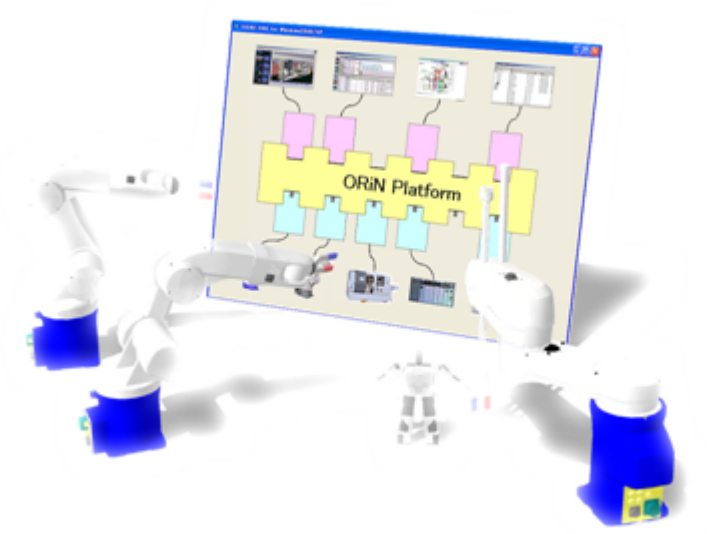
How to use the Slave mode/b-CAP

DENSO Wave inc. 2009



Note

- This document is written with the assumption that the readers have a basic knowledge and experience in b-CAP.
  - Please read “b-CAP Users manual” before reading this document.

- In this version, For using the Slave mode, a teaching pendant is required for settings.

Contents

What's the Slave mode ?

- The Slave mode is a new function to control robot directly by sending a destination position In short interval time.

- Three command are implemented on b-CAP.
  - **slvChangeMode**
    - Changes your RC7 to the Slave mode.
  - **slvGetMode**
    - Get current mode.
  - **slvMove**
    - Move to a destination position.

- In the version 3.0 of RC7, TCP and UDP connection for b-CAP is supported.
  - To use UDP, User may have to implement a retry sequence.

Functions of the Slave mode (1)

- slvChangeMode
  - This function choose the position data type and response style of slvMove.
  - If “Sync” is chosen, then timeout detector (in each 8msec) is enabled.
  - If “Async” is chosen, then you can switch the timeout detector by the teaching pendant.
  - If zero is chosen, then the slave mode is stopped.

Parameter values of slvChangeMode

Parameter Value of slvChangeMode	Position data type (see also slvMove)	Sync/Async
0x000	–	(Stop the slave mode)
0x001	P	Sync
0x002	J	Sync
0x003	T	Sync
0x101	P	Async
0x102	J	Async
0x103	T	Async

  - When changing to the Slave mode with this command, the change mode operation waits until a robot stops completely.
  - However, if this waiting time exceeds 500 msec, an error 600B [Robot is running] will occur.
  - To avoid this error and the waiting time, it is recommended to choose an @E option when using robot motion commands (e.g. Move/Approach/Drive) before changing to slave-mode.

Functions of the Slave mode (2)

- **slvGetMode**
  - This function returns current setting of the Slave mode.
  - Returns value is,
    - 0x000
    - 0x001-0x003
    - 0x101-0x103

Functions of the Slave mode (3)

- **slvMove**
  - This function send the trajectory data, and the data type is specified by the function “slvChangeMode”.
  - (For more detail information ,
    Please see the other document, ‘b-CAP SlaveMode’)
  - Please see also the section of slvChangeMode

Recommendation of the Slave mode

Trajectory data is sent in each 8msec

TCP/UDP	Sync/Async	Retry and Timeout Of slvMove command	Retry and Timeout The other commands
UDP	Sync	Timeout = 2msec Retry count = 7	Timeout = 1000msec Retry count = 4
I.e. the parameter value is 0x001 - 0x003			

Trajectory data is sent in each 1msec

TCP/UDP	Sync/Async	Retry and Timeout Of slvMove command	Retry and Timeout The other commands
UDP	Async	Timeout = 2msec Retry count = 0 (DO NOT SEND Retry packet. Because the bandwidth of the network is wasted by retry packets. Sending in hi-frequency works enough instead of retrying)	Timeout = 1000msec Retry count = 3
I.e. the parameter value is 0x101 - 0x103			

Support functions for the Slave mode

- Convenient functions of b-CAP for supporting the Slave mode
  - StartLog
  - StopLog

- These functions are for getting motion logs of the robot.

Procedures of setting up RC7 and starting your application.

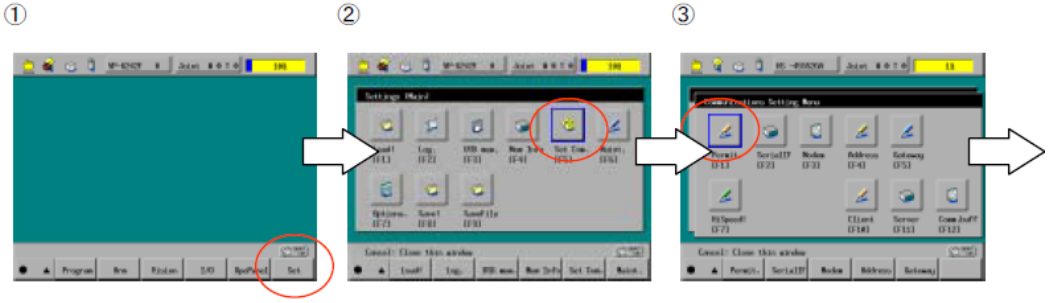
- Setting up your RC7
  - These setting up operations are needed only once.
- Preparation
  - Turn ON the Machine-Lock by the teaching pendant.
  - Change into External Auto mode by the teaching pendant.
- Starting your client application
  - Run “RobSlave.pac” by b-CAP.
    - “RobSlave.pac” must be running to use the Slave mode.
  - Move your robot to the first position to start the Slave mode by b-CAP.
  - Change into the Slave mode by a function **slvChangeMode** of b-CAP.
  - Send trajectories by a function **slvMove** of b-CAP.
  - Stop the Slave mode by a function **slvChangeMode** of b-CAP.
- Clearing error
  - When some function returns a error, must clear the error.

In the following pages, detail information of each procedures are described.

