YASKAWA

DX100/DX200/FS100 OPTIONS INSTRUCTIONS

REFERENCE MANUAL
FOR PROGRAMMING PENDANT CUSTOMIZATION FUNCTION
(MotoConnect.lib/MotoConnectFS.lib API SPECIFICATIONS
FOR DEVELPING eMbedded Visual C++)

Upon receipt of the product and prior to initial operation, read these instructions thoroughly, and retain for future reference.

MOTOMAN INSTRUCTIONS

MOTOMAN-☐☐☐INSTRUCTIONS
DX100/DX200 INSTRUCTIONS
DX100/DX200 OPERATOR'S MANUAL
DX100/DX200 MAINTENANCE MANUAL

FS100 INSTRUCTIONS
FS100 OPERATOR'S MANUAL
FS100 MAINTENANCE MANUAL

The DX100/DX200/FS100 operator's manual above corresponds to specific usage.
Be sure to use the appropriate manual.

Part Number: 166582-1CD
Revision: 0

YASKAWA
MOTOMAN ROBOTICS

MANUAL NO.
HW1481483①

1 Outline

In this manual, details of APIs used when the programming pendant customization application is developed with eMbedded Visual C++ are explained.

2 LIBRARY OPNE/CLOSE API

YppConnectOpen

Execute initialization of the library.

- **Syntax: DX100/DX200 MotoConnect.lib**
  ONG YppConnectOpen();

- **Parameter**
  No value

- **Return Value**
  0 : Normal end

  Other than 0 : Error

- **Syntax: FS100 MotoConnect.lib**
  LONG YppConnectOpen(WCHAR* pYpcAddr, WCHAR* pYppAddr);)S

- **Parameter**
  [in] *pYcpAddr*

  Pointer to the FS100 IP address

  [in] *pYppAddr*

  Pointer to the programming pendant IP address

- **Return Value**
  0 : Normal end

  Other than 0 : Error

YppConnectClose

Execute termination of the library.

- **Syntax**
  LONG YppConnectClose();
- **Parameter**
  No value
- **Return Value**
  0 : Normal end
  Other than 0 : Error

3 SYSTEM MONITOR API

YppGetVarData

Acquires the value of variable (B, I, D, R). More than two variables can be specified.

■ **Syntax**

LONG YppGetVarData(YPP_VAR_INFO* *sData*, LONG* *rData,* LONG *num*);

■ **Parameter**

[in] *sData*

Pointer to the variable specifying data structure

• YPP_VAR_INFO

Variable specifying data structure

Syntax:
```
typedef struct
{
    USHORT usType;
    USHORT usIndex;
} YPP_VAR_INFO;
```

Member: <*usType*> Variable type

Value	Description
YPP_RESTYPE_VAR_B	B variable
YPP_RESTYPE_VAR_I	I variable
YPP_RESTYPE_VAR_D	D variable
YPP_RESTYPE_VAR_R	R variable

<*usIndex*> Variable number

[out] *rData*

Pointer to the variable data

[in] *num*

Number of variable data

■ **Return value**

0 : Normal end

Other than 0 : Error

YppReadIO

Reads I/O. More than two I/Os can be acquired at a time.

■ **Syntax**

LONG YppReadIO(YPP_IO_INFO* *sData*, USHORT* *rData*, LONG *num*);

■ **Parameter**

[in] *sData*

i/O address specifying structure

• YPP_IO_INFO

I/O address specifying structure

Syntax: ypedef struct

{
ULONG*ulAddr;*
} YPP_IO_INFO;

Member: *<ulAddr>* I/O address

<DX100>

Value	Description
10 - 2567	Universal input #00010 - #02567 (2048)
10010 - 12567	Universal output #10010 - #12567 (2048)
30010 - 32567	External output #30010 - #32567 (2048)
40010 - 41607	Specific input #40010 - #41607 (1280)
50010 - 52007	Specific output #50010 - #52007 (1600)
60010 - 60647	I/F panel #60010 - #60647 (512)
70010 - 79997	Auxiliary relay #70010 - #79997 (7992)
80010 - 80647	Control input #80010 - #80647 (512)
82010 - 82207	Pseud input #82010 - #82207 (160)
25010 - 27567	Network input #25010 - #27567 (2048)
35010 - 37567	Network output #35010 - #37567 (2048)
1000000 - 1000559	Register #1000000 - #1000559 (560)

<DX200>

Value	Description
10 - 5127	Universal input #10010 - #05127 (4096)
10010 - 15127	Universal output #10010 - #15127 (4096)
20010 - 25127	External input #20010 - #25127 (4096)
30010 - 35127	External output #30010 - #35127 (4096)
40010 - 41607	Specific input #40010 - #41607 (1280)
50010 - 53007	Specific output #50010 - #53007 (2400)
60010 - 60647	I/F panel #60010 - #60647 (512)
70010 - 79997	Auxiliary relay #70010 - #79997 (7992)
80010 - 80647	Control input #80010 - #80647 (512)
82010 - 82207	Pseud input #82010 - #82207 (160)
27010 - 29567	Network input #27010 - #29567 (2048)
37010 - 39567	Network output #37010 - #39567 (2048)
1000000 - 1000559	Register #1000000 - #1000559 (560)

<FS100>

Value	Description
10 - 1278	Universal input #00010 - #01287 (1024)
10010 - 11278	Universal output #10010 - #11287 (1024)
20010 - 21278	External input #20010 - #21287 (1024)
30010 - 31278	External output #30010 - #31287 (1024)
40010 - 41607	Specific input #40010 - #41607 (1280)
50010 - 52007	Specific output #50010 - #52007 (1600)
60010 - 60647	I/F panel #60010 - #60647 (512)
70010 - 79997	Auxiliary relay #70010 - #79997 (7992)
80010 - 80647	Control input #80010 - #80647 (512)
82010 - 82207	Pseud input #82010 - #82207 (160)
25010 - 26287	Network input #25010 - #26287 (1024)
35010 - 36287	Network output #35010 - #36287 (1024)
1000000 - 1000559	Register #1000000 - #1000559 (560)

[out] *rData*

The value of I/O

[in] *num*

The number of I/Os

■ **Return value**

0 : Normal end

Other than 0 : Error

YppMonitor

Monitors (acquires) the value of the value and I/O in a batch.

■ **Syntax**

LONG YppMonitor
(YPP_MONITOR_INFO* *sData*, LONG* *rData*, LONG *infonum*);

■ **Parameter**

[in] *sData*

Pointer to the monitor information structure

• YPP_MONITOR_INFO

Syntax:
```
typedef struct
{
    ULONG   ulType;
    ULON    ulIndex;
} YPP_MONITOR_INFO;
```

Member: <*ulType*> The type of variable and I/O

Value	Description
YPP_RESTYPE_VAR_B	B variable
YPP_RESTYPE_VAR_I	I variable
YPP_RESTYPE_VAR_D	D variable
YPP_RESTYPE_VAR_R	R variable
YPP_RESTYPE_CIO	I/O

<*ulIndex*> variable number and I/O address

[out] *rData*

The value of the monitor information

[in] *infonum*

The number of monitors

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetPosVarData

Acquires the position-type variable.

■ **Syntax**

LONG YppGetPosVarData
(YPP_VAR_INFO* *sData*, LONG* *rData*, LONG *num*);

■ **Parameter**

[in] *sDatae*

Pointer (input) to the variable information structure array

• YPP_VAR_INFO

Variable information structure

Syntax:
```
typedef struct
{
    USHORT    usType;
    USHORT    usIndex;
} YPP_VAR_INFO;
```

Member: <*usType*> The type of the variable

Value	Description
YPP_RESTYPE_VAR_ROBOT	Robot
YPP_RESTYPE_VAR_BASE	Base
YPP_RESTYPE_VAR_STATION	Station

<*usIndex*> variable number

[out]*rData*

Area for pointer (output) LONG [10] *num to the position variable information is required.

Array	Description	
	Bit No.	**Content**
rData[0]	D05 - D00	Variable type 0 Pulse 16 Cartesian (base coordinates) 17 Cartesian (robot coordinates) 18 Cartesian (tool coordinates) 19 Cartesian (user coordinates) 20 Cartesian (reserved for master tool)
	D07 - D06	Reserved by manufacturer
	D08	0:Front 1:Back
	D09	0:Upper arm 1: Lower arm
	D10	0:Flip 1 :No flip
	D11	0:R<180deg 1:R>=180deg
	D12	0:T<180deg 1:T>=180deg
	D13	0:S<180de 1:S>=180deg
	D14 - D15	Reserved by manufacturer
	D16 - D21	Tool number (0 - 23)
	D22 - D27	User coordinate number
	D28 - D31	Reserved by manufacturer
rData[1]		(Extended attribute)
Array	**Pulse**	**Cartesian**
rData[2]	1st axis (S) pulse value	X-axis coordinate (unit: micron)
rData[3]	2nd axis (L) pulse value	Y-axis coordinate (unit: micron)
rData[4]	3rd axis (U) pulse value	Z-axis coordinate (unit: micron)
rData[5]	4th axis (R) pulse value	Wrist angle Rx (unit: 0.0001deg)
rData[6]	5th axis (B) pulse value	Wrist angle Ry (unit: 0.0001deg)
rData[7]	6th axis (T) pulse value	Wrist angle Rz (unit: 0.0001deg)
rData[8]	7th axis (E) pulse value	angle Re (unit: 0.0001deg)
rData[9]	8th axis pulse value	8th axis pulse value (micron in the case of traveling axis)

[in] *infonum*

The number of arrays

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetAlarmStatus

Acquires the error and alarm status.

