Chapter 13
Online commands

1 Online Command List 13-1
2 Key operation 13-6
3 Utility operation..... 13-12
4 Data handling 13-24
5 Executing the robot language independently ... 13-44
6 Control codes 13-46

1 Online Command List

Online commands can be used to operate the controller via an RS-232C interface or via an Ethernet (option).
This chapter explains the online commands which can be used. For details regarding the RS-232C connection method, refer to the "RCX Controller User's Manual". For details regarding Ethernet, refer to the "RCX Series Ethernet Manual".

About termination codes

During data transmission, the controller adds the following codes to the end of a line of transmission data.

- RS-232C
  - CR (0Dh) and LF (0Ah) are added to the end of the line when the "Termination code" parameter of communication parameters is set to "CRLF".
  - CR (0Dh) is added to the end of the line when the "Termination code" parameter of communication parameters is set to "CR".
- Ethernet (option)
  - CR (0Dh) and LF (0Ah) are added to the end of the line.

When data is received, then the data up to CR (0Dh) is treated as one line regardless of the "Termination code" parameter setting, so LF (0Ah) is ignored.
The termination code is expressed as [cr/lf] in the detailed description of each online command in "12.2 Key operation" onwards.

1.1 Online command list: Function specific

Key operation

Operation type		Command	Option	Condition
Change mode	AUTO mode	AUTO		3
	PROGRAM mode	PROGRAM		
	MANUAL mode	MANUAL		
	SYSTEM mode	SYSTEM		
Program	Reset program	RESET		4
	Execute program	RUN		
	Execute one line	STEP		
	Skip one line	SKIP		
	Execute to next line	NEXT		
	Stop program	STOP		2
Set break point		BREAK	m, n (m: break point No., n: line)	4
Switch execution task		CHGTSK		3
Change manual speed	Main robot	MSPEED	k (k : 1-100)	3
	Sub robot	MSPEED2	k (k : 1-100)	
Move to absolute reset position	Main robot	ABSADJ	k, 0 or k, 1 (k : 1-6)	3
	Sub robot	ABSADJ2	k, 0 or k, 1 (k : 1-6)	
Absolute reset on each axis	Main robot	ABSRESET	k (k : 1-6)	
	Sub robot	ABSRESET2	k (k : 1-6)	
Return-to-origin	Main robot	ORGRTN	k (k : 1-6)	3
	Sub robot	ORGRTN2	k (k : 1-6)	
Manual movement (inching)	Main robot	INCH	k+ or k- (k : X, Y, Z, R, A, B)	3
	Sub robot	INCH2	k+ or k- (k : X, Y, Z, R, A, B)	
Manual movement (jog)	Main robot	JOG	k+ or k- (k : X, Y, Z, R, A, B)	3
	Sub robot	JOG2	k+ or k- (k : X, Y, Z, R, A, B)	

Operation type		Command	Option	Condition
Point data teaching	Main robot	TEACH	m (m : point No.)	3
	Sub robot	TEACH2	m (m : point No.)	

Utility

Operation type		Command	Option	Condition
Acquire program execution status		PADDR		4
Copy program 1 to program 2		COPY	<program 1> TO <program 2>	3
Copy points "m - n" to point "k"			Pm-Pn TO Pk	
Copy point comments "m - n" to point comment "k"			PCm-PCn TO PCk	
Delete program		ERA	<program>	3
Delete points "m - n"			Pm-Pn	
Delete point comments "m - n"			PCm-PCn	
Delete pallet "m"			PLm	
Rename "program 1" to "program 2"		REN	<program 1> TO <program 2>	3
Change program attribute		ATTR	<program> TO s (s : RW/RO)	3
Initialize data	Program	INIT	PGM	3
	Point		PNT	
	Shift		SFT	
	Hand		HND	
	Pallet		PLT	
	Point comment		PCM	
	All data except parameters		MEM	
	Parameter		PRM	
	All data (MEM+PRM)		ALL	
Initialize data	Communication parameter	INIT	CMU	3
Initialize data	Error log	INIT	LOG	3
Setting	Display language	LANGUAGE	k	3
Setting	Point units	UNIT	k	3
	Clear line message	MSGCLR		1
	Setting Access level	ACCESS	k	3
	Setting Execution level	EXELVL	k	3
	Sequence execution flag	SEQUENCE	k	3
Setting	Main hand system	ARMTYP	m, k	3
	Sub hand system	ARMTYP2	m, k	
Reset internal emergency stop flag		EMGRST		1
Check or set date		DATE		2
Check or set time		TIME		2

Conditions: 1. Always executable.
2. Not executable during inputs from the programming box.
3. Not executable during inputs from the programming box, and while the program is running.
4. Not executable during inputs from the programming box, while the program is running, and when specific restrictions apply.

Data handling

Operation type		Command	Option	Condition
Acquiring status	Display language	?	LANGUAGE	1
	Access level		ACCESS	
	Arm status		ARM	
	Break point status		BREAK	
	Controller configuration		CONFIG	
	Execution level		EXELVL	
	Mode indication		MOD	
	Error message		MSG [m, n]	
	Return-to-origin status		ORIGIN	
	Absolute reset status		ABSRST	
	Servo status		SERVO	
	Sequence execution flag status		SEQUENCE	
	AUTO/MANUAL speed status		SPEED	
	Point unit status		UNIT	
	Version information		VER	
	Current main robot position (pulse coordinate)		WHERE	
	Current sub robot position (pulse coordinate)		WHERE2	
	Current main robot position (XY coordinate) (including extended setting)		WHRXY	
	Current sub robot position (XY coordinate) (including extended setting)		WHRXY2	
	Task number		TASKS	
	Task operation status		TSKMON	
	Selected shift status		SHIFT	
	Selected hand status		HAND	
	Remaining memory capacity		MEM	
	Emergency stop status		EMG	
	Error status by self-diagnosis		SELFCHK	
	Option slot status		OPSLOT	
	Main group current torque value		CHKTRQ	
	Sub group current torque value		CHKTRQ2	
	Numerical data		Numerical expression	
	Character string data		Character string expression	
	Point data		Point expression	
	Shift data		Shift expression	
Data readout		READ		2
Data write		WRITE		2

Robot language independent execution

Operation type	Command	Option	Condition
Program switching	SWI	<program>	4
Robot language executable independently			4

Control code

Operation type	Command	Option	Condition
Execution language interruption	^C(=03H)		1

Conditions: 1. Always executable.
2. Not executable during inputs from the programming box.
3. Not executable during inputs from the programming box, and while the program is running.
4. Not executable during inputs from the programming box, while the program is running, and when specific restrictions apply.