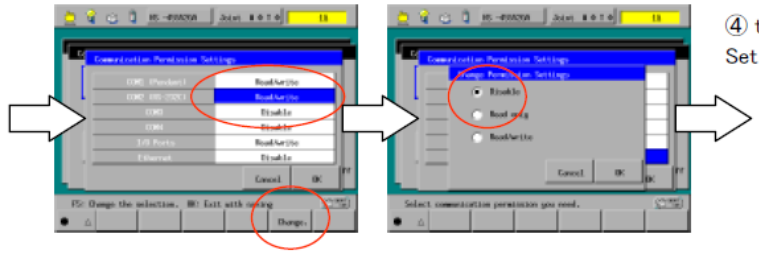
Procedures of setting - Setting up your RC7 (1)

Operations of in this section is the same as section 2 “Setup” of in the b-CAP users manual
○ except “4) Set the “b-CAP Slave mode” in this section.

- 1) Set the “communication settings” with a teaching pendant.
  - Enable the “Ethernet”



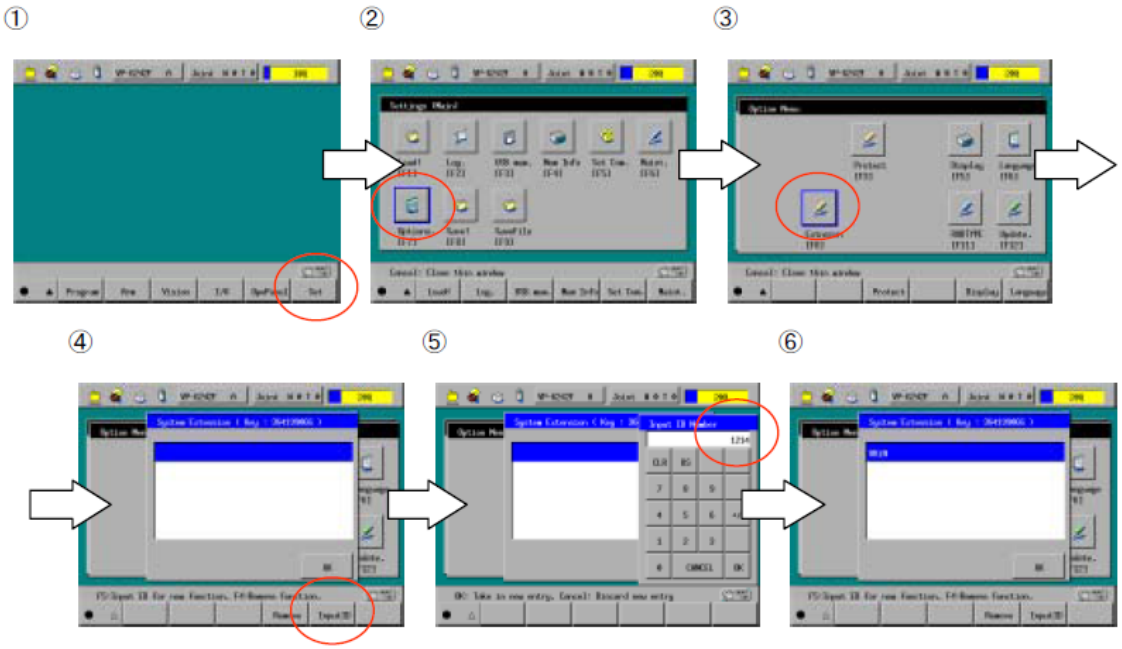
④ to ⑤
Set RS-232C port to disable



⑥ to ⑦
Set Ethernet to enable

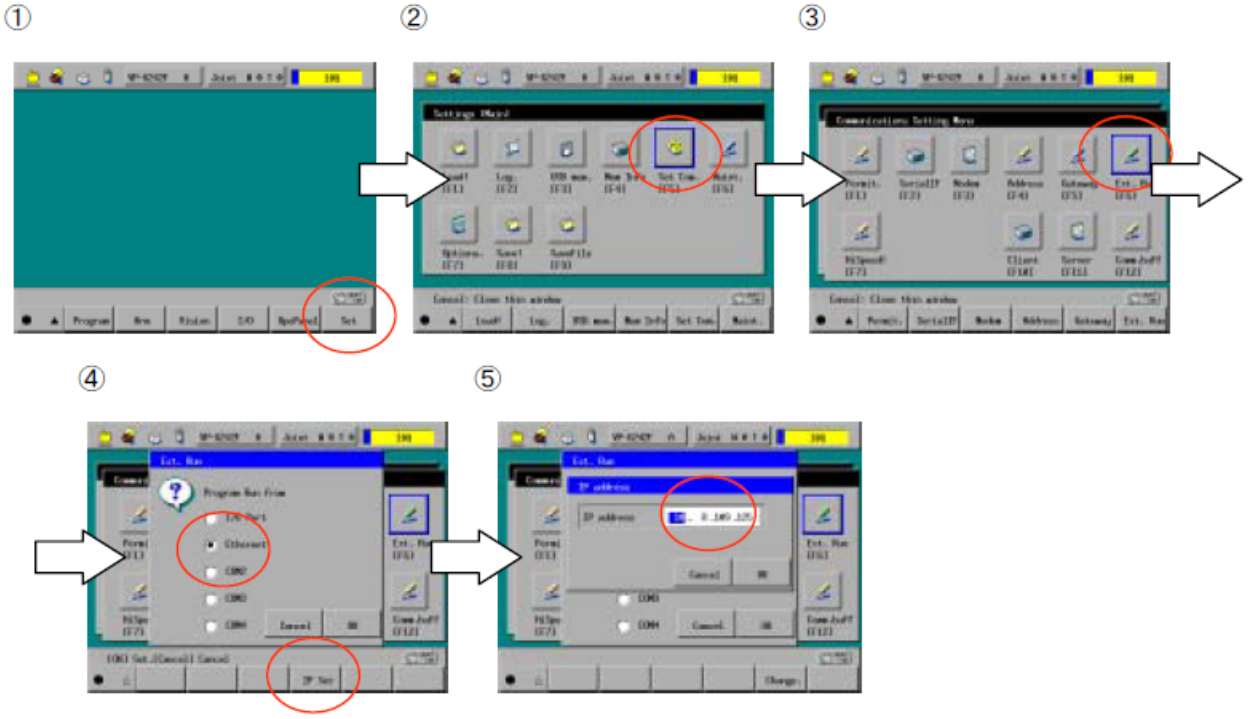
Procedures of setting - Setting up your RC7 (2)

- 2) Set the “Option” with the teaching pendant.
  - Enable the “ORiN” by the number “1214”.



