EZ-ZONE® RMA & EtherNet/IP Configuration & Startup Using an Allen-Bradley CompactLogix PLC

EtherNet/IP Fundamentals

EtherNet/IP is built on the Common Industrial Protocol (CIP) at a foundational level. When communicating using CIP there are two ways to communicate to/from the Master and Slave devices, i.e., Implicitly (real-time I/O messaging) and Explicitly (information/configuration messaging). For your reference, the Watlow device is always the Slave where the PLC is the Master on the network. This document will look closely at both methods of communication.

Explicit Communications - Defined

This type of messaging is executed on demand and can vary in size. Every message must be individually configured to execute a specific Message Type, e.g., CIP Generic and a specific Service Type, e.g., Get Attribute Single. Each device will interpret the message, act upon the task and then generate a response. This message type encapsulates information about the protocol itself as well as the instructions that need to be carried out in a TCP/IP packet. When a message is sent using TCP/IP it requires a response from the device. As stated above, this type of message is generally reserved for diagnostics and configuration.

Implicit Communications - Defined

Because implicit messaging is real-time I/O messaging, it places different demands on the system. Due to the time critical nature of this form of communications the protocol must be able to support multi-casting while also ensuring that the time to execute the task is as fast as possible. To do this effectively, EtherNet/IP incorporates a protocol called User Datagram Protocol/Internet Protocol (UDP). Basically, this protocol contains the data alone without requiring a response from the Slave device. All data that is passed implicitly is defined in the configuration or start up process. Because this method of communications contains the predefined data alone, it is considered to be low overhead and is therefore able to deliver the time-critical requirements for control.

By using both forms of communication EtherNet/IP can prioritize time-critical I/O communications over non-critical messages while allowing for both to occur simultaneously. Watlow EtherNet/IP equipped devices supports both forms (Explicit/Implicit) of communications.

1.0 Getting Started

Prior to configuring the EZ-ZONE RM Access module it is important to think through the needs of the application while also understanding some basic facts that pertain to the RMA module.

Note: This document will not cover basic configuration of the RMA for this is covered in the RMA User's Guide which can be found on the Watlow website; link provided below. http://www.watlow.com/literature/manuals.cfm.

1.1   Noteworthy RMA Facts

1.1.1 The RMA module is a gateway that will allow for communications to take place between dissimilar networks, e.g., Watlow's Standard Bus to EtherNet/IP.

1.1.2 One RMA module per EZ-ZONE RM system.

1.1.3 Fastest refresh should not exceed 250ms.

1.1.4 In this documentation, the RMA input assembly is referred to as the Originator to Target (O to T, instance 1) assembly where the RMA output assembly is referred to as the Target to Originator (T to O, instance 2). The Originator is the Master (usually a PLC) and the Target is the Slave (EZ-ZONE RMA).

1.1.5 All EZ-ZONE assembly members (inputs and outputs) are 32-bits.

1.1.6 The maximum implicit assembly size is 100 inputs and 100 outputs per RM system. Per module, the maximum implicit assembly size is 40 inputs and 40 outputs. See the appropriate user's guide for detail.

1.2 Understanding the Application Requirements

1.2.1 Will there be a need to infrequently read or write parameters between the Master and Slave? Explicit communications can be executed with minimal effort to accomplish this task. Setup and configuration can be found below (see: Explicit Communications Configuration Step-by-Step).

1.2.2 If using implicit communications determine what data (EZ-ZONE parameters) will be transferred implicitly (inputs and outputs) between the Master and Slave ensuring that the maximum number of members is not exceeded for any given module. Refer to the EZ-ZONE module User’s Guide to find parameters of choice as well as limitations in size. Click on the link below to retrieve the document of choice from the Watlow website. http://www.watlow.com/literature/manuals.cfm

1.2.3 Will the default EZ-ZONE module assemblies meet the application requirements or will the module assembly need to be modified? To answer this question, refer to the User’s Guide in the previous step to evaluate the default assemblies for each RM module.

1.2.4 How fast does the assembly information (I/O) need to be refreshed? When communicating implicitly, the Master (PLC) controls the cyclic timing (I/O updates) via a setting referred to as the Requested Packet Interval (RPI).

Note: Suggested RPI setting should be set between 250 and 500ms

2.0 Explicit Communications

2.1 Configuration

It should be noted here that if it is determined that the default Implicit Assemblies need to be changed (step 1.2.3 above), this is the communications method to use to accomplish that task. To establish explicit communications between Master and Slave devices configuration steps need to be executed within the PLC as well as within the RMA module. After the configuration requirements have been met, programming examples will follow.

RMA Configuration - Required Steps Using EZ-ZONE Configurator Software

a. Identify the RMA module on the network via an IP address
b. Enable EtherNet/IP
c. Enable the appropriate gateway instance (EZ-ZONE module address)
d. Define the CIP Instance Offset (if more than one EZ-ZONE module on the Standard Bus network). See note in step 2.4.4 for more information.
e. Define the I/O Implicit Assembly size

PLC Configuration - Required Step Using RSLogix5000 Software

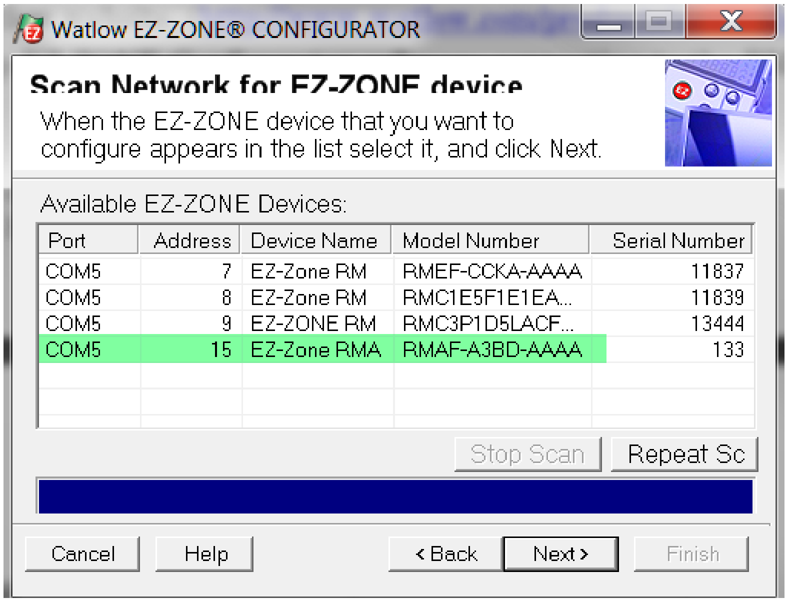
a. Add a Generic Ethernet module to the PLC I/O structure
b. Configure the module properties, e.g., IP address, Assembly size, etc…

2.2 RMA Configuration, Step-by-Step

2.2.1 Give the RMA module a valid network address (IP and Subnet) using EZ-ZONE Configurator software. This software is free of charge and if not already acquired it can be downloaded from the Watlow website. (http://www.watlow.com/products/controllers/software.cfm)

2.2.2 Using a EIA-485 connection, open up EZ-ZONE Configurator software (EZ-ZONE Configurator does not use Ethernet connection) and configure a device while communicating.

Note: If more information is needed in connecting the RMA module to the PC find the RMA User's Guide and turn to the wiring section (Standard Bus EIA-485 Communications).



Scan Network for EZ-ZONE device

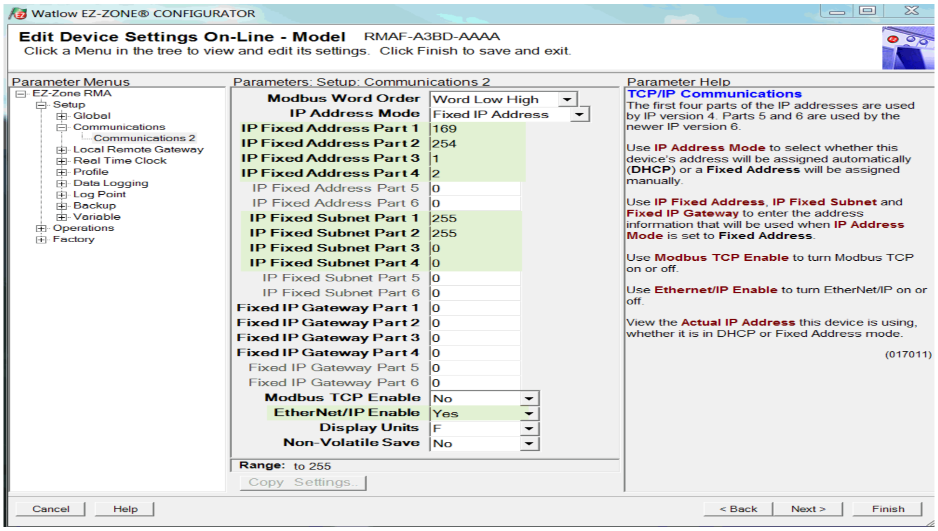
When the EZ-ZONE device that you want to configure appears in the list select it, and click Next.

Available EZ-ZONE Devices:

Port	Address	Device Name	Model Number	Serial Number
COM5	7	EZ-Zone RM	RMEF-CCKA-AAAA	11837
COM5	8	EZ-Zone RM	RMC1E5F1E1EA...	11839
COM5	9	EZ-ZONE RM	RMC3P1D5LACF...	13444
COM5	15	EZ-Zone RMA	RMAF-A3BD-AAAA	133

2.2.3 Click the Next button to connect to the RMA.

2.2.4 Once connected, navigate to the Communications Menu and then to the Communications 2 parameter. Once there enter the required network IP and Subnet address while also enabling EtherNet/IP (green highlight).



Edit Device Settings On-Line - Model RMAF-A3BD-AAAA
Click a Menu in the tree to view and edit its settings. Click Finish to save and exit.

Parameters: Setup: Communications 2

Modbus Word Order	Word Low High
IP Address Mode	Fixed IP Address
IP Fixed Address Part 1	169
IP Fixed Address Part 2	254
IP Fixed Address Part 3	1
IP Fixed Address Part 4	2
IP Fixed Address Part 5	0
IP Fixed Address Part 6	0
IP Fixed Subnet Part 1	255
IP Fixed Subnet Part 2	255
IP Fixed Subnet Part 3	0
IP Fixed Subnet Part 4	0
IP Fixed Subnet Part 5	0
IP Fixed Subnet Part 6	0
Fixed IP Gateway Part 1	0
Fixed IP Gateway Part 2	0
Fixed IP Gateway Part 3	0
Fixed IP Gateway Part 4	0
Fixed IP Gateway Part 5	0
Fixed IP Gateway Part 6	0
Modbus TCP Enable	No
EtherNet/IP Enable	Yes
Display Units	F
Non-Volatile Save	No

Range: to 255

Parameter Help

TCP/IP Communications
The first four parts of the IP addresses are used by IP version 4. Parts 5 and 6 are used by the newer IP version 6.

Use **IP Address Mode** to select whether this device's address will be assigned automatically (**DHCP**) or a **Fixed Address** will be assigned manually.

Use **IP Fixed Address**, **IP Fixed Subnet** and **Fixed IP Gateway** to enter the address information that will be used when **IP Address Mode** is set to **Fixed Address**.

Use **Modbus TCP Enable** to turn Modbus TCP on or off.

Use **Ethernet/IP Enable** to turn EtherNet/IP on or off.

