Isaac EtherNet/IP™ Guide

Abstract

This document is to be used by control engineers and other technical personal responsible for communications between programmable logic controllers (PLC's) and the Isaac multi-tester by Zaxis Inc. The document assumes the reader is well acquainted with EtherNet/IP™ as defined the ODVA association. The reader should also have a good understanding of TCP/IP and UDP/IP protocols.

Last updated
12-Oct-21

Table of Contents

Isaac EtherNet/IP™ Guide 0
Introduction 2
EtherNet/IP™ for the Isaac tester overview 3
Restoring Factory Default Network Configuration 4
Isaac Tester Configuration for Ethernet/IP™ 5
PLC Configuration for Ethernet/IP™ 5
Configuration with the EDS file 5
Configuration using a Generic Ethernet Module 9
Isaac Configuration, Input and Output Tag Groups 12
Configuration Tags 12
cosType – Change-Of-State flags 13
PLC Output Tags 14
PLC Input Tags 15
COS Change-of-State bits 16
Valve Mask 17
Isaac Firmware/Program configuration tags 18
Firmware Parameters 18
Reading and Writing of Firmware Parameters 20
Program Parameters 23
Reading and Writing of Program Parameters 25
Reuse of the MSG tag 27
TCP/IP Network Configuration 28
Trouble Shooting 31
Error Codes Generated by the Isaac 32
General Status Codes 32
Encapsulation Status Codes 33
Definitions 34
Test Type Values 34
Channel/Port Status 34
Test Sequence 35

Introduction

Zaxis's Isaac multi-tester is a high-performance air leak tester with unparalleled test stability and repeatability. Depending on the model, the following tests can be performed: pressure decay, vacuum decay, mass flow, pressure cracking, burst pressure, valve cracking pressure and more. From one channel iKits to the four channel Isaac HD models EtherNet I/P is an optional communications method.

Communications with the Isaac can be accomplished in different ways, RS232/485, TCP/IP and EtherNet/IP™. This document will describe how to setup EtherNet/IP™ using an Allen-Bradley 1769 CompactLogix Controller. Other types of PLCs can be used if compliant to the EtherNet/IP™ specifications as defined the ODVA association.

EtherNet/IP™ for the Isaac tester overview

The "IP" in "EtherNet/IP" refers to "Industrial Protocol" and should not be confused with Ethernet or the TCP/IP and UDP/IP protocols. Although EtherNet/IP™ uses these technologies and protocols to accomplish the desired communications between the Isaac and PLC's.

The Isaac software consists of several different modules, as for EtherNet/IP™ the software follows the recommended implementation as defined in the ODVA specifications. See *The CIP Networks LIBRARY Volume 1 and 2.* The Isaac EtherNet/IP™ software levels are: Application, Session, Transport, Network, Data Link, Physical as show below in Table 1.

Application	Isaac EtherNet/IP
Session	Explicit/Implicit Messaging (CIP)
Transport	TCP/UDP
Network	Internet Protocol (IP)
Data Link	ethernet
Physical	Peer-to-peer, multicast, unicast

Table 1 - Isaac EtherNet/IP™ Adaptation of CIP

The Application and Session layers (as shown in top two rows) are customized for the Isaac, while the lower layers (as shown in bottom four rows) follow the IEEE 802.3 standards with little or no customization. Thus, allowing existing ethernet switches and/or networks to communicate with the Isaac using standard TCP/IP and UDP/IP protocols.

The Isaac supports both DHCP and static ethernet configurations. Factory defaults use the following static network configuration:

I/P Address	192.168.2.130
Network Mask	255.255.255.0
Gateway	192.168.2.1

Table 2 - Factory Default Network Configuration

These defaults can be changed using the Tsi (Touch screen interface) or the PLC MSG command as described below.

Factory defaults can be restored by holding the red and green push buttons in for approximately eight seconds, until both buttons blink once while the unit is powered on. When the lights stop blinking the buttons can be released.

It is recommended that a static configuration be used to ensure devices that communicate to the Isaac can rely on the same I/P addresses for each Isaac tester. However, it is possible to use DHCP if the DHCP server and/or Name server can resolve an Isaac tester hostname to a I/P address.

Because EtherNet/IP™ relays on TCP/UDP it is necessary that the Isaac tester and any device wishing to communicate with the Isaac tester over ethernet have the correct network configuration. The network *ping* command can be used to diagnose network problems etc.

EtherNet/IP™ uses two forms of messaging:

- **Unconnected messaging** is used to: 1) establish connections, 2) low-priority messages, and 3) infrequent messages. Unconnected messages are handled by the Unconnected Message Manager, e.g., the UCMM software module.
- **Connected messaging** allocates resources that are dedicated to a particular purpose, such as real-time I/O data transfers. Connection resources are reserved and configured by the UCMM (Unconnected Message Manager) and then allocated to the desired resource.

Connections are created by a Connection Originator (or Originator for short) and the device that responds to the connection request, called the Connection Target (or just Target).

EtherNet/IP™ specifics two types of message connections:

- **Explicit message connections** are point-to-point relationships. These would be considered general purpose connections and are used to access general information about the Isaac tester. E.g., revision number, serial number, etc. Explicit messages almost always use TCP/IP connections to communicate.
- **Implicit (I/O data) connections** are created to communicate application specific I/O data at regular intervals. Implicit messages use UDP/IP resources to communicate.

The Isaac tester uses explicit messages to create a connection with other devices (e.g., a PLC's etc.) and when requested, creates an implicit connection to periodically return the status of the Isaac tester. E.g., busy, sequence/state, current program etc.

For further information and documentation on message types etc. see OVDA publication PUB00138_EtherNetIP_Ethernet_Technology.pdf.

Restoring Factory Default Network Configuration

The factory default network configuration, as described above, can be restored using the following steps.

1. Power off the tester
2. Press the green and red button at the same time while power is applied.
3. Continue to hold down both buttons (for about 8 seconds).
4. While holding the buttons you may see the green button blink, but don't release the buttons yet.
5. When both buttons blink, release the buttons.
6. The I/P address should now be 192.168.2.130, the network mask 255.255.255.0 and the default gateway 192.168.2.1

Isaac Tester Configuration for Ethernet/IP™

Other than configuring the correct ethernet (TCP/IP) parameters on the Isaac, no other configuration is required for Ethernet/IP™ communications. Taking note of the Isaac I/P address, netmask and gateway (see Table 2 - Factory Default Network Configuration) are required to configure the devices that communicate with the Isaac tester.

PLC Configuration for Ethernet/IP™

There are two ways to configure a PLC to communicate with an Isaac tester.

1. Use an EDS (Electronic Data Sheet) file which can be downloaded from the Zaxis web site https://www.zaxisinc.com/downloads/. Then follow the steps described in the Configuration with the EDS file section below.
2. Add a generic EtherNet I/P module to your PLC project. Then follow the steps described in Configuration using a Generic Ethernet Module below.

Configuration with the EDS file

Once the EDS file has been downloaded from the Zaxis, Inc. web site the EDS file should be registered using the Rockwell Automation's EDS Wizard. Several resources can be found on the Internet showing the proper procedure to register an EDS file.

Once the Isaac tester EDS file has been registered the configuration can be done as follows:

1. Start Logix Designer and Ensure the Communications are offline.
2. Create a new Isaac Ethernet module by right clicking on the **Ethernet** menu item displayed on the **Controller Organizer** window (Figure 1) and select **New Module…**

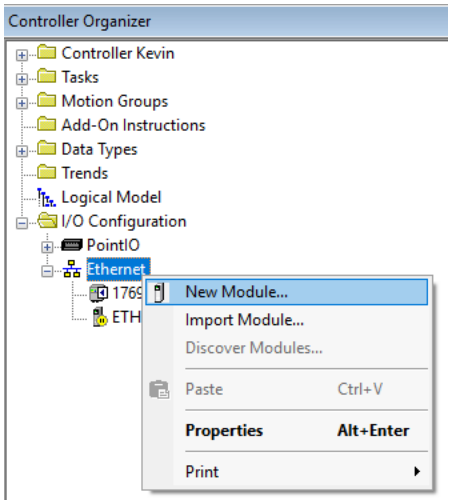


Figure 1 – Create New Ethernet-Module

3. The **Select Module Type** window should be as displayed as seen in Figure 2.
4. Search for Zaxis using the Catalog search field to quickly find the Leak Tester module.

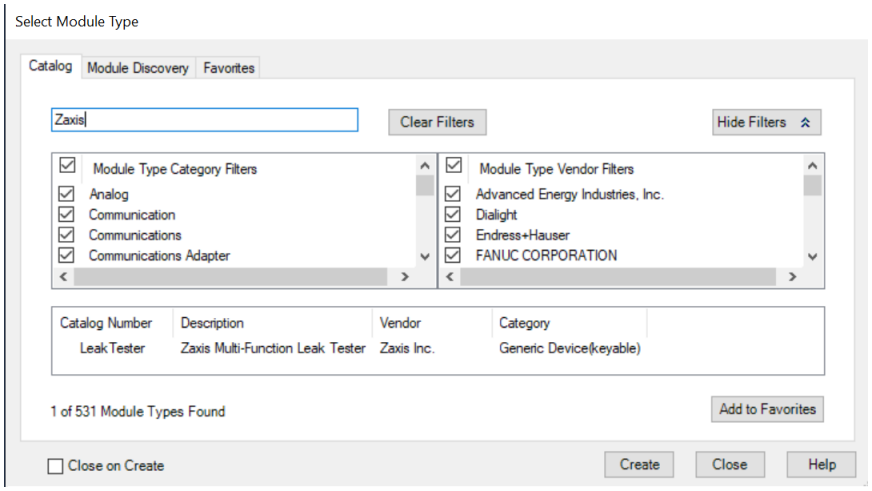


Figure 2 - Create Zaxis Leak Tester

5. Click the Create button to create the Leak Tester module, which should display the following window, see Figure 3.

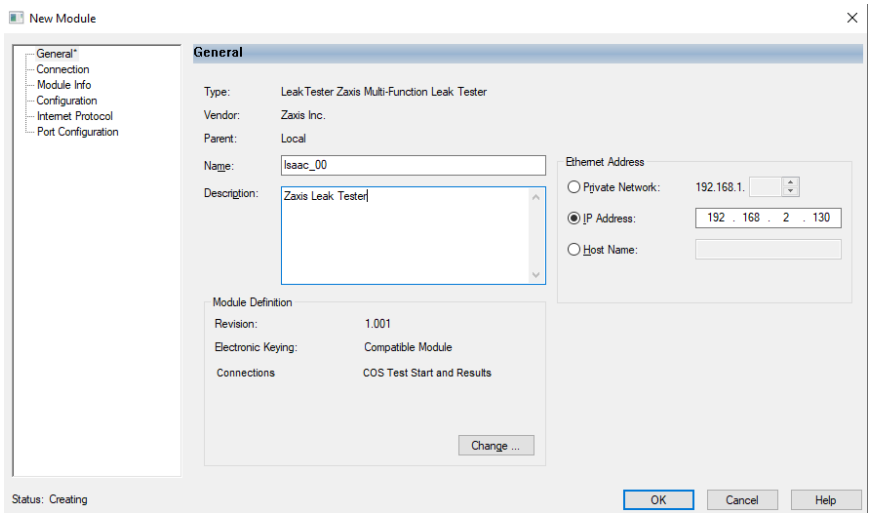


Figure 3 - New Zaxis Module

6. Enter a name for the Isaac tester and enter the assigned I/P address for the tester. A unique I/P address is often assigned to the Isaac test when ordered, if this is the case enter the assigned I/P address.
7. Clicking on the **Connection** menu item will allow changing the type of connection between Change of State and Cyclic. Unless suggested by a Zaxis support personal, or if using a non-Allen-Bradly PLC, it is recommended that the Change of State connection be used.