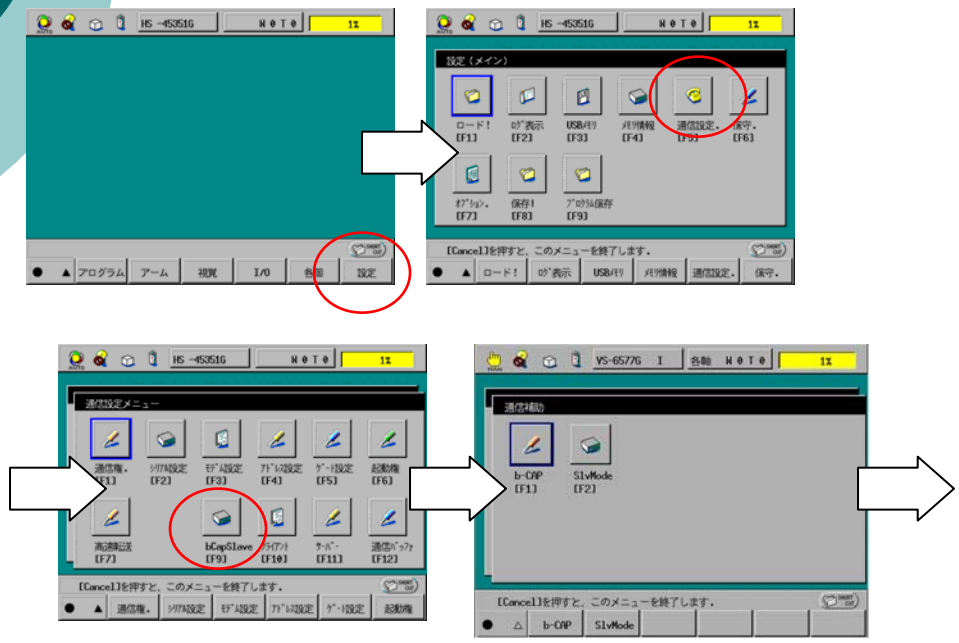
Procedures of setting - Setting up your RC7 (3)

- 3) Set the “Executable Token” with the teaching pendant.
  - Enable “Ethernet”
  - Set your client IP address



Procedures of setting - Setting up your RC7 (4)

- 4) Set the “b-CAP Slave mode” with the teaching pendant.
  - Set the Slave mode option.
  - Please see the section of “slvChangeMode” for more detail information.



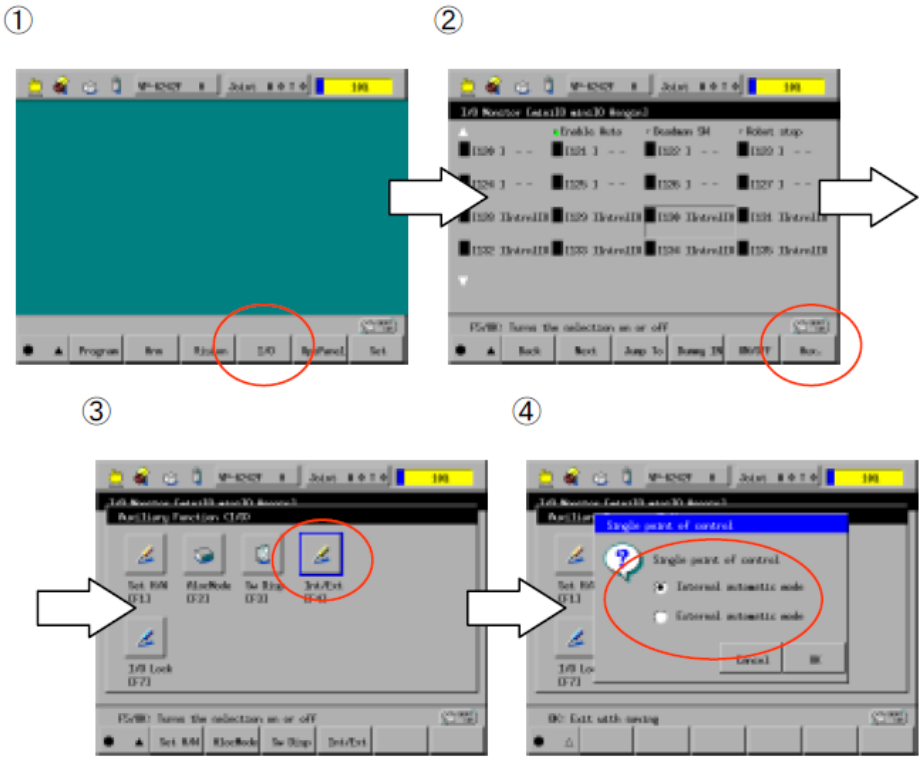
TIPS,

- While testing your application, IF you quit your application without closing the TCP or UDP connection, it may cause to disturb re-connection to the RC7.
- To solve this, Restart the RC7 or Press “Reset” button in the Teaching pendant.

Procedures of setting - Setting up your RC7 (5)

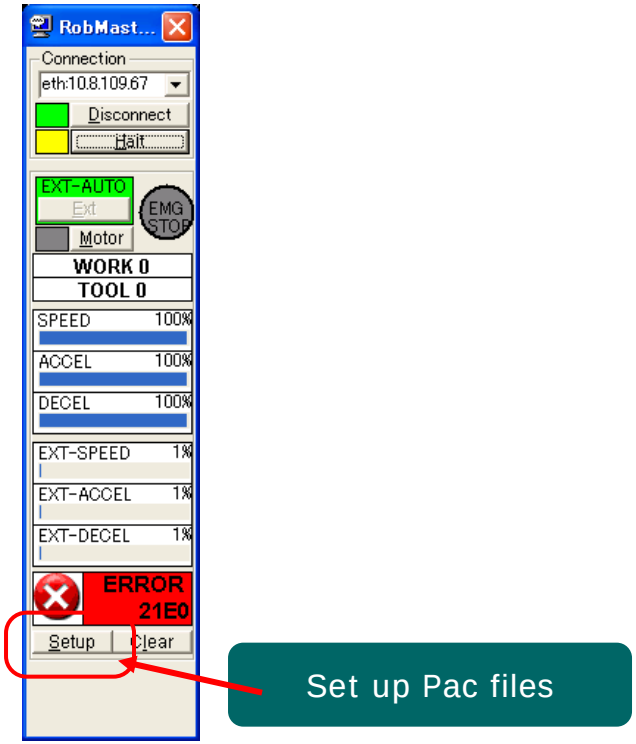
- 5) Set up I/O options with the teaching pendant.
  - Set the ‘External Automatic Mode’
    - To use this Slave mode, The controller must be set to the external automatic mode.

TIP
This setting need not be done in the global type controller.



