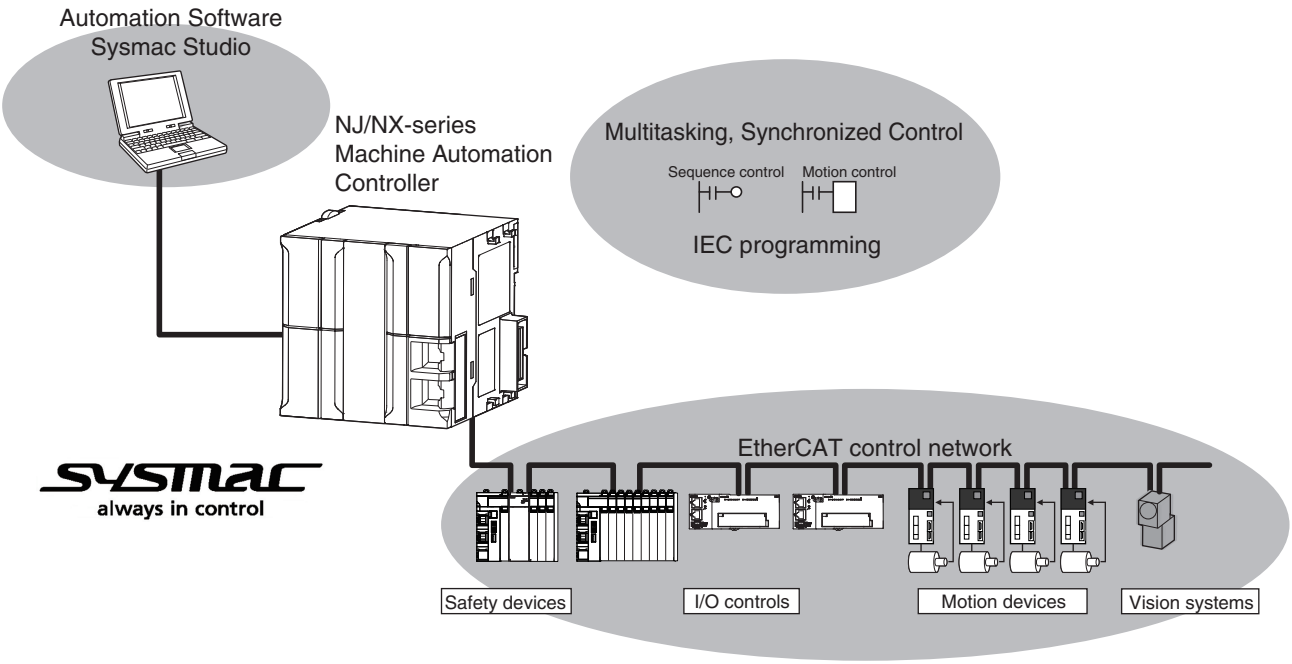
Introduction

What Is a Machine Automation Controller?

The NJ/NX-series Controllers are next-generation machine automation controllers that provide the functionality and high-speed performance that are required for machine control. They provide the safety, reliability, and maintainability that are required of industrial controllers.
The NJ/NX-series Controllers provide the functionality of previous OMRON PLCs, and they also provide the functionality that is required for motion control. Synchronized control of I/O devices on highspeed EtherCAT® can be applied to safety devices, vision systems, motion equipment, discrete I/O, and more.



Features

Hardware Features

Standard-feature EtherCAT Control Network Support
All CPU Units provide an EtherCAT master port for EtherCAT communications.
EtherCAT is an advanced industrial network system that achieves faster, more-efficient communications. It is based on Ethernet. Each node achieves a short fixed communications cycle time by transmitting Ethernet frames at high speed.
The standard-feature EtherCAT control network allows you to connect all of the devices required for machine control (e.g., I/O systems, servo drives, inverters, and vision systems) to the same network.

Support for EtherCAT Slave Terminals
You can use EtherCAT Slave Terminals to save space. You can also flexibly build systems with the wide variety of NX Units.