■ **Syntax**

```
LONG   YppGetAlarmStatus
         ( YPP_ALARM_STATUS_RSP_DATA* rData );
```

■ **Parameter**

[out] *rData*

Pointer (output) to the alarm status receiving data structure

- YPP_ALARM_STATUS_RSP_DATA

Alarm status receiving data structure (output)

Syntax:

```
typedef struct
{
    SHORT    sIsAlarm;
    CHAR     reserved[2];
} YPP_ALARM_STATUS_RSP_DATA;
```

Member: <*sIsAlarm*> Error and alarm status

Bit value	Description
D00	1: In error status
D01	1: In alarm status
D02 - D15	(Unused)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetAlarmCode

Acquires the error and alarm code.

■ **Syntax**

```
LONG   YppGetAlarmCode
                    ( YPP_ALARM_CODE_RSP_DATA* rData );
```

■ **Syntax**

[out] *rData*

Pointer (output) to the alarm code receiving data structure

• PP_ALARM_CODE_RSP_DATA

Alarm code receiving data structure (output)

Syntax:

```
typedef struct
{
    USHORT      usErrorNo;
    USHORT      usErrorData;
    USHORT      usAlarmNum;
    YPP_ALARM_DATA      AlarmData;
} YPP_ALARM_CODE_RSP_DATA;
```

Member:

<*usErrorNo*>	Eror number
<*usErrorData*>	Arror data
<*usAlarmNum*>	The number of alarms
<*AlarmData*>	Alarm data

• YPP_ALARM_DATA

Alarm data structure (output)

Syntax:

```
#define MAX_ALARM_COUNT (4)

typedef struct
{
    USHORT      usAlarmNo[MAX_ALARM_COUNT];
    USHORT      usAlarmData[MAX_ALARM_COUNT];
} YPP_ALARM_DATA;
```

Member: *<usAlarmNo[MAX_ALARM_COUNT]>* Alarm number (4 at maximum)

<usAlarmData [MAX_ALARM_COUNT]> Alarm data (4 at maximum)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetMode

Acquires the operation mode of the system.

■ **Syntax**

LONG YppGetMode(YPP_MODE_RSP_DATA* *rData*);

■ **Parameter**

[out] *rData*

Pointer (output) to the mode receiving data structure

• YPP_MODE_RSP_DATA

Mode receiving data structure (output)

Syntax:
```
typedef struct
{
    SHORT   sMode;
    SHORT   sRemote;
} YPP_MODE_RSP_DATA;
```

Member: <*sMode*> Operation mode

Value	Description
1	Teach mode
2	Play mode

<*sRemote*> Command remote mode

Value	Description
0	Command remote OFF
1	Command remote ON

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetCycle

Acquires the cycle mode of the system.

■ Syntax

LONG YppGetCycle(YPP_CYCLE_RSP_DATA* *rData*);

■ Parameter

[out] *rData*

Pointer (output) to the cycle receiving data structure

• YPP_CYCLE_RSP_DATA

Cycle receiving data structure (output)

Syntax:
```
typedef struct
{
    SHORT    sCycle;
    CHAR     reserved[2];
} YPP_CYCLE_RSP_DATA;
```

Member: <*sCycle*> Cycle

Value	Description
0	Step
1	One Cycle
3	Auto

■ Return value

0 : Normal end

Other than 0 : Error

YppGetServoPower

Acquires the ON/OFF status of the servo power.

Syntax

LONG YppGetServoPower(YPP_SERVO_POWER_RSP_DATA* *rData*);

Parameter

[out] *rData*

Pointer (output) to the servo power supply status receiving data structure

- YPP_SERVO_POWER_RSP_DATA

Servo power supply status receiving data structure (output)

Syntax:

```
typedef struct
{
    SHORT   sServoPower;
    CHAR    reserved[2];
} YPP_SERVO_POWER_RSP_DATA;
```

Member: <*sServoPower*> Servo power

Value	Description
0	Servo Power OFF
1	Servo Power ON

Return value

0 : Normal end

Other than 0 : Error

YppGetPlayStatus

Acquires the operation status of the job.

■ Syntax

LONG YppGetPlayStatus(YPP_PLAY_STATUS_RSP_DATA* *rData*);

■ Parameter

[out] *rData*

Pointer (output) to the play status receiving data structure

- YPP_PLAY_STATUS_RSP_DATA

Play status receiving data structure (output)

```
Syntax:   typedef struct
          {
              SHORT   sStart;
              SHORT   sHold;
          } YPP_PLAY_STATUS_RSP_DATA
```

Member: <*sStart*> Operation status

Value	Description
0	Start OFF
1	Start ON

<*sHold*> Hold status

Value	Description
0	Hold OFF
1	Hold ON

■ Return value

0 : Normal end

Other than 0 : Error

YppGetMasterJob

Retrieves the master job name.

■ **Syntax**
LONG YppGetMasterJob(YPP_TASK_SEND_DATA* *sData*,
YPP_JOB_NAME_RSP_DATA* *rData*);

■ **Parameter**
[in] *sData*

Pointer to the data structure that specifies the task number

• YPP_TASK_SEND_DATA

Data structure that specifies the task number

Syntax:
```
typedef struct
{
    SHORT   sTaskNo;
    CHAR    reserved[2];
} YPP_TASK_SEND_DATA;
```

Member: <*sTaskNo*> Task numberr

Value	Description
0	Master Task
1	SubTask 1
2	SubTask 2
3	SubTask 3
4	SubTask 4
5	SubTask 5
6	SubTask 6
7	SubTask 7
8	SubTask 8
9	SubTask 9
10	SubTask 10
11	SubTask 11
12	SubTask 12
13	SubTask 13
14	SubTask 14
15	SubTask 15

[out] *rData*

Pointer to the data structure that receives the job name

- YPP_JOB_NAME_RSP_DATA

Data structure that receives the job name

Syntax:

```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    CHAR    cJobName[MAX_JOB_NAME_LEN];
    CHAR    reserved[3];
} YPP_JOB_NAME_RSP_DATA;
```

Menber <*cJobName[MAX_JOB_NAME_LEN]*> Master job name
(up to 32 characters for a job name)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetCurJob

Acquires the name, line, step number of the current job.

■ **Syntax**

LONG YppGetCurJob(YPP_TASK_SEND_DATA* *sData,*
YPP_CUR_JOB_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer (output) to the task No. transmitting data structure

- YPP_TASK_SEND_DATA

Task No. transmitting data structure (output)

Syntax:
```
typedef struct
{
    SHORT   sTaskNo;
    CHAR    reserved[2];
} YPP_TASK_SEND_DATA;
```

Member: <*sTaskNo*> Task No.r

Value	Description
0	Master Task
1	SubTask 1
2	SubTask 2
3	SubTask 3
4	SubTask 4
5	SubTask 5
6	SubTask 6
7	SubTask 7
8	SubTask 8
9	SubTask 9
10	SubTask 10
11	SubTask 11
12	SubTask 12
13	SubTask 13
14	SubTask 14
15	SubTask 15

Note: In the FS100 system, Sub task 6 and later tasks are invalid.

[out] *rData*

Pointer (output) to the current job information receiving data structure

- YPP_CUR_JOB_RSP_DATA

Current job information receiving data structure (output)

Syntax:

```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    USHORT    usJobLine;
    USHORT    usStep;
    CHAR      cJobName[MAX_JOB_NAME_LEN];
    CHAR      reserved[3];
} YPP_CUR_JOB_RSP_DATA;
```

Menber

<*usJobLine*> job line

<*usStep*> step

<*cJobName[MAX_JOB_NAME_LEN]*> job name
(32 characters at maximum)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetSpecialOpStatus

Retrieves the status of special operation.

■ **Syntax**

LONG YppGetSpecialOpStatus(YPP_SPECIAL_OP_RSP_DATA* *rData*);

■ **Parameter**

[out] *rData*

Pointer to the data structure that receives the special operation status

• YPP_SPECIAL_OP_RSP_DATA

Data structure that receives the special operation status

Syntax:
```
typedef struct
{
    SHORT   sSpecialOpStatus;
    CHAR    reserved[2];
} YPP_SPECIAL_OP_RSP_DATA;;
```

Member: <*sSpecialOpStatus*> Special operation (Play mode only)

Bit Value	Description
D0	Check operation
D1	Safety speed operation
D2	Dry-run speed operation
D3	Machine lock operation
D4	(Reserved)
D5	Low speed operation
D6	Weaving prohibited
D7-D9	(Reserved)
D10	Pressuring instruction prohibited
D11-D15	(Reserved)

■ **Return value**

0 : Normal end

Other than 0 : Error

■ **Restrictions**

This API is valid only in Play mode.