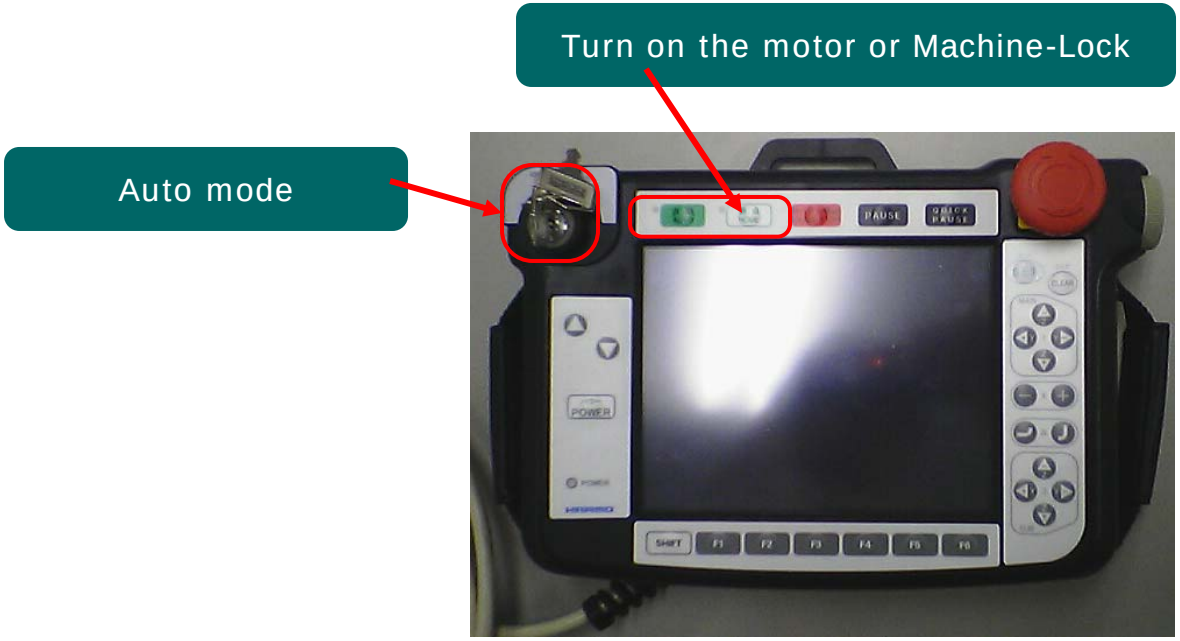
Procedures of setting - Setting up your RC7 (6)

- 6) Load PAC files for supporting the Slave mode.
  - Load RobSlave.Pac, UserExtension.pac and RobSlave.h



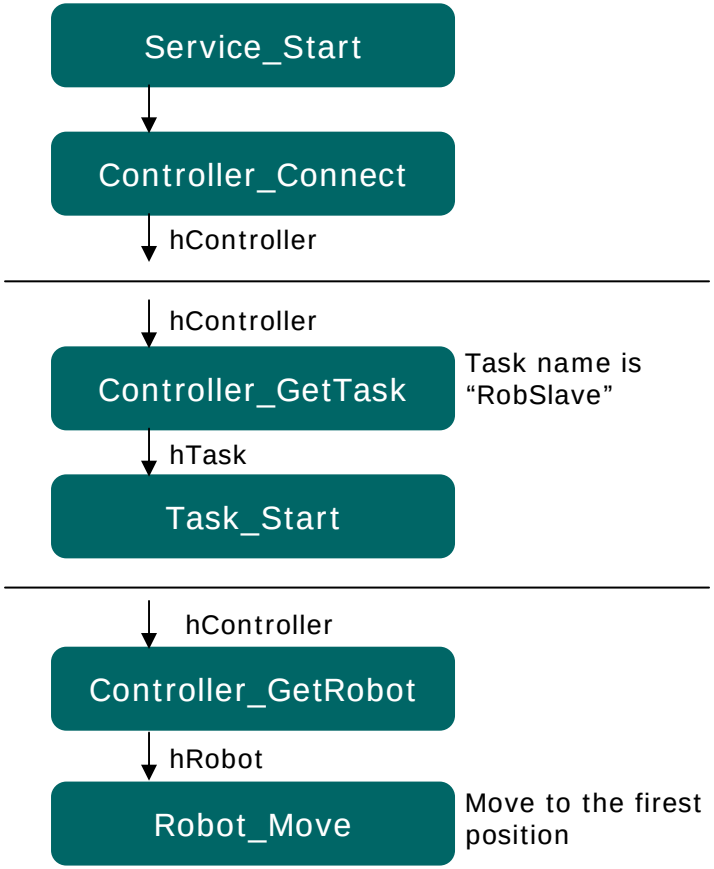
Procedures of setting - Preparation

- 1) Turn ON the Motor or Machine-Lock by the teaching pendant.
- 2) Change into the Auto mode by the teaching pendant.



Starting your client app. (1)

- 1) Start b-CAP
  - Start b-CAP and connect to the controller.
- 2) Run “RobSlave.pac” by b-CAP
  - “RobSlave.pac” must be running to use the Slave mode.
- 3) Move your robot to the first position to start the Slave mode by b-CAP.

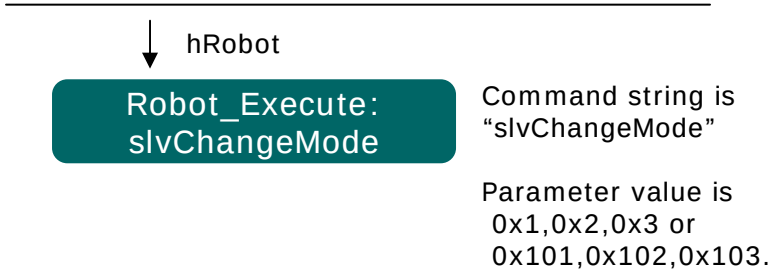


TIPS,

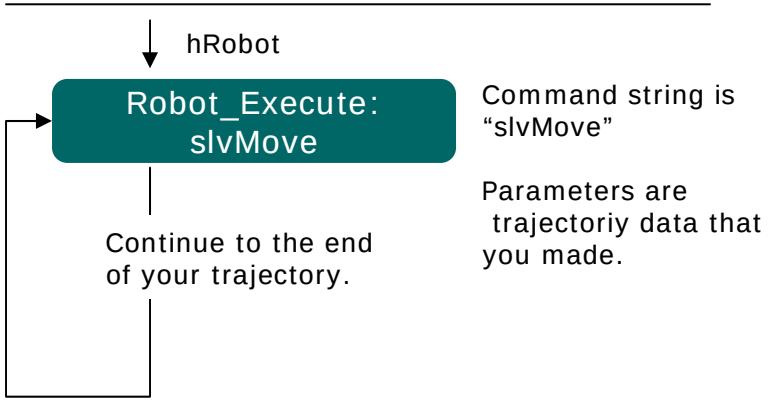
- For the safety reason, confirming your robot type is recommended before moving your robot. The system variable “@TYPE” of robot class of b-CAP returns the type number.

