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: 166581-1CD
Revision: 0
MANDATORY

• This manual explains the programming pendant customization function with the DX100/DX200/FS100 system. Read this manual carefully and be sure to understand its contents before handling the DX100/DX200.

• General items related to safety are listed in Chapter 1: Safety of the DX100/DX200 Instructions. To ensure correct and safe operation, carefully read the DX100/DX200 Instructions before reading this manual.

CAUTION

• Some drawings in this manual are shown with the protective covers or shields removed for clarity. Be sure all covers and shields are replaced before operating this product.

• The drawings and photos in this manual are representative examples and differences may exist between them and the delivered product.

• YASKAWA may modify this model without notice when necessary due to product improvements, modifications, or changes in specifications. If such modification is made, the manual number will also be revised.

• If your copy of the manual is damaged or lost, contact a YASKAWA representative to order a new copy. The representatives are listed on the back cover. Be sure to tell the representative the manual number listed on the front cover.

• YASKAWA is not responsible for incidents arising from unauthorized modification of its products. Unauthorized modification voids your product’s warranty.
Notes for Safe Operation

Read this manual carefully before installation, operation, maintenance, or inspection of the MOTOMAN-MH6-10/MH6F-10.

In this manual, the Notes for Safe Operation are classified as “WARNING”, “CAUTION”, “MANDATORY”, or “PROHIBITED”.

**WARNING**

Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury to personnel.

**CAUTION**

Indicates a potentially hazardous situation which, if not avoided, could result in minor or moderate injury to personnel and damage to equipment. It may also be used to alert against unsafe practices.

**MANDATORY**

Always be sure to follow explicitly the items listed under this heading.

**PROHIBITED**

Must never be performed.

Even items described as “CAUTION” may result in a serious accident in some situations.

At any rate, be sure to follow these important items:

---

**NOTE**

To ensure safe and efficient operation at all times, be sure to follow all instructions, even if not designated as “CAUTION” and “WARNING”.

---
**WARNING**

- Before operating the manipulator, check that servo power is turned OFF pressing the emergency stop buttons on the front door of the DX100/DX200 and the programming. When the servo power is turned OFF, the SERVO ON LED on the programming is turned OFF.

Injury or damage to machinery may result if the emergency stop circuit cannot stop the manipulator during an emergency. The manipulator should not be used if the emergency stop buttons do not function.

*Figure 1: Emergency Stop Button*

- Once the emergency stop button is released, clear the cell of all items which could interfere with the operation of the manipulator. Then turn the servo power ON.

Injury may result from unintentional or unexpected manipulator motion.

*Figure 2: Release of Emergency Stop*

- Observe the following precautions when performing teaching operations within the P-point maximum envelope of the manipulator:
  - View the manipulator from the front whenever possible.
  - Always follow the predetermined operating procedure.
  - Keep in mind the emergency response measures against the manipulator’s unexpected motion toward you.
  - Ensure that you have a safe place to retreat in case of emergency.

Improper or unintended manipulator operation may result in injury.

- Confirm that no person is present in the P-point maximum envelope of the manipulator and that you are in a safe location before:
  - Turning ON the power for the DX100/DX200.
  - Moving the manipulator with the programming.
  - Running the system in the check mode.
  - Performing automatic operations.

Injury may result if anyone enters the P-point maximum envelope of the manipulator during operation. Always press an emergency stop button immediately if there is a problem.

The emergency stop buttons are located on the right of front door of the DX100/DX200 and the programming.
<FS100>

**WARNING**

- Before operating the manipulator, check that servo power is turned OFF when the emergency stop button on the programming is pressed. When the servo power is turned OFF, the SERVO ON LED on the programming is turned OFF.

Injury or damage to machinery may result if the emergency stop circuit cannot stop the manipulator during an emergency.

*Figure 3: Emergency Stop Button*

- In the case of not using the programming, be sure to supply the emergency stop button on the equipment. Then before operating the manipulator, check to be sure that the servo power is turned OFF by pressing the emergency stop button. Connect the external emergency stop button to the 5-6 pin and 16-17 pin of the robot system signal connector (CN2).

- Upon shipment of the FS100, this signal is connected by a jumper cable in the dummy connector. To use the signal, make sure to supply a new connector, and then input it.

If the signal is input with the jumper cable connected, it does not function, which may result in personal injury or equipment damage.

- Once the emergency stop button is released, clear the cell of all items which could interfere with the operation of the manipulator. Then turn the servo power ON.

Injury may result from unintentional or unexpected manipulator motion.

*Figure 4: Release of Emergency Stop*

- Observe the following precautions when performing teaching operations within the manipulator's operating range:
  - Always follow the predetermined operating procedure.
  - Keep in mind the emergency response measures against the manipulator's unexpected motion toward you.
  - Ensure that you have a safe place to retreat in case of emergency.

Improper or unintended manipulator operation may result in injury.

- Confirm that no person is present in the manipulator's operating range and that you are in a safe location before:
  - Turning ON the FS100 power.
  - Moving the manipulator with the programming.
  - Running the system in the check mode.
  - Performing automatic operations.

Injury may result if anyone enters the manipulator's operating range during operation. Always press the emergency stop button immediately if there is a problem. The emergency stop button is located on the right of the programming.
CAUTION

- Perform the following inspection procedures prior to conducting manipulator teaching. If problems are found, repair them immediately, and be sure that all other necessary processing has been performed.
  - Check for problems in manipulator movement.
  - Check for damage to insulation and sheathing of external wires.
- For the DX100/DX200, always return the programming to the hook on the cabinet of the DX100/DX200 after use. For the FS100, always return the programming to a safe place after use.
  The programming can be damaged if it is left in the manipulator's work area, on the floor, or near fixtures.
- Read and understand the Explanation of Warning Labels in the DX100/DX200/FS100 Instructions before operating the manipulator:
Definition of Terms Used Often in This Manual (DX100/DX200)

The MOTOMAN is the YASKAWA industrial robot product.
The MOTOMAN usually consists of the manipulator, the controller, the programming, and supply cables.

In this manual, the equipment is designated as follows:

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Manual Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX100/DX200 controller</td>
<td>DX100/DX200</td>
</tr>
<tr>
<td>DX100/DX200 programming</td>
<td>Programming</td>
</tr>
<tr>
<td>Cable between the manipulator and the controller</td>
<td>Manipulator cable</td>
</tr>
</tbody>
</table>

Definition of Terms Used Often in This Manual (FS100)

The MOTOMAN is the YASKAWA industrial robot product.
The MOTOMAN usually consists of the manipulator, the FS100 controller, manipulator cables, the FS100 programming (optional), and the FS100 programming dummy connector (optional).

In this manual, the equipment is designated as follows:

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Manual Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>FS100 controller</td>
<td>FS100</td>
</tr>
<tr>
<td>FS100 programming</td>
<td>Programming</td>
</tr>
<tr>
<td>Cable between the manipulator and the controller</td>
<td>Manipulator Cable</td>
</tr>
<tr>
<td>FS100 programming dummy connector</td>
<td>Programming dummy connector</td>
</tr>
</tbody>
</table>
Descriptions of the programming keys, buttons, and displays are shown as follows:

<DX100>

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Manual Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming</td>
<td></td>
</tr>
<tr>
<td>Character Keys</td>
<td>The keys which have characters printed on them are denoted with [], ex. [ENTER]</td>
</tr>
<tr>
<td>Symbol Keys</td>
<td>The keys which have a symbol printed on them are not denoted with [], but depicted with a small picture, ex. page key</td>
</tr>
<tr>
<td>Axis Keys / Numeric Keys</td>
<td>“Axis Keys” and “Numeric Keys” are generic names for the keys for axis operation and number input.</td>
</tr>
<tr>
<td>Keys pressed simultaneously</td>
<td>When two keys are to be pressed simultaneously, the keys are shown with “+” sign between them, ex. [SHIFT]+[COORD]</td>
</tr>
<tr>
<td>Displays</td>
<td>The menu displayed in the programming is denoted with {}. ex. {JOB}</td>
</tr>
</tbody>
</table>

<DX200>

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Manual Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming / Pendant</td>
<td></td>
</tr>
<tr>
<td>Character Keys / Symbol Keys</td>
<td>The keys which have characters or its symbol printed on them are denoted with [], ex. [ENTER]</td>
</tr>
<tr>
<td>Axis Keys / Numeric Keys</td>
<td>[Axis Key] and [Numeric Key] are generic names for the keys for axis operation and number input.</td>
</tr>
<tr>
<td>Keys pressed simultaneously</td>
<td>When two keys are to be pressed simultaneously, the keys are shown with “+” sign between them, ex. [SHIFT]+[COORD]</td>
</tr>
<tr>
<td>Displays</td>
<td>The menu displayed in the programming pendant is denoted with {}. ex. {JOB}</td>
</tr>
</tbody>
</table>
Description of the Operation Procedure

In the explanation of the operation procedure, the expression "Select • • • " means that the cursor is moved to the object item and the SELECT key is pressed, or that the item is directly selected by touching the window.

