CIP Over Ethernet

Application Specific Function Block Manual

Version 2.2

G & L Motion Control Inc.

NOTE

These products are being manufactured and sold by G & L Motion Control Inc., a Danaher Motion company (“Danaher Motion”).

Progress is an on-going commitment at Danaher Motion. We continually strive to offer the most advanced products in the industry; therefore, information in this document is subject to change without notice. The text and illustrations are not binding in detail. Danaher Motion shall not be liable for any technical or editorial omissions occurring in this document, nor for any consequential or incidental damages resulting from the use of this document.

DO NOT ATTEMPT to use any Danaher Motion product until the use of such product is completely understood. It is the responsibility of the user to make certain proper operation practices are understood. Danaher Motion products should be used only by qualified personnel and for the express purpose for which said products were designed.

Should information not covered in this document be required, contact the Customer Service Department, Danaher Motion, 672 South Military Road, P.O. Box 1960, Fond du Lac, WI 54936-1960. Danaher Motion can be reached by telephone at (920) 921–7100 or (800) 558-4808 in the United States or by e-mail at glmotion.support@danahermotion.com.

DISCLAIMER: All programs in this release (application demos, application specific function blocks (ASFB's), etc.), are provided "AS IS, WHERE IS", WITHOUT ANY WARRANTIES, EXPRESS OR IMPLIED. There may be technical or editorial omissions in the programs and their specifications. These programs are provided solely for user application development and user assumes all responsibility for their use. Programs and their content are subject to change without notice.

Electronic Version Part No. M.3000.0511

Release 292007

Table of Contents: CIP Over Ethernet ASFBs

NOTES

CHAPTER 1 Application Specific Function Block Guidelines

Installation

The following guidelines are recommended ways of working with Application Specific Function Blocks (i.e. ASFBs) from G&L Motion Control.

The Applications CD includes the ASFB package as follows:

- .LIB file(s) containing the ASFB(s)
- source .LDO(s) from which the ASFB(s) was made
- example LDO(s) with the ASFB(s) incorporated into the ladder which you can then use to begin programming from or merge with an existing application ladder

When you install the Applications CD, the ASFB paths default to:

C:\G&L Motion Control Data\CIP Over Ethernet ASFB **v*xx.x.r***\ASFB

and

C:\G&L Motion Control Data\CIP Over Ethernet ASFB **v*xx.x.r***\Examples

where **v*xx.x*** *is the PiCPro for Windows version number that these ASFBs and examples were built under. The* ***.r*** *is the revision number of the Application software itself.*

The .LIB files and source .LDO files are put in the ASFB subdirectory. The example .LDO files are put in the Examples subdirectory.

Revisions

The first four networks of each ASFB source ladder provide the following information:

Network 1

The first network just informs you that the ASFB is provided to assist your application development.

Network 2

The second network is used to keep a revision history of the ASFB. Revisions can be made by G&L Motion Control personnel or by you.

The network identifies the ASFB, lists the requirements for using this ASFB, the name of the library the ASFB is stored in, and the revision history.

The revision history includes the date, ASFB version (see below), the version of PiCPro used while making the ASFB, and comments about what the revision involved.

When an ASFB is revised, the number of the first input (EN_ _ or RQ_ _) to the function block is changed in the software declarations table. The range of numbers available for G&L Motion Control personnel is 00 to 49. The range of numbers available for you is 50 to 99. See chart below.

Revision	G&L Motion Control revisions	User revisions
1st	EN00	EN50
2nd	EN01	EN51
.	.	.
.	.	.
.	.	.
50th	EN49	EN99

Network 3

The third network describes what you should do if you want to make a revision to the ASFB.

ASFB Input/Output Descriptions

Network 4

The fourth network describes the ASFB and defines all the inputs and outputs to the function block.

Using ASFBs

When you are ready to use the ASFB in your application, there are several approaches you can take as shown below.

- Create a new application LDO starting with the example LDO for the ASFB package. The advantage is that the software declarations table for the ASFB has been entered for you.
- If you already have an application LDO, copy and paste the example LDO into yours. The software declaration tables for both LDOs will also merge.

CHAPTER 2 Configuration and Software Installation

Introduction

The ControlLogix™ CIP Over Ethernet ASFB software package from G&L Motion Control allows the Digital MMC, MMC, MMC for PC, or PiC900/90 to communicate with an Allen-Bradley ControlLogix Logix5000 Controller using the CIP (Control and Information Protocol) Services over Ethernet. The services used are the CIP Read Data Service and the CIP Write Data Service.

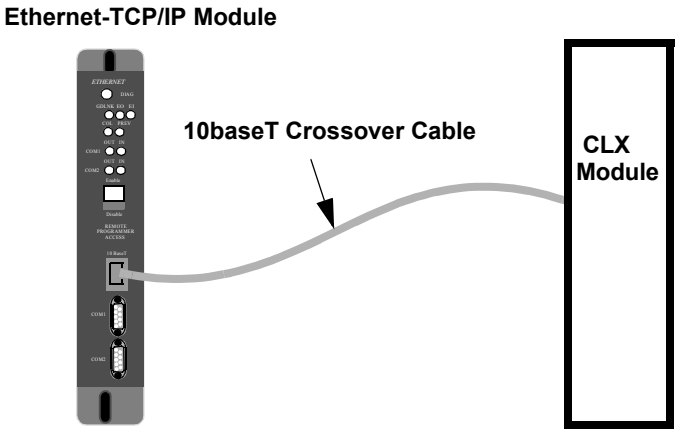
Through out this document GLMC is used as the generic description for the G&L Motion Control Digital MMC, MMC, MMC for PC, or PiC900/90 control platforms. CLX is used to describe the Allen-Bradley ControlLogix Logix5000 Controller.

When communicating with the CLX the GLMC can be programmed to be either a Server or a Client. When the GLMC is a Server it will respond to CIP Generic MSG functions. This function can be used to read and/or write data from/to the GLMC. When the GLMC is a Client it will issue read/write commands to the CLX.

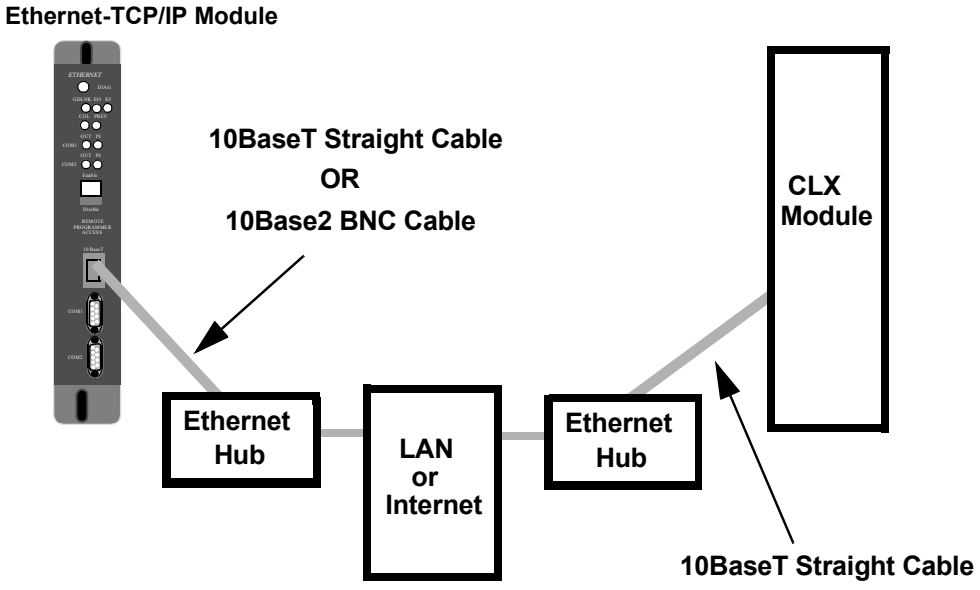
For more information on CIP and the ControlLogix, contact Rockwell Automation.

