Servo

MOTION CONTROLLER Basics (Real Mode:SFC)

This course is a training system for those who establish the motion control system using the motion CPU module of Mitsubishi motion controller Q series for the first time.
Introduction Purpose of the Course

This course is for those who are going to establish the motion control system using the motion CPU module of Mitsubishi motion controller Q series for the first time to acquire the procedures including installing the operating system, setting the system, programming and debugging in the motion SFC language by using the motion controller engineering environment MELSOFT MT Works2.

The main contents of this course are for a person in charge of software. The contents for a person in charge of hardware, such as the system design, installation, wiring and so on are prepared in the "SERVO MOTION CONTROLLER BASICS (HARDWARE)" course.

For this course, you are required to have knowledge about MELSEC-Q series PLC, AC servo and positioning control. For those who take this course for the first time, we recommend to take "MELSEC-Q SERIES BASICS" course, "MELSERVO (MR-J3) BASICS" course, "YOUR FIRST FACTORY AUTOMATION (POSITIONING CONTROL)" course.
Introduction

Course Structure

The contents of this course are as follows.
We recommend that you start from Chapter 1.

Chapter 5 - BASICS OF MOTION CONTROL
You will learn the basics of the motion control system.

Chapter 6 - SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE
You will learn how to select and install the operating system software of the motion CPU module.

Chapter 7 - PARAMETER SETTING
You will learn how to set the system of the motion CPU module and each parameter.

Chapter 8 - OPERATION CHECK
You will learn how to check the operation of the servo motor and execute the home position return.

Chapter 9 - PROGRAM DESIGN
You will learn how to design a program.

Chapter 10 - MOTION SFC PROGRAM
You will learn the basics of the motion SFC program for the motion control.

Chapter 11 - PROGRAMING
You will learn how to program and debug the motion SFC program with MT Developer2.

Final Test
Passing grade: 60% or higher.
<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to the next page</td>
<td>Go to the next page.</td>
</tr>
<tr>
<td>Back to the previous page</td>
<td>Back to the previous page.</td>
</tr>
<tr>
<td>Move to the desired page</td>
<td>&quot;Table of Contents&quot; will be displayed, enabling you to navigate to the desired page.</td>
</tr>
<tr>
<td>Exit the learning</td>
<td>Exit the learning. Window such as &quot;Contents&quot; screen and the learning will be closed.</td>
</tr>
</tbody>
</table>
Safety precautions

When you learn by using actual products, please carefully read the safety precautions in the corresponding manuals.

Precautions in this course

- The displayed screens of the software version that you use may differ from those in this course.

  This course is for the following software version:
  - MT Developer2 Version 1.18U
  - MR Configurator2 Version 1.01B
  - GX Works2 Version 1.55H

Reference

The following is the reference related to the learning. (You can learn without them.)
Click the reference name to download.

<table>
<thead>
<tr>
<th>Reference name</th>
<th>File type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample program</td>
<td>Compressed file</td>
<td>166.5 kB</td>
</tr>
<tr>
<td>Recording paper</td>
<td>Compressed file</td>
<td>5.57 kB</td>
</tr>
</tbody>
</table>
Chapter 5 BASICS OF MOTION CONTROL

The motion controller controls the multiple axes (servo motors) for a conveyor assembly, a processing machine etc. and performs the high-precision positioning control and speed control. In this course, the system building and the program development of the established motion control system are prepared for a person in charge of software.

The application examples of the motion control are introduced in the following. Click the button of the application example which you would like to see.
5.1 Development and Maintenance Environment of Motion Control System

For the development and maintenance environment of the motion control system, use the motion controller engineering environment MELSOFT MT Works2 and the servo setup software package MELSOFT MR Configurator2.

The following lists the main functions of each software.

- MELSOFT MT Works2
- MT Developer2

The development and maintenance environment of the motion control system

- Controlling the project
- Setting the system configuration
- Setting the servo data
- Testing the servo motor operation
- Creating a program in the motion SFC language
- Debugging and monitoring the program
- Writing or reading the program and the parameter
- Installing the operating system software

- MT Simulator2

The simulation environment of the motion SFC program

- MELSOFT MR Configurator2

The setup environment of the servo amplifier and the servomotor
- Setting the servo parameter
- The operation test and gain adjustment of the servo amplifier
5.2 Procedures to Establish the Motion Control System

The following shows the procedure to establish the motion control system. In this course, you will learn the process of the software design along with the establishment procedures.

### Hardware Design

1. SYSTEM DESIGN  
   MOTION CONTROLLER BASICS (HARDWARE) COURSE

2. INSTALLATION AND WIRING  
   MOTION CONTROLLER BASICS (HARDWARE) COURSE

3. WIRING CHECK  
   MOTION CONTROLLER BASICS (HARDWARE) COURSE

### Software Design

4. SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE  
   " Chapter 6 "

5. SYSTEM SETTING  
   " Chapter 7 "

6. OPERATION CHECK  
   " Chapter 8 "

7. PROGRAM DESIGN  
   " Chapter 9 "

8. PROGRAMMING  
   " Chapter 11 "

9. OPERATION

Learning range in this course
5.3 CONTROL FLOW

Check the control mode (control flow) in the sample system for this course by using the animation.

Operate the animation in the following sample system by a mouse according to the instruction of .

Playback speed of the animation
- Normal
- Slow
- Frame advance

Jump to the pointer "P1".

To arrange the next goods onto the pallet, the control flow returns back to the pointer (P1).
5.3.1 Equipment configuration of the sample system for this course

- Personal computer
  - USB cable
  - SSCNET III cable

- Output device
  - Forced stop button
  - Operation indicator lamp
  - Stop indicator lamp
  - Hand open command
  - Hand close command
  - Start button

- Input device

- Servo amplifier MR-J3-10B
  - Axis 1

- Servo motor HF-KP053
  - Axis 2
  - Upper stroke limit
  - Lower stroke limit
  - Proximity dog

*Equipment configuration of axis 2 and 3 are the same as the axis 1.*
## Equipment configuration of the sample system for this course

The following table lists the equipment configuration of the sample system.

<table>
<thead>
<tr>
<th>Item</th>
<th>Configuration component</th>
<th>Quantity</th>
<th>Model name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Motion controller system</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base unit</td>
<td></td>
<td>1</td>
<td>Q380B</td>
<td>A base unit which has 0 slots for mounting each module and supports multiple CPU</td>
</tr>
<tr>
<td>Power supply module</td>
<td></td>
<td>1</td>
<td>Q62P</td>
<td>Supplies power to each module.</td>
</tr>
<tr>
<td>PI C CPU module</td>
<td></td>
<td>1</td>
<td>QN11DPFPU</td>
<td>A CPU module which performs the sequence control</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>* The battery (UdbA1) is enclosed with the CPU module.</td>
</tr>
<tr>
<td>Motion CPU module</td>
<td></td>
<td>1</td>
<td>Q172DPCPU</td>
<td>A CPU module which performs the motion control</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>* The battery (Q6BAT) and the battery holder (Q170DBATC) are enclosed with the CPU module.</td>
</tr>
<tr>
<td>Input module</td>
<td></td>
<td>1</td>
<td>QX41</td>
<td>Inputs the ON/OFF signal from the start button (16 points)</td>
</tr>
<tr>
<td>Output module</td>
<td></td>
<td>1</td>
<td>QY40P</td>
<td>Outputs the ON/OFF signal to the indicator lamp and the device (the hand part) (16 points)</td>
</tr>
<tr>
<td>External power supply</td>
<td></td>
<td>1</td>
<td></td>
<td>Supplies 24VDC power to the I/O devices and the forced stop input</td>
</tr>
<tr>
<td>Start button</td>
<td></td>
<td>1</td>
<td></td>
<td>A push button switch to start the sample system</td>
</tr>
<tr>
<td>Forced stop button</td>
<td></td>
<td>1</td>
<td></td>
<td>A push button switch to stop the servo motors of all the axes at an emergency</td>
</tr>
<tr>
<td>Cable for forced stop input</td>
<td></td>
<td>1</td>
<td>Q170EMICEL_M</td>
<td>Used for wiring the forced stop input to the motion CPU module.</td>
</tr>
<tr>
<td>External I/O device</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hand part of the device</td>
<td></td>
<td>1</td>
<td></td>
<td>The hand part of the device for catching the goods</td>
</tr>
<tr>
<td>Indicator lamp</td>
<td></td>
<td>2</td>
<td></td>
<td>The indicator lamps to inform if the system is in operation or stops</td>
</tr>
<tr>
<td>Servo amplifier</td>
<td></td>
<td>3</td>
<td>MR-J3-10B</td>
<td>A servo amplifiers for 3 axes</td>
</tr>
<tr>
<td>Servo motor</td>
<td></td>
<td>2</td>
<td>HF-KT053</td>
<td>Servo motors for the axis 1 (X axis) and the axis 2 (Y axis)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>HF-KP053B</td>
<td>A servo motor with a brake for the axis 3 (Z axis)</td>
</tr>
<tr>
<td>Stroke limit</td>
<td></td>
<td>6</td>
<td></td>
<td>Sensors to detect the upper limit and the lower limit in the movable range of the device</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Equipment configuration of the sample system for this course

<table>
<thead>
<tr>
<th>Servo system</th>
<th></th>
<th></th>
<th>Sensors to detect the starting position of deceleration at the home position return</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proximity dog</td>
<td>3</td>
<td>–</td>
<td>A cable to conduct the power from the servo amplifier to the servomotor (Length: 2m)</td>
</tr>
<tr>
<td>Motor power supply cable</td>
<td>3</td>
<td>MR-PWS1CB2M-AI-L</td>
<td>A cable to connect the servo amplifier and the encoder of the servomotor (Length: 2m)</td>
</tr>
<tr>
<td>Encoder cable</td>
<td>3</td>
<td>MR-J3ENCBL2M-AI-L</td>
<td>A communication cable between the motion CPU module and the servo amplifier</td>
</tr>
<tr>
<td>SSCNET III cable</td>
<td>3</td>
<td>MR-J33US-M</td>
<td>A personal computer to run the engineering environment software</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Development environment</th>
<th></th>
<th></th>
<th>Software to set the motion CPU module, to program and so on</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personal computer</td>
<td>1</td>
<td>–</td>
<td>Software to set the PLC CPU module, to program and so on</td>
</tr>
<tr>
<td>Engineering environment software</td>
<td>1</td>
<td>MT Works2</td>
<td>Setup software to set the servo amplifier and the servomotor</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>GX Works2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>MR Configurator2</td>
<td></td>
</tr>
<tr>
<td>Operating system software</td>
<td>1</td>
<td>SwDNC-SV13GD</td>
<td>Software to be installed to the motion CPU module</td>
</tr>
<tr>
<td>HSR cable</td>
<td>1</td>
<td>MR-J3USBCBL3M</td>
<td></td>
</tr>
</tbody>
</table>
5.4 **Summary**

The following lists the contents you learned in Chapter 5. The following points are very important, so please check them again.