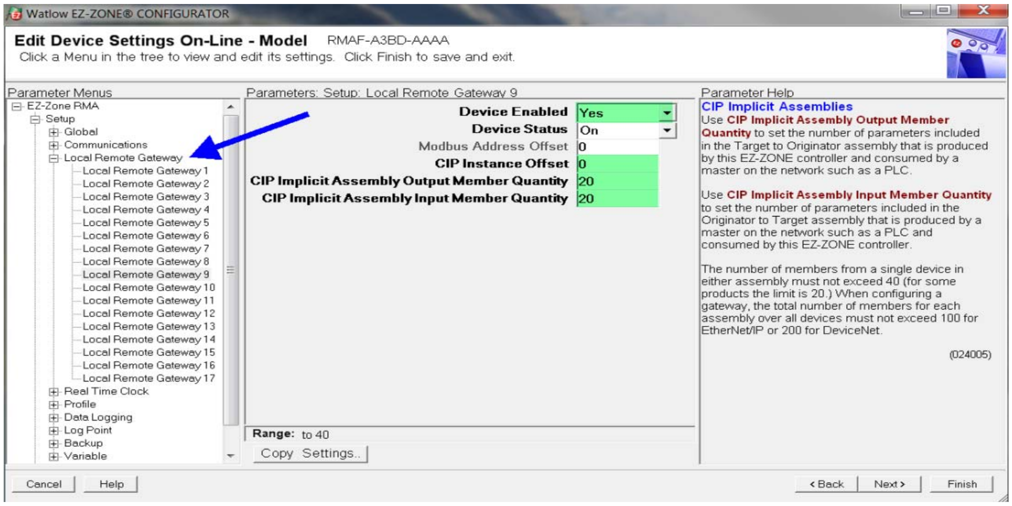
View the **Actual IP Address** this device is using, whether it is in DHCP or Fixed Address mode.

(017011)

2.2.5 In order for the RMA to work in concert with the PLC it is important that the assembly sizes match in the RMA and the PLC configurations. Navigate to the "Local Remote Gateway" and expand the folder structure by clicking on the plus symbol.

Note: The input assembly within the PLC will always be set to n+1 where n = the size of the combined (all modules) input assemblies (see step 2.3.4).

2.2.6 Click on the gateway instance that will pass data through the RMA to the Master device and configure as shown below. The gateway instance is an EZ-ZONE module with an address that ranges from 1 to 17. Although the RMA can be set to any available address (1 to 17), only the RMA defaults to address 17. Using the default address for the RMA ensures that addresses 1 through 16 are available for current and future module additions.



Edit Device Settings On-Line - Model RMAF-A3BD-AAAA
Click a Menu in the tree to view and edit its settings. Click Finish to save and exit.

Parameters: Setup: Local Remote Gateway 9

Device Enabled	Yes
Device Status	On
Modbus Address Offset	0
CIP Instance Offset	0
CIP Implicit Assembly Output Member Quantity	20
CIP Implicit Assembly Input Member Quantity	20

Range: to 40

Parameter Help

CIP Implicit Assemblies
Use **CIP Implicit Assembly Output Member Quantity** to set the number of parameters included in the Target to Originator assembly that is produced by this EZ-ZONE controller and consumed by a master on the network such as a PLC.

Use **CIP Implicit Assembly Input Member Quantity** to set the number of parameters included in the Originator to Target assembly that is produced by a master on the network such as a PLC and consumed by this EZ-ZONE controller.

The number of members from a single device in either assembly must not exceed 40 (for some products the limit is 20.) When configuring a gateway, the total number of members for each assembly over all devices must not exceed 100 for EtherNet/IP or 200 for DeviceNet.

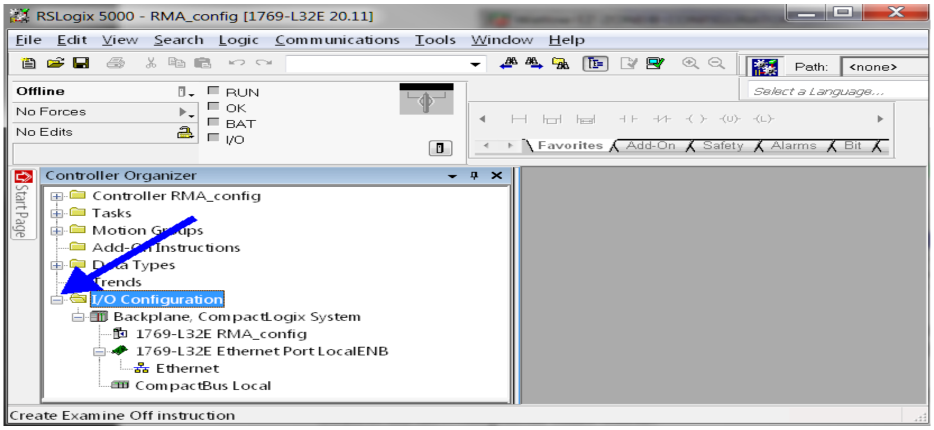
(024005)

2.2.7 If explicit messaging will be used alone, the minimum assembly size requirement (from the PLC perspective) is 1 input and 1 output. Again, the assembly size must be the same in the PLC and the RMA. In the previous step, the graphic shows the assemblies set to twenty. This was done now for implicit communication examples that will follow later in this document.

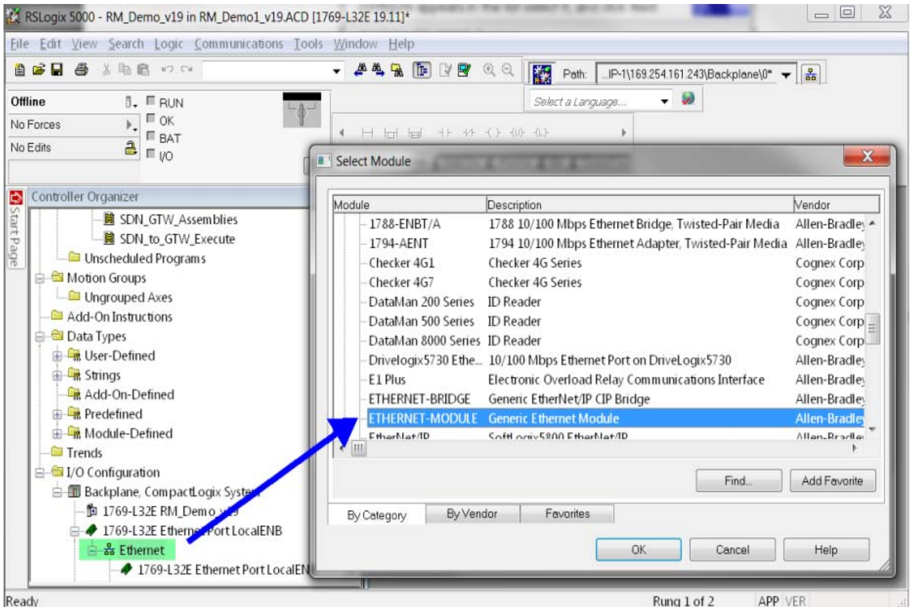
2.3 PLC Configuration, Step-by-Step

2.3.1 Open RSLogix5000 software and add an additional I/O module. Follow the steps below to accomplish this task.

2.3.2 Navigate to the I/O Configuration folder structure. If not already expanded, do so now by clicking the plus sign next to it.



2.3.3 Right click the Ethernet port to add a new module. To narrow the search select the "Communications" category for available modules, select a "Generic Ethernet Module" and then click OK.



Module	Description	Vendor
1788-ENBT/A	1788 10/100 Mbps Ethernet Bridge, Twisted-Pair Media	Allen-Bradley
1794-AENT	1794 10/100 Mbps Ethernet Adapter, Twisted-Pair Media	Allen-Bradley
Checker 4G1	Checker 4G Series	Cognex Corp
Checker 4G7	Checker 4G Series	Cognex Corp
DataMan 200 Series	ID Reader	Cognex Corp
DataMan 500 Series	ID Reader	Cognex Corp
DataMan 8000 Series	ID Reader	Cognex Corp
Drivelogix5730 Ethe...	10/100 Mbps Ethernet Port on DriveLogix5730	Allen-Bradley
E1 Plus	Electronic Overload Relay Communications Interface	Allen-Bradley
ETHERNET-BRIDGE	Generic EtherNet/IP CIP Bridge	Allen-Bradley
ETHERNET-MODULE	Generic Ethernet Module	Allen-Bradley
EtherNet/IP	SoftLogix5800 EtherNet/IP	Allen-Bradley

2.3.4 Define the RMA module properties. Fields that must be completed include:

Name: Given name becomes controller tags to be used in program.

Comm Format: Defines how data is to be treated within PLC.

***Note*: All EZ-ZONE assembly members are 32-bits in length. If the Comm Format is set to something other than DINT, ensure the size changes in a corresponding fashion. As an example, if 20 (32-bit) members are in use, the appropriate Comm Formats would be:

DINT (32-bit):	*Inputs = 21, Outputs = 20
INT (16-bit):	*Inputs = 42, Outputs = 40
SINT (8-bit):	*Inputs = 84, Outputs = 80

* The input assemblies within EZ-ZONE modules have a dedicated Status member that is always present. The input assembly size will always be n+1 where n = the size of the combined (all modules) input assembly.

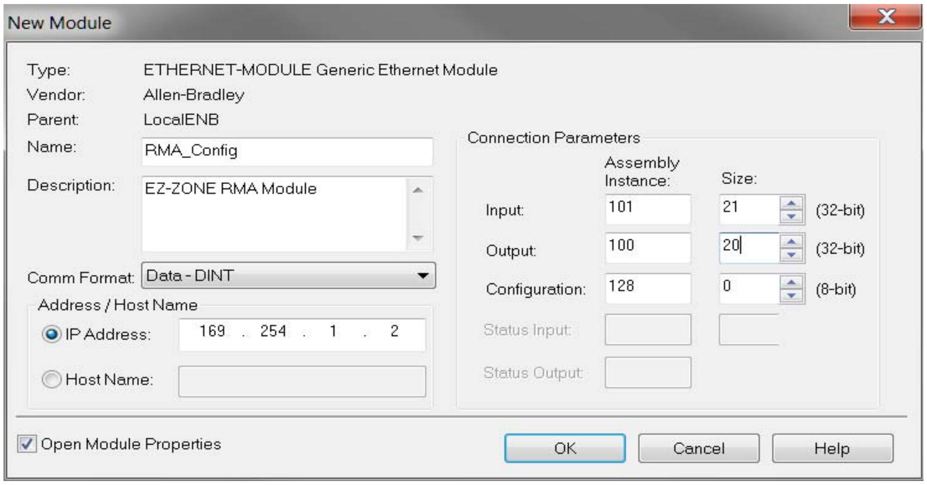
IP Address: Network RMA module Ethernet address.

Assembly Instance

Input (101): Defines number of members to be included in the Input Implicit Assembly (as seen in EZ-ZONE Configurator software “CIP Implicit Assembly Output Member Quantity” from EZ-ZONE devices).

Output (100): Defines number of members to be included in the Output Implicit Assembly (as seen in EZ-ZONE Configurator software “CIP Implicit Assembly Input Member Quantity from master devices).

Configuration(128): Enter zero.