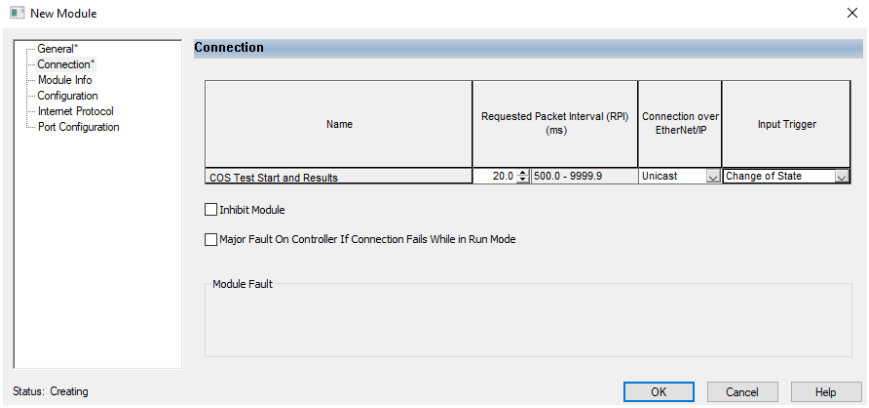


Figure 4 - Connection Parameters

8. The Connection window allows the Requested Packet Interval (RPI) value to be changed, the default value of 20.0 milliseconds is the recommended value. Unless the PLC will be changing program numbers, the recommend RPI would then be 300 milliseconds allowing the Isaac tester enough time to read/write flash values as the programs are changed.
9. The Configuration window allows the modification of the configuration parameters. The Configuration parameters are sent to the Isaac whenever the PLC creates an I/O type connection. The definition of each configuration parameters is provided below (see

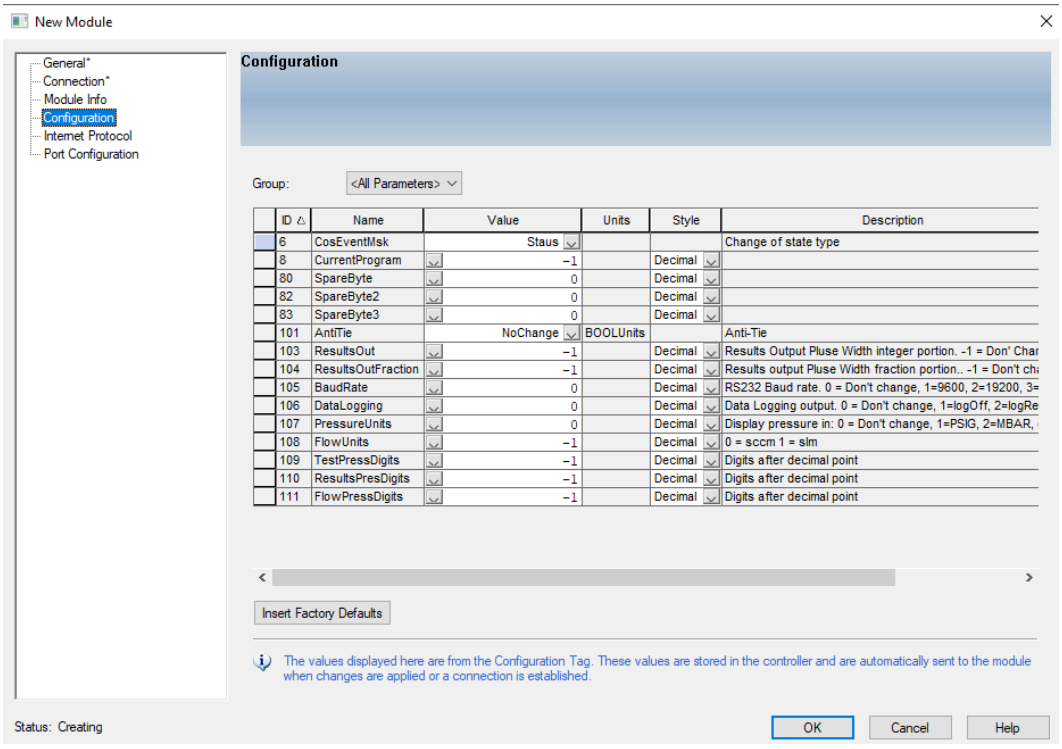


Figure 5 - Configuration Parameters

Configuration Tags). If an invalid value is provided (e.g., -1 for CurrentProgram) the current value of the tag contained in the Isaac will be used and the invalid value will be ignored.

10. Clicking the **OK** button will finish the creation of the module, causing the newly created item to appear, as shown below in Figure 6 below.

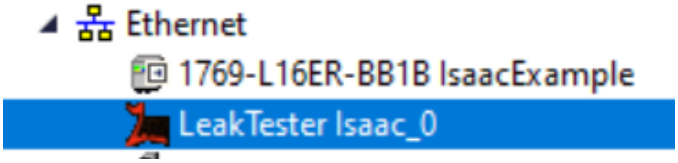


Figure 6 - Ethernet module

Other module configuration items can be completed by double clicking on the newly created module and setting the desired parameters etc.

Configuration using a Generic Ethernet Module

Configuring the PLC for communications with the Isaac tester via EtherNet/IP™ consists of creating a new *Generic Ethernet Module*. Use the following steps.

1. Start Logix Designer and Ensure the Communications are offline.
2. Create a new ETHERNET-MODLE by right clicking on the **Ethernet** menu item displayed on the **Controller Organizer** window (Figure 7) and select **New Module…** which should display the **Select Module Type** window (Figure 8).

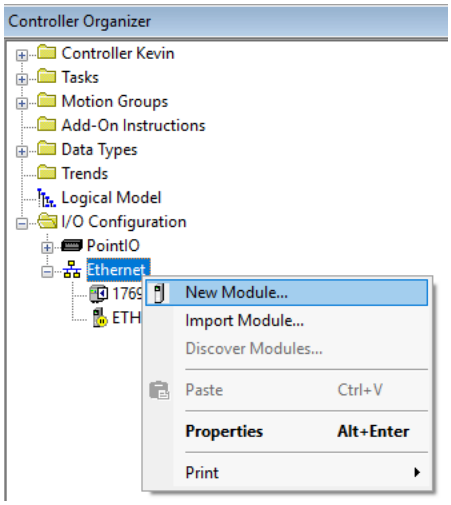


Figure 7 - Create New Ethernet-Module

3. In the **Select Module Type** window use the filter text box to find a **Generic Ethernet Module** as show in Figure 8. Then click the **Create** button.

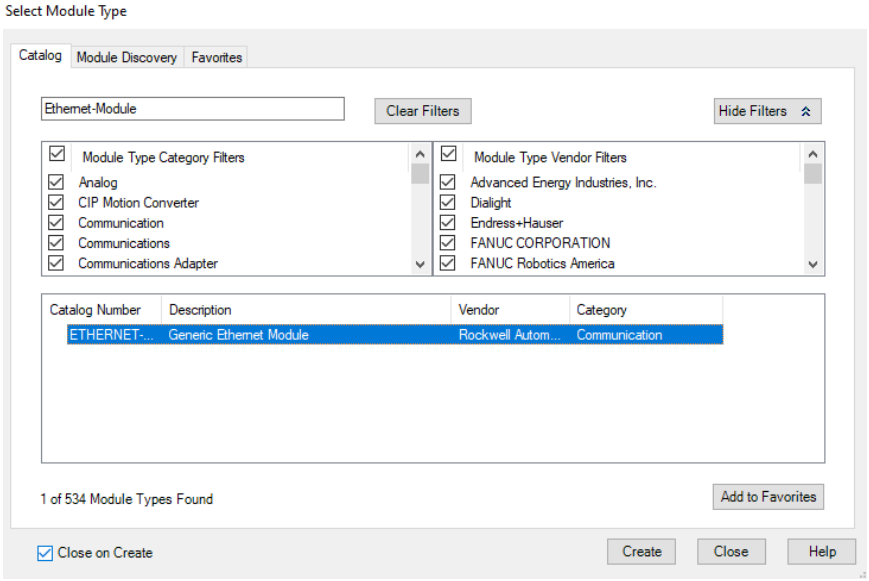


Figure 8 - Generic Ethernet Module

4. After clicking on the **Create** button the **New Module** information can be entered.
   a. Enter a valid name.
   b. Enter a description, if desired.
   c. Ensure the **Comm Format** is set to **Data – DINT.**
   d. Enter the **IP address** of the Isaac tester.
   e. Enter **1** for the **Input** Assembly Instance with a Size of **17**.
   f. Enter **2** for the **Output** Assembly Instance with a Size of **4**.
   g. Enter **4** for the **Configuration** with a Size of **16** (or **0** if Configuration data is not desired).

