

IND570 Weighing Terminal



METTLER TOLEDO

IND570 Weighing Terminal

METTLER TOLEDO Service

Essential Services for Dependable Performance of Your IND570 Weighing Terminal

Congratulations on choosing the quality and precision of METTLER TOLEDO. Proper use of your new equipment according to this Manual and regular calibration and maintenance by our factory-trained service team ensures dependable and accurate operation, protecting your investment. Contact us about a service agreement tailored to your needs and budget. Further information is available at www.mt.com/service.

There are several important ways to ensure you maximize the performance of your investment:

1. **Register your product:** We invite you to register your product at www.mt.com/productregistration so we can contact you about enhancements, updates and important notifications concerning your product.
2. **Contact METTLER TOLEDO for service:** The value of a measurement is proportional to its accuracy – an out of specification scale can diminish quality, reduce profits and increase liability. Timely service from METTLER TOLEDO will ensure accuracy and optimize uptime and equipment life.
 - a. **Installation, Configuration, Integration and Training:** Our service representatives are factory-trained, weighing equipment experts. We make certain that your weighing equipment is ready for production in a cost effective and timely fashion and that personnel are trained for success.
 - b. **Initial Calibration Documentation:** The installation environment and application requirements are unique for every industrial scale so performance must be tested and certified. Our calibration services and certificates document accuracy to ensure production quality and provide a quality system record of performance.
 - c. **Periodic Calibration Maintenance:** A Calibration Service Agreement provides on-going confidence in your weighing process and documentation of compliance with requirements. We offer a variety of service plans that are scheduled to meet your needs and designed to fit your budget.
 - d. **GWP® Verification:** A risk-based approach for managing weighing equipment allows for control and improvement of the entire measuring process, which ensures reproducible product quality and minimizes process costs. GWP (Good Weighing Practice), the science-based standard for efficient life-cycle management of weighing equipment, gives clear answers about how to specify, calibrate and ensure accuracy of weighing equipment, independent of make or brand.
 - e. **InTouchSM Remote Services:** Confidently and securely improve the performance and uptime of your weighing systems, with InTouch Remote Service, only available from METTLER TOLEDO. Working within your existing IT security policies, IND570 with embedded InTouch connectivity actively monitors system performance. Proactive alerts to remote service technicians allow a real-time response to performance issues, increasing uptime, overall asset utilization and reduction of unforeseen expenses.

© METTLER TOLEDO 2024

No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of METTLER TOLEDO.

U.S. Government Restricted Rights: This documentation is furnished with Restricted Rights.

Copyright 2024 Mettler-Toledo, LLC. This documentation contains proprietary information of METTLER TOLEDO. It may not be copied in whole or in part without the express written consent of METTLER TOLEDO.

METTLER TOLEDO reserves the right to make refinements or changes to the product or manual without notice.

COPYRIGHT

METTLER TOLEDO® is a registered trademark of Mettler-Toledo, LLC. All other brand or product names are trademarks or registered trademarks of their respective companies.

METTLER TOLEDO RESERVES THE RIGHT TO MAKE REFINEMENTS OR CHANGES WITHOUT NOTICE.

FCC Notice

This device complies with Part 15 of the FCC Rules and the Radio Interference Requirements of the Canadian Department of Communications. Operation is subject to the following conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his or her expense.

- Declaration of Conformity may be found at <http://glo.mt.com/us/en/home/search/compliance.html/compliance/>.

Statement regarding harmful substances

We do not make direct use of harmful materials such as asbestos, radioactive substances or arsenic compounds. However, we purchase components from third party suppliers, which may contain some of these substances in very small quantities.

Manuals Download

Customers can click the link or scan the QR Code to download product manuals.

www.mt.com/IND570




















www.mt.com/IND570xx



Warnings and Cautions

- READ this manual BEFORE operating or servicing this equipment and FOLLOW these instructions carefully.
- SAVE this manual for future reference.

