching
Position 1	Placing position	
Position 10	Movement distance above position 1	By numerical value setting
Position 20	Movement distance above position 2	

3) Operation flow

Fig. 4-3 shows the flow of operation.

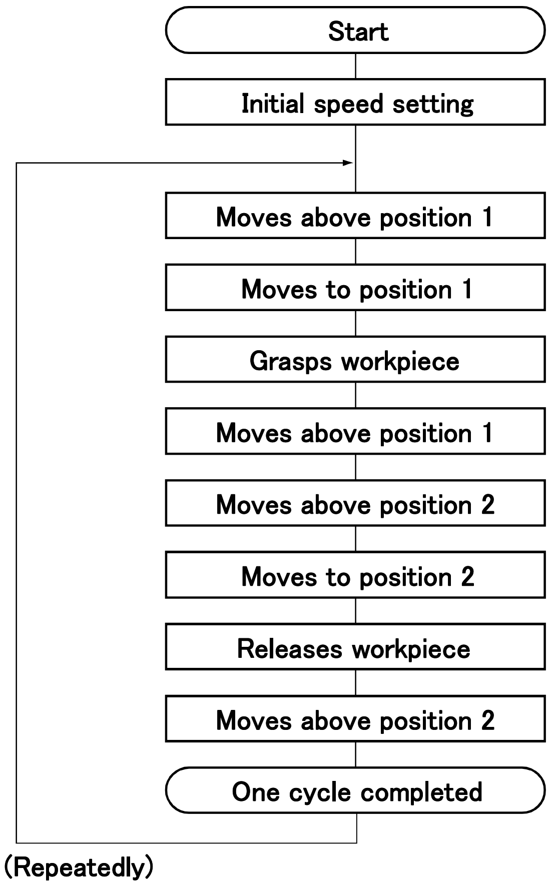


Fig.4-3 : Flowchart

4) Schematic diagram

Fig. 4–4 shows general description of operation.

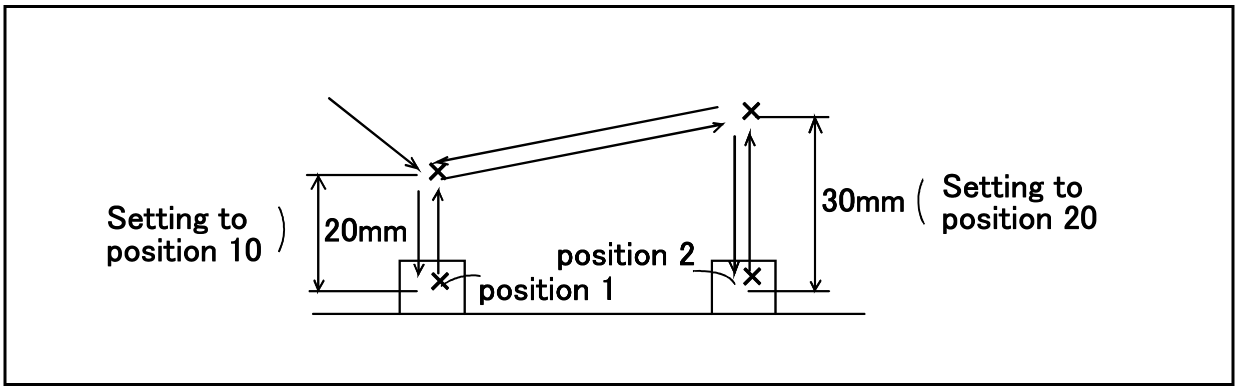


Fig.4–4 : Pick and place operation

5) Example program

```
10   PD 10,0,0,20,0,0,0 Note1)   ; Defines the aerial distance of travel from position 1 (Z=20 mm) .
20   PD 20,0,0,30,0,0,0 Note2)   ; Defines the aerial distance of travel from position 2 (Z=30 mm) .
30   SP 17                       ; Sets the initial speed.
40   MA 1,10,O                   ; Opens hand and moves 20 mm above the position 1.
50   MO 1,O                      ; Moves to the position 1.
60   GC                          ; Closes hand and grasps the workpiece.
70   MA 1,10,C                   ; Moves 10 mm above the position 1 with hand closed.
80   MA 2,20,C                   ; Moves 30 mm above the position 2 with hand closed.
90   MO 2,C                      ; Puts the workpiece.
100  GO                          ; Opens hand and releases the workpiece.
110  MA 2,20,O                   ; Moves 30 mm above the position 2 with hand opened.
120  GT 40                       ; Repeats this program. (Jumps to the line 40.)
```

Note1) For 5–axis robot this is 10 PD 10, 0, 0, 20, 0, 0. For 4–axis robot this is 10 PD 10, 0, 0, 20, 0.
Note2) For 5–axis robot this is 20 PD 20, 0, 0, 30, 0, 0. For 4–axis robot this is 10 PD 10, 0, 0, 30, 0.

(2) Application of interrupt motion

1) Work description

The robot grasps the workpieces that have different height using a limit switch fixed inside the hand. The robot gets a signal of the limit switch through hand check I/O.

2) Defined position

Position No.	Position description	Teaching way
Position 1	Position above work	By actual teaching

3) Signal I/O

I/O	Description	Bit
Input	Work presence signal	Bit 900

4) Operation flow

Fig. 4-5 shows the flow of operation.