Refer to the Digital MMC, MMC, MMC for PC and PiC900 Hardware Manuals and the CLX hardware documentation for detailed information on configuring the hardware.

The GLMC and CLX Ethernet modules can be connected directly together using a 10baseT crossover cable.



If you are communicating over a plant network you can connect the GLMC to an Ethernet HUB using a 10baseT straight cable or a 10Base2 BNC cable. The CLX is then connected to an Ethernet HUB using a 10baseT straight cable. Both of the Hub's would then be connected to the plant network.



GLMC Server - CLX Client Configurations

There are several different Client/Server configurations possible between the GLMC and CLX. When the GLMC is a Server it will respond to read/write requests from one or more CLX's which are the Client(s). The GLMC ladder will require two ASFB's to operate as the server. The E_CIP_CM ASFB is used to manage the TCP/IP Connection between the GLMC and one or more CLX's. The E_CIP_MR function is the Message Router for processing CIP over Ethernet requests from a CLX.

There are example ladders included for the GLMC, E_CLSVEX.LDO/REM and CLX, E_CLSVEX.ACD. These examples show the logic needed for the application. You can cut and paste the networks required from the example ladders into your ladders. This will speed up your development time and eliminate typing errors caused by re-entering the logic.

The following is a list of requirements for configuring the GLMC Server – CLX Client.

1. A TCP/IP connection is a point to point connection between an Ethernet card in the GLMC and an Ethernet card in the CLX.
2. One or more Ethernet cards in the GLMC.
3. One or more Ethernet cards in the CLX.
4. You will need one instance of the E_CIP_CM ASFB in the GLMC ladder for each Ethernet card in the GLMC used as a Server. In most cases the GLMC will only have one Ethernet card, thus you would have one E_CIP_CM in your ladder. To improve system through put you can put more Ethernet cards in the GLMC, thus you would then have multiple instances of the E_CIP_CM. A single instance of the E_CIP_CM can handle up to 64 connections.
5. You will need one instance of the E_CIP_MR in the GLMC ladder for each CLX Ethernet card used by a CLX Client. In most cases each CLX will contain a single Ethernet card, thus each CLX system or rack would require a separate instance of the E_CIP_MR in the GLMC ladder. If you have one CLX system you need one instance of E_CIP_MR. If you have two CLX systems you need two instances of E_CIP_MR, on so on, up to a maximum of 64 connections or E_CIP_MR functions.
6. The CLX can have one or more CPU's in one rack all communicating to the GLMC via the same Ethernet card. Each CLX CPU would communicate with the GLMC via the same E_CIP_MR ASFB.
7. The CLX can have multiple CPU's and multiple Ethernet cards in one rack all communicating to the GLMC. Each CLX CPU/Ethernet combination would a separate instance of the E_CIP_MR ASFB.
8. The CLX CIP over Ethernet protocol requires that the Server or the GLMC E_CIP_CM, use IP Port Number 16#AF12. This port should be reserved for this protocol and not used for any other Ethernet connection.

9. The CLX will use the CIP Generic MSG function to read/write data to/from the GLMC. The CIP Generic MSG function can be configured to read and write data, to only read data, or to only write data in one message. Each message can contain a maximum of 432 bytes of read data and/or 432 bytes for write data.

GLMC Client - CLX Server Configurations

When the GLMC is the Client the E_CIP_CL function will be used to issue read/write commands to the CLX. The CLX does not require any additional functions when it is the Server.

There are example ladders included for the GLMC, E_CLCLEX.LDO/REM and CLX, E_CLCLEX.ACD. These examples show the logic needed for the application. You can cut and paste the networks required from the example ladders into your ladders. This will speed up your development time and eliminate typing errors caused by re-entering the logic.

The following is a list of requirements for configuring the GLMC Client and CLX Server.

1. One or more Ethernet cards in the GLMC.
2. One or more Ethernet cards in the CLX.
3. A separate instance of the E_CIP_CL function is required for each CLX CPU that you want to talk to. This is true even if the CPU's are in the same rack.
4. The CLX CIP over Ethernet protocol requires that the Server or the CLX use IP Port Number 16#AF12. This port should be reserved for this protocol and not used for any other Ethernet connection.
5. The E_CIP_CL function can only read or write data in a single message.
6. Each message can contain a maximum of 432 bytes of read data or 432 bytes for write data.
7. The CLX does not require any additional functions to operate as a Server. The only requirement is that the Controller Tag that the GLMC is accessing does exist.

Software Requirements

- G&L Motion Control ControlLogix CIP over Ethernet ASFB software
- G&L Motion Control PiCPro V16.1 or higher Programming Software
- Rockwell RSLinx and RSLogix 5000 Programming Software

Insert the G&L Motion Control ControlLogix CIP over Ethernet ASFB software CD. If the CD doesn't auto run go to Start-Run-Setup.exe to install the software. The files will be copied to the default folder c:\G&L Motion Control Data\CIP Over Ethernet ASFB ***vxx.x.r***\Examples. During installation you can change the destination folder.

NOTES

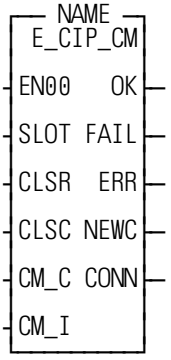
CHAPTER 3 CIP Over Ethernet ASFBs

The following table lists the files for the CIP Over Ethernet application specific function blocks.

NOTE: Every .LDO file on the CD has a corresponding .REM file. The REM files contain all the comments found in the LDO files. If you move an .LDO file to a different location, be sure to move its REM file to the same directory.

GLMC Server ASFBs		
	E_CLSVEX.LDO	Example ladder where the GLMC is the Server.
	E_CLSVEX.ACD	Example ladder where the CLX is the Client.
	E_CIP_CM.LDO	Connection Manager ASFB, manages the TCP/IP connections between the GLMC and one or more CLXs.
	E_CIP_MR.LDO	Message Router ASFB, processes CIP message requests from a CLX.
GLMC Client ASFBs		
	E_CLCLEX.LDO	Example ladder where the GLMC is the client.
	E_CLCLEX.ACD	Example ladder where the CLX is the Server.
	E_CIP_CL.LDO	The E_CIP_CL ASFB, communicates as the client with CLXs.
PiCPro LIB ASFBs		
	E_CIP.LIB	Library containing the ASFBs.
	E_CIP.CHM	Online function block help file.
CIP Support ASFBS		
	E_LISTSV.LDO	This function block processes the CIP ListServices Command and generates the ListServices Reply.
	E_REGSV.LDO	This function block processes the CIP RegisterSession Reply.
	E_NOP_SV.LDO	This function block processes the reply for a NOP command.
	E_SRRDSV.LDO	This function block processes the CIP SendRRData Command and generates the SendRRData Reply.
	E_LISTCL.LDO	This function block generates the CIP ListServices Command and processes the ListServices Reply.