Registered Trademark

In this manual, names of companies, corporations, or products are trademarks, registered trademarks, or brand names for each company or corporation. The indications of (R) and TM are omitted.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Programming Pendant Customizing Function</td>
<td>1-1</td>
</tr>
<tr>
<td>2 Function Outline</td>
<td>2-1</td>
</tr>
<tr>
<td>3 Function Setting</td>
<td>3-1</td>
</tr>
<tr>
<td>4 Parameter Setting</td>
<td>4-1</td>
</tr>
<tr>
<td>5 API for Programming Pendant Customization Function (MotoConnect API)</td>
<td>5-1</td>
</tr>
<tr>
<td>6 Interface with Programming Main Software</td>
<td>6-1</td>
</tr>
<tr>
<td>7 Startup Method of the User-Customization Application</td>
<td>7-1</td>
</tr>
<tr>
<td>7.1 Start from the Main Menu</td>
<td>7-1</td>
</tr>
<tr>
<td>7.2 Auto Run at Power ON</td>
<td>7-1</td>
</tr>
<tr>
<td>7.3 One Touch Run with a Button</td>
<td>7-2</td>
</tr>
<tr>
<td>8 Installing Applications</td>
<td>8-1</td>
</tr>
<tr>
<td>8.1 Create the Installing Data</td>
<td>8-1</td>
</tr>
<tr>
<td>8.1.1 Zip File</td>
<td>8-1</td>
</tr>
<tr>
<td>8.1.2 INI File</td>
<td>8-1</td>
</tr>
<tr>
<td>8.2 Creating Installer</td>
<td>8-5</td>
</tr>
<tr>
<td>8.2.1 Necessary File</td>
<td>8-5</td>
</tr>
<tr>
<td>8.2.2 Describing &quot;pp_install.ini&quot;</td>
<td>8-6</td>
</tr>
<tr>
<td>8.3 Installation</td>
<td>8-7</td>
</tr>
<tr>
<td>9 Guideline for Creating Application</td>
<td>9-1</td>
</tr>
<tr>
<td>9.1 Priority of the Application</td>
<td>9-1</td>
</tr>
<tr>
<td>9.2 CPU Utilization by Application</td>
<td>9-1</td>
</tr>
<tr>
<td>9.3 Successive Access to the DX100/DX200/FS100.</td>
<td>9-1</td>
</tr>
<tr>
<td>9.4 Files Created by Application</td>
<td>9-2</td>
</tr>
<tr>
<td>9.5 Response to Alarm</td>
<td>9-2</td>
</tr>
<tr>
<td>9.6 Application Test</td>
<td>9-3</td>
</tr>
<tr>
<td>9.6.1 Standard Operation Test</td>
<td>9-3</td>
</tr>
<tr>
<td>9.6.2 Response to Alarm</td>
<td>9-4</td>
</tr>
<tr>
<td>9.6.3 Continuous Run</td>
<td>9-4</td>
</tr>
<tr>
<td>9.6.4 Window Switching</td>
<td>9-4</td>
</tr>
<tr>
<td>9.6.5 Cycle Time Check</td>
<td>9-4</td>
</tr>
<tr>
<td>9.7 Other</td>
<td>9-4</td>
</tr>
</tbody>
</table>
# Table of Contents

10 Development of Application in C# .............................................................. 10-1

10.1 Preparation ................................................................................................... 10-1

10.1.1 Developing Environment (Developing PC Environment) ...................... 10-1

10.2 Development of Application for the DX100/DX200 .................................... 10-1

10.2.1 Creation of Software (PC for Development) ........................................... 10-1

10.2.2 Controller Communication Library ....................................................... 10-2

10.2.3 Library Initialization Process ................................................................. 10-3

10.2.4 Interface with Programming Pendant Main Software ........................... 10-4

10.3 Development of Application for the FS100 ................................................. 10-8

10.3.1 Creation of Software (PC for Development) ........................................... 10-8

10.3.2 Controller Communication Library ....................................................... 10-8

10.3.3 Library Initialization Process ................................................................. 10-9

10.3.4 Interface with Programming Pendant Main Software ........................... 10-11

10.4 Application Debugging ............................................................................. 10-15

10.4.1 Parameter Setting ................................................................................. 10-15

10.4.2 Debugging ......................................................................................... 10-15

11 Interface Specification with Programming Pendant Main Software ................ 11-1

11.1 Command from the User Customized Application to Programming Pendant Main Software ................................. 11-1

11.1.1 Completion of Generation .................................................................... 11-1

11.1.2 Closing of Application ....................................................................... 11-1

11.1.3 Keymask .......................................................................................... 11-2

11.1.4 Text Input Function Start .................................................................... 11-3

11.1.5 Switching Control ............................................................................ 11-3

11.1.6 Setting of Notifying Information ......................................................... 11-4

11.1.7 Information Request .......................................................................... 11-4

11.1.8 Version Information Acquisition ........................................................ 11-5
11.2 Command from the Programming Pendant Main Software to the User-Customized Application

11.2.1 Input Character

11.2.2 Request of Control System Switching

11.2.3 Status Information

11.2.4 Alarm

11.2.5 Message

11.2.6 Language

11.2.7 Version
1 Programing Pendant Customizing Function

With this function, user-customized windows can be created by, into the DX100/DX200/FS100 programming, incorporating a customized application (WindowsCE application software) developed by the user (system integrator).

Also, this function provides APIs for the customized function (function group) which are available at the customized application (Windows CE application software) developed by the user (system integrator).

With these APIs, from the user-developing applications, acquiring of the DX100/DX200/FS100 status information or writing/reading of the data are possible.

The size of the windows that can be created with this function are full-screen sized window only.

Indication of the user-customized window when turning ON the power supply of the DX100/DX200/FS100 or the standard window when it is online status can be chosen.
2 Function Outline

This function enables creating windows for the user-customized (a separate exe file from the YASKAWA standard application) and incorporating them into the programming. To execute this function, the following three functions are provided to realize this function.

1. The API that can access (Read/Write) the data in the DX100/DX200/FS100.

2. Data sending/receiving function (communication function between processes) between the YASKAWA standard application and the application for the user-customized window.

3. Function to install the user-customized application to the DX100/DX200/FS100 and register it to the menu of the system.
3 Function Setting

This function is an optional function of the DX100/DX200/FS100. To invalidate this function, like other functions, controller-based setting is necessary.
## Parameter Setting

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
<th>Setting Value</th>
<th>Default</th>
</tr>
</thead>
</table>
| S2C531 | Automatic start of user-customized application function in the programming when start-up. | 0: Invalid  
1: Valid  | 0 (Invalid) |
| S2C541 | Writing variables and I/O during play mode.                               | 0: Permitted  
1: Not permitted  | 1 (Not permitted) |
| S2C542 | Writing variables and I/O during edit-lock status.                       | 0: Invalid  
1: Valid  | 1 (Not permitted) |

### NOTE

Writing of variables and I/Os become valid when “0” is set to S2C541, however, it may influence the manipulator’s cycle time.

### SUPPLEMENT

The term “during edit-lock status” in the explanation of parameter No. S2C542 above means following statuses.

- During alarm occurrence
- During external memory device operation
- During the execution of data transmission function
- During specific input “EDIT_LOCK(#40064)” ON
5 API for Programming Pendant Customization Function (MotoConnect API)

For reading/writing data to/from the user-customized application to/from the DX100/DX200/FS100, APIs are prepared.