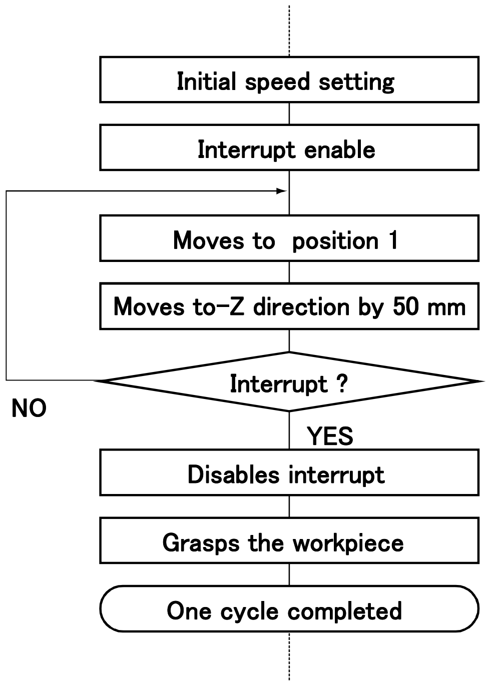


Fig.4-5 : Flowchart

5) Schematic diagram

Fig. 4-6 shows brief description of motion.

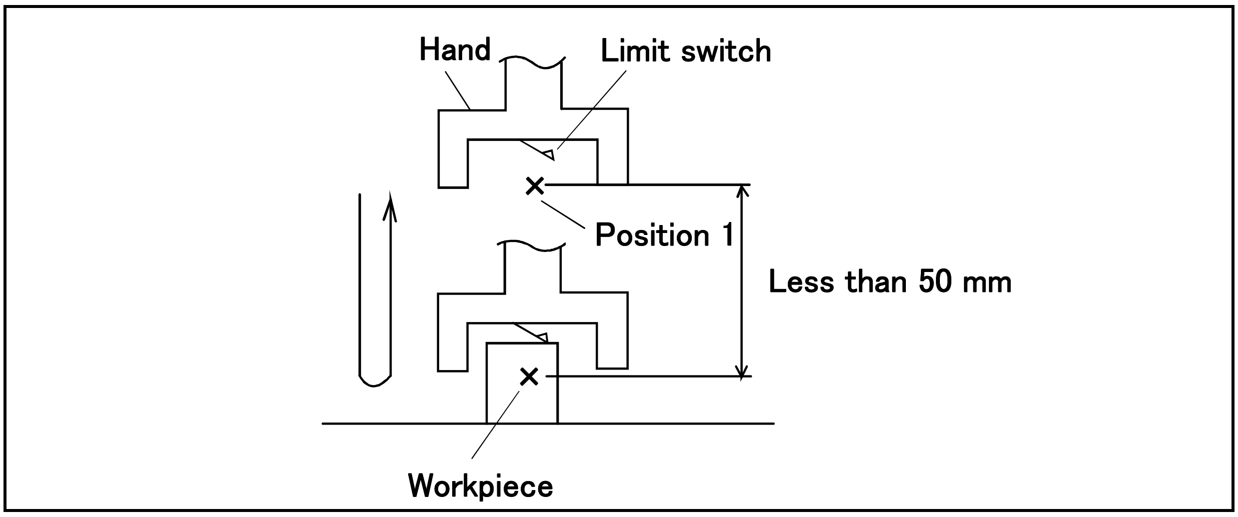


Fig.4-6 : Interrupt motion

6) Example program

```
    :
90 SP 20           ; Sets speed 20.
100EA+900,140      ; Enables interrupt of bit 900.
110MO 1,O          ; Moves above a workpiece.
120DS 0,0,-50      ; Moves 50 mm in the -Z direction. (Linear interpolation)
130GT 110          ;Jumps to the line 110 to return to the position 1 as no workpiece has been detected.
140DA 900          ; Disables interrupt of bit 900.
150GC              ; Closes hand and it grasps the workpiece.
160MO 1,C          ; Moves to position 1 with hand closed.
  :
```

7) Explanation

In this example, the robot moves 50 mm in the -Z direction in line 120. If there is a workpiece, the limit switch signal is input and the robot stopped . Then, the robot jumps to line 140, grasps the workpiece after disabling interrupt and moves to position 1. If there is no workpiece, the robot jumps from line 130 to line 110 returning to position 1. Thus, the robot repeats the same operation again.

(3) Application of palletizing

1) Work description

The robot picks up a workpiece from a feeding pallet and places it on an inspection equipment. After inspection, the robot picks up and places it in another pallet. This program assumes that the shapes of the two pallets are different.

2) Defined position

Position No.	Position description	Teaching way
Position 1	Palette 1 setting position	Defined by PT command
Position 2	Palette 2 setting position	
Position 10	Palette 1 reference position	By actual teaching
Position 11	Palette 1 column terminating position	
Position 12	Palette 1 row terminating position	
Position 13	Palette 1 corner position opposite to reference	
Position 20	Palette 2 reference position	
Position 21	Palette 2 column terminating position	
Position 22	Palette 2 row terminating position	
Position 23	Palette 2 corner position opposite to reference	
Position 30	Test equipment set position	
Position 50	Distance of travel from pallets	

3) Defined counter

Counter No.	Description
Counter 11	Palette 1 column counter
Counter 12	Palette 1 row counter
Counter 21	Palette 2 column counter
Counter 22	Palette 2 row counter

4) Defined input signal

I/O	Description	Bit
Input	Test completion signal	Bit 7

5) Operation flow

Fig. 4-7 and Fig. 4-8 shows the flow of operation.