1.2 Online command list: In alphabetic order

Command	Option	Meaning	Condition
?	ABSRST	Acquire absolute reset status	1
	ACCESS	Acquire access level	1
	ARM	Acquire arm status	1
	BREAK	Acquire break point status	1
	CHKTRQ	Acquire main group current torque value	1
	CHKTRQ2	Acquire sub group current torque value	1
	CONFIG	Acquire controller configuration	1
	EMG	Acquire emergency stop status	1
	EXELVL	Acquire execution level	1
	HAND	Acquire selected hand status	1
	LANGUAGE	Acquire display language	1
	MEM	Acquire remaining memory capacity	1
	MOD	Acquire mode indication	1
	MSG [m, n]	Acquire error message	1
	OPSLOT	Acquire option slot status	1
	ORIGIN	Acquire return-to-origin status	1
	SELFCHK	Acquire error status by self-diagnosis	1
	SEQUENCE	Acquire sequence execution flag status	1
	SERVO	Acquire servo status	1
	SHIFT	Acquire selected shift status	1
	SPEED	Acquire AUTO/MANUAL speed status	1
	TASKS	Acquire task number	1
	TSKMON	Acquire task operation status	1
	UNIT	Acquire point position status	1
	VER	Acquire version	1
	WHERE	Acquire current main robot position (pulse coordinate)	1
	WHERE2	Acquire current sub robot position (pulse coordinate)	1
	WHRXY	Acquire current main robot position (XY coordinate)	1
	WHRXY2	Acquire current sub robot position (XY coordinate)	1
	Shift expression	Acquire shift data	1
	Point expression	Acquire point data	1
	Numeric expression	Acquire numeric data	1
	Character string expression	Acquire character string data	1
^C (=03H)		Execution language interruption	1
ABSADJ	k, 0 or k, 1 (k : 1-6)	Move to absolute reset position Main robot	3
ABSADJ2	k, 0 or k, 1 (k : 1-6)	Move to absolute reset position Sub robot	3
ABSRESET	k (k : 1-6)	Absolute reset on each axis Main robot	3
ABSRESET2	k (k : 1-6)	Absolute reset on each axis Sub robot	3
ACCESS	k	Set access level	3
ARMTYP	m, k	Set main hand system	3
ARMTYP2	m, k	Set sub hand system	3
ATTR	<program> TO s (s : RW/RO)	Change program attribute	3
AUTO		Change mode: AUTO mode	3
BREAK	m, n (m: break point No., n: line)	Set break point	4
CHGTSK		Switch execution task	3
COPY	<program1> to <program2>	Copy program 1 to program 2	3
	PCm-PCn TO PCk	Copy point comments "m - n" to point comments "k"	3
	Pm-Pn TO Pk	Copy points "m - n" to points "k"	3
DATE		Check or set the date	2
EMGRST		Reset internal emergency stop flag	1
ERA	<program>	Delete program	3
	PCm-PCn	Delete point comments "m" to "n"	3
	PLm	Delete pallet "m"	3
	Pm-Pn	Delete points "m" to "n"	3

Command	Option	Meaning	Condition
EXELVL	k	Execution level	3
INCH	k+ or k- (k : X, Y, Z, R, A, B)	Manual movement (inching) Main robot	3
INCH2	k+ or k- (k : X, Y, Z, R, A, B)	Manual movement (inching) Sub robot	3
INIT	ALL	Initialize all data (MEM+PRM)	3
	CMU	Initialize communication parameter	3
	HND	Initialize hand data	3
	LOG	Initialize error history	3
	MEM	Initialize all memory data except parameters	3
	PCM	Initialize point comment data	3
	PGM	Initialize program data	3
	PLT	Initialize pallet data	3
	PNT	Initialize point data	3
	PRM	Initialize parameter data	3
	SFT	Initialize shift data	3
JOG	k+ or k- (k : X, Y, Z, R, A, B)	Manual movement (jog) Main robot	3
JOG2	k+ or k- (k : X, Y, Z, R, A, B)	Manual movement (jog) Sub robot	3
LANGUAGE	k	Set display language	3
MANUAL		Change mode: MANUAL mode	3
MSGCLR		Setting Clear line message	1
MSPEED	k (k : 1-100)	Change manual speed Main robot	3
MSPEED2	k (k : 1-100)	Change manual speed Sub robot	3
NEXT		Execute program to next line	4
ORGRTN	k (k : 1-6)	Return-to-origin Main robot	3
ORGRTN2	k (k : 1-6)	Return-to-origin Sub robot	3
PADDR		Acquire program execution status	4
PROGRAM		Change mode: PROGRAM mode	3
READ		Read data	2
REN	<program 1> TO <program 2>	Change program name from "1" to "2"	3
RESET		Reset program	4
RUN		Execute program	4
SEQUENCE	k	Set sequence execution flag	3
SKIP		Program: Skip one line	4
STEP		Program: Execute one line	4
STOP		Stop program	2
SWI	<program>	Switch programs	4
SYSTEM		Change mode: SYSTEM mode	3
TEACH	m (m: point number)	Point data teaching Main robot	3
TEACH2	m (m: point number)	Point data teaching Sub robot	3
TIME		Check or set time	2
UNIT	k	Set point unit system	3
WRITE		Write data	2
-		Robot language executable independently	4

Conditions: 1. Always executable.

2. Not executable during inputs from the programming box.
3. Not executable during inputs from the programming box, and while the program is running.
4. Not executable during inputs from the programming box, while the program is running, and when specific restrictions apply.

2 Key operation

2.1 Changing the mode

Command format

```
@AUTO[cr/lf]
@PROGRAM[cr/lf]
@MANUAL[cr/lf]
@SYSTEM[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE

- Basically, a response "OK" appears when an instruction from key operation online command is received.
- An error message responds if online commands cannot be executed due to error.

Meaning Changes the mode.

AUTO ..... Changes to AUTO mode.
PROGRAM ..... Changes to PROGRAM mode.
MANUAL ..... Changes to MANUAL mode.
SYSTEM ..... Changes to SYSTEM mode.

SAMPLE

```
Command:   @AUTO[cr/lf]
Response:  OK[cr/lf]
```

2.2 AUTO mode operation

1. Program execution

Command format

```
@RESET[cr/lf]
@RUN[cr/lf]
@STEP[cr/lf]
@SKIP[cr/lf]
@NEXT[cr/lf]
@STOP[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE

- Programs can be executed only in AUTO mode.

Meaning Executes or stops the current program.

RESET.....Resets the program.
RUN .....Executes the program.
STEP.....Executes one line of the program. (Enters the subroutine.)
SKIP .....Skips one line of program. (Skips one line of the subroutine.)
NEXT .....Executes to the next line. (Executes the subroutine as one line.)
STOP .....Stops the program.

SAMPLE

```
Command:   @RUN[cr/lf]
Response:  OK[cr/lf]
```

2. Setting a break point

Command format

```
@BREAK  m,n[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- Programs can be executed only in AUTO mode.

Values
m ..... Break point number: 1 to 4
n ..... Line number to set a break point: 1 to 9999

Meaning
Sets a break point used to temporarily stop execution of the program.
When setting a break point in the COMMON program, the line number should be +10000.
Break point is cleared when 0 is specified as the line number.

SAMPLE

```
Command:   @BREAK 1,28[cr/lf]
Response:  OK[cr/lf]
```

3. Switching the execution task

Command format

```
@CHGTSK[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- Programs can be executed only in AUTO mode.

Meaning
Switches the selected task while program execution is stopped.
The ongoing task is switched to another task not being executed in order from task 1 → 2 → ... → 8 → 1.