<table>
<thead>
<tr>
<th>Motion control overview</th>
<th>The motion controller controls the multiple axes (servo motors) for a conveyor assembly, a processing machine etc. and performs the high-precision positioning control and speed control.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development and maintenance environment of motion control system</td>
<td>For the development and maintenance environment of the motion control system, use the motion controller engineering environment MELSOFT MT Works2 and the servo setup software package MELSOFT MR Configurator2.</td>
</tr>
</tbody>
</table>
In Chapter 6, you will learn how to select and install the operating system software of the motion CPU module.

**Learning procedure of Chapter 6**

6.1 Type and Selection of the Operating System Software
6.2 Installing the Operating System Software
### Type and Selection of the Operating System Software

Select a motion CPU module and install the operating system software (control software) according to the application of a conveyor assembly, a processing machine and so on.

There are the following 3 types of the operating system software according to the application.

In the sample system, select and install SV13 which is for a conveyor assembly.

<table>
<thead>
<tr>
<th>Item</th>
<th>Conveyor assembly use (SV13)</th>
<th>Automatic machinery use (SV22)</th>
<th>Machine tool peripheral use (SV43)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td><img src="image.conveyor" alt="Conveyor assembly" /></td>
<td><img src="image.food" alt="Food processing" /></td>
<td><img src="image.grinding" alt="Grinding machine" /></td>
</tr>
<tr>
<td>Equipment example</td>
<td>Electronic components assembly equipment, conveying equipment, paint applicator, chip mounting, wafer slicer, loader and unloader, bonding machine, X-Y table</td>
<td>Food packaging machine, food processing machine, winding machine, spinning machine, textile machine, printing machine, book binder, press feeder, tire molder</td>
<td>Grinding machine, transfer machine, woodworker, loader and unloader</td>
</tr>
<tr>
<td>Positioning program</td>
<td>Dedicated language supporting the motion SFC</td>
<td>Mechanical support language supporting the motion SFC</td>
<td>EIA language (G code)</td>
</tr>
<tr>
<td>Dedicated language</td>
<td>Control method by the programming language suitable to the motion control such as the positioning control and other</td>
<td>Method performing the synchronous control only by writing the configuration of the mechanical system</td>
<td>Method of using the normalized (coded) numerical value (00 to 101) which specifies the control function of the axis in the NC device</td>
</tr>
</tbody>
</table>

### Precautions

- The operating system software is not installed at the time of the motion CPU module purchase. Please install the software following the procedures on the next screen.
- The operating system software is sold separately. Purchase the operating system software with the motion CPU module.
6.2 Installing the Operating System Software

Install the operating system software to the motion CPU module. Follow the procedure below.

① Turn off the motion controller.
   Switch the RUN/STOP switch of the motion CPU module to STOP.
   Connect a personal computer and a PLC CPU module by a USB cable.

② Change the rotary switch for the function select of the motion CPU module to "Installation mode".
   (Switch for function select 1: "A", Switch for function select 2: "0")

③ Turn on the motion controller.
   LED display becomes "INS" (Installation mode).

④ Start the MT Developer2 and set the transfer setup.
   (Install the USB driver as necessary.)

⑤ Set the CD-ROM of the operating system software to the personal computer and execute the installation from the MT Developer2.
   After the installation, turn off the motion controller.

⑥ Change the rotary switch for the function select.
   (Switch for function select 1: "0", Switch for function select 2: "0")

⑦ Turn on the motion controller.
   LED display becomes "AL" (Motion error).
   * "AL" is displayed because the parameter is not set at this moment, but it is not a problem.
6.3 Summary

The following lists the contents you learned in Chapter 6. The following points are very important, so please check them again.

| Type and selection of the operating system software | · Select a motion CPU module and install the operating system software (control software) according to the application of a conveyor assembly, processing machine and so on.  
  Conveyor assembly use (SV13)  
  Automatic machinery (SV22)  
  Machine tool peripheral use (SV43)  
  · The operating system software is not installed at the time of the motion CPU module purchase.  
  · The operating system software is sold separately.  
  Purchase the operating system software with the motion CPU module. |
| Select and installing the operating system software | · Before the installation, change the rotary function select switch of the motion CPU module to the installation mode. (Switch for function select 1: "A", Switch for function select 2: "0")  
  After the installation, return the rotary function select 1 switch to "0" the rotary function select 2 switch to "0".  
  · Execute the installation by the installation function of MT Developer2. |
Chapter 7  SYSTEM SETTING

In Chapter 7, you will learn how to set the system of the motion CPU module and each parameter.

SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE  ................................................. "Chapter 6"

SYSTEM SETTING  ........................................................................................................... "Chapter 7"

OPERATION CHECK  ........................................................................................................ "Chapter 8"

PROGRAM DESIGN  ................................................. "Chapter 9" (BASICS OF MOTION SFC: Chapter 10)

PROGRAMMING  ................................................. "Chapter 11"

OPERATION

Learning procedure of Chapter 7
7.1 Transfer Setup
7.2 Creating a Project
7.3 System Setting
  7.3.1 Basic system setting
  7.3.2 System configuration setting
  7.3.3 SSCNET configuration setting
7.4 Servo Data Setting
  7.4.1 Fixed parameter setting
  7.4.2 Home position return data setting
  7.4.3 JOG operation data setting
7.5 Servo Parameter Setting
7.6 Parameter Block Setting
7.7 Saving a Project
7.8 Writing Parameters to the Motion CPU Module
7.1 Enabling Communications between the Personal Computer and the Motion CPU Module

Before setting parameters, enable communications between the personal computer to which MT Developer2 is installed and the motion CPU module, and apply setting data to the motion CPU module.

Setting procedure
- Connect the personal computer and the PLC CPU module by a USB cable.
- Set the transfer setup with MT Developer2.
The screen of transfer setup and the operation are same as those of GX Works2.

Point of the transfer setup
Since the communication target motion CPU module is mounted to the CPU slot 2 of the base unit, select PLC No.2 in the transfer setup.

Select PLC No.2.
7.2 Creating a Project

After completion of the transfer setup, create a new project. A project is a unit that is used to control various parameters and programs by MT Developer2.

Set the following to create a project.
Select the motion CPU module type and the operation system software type.

**CPU Type**
Select the model name of the motion CPU module to be used. In this course, select "Q172D".

**OS Type**
Select the model name of the operation system software installed to the motion CPU module. In this course, select "SW8-SV13QD" (SV13).
7.3 **System Setting**

After creating a project, set the **system first**.
Set the motion CPU module and servo according to the actual system configuration.

7.3.1 **Basic system setting**

First, set the **Basic Setting**. (After creating a project, a dialog box appears.)
The basic system setting includes such as base unit, multiple CPU, etc.
In this course, set the parameters in **Base Setting**. (For the other settings, use the default values.)

**Main Base**
Select the number of slots of the main base unit to be used.
In this course, select "8 Slots" since Q38DE is selected for the base unit.

**Extension Base**
Select whether to use the extension base unit or not and the number of slots of the expansion base unit.
In this course, select "Nothing" for all extension base stages.
System configuration setting

Next, set the module configuration used for the main base unit and the extension base unit. Assign the motion module, I/O module, and other modules which are controlled by the motion CPU module to empty slots of the base unit.

In the sample system, assign an input module and an output module to the main base unit.

<table>
<thead>
<tr>
<th>Slot No.</th>
<th>Module model name</th>
<th>I/O Type</th>
<th>Points</th>
<th>First I/O No.</th>
<th>High-Speed Read Setting</th>
<th>I/O Response Time Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slot 1</td>
<td>QY40P</td>
<td>Output</td>
<td>16</td>
<td>0000</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Slot 2</td>
<td>QX40</td>
<td>Input</td>
<td>16</td>
<td>0010</td>
<td>Not used</td>
<td>10ms</td>
</tr>
</tbody>
</table>

Let's set a system configuration in the next screen.
7.3.2 System configuration setting

The system configuration setting is completed. Click to go to the next screen.
7.3.3 SSCNET structure setting

Next, set the servo amplifier configuration used for the system. Assign a servo amplifier connected to the motion CPU module with SSCNET III cable according to each control axis number.

In the sample system, assign three servo amplifiers to three control axis numbers (d01 to d03).

<table>
<thead>
<tr>
<th>Control axis number at the servo amplifier side</th>
<th>Axis No.</th>
<th>Amplifier Type</th>
<th>External Signal Input Type</th>
<th>Allowable Travel during Power-Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>d01</td>
<td>1</td>
<td>MR-J3 (W) - B</td>
<td>Amplifier Input Valid (Input Filter Setting: 3.5ms)</td>
<td>10 Revolution</td>
</tr>
<tr>
<td>d02</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>d03</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Precautions**

*Axis No.* set in the SSCNET III structure differs from the control axis number set using a rotary switch on a servo amplifier. The *Axis No.* set here is used to specify a control axis from the program.

Let’s set the SSCNET III structure in the next screen.
The SSCNET configuration setting is completed.
Click to go to the next screen.
7.4 Servo Data Setting

Next, set the servo data. Set the necessary data for positioning control to each axis which are set in the SSCNET configuration. Servo data are classified into the following three categories.