YppGetJobDate

Acquires YppGetJobDate job date.

■ **Syntax**

LONG YppGetJobDate(YPP_JOB_NAME_SEND_DATA* *sData,*
YPP_SYS_TIME_RSP_DATA* *rData)*;

■ **Parameter**

[in] *sData*

Pointer (output) to the job name transmitting data structure

- YPP_JOB_NAME_SEND_DATA

Job name transmitting data structure (output)

Syntax:
```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    CHAR    cJobName[MAX_JOB_NAME_LEN];
    CHAR    reserved[3];
} YPP_JOB_NAME_SEND_DATA;
```

Member: <*cJobName[MAX_JOB_NAME_LEN]*> job name
(32 characters at maximum)

[out] *rData*

Pointer (output) to the system time receiving data structure

YPP_SYS_TIME_RSP_DATA

System time receiving data structure (output)

Syntax:

```
typedef struct
{
    SHORT   sStartYear;
    SHORT   sStartMonth;
    SHORT   sStartDay;
    SHORT   sStartHour;
    SHORT   sStartMin;
    SHORT   sStartSec;
    LONG    lElapsedTime;
} YPP_SYS_TIME_RSP_DATA;
```

Member:

<*sStartYear*>	Year
<*sStartMonth*>	Month
<*sStartDay*>	Day
<*sStartHour*>	Hour
<*sStartMin*>	Mminute
<*sStartSec*>	Second
<*lElapsedTime*>	Unused

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetCartPosEx

Acquires the current position in Cartesian Coordinates with a specified coordinate frame (Robot, Base, User).

■ Syntax

LONG YppGetCartPosEx(YPP_CARTPOS_EX_SEND_DATA* *sData*, YPP_CART_POS_EX_RSP_DATA**rData*);

■ Parameter

[[in] *sData*

Pointer (input) to the current position transmitting data structure

- YPP_CARTPOS_EX_SEND_DATA

Current position transmitting data structure (input)

```
Syntax:   typedef struct
          {
              SHORT   sRobotNo;
              SHORT   sFrame;
              SHORT   sToolNo;
              CHAR    reserved[2];
          } YPP_CARTPOS_EX_SEND_DATA;
```

Member: <*sRobotNo*> Robot number

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)

Note: In the FS100 system, R5 (Robot 5) and later values are invalid.

[*sFrameNO*] specified coordinate frame

Value	Description
0	Base coordinate
1	Robot coordinate
2, 3, …	User coordinate1, 2, …

[in] *ToolNo* Tool number 0 to 63

Note: In the FS100 system the tool number is from 1 to 15.

[out] *rData*

- Pointer (output) to the current position receiving data structure
- YPP_CART_POS_RSP_DATA

Current position receiving data structure (output)

Syntax:

```
#define MAX_CART_AXES_EX  (12)

typedef struct
{
    LONG    lPos[MAX_CART_AXES_EX];
    SHORT   sConfig;
    CHAR    reserved[2];
} YPP_CART_POS_EX_RSP_DATA;
```

Member: *<lPos[MAX_CART_AXES_EX]>* Position data (12 at maximum).

Value	Description
lPos[0]	X-axis coordinate (unit: micron)
lPos[1]	Y-axis coordinate (unit: micron)
lPos[2]	Z-axis coordinate (unit: micron)
lPos[3]	Wrist angle Rx (unit: 0.0001deg)
lPos[4]	Wrist angle Ry (unit: 0.0001deg)
lPos[5]	Wrist angle Rz (unit: 0.0001deg)
lPos[6]	angle Re (unit: 0.0001deg)
lPos[7]	(Reserved)
lPos[8]	1st external axis pulse value (micron in the case of traveling axis)
lPos[9]	2nd external axis pulse value (micron in the case of traveling axis)
lPos[10]	3rd external axis pulse value (micron in the case of traveling axis)
lPos[11]	(Unused)

[*sConfig]* Figure information

Value	Description
D00	0:Front 1:Back
D01	0:Upper arm 1:Lower arm
D02	0:Flip 1:No flip
D03	0:R<180deg 1:R>=180deg
D04	0:T<180deg 1:T>=180deg
D05	0:S<180deg 1:S>=180deg
D06	0:L<0deg 1:L>=0deg
D07 -D15	Reserved

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetPulsePos

Acquires the current position in pulse count

■ **Syntax**

LONG YppGetPulsePos(YPP_CTRL_GRP_SEND_DATA* *sData,*
YPP_PULSE_POS_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer (input) to the control group transmitting data structure

• YPP_CTRL_GRP_SEND_DATA

Control group transmitting data structure (input)

Syntax:
```
typedef unsigned long CTRLG_T;

typedef struct
{
    CTRLG_T    CtrlGrp;
} YPP_CTRL_GRP_SEND_DATA;
```

Member: <*CtrlGrp*> control group

<DX100/DX200>

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<FS100>

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	(Reserve)
5	(Reserve)
6	(Reserve)
7	(Reserve)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	(Reserve)
13	(Reserve)
14	(Reserve)
15	(Reserve)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)

[out] *rData*

Pointer (output) to the pulse coordinate position receiving data structure

- YPP_PULSE_POS_RSP_DATA

Pulse coordinate position receiving data structure

Syntax:

```
#define MAX_PULSE_AXES (8)

typedef struct
{
    LONG    lPos[MAX_PULSE_AXES];
} YPP_PULSE_POS_RSP_DATA;
```

Member: *<lPos[MAX_PULSE_AXES]>* Pulse coordinate position data (12 at maximum)

Value	Description
lPos[0]	1st axis (S) pulse value
lPos[1]	2nd axis (L) pulse value
lPos[2]	3rd axis (U) pulse value
lPos[3]	4th axis (R) pulse value
lPos[4]	5th axis (B) pulse value
lPos[5]	6th axis (T) pulse value
lPos[6]	7th axis (E) pulse value
lPos[7]	8th axis pulse value

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetFBPulsePos

Retrieves the feedback position in pulse count.

■ **Syntax**
LONG YppGetFBPulsePos(YPP_CTRL_GRP_SEND_DATA* *sData,*
YPP_FB_PULSE_POS_RSP_DATA* *rData*);

■ **Parameter**
[in] *sData*

Pointer to the data structure that specifies the control group

• YPP_CTRL_GRP_SEND_DATA

Data structure that specifies the control group

Syntax:
```
typedef unsigned long CTRLG_T;

typedef struct
{
    CTRLG_T    CtrlGrp;
} YPP_CTRL_GRP_SEND_DATA;
```

Member: <*CtrlGrp*> control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

[out] *rData*

Pointer to the data structure that receives the feedback position coordinates in pulse

- YPP_FB_PULSE_POS_RSP_DATA

Data structure that receives the feedback position coordinates in pulse

Syntax:
```
#define MAX_PULSE_AXES (8)

typedef struct
{
    LONG    lPos[MAX_PULSE_AXES];
} YPP_FB_PULSE_POS_RSP_DATA;
```

Member: *<lPos[MAX_PULSE_AXES]>* Position coordinates in pulse (up to 8 arrays)

Value	Description
lPos[0]	1st axis (S) pulse value
lPos[1]	2nd axis (L) pulse value
lPos[2]	3rd axis (U) pulse value
lPos[3]	4th axis (R) pulse value
lPos[4]	5th axis (B) pulse value
lPos[5]	6th axis (T) pulse value
lPos[6]	7th axis (E) pulse value
lPos[7]	8th axis pulse value

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetServoSpeed

Retrieves the current servo speed by seconds.

■ **Syntax**
LONG YppGetServoSpeed(YPP_CTRL_GRP_SEND_DATA* *sData*, YPP_SERVO_SPEED_RSP_DATA* *rData*);

■ **Parameter**
[in] *sData*

Pointer to the data structure that specifies the control group

• YPP_CTRL_GRP_SEND_DATA

Data structure that specifies the control group

Syntax:
```
typedef unsigned long CTRLG_T;

typedef struct
{
    CTRLG_T    CtrlGrp;
} YPP_CTRL_GRP_SEND_DATA;
```

Member: <*CtrlGrp*> control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

[out] *rData*

Pointer to the data structure that receives the servo speed

- YPP_SERVO_SPEED_RSP_DATA

Data structure that receives the servo speed

Syntax:

```
#define MAX_PULSE_AXES (8)

typedef struct
{
    LONG    lSpeed[MAX_PULSE_AXES];
} YPP_SERVO_SPEED_RSP_DATA;
```

Member: <*lSpeed[MAX_PULSE_AXES]*> Speed (up to 8 arrays)

Value	Description
Speed[0]	1st axis (S) direction (unit: pulses per second - pps)
Speed[1]	2nd axis (L) direction (unit: pps)
Speed[2]	3rd axis (U) direction (unit: pps)
Speed[3]	4th axis (R) direction (unit: pps)
Speed[4]	5th axis (B) direction (unit: pps)
Speed[5]	6th axis (T) direction (unit: pps)
Speed[6]	7th axis (E) direction (unit: pps)
Speed[7]	8th axis direction (unit: pps)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetFBSpeed

Retrieves the feedback speed by seconds.