Achieving a Safety Subsystem on EtherCAT
You can use NX-series Safety Control Units to integrate safety controls in a sequence and motion control system as a subsystem on EtherCAT.

CJ-series Units
NJ-series CPU Units allow you to mount CJ-series Basic I/O Units and Special Units on the I/O bus, in addition to EtherCAT network slaves.

Standard-feature EtherNet/IP Communications Port
All CPU Units provide an EtherNet/IP port for EtherNet/IP communications.
EtherNet/IP is a multi-vendor industrial network that uses Ethernet. You can use it for networks between Controllers or as a field network. The use of standard Ethernet technology allows you to connect to many different types of general-purpose Ethernet devices.

Standard-feature USB Port
You can connect a computer that runs the Support Software directly to the CPU Unit with a USB connection.

Standard-feature SD Memory Card Slot
You can access an SD Memory Card that is mounted in the CPU Unit from the user program.

Highly Reliable Hardware
The NJ/NX-series Controllers provide the hardware reliability and RAS functions that you expect of a PLC.

Parallel Execution of Tasks with a Multi-core Processor
The NX701-□□□□ CPU Unit has a multi-core processor that can execute more than one task in parallel. This enables high-speed control of even large-scale devices.

Software Features

Integrated Sequence Control and Motion Control
An NJ/NX-series CPU Unit can perform both sequence control and motion control. You can simultaneously achieve both sequence control and multi-axes synchronized control. Sequence control, motion control, and I/O refreshing are all executed in the same control period. The same control period is also used for the process data communications cycle for EtherCAT. This enables precise sequence and motion control in a fixed period with very little deviation.

Multitasking
You assign I/O refreshing and programs to tasks and then specify execution conditions and execution order for them to flexibly combine controls that suit the application.

Programming Languages Based on the IEC 61131-3 International Standard
The NJ/NX-series Controllers support language specifications that are based on IEC 61131-3. To these, OMRON has added our own improvements. Motion control instructions that are based on PLCopen ® standards and an instruction set (POUs) that follows IEC rules are provided.

Programming with Variables to Eliminate Worrying about the Memory Map
You access all data through variables in the same way as for the advanced programming languages that are used on computers. Memory in the CPU Unit is automatically assigned to the variables that you create so that you do not have to remember the physical addresses.

A Wealth of Security Features
The many security features of the NJ/NX-series Controllers include operation authority settings and restriction of program execution with IDs.

Complete Controller Monitoring
The CPU Unit monitors events in all parts of the Controller, including mounted Units and EtherCAT slaves. Troubleshooting information for errors is displayed on the Sysmac Studio or on an NA-series PT. Events are also recorded in logs.

Automation Software Sysmac Studio
The Sysmac Studio provides an integrated development environment that covers not only the Controller, but also covers peripheral devices and devices on EtherCAT. You can use consistent procedures for all devices regardless of differences in the devices. The Sysmac Studio supports all phases of Controller application, from designing through debugging, simulations, commissioning, and changes during operation.

A Wealth of Simulation Features
The many simulation features include execution, debugging, and task execution time estimates on a virtual controller.