There are three main types of APIs as follows. For the details of each API, refer to “DX100/DX200/FS100 OPTIONS INSTRUCTIONS REFERENCE MANUAL FOR TEACHING CUSTOMIZING FUNCTION” (Part No.: 166582-1CD).

<table>
<thead>
<tr>
<th>System</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System monitor</td>
<td>Acquires the DX100/DX200/FS100 data. APIs for data reading.</td>
</tr>
<tr>
<td>System control</td>
<td>Operates the DX100/DX200/FS100. APIs for data writing are included.</td>
</tr>
<tr>
<td>File transmission</td>
<td>Transfer files with the function equivalent to External Memory Function.</td>
</tr>
</tbody>
</table>
6  Interface with Programming Main Software

The interface to perform screen switching and data transmission between the YASKAWA standard applications and the user-customized applications are prepared.

For the details of the interface with the YASKAWA standard window applications, refer to “DX100/DX200/FS100 OPTIONS INSTRUCTIONS REFERENCE MANUAL FOR TEACHING CUSTOMIZING FUNCTION” (Part No.: 166582-1CD).
7 Startup Method of the User-Customization Application

There are three ways of starting method of the user-customized application as follows.

- Start from the main menu.
- Auto run at power ON.
- One touch run with a button.

7.1 Start from the Main Menu

Like indicating the standard window, this application can be started from the main menu. Where to register on the main menu or which notation to indicate can be specified on the application basis.

7.2 Auto Run at Power ON

The user-customized application can be started same time when the power supply of the DX100/DX200/FS100 is turned ON. Set this auto run function when installing the application.

Only one user-customized application can be automatically started in the DX100/DX200/FS100 system and this application can be started also from the main menu.

Before using this function, be sure to set “1(valid)” to the programming pendant customization application (S2C531=1) Auto Run function of at power ON.
7.3 One Touch Run with a Button

To start the user-customized application by a single operation, start-up function of this application can be allocated to a button at the lower right of the programming pendant window.

Allocate the application to the button when installing it.

Only one application can be allocated to the button for One Touch Run function and this application can be started also from the main menu.

When putting a name to the button, the name will be the same one as the name of the sub menu. Up to fourteen characters can be used as its name and it may not be indicated properly if it exceeded.
8 Installing Applications

8.1 Create the Installing Data

8.1.1 Zip File

Zip applications (*.exe), accessory files (dll, data files, etc.), and accessory folders with its application’s name. And place the application (*.exe) on the root of zip file.

For example, if the name of the created application is AmpleApp.exe, and that of the library used in the application is SampleApp.dll and zip them. The name of the zip file should be SampleApp.zip.

For zipping, use the zipping tool regularly used on the desk top.

8.1.2 INI File

To an INI file, please describe the information necessary for installing the application. For the name of the INI file, put the same name as the zip file.

Following is an example format of the SampleApp’s INI file (SampleApp.ini).

```
[APPLICATION_INFO]
APP_NAME = SampleApp
APP_VER = 1.00-00-YE
SYS_VER = NS0.00-00
FILE_SIZE = 53
FOLDER_NAME = SampleNx
APP_TYPE = APP_TYPE_OVERLAPPED_EXE
SYS_STATUS = 0x01
AUTO_RUN = FALSE

[MENU_INFO]
TOPMENU.ID = 12
SUBMENU_NAME_JAPANESE = “サンプル アプリ”
SUBMENU_NAME_ENGLISH = “SampleApp”
SUBMENU_ICON = SampleApp.ico
```
## Installing Applications

### 8.1 Create the Installing Data

#### [APPLICATION_INFO]

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>APP_NAME</strong></td>
<td>Name of the application. It is same as application root folder name and exe.file name (extension is excluded). This name functions as the identifier of this application. Restrictions: Within 12 or less of single-byte characters.</td>
</tr>
<tr>
<td><strong>APP_VER</strong></td>
<td>Version of the application itself, and it is described as &quot;<strong>.</strong>-<strong>-developer name&quot;. * one-byte character Developer name: abbreviated name of the company which developed the application. Ex.1.00-00-YE Restrictions: Use 11 of single-byte characters like &quot;</strong>.<strong>-</strong>&quot;.</td>
</tr>
<tr>
<td><strong>SYS_VER</strong></td>
<td>To keep compatibility with the controllers, the version is not checked. Specify &quot;SYS_VER=NS0.00-00&quot;.</td>
</tr>
<tr>
<td><strong>FILE_SIZE</strong></td>
<td>Describes the total size of decompressed file group. (including files in the sub folders) It is used to check the disk capacity when installing the file. Unit: KB</td>
</tr>
<tr>
<td><strong>FOLDER_NAME</strong></td>
<td>Name of the folder created in the programming pendant when this application is installed. The application is decompressed and saved under DiskOnChip/PP_APP &quot;folder name&quot; when installed. If the name is not specified, the application name is adopted instead.</td>
</tr>
<tr>
<td><strong>APP_TYPE</strong></td>
<td>Specify &quot;APP_TYPE_OVERLAPPED_EXE&quot;. In the programming pendant customization function, only the application with which a full-screen sized window type can be created.</td>
</tr>
<tr>
<td><strong>SYS_STATUS</strong></td>
<td>(Displaycondition on the basis of system state) Indication of the programming pendant customization function window can be restricted on the basis of its system status. When a certain status cannot be displayed, the menu of the status will not be displayed on the main menu.</td>
</tr>
<tr>
<td></td>
<td>0x01: Concealed when edit is locked 0x02: Concealed when alarming 0x04: Concealed in the teach mode 0x08: Concealed in the play mode 0x00: Displayed in the operation mode or higher 0x40: Displayed in the edit mode or higher 0x80: Displayed in the management mode or higher</td>
</tr>
<tr>
<td></td>
<td>Note: Specified in hex. notation. Ex.SYS_STATUS=0x4a  Displayed as a menu on the main menu in the edit or management mode. Concealed from the main menu in the play mode or when alarming.</td>
</tr>
<tr>
<td><strong>AUTO_RUN</strong></td>
<td>(Auto Run Function Enabled/Disabled) Auto Run function valid = &quot;TRUE&quot; Auto Run function valid = &quot;FAULTSE&quot; If a certain application is to be started up automatically, set &quot;TRUE&quot; to Auto Run function when installing the application. Note: It is usually set &quot;TRUE&quot; when the operation from the programming pendant is not necessary.</td>
</tr>
</tbody>
</table>
# Programming Pendant

## Customization

### 8 Installing Applications

#### 8.1 Create the Installing Data

> <DX100/FS100>

<table>
<thead>
<tr>
<th><strong>[MENU_INFO]</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TOPMENU_ID</strong></td>
<td>(Top menu) Specify a top menu to add the application.</td>
</tr>
<tr>
<td>Top menu</td>
<td>Setting value</td>
</tr>
<tr>
<td>JOB</td>
<td>1</td>
</tr>
<tr>
<td>WORK CONDITION</td>
<td>2</td>
</tr>
<tr>
<td>VARIABLE</td>
<td>3</td>
</tr>
<tr>
<td>I/O</td>
<td>4</td>
</tr>
<tr>
<td>ROBOT</td>
<td>5</td>
</tr>
<tr>
<td>SYSTEM INFO.</td>
<td>6</td>
</tr>
<tr>
<td>FD/PC CARD</td>
<td>7</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>8</td>
</tr>
<tr>
<td>CONTROLLER SETTING</td>
<td>9</td>
</tr>
<tr>
<td>OPTIONAL</td>
<td>10</td>
</tr>
<tr>
<td>MANUFACTURE RESERVE</td>
<td>11</td>
</tr>
<tr>
<td>(FUTURE EXPANSION)</td>
<td>12-20</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>21</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SUBMENU_NAME_</strong>*</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Submenu Name) Specify a character string for indication in the sub menu. Desired language can be chosen from the followings. Note: Up to 20 single-byte characters. Up to 14 single-byte characters for the name of OneTouch Run function button.</td>
<td></td>
</tr>
<tr>
<td>SUBMENU_NAME_JAPANESE</td>
<td>&quot;12345678901234567890&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_ENGLISH</td>
<td>&quot;BBB&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_GERMAN</td>
<td>&quot;CCC&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_SWEDISH</td>
<td>&quot;DDD&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_FRENCH</td>
<td>&quot;EEE&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_FINNISH</td>
<td>&quot;FFF&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_ITALIAN</td>
<td>&quot;GGG&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_SPANISH</td>
<td>&quot;HHH&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_HANGLER</td>
<td>&quot;III&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_CHINESE</td>
<td>&quot;JJJ&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_TAIWANESE</td>
<td>&quot;KKK&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_CZECH</td>
<td>&quot;LLL&quot;</td>
</tr>
<tr>
<td>SUBMENU_NAME_POLISH</td>
<td>&quot;MMM&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SUBMENU_ICON</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Submenu Icon) (for future expansion) From the icon file, specify an icon to be indicated in the sub menu.</td>
<td></td>
</tr>
</tbody>
</table>
8 Installing Applications
8.1 Create the Installing Data