<table>
<thead>
<tr>
<th>Classification</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Parameter</td>
<td>Refer to the section 7.4.1.</td>
</tr>
<tr>
<td>Home Position Return Data</td>
<td>Set the data necessary for executing the home position return. Home position return is a function that moves a machine to the home position and matches the home position addresses of the machine and the motion CPU module at the position.</td>
</tr>
<tr>
<td>JOG Operation Data</td>
<td>Set the data necessary for executing the JOG operation. JOG operation is a function that operates a servo motor manually in the forward or reverse rotation direction at the constant speed. It is used for the teaching or test operation when a system is when a system is installed.</td>
</tr>
</tbody>
</table>

7.4.1 Fixed parameter setting

Set the characteristic value necessary for machine operation of the system. Set the data and movement range of the machine to convert the command value of "address (travel value) and speed" which is called electric gear into the pulse unit.

In the sample system, set the following fixed parameters to axes 1 to 3.

<table>
<thead>
<tr>
<th>Parameter item</th>
<th>Set value of axes 1 to 3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Setting</td>
<td>0:mm</td>
<td>In the sample system, “mm” unit is used.</td>
</tr>
<tr>
<td>Number of Pulses per Revolution</td>
<td>262144[PLS]</td>
<td>Usually, set the resolution value of the servo motor to be used.</td>
</tr>
<tr>
<td>Travel Value per Revolution</td>
<td>100000.0[μm]</td>
<td>Ball screws (lead: 10mm) are used for the machine.</td>
</tr>
<tr>
<td>Upper Stroke Limit</td>
<td>200000.0[μm]</td>
<td>Set the movement range of the machine to prevent an overrun.</td>
</tr>
<tr>
<td>Lower Stroke Limit</td>
<td>-100000.0[μm]</td>
<td></td>
</tr>
</tbody>
</table>

Let’s set the fixed parameters in the next screen.
### Fixed parameter setting

The fixed parameter setting of axis 1 is completed.

Click ![Next](button) to go to the next screen.
### Home position return data setting

Set the data necessary for executing the home position return. Home position return is a function that moves a machine to the home position and matches the home position addresses of the machine and the motion CPU module at the position.

In the sample system, set the following home position return data to axes 1 to 3.

<table>
<thead>
<tr>
<th>Parameter Item</th>
<th>Set value of axes 1 to 3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPR Direction</td>
<td>0: Reverse Direction</td>
<td></td>
</tr>
<tr>
<td>HPR Method</td>
<td>0: Proximity Dog Type 1</td>
<td>In the sample system, use &quot;Proximity Dog Type 1&quot;.</td>
</tr>
<tr>
<td>Home Position Address</td>
<td>0.0[μm]</td>
<td></td>
</tr>
<tr>
<td>HPR Speed</td>
<td>20000.00[mm/min]</td>
<td></td>
</tr>
<tr>
<td>Creep Speed</td>
<td>100.00[mm/min]</td>
<td></td>
</tr>
<tr>
<td>Travel Value after Proximity Dog ON</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter Block Setting</td>
<td>1</td>
<td>For details, refer to Parameter Block Setting.</td>
</tr>
<tr>
<td>HPR Retry Function</td>
<td>0: Invalid</td>
<td></td>
</tr>
<tr>
<td>Dwell Time at the HPR Retry</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Home Position Shift Amount</td>
<td>0.0[μm]</td>
<td></td>
</tr>
<tr>
<td>Speed Set at Home Position Shift</td>
<td>0: HPR Speed</td>
<td></td>
</tr>
<tr>
<td>Torque Limit Value at Creep Speed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operation for HPR Incompletion</td>
<td>1: Not Execute Servo Program</td>
<td></td>
</tr>
</tbody>
</table>

Let’s set the home position return data in the next screen.
7.4.2 Home position return data setting

The home position return data setting of Axis 1 is completed.

Click [next] to go to the next screen.
7.4.3 JOG operation data setting

Set the data necessary for executing the JOG operation.
JOG operation is a function that operates a servomotor manually in the forward or reverse rotation direction at the constant speed.
It is used for the teaching or test operation when a system is constructed.
In the sample system, set the following JOG operation data to axes 1 to 3.

<table>
<thead>
<tr>
<th>Parameter item</th>
<th>Set value of axes 1 to 3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOG Operation Data</td>
<td>JOG Speed Limit Value</td>
<td>15000.00[mm/min]</td>
</tr>
<tr>
<td></td>
<td>Parameter Block Setting</td>
<td>0</td>
</tr>
</tbody>
</table>

Let’s set the home position return data in the next screen.
7.4.3 JOG operation data setting

The JOG operation data setting of axis 1 and the servo data setting of axes 1 to 3 are completed.

Click to go to the next screen.
7.5 Servo Parameter Setting

Next, set the servo specific parameters to each axis. For the setting of servo parameter, servo setup software MELSOFT MR Configurator2 is needed separately. Download and install MR Configurator2 before setting parameters.

In the sample system, set the following servo parameters to axes 1 to 3.

<table>
<thead>
<tr>
<th>Parameter item</th>
<th>Set value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation direction selection</td>
<td>CCW dir. during fwd. pls input, CW dir. during rev pls. input</td>
</tr>
<tr>
<td>Servo forced stop selection</td>
<td>Invalid (Not use forced stop input (EM1) )</td>
</tr>
<tr>
<td>Absolute position detection system</td>
<td>Used in incremental system</td>
</tr>
<tr>
<td>Home position set condition selection</td>
<td>Z-phase must. not be passed.</td>
</tr>
<tr>
<td>In-position range</td>
<td>100 [PLS]</td>
</tr>
</tbody>
</table>

* For parameters which are not used in this course, use the default values.

* Let's set the servo parameter in the next screen.
MR Configurator2 exits.
The servo parameter setting is completed.
Click ▶️ to go to the next screen.
### 7.6 Parameter Block Setting

Set acceleration/deceleration parameters for each control pattern. Up to 64 acceleration/deceleration patterns can be created. Set an arbitrary parameter block No. to each control pattern in positioning control.

In the sample system, set the following parameters to the blocks No.1 and No. 2.

<table>
<thead>
<tr>
<th>Parameter item</th>
<th>Block No. 1</th>
<th>Block No. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control pattern</td>
<td>For positioning control and home position return</td>
<td>For JOG operation</td>
</tr>
<tr>
<td>Interpolation Control Unit</td>
<td>0.0 mm</td>
<td>0.0 mm</td>
</tr>
<tr>
<td>Speed Limit Value</td>
<td>60000.00[mm/min]</td>
<td>15000.00[mm/min]</td>
</tr>
<tr>
<td>Acceleration Time</td>
<td>500[ms]</td>
<td>300[ms]</td>
</tr>
<tr>
<td>Deceleration Time</td>
<td>500[ms]</td>
<td>300[ms]</td>
</tr>
<tr>
<td>Rapid Stop Deceleration Time</td>
<td>100[ms]</td>
<td>100[ms]</td>
</tr>
<tr>
<td>S-curve Ratio</td>
<td>100[%]</td>
<td>100[%]</td>
</tr>
<tr>
<td>Torque Limit Value</td>
<td>300[%]</td>
<td>300[%]</td>
</tr>
<tr>
<td>Deceleration Process on STOP</td>
<td>0: Deceleration Stop</td>
<td>0: Deceleration Stop</td>
</tr>
<tr>
<td>Allowable Error Range for Circular Interpolation</td>
<td>10.0[um]</td>
<td>10.0[um]</td>
</tr>
<tr>
<td>Acceleration/Deceleration System</td>
<td>0: Trapezoid/ S-curve</td>
<td>0: Trapezoid/ S-curve</td>
</tr>
</tbody>
</table>

Let's set the parameter block setting in the next screen.
7.6 Parameter Block Setting

Set the data such as the acceleration/deceleration function control used for each parameter block.

- **Parameter Block**
  - Interpolation Control Limit
  - Speed Limit Value: 0.0000 [mm/min], 10000.00 [mm/min], 200000 [mm/sec]
  - Acceleration Time: 0.00 [ms], 300 [ms], 3000 [ms]
  - Deceleration Time: 0.00 [ms], 300 [ms], 3000 [ms]
  - Rapid Stop Deceleration Time: 0.00 [ms], 300 [ms], 3000 [ms]
  - S-curve Ratio: 100 [%], 100 [%], 0 [%]

- **Torque Limit Value**
  - Deceleration Process on STOP: 0 [Deceleration Stop], 0 [Deceleration Stop], 0 [Deceleration Stop], 0 [Deceleration Stop]
  - Allowable Error Range for Circular Interpolation: 0.0 [μm], 0.1 [μm], 0.1 [μm], 0.1 [μm]
  - Bias Speed at Start: 0.00 [mm/min], 0.00 [mm/min], 0.00 [mm/sec], 0.00 [mm/sec]

- **Advanced S-curve Acceleration/Deceleration**
  - Acceleration/Deceleration 1: 0 [Trapezoid/S-curve], 0 [Trapezoid/S-curve], 0 [Trapezoid/S-curve], 0 [Trapezoid/S-curve]
  - Acceleration 1 Ratio: 0 [%], 100 [%], 100 [%], 0 [%]

Set the data of advanced S-curve acceleration/deceleration, which performs the acceleration/deceleration process by converting the speed smoothly.

The settings of parameter block No.1 and 2 are completed.
Click ➡️ to go to the next screen.
7.7 Saving a Project

Save a project including parameters after the parameter setting.
If you exit MT Developer2 without saving a project, the set parameters will be discarded.
If you save a new project, set the following project information.
It is recommended to give a name which you can easily recognize the contents of the project (control contents, system name, etc.).

![Save As dialog box]

- **Save Folder Path** *Compulsory*
  Specify a folder to create a workspace.

- **Workspace/Project List**
  If one or more workspaces exist in the save folder path, they are displayed in list.
  Double-clicking a workspace name displays a list of projects.

- **Workspace Name** *Compulsory*
  Specify a workspace name. (up to 128 characters)

- **Project Name** *Compulsory*
  Specify a project name. (up to 128 characters)

- **Title**
  Specify a title. (up to 128 characters)
  Use here when you want to give a name which exceeds 128 characters. (Inputting title is not necessary.)
7.8 Writing Parameters to the Motion CPU Module

After saving the project, write parameters to the motion CPU module.
Before writing, check the following.