	<p style="text-align: center;"> WARNING</p> <p>FOR CONTINUED PROTECTION AGAINST SHOCK HAZARD CONNECT THE TERMINAL TO PROPERLY GROUNDED OUTLET ONLY. DO NOT REMOVE THE GROUND PRONG.</p>
	<p style="text-align: center;"> WARNING</p> <p>ONLY PERMIT QUALIFIED PERSONNEL TO SERVICE THE TERMINAL. EXERCISE CARE WHEN MAKING CHECKS, TESTS AND ADJUSTMENTS THAT MUST BE MADE WITH POWER ON. FAILING TO OBSERVE THESE PRECAUTIONS CAN RESULT IN BODILY HARM AND/OR PROPERTY DAMAGE.</p>
	<p style="text-align: center;"> WARNING</p> <p>DO NOT INSTALL, DISCONNECT OR PERFORM ANY SERVICE ON THIS EQUIPMENT BEFORE POWER HAS BEEN SWITCHED OFF AND THE AREA HAS BEEN SECURED AS NON-HAZARDOUS BY PERSONNEL AUTHORIZED TO DO SO BY THE RESPONSIBLE PERSON ON-SITE.</p>
	<p style="text-align: center;"> WARNING</p> <p>NOT ALL VERSIONS OF IND570 ARE DESIGNED FOR USE IN HAZARDOUS (EXPLOSIVE) AREAS. REFER TO THE DATA PLATE OF THE IND570 TO DETERMINE IF A SPECIFIC TERMINAL IS APPROVED FOR USE IN AN AREA CLASSIFIED AS HAZARDOUS BECAUSE OF COMBUSTIBLE OR EXPLOSIVE ATMOSPHERES. TERMINALS THAT ARE NOT FACTORY LABELED AS DIVISION 2 OR EUROPEAN CATEGORY 3 APPROVED MUST NOT BE INSTALLED IN A DIVISION 2 OR ZONE 2/22 ENVIRONMENT.</p>
	<p style="text-align: center;"> WARNING</p> <p>THE INTERNAL DISCRETE I/O <u>RELAY</u> OPTIONS MUST NOT BE USED IN AREAS CLASSIFIED AS HAZARDOUS BECAUSE OF COMBUSTIBLE OR EXPLOSIVE ATMOSPHERES. FAILURE TO COMPLY WITH THIS WARNING COULD RESULT IN BODILY HARM AND/OR PROPERTY DAMAGE.</p>
	<p style="text-align: center;"> WARNING</p> <p>WHEN THIS EQUIPMENT IS INCLUDED AS A COMPONENT PART OF A SYSTEM, THE RESULTING DESIGN MUST BE REVIEWED BY QUALIFIED PERSONNEL WHO ARE FAMILIAR WITH THE CONSTRUCTION AND OPERATION OF ALL COMPONENTS IN THE SYSTEM AND THE POTENTIAL HAZARDS INVOLVED. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY HARM AND/OR PROPERTY DAMAGE.</p>
	<p style="text-align: center;"> WARNING</p> <p>ONLY THE COMPONENTS SPECIFIED ON THE IND570 DOCUMENTATION CD CAN BE USED IN THIS TERMINAL. ALL EQUIPMENT MUST BE INSTALLED IN ACCORDANCE WITH THE INSTALLATION INSTRUCTIONS DETAILED IN THE INSTALLATION MANUAL. INCORRECT OR SUBSTITUTE COMPONENTS AND/OR DEVIATION FROM THESE INSTRUCTIONS CAN IMPAIR THE SAFETY OF THE TERMINAL AND COULD RESULT IN BODILY HARM AND/OR PROPERTY DAMAGE.</p>

	 CAUTION
	<p>BEFORE CONNECTING/DISCONNECTING ANY INTERNAL ELECTRONIC COMPONENTS OR INTERCONNECTING WIRING BETWEEN ELECTRONIC EQUIPMENT ALWAYS REMOVE POWER AND WAIT AT LEAST THIRTY (30) SECONDS BEFORE ANY CONNECTIONS OR DISCONNECTIONS ARE MADE. FAILURE TO OBSERVE THESE PRECAUTIONS COULD RESULT IN DAMAGE TO OR DESTRUCTION OF THE EQUIPMENT AND/OR BODILY HARM.</p>
	NOTICE
	<p>OBSERVE PRECAUTIONS FOR HANDLING ELECTROSTATIC SENSITIVE DEVICES.</p>

Disposal of Electrical and Electronic Equipment



In conformance with the European Directive 2012/19/EC on Waste Electrical and Electronic Equipment (WEEE) this device may not be disposed of in domestic waste. This also applies to countries outside the EU, per their specific requirements.

Please dispose of this product in accordance with local regulations at the collecting point specified for electrical and electronic equipment.

If you have any questions, please contact the responsible authority or the distributor from which you purchased this device.

Should this device be passed on to other parties (for private or professional use), the content of this regulation must also be related.

Thank you for your contribution to environmental protection.

Contents

1	Analog Output.....	1-1
1.1.	Specifications	1-1
1.2.	Analog Output Operation	1-2
1.3.	Installation	1-5
1.4.	Configuration.....	1-6
1.5.	Wiring	1-8
1.6.	Spare Parts	1-9
1.7.	Troubleshooting	1-9
2	ControlNet.....	2-1
2.1.	Preface	2-1
2.2.	ControlNet Interface Board.....	2-1
2.3.	Overview.....	2-2
2.4.	ControlNet Characteristics	2-2
2.5.	Data Definition.....	2-4
2.6.	Controlling the Discrete I/O Using a PLC Interface	2-7
2.7.	Hardware Setup	2-7
2.8.	Software Setup.....	2-8
2.9.	Troubleshooting	2-10
2.10.	Programming Examples	2-11
3	DeviceNet™	3-1
3.1.	Preface	3-1
3.2.	Overview.....	3-1
3.3.	DeviceNet Interface.....	3-1
3.4.	Data Definition.....	3-2
3.5.	Floating Point	3-4
3.6.	Controlling Discrete I/O Using a PLC Interface	3-4
3.7.	Hardware Setup	3-5
3.8.	Software Setup.....	3-6
3.9.	Troubleshooting	3-7
3.10.	DeviceNet Option Kit.....	3-8
3.11.	DeviceNet Commissioning and Configuration Examples.....	3-8
4	EtherNet/IP™	4-1
4.1.	Preface	4-1