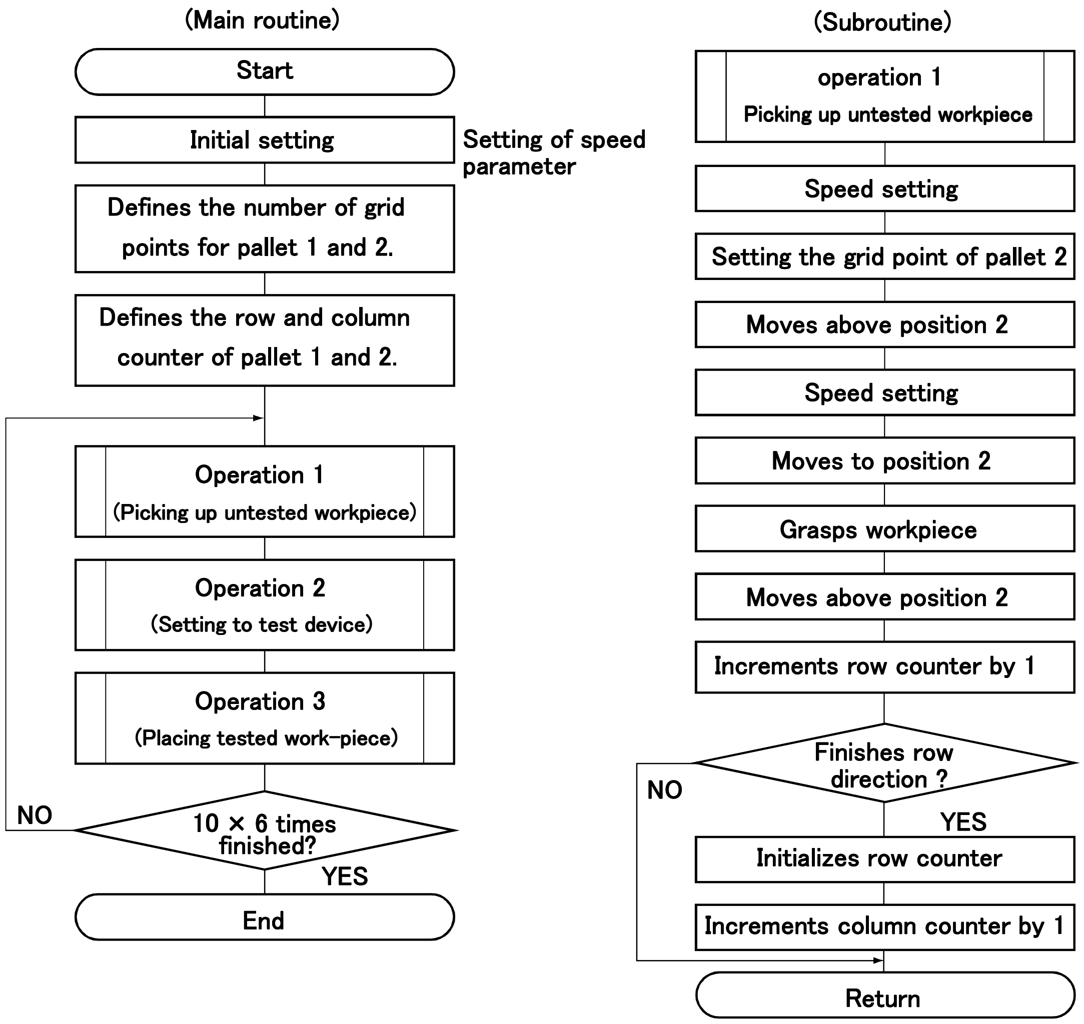
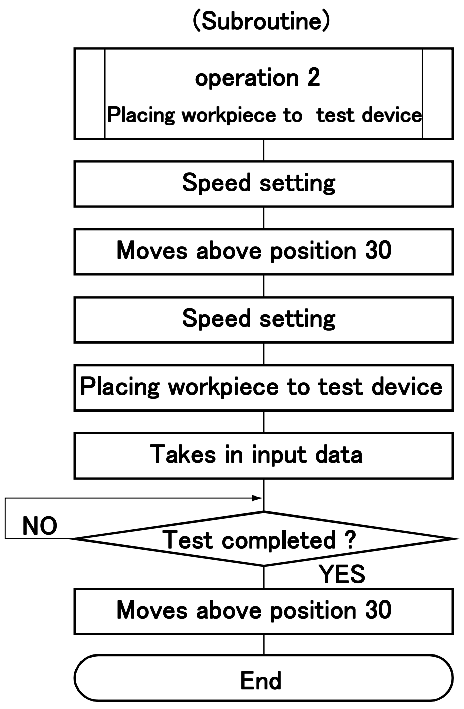

(Main routine)
Start
Initial setting
Setting of speed parameter
Defines the number of grid points for pallet 1 and 2.
Defines the row and column counter of pallet 1 and 2.
Operation 1 (Picking up untested workpiece)
Operation 2 (Setting to test device)
Operation 3 (Placing tested work-piece)
10 × 6 times finished?
NO
YES
End
(Subroutine)
operation 1 Picking up untested workpiece
Speed setting
Setting the grid point of pallet 2
Moves above position 2
Speed setting
Moves to position 2
Grasps workpiece
Moves above position 2
Increments row counter by 1
Finishes row direction ?
NO
YES
Initializes row counter
Increments column counter by 1
Return