SAMPLE

```
Command:   @CHGTSK[cr/lf]
Response:  OK[cr/lf]
```

2.3 MANUAL mode operation

1. Changing the MANUAL mode speed

Command format

```
@MSPEED  k[cr/lf]
@MSPEED2  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- The MANUAL mode speed can be changed only in MANUAL mode.

Values k .....Manual movement speed: 1 to 100

Meaning Changes the MANUAL mode movement speed.
MSPEED: Changes manual movement speed for main robot.
MSPEED2: Changes manual movement speed for sub robot.

SAMPLE

```
Command:   @MSPEED 50[cr/lf]
Response:  OK[cr/lf]
```

2. Absolute reset

Command format

```
@ABSADJ  k,f[cr/lf]
@ABSADJ2  k,f[cr/lf]
@ABSRESET  k[cr/lf]
@ABSRESET2  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- The MANUAL mode speed can be changed only in MANUAL mode.

Values k .....Designated axis: 1 to 6
f .....Movement direction / 0: + direction, 1: - direction

Meaning Performs absolute reset.
ABSADJ.....Moves the main robot axis to an absolute reset position.
ABSADJ2.....Moves the sub robot axis to an absolute reset position.
ABSRESET .....Performs absolute reset on the main robot axis.
ABSRESET2 .....Performs absolute reset on the sub robot axis.

SAMPLE

```
Command:   @ABSADJ 1,0[cr/lf]
Response:  OK[cr/lf]
```

MEMO
- ABSADJ and ABSADJ2 can be used at mark format axes.

3. Return-to-origin operation

NOTE
- An axis can be specified from software version 8.45 onwards.

Command format

```
@ORGRTN  k[cr/lf]
@ORGRTN2  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- The MANUAL mode speed can be changed only in MANUAL mode.

Values k .....Specified axis: 1 to 6

Meaning Performs return-to-origin on the specified axis.
Performs return-to-origin on an incremental mode axis when return-to-origin is executed.
Performs absolute search on a semi-absolute mode axis when return-to-origin is executed.
ORGRTN .....Performs return-to-origin on the specified main robot axis.
ORGRTN2 .....Performs return-to-origin on the specified sub robot axis.

NOTE
- Response is transmitted after movement is complete.

SAMPLE

```
Command:  @ORGRTN 1[cr/lf]
Response:  OK[cr/lf]
```

4. Manual movement: inching

Command format

```
@INCH  km[cr/lf]
@INCH2  km[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE
- The MANUAL mode speed can be changed only in MANUAL mode.

Values k .....Specified axis: X, Y, Z, R, A, B
m .....Movement direction / +, -

Meaning Manually moves (inching motion) the specified axis. The robot performs the same motion as when moved manually in inching mode with the programming box jog keys (moves a fixed distance each time a jog key is pressed).
INCH .....Moves the specified main robot axis in inching mode.
INCH2 .....Moves the specified sub robot axis in inching mode.

SAMPLE

```
Command:  @INCH X+[cr/lf]
Response:  OK[cr/lf]
```

5. Manual movement: jog

Command format

```
@JOG  km[cr/lf]
@JOG2  km[cr/lf]
```

Response format

```
OK[cr/lf]
```

NOTE

- Response is transmitted after movement is complete.

Values

k ..... Specified axis: X, Y, Z, R, A, B
m ..... Movement direction / +, -

Meaning

Manually moves (jog motion) the specified axis. The robot performs the same motion as when holding down the programming box jog keys in manual mode.
After the robot has started moving, it will stop when any of the following occurs.

- When software limit was reached.
- When interlock signal was turned off.
- When STOP key on the programming box was pressed.
- When an online command (^C (=03H)) to interrupt execution was input.

JOG ..... Moves the specified main robot axis in jog mode.
JOG2 ..... Moves the specified sub robot axis in jog mode.

SAMPLE

```
Command:  @JOG X+[cr/lf]
Response:  OK[cr/lf]
```

6. Point data teaching

NOTE

- At controllers with software version 8.28 or earlier, point numbers 0 to 4000 can be specified by point variables.
- The MANUAL mode speed can be changed only in MANUAL mode.

Command format

```
@TEACH  mmmm[cr/lf]
@TEACH2  mmmm[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values

mmmm ..... Point number for registering point data: 0 to 9999

Meaning

Registers the current robot position as point data for the specified point number. If point data is already registered in the specified point number, then that point data will be overwritten. Point data is registered in the same format as the currently selected unit system.

TEACH ..... Registers the current position of the main group as point data for the specified point number.
TEACH2 ..... Registers the current position of the sub group as point data for the specified point number.

SAMPLE

```
Command:  @TEACH 100[cr/lf]
Response:  OK[cr/lf]
```

3 Utility operation

3.1 Acquiring the program execution status

Command format

```
@PADDR[cr/lf]
```

Response format

```
<program name> Tn, m, k[cr/lf]
```

NOTE

- The current program execution status can be acquired only when the program is stopped during AUTO mode.

Values

<program name> .....Currently selected program name
Tn .....Current task number: 1 to 8
m .....Current program line number: 1 to 9999
k .....Current task priority level:17 to 47

Meaning Acquires the current program execution status.

MEMO

- If the COMMON program is used, the response format might become as follows.
  <COMMON>/<program name>, Tn,m,k[cr/lf]

SAMPLE

```
Command:   @PADDR[cr/lf]
Response:  <TEST>,T3,134,32[cr/lf]
```

3.2 Copy

1. Copying a program

Command format

```
COPY <program name 1> TO <program name 2> [cr/lf]
```

Response format

```
@COPY <program name 1> TO <program name 2> [cr/lf]
```

Values

<program name 1> .....Program name in copy source (8 characters or less consisting of alphanumeric characters and underscore)
<program name 2> .....Program name in copy destination (8 characters or less consisting of alphanumeric characters and underscore)

Meaning Copies the contents of program name 1 under program name 2.

SAMPLE

```
Command:   @COPY <TEST1> TO <TEST2>[cr/lf]
Response:  OK[cr/lf]
```

2. Copying point data

Command format

```
@COPY   Pmmmm-Pnnnn   TO   Pkkkk[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values
- mmmm ..... Top point number in copy source: 0 to 9999
- nnnn ..... Last point number in copy source: 0 to 9999
- kkkk ..... Top point number in copy destination: 0 to 9999

Meaning Copies the point data between Pmmmm and Pnnnn to Pkkkk.

SAMPLE

```
Command:   @COPY P101-P200 TO P1101[cr/lf]
Response:  OK[cr/lf]
```

NOTE
- In controllers whose software version is earlier than 8.28, point numbers from 0 to 4000 can be specified as the point variables.

3. Copying point comments

Command format

```
@COPY   PCmmmm-PCnnnn   TO   PCkkkk[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values
- mmmm ..... Top point comment number in copy source: 0 to 9999
- nnnn ..... Last point comment number in copy source: 0 to 9999
- kkkk ..... Top point comment number in copy destination: 0 to 9999

Meaning Copies the point comments between PCmmmm and PCnnnn to PCkkkk.

SAMPLE

```
Command:   @COPY PC101-PC200 TO PC1101[cr/lf]
Response:  OK[cr/lf]
```

NOTE
- In controllers whose software version is earlier than 8.28, point numbers from 0 to 4000 can be specified as the point variables.

3.3 Erase

1. Erasing a program

Command format

```
@ERA <program name> [cr/lf]
```

Response format

```
OK[cr/lf]
```

Values <program name> .....Program name to be erased (8 characters or less consisting of alphanumeric characters and underscore)

Meaning Erases the designated program.

SAMPLE

```
Command:   @ERA <TEST1>[cr/lf]
Response:  OK[cr/lf]
```

2. Erasing point data

Command format

```
@ERA   Pmmmm-Pnnnn[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values mmmm .....Top point number to be erased: 0 to 9999
nnnn .....Last point number to be erased: 0 to 9999

Meaning Erases the point data between Pmmmm and Pnnnn.