E_FWDOCL.LDO	This function block generates the CIP FowardOpen Request and processes the ForwardOpen Reply
E_REGCL.LDO	This function block generates the CIP RegisterSession Command and processes the RegisterSession Reply.
E_SHNDCL.LDO	This function block generates the SendUnitData Command for a read of zero for each Custom Data Area to the CLX and processes the response. The response will give us the Structure Handle for each User Defined Data Structure in the CLX.
E_SUDCL.LDO	This function block generates the SendUnitData Command for a read or write to he CLX and processes the response.
E_SYMBCL.LDO	This function block reads the symbolic name string to the IOI String of the transport PDU.
E_PDU2DI.LDO	This function block moves the data from the CLX SendUnitData Reply - PDU data area to the DINT array.
E_DI2PDU.LDO	This function block moves the data from a DINT array to the CLX SendUnitData Reply - PDU data area.



Inputs: EN00 (BOOL) - enables execution, should be enabled all the time

SLOT (USINT) - slot number of the G&L Motion Control Ethernet module

CLSR (BOOL) - request to close a TCP/IP session, one-shot

CLSC (USINT) - connection number of session number to close

CM_C (STRUCT) - connection manager object class attributes

CM_I (STRUCT) - connection manager object instance attributes

Outputs: OK (BOOL) - execution complete without error

FAIL (BOOL) - error occurred when initializing the Ethernet module

ERR (INT) - error number

NEWC (BOOL) - a new TCP/IP connection has been established

CONN (USINT) - new connection number, 0-63

<<INSTANCE NAME>>:E_CIP_CM(EN00 := <<BOOL>>, SLOT := <<USINT>>, CLSR := <<BOOL>>, CLSC := <<USINT>>, CM_C := <<STRUCT>>, CM_I := <<MEMORY AREA>>, OK => <<BOOL>>, FAIL => <<BOOL>>, ERR => <<INT>>, NEWC => <<BOOL>>, CONN => <<USINT>>);

This function block is used to mange the TCP/IP connections between the GLMC and one or more CLXs. You will need one instance of the E_CIP_CM ASFB in the GLMC ladder for each Ethernet card in the GLMC used as a Server.

In most cases the GLMC will only have one Ethernet card, thus you would have one E_CIP_CM in your ladder. To improve system bandwidth you can put more Ethernet cards in the GLMC, thus you would have multiple instances of the E_CIP_CM. A single instance of the E_CIP_CM can handle up to 64 connections.

The CLX CIP over Ethernet protocol requires that the Server or the GLMC E_CIP_CM, use IP Port Number 16#AF12. This port should be reserved for this protocol and not used for any other Ethernet connection.

The first time this function is scanned it will open up an IP Socket on port 16#AF12. It will then start to listen for TCP/IP connection requests. Each time a request is made it will accept the connection, up to 64 connections. The Connection Manager instance object will contain the connection parameters for the new connection. These parameters include the IP address of the Client and the IP Socket handle for the new connection. The connection will remain open until the Message Router determines that the connection is broken or a close connection request was received from the Client.

The following is a description of the inputs and outputs.

INPUTS:

EN00 - The enable input should be energized every scan. The Ethernet socket will be opened when the function is first enabled. It will then start to listen for TCP/IP connection requests. If the enable is dropped it will not be able to establish any new connections.

SLOT - Slot number of the G&L Motion Control Ethernet module. The Ethernet module must be in the main rack of a PiC900/90 it can not be in a remote rack. For a Standalone MMC the slot number must be 3 or 4. For the MMC for PC the slot number must be 1.

CLSR - Request to close a TCP/IP connection or session, one-shot.

CLSC - Connection number of session to close when CLSR is on.

CM_C – The CM_C is a structure that contains the Connection Manager Object Class Attributes as defined by the ControlNet Specification. The structure for the CM_C input is defined below.

Name	Type	Description
CM_CLASS	STRUCT	Connection Manager Object Class Attributes
.REVISION	UINT	Revision of this object
.MAXINSTA	UDINT	Largest instance # of Created Object
.NUM_ATTR	UINT	Number of attributes in OPT_ATTR
.OPT_ATTR	UINT (0..19)	List of optional attribute numbers
	END _STRUCT	

CM_I - The CM_I is a structure that contains the Connection Manager Object Instance Attributes as defined by the ControlNet Specification. Note: G&L Motion Control does not support all of these attributes. The structure for the CM_I input is defined in the following table.

Name	Type	Description
CM_INSTA	STRUCT	Connection Manager Instance Attributes
.OPENREQS	UINT	Number of open requests received including null open requests
.OP_F_REJ	UINT	Number of open requests rejected by this node due to format errors
.OP_R_REJ	UINT	Number of open requests rejected by this node.
.OP_O_REJ	UINT	Number of open requests rejected or timed out by downstream nodes
.CL_REQS	UINT	Number of close requests received
CL_F_REJ	UINT	Number of close requests rejected by this node due to format errors
.CLO_O_REJ	UINT	Number of close requests rejected or timed out by downstream nodes
.CONN_TO	UNIT	Number of connections that have been timed out by this node after they were opened
.NUM_CON	UINT	Number of ConnOpen BOOLs used in CON-NOPEN. Should be set to 64.
.CONNOPEN	BOOL (0..63)	List of connection data. 0 = no connection, 1 = connection established
.CONNHNDL	UINT (0..63)	List of IP_ACCEPT connection handles
.CPU_UTIL	UINT	CPU Utilization in 0.1% 0-1000
.MAX_BUFF	UDINT	Max size of buffer bytes
.BUF_SIZE	UDINT	Amount of buffer bytes available
.IPZ	STR [16] (0..63)	List of IP Addresses of the Clients
	END _STRUCT	

OUTPUTS:

OK - The function executed without error.

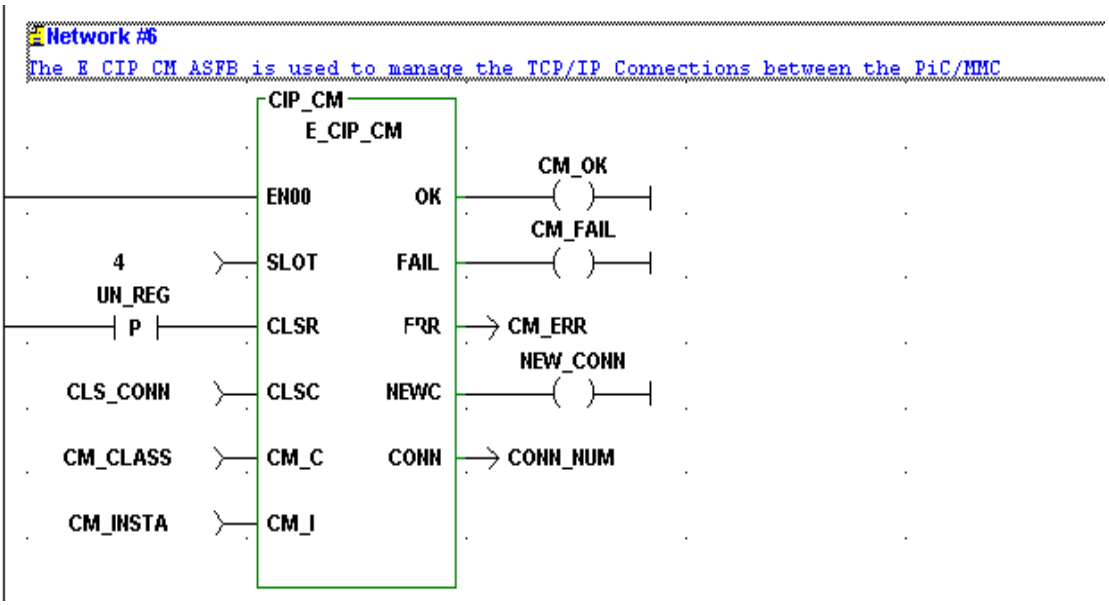
FAIL - An error occurred when initializing the Ethernet module.