<DX200>

<table>
<thead>
<tr>
<th>[MENU_INFO]</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPMENU_ID</td>
<td>Specify a top menu to add the application.</td>
</tr>
<tr>
<td></td>
<td>Setting value</td>
</tr>
<tr>
<td>TOP MENU</td>
<td></td>
</tr>
<tr>
<td>JOB</td>
<td>1</td>
</tr>
<tr>
<td>WORK CONDITION</td>
<td>2</td>
</tr>
<tr>
<td>VARIABLE</td>
<td>3</td>
</tr>
<tr>
<td>IN/OUT</td>
<td>4</td>
</tr>
<tr>
<td>ROBOT</td>
<td>5</td>
</tr>
<tr>
<td>SYSTEM INFO.</td>
<td>6</td>
</tr>
<tr>
<td>EX.MEMORY</td>
<td>7</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>8</td>
</tr>
<tr>
<td>SETUP</td>
<td>9</td>
</tr>
<tr>
<td>SAFETY FUNC.</td>
<td>10</td>
</tr>
<tr>
<td>PM</td>
<td>11</td>
</tr>
<tr>
<td>OPTION</td>
<td>12</td>
</tr>
<tr>
<td>MANUFACTURE RESERVE</td>
<td>13-20</td>
</tr>
<tr>
<td>APPLICATION</td>
<td>21</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUBMENU_NAME_*</th>
<th>Specify a character string for indication in the sub menu.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desired language can be chosen from the followings.</td>
</tr>
<tr>
<td></td>
<td>Note: Up to 20 single-byte characters.</td>
</tr>
<tr>
<td></td>
<td>Up to 14 single-byte characters for the name of OneTouch Run function button.</td>
</tr>
<tr>
<td>SUBMENU_NAME_JAPANESE</td>
<td>“12345678901234567890”</td>
</tr>
<tr>
<td>SUBMENU_NAME_ENGLISH</td>
<td>“BBB”</td>
</tr>
<tr>
<td>SUBMENU_NAME_GERMAN</td>
<td>“CCC”</td>
</tr>
<tr>
<td>SUBMENU_NAME_SWEDISH</td>
<td>“DDD”</td>
</tr>
<tr>
<td>SUBMENU_NAME_FRENCH</td>
<td>“EEE”</td>
</tr>
<tr>
<td>SUBMENU_NAME_FINNISH</td>
<td>“FFF”</td>
</tr>
<tr>
<td>SUBMENU_NAME_ITALIAN</td>
<td>“GGG”</td>
</tr>
<tr>
<td>SUBMENU_NAME_SPANISH</td>
<td>“HHH”</td>
</tr>
<tr>
<td>SUBMENU_NAME_HANGE</td>
<td>“III”</td>
</tr>
<tr>
<td>SUBMENU_NAME_CHINESE</td>
<td>“JJJ”</td>
</tr>
<tr>
<td>SUBMENU_NAME_TAIWANESE</td>
<td>“KKK”</td>
</tr>
<tr>
<td>SUBMENU_NAME_CZECH</td>
<td>“LLL”</td>
</tr>
<tr>
<td>SUBMENU_NAME_POLISH</td>
<td>“MMM”</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUBMENU_ICON</th>
<th>(Submenu Icon) (for future expansion)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>From the icon file, specify an icon to be indicated in the sub menu.</td>
</tr>
</tbody>
</table>
8.2 Creating Installer

8.2.1 Necessary File

Create PP_APP folder on the CF for version up root folder.

- YE_AppInstall.exe: Application installer
- pp_install.ini: Configuration file of the installer. Specifies the default selection of the added application when startup.
- Unzip.dll: Zip file decompresses
- *.zip: Application compressed file
- *.ini: Application configuration information file

Example of PP_APP folder
8 Installing Applications
8.2 Creating Installer

8.2.2 Describing “pp_install.ini”

For the applications described in pp_install.ini, “add” is automatically set to (Operation) when the installer is started-up.

And for the applications not described in pp_install.ini, they can be installed by setting “add” before installation.

APP_NUM: Specify the numbers of applications to be set
APP_NAME1: Specify the name of applications to be set

[Default Setting]
APP_NUM=3
AMM_NAME1=APITest1
AMM_NAME2=APITest2
AMM_NAME3=OperationArea
8.3 Installation

Press [INTERLOCK] + [7]+ [SELECT] keys and turn ON the power supply of the DX100/DX200/FS100.

The user-customized application installation window opens if the user-customized application installing environment is equipped to the CF or USB memory.

1. Check that “Add” is set to “Operation” of all the installing applications.

2. Press {Option} button when installing the application which uses the following functions.
   - Auto Run function or
   - One Touch Run function
3. Optional setting window appears. Set Auto Run application and One Touch Run application. The already-set applications are indicated if any.

[Auto Run Application]
Select an application which specifies the Auto Run function at power-ON. In the list, an application which specifies "AUTO_RUN=TRUE" with the "application name.ini" file is indicated. In case canceling this section, select a blank line.

[One Touch Run Application]
Select an application which specifies the One Touch Run with a Button function. In the list, already-registered applications and those applications to which "Add" is set to "Operation" on the window are indicated. In case canceling this selection, select a blank line.
8. Installing Applications
8.3 Installation

4. Press (Return) button after the above setting. The window returns to the previous one. The indication varies depending on the setting as follows.

- “A” is indicated to the application specified as the Auto Run application at power ON.
- “O” is indicated to the application specified as the One Touch Run with a button.
- “*” is indicated to the application to which both functions are specified.

5. After all settings are completed, press (Execute) button. Installation of the applications starts. Do not turn OFF the power supply before the installation is done because the files are being developed to both DX100/DX200/FS100 and programming pendant.
6. When the installation is completed, the “Operation” section is cleared and versions and sizes of each application are indicated. (This state is granted as completion of installation.)

The installed application data is controlled by both DX100/ DX200/FS100 and programming pendant. Therefore, for the data consistency, within the installed data, the inconsistent data between the DX100/DX200/FS100 and the programming pendant are deleted next time when the data is newly installed.

In case the programming pendant is replaced after the user-customized application is installed, the inconsistency of the data may occur between the DX100/DX200/FS100 and the programming pendant.
9  Guideline for Creating Application

The created user-customized application is incorporated to the manipulator and functions as a part of the manipulator system. Therefore, it may interfere the system if it is not processed properly. To avoid this, a guideline for creating the user-customized application is provided. Follow this guideline when creating user-customized applications.

9.1 Priority of the Application

For the system stability, it is recommended to set “BELOW_NORMAL” to the thread priority of the user-customized application. However, it is prohibited to set higher than “NORMAL” to the thread priority since it may influence a lot to the YASKAWA standard programming pendant window.

In case the higher priority than “NORMAL” is set to the application, the YASKAWA standard programming pendant window does not work properly and communication faults (PP communication fault) with the DX100/DX200/FS100 may occur.

9.2 CPU Utilization by Application

It is prohibited to execute a process that the user-customized application continuously occupies the CPU. To avoid this to happen, create a software in which the user-customized application does not occupy the CPU for more than 500 sec (within 300sec is recommended).

In case a process is execute which the created application continuously occupies the DX100/DX200/FS100, the YASKAWA standard programming pendant window does not work properly and communication faults (PP communication fault) with the DX100/DX200/FS100 may occur.

9.3 Successive Access to the DX100/DX200/FS100

It is prohibited to execute a process that the user-customized application continuously make access to the DX100/DX200/FS100. (Consecutive writing of the variables, for example.) The application is recommended to make a call to the API at intervals of more than 50 sec (100 msec is strongly recommended).