The power supplies of the motion controller and the servo amplifier are on.
The RUN/STOP switch of the motion CPU module is switched to STOP.
The personal computer and the PLC CPU module are connected correctly.

Check parameters in the Write to CPU screen and perform writing.
The following lists the contents you learned in Chapter 7.
The following points are very important, so please check them again.

| Transfer setup | Before setting parameters, enable communications between the personal computer and the motion CPU module.  
<table>
<thead>
<tr>
<th></th>
<th>Since the communication target motion CPU module is mounted to the CPU slot 2 of the base unit; select PLC No.2 in the transfer setup.</th>
</tr>
</thead>
</table>
| Project        | A project is a unit that is used to control various parameters and programs by MT Developer2.  
|                | Set an operating system type and the model name of a motion CPU module to be used for creating a project.                       |
| Basic system setting | The basic system setting includes such as base unit, multiple CPU, etc.                                                  |
| System configuration | Set the module configuration used for the main base unit and the extension base unit.  
|                  | Assign the motion module, I/O module, and other modules which are controlled by the motion CPU module to empty slots of the base unit. |
| SSCNET configuration | Assign a servo amplifier connected to the motion CPU module with SSCNET III cable according to each control axis number.  
|                   | Axis No. set in the SSCNET configuration differs from the control axis number set using a rotary switch on a servo amplifier.  
|                   | The axis No. is used to specify a control axis from the program.                                                          |
| Fixed parameter | Set the characteristic value necessary for machine operation of the system.  
|                 | Set the data and movement range of the machine to convert the command value of “address (travel value) and speed” which is called electric gear into the pulse unit. |
| Home position return data | Set the data necessary for executing the home position return.  
|                          | Home position return is a function that moves the machine to the home position and matches the home positions between the machine and the motion CPU module at the position. |
| JOG operation data | Set the data necessary for executing the JOG operation.  
|                   | JOG operation is a function that operates a servo motor manually in the forward or reverse rotation direction at the constant speed.  
|                   | It is used for the teaching or test operation when a system is constructed.                                               |
| Servo parameter | Set the servo specific parameters to each axis.  
|                 | For the setting of servo parameter, servo setup software MELSOFT MR Configurator2 is needed separately.                     |
| Parameter block | Set the acceleration/deceleration process for each control pattern. Up to 64 acceleration/deceleration patterns can be created.  
|                 | Specify an arbitrary parameter block No. to each control pattern in positioning control.                                    |
| Saving a project | Save a project including parameters after the parameter setting.                                                        |
## 7.9 Summary

<table>
<thead>
<tr>
<th>Saving a project</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Save a project including parameters after the parameter setting.</td>
<td></td>
</tr>
<tr>
<td>• If MELSOFT MT Developer2 exits without saving a project, the set parameter contents will be discarded.</td>
<td></td>
</tr>
<tr>
<td>• Give a name which the contents of the project (control contents, system name, etc.) are easily recognized.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Writing parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Write parameters to the motion CPU module. Before writing, check the following.</td>
<td></td>
</tr>
<tr>
<td>• The power supplies of the motion controller and the servo amplifier are on.</td>
<td></td>
</tr>
<tr>
<td>• The RUN/STOP switch of the motion CPU module is STOP.</td>
<td></td>
</tr>
<tr>
<td>• A personal computer and the PLC CPU module are connected correctly.</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 8  OPERATION CHECK

In Chapter 8, you will learn how to check the operation of a servo motor and how to perform the home position return. When turning on a servo amplifier and a servomotor for the first time, before installing the servomotor to a machine, make sure to check the operation to prevent an accident such as machine damage due to malfunction such as miswiring or incorrect parameter settings.

SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE ........................................... "Chapter 6"

SYSTEM SETTING ........................................... "Chapter 7"

OPERATION CHECK ........................................... "Chapter 8"

PROGRAM DESIGN ........................................ "Chapter 9"(BASICS OF MOTION SFC: Chapter 10)

PROGRAMMING ........................................ "Chapter 11"

OPERATION

**Learning procedure of Chapter 8**
8.1 Checking the Operation of a Servo motor
8.2 Connection of a Servo motor with a Machine
8.3 Performing the Home Position Return
8.1 Checking the Operation of a Servo motor

Check the status of a servo amplifier (for error), rotating direction of a servo motor, operation of the upper and lower stroke limits, and stopping accuracy of the home position return, by using the test function of MT Developer2.

The following shows the list of the test functions used in this course.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servo on and off</td>
<td>Outputs the servo on or servo off command to all or desired axes of servo motors.</td>
</tr>
<tr>
<td>Initial check</td>
<td>Displays the status of a servo amplifier. If an error exists, the error code and the error name can be checked.</td>
</tr>
<tr>
<td>Upper and lower LS check</td>
<td>Performs JOG operation in a forward or reverse rotation to check if the upper or lower stroke limit operates normally.</td>
</tr>
<tr>
<td>JOG operation</td>
<td>Performs JOG operation of a connected motor. Before performing JOG operation, make sure to set the JOG operation data and set the data in the parameter blocks to be used.</td>
</tr>
<tr>
<td>Home position return test</td>
<td>Performs the home position return to check for an error between a stop position and a machine home position.</td>
</tr>
</tbody>
</table>

Let's check operation using the test function in the next screen.
8.1 Checking the Operation of a Servo motor

**Test Mode Function**

The test mode supports the initial check at a system start. From the tool button, choose the function you want to perform.

*Starting procedure outline*

**Test Mode**

[Program Start]

- Check whether the servo motor runs in accordance with the servo program written to the motion controller.
- Perform operation with PLC ready (M2000) OFF.

Next

**Debug Mode**

Motion SFC program debugging is supported.

- By turning ON PLC ready (M2000), the motion controller is placed in the ordinary operation mode and starts the SFC program running.
- Debug operation is supported on the monitor screen of the program editor function.

Next

**Program Start**

The operation check of the servomotor is completed.

Click Play and go to the next screen.
8.2 Connection of a Servo motor with a Machine

Next, install a machine to the rotational axis of a servo motor.

Before installation, check the operation of a servo motor without a machine in order to prevent machine damage due to malfunction of a servo system.

After completing the installation of a machine, check the normal operation of both a servo motor and a machine by using JOG operation again.
8.3 Performing the Home Position Return

After connecting a servo motor with a machine, check the normal operation of the home position return. The home position return is an operation to match a home position stored in a motion CPU module to a machine home position. The mismatched home positions cause an error of a stop position. To prevent the error, perform the home position return test to confirm there is no error between a stop position and a machine home position.

Let's check the operation using the home position return test function in the next screen.
Performing the Home Position Return

Test Mode Function

The test mode supports the initial check at a system start. From the tool button, choose the function you want to perform.

<Starting procedure outline>

Test Mode

[Servo Start]
Check whether the motion controller and servo amplifier are connected properly, and check the servomotor runs properly.
- Perform operation with PLC ready (M2000) OFF.

Debug Mode Motion SFC program debugging is supported.
- By turning ON PLC ready (M2000), the motion controller is placed in the ordinary operation mode and starts the SFC program running.
- Debug operation is supported on the monitor screen of the program editor function.

Program Start

The home position return test has been completed.
Click "Next" and go to the next screen.
The following lists the contents you learned in Chapter 8. The following points are very important, so please check them again.

<table>
<thead>
<tr>
<th>Operation check of servo motor</th>
<th>Check the status of a servo amplifier, rotating direction of a servo motor, operation of the upper and lower stroke limits by using the test function of MT Developer2.</th>
</tr>
</thead>
</table>
| Connection of a Servo motor with a Machine | * Before installation, check the operation of a servomotor without a machine in order to prevent machine damage due to malfunction of a servo system.  
* After completing the installation of a machine, check the normal operation of both a servo motor and a machine by using JOG operation again. |
| Operation check of the home position return | After connecting a servo motor with a machine, check the normal operation of the home position return.  
After the home position return is executed in the home position return test, confirm there is no error between a stop position and a machine home position. |
Chapter 9 PROGRAM DESIGN

In Chapter 9, you will learn how to design a program necessary for motion control.

SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE  "Chapter 6"

SYSTEM SETTING  "Chapter 7"

OPERATION CHECK  "Chapter 8"

PROGRAM DESIGN  "Chapter 9" (BASICS OF MOTION SFC: Chapter 10)

PROGRAMMING  "Chapter 11"

OPERATION

Learning procedure of Chapter 9
9.1 Programming Language for Motion Control
9.2 Flowcharting the Sequence of Control
9.3 Creating the Correspondence Table of I/O Devices and Device Nos.
9.4 Designing a Servo Program
  9.4.1 Servo instruction
  9.4.2 Positioning data
9.5 Creating a Servo Program
Programming Language for Motion Control

The following three types of programming languages enable motion control.

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence program</td>
<td>A motion SFC program is started with the motion dedicated sequence instruction &quot;D(P).SFCS&quot;. * When &quot;Auto.&quot; is set to &quot;Yes&quot; in the parameter setting, a sequence program for start is unnecessary. * A specified servo program can be directly started with the motion dedicated sequence instruction &quot;D(P).SVST&quot;</td>
</tr>
<tr>
<td>Motion SFC program</td>
<td>The sequence of motion control is written in a flowchart-like format. In positioning control, a servo program is executed with the motion control step.</td>
</tr>
<tr>
<td>Servo program</td>
<td>The pattern of positioning control is written with servo instructions.</td>
</tr>
</tbody>
</table>

The following figure shows the relationship among the sequence program, motion SFC program, and servo program.
Motion SFC language is a programming language similar to a flowchart. Expressing the sequence of control in a flowchart makes designing a motion SFC program easier. The following shows the flowchart of the control of the sample system.

Put mouse pointer over the flowchart to display the details of each control.
Flowcharting the Sequence of Control

1. Axis 2 start
2. Home position return for axis 3
3. Axis 3 start
4. Start acceptance OFF
5. Positioning control
6. Start acceptance OFF
7. Start flag ON
8. Lamp ON/OFF
9. Start acceptance OFF
10. Positioning control
11. Start acceptance OFF
12. Start flag OFF
13. Hand open and close command
14. Open command
15. Stop for 0.5 second
16. Positioning control
17. Start acceptance OFF
9.2 Flowcharting the Sequence of Control

Start signal

Hand close command ON

Close command

Stop for 0.5 seconds

Positioning control

Start acceptance OFF

Number of arranged goods count

Number of arranged goods

Number of arranged goods reset

Lamp ON/OFF

Jump to P1
Creating the Correspondence Table of I/O Devices and Device Nos.

