Logix5000 Controllers I/O and Tag Data

Allen-Bradley

Catalog Numbers 1756 ControlLogix, 1756 GuardLogix, 1768 Compact GuardLogix, 1769 CompactLogix, 1789 SoftLogix, PowerFlex with DriveLogix

Programming Manual

Allen-Bradley • Rockwell Software

Rockwell Automation

Important User Information

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

WARNING ⚠	Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.
IMPORTANT	Identifies information that is critical for successful application and understanding of the product.
ATTENTION ⚠	Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.
SHOCK HAZARD ⚠	Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.
BURN HAZARD ⚠	Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

Summary of Changes

Introduction

The release of this document contains new information.

New Information

New information is marked by change bars in the side column, as shown to the right.

Section	Changes
Chapter 1	New information and examples for electronic keying.
Chapter 4	Procedures and descriptions for the external access and constant tag attributes that control access to controller tags.

Notes:

Purpose of This Manual

This manual shows how to access I/O and tag data in Logix5000 controllers. This manual is one of a set of related manuals that show common procedures for programming and operating Logix5000 controllers.

For a complete list of common procedures manuals, refer to the Logix 5000 Controllers Common Procedures Programming Manual, publication 1756-PM001.

The term Logix5000 controller refers to any controller that is based on the Logix5000 operating system, such as:

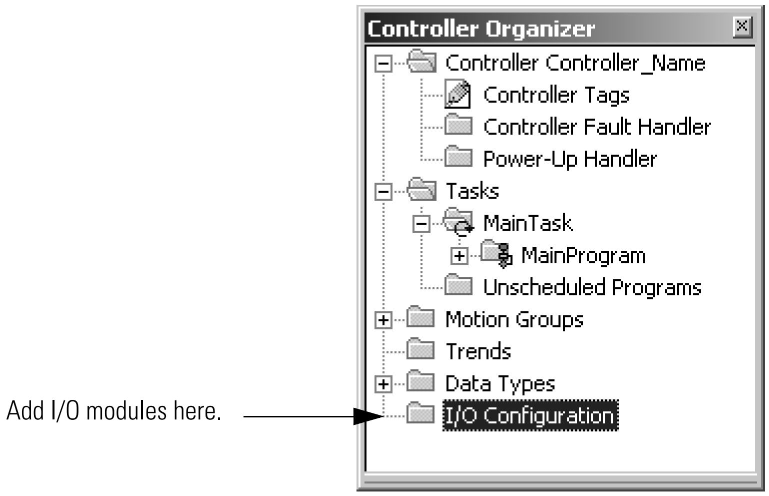
- CompactLogix controllers.
- ControlLogix controllers.
- DriveLogix controllers.
- FlexLogix controllers.
- SoftLogix5800 controllers.

Notes:

Communicate with I/O Modules

Introduction

To communicate with an I/O module in your system, you add the module to the I/O Configuration folder of the controller.



When you add the module, you also define a specific configuration for the module. While the configuration options vary from module to module, these are some common options that you typically configure:

- Requested Packet Interval
- Communication Format
- Electronic Keying

Requested Packet Interval

The Logix5000 controller uses connections to transmit I/O data.

Term	Definition
Connection	A communication link between two devices, such as between a controller and an I/O module, PanelView terminal, or another controller. Connections are allocations of resources that provide more reliable communications between devices than unconnected messages. The number of connections that a single controller can have is limited. You indirectly determine the number of connections the controller uses by configuring the controller to communicate with other devices in the system. The following types of communication use connections: • I/O modules • produced and consumed tags • certain types of Message (MSG) instructions (not all types use a connection)
Requested packet interval (RPI)	The RPI specifies the period at which data updates over a connection. For example, an input module sends data to a controller at the RPI that you assign to the module. • Typically, you configure an RPI in milliseconds (ms). The range is 0.2 ms (200 microseconds)…750 ms. • If a ControlNet network connects the devices, the RPI reserves a slot in the stream of data flowing across the ControlNet network. The timing of this slot may not coincide with the exact value of the RPI, but the control system guarantees that the data transfers at least as often as the RPI.

In Logix5000 controllers, I/O values update at a period that you configure via the I/O configuration folder of the project. The values update asynchronous to the execution of logic. At the specified interval, the controller updates a value independently from the execution of logic.

ATTENTION ⚠ Make sure that data memory contains the appropriate values throughout a task's execution. You can duplicate or buffer data at the beginning of the scan to provide reference values for your logic.

- Programs within a task access input and output data directly from controller-scoped memory.
- Logic within any task can modify controller-scoped data.
- Data and I/O values are asynchronous and can change during the course of a task's execution.
- An input value referenced at the beginning of a task's execution can be different when referenced later.
- To prevent an input value from changing during a scan, copy the value to another tag and use the data from there (buffer the values).

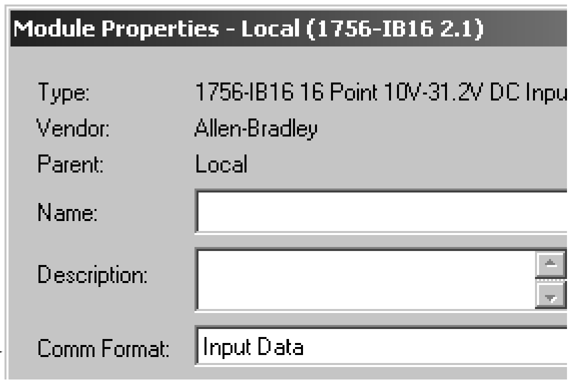
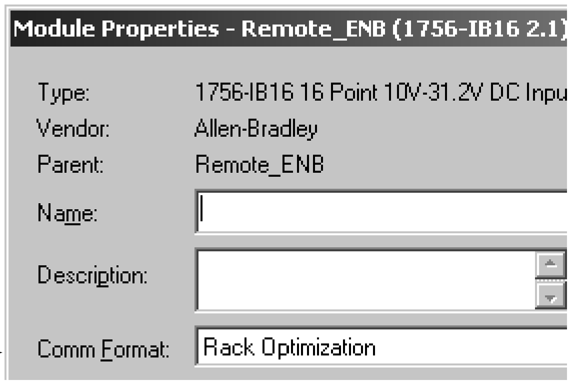
Communication Format

The communication format that you choose determines the data structure for the tags that are associated with the module. Many I/O modules support different formats. Each format uses a different data structure. The communication format that you choose also determines:

- Direct or Rack-Optimized Connection.
- Ownership.

Direct or Rack-Optimized Connection

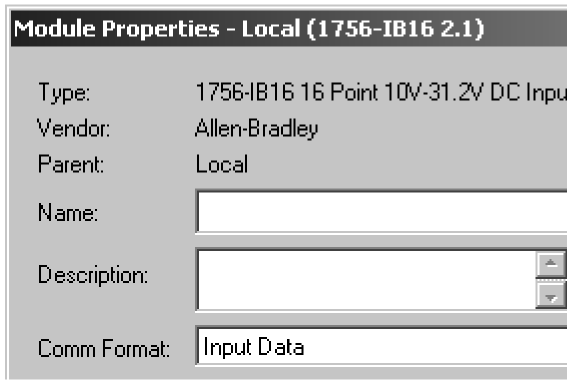
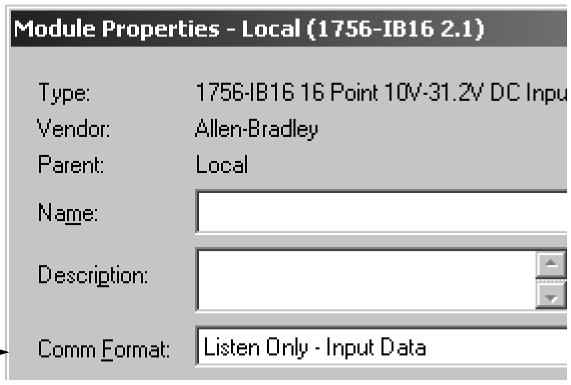
The Logix5000 controller uses connections to transmit I/O data. These connections can be direct connections or rack-optimized connections.

Term	Definition
Direct connection	A direct connection is a real-time, data transfer link between the controller and an I/O module. The controller maintains and monitors the connection with the I/O module. Any break in the connection, such as a module fault or the removal of a module while under power, sets fault bits in the data area associated with the module.  Module Properties - Local (1756-IB16 2.1) Type: 1756-IB16 16 Point 10V-31.2V DC Inpu Vendor: Allen-Bradley Parent: Local Name: Description: Comm Format: Input Data A direct connection is any connection that does not use the Rack Optimization Comm Format.
Rack-optimized connection	For digital I/O modules, you can select rack-optimized communication. A rack-optimized connection consolidates connection usage between the controller and all the digital I/O modules in the chassis (or DIN rail). Rather than having individual, direct connections for each I/O module, there is one connection for the entire chassis (or DIN rail).  Module Properties - Remote_ENB (1756-IB16 2.1) Type: 1756-IB16 16 Point 10V-31.2V DC Inpu Vendor: Allen-Bradley Parent: Remote_ENB Name: Description: Comm Format: Rack Optimization

Ownership

In a Logix5000 system, modules multicast data. This means that multiple devices can receive the same data at the same time from a single device.

When you choose a communication format, you have to choose whether to establish an owner or listen-only relationship with the module.

Owner controller	The controller that creates the primary configuration and communication connection to a module. The owner controller writes configuration data and can establish a connection to the module.  An owner connection is any connection that does not include Listen-Only in its Comm Format. Module Properties - Local (1756-IB16 2.1) Type: 1756-IB16 16 Point 10V-31.2V DC Inpu Vendor: Allen-Bradley Parent: Local Name: Description: Comm Format: Input Data
Listen-only connection	An I/O connection where another controller owns/provides the configuration data for the I/O module. A controller using a listen-only connection only monitors the module. It does not write configuration data and can only maintain a connection to the I/O module when the owner controller is actively controlling the I/O module.  Listen-only Connection Module Properties - Local (1756-IB16 2.1) Type: 1756-IB16 16 Point 10V-31.2V DC Inpu Vendor: Allen-Bradley Parent: Local Name: Description: Comm Format: Listen Only - Input Data

Use the following table to choose the type of ownership for a module.

Choose the Type of Ownership

If module is	And another controller	And you want to	Then use this type of connection
Input module	Does not own the module	⟶	Owner (not listen-only)
	Owns the module	Maintain communication with the module if it loses communication with the other controller	Owner (not listen-only) Use the same configuration as the other owner controller.
		Stop communication with the module if it loses communication with the other controller	Listen-only
Output module	Does not own the module	⟶	Owner (such as, not listen-only)
	Owns the module	⟶	Listen-only

There is a noted difference in controlling input modules versus controlling output modules.

Control Input and Output Modules

Controlling	This Ownership	Description
Input modules	Owner	An input module is configured by a controller that establishes a connection as an owner. This configuring controller is the first controller to establish an owner connection. Once an input module has been configured (and owned by a controller), other controllers can establish owner connections to that module. This lets additional owners to continue to receive multicast data if the original owner controller breaks its connection to the module. All other additional owners must have the identical configuration data and identical communications format that the original owner controller has, otherwise, the connection attempt is rejected.
	Listen-only	Once an input module has been configured (and owned by a controller), other controllers can establish a listen-only connection to that module. These controllers can receive multicast data while another controller owns the module. If all owner controllers break their connections to the input module, all controllers with listen-only connections no longer receive multicast data.
Output modules	Owner	An output module is configured by a controller that establishes a connection as an owner. Only one-owner connection is allowed for an output module. If another controller attempts to establish an owner connection, the connection attempt is rejected.
	Listen-only	Once an output module has been configured (and owned by one controller), other controllers can establish listen-only connections to that module. These controllers can receive multicast data while another controller owns the module. If the owner controller breaks its connection to the output module, all controllers with listen-only connections no longer receive multicast data.

Electronic Keying

The electronic keying feature automatically compares the expected module, as shown in the RSLogix 5000 I/O Configuration tree, to the physical module before I/O communication begins. You can use electronic keying to help prevent communication to a module that does not match the type and revision expected.

For each module in the I/O Configuration tree, the user-selected keying option determines if, and how, an electronic keying check is performed. Typically, three keying options are available.

- Exact Match
- Compatible Keying
- Disable Keying

You must carefully consider the benefits and implications of each keying option when selecting between them. For some specific module types, fewer options are available.

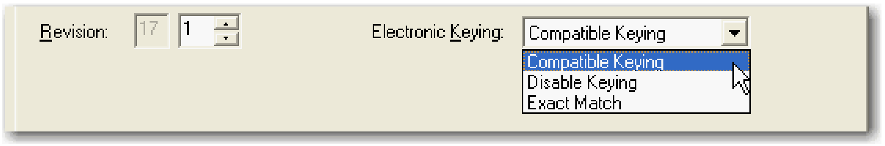
Electronic keying is based on a set of attributes unique to each product revision. When a Logix5000 controller begins communicating with a module, this set of keying attributes is considered.

Keying Attributes

Attribute	Description
Vendor	The manufacturer of the module, for example, Rockwell Automation/Allen-Bradley.
Product Type	The general type of the module, for example, communication adapter, AC drive, or digital I/O.
Product Code	The specific type of module, generally represented by its catalog number, for example, 1756-IB16I.
Major Revision	A number that represents the functional capabilities and data exchange formats of the module. Typically, although not always, a later, that is higher, Major Revision supports at least all of the data formats supported by an earlier, that is lower, Major Revision of the same catalog number and, possibly, additional ones.
Minor Revision	A number that indicates the module's specific firmware revision. Minor Revisions typically do not impact data compatibility but may indicate performance or behavior improvement.

You can find revision information on the General tab of a module's Properties dialog box.

General Tab


Revision: 17 1

Electronic Keying: Compatible Keying

IMPORTANT Changing electronic keying selections online may cause the I/O communication connection to the module to be disrupted and may result in a loss of data.

Exact Match

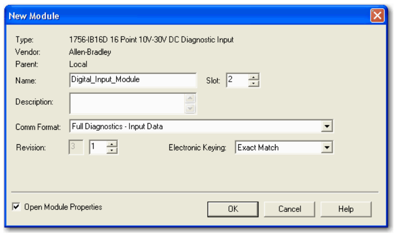
Exact Match keying requires all keying attributes, that is, Vendor, Product Type, Product Code (catalog number), Major Revision, and Minor Revision, of the physical module and the module created in the software to match precisely in order to establish communication. If any attribute does not match precisely, I/O communication is not permitted with the module or with modules connected through it, as in the case of a communication module.

Use Exact Match keying when you need the system to verify that the module revisions in use are exactly as specified in the project, such as for use in highly-regulated industries. Exact Match keying is also necessary to enable Automatic Firmware Update for the module via the Firmware Supervisor feature from a Logix5000 controller.

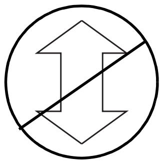
EXAMPLE In the following scenario, Exact Match keying prevents I/O communication:
The module configuration is for a 1756-IB16D module with module revision 3.1. The physical module is a 1756-IB16D module with module revision 3.2. In this case, communication is prevented because the Minor Revision of the module does not match precisely.

Module Configuration

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 3
Minor Revision = 1



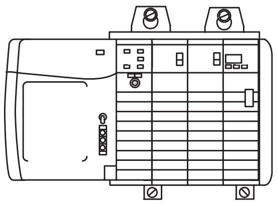
Type: 1756-IB16D 16 Point 10V-30V DC Diagnostic Input
Vendor: Allen-Bradley
Parent: Local
Name: Digital_Input_Module Slot: 2
Description:
Comm Format: Full Diagnostics - Input Data
Revision: 3 1 Electronic Keying: Exact Match
☑ Open Module Properties



Communication is prevented

Physical Module

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 3
Minor Revision = 2



IMPORTANT Changing electronic keying selections online may cause the I/O Communication connection to the module to be disrupted and may result in a loss of data.

Compatible Keying

Compatible Keying indicates that the module determines whether to accept or reject communication. Different module families, communication adapters, and module types implement the compatibility check differently based on the family capabilities and on prior knowledge of compatible products.

Compatible Keying is the default setting. Compatible Keying allows the physical module to accept the key of the module configured in the software, provided that the configured module is one the physical module is capable of emulating. The exact level of emulation required is product and revision specific.

With Compatible Keying, you can replace a module of a certain Major Revision with one of the same catalog number and the same or later, that is higher, Major Revision. In some cases, the selection makes it possible to use a replacement that is a different catalog number than the original. For example, you can replace a 1756-CNBR module with a 1756-CN2R module.

Release notes for individual modules indicate the specific compatibility details.

When a module is created, the module developers consider the module's development history to implement capabilities that emulate those of the previous module. However, the developers cannot know future developments. Because of this, when a system is configured, we recommend that you configure your module using the earliest, that is, lowest, revision of the physical module that you believe will be used in the system.

By doing this, you can avoid the case of a physical module rejecting the keying request because it is an earlier revision than the one configured in the software.

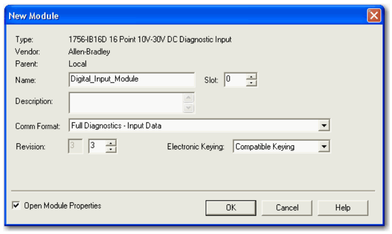
EXAMPLE

In the following scenario, **Compatible Keying prevents I/O communication**:

The module configuration is for a 1756-IB16D module with module revision 3.3. The physical module is a 1756-IB16D module with module revision 3.2. In this case, communication is prevented because the minor revision of the module is lower than expected and may not be compatible with 3.3.

Module Configuration

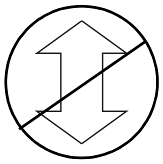
Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 3
Minor Revision = 3


New Module

Type: 1756-IB16D 16 Point 10V-30V DC Diagnostic Input
Vendor: Allen-Bradley
Parent: Local
Name: Digital_Input_Module
Slot: 0
Description:
Comm Format: Full Diagnostics - Input Data
Revision: 3 3
Electronic Keying: Compatible Keying

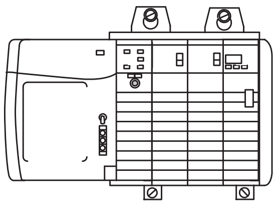
☑ Open Module Properties

Communication is prevented



Physical Module

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 3
Minor Revision = 2



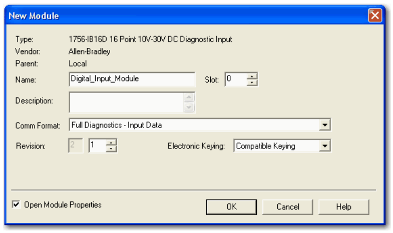
EXAMPLE

In the following scenario, **Compatible Keying allows I/O communication**:

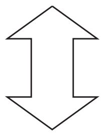
The module configuration is for a 1756-IB16D module with module revision 2.1. The physical module is a 1756-IB16D module with module revision 3.2. In this case, communication is allowed because the major revision of the physical module is higher than expected and the module determines that it is compatible with the prior major revision.