In case a process is execute which the created application continuously access to DX100/DX200/FS100, processing operation in the DX100/DX200/FS100 may be influenced as follows.

- Operations during the teach mode may slow down.
- Cycle times when play back operation may slow down.
9.4 Files Created by Application

Save the file created by the application to the CF (Storage Card) in the programming pendant, to the USB memory or to the root folder (on the memory). Saving it to the flash memory (DiskOnChip) in the programming pendant, including to the temporary file, is prohibited.

Since necessary files for the robot controller system is saved in the DiskOnChip, the YASKAWA standard programming pendant window does not work properly or it won’t start up in case those files are influenced.

9.5 Response to Alarm

An alarm occurs when an error is found in the robot controller system. When an alarm occurs while the user-customized application window is displayed, it is reported from the YASKAWA standard window to the user-customized application.

To deal with this error, please prepare methods at the user-customized application side.

Create applications when an alarm occurs as follows.

- Create an application which closes the user-customized application and indicates an alarm window on the pendant YASKAWA standard window to notify the user that it is alarming.
- Create an application which creates methods to notify the user that an alarm is occurring in the user-customized application.
- Other than above.
9.6 Application Test

Carry out sufficient test to the created user-customized applications.

9.6.1 Standard Operation Test

As tests for the application’s standard operations, check the occurrence of errors or alarms while the created application is running. By conducting these tests, influences by the user-customized application to the robot controller system and to the YASKAWA standard window can be confirmed.

The definition of the standard operations of the created applications are as follows.

1. Change the mode to the teach mode.
2. Create a new job.
3. After moving by the job operation, teach the move instruction. (MOVJ VJ=10 (or around 10))
4. Insert JUMP*LABEL to the end of the job.
5. Insert *LABEL to the head of the job.
6. Move the “First step” by pressing [FWD] key to move.
7. Change the mode to the play mode.
8. Indicate the cycle time.
9. Indicate the job stack.
10. Press {Start} button and execute playback operation of the Job.

Input B variables by following the procedures below. (For inputting I, D, and R variables procedures, substitute the same procedures.)

1. Select {VARIABLE} - {BYTE VARIABLE} on the main menu.
2. Change the valuable on B variable window. Input 0, 128, 255, 256 or 1024. When inputting 256 or 1024, check that the range check works properly.
9.6.2 Response to Alarm

To confirm the alarming condition when indicating the user-created application, conduct tests to the following two cases. And check if the alarm is properly processed as defined at the user-created application side.

- 24V power supply error: Start an alarm by disconnecting YIF board.
- Alarm during play back operation: Intentionally exceed the limit by a move instruction in which the P variable (cartesian value) is used.

9.6.3 Continuous Run

As for the Jobs for continuous run tests, prepare Jobs for programming pendant continuous test including layer check of variety of calculation, R1,S1 and call job. Execute continuous run of these Jobs and check the occurrence of alarms or errors.

9.6.4 Window Switching

Test if switching of the window from the YASKAWA standard window to the user-customized application window is possible.

For the user-customized application, keep indicating the window to call up the API while test is executed. And check if those windows alternate properly.

9.6.5 Cycle Time Check

Prepare a cycle time checking Job. Before the test, record the cycle time (in the status that any APIs are not executed in the user-customized application), then check the cycle time after API is executed in the application. There will be an error for several segments time to the cycle time due to the Job start time, but it can be ignored.

9.7 Other

Other than mentioned above, it prohibited to execute any processes than may influence the whole robot controller system or the YASKAWA standard programming pendant window.
10 Development of Application in C#

10.1 Preparation

10.1.1 Developing Environment (Developing PC Environment)

1. Windows CE 5.0: Install

2. MS Visual Studio: Install
   - DX100/FS100: MS Visual Studio 2005
     Note: The debugging procedure is partially added in case Visual studio 2005 ServicePack is installed.
   - DX200: MS Visual Studio 2008

3. MS .NET Compact Framework: Install
   - DX100/FS100: MS .NET Compact Framework 2.0 SP2
   - DX200: MS .NET Compact Framework 3.5

4. SDK: Install SDK for programming pendant

5. IP address setting for development PC
   Set the IP address of the developing PC as follows.
   
   IP Address : 10.0.0.3
   Sub-net Mask : 255.255.255.0

10.2 Development of Application for the DX100/DX200

There are following differences between the DX100/DX200 and the FS100 when developing applications. Refer to Section 10.3 “Development of Application for the FS100” on page 10-10 developing applications for the FS100.

- Initializing process
  In the FS100 system, modification of the FS100 IP address and the programming pendant is possible. Therefore, it is required to receive them from the programming pendant main application as the parameter of the application, and deliver them as the argument of YPPConnectOpen() when the application starts up.

NOTE

The library for C# (MotoConnectCSU.DLL) is common for both DX100, DX200 and FS100.

10.2.1 Creation of Software (PC for Development)

1. With Visual Studio, newly create a project for creating application for Windows CE 5.0.
10.2.2 Controller Communication Library

1. Copy the MotoConnect in the folder under the Develop folder to the application folder to be created.

2. Add MotoConnectCSU.dll to the project by “Reference setting”.

![Controller Communication Library Diagram](image-url)
10.2.3 **Library Initialization Process**

Create a MotoConnectCS object in order to utilize the API of MotoConnectCS.

**[Code example]**

```csharp
public partial class Form1 : Form
{
    #region Initialize
    /// <summary>
    /// Command Line
    /// </summary>
    private string[] CommandLine;
    public string[] SetCommandLine
    {
        set
        {
            CommandLine = value;
        }
    }

    /// <summary>
    /// AppOption : Analize Command Line
    /// </summary>
    private AppOption AppOpt;
    /// <summary>
    /// AppMessageWindow : Communicate YppMain
    /// </summary>
    private AppMessageWindow AppMsg;

    /// <summary>
    /// Endian true:big / false:little
    /// </summary>
    private bool EndianData = false;

    /// <summary>
    /// MotoConnectCSU
    /// </summary>
    private MotoConnectCSU.MotoConnect MotoConnect;
    #endregion
}
```
10.2 Development of Application for the DX100/DX200

For sending and receiving data with the programming pendant main software, the user-customized application employs WM_COPYDATA for the communication between APIs.

Following are the important processing operations.

**Initializing Process <1>**

The following two arguments are delivered from the pendant main software, when the customized application is started up.

- Language code
- Window handle of the programming pendant main software

For the user-customized application, obtain and save the argument data.

[Code example]

```csharp
static class Program
{
    public Form1()
    {
        InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        this.Visible = false;

        // Set the command line argument
        AppOpt = new AppOption();
        AppOpt.SetOptionData( CommandLine );

        AppMsg =
            new AppMessageWindow(this, AppOpt.GetMainHWnd );

        MotoConnect =
            newMotoConnectCS.MotoConnect(EndianData);

        // Completion of start up command sending process
        AppMsg.SendCreateCommand(Handle);

        this.Visible = true;
    }
}
```
{  
    /// <summary>
    /// It is the main entry point of the application
    /// </summary>
    [MTAThread]
    static void Main(string[] args)
    {
        Form1 form1 = new Form1();
        form1.SetCommandLine = args;
        Application.Run(form1);
    }
}

public partial class Form1 : Form
{
    #region Initialize
    /// <summary>
    /// Command Line
    /// </summary>
    private string[] CommandLine;
    public string[] SetCommandLine
    {
        set
        {
            CommandLine = value;
        }
    }
    /// <summary>
    /// AppOption : Analyze Command Line
    /// </summary>
    /// <summary>
    /// AppMessageWindow : Communicate YppMain
    /// </summary>
    private AppOption AppOpt;
    private AppMessageWindow AppMsg;
}
For the AppOption class, refer to its source code (AppOption.cs) which will be provided.

**Initializing Process <2>**

Send the window handle of the application to the programming pendant main software at the timing when the user-customized application window is displayed. The programming pendant main software sends/receives messages to/from the window to which the window handle is sent.