Fig.4-7 : Flowchart



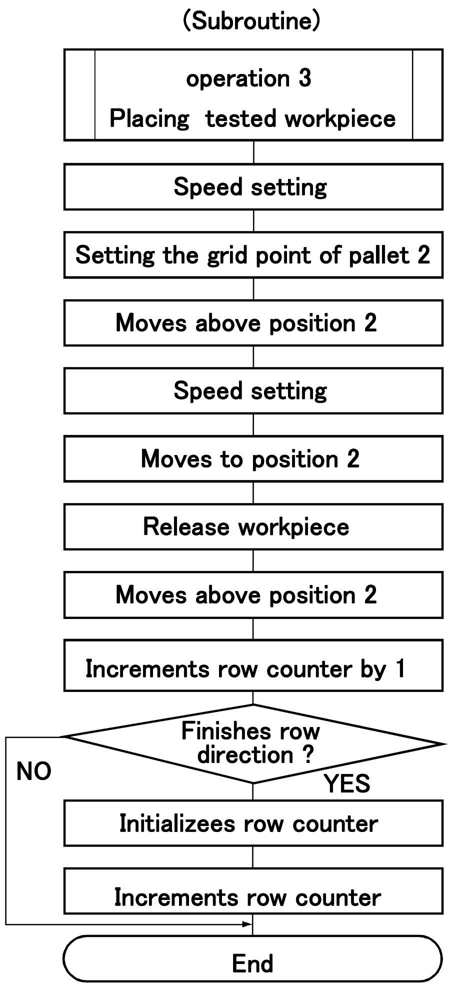


Fig.4-8 : Flowchart (Continue)

6) Schematic diagram

Fig. 4-9 shows general description of operation.

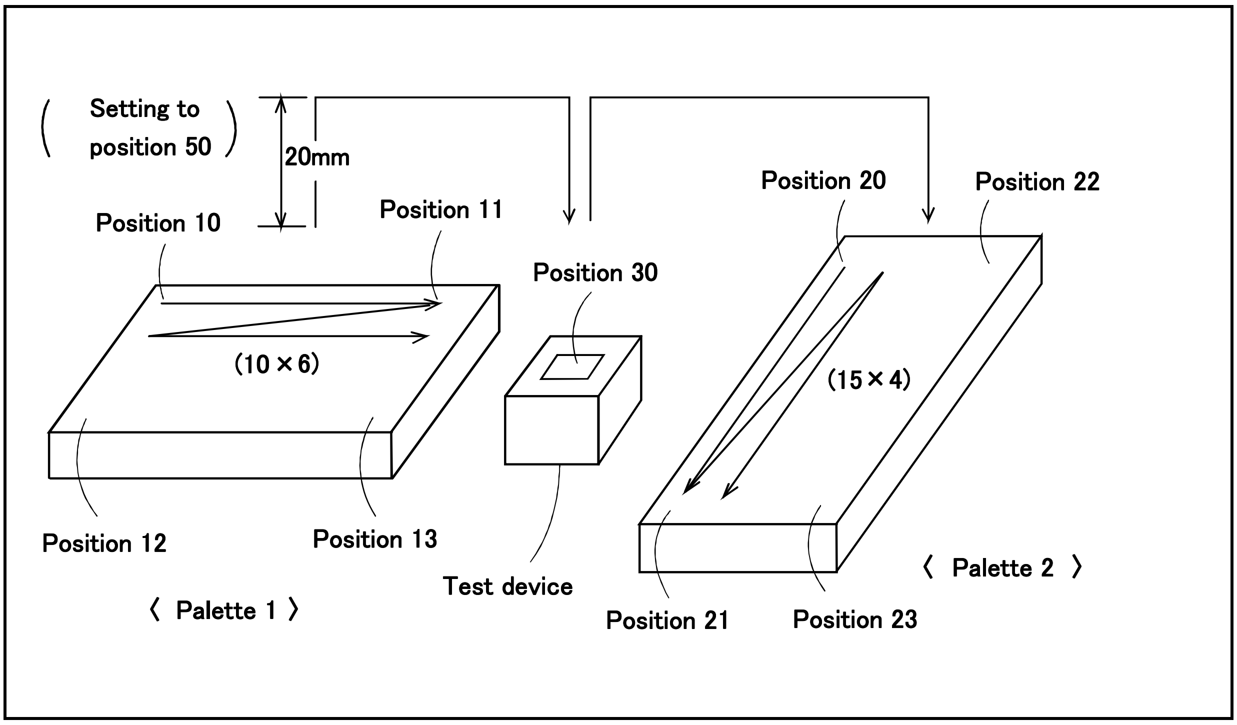


Fig.4-9 : Palletizing

7) Example program

```
(Initial setup)
10   PD 50,0,0,20,0,0,0 Note1)  ;Defines the aerial distance of travel (Z=20 ㎜) as position 50.
15   TL  145               ;Sets tool length at 145 ㎜.
20   GP 10,8,10            ;Sets hand open/close parameters.
25   PA 1,10,6             ;Defines the number of grid points in the column and row directions for pallet
                            1. (ver.10 × hor.6)
30   PA 2,15,4             ;Defines the number of grid points in the column and row directions for pallet
                            2. (ver.15 × hor.4)
35   SC 11,1               ;Initializes the column counter 11 of the pallet 1. (Sets 1 to the counter.)
40   SC 12,1               ;Initializes the row counter 12 of the pallet 1. (Sets 1 to the counter.)
45   SC 21,1               ;Initializes the column counter 11 of the pallet 2. (Sets 1 to the counter.)
50   SC 22,1               ;Initializes the row counter 12 of the pallet 2. (Sets 1 to the counter.)

(Main routine)
100  RC 60                 ;Sets the number of repeat cycles of a loop up to line 140.
110  GS 200                ;Calls the subroutine of picking workpieces from the 1st pallet.
120  GS 300                ;Calls the subroutine of setting workpieces on the inspection equipment.
130  GS 400                ;Calls the subroutine of placing workpieces on the 2nd pallet.
140  NX                    ;Returns to line 100.
150  ED                    ;End
```