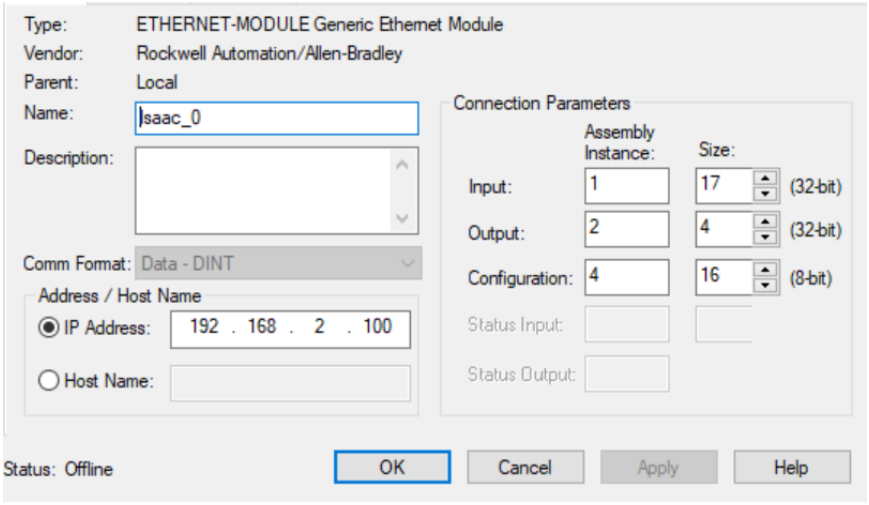


Figure 9 - Generic Ethernet Module

The **Configuration** size is optional, it can be 0 (zero) or 16. When zero, no initial configuration will be done when communications to the Isaac tester is established. When size is set to 16, a block of 16 bytes will be sent to the tester, used to do initial configuration. See Configuration Tags below.

5. Clicking the **OK** button creates the new module as well as the necessary controller tags used to interface with the Isaac tester and displays the **Connection** tab of the newly created module as seen in Figure 10 below. Set the **Requested Packet Interval (RPI)** to 20 e.g., 20 milliseconds.

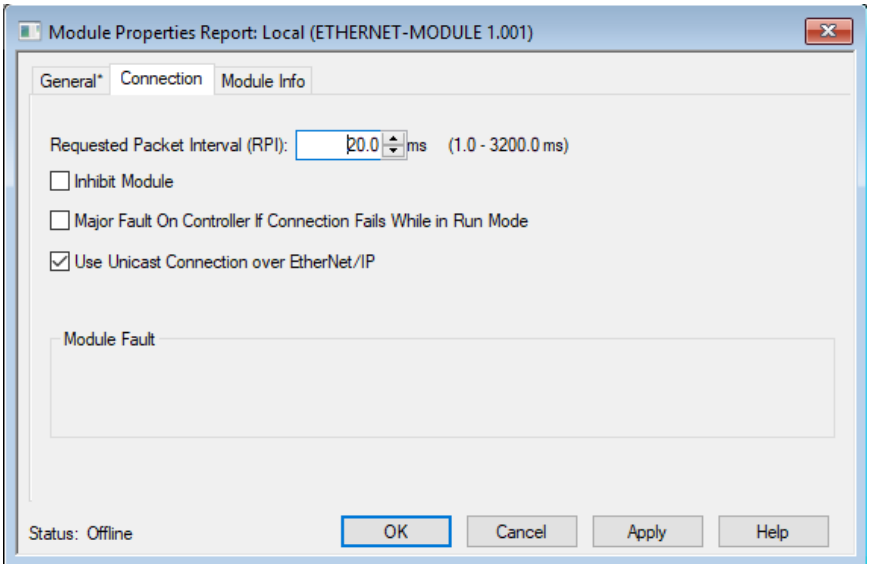


Figure 10 - Setting the RPI

6. Click the **Apply** button to save the properties of the newly created module.
7. Currently the Module Info tab is not functioning. The Zaxis vendor information must be registered and published by OVDA before he Module Info tab provides any meaningful information.
8. Open the **Controller Tags** menu item to display the newly created tags for the Isaac module as shown in Figure 11. E.g. the Isaac_0:C, Isaac_0:I, Isaac_0:O tags.

Name	Value	Force Mask	Style	Data Type	Description
Isaac_0:C	{...}	{...}		AB:ETHERNET_MODULE:C:0	
Isaac_0:I	{...}	{...}		AB:ETHERNET_MODULE_DINT_68Bytes:I:0	
Isaac_0:O	{...}	{...}		AB:ETHERNET_MODULE_DINT_16Bytes:O:0	

Figure 11 - Controller Tags

Isaac Configuration, Input and Output Tag Groups

The *Controller Tags* window (see Figure 11) will always have the Input (`Isaac_0:I`) and Output (`Isaac_0:O`) tags, the Configuration (`Isaac_0:C`) tags will only be available when a size other than zero is placed in the **Size** field of the Configuration Instance. The following tables: Table 3, Table 4 and Table 5, represent the structure of an Isaac tester module.

These three tag groups are defined in the EDS file or can be imported as UDTs (User Defined Types). Both the EDS file and the UDTs (found in Isaac_Udts.xlsx) are available from Zaxis support and/or the Zaxis download web page.

Configuration Tags

The configuration tags are used to initialize selected parameters within the Isaac tester. These tags are not required and will not be sent to the Isaac if the size of the Configuration instance is set to zero when using the. If the Configuration size is non-zero, it must be set to the size indicated in step 4.g above, or undesirable results will occur.

When using the Configuration with the EDS file method these tags are always sent and therefore must contain the deaired values.

Configuration tags are sent to the Isaac tester when the communications channel is created, e.g. when the Ethernet module comes online.

Name	Data Type	Valid Values	Description
AntiTie	SINT	1,2,4	Test start when: 1 = input 1, 2 = input 1 & 2, 4 = input 1 & start button.
CurrentProgram	SINT	0 thru 99	Starting program number.
ResultsOut	SINT	> 0	Integer value of results out pulse width.
ResultsOutFraction	SINT	> 0	Fraction of results out pulse width.
BaudRate	SINT	1 thru 4	Baud rate for RS232 port. 1 = 9600, 2=19200, 3=57600, 4=115200
DataLogging	SINT	1 thru 4	When to log data on RS232 port. 1=Off, 2=Results, 3=0.1 sec, 4=1 sec
PressureUnits	SINT	1 thru 7	Units for pressure 1=PSIG, 2=MBAR, 3=MMHG, 4=INH2O, 5=KPA, 6=CMH20, 7=INHG
FlowUnits	SINT	0 or 1	Units for flow. 0=sccm, 1=slm
TestPresDigits	SINT	0 thru 3	Number of digits after decimal. 0 thru 3
ResultsPresDigits	SINT	0 thru 5	Number of digits after decimal. 0 thru 5
FlowPressureDigits	SINT	0 thru 3	Number of digits after decimal. 0 thru 3
SpareCfg1	SINT	0	Spare data
cosType	INT	0 to 0xFF	Indicates which change-of-states events should be reported. Should be set to 0.
SpareCfg2	SINT	0	Spare data
SpareCfg3	SINT	0	Spare data

Table 3 – Ethernet-Module Configuration Tags

The TSi (touch screen interface) has input fields with the same or similar names to the configuration tags, more detailed information as to the purpose of these tags may be found in the Isaac User's Manual. These configuration tags preform the same functions as the Tsi fields.

When an invalid value is given, such as a -1 for the *CurrentProgram*, is given the value will be ignored by the Isaac tester and the existing value will be used. This prevents the PLC from overwriting these values when they have been set using the TSi. For example, suppose the Tsi was used to set the number of test pressure digits to 3 *(TestPresDigits)* if the PLC has a value of 0, each time the PLC connected to the Isaac the 3 would be changed to 0. To prevent this overwriting, if the PLC has a value of -1 (e.g., and invalid value) the -1 will be ignored by the Isaac and the existing value of 3 would be used.

cosType – Change-Of-State flags

The change-of-state flags indicate which events should be reported by the producer (the Isaac tester) to the consumer (usually the PLC). Setting one of the change-of-state flags is the indicator to the Isaac tester which change-of-state events the consumer wants to be notified of. The default is to only report the test results, which means when the EDS file is used to create the Isaac Ethernet modules and the Change of State connection type is selected the only messages sent by the Isaac will be status message (which will occur every RPI milliseconds) and the test results message. Thus, ensuring that the PLC is sent a "heartbeat" e.g., a status message every RPI milliseconds and the test results at the end of each test.

The change-of-state bits indicate the reason the message was produced by the Isaac tester.

- CosStatus – 0x00 when a status message has been requested by the originator (e.g., the PLC) or when the RPI has expired.
- CosSequencePts – 0x01 when a test is running, and the sequence has changed and/or the elapse time has changed.
- CosTestStart – 0x02 at the start/beginning of a test.
- CosTestsResults – 0x04 when a test has completed.
- CosPressure – 0x08 when a test is not running but the pressure changes.
- CosProgram – 0x10 when a program parameter has changed.
- CosFirmwareCfg – 0x20 when a firmware parameter changed.
- CosProgNumb – 0x40 when a flash parameter has changed.
- CosTestValve – 0x80 valve is manually opened/closed.

It is Zaxis recommendation that only the CosTestResults bit be set for normal operation. Setting the other bits may cause communication delays and communication disconnects.

PLC Output Tags

Output tags are sent by the PLC to the Isaac tester every **Request Packet Interval (RPI)** milliseconds as configured above, see Figure 10 - Setting the RPI. Once the Isaac scans i.e., process, the output tags from the PLC, the input tags are sent from the Isaac tester to the PLC.

Name	Data Type	Valid Values	Description
Start	BOOL	0 or 1	Bit 0 = Start the tests
StartAntiTie	BOOL	0 or 1	Bit 1 = Start on input 1 and 2 (Anti-Tie)
Abort	BOOL	0 or 1	Bit 2 = Stop the current test
Retest	BOOL	0 or 1	Bit 3 = Rerun the failed test
Valve1On	BOOL	0 or 1	Bit 4 = Turn pressure valve one on
Valve2On	BOOL	0 or 1	Bit 5 = Turn pressure valve two on
Valve3On	BOOL	0 or 1	Bit 6 = Turn pressure valve three on
Valve4On	BOOL	0 or 1	Bit 7 = Turn pressure valve four on
IgnoreCmds	BOOL	0 or 1	Bit 8 = Ignore EtherNet I/P commands
StatusGet	BOOL	0 or 1	Bit 9 = Get current device status
StartWInterlock	BOOL	0 or 1	Bit 10 = Start the tests when StartAntiTie is high.
ChanDisabledEIP	BOOL	0 or 1	Bit 11 = Allow Channels to be disabled via EtherNet I/P
Chan1Disable	BOOL	0 or 1	Bit 12 = Disable Channel 1
Chan2Disable	BOOL	0 or 1	Bit 13 = Disable Channel 2
Chan3Disable	BOOL	0 or 1	Bit 14 = Disable Channel 3
Chan4Disable	BOOL	0 or 1	Bit 15 = Disable Channel 4
Spare	BOOL	0	Bit 16 = Spare bit for future use
Spare	BOOL	0	Bit 17 = Spare bit for future use
LeakStandard	BOOL	0 or 1	Bit 18 = Run current program with Leak Standard
CurrentProgram	SINT	-1 thru 99	Set to current program. -1 means don't change current program.
SpareByteOut1	SINT		Spare data
SpareIntOut1	INT		
SpareDIntOut1	DINT		
SpareDIntOut2	DINT		

Table 4 - Output Tags

Again, the tag names are similar or the same as the Tsi field names, or buttons, e.g. Start is the green push button, Abort the red push button. The IgnoreCmds bit tells the Isaac tester to ignore any commands via EtherNet I/P. This bit should be set when Tsi is being used to configure the Isaac tester to ensure there is no conflict between the PLC and the Tsi.

