Connection with personal computer[RS-232C/Ethernet]

Revision V: Nov 15, 2018
Revision U: May 29, 2018
Revision T: Nov 30, 2016
Revision S: Oct 21, 2015
Revision R: Oct 9, 2014
Revision Q: Jun 16, 2014
Revision P: Aug 28, 2013
Revision N: Oct 11, 2011
Revision M: Mar 26, 2009
Revision L: Oct 23, 2008
Revision K: Feb 15, 2005
Revision J: Jul 7 , 2004
Revision H: Feb 10, 2004
Revision G: Jan 23, 2004
Revision F: Aug 19, 2003

-NOTICE-
This is a preliminary printing.

Contents

1 INTRODUCTION........5

2 CONFIGURATION........5

3 PROTOCOL........6

3.1 NON-PROCEDURAL........6
3.2 PROCEDURAL (FOR RS-232)........6
3.3 PROTOCOL FORMAT........8
3.3.1 Transmit data........8
3.3.2 Receive data........8
3.3.3 Communication interval........8

4 COMMAND EXAMPLE........10

4.1 CONNECT AND DISCONNECT........10
4.2 JOG OPERATION........10
4.3 PROGRAM EDIT........11
4.4 TEACHING........11
4.5 STEP RUNNING........12
4.6 RUN........12
4.7 VARIABLE MONITOR AND WRITE........13

5 COMMAND SPECIFICATION........14

5.1 COMMAND LISTS........14
5.2 DETAILS EXPLANATION OF COMMAND WORDS........17
5.2.1 OPEN= (Communication open)........17
5.2.2 CLOSE (Communication close)........18
5.2.3 CNTL (Operation enable or disable)........18
5.2.4 LOAD= (Program open)........19
5.2.5 SAVE (Program save and close)........19
5.2.6 NEW (Program quit and close)........20
5.2.7 EDATA (Edit program)........20
5.2.8 EMDAT (More line edit program)........21
5.2.9 EDINS (Insert program)........21
5.2.10 VAL= (Variable data write)........22
5.2.11 LISTI (Program list read)........23
5.2.12 LISTL (Program more list read)........23
5.2.13 LISTP (Program position read)........24
5.2.14 VTYPRD (Variable type read)........25
5.2.15 LISTCNT (Count program lines)........25
*5.2.16 EXEC * (Direct execution)........26*
*5.2.17 EXEC2= * (Direct execution)........26*
*5.2.18 STEP * (Step execution)........27*
5.2.19 ECLR (Clear program contents)........27
5.2.20 DELETE (Delete program lines)........28
5.2.21 RENUM (Renumber)........28
5.2.22 PDIR (Program directory)........29
5.2.23 FDIR (File directory)........30
5.2.24 FCHECK (File check)........31
5.2.25 FPATH= (File path)........31
5.2.26 FCOPY (File copy)........32
5.2.27 FDEL (File delete)........32
5.2.28 FRENAME (File rename)........33
5.2.29 FATTRIB (File attribute)........33
5.2.30 FINIT (File init)........34
5.2.31 FOPEN (File block open)........34
5.2.32 FCLOSE (File block close)........35
5.2.33 FREAD (Block read)........35

5.2.34 FWRITE (Block write)..........36
5.2.35 EFREE (Read file size)..........36
5.2.36 ESEARCH (Search string)..........37
*5.2.37 PRGLOAD= * (Program load)..........37*
*5.2.38 PRG<UP/DOWN> * (Program select)..........38*
5.2.39 PRGRD (Execution program name read)..........38
*5.2.40 LINENO= * (Execution line number change)..........39*
5.2.41 LINENO (Execution Line number read)..........39
5.2.42 LINERD (Execution Line contents)..........40
5.2.43 LINESRD (Execution more Line contents)..........40
*5.2.44 SRV＜ON/OFF> * (Servo ON or OFF)..........41*
*5.2.45 OVRD= * (OP override change)..........42*
5.2.46 OVRD (OP override read)..........42
*5.2.47 RUN * (Program start)..........43*
5.2.48 STOP (STOP)..........43
5.2.49 STOP<ON/OFF> (STOP ON or OFF)..........44
5.2.50 CSTOP (Cycle STOP)..........44
5.2.51 RSTALRM (Error reset)..........45
5.2.52 SLOTINIT (All program reset)..........45
5.2.53 RSTPRG (Each program reset)..........45
5.2.54 RSTIO (Output signal reset)..........46
5.2.55 MLOCK<ON/OFF> (Machine lock ON or OFF)..........46
5.2.56 HND<ON/OFF> (HAND open or close)..........47
*5.2.57 ALIGN * (Aligning the hand)..........47*
*5.2.58 MOVSP * (MOVE safe position)..........48*
*5.2.59 JOG * (JOG operation)..........48*
*5.2.60 JOG * (JOG operation for Multifunctional Electric Hand)..........49*
5.2.61 LS<ON/OFF> (Limit switch ON or OFF)..........50
5.2.62 AUE<ON/OFF> (Program start enable or disable)..........50
5.2.63 ATENA (Status can be start)..........51
5.2.64 BRKPTSET (Set breakpoint)..........51
5.2.65 BRKPTCLR (Delete breakpoint)..........52
5.2.66 BRKPTGET (List breakpoint)..........52
5.2.67 TOOLSET (Set Tool number)..........53
5.2.68 TOOLRD (Read Tool number)..........54
5.2.69 SAFE= (Low speed mode)..........54
5.2.70 STATE (Read run status)..........55
5.2.71 DSTATE (Read stop status)..........56
5.2.72 CALIB (Install status)..........57
5.2.73 IOSIGNAL (Input and output signal read)..........57
5.2.74 IN (Input signal read)..........58
5.2.75 OUT (Output signal read)..........58
5.2.76 OUT= (Output signal write)..........59
5.2.77 DIN (CC-Link's input register data read)..........59
5.2.78 DOUT (CC-Link's output register data read)..........60
5.2.79 DOUT= (CC-Link's output register data write)..........60
5.2.80 INDMY (Set pseudo input)..........61
5.2.81 INSET (Reset pseudo input)..........61
5.2.82 IN= (Write pseudo input data)..........62
5.2.83 DIN= (Write pseudo input register)..........62
5.2.84 STPSIG (Stop signal read)..........63
5.2.85 HNDSTS (Hand output signal read)..........63
5.2.86 USERAREASTS (User specified area read)..........64
5.2.87 JPOS,PPOS,XPOS,RPOS (Current position read)..........64
5.2.88 GJPOS,GPPOS (Destination position read)..........65
5.2.89 TIME (Time read)..........66
5.2.90 TIME= (Time change)..........67
5.2.91 PTIME (Hour meter read)..........67

5.2.92 PTIMEDEL= (Hour meter clear)........68
5.2.93 CYCLETIME (Cycle time read)........69
5.2.94 CYCLECLR (Cycle time clear)........69
5.2.95 ERROR (Error number read)........70
5.2.96 ERRORMES (Error contents read)........70
5.2.97 ERRORLOG (Error history read)........71
5.2.98 ERRLOG2= (Error history reading. / Error details number narrowing seeing.)........72
5.2.99 ERRLOGCLR (Error history clear)........73
5.2.100 ERRSUM (Error summary)........73
5.2.101 ERRSUM2= (error summary)........74
5.2.102 ERRSUMCLR (Clear error summary)........75
5.2.103 SUMDATE (Date when error logging function began)........76
5.2.104 VAL (Variable data read)........76
5.2.105 VALS (More Variable data read)........77
5.2.106 GVAL (Global variable data read)........78
5.2.107 GVALS (More Variable data read)........78
5.2.108 HOT (Variable data write)........79
5.2.109 OPNUMRD (Option slot number read)........80
5.2.110 OPSTSRD (Option information read)........80
5.2.111 THMRD (Controller temperature read)........81
5.2.112 ETEMP (Encoder temperature read)........81
5.2.113 EMISS (Encoder miscount read)........82
5.2.114 SRVENC (Servo encoder read)........82
5.2.115 SRVDRP (Servo droop read)........83
5.2.116 SRVSPD (Servo speed read)........84
5.2.117 SRVCUR (Servo current read)........84
5.2.118 SRVLCR (Servo load current)........86
5.2.119 SRVVOL (Servo voltage)........86
5.2.120 SVMONRST= (Reset servo monitor maximum)........87
5.2.121 RAREAD= (Read serial number)........87
5.2.122 PRMINIT (Parameter initial)........88
5.2.123 PNR (Parameter read)........88
5.2.124 PRM= (Parameter write)........89
5.2.125 PAR (Parameter read)........89
5.2.126 PAW= (Parameter write)........90
5.2.127 PAW2= (Parameter write (need to reboot.))........90
5.2.128 PRMUNDO (Parameter undo)........91
5.2.129 PRM= (Read change parameter list)........92
5.2.130 KEYWD (Keyword input)........92
5.2.131 SLOTRD (Slot table read)........93
5.2.132 SLOTSET (Slot table write)........93
5.2.133 ENCBATTM (Battery remain time)........94
*5.2.134 BREAKON * (Release brake)........94*
*5.2.135 BREAKONF * (Release brake)........95*
*5.2.136 HOME * (Setting the origin)........96*
*5.2.137 AXDATINST * (Additional axis add for DATINST and DATRD)........96*
*5.2.138 DATINST * (Data input origin set)........97*
5.2.139 DATRD (Data input origin set)........97
5.2.140 RSTPWR (Reset power)........98
5.2.141 RPWRCHK= (Reset power check)........98
5.2.142 MFTIME= (Maintenance forecast date)........99
5.2.143 MFRST= (Maintenance forecast reset)........100
5.2.144 MFFCST= (Maintenance forecast read)........100

1 Introduction

In this book, the communication specification to do peripherals such as personal computers and the communication between robot controllers is described.

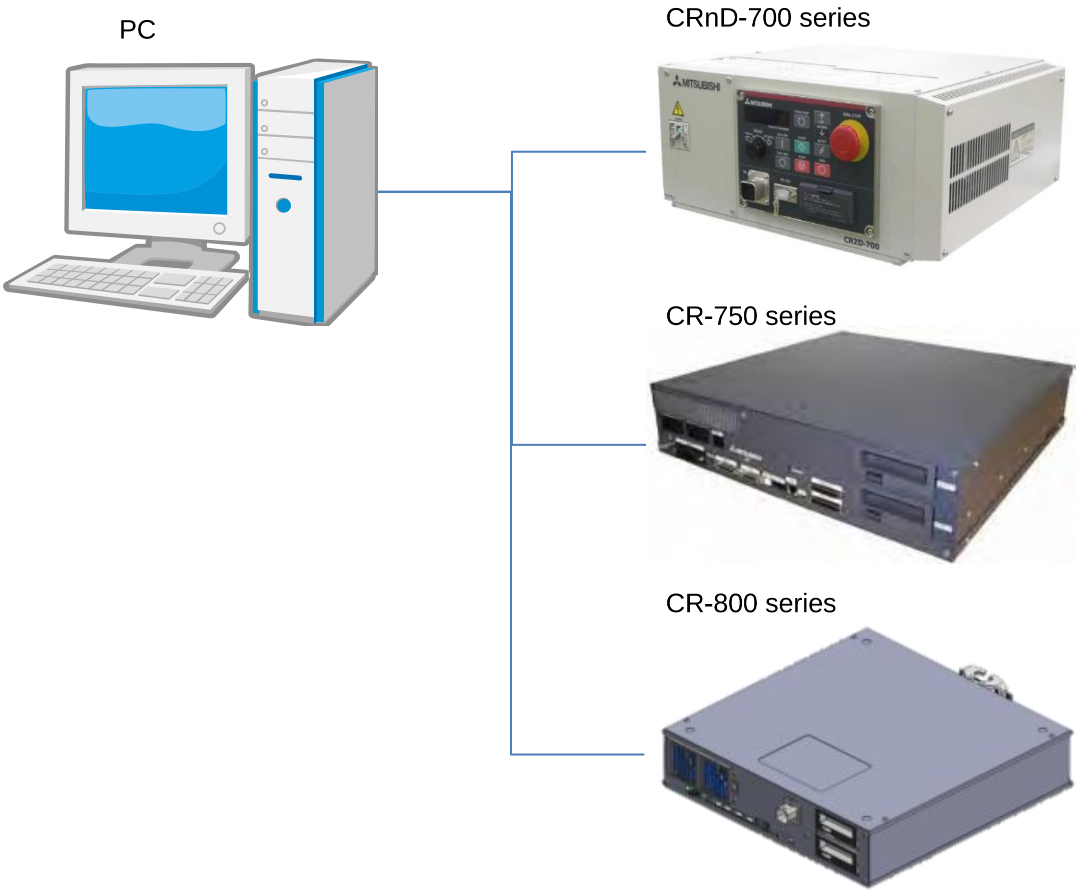
The robot controller reacts only to the command demand by peripherals, and returns the answer of the command. Therefore, the thing to lodge the command demand from the robot controller voluntarily is not done.

The kind of the protocol is as follows.

(1) Non-Procedural

(2) Procedural

2 Configuration



3 Protocol

3.1 Non-Procedural

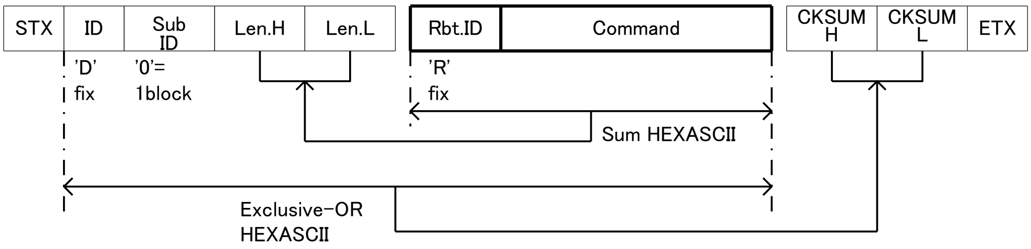
[Data syntax]

The terminal character can be set according to the parameter. (CR or CR+LF)

Maximum command size is 255 bytes.