New Module

Type: ETHERNET-MODULE Generic Ethernet Module
Vendor: Allen-Bradley
Parent: LocalENB
Name: RMA_Config
Description: EZ-ZONE RMA Module
Comm Format: Data - DINT

Address / Host Name
● IP Address: 169 . 254 . 1 . 2
○ Host Name:

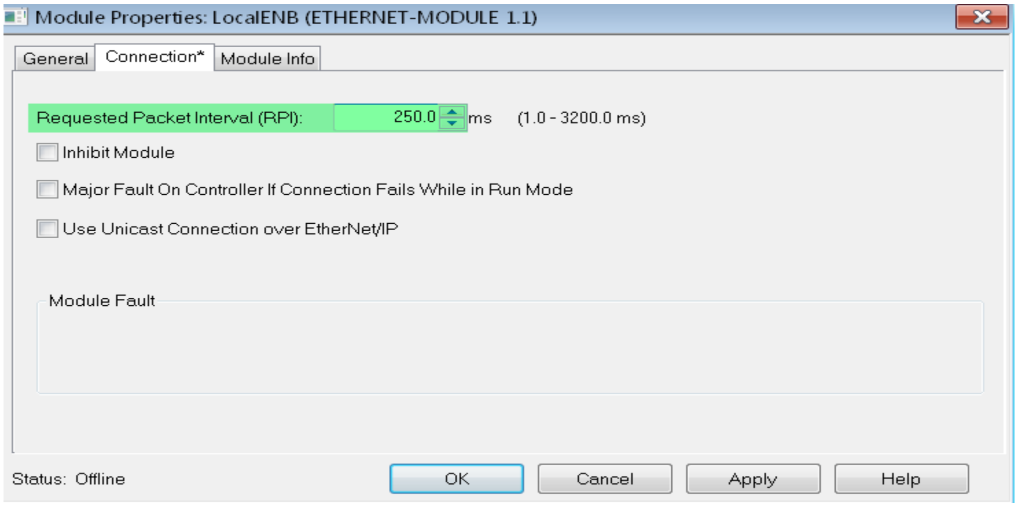
Connection Parameters

	Assembly Instance:	Size:	
Input:	101	21	(32-bit)
Output:	100	20	(32-bit)
Configuration:	128	0	(8-bit)
Status Input:			
Status Output:			

☑ Open Module Properties

2.3.5 Click OK when all fields have been entered.

2.3.6 Enter the Requested packet interval, used with implicit messaging (250 to 500ms).



Module Properties: LocalENB (ETHERNET-MODULE 1.1)

Requested Packet Interval (RPI): 250.0 ms (1.0 - 3200.0 ms)

□ Inhibit Module

□ Major Fault On Controller If Connection Fails While in Run Mode

□ Use Unicast Connection over EtherNet/IP

Module Fault

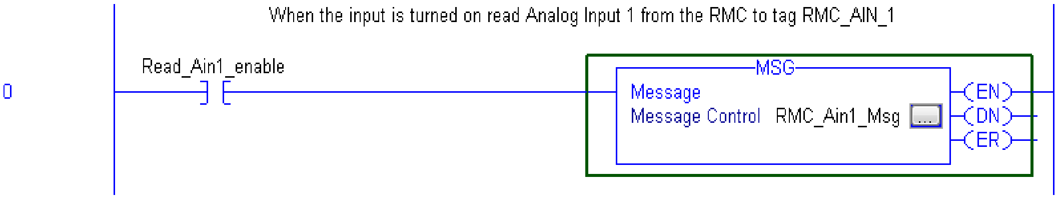
Status: Offline

2.3.7 The RMA configuration is now complete.

2.4 Explicit Programming Examples, Step-by-Step

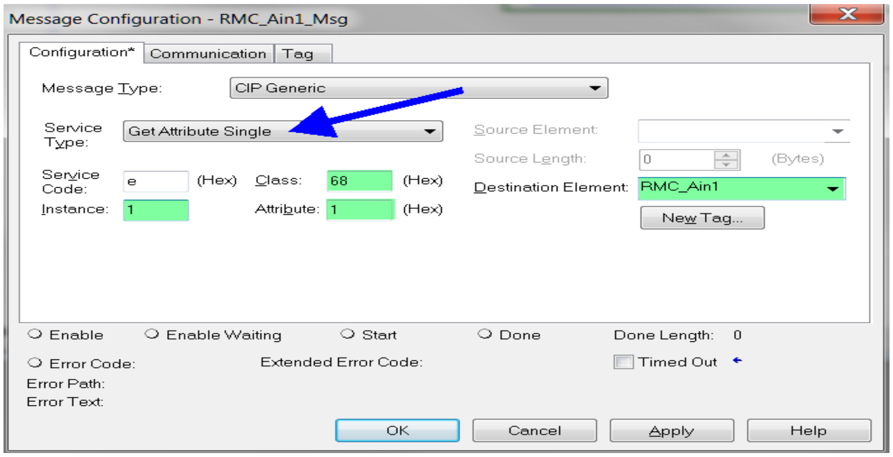
The examples below will use a very simple and straight forward way to execute an explicit message. There are other ways to do this within the PLC.

2.4.1 The first example will read the first Analog input from an RMC module. To do this, create a rung of logic similar to that shown below.



When the input is turned on read Analog Input 1 from the RMC to tag RMC_AIN_1
0
Read_Ain1_enable
MSG
Message
Message Control RMC_Ain1_Msg
...
EN
DN
ER

2.4.2 Message instruction configuration with explanations.



Message Configuration - RMC_Ain1_Msg

Message Type: CIP Generic

Service Type: Get Attribute Single

Service Code: e (Hex) Class: 68 (Hex)

Instance: 1 Attribute: 1 (Hex)

Source Element:

Source Length: 0 (Bytes)

Destination Element: RMC_Ain1

○ Enable ○ Enable Waiting ○ Start ○ Done Done Length: 0

○ Error Code: Extended Error Code: □ Timed Out ←

Error Path:
Error Text:

Explanation of fields in graphic above follows.

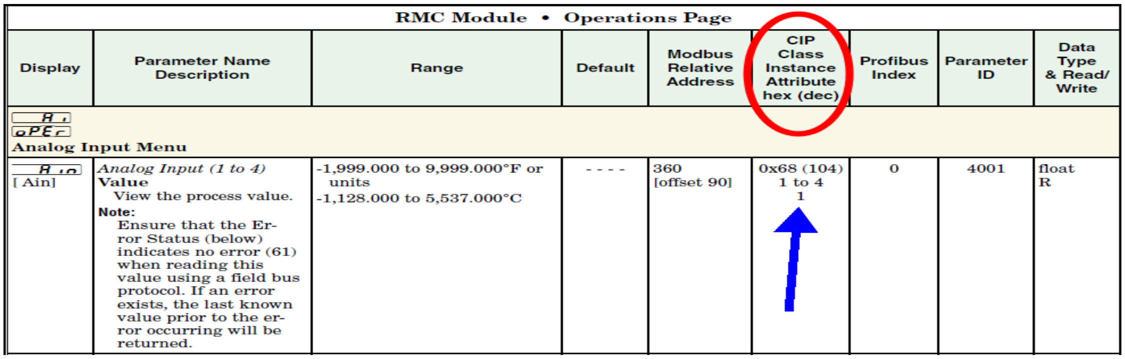
- *Service Type*

This particular example will read a single parameter (attribute) from the RMC, therefore the Service Type is “Get Attribute Single”.

- *Class, Instance and Attribute*

Note: The Class and Attribute are always entered in hexadecimal where the instance is entered in decimal.

These fields represent the actual CIP address within the RMC for the Analog Input. This address can be found in the RMC User’s Guide in the Operations Page.



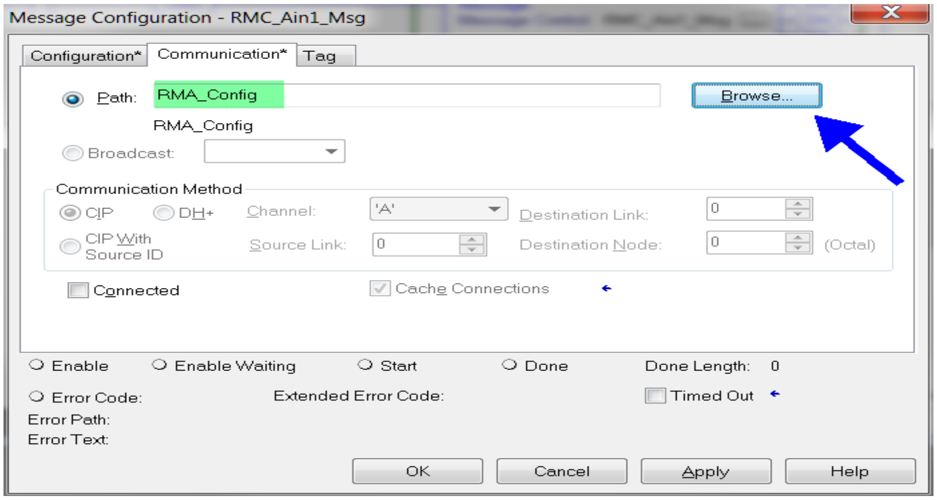
RMC Module • Operations Page

Display	Parameter Name Description	Range	Default	Modbus Relative Address	CIP Class Instance Attribute hex (dec)	Profibus Index	Parameter ID	Data Type & Read/ Write
A i oPEr Analog Input Menu								
A in [Ain]	*Analog Input (1 to 4)* **Value** View the process value. **Note:** Ensure that the Error Status (below) indicates no error (61) when reading this value using a field bus protocol. If an error exists, the last known value prior to the error occurring will be returned.	-1,999.000 to 9,999.000°F or units -1,128.000 to 5,537.000°C	- - - -	360 [offset 90]	0x68 (104) 1 to 4 1	0	4001	float R

- *Destination Element*

The tag shown above (RMC_Ain1) must be created by the user and represents the location in which the Analog Input value will be found when the message instruction is executed.

2.4.3 When all fields in the graphic above are filled in, click the Communications tab to identify the path to the RMC module. Click the Browse button and select the RMA module that was configured in step 2.3.4. Lastly, click the OK button to finish the message instruction configuration.