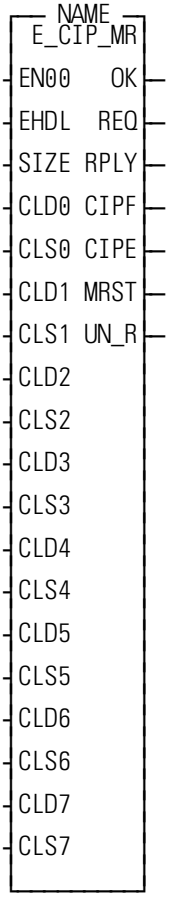
ERR - Error number, see Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide for error description.
Note: ERR = 30000, There are no open connections, all 64 connections are in use.

NEWC - A new TCP/IP Connection has been established.

CONN - Connection number, 0-63, for the new connection, it is used as index into CM_INSTA.CONNHNDL(0..63) which is the socket handle for the connection and CM_INSTA.IPZ(0..63) which is the IP address of the Client for the new connection.

An Example Ladder for the E_CIP_CM ASFB follows:





Inputs: EN00 (BOOL) - enables the function, energizes after TCP/IP connection has been made

EHDL (UINT) - tcp/ip socket handle from E_CIP_CM

SIZE (STRUCT (0..7) - Defines the size of the custom data base areas CLD (0..7) and CLS (0..7)

CLD (0..7)) (STRUCT) - destination structure, CIP Generic Source structure data will be written to this structure

CLS (0..7)) (STRUCT) - source structure, this structure will be written to the CIP Generic destination structure

Outputs: OK (BOOL) - function is enabled

REQ (BOOL) - a CIP request was received

RPLY (BOOL) - a CIP reply has been sent

CIPF (BOOL) - a CIP error or fault code was returned in the reply

CIPE (UDINT) - error number returned the CIP header status

MRST (UINT) - MR request status code returned

UN_R (BOOL) - un-register the TCP/IP connection

<<INSTANCE NAME>>:E_CIP_MR(EN00 := <<BOOL>>, EHDL := <<BOOL>>, SIZE := <<MEMORY AREA>>, CLD := <<MEMORY AREA>>, CLS := <<MEMORY AREA>>, OK => <<BOOL>>, REQ => <<BOOL>>, RPLY => <<BOOL>>, CIPF => <<BOOL>>, CIPE => <<UDINT>>, MRST => <<UINT>>, UN_R => <<BOOL>>);

The E_CIP_MR function block is the Message Router for processing CIP over Ethernet requests from the CLX. This function will process messages for a single TCP/IP connection. The TCP/IP connection will be establish via the E_CIP_CM function. The CLX will issue the read/write data requests via the CIP Generic MSG function. See CHAPTER 4, CIP Generic MSG Function.

The following is a description of the inputs and outputs for the E_CIP_MR function.

INPUTS

EN00 – The enable input should be energized after a TCP/IP connection has been made. When a new connection is made the E_CIP_CM output NEWC will be on for one scan and the CONN output will hold the connection number. The CM_INSTA.IPZ(CONN) which is the IP address of the Client for the new connection should be compared to the IP address for the corresponding node. If they are equal then the enable should energized to enable the messages to be processed.

EHDL - TCP/IP socket handle from C_CIP_CM CM_INSTA.CONNHNDL(x) for the connection, where x is equal to value of CONN when the connection was established.

SIZE - Defines the size in bytes of the custom data areas CLDx & CLSx, where x = 0-7. The SIZE_OF function can be used to determine the size of the structures. The structure for the SIZE input is defined in the following table.

.

SIZE	STRUCT (0..7)	
.SOURCE	UINT	Structure in bytes of the CLS structure. This is the number of bytes written to the CLX in the reply message. The size of the Destination structure in the CLX must be equal to or greater than this size. Note: If the number of bytes of data returned to the CLX is greater than the size of the destination structure, it does not return an error. It is unknown if the data goes to memory or if it is ignored.
.DEST	UINT	Size in bytes of the CLD structure. The number of bytes written from the CLX Source structure to the GLMC Destination structure must not be greater than this size.
	END_STRUC	

CLD 0-7 – GLMC Destination Structure, the CIP Generic MSG Source structure data will be written to this structure.

CLS 0-7 – GLMC Source Structure, this structure will be written to the CIP Generic MSG Destination structure.

The following is a summary of how the Custom Data Areas or structures need to be defined in the GLMC and CLX.

1. The Custom Data Areas CLD0-7 and CLS0-7 are data structures or arrays that are used to pass data between the GLMC and the CLX.
2. The data types supported are INT, DINT, REAL and BOOL.
3. The CLX requires that the data in a User-Defined structure be in groups of 32 bits, otherwise there will be a gap or unused word in memory.
4. To make the structures in the GLMC and CLX match up byte for byte the INT data needs to be such that you have an even number of INT's, i.e. 2,4,6, … elements.
5. The DINT and REAL data types consume 4 bytes of memory so you can have any number of them.
6. The BOOLs need to be packed into Double Words or in groups of 32 bits.
7. The GLMC ladder will need to pack or unpack the BOOLs into/from DWORD's using the G_BOOLDW and G_DW2BOO ASFBs. This is the case because the GLMC uses 1 byte of memory for each BOOL. Internally the GLMC uses bits in this byte for the current, previous, positive transition and negative transition states. The CLX stores BOOLs as bits in 32 bit increments and uses functions to determine transitional states.
8. You can define the BOOLs in the CLX as individual BOOLs in groups of 32, as BOOL[32] arrays, or as DINT's and reference each bit.
9. The maximum size of data that can pass each way in one message is 432 bytes. You can read 432 bytes and write 432 bytes in the same message. If you have more that 432 bytes of data to read or write you will have to break the data up into multiple Custom Data Area's and use multiple messages to pass the data.

OUTPUTS

OK – Indicates that the function is enabled.

REQ – Indicates that a CIP request was received, one shot.

RPLY - Indicates a CIP reply has been sent, one shot.