The following program shows subroutines used in the main program:

```
(Subroutine:  Picking up the workpieces to be tested)
200  SP 25                 ;Sets speed.
202  PT 1                  ;Sets the grid point of the pallet 1 to the position 1.
204  MA  1,50, O           ;Moves to a location above position 1.
```

Note1)For 5-axis robot, this is "10   PD 50,0,0,20,0,0". For 4-axis robot, this is "10   PD 50,0,0,20,0".

```
206  SP 8                 ;Sets speed.
208  MO  1, O             ;Moves to the position 1.
210  GC                   ;Closes hand and grasps the workpiece.
212  MA 1,50, C           ;Moves to a location 20  mm above the position 1 with the workpiece grasped.
214  IC  11               ;Increments the column counter 11 of the  pallet 1 by 1.
216  CP 11                ;Sets the value of counter 11 to the internal register.
218  EQ 11,230            ;Jumps to line 230 on completing the column line. (compares with value 11.)
220  RT                   ;Ends the subroutine otherwise.
230  SC 11,1              ;Initializes counter 11. (Sets 1 to the counter.)
232  IC 12                ;Increments the row counter 12 of the pallet 2.
234  RT                   ;Ends the subroutine.

(Subroutine: Setting up the workpieces on the test equipment)
300  SP 25                ;Sets speed.
302  MT 30,-50, C         ;Moves to a location 50mm ahead of the test equipment.
304  SP 8                 ;Sets speed.
306  MO  30, C            ;Sets the workpiece to the inspection equipment.
308  ID                   ;Inputs external data.
310  TB  -7, 308          ;Waits for the test to complete.
312  MT 30,-50, C         ;Moves to a position 50 mm ahead the inspection equipment.
314  RT                   ;Ends the subroutine.
(Subroutine: Placing the tested workpiece in pallet 2)
400  SP 25                ;Sets speed.
402  PT 2                 ;Sets the grid point of the pallet 2 to the position 2.
404  MA 2,50, C           ;The robot moves to the sky after position 2.
406  SP 8                 ;The robot sets speed.
408  MO 2, C              ;The robot moves to the 2nd position.
410  GO                   ;The robot opens the hand and it puts the work.
412  MA 2,50, O           ;Moves to a location 20 mm above the position 2.
414  IC 21                ;Increments the column counter 21 of the pallet 2.
416  CP 21                ;Sets the value of the counter 21 to the internal register.
418  EQ 16,430            ;Jumps to line 430 on completing the column line. (compares with value 16.)
420  RT                   ;Ends the subroutine otherwise.
430  SC 21,1              ;Initializes counter 21. (Sets 1 into the counter.)
432  IC 22                ;Increments the row counter 22 of the pallet 2.
434  RT                   ;Ends the subroutine.
```

8) Explanation

① In this example, the robot increments the column counter of each pallet. The robot initializes them to return to the top of column when it reaches the end of the column. And it increments the row counter of each pallet to move to the next column. (From the 214 line to the 232nd lines and from the 414th line to the 432nd line)

② The robot waits for the test completion signal. (The 310th line).

③ The completion of the entire sequence is determined by the number of main program cycles. (See line 100)

(4) Example of connection with external I/O equipment

1) Work description
The following program causes the robot to select any of 8 jobs through 8 switches connected to the input for use as external I/O equipment and display the job currently being executed by any of the 8 LEDs connected to the outputs.

2) Connection
Fig. 4-10 shows connection example.