Starting your client app. (2)

- 4) Change to the Slave mode by slvChangeMode.

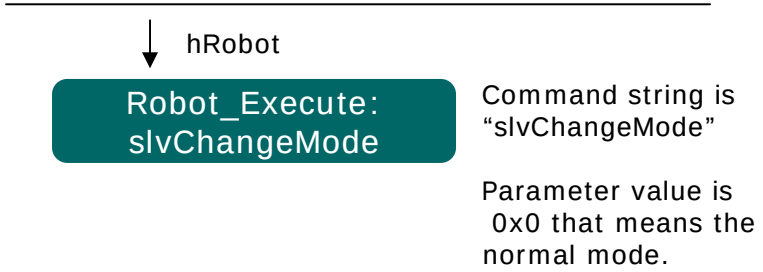


- 5) Send trajectories by slvMove.



Quitting or stopping your client application.

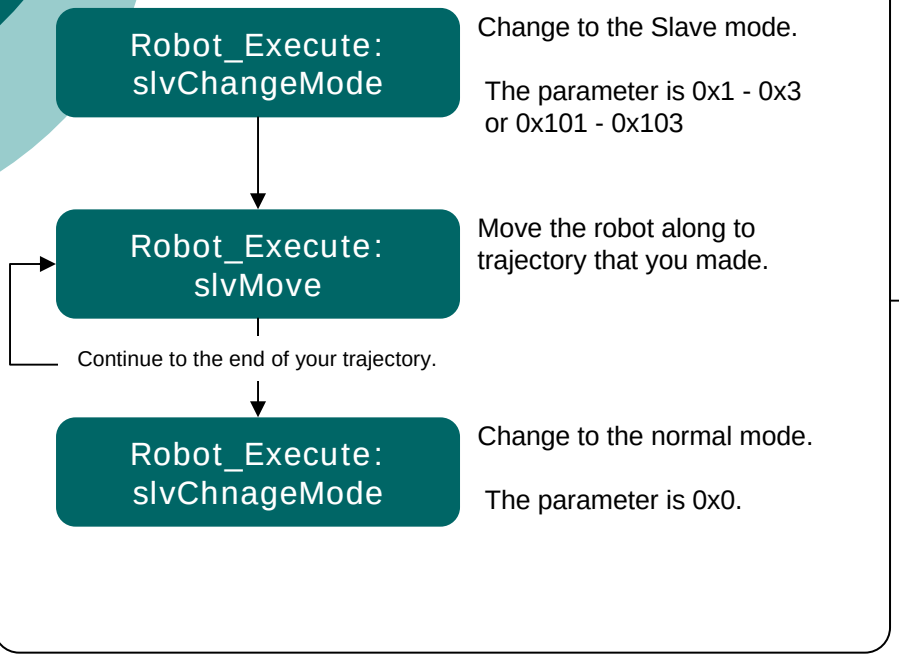
- 6) Change to the normal mode before quitting your application by slvChangeMode.



Clearing errors

- When some function returns a error,
  Clear the error in the following sequence.

Typical control sequence in the Slave mode



Error , and the sequence of clearing error

If Return code NOT EQUAL S_OK (S_OK = 0x0),
then this means error has been occurred.

By occurring the error ,
・RoboSlave.PAC is stopped.
・The Slave mode is cleared and changed to the normal mode.
・In the teaching pendant some error may be displayed

So, for clearing the error, following sequence is required.
1) Clear Error by b-CAP
2) Restart “RobSlave.PAC”

3) Move your robot to the first position by b-CAP.

4) Execute “slvChangeMode” to change
to the Slave Mode

Function references

- The commands of the Slave mode are implemented as one of Robot_Execute of b-CAP.

Function	HRESULT Robot_Execute()			
Function ID	64			
Argument	[in]	long	hRobot	Handle of robot
	[in]	BSTR	bstrCommand	Command name
	[in]	VARIANT	vntParam	Parameter
	[out]	VARIANT	pVal	Result
Return Value	Given return codes			
Description	Execute the command of the robot "hRobot"			

In this function, commands can be executed by using the following command names for "bstrCommand".

Command	Parameters	Return value	Function
slvChangeMode	VT_I2 0x0:Disable Slave mode 0x1:Slave mode(P type、**Sync**) 0x2:Slave mode (J type、**Sync**) 0x3:Slave mode (Ttype、**Sync**) 0x101:Slave mode (P type、**Async**) 0x102:Slave mode (J type、**Async**) 0x103:Slave mode (Ttype、**Async**)	None	Enable/Disable the Slave mode
slvGetMode	None	VT_I2 0x0:Disable Slave mode 0x1:Slave mode(P type、**Sync**) 0x2:Slave mode (J type、**Sync**) 0x3:Slave mode (Ttype、**Sync**) 0x101:Slave mode (P type、**Async**) 0x102:Slave mode (J type、**Async**) 0x103:Slave mode (Ttype、**Async**)	Returns current mode
slvMove	<Ptype:VT_R4|VT_ARRAY>	<Jtype:VT_R4|VT_ARRAY> Current joint angle	Move to a destination position

Function references

- slvChangeMode

Communication sample - slvChangeMode			
Enabling or disabling the Slave mode			
Packet TX	TX(Client->Server): 01 52 00 00 00 05 00 00 00 40 00 00 00 03 00 0A 00 00 00 03 00 01 00 00 00 **01 00 00 00** 24 00 00 00 08 00 01 00 00 00 1A 00 00 00 **73 00 6C 00 76** **00 43 00 68 00 61 00 6E 00 67 00 65 00 4D 00 6F** **00 64 00 65 00** 08 00 00 00 02 00 01 00 00 00 **02** **00** 04		
	Parameter	Description	Type	Value
		Binary		
	hRobot	The handle of robot (See Controller_GetRobot)	VT_I4	0x0000001
		00 00 00 03 00 01 00 00 00 **01 00 00 00** 0A		
	bstrCommand	Command string	VT_BSTR	“slvChangeMode”
		00 08 00 01 00 00 00 1A 00 00 00 **73 00 6C 00 76** 24 00 00 **00 43 00 68 00 61 00 6E 00 67 00 65 00 4D 00 6F** **00 64 00 65 000**		
	vntParam	Mode value	VT_I2	0x0002
		08 00 00 00 02 00 01 00 00 00 **02** **00**		
Packet RX	RX(Server->Client): 01 10 00 00 00 08 00 00 00 **00 00 00 00** 00 00 04			
	Parameter	Description	Type	Value
		Binary		
	None	–	–	–
		–		