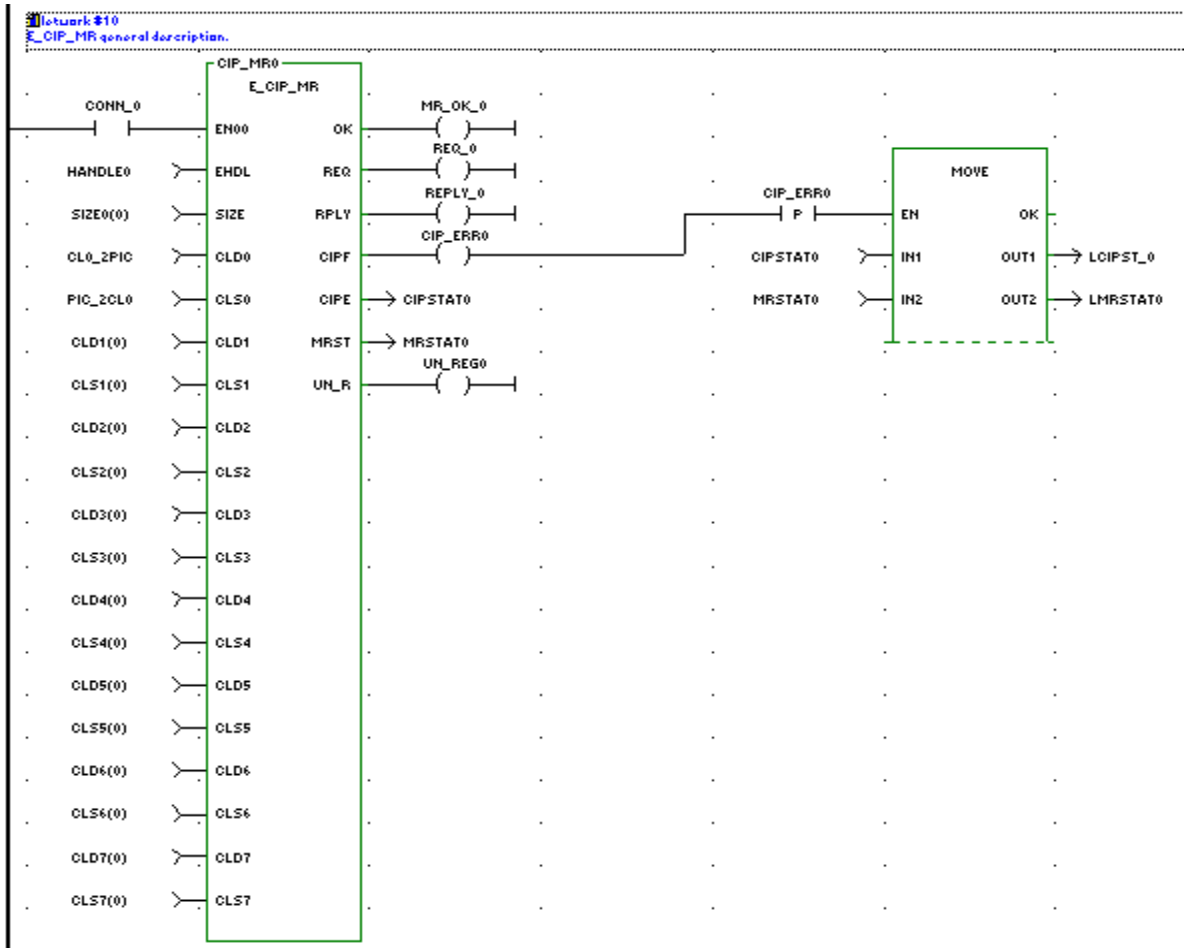
CIPF - Indicates that a CIP error or fault code was returned in the reply or a connection error has occurred.

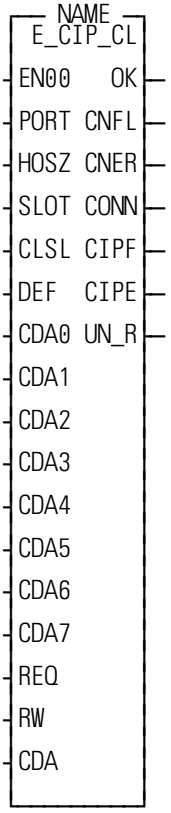
CIPE - Error number returned in the CIP header status. See Chapter 4.

MRST – Message Router Request status code. Refer to See Chapter 4.

UN_R – The Message Router has determined that the TCP/IP connection has been broken or an Un-Register request has been received. The TCP/IP connection has been closed and the connection needs to be closed by the E_CIP_CM. If the Message Router goes 30 seconds without receiving a message it considers the connection broken and closes the session.

An example ladder for the E_CIP_MR ASFB follows:





Inputs: EN00 (BOOL) - enables execution, should be tied to the rail

PORT (UINT) - protocol part number; AF12 is reserved for the CIP Server and should not be used for the client

HOSZ (STRING) - name of the address of target host or target CLX zero terminated

SLOT (USINT) - slot number for the PiC's Ethernet card

CLSL (USINT) - slot number for the ControlLogix CPU

DEF (STRUCT) (0..7) - Defines the size and name of the custom data areas

CDA (0..7) (STRUCT) - This structure will contain the data that will be passed between the PiC and ControlLogix; max size is 432 bytes

REQ (BOOL) - one-shot to request a read or write command

RW (BOOL) - Off = Read command; On = Write command

CDA (USINT) - selects the custom data area where the data will be read from or written to

Outputs: OK (BOOL) - function is enabled

CNFL (BOOL) - failed to make a TCP/IP connection

CNER (INT) - error code failed TCP/IP connection

CONN (BOOL) - a TCP/IP connection and a CIP session have been established with the target CLX CPU

RWDN (BOOL) - the requested read/write command is done

CIPF (BOOL) - an error occurred while trying to do a read or write

CIPE (DINT) - error code for read/write error

UN_R (BOOL) - the CLX has sent an UnRegisterSession command and the GLMC has closed the TCP/IP connection or the command has timed out

```
<<INSTANCE NAME>>:E_CIP_CL(EN00 := <<BOOL>>, PORT :=
  <<UINT>>, HOSZ := <<STRING>>, SLOT := <<USINT>>, CLSL :=
  <<USINT>>, DEF := <<MEMORY AREA>>, CDA := <<MEMORY
  AREA>>, REQ := <<BOOL>>, RW := <<BOOL>>, CDA := <<USINT>>, OK
  => <<BOOL>>, CNFL => <<BOOL>>, CNER => <<INT>>, CONN =>
  <<BOOL>>, RWDN => <<BOOL>>, CIPF => <<BOOL>>, CIPE =>
  <<DINT>>, UN_R => <<BOOL>>);
```

The E_CIP_CL function bkock is used to communicate as the client with a CLX. Separate functions are required to communicate with each CLX.

The following is a description of the inputs and outputs for the E_CIP_CL function.

INPUTS

EN00 – When the function is first enabled, it will try once to make a TCP/IP connection, if this fails the CNER will be set. After a TCP/IP connection has been made this function will try to open a CIP Session. If a CIP Session is not opened within 30 seconds the attempt fails and the CNER will be set. After the CIP Session is established, the requested commands will be sent to the CLX. If there is not a reply in 2 seconds the command will be re-sent. Note: the Ethernet card continues to re-send the command. If there is no reply within 30 seconds the connection will close and the response will be CNER. To re-establish a connection you must drop the ENxx or enable input for at least one scan and then enable it again.

PORT – IP Protocol Port Number. 16#AF12 is reserved for the CIP Server and should not be used for the client.

HOSZ – Name or IP Address of Target CLX, zero terminated.

CLSL - Slot number of the CLX CPU.

DEF – Defines the name and the size in bytes of the custom data areas CLDx & CLSx, where x = 0-7. The SIZE_OF function can be used to determine the size of the structures. The structure for the DEF input is defined in the following table:

DEF	STRUCT (0..7)	
.NAME	STRING[25]	Name of the Controller Tag in the CLX of the data being accessed
.SIZE	UINT	Size bytes of the Custom Data Structure, CDAx structure in the GLMC. This determines the number of bytes written to or read form CLX.
	END_STRUCT	

CDA0-7 – GLMC Custom Data Area. These data structures will be either written to or read from the CLX. The following is summary of how the Custom Data Areas or structures need to be defined in the GLMC and CLX.