Module Configuration

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 2
Minor Revision = 1

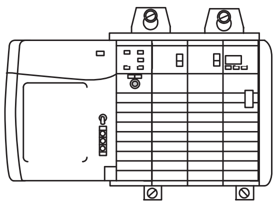

New Module

Type: 1756-IB16D 16 Point 10V-30V DC Diagnostic Input
Vendor: Allen-Bradley
Parent: Local
Name: Digital_Input_Module
Slot: 0
Description:
Comm Format: Full Diagnostics - Input Data
Revision: 2 1
Electronic Keying: Compatible Keying



Physical Module

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16D
Major Revision = 3
Minor Revision = 2



IMPORTANT

Changing electronic keying selections online may cause the I/O communication connection to the module to be disrupted and may result in a loss of data.

Disabled Keying

Disabled Keying indicates the keying attributes are not considered when attempting to communicate with a module. Other attributes, such as data size and format, are considered and must be acceptable before I/O communication is established. With Disabled Keying, I/O communication may occur with a module other than the type specified in the I/O Configuration tree with unpredictable results. We generally do not recommend using Disabled Keying.

ATTENTION

Be extremely cautious when using Disabled Keying; if used incorrectly, this option can lead to personal injury or death, property damage, or economic loss.

If you use Disabled Keying, you must take full responsibility for understanding whether the module being used can fulfill the functional requirements of the application.

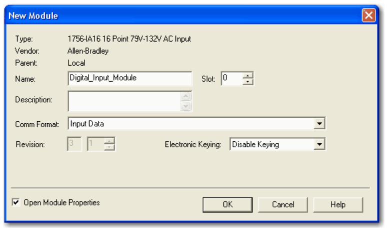
EXAMPLE

In the following scenario, **Disable Keying prevents I/O communication**:

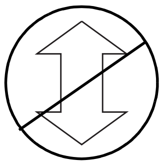
The module configuration is for a 1756-IA16 digital input module. The physical module is a 1756-IF16 analog input module. In this case, **communication is prevented because the analog module rejects the data formats that the digital module configuration requests**.

Module Configuration

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IA16
Major Revision = 3
Minor Revision = 1

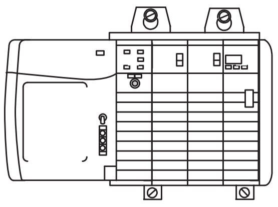

New Module

Type: 1756-IA16 16 Point 79V-132V AC Input
Vendor: Allen-Bradley
Parent: Local
Name: Digital_Input_Module
Slot: 0
Description:
Comm Format: Input Data
Revision: 3 1
Electronic Keying: Disable Keying
☑ Open Module Properties


Communication is prevented

Physical Module

Vendor = Allen-Bradley
Product Type = Analog Input Module
Catalog Number = 1756-IF16
Major Revision = 3
Minor Revision = 2



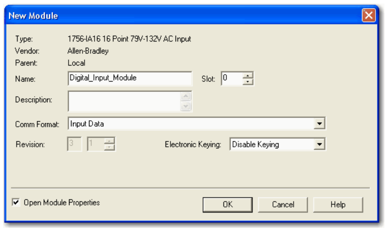
EXAMPLE

In the following scenario, **Disable Keying allows I/O communication**:

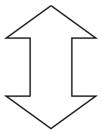
The module configuration is for a 1756-IA16 digital input module. The physical module is a 1756-IB16 digital input module. In this case, communication is allowed because the two digital modules share common data formats.

Module Configuration

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IA16
Major Revision = 2
Minor Revision = 1

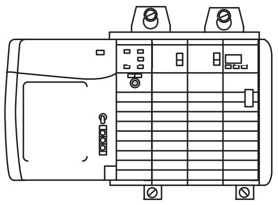

New Module

Type: 1756-IA16 16 Point 79V-132V AC Input
Vendor: Allen-Bradley
Parent: Local
Name: Digital_Input_Module
Slot: 0
Description:
Comm Format: Input Data
Revision: 3 1
Electronic Keying: Disable Keying
☑ Open Module Properties



Physical Module

Vendor = Allen-Bradley
Product Type = Digital Input Module
Catalog Number = 1756-IB16
Major Revision = 3
Minor Revision = 2



IMPORTANT

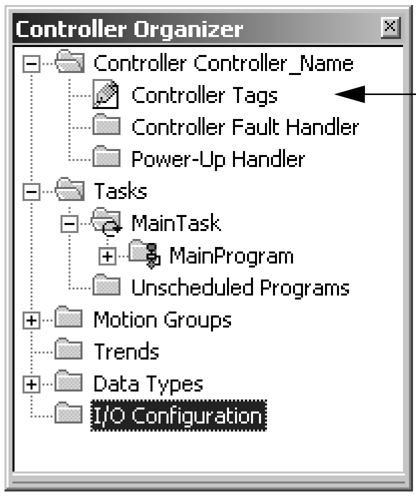
Changing electronic keying selections online may cause the I/O communication connection to the module to be disrupted and may result in a loss of data.

Address I/O Data

I/O information is presented as a set of tags.

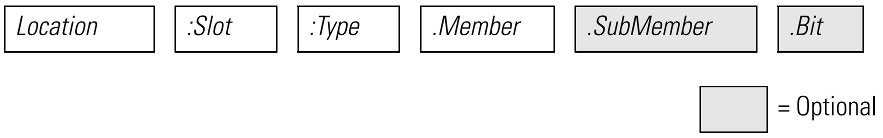
- Each tag uses a structure of data. The structure depends on the specific features of the I/O module.
- The name of the tag is based on the location of the I/O module in the system.

When you add a module to the I/O Configuration folder…



…the software automatically creates controller-scoped tags for the module.

An I/O address follows this format:



Where	Is
Location	Network location LOCAL = same chassis or DIN rail as the controller ADAPTER_NAME = identifies remote communication adapter or bridge module
Slot	Slot number of I/O module in its chassis or DIN rail

Type	Type of data
	I = input
	O = output
	C = configuration
	S = status

Member	Specific data from the I/O module; depends on what type of data the module can store. • For a digital module, a Data member usually stores the input or output bit values. • For an analog module, a Channel member (CH#) usually stores the data for a channel.
SubMember	Specific data related to a Member.
Bit	Specific point on a digital I/O module; depends on the size of the I/O module (0…31 for a 32-point module)

Buffer I/O

Buffering is a technique that logic does not directly reference or manipulate the tags of real I/O devices. Instead, the logic uses a copy of the I/O data. Buffer I/O in the following situations:

- To prevent an input or output value from changing during the execution of a program. (I/O updates asynchronous to the execution of logic.)
- To copy an input or output tag to a member of a structure or element of an array.

Follow these steps to buffer I/O.

1. On the rung before the logic for the function, copy or move the data from the required input tags to their corresponding buffer tags.

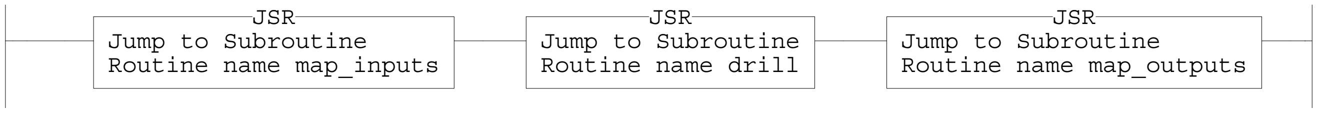
2. In the logic of the function, reference the buffer tags.

3. On the rung after the function, copy the data from the buffer tags to the corresponding output tags.

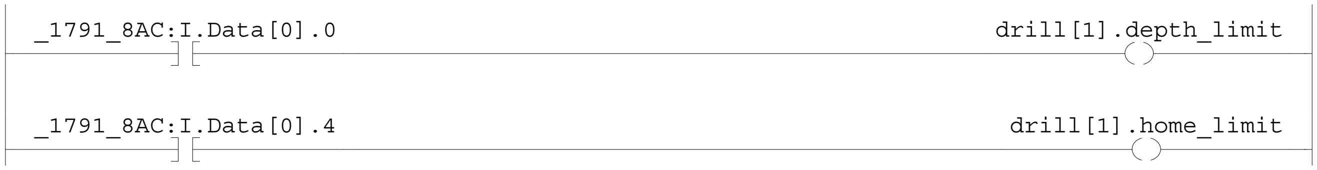
This example copies inputs and outputs to the tags of a structure for a drill machine.

EXAMPLE Buffer I/O

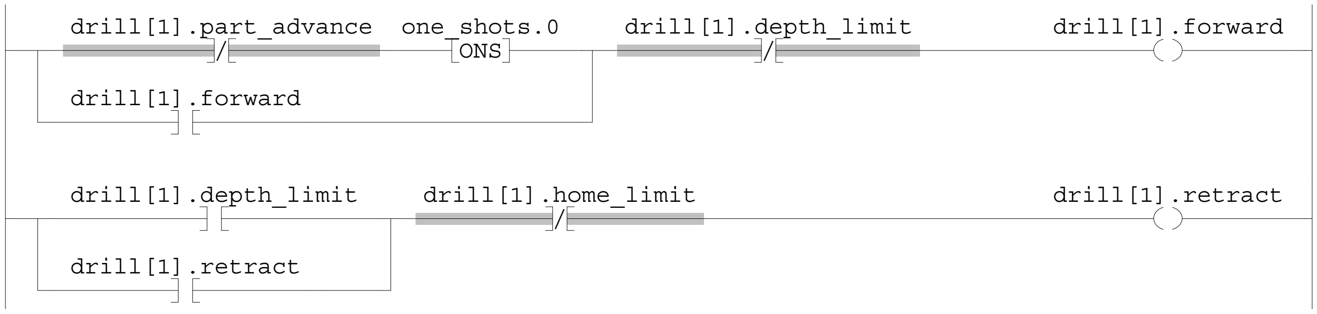
The main routine of the program executes the following subroutines in this sequence.



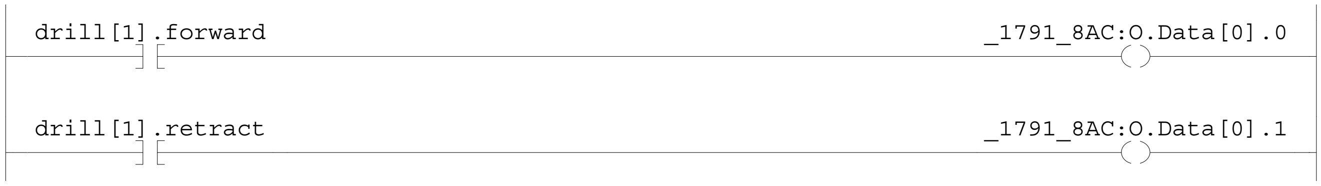
The map_inputs routine copies the values of input devices to their corresponding tags that are used in the drill routine.



The drill routine executes the logic for the drill machine.



The map_outputs routine copies the values of output tags in the drill routine to their corresponding output devices.



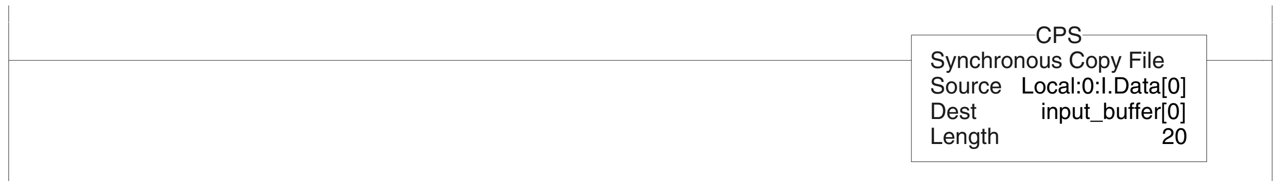
42369

This example uses the CPS instruction to copy an array of data that represent the input devices of a DeviceNet network.

EXAMPLE Buffer I/O

Local:0:I.Data stores the input data for the DeviceNet network that is connected to the 1756-DNB module in slot 0. To synchronize the inputs with the application, the CPS instruction copies the input data to input_buffer.

- While the CPS instruction copies the data, no I/O updates can change the data.
- As the application executes, it uses for its inputs the input data in input_buffer.



42578

Organize Tags

Introduction

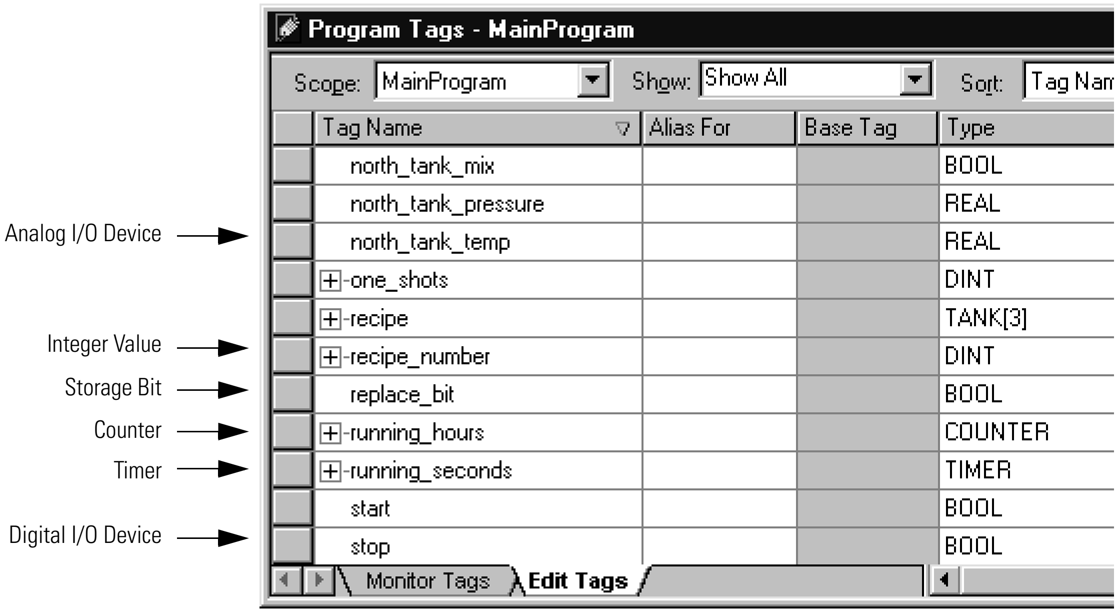
With a Logix5000 controller, you use a tag (alphanumeric name) to address data (variables).

Term	Definition
Tag	A text-based name for an area of the controller's memory where data is stored. • Tags are the basic mechanism for allocating memory, referencing data from logic, and monitoring data. • The minimum memory allocation for a tag is four bytes. • When you create a tag that stores data that requires less than four bytes, the controller allocates four bytes, but the data only fills the part it needs.

The controller uses the tag name internally and doesn't need to cross-reference a physical address.

- In conventional programmable controllers, a physical address identifies each item of data.
  - Addresses follow a fixed, numeric format that depend on the type of data, such as N7:8, F8:3.
  - Symbols are required to make logic easier to interpret.
- In Logix5000 controllers, there is no fixed, numeric format. The tag name itself identifies the data. This lets you:
  - organize your data to mirror your machinery.
  - document (through tag names) your application as you develop it.

EXAMPLE Tags



Scope: MainProgram Show: Show All Sort: Tag Nam

Tag Name	Alias For	Base Tag	Type
north_tank_mix			BOOL
north_tank_pressure			REAL
north_tank_temp			REAL
+-one_shots			DINT
+-recipe			TANK[3]
+-recipe_number			DINT
replace_bit			BOOL
+-running_hours			COUNTER
+-running_seconds			TIMER
start			BOOL
stop			BOOL

Tag Type

The tag type defines how the tag operates within your project.

If you want the tag to	Then choose this type
Store a value or values for use by logic within the project	Base
Represent another tag	Alias
Send data to another controller	Produced
Receive data from another controller	Consumed

If you plan to use produced or consumed tags, you must follow additional guidelines as you organize your tags.

See the Logix5000 Controllers Produced and Consumed Tags Programming Manual, publication 1756-PM011.

Data Type

Term	Definition
Data type	The data type defines the type of data that a tag stores, such as a bit, integer, floating-point value, string, and so forth.
Structure	A data type that is a combination of other data types. • A structure is formatted to create a unique data type that matches a specific need. • Within a structure, each individual data type is called a member. • Like tags, members have a name and data type. • A Logix5000 controller contains a set of predefined structures (data types) for use with specific instructions such as timers, counters, Function Blocks, and so forth. • You can create your own structures, called a user-defined data type.

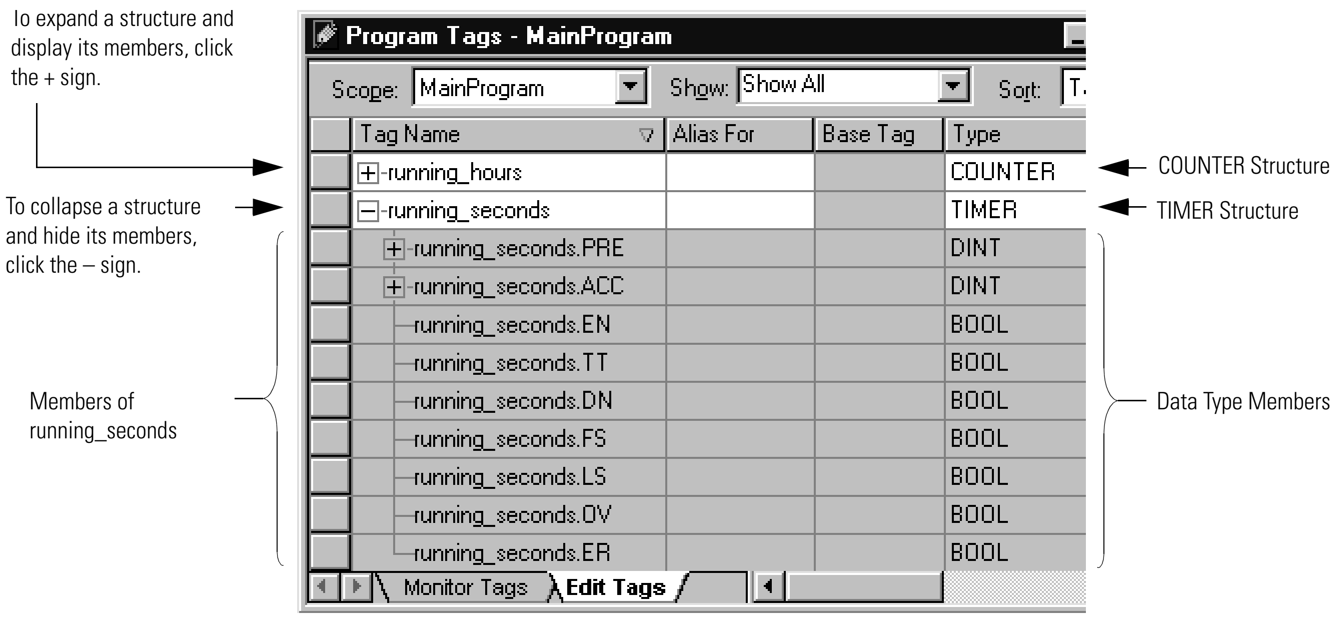
The following table outlines the most common data types and when to use each.

For	Select
Analog device in floating-point mode	REAL
Analog device in integer mode (for very fast sample rates)	INT
ASCII characters	String
Bit	BOOL
Counter	COUNTER
Digital I/O point	BOOL
Floating-point number	REAL
Integer (whole number)	DINT
Sequencer	CONTROL
Timer	TIMER

The minimum memory allocation for a tag is four bytes. When you create a tag that stores data that requires less than four bytes, the controller allocates four bytes, but the data only fills the part it needs.