System Configurations

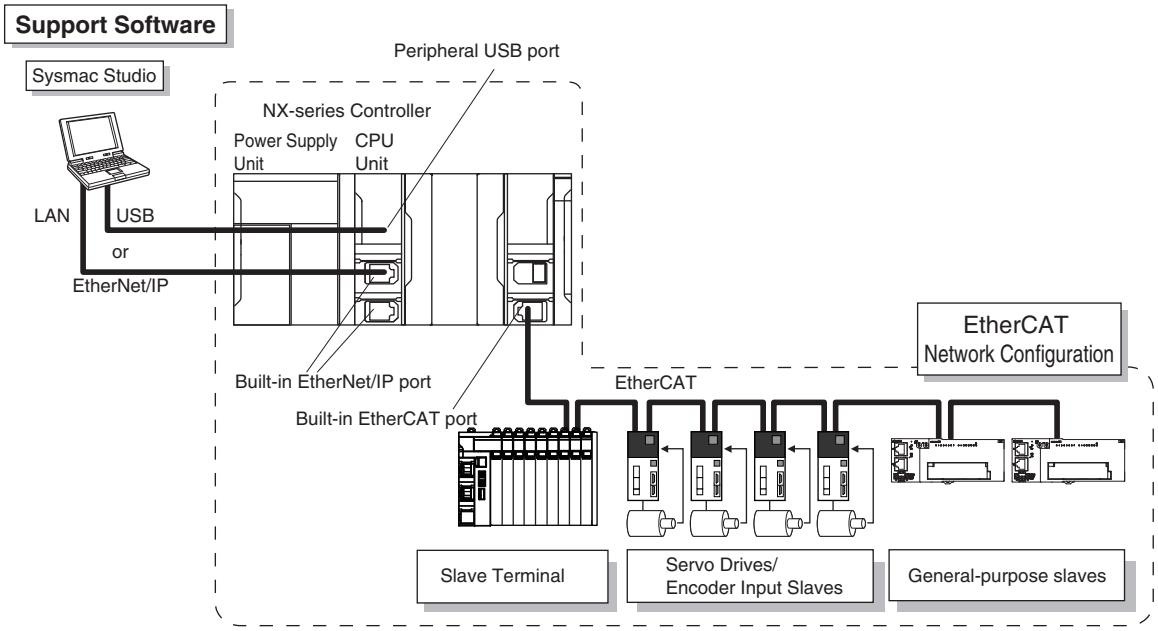
Introduction to the System Configurations of the NX-series Controllers

The NX Series supports the following system configurations.

Basic System Configurations

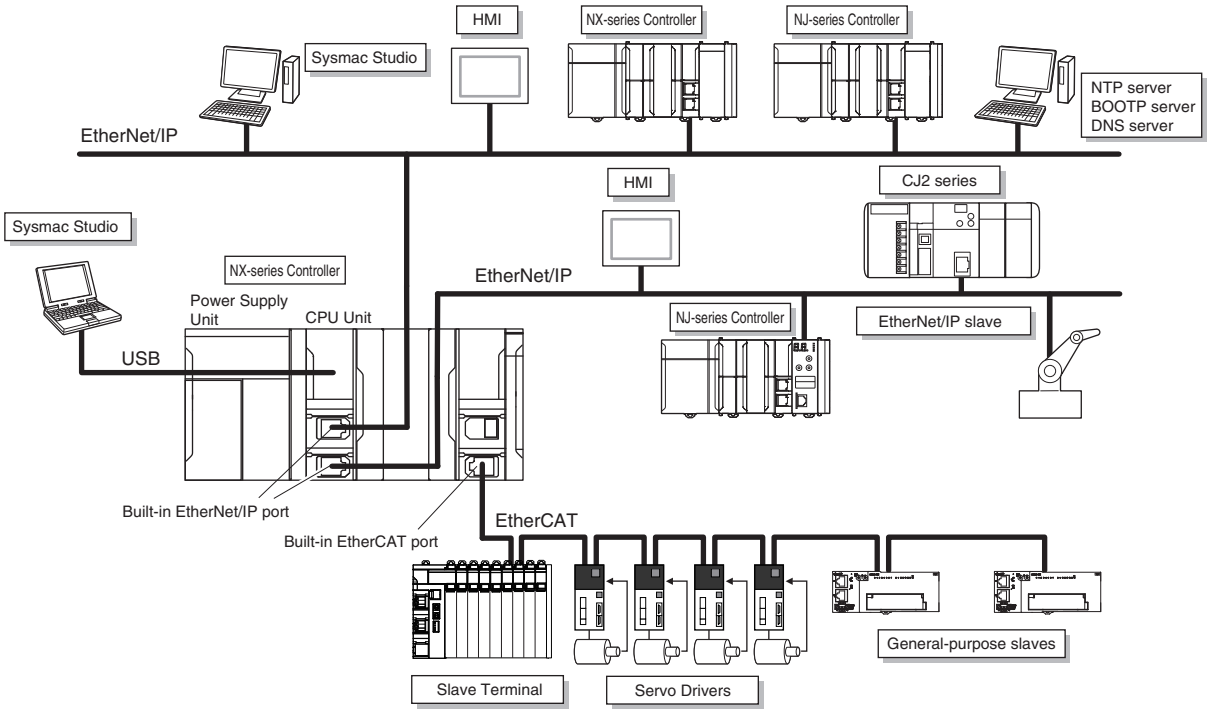
The NX-series configurations include the EtherCAT network configuration and the Support Software.