4.2.	EtherNet/IP Interface Board	4-1
4.3.	Overview.....	4-1
4.4.	EtherNet/IP Characteristics.....	4-2
4.5.	Data Definition.....	4-4
4.6.	Shared Data Mode	4-7
4.7.	Controlling Discrete I/O Using a PLC Interface	4-7
4.8.	Software Setup.....	4-8
4.9.	Troubleshooting	4-9
4.10.	Programming Examples	4-11
5	Modbus TCP	5-1
5.1.	Modbus TCP Interface.....	5-1
5.2.	Overview.....	5-1
5.3.	Modbus TCP Characteristics	5-1
5.4.	Data Definition.....	5-2
5.5.	Controlling Discrete I/O Using a PLC Interface	5-5
5.6.	Software Setup.....	5-5
5.7.	Troubleshooting	5-7
5.8.	Modbus TCP Configuration Example.....	5-8
6	PROFIBUS	6-1
6.2.	Communications	6-2
6.3.	Data Definition.....	6-3
6.4.	Shared Data	6-3
6.5.	IND570 PROFIBUS I/O Mapping.....	6-5
6.6.	Controlling Discrete I/O Using a PLC Interface	6-7
6.7.	Hardware Setup	6-7
6.8.	Software Setup.....	6-8
6.9.	Troubleshooting	6-9
6.10.	Interfacing Examples	6-11
6.11.	Sample PLC Program	6-11
7	PROFINET	7-1
7.1.	Overview.....	7-1
7.2.	PROFINET Interface	7-1
7.3.	Data Definition.....	7-6
7.4.	Controlling the Discrete I/O Using a PLC Interface	7-8
7.5.	Shared Data Access	7-8

7.6.	Software Setup.....	7-9
7.7.	PROFINET GSDML File	7-10
7.8.	Assigning the IP Address and Device Name.....	7-12
7.9.	Troubleshooting	7-15
7.10.	Siemens S7-300 Programming Examples.....	7-17
8	Modbus RTU	8-1
8.1.	Overview.....	8-1
8.2.	Parameters	8-1
8.3.	Data Definition.....	8-2
A.	Integer and Division Formats	A-1
B.	Floating Point Format	B-1
B.1.	Operational Overview.....	B-1
B.2.	Floating Point Data Format and Compatibility	B-1
B.3.	Floating Point Data Format Definitions	B-2
B.4.	Floating Point Command Examples.....	B-10
C.	Common Data Features	C-1
C.1.	Data Formats	C-1
C.2.	Byte Order	C-2
C.3.	Controlling Discrete I/O Using a PLC Interface	C-2

1 Analog Output

The Analog Output option kit provides an isolated 4-20 mA or 0-10 VDC analog signal output for displayed weight, ABS-displayed weight (absolute displayed weight), gross weight, rate or ABS-rate. The analog output uses a 16-bit D/A converter for a very precise output.

The outputs are active, which means that no external power supply is required – nor is there any provision for using an external power supply in the circuit.

The Analog Output sub-block lets you select the data source for the analog signal and provides a method to calibrate the analog zero and high limit values. The IND570 terminal must be calibrated to the desired scale capacity before making Analog Output adjustments. If rate is to be used as the source for the analog output signal, it must be enabled in the **Scale > Rate** branch of setup. The Analog Output card provides one channel - it may be either current (4-20 mA) or voltage (0-10 VDC).

1.1. Specifications

Maximum Cable Length	0-10 VDC – 50 ft (15.2 m) 4-20mA – 1000 ft (300 m)
Min/Max Load Resistance	0-10 VDC – 100k ohms minimum 4-20 mA – 500 ohms maximum
Outputs	1 channel capable of supplying 4-20 mA or 0-10 VDC
Resolution	16 bit resolution - 65536 levels across entire range

- Note that if the load resistance ratings are exceeded, the analog output will not operate properly.

Figure 1-1 shows an Analog Output Option Board with its connector at bottom left.