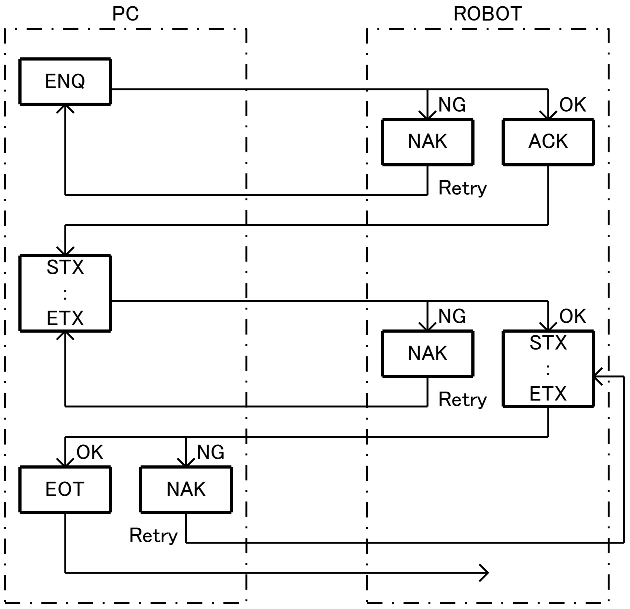
3.2 Procedural (for RS-232)

[Data syntax]



Maximum command size is 255 bytes.

[Communication procedure]



[State chart]

	ID	State Status	ENQ	Cmd. OK	Cmd. ERR	ACK	NAK	EOT	Time out	Retry error
Receive	1	Wait ENQ	State 2	State 3	State 3	State 3	State 3	State 3	---	---
	2	Send ACK and Wait	State 3	State 2	State 3	State 3	State 3	State 1	State 1	---
	3	Send NAK and Wait	State 3	State 2	State 3	State 3	State 3	State 1	State 1	State 1
Send	4	Send ENQ Wait ACK	State 6	State 6	State 6	State 5	State 4	State 6	---	---
	5	Send Command Wait ACK	State 6	State 6	State 6	State 5	State 5	State 6	---	---
	6	Send NAK Wait ACK	State 6	State 6	State 6	State 5	State 6	State 6	---	---

3.3 Protocol format

3.3.1 Transmit data

[<Robot No.>];[<Slot No>];<Command><Argument>

<Robot No.>: The robot number to be operated is specified. (0, 1, 2 or 3)
It is possible to omit it. Omitting it is 1.
There are commands that influences all robots if 0 is specified.

< Slot No >: The slot number to be operated is specified. (0, 1 - 33)
Parameter “TASKMAX” is a number of task slots used by the multitask. When the program is edited from the PC, the edit slot is used. The slot number of the edit slot is parameter TASKMAX+1. In this case, because an initial value of TASKMAX is 8, the number of the edit slot is 9.
It is possible to omit it. Omitting it is 1.
There are commands that influences all slots if 0 is specified.

< Command >< Argument >: It differs in each command, and refer to the explanation of each command, please.

<Argument>: There are ‘;’ or VT(0b) delimiters for argument.

Example1) 1;1; DELETE10;50 … Command DELETE

1	;	1	;	D	E	L	E	T	E	1	0	;	5	0	
31	3B	31	3B	44	45	4C	45	54	45	31	30	3B	35	30	

Example2) 1;1;EMDAT10 MOV P10b20 MOV P2 … EMDAT Command

1	;	1	;	E	M	D	A	T	1	0		M	O	V	
31	3B	31	3B	45	4D	44	41	54	31	30	20	4D	4F	56	20
P	1	VT	2	0		M	O	V		P	2				
50	31	0B	32	30	20	4D	4F	56	20	50	32				

3.3.2 Receive data

QoK<Answer>
or
QeR<Error No.>

< Answer >: It differs in each command, and refer to the explanation of each command, please.
There are ‘;’ or VT(0b) delimiters for answer.

Example1) QoK09060;17;1 … DSTATE answer

Q	O	K	0	9	0	6	0	;	1	7	;	1			
51	6F	4B	30	39	30	36	30	3B	31	37	3B	31			

Example2) QoK10 MOV P10b20 MOV P2 … LISTL answer

Q	O	K	1	0	M	O	V		P	1	VT	2	0		M
51	6F	4B	31	30	4D	4F	56	20	50	31	0B	32	30	20	4D
O	V		P	2											
4F	56	20	50	32											

< Error No.>: It replies the error number when the command cannot be executed. Please refer to the troubleshooting manual of the robot for the number.

3.3.3 Communication interval

When receiving the next data after receiving the received data while operating the robot program, please communicate at an interval (recommended value 100 ms or more).

Example)

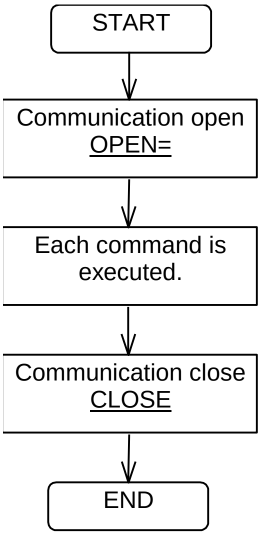
1;1;DSTATE
QoK09060;17;1
(Interval of 100 ms or more)
1;1;DSTATE

QoK09060;17;1
:

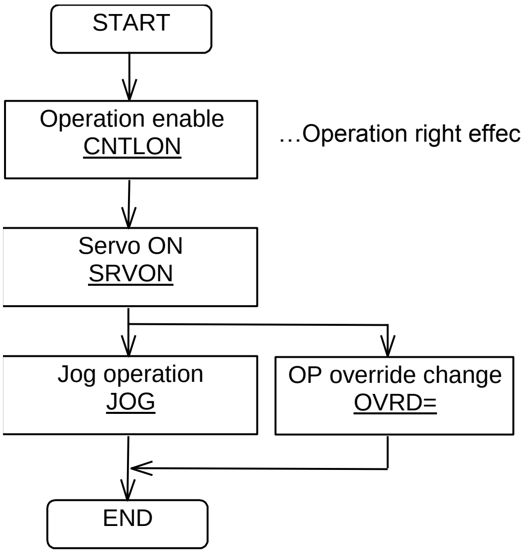
4 Command example

Two or more commands are combined and the example of using the command is shown about the operation used.

4.1 Connect and disconnect



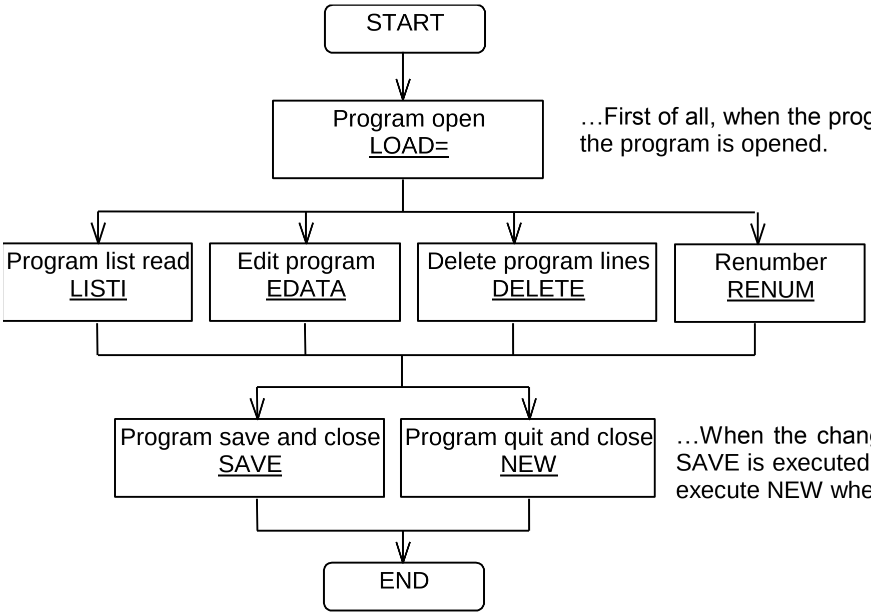
4.2 JOG operation



…Operation right effective

…The robot is made to work by the jog sending and the override setting.

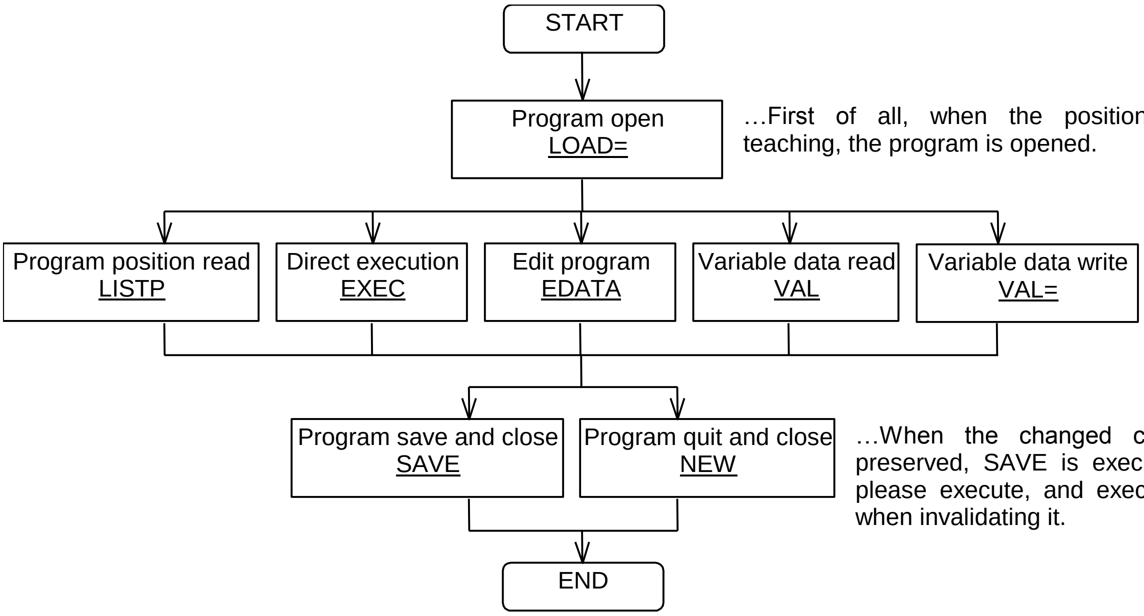
4.3 Program edit



…First of all, when the program is edited, the program is opened.

…When the changed content is preserved, SAVE is executed, and please execute, and execute NEW when invalidating it.

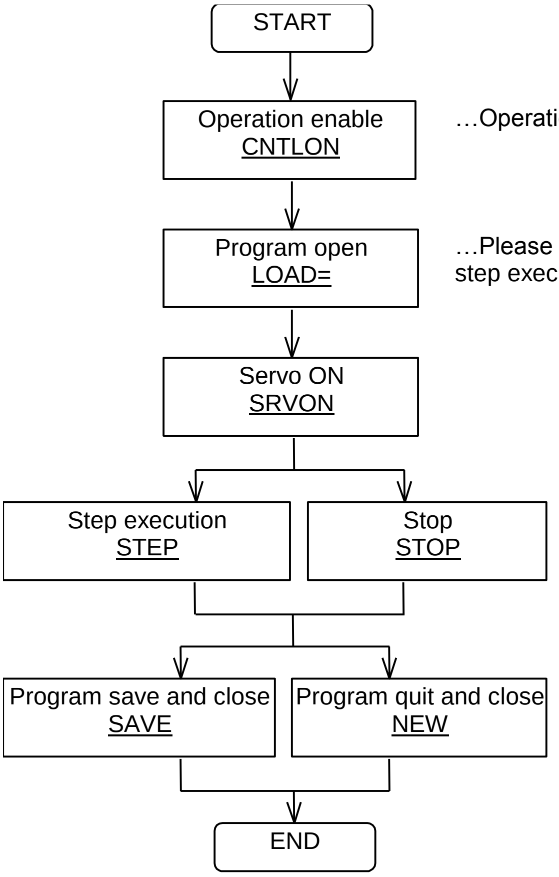
4.4 Teaching



…First of all, when the position is teaching, the program is opened.

…When the changed content is preserved, SAVE is executed, and please execute, and execute NEW when invalidating it.

4.5 STEP running

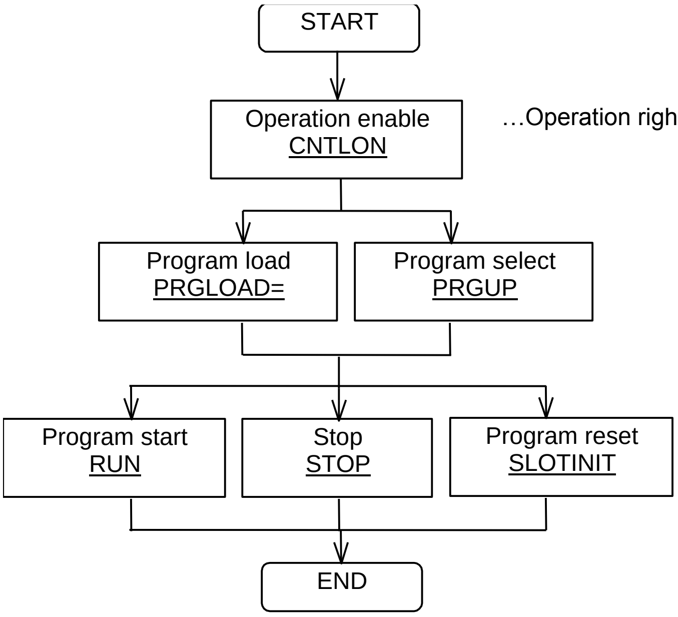


…Operation right effective

…Please open the program when the step execution.

…When the changed content is preserved, SAVE is executed, and please execute, and execute NEW when invalidating it.

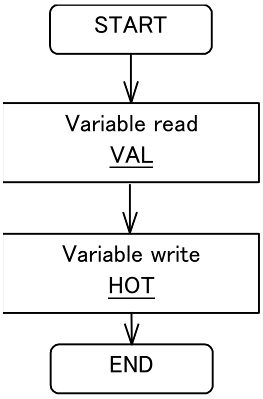
4.6 RUN



…Operation right effective

* When the program pausing, neither the program loading(PRGLOAD=) nor the program selection(PRGUP) can be done. Please do after doing program reset (SLOTINIT).