SAMPLE

```
Command:   @ERA P101-P200[cr/lf]
Response:  OK[cr/lf]
```

NOTE

- In controllers whose software version is earlier than 8.28, point numbers from 0 to 4000 can be specified as the point variables.

3. Erasing point comments

Command format

```
@ERA  PCmmmm-PCnnnn[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values mmmm ..... Top point comment number to be erased: 0 to 9999
nnnn ..... Last point comment number to be erased: 0 to 9999

Meaning Erases the point comments between PCmmmm and PCnnnn.

SAMPLE

```
Command:  @ERA PC101-PC200[cr/lf]
Response: OK[cr/lf]
```

NOTE

- In controllers whose software version is earlier than 8.28, point numbers from 0 to 4000 can be specified as the point variables.

4. Erasing pallet data

Command format

```
@ERA  PLm[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values m ..... Pallet number to be erased: 0 to 19

Meaning Erases the PLm pallet data.

SAMPLE

```
Command:  @ERA PL1[cr/lf]
Response: OK[cr/lf]
```

3.4 Rename program name

Command format

```
@REN <program name 1> TO <program name 2> [cr/lf]
```

Response format

```
OK[cr/lf]
```

Values

<program name 1> ..... Program name before renaming (8 characters or less consisting of alphanumeric characters and underscore)

<program name 2> ..... Program name after renaming (8 characters or less consisting of alphanumeric characters and underscore)

Meaning Changes the name of the specified program.

SAMPLE

```
Command:   @REN <TEST1> TO <TEST2>[cr/lf]
Response:  OK[cr/lf]
```

3.5 Changing the program attribute

Command format

```
@ATTR <program name> TO s[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values

<program name> ..... Program name to change the attribute (8 characters or less consisting of alphanumeric characters and underscore)

s ..... Attribute / RW: read & write, RO: read only

Meaning Changes the attribute of the designated program.

SAMPLE

```
Command:   @ATTR <TEST1> TO RO[cr/lf]
Response:  OK[cr/lf]
```

3.6 Initialize

1. Initializing the memory

Command format

```
@INIT <memory area>[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values

<memory area> ..... One of the following memory areas is specified.
PGM ..... Initializes the program area.
PNT ..... Initializes the point data area.
SFT ..... Initializes the shift data area.
HND ..... Initializes the hand data area.
PLT ..... Initializes the pallet data area.
PCM ..... Initializes the point comment area.
MEM ..... Initializes the above areas (PGM ... all data up to PCM).
PRM ..... Initializes the parameter area.
ALL ..... Initializes all areas (MEM+PRM).

Meaning Initializes the memory.

SAMPLE

```
Command:   @INIT PGM[cr/lf]
Response:  OK[cr/lf]
```

2. Initializing the communication port

Command format

```
@INIT  CMU[cr/lf]
```

Response format

```
OK[cr/lf]
```

Meaning Initializes the communication port parameters.
For information about the communication port initial settings, refer to the Controller user's manual.

SAMPLE

```
Command:   @INIT CMU[cr/lf]
Response:  OK[cr/lf]
```

3. Initializing the error log

Command format

```
@ INIT LOG[cr/lf]
```

Response format

```
OK[cr/lf]
```

Meaning Initializes the error log.

SAMPLE

```
Command:   @INIT LOG[cr/lf]
Response:  OK[cr/lf]
```

3.7 Setting the display language

Command format

```
@LANGUAGE  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values k .....Display language / 0: Japanese, 1: English

Meaning Sets the controller display language.

SAMPLE

```
Command:   @ LANGUAGE 1[cr/lf]
Response:  OK[cr/lf]
```

3.8 Setting the coordinates and units in MANUAL mode

Command format

```
@UNIT  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values k ..... Unit definition
0 : pulses
1 : mm or degrees
2 : mm or degrees in tool coordinate mode

Meaning Select the display unit to indicate current position.
k=2 (tool coordinate mode) can be selected only when the hand attached to the R-axis of a Cartesian or SCARA robot is selected.

NOTE

- k=2 (mm or degree units in tool coordinate mode) is available from software version 8.19 onwards.

SAMPLE

```
Command:   @UNIT 1[cr/lf]
Response:  OK[cr/lf]
```

3.9 Clearing the programming box error message

Command format

```
@MSGCLR[cr/lf]
```

Response format

```
OK[cr/lf]
```

Meaning Clears the error messages displayed on the programming box.

SAMPLE

```
Command:   @MSGCLR[cr/lf]
Response:  OK[cr/lf]
```

3.10 Setting the UTILITY mode

1. Setting the access level

Command format

```
@ACCESS  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

REFERENCE

- For details regarding the access level, refer to the Controller user's manual.

Values k ..... Access level: 0 to 3

Meaning Sets the access level.

SAMPLE

```
Command:  @ ACCESS 1[cr/lf]
Response: OK[cr/lf]
```

2. Setting the execution level

Command format

```
@EXELVL  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

REFERENCE

- For details regarding the execution level, refer to the Controller user's manual.

Values k ..... Execution level: 0 to 8

Meaning Sets the execution level.

SAMPLE

```
Command:  @ EXELVL 1[cr/lf]
Response: OK[cr/lf]
```

3. Setting the sequence program execution flag

Command format

```
@SEQUENCE  k[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values k .....Execution flag / 0: disable, 1: enable, 3: enable (DO reset)

Meaning Sets the sequence program execution flag.

SAMPLE

```
Command:   @ SEQUENCE 1[cr/lf]
Response:  OK[cr/lf]
```

4. Setting the SCARA robot hand system

Command format

```
@ARMTYP  m,k[cr/lf]
@ARMTYP2  m,k[cr/lf]
```

Response format

```
OK[cr/lf]
```

Values m .....Current hand system / 0: right-handed system, 1: left-handed system
k .....Hand system at program reset / 0: right-handed system, 1: left-handed system

Meaning Sets the SCARA robot hand system.
ARMTYP .....Changes the main robot hand system.
ARMTYP2 .....Changes the sub robot hand system.

SAMPLE

```
Command:   @ ARMTYP 0, 0 [cr/lf]
Response:  OK[cr/lf]
```

5. Resetting the internal emergency stop flag

Command format

```
@EMGRST[cr/lf]
```

Response format

```
OK[cr/lf]
```

Meaning Resets the internal emergency stop flag.

SAMPLE

```
Command:   @ EMGRST[cr/lf]
Response:  OK[cr/lf]
```

3.11 Checking and setting the date

NOTE

- The date can be checked and set on the following software versions.

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Command format

```
@DATE[cr/lf]
```

Response format

```
current date: yy/mm/dd[cr/lf]
enter new date: (YY/MM/DD)[cr/lf]
```

Values

yy/mm/dd.....Current date (year, month, day)
yy.....Lower 2 digits of the year (00 to 99)
mm .....Month (01 to 12)
dd .....Day (01 to 31)

Meaning Acquires the current date in the controller and sets a new date.

Command format