- EtherCAT Network Configuration
  You can use the built-in EtherCAT port to connect to EtherCAT Slave Terminals, to general-purpose slaves for analog and digital I/O, and to Servo Drives and encoder input slaves. An EtherCAT network configuration enables precise sequence and motion control in a fixed cycle with very little deviation.
- Support Software
  The Support Software is connected to the peripheral USB port on the CPU Unit with a commercially available USB cable. You can also connect it through an Ethernet cable that is connected to the built-in EtherNet/IP port.



NX-series System Configuration Example

You can use the NX-series System to build the communications system shown below.



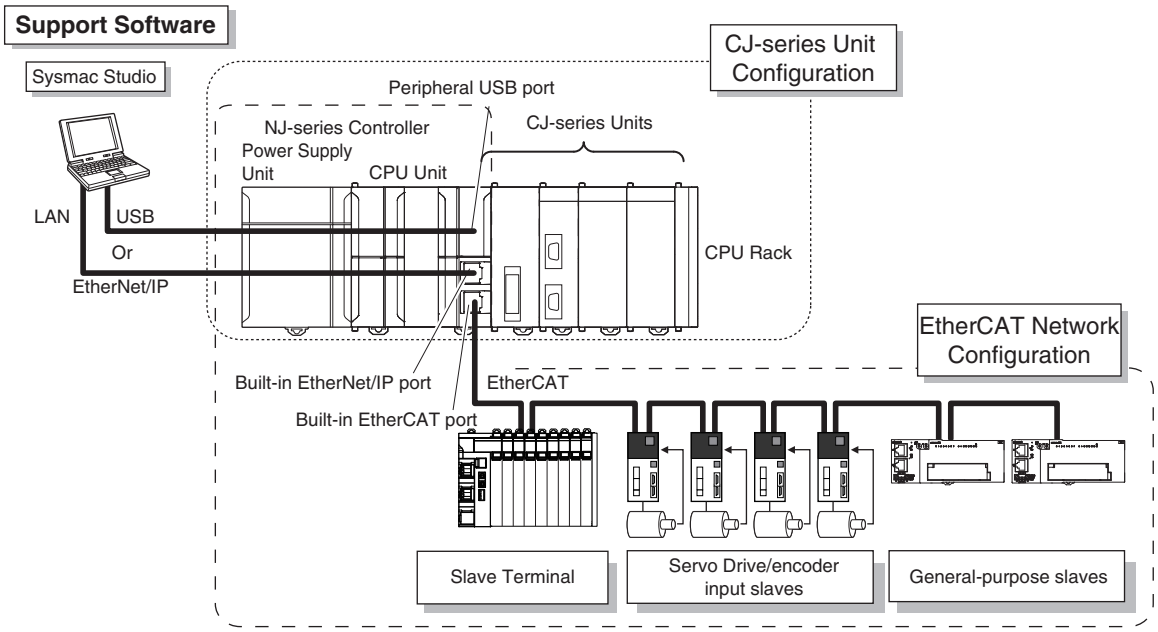
Introduction to the System Configurations of the NJ-series Controllers

The NJ Series supports the following system configurations.