**[Code example]**

```csharp
public partial class Form1 : Form
{
    ...

    private void Form1_Load(object sender, EventArgs e)
    {
        this.Visible = false;

        // Set the command line argument
        AppOpt = new AppOption();
        AppOpt.SetOptionData(CommandLine);

        AppMsg = new AppMessageWindow(this, AppOpt.GetMainHWnd);
        MotoConnect = new MotoConnectCS.MotoConnect(EndianData);

        // Completion of start up command sending process
        AppMsg.SendCreateCommand(Handle);

        this.Visible = true;
    }

    ...
```
Closing Process

Notify the programming pendant main software of the closing when the customized application is closed.

[Code example]

```csharp
public partial class Form1 : Form
{
    . . .

    private void button2_Click(object sender, EventArgs e)
    {
        // Close
        AppMsg.SendCloseCommand();
    }
}
```

For the AppMessageWindow class, refer to its source code (AppMessageWindow.cs) which will be provided.

AppMessageWindow.cs includes the following codes.
- WM_COPYDATA message receiving from the programming pendant main software
- WM_COPYDATA message sending to the programming pendant main software
- Examples of important process such as Initializing or closing process.
10.3 Development of Application for the FS100

10.3.1 Creation of Software (PC for Development)

With Visual Studio 2005, newly create a project for creating application for Windows CE 5.0.

10.3.2 Controller Communication Library

1. Copy the MotoConnect in the folder under the Develop folder to the application folder to be created.

2. Add MotoConnectCSU.dll to the project by “Reference setting”.

![Visual Studio Project Screenshot]
10.3.3 Library Initialization Process

Create a MotoConnectCS object in order to utilize the API of MotoConnectCS.

**Code example**

```csharp
public partial class Form1 : Form
{
    #region Initialize
    /// <summary>
    /// Command Line
    /// </summary>
    private string[] CommandLine;
    public string[] SetCommandLine
    {
        set
        {
            CommandLine = value;
        }
    }
    /// <summary>
    /// AppOption : Analize Command Line
    /// </summary>
    private AppOption AppOpt;
    /// <summary>
    /// AppMessageWindow : Communicate YppMain
    /// </summary>
    private AppMessageWindow AppMsg;
    /// <summary>
    /// Endian true:big / false:little
    /// </summary>
    private bool EndianData = false;
    /// <summary>
    /// MotoConnectCSU
    /// </summary>
    private MotoConnectCSU.MotoConnect MotoConnect;
    #endregion
}
```

/// <summary>
/// MotoConnectCSU
/// </summary>
private MotoConnectCSU.MotoConnect MotoConnect;
public Form1()
{
    InitializeComponent();
}

private void Form1_Load(object sender, EventArgs e)
{
    this.Visible = false;

    // Set the command line argument
    AppOpt = new AppOption();
    AppOpt.SetOptionData(CommandLine);

    AppMsg = new AppMessageWindow(this, AppOpt.GetMainHWnd);

    MotoConnect = new MotoConnectCS.MotoConnect(EndianData,

    // Completion of start up command sending process
    AppMsg.SendCreateCommand(Handle);

    this.Visible = true;
}

...
10.3.4 Interface with Programming Pendant Main Software

For sending and receiving data with the programming pendant main software, the user-customized application employs WM_COPYDATA for the communication between APIs.

Following are the important processing operations.

**Initializing Process <1>**

The following two arguments are delivered from the pendant main software, when the customized application is started up.

- Language code
- Window handle of the programming pendant main software

For the user-customized application, obtain and save the argument data.

**[Code example]**

```csharp
static class Program
{
    /// <summary>
    /// It is the main entry point of the application
    /// </summary>
    [MTAThread]
    static void Main(string[] args)
    {
        Form1 form1 = new Form1();
        form1.SetCommandLine = args;
        Application.Run(form1);
    }
}

public partial class Form1 : Form
{
    #region Initialize
    /// <summary>
    /// Command Line
    /// </summary>
    private string[] CommandLine;
    public string[] SetCommandLine
    {
        set
        {
            CommandLine = value;
        }
    }
}
```
For the AppOption class, refer to its source code (AppOption.cs) which will be provided.
Initializing Process <2>

Send the window handle of the application to the programming pendant main software at the timing when the user-customized application window is displayed. The programming pendant main software sends/receives messages to/from the window to which the window handle is sent.

[Code example]

```csharp
public partial class Form1 : Form
{

    . . .

    private void Form1_Load(object sender, EventArgs e)
    {
        this.Visible = false;

        // Set the command line argument
        AppOpt = new AppOption();
        AppOpt.SetOptionData( CommandLine );

        AppMsg = new AppMessageWindow
        ( this, AppOpt.GetMainHWnd );

        MotoConnect = new 
        MotoConnectCS.MotoConnect(EndianData,

        // Completion of start up command sending process
        AppMsg.SendCreateCommand( Handle );

        this.Visible = true;
    }

    . . .

}
```
Programming Pendant
Customization

10 Development of Application in C#
10.3 Development of Application for the FS100

Closing Process

Notify the programming pendant main software of the closing when the customized application is closed.

[Code example]

```csharp
public partial class Form1 : Form
{

    . . .

    private void button2_Click(object sender, EventArgs e)
    {
        // Close
        AppMsg.SendCloseCommand();
    }

    . . .
}
```

For the AppMessageWindow class, refer to its source code (AppMessageWindow.cs) which will be provided.

AppMessageWindow.cs includes the following codes.

• WM_COPYDATA message receiving from the programming pendant main software
• WM_COPYDATA message sending to the programming pendant main software
• Examples of important process such as Initializing or closing process.
10.4 Application Debugging

10.4.1 Parameter Setting

For the parameter set S2C533=1 for debugging the created applications.

NOTE
When releasing the debugging, do not fail to set S2C533=0 to return the parameter.

10.4.2 Debugging

Additional procedures are required as follows only when the versions except for Visual studio specified in Section 10.1.1 “Developing Environment (Developing PC Environment)” (including the versions of ServicePack) is installed. And this operation is necessary every time the power is turned ON.

<Additional procedures>

Copy the following files to the \Window\ folder in the programming pendant. Based on the rules, those files are stored in \Program Files\Common Files\Microsoft Shared\CoreCon\1.0\Target\wce400\<CPU>.

- Clientshutdown.exe
- ConmanClient2.exe
- CMaccept.exe
- eDbgTL.dll

1. Select {DEBUG} - {DEBUG OVERLAP}.

2. "for ExternalExe Debug" window appears.
3. Press [C# Debug] button. With this operation, the following processing will be executed and the pendant will be ready to connect a device.

- Expansion of the memory size
  (Enable to download the user-customized application)
- Execution of the device connection program
  (After execution, implement device connection within three minutes)

4. In Visual Studio on the desktop, press [Project] - [Property] to select [Debug] tab. On the window, also select “Debug” tab, and then, to “Program argument” section, input “argument:” which was indicated on the “for ExternalExe Debug”.

5. Press {Connect} button on “for EXternal Debug” window. It closes.


- When the following window is shown, debugging is ready.
7. Select {Menu} - {Debug} - {Start Debugging} to start the debugging operation.

8. If the following window is shown when debugging is started, select “OMRON NewPendant ARMv4i Release” and then press [Deploy] button.
11 Interface Specification with Programming Pendant Main Software

11.1 Command from the User Customized Application to Programming Pendant Main Software

11.1.1 Completion of Generation

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXE2YPP_CREATE_CMD</td>
<td>HWND hwnd : Window handle for message sending destination</td>
</tr>
</tbody>
</table>

.DESCRIPTION

The user-customized application sends this command to the programming pendant software when it is started up.

The sending window handle is the main frame handle of the user-customized application.

The programming pendant main software sends messages to this window while the user-customized application is running.

11.1.2 Closing of Application

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXE2YPP_CLOSE_CMD</td>
<td>Not exist</td>
</tr>
</tbody>
</table>

.DESCRIPTION

The user-customized application sends this command to the programming pendant software when it is closed.

After receiving this command, the software side will discard or release each object.
### Keymask

**<Description>**

With this command, keys sent to the DX100/DX200/FS100 can be limited from the user-customized application. When setting “1” to the objective key, it is sent to the DX100/DX200/FS100.