The CurrentProgram tag allows the currently running program to be changed to a different program. This could be used if the product being tested needs to have a pressure decay test as well as a flow test. Program zero could be configured to run a pressure decay test and program one a flow test. The PLC could set the CurrentProgram to zero and start the test (set Start) if the product passed the pressure decay test, the PLC could change the CurrentProgram to one and start the test (set Start), which would then run the flow test.

The Start bit is ignored until a transition of 0 to 1 is seen and the tester is not busy (e.g. Busy bit is 0 see PLC Input Tags). When Start bit transition is seen the current program/test will be run. The PLC should set the Start bit to 0 when the Busy bit goes to 1 e.g., high. The Start bit should not be kept high once the Busy bit goes high, or if communication between the PLC and the tester is lost. Keeping the Start bit high may cause false starts when communication is reestablished and/or when the test has been completed.

The ability to disable/enable a given channel when the tester has multiple channels and is in concurrent mode is provided by the ChanDisableEIP and Chan?Disable bits. The ChanDisableEIP bit must be set, indicating that the PLC wishes to Disable/Enable channels. With the ChanDisableEIP bit set the four Chan?Disable bits indicate which channels should be disabled during the current test. Thus, allowing one or more channels to be disabled when a part is not able to be presented on the given channel.

The Chan?Disable bits are not written to flash and therefore when power is cycled this information is lost.

PLC Input Tags

Input tags are sent to the PLC from the Isaac tester after the Output tags have been scanned, or processed, by the Isaac tester. Input tags provide the current state of the Isaac to the PLC. The following table list the tags returned from the Isaac tester.

Name	Data Type	Valid Values	Description
CosStatus	BOOL	0 or 1	See: cosType – Change-Of-State flags
CosSequencePts	BOOL	0 or 1	
CosTestStart	BOOL	0 or 1	
CosTestResults	BOOL	0 or 1	
CosPressure	BOOL	0 or 1	
CosProgram	BOOL	0 or 1	
CosFirmwareCfg	BOOL	0 or 1	
CosProgNumb	BOOL	0 or 1	
CosTestValve	BOOL	0 or 1	
ClassId	INT	0 or 1	Id of the class that changed state
Instance	INT	1 thru 99	Instance Id of the class that changed state
Attribute	INT	1 thru 99	Attribute number that changed state
Busy	BOOL	0 or 1	Bit 0 = Isaac is busy
GlobalPass	BOOL	0 or 1	Bit 1 = All the ports passed the test
GlobalFail	BOOL	0 or 1	Bit 2 = One or more ports failed the test
Fail_1	BOOL	0 or 1	Bit 3 = Port 1 failed
Fail_2	BOOL	0 or 1	Bit 4 = Port 2 failed
Fail_3	BOOL	0 or 1	Bit 5 = Port 3 failed
Fail_4	BOOL	0 or 1	Bit 6 = Port 4 failed
Pass_1	BOOL	0 or 1	Bit 7 = Port 1 Passed
Pass_2	BOOL	0 or 1	Bit 8 = Port 2 Passed

Pass_3	BOOL	0 or 1	Bit 9 = Port 3 Passed
Pass_4	BOOL	0 or 1	Bit 10 = Port 4 Passed
AntiTie	BOOL	0 or 1	Bit 11 = Anti Tie is enabled
Interlock	BOOL	0 or 1	Bit 12 = Interlock is required
Abort	BOOL	0 or 1	Bit 13 = Test Aborted
LeakStandard	BOOL	0 or 1	Bit 14 = Program is running with Leak Standard
Fault	BOOL	0 or 1	Bit 15 = Faulted
SpareByteIn1	SINT	0	Spare data
TestPresValve1	BOOL	0 or 1	Bit 0 = State of Test Pressure Valve 1
TestPresValve2	BOOL	0 or 1	Bit 1 = State of Test Pressure Valve 2
TestPresValve3	BOOL	0 or 1	Bit 2 = State of Test Pressure Valve 3
TestPresValve4	BOOL	0 or 1	Bit 3 = State of Test Pressure Valve 4
CurrentProgram	SINT	0 thru 99	Current program
TestType	SINT	0 thru 11	Type of test being run. PD, POCC, VD, VOCC, Flow, etc.
PortEnabled1	BOOL	0 or 1	Bit 0 = Port 1 enabled
PortEnabled2	BOOL	0 or 1	Bit 1 = Port 2 enabled
PortEnabled3	BOOL	0 or 1	Bit 2 = Port 3 enabled
PortEnabled4	BOOL	0 or 1	Bit 3 = Port 4 enabled
Sequence	SINT	0 thru 9	Current test sequence 0 = Idle.
ValveMask	INT	0 thru 15	Bit mask showing which valves are on/off.
ElapseTime	DINT	> 0	Sequence elapse time in 1/10's seconds.
PortStatus1	SINT	0 thru 17, 250 thru 255	Status of Port 1. E.g., Testing, Leak, Occl, Perr
PortStatus2	SINT		Status of Port 2. E.g., Testing, Leak, Occl, Perr
PortStatus3	SINT		Status of Port 3. E.g., Testing, Leak, Occl, Perr
PortStatus4	SINT		Status of Port 4. E.g., Testing, Leak, Occl, Perr
PressurePort1	DINT		Test pressure * 1000. E.g., fixed-point decimal
PressurePort2	DINT		Test pressure * 1000. E.g., fixed-point decimal
PressurePort3	DINT		Test pressure * 1000. E.g., fixed-point decimal
PressurePort4	DINT		Test pressure * 1000. E.g., fixed-point decimal
TestResults1	DINT		Test results for port 1. The pressure delta between the start and end of the test, or the flow at the end of the test.
TestResults2	DINT		Test results for port 2
TestResults3	DINT		Test results for port 3
TestResults4	DINT		Test results for port 4
Flow	DINT		Current flow * 1000. E.g., fixed-point decimal.
SpareDIntIn1	DINT		Spare data

Table 5 - Input Tags

COS Change-of-State bits

Change-of-state bits indicate to the originator (e.g., usually the PLC) why a message was sent from the Isaac. Allowing the PLC (or originator) to determine the current state of the Isaac. For example, when the CosTestResults bit is set the values in the TestResults fields are valid end of test results.

The Busy bit preforms the same functions as the Busy input signals as described in the User's Manual, e.g., the tester is busy preforming a test.

The other bits are described in the Isaac User's manual.

Valve Mask

The valve mask field indicates which valves are on/off, 1 indicating on and 0 off. Valve mask change based on the current type of testing being run and/or the current valve configuration. However, the standard masks are shown below.

Valve Name	Hex Value		Valve Name	Hex Value
Leak Standard	0x10		Vacuum	0x04
Clamp 1	0x80		Clamp 2	0x40
PD Pre-Fill	0x03		VD Pre-Fill	0x11
PD Fill	0x03		VD Fill	0x11
PD Settle	0x02		VD Settle	0x00
PD Test	0x00		VD Test	0x00
Occ Pre-Fill	0x03		VOcc Pre-Fill	0x01
Occ Fill	0x03		VOcc Fill	0x01
Occ Settle	0x02		VOcc Settle	0x00
Occ Test	0x08		VOcc Test	0x08
Flow Pre-Fill	0x18			
Flow Fill	0x18			
Flow Test	0x18			

Table 6 - Valve Masks

Contact Zaxis support to confirm the valve masks for your Leak testers.

Isaac Firmware/Program configuration tags

The Isaac has two different types of configuration or setup tags: 1) tags that remain the same regardless of which program is running and 2) tags for each individual program. This document will refer to these different types as firmware paraments and program parameters respectively.

Regardless of the type of parameter (firmware or program) both are saved in flash memory and are read into RAM (random access memory) each time the Isaac is powered up and/or reset. The flash memory doesn't require a battery and is therefore more reliable for longer periods of time than battery backed up RAM.

This also implies that these parameters do not (and should not) be written to the Isaac tester each time a program (test) is run. Doing so causes unnecessary overhead and slows the Isaac's response times which may cause timeout problems.

Firmware Parameters

Firmware parameters are those parameters that remain the same regardless of which program (or test) is being executed. Firmware parameters are typically set once on initial configuration of the Isaac tester and very seldom or never, change. The following table describes the Firmware parameters.

Name	Data Type	Valid Values	Description
CosType	INT	Read Only	Change-of-State flags
ClassId	INT	Read Only	Id of the class that changed state
Instance	INT	Read Only	Instance Id of the class that changed state
Attribute	INT	Read Only	Attribute number that changed state
LockPrgEdits	BOOL	0 or 1	Tsi Program parameter editing is locked
LockCurrentPrgEdits	BOOL	0 or 1	Tsi Current program editing is locked
LockCalibrationEdits	BOOL	0 or 1	Tsi Calibration edits are locked
ElectronicReg	BOOL	0 or 1	Electronic regulator installed
ManualReg	BOOL	0 or 1	Manual regulator installed
Concurrent	BOOL	0 or 1	Run test concurrently on all channels
BuzzerEnabled	BOOL	0 or 1	Buzzer is enabled
BusyImm	BOOL	0 or 1	0 = send busy msessage after all overhead, 1 = send busy with pass/fail
DhcpEnabled	BOOL	0 or 1	DHCP is enabled
OkToResetDevice	BOOL	0 or 1	Ok to reset the device because of date/time changes etc.
LinkedFail	BOOL	0	Test fails when linked program fails.
HighBleed	BOOL	0	High bleed valve enabled.
ElectRegAlwaysOn	BOOL	0 or 1	Electronic regulator is always on.
SpareBit13	BOOL	0	Spare data
SpareBit14	BOOL	0	Spare data
SpareBit15	BOOL	0	Spare data
TesterNumber	SINT	1 to 128	Device number on 485 bus