■ **Syntax**

LONG YppGetFBSpeed(YPP_CTRL_GRP_SEND_DATA* *sData*, YPP_FB_SPEED_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer to the data structure that specifies the control group

- YPP_CTRL_GRP_SEND_DATA

Data structure that specifies the control group

Syntax:

```
typedef unsigned long CTRLG_T;

typedef struct
{
    CTRLG_T    CtrlGrp;
} YPP_CTRL_GRP_SEND_DATA;
```

Member: <*CtrlGrp*> control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

[out] *rData*

Pointer to the data structure that receives the feedback speed

- YPP_FB_SPEED_RSP_DATA

Data structure that receives the feedback speed

Syntax:

```
#define MAX_PULSE_AXES (8)

typedef struct
{
    LONG    lSpeed[MAX_PULSE_AXES];
} YPP_FB_SPEED_RSP_DATA;
```

Member: <*lSpeed[MAX_PULSE_AXES]*> Speed (up to 8 arrays)

Value	Description
lSpeed[0]	1st axis (S) direction (unit: pulses per second - pps)
lSpeed[1]	2nd axis (L) direction (unit: pps)
lSpeed[2]	3rd axis (U) direction (unit: pps)
lSpeed[3]	4th axis (R) direction (unit: pps)
lSpeed[4]	5th axis (B) direction (unit: pps)
lSpeed[5]	6th axis (T) direction (unit: pps)
lSpeed[6]	7th axis (E) direction (unit: pps)
lSpeed[7]	8th axis direction (unit: pps)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetTorque

Retrieves the percentage to the maximum current servo torque value.

■ **Syntax**

LONG YppGetTorque(YPP_CTRL_GRP_SEND_DATA* *sData*, YPP_TORQUE_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer to the data structure that specifies the control group

- YPP_CTRL_GRP_SEND_DATA

Data structure that specifies the control group

Syntax:
```
typedef struct
{
    CTRLG_T    CtrlGrp;
} YPP_CTRL_GRP_SEND_DATA;
```

Member: <*CtrlGrp*> control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

[out] *rData*

Pointer to the data structure that receives the torque value

- YPP_TORQUE_RSP_DATA

Data structure that receives torque the value

Syntax:

```
#define MAX_PULSE_AXES (8)

typedef struct
{
    LONG    lTorquePcnt[MAX_PULSE_AXES];
} YPP_TORQUE_RSP_DATA;
```

Member: *<lTorquePcnt[MAX_PULSE_AXES]>* Torque percentage (up to 8 arrays)

Value	Description
TorquePercent[0]	1st axis (S) direction (unit: 0.01% of max torque)
TorquePercent[1]	2nd axis (L) direction (unit: 0.01%)
TorquePercent[2]	3rd axis (U) direction (unit: 0.01%)
TorquePercent[3]	4th axis (R) direction (unit: 0.01%)
TorquePercent[4]	5th axis (B) direction (unit: 0.01%)
TorquePercent[5]	6th axis (T) direction (unit: 0.01%)
TorquePercent[6]	7th axis (E) direction (unit: 0.01%)
TorquePercent[7]	8th axis direction (unit: 0.01%)

■ **Return value**

0 : Normal end

Other than 0 : Error

YppGetSysTimes

Acquires the current system time.

■ **Syntax**

LONG YppGetSysTimes(YPP_SYS_TIME_SEND_DATA* *sData*,
YPP_SYS_TIME_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer (input) to the transmitting data for aquiring system time structure

- YPP_SYS_TIME_SEND_DATA

Transmitting data for aquiring system time structure (input)

System:
```
typedef struct
{
    SHORT   sTimeType;
    CHAR    reserved[2];
} YPP_SYS_TIME_SEND_DATA;
```

Member: *<sTimeType>* Type of the system time

Value	Description
YPP_POWER_ON_TIME_ID	Time during the power supply of the controller is ON
YPP_SERVO_ON_TIME_ID	Time during the servo power supply is ON
YPP_PLAYBACK_TIME_ID	Time during the play back operation
YPP_MOVING_TIME_ID	Moving time
YPP_OPERATION_TIME_ID	Operation time

[out] *rData*

Pointer (output) to the system time receiving data structure

- YPP_SYS_TIME_RSP_DATA

System time receiving data structure (output)

Syntax:

```
typedef struct
{
    SHORT   sStartYear;
    SHORT   sStartMonth;
    SHORT   sStartDay;
    SHORT   sStartHour;
    SHORT   sStartMin;
    SHORT   sStartSec;
    LONG    lElapsedTime;
} YPP_SYS_TIME_RSP_DATA;
```

Member:

<*sStartYear*>	Year
<*sStartMonth*>	Month
<*sStartDay*>	Day
<*sStartHour*>	Hour
<*sStartMin*>	Minute
<*sStartSec*>	Second
<*lElapsedTime*>	Elapsed time

■ **Return value**

0 : Normal end

Other than 0: Error

YppGetJogSpeed

Acquires the current jog speed.

Syntax

LONG YppGetJogSpeed(YPP_JOGSPEED_RSP_DATA* *rData*);

Parameter

[out] *rData*

Pointer (output) to the jog speed receiving data structure

- YPP_JOGSPEED_RSP_DATA

Jog speed receiving data structure (output)

Syntax:
```
typedef struct
{
    SHORT   sJogSpeed;
    CHAR    reserved[2];
} YPP_JOGSPEED_RSP_DATA;
```

Member: <*sJogSpeed*> Jog speed

Value	Description
0	Inching
1	Low speed
2	Middle speed
3	High speed
4	Maxinum speed

Return value

0 : Normal end

Other than 0 : Error

YppGetJogCoord

Acquires the current jog coordinates.

■ Syntax

LONG YppGetJogCoord(YPP_JOGCOORD_RSP_DATA* *rData*);

■ Parameter

[out] *rData*

Pointer (output) to the jog code receiving data structure

- YPP_JOGCOORD_RSP_DATA

Jog code receiving data structure (output)

```
Syntax:   typedef struct
          {
              SHORT   sJogCoord;
              CHAR    reserved[2];
          } YPP_JOGCOORD_RSP_DATA;
```

Member: <*sJogCoord*> Jog code

Value	Description
0	Joint coordinates
1	Cartesian coordinates
2	Cylindrical coordinates
3	Tool coordinates
4	User coordinates
5	External reference point coordinates
6	Teaching line coordinates

■ Return value

0 : Normal end

Other than 0 : Error

YppGetSVarInfo

Acquires the value (character strings) seto to S variable. More than one variables can be specified.

Syntax

LONG YppGetSVarInfo
(YPP_VAR_INFO* *sData,* YPP_SVAR_RECV_INFO* *rData,* LONG *num*);

Parameter

[in] *sData*

Pointer to variable specifying data structure

- YPP_SVAR_INFO

Variable specifying data structure

Syntax:
```
typedef struct
{
    USHORT    usType;
    USHORT    usIndex;
} YPP_VAR_INFO;
```

Member: <*usType*> Variable type

Value	Description
YPP_RESTYPE_VAR_S	S variable

<*usIndex*> Variable number

[out] *rData*

Pointer to the variable data

- YPP_SVAR_RECV_INFO

S variable data structure

Syntax:
```
typedef struct
{
    UCHAR   ushSVar[17];
    Note: Maximum character used for the S variable is (16) + 1
    CHAR    reserved[3];
} YPP_VAR_INFO;
```

Member: *<ushSVar> Character strings set to the S variable*

[in] *num*

The number of variable data (10 at maximum)

■ **Return value**

0 : Normal end

Other than 0 : Error

4 SYSTEM CONTROL API

YppSetVarData

Writes the variable.

■ **Syntax**

LONG YppPutVarData(YPP_VAR_DATA* *sData*, LONG *num*);

■ **Parameter**

[in] *sData*

Pointer to the variable data writing structure

• YPP_VAR_DATA

Variable data writing structure

Syntax:
```
typedef struct
{
    USHORT    usType;
    USHORT    usIndex;
    LONG      ulValue;
} YPP_VAR_DATA;
```

Member: <*usType*> Variable type

Value	Description
YPP_RESTYPE_VAR_B	B variable
YPP_RESTYPE_VAR_I	I variable
YPP_RESTYPE_VAR_D	D variable
YPP_RESTYPE_VAR_R	R variable

<*usIndex*> Variable index

<*ulValue*> Variable value

[in] *num*

The number of variable data

■ **Return value**

0 : Normal end

Other than 0 : Error

■ **Relevant parameter**

Number	Function	Setting value	Default
S2C541	Permission of variable and I/O writing during play mode	0: Permitted 1: Not permitted	1 (Not permitted)
S2C542	Permission of variable and I/O writing during edit lock	0: Permitted 1: Not permitted	1 (Not permitted)

CAUTION

When “0” is set to “S2C541”, writing operation during play back mode becomes valid. However, please operate the manipulator with caution since this operation may influence the cycle time.

CAUTION

The “edit lock status” to which S2C542 can specify are following status.

– In alarm status
– External memory is being used
– Data transmitting function is being used
– Specified input EDIT_LOCK(#40064) is turned ON.