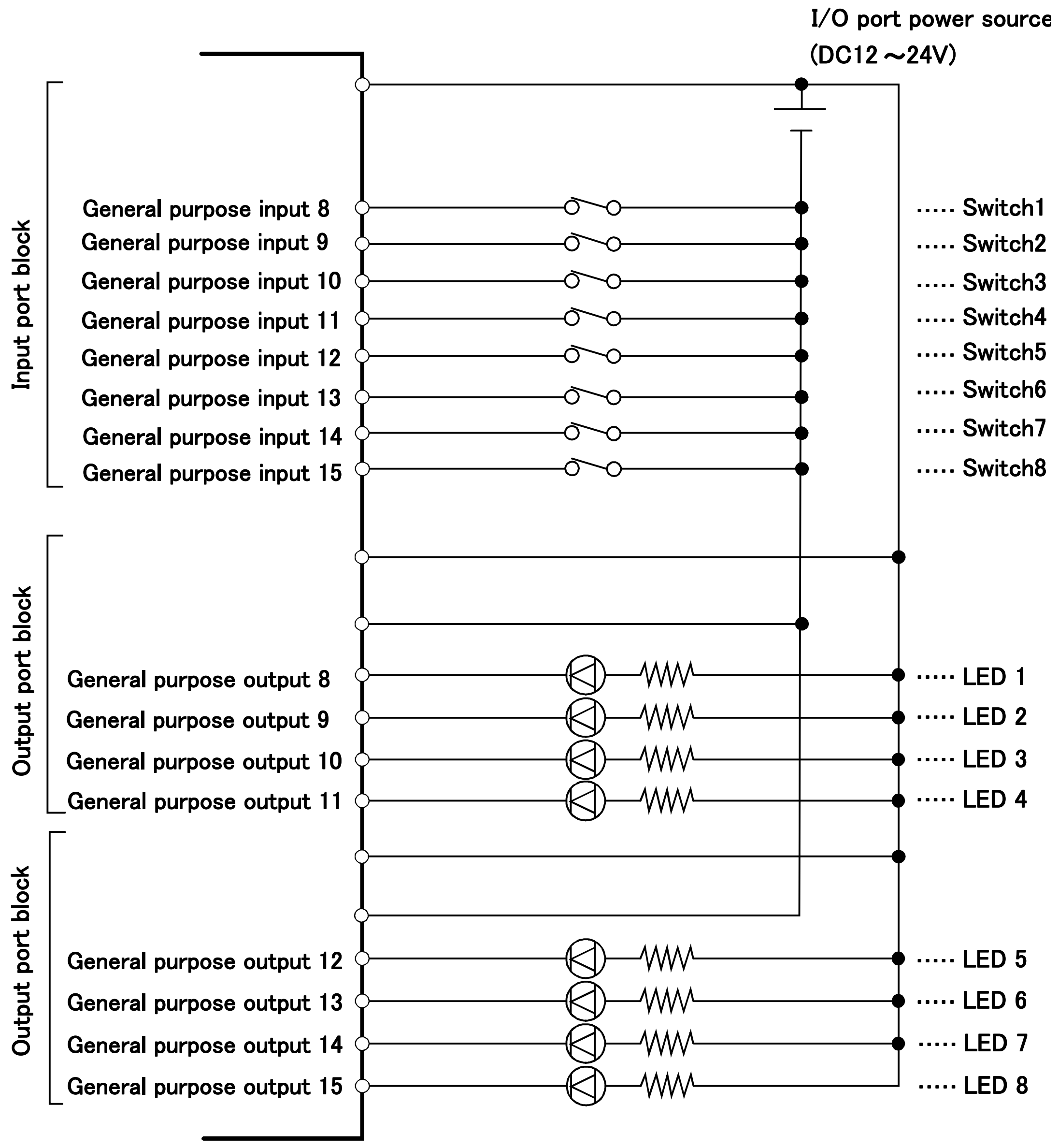


Fig.4-10 : Connection example with external I/O equipment.

3) Operation flow

Fig. 4-11 shows the flow of operation.