Data type	**Bits**			
	31 … 16	**15 … 8**	**7 … 1**	**0**
BOOL	Not used			0 or 1
SINT	Not used		-128…+127	
INT	Not used	-32,768…+32,767		
DINT	-2,147,483,648…+2,147,483,647			
REAL	$-3.40282347E^{38}$ …$-1.17549435E^{-38}$ (negative values) 0 $1.17549435E^{-38}$ …$3.40282347E^{38}$ (positive values)			

The COUNTER and TIMER data types are examples of commonly used structures.


To expand a structure and display its members, click the + sign.
To collapse a structure and hide its members, click the – sign.
Members of running_seconds
COUNTER Structure
TIMER Structure
Data Type Members
Program Tags - MainProgram
Scope: MainProgram
Show: Show All
Sort: T
Monitor Tags
Edit Tags

Tag Name	Alias For	Base Tag	Type
+-running_hours			COUNTER
–-running_seconds			TIMER
+-running_seconds.PRE			DINT
+-running_seconds.ACC			DINT
running_seconds.EN			BOOL
running_seconds.TT			BOOL
running_seconds.DN			BOOL
running_seconds.FS			BOOL
running_seconds.LS			BOOL
running_seconds.OV			BOOL
running_seconds.ER			BOOL

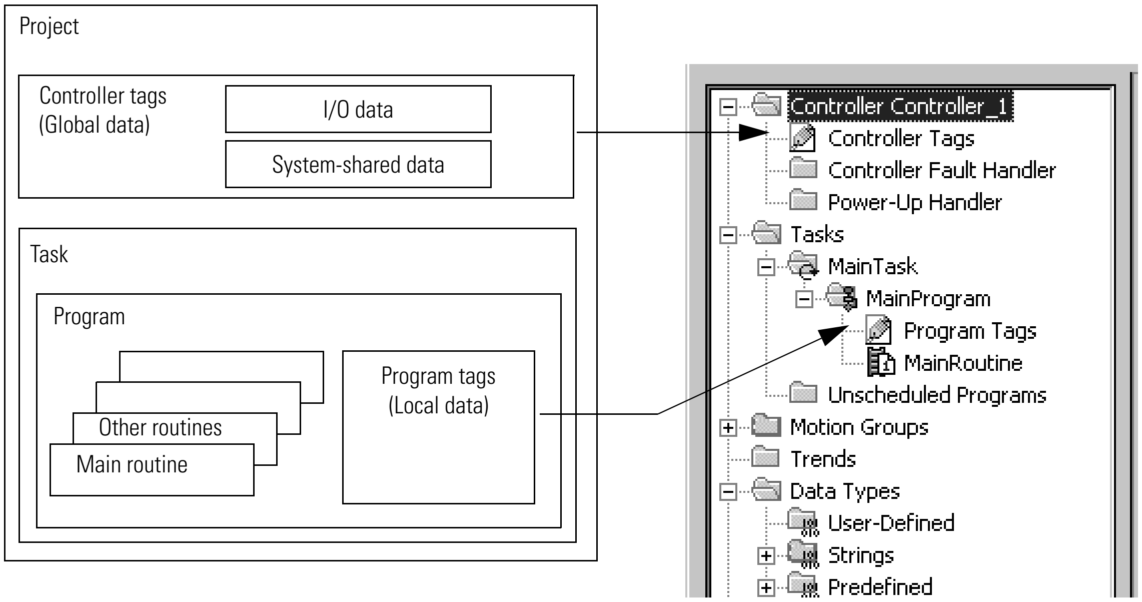
42365

To copy data to a structure, use the COP instruction.

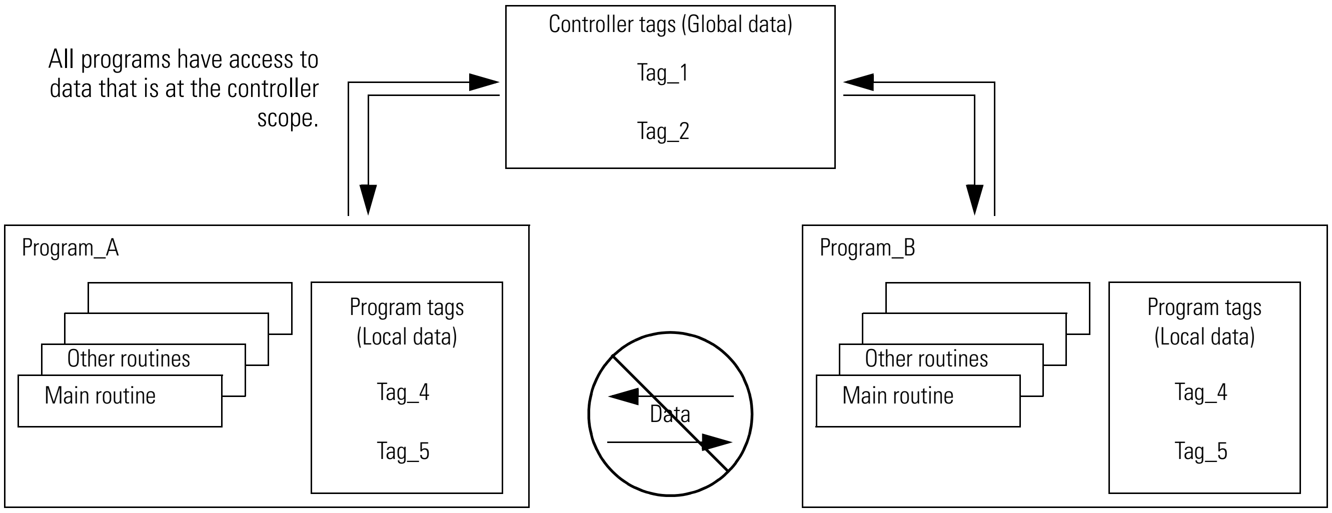
Refer to the Logix5000 Controllers General Instructions Reference Manual, publication 1756-RM003.

Scope

When you create a tag, you define it as either a controller tag (global data) or a program tag for a specific program (local data).



A Logix5000 controller lets you divide your application into multiple programs, each with its own data. There is no need to manage conflicting tag names between programs. This makes it easier to reuse both code and tag names in multiple programs.



Data at the program scope is isolated from other programs.

- Routines cannot access data that is at the program scope of another program.
- You can reuse the tag name of a program-scoped tag in multiple programs.

  For example, both Program_A and Program_B can have a program tag named Tag_4.

Avoid using the same name for both a controller tag and a program tag. Within a program, you cannot reference a controller tag if a tag of the same name exists as a program tag for that program.

Certain tags must be controller scope (controller tag).

Controller Scope Tags

If you want to use the tag	Then assign this scope
In more than one program in the project	Controller scope (controller tags)
In a Message (MSG) instruction	
To produce or consume data	
In any of the seven AXIS data types	
To communicate with a PanelView terminal	
None of the above	Program scope (program tags)

Guidelines for Tags

Use the following guidelines to create tags for a Logix5000 project.

Tag Guidelines

Guideline	Details
Create user-defined data types	User-defined data types (structures) let you organize data to match your machine or process. A user-defined data type provides these advantages: • One tag contains all the data related to a specific aspect of your system. This keeps related data together and easy to locate, regardless of its data type. • Each individual piece of data (member) gets a descriptive name. This automatically creates an initial level of documentation for your logic. • You can use the data type to create multiple tags with the same data layout. For example, use a user-defined data type to store all the parameters for a tank, including temperatures, pressures, valve positions, and preset values. Then create a tag for each of your tanks based on that data type.
Use arrays to quickly create a group of similar tags	An array creates multiple instances of a data type under a common tag name. • Arrays let you organize a block of tags that use the same data type and perform a similar function. • You organize the data in one, two, or three dimensions to match what the data represents. For example, use a two-dimensional array to organize the data for a tank farm. Each element of the array represents a single tank. The location of the element within the array represents the geographic location of the tank. Important: Minimize the use of BOOL arrays. Many array instructions do not operate on BOOL arrays. This makes it more difficult to initialize and clear an array of BOOL data. • Typically, use a BOOL array for the bit-level objects of a PanelView screen. • Otherwise, use the individual bits of a DINT tag or an array of DINTs.

Tag Guidelines

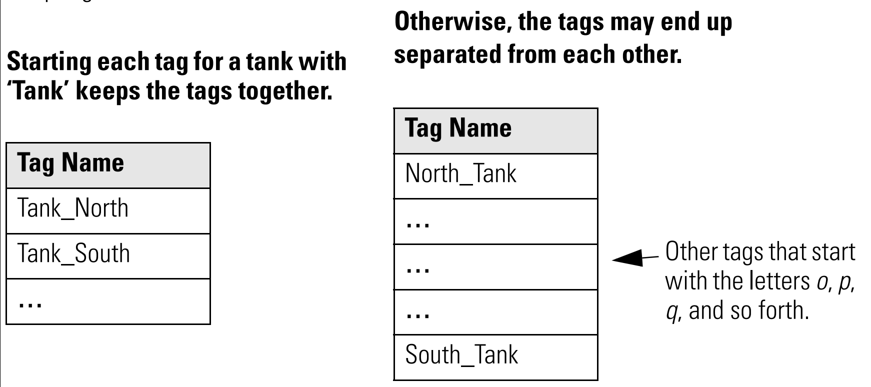
Guideline	Details	
Take advantage of program-scoped tags	If you want multiple tags with the same name, define each tag at the program scope (program tags) for a different program. This lets you reuse both logic and tag names in multiple programs. Avoid using the same name for both a controller tag and a program tag. Within a program, you cannot reference a controller tag if a tag of the same name exists as a program tag for that program. Certain tags must be controller scope (controller tag).	
	If you want the tag	**Then assign this scope**
	In more than one program in the project	Controller scope (controller tags)
	In a Message (MSG) instruction	
	To produce or consume data	
	In any of the seven AXIS data types	
	To communicate with a PanelView terminal	
	None of the above	Program scope (program tags)
For integers, use the DINT data type	To increase the efficiency of your logic, minimize the use of SINT or INT data types. Whenever possible, use the DINT data type for integers. • A Logix5000 controller typically compares or manipulates values as 32-bit values (DINTs or REALs). • The controller typically converts a SINT or INT value to a DINT or REAL value before it uses the value. • If the destination is a SINT or INT tag, the controller typically converts the value back to a SINT or INT value. • The conversion to or from SINTs or INTs occurs automatically with no extra programming. But it takes extra execution time and memory.	
Use most restrictive external access	External access limits the exposure of controller tags by defining a user's ability to edit tags to Read/Write, Read Only and None. This helps: • reduce the risk of inadvertently changing tags. • reduce the number of tags to browse when configuring HMI. See External Access on page 63.	
Enable constant attribute for tags that should not be changed by logic	A constant value can be assigned to a tag to prevent the table-backed data from being changed programmatically. This helps reduce the risk of inadvertently changing tags. See Constant Value Tags on page 79.	

Tag Guidelines

Guideline	Details
Limit a tag name to 40 characters	Here are the rules for a tag name: • Only alphabetic characters (A-Z or a-z), numeric characters (0…9), and underscores (_) • Must start with an alphabetic character or an underscore • No more than 40 characters • No consecutive or trailing underscore characters (_) • Not case sensitive
Use mixed case	Although tags are not case sensitive (upper case *A* is the same as lower case *a*), mixed case is easier to read.

These tags are easier to read	Than these tags
Tank_1	TANK_1
Tank1	TANK1
	tank_1
	tank1

Guideline	Details
Consider the alphabetical order of tags	RSLogix 5000 software displays tags of the same scope in alphabetical order. To make it easier to monitor related tags, use similar starting characters for tags that you want to keep together.



Starting each tag for a tank with 'Tank' keeps the tags together.

Tag Name
Tank_North
Tank_South
…

Otherwise, the tags may end up separated from each other.

Tag Name
North_Tank
…
…
…
South_Tank

Other tags that start with the letters *o*, *p*, *q*, and so forth.

Create a Tag

The Tag Editor window lets you create and edit tags by using a spreadsheet-style view of the tags.

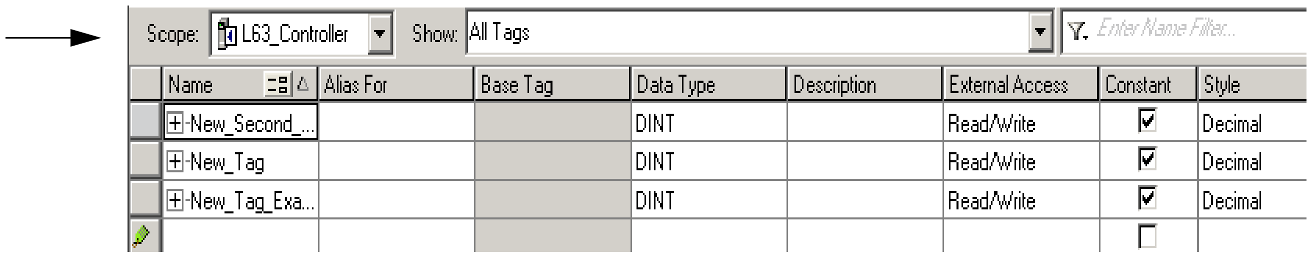
IMPORTANT RSLogix 5000 programming software also automatically creates tags when you:

- add an element to a sequential function chart (SFC).
- add a function block instruction to a function block diagram.

Follow these steps to create a tag by using the RSLogix 5000 programming software.

1. On the Controller Organizer, right-click Controller Tags and choose Edit Tags.

   The Tag Editor window appears.



Scope: L63_Controller Show: All Tags Enter Name Filter...

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
New_Second_...			DINT		Read/Write	✓	Decimal
New_Tag			DINT		Read/Write	✓	Decimal
New_Tag_Exa...			DINT		Read/Write	✓	Decimal

42350

2. Choose a scope for the tag.

If You Use The Tag	Then Select
In more than one program within the project	Name_of_controller
As a producer or consumer	
In any of the seven AXIS data types	
In a message	
In only one program within the project	Program that will use the tag

3. Type a name, data type, and description (optional) for the tag.

4. Specify the External Access and Constant attributes.

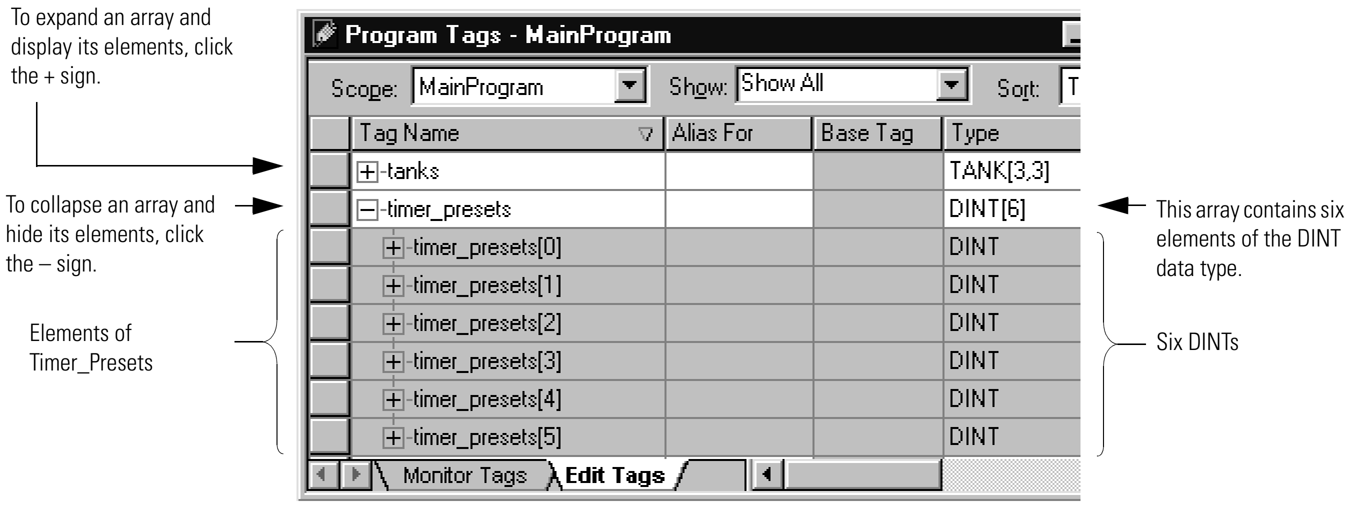
   See Chapter 4 on page 63 for information on the External Access and Constant attributes.

Create an Array

Logix5000 controllers also let you use arrays to organize data.

Term	Definition
Array	A tag that contains a block of multiple pieces of data. • An array is similar to a file. • Within an array, each individual piece of data is called an element. • Each element uses the same data type. • An array tag occupies a contiguous block of memory in the controller, each element in sequence. • You can use array and sequencer instructions to manipulate or index through the elements of an array • You organize the data into a block of one, two, or three dimensions.

A subscript (s) identifies each individual element within the array. A subscript starts at 0 and extends to the number of elements minus 1 (zero based).


To expand an array and display its elements, click the + sign.
To collapse an array and hide its elements, click the – sign.
Elements of Timer_Presets
This array contains six elements of the DINT data type.
Six DINTs
Program Tags - MainProgram
Scope: MainProgram
Show: Show All
Sort: T
Monitor Tags
Edit Tags

Tag Name	Alias For	Base Tag	Type
⊞-tanks			TANK[3,3]
⊟-timer_presets			DINT[6]
⊞-timer_presets[0]			DINT
⊞-timer_presets[1]			DINT
⊞-timer_presets[2]			DINT
⊞-timer_presets[3]			DINT
⊞-timer_presets[4]			DINT
⊞-timer_presets[5]			DINT

42367

The following example compares a structure to an array.

This is a tag that uses the Timer structure (data type).

Tag Name	Data Type
⊟ Timer_1	TIMER
⊞ Timer_1.PRE	DINT
⊞ Timer_1.ACC	DINT
Timer_1.EN	BOOL
Timer_1.TT	BOOL
Timer_1.DN	BOOL

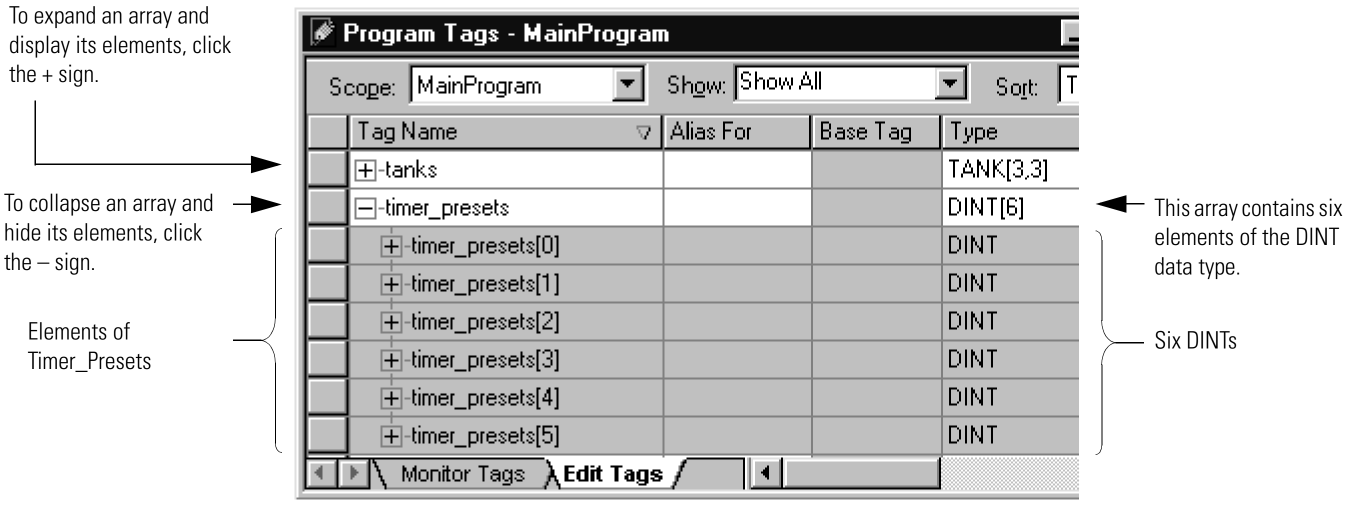
This is a tag that uses an array of the Timer data type.