Message Configuration - RMC_Ain1_Msg

Configuration* | Communication* | Tag

● Path: RMA_Config

RMA_Config

○ Broadcast:

Communication Method

○ CIP ○ DH+ Channel: 'A' Destination Link: 0

○ CIP With Source ID Source Link: 0 Destination Node: 0 (Octal)

☐ Connected ☑ Cache Connections

○ Enable ○ Enable Waiting ○ Start ○ Done Done Length: 0

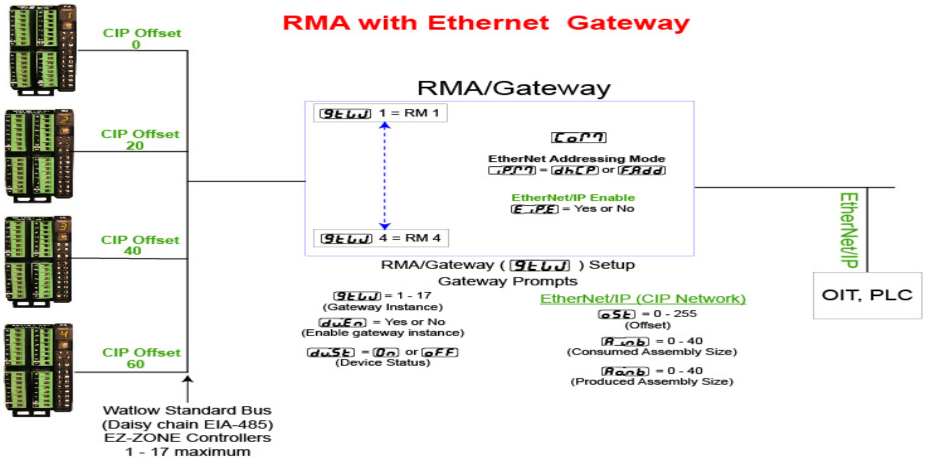
○ Error Code: Extended Error Code: ☐ Timed Out

Error Path:

Error Text:

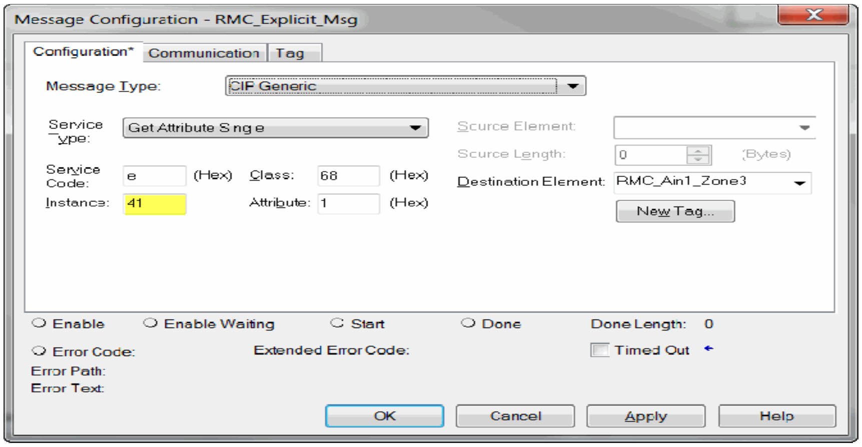
2.4.4 Looking at the graphic in step 2.4.1 when the contact identified as “Read_Ain1_enable” comes on, the message instruction will be executed and the Analog Input will be read from the RMC and then stored in the PLC tag called “RMC_Ain1”.

Note: If there is more than one RMC module on the Standard Bus network and there is a desire to read Analog Input 1 explicitly from each, the CIP Instance Offset prompt must be used. CIP Instance Offset is used exclusively with explicit messages; it’s unique to each gateway instance (module), never overlapping, and defines the module on the network as well as the parameter instance within the module. Using the CIP Instance Offset (entered in decimal format) allows for a message originating from the Master (Originator) to make its way to the appropriate Slave (Target).



Note: If it is desired to use all 40 members of the assemblies the CIP Offset must be changed accordingly: Zone 1 = 0, Zone 2 = 40, Zone 3 = 80 and Zone 4 = 120.

2.4.5 To read Analog Input 1 from the RM module at zone address 3 add the CIP Offset (in this case 40) to the desired instance. The PLC message instruction configuration would change as shown in the graphic below.


Message Configuration - RMC_Explicit_Msg

Configuration* | Communication | Tag

Message Type: CIP Generic

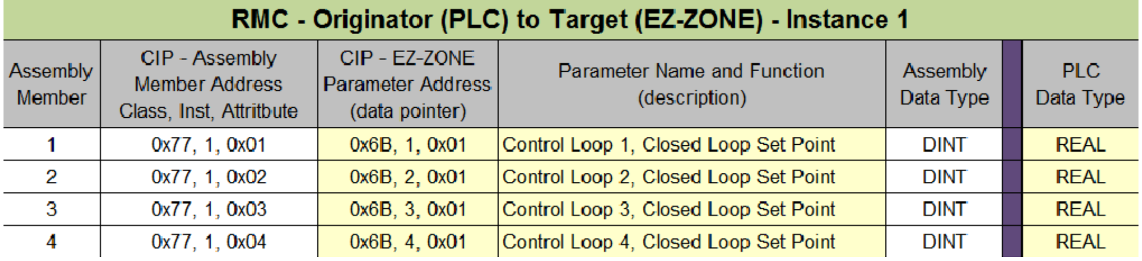
Service Type: Get Attribute Single

Service Code: e (Hex)
Class: 68 (Hex)
Instance: 41
Attribute: 1 (Hex)

Source Element:
Source Length: 0 (Bytes)
Destination Element: RMC_Ain1_Zone3

2.5 Modifying Implicit Assemblies Using Explicit Messages, Step-by-Step

2.5.1 When modifying the implicit assemblies of *multiple* RM modules on the network, the CIP Instance Offset must be used. The first four default members of the RMC Originator (PLC) to Target (EZ-ZONE module) assembly is shown below.

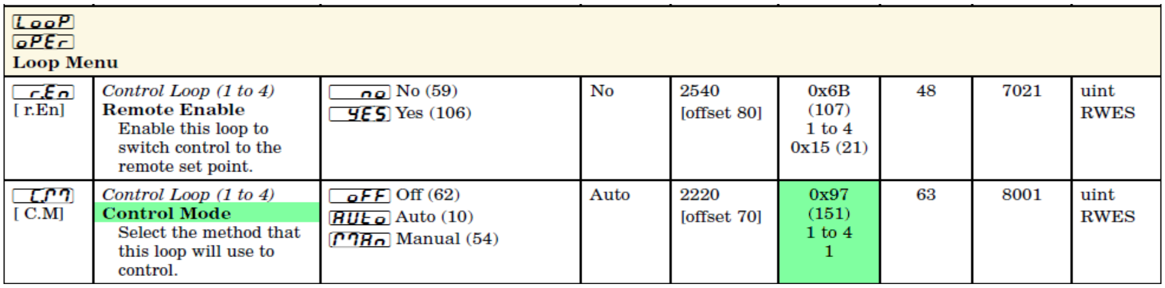


RMC - Originator (PLC) to Target (EZ-ZONE) - Instance 1

Assembly Member	CIP - Assembly Member Address Class, Inst, Attritbute	CIP - EZ-ZONE Parameter Address (data pointer)	Parameter Name and Function (description)	Assembly Data Type	PLC Data Type
1	0x77, 1, 0x01	0x6B, 1, 0x01	Control Loop 1, Closed Loop Set Point	DINT	REAL
2	0x77, 1, 0x02	0x6B, 2, 0x01	Control Loop 2, Closed Loop Set Point	DINT	REAL
3	0x77, 1, 0x03	0x6B, 3, 0x01	Control Loop 3, Closed Loop Set Point	DINT	REAL
4	0x77, 1, 0x04	0x6B, 4, 0x01	Control Loop 4, Closed Loop Set Point	DINT	REAL

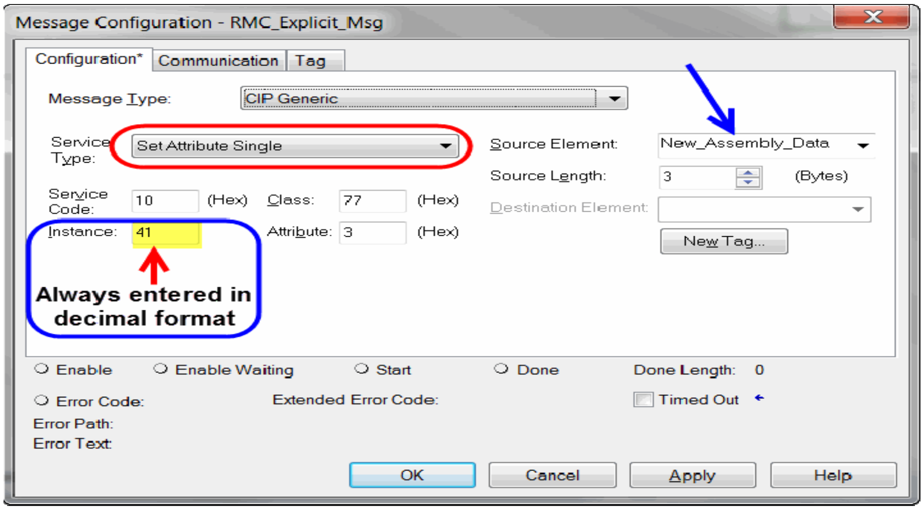
Note: All numbers in the graphic above that are preceded by 0x are in hexadecimal format; numbers without the prefix of 0x are in decimal format.

To change the 3rd Assembly Member at *Zone address 3* from what is shown above to Control Mode loop 4, first find the appropriate CIP address in the RMC User's Guide (shown below).



LooP oPEr **Loop Menu**									
r.En [r.En]	*Control Loop (1 to 4)* **Remote Enable** Enable this loop to switch control to the remote set point.	no No (59) YES Yes (106)	No	2540 [offset 80]	0x6B (107) 1 to 4 0x15 (21)	48	7021	uint RWES
C.M [C.M]	*Control Loop (1 to 4)* **Control Mode** Select the method that this loop will use to control.	oFF Off (62) AUto Auto (10) MAn Manual (54)	Auto	2220 [offset 70]	0x97 (151) 1 to 4 1	63	8001	uint RWES

The explicit message instruction configuration (previously discussed in step 2.4.2) now becomes a set (write) operation while a specific tag must be created which contains the new parameter address pointer (New_Assembly_Data) to be written to the RMC assembly. The message configuration would change as shown below.



Message Configuration - RMC_Explicit_Msg

Configuration* | Communication | Tag

Message Type: CIP Generic

Service Type: Set Attribute Single

Service Code: 10 (Hex) Class: 77 (Hex)

Instance: 41 Attribute: 3 (Hex)

Always entered in decimal format

Source Element: New_Assembly_Data

Source Length: 3 (Bytes)

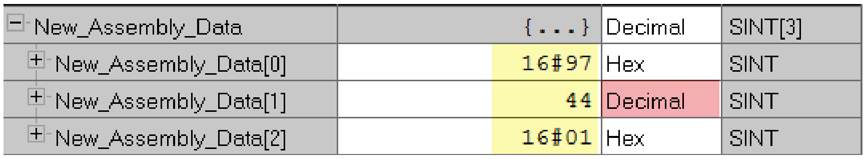
Destination Element:

○ Enable ○ Enable Waiting ○ Start ○ Done Done Length: 0

○ Error Code: Extended Error Code: □ Timed Out

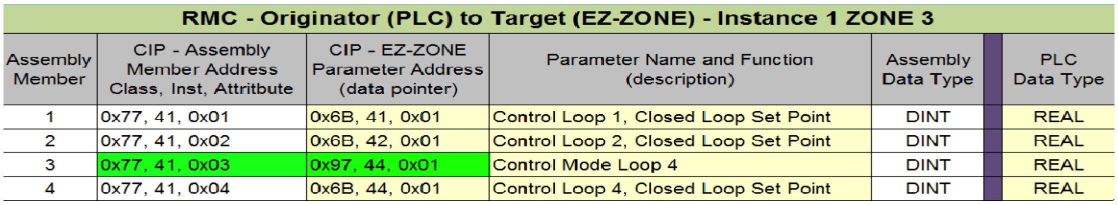
Error Path:
Error Text:

The Source Element was created as a 3 dimensional array using the SINT data type because the Class, Instance and Attribute objects are 8-bits in length.