Basic System Configurations

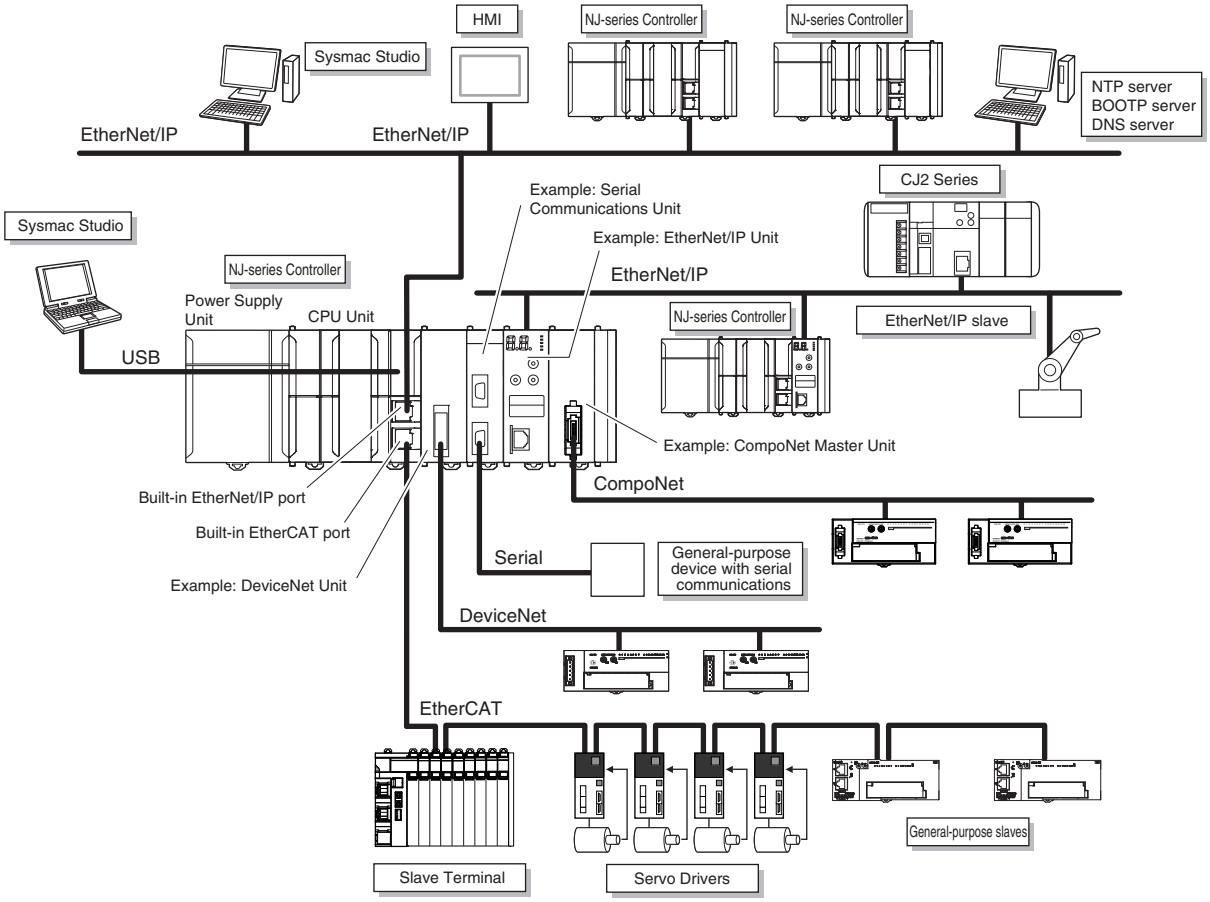
The NJ-series basic configurations include the EtherCAT network configuration, CJ-series Unit configuration, and the Support Software.

- EtherCAT Network Configuration
  You can use the built-in EtherCAT port to connect to EtherCAT Slave Terminals, to general-purpose slaves for analog and digital I/O, and to Servo Drives and encoder input slaves. An EtherCAT network configuration enables precise sequence and motion control in a fixed cycle with very little deviation.
- CJ-series Unit Configuration
  In addition to the EtherCAT network, you can also mount CJ-series Basic I/O Units and Special Units. CJ-series Units can be mounted both to the CPU Rack where the CPU Unit is mounted and to Expansion Racks.
- Support Software
  The Support Software is connected to the peripheral USB port on the CPU Unit with a commercially available USB cable. You can also connect it through an Ethernet cable that is connected to the built-in EtherNet/IP port.