**<Key-Bit table>**

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>W0</td>
<td>X+/R+</td>
<td>ENTER</td>
<td>INSERT</td>
<td>FWD</td>
<td>SHIFT</td>
<td>RIGHT</td>
<td>w+</td>
</tr>
<tr>
<td>W1</td>
<td>Y-/B-</td>
<td>X-/R-</td>
<td>MODIFY</td>
<td>TEST</td>
<td>RUN</td>
<td>BWD</td>
<td>DELETE</td>
</tr>
<tr>
<td>W2</td>
<td>INTER LOCK</td>
<td>MOTION</td>
<td>TYPE</td>
<td>Z+/U+</td>
<td>USAGE</td>
<td>Y+/L+</td>
<td>X+/S+</td>
</tr>
<tr>
<td>W3</td>
<td>SHIFT</td>
<td>LEFT</td>
<td>AUX</td>
<td>Z-/U-</td>
<td>EX AXIS</td>
<td>ROBOT</td>
<td>Y-/L-</td>
</tr>
<tr>
<td>W4</td>
<td>.</td>
<td>0</td>
<td>7</td>
<td>ARROW DOWN</td>
<td>ARROW LEFT</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>W5</td>
<td>SERVO ON READY</td>
<td>5</td>
<td>-</td>
<td>3</td>
<td>8</td>
<td>FAST</td>
<td>SLOW</td>
</tr>
<tr>
<td>W6</td>
<td>9</td>
<td>SELECT</td>
<td>AREA</td>
<td>CANCEL</td>
<td>ASSIST</td>
<td>PAGE</td>
<td>DIRECT OPEN</td>
</tr>
<tr>
<td>W7</td>
<td>Spare</td>
<td>Spare</td>
<td>COORD</td>
<td>SIMPLE</td>
<td>MENU</td>
<td>MULTI</td>
<td>ARROW UP</td>
</tr>
</tbody>
</table>

**Ex.**

Set as follows when sending only [CANCEL] key to the DX100/DX200/FS100.

byte[8] = {0x00,0x00,0x00,0x00,0x00,0x00,0x10,0x00};

When there are no sending data, present specified key mask at the programming pendant software side is set.
11.1.4 Text Input Function Start

**Code name** | **Data configuration (Type)**
--- | ---
EXE2YPP_KEYPAD_CMD | DWORD type : key pad type (English/Japanese, etc.)
 | DWORD maxlen : limit of inputting character
 | SCHAR buf[128] : default character strings for keypad

<Description>
With this command, the character inputting key pad becomes available from the user-customized application.

11.1.5 Switching Control

**Code name** | **Data configuration (Type)**
--- | ---
EXE2YPP_CONTROL_CMD | DWORD target
 | 1: programming pendant main software
 | 2: user-customized application

<Description>
With this command, the control system can be alternated between the programming pendant main software and the user-customized application by sending data.

If it is switched to the programming pendant main software, the software side changes the display or releases the keymask, etc.

If it is switched to the user-customized application, the application side acquires the latest necessary information and displays it, and it should execute some initializing operations such as re-setting the keymask, etc.
11.1.6 Setting of Notifying Information

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXE2YPP_GET_INFOAUTO_CMD</td>
<td>DWORD notifyingfo Information to be notified</td>
</tr>
<tr>
<td></td>
<td>(specified by bit)</td>
</tr>
</tbody>
</table>

**Description**

With this command, setting of notifying information such as changes of the programming pendant software side status or information sent to the user-customized application when it is requested become available.

If it is not set, all the information is notified.

Bit is used for this setting.

- 0 bit: Group axis
- 1 bit: Motion coordinate system
- 2 bit: Manual speed
- 3 bit: Security
- 4 bit: Motion cycle
- 5 bit: Executing status
- 6 bit: Page shift
- 7 bit: Synchronizing status
- 8 bit: Battery
- 9 bit: Teach/play mode
- 10 bit: Multi window mode

11.1.7 Information Request

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXE2YPP_GET_INFO_CMD</td>
<td>Not exist</td>
</tr>
</tbody>
</table>

**Description**

With this command, request of the information to the programming pendant main software becomes available.

To this request, the software side sends the information previously set as sending information to the user-customized application.
11 Interface Specification with Programming Pendant Main Software

11.1 Command from the User Customized Application to Programming Pendant Main Software

11.1.8 Version Information Acquisition

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXE2YPP_GET_VERSION_CMD</td>
<td>Not exist</td>
</tr>
</tbody>
</table>

*Description*

With this command, request of the version to the programming pendant main software becomes available.

To this request, the software side sends the version which is under running to the user-customized application.
11.2 Command from the Programming Pendant Main Software to the User-Customized Application

11.2.1 Input Character

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_KEYPADRET_CMD</td>
<td>WORD okcancel 0=OK / 1=Cancel</td>
</tr>
<tr>
<td></td>
<td>WCHAR buf[128] Input character string</td>
</tr>
</tbody>
</table>

<Description>

The programming pendant main software sends this command after completion of inputting character strings keypad started by the user-customized application.

Receiving this command, the application side can acquire the input character strings.

11.2.2 Request of Control System Switching

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_CONTROL_CMD</td>
<td>DWORD target 1= Programming pendant main software</td>
</tr>
<tr>
<td></td>
<td>2= User-customized application</td>
</tr>
</tbody>
</table>

<Description>

With this command, the control system can be switched between the programming pendant main software and the user-customized application as specified by the sent data.

When switching to the programming pendant main software control system, the software side will change the indication or release the keymask after receiving this command.

When switching to the user-customized application control system, the application side should execute various initializing operations such as acquiring and indicating the necessary information, or resetting the keymask, etc. after receiving this command.
11 Interface Specification with Programming Pendant Main Software
11.2 Command from the Programming Pendant Main Software to the User-Customized Application

11.2.3 Status Information

<Description>
With this command, the programming pendant software side sends the status information when

– the status information which is previously-set as sending information is changed

or

– there is a information which is set as a sending information when requested by the user-customized application.

When the sending information is changed, “1” is set to “enable”. And a status value is set to “value”.

Followings are the element numbers of both “enable” and “value” arrangements.

<table>
<thead>
<tr>
<th>Code name Data configuration (Type)</th>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_STATUS_CMD</td>
<td>DWORD</td>
<td>valid information</td>
</tr>
<tr>
<td></td>
<td>DWORD</td>
<td>status value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Group axis</td>
</tr>
<tr>
<td>1</td>
<td>Motion coordinate system</td>
</tr>
<tr>
<td>2</td>
<td>Manual speed</td>
</tr>
<tr>
<td>3</td>
<td>Security</td>
</tr>
<tr>
<td>4</td>
<td>Motion cycle</td>
</tr>
<tr>
<td>5</td>
<td>Executing status</td>
</tr>
<tr>
<td>6</td>
<td>Page shift</td>
</tr>
<tr>
<td>7</td>
<td>Synchronizing status</td>
</tr>
<tr>
<td>8</td>
<td>Battery/Accessing to CF/Touch operation invalid</td>
</tr>
<tr>
<td>9</td>
<td>Teach/play mode</td>
</tr>
<tr>
<td>10</td>
<td>Multi window mode</td>
</tr>
<tr>
<td>11-15</td>
<td>(Unused)</td>
</tr>
</tbody>
</table>
Followings are the status value of each element.