⊟ New_Assembly_Data	{...}	Decimal	SINT[3]
⊞ New_Assembly_Data[0]	16#97	Hex	SINT
⊞ New_Assembly_Data[1]	44	Decimal	SINT
⊞ New_Assembly_Data[2]	16#01	Hex	SINT

Notice in the graphic above and below that the instance for the 3rd assembly location as well as the parameter to be written have the listed offset (40) added to each.



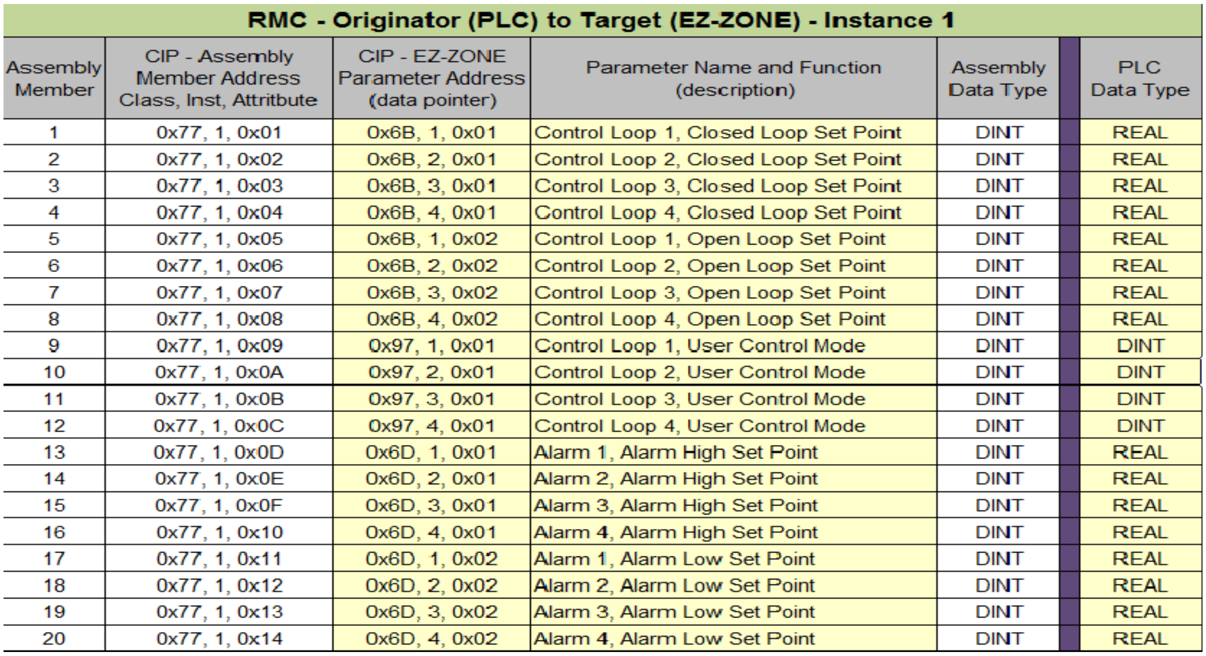
RMC - Originator (PLC) to Target (EZ-ZONE) - Instance 1 ZONE 3					
Assembly Member	CIP - Assembly Member Address Class, Inst, Attritbute	CIP - EZ-ZONE Parameter Address (data pointer)	Parameter Name and Function (description)	Assembly Data Type	PLC Data Type
1	0x77, 41, 0x01	0x6B, 41, 0x01	Control Loop 1, Closed Loop Set Point	DINT	REAL
2	0x77, 41, 0x02	0x6B, 42, 0x01	Control Loop 2, Closed Loop Set Point	DINT	REAL
3	0x77, 41, 0x03	0x97, 44, 0x01	Control Mode Loop 4	DINT	REAL
4	0x77, 41, 0x04	0x6B, 44, 0x01	Control Loop 4, Closed Loop Set Point	DINT	REAL

Note: From the perspective of the RMC module at zone address 3, the O to T assembly instance is 1. However, when reading or modifying it through the gateway, the CIP Instance Offset must be added to the member and parameter instance as shown above.

3.0 Implicit Communications

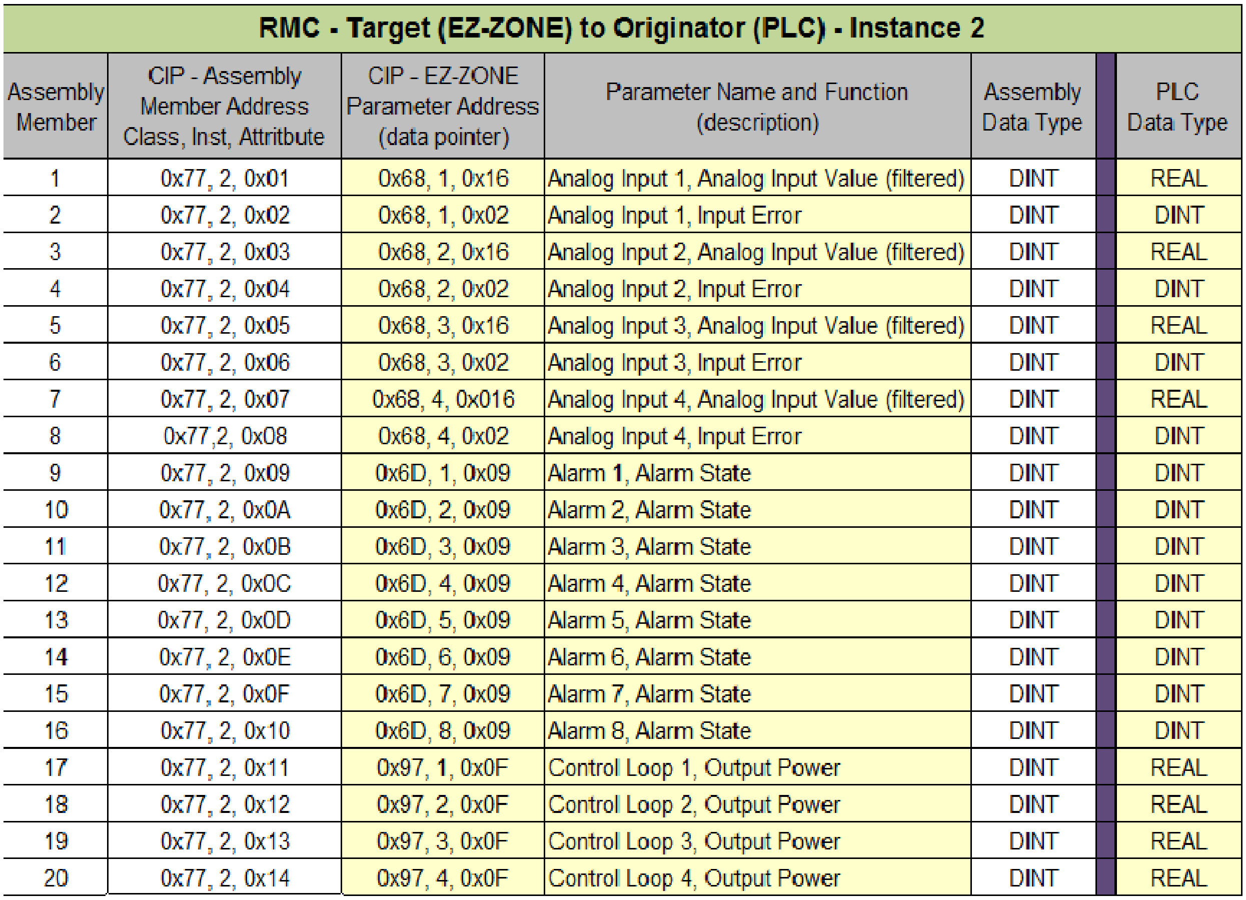
3.1 PLC Configuration

Each RM module has a built-in implicit assembly. The RMC I/O assemblies (shown below) are partially populated by default (20 out of 40 available members). All of the other RM module assemblies are not populated being empty containers by default.



RMC - Originator (PLC) to Target (EZ-ZONE) - Instance 1					
Assembly Member	CIP - Assembly Member Address Class, Inst, Attritbute	CIP - EZ-ZONE Parameter Address (data pointer)	Parameter Name and Function (description)	Assembly Data Type	PLC Data Type
1	0x77, 1, 0x01	0x6B, 1, 0x01	Control Loop 1, Closed Loop Set Point	DINT	REAL
2	0x77, 1, 0x02	0x6B, 2, 0x01	Control Loop 2, Closed Loop Set Point	DINT	REAL
3	0x77, 1, 0x03	0x6B, 3, 0x01	Control Loop 3, Closed Loop Set Point	DINT	REAL
4	0x77, 1, 0x04	0x6B, 4, 0x01	Control Loop 4, Closed Loop Set Point	DINT	REAL
5	0x77, 1, 0x05	0x6B, 1, 0x02	Control Loop 1, Open Loop Set Point	DINT	REAL
6	0x77, 1, 0x06	0x6B, 2, 0x02	Control Loop 2, Open Loop Set Point	DINT	REAL
7	0x77, 1, 0x07	0x6B, 3, 0x02	Control Loop 3, Open Loop Set Point	DINT	REAL
8	0x77, 1, 0x08	0x6B, 4, 0x02	Control Loop 4, Open Loop Set Point	DINT	REAL
9	0x77, 1, 0x09	0x97, 1, 0x01	Control Loop 1, User Control Mode	DINT	DINT
10	0x77, 1, 0x0A	0x97, 2, 0x01	Control Loop 2, User Control Mode	DINT	DINT
11	0x77, 1, 0x0B	0x97, 3, 0x01	Control Loop 3, User Control Mode	DINT	DINT
12	0x77, 1, 0x0C	0x97, 4, 0x01	Control Loop 4, User Control Mode	DINT	DINT
13	0x77, 1, 0x0D	0x6D, 1, 0x01	Alarm 1, Alarm High Set Point	DINT	REAL
14	0x77, 1, 0x0E	0x6D, 2, 0x01	Alarm 2, Alarm High Set Point	DINT	REAL
15	0x77, 1, 0x0F	0x6D, 3, 0x01	Alarm 3, Alarm High Set Point	DINT	REAL
16	0x77, 1, 0x10	0x6D, 4, 0x01	Alarm 4, Alarm High Set Point	DINT	REAL
17	0x77, 1, 0x11	0x6D, 1, 0x02	Alarm 1, Alarm Low Set Point	DINT	REAL
18	0x77, 1, 0x12	0x6D, 2, 0x02	Alarm 2, Alarm Low Set Point	DINT	REAL
19	0x77, 1, 0x13	0x6D, 3, 0x02	Alarm 3, Alarm Low Set Point	DINT	REAL
20	0x77, 1, 0x14	0x6D, 4, 0x02	Alarm 4, Alarm Low Set Point	DINT	REAL

The first 20 members of the RMC default T to O assembly are shown below; the remaining 20 are undefined.