BaudRate	SINT	0 thru 4	0 = Don't change, 1=9600, 2=19200, 3=57600, 4=1152
DataLogging	SINT	0 thru 4	0 = Don't change, 1=LogOff, 2=logResult, 3=log 0.1 secs, 4= log 1.0 sec
PressureUnits	SINT	0 thru 7	0 = Don't change, 1=PSIG, 2=MBAR, 3=MMHG,4=INH2O, 5=KPA, 6=CMH2O, 7=INHG
FlowUnits	SINT	0 or 1	0 = sccm 1 = slm
TestPresDigits	SINT		Number of digits after test pressures 0 to 3
ResultsPresDigits	SINT	0 thru 5	Number of digits after results pressures 0 to 5
FlowPresDigits	SINT	0 thru 3	Number of digits after flow pressures 0 to 3
BCD	SINT	0 thru 99	BCD Inputs. See documentation
NumberChannels	SINT	1 thru 4	Number of installed channels/ports
Sensors	SINT	1 thru 4	Number of installed sensors. 1st nibble = Pressures sensors, 2nd nibble = Flow.
AntiTie	SINT	1, 2, or 4	Test start when: 1 = input 1, 2 = input 1 & 2, 4 = input 1 & start button
AvailTestTypes	SINT	O thru 10	Types of tests that can be preformed
SpareByte1	SINT	0	Spare data
SpareInit1	INT	0	Spare data
IoResetTime	INT	0 thru 3000	Number of milliseconds the I/O remains in a given state
FilterPress	DINT	0 thru 65000	Read only Pressure filter cutoff
FilterFlow	DINT	0 thru 65000	Read only Flow filter cutoff
FilterFillSettle	DINT	0 thru 65000	Read only Fill filter cutoff
SpareDint1	DINT	0	Spare data
HighBleedPressure	INT	0 thru 65000	High Bleed pressure limit
SpareInt2	INT	0	Spare data
AtmPress	DINT	0 to 100	Atmospheric Pressure
SerialNumber	STRING_16	Read Only	Serial number. 16 bytes.
IpAddress	STRING_16	Valid I/P	I/P Address. 16 bytes.
IpNetmask	STRING_16	Valid I/P	I/P Netmask. 16 bytes.
IpGateway	STRING_16	Valid I/P	I/P Gateway. 16 bytes.
MacAddress	STRING_20	Read Only	MAC Address. 20 bytes.
DateTime	STRING_20	Valid Date/Time	Date and time: YYYY/MM/DD hh:mm:ss
ModelNumber	STRING_24	Read Only	Model number or TIN
OptionsAndFittings	STRING_32	Read Only	Installed options and fittings
SccsRevDate	STRING_24	Read Only	Source Code Control Date stamp
SccsRev	STRING_12	Read Only	Source Code Revision
SccsRevRange	STRING_24	Read Only	Source Coe Revision range

Table 7 - Configuration Parameters

All of the above firmware parameters are typically defined as a User-Defined Type (UDT) and imported from a Microsoft Excel spread sheet (Isaac_Utds.xlsx) which can be downloaded from the Zaxis web site.

Reading and Writing of Firmware Parameters

Firmware parameters can be read and written using the PLC Message (MSG) instruction.

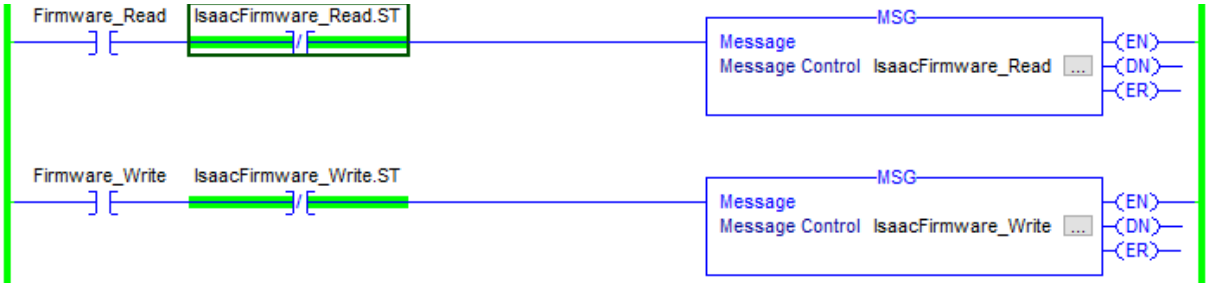


Figure 12 – Firmware Read/Write Ladder Logic

Firmware parameters are read/written using the CIP Read/Write All Attributes message. These CIP messages (Read/Write All Attributes) are created on the PLC using the *CIP Generic* **Message Type** see Figure 12.

Figure 13 shows the required parameters for the MSG tag.

- **Service Code** of 1 (e.g. Read all attributes) or 2 (e.g. Write all attributes)
- Vendor specific **Class** code of 0x64 (Hex)
- **Instance** of 1.
- **Destination Element** should be a *User-Defined* data type which minors the tags described in Table 7.

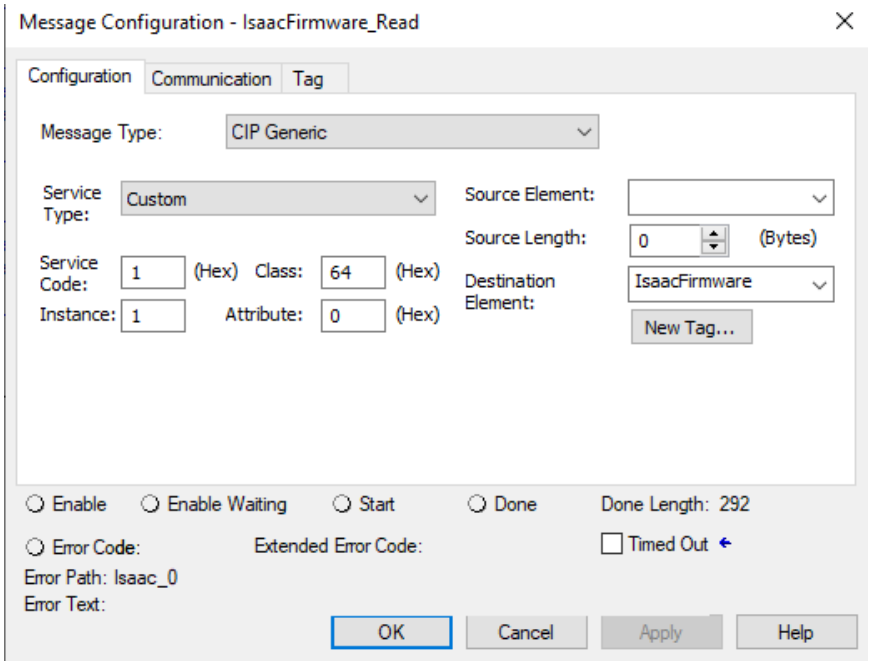


Figure 13 - Read Isaac Configuration MSG

The *Communication* tab references the **Path** of the Isaac Ethernet module that was added above in the Configuration with the EDS file or Configuration using a Generic Ethernet Module sections, see the Path field shown below in Figure 14.

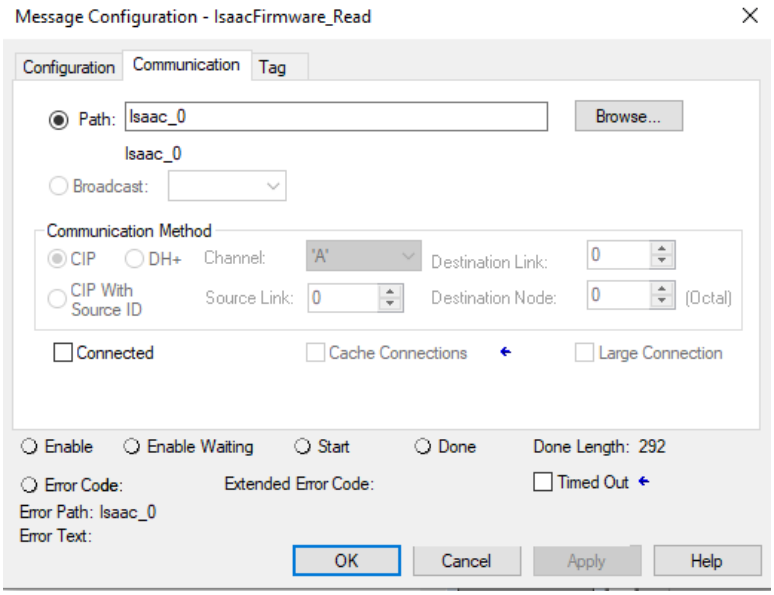


Figure 14 - MSG Communication tab

The MSG Tag can be given any meaningful name as shown in Figure 15 below to complete the configuration.

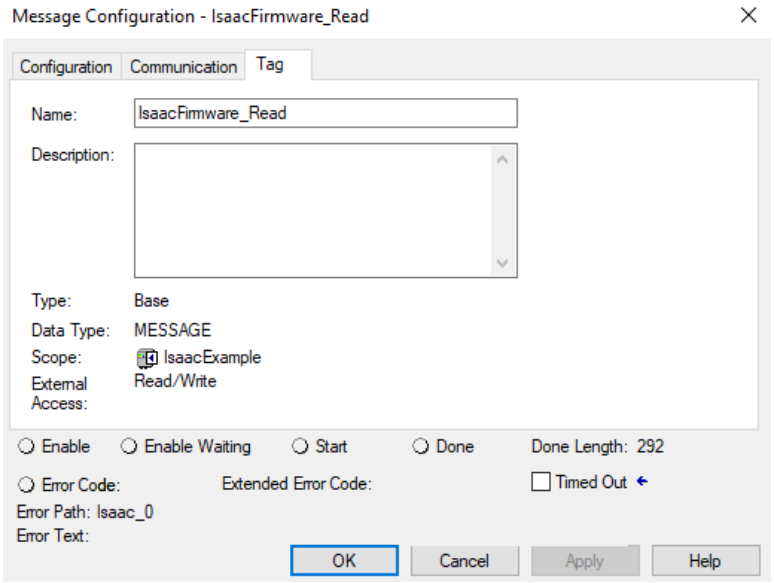


Figure 15 - MSG Tag tab

The basic difference between reading and writing setup parameters is the **Service Code** and the **Source Element**. The Service Code changes from 1 to a 2, and the **Source Element** now references the user defined data to be written. The MSG parameters for writing are show in Figure 16.

The *Communication* tab and *Tag* tab have the same parameters as reading the setup.