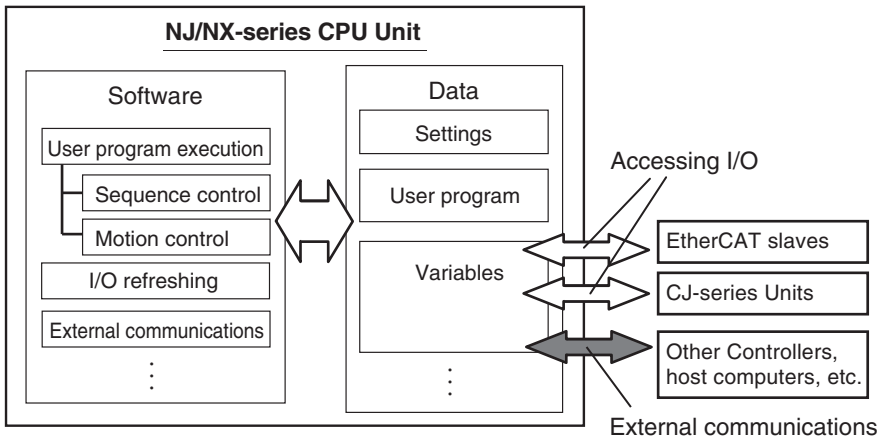
NJ-series System Configuration Example

You can use the NJ-series System to build the communications system shown below.



CPU Unit Operation

The NJ/NX-series CPU Unit executes the user program for sequence control and motion control. It also performs other processing, such as I/O refreshing and external communications. These processes are performed by the software in the CPU Unit.
The CPU Unit also contains settings, the user program, variables, and other data. The CPU Unit uses this data to perform processing. Of this data, variables are used to access the CPU Unit and I/O, and for external communications.
The internal software and the use of variables for I/O access enable the CPU Unit to execute both sequence control and motion control.



Note: You can use CJ-series Units only with NJ-series CPU Units.

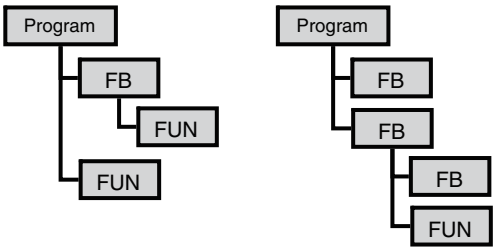
Program Organization Units (POUs)

A POU (program organization unit) is a unit that is defined in the IEC 61131-3 user program execution model.
You combine POUs to build a complete user program.
There are three types of POUs, as described below.

POU configuration element	Description
Programs	A program corresponds to a main routine. It is the main type of POU that is used for algorithms. You can place any instruction, function, or function block in the algorithm of a program.
Function Blocks (FBs)	A function block can output different values even with the same inputs. Function blocks are executed when they are called from a program or another function block. To use a function block in a program, an instance of the function block must be placed in the program. You can retain the values of internal variables. Therefore, you can retain status, such as for timers and counters.
Functions (FUNs)	A function always outputs the same values for the same inputs. Functions are executed when they are called from a program, another function, or a function block.

Easy-to-Read Programming

Programs can be organized in layers by calling POUs from other POUs. For example, you can increase the readability of your programs by structuring them according to units of control.
The following figure shows the structure as examples.