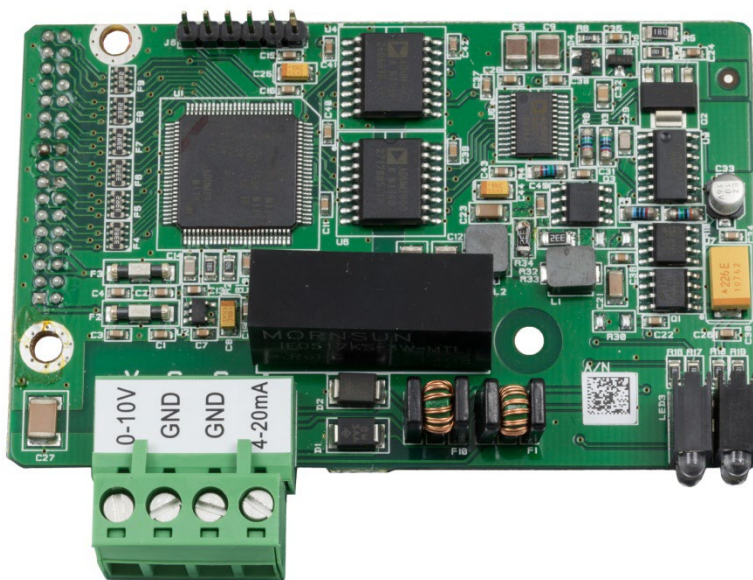


Figure 1-1: Analog Output Option Board

1.2. Analog Output Operation

When the source of the analog output is displayed weight, gross weight or rate, the output signals will be at the lower limit (0 VDC or 4 mA) when the value represented is at zero. When the value reaches its maximum limit, the output signal will increase to the high limit (10 VDC or 20 mA). Any value between zero and the maximum limit will be represented as a percentage of the output proportional to the percentage of the value.

The ABS (absolute)-displayed weight and ABS-rate are intended for use when material is being transferred off of a scale in Net mode. In these cases, the displayed weight and rate will show negative values, but the analog output signal will treat them as absolute values (disregarding their negative status). The output signals will increase as the interpreted absolute weight value or absolute rate value increases.

How the analog output functions under zero and over the high limit is determined by the selection for the source field selected – Displayed Weight, ABS – Displayed Weight, Gross Weight, Rate or ABS – Rate and the type of analog signal (4-20 mA or 10 VDC). Table 1-1 details how the analog output reacts under these conditions.

In order to use Rate as the source, it must be enabled in setup at **Scale > Rate**. For information on Rate configuration, refer to Chapter 3, **Configuration**.





Table 1-1: Analog Output Functions by Source

Source = Displayed Weight, Mode = 4-20 mA	
Under Zero	When the displayed weight (gross or net) drops below zero, the analog signal continues to decrease. When the under zero display blanking point is reached, or the analog signal negative range is exceeded, the analog output immediately switches to approximately 0 mA and remains there until the display is no longer blanked or the analog signal returns to within range.
Over High Limit	When the displayed weight (gross or net) exceeds the high limit, the analog signal continues to increase. When the display blanking point is reached, or the analog signal positive range is exceeded, the analog output immediately switches to approximately 24 mA and remains there until the weight display is no longer blanked or the analog signal returns to within range.
Source = ABS – Displayed Weight, Mode = 4-20 mA	
Under Zero	As the ABS-displayed weight increases in value, the analog output signal will climb towards the maximum output (20mA or 10VDC). When the true under zero blanking point is reached, the analog output immediately switches to approximately 0 mA and remains there until the display is no longer blanked.
Over High Limit	Only possible when weighing in gross mode with ABS-displayed weight as the source. Works the same as the standard “Displayed Weight” setting in this case.
Source = Gross Weight, Mode = 4-20 mA	
Under Zero	When the gross weight drops below zero, the analog signal continues to decrease. When the under zero display blanking point is reached, or the analog signal negative range is exceeded, the analog output immediately switches to approximately 0 mA and remains there until the display is no longer blanked or the analog signal returns to within range.
Over High Limit	When the gross weight exceeds the high limit, the analog signal continues to increase. When the display blanking point is reached, or the analog signal positive range is exceeded, the analog output immediately switches to approximately 24 mA and remains there until the weight display is no longer blanked or the analog signal returns to within range.
Source = Rate, Mode = 4-20 mA	
Under Zero	When the rate drops below zero, the analog output quickly switches to approximately 0 mA and remains there until the rate returns to within the programmed range. The jump to 0 mA will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal’s setup menu.
Over High Limit	When the rate exceeds the high limit, the analog output quickly switches to approximately 24 mA and remains there until the rate returns to within the programmed range. The jump to 24 mA will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal’s setup menu.