Next, create the correspondence table of I/O devices and device Nos. to be used in the sample system. Creating a correspondence table reduces programming glitches and streamlines your programming.

As an example, the following table shows the correspondence of I/O devices and device Nos in the sample system.

<table>
<thead>
<tr>
<th>I/O device name</th>
<th>Device No.</th>
<th>Input or output</th>
<th>Type</th>
<th>Data type</th>
<th>Range</th>
<th>Initial value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start button</td>
<td>PX12</td>
<td>Input</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>A push button switch to start the system.</td>
</tr>
<tr>
<td>Hand open command</td>
<td>PY0</td>
<td>Output</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>off</td>
<td>Output for controlling opening and closing the hand part of the device</td>
</tr>
<tr>
<td>Hand close command</td>
<td>PY1</td>
<td>Output</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operation indicator lamp</td>
<td>PY2</td>
<td>Output</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>The lamp illuminates during system operation.</td>
</tr>
<tr>
<td>Stop indicator lamp</td>
<td>PY3</td>
<td>Output</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>The lamp illuminates during system stop.</td>
</tr>
<tr>
<td>Devices used in program</td>
<td>D2000</td>
<td>—</td>
<td>Word</td>
<td>16-bit integer</td>
<td>0〜500</td>
<td>0</td>
<td>The travel value of the X-axis (axis 1) of the device is stored.</td>
</tr>
<tr>
<td></td>
<td>D2002</td>
<td>—</td>
<td>Word</td>
<td>16-bit integer</td>
<td>0〜1,100</td>
<td>0</td>
<td>The travel value of the Y-axis (axis 2) of the device is stored.</td>
</tr>
<tr>
<td></td>
<td>D2100</td>
<td>—</td>
<td>Word</td>
<td>16-bit integer</td>
<td>0〜6</td>
<td>0</td>
<td>The number of goods arranged on the pallet is stored.</td>
</tr>
<tr>
<td></td>
<td>M7100</td>
<td>—</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>Bit data to be output to the hand open command (PY0) is stored.</td>
</tr>
<tr>
<td></td>
<td>M7101</td>
<td>—</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>Bit data to be output to the hand close command (PY1) is stored.</td>
</tr>
<tr>
<td></td>
<td>M8001</td>
<td>—</td>
<td>Bit</td>
<td>—</td>
<td>—</td>
<td>OFF</td>
<td>Bit data input from the start button (PX12) is stored.</td>
</tr>
</tbody>
</table>
9.4 Designing a Servo Program

Next, design a servo program. A servo program is a programmed pattern of positioning control. The program consists of servo instructions, axis Nos., addresses (travel value), command speed, and acceleration pattern, etc.

Register the pattern of positioning control as a servo program beforehand. In positioning control with a Motion SFC program, a specified servo program No. is executed according to the control pattern.

Programs are executed in descending order.

Servo program No. [K 22]
Servo instruction CPSTART3

| Axis | 1 |
| Axis | 2 |
| Axis | 3 |

Controlled axis No.

Speed

1 INC-3

Axis 1, 0.0 μm
Axis 2, 0.0 μm
Axis 3, 250000.0 μm

Command speed (Vector-speed) 20000.00 mm/min
Addresses of each axis (travel value)

2 INC<

Axis 2, 50000.0 μm
Axis 3, 50000.0 μm

You can specify parameters other than an address and speed, depending on the servo instruction type.

3 INC-3

Axis 1, 0.0 μm
Axis 2, D 2002 μm
Axis 3, 0.0 μm

You can specify an address (travel value) by using a word device.
9.4.1 Servo instruction

Next, you will learn servo instructions used in a servo program.
Many servo instructions, such as simple linear positioning with one axis and linear and circular interpolations with two or more axes, are prepared. In this course, the six servo instructions used in the sample system is explained.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
</table>
| INC-3  | Incremental 3-axes linear interpolation | - For three specified axes, 3-axes linear interpolation control is performed in the range of travel value specified for each axis, starting from a current stop position.  
- The movement directions of each axis depend on the sign (+ or -) of the travel value specified for each axis.  
  * When the travel value is positive: Positioning in forward direction (The address increases.)  
  * When the travel value is negative: Positioning in reverse direction (The address decreases.) |

Example: Positioning is performed as follows when the movement distances are 10000 for X-axis direction, 5000 for Y-axis direction, and 6000 for Z-axis direction.
## 9.4.1 Servo instruction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
</table>
| ABS-3  | Absolute 3-axes linear interpolation         | • By the absolute method, 3-axes linear interpolation control is performed for specified three axes in the range from a current stop position (X1, Y1, Z1) to a specified position (X2, Y2, Z2).
• The movement directions of each axis depend on their stop addresses and specified addresses. |

※□ indicates the data to be set.

![Diagram 1](image1.png)  
Example:
Positioning is performed as follows when the current stop address is (1000, 2000, 1000) and the positioning address is (4000, 8000, 4000).
### 9.4.1 Servo instruction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
</table>
| CPSTART3     | Constant-speed control  | - Constant-speed control of 3-axes linear interpolation control or 2-axis circular interpolation control is performed. Once started, constant-speed control is performed for a specified axis, with passing a preset pass point, until constant-speed control end (CPEND). In interpolation control, a vector speed is used as a command speed.  
- As shown in the figures below, the control for changing to the specified speed at the pass point differs depending on ON/OFF of "CP completion point specification flag (M2040)" at the time when the constant-speed control is started. |

**CP completion point specification flag: OFF**  
......Specifies the point where the speed change is started.  

The speed is started to change at the pass point.

![Graph showing speed change at pass point](image1)

**CP completion point specification flag: ON**  
 ......Specifies the point where the speed change is completed.

The speed is completely changed at the pass point.

![Graph showing complete speed change at pass point](image2)
### 9.4.1 Servo instruction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
</table>
| INC    | Incremental auxiliary point-specified circular interpolation | - By the incremental method, 2-axis circular interpolation control is performed from a current stop position (start point) to a specified relative end address (X1, Y1) with an auxiliary address (pass point) (X2,Y2) passed.  
- The center of the arc is the point of intersection of the perpendicular bisectors of the start point (current stop) to the auxiliary point, and the auxiliary point to the end point. |

![Diagram of servo instruction](image)
## 9.4.1 Servo instruction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
</table>
| CHGA   | Current value change | - The current value of the specified axis is changed.  
- Only the current value of a stopped axis can be changed.  
- A current value change for an operating axis causes a minor error 101.  
- A current value change is performed according to the following procedure.  
  1. Turn on "Start accept flag" corresponding to the specified axes.  
  2. Change the current values of the specified axes to the specified addresses.  
  3. Turn off "Start accept flag" on completion of the current value change.  
- Start accept flag: M200n (n: axis No.) |

![Diagram](image.png)
## 9.4.1 Servo instruction

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Instruction name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZERO</td>
<td>Home position return</td>
<td>- Perform the home position return when a machine home position needs to be checked, such as when a power supply is turned on.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Several methods are prepared for the home position return. Select a suitable method for system configuration or application.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Specify a home position return method in the servo data setting screen.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- In the sample system, &quot;Proximity dog type 1&quot; is used.</td>
</tr>
</tbody>
</table>

![Graph showing speed, creep speed, and zero-point signal over time.](image)

In the sample system, "Proximity dog type 1" is used.
9.4.2 Designing the servo program of the sample system

Goods arrangement points
The following figure and table show the positioning control pattern of the sample system and the servo instructions used for positioning control at each point.

![Diagram of goods arrangement points]

<table>
<thead>
<tr>
<th>No.</th>
<th>Servo Instruction</th>
<th>Movement range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ZERO Servo instruction</td>
<td>-</td>
<td>Home position return of axis 1 (X-axis)</td>
</tr>
<tr>
<td>2</td>
<td>ZERO Servo instruction</td>
<td>-</td>
<td>Home position return of axis 2 (Y-axis)</td>
</tr>
<tr>
<td>3</td>
<td>ZERO Servo instruction</td>
<td>-</td>
<td>Home position return of axis 3 (Z-axis)</td>
</tr>
<tr>
<td>10</td>
<td>INC-3 Incremental 3-axes linear interpolation</td>
<td>Home position -&gt; A</td>
<td>The device is moved from the retract position to the offset position.</td>
</tr>
<tr>
<td>11</td>
<td>CHGA Current value change</td>
<td>-</td>
<td>The current value of the offset position is changed to &quot;0&quot;.</td>
</tr>
<tr>
<td>12</td>
<td>INC-3 Incremental 3-axes linear interpolation</td>
<td>A -&gt; B</td>
<td>The hand of the device (Z-axis) is lowered.</td>
</tr>
<tr>
<td>21</td>
<td>INC-3 Incremental 3-axes linear interpolation</td>
<td>A -&gt; B</td>
<td>Constant-speed control is started.</td>
</tr>
<tr>
<td></td>
<td>CPSTART3 3-axis constant-speed control start</td>
<td>-</td>
<td>The hand of the device (Z-axis) is lifted.</td>
</tr>
<tr>
<td>1</td>
<td>INC-3 Incremental 3-axes linear interpolation</td>
<td>B -&gt; C</td>
<td>The device is moved to the arrangement position on the pallet.</td>
</tr>
<tr>
<td>2</td>
<td>INC Incremental radius-specified circular interpolation less than 180°</td>
<td>C -&gt; D</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>INC-3 Incremental 3-axes linear interpolation</td>
<td>D -&gt; E</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>INC Incremental radius-specified circular interpolation less than 180°</td>
<td>E -&gt; F</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>INC Incremental 3-axes linear interpolation</td>
<td>F -&gt; G</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>INC Incremental radius-specified circular interpolation less than 180°</td>
<td>G -&gt; H</td>
<td></td>
</tr>
</tbody>
</table>
### 9.4.2 Designing the servo program of the sample system

<table>
<thead>
<tr>
<th>Step</th>
<th>Instruction</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>INC-3</td>
<td>H→I</td>
<td>The hand of the device (Z-axis) is lowered.</td>
</tr>
<tr>
<td></td>
<td>CPEND</td>
<td></td>
<td>Constant-speed control is completed.</td>
</tr>
<tr>
<td></td>
<td>CPSTART3</td>
<td></td>
<td>Constant-speed control is started.</td>
</tr>
<tr>
<td>23</td>
<td>INC-3</td>
<td>I→J</td>
<td>The hand of the device (Z-axis) is lifted.</td>
</tr>
<tr>
<td>2</td>
<td>ABS-3</td>
<td>J→A</td>
<td>The device is moved to the retract position.</td>
</tr>
<tr>
<td></td>
<td>CPEND</td>
<td></td>
<td>Constant-speed control is completed.</td>
</tr>
</tbody>
</table>
9.4.2 Designing the servo program of the sample system
Next, with MT Developer2, create the servo program you have designed.
In the sample system, you will create the following ten servo programs.
Let's create the servo programs in the next screen.