```
aa/bb/cc[cr/lf]
```

Response format

```
OK[cr/lf] or "error message" [cr/lf]
```

NOTE

- To change only the year or month, the slash (/) following it can be omitted.

  Example:

  To set the year to 2007, enter 07(cr/lf).

  To set the month to June, enter /06(cr/lf).

Values

aa/bb/cc .....Date to be set. (year/month/day)
aa.....Lower 2 digits of the year (00 to 99) *This can be omitted.
bb .....Month (01 to 12) *This can be omitted.
cc.....Day (01 to 31) *This can be omitted.

MEMO

- The currently set values are used for the omitted items.
- If only [cr/lf] is transmitted, then the date remains unchanged.
- If an improbable date is entered, then "5.2: Data error" occurs.

SAMPLE 1

```
To change only the day,
 //15[cr/lf]·····Day is set to 15th.
```

SAMPLE 2

```
Command:   @DATE[cr/lf]
Response:  current date: 07/05/10[cr/lf]
           enter new date: (YY/MM/DD)[cr/lf]
Transmission: 07/05/11[cr/lf]
Response:  OK[cr/lf]
```

3.12 Checking and setting the time

NOTE

- The time can be checked and set on the following software versions.

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Command format

```
@TIME[cr/lf]
```

Response format

```
current time: hh:mm:ss[cr/lf]
enter new time: (HH:MM:SS)[cr/lf]
```

Values

hh:mm:ss ..... Current time
hh ..... hour (00 to 23)
mm ..... minute (00 to 59)
ss ..... second (00 to 59)

Meaning Acquires the current time in the controller and sets a new time.

Command format

```
aa:bb:cc[cr/lf]
```

Response format

```
OK[cr/lf] or "error message" [cr/lf]
```

Values

aa:bb:cc ..... Time to be set.
aa ..... hour (00 to 23) *This can be omitted.
bb ..... minute (00 to 59) *This can be omitted.
cc ..... second (00 to 59) *This can be omitted.

MEMO

- The currently set values are used for the omitted items.
- If only [cr/lf] is transmitted, then the time remains unchanged.
- If an improbable time is entered, then "5.2: Data error" occurs.

SAMPLE 1

```
To change only the minute,
 :20:[cr/lf]·····Minute is set to 20 minutes.
```

SAMPLE 2

```
Command:   @TIME[cr/lf]
Response:  current time: 10:21:35[cr/lf]
           enter new time:(HH/MM/SS)[cr/lf]
Transmission: 10:25:00[cr/lf]
Response:  OK[cr/lf]
```

4 Data handling

4.1 Acquiring the display language

Command format

```
@?LANGUAGE[cr/lf]
```

Response format

```
m[cr/lf]
```

Values m .....Display language / JAPANESE or ENGLISH

Meaning Acquires the language for displaying messages.

SAMPLE

```
Command  @?LANGUAGE[cr/lf]
Response JAPANESE[cr/lf]
```

4.2 Acquiring the access level

Command format

```
@?ACCESS[cr/lf]
```

Response format

```
LEVELk[cr/lf]
```

Values k .....Access level: 0 to 3

Meaning Acquires the access level.

SAMPLE

```
Command  @?ACCESS[cr/lf]
Response LEVEL2[cr/lf]
```

REFERENCE

- For a detailed description of the access level, refer to the Controller user's manual.

4.3 Acquiring the arm status

Command format

```
@?ARM[cr/lf]
```

Response format

```
m1/s1,m2/s2[cr/lf]
```

Values

Main robot

m1 ..... Current arm setting status / RIGHTY: right-handed system, LEFTY: left-handed system

m2 ..... Arm setting status at program reset / RIGHTY: right-handed system, LEFTY: left-handed system

Sub robot

s1 ..... Current arm setting status / RIGHTY: right-handed system, LEFTY: left-handed system

s2 ..... Arm setting status at program reset / RIGHTY: right-handed system, LEFTY: left-handed system

Meaning Acquires the arm setting status.

MEMO

- Valid only on SCARA robots.
- "s1" and "s2" are not displayed when sub robot is not set.

SAMPLE

```
Command @?ARM[cr/lf]
Response RIGHTY,RIGHTY[cr/lf]
```

4.4 Acquiring the break point status

Command format

```
@?BREAK[cr/lf]
```

Response format

```
k1,k2,k3,k4[cr/lf]
```

Values kn ..... Line number on which break point "n" is set: 1 to 9999

Meaning Acquires the break point status.
When kn is 0, this means no break point is set.
When a break point is set in the COMMON program, the line number shows +10000.

SAMPLE

```
Command @?BREAK[cr/lf]
Response 12,35,0,0[cr/lf]
```

4.5 Acquiring the controller configuration status

Command format

```
@?CONFIG[cr/lf]
```

Response format

```
mr/sr-ma/sa-r-o1-o2[cr/lf]
```

Values
- mr ..... Main robot name
- sr ..... Sub robot name
- ma ..... Main group axis setting (Auxiliary axes are shown separated by "+".)
- sa ..... Sub group axis setting (Auxiliary axes are shown separated by "+".)
- r ..... Standard interface unit
- o1 ..... Option unit
- o2 ..... Other setting

Meaning Acquires the controller configuration status.

MEMO
- "sr" and "sa" are not displayed when sub robot is not set.

SAMPLE

```
Command  @?CONFIG[cr/lf]
Response YK250X-XYZR-SRAM/196kB,DIO_N-DIO_N(1/2)[cr/lf]
```

4.6 Acquiring the execution level

Command format

```
@?EXELVL[cr/lf]
```

Response format

```
LEVELk[cr/lf]
```

Values
- k ..... Execution level: 0 to 8

Meaning Acquires the execution level.

REFERENCE
- For a detailed description of the execution level, refer to the Controller user's manual.

SAMPLE

```
Command  @?EXELVL[cr/lf]
Response LEVEL2[cr/lf]
```

4.7 Acquiring the mode status

Command format

```
@?MOD[cr/lf]
```

Response format

```
s[cr/lf]
```

Values s.....Mode status

s		Meaning
English	Japanese	
AUTO	ジドウ	AUTO mode
PROGRAM	プログラム	PROGRAM mode
MANUAL	シュドウ	MANUAL mode
SYSTEM	システム	SYSTEM mode

Meaning Acquires the controller mode status.

SAMPLE

```
Command @?MOD[cr/lf]
Response AUTO[cr/lf]
```

4.8 Acquiring the message

Command format 1

```
@?MSG[c/r]
```

Response format 1

```
gg,bb: msg[c/r]or OK[c/r]
```

Command format 2

```
@?MSG  m,n[cr/lf]
```

Response format 2

```
yy/mm/dd,hh:mm:ss  gg.bb:msg[cr/lf]
yy/mm/dd,hh:mm:ss  gg.bb:msg[cr/lf]
:
OK[cr/lf]
```

Values

gg.....Error group
bb .....Error category
msg .....Error message
m .....Top number to be acquired: 1 to 500
n .....Last number to be acquired: 1 to 500
yy/mm/dd.....Date (year/month/day) when error occurred
hh:mm:ss .....Time (hour:minute:second) when error occurred

Meaning Command format 1 acquires information on the message line displayed on the programming box.
Command format 2 acquires error history message.

SAMPLE 1

```
Command @?MSG[cr/lf]
Response 5.30: Undefined identifier[cr/lf] or OK[cr/lf]
```

SAMPLE 2