Source = ABS - Rate, Mode = 4-20 mA	
Under Zero	Not applicable. ABS-Rate recognizes negative rate values as valid.
Over High Limit	When the rate exceeds the analog output high limit, the analog output quickly switches to approximately 24 mA and remains there until the rate returns to within the programmed range. The jump to 24mA will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal's setup menu.
Source = Displayed Weight, Mode = 0-10 VDC	
Under Zero	When the displayed weight (gross or net) drops below zero, the analog signal continues to decrease. When the under zero display blanking point is reached, or the analog signal negative range is exceeded, the analog output immediately switches to approximately -2.4 VDC and remains there until the display is no longer blanked or the analog signal returns to within range.
Over High Limit	When the displayed weight (gross or net) exceeds the high limit, the analog signal continues to increase. When the display blanking point is reached, or the analog signal positive range is exceeded, the analog output immediately switches to approximately 12.5 VDC and remains there until the weight display is no longer blanked or the analog signal returns to within range.
Source = ABS – Displayed Weight, Mode = 0-10 VDC	
Under Zero	As the ABS-displayed weight increases in value, the analog output signal will climb towards the maximum output (20mA or 10VDC). When the true under zero blanking point is reached, the analog output immediately switches to approximately -2.4V and remains there until the display is no longer blanked.
Over High Limit	Only possible when weighing in gross mode with ABS-displayed weight as the source. Works the same as the standard "Displayed Weight" setting in this case.
Source = Gross Weight, Mode = 0-10 VDC	
Under Zero	When the gross weight drops below zero, the analog signal continues to decrease. When the under zero display blanking point is reached, or the analog signal negative range is exceeded, the analog output immediately switches to approximately -2.4 VDC and remains there until the display is no longer blanked or the analog signal returns to within range.
Over High Limit	When the gross weight exceeds the high limit, the analog signal continues to increase. When the display blanking point is reached, or the analog signal positive range is exceeded, the analog output immediately switches to approximately 12.5 VDC and remains there until the weight display is no longer blanked or the analog signal returns to within range.

Source = Rate, Mode = 0-10 VDC	
Under Zero	When the rate drops below zero, the analog output quickly switches to approximately -2.4 VDC and remains there until the rate returns to within the programmed range. The jump to -2.4 VDC will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal's setup menu.
Over High Limit	When the rate exceeds the high limit, the analog output quickly switches to approximately 12.5 VDC and remains there until the rate returns to within the programmed range. The jump to 12.5 VDC will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal's setup menu.
Source = ABS - Rate, Mode = 0-10 VDC	
Under Zero	Not applicable. ABS-Rate recognizes negative rate values as valid.
Over High Limit	When the rate exceeds the analog output high limit, the analog output quickly switches to approximately 12.5 VDC and remains there until the rate returns to within the programmed range. The jump to 12.5 VDC will occur only as quickly as the rate value is updated in the terminal. Rate update is based on Measurement Period and Output Average, selected in the terminal's setup menu.

1.3. Installation

	 WARNING
	DISCONNECT ALL POWER TO THIS UNIT BEFORE REMOVING THE FUSE OR SERVICING.
 WARNING	
DO NOT APPLY POWER TO THE IND570 TERMINAL UNTIL INSTALLATION OF COMPONENTS AND EXTERNAL WIRING HAS BEEN COMPLETED.	
	NOTICE
	OBSERVE PRECAUTIONS FOR HANDLING ELECTROSTATIC SENSITIVE DEVICES.

The analog output option for either enclosure type can be installed at the factory or it can be ordered as a kit and installed in the field. The option kit includes detailed drawings to assist in the installation.

The recommended wiring for the analog output is two-conductor, 20 GA cable available from METTLER TOLEDO (part number 510220190). It is equivalent to Belden #8762.

1.4. Configuration

Figure 1-2 illustrates the setup procedures for configuring the Analog Output option for the IND570 terminal.