1. The Custom Data Areas CDA0-7 are data structures or arrays that are used to pass data between the GLMC and the CLX.
2. The data types supported are INT, DINT, REAL and BOOL.
3. The CLX requires that the data in a User-Defined structure be in groups of 32 bits, otherwise there will be a gap or unused word in memory.
4. To make the structures in the GLMC and CLX match up byte for byte the INT data needs to be such that you have an even number of INT's, i.e. 2,4,6, … elements.
5. The DINT and REAL data types consume 4 bytes of memory so you can have any number of them.
6. The BOOLs need to be packed into Double Words or in groups of 32 bits.
7. The GLMC ladder will need to pack or unpack the BOOLs into/from DWORD's using the G_BOOLDW and G_DW2BOO ASFBs. The reason for this is that the GLMC uses 1 byte of memory for each BOOL. Internally the GLMC uses bits in this byte for the current, previous, positive transition, and negative transition states. The CLX stores BOOLs as bits in 32 bit increments and uses functions to determine transitional states.
8. You can define the BOOLs in the CLX as individual BOOLs in groups of 32, as BOOL[32] arrays, or as DINT's and reference each bit.
9. The maximum size of data that you can pass each way in one message is 432 bytes. You can read 432 bytes or write 432 bytes in one message. You can not read and write data with the same message. If you have more that 432 bytes of data to read or write you will have break the data up into multiple Custom Data Area's and use multiple messages to pass the data.

REQ - Request to read/write data from/to the CLX, one-shot. Once a request has been made, you can not make another request until the current request is done (RWDN). If you make another request while one is being done, the second request will be ignored.

RW – Off = Read data from the CLX Controller Tag defined in DEF(x).NAME, the data will be stored in GLMC CDAx. On = Write the data from GLMC CDAx, to CLX Controller Tag defined in DEF(x).NAME. Where x is equal to CDA input below.

CDA - Selects the Custom Data Area where the data will be stored during a read, or written from during a write.

Outputs

OK – Indicates that the function is enabled.

CNFL - Failed to make a TCP/IP Connection.

CNER - Error number, see Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide for error description.

CONN - A TCP/IP Connection and a CIP Session has been established with the target CLX CPU. Read and write requests can now be made to the CLX. See Chapter 4.

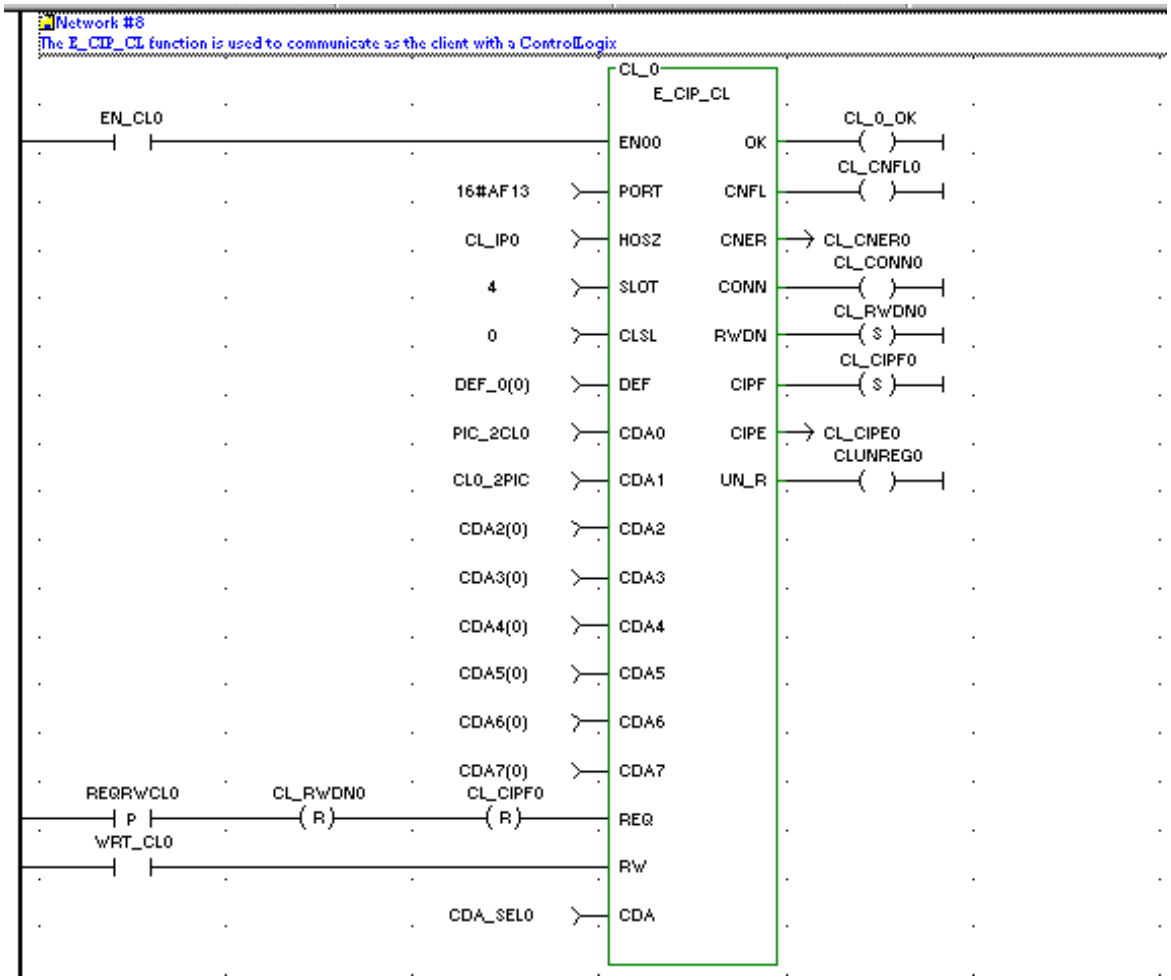
RWDN - The requested read/write command is done. After making a request, can not make another request until the RWDN goes on.

CIPF - An error occurred while trying to do a read or write.

CIPE - Error code for read/write error. See Chapter 4.

UN_R - The CLX has sent an UnRegisterSession command and the GLMC has closed the TCP/IP connection or the connection has timed out.