Reusable Programming

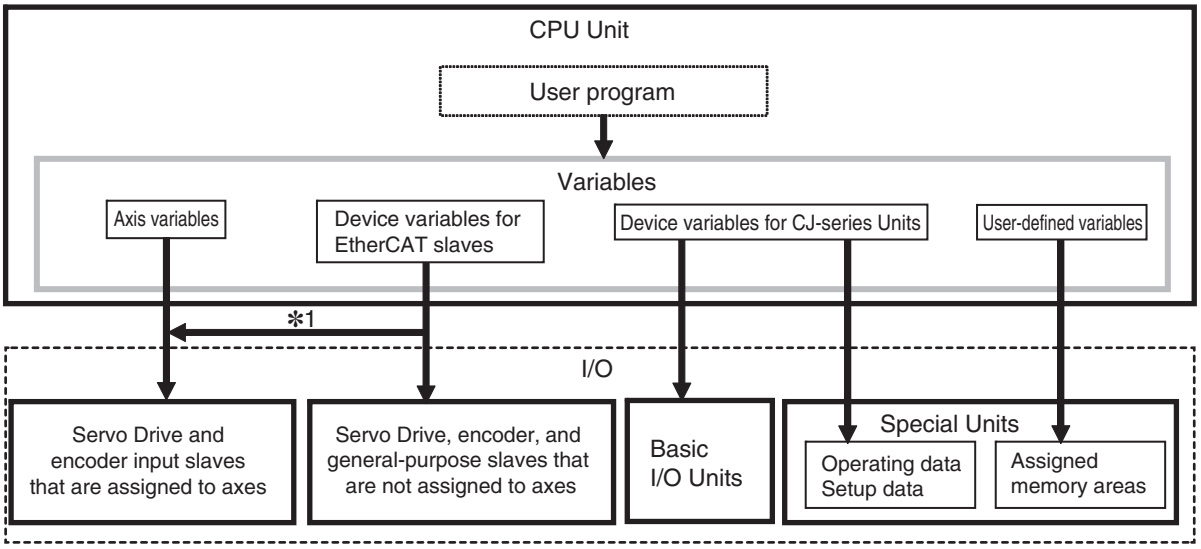
Function blocks and functions are used to divide programs into smaller, more manageable objects. If processes are divided up into function blocks, you can call instances of those function blocks to reuse them in other devices that require those same processes.

Accessing I/O with Variables

In the CPU Unit, variables are used in the user program. Variables access the data of the assigned I/O. The following table shows how I/O and variables are assigned in the CPU Unit. The type of variable that is used by a CJ-series Special Unit depends on the data to access.

I/O		Data	Variables
EtherCAT slaves	EtherCAT slaves to which axes are not assigned ✻1	---	Device variables for Ether-CAT slaves
	EtherCAT slaves to which axes are assigned	---	Axis variables
CJ-series Units ✻2	Basic I/O Units	---	Device variables for CJseries Units
	Special Units	• Operating data • Setup data	Device variables for CJseries Units
		Assigned memory area data ✻3	User-defined variables

✻1. With the Sysmac Studio version 1.09 or higher, the EtherCAT slaves to which axes are assigned can also be accessed via EtherCAT slave device variables.
✻2. You can use CJ-series Units only with NJ-series CPU Units.
✻3. This data is for extended functions and slave I/O that you assign by specifying addresses in memory. You cannot access assigned memory area data with device variables.



Explanation of Terms

AT Specification
One of the attributes of a variable.
This attribute allows the user to specify what is assigned to a variable.
An I/O port or an address in memory used for CJ-series Units can be specified.

FB
An acronym for "function block."

FUN
An abbreviation for "function."

I/O Port
A logical interface that is used by the CPU Unit to exchange data with an external device (slave or Unit).

I/O Map Settings
Settings that assign variables to I/O ports.
Assignment information between I/O ports and variables.

PDO Communications
An abbreviation for process data communications. Data is exchanged between the master and slaves on a process data communications cycle. (The process data communications cycle is the same as the task period of the primary periodic task.)

POU (Program Organization Unit)
A POU is a unit in a program execution model that is defined in IEC 61131-3. It contains an algorithm and a local variable table and forms the basic unit used to build a user program. There are three types of POUs: programs, functions, and function blocks.

SDO Communications
One type of EtherCAT communications in which service data objects (SDOs) are used to transmit information whenever required.

Basic Data Type
Any of the data types that are defined by IEC 61131-3.
They include Boolean, bit string, integer, real, duration, date, time of day, date and time, and text string data types.
"Basic data type" is used as opposed to derivative data types, which are defined by the user.