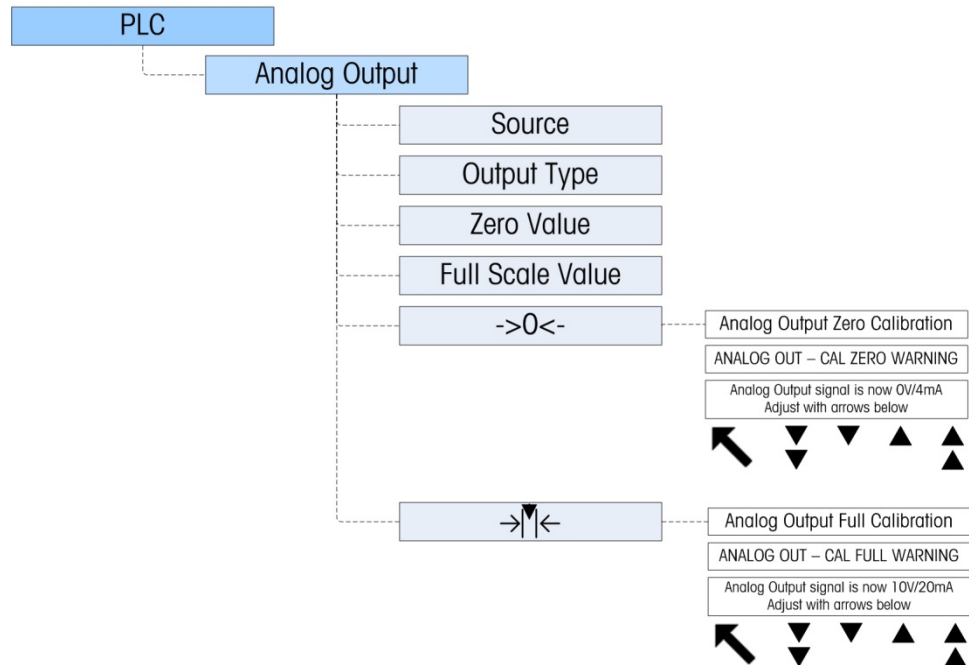


Figure 1-2: Setup Procedures for Configuring the Analog Output Option Card

1.4.1. Analog Output Setup Sub-Block





To configure the Analog Output Kit Option:


1. With power to the IND570 terminal removed, connect a volt or current meter to the appropriate output. If the customer's device is already connected, the meter is not necessary.
2. Apply power to the terminal and enter Setup. Navigate to PLC sub-block.
3. Select the **Analog Output** branch then select the source. Choices are **None**, **Displayed Weight** (the default), **ABS – Displayed Weight**, **Gross Weight**, **Rate** and **ABS – Rate**. **None** disables the analog output. **Displayed Weight** outputs an analog signal based on the displayed net or gross weight. When **Gross Weight** is selected, the analog signal is based on the gross weight regardless of what the net weight might be. In order to be available as a source, **Rate** must be configured at **Scale > Rate**.
4. Next, select the **Channel**. Options are **Scale** and **None**. Scale is the only option available now; None is reserved for future use.
5. At the **Zero Value** prompt, enter the desired source value for which the analog output should be zero. Typically this would be "0" in most applications; however, any valid value below the high limit can be used.
6. At the **Full Scale Value** prompt, enter the desired source value at which the analog output should be at its high limit. For sources of weight, this would typically be scale capacity, but it

could be lower. For rate, this should be the rate value that should provide a full analog output signal.

7. After all these parameters have been entered, the analog output can be adjusted to meet the customer's requirements using the ZERO softkey $\rightarrow 0 \leftarrow$ and the SPAN softkey $\rightarrow] \leftarrow$. To adjust the zero reference analog signal, press the ZERO softkey $\rightarrow 0 \leftarrow$.
8. Note that a display message is shown warning that during the adjustment, the analog output will be set to zero and will not monitor changes in the source value. Press the ESCAPE softkey **Esc** to exit the zero adjustment process or press the OK softkey **OK** to continue the adjustment process.
9. At the **Analog Output - Cal Zero** screen, use the softkeys to adjust the analog output signal to be exactly zero on the customer's device. The available softkeys are described in Table 1-2.

Table 1-2: Softkey Descriptions

	Coarse Down	This adjusts the analog signal level down in large steps.
	Fine Down	This adjusts the analog signal level down in small steps.
	Fine Up	This adjusts the analog signal level up in small steps.
	Coarse Up	This adjusts the analog signal level up in large steps.

10. When the zero adjustment is complete, press the EXIT softkey  to return to the Analog Output screen.
11. Now, the full scale analog output value can be adjusted by pressing the SPAN softkey $\rightarrow] \leftarrow$. A similar warning message will be shown indicating the analog output will be set to the high value and will not monitor changes in the source. Press the ESCAPE softkey **Esc** to exit the span adjustment process or press the OK softkey **OK** to continue the adjustment process.
12. At the **Analog Output - Cal Full** screen, use the softkeys to adjust the analog output signal to be exactly what the customer's device requires for its high limit. The available softkeys are described in Table 1-2.

1.4.1.1. Setting a Negative Value

It is sometimes necessary to set a negative value to define the lower end of the span. For example, the 4mA output might be set to correspond to a weight value of -20 kg.

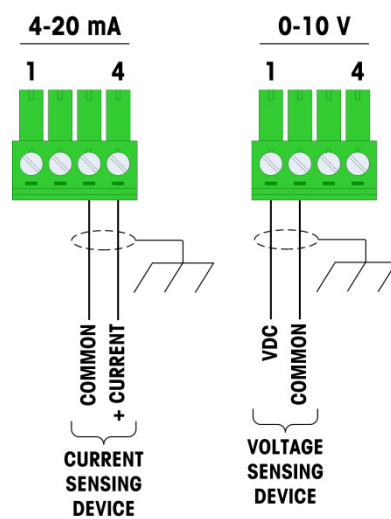
It is not possible to set a negative weight value directly from the IND570 operator interface. However, there are two ways of setting a negative value:

- Use an external QWERTY keyboard connected to the terminal's (optional) USB port to enter the negative value in setup on the terminal, in the Analog Output Zero Value field.
- Enter the negative value directly into Shared Data variable ao0103 (Analog Output Zero Preset).