An example ladder for the E_CIP_CL ASFB follows:

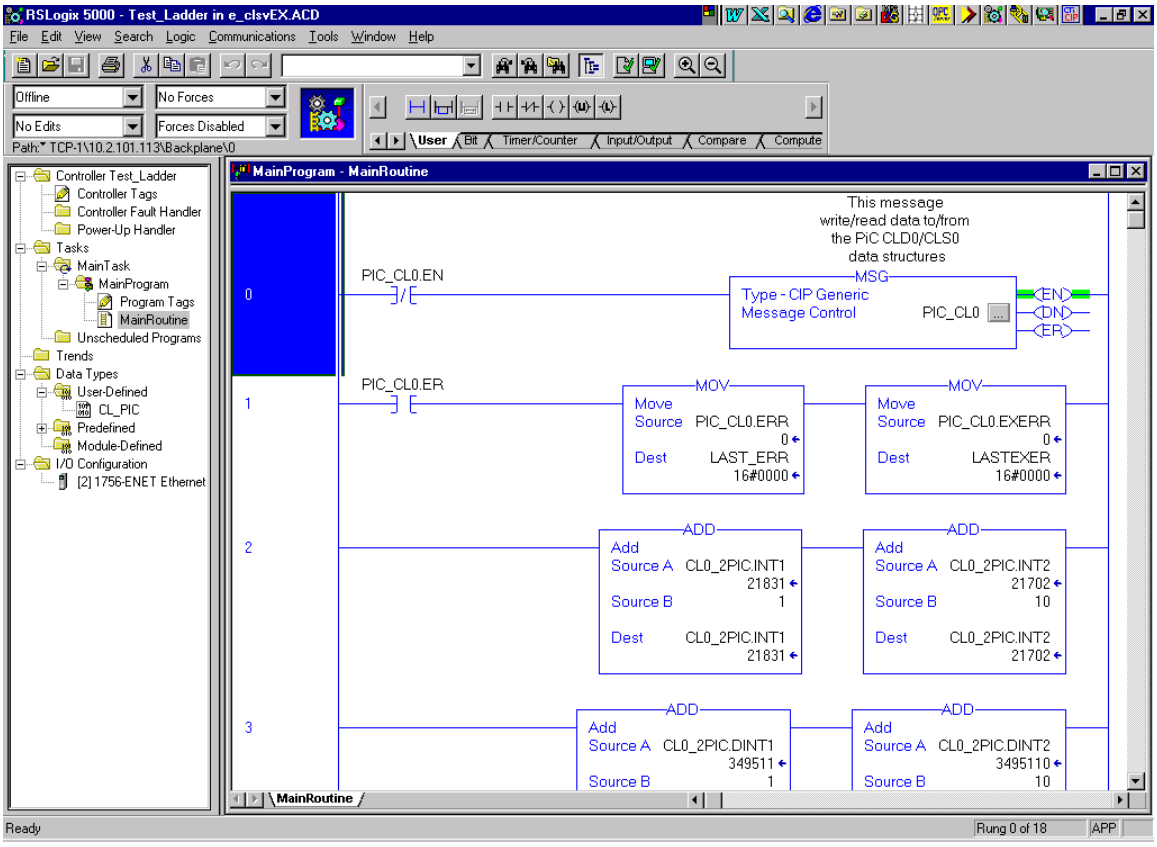


CHAPTER 4 CIP Generic MSG Function

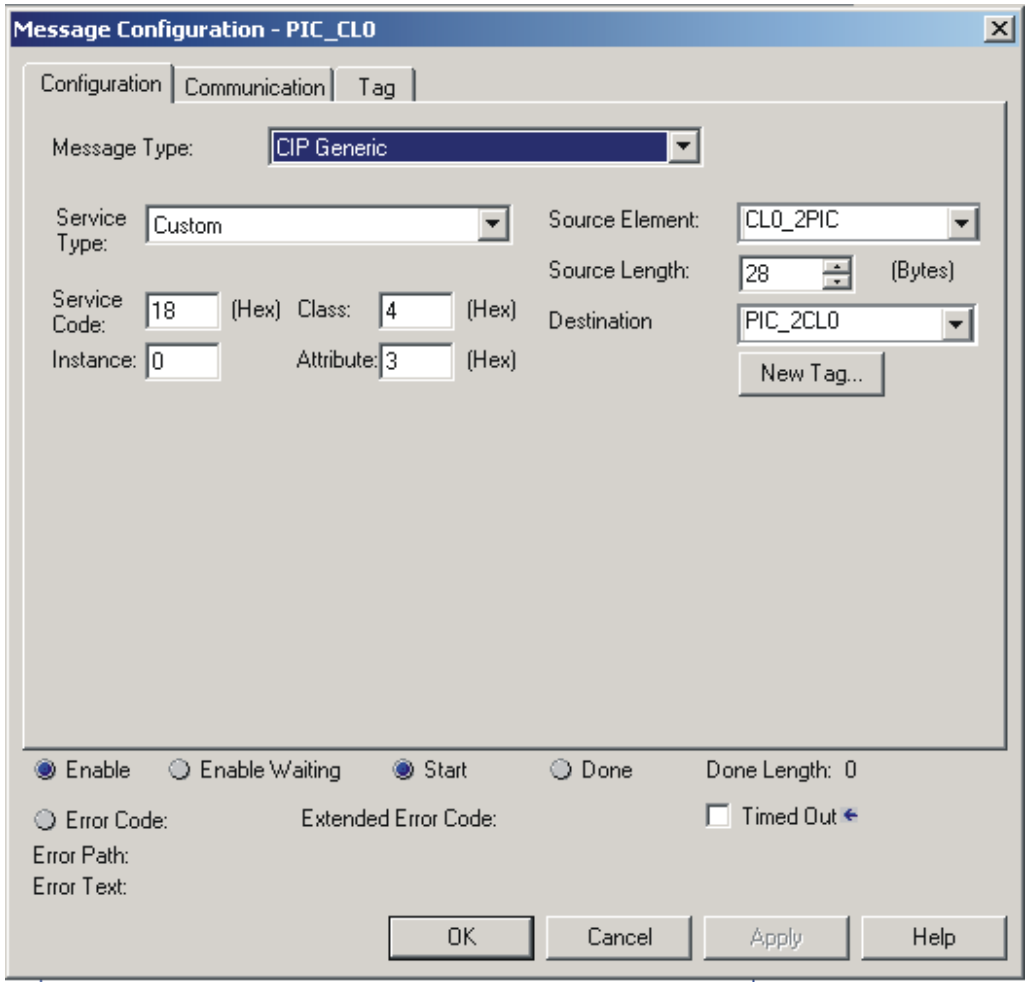
This function is used in the CLX ladder to pass data between the GLMC and the CLX. Refer to the RSLogix5000 online help for a detailed description of the MSG function.

To setup the MSG function, do the following:

1. Open the Main Routine window in the RSLogix5000 software so the following window appears.



2. Access the Message Configuration window by double clicking on the MSG function. Fill in the data fields for the Configuration tab as follows.



The following is a description of the fields in the Configuration tab:

Message Type – CIP Generic.

Service Type – Custom.

Service Code – 0x18, Get Member command.

Object Type – 0x4, Assembly Object.

Instance – Instance number, 0-7, the instance corresponds to CLDx and CLSx, where x = 0-7. This selects the custom data area in the GLMC where the data is going to read from or written to.