Union
One of the derivative data types. It allows you to handle the same data as different data types.

Global Variable
A variable that can be read or written from all POUs (programs, functions, and function blocks).

Structure
One of the derivative data types. It consists of multiple data types placed together into a layered structure.

Constant
One of the attributes of a variable.
If you specify the Constant attribute for a variable, the value of the variable cannot be written by any instructions, ST operators, or CIP message communications.

Axis
A functional unit within the Motion Control Function Module. An axis is assigned to the drive mechanism in an external Servo Drive or the sensing mechanism in an external Encoder Input Slave Unit.

Axes Group
A functional unit that groups together axes within the Motion Control Function Module.

Axes Group Variable
A system-defined variable that is defined as a structure and provides status information and some of the axes parameters for an individual axes group.
An Axes Group Variable is used to specify an axes group for motion control instructions and to monitor the command interpolation velocity, error information, and other information for the axes group.

Axis Variable
A system-defined variable that is defined as a structure and provides status information and some of the axis parameters for an individual axis.
An Axis Variable is used to specify an axis for motion control instructions and to monitor the command position, error information, and other information for the axis.

System-defined Variable
A variable for which all attributes are defined by the system and cannot be changed by the user.

Initial Value
One of the attributes of a variable.
The variable is set to the initial value in the following situations.
- When power is turned ON
- When the CPU Unit changes to RUN mode
- When you specify to initialize the values when the user program is transferred
- When a major fault level Controller error occurs

Task
An attribute that defines when a program is executed.

Task Period
The interval at which the primary periodic task or a periodic task is executed.

Periodic Task
A task for which user program execution and I/O refreshing are performed each period.

Device Variable
A variable that is used to access a specific device through an I/O port.

Namespace
A system that is used to group and nest the names of functions, function block definitions, and data types.

Network Publish
One of the attributes of a variable.
This attribute allows you to use CIP message communications or tag data links to read/write variables from another Controller or from a host computer.

Array Specification
One of the variable specifications.
An array variable contains multiple elements of the same data type. The elements in the array are specified by serial numbers called subscripts that start from the beginning of the array.

Derivative Data Type
A data type that is defined by the user.
Structures, unions, and enumerations are derivative data types.

Function
A POU that is used to create an object that determines a unique output for the same input, such as for data processing.

Function Block
A POU that is used to create an object that can have a different output for the same input, such as for a timer or counter.

Primary Periodic Task
The task with the highest priority.

Program
One of three POUs. The others are functions and function blocks.
Programs are assigned to tasks to execute them.

Process Data Communications
One type of EtherCAT communications in which process data objects (PDOs) are used to exchange information cyclically and in realtime. It is also called “PDO communications.”.

Variable
A representation of data, such as a numeric value or character string, that is used in a user program.
You can change the value of a variable by assigned the required value. “Variable” is used as opposed to “constant,” for which the value does not change.

Variable Memory
A memory area that contains the present values of variables that do not have AT specifications.
It can be accessed only with variables without an AT attribute.

Retain
One of the attributes of a variable.
The values of variables with a Retain attribute are held at the following times. (Variables without a Retain attribute are set to their initial values.)
- When power is turned ON after a power interruption
- When the CPU Unit changes to RUN mode
- When you specify to not initialize the values when the user program is transferred

Motion Control Instruction
A function block instruction that executes motion control.
The Motion Control Function Module supports instructions that are based on function blocks for PLCopen® motion control as well as instructions developed specifically for the Motion Control Function Module.

User-defined Variable
A variable for which all of the attributes are defined by the user and can be changed by the user.

Literal
A constant expression that is used in a user program.

Enumeration
One of the derivative data types.
This data type takes one item from a prepared name list of enumerators as its value.

Enumerator
One of the values that an enumeration can take expressed as a character string.
The value of an enumeration is one of the enumerators.

Local Variable
A variable that can be accessed only from inside the POU in which it is defined.
“Local variable” is used as opposed to “global variable.” Local variables include internal variables, input variables, output variables, in-out variables, and external variables.