Function references

- slvGetMode

Communication sample - slvGetMode

Get current the Slave mode

Packet TX

TX(Client->Server):

```
01 4E 00 00 00 05 00 00  00 40 00 00 00 03 00 0A
00 00 00 03 00 01 00 00  00 01 00 00 00 1E 00 00
00 08 00 01 00 00 00 14  00 00 00 73 00 6C 00 76
00 47 00 65 00 74 00 4D  00 6F 00 64 00 65 00 0A
00 00 00 08 00 01 00 00  00 00 00 00 00 04
```

Parameter	Description	Type	Value
	Binary		
hRobot	The handle of robot (See Controller_GetRobot)	VT_I4	0x0000001
	0A 00 00 00 03 00 01 00 00 00 01 00 00 00		
bstrCommand	Command string	VT_BSTR	"slvGetMode"
	1C 00 00 00 08 00 01 00 00 00 12 00 00 00 53 00 4C 00 41 00 56 00 45 00 4D 00 4F 00 44 00 45 00		
vntParam	Option string	VT_BSTR	Null string = ""
	0A 00 00 00 08 00 01 00 00 00 00 00 00 00		

Packet RX

RX(Server->Client):

```
01 1C 00 00 00 05 00 00  00 00 00 00 00 01 00 08
00 00 00 02 00 01 00 00  00 02 00 04
```

Parameter	Description	Type	Value
	Binary		
pVal	Current slave mode is returned	VT_I2	2
	08 00 00 00 02 00 01 00 00 00 02 00		

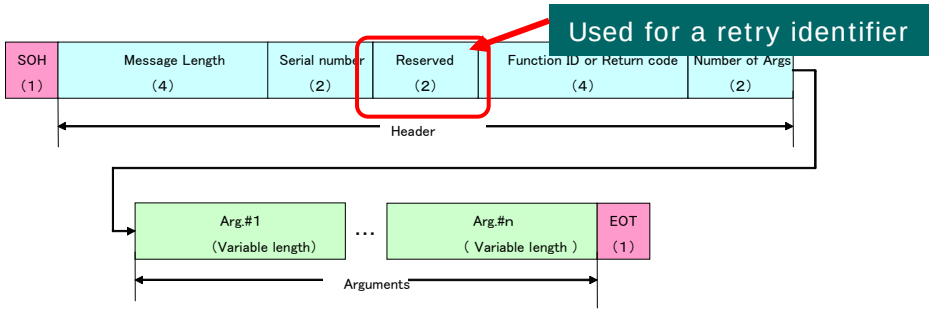
Function references

- slvMove

Communication sample - SlvMove			
Send robot move position (periodical)			
Transmitted packet	Client →server: 01 60 00 00 00 08 00 00 00 40 00 00 00 03 00 0A 00 00 00 03 00 01 00 00 00 01 00 00 00 18 00 00 00 08 00 01 00 00 00 0E 00 00 00 53 00 4C 00 56 00 4D 00 4F 00 56 00 45 00 22 00 00 00 04 20 07 00 00 00 00 00 20 41 00 00 A0 41 00 00 F0 41 00 00 20 42 00 00 48 42 00 00 70 42 00 00 80 3F 04		
	Name	Description	Type	Value
		Binary		
	hRobot	Handle of controller (Ref.:Controller_GetRobot)	VT_I4	0x0000001
		00 00 00 03 00 01 00 00 00 01 00 00 00 0A		
	bstrCommand	Command parameter	VT_BSTR	"SLVMOVE"
		00 08 00 01 00 00 00 0E 00 00 00 53 00 4C 00 56 00 4D 00 4F 00 56 00 45 00 18 00 00		
	vntParam	Command parameter	VT_R4|ARRAY	10.0 20.0 30.0 40.0 50.0 60.0 1.0
		00 00 00 00 00 20 41 00 00 A0 41 00 00 F0 41 00 00 20 42 00 00 48 42 00 00 70 42 00 00 80 3F 00 22 00 00 00 04 20 07		

Retry sequence in UDP connection (1)

- To use UDP, Retry sequence must be implemented in user apps.
- “Reserved” in the header of b-CAP packet is used for a retry identifier.



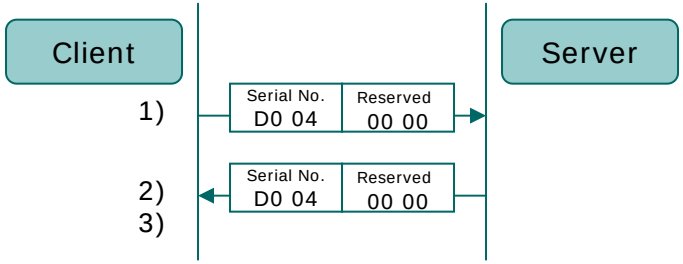
Retry sequence in UDP connection (2)

The normal communication

1) Send a new packet

2) Receive a packet

3) The sent serial number and received serial number is same,
Then a communication is completed.