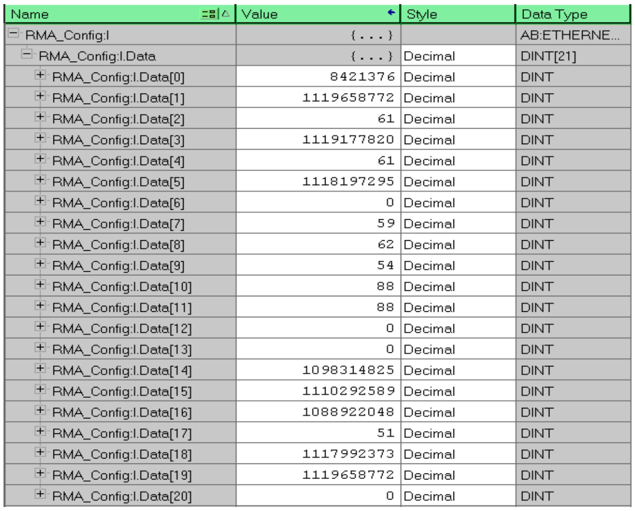


RMC - Target (EZ-ZONE) to Originator (PLC) - Instance 2					
Assembly Member	CIP - Assembly Member Address Class, Inst, Attribute	CIP - EZ-ZONE Parameter Address (data pointer)	Parameter Name and Function (description)	Assembly Data Type	PLC Data Type
1	0x77, 2, 0x01	0x68, 1, 0x16	Analog Input 1, Analog Input Value (filtered)	DINT	REAL
2	0x77, 2, 0x02	0x68, 1, 0x02	Analog Input 1, Input Error	DINT	DINT
3	0x77, 2, 0x03	0x68, 2, 0x16	Analog Input 2, Analog Input Value (filtered)	DINT	REAL
4	0x77, 2, 0x04	0x68, 2, 0x02	Analog Input 2, Input Error	DINT	DINT
5	0x77, 2, 0x05	0x68, 3, 0x16	Analog Input 3, Analog Input Value (filtered)	DINT	REAL
6	0x77, 2, 0x06	0x68, 3, 0x02	Analog Input 3, Input Error	DINT	DINT
7	0x77, 2, 0x07	0x68, 4, 0x016	Analog Input 4, Analog Input Value (filtered)	DINT	REAL
8	0x77,2, 0x08	0x68, 4, 0x02	Analog Input 4, Input Error	DINT	DINT
9	0x77, 2, 0x09	0x6D, 1, 0x09	Alarm 1, Alarm State	DINT	DINT
10	0x77, 2, 0x0A	0x6D, 2, 0x09	Alarm 2, Alarm State	DINT	DINT
11	0x77, 2, 0x0B	0x6D, 3, 0x09	Alarm 3, Alarm State	DINT	DINT
12	0x77, 2, 0x0C	0x6D, 4, 0x09	Alarm 4, Alarm State	DINT	DINT
13	0x77, 2, 0x0D	0x6D, 5, 0x09	Alarm 5, Alarm State	DINT	DINT
14	0x77, 2, 0x0E	0x6D, 6, 0x09	Alarm 6, Alarm State	DINT	DINT
15	0x77, 2, 0x0F	0x6D, 7, 0x09	Alarm 7, Alarm State	DINT	DINT
16	0x77, 2, 0x10	0x6D, 8, 0x09	Alarm 8, Alarm State	DINT	DINT
17	0x77, 2, 0x11	0x97, 1, 0x0F	Control Loop 1, Output Power	DINT	REAL
18	0x77, 2, 0x12	0x97, 2, 0x0F	Control Loop 2, Output Power	DINT	REAL
19	0x77, 2, 0x13	0x97, 3, 0x0F	Control Loop 3, Output Power	DINT	REAL
20	0x77, 2, 0x14	0x97, 4, 0x0F	Control Loop 4, Output Power	DINT	REAL

3.1.1 In step 2.2.6 above, the Implicit Assembly (input and output) was configured for 20 members. With the RMA configuration complete, it is time to look closer at the PLC side.

3.1.2 In step 2.3.2 above, the PLC was configured to include a Generic Ethernet module. The module parameters, specifically, the assembly sizes were configured at this time to be the same as the RMA.

3.1.3 Prior to making any changes to the current PLC configuration let’s look closer at what is currently there. Recall that when a Generic Ethernet module was added to the PLC I/O structure (step 2.3.4) it was also given a name (RMA_Config) and that name became a controller tag. While being connected to the PLC on-line and with the PLC in the run mode the input tag (RMA_Config:I) clearly shows that it is dynamically receiving raw data from the Slave (see the screenshot below).

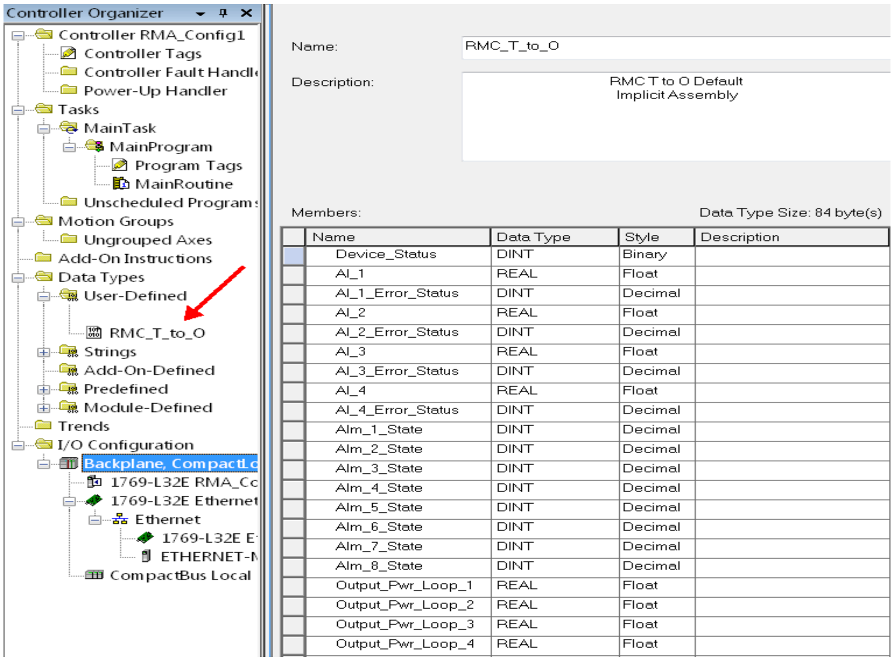


Name	Value	Style	Data Type
RMA_Config:I	{...}		AB:ETHERNE...
RMA_Config:I.Data	{...}	Decimal	DINT[21]
RMA_Config:I.Data[0]	8421376	Decimal	DINT
RMA_Config:I.Data[1]	1119658772	Decimal	DINT
RMA_Config:I.Data[2]	61	Decimal	DINT
RMA_Config:I.Data[3]	1119177820	Decimal	DINT
RMA_Config:I.Data[4]	61	Decimal	DINT
RMA_Config:I.Data[5]	1118197295	Decimal	DINT
RMA_Config:I.Data[6]	0	Decimal	DINT
RMA_Config:I.Data[7]	59	Decimal	DINT
RMA_Config:I.Data[8]	62	Decimal	DINT
RMA_Config:I.Data[9]	54	Decimal	DINT
RMA_Config:I.Data[10]	88	Decimal	DINT
RMA_Config:I.Data[11]	88	Decimal	DINT
RMA_Config:I.Data[12]	0	Decimal	DINT
RMA_Config:I.Data[13]	0	Decimal	DINT
RMA_Config:I.Data[14]	1098314825	Decimal	DINT
RMA_Config:I.Data[15]	1110292589	Decimal	DINT
RMA_Config:I.Data[16]	1088922048	Decimal	DINT
RMA_Config:I.Data[17]	51	Decimal	DINT
RMA_Config:I.Data[18]	1117992373	Decimal	DINT
RMA_Config:I.Data[19]	1119658772	Decimal	DINT
RMA_Config:I.Data[20]	0	Decimal	DINT

3.2 PLC Programming

Being that the graphic above shows the T to O assembly, 21 members are present. Some of the values coming in reflect numbers that are expected (members [0], [2] and [4]) but some do not (members [1], [3] and [5]) due to the data format shown and the data format of the given parameter.

3.2.1 To start the programming process, it is suggested that a User Defined Data Type be created for both implicit assemblies which will reflect the appropriate data format and will also simplify the programming when transferring I/O data between Master (PLC) and Slave (RMC module) Device. As can be seen below, a user defined data type was created (RMC_T_to_O) using the RMC default assembly.



Controller Organizer
- Controller RMA_Config1
  - Controller Tags
  - Controller Fault Handl
  - Power-Up Handler
- Tasks
  - MainTask
    - MainProgram
      - Program Tags
      - MainRoutine
  - Unscheduled Program
- Motion Groups
  - Ungrouped Axes
- Add-On Instructions
- Data Types
  - User-Defined
    - RMC_T_to_O
  - Strings
  - Add-On-Defined
  - Predefined
  - Module-Defined
- Trends
- I/O Configuration
  - Backplane, CompactL
    - 1769-L32E RMA_Co
    - 1769-L32E Ethernet
      - Ethernet
        - 1769-L32E E
        - ETHERNET-N
    - CompactBus Local

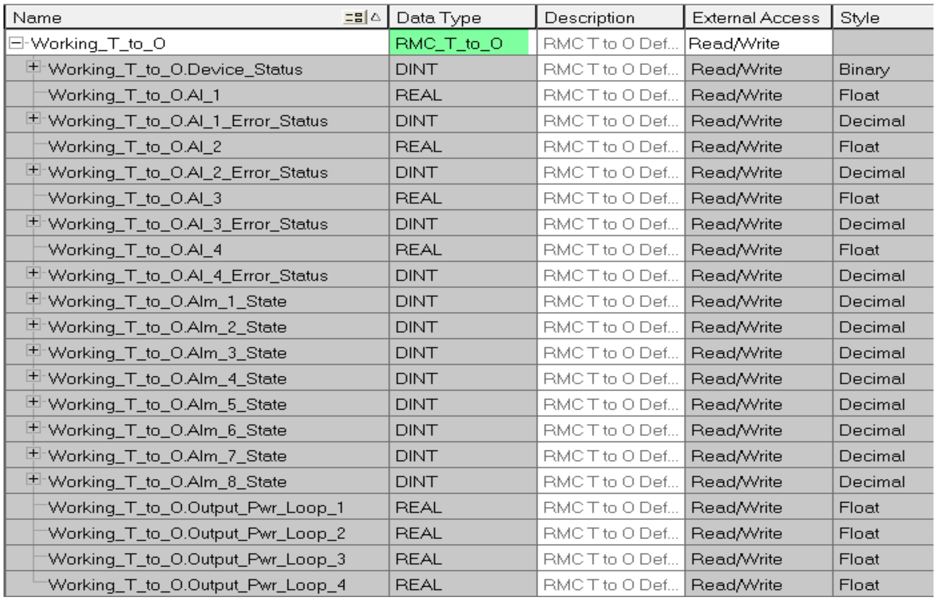
Name: RMC_T_to_O

Description: RMC T to O Default Implicit Assembly

Members: Data Type Size: 84 byte(s)

Name	Data Type	Style	Description
Device_Status	DINT	Binary	
AI_1	REAL	Float	
AI_1_Error_Status	DINT	Decimal	
AI_2	REAL	Float	
AI_2_Error_Status	DINT	Decimal	
AI_3	REAL	Float	
AI_3_Error_Status	DINT	Decimal	
AI_4	REAL	Float	
AI_4_Error_Status	DINT	Decimal	
Alm_1_State	DINT	Decimal	
Alm_2_State	DINT	Decimal	
Alm_3_State	DINT	Decimal	
Alm_4_State	DINT	Decimal	
Alm_5_State	DINT	Decimal	
Alm_6_State	DINT	Decimal	
Alm_7_State	DINT	Decimal	
Alm_8_State	DINT	Decimal	
Output_Pwr_Loop_1	REAL	Float	
Output_Pwr_Loop_2	REAL	Float	
Output_Pwr_Loop_3	REAL	Float	
Output_Pwr_Loop_4	REAL	Float	

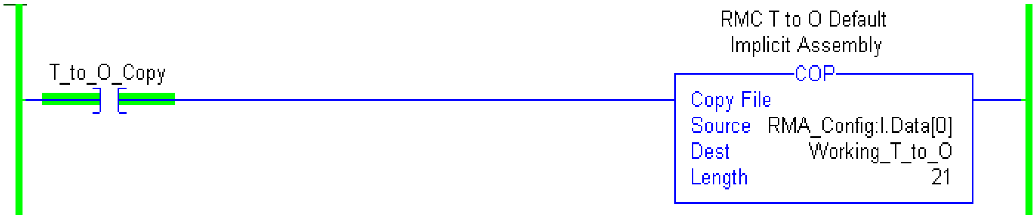
3.2.2 Once created, a controller tag should be created using the User Defined Data Type as its data type. Below, a controller tag was created (Working_T_to_O) where the data type (green highlight) is the User Defined Data Type created in the previous step.