4.7 Variable monitor and write



5 Command specification

5.1 Command lists

Class	Functions	Command * is enable commands	Only FR/F/SD/SQ
Communication	Communication open	OPEN=	
	Communication close	CLOSE	
	Operation enable or disable	CNTL<ON/OFF>	
Program edit	Program open	LOAD=	
	Program save and close	SAVE	
	Program quit and close	NEW	
	Edit program	EDATA	
	More line edit program	EMDAT	
	Insert program	EDINS	o
	Variable data write	VAL=	
	Program list read	LISTI	
	Program more list read	LISTL	
	Program position read	LISTP	
	Variable type read	VTYPRD	
	Count program lines	LISTCNT	
	Direct execution	EXEC *	
	Direct execution	EXEC2= *	
	Step execution	STEP *	
	Clear program contents	ECLR	
	Delete program lines	DELETE	
	Renumber	RENUM	
File operation	Program directory	PDIR	
	File directory	FDIR	
	File check	FCHECK	
	File path	FPATH=	o
	File copy	FCOPY	
	File delete	FDEL	
	File rename	FRENAME	
	File attribute	FATTRIB	
	File init	FINIT	
	File block open	FOPEN	
	File block close	FCLOSE	
	Block read	FREAD	
	Block write	FWRITE	
	Read file size	EFREE	
	String search	ESEARCH	
Running	Program load	PRGLOAD= *	
	Program select	PRG<UP/DOWN> *	
	Execution program name read	PRGRD	
	Execution line number change	LINENO= *	
	Execution Line number read	LINENO	
	Execution Line contents	LINERD	
	Execution more Line contents	LINESRD	
	Servo ON or OFF	SRV<ON/OFF> *	
	OP override change	OVRD= *	
	OP override read	OVRD	
	Program start	RUN *	
	STOP	STOP	
	STOP ON or OFF	STOP<ON/OFF>	

	Cycle STOP	CSTOP	
	Error reset	RSTALRM	
	All program reset	SLOTINIT	
	Each program reset	RSTPRG	
	Output signal reset	RSTIO	
	Machine lock ON or OFF	MLOCK<ON/OFF>	
	HAND open or close	HND<ON/OFF>	
	Aligning the hand	ALIGN *	
	MOV safe position	MOVSP *	
	JOG operation	JOG *	
	Limit switch ON or OFF	LS<ON/OFF>	
	Program start enable or disable	AUE<ON/OFF>	
	Status can be start	ATENA	
	Set breakpoint	BRKPTSET	o
	Delete breakpoint	BRKPTCLR	o
	List breakpoint	BRKPTGET	o
	Set Tool number	TOOLSET$_{Q}$	o
	Read Tool number	TOOLRD$_{Q}$	o
	Low speed mode	SAFE=	o
Monitor	Read run status	STATE	
	Read stop status	DSTATE	
	Install status	CALIB	
	Input and output signal read	IOSIGNAL	
	Input signal read	IN	
	Output signal read	OUT	
	Output signal write	OUT=	
	CC-Link's input register data read	DIN	
	CC-Link's output register data read	DOUT	
	CC-Link's output register data write	DOUT=	
	Set pseudo input	INDMY	
	Reset pseudo input	INSET	
	Write pseudo input data	IN=	
	Write pseudo input register	DIN=	
	Stop signal read	STPSIG	
	Hand output signal read	HNDSTS	
	User specified area read	USERAREASTS	
	Current position read	JPOS, PPOS, XPOS, RPOS	
	Destination position read	GJPOS, GPPOS$_{Q}$	o
	Time read	TIME	
	Time change	TIME=	
	Hour meter read	PTIME	
	Hour meter clear	PTIMEDEL=	
	Cycle time read	CYCLETIME	
	Cycle time clear	CYCLECLR	
	Error number read	ERROR	
	Error contents read	ERRORMES	
	Error history read	ERRORLOG	
	Error history read 2	ERRLOG2=	
	Error history clear	ERRORLOGCLR	
	Error summary	ERRSUM	
	Error summary 2	ERRSUM2=	
	Clear error summary	ERRSUMCLR	
	Date when error logging function began	SUMDATE	
	Variable data read	VAL	

	More Variable data read	VALS	
	Global variable data read	GVAL	
	More Global variable data read	GVALS	
	Variable data write	HOT	
	Option slot number read	OPNUMRD	
	Option information read	OPSTSRD	
	Controller temperature read	THMRD	
	Encoder temperature read	ETEMP	
	Encoder miscount read	EMISS$_Q$	o
	Servo encoder read	SRVENC$_Q$	o
	Servo droop read	SRVDRP$_Q$	o
	Servo speed read	SRVSPD$_Q$	o
	Servo current read	SRVCUR$_Q$	o
	Servo load current read	SRVLCR$_Q$	o
	Servo voltage read	SRVVOL$_Q$	o
	Reset servo monitor maximum	SVMONRST=$_Q$	o
	Read serial number	RAREAD=$_Q$	o
Maintenance	Parameter initial	PRMINIT	
	Parameter read	PNR	
	Read change parameter list	PRM=	
	Parameter compulsion read	PAR	
	Parameter compulsion write	PAW=	
	Parameter write (need to reboot)	PAW2=	
	Parameter undo	PRMUNDO	
	Read change parameter list	PRM=	
	Keyword input	KEYWD	
	Slot table read	SLOTRD	
	Slot table write	SLOTSET	
	Battery remain time	ENCBATTM	
	Release brake	BREAKON *	
	Release brake (No operation right required)	BREAKONF	
	Setting the origin	HOME *	
	Additional axis add for DATINST and DATRD	AXDATINST *	
	Data input origin set	DATINST *	
	Data input origin read	DATRD	
	Reset power	RSTPWR	o
	Reset power check	RPWRCHK=$_Q$	o
	Maintenance forecast date	MFTIME=$_Q$	
	Maintenance forecast reset	MFRST=$_Q$	
	Maintenance forecast read	MFFCST=$_Q$	

5.2 Details explanation of command words

5.2.1 OPEN= (Communication open)

[Function]

Communication open. The commands sent most first when communicating from the peripheral equipment such as personal computers.

[Format]

```
OPEN=<Device name>[;<LANG>]
```

<Device name>	An arbitrary name is specified in the alphanumeric character.
<LANG>	Specify the display language of the error message.

[Answer]

```
QoK<XYZ axes pattern>;<JOINT axes pattern>;<Structure flag1>,< Structure flag2>;
    <JOG speed>;
    <Program ext.>;<Parameter ext.>;<Robot type>;<Controller>;<Series>;<DATE>;
    <Version>;<Language>;<Copyright>;<Robot info.>;<Serial No>;
    <Multitask No>
```

<XYZ axes pattern>	XYZ axes pattern of HEX number. (00～FF)
<JOINT axes pattern>	JOINT axes pattern of HEX number. (00～FF)

axis pattern

```
00000000B    XYZ / JOINT
_______1      X     J1
______1_      Y     J2
_____1__      Z     J3
____1___      A     J4
___1____      B     J5
__1_____      C     J6
_1______      L1    J7
1_______      L2    J8
```

< Structure flag1>	Default structure flag1
< Structure flag2>	Default structure flag2
<JOG speed>	The JOG speed of the HEX number is 7 pieces. (00～FF)
<Program ext.>	Program extension (MB4, MB5 or MB6)
<Parameter ext.>	Parameter extension (PRM)
<Robot type>	Robot name
<Controller>	”CRn-5xx” , “CRn-7xx” , “CR75x” or “CR800”
<Series>	”MELFA” (fix)
<DATE>	Release data
<Version>	System software version
<Language>	JPN(Japanese) or ENG(English)
<Copyright>	Copyright message
<Robot info.>	Multi robot information

<Serial No>	1 (fix)
<Multitask No>	Value of parameter ”TASKMAX”

[Reference command]

```
1;1;OPEN=USERTOOL
```

```
QoK3F;3F;7,0;3,5,A,1E,32,46,64;MB5;PRM;RV-4A;CRn-5xx;MELFA;03-11-19;Ver.J4;ENG;
COPYRIGHT(C)1999-2003 MITSUBISHI ELECTRIC CORPORATION ALL RIGHTS
  RESERVED;1;1;8;
```

[Related commands]

CLOSE

5.2.2 CLOSE (Communication close)

[Function]

Communication close. The command sent when the communication is ended from the peripheral equipment such as personal computers.

[Format]

```
CLOSE
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;CLOSE
```

```
QoK
```

[Related commands]

OPEN=

5.2.3 CNTL (Operation enable or disable)

[Function]

Operation enable or disable. When the command which needs the operation right such as Program start, Servo ON and more is used, the operation right should be made effective.

[Format]

```
CNTL<ON/OFF>
```

<ON/OFF> Select ON or OFF

[Answer]

```
QoK
```

[Reference command]

```
1;1;CNTLON
QoK
```

[Related commands]

5.2.4 LOAD= (Program open)

[Function]

Open the program for edit.

[Format]

```
LOAD=<Program name>
```

<Program name> Edit program name

[Answer]

```
QoK
```

[Reference command]

```
1;1;LOAD=100
QoK
```

[Related commands]

SAVE, NEW

5.2.5 SAVE (Program save and close)

[Function]

The content of the edit is preserved and the program is closed.

[Format]

```
SAVE
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;SAVE
QoK
```

[Related commands]

LOAD=, NEW

5.2.6 NEW (Program quit and close)

[Function]

The program is closed annulling the content of the edit.

[Format]

```
NEW
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;NEW
QoK
```

[Related commands]

LOAD=, SAVE

5.2.7 EDATA (Edit program)

[Function]

The line and the position are registered in the program. It is effective in the edit slot.

[Format]

```
EDATA<Line or Position>
```

<Line or Position> Line data and positional data are specified.

[Answer]

```
QoK
or
```

```
Qer<Error no>;<Error character>
```

<Error no>	Error number when registering
<Error character>	Character position of cause of error

[Reference command]

```
1;1;EDATA10 MOV P1
QoK
```

[Related commands]

EMDAT

5.2.8 EMDAT (More line edit program)

[Function]

More line and position are registered in the program. It is effective in the edit slot.

[Format]

```
EMDAT<Line or Position>[␋<Line or Position>…]
```

<Line or Position>	Line data and positional data are specified.

[Answer]

```
QoKor
Qer<Error no>;<Error line no>;<Error character>
```

<Error no>	Error number when registering
<Error line no>	The error line is piece how many.
<Error character>	Character position of cause of error

[Reference command]

```
1;1;EMDAT10 MOV P1␋20 MOV P2␋30 MOV P3
QoK
```

[Related commands]

EDATA

5.2.9 EDINS (Insert program)

[Function]

The line is inserted in the program. It is effective in the edit slot.

[Format]

```
EDINS=<Insert line>;<Line data>
```

<Insert line>	Insert line number
<Line data>	Line data are specified.

[Answer]

```
QoK
or
Qer<Error no>;<Error character>
```

<Error no>	Error number when registering
<Error character>	Character position of cause of error

[Reference command]

```
1;1;EDINS=3;MOV P1
QoK
```

[Related commands]

EMDAT

5.2.10VAL= (Variable data write)

[Function]

The value of the variable is changed. It is effective in the edit slot.

[Format]

```
VAL=<Variable name>=<Value>
```

<Variable name>	Variable name
<Value>	Changed value

[Answer]

```
QoK
```

[Reference command]

```
1;9;VAL=M1=3
QoK
```

[Related commands]

5.2.11LISTI (Program list read)

[Function]

The line data is read from the program. It is effective in the edit slot.

[Format]

```
LISTI<Line posi.>
```

<Line posi.>　The read line is specified.
TOP: Top line
END: Bottom line
+1:Next line
-1:Previous line
Line number: Specified line

[Answer]

```
QoK[<Line contents>]
```

<Line contents>　Specified line data
Only QoK is red when there is no specified line.

[Reference command]

```
1;1;LISTITOP
QoK10 MOV P1
```

[Related commands]

LISTL

5.2.12LISTL (Program more list read)

[Function]

More line data is read from the program. It is effective in the edit slot. .

[Format]

```
LISTL<Line pos.>
```

<Line posi.>　The read line is specified.
TOP: Top line
END: Bottom line
+1: Next line
-1: Previous line
Line number: Specified line

[Answer]

```
QoK<Line contents>[0b<Line contents>…];<Count>;<Continue>
```

<Line contents>	Specified line data Only QoK is red when there is no specified line.
<Count>	The read number of lines.
<Continue>	1: There is a continuation line. 0: Continuation line none

[Reference command]

```
1;1;LISTLTOP
QoK10 MOV P10b20 MOV P20b30 M1=1;3;0
```

[Related commands]

LISTI

5.2.13LISTP (Program position read)

[Function]

Positional data is read from the program. It is effective in the edit slot.

[Format]

```
LISTP<Position posi.>
```

<Position posi.>	The read position is specified. TOP: Top position END: Bottom position +1: Next position -1: Previous position Position name: Specified position

[Answer]

```
QoK<Position contents>
```

<Position contents>	Specified Positional data Only QoK is red when there is no specified position.

[Reference command]

```
1;1;LISTPTOP
QoKP1=(+400.13,+0.00,+644.62,+180.00,+0.00,+180.00)(7,0)
```

[Related commands]

5.2.14VTYPRD (Variable type read)

[Function]

The type of the variable is read. It is effective in the edit slot.

[Format]

```
VTYPRD<Variable name>
```

<Variable name>	Variable name

[Answer]

```
QoK<Type>
```

<Type>	Variable type. (M:Integer, C:Character, P:Position, J:Joint) Only QoK is red when there is no specified variable.

[Reference command]

```
1;9;VTYPRDP1
QoKP
```

[Related commands]

5.2.15LISTCNT (Count program lines)

[Function]

The number of lines from the start line to the end line are counted.

[Format]

```
LISTCNT<Start line>;<End line>
```

<Start line>	Counted start line number
<End line>	Counted end line number

[Answer]

```
QoK<Number >
```

< Number >	Number of lines

[Reference command]

```
1;1;LISTCNT10;50
QoK0005
```

[Related commands]

5.2.16EXEC * (Direct execution)

[Function]

The instruction is executed directly.

Returns the response when you accepted.

[Format]

```
EXEC<Instruction>
```

<Instruction> Instruction of MELFA-BASICⅣtoⅥ or MOVEMASTER commands.

[Answer]

```
QoK
```

[Reference command]

```
1;9;EXECMOV P1
QoK
```

[Related commands]

EXEC2=

5.2.17EXEC2= * (Direct execution)

[Function]

The instruction is executed directly.

Returns the response when you work completed.

[Format]

```
EXEC2=<Instruction>
```

<Instruction> Instruction of MELFA-BASICⅣtoⅥ or MOVEMASTER commands.

[Answer]

```
QoK
```

[Reference command]

```
1;9;EXEC2=MOV P1
QoK
```

[Related commands]

EXEC

5.2.18STEP * (Step execution)

[Function]

The step is executed. It is effective in the edit slot.

[Format]

```
STEP<Method>
```

<Method> 1:Execute 1 step forward
R: Execute continuous step forward
B: Execute 1 step backward
S: Execute step stop

[Answer]

```
QoK
```

[Reference command]

```
1;9;STEP1
QoK
```

[Related commands]

5.2.19ECLR (Clear program contents)

[Function]

Clear program contents. It is effective in the edit slot.

[Format]

```
ECLR
```

[Answer]

```
QoK
```

[Reference command]

```
1;9;ECLR
QoK
```

[Related commands]

DELETE

5.2.20DELETE (Delete program lines)

[Function]

The line from the start line to the end line is deleted.