Tag Name	Data Type
⊟ Timers	TIMER[3]
⊞ Timer[0]	TIMER
⊞ Timer[1]	TIMER
⊞ Timer[2]	TIMER

EXAMPLE

Single-dimension array

In this example, a single timer instruction times the duration of several steps. Each step requires a different preset value. Because all the values are the same data type (DINTs) an array is used.


To expand an array and display its elements, click the + sign.
To collapse an array and hide its elements, click the – sign.
Elements of Timer_Presets
This array contains six elements of the DINT data type.
Six DINTs
Program Tags - MainProgram
Scope: MainProgram
Show: Show All
Sort:
Monitor Tags
Edit Tags

Tag Name	Alias For	Base Tag	Type
⊞-tanks			TANK[3,3]
⊟-timer_presets			DINT[6]
⊞-timer_presets[0]			DINT
⊞-timer_presets[1]			DINT
⊞-timer_presets[2]			DINT
⊞-timer_presets[3]			DINT
⊞-timer_presets[4]			DINT
⊞-timer_presets[5]			DINT

42367

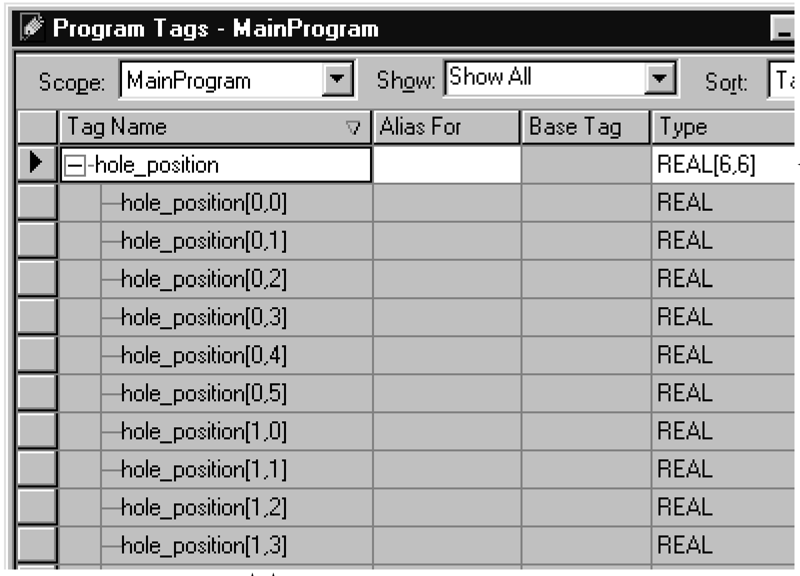
EXAMPLE

Two-dimension array

A drill machine can drill one…five holes in a book. The machine requires a value for the position of each hole from the leading edge of the book. To organize the values into configurations, a two-dimension array is used. The first subscript indicates the hole that the value corresponds and the second subscript indications how many holes will be drilled (one…five).

	Subscript of Second Dimension							**Description**
		0	1	2	3	4	5	
	0							
	1		1.5	2.5	1.25	1.25	1.25	Position of first hole from leading edge of book
Subscript of First Dimension	2			8.0	5.5	3.5	3.5	Position of second hole from leading edge of book
	3				9.75	7.5	5.5	Position of third hole from leading edge of book
	4					9.75	7.5	Position of fourth hole from leading edge of book
	5						9.75	Position of fifth hole from leading edge of book

In the Tags window, the elements are in the order depicted below.



Program Tags - MainProgram

Scope: MainProgram Show: Show All Sort: Ta

Tag Name	Alias For	Base Tag	Type
⊟-hole_position			REAL[6,6]
hole_position[0,0]			REAL
hole_position[0,1]			REAL
hole_position[0,2]			REAL
hole_position[0,3]			REAL
hole_position[0,4]			REAL
hole_position[0,5]			REAL
hole_position[1,0]			REAL
hole_position[1,1]			REAL
hole_position[1,2]			REAL
hole_position[1,3]			REAL

This array contains a two-dimensional grid of elements, six elements x six elements.

42367

The rightmost dimension increments to its maximum value then starts over.

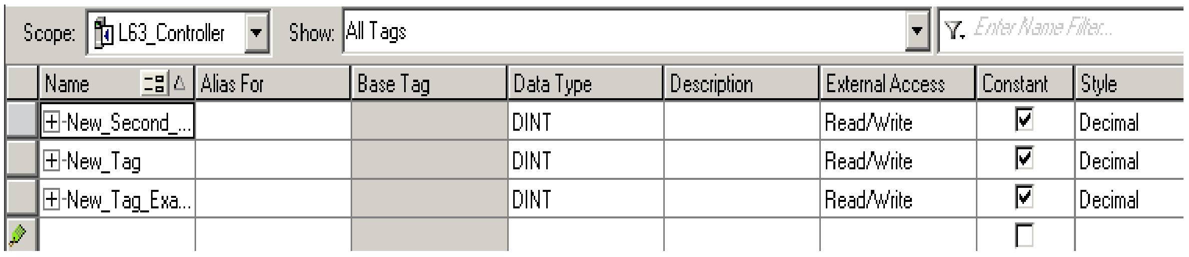
When the rightmost dimension starts over, the dimension to the left increments by one.

Configure an Array

To create an array, you create a tag and assign dimensions to the data type.

1. On the Controller Organizer, right-click Controller Tags and choose Edit Tags.

   The Tag Editor window appears.



Scope: L63_Controller Show: All Tags Enter Name Filter...

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
⊞-New_Second_...			DINT		Read/Write	✓	Decimal
⊞-New_Tag			DINT		Read/Write	✓	Decimal
⊞-New_Tag_Exa...			DINT		Read/Write	✓	Decimal
						☐	

42350

2. Type a name for the tag and select a scope for the tag.

3. Assign the array dimensions.

If the tag is	Then type	Where
One-dimension array	Data_type[x]	Data_type is the type of data that the tag stores. X is the number of **elements** in the first dimension. Y is the number of elements in the second dimension. Z is the number of elements in the third dimension.
Two-dimension array	Data_type[x,y]	
Three-dimension array	Data_type[x,y,z]	

Create a User-defined Data Type

User-defined data types (structures) let you organize your data to match your machine or process.

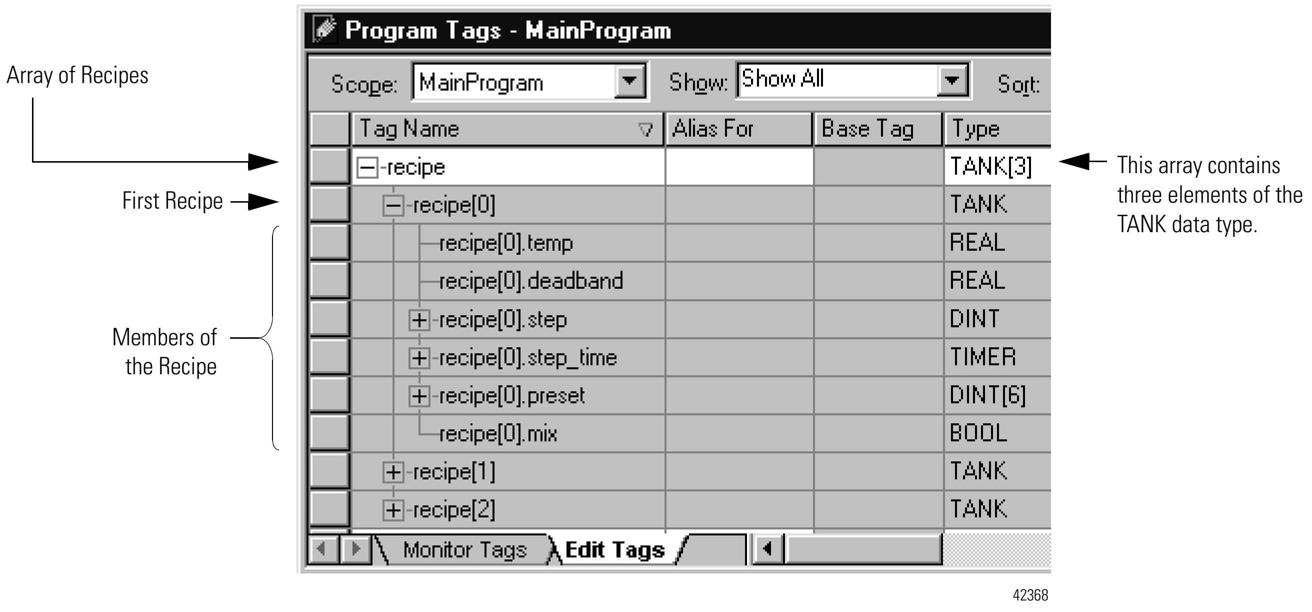
EXAMPLE

User-defined data type that stores a recipe.

In a system of several tanks, each tank can run a variety of recipes. Because the recipe requires a mix of data types (REAL, DINT, BOOL, so forth), a user-defined data type is used.

Name (of data type): TANK	
Member Name	**Data Type**
Temp	REAL
Deadband	REAL
Step	DINT
Step_time	TIMER
Preset	DINT[6]
Mix	BOOL

An array that is based on this data type would look like this example.


Array of Recipes
First Recipe
Members of the Recipe
This array contains three elements of the TANK data type.
Program Tags - MainProgram
Monitor Tags
Edit Tags
42368

Scope: MainProgram Show: Show All Sort:

Tag Name	Alias For	Base Tag	Type
⊟-recipe			TANK[3]
⊟-recipe[0]			TANK
recipe[0].temp			REAL
recipe[0].deadband			REAL
⊞-recipe[0].step			DINT
⊞-recipe[0].step_time			TIMER
⊞-recipe[0].preset			DINT[6]
recipe[0].mix			BOOL
⊞-recipe[1]			TANK
⊞-recipe[2]			TANK

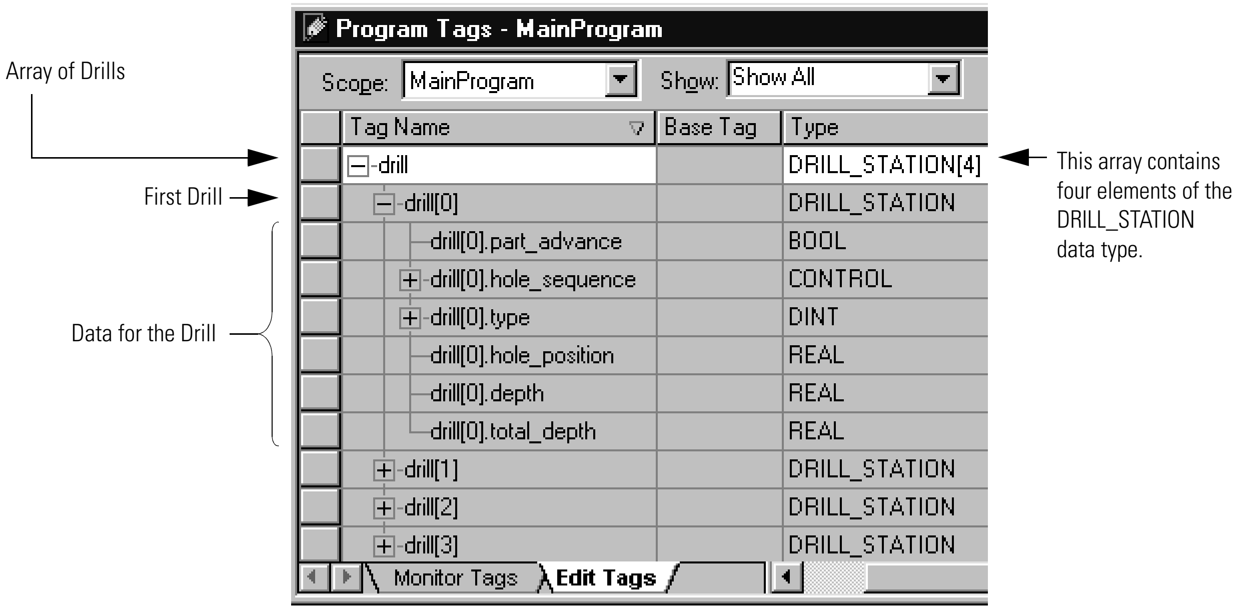
EXAMPLE

User-defined data type that stores the data that is required to run a machine.

Because several drill stations require the following mix of data, use a user-defined data type.

Name (of data type): DRILL_STATION	
Member Name	**Data Type**
Part_advance	BOOL
Hole_sequence	CONTROL
Type	DINT
Hole_position	REAL
Depth	REAL
Total_depth	REAL

An array that is based on this data type looks like this example.



Tag Name	Base Tag	Type
drill		DRILL_STATION[4]
drill[0]		DRILL_STATION
drill[0].part_advance		BOOL
drill[0].hole_sequence		CONTROL
drill[0].type		DINT
drill[0].hole_position		REAL
drill[0].depth		REAL
drill[0].total_depth		REAL
drill[1]		DRILL_STATION
drill[2]		DRILL_STATION
drill[3]		DRILL_STATION

42583

Guidelines for User-defined Data Types

When you create a user-defined data type, use these guidelines:

- If you include members that represent I/O devices, you must use logic to copy the data between the members in the structure and the corresponding I/O tags. Refer to Address I/O Data on page 21.
- If you include an array as a member, limit the array to a single dimension. Multi-dimension arrays are *not* permitted in a user-defined data type.
- When you use the BOOL, SINT, or INT data types, place members that use the same data type in sequence.

More Efficient

BOOL
BOOL
BOOL
DINT
DINT

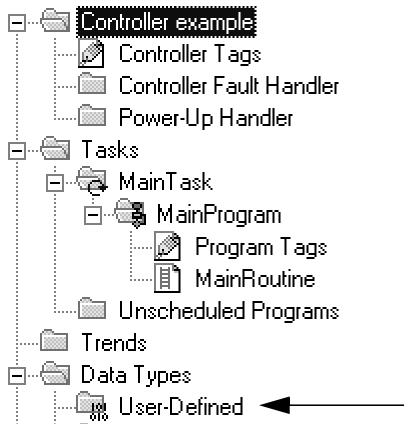
Less Efficient

BOOL
DINT
BOOL
DINT
BOOL

Create a User-defined Data Type

1. On the Controller Organizer from the User-defined folder under Data Types, right-click User-Defined.

2. Choose New Data Type.

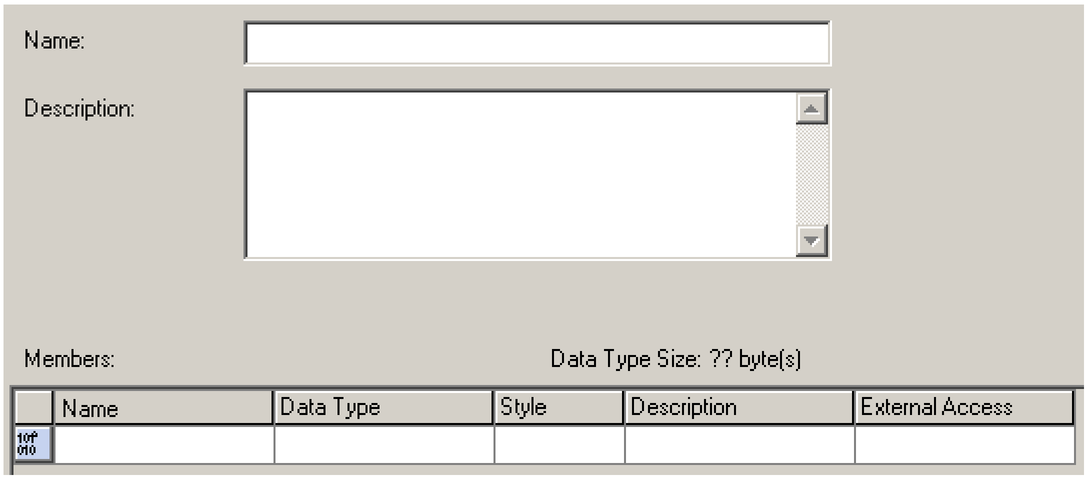


3. Type a name and description for the user-defined data type.

   A description is optional.

4. For each member of the user-defined data type, type a name, data type, style, and description.

5. Click the External Access column, and choose an attribute.



Name:

Description:

Members:

Data Type Size: ?? byte(s)

	Name	Data Type	Style	Description	External Access

42196

   Limit any arrays to a single dimension.

   To display the value of the member in a different style (radix), select the style.

6. Click Apply.

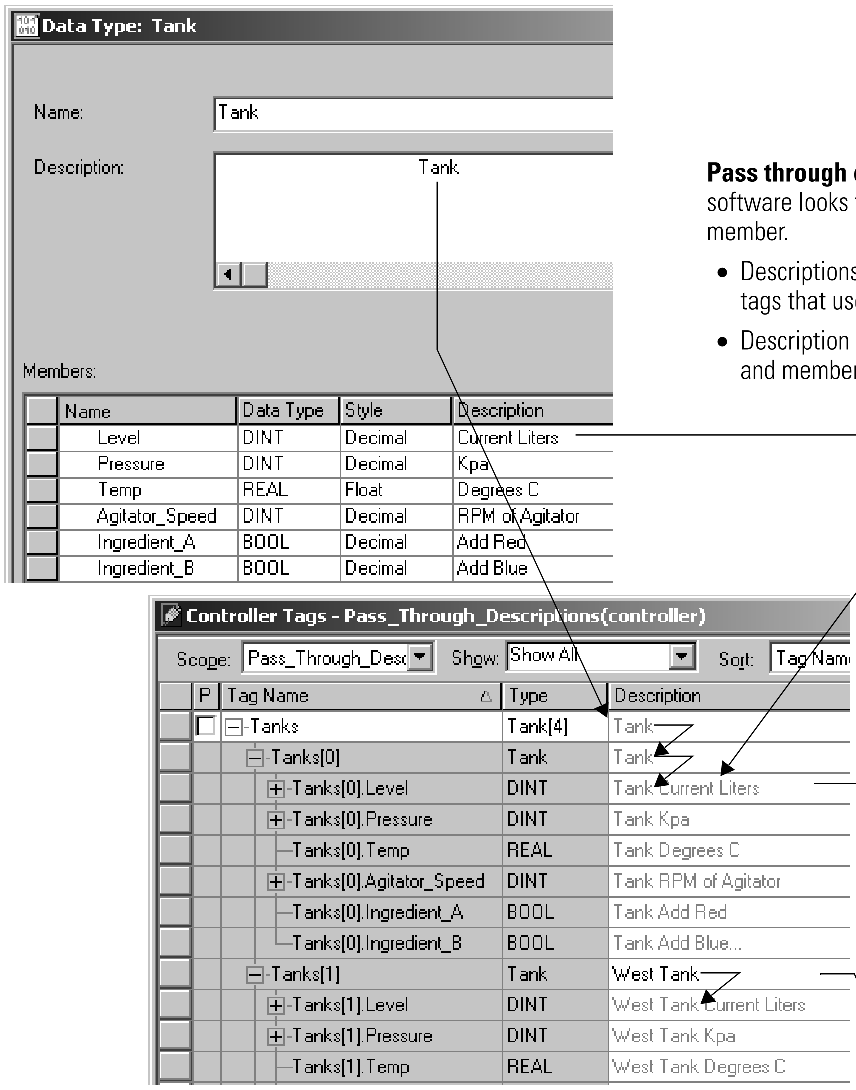
7. Add as many members as needed.