```
Command @?MSG  1,5[cr/lf]
Response 01/10/28,14:20:20 5.30: Undefined identifier[cr/lf]
01/10/28,14:18:34 5.1: Syntax error[cr/lf]
01/10/28,14:10:54 5.30: Undefined identifier[cr/lf]
01/10/28,14:05:40 14.22: No start code(@)[cr/lf]
01/10/28,14:05:00 5.52: Command doesn't exist[cr/lf]
OK[cr/lf]
```

4.9 Acquiring return-to-origin status

Command format

```
@?ORIGIN[cr/lf]
```

Response format

```
COMPLETE[cr/lf] or INCOMPLETE[cr/lf]
```

Meaning Acquires return-to-origin status.
Response format .....COMPLETE: Return-to-origin is complete.
INCOMPLETE: Return-to-origin is incomplete.

SAMPLE

```
Command @?ORIGIN[cr/lf]
Response COMPLETE[cr/lf]
```

4.10 Acquiring the absolute reset status

Command format

```
@?ABSRST[cr/lf]
```

Response format

```
COMPLETE[cr/lf] or INCOMPLETE, xxxxxxxx[cr/lf]
```

Values xxxxxxxx.....Absolute reset status of each axis (axis 8 to axis 1 from the left)
0: Incomplete
1: Complete
9: Not applicable

Meaning Acquires the absolute reset status.
Response format .....COMPLETE: Return-to-origin is complete.
INCOMPLETE: Return-to-origin is incomplete.

SAMPLE

```
Command @?ABSRST[cr/lf]
Response INCOMPLETE,99991011[cr/lf]
```

4.11 Acquiring the servo status

Command format

```
@?SERVO[cr/lf]
```

Response format

```
OFF,xxxxxxxx [cr/lf] or ON,xxxxxxxx[cr/lf]
```

Values xxxxxxxx.....Status of each axis (axis 8 to axis 1 from the left)
0: Mechanical break ON + dynamic break ON
1: Servo ON
2: Mechanical break OFF + dynamic break OFF
9: Not applicable

Meaning Acquires the servo status.
Response outputs are defined as follows:
ON .....Motor power is ON.
OFF.....Motor power is OFF.

SAMPLE

```
Command @?SERVO[cr/lf]
Response ON,99991011[cr/lf]
```

4.12 Acquiring the sequence program execution status

Command format

```
@?SEQUENCE[cr/lf]
```

Response format

```
1. ENABLE,s[cr/lf]
2. ENABLE(RST.DO),s[cr/lf]
3. DISABLE[cr/lf]
```

Values s.....The sequence program's execution status is indicated as "RUNNING" or "STOP".
RUNNING.....Program execution is in progress.
STOP .....Program execution is stopped.

Meaning Acquires the sequence program execution status.
Response output means as follows:
ENABLE .....Enabled
ENABLE(RST.DO) .....Enabled and output is cleared at emergency stop
DISABLE .....Disabled

SAMPLE

```
Command @? SEQUENCE[cr/lf]
Response DISABLE[cr/lf]
```

4.13 Acquiring the speed setting status

Command format

```
@?SPEED[cr/lf]
```

Response format

```
ma/sa,mm/sm[cr/lf]
```

Values

Main group

ma .....Automatic movement speed setting status: 1 to 100

mm .....Manual movement speed setting status: 1 to 100

Sub group

sa .....Automatic movement speed setting status: 1 to 100

sm .....Manual movement speed setting status: 1 to 100

Meaning Acquires the speed setting status.

MEMO

- "sa" and "sm" are not displayed unless the sub robot is set.

SAMPLE

```
Command @?SPEED[cr/lf]
Response 100,50[cr/lf]
```

4.14 Acquiring the point coordinates and units

Command format

```
@?UNIT[cr/lf]
```

Response format

```
s[cr/lf]
```

Values s: Coordinates and units .....PULSE: joint coordinate in "pulse" units

MM: Cartesian coordinate in "mm" or “deg.” units

Meaning Acquires the coordinates and units for point data.

SAMPLE

```
Command @?UNIT[cr/lf]
Response PULSE[cr/lf]
```

4.15 Acquiring the version information

Command format

```
@?VER[cr/lf]
```

Response format

```
cv,cr-mv-d1/d2/d3/d4/d5/d6/d7/d8{-ov}[cr/lf]
```

Values
- cv ..... Host version number (RCX14x: V8.xx, RCX22x: V9.xx)
- cr ..... Host revision number (Rxxxx)
- mv ..... Programming box version number (Vx.xx)
- d?(?:1-8) ..... Driver version number (Vx.xx)
- ov ..... Option unit version number (Vx.xx)

Meaning Acquires the version information.

SAMPLE

```
Command @?VER[cr/lf]
Response V8.02,R1021-V5.10-V1.01/V1.01/V1.01/V1.01/-----/-----/-----/-----/[cr/lf]
```

4.16 Acquiring the current positions

1. Acquiring the current positions on pulse unit coordinates

Command format

```
@?WHERE[cr/lf]
@?WHERE2[cr/lf]
```

Response format

```
[POS]xxxxxx yyyyyy zzzzzz rrrrrr aaaaaa bbbbbb[cr/lf]
```

Values
- xxxxxx ..... Current position of axis 1 in "pulse" units
- yyyyyy ..... Current position of axis 2 in "pulse" units
- :
- bbbbbb ..... Current position of axis 6 in "pulse" units

Meaning Acquires the current positions.
WHERE: Acquires the current positions of main group axes.
WHERE2: Acquires the current positions of sub group axes.

MEMO
- WHERE2 cannot be used unless the sub robot is set.

SAMPLE

```
Command @?WHERE[cr/lf]
Response [POS]  1000   2000    3000   -40000     0     0[cr/lf]
```

2. Acquiring the current positions on XY coordinates

Command format

```
@?WHRXY[cr/lf]
@?WHRXY2[cr/lf]
```

Response format

```
[POS]xxxxxx yyyyyy zzzzzz rrrrrr aaaaaa bbbbbb[cr/lf]
```

Values

xxxxxx ..... Current position of axis 1 in "mm" or "deg" units
yyyyyy ..... Current position of axis 2 in "mm" or "deg" units
:
bbbbbb ..... Current position of axis 6 in "mm" or "deg" units

Meaning Acquires the current positions.
WHRXY: Acquires the current positions of main group axes.
WHRXY2: Acquires the current positions of sub group axes.

MEMO

- WHRXY2 cannot be used unless the sub robot is set.

SAMPLE

```
Command @?WHRXY[cr/lf]
Response [POS] 100.00  200.00  300.00  -40.00  0.00  0.00[cr/lf]
```

3. XY coordinate system current position (including extended setting) acquisition

NOTE

- The "XY coordinate system current position (including extended setting) acquisition" function is only available in software version 10.66 onwards.

Command format