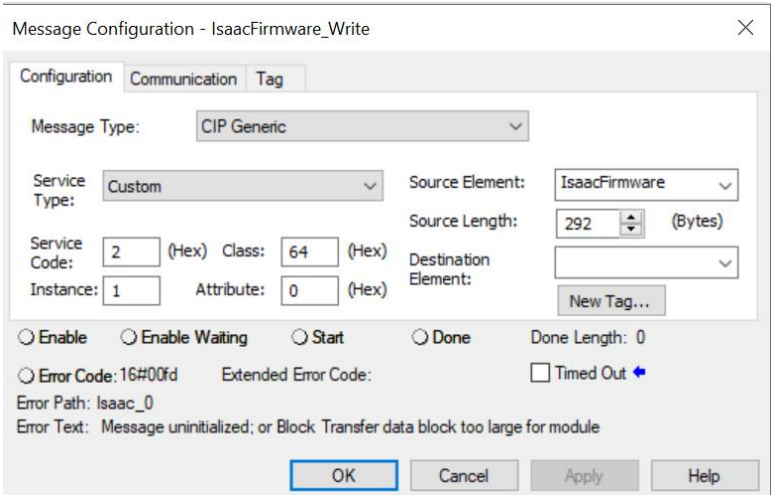


Figure 16 - Write Isaac Firmware Configuration MSG

Program Parameters

Program parameters allow the Isaac tester to be configured such that different programs, loops or tests can be run by the same tester, yet perform completely different tasks. Programs can be linked allowing one program to execute after the parent program, with each program having different parameters. Table 8 shows the different program parameters and their meaning.

Name	Data Type	Valid Values	Description
CosType	INT	Read Only	Change-of-State flags
ClassId	INT	Read Only	Id of the class that changed state
Instance	INT	Read Only	Instance Id of the class that changed state
Attribute	INT	Read Only	Attribute number that changed state
PrgNumb	SINT	0 thru 99	Program number
SpareByte1	SINT	0	Spare Byte 1
SpareIntIn1	INT	0	Spare integer
PortEnabled1	BOOL	0 or 1	Port 1 is enabled
PortEnabled2	BOOL	0 or 1	Port 2 is enabled
PortEnabled3	BOOL	0 or 1	Port 3 is enabled
PortEnabled4	BOOL	0 or 1	Port 4 is enabled
FixtureValve1	BOOL	0 or 1	Keep Fixture valve 1 on during test step
FixtureValve2	BOOL	0 or 1	Keep Fixture valve 2 on during test step
ClampHoldFail	BOOL	0 or 1	Hold clamps on fail
Rslts232	BOOL	0 or 1	Output results to RS232 port
RsltsEthr	BOOL	0 or 1	Output results to Ethernet port 23
AutoFFill	BOOL	0 or 1	Auto Fast Fill or timed
LeakStd	BOOL	0 or 1	Leak Standard
AutoVent	BOOL	0 or 1	Auto Vent or timed
IncreaseLimit	BOOL	0 or 1	Enable Increase Limit
EvalAtEnd	BOOL	0 or 1	Evaluate at End of Test
ElectReg	BOOL	0 or 1	Read only Use Last Good DAC when Electronic Reg available.
SaveDacOnAbort	BOOL	0 or 1	Save DAC reading on Abort when in settle or test sequence.
TestType	SINT	0 thru 10	Type of test to run. PD, POCC, VD, VOCC, Flow, Burst, Vburst, Crack, Creep, Flow2, Reseat
NextPrg	SINT	-1 thru 99	Next program to run. -1 means not linked.
ElectRegDelay	SINT	0 thru 100	% of fill timer to allow for Electronic Regulator to seek
SpareByte2	SINT	0	Spare
TimeIdle	DINT	0	Reserved. Do not use.
TimeClmp1	DINT	0 thru 10000	Elapse time in 1/10 seconds for Clamp 1 Sequence
TimeClmp2	DINT	0 thru 10000	Elapse time in 1/10 seconds for Clamp 2 Sequence
TimeFFill	DINT	0 thru 10000	Elapse time in 1/10 seconds for Fast Fill (Pre-Fill) Sequence

TimeFill	DINT	0 thru 10000	Elapse time in 1/10 seconds for Fill Sequence
TimeSettle	DINT	0 thru 10000	Elapse time in 1/10 seconds for Settle Sequence
TimeTest	DINT	0 thru 10000	Elapse time in 1/10 seconds for Test Sequence
TimeVent	DINT	0 thru 10000	Elapse time in 1/10 seconds for Vent Sequence
TimeUnClmp1	DINT	0 thru 10000	Elapse time in 1/10 seconds for Unclamp 2 Sequence
TimeUnClmp2	DINT	0 thru 10000	Elapse time in 1/10 seconds for unclamp 1 Sequence
SpareDInt1	DINT	0	Spare
SpareDInt2	DINT	0	Spare
TestPres	DINT		Test Pressure as a fixed decimal number
TestPresMin	DINT		Minimum Test Pressure as a fixed decimal number
TestPresMax	DINT		Maximum Test Pressure as a fixed decimal number
RegKp	DINT	0 thru 10000	Regulator P-I Kp
RegKi	DINT	0 thru 10000	Regulator P-I Ki
RegKd	DINT	0 thru 10000	Regulator PI Kd
PresFFill	DINT		Pre-Fill Pressure
PresTestMin	DINT	0 thru 10000	Minimum test pressure. E.g., Decay, Min Crack/Burst/Flow etc.
PresTestMax	DINT	0 thru 10000	Maximum test pressure e.g., Increase, Max Crack/Burst/Flow etc.
PresTestTrigger	DINT	0 thru 10000	Trigger test pressure e.g., Trigger Crack/Burst etc.
Volume	DINT	0 thru 10000	Air volume used to calculate approx. Leak rate. 0 to disable calculation.
RampRate	DINT	0 thru 100	How fast to ramp up the Fast Fill pressure
SpareDInt3	DINT	0	Spare
PrgName	STRING_16	Printable Chars	Program name length (4 bytes) + Program name (16 bytes)

Table 8 - Program Configuration Parameters

All of the above program parameters are typically defined as a User-Defined Type (UDT) and imported from a Microsoft Excel spread sheet (Isaac_Utds.xlsx) which can be downloaded from the Zaxis web site.

Reading and Writing of Program Parameters

Program parameters are read and written using the CIP Read/Write All Attributes message. These CIP messages (Read/Write All Attributes) are configured on the PLC using the *CIP Generic* **Message Type** (see Figure 17).

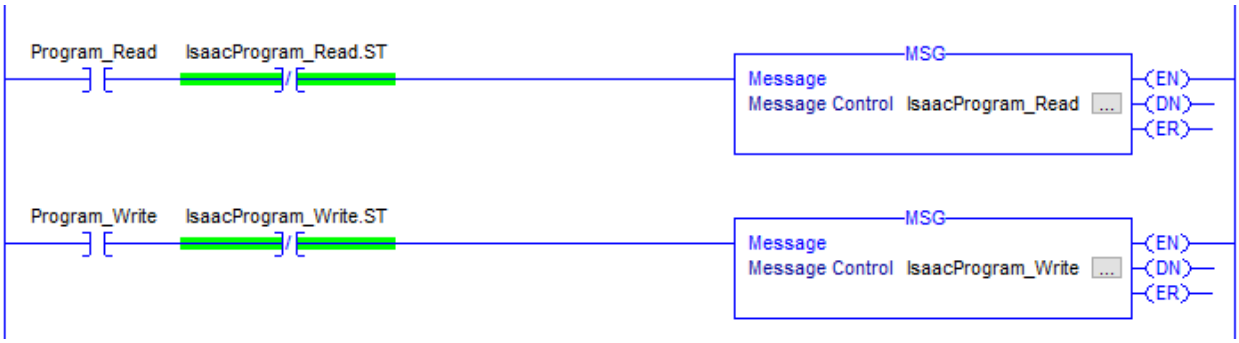


Figure 17 - Program Read/Write Ladder Logic

Figure 18 shows the required parameters for the MSG tag.

- **Service Code** of 1 (e.g., Read all attributes) or 2 (e.g., Write all attributes)
- Vendor specific **Class** code of 0x65 (Hex)
- **Instance** of 0 to 99.
- **Destination Element** should be a *User-Defined* data type which minors the tags described in Table 8.

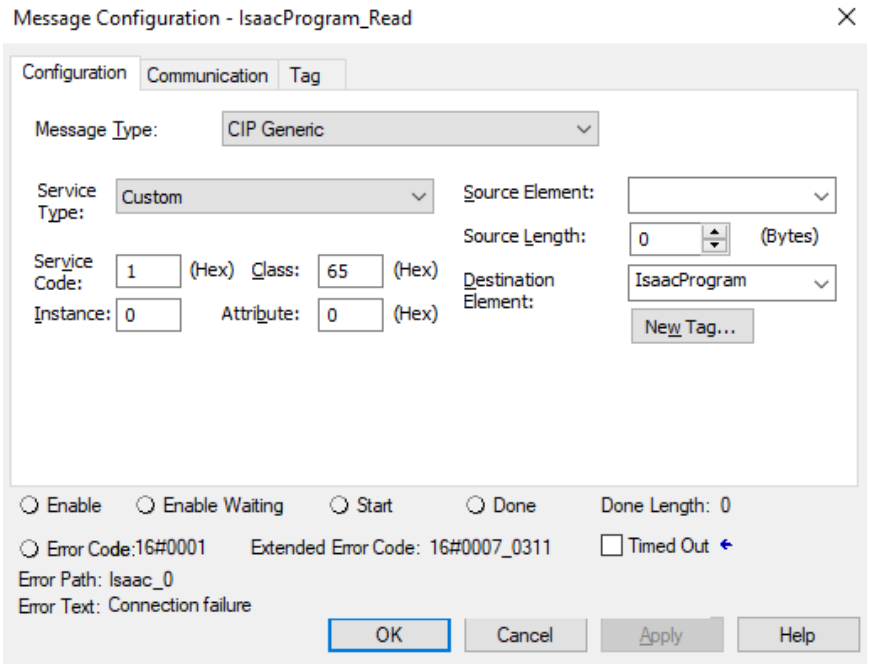


Figure 18 - Read Programs Parameters Configuration Tab

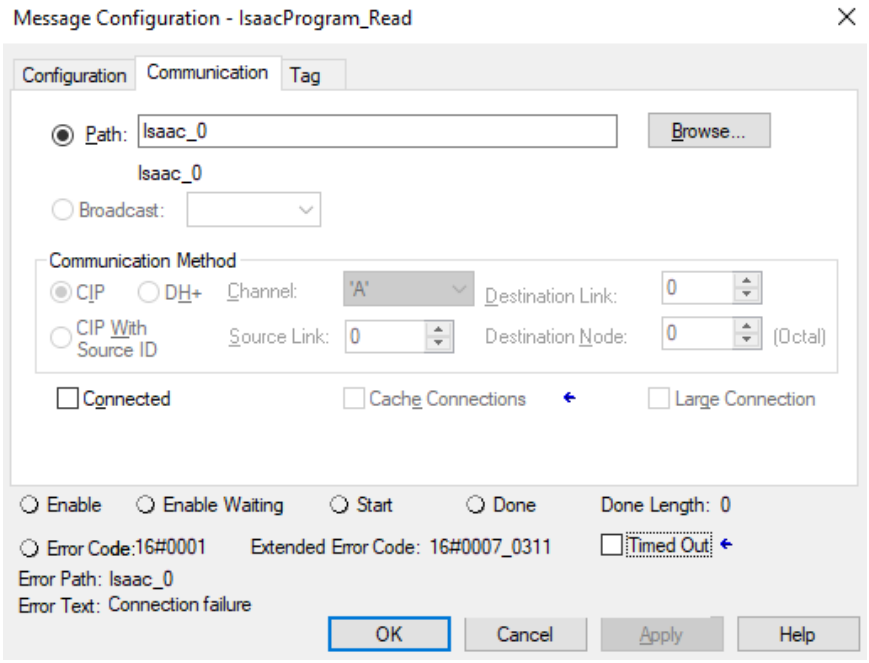


Figure 19 – Read Isaac Program Parameters Communication Tab

The basic difference between reading and writing program parameters is the **Service Code** and the **Source Element**. The Service Code changes from 1 to a 2, and the **Source Element** now references the user defined data to be written. The MSG parameters for writing are show in Figure 20.