<table>
<thead>
<tr>
<th>No.</th>
<th>Servo Instruction</th>
<th>Movement range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ZERO</td>
<td></td>
<td>Home position return for axis 1 (X-axis)</td>
</tr>
<tr>
<td>2</td>
<td>ZERO</td>
<td></td>
<td>Home position return for axis 2 (Y-axis)</td>
</tr>
<tr>
<td>3</td>
<td>ZERO</td>
<td></td>
<td>Home position return for axis 3 (Z-axis)</td>
</tr>
<tr>
<td>10</td>
<td>INC-3</td>
<td>Home position A</td>
<td>The device is moved from the start position to the offset position.</td>
</tr>
<tr>
<td>11</td>
<td>CHGA</td>
<td></td>
<td>The current value of offset position of the axis 1 (X-axis) is changed to &quot;0mm&quot;.</td>
</tr>
<tr>
<td>12</td>
<td>CHGA</td>
<td></td>
<td>The current value of offset position of the axis 2 (Y-axis) is changed to &quot;0mm&quot;.</td>
</tr>
<tr>
<td>13</td>
<td>CHGA</td>
<td></td>
<td>The current value of offset position of the axis 3 (Z-axis) is changed to &quot;0mm&quot;.</td>
</tr>
<tr>
<td>21</td>
<td>INC-3</td>
<td>A → B</td>
<td>The hand of the device (Z-axis) is lowered.</td>
</tr>
<tr>
<td>22</td>
<td>CPS/START3</td>
<td></td>
<td>Constant-speed control is started.</td>
</tr>
<tr>
<td>1</td>
<td>INC-3</td>
<td>B → C</td>
<td>The hand of the device (Z-axis) is lifted.</td>
</tr>
<tr>
<td>2</td>
<td>INC-3</td>
<td>C → D</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>INC-3</td>
<td>D → E</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>INC-3</td>
<td>E → F</td>
<td>The device is moved to the arrangement position on the pallet.</td>
</tr>
<tr>
<td>5</td>
<td>INC-3</td>
<td>F → G</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>INC-3</td>
<td>G → H</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>INC-3</td>
<td>H → I</td>
<td>The hand of the device (Z-axis) is lowered.</td>
</tr>
<tr>
<td>8</td>
<td>CPE/END</td>
<td></td>
<td>Constant-speed control is completed.</td>
</tr>
</tbody>
</table>
# Creating a Servo Program

<table>
<thead>
<tr>
<th>CPEND</th>
<th>Constant-speed control end</th>
<th>Constant-speed control is completed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSTART3</td>
<td>3-axis constant-speed control start</td>
<td>Constant-speed control is started.</td>
</tr>
<tr>
<td>1 INC-3</td>
<td>Incremental 3-axes linear interpolation</td>
<td>I → J</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The hand of the device (Z-axis) is lifted.</td>
</tr>
<tr>
<td>2 ABS-3</td>
<td>Absolute 3-axes linear interpolation</td>
<td>J → A</td>
</tr>
<tr>
<td>CPEND</td>
<td>Constant-speed control end</td>
<td>Constant-speed control is completed.</td>
</tr>
</tbody>
</table>
Creating a Servo Program

The servo programs have been created.

Click and go to the next screen.
## Summary

The following lists the contents you learned in Chapter 9.
The following points are very important, so please check them again.

<table>
<thead>
<tr>
<th>Flowcharting the Sequence of Control</th>
<th>Motion SFC language is a programming language similar to a flowchart. Exressing the sequence of control in a flowchart facilitates the design of a motion SFC program.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servo program</td>
<td>• A servo program is a programmed pattern of positioning control. The program consists of servo instructions, axis Nos., addresses (travel value), movement speed, and acceleration pattern, etc. • In positioning control with a motion SFC program, specified servo programs are executed according to the control pattern.</td>
</tr>
<tr>
<td>Servo instruction</td>
<td>Instructions to command positioning control. Many servo instructions, such as simple linear positioning with one axis and linear and circular interpolations with two or more axes, are prepared.</td>
</tr>
</tbody>
</table>
Chapter 10 MOTION SFC PROGRAM

In Chapter 10, you will learn the basics of the motion SFC program. In the end of the chapter, you will design a motion SFC program in the control procedure (flow chart) of the sample system.

Learning procedure of Chapter 10
10.1 Features of the Motion SFC Program
10.2 Configuration Component of the Motion SFC Program
10.3 Available Device Type
10.4 Operation Sequence of the Motion SFC Program
10.5 Creating a Motion SFC Program of the Sample System
10.6 Start Method of the Motion SFC Program
### 10.1 Features of the Motion SFC Program

**Motion SFC program**

Motion SFC program is a program similar to the flow chart to diagrammatize a flow of a programming process. It is also easy to handle for a person who learns the motion control programming for the first time. The following shows the features of the motion SFC program.

<table>
<thead>
<tr>
<th>Point</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program which can be used by anyone</td>
<td>By associating each operation of the machine with each operation step, a program which anyone can easily understand can be created using the flow chart format. Therefore, the maintainability is improved.</td>
</tr>
<tr>
<td>Not affected by the scan time</td>
<td>Since the motion CPU module determines the transition condition and starts positioning, there is no variation in the response time affected by the scan time at the PLC CPU module side.</td>
</tr>
<tr>
<td>Shortened tact time</td>
<td>The motion CPU module can process not only the positioning control but also the numerical operations, SET or RST of device, etc. This makes operations via a PLC CPU module unnecessary and shortens the tact time.</td>
</tr>
<tr>
<td>Transition condition description specific to the motion SFC</td>
<td>The transition condition description specific to the motion SFC allows a command be given to the servo amplifier after the start condition is satisfied. In addition, a transition to the next step can be performed after a positioning is started, without waiting for completion.</td>
</tr>
</tbody>
</table>

**Supplement**
The motion controller can be controlled from the sequence program, by using the motion dedicated sequence instruction. For details, refer to the manuals.

- **F**: Operation control step
  - Executes the arithmetic operation and the I/O control.
- **G**: Transition (condition wait)
  - Judges the transition condition.
- **K**: Motion control step
  - Executes the positioning control and the speed control of the servo motor.
10.2 Configuration Component of the Motion SFC Program

The configuration components of the motion SFC program (SFC chart symbols) are explained by taking a sample program as an example.

Pointing the mouse cursor on the figure of each configuration component displays how to use these components.

Program example: Main program

- **Main**
- **F1**
- **G1**
- **IFB1**
- **P1**
- **G2**
- **F0**
- **Subroutine**
- **G3**
- **IFE1**
- **P1**

Program example: Subroutine program

- **Subroutine**
- **K1**
- **Q4**
- **PAB1**
- **F3**
- **F2**
- **PAE1**
- **F4**
- **END**
## Available Device Type

You can use the following devices in the motion SFC program.

<table>
<thead>
<tr>
<th>Device</th>
<th>Symbol</th>
<th>Number of points</th>
<th>Read</th>
<th>Write</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>X</td>
<td>8192 points</td>
<td>○</td>
<td>○</td>
<td>Note) A motion CPU module cannot access the I/O module with &quot;X&quot; and &quot;Y&quot;. Use &quot;PX&quot; and &quot;PY&quot; instead.</td>
</tr>
<tr>
<td>Output</td>
<td>Y</td>
<td></td>
<td>○</td>
<td>○</td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>PX</td>
<td>256 points</td>
<td>○</td>
<td>×</td>
<td>Devices for the I/O modules controlled by the motion CPU module Use &quot;PX&quot; and &quot;PY&quot; when accessing the I/O module.</td>
</tr>
<tr>
<td>Output</td>
<td>DY</td>
<td></td>
<td>○</td>
<td>○</td>
<td></td>
</tr>
<tr>
<td>Internal relay</td>
<td>M</td>
<td>12288 points</td>
<td>○</td>
<td>○</td>
<td>This device can be used within the range of M0 to M8191.</td>
</tr>
<tr>
<td>Link relay</td>
<td>B</td>
<td>8192 points</td>
<td>○</td>
<td>○</td>
<td>—</td>
</tr>
<tr>
<td>Annunciator</td>
<td>F</td>
<td>2048 points</td>
<td>○</td>
<td>○</td>
<td>—</td>
</tr>
<tr>
<td>Special relay</td>
<td>SM</td>
<td>2256 points</td>
<td>○</td>
<td>○</td>
<td>—</td>
</tr>
<tr>
<td><strong>Word</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data register</td>
<td>D</td>
<td>8192 points</td>
<td>○</td>
<td>○</td>
<td>This device can be used within the range of D0 to D8191.</td>
</tr>
<tr>
<td>Link register</td>
<td>W</td>
<td>8192 points</td>
<td>○</td>
<td>○</td>
<td>—</td>
</tr>
<tr>
<td>Special register</td>
<td>SD</td>
<td>2256 points</td>
<td>○</td>
<td>○</td>
<td></td>
</tr>
<tr>
<td>Motion register</td>
<td>#</td>
<td>12288 points</td>
<td>○</td>
<td>○</td>
<td>Use #8000 to #8639 as monitor devices and #8640 to #8735 as motion error history devices.</td>
</tr>
</tbody>
</table>

### Multiple CPU shared device

<table>
<thead>
<tr>
<th>CPU</th>
<th>Symbol</th>
<th>Number of points</th>
<th>Read</th>
<th>Write</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self CPU</td>
<td>U G</td>
<td>Max. 14336 points*</td>
<td>○</td>
<td>○</td>
<td>You can share the device range assigned in the multiple CPU setting between the CPU modules and also access the devices controlled with the PLC CPU module.</td>
</tr>
<tr>
<td>Other CPU</td>
<td></td>
<td></td>
<td>○</td>
<td>×</td>
<td>* The available points differ depending on the system setting.</td>
</tr>
</tbody>
</table>

* The available points differ depending on the system setting.
10.3 Available Device Type

Positioning dedicated device
This is a device which can access the status of the motion CPU module and each axis. It uses a part of the range in the internal relay (M) and the data register (D). For details, click the button below. You can browse a list of the devices in PDF format.