Name	Data Type	Description	External Access	Style
⊟ Working_T_to_O	RMC_T_to_O	RMC T to O Def...	Read/Write	
⊞ Working_T_to_O.Device_Status	DINT	RMC T to O Def...	Read/Write	Binary
Working_T_to_O.AI_1	REAL	RMC T to O Def...	Read/Write	Float
⊞ Working_T_to_O.AI_1_Error_Status	DINT	RMC T to O Def...	Read/Write	Decimal
Working_T_to_O.AI_2	REAL	RMC T to O Def...	Read/Write	Float
⊞ Working_T_to_O.AI_2_Error_Status	DINT	RMC T to O Def...	Read/Write	Decimal
Working_T_to_O.AI_3	REAL	RMC T to O Def...	Read/Write	Float
⊞ Working_T_to_O.AI_3_Error_Status	DINT	RMC T to O Def...	Read/Write	Decimal
Working_T_to_O.AI_4	REAL	RMC T to O Def...	Read/Write	Float
⊞ Working_T_to_O.AI_4_Error_Status	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_1_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_2_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_3_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_4_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_5_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_6_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_7_State	DINT	RMC T to O Def...	Read/Write	Decimal
⊞ Working_T_to_O.Alm_8_State	DINT	RMC T to O Def...	Read/Write	Decimal
Working_T_to_O.Output_Pwr_Loop_1	REAL	RMC T to O Def...	Read/Write	Float
Working_T_to_O.Output_Pwr_Loop_2	REAL	RMC T to O Def...	Read/Write	Float
Working_T_to_O.Output_Pwr_Loop_3	REAL	RMC T to O Def...	Read/Write	Float
Working_T_to_O.Output_Pwr_Loop_4	REAL	RMC T to O Def...	Read/Write	Float

The example above was based on the default RMC T to O assembly. This assembly will typically be modified by each user and would look different based on each assembly configuration and the module in use.

3.2.3 Enter the rung of logic shown below to read the Target data into the PLC tag created above.

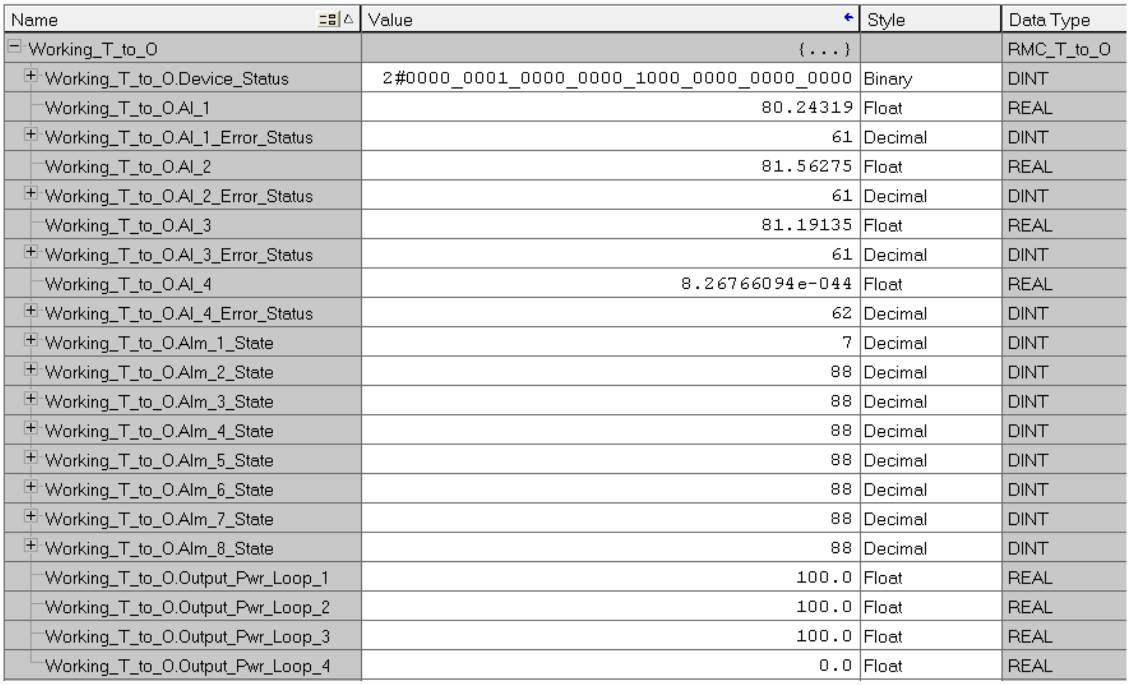


RMC T to O Default
Implicit Assembly

T_to_O_Copy

COP
Copy File
Source RMA_Config:I.Data[0]
Dest Working_T_to_O
Length 21

Notice that the source of the copy instruction has the same name as the name given to the module back in step 2.3.4. Recall that when the module was added that there were entries for the input, output and configuration assemblies. The input assembly was defined as having 21 members as was the destination tag. Therefore, the length is defined as 21. Once the contact (T_to_O_Copy) is enabled the source data will be copied to the destination as can be seen below.



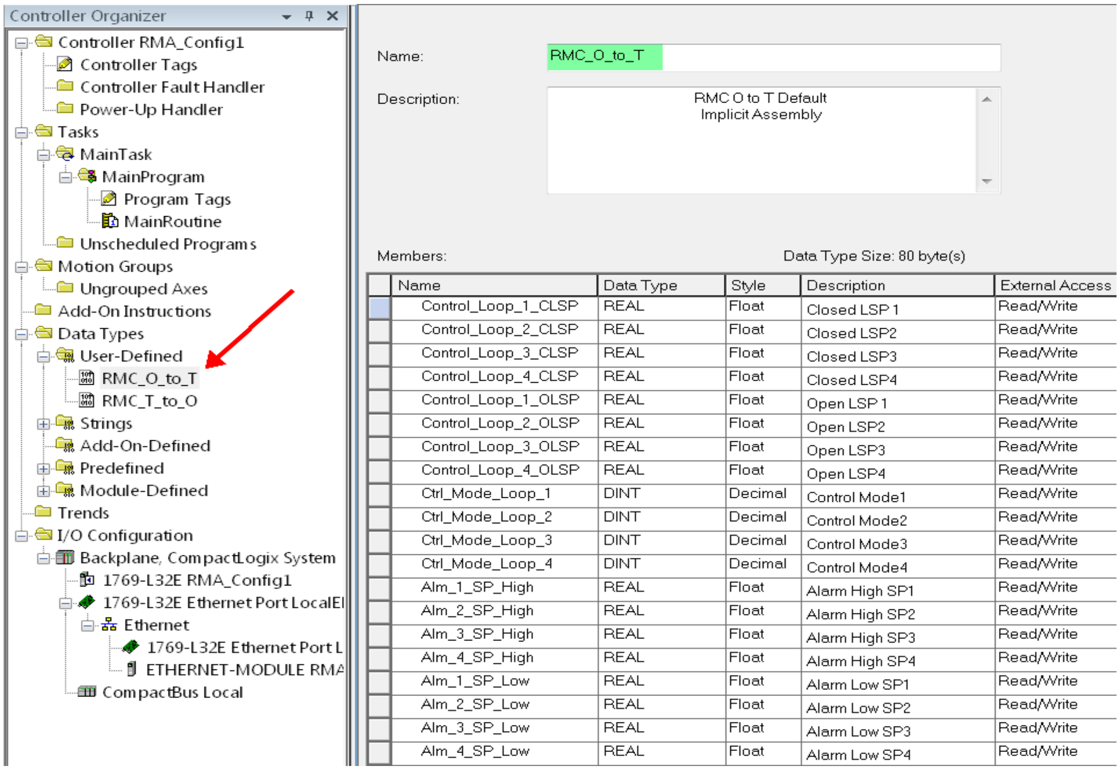
Name	Value	Style	Data Type
⊟ Working_T_to_O	{...}		RMC_T_to_O
⊞ Working_T_to_O.Device_Status	2#0000_0001_0000_0000_1000_0000_0000_0000	Binary	DINT
Working_T_to_O.AI_1	80.24319	Float	REAL
⊞ Working_T_to_O.AI_1_Error_Status	61	Decimal	DINT
Working_T_to_O.AI_2	81.56275	Float	REAL
⊞ Working_T_to_O.AI_2_Error_Status	61	Decimal	DINT
Working_T_to_O.AI_3	81.19135	Float	REAL
⊞ Working_T_to_O.AI_3_Error_Status	61	Decimal	DINT
Working_T_to_O.AI_4	8.26766094e-044	Float	REAL
⊞ Working_T_to_O.AI_4_Error_Status	62	Decimal	DINT
⊞ Working_T_to_O.Alm_1_State	7	Decimal	DINT
⊞ Working_T_to_O.Alm_2_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_3_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_4_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_5_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_6_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_7_State	88	Decimal	DINT
⊞ Working_T_to_O.Alm_8_State	88	Decimal	DINT
Working_T_to_O.Output_Pwr_Loop_1	100.0	Float	REAL
Working_T_to_O.Output_Pwr_Loop_2	100.0	Float	REAL
Working_T_to_O.Output_Pwr_Loop_3	100.0	Float	REAL
Working_T_to_O.Output_Pwr_Loop_4	0.0	Float	REAL

Now that the data formats correspond to each parameter (member) data type we see values that are more in alignment with expectations. As can be seen in the 8th member down from the top (AI_4), there appears to be an error in the incoming value. In this case, it is not an error because the RMC in use has only 3 analog inputs. With this being the case, the member that follows it is also inaccurate. This illustrates why a user would want to modify the default assembly to their specific hardware.

Again, looking at the graphic above, notice the first member referred to as "Device Status". This member is sourced from the RMA module. Regardless of how many RM modules are connected to the RMA there is only one assembly member referred to as the Device Status. Counting the bits from right to left, bit 15 is shown as being set to a "1". This bit represents the RMA module being present in the PLC I/O structure. Bits 16 through 31 represent gateway instances (EZ-ZONE modules on Standard Bus). Once the gateway instance is enabled, as long as there is successful communications to the RMA module the corresponding bit will be set to a one. Above, bit 24 (RMC Standard Bus address 9) is set to a "1" because it was enabled and is communicating successfully with the RMA module.