```
@?WHRXYEX[cr/lf]
@?WHRXYEX2[cr/lf]
```

Response format

```
[POS]xxxxxx yyyyyy zzzzzz rrrrrr aaaaaa bbbbbb n xr yr[cr/lf]
```

Values

xxxxxx ..... Axis 1 current position in "mm" units.
yyyyyy ..... Axis 2 current position in "mm" units.
:
bbbbbb ..... Axis 6 current position in "mm" units.
n ..... SCARA robot extended hand system flag (*1)
1: Right-handed system; 2: Left-handed system
xr ..... Extended setting's X-arm rotation information (*2).
0: The "mm → pulse" converted angle data x (*3) range is –180.00° < x < = 180.00°.
1: The "mm → pulse" converted angle data x (*3) range is 180.00° < x < = 540.00°.
-1: The "mm → pulse" converted angle data x (*3) range is –540.00° < x < = -180.00°.
yr ..... Extended setting's Y-arm rotation information (*2).
0: The "mm → pulse" converted angle data y (*3) range is –180.00° < y < = 180.00°.
1: The "mm → pulse" converted angle data y (*3) range is 180.00° < y < = 540.00°.
-1: The "mm → pulse" converted angle data y (*3) range is –540.00° < y < = -180.00°.

*1: The hand system flag is "0" on all robots other than the SCARA robot.
*2: The arm rotation information is "0" on all robots other than the YK500TW robot.
*3: The joint-coordinates-converted pulse data represents each arm's distance (converted to angular data) from its mechanical origin point.

Meaning
- The acquired current position data includes additional dedicated YK500TW information.

WHRXYEX: Acquires the current position of a main group axis.
WHRXYEX2: Acquires the current position of a sub group axis.

MEMO
- WHRXYEX2 cannot be used unless the sub robot is set.

NOTE
- The "XY coordinate system current position (including extended setting) acquisition" function is only available in software version 10.66 onwards.

SAMPLE

```
Command @?WHRXYEX
Response [POS]   13.44  206.06    0.00   83.24    0.00    0.00 1 0 -1[cr/lf]
```

4.17 Acquiring the tasks in RUN or SUSPEND status

Command format

```
@?TASKS[cr/lf]
```

Response format

```
n{,n{,{...}}}[cr/lf]
```

Values n: Task number .....1 to 8 (Task currently run or suspended)

Meaning Acquires the tasks in RUN or SUSPEND status.

SAMPLE

```
Command @?TASKS[cr/lf]
Response 1,3,4,6[cr/lf]
```

4.18 Acquiring the tasks operation status

Command format

```
@?TSKMON[cr/lf]
```

Response format

```
nfp,{nfp},{nfp},{nfp},{nfp},{nfp},{nfp},{nfp}[cr/lf]
```

Values
n : Line number being executed in each task..... 1 to 9999
f : Status of each task ..... R: RUN
U: SUSPEND
S: STOP
p : Priority level of each task ..... 17 to 47

Meaning Acquires the status of each task in order from Task 1 to Task 8.

SAMPLE

```
Command @?TSKMON[cr/lf]
Response 11R32,,43U32,,,,129R31,[cr/lf]
```

4.19 Acquiring the shift status

Command format

```
@?SHIFT[cr/lf]
```

Response format

```
m/s[cr/lf]
```

Values
m .....Shift number selected for main robot: 0 to 9
s.....Shift number selected for sub robot: 0 to 9

Meaning Acquires the shift status.

MEMO
- "s" is not displayed unless the sub robot is set.

SAMPLE

```
Command @?SHIFT[cr/lf]
Response 1[cr/lf]
```

4.20 Acquiring the hand status

Command format

```
@?HAND[cr/lf]
```

Response format

```
m/s[cr/lf]
```

Values
m ..... Hand number selected for main robot: 0 to 3
s ..... Hand number selected for sub robot: 4 to 7

Meaning Acquires the hand status.

MEMO
- "s" is not displayed unless the sub robot is set.

SAMPLE

```
Command @?HAND[cr/lf]
Response 1[cr/lf]
```

4.21 Acquiring the remaining memory capacity

Command format

```
@?MEM[cr/lf]
```

Response format

```
k/m[cr/lf]
```

Values
k ..... Remaining source area (unit: bytes)
m ..... Remaining object area (unit: bytes)

Meaning Acquires the remaining memory capacity.

SAMPLE

```
Command @?MEM[cr/lf]
Response 102543/1342[cr/lf]
```

4.22 Acquiring the emergency stop status

Command format

```
@?EMG[cr/lf]
```

Response format

```
k[cr/lf]
```

Values k ..... Emergency stop status / 0: normal operation, 1: emergency stop

Meaning Acquires the emergency stop status by checking the internal emergency stop flag.

SAMPLE

```
Command @?EMG[cr/lf]
Response 1[cr/lf]
```

4.23 Acquiring the error status by self-diagnosis

Command format

```
@?SELFCHK[cr/lf]
```

- When no error was found

Response format

```
OK[cr/lf]
```

- If an error occurred

Response format

```
m.n: "message" [cr/lf]
:
END [cr/lf]
```

Values m ..... Error group
n ..... Error category
"message" ..... Show error message.

Meaning Acquires the error status by self-diagnosis that checks for errors inside the controller.

SAMPLE

```
Command @?SELFCHK[cr/lf]
Response 12.1: Emg.stop on[cr/lf]
    END[cr/lf]
```

4.24 Acquiring the option slot status

Command format

```
@?OPSLOT[cr/lf]
```

Response format

```
OP.1 : <option board name> [cr/lf]
OP.2 : <option board name> [cr/lf]
OP.3 : <option board name> [cr/lf]
OP.4 : <option board name> [cr/lf]
```

Values

<option board name> .....Name of option board installed in the controller.
DIO_Nm.....DIO board with NPN specifications (m: board ID)
DIO_Pm..... DIO board with PNP specifications (m: board ID)
CCLnk..... CC-Link compatible board
D_Net..... DeviceNet compatible board
Profi..... Profibus compatible board
E_Net..... Ethernet compatible board
no board..... No option board is installed.
illegal board.....Incompatible board is installed.

Meaning Acquires the option slot status by checking the option boards.

SAMPLE

```
Command: @?OPSLOT[cr/lf]
Response: OP.1: DIO_N2[cr/lf]
          OP.2: DIO_N1[cr/lf]
          OP.3: no board[cr/lf]
          OP.4: CCLnk [cr/lf]
```

4.25 Acquiring various values

1. Acquiring the value of a numerical expression

NOTE

- Numerical expression values can be acquired on the following software versions.

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Command format

```
@? "numerical expression" [cr/lf]
```

Response format

```
"numerical value" [cr/lf]
```

Meaning Acquires the value of the specified numerical expression.
The numerical expression's value format is "decimal" or "real number".

SAMPLE 1

```
Command:   @?SQR(100*5)[cr/lf]
Response:  2.23606E01[c/lf]
```

SAMPLE 2

```
Command:   @?LOCX(WHERE)[cr/lf]
Response:  102054[cr/lf]
```

2. Acquiring the value of a character string expression

NOTE

- Numerical expression values can be acquired on the following software versions.

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Command format

```
@? "character string expression" [cr/lf]
```

Response format

```
"character string " [cr/lf]
```

Meaning Acquires the value (character string) of the specified character string expression.

SAMPLE

```
If A$="ABC" and B$="DEF"
Command:   @?A$+B$+"123"[cr/lf]
Response:  ABCDEF123[cr/lf]
```

3. Acquiring the value of a point expression

NOTE

- Numerical expression values can be acquired on the following software versions.
  RCX14x Ver. 8.64 onwards
  RCX22x Ver. 9.11 onwards

Command format