Describe a User-defined Data Type

RSLogix 5000 software 13.0 or later

RSLogix 5000 programming software lets you automatically build descriptions out of the descriptions in your user-defined data types. This greatly reduces the amount of time you have to spend documenting your project.

As you organize your user-defined data types, keep in mind the following features of RSLogix 5000 software.



Data Type: Tank

Name: Tank

Description: Tank

Members:

Name	Data Type	Style	Description
Level	DINT	Decimal	Current Liters
Pressure	DINT	Decimal	Kpa
Temp	REAL	Float	Degrees C
Agitator_Speed	DINT	Decimal	RPM of Agitator
Ingredient_A	BOOL	Decimal	Add Red
Ingredient_B	BOOL	Decimal	Add Blue

Controller Tags - Pass_Through_Descriptions(controller)

Scope: Pass_Through_Desc Show: Show All Sort: Tag Nam

P	Tag Name	Type	Description
	-Tanks	Tank[4]	Tank
	-Tanks[0]	Tank	Tank
	+-Tanks[0].Level	DINT	Tank Current Liters
	+-Tanks[0].Pressure	DINT	Tank Kpa
	-Tanks[0].Temp	REAL	Tank Degrees C
	+-Tanks[0].Agitator_Speed	DINT	Tank RPM of Agitator
	-Tanks[0].Ingredient_A	BOOL	Tank Add Red
	-Tanks[0].Ingredient_B	BOOL	Tank Add Blue...
	-Tanks[1]	Tank	West Tank
	+-Tanks[1].Level	DINT	West Tank Current Liters
	+-Tanks[1].Pressure	DINT	West Tank Kpa
	-Tanks[1].Temp	REAL	West Tank Degrees C

Pass through of descriptions – When possible, RSLogix 5000 software looks for an available description for a tag, element, or member.

- Descriptions in user-defined data types ripple through to the tags that use that data type.
- Description of an array tag ripples through to the elements and members of the array.

Append description to base tag – RSLogix 5000 software automatically builds a description for each member of a tag that uses a user-defined data type. It starts with the description of the tag and then adds the description of the member from the data type.

Paste pass-through description – Use the data type and array description as a basis for more specific descriptions.

In this example, Tank became West Tank.

RSLogix 5000 software uses different colors for descriptions.

Color Descriptions

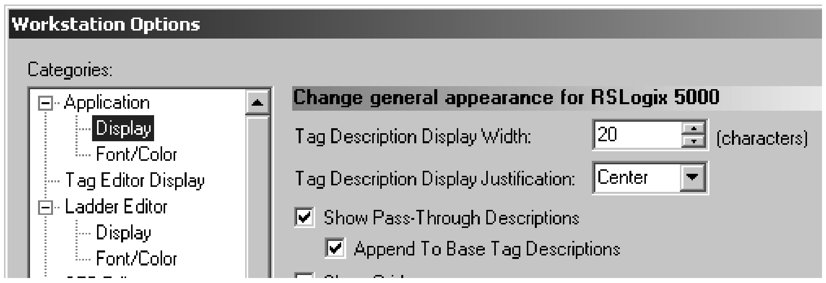
If Color Description Is	This Is
Gray	Pass-through description
Black	Manually entered description

Activate Pass-Through and Append Descriptions

Follow these steps to use pass-through descriptions and append to base tag descriptions.

1. In the RSLogix 5000 programming software, from the Tools menu choose Options.

   The Work Station Options screen appears.



Change general appearance for RSLogix 5000

Tag Description Display Width: 20 (characters)

Tag Description Display Justification: Center

[✓] Show Pass-Through Descriptions

[✓] Append To Base Tag Descriptions

2. Under Application, select Display.

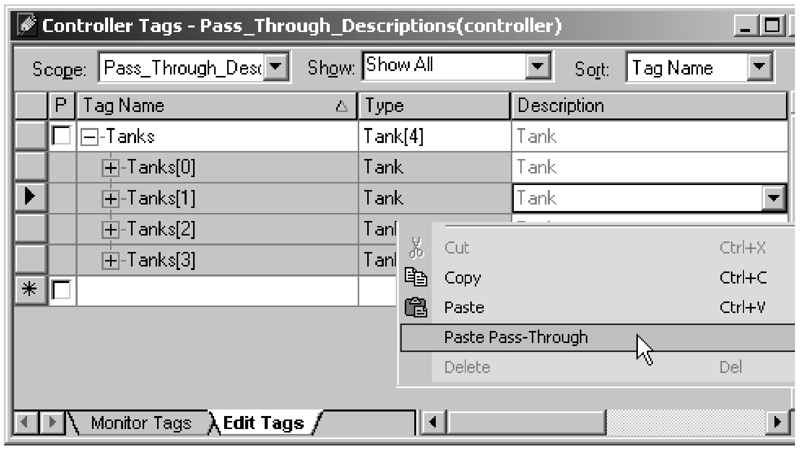
3. Check Show Pass-Through Descriptions and Append to Base Tag Descriptions.

4. Click OK.

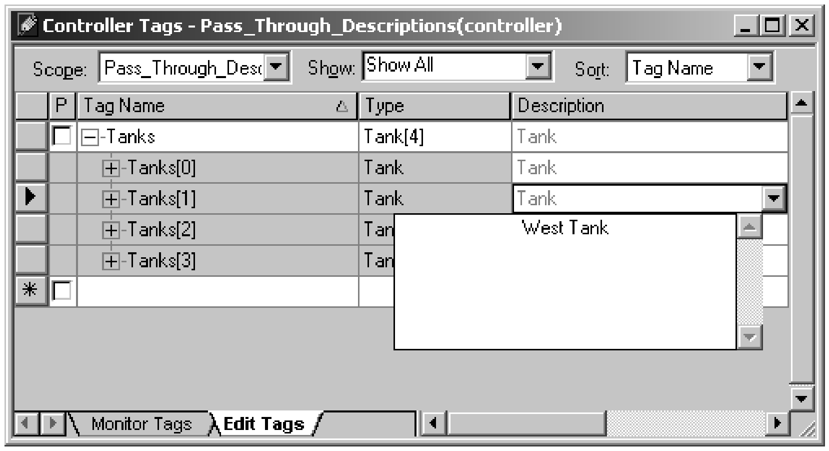
Paste a Pass-Through Description

Follow these steps to use a pass-through description as the starting point for a more specific description.

1. On the Controller Tags screen, right-click the pass-through description, and choose Paste Pass-Through.



2. Edit the description and press CTRL + Enter.



Address Tag Data

A tag name follows this format.

Name *[Element]* *.Member* *[Element]* *.Bit*
or
.[Index]

▨ = Optional

Where	Is
Name	Name that identifies this specific tag.
Element	Subscript or subscripts that point to a specific element within an array. • Use the element identifier only if the tag or member is an array. • Use one subscript for each dimension of the array. For example: [5], [2,8], [3,2,7]. To indirectly (dynamically) reference an element, use a tag or numeric expression that provides the element number. • A numeric expression uses a combination of tags, constants, operators, and functions to calculate a value. For example, Tag_1-Tag_2, Tag_3+4, ABS (Tag_4). • Keep the value of the tag or numeric expression within the dimensions of the array. For example, if a dimension of an array contains 10 elements, then the value of the tag or numeric expression must be 0…9 (10 elements).
Member	Specific member of a structure. • Use the member identifier only if the tag is a structure. • If the structure contains another structure as one of its members, use additional levels of the.Member format to identify the required member.
Bit	Specific bit of an integer data type (SINT, INT, or DINT).
Index	To indirectly (dynamically) reference a bit of an integer, use a tag or numeric expression that provides the bit number. • A numeric expression uses a combination of tags, constants, operators, and functions to calculate a value. For example, Tag_1-Tag_2, Tag_3+4, ABS(Tag_4). • Keep the value of the tag or numeric expression within the range of bits of the integer tag. For example, if the integer tag is a Dint (32-bits), then the value of the index must be 0…31 (32-bits).

Alias Tags

An alias tag lets you create one tag that represents another tag.

- Both tags share the same value.
- When the value of one of the tags changes, the other tag reflects the change as well.

Use aliases in the following situations:

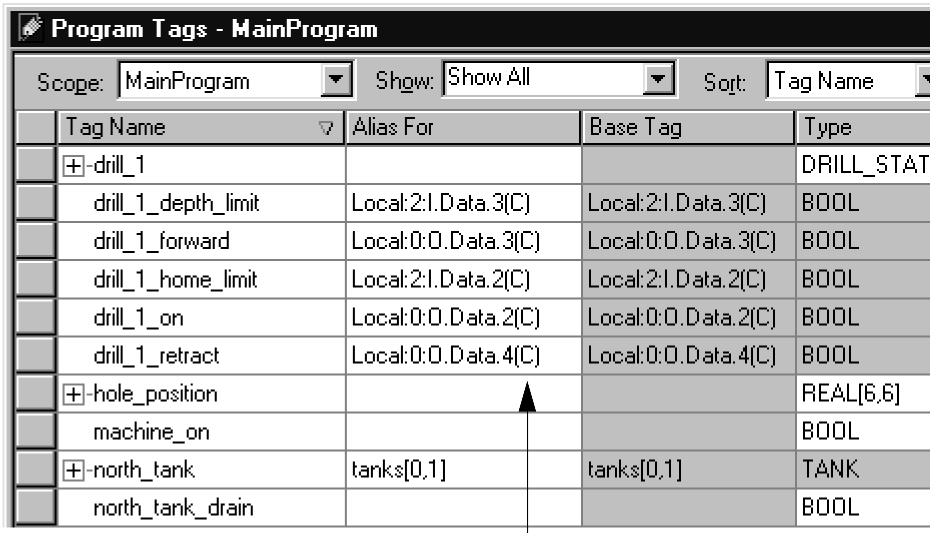
- Program logic in advance of wiring diagrams.
- Assign a descriptive name to an I/O device.
- Provide a more simple name for a complex tag.
- Use a descriptive name for an element of an array.

The tags window displays alias information.

drill_1_depth_limit is an alias for Local:2:I.Data.3 (a digital input point). When the input turns on, the alias tag also turns on.

drill_1_on is an alias for Local:0:O.Data.2 (a digital output point). When the alias tag turns on, the output tag also turns on.

north_tank is an alias for tanks[0,1].



Tag Name	Alias For	Base Tag	Type
⊞-drill_1			DRILL_STAT
drill_1_depth_limit	Local:2:I.Data.3(C)	Local:2:I.Data.3(C)	BOOL
drill_1_forward	Local:0:O.Data.3(C)	Local:0:O.Data.3(C)	BOOL
drill_1_home_limit	Local:2:I.Data.2(C)	Local:2:I.Data.2(C)	BOOL
drill_1_on	Local:0:O.Data.2(C)	Local:0:O.Data.2(C)	BOOL
drill_1_retract	Local:0:O.Data.4(C)	Local:0:O.Data.4(C)	BOOL
⊞-hole_position			REAL[6,6]
machine_on			BOOL
⊞-north_tank	tanks[0,1]	tanks[0,1]	TANK
north_tank_drain			BOOL

42360

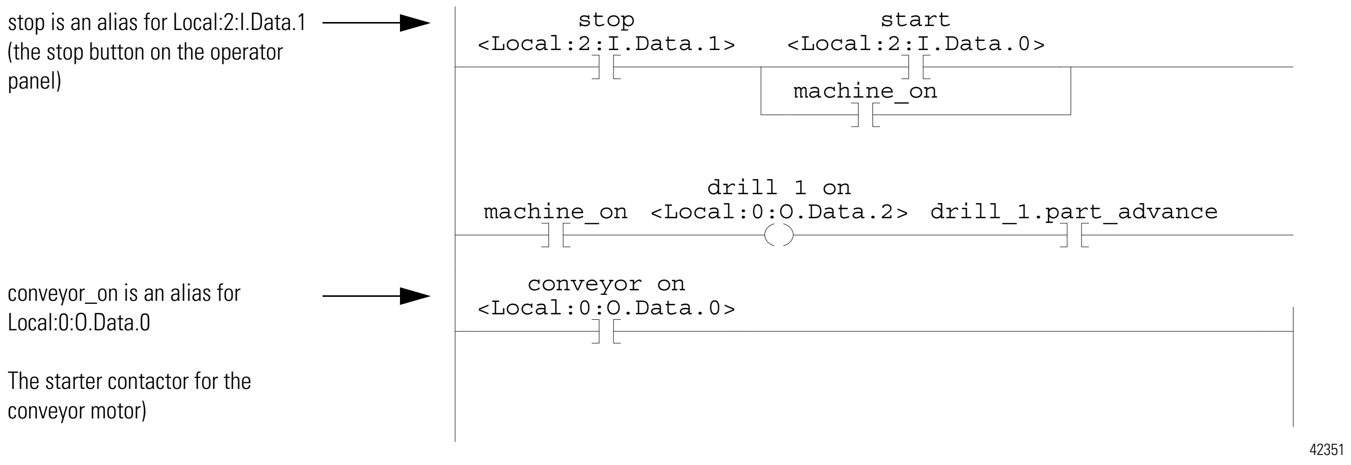
The (C) indicates that the tag is at the controller scope.

A common use of alias tags is to program logic before wiring diagrams are available.

1. For each I/O device, create a tag with a name that describes the device, such as conveyor for the conveyor motor.
2. Program your logic by using the descriptive tag names.

   You can even test your logic without connecting to the I/O.
3. Later, when wiring diagrams are available, add the I/O modules to the I/O configuration of the controller.
4. Finally, convert the descriptive tags to aliases for their respective I/O points or channels.

The following logic was initially programmed by using descriptive tag names, such as stop and conveyor_on. Later, the tags were converted to aliases for the corresponding I/O devices.


stop is an alias for Local:2:I.Data.1 (the stop button on the operator panel)
conveyor_on is an alias for Local:0:O.Data.0
The starter contactor for the conveyor motor)
stop
<Local:2:I.Data.1>
start
<Local:2:I.Data.0>
machine_on
drill 1 on
machine_on
<Local:0:O.Data.2>
drill_1.part_advance
conveyor on
<Local:0:O.Data.0>
42351

Display Alias Information

Follow these steps to show (in your logic) the tag to which an alias points.

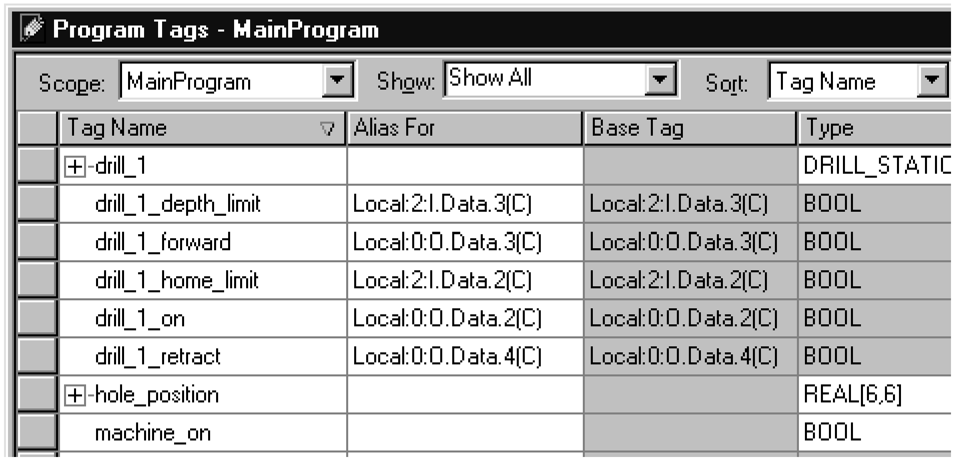
1. From the Tools menu, choose Options.
2. Click the Ladder Display tab.
3. Check Show Tag Alias Information.
4. Click OK.

Assign an Alias

Follow these steps to assign a tag as an alias tag for another tag.

1. On the Controller Organizer, right-click Controller Tags and choose Edit Tags.

   The Tag Editor window appears.



Program Tags - MainProgram

Scope: MainProgram Show: Show All Sort: Tag Name

Tag Name	Alias For	Base Tag	Type
⊞-drill_1			DRILL_STATIC
drill_1_depth_limit	Local:2:I.Data.3(C)	Local:2:I.Data.3(C)	BOOL
drill_1_forward	Local:0:O.Data.3(C)	Local:0:O.Data.3(C)	BOOL
drill_1_home_limit	Local:2:I.Data.2(C)	Local:2:I.Data.2(C)	BOOL
drill_1_on	Local:0:O.Data.2(C)	Local:0:O.Data.2(C)	BOOL
drill_1_retract	Local:0:O.Data.4(C)	Local:0:O.Data.4(C)	BOOL
⊞-hole_position			REAL[6,6]
machine_on			BOOL

42360

2. Select the scope of the tag.

3. To the right of the tag name, click the 'Alias For' cell.

   The cell displays a ▼.

4. Click ▼.

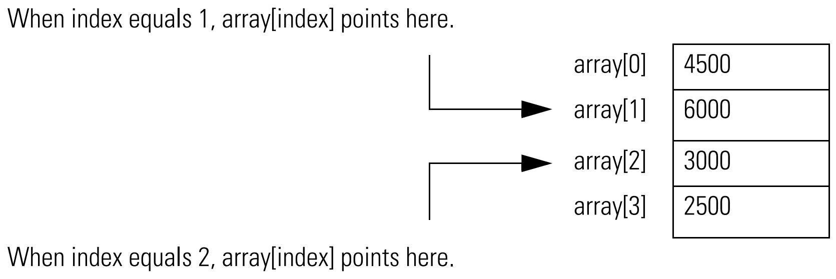
5. Choose the tag that the alias will represent.

To	Do This
Select a tag	Double-click the tag name.
Select a bit number	A. Click the tag name. B. To the right of the tag name, click ▼. C. Click the required bit.

6. Click another cell.

Assign an Indirect Address

If you want an instruction to access different elements in an array, use a tag in the subscript of the array (an indirect address). By changing the value of the tag, you change the element of the array that your logic references.


When index equals 1, array[index] points here.
array[0]
4500
array[1]
6000
array[2]
3000
array[3]
2500
When index equals 2, array[index] points here.