Note: The other bits (16 through 31) being "0" could represent a module problem or may simply mean those instances have not been enabled.

3.2.4 To write data out to the Target, perform the same steps in the creation of the O to T assembly; as can be seen below, the User Defined Data Type is created based on the RMC default. Keep in mind that this assembly represents the output from the PLC or Originator which will be sent to the Target or EZ-ZONE module. The program within the PLC would write values to these tags and they would then be sent out to the Target at the rate of the setting for the RPI (see step 2.3.6).



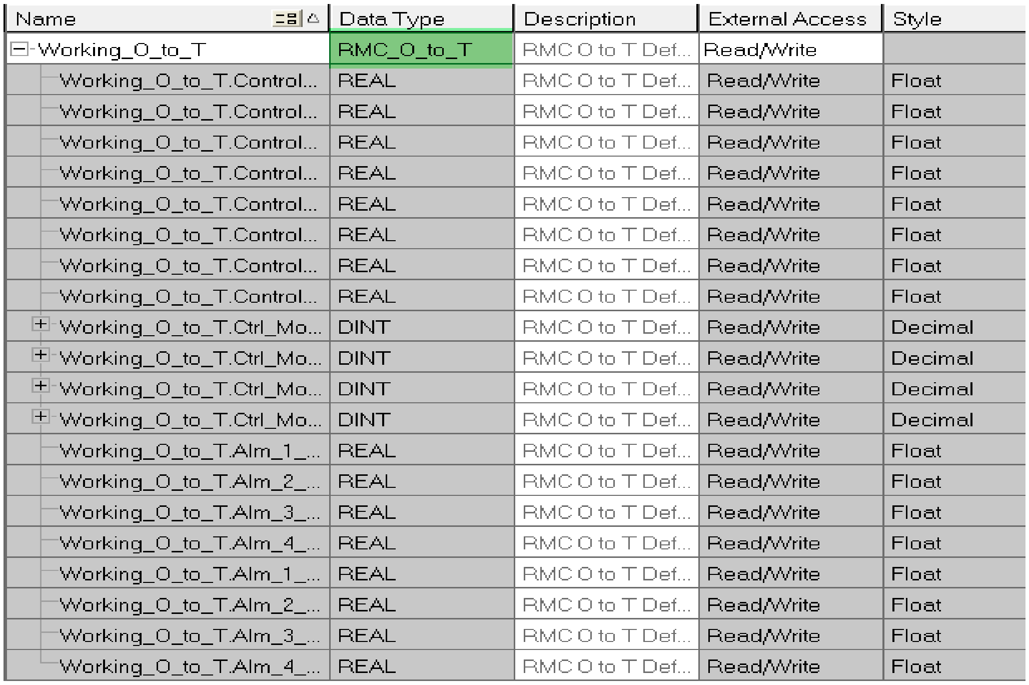
Name: RMC_O_to_T

Description: RMC O to T Default
Implicit Assembly

Members: Data Type Size: 80 byte(s)

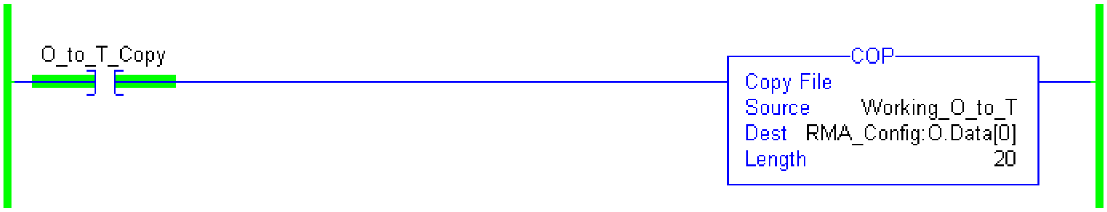
Name	Data Type	Style	Description	External Access
Control_Loop_1_CLSP	REAL	Float	Closed LSP 1	Read/Write
Control_Loop_2_CLSP	REAL	Float	Closed LSP2	Read/Write
Control_Loop_3_CLSP	REAL	Float	Closed LSP3	Read/Write
Control_Loop_4_CLSP	REAL	Float	Closed LSP4	Read/Write
Control_Loop_1_OLSP	REAL	Float	Open LSP 1	Read/Write
Control_Loop_2_OLSP	REAL	Float	Open LSP2	Read/Write
Control_Loop_3_OLSP	REAL	Float	Open LSP3	Read/Write
Control_Loop_4_OLSP	REAL	Float	Open LSP4	Read/Write
Ctrl_Mode_Loop_1	DINT	Decimal	Control Mode1	Read/Write
Ctrl_Mode_Loop_2	DINT	Decimal	Control Mode2	Read/Write
Ctrl_Mode_Loop_3	DINT	Decimal	Control Mode3	Read/Write
Ctrl_Mode_Loop_4	DINT	Decimal	Control Mode4	Read/Write
Alm_1_SP_High	REAL	Float	Alarm High SP1	Read/Write
Alm_2_SP_High	REAL	Float	Alarm High SP2	Read/Write
Alm_3_SP_High	REAL	Float	Alarm High SP3	Read/Write
Alm_4_SP_High	REAL	Float	Alarm High SP4	Read/Write
Alm_1_SP_Low	REAL	Float	Alarm Low SP1	Read/Write
Alm_2_SP_Low	REAL	Float	Alarm Low SP2	Read/Write
Alm_3_SP_Low	REAL	Float	Alarm Low SP3	Read/Write
Alm_4_SP_Low	REAL	Float	Alarm Low SP4	Read/Write

3.2.5 The corresponding O to T controller tag is created as shown below.



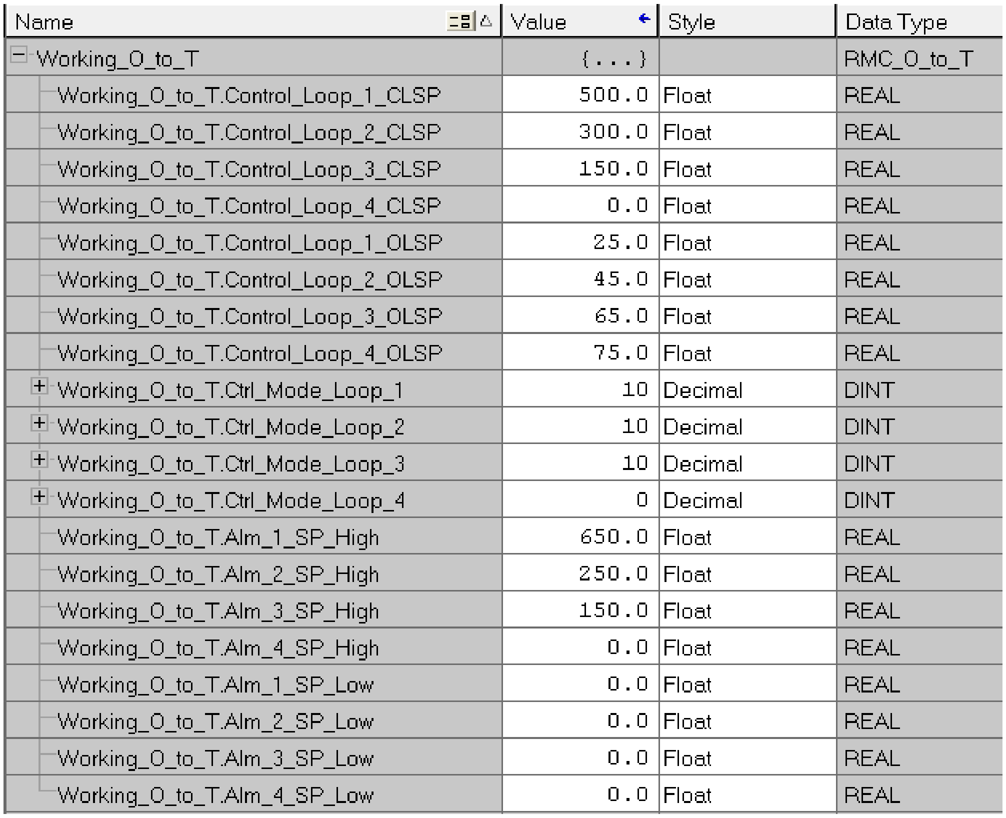
Name	Data Type	Description	External Access	Style
Working_O_to_T	RMC_O_to_T	RMC O to T Def...	Read/Write	
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Control...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Ctrl_Mo...	DINT	RMC O to T Def...	Read/Write	Decimal
Working_O_to_T.Ctrl_Mo...	DINT	RMC O to T Def...	Read/Write	Decimal
Working_O_to_T.Ctrl_Mo...	DINT	RMC O to T Def...	Read/Write	Decimal
Working_O_to_T.Ctrl_Mo...	DINT	RMC O to T Def...	Read/Write	Decimal
Working_O_to_T.Alm_1_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_2_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_3_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_4_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_1_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_2_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_3_...	REAL	RMC O to T Def...	Read/Write	Float
Working_O_to_T.Alm_4_...	REAL	RMC O to T Def...	Read/Write	Float

3.2.6 Enter the rung of logic shown below to write data from the Originator to the Target.



Copy File
Source Working_O_to_T
Dest RMA_Config:O.Data[0]
Length 20

Notice that the source of the copy instruction is now the controller tag created above where the destination is the same name given to the module back in step 2.3.4. In this case, the output assembly was defined as having 20 members as was the destination tag. Therefore, the length is defined as 20. Once the contact (O_to_T_Copy) is enable the source data will be sent to the destination as can be seen below.



Name	Value	Style	Data Type
⊟ Working_O_to_T	{...}		RMC_O_to_T
Working_O_to_T.Control_Loop_1_CLSP	500.0	Float	REAL
Working_O_to_T.Control_Loop_2_CLSP	300.0	Float	REAL
Working_O_to_T.Control_Loop_3_CLSP	150.0	Float	REAL
Working_O_to_T.Control_Loop_4_CLSP	0.0	Float	REAL
Working_O_to_T.Control_Loop_1_OLSP	25.0	Float	REAL
Working_O_to_T.Control_Loop_2_OLSP	45.0	Float	REAL
Working_O_to_T.Control_Loop_3_OLSP	65.0	Float	REAL
Working_O_to_T.Control_Loop_4_OLSP	75.0	Float	REAL
⊞ Working_O_to_T.Ctrl_Mode_Loop_1	10	Decimal	DINT
⊞ Working_O_to_T.Ctrl_Mode_Loop_2	10	Decimal	DINT
⊞ Working_O_to_T.Ctrl_Mode_Loop_3	10	Decimal	DINT
⊞ Working_O_to_T.Ctrl_Mode_Loop_4	0	Decimal	DINT
Working_O_to_T.Alm_1_SP_High	650.0	Float	REAL
Working_O_to_T.Alm_2_SP_High	250.0	Float	REAL
Working_O_to_T.Alm_3_SP_High	150.0	Float	REAL
Working_O_to_T.Alm_4_SP_High	0.0	Float	REAL
Working_O_to_T.Alm_1_SP_Low	0.0	Float	REAL
Working_O_to_T.Alm_2_SP_Low	0.0	Float	REAL
Working_O_to_T.Alm_3_SP_Low	0.0	Float	REAL
Working_O_to_T.Alm_4_SP_Low	0.0	Float	REAL