YppWriteIO

Writes I/O.

■ **Syntax**
LONG YppWriteIO(YPP_IO_DATA* *sData,* LONG *num*);

■ **Parameter**
[in] *sData*

Pointer to the I/O data writing structure

• YPP_IO_DATA

I/O data writing structure

Syntax:
```
typedef struct
{
    ULONG  ulAddr;
    ULONG  ulValue;
} YPP_IO_DATA;
```

Member: *<ulAddr>* I/O address

<DX100>

Value	Description
10010 to12567	Universal output #10010 - #12567(2048)
60010 - 60647	I/F panel #60010 - #60647(512)
25010 - 27567	Network input #25010 - #27567(2048)
1000000 - 1000559	Register #1000000?#1000559(560)

<DX200>

Value	Description
10010 - 15127	Universal output #10010 - #15127 (4096)
60010 - 60647	I/F panel #60010 - #60647 (512)
27010 - 29567	Network input #27010 - #29567 (2048)
1000000 - 1000559	Register #1000000 - #1000559 (560)

<FS100>

Value	Description
1001 - 11287	Universal output #10010 - #11287(1024)
60010 - 60647	I/F panel #60010 - #60647(512)
25010 - 26287	Network input #25010 - #26287(1024)
1000000 - 1000559	Register #1000000 - #1000559(560)

<*ulValue*> I/Ovalue

[in] *num*

The number of I/O data

■ **Return value**

0 : Normal end

Other than 0 : Error

■ **Relevant parameter**

Number	Function	Setting value	Default
S2C541	Permission of variable and I/O writing during play mode	0: Permitted 1: Not permitted	1 (Not permitted)
S2C542	Permission of variable and I/O writing during edit lock	0: Permitted 1: Not permitted	1 (Not permitted)

CAUTION

When “0” is set to “S2C541”, writing operation during play back mode becomes valid. However, please operate the manipulator with caution since this operation may influence the cycle time.

CAUTION

The “edit lock status” to which S2C542 can specify are following status.

– In alarm status
– External memory is being used
– Data transmitting function is being used
– Specified input EDIT_LOCK(#40064) is turned ON.

YppPutPosVarData

Sets the position-type variable

■ **Syntax**

LONG YppPutPosVarData(YPP_POSVAR_DATA* *sData,* LONG *num*);

■ **Parameter**

[in] *sData*

Pointer (input) to the array of position-type variable setting information structure

• YPP_POSVAR_DATA

Position-type variable setting information structure (input)

Syntax:
```
typedef struct
{
    USHORT    usType;
    USHORT    usIndex;
    LONG      ulValue[10];
} YPP_POSVAR_DATA;
```

Member: <*usType*> Type of variable

Value	Description
YPP_RESTYPE_VAR_ROBOT	Robot
YPP_RESTYPE_VAR_BASE	Base
YPP_RESTYPE_VAR_STATION	Station

<*usIndex*> The number of variables

<*ulValue[10]*> Position data (defined value)

Array	**Description**	
	Bit No.	**Content**
rData[0]	D05 - D00	Variable type 0 Pulse 16 Cartesian (base coordinates) 17 Cartesian (robot coordinates) 18 Cartesian (tool coordinates) 19 Cartesian (user coordinates) 20 Cartesian (reserved for master tool)
	D07 - D06	Reserved by manufacturer
	D08	0:Front 1:Back
	D09	0:Upper arm 1: Lower arm
	D10	0:Flip 1:No flip
	D11	0:R<180deg 1:R>=180deg
	D12	0:T<180deg 1:T>=180deg
	D13	0:S<180de 1:S>=180deg
	D14 - D15	Reserved by manufacturer
	D16 - D21	Tool number (0 - 23)
	D22 - D27	User coordinate number
	D28 - D31	Reserved by manufacturer
rData[1]		(Extended attribute)
Array	**Pulse**	**Cartesian**
rData[2]	1st axis (S) pulse value	X-axis coordinate (unit: micron)
rData[3]	2nd axis (L) pulse value	Y-axis coordinate (unit: micron)
rData[4]	3rd axis (U) pulse value	Z-axis coordinate (unit: micron)
rData[5]	4th axis (R) pulse value	Wrist angle Rx (unit: 0.0001deg)
rData[6]	5th axis (B) pulse value	Wrist angle Ry (unit: 0.0001deg)
rData[7]	6th axis (T) pulse value	Wrist angle Rz (unit: 0.0001deg)
rData[8]	7th axis (E) pulse value	angle Re (unit: 0.0001deg)
rData[9]	8th axis pulse value	8th axis pulse value (micron in the case of traveling axis)

[in] *num*

The number of variable data

■ **Return value**

0 : Normal end

Other than 0 : Error

■ **Relevant parameter**

Number	Function	Setting value	Default
S2C541	Permission of variable and I/O writing during play mode	0: Permitted 1: Not permitted	1 (Not permitted)
S2C542	Permission of variable and I/O writing during edit lock	0: Permitted 1: Not permitted	1 (Not permitted)

CAUTION

When “0” is set to “S2C541”, writing operation during play back mode becomes valid. However, please operate the manipulator with caution since this operation may influence the cycle time.

CAUTION

The “edit lock status” to which S2C542 can specify are following status.

– In alarm status
– External memory is being used
– Data transmitting function is being used
– Specified input EDIT_LOCK(#40064) is turned ON.

YppCancelError

Releases the error status.

■ **Syntax**

LONG YppCancelError(YPP_STD_RSP_DATA* *rData*);)

■ **Parameter**

[out] *rData*

Pointer (output) to the basic receiving data structure

• YPP_STD_RSP_DATA

Basic receiving data structure (output)

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Normal end

■ **Return value**

0 : Normal end

Other than 0 : Error

YppResetAlarm

Resets the alarm

■ **Syntax**

LONG YppResetAlarm(YPP_STD_RSP_DATA* *rData*);

■ **Parameter**

[out] *rData*

Pointer (output) to the basic receiving data structure

- YPP_STD_RSP_DATA

Basic receiving data structure (output)

```
Syntax:   typedef struct
          {
              USHORT    err_no;
              CHAR      reserved[2];
          } YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Normal

■ **Return value**

0 : Normal end

Other than 0 : Error

YppSetCycle

Sets the cycle mode.

Syntax

LONG YppSetCycle(YPP_CYCLE_SEND_DATA* *sData,*
YPP_STD_RSP_DATA* *rData*);

Parameter

[in] *sData*

Pointer (input) to the cycle transmitting data structure

- YPP_CYCLE_SEND_DATA

Cycle transmitting data structure (input)

Syntax:
```
typedef struct
{
    SHORT   sCycle;
    CHAR    reserved[2];
} YPP_CYCLE_SEND_DATA;
```

Member: <*sCycle*> Cycle

Value	Description
1	Step
2	One Cycle
3	Auto

[out] *rData*

Pointer (output) to the basic receiving data structure

- YPP_STD_RSP_DATA

Basic receiving data structure (output)

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Normal
0x2060	In alarm status

■ **Return value**

0 : Normal end

Other than 0 : Error

YppSetServoPower

Sets ON/OFF of the servo power.

■ **Syntax**

LONG YppSetServoPower(YPP_SERVO_POWER_SEND_DATA* *sData*, YPP_STD_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer to the data structure that transmits the ON/OFF of servo power

- YPP_SERVO_POWER_SEND_DATA

Data structure that transmits the ON/OFF of servo power

Syntax:
```
typedef struct
{
    SHORT   sServoPower;
    CHAR    reserved[2];
} YPP_SERVO_POWER_SEND_DATA;
```

Member: <*sServoPower*> Servo power

Value	Description
0	Servo Power OFF
1	Servo Power ON

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Succeeded
0x2060	In alarm/error status (The case of Servo ON only)
0x3450	Servo power ON failed

■ **Return value**

0 : Normal end

Other than 0 : Error

YppSetMasterJob

Registers a specified job as a master job.

Syntax

LONG YppSetMasterJob(YPP_MASTER_JOB_SEND_DATA* *sData*, YPP_STD_RSP_DATA* *rData*);

Parameter

[in] *sData*

Pointer to the data structure that transmits the master job information

- YPP_MASTER_JOB_SEND_DATA

Data structure that transmits the master job information

Syntax:

```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    SHORT   sTaskNo;
    CHAR    cJobName[MAX_JOB_NAME_LEN];
    CHAR    reserved[5];
} YPP_MASTER_JOB_SEND_DATA;
```

Member: <*sTaskNo*> Task numberr

Value	Description
0	Master Task
1	SubTask 1
2	SubTask 2
3	SubTask 3
4	SubTask 4
5	SubTask 5
6	SubTask 6
7	SubTask 7
8	SubTask 8
9	SubTask 9
10	SubTask 10
11	SubTask 11
12	SubTask 12
13	SubTask 13
14	SubTask 14
15	SubTask 15

<*cJobName[MAX_JOB_NAME_LEN]*> Job name
(up to 32 characters for a job name)

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2060	In alarm/error status
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Normal end

Other than 0 : Error

YppSetCurJob

Set the specified job and line number as the current job.

■ **Syntax**

LONG YppSetCurJob(YPP_CUR_JOB_SEND_DATA* *sData,*
YPP_STD_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer (input) to the current job information transmitting data structure

• YPP_CUR_JOB_SEND_DATA

Current job information transmitting data structure (input)

Syntax:

```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    USHORT    usJobLine;
    CHAR      cJobName[MAX_JOB_NAME_LEN];
    CHAR      reserved[5];
} YPP_CUR_JOB_SEND_DATA;
```

Member: <*usJobLine*> Job line

<*cJobName[MAX_JOB_NAME_LEN]*>
Job name (Job name: 32 characters at maximum)

[out] *rData*

Pointer (output) to basic receiving data structure

- YPP_STD_RSP_DATA

Basic receiving data structure (output)

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
    } YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Normal
0x2010	Manipulator is in operation
0x2060	In alarm/error status
0x2110	Inaccessible job
0x4040	Specified job not found
0x5200	Specified Line No. is out of range

■ **Return value**

0 : Normal end

Other than 0 : Error

YppStartJob

Registers a specified job as a master job and starts it.

Syntax

LONG YppStartJob(YPP_START_JOB_SEND_DATA* *sData*, YPP_STD_RSP_DATA* *rData*);

Parameter

[in] *sData*

Pointer to the data structure that transmits the job to be started

- YPP_START_JOB_SEND_DATA

Data structure that transmits the job to be started

Syntax:

```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    SHORT   sTaskNo;
    CHAR    cJobName[MAX_JOB_NAME_LEN];
    CHAR    reserved[5];
} YPP_START_JOB_SEND_DATA;
```

Member: <*sTaskNo*> Task number

Value	Description
0	Master Task
1	SubTask 1
2	SubTask 2
3	SubTask 3
4	SubTask 4
5	SubTask 5
6	SubTask 6
7	SubTask 7
8	SubTask 8
9	SubTask 9
10	SubTask 10
11	SubTask 11
12	SubTask 12
13	SubTask 13
14	SubTask 14
15	SubTask 15

<*cJobName[MAX_JOB_NAME_LEN]*> Job name
(up to 32 characters for a job name)

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Normal end

Other than 0 : Error

YppHold

Turns ON/OFF the hold function.

■ **Syntax**

LONG YppHold(YPP_HOLD_SEND_DATA* *sData*,
YPP_STD_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer to the data structure that transmits the ON/OFF of hold function

• YPP_HOLD_SEND_DATA

Data structure that transmits the ON/OFF of hold function

Syntax:
```
typedef struct
{
    SHORT   sHold;
    CHAR    reserved[2];
} YPP_HOLD_SEND_DATA;
```

Member: <*sHold*> Hold status

Value	Description
0	Hold OFF
1	Hold ON

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Succeeded

■ **Return value**

0 : Normal end

Other than 0 : Error

YppWaitForJobEnd

Waits for the job completion or expiration of specified time duration.

Syntax

LONG YppWaitForJobEnd(YPP_WAIT_JOB_SEND_DATA* *sData*, YPP_STD_RSP_DATA* *rData*);

Parameter

[in] *sData*

Pointer to the data structure that transmits the WaitJOB

- YPP_WAIT_JOB_SEND_DATA

Data structure that transmits the WaitJOB

Syntax:
```
typedef struct
{
    SHORT    sTaskNo;
    SHORT    sTime;
} YPP_WAIT_JOB_SEND_DATA;
```

Member: <*sTaskNo*> Task number

Value	Description
0	Master Task
1	SubTask 1
2	SubTask 2
3	SubTask 3
4	SubTask 4
5	SubTask 5
6	SubTask 6
7	SubTask 7
8	SubTask 8
9	SubTask 9
10	SubTask 10
11	SubTask 11
12	SubTask 12
13	SubTask 13
14	SubTask 14
15	SubTask 15

<*sTime*> Waiting time (If “-1” is specified, waits unlimitedly)

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: *<err_no>* Error number

Value	Description
0x0000	Succeeded
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0xFFFF	Other error

■ **Return value**

0 : Normal end

Other than 0 : Error

YppDeleteJob

Deletes the specified job.

■ **Syntax**

LONG YppDeleteJob(YPP_DELETE_JOB_SEND_DATA* *sData,*
YPP_STD_RSP_DATA* *rData*);

■ **Parameter**

[in] *sData*

Pointer (input) to the deleted job information transmitting data structure

• YPP_DELETE_JOB_SEND_DATA

Deleted job information transmitting data structure

Syntax:
```
#define MAX_JOB_NAME_LEN (33)

typedef struct
{
    CHAR    cJobName[MAX_JOB_NAME_LEN];
    CHAR    reserved[7];
} YPP_DELETE_JOB_SEND_DATA;
```

Member: *<cJobName[MAX_JOB_NAME_LEN]>*
Job name (Job name: 32 characters at maximum)

[out] *rData*

Pointer (output) to the basic receiving data structure

• YPP_STD_RSP_DATA

Basic receiving data structure (output)

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Normal
0x2010	Manipulator is in operation or cannot delete JOBs during the robot operation
0x2060	In alarm/error status
0x2080	Play mode
0x2090	Accessing to the specified job
0x4020	Edit lock job
0x4040	Specified JOB not found

■ **Return value**

0 : Normal end

Other than 0 : Error

YppPutSVarInfo

Writes the S variable

■ **Syntax**

LONG YppPutSVarInfo(YPP_SVAR_INFO* *sData*, LONG *num*);

■ **Parameter**

[in] *sData*

Pointer to the variable data write structure

• YPP_SVAR_DATA

Variable data write structure

Syntax:
```
typedef struct
{
    USHORT    usType;
    USHORT    usIndex;
    UCHAR     uchValue[17];
              Note: Maximum character used for the S variable is (16) + 1
    CHAR      reserved[3];
} YPP_VAR_DATA;
```

Member: <*usType*> Variable type

Value	Description
YPP_RESTYPE_VAR_S	S variable

<*usIndex*> Variable index

<*uchValue*> Character strings defined to the S variable

[in] *num*

The number of variable data

■ **Return value**

0 : Normal end

Other than 0 : Error

■ **Relevant parameter**

Number	Function	Setting value	Default
S2C541	Permission of variable and I/O writing during play mode	0: Permitted 1: Not permitted	1 (Not permitted)
S2C542	Permission of variable and I/O writing during edit lock	0: Permitted 1: Not permitted	1 (Not permitted)

CAUTION

When “0” is set to “S2C541”, writing operation during play back mode becomes valid. However, please operate the manipulator with caution since this operation may influence the cycle time.

CAUTION

The “edit lock status” to which S2C542 can specify are following status.

– In alarm status
– External memory is being used
– Data transmitting function is being used
– Specified input EDIT_LOCK(#40064) is turned ON.

5 MOTION CONTROL API

YppIMOV

Moves the current position of the manipulator with the incremental value of linear motion.

Syntax

LONG YppIMOV(YPP_IMOV_SEND_DATA* sData,
YPP_STD_RSP_DATA* rData);

Parameters

[in] *sData*

Pointer to the data structure that transmits the information of incremental move

- YPP_IMOV_SEND_DATA

Data structure that transmits the information of incremental move

Syntax:

```
#define MAX_NO_OF_AXES  (12)

typedef struct
{
    CTRLG_T    CtrlGrp;
    LONG       lSpeed;
    SHORT      sVType;
    SHORT      sFrame;
    SHORT      sToolNo;
    CHAR       reserved[2];
    LONG       lPos[MAX_NO_OF_AXES];
} YPP_IMOV_SEND_DATA;
```

Member: <*CtrlGrp*> Control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)

Value	Description
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<*lSpeed*> Motion speed (Unit : micron/s, 0.0001deg/s)

<*sVType*> Selection of motion speed

Value	Description
0	Control point
1	Position angular

<*sFrame*> Coordinate system ID

Value	Description
0	Base coordinate
1	Robot coordinate
2, 3, …	User coordinate1, 2, …

<*sToolNo*> Tool number (0 to 63)

<*lPos[MAX_CART_AXES]*> Specified position (up to 12 arrays)

Robot axis

Value	Description
lPos[0]	X-axis coordinate (unit: micron)
lPos[1]	Y-axis coordinate (unit: micron)
lPos[2]	Z-axis coordinate (unit: micron)
lPos[3]	Wrist angle Rx (unit: 0.0001deg)
lPos[4]	Wrist angle Ry (unit: 0.0001deg)
lPos[5]	Wrist angle Rz (unit: 0.0001deg)
lPos[6]	angle Re (unit: 0.0001deg)
lPos[7]	(Reserved)
lPos[8]	(Not in use)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

External axis

Value	Description
lPos[0]	(Not in use)
lPos[1]	(Not in use)
lPos[2]	(Not in use)
lPos[3]	(Not in use)
lPos[4]	(Not in use)
lPos[5]	(Not in use)
lPos[6]	1st external axis pulse value (micron in the case of traveling axis)
lPos[7]	2nd external axis pulse value (micron in the case of traveling axis)
lPos[8]	3rd external axis pulse value (micron in the case of traveling axis)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

Set "0" for data lPos[8] to lPos[11] if the system has no external axis.

Set “0” for data lPos[0] to lPos[5] and lPos[9] to lPos[11] if the system has any external axes.

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Function succeeded.

Nonzero : Function failed.

YppMOVJ

Moves the manipulator to the specified position with the joint motion.

Syntax

LONG YppMOVJ(YPP_MOVJ_SEND_DATA* sData,
YPP_STD_RSP_DATA* rData);

Parameters

[in] *sData*

Pointer to the data structure that transmits the MOVJ information

- YPP_MOVJ_SEND_DATA

Data structure that transmits the MOVJ information

Syntax:

```
#define MAX_NO_OF_AXES  (12)

typedef struct
{
    CTRLG_T    CtrlGrp;
    LONG       lSpeed;
    SHORT      sFrame;
    SHORT      sConfig;
    SHORT      sToolNo;
    CHAR       reserved[2];
    LONG       lPos[MAX_NO_OF_AXES];
} YPP_MOVJ_SEND_DATA;
```

Member: <*CtrlGrp*> Control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)

Value	Description
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<*lSpeed*> Motion speed (Input range : 1 to 10000, Unit: 0.01%)

<*sFrame*> Coordinate system ID

Value	Description
0	Base coordinate
1	Robot coordinate
2, 3, …64	User coordinate1, 2, …63

<*sConfig*> Configuration

Value	Description
D00	0:Front 1:Back
D01	0:Upper arm 1:Lower arm
D02	0:Flip 1:No flip
D03	0:R<180deg 1:R>=180deg
D04	0:T<180deg 1:T>=180deg
D05	0:S<180deg 1:S>=180deg
D06-D15	Reserved

<*sToolNo*> Tool number (0 to 63)

<*lPos[MAX_CART_AXES]*> Specified position (up to 12 arrays)

Robot axis

Value	Description
lPos[0]	X-axis coordinate (unit: micron)
lPos[1]	Y-axis coordinate (unit: micron)
lPos[2]	Z-axis coordinate (unit: micron)
lPos[3]	Wrist angle Rx (unit: 0.0001deg)
lPos[4]	Wrist angle Ry (unit: 0.0001deg)
lPos[5]	Wrist angle Rz (unit: 0.0001deg)
lPos[6]	angle Re (unit: 0.0001deg)
lPos[7]	(Reserved)
lPos[8]	(Not in use)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

External axis

Value	Description
lPos[0]	(Not in use)
lPos[1]	(Not in use)
lPos[2]	(Not in use)
lPos[3]	(Not in use)
lPos[4]	(Not in use)
lPos[5]	(Not in use)
lPos[6]	1st external axis pulse value (micron in the case of traveling axis)
lPos[7]	2nd external axis pulse value (micron in the case of traveling axis)
lPos[8]	3rd external axis pulse value (micron in the case of traveling axis)
lPos[9]	(Not in use)
lPos[10]	(Not in use)

Value	Description
lPos[11]	(Not in use)

Set "0" for data lPos[8] to lPos[11] if the system has no external axis.

Set “0” for data lPos[0] to lPos[5] and lPos[9] to lPos[11] if the system has any external axes.

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Function succeeded.

Nonzero : Function failed.

YppMOVL

Moves the manipulator to the specified position with the linear motion.

Syntax

LONG YppMOVL(YPP_MOVL_SEND_DATA* sData,
YPP_STD_RSP_DATA* rData);

Parameters

[in] *sData*

Pointer to the data structure that transmits the MOVL information

- YPP_MOVL_SEND_DATA

Data structure that transmits the MOVL information

Syntax:

```
#define MAX_NO_OF_AXES  (12)

typedef struct
{
    CTRLG_T    CtrlGrp;
    LONG       lSpeed;
    SHORT      sVType;
    SHORT      sFrame;
    SHORT      sConfig;
    SHORT      sToolNo;
    LONG       lPos[MAX_NO_OF_AXES];
} YPP_MOVL_SEND_DATA;
```

Member: <*CtrlGrp*> Control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)

Value	Description
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<*lSpeed*> Motion speed (Unit : micron/s, 0.0001deg/s)

<*sVType*> Selection of motion speed

Value	Description
0	Control point
1	Position angular

<*sFrame*> Coordinate system ID

Value	Description
0	Base coordinate
1	Robot coordinate
2, 3, …	User coordinate1, 2, …

<*sConfig*> Configuration

Value	Description
D00	0:Front 1:Back
D01	0:Upper arm 1:Lower arm
D02	0:Flip 1:No flip
D03	0:R<180deg 1:R>=180deg
D04	0:T<180deg 1:T>=180deg
D05	0:S<180deg 1:S>=180deg
D06-D15	Reserved

<*sToolNo*> Tool number (0 to 63)

<*lPos[MAX_CART_AXES]*> Specified position (up to 12 arrays)

Robot axis

Value	Description
lPos[0]	X-axis coordinate (unit: micron)
lPos[1]	Y-axis coordinate (unit: micron)
lPos[2]	Z-axis coordinate (unit: micron)
lPos[3]	Wrist angle Rx (unit: 0.0001deg)
lPos[4]	Wrist angle Ry (unit: 0.0001deg)
lPos[5]	Wrist angle Rz (unit: 0.0001deg)
lPos[6]	angle Re (unit: 0.0001deg)
lPos[7]	(Reserved)
lPos[8]	(Not in use)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

External axis

Value	Description
lPos[0]	(Not in use)
lPos[1]	(Not in use)
lPos[2]	(Not in use)
lPos[3]	(Not in use)
lPos[4]	(Not in use)
lPos[5]	(Not in use)
lPos[6]	1st external axis pulse value (micron in the case of traveling axis)
lPos[7]	2nd external axis pulse value (micron in the case of traveling axis)
lPos[8]	3rd external axis pulse value (micron in the case of traveling axis)
lPos[9]	(Not in use)
lPos[10]	(Not in use)

Value	Description
lPos[11]	(Not in use)

Set "0" for data lPos[8] to lPos[11] if the system has no external axis.

Set “0” for data lPos[0] to lPos[5] and lPos[9] to lPos[11] if the system has any external axes.

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Function succeeded.

Nonzero : Function failed.

YppPulseMOVJ

Moves the manipulator to the specified pulse position with the joint motion.

Syntax

LONG YppPulseMOVJ(YPP_PMOVJ_SEND_DATA* sData, YPP_STD_RSP_DATA* rData);

Parameters

[in] *sData*

Pointer to the data structure that transmits the PMOVJ information

- YPP_PMOVJ_SEND_DATA

Data structure that transmits the PMOVJ information

Syntax:

```
#define MAX_NO_OF_AXES  (12)

typedef struct
{
    CTRLG_T   CtrlGrp;
    LONG      lSpeed;
    SHORT     sToolNo;
    CHAR      reserved0[2];
    LONG      lPos[MAX_NO_OF_AXES];
    CHAR      reserved1[4];
} YPP_PMOVJ_SEND_DATA;
```

Member: <*CtrlGrp*> Control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)
11	B4 (Base 4)

Value	Description
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<*lSpeed*> Motion speed (Input range : 1 to 10000, Unit: 0.01%)

<*sToolNo*> Tool number (0 to 63)

<*lPos[MAX_CART_AXES]*> Specified position (up to 12 arrays)

Robot axis

Value	Description
lPos[0]	1st axis (S) pulse value
lPos[1]	2nd axis (L) pulse value
lPos[2]	3rd axis (U) pulse value
lPos[3]	4th axis (R) pulse value
lPos[4]	5th axis (B) pulse value
lPos[5]	6th axis (T) pulse value
lPos[6]	7th axis (E) pulse value

Value	Description
lPos[7]	8th axis pulse value
lPos[8]	(Not in use)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

External axis

Value	Description
lPos[0]	(Not in use)
lPos[1]	(Not in use)
lPos[2]	(Not in use)
lPos[3]	(Not in use)
lPos[4]	(Not in use)
lPos[5]	(Not in use)
lPos[6]	(Not in use)
lPos[7]	(Not in use)
lPos[8]	1st extermal axis pulse value
lPos[9]	2nd extermal axis pulse value
lPos[10]	3rd extermal axis pulse value
lPos[11]	4th extermal axis pulse value

Set "0" for data lPos[8] to lPos[11] if the system has no external axis.

Set “0” for data lPos[0] to lPos[5] if the system has any external axes.

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:

```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Function succeeded.

Nonzero : Function failed.

YppPulseMOVL

Moves the manipulator to the specified pulse position with the linear motion.