[Format]

```
DELETE<Start line>;<End line>
```

<Start line>	Deleted start line number
<End line>	Deleted end line number

[Answer]

```
QoK
```

[Reference command]

```
1;1;DELETE10;50
QoK
```

[Related commands]

ECLR

5.2.21RENUM (Renumber)

[Function]

Renumber for program. It is effective in the edit slot.

[Format]

```
RENUM<New line no>;<Old start line no>;<Step>;<Old end line no>
```

<New line no>	New line number. (0:10)
<Old start line no>	Old start line number. (0:Top)
<Step>	Step of line number. (0:10)
<Old end line no>	Old end line number. (0:Bottom)

[Answer]

```
QoK
```

[Reference command]

```
1;9;RENUM100;0;0;0
QoK
```

[Related commands]

5.2.22 PDIR (Program directory)

[Function]

Read program directories.

[Format]

```
PDIR<Posi.>
```

<Posi.>	The place read is specified. TOP: First information Value: Information at specified position

[Answer]

```
QoK<Program name>;<Size>;<DATE><TIME>;<Count>;<Remain size>;<Attrib>;;
  <Line count>;<Position Count>;<Operation time>;<Latest tact>;<Avg.tact>;<Cycle count>
or
QoK<Program name>;<Size>;<DATE><TIME>;<Attrib>;;<Line count>;<Position Count>;
  <Running>;<Latest tact>;<Avg.tact>;<Cycle count>
```

<Program name>	Program name
<Size>	Number of bytes used by program.
<DATE>	Last update date. (yy-mm-dd)
<TIME>	Last update time (hh:mm:ss)
<Count>	Program count in robot.
<Remain size>	Remain size of file system.
<Attrib>	Program attrib
<Line count>	Line count in program
<Position count>	Position count in program
<Operation time>	Operation time. (msec)
<Latest.tact>	Latest of tact time. (msec)
<Avg.tact>	Average of tact time. (msec)
<Cycle count>	Operation count

[Reference command]

```
1;1;PDIRTOP
QoK1.MB5;235;04-02-0519:31:28;34;105984;12;;2;0;115850;60;60;1903
1;1;PDIR1
QoK1.MB5;235;04-02-0519:31:28;12;;2;0;115850;60;60;1903
```

[Related commands]

FDIR

5.2.23FDIR (File directory)

[Function]

Read file directories.

[Format]

```
FDIR<Posi.>
```

<Posi.>	The place read is specified. TOP: First information Value: Information at specified position <*.*: First information for all files

[Answer]

```
QoK<File name>;<Size>;<DATE><TIME>;<Count>;<Remain size>;<Attrib>
or
QoK<File name>;<Size>;<DATE><TIME>;<Attrib>
```

<File name>	File name
<Size>	Number of bytes used by file.
<DATE>	Last update date. (yy-mm-dd)
<TIME>	Last update time (hh:mm:ss)
<Count>	File count in robot.
<Remain size>	Remain size of file system.
<Attrib>	Program attrib

[Reference command]

```
1;1;FDIRTOP
QoK1.MB5;235;04-02-0519:31:28;46;105984;12
1;1;FDIR1
QoK1.MB5;235;04-02-0519:31:28;12
1;1;FDIR<*.*
```

```
QoK1.MB5;235;04-02-0519:31:28;46;105984;12
```

[Related commands]

PDIR

5.2.24FCHECK (File check)

[Function]

The existence of the file is confirmed.

[Format]

```
FCHECK<File name>
```

<File name>	The checked file is specified. When the extension is omitted, it becomes MB5.

[Answer]

```
QoK<Status>
```

<Status>	N:Exist,F:No-exist

[Reference command]

```
1;1;FCHECK100
QoKN
```

[Related commands]

5.2.25FPATH= (File path)

[Function]

Get the save path name of the file.

[Format]

```
FPATH<File name>
```

<File name>	Specifies the file to obtain the path name. When the extension is omitted, it becomes MB5.

[Answer]

```
QoK<File name>
```

< File name >	File name in the path name with

[Reference command]

```
1;1;FPATH=1.MB5
QoK/robprg/dat/1.MB5
```

[Related commands]

5.2.26FCOPY (File copy)

[Function]

The file is copied.

[Format]

```
FCOPY<Src.file>;<Dst.file>
```

<Src.file>	Source file name is specified.
<Dst.file>	Destination file name is specified. When the extension is omitted, it becomes MB5.

[Answer]

```
QoK
```

[Reference command]

```
1;1;FCOPY1;2
QoK
```

[Related commands]

5.2.27FDEL (File delete)

[Function]

The file is deleted.

[Format]

```
FDEL<File name>
```

<File name>	The deleted file is specified When the extension is omitted, it becomes MB5.

[Answer]

```
QoK
```

[Reference command]

```
1;1;FDEL1
QoK
```

[Related commands]

5.2.28FRENAME (File rename)

[Function]

The file name is renamed.

[Format]

```
FRENAME<Src.file>;<Dst.file>
```

<Src.file>	Source file name is specified.
<Dst.file>	Destination file name is specified. When the extension is omitted, it becomes MB5.

[Answer]

```
QoK
```

[Reference command]

```
1;1;FRENAME1;2
QoK
```

[Related commands]

5.2.29FATTRIB (File attribute)

[Function]

The attribute of the file is changed.

[Format]

```
FATTRIB<File name>;<Attrib>
```

<File name>	File name
<Attrib>	Attribute +p:Read only lines -p:Read and write lines +q:Read only variable -q:Read and write variable

[Answer]

```
QoK
```

[Reference command]

```
1;1;FATTRIB1;+p
QoK
```

[Related commands]

5.2.30 FINIT (File init)

[Function]

The file is initialized.

[Format]

```
FINIT
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;FINIT
QoK
```

[Related commands]

5.2.31 FOPEN (File block open)

[Function]

The file is opened for the block reading and writing.

[Format]

```
FOPEN<File name>;<Mode>
```

<File name>	File name
<Mode>	w:Block write
	r:Block read

[Answer]

```
QoK
```

[Reference command]

```
1;1;FOPEN1.MB5;r
QoK
```

[Related commands]

FCLOSE, FREAD, FWRITE

5.2.32FCLOSE (File block close)

[Function]

The file which does the block reading and writing is closed.

[Format]

```
FCLOSE
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;FCLOSE
QoK
```

[Related commands]

FOPEN, FREAD, FWRITE

5.2.33FREAD (Block read)

[Function]

It block reads it from the file.

[Format]

```
FREAD
```

[Answer]

```
QoK<Contents>;<Continue>
```

<Contents>	The content of the file is returned with HEX ASCII.
<Continue>	1: There is a continuation data.

0: Continuation data none

[Reference command]

```
1;1;FREAD
```

```
QoK524F4250524700000000000000000000110100000000000200000000000000000000
00A000A003C5217003F550200065502000A000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000;1
```

[Related commands]

FOPEN, FCLOSE, FWRITE

5.2.34FWRITE (Block write)

[Function]

It blocks and it writes it in the file.

[Format]

```
FWRITE<Data size>;<Contents>
```

<Data size>	The size of the content is specified.
<Contents>	The content is specified with HEX ASCII.

[Answer]

```
QoK
```

[Reference command]

```
1;1;FWRITE120;524F425052470000000000000000000011010000000000020000000000
000000000000A000A003C5217003F550200065502000A00000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000
```

```
QoK
```

[Related commands]

FOPEN, FCLOSE, FWRITE

5.2.35EFREE (Read file size)

[Function]

The size of the file system is read.

[Format]

```
EFREE
```

[Answer]

```
QoK<Total size>;<Used size>;<Use rate>
```

<Total size>	Total sizes of file system.
<Used size>	Used size.
<Use rate>	Use rate. (%)

[Reference command]

```
1;1;EFREE
```

```
QoK125696;15616;12
```

[Related commands]

5.2.36 ESEARCH (Search string)

[Function]

Find the specified string from the editing program.

[Format]

```
ESEARCH<String>
```

<String>	Specify a search string.

[Answer]

```
QoK<Contents>;<Column>
```

<Contents>	The content of the line.
<Colum>	Column of find a string.

[Reference command]

```
1;9;ESEARCHMov
```

```
QoK3 Mov P1;2
```

[Related commands]

5.2.37 PRGLOAD= * (Program load)

[Function]

The program is loaded into the task slot.

[Format]

```
PRGLOAD=<Program name>
```

<Program name>   Program name.

[Answer]

```
QoK
```

[Reference command]

```
1;1;PRGLOAD=100
QoK
```

[Related commands]

PRG<UP/DOWN>

5.2.38PRG<UP/DOWN> * (Program select)

[Function]

The program of the task slot is selected.

[Format]

```
PRG<UP/DOWN>
```

<UP/DOWN>   Select UP or DOWN

[Answer]

```
QoK
```

[Reference command]

```
1;1;PRGUP
QoK
```

[Related commands]

PRGLOAD

5.2.39PRGRD (Execution program name read)

[Function]

The program name of the task slot is read.

[Format]

```
PRGRD
```

[Answer]

```
QoK<Program name>
```

<Program name> Program name.

[Reference command]

```
1;1;PRGRD
QoK100.MB5
```

[Related commands]

5.2.40LINENO= * (Execution line number change)

[Function]

The execution line number is changed.

[Format]

```
LINENO=<Line no.>
```

<Line no.> Line number

[Answer]

```
QoK
```

[Reference command]

```
1;1;LINENO=100
QoK
```

[Related commands]

LINENO

5.2.41LINENO (Execution Line number read)

[Function]

The execution line number is read.

[Format]

```
LINENO
```

[Answer]

```
QoK<Line no.>
```

< Line no.>	Line number

[Reference command]

```
1;1;LINENO
QoK100
```

[Related commands]

LINENO=

5.2.42 LINERD (Execution Line contents)

[Function]

The content of the line execution is read.

[Format]

```
LINERD
```

[Answer]

```
QoK<Contents>
```

<Contents>	The content of the line.

[Reference command]

```
1;1;LINERD
QoK10 MOV P1
```

[Related commands]

LINESRD

5.2.43 LINESRD (Execution more Line contents)

[Function]

More content of the line under execution is read.

[Format]

```
LINESRD<Continue>;<Prev.line>;<Next line>
```

<Continue>	0:New read, 1:Continuous read
<Prev.line>	Previous line count of the execution line
<Next line>	Next line count of the execution line

[Answer]

```
QoK<Continue>;<Line count>;<Contents>[0b< Contents >…];<Line no.>
```

<Continue>	1: There is a continuation line.
	0: Continuation line none
<Line count>	The read count of lines
<Contents>	The content of the line.
<Line no.>	Execution line number

[Reference command]

```
1;1;LINESRD0;2;2
QoK0;5;10 '### Program ###0b20 '0b30 *SMAIN0b40   MOV P00b50   DLY 1;0010
```

[Related commands]

LINERD

5.2.44SRV＜ON/OFF> * (Servo ON or OFF)

[Function]

The servo power supply is turned on and off.

[Format]

```
SRV<ON/OFF>
```

<ON/OFF>	Select ON or OFF

[Answer]

```
QoK
```

[Reference command]

```
1;1;SRVON
QoK
```

[Related commands]

5.2.45 OVRD= * (OP override change)

[Function]

The OP override is changed.

[Format]

```
OVRD=<Override>
```

<Override>	The set override is specified by 1-100.

[Answer]

```
QoK<Override>
```

<Override>	The set override value is returned.

[Reference command]

```
1;1;OVRD=50
QoK50
```

[Related commands]

OVRD

5.2.46 OVRD (OP override read)

[Function]

The OP override is read.

[Format]

```
OVRD
```

[Answer]

```
QoK<Override>
```

<Override>	A present override value is read.

[Reference command]

```
1;1;OVRD
QoK50
```

[Related commands]

OVRD=

5.2.47RUN * (Program start)

[Function]

The program is started.

[Format]

```
RUN[<Program name>;<Mode>]
```

<Program name>	Program name When omitting it, the program under the selection is started.
<Mode>	0:Repeat start 1:Cycle start When omitting it, repeat start.

[Answer]

```
QoK
```

[Reference command]

```
1;1;RUN100;1
QoK
```

[Related commands]

STOP, CSTOP

5.2.48STOP (STOP)

[Function]

The start is stopped.

[Format]

```
STOP
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;STOP
QoK
```

[Related commands]

STOP<ON/OFF>,RUN

5.2.49STOP<ON/OFF> (STOP ON or OFF)

[Function]

The start is stopped. After it STOPON it, it is not possible to start until the STOPOFF.

[Format]

```
STOP<ON/OFF>
```

<ON/OFF>	Select ON or OFF

[Answer]

```
QoK
```

[Reference command]

```
1;1;STOPON
QoK
```

[Related commands]

STOP, CSTOP, RUN

5.2.50CSTOP (Cycle STOP)

[Function]

The program under the start stops at the cycle.

[Format]

```
CSTOP
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;CSTOP
QoK
```

[Related commands]

STOP, STOP<ON/OFF>, RUN

5.2.51 RSTALRM (Error reset)

[Function]

The error is reset.

[Format]

```
RSTALRM
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;RSTALRM
QoK
```

[Related commands]

5.2.52 SLOTINIT (All program reset)

[Function]

The program resets all slots.

[Format]

```
SLOTINIT
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;SLOTINIT
QoK
```

[Related commands]

RSTPRG

5.2.53 RSTPRG (Each program reset)

[Function]

The program resets a specified slot.

[Format]

```
RSTPRG
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;RSTPRG
QoK
```

[Related commands]

SLOTINIT

5.2.54 RSTIO (Output signal reset)

[Function]

The general-purpose output signal is reset.

[Format]

```
RSTIO
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;RSTIO
QoK
```

[Related commands]

5.2.55 MLOCK<ON/OFF> (Machine lock ON or OFF)

[Function]

The machine lock is set.

[Format]

```
MLOCK<ON/OFF>
```

<ON/OFF>	Select ON or OFF

[Answer]

```
QoK
```

[Reference command]

```
1;1;MLOCKON
QoK
```

[Related commands]

5.2.56 HND<ON/OFF> (HAND open or close)

[Function]

The hand is opened and close.

[Format]

```
HND<ON/OFF><Hand no.>
```

<ON/OFF>	Select ON or OFF
<Hand no.>	Hand number 1-8 is specified.

[Answer]

```
QoK
```

[Reference command]

```
1;1;HNDON1
QoK
```

[Related commands]

5.2.57 ALIGN * (Aligning the hand)

[Function]

Aligning the hand.

[Format]

```
ALIGN
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;ALIGN
QoK
```

[Related commands]

5.2.58 MOVSP * (MOVE safe position)

[Function]

It moves to the safe position (the parameter "JSAFE").

[Format]

```
MOVSP
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;MOVSP
QoK
```

[Related commands]

5.2.59 JOG * (JOG operation)

[Function]

The jog operates. If the command is not received between about 140msec, the jog operation is automatically stopped.