1.5. Wiring

WARNING
<p>DO NOT APPLY POWER TO THE IND570 TERMINAL UNTIL INSTALLATION OF COMPONENTS AND EXTERNAL WIRING HAS BEEN COMPLETED.</p>
WARNING
<p>IF THIS DEVICE IS USED IN AN AUTOMATIC OR MANUAL FILLING CYCLE, ALL USERS MUST PROVIDE A HARD-WIRED EMERGENCY STOP CIRCUIT OUTSIDE THE DEVICE OF CIRCUITRY. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.</p>

The maximum recommended cable length for the 0-10 VDC output is 50 feet (15.2 meters). The maximum recommended cable length for the 4-20 mA output is 1,000 feet (300 meters). The recommended cable for use with the analog output is shielded two-conductor stranded 20-gauge cable (Belden #8762 or equivalent), which is available from METTLER TOLEDO using part number 510220190. See Figure 1-3 for connection and termination information.



NOTES:

- USE TWO-CONDUCTOR SHIELDED CABLE.**
- MINIMUM RESISTANCE OF DEVICE LOAD: 500 OHMS.**
- WIRE SIZE: 18 AWG (.823 mm²) MAXIMUM
24 AWG (0.205 mm²) MINIMUM.**

Figure 1-3: Analog Output Kit Wiring Connections

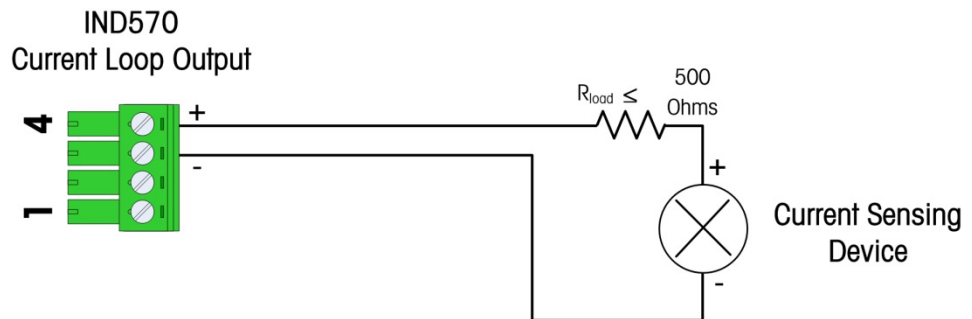


Figure 1-4: Typical Current Loop connection

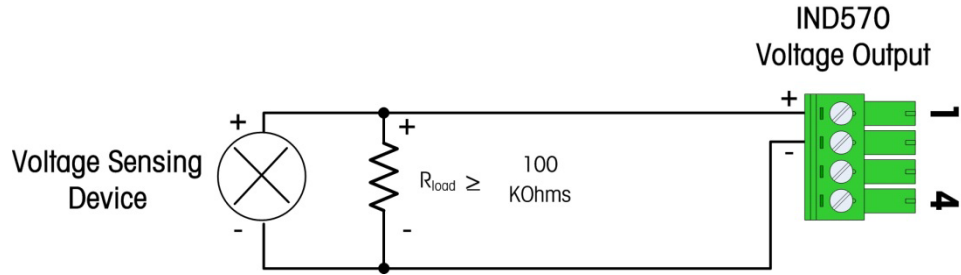


Figure 1-5: Typical Voltage Measurement Connection

1.6. Spare Parts

There are no associated spare parts with the Analog Output option kit. The kit number is 30113588. Table 1-3 shows what the kit contains.

Table 1-3: Analog Output Option Kit

Description	Qty.
Installation Instructions	1
PCB Package	1
Installation Kit	1
Gland Kit	1

1.7. Troubleshooting

Before attempting to troubleshoot the output, be sure to read the “Analog Output Operation” section of the manual – paying particular attention to Table 1-1, which explains the various operation modes for the output for the conditions of Under Zero and Over High Limit.

Check the LED Status as indicated in **Figure 1-6**. Use Table 1-4: to verify that the operational state of the Analog Output board matches the configuration selected.

1.7.1. No Signal

If no signal is being received from the IND570, check the following

- Confirm that a Source has been selected in the Analog Output configuration.
- For the 4-20mA configuration:
 - Confirm that the Output Type was set to “4-20mA” in the Analog Output Configuration.
 - Confirm proper wiring by referring to Figure 1-4.
 - Insert a current measuring device (DVM placed in mA measuring mode) in series with the circuit, vary the input to the IND570, confirm that the input is moving by observing the display, verify that the output changes as the input changes.