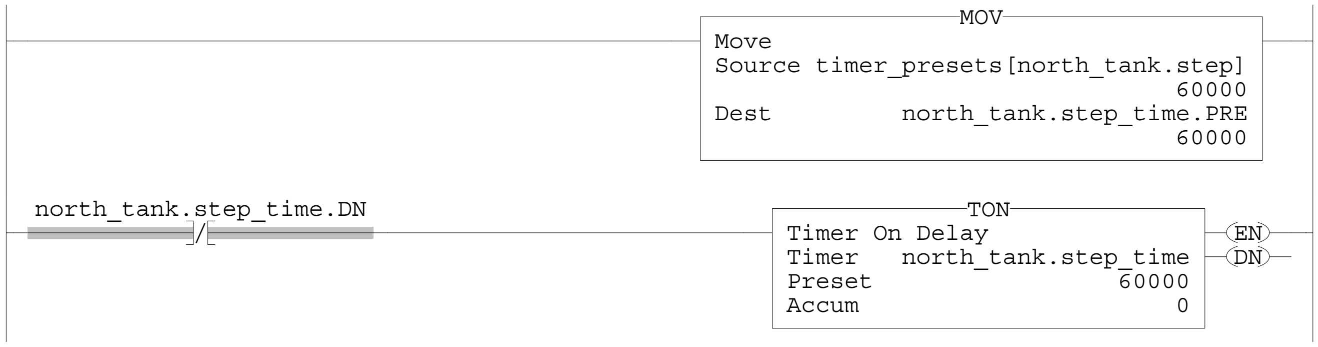
The following table outlines some common uses for an indirect address.

To	Use a tag in the subscript and
Select a recipe from an array of recipes	Enter the number of the recipe in the tag.
Load a specific machine setup from an array of possible setups	Enter the desired setup in the tag.
Load parameters or states from an array, one element at a time	A. Perform the required action on the first element. B. Use an ADD instruction to increment the tag value and point to the next element in the array.
Log error codes	
Perform several actions on an array element and then index to the next element	

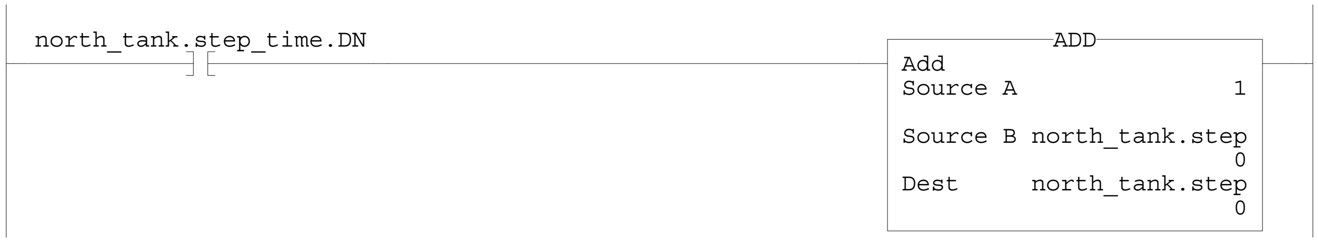
The following example loads a series of preset values into a timer, one value (array element) at a time.

EXAMPLE Step through an array.

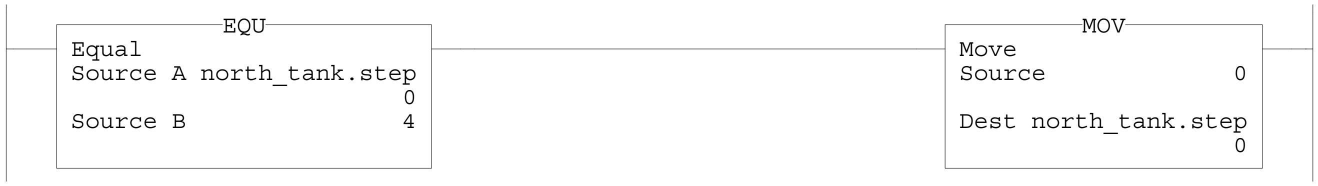
The timer_presets array stores a series of preset values for the timer in the next rung. The north_tank.step tag points to which element of the array to use. For example, when north_tank.step equals 0, the instruction loads timer_presets[0] into the timer (60,000 ms).


MOV
Move
Source timer_presets[north_tank.step]
60000
Dest north_tank.step_time.PRE
60000
north_tank.step_time.DN
TON
Timer On Delay
Timer north_tank.step_time
Preset 60000
Accum 0
EN
DN

When north_tank.step_time is done, the rung increments north_tank.step to the next number and that element of the timer_presets array loads into the timer.



When north_tank.step exceeds the size of the array, the rung resets the tag to start at the first element in the array. (The array contains elements 0…3.)



42358

Expressions

You can also use an expression to specify the subscript of an array.

- An expression uses operators, such as + or -, to calculate a value.
- The controller computes the result of the expression and uses it as the array subscript.

You can use these operators to specify the subscript of an array.

Operator	Description
+	Add
-	Subtract/negate
*	Multiply
/	Divide
ABS	Absolute value
AND	AND
FRD	BCD to integer

Operator	Description
MOD	Modulo
NOT	Complement
OR	OR
SQR	Square root
TOD	Integer to BCD
TRN	Truncate
XOR	Exclusive OR

Format your expressions as follows.

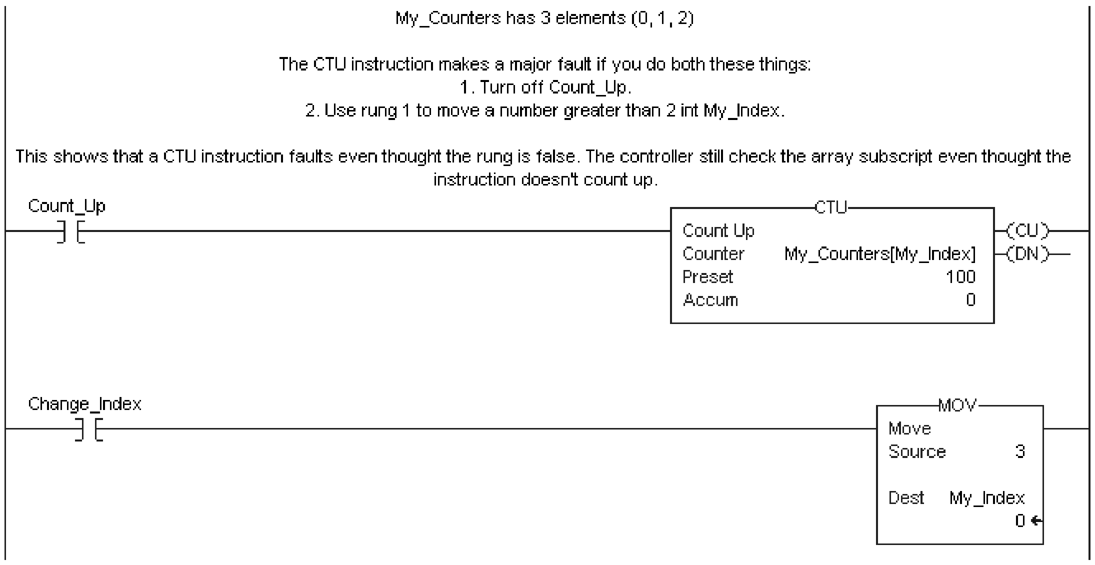
Format Expressions

If the operator requires	Use this format	Example
One value (tag or expression)	operator(value)	ABS(tag_a)
Two values (tags, constants, or expressions)	value_a operator value_b	• tag_b + 5 • tag_c AND tag_d • (tag_e ** 2) MOD (tag_f / tag_g)

Array Subscript Out of Range

Every instruction generates a major fault if the array subscript is out of range. Transitional instructions also generate a major fault even if the rung is false. The controller checks the array subscript in these instructions even if the rung is false.

EXAMPLE


My_Counters has 3 elements (0, 1, 2)
The CTU instruction makes a major fault if you do both these things:
1. Turn off Count_Up.
2. Use rung 1 to move a number greater than 2 int My_Index.
This shows that a CTU instruction faults even thought the rung is false. The controller still check the array subscript even thought the instruction doesn't count up.
Count_Up
CTU
Count Up
Counter My_Counters[My_Index]
Preset 100
Accum 0
CU
DN
Change_Index
MOV
Move
Source 3
Dest My_Index
0

For more information on handling major faults, refer to the Logix5000 Controllers Major and Minor Faults Programming Manual, publication 1756-PM014.

Tag Documentation

The table outlines the four types of tags that can be created and the descriptions that you can document for each one.

IMPORTANT RSLogix 5000 programming software automatically assigns what are called pass-through descriptions of the tags you have created, descriptions you may or may not want to use.

Tag	Description
Base	When you create a tag without specifying a tag type, RSLogix 5000 automatically assigns your tag a default type of Base. Since base tags enable you to create your own internal data storage, you can document in your tag description the nature of the data being stored.
Alias	By creating an Alias tag, you can assign your own name to an existing tag, structure tag member, or bit. In the description of your Alias tag, you can describe the tag that your alias tag references.
Produced	A Produced tag refers to a tag that is consumed by another controller. In the description of your Produced tag, you can describe the remote controllers that you want to make your Produced tag available through controller-to-controller messaging.
Consumed	A Consumed tag refers to a tag that is produced by another controller and whose data you want to use in your controller. In the description of your Consumed tag, you can describe how you want to use a produced tag's data or the data-producing controller.

Language Switching

With RSLogix 5000 software, version 17 and later, you have the option to display project documentation, such as tag descriptions and rung comments for any supported localized language. You can store project documentation for multiple languages in a single project file rather than in language-specific project files. You define all the localized languages that the project will support and set the current, default, and optional custom localized language. The software uses the default language if the current language's content is blank for a particular component of the project. However, you can use a custom language to tailor documentation to a specific type of project file user.

Enter the localized descriptions in your RSLogix 5000 project, either when programming in that language or by using the import/export utility to translate the documentation off-line and then import it back into the project. Once you enable language switching in RSLogix 5000 software, you can dynamically switch between languages as you use the software.

Project documentation that supports multiple translations within a project includes the following:

- Component descriptions in tags, routines, programs, user-defined data types, and Add-On Instructions.
- Equipment phases.
- Trends.
- Controllers.
- Alarm Messages (in ALARM_ANALOG and ALARM_DIGITAL configuration).
- Tasks.
- Property descriptions for modules in the Controller Organizer.
- Rung comments, SFC text boxes, and FBD text boxes.

For more information on enabling a project to support multiple translations of project documentation, see the online help.

Force I/O

Introduction

Use a force to override data that your logic either uses or produces. For example, use forces to:

- test and debug your logic.
- check wiring to an output device.
- temporarily keep your process functioning when an input device has failed.

Use forces only as a temporary measure. They are not intended to be a permanent part of your application.

Precautions

When you use forces, take these precautions.

ATTENTION

Forcing can cause unexpected machine motion that could injure personnel. Before you use a force, determine how the force will effect your machine or process and keep personnel away from the machine area.

- Enabling I/O forces causes input, output, produced, or consumed values to change.
- Enabling SFC forces causes your machine or process to go to a different state or phase.
- Removing forces may still leave forces in the enabled state.
- If forces are enabled and you install a force, the new force immediately takes effect.

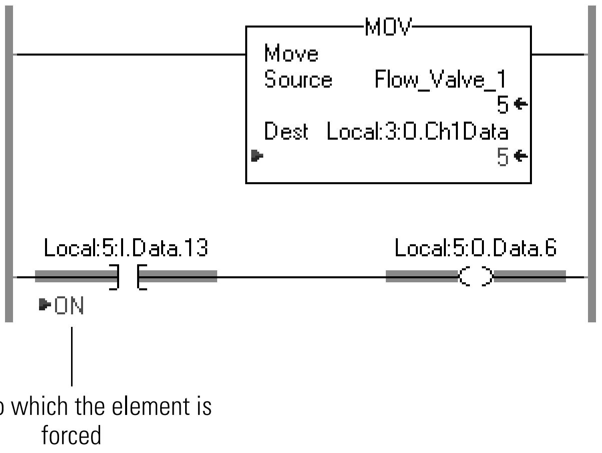
Enable Forces

For a force to take effect, you enable forces. You can only enable and disable forces at the controller level.

- You can enable I/O forces and SFC forces separately or at the same time.
- You cannot enable or disable forces for a specific module, tag collection, or tag element.

IMPORTANT If you download a project that has forces enabled, the programming software prompts you to enable or disable forces after the download completes.

When forces are in effect (enabled), a ▶ appears next to the forced element.



Disable or Remove a Force

To stop the effect of a force and let your project execute as programmed, disable or remove the force.

- You can disable or remove I/O and SFC forces at the same time or separately.
- Removing a force on an alias tag also removes the force on the base tag.

ATTENTION Changes to forces can cause unexpected machine motion that could injure personnel. Before you disable or remove forces, determine how the change will effect your machine or process and keep personnel away from the machine area.

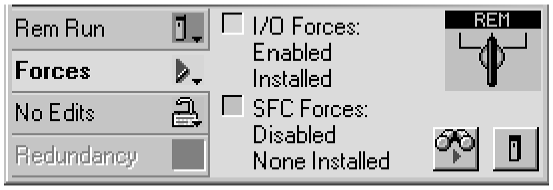
Check Force Status

Before you use a force, determine the status of forces for the controller. You can check force status.

To determine status	Use any of the following
I/O forces	• Online toolbar • FORCE status indicator • GSV instruction
SFC forces	Online toolbar

The Online toolbar shows the status of forces. It shows the status of I/O forces and SFC forces separately.

Forces tab



This	Means
Enabled	• If the project contains any forces of this type, they **are** overriding your logic. • If you add a force of this type, the new force immediately takes effect
Disabled	Forces of this type are inactive. If the project contains any forces of this type, they **are not** overriding your logic.
Installed	At least one force of this type exists in the project.
None Installed	No forces of this type exist in the project.

FORCE Status Indicator

If your controller has a FORCE Status Indicator, use it to determine the status of any I/O forces.

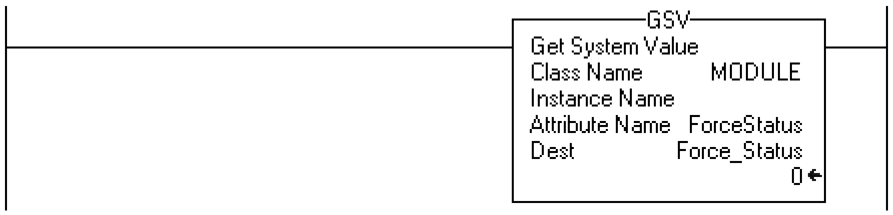
IMPORTANT The FORCE Status Indicator shows only the status of I/O forces. It does not show that status of SFC forces.

FORCE Status Indicator	Then
Off	• No tags contain force values. • I/O forces are inactive (disabled).
Flashing	• At least one tag contains a force value. • I/O forces are inactive (disabled).
Solid	• I/O forces are active (enabled). • Force values may or may not exist.

GSV Instruction

IMPORTANT The ForceStatus attribute shows only the status of I/O forces. It does not show the status of SFC forces.

This example shows how to use a GSV instruction to get the status of forces.



where:

Force_Status is a DINT tag.

To determine if	Examine this bit	For this value
Forces are installed	0	1
No forces are installed	0	0
Forces are enabled	1	1
Forces are disabled	1	0

When to Use I/O Force

Use an I/O force to:

- override an input value from another controller (that is, a consumed tag).
- override an input value from an input device.
- override your logic and specify an output value for another controller (that is, a produced tag).
- override your logic and specify the state of an output device.

IMPORTANT Forcing increases logic execution time. The more values you force, the longer it takes to execute the logic.

IMPORTANT I/O forces are held by the controller and not by the programming workstation. Forces remain even if the programming workstation is disconnected.

Use these guidelines when forcing an I/O value.

- You can force all I/O data, except for configuration data.
- If the tag is an array or structure, such as an I/O tag, force a BOOL, SINT, INT, DINT, or REAL element or member.
- If the data value is a SINT, INT, or DINT, you can force the entire value or you can force individual bits within the value. Individual bits can have a force status of:
  - No force
  - Force on
  - Force off
- You can also force an alias to an I/O structure member, produced tag, or consumed tag.
  - An alias tag shares the same data value as its base tag, so forcing an alias tag also forces the associated base tag.
  - Removing a force from an alias tag removes the force from the associated base tag.
- If a produced tag is also Constant, you cannot use forces.
- If a produced tag is forced, you cannot make it Constant.

Force an Input Value

Forcing an input or consumed tag:

- overrides the value regardless of the value of the physical device or produced tag.
- does not affect the value received by other controllers monitoring that input or produced tag.

Force an Output Value

Forcing an output or produced tag overrides the logic for the physical device or other controller. Other controllers monitoring that output module in a listen-only capacity will also see the forced value.

Add an I/O Force

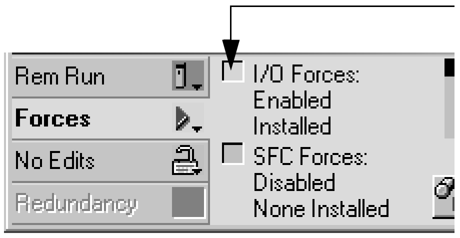
To override an input value, output value, produced tag, or consumed tag, use an I/O force.

ATTENTION

Forcing can cause unexpected machine motion that could injure personnel. Before you use a force, determine how the force will effect your machine or process and keep personnel away from the machine area.

- Enabling I/O forces causes input, output, produced, or consumed values to change.
- If forces are enabled and you install a force, the new force immediately takes effect.

1. What is the state of the I/O Forces status indicator?



If	Then note
Off	No I/O forces currently exist.
Flashing	No I/O forces are active. But at least one force already exists in your project. When you enable I/O forces, **all** existing I/O forces will also take effect.
Solid	I/O forces are enabled (active). When you install (add) a force, it immediately takes effect.

2. Open the routine that contains the tag that you want to force.

3. Right-click the tag and choose Monitor.

   If necessary, expand the tag to show the value that you want to force (that is, BOOL value of a DINT tag).

4. Install the force value.

To force a	Do this
BOOL value	Right-click tag and choose Force On or Force Off.
Non-BOOL value	In the Force Mask column for the tag, type the value that you want to force the tag. Press Enter.

5. Are I/O forces enabled? (See step 1.)

If	Then
No	From the Logic menu, choose I/O Forcing > Enable All I/O Forces. Choose Yes to confirm.
Yes	Stop.

Remove or Disable Forces

This section describes how to remove and disable forces.

ATTENTION ⚠

Changes to forces can cause unexpected machine motion that could injure personnel. Before you disable or remove forces, determine how the change will effect your machine or process and keep personnel away from the machine area.

If you want to	And	Then
Stop an individual force	Leave other forces enabled and in effect	Remove an Individual Force
Stop all I/O forces but leave all SFC forces active	Leave the I/O forces in the project	Disable All I/O Forces
	Remove the I/O forces from the project	Remove All I/O Forces

Remove an Individual Force

ATTENTION

⚠

If you remove an individual force, forces remain in the enabled state and any new force immediately takes effect.

Before you remove a force, determine how the change will effect your machine or process and keep personnel away from the machine area.

1. Open the routine that contains the force that you want to remove.
2. What is the language of the routine?

If	Then
SFC	Go to step 4.
Ladder logic	Go to step 4.
Function block	Go to step 3.
Structured text	Go to step 3.

3. Right-click a tag that has the force and choose Monitor.

   If necessary, expand the tag to show the value that is forced, for example, BOOL value of a DINT tag.