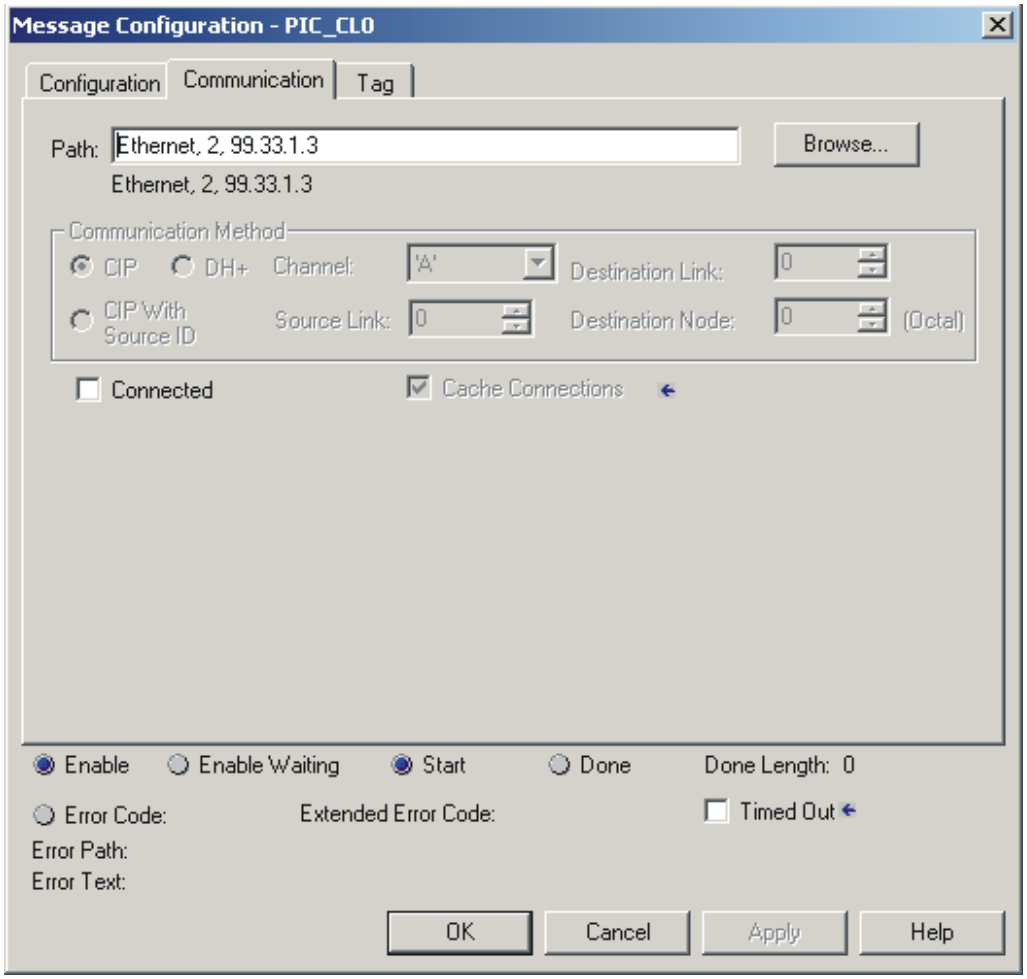
Attribute – 0x3, All of the member data is packed into one array or data area.

Source Element – CLX Controller Tag from where the data is to be written from. This data will be written to the GLMC structure CLDx.

Source Length – Number of bytes to be written from the CLX Source to the GLMC Destination CLDx structure. The size in bytes of the CLX Controller Tag can be found under Properties for the Data Types - User-Defined - UserType.

Destination – CLX Controller Tag where the data read from the GLMC Source CLSx Structure will be written to in the CLX. The number of bytes read is the value of the SIZE(x).SOURCE input to the GLMC E_CIP_MR function.

3. Click on the Communication tab. Fill in the data fields for the Communication tab as follows.



The following is a description of the fields in the Communication tab:

Path – The connection path to which the message is going to be sent (target device).

Communication Method – CIP, click on this button to route the message to the ControlLogix backplane/Ethernet Network.

Cache Connections – Check this box to enable cache connections. To optimize execution time, you should keep this connection open when the MSG instruction executes repeatedly. If you don't check the box, the connection will be released after the message has been executed. This should only be done if the MSG executes infrequently.

NOTES

CHAPTER 5 Status and Error Codes for CIP Over Ethernet ASFBs

Status and Error Codes for E_CIP_MR ASFBs

CIPE Status Codes	Description
0x00	Sucess
0x01	The sender issued an invalid or unsupported encapsulation command.
0x02	Insufficient memory resources in the receiver to handle the command. The size of the message exceeds 512 bytes or 432 bytes of data.
0x03	Poorly formed or incorrect data in the data portion of the encapsulation message. Check the parameters in the CIP Generic MSG. The Service code must be 0x18, the object code type must be 4, the Object ID must be 0-7 corresponding to CLDx/CLSx and the Object Attribute must be 3.
0x64	An originator used an invalid session handle when sending an encapsulation message to the target.
0x65	The target received a message of invalid length.
0x69	Unsupported protocol revision.
0x04-0x63	Allocated for compatibility with existing protocols.
0x66-0x68	Allocated for compatibility with existing protocols.
0x6A-0x7FFF	Allocated for compatibility with existing protocols.
0x8000-0xFFFF	Reserved for future expansion.

MRST Status Codes	Description
0x15	The CLX attempted to write more data than the size of the CLDx structure. Check the Num. of Elements in the CLX MSG function; it can not be greater than SIZE(x).DEST in the GLMC.
	If CIPE = 30000-30002, see Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide for error description.

Connection Errors	Description
30000	Timeout, no message requests have been received in the last 30 seconds. The TCP/IP session has been closed. Possible causes include: • The Ethernet cable has been disconnected or broken. • Client has been powered down or has stopped running the ladder. • The network is down.
30001	The IPREAD function has failed. MRST will contain the error number. See the Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide for error description.
30002	The IPWRITE function has failed. MRST will contain the error number. See the Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide for error description.
30003	Loss of communication between the MMC for PC and the Windows OS Ethernet Stack.

CNER Error Codes	Description
0-1004	See Ethernet-TCP/IP Errors under IPWRITE function in the Function/Function Block Reference Guide
30000	Timeout, no message requests have been received in the last 30 seconds. The TCP/IP session has been closed. Possible causes include: • The Ethernet cable has been disconnected or broken. • Server has been powered down or has stopped running the ladder. • The network is down.
30001	The CIP ListServicesReply was invalid, consult factory
30002	The CIP RegisterSession Reply was invalid, consult factory
30003	The CIP ForwardOpen Reply was invalid, consult factory
30004	The CIP SendUnitDataReply was invalid when the attempt was made to get the Structure Handle for each User Defined Data Structure in the CLX. Check to make sure that the Controller Tag exists in the CLX. If the Controller Tag exists in the CLX consult the factory.
30005	The CIP SendUnitDataReply was invalid when we tried to read or write data. Consult the factory.
30006	Loss of communication between the MMC for PC and the Windows OS Ethernet Stack.

CIPE Status Codes	Description
0x00	Success
0x01	The sender issued an invalid or unsupported encapsulation command.
0x02	Insufficient memory resources in the receiver to handle the command. The size of the message exceeds 512 bytes or 432 bytes of data.
0x03	Poorly formed or incorrect data in the data portion of the encapsulation message. Check the parameters in the CIP Generic MSG. The Service code must be 0x18, the object code type must be 4, the Object ID must be 0-7 corresponding to CLDx/CLSx and the Object Attribute must be 3.
0x64	An originator used an invalid session handle when sending an encapsulation message to the target.
0x65	The target received a message of invalid length.
0x69	Unsupported protocol revision.
0x04-0x63	Allocated for compatibility with existing protocols.
0x66-0x68	Allocated for compatibility with existing protocols.
0x6A-0x7FFF	Allocated for compatibility with existing protocols.
0x8000-0xFFFF	Reserved for future expansion.
0x10000	DEF (CDA). SIZE is greater than 432 bytes. A message can not have more than 432 bytes of data. The message will have to be broken up into multiple messages.

Index

NOTES