The retry communication

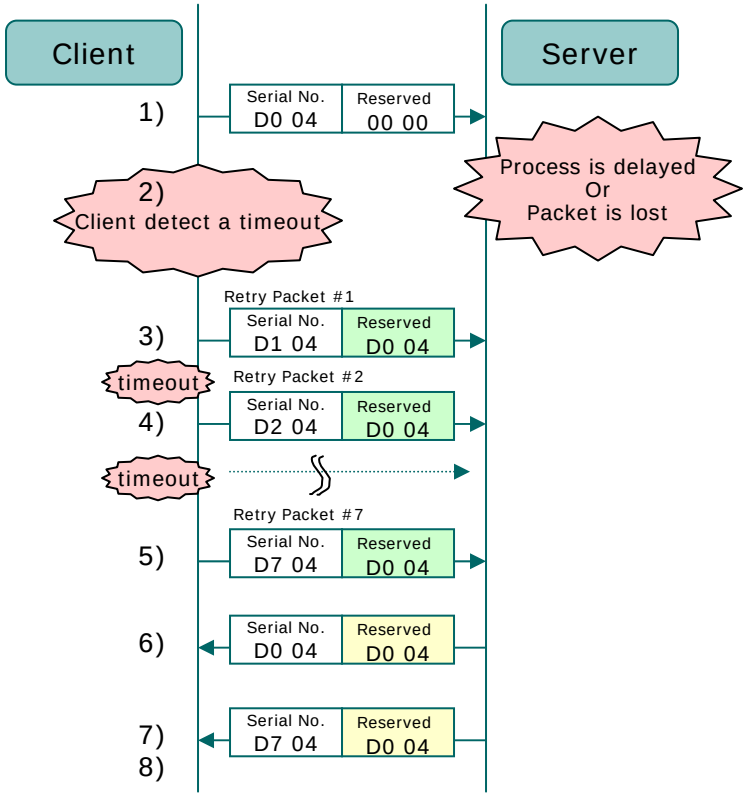
1) Send a new packet

2) But Timeout is detected

3) - 5) Create and send a retry packet
- The serial number is increased.
- In the Reserved, it is stored the serial number of original packet.
I.e. If the reserved is not 0, then it is a retry packet.

6) Packet is received.
But the last packet that client send has a different serial number,
Then this packet must be destructed.

7) Packet is received.
8) The last packet that client send has a same serial number,
Then a communication is completed.



Retry sequence in UDP connection (3)

- This is a normal packet.
  - The serial number of a packet is “D0 04”.
  - The reserved is “00 00”.

Packet TX(Client->Server)

```
01 64 00 00 00 D0 04 00 00 40 00 00 00 03 00 0A 00
00 00 03 00 01 00 00 00 01 00 00 00 18 00 00 00 08 00
01 00 00 00 0E 00 00 00 73 00 6C 00 76 00 4D 00 6F 00
76 00 65 00 26 00 00 00 04 20 08 00 00 00 B3 A3 5F C2
66 DB CB 41 86 09 33 43 60 EF 07 42 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 04
```

Packet RX(Server->Client)

```
01 3A 00 00 00 D0 04 00 00 00 00 00 01 00 26 00
00 00 04 20 08 00 00 00 67 B2 65 C2 00 28 CE 41 81 77
32 43 13 76 09 42 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 04
```

- Here is a sample of retry packets.

Packet TX(Client->Server)

```
01 64 00 00 00 D0 04 00 00 40 00 00 00 03 00 0A 00
```

Packet TX(Client->Server) Retry #1

```
01 64 00 00 00 D1 04 D0 04 40 00 00 00 03 00 0A 00
```

Packet TX(Client->Server) Retry #2

```
01 64 00 00 00 D2 04 D0 04 40 00 00 00 03 00 0A 00
```

Packet TX(Client->Server) Retry #3

```
01 64 00 00 00 D3 04 D0 04 40 00 00 00 03 00 0A 00
00 00 03 00 01 00 00 00 01 00 00 00 18 00 00 00 08 00
01 00 00 00 0E 00 00 00 73 00 6C 00 76 00 4D 00 6F 00
76 00 65 00 26 00 00 00 04 20 08 00 00 00 B3 A3 5F C2
66 DB CB 41 86 09 33 43 60 EF 07 42 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 04
```

Packet RX(Server->Client) #1

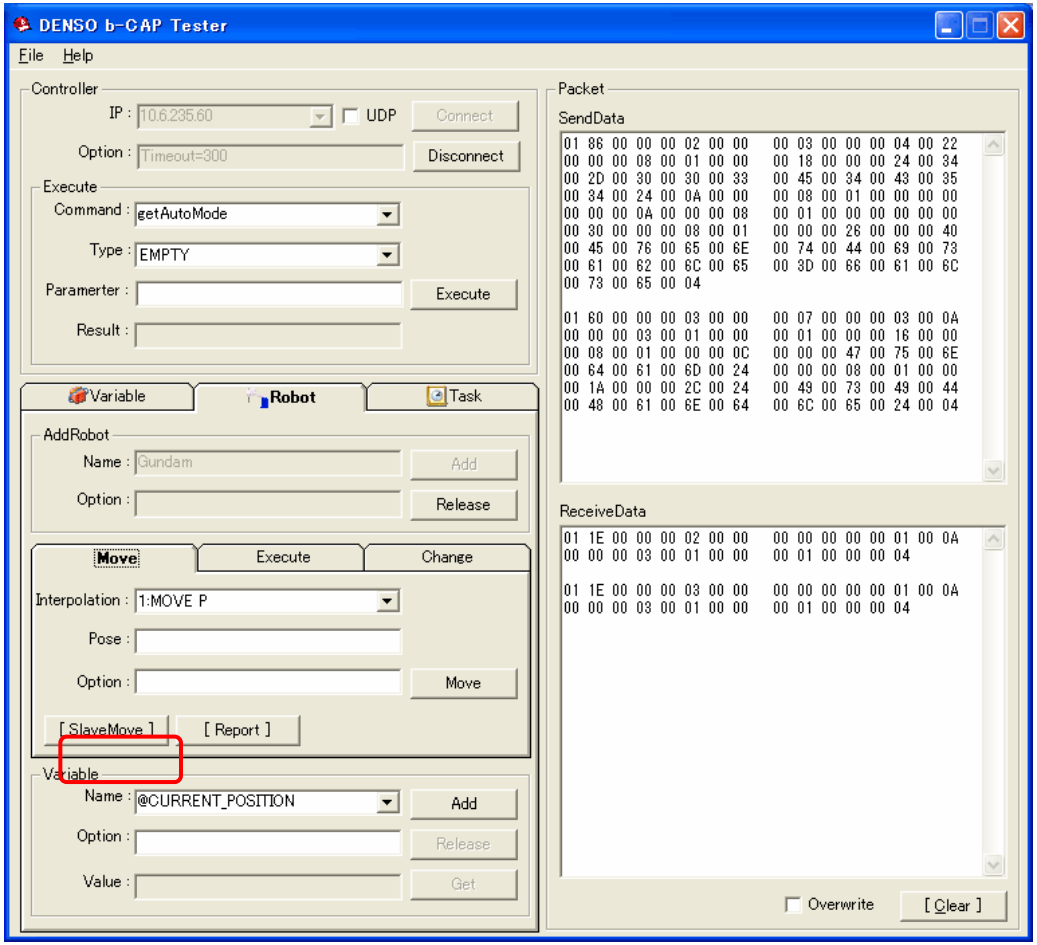
```
01 3A 00 00 00 D1 04 D0 04 00 00 00 00 01 00 26 00
```

Packet RX(Server->Client) #2

```
01 3A 00 00 00 D3 04 D0 04 00 00 00 00 01 00 26 00
00 00 04 20 08 00 00 00 67 B2 65 C2 00 28 CE 41 81 77
32 43 13 76 09 42 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 04
```