4. Right-click a tag or element that has the force and choose Remove Force.

Disable All I/O Forces

To disable, choose Logic>I/O Forcing>Disable All I/O Forces. Click Yes to confirm.

Remove All I/O Forces

To remove, choose Logic>I/O Forcing>Remove All I/O Forces. Click Yes to confirm.

Data Access Control

Introduction

In the Logix platform, software version 18 or later, there are two tag attributes that allow you to control access to tag data. These attributes are:

- External Access
- Constant

The External Access attribute controls how external applications, such as HMIs, can access tags. It has possible values of Read/Write, Read Only, and None. See Configure External Access on page 64.

The Constant attribute value determines if a tag can be modified by controller logic. Also, by using FactoryTalk Security software, it is possible to control which users are permitted to change tags designated as constants in RSLogix 5000 software. See page 79 for more information on the Constant attribute.

By using these two attributes, you can help safeguard tag data by preventing unwanted changes to tag values. Also, by reducing the number of tags exposed to external applications, you can also reduce the time required to develop HMI screens.

External Access

By using the External Access attribute, you can control how external applications and devices can access tags.

This process can help you manage the thousands of tags you might have in a project that have similar names that can get easily confused when referencing them in applications or devices.

Using this attribute also can help improve system performance by reducing the number of tags RSLinx has to maintain, scan, and cache. This volume can impact the performance of the RSLinx data server and other related applications.

External applications and devices include:

- RSLinx Classic and RSLinx Enterprise software.
- other Logix controllers.
- PanelView terminals.
- PLC/SLC controllers.
- FactoryTalk Historian software.
- other third-party software.

Configure External Access

You configure external access from a pull-down menu when you create a new tag or data type. You can also modify that value just like other tag attributes. These changes can be made throughout the software. For example, they can be made in the User-defined Data Type Editor, New Tag Dialog, and the Tag Properties Dialog.

External Access Settings	Description
Read/Write	External applications and devices have full access to the tag and can read and change the tag's value.
Read Only	External applications can read, but cannot change, the tag's value.
None	External applications cannot read or change the tag's value.

IMPORTANT

RSLogix 5000 programming software has full access to all tags, regardless of their External Access settings. External access applies to all program, controller, and Add-On Instruction scoped tags.

If the controller is in safety locked mode, only the safety tags will be disabled from being accessed. The standard tags will have the same behavior as in the unlocked mode.

External Access Options

You can choose one of three options—Read/Write, Read Only, None from the External Access box on the following RS Logix 5000 dialog boxes:

- New Tag (See page 66)
- Tag Properties (See page 68)

The default value in the External Access box is dependent on the usage, and type of the tag. The following table describes the values.

Default Tag's External Access

If the tag is	Default value is
Alias	Same as its target. See Important note below.
Controller/program scoped and equipment phase input parameters	Out-of-box is Read/Write. Thereafter, when creating a new tag, the default external access tag retains the value of the user's previous choice.(1)
Equipment phase output parameters	Out-of-box is Read Only. Thereafter, when creating a new tag, the default external access tag retains the value of the user's previous choice.(1)

(1) The External Access default value for tag creation is stored per Windows login account.

IMPORTANT

For Alias type, the External Access box is disabled. You are not allowed to change the external access of an alias tag. However, the External Access box will update its value to be the same as the external access of the base target.

See 'Go To' Search Menu on page 70 for procedures to locate the base tag for an alias.

See External Access Availability on page 71 for additional tag considerations.

Configure External Access in the New Tag Dialog Box

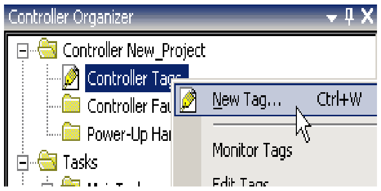
You can create these types of tags on the New Tag dialog box:

- Base tag
- Alias tag
- Produced tag
- Consumed tag

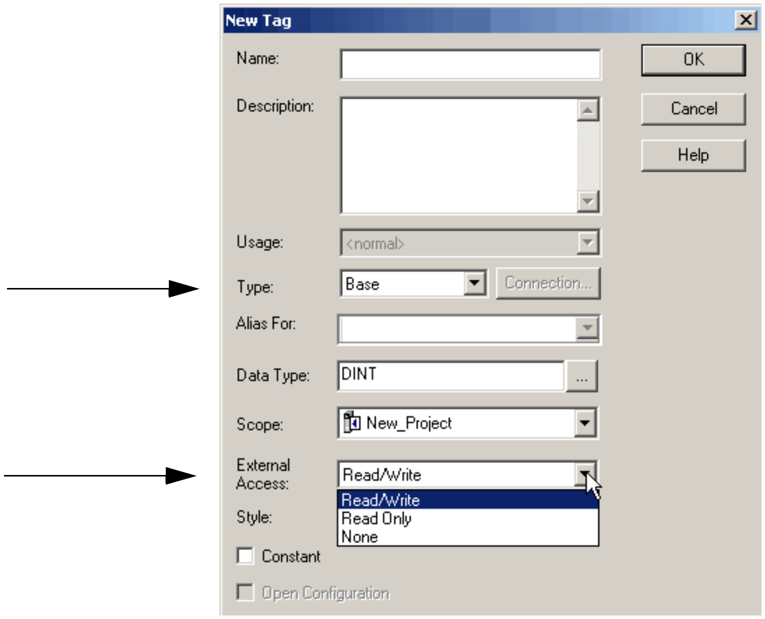
The parameters on the dialog box depend on the type of tag you are creating. For tag descriptions, see page 26.

The External Access box on the New Tag dialog box lets you assign the external access attribute for the tag being created. Follow these steps.

1. On the Controller Organizer, right-click Controller Tags and choose New Tag.



The New Tag Dialog Box appears.



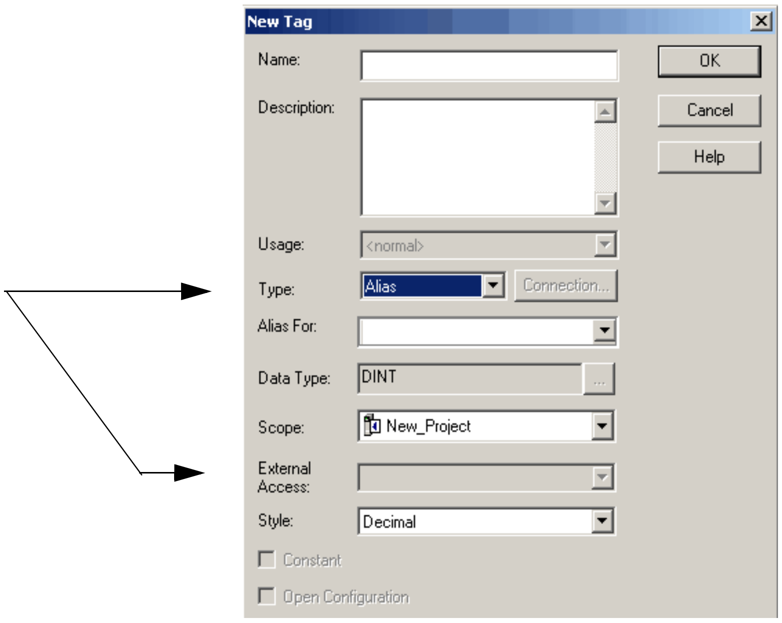
Name:
Description:
Usage: <normal>
Type: Base
Alias For:
Data Type: DINT
Scope: New_Project
External Access: Read/Write
Style:

2. From the Type pull-down menu, choose a tag type.

3. From the External Access pull-down menu, choose an external access option.

4. Click OK.

As shown in the example below, the External Access box is dimmed for an alias tag.



Name:

Description:

Usage: <normal>

Type: Alias

Alias For:

Data Type: DINT

Scope: New_Project

External Access:

Style: Decimal

There may be many alias tags in a program. To locate an associated base tag to assign an external access, use the 'Go To' feature. See page 70 for details.

For other tag considerations, see External Access Availability on page 71.

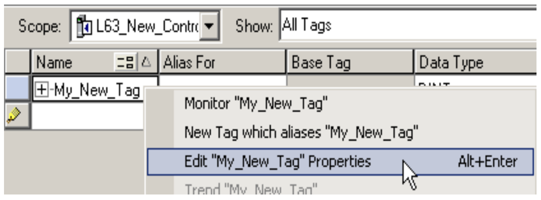
The Connection button (next to the Type box) becomes active when either a produced or consumed tag type is selected. The button accesses a dialog box for setting up produced/consumed tag connections. See the Logix5000 Controllers Produced and Consumed Tags Programming Manual, publication 1756-PM011.

Set Up External Access in the Tag Properties Dialog Box

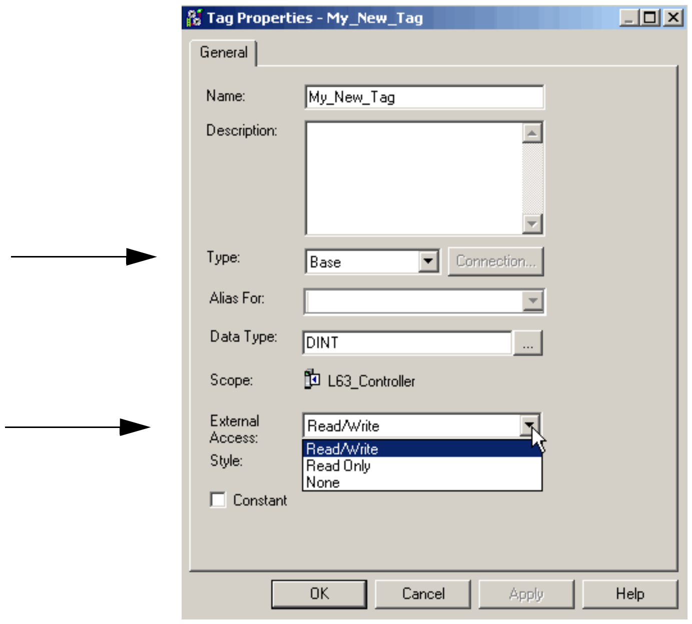
The Tag Properties dialog box is used to edit properties of existing tags. You can change tag attributes and modify tag types, such as base and alias.

Follow these steps to choose an external access option for an existing tag.

1. On the Tag Editor window, right-click a tag and choose Edit (tag name) Properties.



   The Tag Properties dialog box appears.



Name: My_New_Tag
Description:
Type: Base
Alias For:
Data Type: DINT
Scope: L63_Controller
External Access: Read/Write
Style:
Constant

2. From the Type pull-down menu, choose a tag type.

3. From the External Access pull-down menu, choose an external access option.

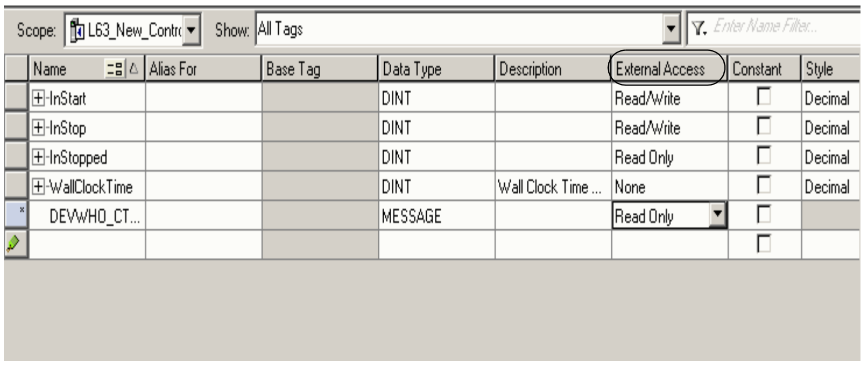
   The External Access box is dimmed for an alias tag. If a tag is a module tag, the only external access option is Read/Write.

   See External Access Availability on page 71 for other considerations.

4. Click OK.

View and Select External Access Status on the Tag Editor Window

You can view the external access status of a tag in the Tag Editor window. The External Access column displays the tag as 'Read/Write', 'Read Only', or 'None'.



Scope: L63_New_Contr Show: All Tags Enter Name Filter...

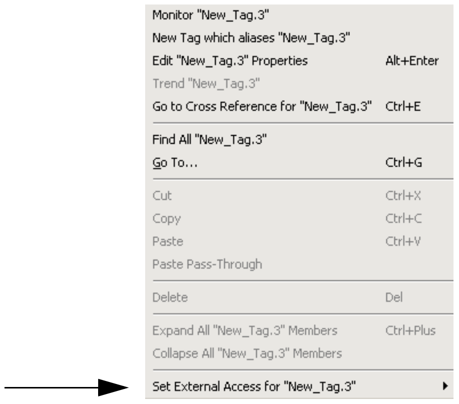
Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
InStart			DINT		Read/Write	☐	Decimal
InStop			DINT		Read/Write	☐	Decimal
InStopped			DINT		Read Only	☐	Decimal
WallClockTime			DINT	Wall Clock Time ...	None	☐	Decimal
DEVWHO_CT...			MESSAGE		Read Only	☐	
						☐	

Follow these steps to select multiple rows and set the external access at one time on the Tag Editor.

1. To select multiple individual rows, hold down Ctrl and click the desired rows.

2. Right-click a selected tag.

   A pull-down menu displays.


Monitor "New_Tag.3"
New Tag which aliases "New_Tag.3"
Edit "New_Tag.3" Properties Alt+Enter
Trend "New_Tag.3"
Go to Cross Reference for "New_Tag.3" Ctrl+E
Find All "New_Tag.3"
Go To... Ctrl+G
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Paste Pass-Through
Delete Del
Expand All "New_Tag.3" Members Ctrl+Plus
Collapse All "New_Tag.3" Members
Set External Access for "New_Tag.3"

3. Click 'Set External Access for (tag name)' to select an external access option.

   All highlighted rows that are enabled for changing External Access will change their external access setting.

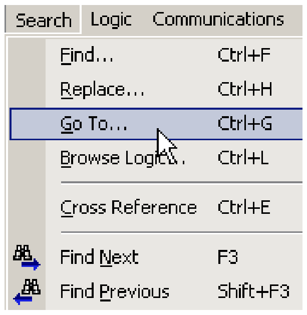
See External Access Availability on page 71 for considerations when the External Access column is disabled.

'Go To' Search Menu

The external access setting of an alias tag can only be changed through its base tag. The 'Go To' option on the Search menu of the RSLogix 5000 programming software is a convenient way to find the base tag among all the cross-reference records.

Follow these steps to locate a base tag.

1. With the Tag Editor window open, from the RSLogix 5000 Search menu, select the desired alias tag and choose Go To.



   The Go To window appears.

2. In the 'Go to what column', choose Base Tag.

   The box will display the target of the alias tag. If there is an alias chain, all alias tags in this chain will display in a pull-down list in the Go To column.

3. From the Go To pull-down menu, choose a target of the alias tag.

4. Click Go To.

   The target is located with a black box around it.

External Access Availability

The following table describes the conditions in which the External Access box is disabled.

IMPORTANT The External Access box is always disabled for any tag whose data type is Alarm Analog or Alarm Digital. The external access status is always Read/Write for these data types.

External Access Variables

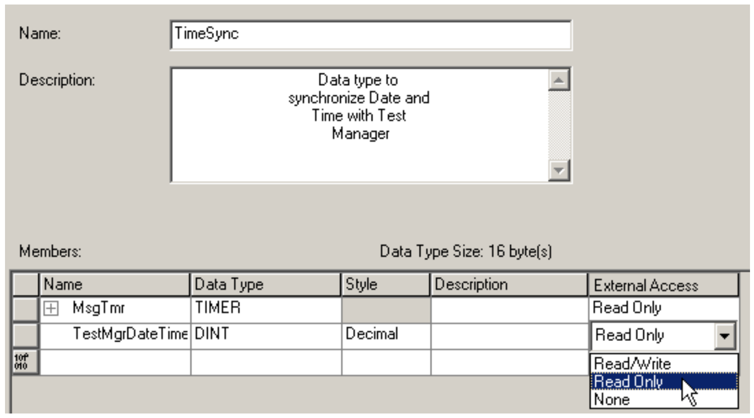
Dialog Box/Window	Considerations
New Tag	The External Access box is disabled if: • the tag is an alias tag. • the controller is user locked online. If you change the Type box from 'Base' to 'Alias,' the External Access box becomes disabled and appears blank. If you choose a target for an alias tag in the Alias For box, the External Access box remains disabled and the external access value appears in the External Access box. The external access setting of an alias tag can only be changed through its base tag.
Tag Properties	The External Access box is disabled if: • you do not have permission to change the external access settings. • the redundancy controller is in any state that does not allow changes. • the controller is user-locked online from another computer. • the controller is safety-locked and the tag is a safety tag. • the Scope is an equipment phase and the equipment phase feature is not activated in the current license. • the tag is an alias tag. • the controller is in hard-run mode.

External Access Variables

Dialog Box/Window	Considerations
Tag Editor	The External Access box is disabled if: • you do not have permission to change the external access settings. • the redundancy controller is in any state that does not allow changes. • the controller is user-locked online. • the controller is safety-locked and the tag is a safety tag. Only the safety tags' External Access cell is disabled. • the Scope is an equipment phase and the equipment phase feature is not activated in the current license. • the tag is an alias tag. • the controller is in hard-run mode. • the row represents an expanded array dimension, bit, or data member. For tags of Predefined (Atomic and Structural), Module-defined Data Types and String, all of these tag members will have the same external access level because: • they are all hard-coded to 'Read/Write' and you can only view, not change, this value. You also cannot change external access for the **data type members**. • an external access change on the tag results in an update on all tag members. For Array tags, all elements: • must have the same external access level. • of all data members for predefined or module-defined data types will have the same external access setting. • of each data member for user-defined type (UDT) and Add-On Instruction will have the more restrictive external access setting between the element external access setting and the external access setting of the member in the type definition.

User-defined Type Considerations

The three external access options—Read/Write (default), Read Only, None—are chosen from the External Access column on the Data Type dialog box.



Name: TimeSync

Description: Data type to synchronize Date and Time with Test Manager

Members: Data Type Size: 16 byte(s)

Name	Data Type	Style	Description	External Access
⊞ MsgTmr	TIMER			Read Only
TestMgrDateTime	DINT	Decimal		Read Only

Three external access rules apply for members of User-defined data types.

- You can only set external access for the top members of that User-defined data type. External Access cells for the child-members are disabled on the User-defined Data Type dialog box.
- If the member's data type is Predefined structural, Module-defined, or String, you cannot set external access of child-members. The external access level of the parent member is given to its child-members.
- If the member's data type is User-defined and the child-member has a different external access level from its parent, the more restrictive external access level is applied.

The following table describes the conditions in which the External Access column is disabled.

Availability of the External Access Setting for Structured Data Types

Topic	Considerations
Modify existing data type	The External Access column is disabled if: • you do not have permission to change the external access settings.(1) • the redundancy controller is in any state that does not allow changes. • the data type is applied to tags and the controller is online. Note: Data type size is not affected by the external access attribute.
Predefined, module-defined, Strings type	The external access column is always visible but disabled. The 'Set External Access' entry is added to the bottom of the row header context menu, but it is always disabled.

(1) If you have User-defined Data Type Modify permission, you also can modify external access of a User-defined data type.