In the sample system, use the following positioning dedicated devices.

<table>
<thead>
<tr>
<th>Device No.</th>
<th>Application</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2042</td>
<td>Set all the axes to the servo-on status.</td>
<td></td>
</tr>
<tr>
<td>M2415</td>
<td>Used to check the servo-on status for the axis 1</td>
<td>The device is turned on in the servo-on status.</td>
</tr>
<tr>
<td>M2435</td>
<td>Used to check the servo-on status for the axis 2</td>
<td></td>
</tr>
<tr>
<td>M2001</td>
<td>Used to check the start acceptance status for the axis 1</td>
<td>The device is turned on when the servo is operating.</td>
</tr>
<tr>
<td>M2002</td>
<td>Used to check the start acceptance status for the axis 2</td>
<td></td>
</tr>
<tr>
<td>M2003</td>
<td>Used to check the start acceptance status for the axis 3</td>
<td></td>
</tr>
</tbody>
</table>

Motion register
This is a device which can access the monitor value and the error history of each axis. 
"#" is used for the device symbol. (In the sample system, the motion resister is not used.) For details, click the button below. You can browse a list of the devices in PDF format.

List of motion resisters
10.4 Operation Sequence of the Motion SFC Program

The basic operation sequence of the motion SFC program starts from the "START" symbol and finishes with the "END" symbol. If the program shifts to the transition during the operation, it does not shift to the next step until the conditions are satisfied. (Waiting for the conditions to be satisfied)

Also, the operation sequence changes if condition branches, jumps and calls of subroutine are included.

Let's check the operation sequence by taking a sample program as an example. Click the "PLAY" button to play the animation.

The process starts from the position of the pointer. This example program repeats processing within the range between the pointer and the jump (loop process).
## 10.5 Creating a Motion SFC Program of the Sample System

Create a motion SFC program in the control procedure (flow chart) of the sample system. The following shows motion SFC programs to be created.

<table>
<thead>
<tr>
<th>No.</th>
<th>Program name</th>
<th>Program description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Main</td>
<td>This is a main program to be executed from the sequence program for start. This program executes each subroutine of &quot;Servo-on&quot;, &quot;Retract position movement&quot;, &quot;Home position return&quot; when the system starts. After completion of the above subroutines, the program waits for the input of the start signal (PY12). When the start signal (PY12) is turned on, the &quot;Positioning 1&quot; subroutine is executed and the goods move onto the pallet. Until the number of arranged goods on the pallet becomes 6, the subroutine is executed repeatedly.</td>
</tr>
<tr>
<td>11</td>
<td>Servo-on</td>
<td>This is a subroutine program executed when the system starts from the main program. It sets all the axes to the servo-on status.</td>
</tr>
<tr>
<td>12</td>
<td>Retract position</td>
<td>movement</td>
</tr>
<tr>
<td>13</td>
<td>Home position</td>
<td>return</td>
</tr>
<tr>
<td>14</td>
<td>Positioning 1</td>
<td></td>
</tr>
</tbody>
</table>

**Point**

Writing all the processes to a single program makes the program complicated and difficult to read when creating a program. Dividing the program according to the control contents (creating subroutines), calling and executing them from the main program makes the program simple and the program becomes easy to read.

In addition, the programming efficiency improves since there is no need to write the same processing details many times.
10.5 Creating a Motion SFC Program of the Sample System

The following shows the motion SFC programs in the sample system. Pointing the mouse cursor to the figure of each configuration component displays each processing detail.
10.6 Start Method of the Motion SFC Program

The following two methods are available for starting the motion SFC program.

Automatically start when the motion CPU module is executed
By turning on the PLC READY flag (M2000) of the motion CPU module, any motion SFC program can be automatically started. The sequence program to start the motion SFC program is unnecessary. In the default setting, setting the STOP/RUN switch of the motion CPU module to RUN turns on M2000.

Set the program to be started automatically by the motion SFC parameter of MT Developer2. Set "Auto." item of the program to "Yes". (The default setting is "No".)

The program whose "Auto." item is set to "Yes" automatically starts. The set value can be changed in the dialog box displayed by double-clicking each program.
10.6 Start Method of the Motion SFC Program

Start by using the motion dedicated sequence instruction from the sequence program.
Any motion SFC program can be started by the motion SFC start request "D(P).SFCS" command from the sequence program.
The motion control system coupled with the sequence control can be established.

<table>
<thead>
<tr>
<th>Setting data</th>
<th>Setting details</th>
<th>Set by</th>
<th>Data type</th>
</tr>
</thead>
</table>
| (n1)         | (First I/O No. of the target CPU)/16  
Value to specify actually is the following.  
CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H  
(Note): Motion CPU cannot be set as CPU No.1 in the Multiple CPU configuration. | User | 16-bit binary |
| (n2)         | Motion SFC program No. to start. | User | 16-bit binary |
| (D1) (Note-1) | Completed devices  
(D1=0): Device which make turn on for one scan at accept completion of instruction.  
(D1=1): Device which make turn on for one scan at accept abnormal completion of instruction.  
(On completion of the error, D1 + 0 is turned on as well.) | System | bit |
| (D2) (Note-3) | Complete status storage Device | System | 16-bit binary |

Note-1: Omission possible with both of (D1) and (D2) omission.
10.7 Summary

The following lists the contents you learned in Chapter 10. The following points are very important, so please check them again.

<table>
<thead>
<tr>
<th>Motion SFC program</th>
<th>This is a program similar to the flow chart for the motion control. It is also easy to handle for a person who learns the motion control programming for the first time.</th>
</tr>
</thead>
</table>
| Configuration      | The motion SFC program is described by arranging and connecting the configuration component (SFC chart symbols).  
Component of the motion SFC | - START  
  Starts a program.  
- Operation control step  
  Executes the operation control program.  
- Motion control step  
  Executes the servo program (positioning control).  
- Subroutine call/start step  
  Executes the subroutine program (other motion SFC programs).  
- Shift transition  
  Without waiting for completion of the previous step, shifts to the next step when the transition conditions are satisfied.  
- WAIT transition  
  After waiting for completion of the previous step, shifts to the next step when the transition conditions are satisfied.  
- Jump and pointer  
  Moves a process to the specified pointer position.  
- END  
  Finishes a program. |
| Operation sequence  | The basic operation sequence starts from "START" and finishes with "END". If a program shifts to the transition, it does not shift to the next step until the conditions are satisfied. Besides that, the sequence changes if the branch, subroutine, and others are executed. |
| of the motion SFC   |                                                                                                                                                                                                  |
| Positioning dedicated device | This is a device which can access the status of the motion CPU module and each axis. A part of the range in the internal relay (M) and the data register (D) is assigned. |
| Motion register     | With the motion CPU-dedicated device (Symbol #:), a motion controller can access the monitor values and error histories of each axis. |
Chapter 11 PROGRAMMING

In Chapter 11, you will learn the method of programming and debugging the motion SFC program of the sample system with MT Developer2.

SELECTING AND INSTALLING THE OPERATING SYSTEM SOFTWARE

SYSTEM SETTING

OPERATION CHECK

PROGRAM DESIGN

PROGRAMMING

OPERATION

Learning procedure of Chapter 11

11.1 Creating a Motion SFC Program
11.2 Converting a Program into Practicable Format
11.3 Creating a Sequence Program for Start
11.4 Debugging a Motion SFC Program
11.4.1 Debugging without using a motion CPU module
11.4.2 Debugging the program of the sample system
11.5 Writing a Motion SFC Program
11.6 Executing the Motion SFC Program
11.6 Completion of the Sample System
11.1 Creating a Motion SFC Program

Program the motion SFC program of the sample system with MT Developer2. You will learn the basic operation such as selection, arrangement, alignment of figures and connection and disconnection of cables through programming.

Let’s operate the programming of the motion SFC in the next screen.
11.1 Creating a Motion SFC Program

The motion SFC program is created.

Click and go to the next screen.
11.2 Converting a Program into Practicable Format

After creating a program, convert it into a practicable format for the motion CPU module. Programs which are not converted cannot be executed or stored.

Click the **Batch Conversion** on the toolbar.
* From the menu, [Check/Convert] - [Batch Conversion] is also possible.

**Output**
The conversion results of the programs are displayed.
11.3 Creating a Sequence Program for Start

Start the motion SFC program of the sample system with the motion dedicated PLC instruction "D (P).SFCS" from the sequence program for start.

The following shows the sequence program for start of the sample system. The motion SFC program No. 10 (Main) of the motion CPU module (No. 2) is started when M0 turns on.

* SM403 in the program is the special relay which turns off only at the first scan after the PLC CPU module has started.

Create the sequence program with GX Works2. (You cannot create sequence programs with MT Works2.) Write the created programs to the PLC CPU module using Write to PLC of GX Works2.
11.4 Debugging a Motion SFC Program

After the completion of the programming, check that the program operates according to the design. We call the cause of incorrect operation (fault point) Bug and research and correcting works Debug.

Debug is an important work. Do not execute programs in the actual system without debugging. If bugs are left, they may cause abnormal stop, malfunctions or troubles.