[Format]

```
JOG<Jog mode>;<Reserve>;<-Dire.>;<+Dire.>;<Inch>
```

<Jog mode>	00:JOINT
	01:XYZ
	02:TOOL
	03:Reserve
	04:3-axis XYZ
	05:CYLNDER
<Reserve>	00:fix
<-Dire.>	The direction of operation is specified by the axis pattern.

<+Dire.>

00000000B	XYZ / JOINT	
_______1	X	J1
______1_	Y	J2
_____1__	Z	J3
____1___	A	J4
___1____	B	J5
__1_____	C	J6
_1______	L1	J7
1_______	L2	J8

<Inch> 00:Incing OFF

01:Inching High

02:Incing Low

[Answer]

```
QoK
```

[Reference command]

```
JOG01;00;01;00;00;00
QoK
```

[Related commands]

5.2.60JOG * (JOG operation for Multifunctional Electric Hand)

[Function]

The jog operation for Multifunctional Electric Hand. If the command is not received between about 140msec, the jog operation is automatically stopped.

[Format]

```
JOG<Jog mode>;<Reserve>;<Speed>;<Force>;<Point No>
```

<Jog mode>

00000000B	
_______1	bit0
______1_	bit1
_____1__	bit2
____1___	bit3
___1____	bit4
__1_____	bit5
_1______	bit6
1_______	bit7

Bit0 to bit2 : Mode

000B : Origin setting (Equivalent to EHOrg command)
001B : Hand open (Equivalent to EHOpen command)
010B : Hand close (Equivalent to EHClose command)
011B : Hand move (Equivalent to EHMov command)
100B : Hand hold (Equivalent to EHHold command)

<Reserve> 00:fix

<Speed> Specify the movement speed

The range of values is 01 to 64 (Hex).

<Force> Specify the holding force

The range of values is 01 to 64 (Hex).

<Point No> Specify the point number

The range of values is 00 to 20 (Hex).

[Answer]

```
QoK
```

[Reference command]

```
JOG44;00;46;32;0A
QoK
```

[Related commands]

5.2.61LS<ON/OFF> (Limit switch ON or OFF)

[Function]

The limit switch is turned on and off.

[Format]

```
LS<ON/OFF>
```

<ON/OFF> Select ON or OFF

ON:LS enable, OFF:LS disable

[Answer]

```
QoK
```

[Reference command]

```
1;1;LSON
QoK
```

[Related commands]

5.2.62AUE<ON/OFF> (Program start enable or disable)

[Function]

The program start is made enable or disable..

[Format]

```
AUE<ON/OFF>
```

<ON/OFF> Select ON or OFF

ON: Enable, OFF: Disable

[Answer]

```
QoK
```

[Reference command]

```
1;1;AUEON
QoK
```

[Related commands]

ATENA

5.2.63ATENA (Status can be start)

[Function]

The state which can be started is read.

[Format]

```
ATENA
```

[Answer]

```
QoK<Status>
```

< Status >   0:Start enable, 1:Start disable

[Reference command]

```
1;1;ATENA
QoK1
```

[Related commands]

AUE<ON/OFF>

5.2.64BRKPTSET (Set breakpoint)

[Function]

The breakpoint is set. It is not possible to set it while running.

[Format]

```
BRKPTSET=<Program name>;<Line number>;<Breakpoint type>
```

<Program name>   Program name

<Line number>	Breakpoint line number
< Breakpoint type >	0: Continues breakpoint, 1: One time breakpoint

When the plural is set to the same line, the breakpoint type set later is effective.

[Answer]

```
QoK
```

[Reference command]

```
1;1; BRKPTSET10;20;0
QoK1
```

[Related commands]

BRKPTCLR, BRKPTGET

5.2.65 BRKPTCLR (Delete breakpoint)

[Function]

The breakpoint is deleted.

[Format]

```
BRKPTCLR=<Program name>;<Line number>
```

<Program name>	Program name
<Line number>	Breakpoint line number 0 is all deleted.

[Answer]

```
QoK
```

[Reference command]

```
1;1; BRKPTCLR10;20
QoK1
```

[Related commands]

BRKPTSET, BRKPTGET

5.2.66 BRKPTGET (List breakpoint)

[Function]

The breakpoint lists is read.

[Format]

```
BRKPTGET=<Program name>;<Breakpoint number>;<Stat line>;<End line>
```

<Program name>	Program name When the program name is omitted, it is all program.
< Breakpoint number>	Breakpoint number 1 to 128
<Start line>	Reading start line number
<End line>	Reading end line number

[Answer]

```
QoK<Program name>;<Line number>;<Breakpoint type>
```

<Program name>	Program name
<Line number>	Breakpoint line number
<Breakpoint type>	0: Continues breakpoint, 1: One time breakpoint

[Reference command]

```
1;1; BRKPTSET
QoK1
```

[Related commands]

BRKPTSET, BRKPTCLR

5.2.67TOOLSET (Set Tool number)

[Function]

The tool number is set.

[Format]

```
TOOLSET=<Tool number>
```

<Tool number>	Tool number (0 to 16)

[Answer]

```
QoK
```

[Reference command]

```
1;1;TOOLSET2
QoK
```

[Related commands]

TOOLRD

5.2.68TOOLRD (Read Tool number)

[Function]

The tool number is read.

[Format]

```
TOOLRD
```

[Answer]

```
QoK<Tool number>
```

<Tool number>   Tool number (0 to 16)

[Reference command]

```
1;1;TOOLRD
QoK2
```

[Related commands]

TOOLSET

5.2.69SAFE= (Low speed mode)

[Function]

Set low speed operation mode.

[Format]

```
SAFE=<1/0>
```

[Answer]

```
QoK<Tool number>
```

<1/0>   1: Setting valid or 0: setting invalid

[Reference command]

```
1;1;SAFE=1
QoK2
```

[Related commands]

5.2.70 STATE (Read run status)

[Function]

The run state is read.

[Format]

```
STATE
```

[Answer]

```
QoK< Program name >;<Line no.>;<Override>;<Edit sts.>;<Run sts.><Stop sts.><Error no.>;
   <Step no.>;<Mech info.>;;;;;;;;;<Task prg.name>;<Task mode>;<Task cond.>;
   <Task pri.>;<Mech no.>
```

<Program name>	Program name loaded into task slot
<Line no.>	Execution line number
<Override>	A present override value is read.
<Edit sts.>	Edit status 00000000B _______1 Editing ______1_ Running _____1__ Changed
<Run sts.>	Run status by 2 HEX number fixation 00000000B 0 / 1 _______1 Cycle / Repeat ______1_ Cycle stop ON / OFF _____1__ MLOCK OFF / ON ____1___ Auto / Teach ___1____ Running of Teach mode __1_____ Servo OFF / ON _1______ STOP / RUN 1_______ Operation disable / enable
<Stop sts.>	Stop status by 2 HEX number fixation 00000000B _______1 EMG STOP ______1_ STOP _____1__ WAIT ____1___ STOP signal ON / OFF ___1____ Program select enable __1_____ (reserve) _1______ Pseudo input
<Error no.>	Error number. (0:No error)
<Step no.>	Execution step number
<Mech info.>	00000000B _______1 Mech 1 ______1_ Mech 2 _____1__ Mech 3
<Task prg.name>	Program name of slot table.
<Task mode>	Operation mode of slot table. (REP/CYC)
<Task cond.>	Stating conditions of slot table. (START/ALWAYS/ERROR)
<Task pri.>	Priority of slot table. (1 – 31)

<Mech no.>	Mech number under use

[Reference command]

```
1;1;STATE
Qok2.MB5;100;100;5;A1124220;1;1;;;;;;;;REP;START;1;0
```

[Related commands]

5.2.71DSTATE (Read stop status)

[Function]

The stop state is read.

[Format]

```
DSTATE
```

[Answer]

```
QoK<Run sts.><Stop sts.><Error no.>;<Step no.>;<Mech no.>
```

<Run sts.>	Run status by 2 HEX number fixation	
	00000000B	0 / 1
	_______1	Cycle / Repeat
	______1_	Cycle stop ON / OFF
	_____1__	MLOCK OFF / ON
	____1___	Auto / Teach
	___1____	Running of Teach mode
	__1_____	Servo OFF / ON
	_1______	STOP / RUN
	1_______	Operation disable / enable
<Stop sts.>	Stop status by 2 HEX number fixation	
	00000000B	
	_______1	EMG STOP
	______1_	STOP
	_____1__	WAIT
	____1___	STOP signal ON / OFF
	___1____	Program select enable
	__1_____	(reserve)
	_1______	Pseudo input
<Error no.>	Error number. (0:No error)	
<Step no.>	Execution step number	
<Mech no.>	Mech number under use	

[Reference command]

```
1;1;DSTATE
QoK09060;17;1
```

[Related commands]

5.2.72CALIB (Install status)

[Function]

The install status is read.

[Format]

```
CALIB
```

[Answer]

```
QoK<Status><Axis pattern><Reserve>
```

<Status> Install status (N:Defined、F:Not defined)

<Axis pattern> Completion axis pattern by 2 HEX number fixation

```
00000000B
_______1    J1
______1_    J2
_____1__    J3
____1___    J4
___1____    J5
__1_____    J6
_1______    J7
1_______    J8
```

[Reference command]

```
1;1; CALIB
QoKN3F3F
```

[Related commands]

5.2.73IOSIGNAL (Input and output signal read)

[Function]

The state of the input signal and the state of the output signal are read.

[Format]

```
IOSIGNAL<IN no.>;<OUT no.>
```

<IN no.> Input signal number.

<OUT no.> Output signal number.

[Answer]

```
QoK<IN state><OUT state>
```

<IN state> Input signal state by 4 HEX number fixation

<OUT state> Output signal state by 4 HEX number fixation

[Reference command]

```
1;1;IOSIGNAL0;0
QoK00000002
```

[Related commands]

5.2.74IN (Input signal read)

[Function]

The state of the input signal is read.

[Format]

```
IN<IN no.>
```

<IN no.>   Input signal number.

[Answer]

```
QoK<IN state>
```

<IN state>   Input signal state by 4 HEX number fixation

[Reference command]

```
1;1;IN0
QoK0000
```

[Related commands]

5.2.75OUT (Output signal read)

[Function]

The state of the output signal is read.

[Format]

```
OUT<OUT no.>
```

<OUT no.>   Output signal number.

[Answer]

```
QoK<OUT state>
```

<OUT state>   Output signal state by 4 HEX number fixation

[Reference command]

```
1;1;OUT0
QoK0001
```

[Related commands]

5.2.76OUT= (Output signal write)

[Function]

The output signal is compelling output.

[Format]

```
OUT=<OUT no.>;<OUT val.>
```

<OUT no.>	Output signal number.
<OUT val.>	Output signal value by 4 HEX number fixation

[Answer]

```
QoK
```

[Reference command]

```
1;1;OUT=0;0802
QoK
```

[Related commands]

5.2.77DIN (CC-Link's input register data read)

[Function]

The state of the CC-Link's input register is read. It is effective only to install the CC-Link option.

[Format]

```
DIN<IN no.>
```

<IN no.>	CC-Link’s input register number.

[Answer]

```
QoK<IN reg.state>;…(16 pieces)
```

<IN reg.state>	The input register is returned by the DEC.

[Reference command]

```
1;1;DIN6000
QoK0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
```

[Related commands]

DOUT

5.2.78DOUT (CC-Link's output register data read)

[Function]

The state of the CC-Link's output register is write. It is effective only to install the CC-Link option.

[Format]

```
DOUT<OUT no.>
```

<OUT no.>	CC-Link’s output register number.

[Answer]

```
QoK<OUT reg.state>;…   (16 pieces)
```

<OUT reg.state>	The output register is returned by the DEC.

[Reference command]

```
1;1;DOUT6000
QoK0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
```

[Related commands]

DIN

5.2.79DOUT= (CC-Link's output register data write)

[Function]

The CC-Link’s output register is compelling output. It is effective only to install the CC-Link option

[Format]

```
DOUT=<OUT no.>;<OUT val.>
```

<OUT no.>	CC-Link’s output register number.
<OUT val.>	Output register value by 4 HEX number fixation

[Answer]

```
QoK
```

[Reference command]

```
1;1;DOUT=6000;0002
QoK
```

[Related commands]

DIN, DOUT

5.2.80 INDMY (Set pseudo input)

[Function]

A pseudo-input of the input signal is set.

[Format]

```
INDMY
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;INDMY
QoK
```

[Related commands]

INSET, IN=, DIN=

5.2.81 INSET (Reset pseudo input)

[Function]

A pseudo-input of the input signal is reset.

[Format]

```
INSET
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;INSET
QoK
```

[Related commands]

INDMY, IN=, DIN=

5.2.82 IN= (Write pseudo input data)

[Function]

The input signal is pseudo-input.

[Format]

```
IN=<IN no.>;<IN val.>
```

<IN no.>	Input signal number.
<IN val.>	Pseudo-input signal value by 4 HEX number fixation

[Answer]

```
QoK
```

[Reference command]

```
1;1;IN=0;0802
QoK
```

[Related commands]

INDMY, INSET, DIN=

5.2.83 DIN= (Write pseudo input register)

[Function]

The CC-Link's input register is pseudo-input. It is effective only to install the CC-Link option.

[Format]

```
DIN=<IN no.>;<IN reg.val>
```

<IN no.>	Input register number.
<IN reg.val>	Pseudo-input register value by 4 HEX number fixation

[Answer]

```
QoK
```

[Reference command]

```
1;1;DIN=6000;0001
```

```
QoK
```

[Related commands]

INDMY, INSET, IN=

5.2.84STPSIG (Stop signal read)

[Function]

The state of the stop signal is read.

[Format]

```
STPSIG
```

[Answer]

```
QoK<Stop sts.>;<EMG sts.>
```

```
<Stop sts.>      The state of the stop signal is returned by the HEX.
                 00000000B
                 _______1    T/B(RS-422)
                 ______1_    P/C(RS-232C)
                 _____1__    I/O
                 ____1___    O/P
<EMG sts.>       The state of the EMG stop signal is returned by the HEX.
                 00000000B
                 _______1    I/O EMG
                 ______1_    O/P EMG
                 _____1__    T/B EMG
```

[Reference command]

```
1;1;STPSIG
```

```
QoK0;0
```

[Related commands]

5.2.85HNDSTS (Hand output signal read)

[Function]

The setting and the output of the hand are read.

[Format]

```
HNDSTS
```

[Answer]

```
QoK<Output no.>;<Hand sts.>;<ハンドタイプ>;…   (8 pieces)
```

<Output no.> Signal number allocated in hand.