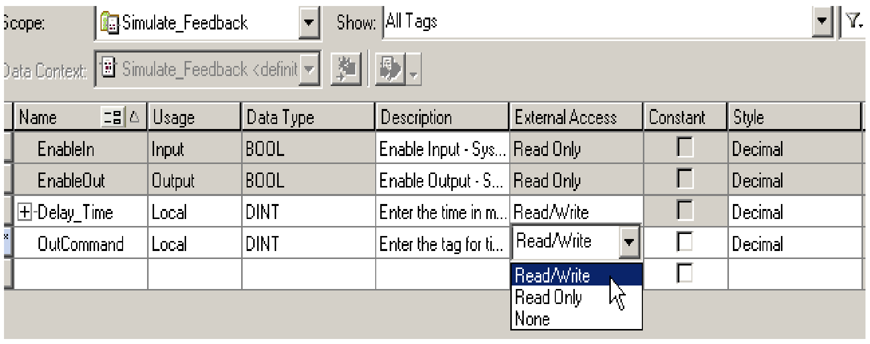
Add-On Instructions External Access Considerations

External Access settings can be used with parameters and local tags of Add-On Instructions. For example, if an input parameter is defined with external access of ‘read only’, the member that represents that parameter in the Add-On Instruction data type cannot be written.

The table below describes the External Access options for various Add-On Instruction parameters and tags.

Add-On Instruction Parameters and Tags	External Access Options
Local tag	Read/Write Read Only None
Input parameter	
Output parameter	
EnableIn parameter	Read Only
EnableOut parameter	
InOut parameter	Not Applicable

The external access for an Add-On Instruction tag can be chosen from the box on the New Tag dialog box or from the External Access column on the Tag Editor window.


Scope: Simulate_Feedback Show: All Tags
Data Context: Simulate_Feedback <definit

Name	Usage	Data Type	Description	External Access	Constant	Style
EnableIn	Input	BOOL	Enable Input - Sys...	Read Only	☐	Decimal
EnableOut	Output	BOOL	Enable Output - S...	Read Only	☐	Decimal
⊞-Delay_Time	Local	DINT	Enter the time in m...	Read/Write	☐	Decimal
OutCommand	Local	DINT	Enter the tag for ti...	Read/Write	☐	Decimal
					☐	

The external access of an Add-On Instruction’s parameters and local tags can be configured in the Add-On Instruction Definition dialog box and on the Add-On Instruction Parameters and Local Tags dialog boxes.

For alias parameters, the external access type is equal to the type configured for the base local tag.

Add-On Instruction External Access Variables

Dialog Box/Window	Considerations
New Add-On Instruction Parameter or Local Tag	If the current usage is: • Input parameter - the External Access box is enabled and the displayed value is your last selection when creating an equipment phase input parameter or Add-On Instruction input parameter. • Output parameter - the External Access box is enabled and the displayed value is your last selection when creating an equipment phase output parameter or Add-On Instruction output parameter. • InOut parameter - the External Access box is disabled and blank. • Local tag - the External Access box is disabled and the displayed value is 'None'.
Parameters/Local Tab Properties	No change is applied to the External Access box if you switch the usage among Input parameter, Output parameter or Local tag, except when the usage is a Local tag, the box is disabled. If you change the usage from InOut parameter to: • Input or output parameter - the External Access box is enabled and your last selection for creating an equipment phase/Add-On Instruction input parameter or an equipment phase/Add-On Instruction output parameter is displayed accordingly. • Local tag - the External Access is updated to 'None', and the box is disabled. The External Access box also is disabled if: • you do not have permission to change external access settings.(1) • the controller is online. • the tag is an alias tag. • the Add-On Instruction is in Source Protection mode.

Add-On Instruction External Access Variables

Dialog Box/Window	Considerations
Add-On Instruction Definition - Parameters Tab	The External Access column is disabled if: • InOut parameters, which are blank. • EnableIn and EnableOut parameters, which default 'Read Only'. • you do not have permission to change the external access settings.(1) • the controller is online. • the tag is an alias tag. • the Add-On Instruction is in Source Protection mode. • the row represents an expanded bit, or data member. When creating a new parameter, changing usage causes the External Access column auto update to default to: • Input parameter - equipment phase input parameter and Add-On Instruction input parameter. • Output parameter - equipment phase output parameter and Add-On Instruction output parameter. • InOut parameter - External Access column cell is blank and disabled. Changing external access attributes will cause: • an error message if you change a tag from Input or Output parameter to InOut parameter and the present attribute is either 'Read/Write,' or 'Read Only'. • no change if you switch between Input and Output parameters. • the value of the external access updates to the new target for an alias.
Add-On Instruction Definition - Local Tags Tab	The External Access column is disabled if: • you do not have permission to change external access settings.(1) • the controller is online. • the Add-On Instruction is in Source Protection mode. • the row represents an expanded array dimension, bit, or data member.

Add-On Instruction External Access Variables

Dialog Box/Window	Considerations
Add-On Instruction Edit Tags	Note: External access is not applicable for InOut parameters because they are just references until invoked. The External Access column is disabled if: • EnableIn and EnableOut parameters, which default 'Read Only'. • you do not have permission to change the external access settings.(1) • the controller is online. • the tag is an alias tag. • the Add-On Instruction is in Source Protection mode. • the row represents an expanded array dimension, bit, or data member. When creating a new parameter, changing usage causes the External Access column auto update to default to: • Input parameter - equipment phase input parameter and Add-On Instruction input parameter. • Output parameter - equipment phase output parameter and Add-On Instruction output parameter. • InOut parameter - External Access column cell is blank and disabled. • Local tag - external access is updated to 'None'. Changing external access attributes will cause: • a warning message if you change a tag from Input or Output parameter to InOut parameter and the present attribute is either 'Read/Write,' or 'Read Only'. • no change if you switch between Input, Output parameters and Local tag • the value of the external access updates to the new target for an alias.

(1) If you have Add-On Instruction Modify permission, you also can modify external access of an Add-On Instruction tag.

Tag Mapping Considerations

Only tags with external access settings of Read/Write or Read Only can be mapped to a PLC-2 controller and PLC-5/SLC controllers.

PLC-2, PLC-5/SLC External Access Variables

Dialog Box/Window	Considerations
PLC-2, PLC-5/SLC Mapping	To map a tag: • Type a file number. • Choose a tag from the Name box. Only eligible tags that are set to either Read/Write or Read Only will display in the pull-down menu. If you manually type the name of a tag whose external access is set to None, an error message displays. • Click OK.

Imported Tag Behavior

The RSLogix 5000 programming software preforms a check to verify an imported program file has a valid external access value. A default value is assigned to unspecified tags that are imported from programs that have software with versions earlier than 18.

An error message displays in RSLogix 5000 software for imported files that contain tags with any value other than Read/Write, Read Only, and None.

Default External Access Values for Imported Program Files

Object Name	Default External Access
Controller and program-scoped standard tags	Read/Write
All safety tags	Read Only
Add-on Instruction local tags	Read/Write
Add-on Instruction Input parameters	Read/Write
Add-on Instruction Output, EnableIn and EnableOut parameters	Read Only
Add-on Instruction InOut parameters	N/A
Equipment phase output parameters	Read Only
Members of all data types	Read/Write

Constant Value Tags

In RSLogix 5000 software version 18 and later, you can designate tags as constants to protect them from being changed programatically via:

- the controller programming application.
- logic in the controller.

Tags that cannot be designated as constants are User-defined type members, Add-On Instruction input and output parameters, and local tags. A check mark in the Constant box on tag creation dialog boxes and tag editor/monitor windows indicates a ‘constant’ designation.

FactoryTalk security is used to control who is permitted to modify values of constants and who can modify the constant attribute of a tag. To change the value of a constant, you must have the Tag: Modify Constant Tag Values permission. To modify the constant attribute of a tag, you must have the Tag: Modify Constant Property permission.

For details on setting permissions, see the FactoryTalk Security System Configuration Guide, publication FTSEC-QS001.

For an alias tag, the default constant setting of this tag is the same as its target tag. For all other conditions, the default value is unchecked, indicating the tag is not a constant value tag.

When you designate an InOut parameter as a constant, it cannot be written to within the Add-On Instruction.

TIP You cannot pass a constant value tag as an argument to an Output parameter of an Add-On Instruction. You cannot pass a constant tag to an InOut parameter that is not also designated as a constant value.

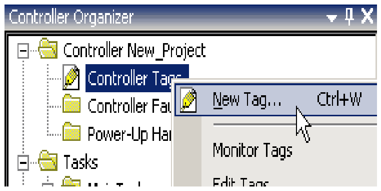
Configure Constant Tags

This section describes the various ways a constant attribute can be configured.

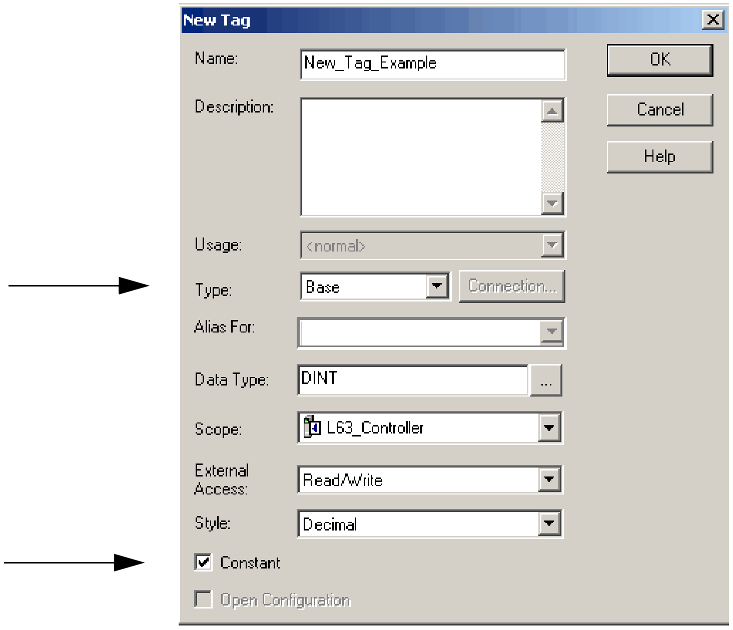
Set Up a Constant in the New Tag Dialog Box

Follow these steps to configure a tag as a constant on the New Tag dialog box.

1. On the Controller Organizer, right-click Controller Tags and choose New Tag.



The New Tag Dialog Box appears.



Name: New_Tag_Example
Description:
Usage: <normal>
Type: Base
Alias For:
Data Type: DINT
Scope: L63_Controller
External Access: Read/Write
Style: Decimal
☑ Constant
☐ Open Configuration

2. From the Type pull-down menu, choose a tag type.

3. Check Constant.

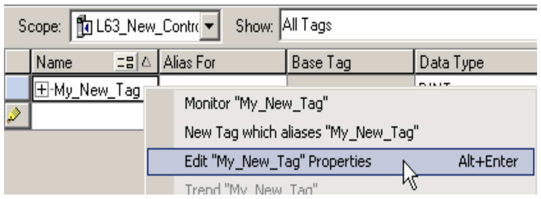
4. Click OK.

See Constant Checkbox Availability on page 84 for considerations.

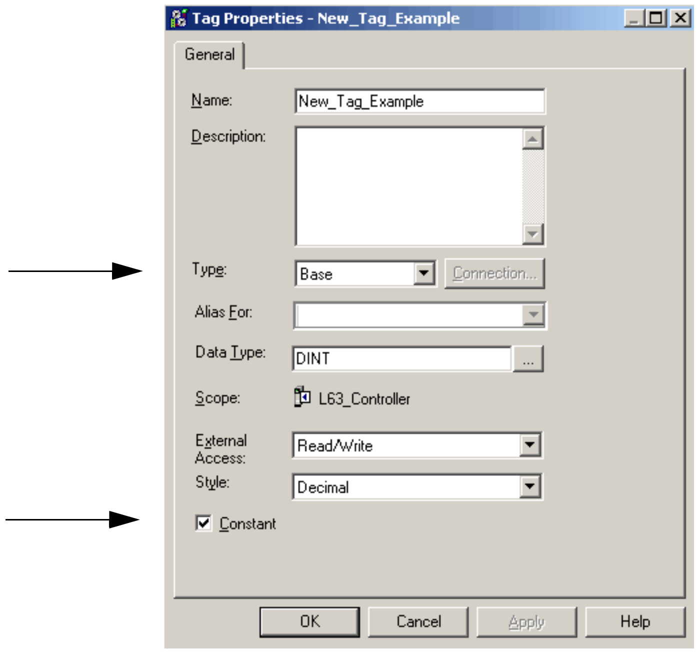
Configure a Constant in the Tag Properties Dialog Box

Follow these steps to designate a tag as a constant on the Tag Properties dialog box.

1. On the Tag Editor window, right-click a tag and choose Edit (tag name) Properties.



The Tag Properties dialog box appears.



Name: New_Tag_Example
Description:
Type: Base
Alias For:
Data Type: DINT
Scope: L63_Controller
External Access: Read/Write
Style: Decimal
☑ Constant

2. From the Type pull-down menu, choose a tag type.

3. Check Constant.

4. Click OK.

See Constant Checkbox Availability on page 84 for considerations.

Designate a Constant in the Tag Editor

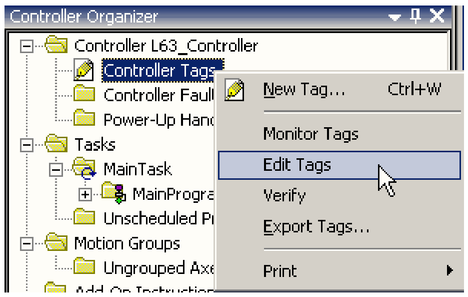
The Constant column on the Tag Editor window lets you designate tags that cannot be modified in the RSLogix 5000 software program. The Constant property applies to an entire tag; all members of the tag take on the same setting. The Constant column cells are blank for members of the constant tag.

An error message displays if a user tries to change the data type of a constant tag to a data type that cannot be constant.

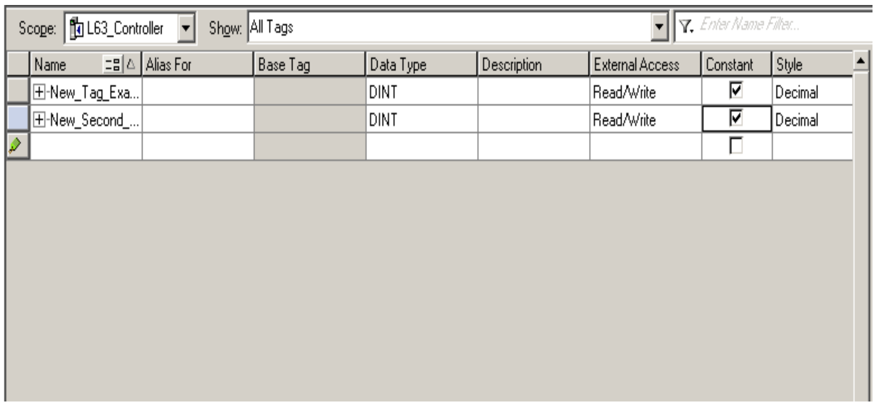
Follow these steps to add a constant value in the Tag Editor window.

1. On the Controller Organizer, right-click Controller Tags and choose Edit Tags.

   A pull-down menu appears.



   The Tag Editor window appears.



Scope: L63_Controller Show: All Tags Enter Name Filter...

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
New_Tag_Exa...			DINT		Read/Write	☑	Decimal
New_Second_...			DINT		Read/Write	☑	Decimal
						☐	

2. Click the checkbox in the Constant column.

IMPORTANT

In the

In the Tag Monitor window, the constant setting of the tag displays in the same Constant column as shown in the above illustration. However, you cannot change the value.

The Constant column also is available on the Equipment Phase Tag Edit window and Equipment Phase Tag Monitor window.

Constant Checkbox Availability

The state of the Constant checkbox depends on a number of conditions.

Constant Variables

Dialog Box/Window	Considerations
New Tag	The Constant box is disabled if: • the tag is an alias tag. • the Factory Talk Security action is not enabled for changing constant value property of a tag. • you do not have permission to modify tag properties (Factory Talk Security Tag Modify is denied.) • the new tag is a Consumed tag. • the tag's 'Data Type' is not a data table-backed type. • the tag's 'Usage' setting is not 'InOut'. • redundancy controller is in any state that does not allow changes. • the controller is safety-secured and the tag is a safety tag.(1) • if the Scope is an equipment phase and the equipment phase feature is not activated in the current license. • the controller is in hard-run mode. • the Add-On Instruction is in Source Protection mode.
Tag Properties	Same considerations apply as for New Tag.
Tag Editor	
Tag Monitor	The value of a constant tag can be modified by using the Tag Monitor window if you have both standard Tag: Modify Values permission and Tag: Modify Constant Tag Values permission. You cannot modify a constant value in any of the language editors or any other tag browser. The icon ⊖ in the Value column indicates that you are changing a constant value tag's value. Any modifications to the values of constant tags are recorded in the Controller Log for future reference. For controller logging, see the Logix5000 Controllers Information and Status Programming Manual, publication 1756-PM015.

(1) If the controller is in safety-locked mode, only the safety tags will be disabled from being accessed, the standard tags will have the same behavior as in the unlocked mode. The Constant value box will be disabled in the Tag Properties dialog box only if the tag is a safety tag.

Add-On Instructions Constant Value Considerations

The Constant attribute applies only to InOut parameters. The default setting of the property will be 'not a Constant Value'.

The Constant attribute will not apply to Input, Output, EnableIn and EnableOut Add-On Instruction parameters. It will not apply to Add-On Instruction Local tags.

By denoting an InOut parameter of an Add-On Instruction as a constant, it means that within the Add-On Instruction, that parameter cannot be written to. The project will fail verification if this type of write is attempted.

Appropriate usage of Constant tags is monitored by logic verification.

Notes:

Notes:

Rockwell Automation Support

Rockwell Automation provides technical information on the Web to assist you in using its products. At http://www.rockwellautomation.com/support/, you can find technical manuals, a knowledge base of FAQs, technical and application notes, sample code and links to software service packs, and a MySupport feature that you can customize to make the best use of these tools.

For an additional level of technical phone support for installation, configuration, and troubleshooting, we offer TechConnect support programs. For more information, contact your local distributor or Rockwell Automation representative, or visit http://www.rockwellautomation.com/support/.

Installation Assistance

If you experience an anomoly within the first 24 hours of installation, review the information that is contained in this manual. You can contact Customer Support for initial help in getting your product up and running.

United States or Canada	1.440.646.3434
Outside United States or Canada	Use the Worldwide Locator at http://www.rockwellautomation.com/support/americas/phone_en.html, or contact your local Rockwell Automation representative.

New Product Satisfaction Return

Rockwell Automation tests all of its products to ensure that they are fully operational when shipped from the manufacturing facility. However, if your product is not functioning and needs to be returned, follow these procedures.

United States	Contact your distributor. You must provide a Customer Support case number (call the phone number above to obtain one) to your distributor to complete the return process.
Outside United States	Please contact your local Rockwell Automation representative for the return procedure.

Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete this form, publication RA-DU002, available at http://www.rockwellautomation.com/literature/.

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation, Vorstlaan/Boulevard du Souverain 36, 1170 Brussels, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Publication 1756-PM004C-EN-P - October 2009
Supersedes Publication 1756-PM004B-EN-P - July 2008