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>R1</td>
</tr>
<tr>
<td>1</td>
<td>R2</td>
</tr>
<tr>
<td>2</td>
<td>R3</td>
</tr>
<tr>
<td>3</td>
<td>R4</td>
</tr>
<tr>
<td>4</td>
<td>R5</td>
</tr>
<tr>
<td>5</td>
<td>R6</td>
</tr>
<tr>
<td>6</td>
<td>R7</td>
</tr>
<tr>
<td>7</td>
<td>R8</td>
</tr>
<tr>
<td>8</td>
<td>B1</td>
</tr>
<tr>
<td>9</td>
<td>B2</td>
</tr>
<tr>
<td>10</td>
<td>B3</td>
</tr>
<tr>
<td>11</td>
<td>B4</td>
</tr>
<tr>
<td>12</td>
<td>B5</td>
</tr>
<tr>
<td>13</td>
<td>B6</td>
</tr>
<tr>
<td>14</td>
<td>B7</td>
</tr>
<tr>
<td>15</td>
<td>B8</td>
</tr>
<tr>
<td>16</td>
<td>S1</td>
</tr>
<tr>
<td>17</td>
<td>S2</td>
</tr>
<tr>
<td>18</td>
<td>S3</td>
</tr>
<tr>
<td>19</td>
<td>S4</td>
</tr>
<tr>
<td>20</td>
<td>S5</td>
</tr>
<tr>
<td>21</td>
<td>S6</td>
</tr>
<tr>
<td>22</td>
<td>S7</td>
</tr>
<tr>
<td>23</td>
<td>S8</td>
</tr>
<tr>
<td>24</td>
<td>S9</td>
</tr>
<tr>
<td>25</td>
<td>S10</td>
</tr>
<tr>
<td>26</td>
<td>S11</td>
</tr>
<tr>
<td>27</td>
<td>S12</td>
</tr>
<tr>
<td>28</td>
<td>S13</td>
</tr>
<tr>
<td>29</td>
<td>S14</td>
</tr>
<tr>
<td>30</td>
<td>S15</td>
</tr>
<tr>
<td>31</td>
<td>S16</td>
</tr>
<tr>
<td>32</td>
<td>S17</td>
</tr>
<tr>
<td>33</td>
<td>S18</td>
</tr>
<tr>
<td>34</td>
<td>S19</td>
</tr>
<tr>
<td>35</td>
<td>S20</td>
</tr>
<tr>
<td>36</td>
<td>S21</td>
</tr>
<tr>
<td>37</td>
<td>S22</td>
</tr>
<tr>
<td>38</td>
<td>S23</td>
</tr>
<tr>
<td>39</td>
<td>S24</td>
</tr>
</tbody>
</table>
11 Interface Specification with Programming Pendant Main Software
11.2 Command from the Programming Pendant Main Software to the User-Customized Application

1: Motion coordinate system

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Link coordinate</td>
</tr>
<tr>
<td>1</td>
<td>Cartesian coordinate</td>
</tr>
<tr>
<td>2</td>
<td>Cylindrical coordinate</td>
</tr>
<tr>
<td>3</td>
<td>Tool coordinate</td>
</tr>
<tr>
<td>4</td>
<td>User coordinate</td>
</tr>
<tr>
<td>5</td>
<td>External reference point coordinate</td>
</tr>
<tr>
<td>6</td>
<td>Teaching line coordinate</td>
</tr>
<tr>
<td>7</td>
<td>(Unused)</td>
</tr>
<tr>
<td>8</td>
<td>(Unused)</td>
</tr>
<tr>
<td>9</td>
<td>(Unused)</td>
</tr>
<tr>
<td>10</td>
<td>(Unused)</td>
</tr>
<tr>
<td>11-74</td>
<td>User coordinate (1-64)</td>
</tr>
</tbody>
</table>

2: Manual speed

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Inching</td>
</tr>
<tr>
<td>1</td>
<td>Low speed</td>
</tr>
<tr>
<td>2</td>
<td>Middle speed</td>
</tr>
<tr>
<td>3</td>
<td>High speed</td>
</tr>
</tbody>
</table>

3: Security

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Operation mode</td>
</tr>
<tr>
<td>1</td>
<td>Edit mode</td>
</tr>
<tr>
<td>2</td>
<td>Management mode</td>
</tr>
<tr>
<td>3</td>
<td>Safety mode</td>
</tr>
<tr>
<td></td>
<td>*Only for DX200</td>
</tr>
</tbody>
</table>

4: Motion cycle

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Continuous</td>
</tr>
<tr>
<td>1</td>
<td>1 cycle</td>
</tr>
<tr>
<td>2</td>
<td>Step</td>
</tr>
</tbody>
</table>
11 Interface Specification with Programming Pendant Main Software

11.2 Command from the Programming Pendant Main Software to the User-Customized Application

5: Executing status

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Alarming</td>
</tr>
<tr>
<td>1</td>
<td>Stopped by emergency stop</td>
</tr>
<tr>
<td>2</td>
<td>Hold</td>
</tr>
<tr>
<td>3</td>
<td>Playback</td>
</tr>
<tr>
<td>4</td>
<td>Playback</td>
</tr>
<tr>
<td>5</td>
<td>Stopped</td>
</tr>
</tbody>
</table>

6: Page switch

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Page does not switch</td>
</tr>
<tr>
<td>1</td>
<td>Page switches</td>
</tr>
</tbody>
</table>

7: Synchronized motion

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not synchronized</td>
</tr>
<tr>
<td>1</td>
<td>Synchronizing</td>
</tr>
</tbody>
</table>

8: Battery

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Normal</td>
</tr>
<tr>
<td>1</td>
<td>Exhausted</td>
</tr>
<tr>
<td>2</td>
<td>Accessing to CF</td>
</tr>
<tr>
<td>3</td>
<td>Touch operation invalid</td>
</tr>
<tr>
<td>4</td>
<td>Exhausted and touch operation invalid</td>
</tr>
</tbody>
</table>

9: Teach/play mode

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial status</td>
</tr>
<tr>
<td>1</td>
<td>Play mode</td>
</tr>
<tr>
<td>2</td>
<td>Teach mode</td>
</tr>
<tr>
<td>3</td>
<td>Play mode</td>
</tr>
<tr>
<td>4</td>
<td>Teach mode</td>
</tr>
<tr>
<td>5</td>
<td>Teach mode</td>
</tr>
</tbody>
</table>
11 Interface Specification with Programming Pendant Main Software

11.2 Command from the Programming Pendant Main Software to the User-Customized Application

10: Multi window indication mode

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Single mode</td>
</tr>
<tr>
<td>1</td>
<td>Multi mode</td>
</tr>
<tr>
<td>2</td>
<td>Multi mode</td>
</tr>
</tbody>
</table>

Example

When changing the mode from teach to play

```
enable = {0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.0}
value = {0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.0}
```

When changing to return the mode from play to teach

```
enable = {0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.0}
value = {0.0.0.0.0.0.0.0.0.2.0.0.0.0.0.0}
```

More than two “enable” s may become valid when several status change occurred.

In case no information exists, “0xffff” is set to “value”.

11.2.4 Alarm

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_ALARM_CMD</td>
<td>DWORD status</td>
</tr>
<tr>
<td></td>
<td>1 = alarming /</td>
</tr>
<tr>
<td></td>
<td>2 = alarm released</td>
</tr>
</tbody>
</table>

<Description>

The programming pendant main software sends this command when alarm is occurred or released.
11.2.5 Message

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_MESSAGE_CMD</td>
<td>DWORD msgnum struct{ WORD code WCHAR buf[128] }*msgnum</td>
</tr>
<tr>
<td></td>
<td>Numbers of message</td>
</tr>
<tr>
<td></td>
<td>Message code</td>
</tr>
<tr>
<td></td>
<td>Message</td>
</tr>
<tr>
<td></td>
<td>(volume of data for all messages)</td>
</tr>
</tbody>
</table>

<Description>
When the message is modified, the programming pendant software sends this command.
More than two messages are sent at a time when several messages are modified.

Meanings of the message code
1: Error message
2: Warning message
3: System status message

11.2.6 Language

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_LANGUAGE_CMD</td>
<td>DWORD language</td>
</tr>
<tr>
<td></td>
<td>language code</td>
</tr>
</tbody>
</table>

<Description>
When receiving a information request command (EXE2YPP_GET_INFO_cmd) or the language code is modified, the programming pendant software sends this command.

11.2.7 Version

<table>
<thead>
<tr>
<th>Code name</th>
<th>Data configuration (Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPPEXE_VERSION_CMD</td>
<td>WCHAR buf[16]</td>
</tr>
<tr>
<td></td>
<td>version information</td>
</tr>
</tbody>
</table>

<Description>
When the programming pendant software received a request for acquiring version information, it sends this command.
DX100/DX200/FS100 OPTIONS
INSTRUCTIONS
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  +46-480-414-10

YASKAWA Europe GmbH  Robotics Div.
Yaskawastrasse 1, 85391 Allershausen, Germany
Phone  +49-8166-80-0    Fax  +49-8166-80-103

YASKAWA Electric Korea Co., Ltd
9F, KYOBO Securities Bldg., 26-4, Yeoido-Dong Yeoungungpo-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. Boxuan 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, Udjoy Vihar, Phase- IV, Gurgaon, Haryana, India
Phone  +91-124-475-8500    Fax  +91-124-475-8542

Specifications are subject to change without notice
for ongoing product modifications and improvements.