-1:not used

<Hand sts.> Hand output status. (1:Hand open, 2:Hand close)

<Hand type> Hand type (0:Single-solenoid, 1:Double-solenoid)

[Reference command]

```
1;1;HNDSTS
QoK900;2;1;902;1;1;904;1;1;906;1;1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1
```

[Related commands]

5.2.86 USERAREASTS (User specified area read)

[Function]

The state of the user specified area is read.

[Format]

```
USERAREASTS
```

[Answer]

```
QoK<Status>
```

<Status> The state of the user specified area by the HEX.

00000000B	
_______1	Area 1
______1_	Area 2
_____1__	Area 3
____1___	Area 4
___1____	Area 5
__1_____	Area 6
_1______	Area 7
1_______	Area 8

[Reference command]

```
1;1;USERAREASTS
QoK0
```

[Related commands]

5.2.87 JPOS,PPOS,XPOS,RPOS (Current position read)

[Function]

The current position of the robot is read.

[Format]

```
<Type>POS<Axis info.>
```

<Type>	Type at read position J:JOINT, P:XYZ, X:3axis-XYZ, R:CYLNDER
<Axis info.>	Read axis (1 – 8:Single axis, F:Full axis)

[Answer]

```
QoK<Axis name>;<Axis data>;…   (Repeat axis number)
  ;<Flag1>,<Flag2>;<Override>;<End speed>;<+Limit><-Limit><reserve>
```

<Axis name>	Name of axis
<Axis data>	Data of axis
<Flag1>	Structure flag 1 and 2.
<Flag2>	Only XYZ is significant.
<Override>	A present override value is read.
<End speed>	Speed in robot end. (mm/sec)
<+Limit>	Limit status by 2 HEX number fixation.
<-Limit>	

00000000B	XYZ / JOINT	
_______1	X	J1
______1_	Y	J2
_____1__	Z	J3
____1___	A	J4
___1____	B	J5
__1_____	C	J6
_1______	L1	J7
1_______	L2	J8

[Reference command]

```
1;1;PPOSF
QoKX;290.62;Y;-0.09;Z;11.26;A;-179.94;B;-0.26;C;179.93;L1;0.00;;6,0;100;0.00;00000000
```

[Related commands]

5.2.88GJPOS,GPPOS (Destination position read)

[Function]

The current position of the robot is read.

[Format]

```
G<Type>POS<Axis info.>
```

<Type>	Type at read position J:JOINT, P:XYZ

<Axis info.>	Read axis (1 – 8:Single axis, F:Full axis)

[Answer]

```
QoK<Axis name>;<Axis data>;…   (Repeat axis number)
  ;<Flag1>,<Flag2>;<Override>;<End speed>;<+Limit><-Limit><reserve>
```

<Axis name>	Name of axis
<Axis data>	Data of axis
<Flag1>	Structure flag 1 and 2.
<Flag2>	Only XYZ is significant.
<Override>	A present override value is read.
<End speed>	Speed in robot end. (mm/sec)
<+Limit>	Limit status by 2 HEX number fixation.
<-Limit>	

00000000B	XYZ / JOINT	
_______1	X	J1
______1_	Y	J2
_____1__	Z	J3
____1___	A	J4
___1____	B	J5
__1_____	C	J6
_1______	L1	J7
1_______	L2	J8

[Reference command]

```
1;1;GPPOSF
QoKX;290.62;Y;-0.09;Z;11.26;A;-179.94;B;-0.26;C;179.93;L1;0.00;;6,0;100;0.00;00000000
```

[Related commands]

5.2.89TIME (Time read)

[Function]

Time is read.

[Format]

```
TIME
```

[Answer]

```
QoK<DATE><TIME>
```

<DATE>	DATE (yy-mm-dd)
<TIME>	TIME (hh:mm:ss)

[Reference command]

```
1;1;TIME
QoK04-02-0917:54:32
```

[Related commands]

TIME=

5.2.90TIME= (Time change)

[Function]

Time is set.

[Format]

```
TIME=<DATE><TIME>
```

<DATE>	DATE (yy-mm-dd)
<TIME>	TIME (hh:mm:ss)

[Answer]

```
QoK
```

[Reference command]

```
1;1;TIME=04-02-0917:00:00
QoK
```

[Related commands]

5.2.91PTIME (Hour meter read)

[Function]

The operation time of the robot is read.

[Format]

```
PTIME
```

[Answer]

```
QoK<Power time>;<All mech servo time>;<Mech1 servo time>;<Mech2 servo time>;
  <Mech3 servo time>;0;0;0;0;0;0;0;0;0;0;0;0;0<Operation time>;0;<Accumu.time>;
  <Remain time>;<Mech info.>
```

<Power time>	Power on time (Hr)
<All mech servo time>	All mecha servo on time (Hr)

<Mech1 servo time>	Mech n servo on time (Hr)
<Mech2 servo time>	
<Mech3 servo time>	
<Operation time>	Program operation time (Hr)
<Accumu.time>	Battery accumulation time. (Hr)
<Remain time>	Battery remaining time. (Hr)
<Mech info.>	Mech information by the HEX. 00000000B _______1 Mech 1 ______1_ Mech 2 _____1__ Mech 3

[Reference command]

```
1;1;PTIME
QoK149;51;51;0;0;0;0;0;0;0;0;0;0;0;0;0;0;102;0;2715;11885;1
```

[Related commands]

PTIMEDEL=

5.2.92 PTIMEDEL= (Hour meter clear)

[Function]

The operating time of the robot is cleared.

[Format]

```
PTIMEDEL=<Kind>
```

<Kind>	Kind at operation time. T: Power on time, S: Servo on time, D: Program operation time B: Battery accumulation time, Z: All

[Answer]

```
QoK
```

[Reference command]

```
1;1;PTIMEDEL=B
QoK
```

[Related commands]

PTIME

5.2.93CYCLETIME (Cycle time read)

[Function]

The operating time of the program is read.

[Format]

```
CYCLETIME<Program name>
```

<Program name>	Program name

[Answer]

```
QoK<Operation time>;<Latest tact>;<Avg.tact>;<Cycle count>
```

<Operation time>	Operation time. (msec)
<Latest.tact>	Latest of tact time. (msec)
<Avg.tact>	Average of tact time. (msec)
<Cycle count>	Operation count

[Reference command]

```
1;1;CYCLETIME100
QoK224745;156;154;1454
```

[Related commands]

PDIR, CYCLECLR

5.2.94CYCLECLR (Cycle time clear)

[Function]

The operating time of the program is cleared.

[Format]

```
CYCLECLR<Program name>
```

<Program name>	Program name

[Answer]

```
QoK
```

[Reference command]

```
1;1;CYCLECLR100
QoK
```

[Related commands]

PDIR, CYCLETIME

5.2.95ERROR (Error number read)

[Function]

The error number is read.

[Format]

```
ERROR
```

[Answer]

```
QoK[<Error no.>;<Error level>]
```

<Error no.>	Error number When the error does not occur, only QoK is returned.
<Error level>	Error level (1:High, 2:Low, 3:Caution)

[Reference command]

```
1;1;ERROR
Qok4140;2
```

[Related commands]

ERRORMES

5.2.96ERRORMES (Error contents read)

[Function]

The content of the error is read.

[Format]

```
ERRORMES<Error no.>
```

<Error no.>	Error number

[Answer]

```
QoK<Contents>
```

<Contents>	Content of the error.

[Reference command]

```
1;1;ERRORMES4140
```

```
QoKThe program was not found
```

[Related commands]

ERROR

5.2.97 ERRORLOG (Error history read)

[Function]

The error history is read.

[Format]

```
ERRORLOG<Hist.no.>
```

<Hist.no.>	Position of history
	TOP: The newest history
	END: The oldest history
	+1: Previous history
	-1: Next history

[Answer]

```
QoK<DATE>;<TIME>;<Error no.>;<Error contents>;<Error level>;<Program name >;
  <Line no.>;<Error detail no.>
```

<DATE>	Date when error occurs (yy-mm-dd)
<TIME>	Time when error occurs (hh:mm:ss)
<Error no.>	Error number
<Error contents>	Content of the error
<Error level>	Error level
	1:High,2:Low,3:Caution
<Program name>	Program name when error occurs
<Line no.>	Line number when error occurs
<Error detail no.>	Error detail number
	Only QoK is returned when there is no error history.

[Reference command]

```
1;1;ERRORLOGTOP
QoK04-02-09;18:49:52;4140;The program was not found;2;S1;0;414000000
```

[Related commands]

ERRLOGCLR, ERRLOG2=

5.2.98ERRLOG2=　(Error history reading. / Error details number narrowing seeing.)

[Function]

It narrows by the error level and error details number, and error history information is read.

(It is not possible to use by the CRn-500 series.)

(It is possible to use since the edition of main S/W of the CRnQ-700/CRnD-700 series version R2/S2.)

[Format]

```
ERRLOG2=<Hist. No.>;<Error level >;<Error details number>
```

<Hist. No.>	Position of history TOP: The newest history END: The oldest history +1: Previous history -1: Next history
<Error level>	Error level (The range of the extraction is narrowed.) 0:All level,　1:High,　2: Low,　3:Caution,　0 when omitting it When < history number > is "+" and "-" specification, the last specification is used.
<Error details number>	Narrowed error details number (6-9 digit).

[Answer]

```
QoK<DATE>;<TIME>;<Error no.>;<Error contents>;<Error level>;<Program name >;
<Line no.>;<Error detail no.>
```

<DATE>	Date when error occurs (yy-mm-dd)
<TIME>	Time when error occurs (hh:mm:ss)
<Error no.>	Error number
<Error contents>	Content of the error
<Error level>	Error level 1:High,2:Low,3:Caution
<Program name>	Program name when error occurs
<Line no.>	Line number when error occurs
<Error detail no.>	Error detail number Only QoK is returned when there is no error history.

[Reference command]

```
1;1;ERRLOG2=TOP;0;414000000
QoK04-02-09;18:49:52;4140;The program was not found;2;S1;0;414000000
```

[Related commands]

ERRLOGCLR, ERRORLOG

[Note]

When "+" and "-" are specified for < history number >, referring the last positional and last reference error number is used common of the communication command "ERRORLOG" and "ERRORRD".

5.2.99 ERRLOGCLR (Error history clear)

[Function]

The error history is cleared.

[Format]

```
ERRLOGCLR<Error level>
```

<Error level>　　Error level

1:High,2:Low,3:Caution,0:All

[Answer]

```
QoK
```

[Reference command]

```
1;1;ERRLOGCLR1
QoK
```

[Related commands]

ERRLOG

5.2.100 ERRSUM (Error summary)

[Function]

The error summary data is read.

[Format]

```
ERRSUM<No.>
```

<No.>　　number (1-100)　　(in numerical order)

[Answer]

```
QoK<Number of errors>;<Detailed error number>;<Error level>;<Error contents>;<Cause>;
<Treat>;<Count of error >
```

<Number of errors>　　Number of error total data

<Detailed error number>	Detailed error number "000000000" is returned when there is no error history.
<Error level>	Error level (H:High, L:Low, C:Caution) *
<Error contents>	Error contents*
<Cause>	Error cause *
<Treat>	Error treat *
<Count of error >	Count of error generation

* Note : “”(NULL) is returned when there is no error history.

[Reference command]

```
1;1;ERRSUM1
QoK4;006000000;H; EMG signal is input. (O.Panel); EMG signal is input. (O.Panel); Check the O.Panel emergency stop;4
1;1;ERRSUM5
QoK4;000000000;;;;;0
```

[Reference command]

ERRSUMCLR, SUMDATE, ERRSUM2=

5.2.101 ERRSUM2= (error summary)

[Function]

The error summary data is read.

(It is not possible to use by the CRn-500 series.)

(It is possible to use since the edition of main S/W of the CRnQ-700/CRnD-700 series version R2/S2.)

[Format]

```
ERRSUM2=<Hist. no.>
```

<Hist.no.>	number (1-100)	(in numerical order**)**

[Answer]

```
QoK<Number of errors>;<Detailed error number>;<Error level>;<Error contents>;<Cause>;
<Treat>;<Count of error >;<date>;<time>
```

<Number of errors>	Number of error total data
<Detailed error number>	Detailed error number "000000000" is returned when there is no error history.
<Error level>	Error level (H:High, L:Low, C:Caution) *

<Error contents>	Error contents*
<Cause>	Error cause *
<Treat>	Error treat *
<Count of error >	Count of error generation
<date>	The last date when error occurred.
<time>	The last time when error occurred.

* Note : “”(NULL) is returned when there is no error history.

[Reference command]

```
1;1;ERRSUM2=1
QoK4;006000000;H; EMG signal is input. (O.Panel); EMG signal is input. (O.Panel); Check the O.Panel emergency stop;4;10-08-19;18:49:52
1;1;ERRSUM5
QoK4;000000000;;;;;0;;
```

[Reference command]

ERRSUMCLR, SUMDATE, ERRSUM

5.2.102 ERRSUMCLR (Clear error summary)

[Function]

The error summary is cleared.

[Format]

```
ERRLOGCLR
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;ERRLOGCLR
QoK
```

[Reference command]

ERRSUM, ERRSUM2=

5.2.103 SUMDATE (Date when error logging function began)

[Function]

The beginning date of the error logging function (error log / error summary) is read.

[Format]

```
SUMDATE<function no.>
```

<function no.> Data of the acquired beginning date is specified.

0: Error log function

1: Error summary function

[Answer]

```
QoK<date>;<time >
```

<date> Date when data acquisition was begun（yy-mm-dd）

<time> Tiem when data acquisition was begun（hh:mm:ss）

[Reference command]

```
1;1;SUMDATE0
QoK10-08-09;20:23:46
```

[Reference command]

ERRLOGCLR, ERRSUMCLR

5.2.104 VAL (Variable data read)

[Function]

The value of the variable is read.

[Format]

```
VAL<Variable posi.>;<Class>
```

<Variable posi.> The read position is specified.

<: Top variable

>: Bottom variable

+1: Next variable

-1: Previous variable

Variable name: Specified Variable

<Class> Variable class

(M: Integer, C: Character, P: Position, J: JOINT)

[Answer]

```
QoK<Variable name>=<Value>
```

<Variable name>	Variable name
<Value>	Value of the variable

[Reference command]

```
1;1;VAL<;M
QoKM1=+3
```

[Related commands]

VALS

5.2.105 VALS (More Variable data read)

[Function]

The value of more variables is read.