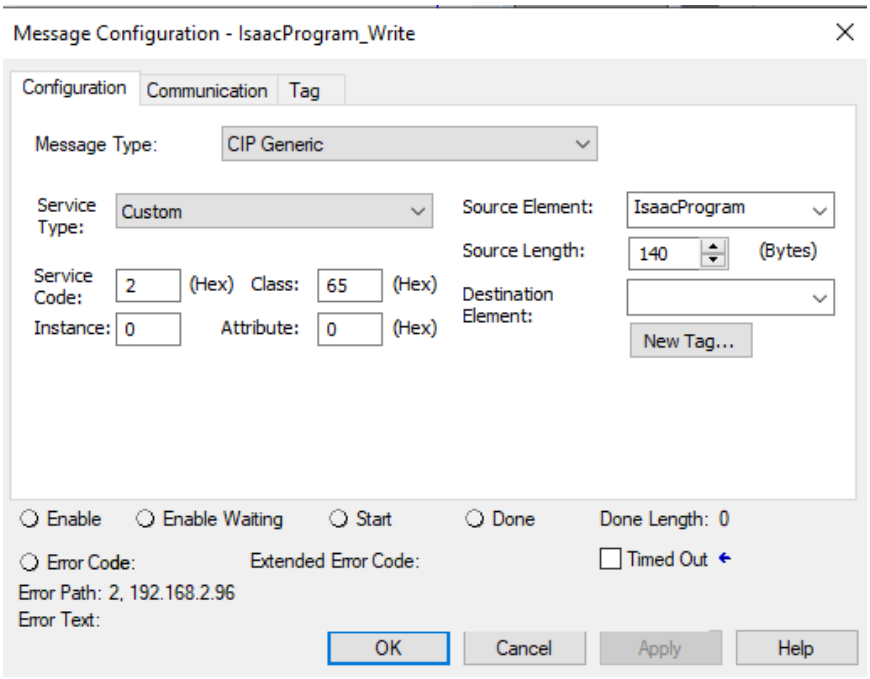


Figure 20 - Write Isaac Program Parameters Configuration Tab

Reuse of the MSG tag

The MSG tag can be reused to configure multiple Isaac testes following the examples shown in the "Logix 5000 Controllers Message Programming Manual" created by Rockwell automation. See https://literature.rockwellautomation.com/idc/groups/literature/documents/pm/1756-pm012_-en-p.pdf Chapter 3 for more information.

TCP/IP Network Configuration

The default TCP/IP settings are provided in Table 2 above, however these values can be reconfigured using the Firmware Parameters or the predefined CIP TCP/IP object. The CIP TCP/IP object definition is shown in Table 9, and is based on the *TCP/IP Interface Object* in *Volume 2: EtherNet/IP Adaptation of CIP, Chapter 5: Object Library*.

Name	Data Type	Valid Values	Description
Status	DINT	Read only	Interface Status bits. 0x02=Configured from DHCP, 0x04=Hardware configuration.
Configuration Capability	DINT	Read only (0x34)	Bit map of interface capability flags: 0x04=DHCP capable 0x10=Interface configurable 0x20=Hardware configurable
Configuration Control	DINT		Interface control flags: 0x00=Use static configuration 0x02=Use DHCP configuration
Path Size	INT	Read only (3)	Number of 16-bit words in the following padded EPATH.
Path	SINT[4]	Read only	Padded EPATH to physical link object.
IP Address	DINT	0 to 0xFFFFFFFF	Devices I/P address.
Network Address	DINT	0 to 0xFFFFFFFF	Devices network address.
Gateway Address	DINT	0 to 0xFFFFFFFF	Devices gateway I/P address.
Name Server	DINT	0 to 0xFFFFFFFF	Primary name server I/P address.
Name Server 2	DINT	0 to 0xFFFFFFFF	Secondary name server I/P address.
Domain Name	STRING	ASCII Characters	Default domain name.
Host Name	STRING	ASCII Characters	Default host name.
Safety Network Number	SINT[6]	0 to 100	Ignored by Isaac.
TTL	SINT	1 to 255	Time-to-Live value for IP Multicast packets. Ignored by Isaac.
Alloc Control	SINT	0	Multicast address allocation. Ignored by Isaac.
Reserved	SINT	0	Reserved for future use.
Num Mcast	INT	0	Number of multicasts addresses to allocate. Ignored by Isaac.
Mcast Start Addr	DINT	0	Starting multicast address. Ignored by Isaac.
Selected Acd	BOOL	0	Activates use of ACD. Ignored by Isaac.
Acd Active	SINT	0	State of ACD activity. Ignored by Isaac.
Remote MAC	SINT[6]	0	MAC Address of remote node. Ignored by Isaac.
Arp Pdu	SINT[28]	0	ARP PDU. Ignored by Isaac.

EthernetIP Quick Connect	BOOL	0	Quick connect feature. Ignored by Isaac.
Encp Inactivity Timeout	INT	0	Inactivity Timeout. Ignored by Isaac.

Table 9 - TCP/IP Object definition

As can be seen from the table above many of the parameters are not required by the Isaac. However, this object does allow the reconfiguration of necessary TCP/IP parameters, e.g. I/P address, enable/disable DHCP etc.

TCP/IP parameters are read using the CIP Read All Attributes message and written using the CIP Write All Attributes message. These CIP messages (Read/Write All Attributes) are configured on the PLC using the *CIP Generic* **Message Type** with a **Service Code** of 1 (e.g. Read all attributes) or 2 (e.g. Write all attributes), a CIP **Class** code of 0xF5 (Hex) and an **Instance** of 1. As show in Figure 21 and Figure 22.

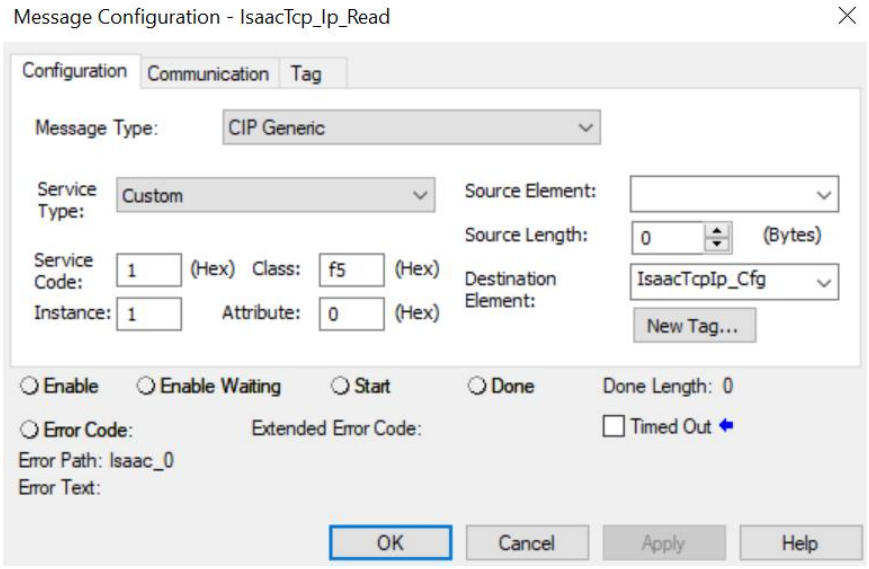


Figure 21 - Reading TCP/IP Configuration Parameters

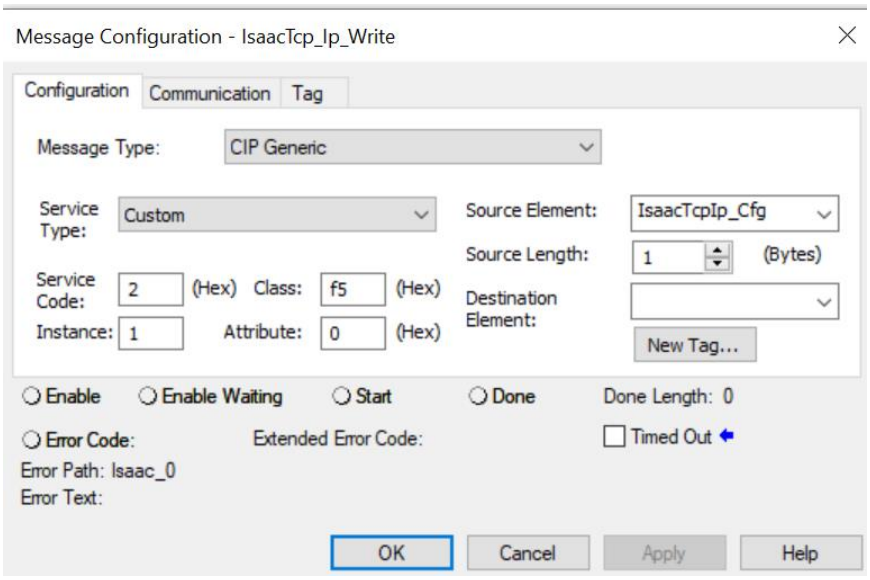


Figure 22 - Write TCP/IP Configuration Parameters

For example, to configure the Isaac to use DHCP, simply set the Configuration Control field to 0x02 and write the TCP/IP object using the MSG instruction.

To set anyone of the I/P addresses (e.g. I/P address, network address, gateway address, etc.) convert each address octet into hex and enter the hex number into the desired field. For example, to set the I/P address of the Isaac to 192.168.168.100 enter the hex value 0xC0A8A864 into the IP Address field and write the TCP/IP object to the Isaac.