Syntax

LONG YppPulseMOVL(YPP_PMOVL_SEND_DATA* sData, YP_STD_RSP_DATA* rData);

Parameters

[in] *sData*

Pointer to the data structure that transmits the PMOVL information

- YPP_PMOVL_SEND_DATA

Data structure that transmits the PMOVL information

Syntax:

```
#define MAX_NO_OF_AXES  (12)

typedef struct
{
    CTRLG_T    CtrlGrp;
    LONG       lSpeed;
    SHORT      sVType;
    SHORT      sToolNo;
    LONG       lPos[MAX_NO_OF_AXES];
    CHAR       reserved[4];
} YPP_PMOVL_SEND_DATA;
```

Member: <*CtrlGrp*> Control group

Value	Description
0	R1 (Robot 1)
1	R2 (Robot 2)
2	R3 (Robot 3)
3	R4 (Robot 4)
4	R5 (Robot 5)
5	R6 (Robot 6)
6	R7 (Robot 7)
7	R8 (Robot 8)
8	B1 (Base 1)
9	B2 (Base 2)
10	B3 (Base 3)

Value	Description
11	B4 (Base 4)
12	B5 (Base 5)
13	B6 (Base 6)
14	B7 (Base 7)
15	B8 (Base 8)
16	S1 (Station 1)
17	S2 (Station 2)
18	S3 (Station 3)
19	S4 (Station 4)
20	S5 (Station 5)
21	S6 (Station 6)
22	S7 (Station 7)
23	S8 (Station 8)
24	S9 (Station 9)
25	S10 (Station 10)
26	S11 (Station 11)
27	S12 (Station 12)
28	S13 (Station 13)
29	S14 (Station 14)
30	S15 (Station 15)
31	S16 (Station 16)
32	S17 (Station 17)
33	S18 (Station 18)
34	S19 (Station 19)
35	S20 (Station 20)
36	S21 (Station 21)
37	S22 (Station 22)
38	S23 (Station 23)
39	S24 (Station 24)

<*lSpeed*> Motion speed (Unit : micron/s, 0.0001deg/s)

<*sVType*> Selection of motion speed

Value	Description
0	Control point
1	Position angular

<*sToolNo*> Tool number (0 to 63)

<*lPos[MAX_CART_AXES]*> Specified position (up to 12 arrays)

Robot axis

Value	Description
lPos[0]	1st axis (S) pulse value
lPos[1]	2nd axis (L) pulse value
lPos[2]	3rd axis (U) pulse value
lPos[3]	4th axis (R) pulse value
lPos[4]	5th axis (B) pulse value
lPos[5]	6th axis (T) pulse value
lPos[6]	7th axis (E) pulse value
lPos[7]	8th axis pulse value
lPos[8]	(Not in use)
lPos[9]	(Not in use)
lPos[10]	(Not in use)
lPos[11]	(Not in use)

External axis

Value	Description
lPos[0]	(Not in use)
lPos[1]	(Not in use)
lPos[2]	(Not in use)
lPos[3]	(Not in use)
lPos[4]	(Not in use)
lPos[5]	(Not in use)
lPos[6]	(Not in use)
lPos[7]	(Not in use)
lPos[8]	1st extermal axis pulse value
lPos[9]	2nd extermal axis pulse value
lPos[10]	3rd extermal axis pulse value
lPos[11]	4th extermal axis pulse value

Set "0" for data lPos[8] to lPos[11] if the system has no external axis.

Set “0” for data lPos[0] to lPos[5] if the system has any external axes.

[out] *rData*

Pointer to the data structure that receives the error information

- YPP_STD_RSP_DATA

Data structure that receives the error information

Syntax:
```
typedef struct
{
    USHORT    err_no;
    CHAR      reserved[2];
```

```
} YPP_STD_RSP_DATA;
```

Member: <*err_no*> Error number

Value	Description
0x0000	Succeeded
0x2010	Robot is in operation
0x2030	In HOLD status (Pendant)
0x2040	In HOLD status (External)
0x2050	In HOLD status (Command)
0x2060	In alarm/error status
0x2070	In SERVO OFF status
0x2080	In TEACH mode status
0x3040	The origin position is not registered
0x3050	Out of range (ABSO data)
0x3400	Cannot operate MASTER JOB
0x3410	In TEACH LOCK status; or the JOB name is already registered
0x4040	Specified JOB not found

■ **Return value**

0 : Function succeeded.

Nonzero : Function failed.

6 FILE TRANSFER API

YppLoadFile

Loads the specified file.

■ **Syntax**

LONG YppLoadFile
(LONG *lMedia,* LPCTSTR *loadPath,* LPCTSTR *fileName*);

■ **Parameter**

[in] *Media*

Media in which loading files exist.

Value	Description
1	Compact Flash (PP)
2	USB memory (PP)

[in] *loadPath*

Pointer of the folder name in which loading files exist.

- When the file exists in the route folder of the specified media, specify an empty character string (“”).
- When the number of hierarchy of the holder is two or more, use “\\” to partition them.
  (Example: aaa\\bbb)

[in] *fileName*

Pointer to the name of the loading file.

■ **Return value**

0 : Normal end

Other than 0 : Error

CAUTION

In consideration of the data migration from the controller, by specifying 0 (IMedia=0) to the first parameter, specification with the full path name, which is same as the full path name of controller, to savePath is enabled.

However, on the other hand, it is recommended to specify “1” or ”2” (IMedia=1 or 2) and use the Compact Flash (PP) or the USB memory for loading/saving the file.

YppSaveFile

Saves the specified file.

■ Syntax

LONG YppSaveFile
(LONG *lMedia*, LPCTSTR *savePath*, LPCTSTR *fileName*);

■ Parameter

[in] *lMedia*

Media for saving files

Value	Description
1	Compact Flash (PP)
2	USB memory (PP)

[in] *savePath*

Folder name in which the file is saved.

- When saving the file in the route folder of the specified media, specify an empty character string (“”).
- When the hierarchy of the holder is two or more, use “\\” to partition them.
  (Example: aaa\\bbb)

[in] *fileName*

File name to be saved.

■ Return value

0 : Normal end

Other than 0 : Error

CAUTION

In consideration of the data migration from the controller, by specifying 0 (IMedia=0) to the first parameter, specification with the full path name, which is same as the full path name of the controller, to savePath is enabled.

However, on the other hand, it is recommended to specify “1” or ”2” (IMedia=1 or 2) and use the Compact Flash (PP) or the USB memory for loading/saving the file.

YppRefreshFileList

Saves the specified file.

Syntax

LONG YppRefreshFileList(SHORT *extension*);

Parameter

[in] *extension*

File type

Value	Description
1	Independent job file (JBI)
2	Related job file (JBR)

Return value

0 : Normal end

Other than 0 : Error

YppGetFileCount

Acquires the file number in the file list.

■ **Syntax**
LONG YppGetFileCount();

■ **Parameter**
No parameter

■ **Return value**
0 : Normal end
Other than 0 : Error

YppGetFileName

Acquires the file name in the file list.

■ **Syntax**

LONG YppGetFileName(INT *index*, CString* *jobname*);

■ **Paramete**

[in] *index*

Index of the acquiring file.

[out] *SavePath*

File name of the specified index (result of acquisition)

■ **Return value**

0 : Normal end

Other than 0 : Error

DX100/DX200/FS100 OPTIONS
INSTRUCTIONS

REFERENCE MANUAL
FOR PROGRAMMING PENDANT CUSTOMIZATION FUNCTION

HEAD OFFICE
2-1 Kurosakishiroishi, Yahatanishi-ku, Kitakyushu 806-0004, Japan
Phone +81-93-645-7745 Fax +81-93-645-7746

YASKAWA America Inc. M Robotics Division
100 Automation Way, Miamisburg, OH 45342, U.S.A.
Phone +1-937-847-6200 Fax +1-937-847-6277

YASKAWA Nordic AB
Box 504 Verkstadsgatan 2, PO Box 504 SE-385 25 Torsås, Sweden
Phone +46-480-417-800 Fax +46-486-414-10

YASKAWA Europe GmbH Robotics Div.
Yaskawastrasse 1, 85391 Allershausen, Germany
Phone +49-8166-90-0 Fax +49-8166-90-103

YASKAWA Electric Korea Co., Ltd
9F, KYOBO Securities Bldg., 26-4, Yeoido-Dong Yeoungeungpo-ku, Seoul, Korea
Phone +82-2-784-7844 Fax +82-2-784-8495

YASKAWA Electric (Singapore) PTE Ltd.
151 Lorong Chuan, #04-02A, New Tech Park, Singapore 556741
Phone +65-6282-3003 Fax +65-6289-3003

YASKAWA Electric (Thailand) Co., Ltd.
252/246, 4th Floor. Muang Thai-Phatra Office Tower II Rachadaphisek Road,
Huaykwang Bangkok, 10320, Thailand
Phone +66-2-693-2200 Fax +66-2-693-4200

YASKAWA Shougang Robot Co. Ltd.
1015. Boxuenan Rd. Maluzhen, Jiading District, Shanghai, China
Phone +86-21-5950-3521 Fax +86-20-3878-0651

YASKAWA ELECTRIC CHINA Co., Ltd.
12F Carlton Building, No. 21-42 Huanghe Road, Shanghai 200003, China
Phone +86-21-5385-2200 Fax +86-21-5385-3299

YASKAWA Robotics India Ltd.
#426, Udyog Vihar, Phase- IV, Gurgaon, Haryana, India
Phone +91-124-475-8500 Fax +91-124-475-8542

YASKAWA
MOTOMAN ROBOTICS

MANUAL NO.
HW1481483 ①