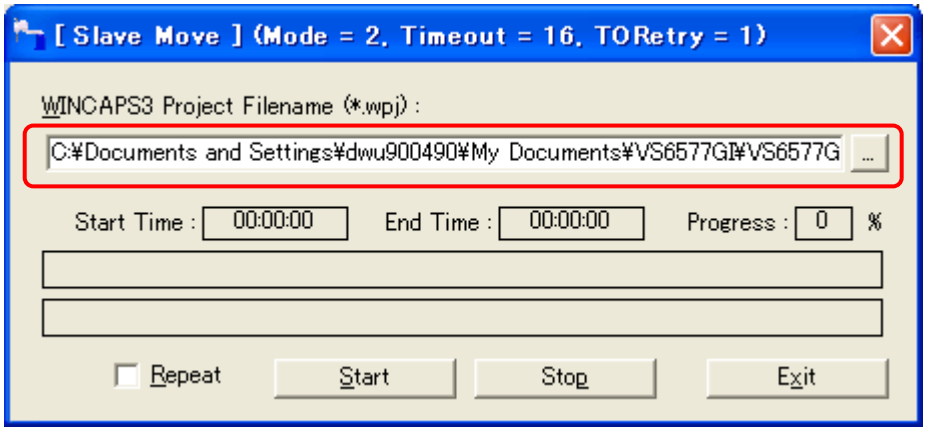

How to use b-CAP Tester(1)

- After connecting robot, push “Slave Mode” button.

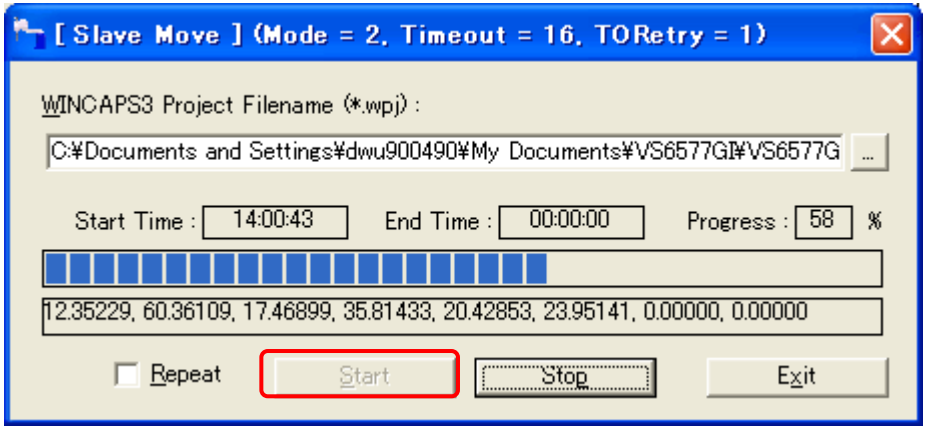


How to use b-CAP Tester(2)

- Chose WINCAPS3 project file pass which has control log.

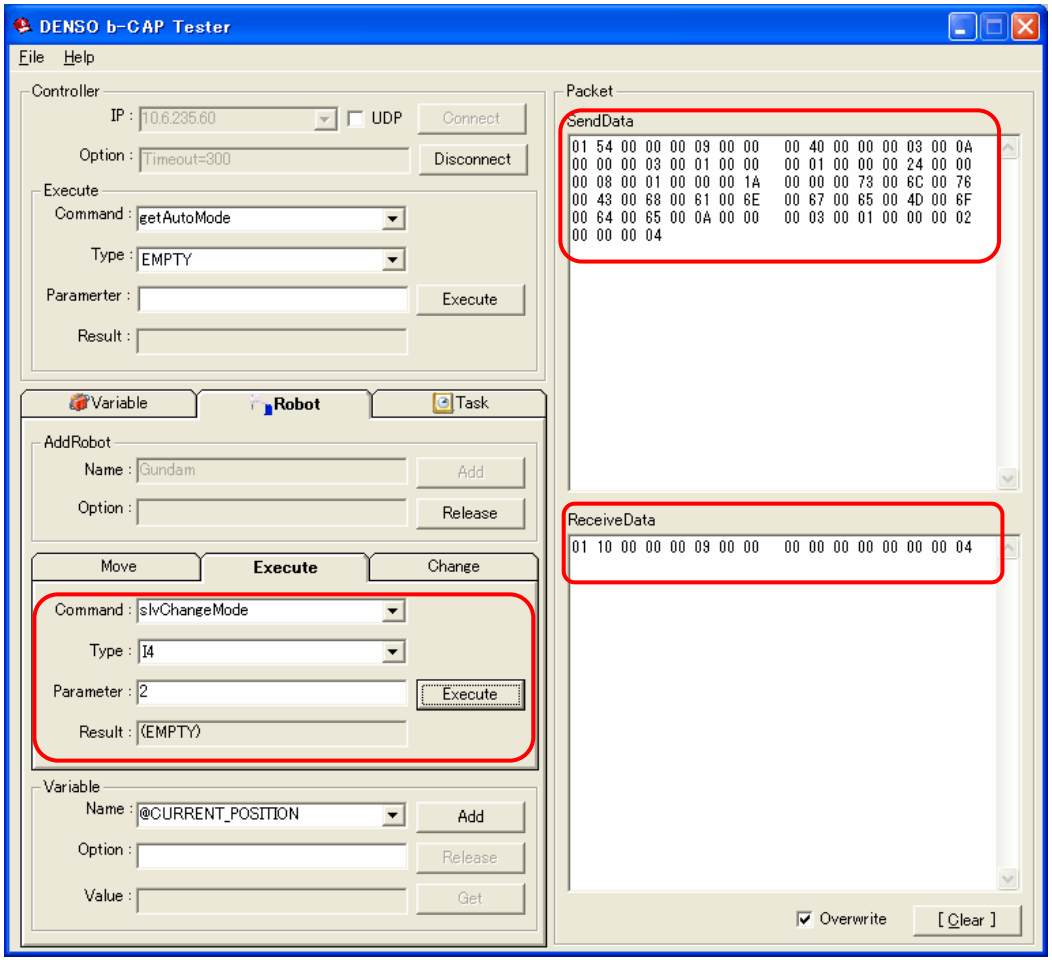


- Robot begins to move when you push “Start” Button.



Show packets(1)

- Change slave mode by “slvChangeMode”.



Show packets(2)

- Move robot by “slvMove”.
  - After running "slvMove", error occurs at RC.