The functions to support debug are equipped in MT Developer2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>A function to simulate the execution of the program without a motion CPU module. Use this in an environment where a motion CPU module cannot be provided for debug.</td>
</tr>
<tr>
<td>Monitor</td>
<td>A function to monitor the execution status and the status of the each device. There are various monitor functions such as the function to monitor only registered devices, to monitor the motion SFC program in execution.</td>
</tr>
<tr>
<td>Device test</td>
<td>A function to perform a test for checking the operations of created programs by turning on/off bit devices, writing the data of word device. Debug can be performed without connecting the I/O devices.</td>
</tr>
</tbody>
</table>

Precautions
Perform debugging without connecting I/O devices or machine to the motion controller system or servo motors. Unintended operations may be caused by bugs.
11.4.1 Debugging without using a motion CPU module

Use a Simulation function when a motion CPU module cannot be provided for debug. The operation of the program can be simulated on the virtual motion CPU module on the software.

<table>
<thead>
<tr>
<th>Item</th>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch</td>
<td>RUN</td>
<td>Executes a virtual motion CPU module.</td>
</tr>
<tr>
<td></td>
<td>STOP</td>
<td>Stops a virtual motion CPU module. (initial status)</td>
</tr>
<tr>
<td></td>
<td>RESET</td>
<td>Resets a virtual motion CPU module. (This can be selected only during STOP.)</td>
</tr>
<tr>
<td>LED</td>
<td></td>
<td>Displays the status of the motion CPU module or errors in 7-segment LED.</td>
</tr>
</tbody>
</table>

Precautions
- There is no guarantee that the motion SFC program operates according to the simulation after debug.
- Input or output of I/O modules are executed by using the memory for simulation.
- Therefore, the operation result of the simulation function may differ from the operation result of the actual motion CPU module.
11.4.2 Debugging the program of the sample system

Debug the motion SFC program of the sample system with the simulation function.
First, shift an execution status of the program to **debug mode**.

In the debug mode, up to 4 stop positions of the program (which is called **break point**) can be specified.
The program stops when it shifts to the step specified as a break point. (This situation is called **During break**.)
During break, programs can be executed with the operation results or operations checked in one step at a time
by using the following functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request or cancel of debug mode</td>
<td>Shifts the execution of the program to the debug mode or releases the debug mode. When the program is shifted to the debug mode, the break function below can be used.</td>
</tr>
<tr>
<td>Execute or continue</td>
<td>Executes again or continues the program when the motion SFC program is during break or forced end.</td>
</tr>
<tr>
<td>One step execution</td>
<td>Shifts the motion SFC program during break from the break point to next step.</td>
</tr>
<tr>
<td>Forced shift</td>
<td>Shifts the program to the next step forcibly when it does not shift to the next step at the transition because the conditions are not satisfied.</td>
</tr>
<tr>
<td>Break</td>
<td>Ends the motion SFC program in execution or during break forcibly regardless of the break point.</td>
</tr>
<tr>
<td>Forced end</td>
<td>Ends the motion SFC program in execution or during break forcibly.</td>
</tr>
</tbody>
</table>

Let's operate the debug of the motion SFC program in the next screen.
11.4.2 Debugging the program of the sample system

IFBl

[G 3]
EXIT // Check start button is ON

IFB2

[G 2]
!M8001
// Check the rise of the start button

[F 6]
SET M8001 // Start rising note
SET PY2 // In open
RST PY3 // Inactive

The simulation function is finished.
Debugging of the motion SFC program is completed.

Click [ ] and go to the next screen.
11.5 Writing a Motion SFC Program

Write a created motion SFC program to the motion CPU module. Before writing, check the following:
- Power supplies of a motion controller and servo amplifier are turned on.
- The RUN/STOP switch of the motion CPU module is STOP.
- A personal computer and the PLC CPU module are connected correctly.

Click the Parameter + Program button in the following Write to CPU window and perform writing.

All parameters and programs are selected.

Execute writing

The motion CPU module information can be checked.
11.6 Executing the Motion SFC Program

Execute the motion SFC program wrote to the motion CPU module.
Operate the switches of the PLC CPU module and the motion CPU module in the following procedures.

1) Reset the PLC CPU module and the motion CPU module.
   Set the RESET/STOP/RUN switch of PLC CPU to RESET.
   Reset is performed by the No. 1 PLC CPU module.
   All CPU modules, including the motion CPU module are reset.

2) Check error generation.

3) Execute the program.
   Set the RESET/STOP/RUN switch of the PLC CPU module and the STOP/RUN switch of the motion CPU module to RUN.
Finally, check the operation of completed sample system using the animation.

Operate the animation in the following sample system by a mouse according to the instruction of the animation.

Playback speed of the animation
- Normal
- Slow
- Frame advance

No. 14: Positioning 1 Subroutine program

【END】
The subroutine program ends and the flow returns to the main program.
The following lists the contents you learned in Chapter 11. The following points are very important, so please check them again.

<table>
<thead>
<tr>
<th>Convert a program</th>
<th>After creating a program, convert it into a practicable format for the motion CPU module. Programs which are not converted cannot be executed or stored.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug</td>
<td>After the completion of the programming, check that the program operates according to design. • We call the cause of incorrect operation (fault point) Bug and research and correcting works Debug.</td>
</tr>
<tr>
<td></td>
<td>• Do not execute programs in the actual system without debugging. If bugs are left, they may cause abnormal stop, malfunctions or troubles.</td>
</tr>
<tr>
<td>Simulation function</td>
<td>Use a simulation function when a motion CPU module cannot be provided. The operation of the program can be simulated on the virtual motion CPU module on the software.</td>
</tr>
<tr>
<td>Debug mode</td>
<td>Up to 4 stop positions of the program (which is called break point) can be specified. The program stops when it shifts to the step specified as a break point. (This situation is called During break.) During break, the program can be executed using the following functions in one step at a time.</td>
</tr>
</tbody>
</table>
| Execution of the motion SFC | 1. Reset the PLC CPU module and the motion CPU module.  
Set the RESET/STOP/RUN switch of PLC CPU to RESET.  
Reset is performed by the No. 1 PLC CPU module.  
All CPU modules, including the motion CPU module are reset.  
2. Checking error generation  
3. Execute the program.  
Set the RESET/STOP/RUN switch of the PLC CPU module and the STOP/RUN switch of the motion CPU module to RUN. |
Now that you have completed all of the lessons of the MOTION CONTROLLER Basics (Real Mode:SFC) Course, you are ready to take the final test.
If you are unclear on any of the topics covered, please take this opportunity to review those topics.
There are a total of 5 questions (23 items) in this Final Test.
You can take the final test as many times as you like.

How to score the test
After selecting the answer, make sure to click the Answer button. Your answer will be lost if you proceed without clicking the Answer button. (Regarded as unanswered question.)

Score results
The number of correct answers, the number of questions, the percentage of correct answers, and the pass/fail result will appear on the score page.

Correct Answers : 2
Total Questions : 9
Percentage : 22%

To pass the test, you have to answer 60% of the questions correct.

- Click the Proceed button to exit the test.
- Click the Review button to review the test. (Correct answer check)
- Click the Retry button to retake the test again.
Select the 3 correct features of the operating system software (described as Operation system software hereinafter).

- Operation system software is shipped with installed on a motion CPU module.
- Operation system software is needed to be installed on a motion CPU module.
- Operation system software is needed to be purchased separately from a motion CPU module.
- Operation system software is enclosed with a motion CPU module.
- Set the motion CPU module to the install mode by the rotary switch before installing the Operation system software.
- Operation system software is already installed, so you can use the motion CPU module soon after the purchase.

Answer  Back
Select the functions of the configuration components (such as step, transition) used in a motion SFC program.

<table>
<thead>
<tr>
<th>Configuration component</th>
<th>Processing detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>Main</td>
</tr>
<tr>
<td>END</td>
<td>END</td>
</tr>
<tr>
<td>Operation control step</td>
<td>F1</td>
</tr>
<tr>
<td>Motion control step</td>
<td>K1</td>
</tr>
<tr>
<td>Subroutine call/start step</td>
<td>Subroutine</td>
</tr>
<tr>
<td>Shift transition</td>
<td>G1</td>
</tr>
<tr>
<td>WAIT transition</td>
<td>G1</td>
</tr>
<tr>
<td>Shift Y/N transition</td>
<td>G1</td>
</tr>
<tr>
<td>Jump</td>
<td>P1</td>
</tr>
<tr>
<td>Pointer</td>
<td>P1</td>
</tr>
</tbody>
</table>

Processing detail

1. Executes the specified motion SFC program.
2. Without waiting for the completion of the previous step, shifts the program to the next step when the transition conditions are satisfied.
3. Jumps to the specified pointer Pn in the program.
4. Finishes the program or the subroutine program.
5. Branches to when the transition conditions are satisfied and not satisfied without waiting for the completion of the previous step.
6. Indicates the jump destination pointer (label).
7. When the previous step is the motion control step, waits for the completion of movement and shifts to the next step at the transition condition satisfied.
8. Executes the specified operation control program.
9. Starts the program or the subroutine program.
10. Executes the specified servo program.
Select the correct program where the program shifts to the next step after the completion of movement of the motion control step.

- Program example 1
  - Test
  - K22
  - G0
  - F2
  - END

- Program example 2
  - Test
  - K22
  - G0
  - F2
  - END

- Program example 3
  - Test
  - K22
  - F2
  - F3
  - END

Answer  Back
Select the 3 processes which should be executed before the positioning control when designing a motion SFC program.

- Servo-on
- Servo-off
- JOG operation
- Home position return
- Current value change
- Confirming that the start accept flag turns on
- Confirming that the start accept flag turns off
Select the functions of the configuration components (such as step, transition) used in a motion SFC program.

- Start program number specification
- Start request instruction
- System setting
  - Fixed parameter
  - Servo parameter
  - Parameter block
  - Home position return data
  - JOG operation data

Servo amplifier

Servo motor

Answer
Back
You have completed the Final Test. Your results are as follows.
To end the Final Test, proceed to the next page.

Correct answers: 0
Total questions: 5
Percentage: 0%

You failed the test.
You have completed the MOTION CONTROLLER Basics (Real Mode:SFC) Course.

Thank you for taking this course.

We hope you enjoyed the lessons and the information you acquired in this course will be useful in the future.

You can review the course as many times as you want.

Review  Close