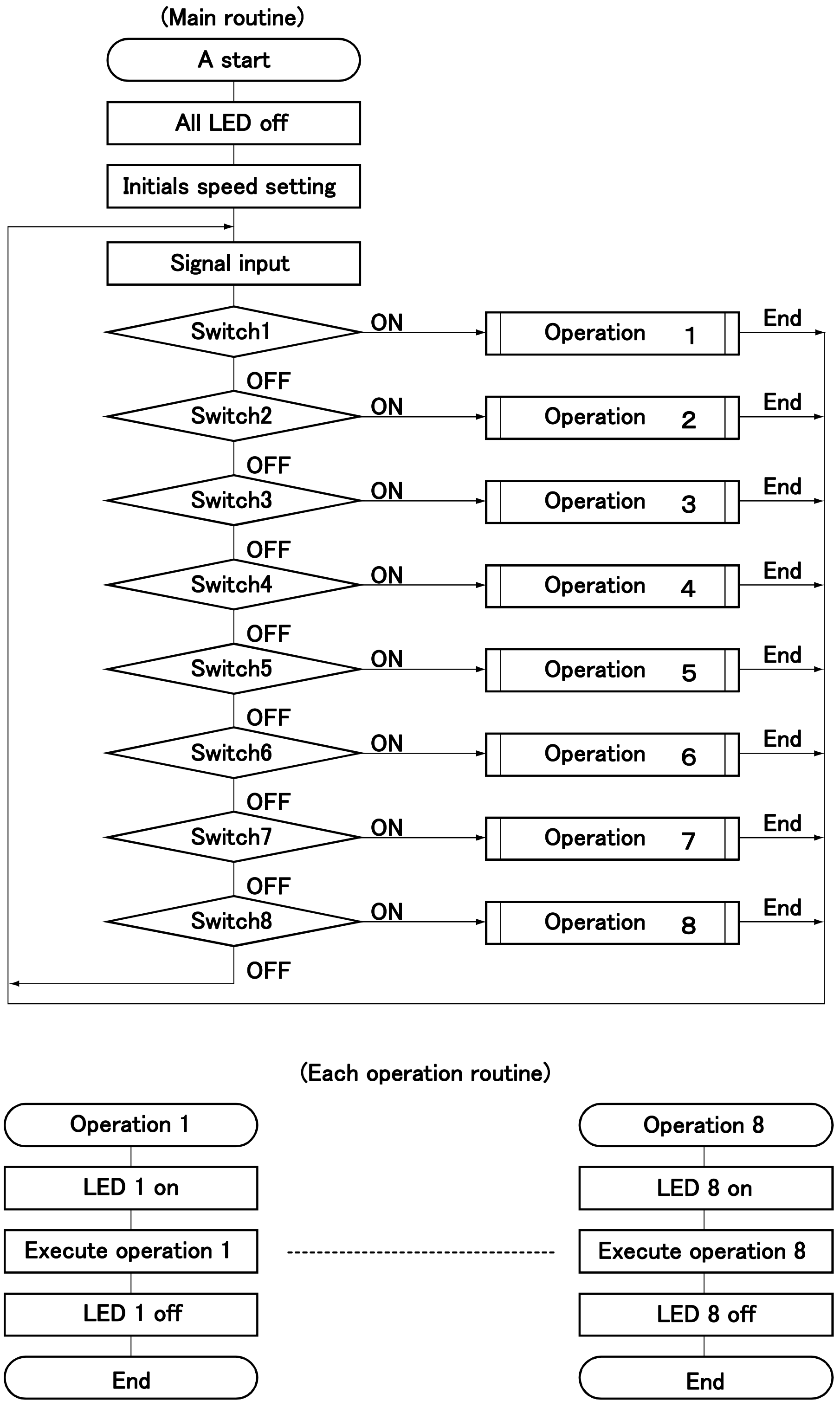


Fig.4-11 : Flowchart

4) Example program

```
(Main routine)
15   OD 0,8,8          ;All LED turns off.
20   SP 10             ;Initializes speed.
25   ID                ;Inputs signals.
30   TB +8,100         ;Jumps to the line 100 when the switch 1 is turned on.(Job 1)
31   TB +9,200         ;Jumps to the line 200 when the switch 2 is turned on.(Job 2)
32   TB +10,300        ;Jumps to the line 300 when the switch 3 is turned on.(Job 3)
33   TB +11,400        ;Jumps to the line 400 when the switch 4 is turned on.(Job 4)
34   TB +12,500        ;Jumps to the line 500 when the switch 5 is turned on.(Job 5)
35   TB +13,600        ;Jumps to the line 600 when the switch 6 is turned on.(Job 6)
36   TB +14,700        ;Jumps to the line 700 when the switch 7 is turned on.(Job 7)
37   TB +15,800        ;Jumps to the line 800 when the switch 8 is turned on.(Job 8)
38   GT 25             ;The robot jumps to the line 25 when all switches are off.

(Each operation routine)
100  OB +8             ;Turns on the 1st LED.(Job  started)
105  MO 10             ;Carries out the 1st operation.
 :                     :The 1st operation.
198  OB -8             ;Turns off the 1st LED.(Job completed)
199  GT 25             ;Jumps to the line 25.

800  OB +15            ;Turns on the 8th LED.(Job  started)
805  MO 80             ;Carries out the 8th operation.
 :                     :The 8th operation.
898  OB -15            ;Turns off the 8th LED.(Job completed)
899  GT 25             ;Jumps to the line 25.
```

MITSUBISHI ELECTRIC

HEADQUARTERS

MITSUBISHI ELECTRIC EUROPE B.V. — EUROPE
German Branch
Gothaer Straße 8
D-40880 Ratingen
Phone: +49 (0)2102 486-0
Fax: +49 (0)2102 486-1120
e mail: megfamail@meg.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. — FRANCE
French Branch
25, Boulevard des Bouvets
F-92741 Nanterre Cedex
Phone: +33 1 55 68 55 68
Fax: +33 1 55 68 56 85
e mail: factory.automation@fra.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. — IRELAND
Irish Branch
Westgate Business Park, Ballymount
IRL-Dublin 24
Phone: +353 (0) 1 / 419 88 00
Fax: +353 (0) 1 / 419 88 90
e mail: sales.info@meir.mee.com

MITSUBISHI ELECTRIC . EUROPE B.V — ITALY
Italian Branch
Via Paracelso 12
I-20041 Agrate Brianza (MI)
Phone: +39 (0)39 / 60 53 1
Fax: +39 (0)39 / 60 53 312
e mail: factory.automation@it.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. — SPAIN
Spanish Branch
Carretera de Rubí 76-80
E-08190 Sant Cugat del Vallés (Barcelona)
Phone: +34 9 3 / 565 3131
Fax: +34 9 3 / 589 2948
e mail: industrial@sp.mee.com

MITSUBISHI ELECTRIC EUROPE B.V. — UK
UK Branch
Travellers Lane
GB-Hatfield Herts. AL10 8 XB
Phone: +44 (0)1707 / 27 61 00
Fax: +44 (0)1707 / 27 86 95
E-mail: automation@meuk.mee.com

MITSUBISHI ELECTRIC CORPORATION — JAPAN
Office Tower "Z" 14 F
8-12,1 chome, Harumi Chuo-Ku
Tokyo 104-6212
Phone: +81 3 6221 6060
Fax: +81 3 6221 6075

MITSUBISHI ELECTRIC AUTOMATION — USA
500 Corporate Woods Parkway
Vernon Hills, IL 60061
Phone: +1 847 / 478 21 00
Fax: +1 847 / 478 22 83

EUROPEAN REPRESENTATIVES

GEVA — AUSTRIA
Wiener Straße 89
AT-2500 Baden
Phone: +43 (0)2252 / 85 55 20
Fax: +43 (0)2252 / 488 60
e mail: office@geva.at