[Format]

```
VALS<Variable posi.>;<Class>
```

<Variable posi.>	The read position is specified. <: Top variable >: Bottom variable +1: Next variable -1: Previous variable
<Class>	Variable class (M: Integer, C: Character, P: Position, J:JOINT)

[Answer]

```
QoK<Variable name>=<Value>[0b<Variable name>=<Value>…];<Count>;<Continue>
```

<Variable name>	Variable name
<Value>	Value of the variable
<Count>	Count of read variables
<Continue>	1: There is a continuation variable. 0: Continuation variable none

[Reference command]

```
1;1;VALS<;M
QoKM1=+1;0bM2=+2;0bM3=+3;3;0
```

[Related commands]

VAL

5.2.106 GVAL (Global variable data read)

[Function]

The value of the global variable is read.

[Format]

```
GVAL<Variable posi.>;<Class>
```

<Variable posi.>	The read position is specified.
	<: Top variable
	>: Bottom variable
	+1: Next variable
	-1: Previous variable
	Variable name: Specified Variable
<Class>	Variable class
	(M: Integer, C: Character, P: Position, J: JOINT)

[Answer]

```
QoK<Variable name>=<Value>
```

<Variable name>	Variable name
<Value>	Value of the variable

[Reference command]

```
1;1;GVAL<;M
QoKM_01=+3
```

[Related commands]

GVALS

5.2.107 GVALS (More Variable data read)

[Function]

The value of more global variables is read.

[Format]

```
GVALS<Variable posi.>;<Class>
```

<Variable posi.>	The read position is specified.

	<: Top variable
	>: Bottom variable
	+1: Next variable
	-1: Previous variable
<Class>	Variable class (M: Integer, C: Character, P: Position, J:JOINT)

[Answer]

```
QoK<Variable name>=<Value>[0b<Variable name>=<Value>…];<Count>;<Continue>
```

<Variable name>	Variable name
<Value>	Value of the variable
<Count>	Count of read variables
<Continue>	1: There is a continuation variable. 0: Continuation variable none

[Reference command]

```
1;1;GVALS<;M
QoKM_01=+1;0bM_02=+2;0bM_03=+3;3;0
```

[Related commands]

GVAL

5.2.108 HOT (Variable data write)

[Function]

The value of the variable is changed. A integer variable is revocable while even starting.

[Format]

```
HOT<Program name>;<Variable name>=<Value>
```

<Program name>	Program name
<Variable name>	Variable name
<Value>	Changed value

[Answer]

```
QoK
```

[Reference command]

```
1;1;HOT100;M1=1
```

```
QoK
```

[Related commands]

5.2.109 OPNUMRD (Option slot number read)

[Function]

The number of option slots is read.

[Format]

```
OPNUMRD
```

[Answer]

```
QoK<Slot no.>
```

<Slot no.>	Number of option slots.

[Reference command]

```
1;1;OPNUMRD
QoK3
```

[Related commands]

OPSTSRD

5.2.110 OPSTSRD (Option information read)

[Function]

Option slot information is read.

[Format]

```
OPSTSRD<Slot no.>
```

<Slot no.>	Number of option slots. (1,2 or 3)

[Answer]

```
QoK<Option info.>
```

<Option info.>	Option information

[Reference command]

```
1;1;OPSTSRD
QoKEthernet;IP Addr 192.168.0.10bPortNo(R-time) 100000b
```

```
PortNo(OPT11-19) 10001/10002/10003/10004/10005/10006/10007/10008/100090b
Connect Count(OPT11-19) 2/0/0/0/0/0/0/0/00bMAC Addr 00:00:00:00:00:000bH/W Ver: 0
```

[Related commands]

OPSTSRD

5.2.111 THMRD (Controller temperature read)

[Function]

Read the temperature internal the controller.

[Format]

```
THMRD
```

[Answer]

```
QoK<Temperature>
```

<Temperature>   Temperature internal the controller

[Reference command]

```
1;1;THMRD
```

```
QoK39.5
```

[Related commands]

ETEMP

5.2.112 ETEMP (Encoder temperature read)

[Function]

Read the temperature the encoder.

[Format]

```
ETEMP
```

[Answer]

```
QoK<J1 Temperature>;<J2 Temperature>;<J3 Temperature>;<J4 Temperature>;
<J5 Temperature>;<J6 Temperature>;<J7 Temperature>;<J8 Temperature>;
<J1 Max.Temperature >;<J2 Max.Temperature>;<J3 Max.Temperature>;
<J4 Max.Temperature>;<J5 Max.Temperature>;<J6 Max.Temperature>;
<J7 Max.Temperature>;<J8 Max.Temperature>
```

<Jn Temperature> Temperature the encoder (n indicate the axis no.)

<Jn Max.Temperature> Maximum value of the temperature of the encoder after power ON (n indicate the axis no.)

[Reference command]

```
1;1;ETEMP
QoK38;39;41;39;36;36;0;0;41;42;44;41;38;37;0;0
```

[Related commands]

THMRD

5.2.113 EMISS (Encoder miscount read)

[Function]

Read the miscount the encoder.

[Format]

```
EMISS
```

[Answer]

```
QoK<J1 Miscount>;<J2 Miscount >;<J3 Miscount >;<J4 Miscount >;
<J5 Miscount >;<J6 Miscount >;<J7 Miscount >;<J8 Miscount >
```

<Jn Temperature> Miscount the encoder (n indicate the axis no.)

[Reference command]

```
1;1;EMISS
QoK0;1;1;0;0;0;0;0
```

[Related commands]

5.2.114 SRVENC (Servo encoder read)

[Function]

Data concerning the position of each axis motor can be monitored.

[Format]

```
SRVENC
```

[Answer]

```
QoK<J1 pos. feedback>;<J1 pos. in 1 rot.>;<J1 Fdt command>;<J1 pos. command>;
<J2 pos. feedback >;<J2 pos. in 1 rot.>;<J2 Fdt command>;<J2 pos. command>;
<J3 pos. feedback>;<J3 pos. in 1 rot.>;<J3 Fdt command>;<J3 pos. command>;
<J4 pos. feedback>;<J4 pos. in 1 rot.>;<J4 Fdt command>;<J4 pos. command>;
<J5 pos. feedback >;<J5 pos. in 1 rot.>;<J5 Fdt command>;<J5 pos. command>;
<J6 pos. feedback >;<J6 pos. in 1 rot.>;<J6 Fdt command>;<J6 pos. command>;
<J7 pos. feedback>;<J7 pos. in 1 rot.>;<J7 Fdt command>;<J7 pos. command>;
<J8 pos. feedback>;<J8 pos. in 1 rot.>;<J8 Fdt command>;<J8 pos. command>
```

<Jn pos. feedback>	Position feedback the encoder (n indicate the axis no.)
<Jn pos. in 1 rot.>	Position in 1 rotation the encoder (n indicate the axis no.)
<Jn Fdt command>	Fdt command the encoder (n indicate the axis no.)
<Jn pos. command>	Position command the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVENC
```

```
QoK-4475890;0;0;-4487192;4912356;0;0;4892959;10225275;0;0;10323293;-940046;0;0;-947
494;-3330789;0;0;-3358040;6508941;0;0;6502064;0;0;0;0;0;0;0;0
```

[Related commands]

5.2.115 SRVDRP (Servo droop read)

[Function]

Data concerning the droop of each axis motor can be monitored.

[Format]

```
SRVDRP
```

[Answer]

```
QoK<J1 pos. droop>;<J1 max. pos. droop>;<J2 pos. droop>;<J2 max. pos. droop>;
<J3 pos. droop>;<J3 max. pos. droop>;<J4 pos. droop>;<J4 max. pos. droop.>;
<J5 pos. droop>;<J5 max. pos. droop>;<J6 pos. droop>;<J6 max. pos. droop>;
<J7 pos. droop>;<J7 max. pos. droop>;<J8 pos. droop>;<J8 max. pos. droop>
```

<Jn pos. droop>	Position droop the encoder (n indicate the axis no.)
<Jn max. pos. droop>	Maximum position droop the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVDRP
QoK16860;1525531;22463;-1686967;126622;-543044;-11984;-318476;36426;2421963;9865;
-2205319;0;0;0;0
```

[Related commands]

5.2.116　SRVSPD (Servo speed read)

[Function]

The following data concerting the rotational speed of each axis motor can be monitored.

[Format]

```
SRVSPD
```

[Answer]

```
QoK<J1 spd. feedback>;<J1 speed max.>;<J1 speed command>;
<J2 spd. feedback>;<J2 speed max.>;<J2 speed command>;
<J3 spd. feedback>;<J3 speed max.>;<J3 speed command>;
<J4 spd. feedback>;<J4 speed max.>;<J4 speed command>;
<J5 spd. feedback>;<J5 speed max.>;<J5 speed command>;
<J6 spd. feedback>;<J6 speed max.>;<J6 speed command>;
<J7 spd. feedback>;<J7 speed max.>;<J7 speed command>;
<J8 spd. feedback>;<J8 speed max.>;<J8 speed command>
```

<Jn spd. feedback>	Speed feedback the encoder (n indicate the axis no.)
<Jn speed max.>	Speed maximum the encoder (n indicate the axis no.)
<Jn speed command>	Speed command the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVSPD
QoK16;3755;68;29;-4148;115;147;-1150;-587;-11;-783;45;41;5964;163;10;-5429;41;0;0;0;0;0;
0
```

[Related commands]

5.2.117　SRVCUR (Servo current read)

[Function]

Data concerning the current value of each axis motor can be monitored.

[Format]

```
SRVCUR<type>
```

<type>　　Read type (0:command, 1:feedback)

[Answer]

type1

```
QoK<J1 cur. feedback>;<J1 max. curr. cmd1>;<J1 max. curr. cmd2>;
<J2 cur. feedback>;<J2 max. curr. cmd1>;<J2 max. curr. cmd2>;
<J3 cur. feedback>;<J3 max. curr. cmd1>;<J3 max. curr. cmd2>;
<J4 cur. feedback>;<J4 max. curr. cmd1>;<J4 max. curr. cmd2>;
<J5 cur. feedback>;<J5 max. curr. cmd1>;<J5 max. curr. cmd2>;
<J6 cur. feedback>;<J6 max. curr. cmd1>;<J6 max. curr. cmd2>;
<J7 cur. feedback>;<J7 max. curr. cmd1>;<J7 max. curr. cmd2>;
<J8 cur. feedback>;<J8 max. curr. cmd1>;<J8 max. curr. cmd2>
```

<Jn cur. feedback>　　Current feedback the encoder (n indicate the axis no.)

<Jn max. curr. cmd1>　　Maximum current command the encoder (n indicate the axis no.)

<Jn max. curr. cmd2>　　Maximum current command the encoder (n indicate the axis no.)

type0

```
QoK<J1 cur. command>;<J1 tolerable cmd+>;<J1 tolerable cmd->;<J1 RMS Current>;
<J2 cur. command >;<J2 tolerable cmd+>;<J2 tolerable cmd->;<J2 RMS Current>;
<J3 cur. command >;<J3 tolerable cmd+>;<J3 tolerable cmd->;<J3 RMS Current>;
<J4 cur. command >;<J4 tolerable cmd+>;<J4 tolerable cmd->;<J4 RMS Current>;
<J5 cur. command >;<J5 tolerable cmd+>;<J5 tolerable cmd->;<J5 RMS Current>;
<J6 cur. command >;<J6 tolerable cmd+>;<J6 tolerable cmd->;<J6 RMS Current>;
<J7 cur. command >;<J7 tolerable cmd+>;<J7 tolerable cmd->;<J7 RMS Current>;
<J8 cur. command >;<J8 tolerable cmd+>;<J8 tolerable cmd->;<J8 RMS Current>
```

<Jn cur. command>　　Current command the encoder (n indicate the axis no.)

<Jn tolerable cmd+>　　tolerable command+ the encoder (n indicate the axis no.)

<Jn tolerable cmd->　　tolerable command- the encoder (n indicate the axis no.)

<Jn RMS Current>　　RMS Current the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVCUR1
```

```
QoK-0.42;0.49;3.30;0.80;1.73;2.63;-0.57;0.67;0.83;0.24;0.24;0.86;0.13;0.39;0.87;-0.24;0.24;0
.93;0.00;0.00;0.00;0.00;0.00;0.00
```

[Related commands]

5.2.118 SRVLCR (Servo load current)

[Function]

The load state of each axis motor and the temperature of the encoder can be monitored.

[Format]

```
SRVLCR
```

[Answer]

```
QoK<J1 load level>;<J1 max. load level>;<J2 load level>;<J2 max. load level>;
<J3 load level>;<J3 max. load level>;<J4 load level>;<J4 max. load level>;
<J5 load level>;<J5 max. load level>;<J6 load level>;<J6 max. load level>;
<J7 load level>;<J7 max. load level>;<J8 load level>;<J8 max. load level>
```

<Jn load level>	Load level the encoder (n indicate the axis no.)
<Jn max. load level>	Maximum load level the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVLCR
QoK3.00;3.00;35.00;35.00;30.00;30.00;14.00;14.00;32.00;32.00;8.00;8.00;0.00;0.00;0.00;0.00
```

[Related commands]

5.2.119 SRVVOL (Servo voltage)

[Function]

The following data concerning robot controller's main circuit power supply can be monitored.

[Format]

```
SRVVOL
```

[Answer]

```
QoK<Power voltate>;<Power voltate (max)>;< Power voltate(min) >;
<J1 regen. level>;<J2 regen. level>;<J3 regen. level>;<J4 regen. level>;
<J5 regen. level>;<J6 regen. level>;<J7 regen. Level>;<J8 regen. level>
```

<Power voltate >	Power voltage the encoder

<Power voltate (max)>	Power voltage the encoder
<Power voltate (min)>	Power voltage the encoder
<Jn regen. level >	Regenaration level the encoder (n indicate the axis no.)

[Reference command]

```
1;1;SRVVOL
QoK0;0;0;0;0;0;0;0;0;0;0
```

[Related commands]

5.2.120 SVMONRST= (Reset servo monitor maximum)

[Function]

The servo monitor data is cleared.

[Format]

```
SVMONRST=
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;SVMONRST=
QoK
```

[Related commands]

5.2.121 RAREAD= (Read serial number)

[Function]

Read the robot serial number and controller serial number.

[Format]

```
RAREAD=
```

[Answer]

```
QoK<Number>;<Serial number >;<Set date>
```

<Number>	Robot number

<Serial number>	Robot serial number or controller serial number
< Set date >	Setting date of serial number

[Reference command]

```
1;1; RAREAD=
QoK1;xxxxxxxxxx;2014/04/09 17:14:02
```

[Related commands]

5.2.122 PRMINIT (Parameter initial)

[Function]

The parameter is initialized, and it puts it into the state when shipping it.

[Format]

```
PRMINIT
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;PRMINIT
QoK
```

[Related commands]

5.2.123 PNR (Parameter read)

[Function]

The parameter is read. The parameter at the level corresponding to keyword (KEYWD) is read.

[Format]