```
@? "point data expression" [cr/lf]
```

Response format

```
"point data" [cr/lf]
```

Meaning Acquires the value (point data) of the specified point expression.

SAMPLE

```
Command:   @?P1+WHRXY[cr/lf]
Response:  10.41   -1.60   52.15   3.00   0.00   0.00   0[cr/lf]
```

4. Acquiring the value of a shift expression

NOTE

- Numerical expression values can be acquired on the following software versions.
  RCX14x Ver. 8.64 onwards
  RCX22x Ver. 9.11 onwards

Command format

```
@? "shift expression" [cr/lf]
```

Response format

```
"shift data" [cr/lf]
```

Meaning Acquires the value (shift data) of the specified shift expression.

SAMPLE

```
Command:   @?s1[cr/lf]
Response:  25.00   12.60   10.00   0.00[cr/lf]
```

4.26 Data readout processing

Command format

```
@READ <readout file> [cr/lf]
```

Response format

```
Response output depends on the designated readout file.
```

NOTE

- For more information about files, refer to the earlier Chapter 11 "Data file description".

Values <readout file> ..... Designate a readout file name.

Meaning Reads out the data from the designated file.

Online commands that are input through the RS-232C port have the same meaning as the following command.

- SEND <readout file> TO CMU

Commands via Ethernet have the same meaning as the following command.

- SEND <readout file> TO ETH

NOTE

- The point comment separate format (PCn) is available on the following software versions.

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Type	Readout file name	Definition format	
		All	Separate file
User memory	All files	ALL	————
	Program	PGM	<bbbbbbbb>
	Point data	PNT	Pn
	Point comment	PCM	PCn
	Parameter	PRM	/cccccc/
	Shift definition	SFT	Sn
	Hand definition	HND	Hn
	Pallet definition	PLT	PLn
Variable, constant	Variable	VAR	ab...by
	Array variable	ARY	ab...by(x)
	Constant		"cc...c"
Status	Program directory	DIR	<<bbbbbbbb>>
	Parameter directory	DPM	————
	Machine reference	MRF	————
	Error history (log)	LOG	————
	Memory size	MEM	————
Device	DI port	DI()	DIn()
	DO port	DO()	DOn()
	MO port	MO()	MOn()
	TO port	TO()	TOn()
	LO port	LO()	LOn()
	SI port	SI()	SIn()
	SO port	SO()	SOn()
	SIW port	SIW()	SIWn()
	SOW port	SOW()	SOWn()
Others	File end code	EOF	————

a: Alphabetic character n: Number

b: Alphanumeric character or underscore (_) x: Expression (Array argument)

c: Alphanumeric character or symbol y: variable type

SAMPLE

```
Command: @READ PGM[cr/lf]
         @READ P100[cr/lf]
```

4.27 Data write processing

Command format

```
@WRITE <write file> [cr/lf]
```

Response format

```
Input request display    ▸ ***Input the data! [cr/lf]
After input is completed ▸ OK [cr/lf]
```

NOTE
- For more information about files, refer to the earlier Chapter 11 "Data file description".

Values <write file> ..... Designate a write file name.

Meaning Writes the data in the designated file.
Online commands that are input through the RS-232C port have the same meaning as the following command.
- SEND CMU TO <write file>

Commands via Ethernet have the same meaning as the following command.
- SEND ETH TO <write file>

MEMO
- At the DO, MO, TO, LO, SO, SOW ports, an entire port (DO(), MO(), etc.) cannot be designated as a WRITE file.
- Some separate files (DOn(), MOn(), etc.) cannot be designated as a WRITE file. For details, see Chapter 11 "Data File Details".

NOTE
- The point comment separate format (PCn) is available on the following software versions

  RCX14x Ver. 8.64 onwards

  RCX22x Ver. 9.11 onwards

Type	Write file name	Definition format	
		All	Separate file
User memory	All files	ALL	————
	Program	PGM	<bbbbbbbb>
	Point data	PNT	Pn
	Point comment	PCM	PCn
	Parameter	PRM	/cccccc/
	Shift definition	SFT	Sn
	Hand definition	HND	Hn
	Pallet definition	PLT	PLn
Variable, constant	Variable	VAR	ab...by
	Array variable	ARY	ab...by(x)
Device	DO port	————	DOn()
	MO port	————	MOn()
	TO port	————	TOn()
	LO port	————	LOn()
	SO port	————	SOn()
	SOW port	————	SOWn()

a: Alphabetic character n: Number
b: Alphanumeric character or underscore (_) x: Expression (Array argument)
c: Alphanumeric character or symbol y: variable type

SAMPLE 1

```
Command: @WRITE PRM[cr/lf]
         @WRITE P100[cr/lf]
```

4.28 Current torque value acquisition

Command format

```
@?CHKTRQ k[cr/lf]
@?CHKTRQ2 k[cr/lf]
```

Values k ... Axis setting (k = 1 to 6).

Response format

```
n[cr/lf]
```

Values n .....Current torque value (n = -100 to 100) of the specified axis.
* The plus/minus sign indicates the direction.

Meaning • Acquires the current torque value of the specified axis.
CHKTRQ: Acquires the current torque value of a main group axis.
CHKTRQ2: Acquires the current torque value of a sub group axis.

SAMPLE

```
Command:
Response:
```

NOTE

- The "@?CHKTRQ" and "@?CHKTRQ2" commands are available from the following software versions:
  RCX240 Ver. 10.65 onwards,
  RCX22x Ver. 9.36 onwards
- If the specified axis has been set to "no axis" in the system generation, or if that axis uses the YC-Link or a power gripper, a "5.37: Specification mismatch" error message displays and command execution is stopped.

5 Executing the robot language independently

5.1 Switching the program

Command format

```
@SWI <program name>[cr/lf]
```

Response format

```
OK[cr/lf] or LINEx,m,n: "message"
```

Values

OK ..... Program was switched correctly.
LINEx,m,n : "message" ..... Error occurred during compiling.
x ..... Line number on which error occurred
m ..... Error group
n ..... Error category
"message" ..... Error message

Meaning Switches the program.

- In AUTO mode, the program that is switched to will be compiled.
- In other modes, the program is only switched.

However, when "SEQUENCE" program is designated, a sequence object is created.

SAMPLE 1

```
In AUTO mode:
Command @SWI <TEST1>[cr/lf]
Response Line2,5.39:Illegal identifier[cr/lf]
```

SAMPLE 2

```
In other modes:
Command @SWI <TEST1>[cr/lf]
Response OK[cr/lf]
```

5.2 Other robot language command processing

Command format

```
@"robot language"[cr/lf]
```

Response format

```
OK[cr/lf] or ***Aborted
```

Values
OK ..... Command ended correctly.
***Aborted ..... An error occurred.

Meaning Robot language commands can be executed.
- Independently executable commands can only be executed.
- Command format depends on each command to be executed.

SAMPLE 1

```
Command @SET DO(20) [cr/lf]
Response OK[cr/lf]
```

SAMPLE 2

```
Command @MOVE P,P100,S=20[cr/lf]
Response OK[cr/lf]
```

6 Control codes

6.1 Interrupting the command execution

Command format

```
^C (=03H)
```

Response format

```
***Aborted
```

Meaning Interrupts execution of the current command.

SAMPLE

```
Command:   @MOVE P,P100,S=20[cr/lf]
           ^C
Response:  ***Aborted[cr/lf]
```