Getronics b.v. — BELGIUM
Control Systems
Pontbeeklaan 43
BE-1731 Asse-Zellik
Phone: +32 (0)2 / 4 67 17 51
Fax: +32 (0)2 / 4 67 17 45
e mail: infoautomation@getronics.com

INEA CR d.o.o. — CROATIA
Drvinje 63
HR-10000 Zagreb
Phone: +385 (0)1/ 36 67 140
Fax: +385 (0)1/ 36 67 140
e mail: —

AutoCont — CZECH REPUBLIC
Control Systems s.r.o.
Nemocnicni 12
CZ-70200 Ostrava 2
Phone: +420 (0)69 / 6152 111
Fax: +420 (0)69 / 6152 562
e mail: consys@autocont.cz

Kocks Ostrava s.r.o. — CZECH REPUBLIC
Ul. Jezdiste 808
CZ-72000 Ostrava Hrabova
Phone: +420 596 / 735 095
Fax: +420 596 / 782 707
e mail: —

louis poulsen — DENMARK
industri & automation
Geminivej 32
DK-2670 Greve
Phone: +45 (0)43 / 95 95 95
Fax: +45 (0)43 / 95 95 91
e mail: lpia@lpmail.com

Beijer Electronics OY — FINLAND
Ansatie 6a
FIN-01740 Vantaa
Phone: +358 (0)9 / 886 77 500
Fax: +358 (0)9 / 886 77 555
e mail: info@beijer.fi

Kouvalias — GREECE
Industrial Robots
25, El. Venizelou Ave.
GR-17671 Kallithea
Phone: +302 (0)10 / 958 92 43
Fax: +302 (0)10 / 953 65 14
e mail: robots@acci.gr

Axicont Automatika Kft. — HUNGARY
Reitter F. U. 132
HU-1131 Budapest
Phone: +36 (0)1 / 412-0882
Fax: +36 (0)1 / 412-0883
e mail: office@axicont.hu

EUROPEAN REPRESENTATIVES

Meltrade Automatika Kft. — HUNGARY
55, Harmat St.
HU-1105 Budapest
Phone: +36 (0)1 / 2605 602
Fax: +36 (0)1 / 2605 602
e mail: office@meltrade.hu

Getronics — NETHERLANDS
Industrial Automation B.V.
Donauweg 2 B
NL-1043 AJ Amsterdam
Phone: +31 (0)20 / 587 6700
Fax: +31 (0)20 / 587 6839
e mail: info.gia@getronics.com

Beijer Electronics AS — NORWAY
Teglverksveien 1
NO-3002 Drammen
Phone: +47 (0)32 / 24 30 00
Fax: +47 (0)32 / 84 85 77
e mail: info@beijer.no

MPL Technology Sp. z o.o. — POLAND
ul. Sliczna 36
PL-31-444 Kraków
Phone: +48 (0)12 / 632 28 85
Fax: +48 (0)12 / 632 47 82
e mail: krakow@mpl.pl

ACP Autocomp a.s. — SLOVAKIA
Chalupkova 7
SK-81109 Bratislava
Phone: +421 (02)5292 2254
Fax: +421 (02)5292 2248
e mail: info@acp-autocomp.sk

INEA d.o.o. — SLOVENIA
Stegne 11
SI-1000 Ljubljana
Phone: +386 (0)1- 513 8100
Fax: +386 (0)1- 513 8170
e mail: inea@inea.si

Beijer Electronics AB — SWEDEN
Box 426
S-20124 Malmö
Phone: +46 (0)40 / 35 86 00
Fax: +46 (0)40 / 35 86 02
e mail: info@beijer.se

ECONOTEC AG — SWITZERLAND
Postfach 282
CH-8309 Nürensdorf
Phone: +41 (0)1 / 838 48 11
Fax: +41 (0)1 / 838 48 12
e mail: info@econotec.ch

GTS — TURKEY
Darülaceze Cad. No. 43A Kat. 2
TR-80270 Okmeydani-Istanbul
Phone: +90 (0)212 / 320 1640
Fax: +90 (0)212 / 320 1649
e mail: gts@turk.net

EURASIAN REPRESENTATIVES

ELEKTROSTYLE — RUSSIA
ul. Garschina 11
RU-140070 Moscow Oblast
Phone: +7 095 557-9756
Fax: +7 095 746-8880
e mail: mjuly@elektrostyle.ru

ELEKTROSTYLE — RUSSIA
Krasnij Prospekt 220-1,Office 312
RU-630049 Novosibirsk
Phone: +7 3832 / 10 66 18
Fax: +7 3832 / 10 66 26
e mail: elo@elektrostyle.ru

ICOS — RUSSIA
Ryazanskij Prospekt, 8A, Office 100
RU-109428 Moscow
Phone: +7 095 232-0207
Fax: +7 095 232-0327
e mail: mail@icos.ru

MIDDLE EAST REPRESENTATIVE

ILAN & GAVISH LTD — ISRAEL
Automation Service
24 Shenkar St., Kiryat Arie
IL-49001 Petach-Tiqva
Phone: +972 (0) 3 / 922 18 24
Fax: +972 (0) 3 / 924 07 61
e mail: iandg@internet-zahav.net

AFRICAN REPRESENTATIVE

CBI Ltd — SOUTH AFRICA
Private Bag 2016
ZAF-1600 Isando
Phone: +27 (0) 11 / 928 2000
Fax: +27 (0) 11 / 392 2354
e mail: cbi@cbi.co.za

ROB 10/04 - Printed in Germany