```
PNR<Parameter name>
```

<Parameter name>	Parameter name

[Answer]

```
QoK<Parameter name>;<Value>;<Value count>
```

<Parameter name>	Parameter name
<Value>	Value of parameter

<Value count>　　Count of value

[Reference command]

```
1;1;PNRBZR
QoKBZR;1;1
```

[Related commands]

PRM=

5.2.124　PRM= (Parameter write)

[Function]

The parameter is changed. The parameter at the level corresponding to keyword (KEYWD) is changed.

[Format]

```
PRM=<Parameter name>;<Value>
```

<Parameter name> Parameter name

<Value>　　Value of parameter

[Answer]

```
QoK
```

[Reference command]

```
1;1;PRM=BZR;1
QoK
```

[Related commands]

PNR

5.2.125　PAR (Parameter read)

[Function]

The parameter is able to read. All parameters are able to read regardless of keyword (KEYWD).

[Format]

```
PAR<Parameter name>
```

<Parameter name> Parameter name

[Answer]

```
QoK<Parameter name>;<Value>;<Value count>
```

<Parameter name>	Parameter name
<Value>	Value of parameter
<Value count>	Count of value

[Reference command]

```
1;1;PARBZR
QoKBZR;1;1
```

[Related commands]

PAW=

5.2.126 PAW= (Parameter write)

[Function]

The parameter is changed. All parameters are changed regardless of keyword (KEYWD).

[Format]

```
PAW=<Parameter name>;<Value>
```

<Parameter name>	Parameter name
<Value>	Value of parameter

[Answer]

```
QoK
```

[Reference command]

```
1;1;PAW=BZR;1
QoK
```

[Related commands]

PAR

5.2.127 PAW2= (Parameter write (need to reboot.))

[Function]

The parameter is able to write. All parameters are able to write regardless of keyword (KEYWD). However, turning the power off-on is needed without fail for the reflection of the parameter setting value.

(It is not possible to use by the CRn-500 series.)

(It is possible to use since the edition of main S/W of the CRnQ-700/CRnD-700 series version R2/S2.)

[Format]

```
PAW2=<name>;<value>
```

<name>	parameter name to write
<value>	parameter value to write

[Answer]

```
QoK
```

[Reference command]

```
1;1;PAW2=BZR;1
QoK
```

[Related commands]

PAR、PAW=

5.2.128 PRMUNDO (Parameter undo)

[Function]

It returns it to the value when the parameter is shipped.

[Format]

```
PRMUNDO<Mech No.>;<Parameter name>
```

<Mech no.>	Mech number. (0:Common parameter)
<Parameter name>	Parameter name

[Answer]

```
QoK
```

[Reference command]

```
1;1;PRMUNDO0;BZR
QoK
```

[Related commands]

PRMINIT

5.2.129 PRM= (Read change parameter list)

[Function]

The list of the parameter name that has been changed is read.

(It is possible to use it since the edition of main S/W version N6/P6.)

[Format]

```
PMR=<Continuation specification>
```

< Continuation specification > The reading method is specified.

0: Newly reading

1: Continuance reading.

[Answer]

```
QoK<Continuation>;<num>;<name>[0b<name>…]
```

<Continuation>	0: Continuation - off , 1: Continuation - on
<num>	number of read parameters
<name>	parameter name It is delimited with 0x0b code between parameter name.

[Answer]

```
1;1;PMR=0
QoK0;6;SVSSFR0bMEOFFZ0bMEINST0bMEINSZ0bMEINSD0bMEJAR
```

[Related commands]

5.2.130 KEYWD (Keyword input)

[Function]

The level of opening to the public of the parameter is changed.

[Format]

```
KEYWDp<Keyword>
```

<Keyword> Keyword

[Answer]

```
QoK
```

[Reference command]

```
1;1;KEYWDpMELFA
QoK
```

[Related commands]

PNR, PRM=

5.2.131 SLOTRD (Slot table read)

[Function]

The slot table is read.

[Format]

```
SLOTRD
```

[Answer]

```
QoK<Task prg.name>;<Task mode>;<Task cond.>;<Task pri>
```

<Task prg.name>	Program name of slot table.
<Task mode>	Operation mode of slot table. (REP/CYC)
<Task cond.>	Stating conditions of slot table. (START/ALWAYS/ERROR)
<Task pri.>	Priority of slot table. (1 – 31)

[Reference command]

```
1;2;SLOTRD
```

```
QoK2;REP;START;1
```

[Related commands]

SLOTSET

5.2.132 SLOTSET (Slot table write)

[Function]

The slot table is changed. ("SLTn" parameter is changed. It is necessary to reboot.)

[Format]

```
SLOTSET<Task prg.name>;<Task mode>;<Task cond.>;<Task pri>
```

<Task prg.name>	Program name of slot table.
<Task mode>	Operation mode of slot table. (REP/CYC)
<Task cond.>	Stating conditions of slot table. (START/ALWAYS/ERROR)
<Task pri.>	Priority of slot table. (1 – 31)

[Answer]

```
QoK
```

[Reference command]

```
1;2;SLOTSET2;REP;START;1
QoK
```

[Related commands]

SLOTRD

5.2.133 ENCBATTM (Battery remain time)

[Function]

The battery remain time is read.

[Format]

```
ENCBATTM
```

[Answer]

```
QoK<Power time>;<Remain time>
```

<Power time>	Power on time (Hr)
<Remain time>	Battery remaining time. (Hr)

[Reference command]

```
1;1;ENCBATTM
QoK51;11885
```

[Related commands]

PTIMEDEL=

5.2.134 BREAKON * (Release brake)

[Function]

The brake is released.

[Format]

```
BREAKON<Axis pattern>
```

<Axis pattern> Axis pattern by the HEX

```
00000000B
_______1    J1
______1_    J2
_____1__    J3
____1___    J4
```

___1____	J5
__1_____	J6
_1______	J7
1_______	J8

When 00 is specified, the brake is locked.

[Answer]

```
QoK
```

[Reference command]

```
1;1;BREAKON3F
QoK
```

[Related commands]

BREAKONF

5.2.135 BREAKONF * (Release brake)

[Function]

The brake is released. (No operation right required)

[Format]

```
BREAKONF<Axis pattern>
```

<Axis pattern> Axis pattern by the HEX

00000000B

_______1	J1
______1_	J2
_____1__	J3
____1___	J4
___1____	J5
__1_____	J6
_1______	J7
1_______	J8

When 00 is specified, the brake is locked.

[Answer]

```
QoK
```

[Reference command]

```
1;1;BREAKONF3F
QoK
```

[Related commands]

BREAKON

5.2.136 HOME * (Setting the origin)

[Function]

The origin is set.

[Format]

```
HOME<Org.type>;<Axis pattern>
```

<Org.type> 0: User origin

1: Mechanical stopper

2: (reserve)

3: Calibration jig

4: ABS

5: Eye mark

(6: Origin data input)

7: User ABS

<Axis pattern> Axis pattern by the HEX

```
00000000B
_______1   J1
______1_   J2
_____1__   J3
____1___   J4
___1____   J5
__1_____   J6
_1______   J7
1_______   J8
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;HOME3;3F
QoK
```

[Related commands]

DATINST

5.2.137 AXDATINST * (Additional axis add for DATINST and DATRD)

[Function]

The addition axis is added to the origin setting data by the data input. The argument of DATINST after the AXDATINST command is executed and the DATRD command becomes it with the addition axis.

[Format]

```
AXDATINST
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;AXDATINST
QoK
```

[Related commands]

DATINST, DATRD

5.2.138 DATINST * (Data input origin set)

[Function]

The origin is set by the data input.

[Format]

```
DATINST<J1 data>;<J2 data>;<J3 data>;<J4 data>;<J5 data>;<J6 data>;<Checksum>
or
DATINST<J1 data>;<J2 data>;<J3 data>;<J4 data>;<J5 data>;<J6 data>;<J7 data>;
    <J8 data>;<Checksum>
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;DATINSTZ2UOGQ;Z2VX5I;Z2#46Z;Z2NXW5;004QA5;0XB462;Y91%K5
QoK
```

[Related commands]

AXDATINST, DATRD

5.2.139 DATRD (Data input origin set)

[Function]

The origin data value by the data input is read.

[Format]

```
DATRD
```

[Answer]

```
QoK<J1 data>;<J2 data>;<J3 data>;<J4 data>;<J5 data>;<J6 data>;<Checksum>
or
QoK<J1 data>;<J2 data>;<J3 data>;<J4 data>;<J5 data>;<J6 data>;<J7 data>;
  <J8 data>;<Checksum>
```

[Reference command]

```
1;1;DATRD
QoKZ2UOGQ;Z2VX5I;Z2#46Z;Z2NXW5;004QA5;0XB462;Y91%K5
```

[Related commands]

AXDATINST, DATINST

5.2.140 RSTPWR (Reset power)

[Function]

Power supply reset (reboot) of the controller is executed.

[Format]

```
RSTPWR
```

[Answer]

```
QoK
```

[Reference command]

```
1;1;RSTPWR
QoK
```

[Related commands]

RPWRCHK=

5.2.141 RPWRCHK= (Reset power check)

[Function]

Check Power supply reset (reboot) of the controller.

[Format]

```
RPWRCHK=
```

[Answer]

```
QoK<Status>
```

<Status>	0:Impossible, 1:Possible

[Reference command]

```
1;1;RPWRCHK=
QoK1
```

[Related commands]

RSTPWR

5.2.142 MFTIME= (Maintenance forecast date)

[Function]

Read Maintenance forecast reset date.

[Format]

```
MFTIME=<Type>
```

<Type>	Read type. 1: Battery, 2: Grease, 3: Belt

[Answer]

```
QoK<J1 reset date>;<J2 reset date>;<J3 reset date>;<J4 reset date>;
<J5 reset date>;<J6 reset date>;<J7 reset date>;<J8 reset date>;
<J1 reset count>;<J2 reset count>;<J3 reset count>;<J4 reset count>;
<J5 reset count>;<J6 reset count>;<J7 reset count>;<J8 reset count>
```

<Jn reset date>	Reset date.
<Jn reset count>	Reset count.

[Reference command]

```
1;1;MFTIME=1
QoK2014/04/09;17:06:16;2014/04/09;17:06:16;2014/04/09;17:06:16;2014/04/09;17:06:16;201
4/04/09;17:06:16;2014/04/09;17:06:16;2014/04/09;17:06:16;2014/04/09;17:06:16;0;0;0;0;0;0;
```

```
0;0
```

[Related commands]

5.2.143 MFRST= (Maintenance forecast reset)

[Function]

Reset Maintenance forecast data.

[Format]

```
MFRST=<Axis bit>;<Reset bit>
```

<Axis bit> Axis bit pattern of HEX number. (00〜FF)

axis pattern

00000000B	JOINT
_______1	J1
______1_	J2
_____1__	J3
____1___	J4
___1____	J5
__1_____	J6
_1______	J7
1_______	J8

<Reset bit> Reset bit pattern of HEX number. (00〜FF)

reset pattern

00000000B	
_______1	Grease
______1_	Belt

[Answer]

```
QoK
```

[Reference command]

```
1;1;MFRST=20;1
```

```
QoK
```

[Related commands]

5.2.144 MFFCST= (Maintenance forecast read)

[Function]

Read Maintenance forecast data.

[Format]

```
MFFCST=<Type>
```

<Type> Read type.

0: Battery, 1: Grease, 2: Belt

[Answer]

```
QoK<J1 forecast hour>;<J2 forecast hour>;<J3 forecast hour>;<J4 forecast hour>;
<J5 forecast hour>;<J6 forecast hour>;<J7 forecast hour>;<J8 forecast hour>
```

<Jn forecast hour> Maintenance forecast hour.

[Reference command]

```
1;1;MFFCST=1
```

```
QoK23999;23999;23999;23999;23999;23999;0;0
```

[Related commands]

Index

ALIGN, 47
ATENA, 51
AUE<ON/OFF>, 50
AXDATINST, 96
BREAKON, 94, 95
BRKPTCLR, 52
BRKPTGET, 52
BRKPTSET, 51, 53, 54
CALIB, 57
CLOSE, 18
CNTL, 18
CSTOP, 44
CYCLECLR, 69
CYCLETIME, 69
DATINST, 97
DATRD, 97
DELETE, 28
DIN, 59
DIN=, 62
DOUT, 60
DOUT=, 60
DSTATE, 56
ECLR, 27
EDATA, 20
EDINS, 21
EFREE, 36, 37
EINS, 21
EMDAT, 21
ENCBATTM, 94
ERRLOG2=, 72
ERRLOGCLR, 73
ERROR, 70
ERRORLOG, 71
ERRORMES, 70
ERRSUM, 73
ERRSUM2=, 74
ERRSUMCLR, 75
EXEC, 26
FATTRIB, 33
FCHECK, 31
FCLOSE, 35
FCOPY, 32
FDEL, 32
FDIR, 30
FINIT, 34
FOPEN, 34
FREAD, 35
FRENAME, 33
FWRITE, 36
HND<ON/OFF>, 47
HNDSTS, 63
HOME, 96
HOT, 79
IN, 58
IN=, 62
INDMY, 61
INSET, 61
IOSIGNAL, 57
JOG, 48, 49
JPOS, 64, 65
KEYWD, 92
LINENO, 39
LINENO=, 39
LINERD, 40
LINESRD, 40
LISTCNT, 25
LISTI, 23
LISTL, 23
LISTP, 24
LOAD=, 19
LS<ON/OFF>, 50
MLOCK<ON/OFF>, 46
MOVSP, 48
NEW, 20
OPEN=, 17
OPNUMRD, 80
OPSTSRD, 80
OUT, 58
OUT=, 59
OVRD, 42
OVRD=, 42
PAR, 89
PAW=, 90
PAW2=, 90
PDIR, 29
PNR, 88
PPOS, 64, 65
PRG<UP/DOWN>, 38
PRGLOAD=, 37
PRGRD, 38
PRM=, 89, 92
PRMINIT, 88
PRMUNDO, 91
PTIME, 67
PTIMEDEL=, 68
RENUM, 28
RPOS, 64, 65
RSTALRM, 45
RSTIO, 46
RSTPRG, 45
RSTPWR, 98
RUN, 43
SAVE, 19
SLOTINIT, 45
SLOTRD, 93
SLOTSET, 93
SRV＜ON/OFF>, 41
STATE, 55
STEP, 27
STOP, 43
STOP<ON/OFF>, 44
STPSIG, 63
SUMDATE, 76
TIME, 66

TIME=, 67

USERAREASTS, 64

VAL, 76, 78

VAL=, 22

VALS, 77, 78

VTYPRD, 25

XPOS, 64, 65