Decimal:	192	168	168	100
Hex:	0xC0	0xA8	0xA8	0x64

Table 10 - I/P Address to Hex conversion example

The Isaac default TCP/IP configuration can always be restored by holding the red and green push buttons in for approximately eight seconds while turning the unit on until both buttons blink once. See the EtherNet/IP™ for the Isaac tester overview section for more information.

Trouble Shooting

When problems occur, either in setting up the communications, or communicating with, the Isaac, sometimes it is helpful to get more detailed than a simple fault error. The Isaac displays error and status messages when a network connection has been established on port 8080. These diagnostic messages may give more detail and/or a better understanding of the problems.

A connection to the Isaac diagnostics console can be created using the *Telnet* utility, available on all Unix based system and most Windows system. However, it may need to be turned on under windows. The following figure shows a check mark next to the **Telnet Client**, indicating that it is available for use.

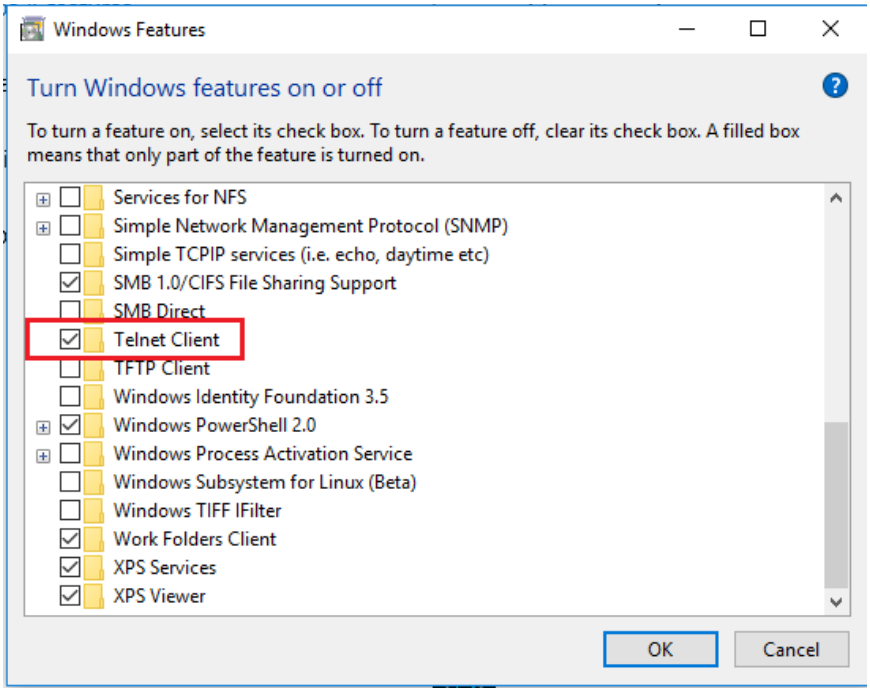


Figure 23 - Windows Telnet Client

With the *Telnet* utility available a connection can, be made using the following command, within a command window (which can be created by "*run cmd*"):

telnet <Isaac_IP_address> 8080 e.g., *telnet 192.168.2.130 8080*

Figure 24 shows the output from the diagnostics console. With several lines showing an invalid connection size. In this case the error was caused by using a size of 16 for the Input Assembly Instance when it should have been 17.

Other and more detailed information can be obtained from the diagnostics console; however, it is recommended that a Zaxis support personal help you in diagnosing and resolving more advanced problems.

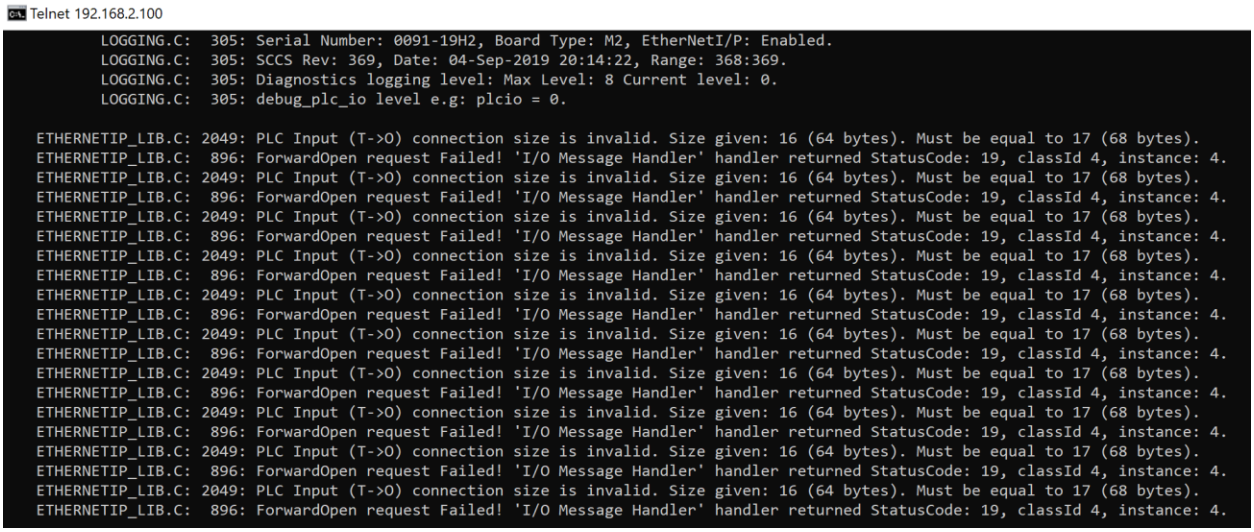


Figure 24 - Diagnostics Console

Error Codes Generated by the Isaac

The following table lists the different error codes that can be returned from the Isaac tester. This table (General Status Codes) indicates the overall success/failure of the given request. These codes are defined by Common Industrial Protocol (CIP). Typically, these codes are display by the PLC.

General Status Codes

Numeric Status Code	Status Code Name	Description
0x00	Success	Service was successfully preformed by the specified object.
0x01	Communications Related Problem	A communications subsystem related problem was detected along the communications path.
0x03	Invalid parameter value	A parameter associated with the request was invalid. Such as an invalid instance id, invalid I/P address, invalid encapsulation parameter,
0x04	Path segment error	The path segment identifier or the segment syntax was not understood by the Isaac.
0x06	Partial transfer	Only part of the expected data was transferred.
0x08	Service not supported	The requested service was not implemented or was not defined for this Object Class/Instance.
0x0C	Object state conflict	The object cannot perform the requested service in its current mode/state. Usually occurs when a calibration command is sent but the Isaac is not in calibration mode.
0x13	Not enough data	The service did not supply enough data to perform the specified operation. Typically caused by an invalid Assembly size.
0x14	Attribute not supported	The attribute specified in the request is not supported. Typically caused by an invalid Assembly size.

0x15	Too much data	The service supplied more data than was expected.
0x16	Object instance does not exist	The object instance specified does not exist in the device.
0x1A	Routing failure, request packet too large	The service request packet was too large. Typically caused by an invalid Assembly size.
0x1F	Vendor specific error	This occurs when the Isaac fails to resolve an I/P address in the ARP cache. This indicates a slow or faulty hub/switch.
0x25	Key Failure in path	The Key Segment that was included as the first segment in the path does not match the destination module.
0x2E	Service Not Supported for Specified Path	The object supports the service, but not for the designated application path (e.g., attribute).

This second table (Encapsulation Status Codes) are a part of the EtherNet I/P encapsulation header. Generally, these codes are not display by a PLC, but can be seen when using a network analyzer such as Wireshark.

Encapsulation Status Codes

Numeric Status Code	Status Code Name	Description
0x00	Success	Service was successfully performed by the specified object.
0x01	Invalid Command	An invalid or unsupported EtherNet I/P encapsulation command was received. A command other than: List Services, List Identity, List Interfaces, Register Session, Unregister Session, Send RR data request or Send Unit Data was received.
0x64	Invalid Session Handle	An incorrect session handle was given when registering an EtherNet I/P session.
0x65	Invalid Length	The length of the encapsulation header was invalid.
0x69	Unsupported Protocol	A protocol version other than 1 was given in the EtherNet I/P encapsulation header.

Definitions

Test Type Values

The test type value reported back to the PLC is a numeric value. No Leak tester supports all test type and typically only has two or three valid types depending on the physical configuration. For example, an iKit typically has at most two test types Pressure Decay and Occlusion. The following table lists all possible types for completeness only.

Numeric Value	Description
0	Pressure Decay
1	Occlusion
2	Vacuum Decay
3	Vacuum Occlusion
4	Flow
5	Burst
6	Vacuum Burst
7	Crack
8	Creep
9	Flow 2
10	Valve Reseal
11	Null

Table 11 - Test Types

Channel/Port Status

A channel (or port) can be in several different states, depending on the type of test currently being run. Because it is possible to have multiple channels that can be run in sequence or concurrently each channel has its own status. The following table lists all the possible channel status. Based on the physical configuration of the tester some status will never be reported. For example, a pressure decay test will never report Burst Pass status.

Numeric Value	Description
0	Testing – The default value. Is testing or ready to test.
1	Leak Error
2	Occlusion Error
3	Pressure High Error
4	Pressure Low Error
5	Gross Leak Error
6	High Flow Error
7	Low Flow Error
8	No Burst at High pressure
9	No Burst at Low pressure
10	Low Burst
11	No Burst occurred
12	High Crack Error

13	Low Crack Error
14	No Crack Error
15	Increase Limit Exceeded Error
16	High Reseat Error
17	Low Reseat Error
250	Test Aborted
252	Burst Test Passed
253	Crack Test Passed
254	Occlusion Test Passed
255	Test Passed

Table 12 - Channel/Port Status Values

It is recommended that the Boolean Pass/Fail flags be used to determine if a test passed and/or failed and the port status only be used to indicate why the failure.

Test Sequence

Based on the selected program settings a test consists of several steps or sequences. Some sequences are not needed if a given setting is zero or not selected. For example, if the pre-fill time is set to zero, no pre-fill sequence will be performed. The following table lists all possible sequences.

Numeric Value	Description
0	Idle – Default value.
1	Clamp 1
2	Clamp 2
3	Pre-Fill
4	Fill
5	Settle
6	Testing
7	Venting
8	Unclamp 1
9	Unclamp 2

Table 13 - Test Sequences

Because of timing not all sequences will be seen by the PLC. For example, if the Venting sequence is set to auto, it is possible that parting being tested could be vented before the next RPI and thus not seen by the PLC.