- For the Voltage Output configuration:
 - Confirm that the Output Type was set to “0-10V” in the Analog Output Configuration.
 - Confirm proper wiring by referring to Figure 1-5.
 - Place a voltage measuring device (DVM placed in DC Voltage measuring mode) in parallel with the circuit, vary the input to the IND570, confirm that the input is moving by observing the display, verify that the output changes as the input changes.
- Confirm that the Analog Card is properly inserted into the unit, and that all three anchor screws are installed.
- Make sure that the terminal is NOT in Setup mode.
- Cycle power on the IND570 and check the output again.
- If the PLC interface pcb was changed from another type, like DeviceNet or ControlNet, a master reset of the IND570 should be performed. Contact METTER TOLEDO service for assistance.
- Contact METTLER TOLEDO service for replacement Analog Output card.

1.7.2. Incorrect Signal

If an incorrect signal is being received from the IND570, check the following:

- For the 4-20mA output, confirm proper wiring by referring to Figure 1-4, and verify that the output load does not excel 500 Ohms.
- For the 0-10V output, confirm proper wiring by referring to Figure 1-5, and verify that the output load is 100 KOhms or greater.
- Refer to the Analog Output Calibration instructions in section 1.4.1.

1.7.3. Status LEDs

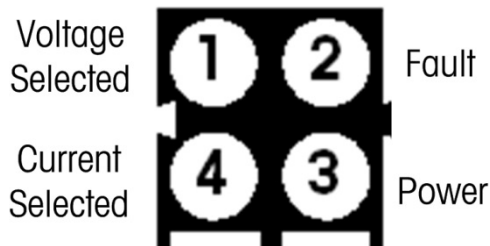


Figure 1-6: Status LEDs

Table 1-4: LED Status

LED #	State	Status
1 – Voltage Selected	Off	Not Selected
	Green	Voltage Output Mode Selected
2 - Fault	Off	No Fault
	Red	Current Mode Open Circuit detected. Over temperature Detected
3 - Power	Off	No Power on board

LED #	State	Status
	Green	Power On
4 – Current Selected	Off	Not Selected
	Green	Current Output Mode Selected

2 ControlNet

2.1. Preface

Users should note that the ControlNet option PCB used in the IND570 terminal is also used in the METTLER TOLEDO IND131, IND331 and IND780 terminals. There are minor differences in the Floating Point polled data between the terminals, so care should be taken to use the appropriate PLC data format guide for each terminal family.

This chapter describes connections and setup that are specific to the ControlNet option for IND570. The formats of the data that is transferred between the IND570 and the PLC are described in Appendix A and Appendix B.

2.2. ControlNet Interface Board

Due to space constraints, the ControlNet interface option can only be used with panel-mount versions of the IND570 terminals.

Figure 2-1 shows the ControlNet interface module and its components.

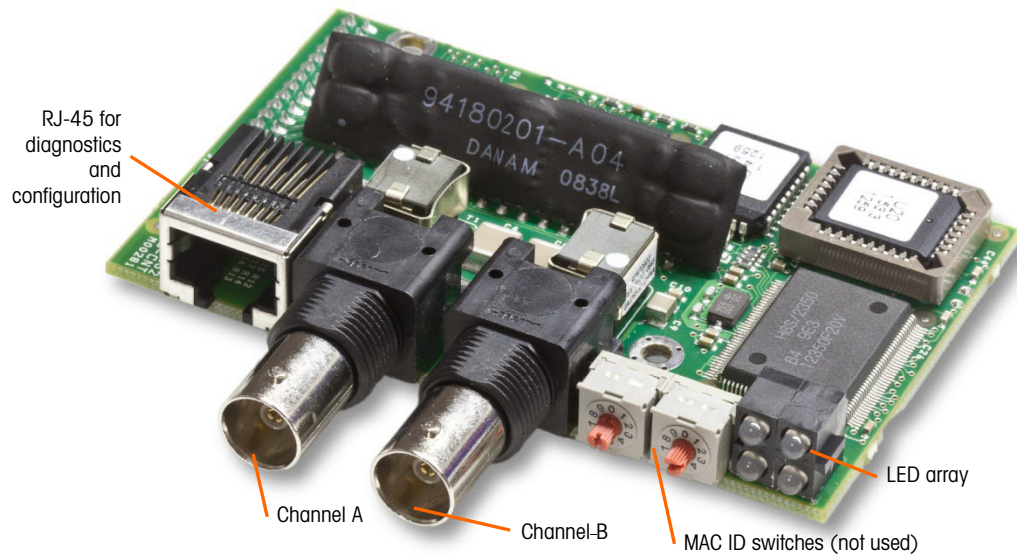


Figure 2-1: ControlNet PLC Module and its Components

- Do not plug an Ethernet cable into the RJ-45 connector shown at left in Figure 2-1. This connection is not used.

2.3. Overview

The ControlNet option enables the IND570 terminal to communicate to ControlNet Programmable Logic Controllers (PLCs) through direct connection to the ControlNet network.

2.4. ControlNet Characteristics

The ControlNet option has the following features:

- User-programmable node (MAC ID) address.
- Capability for bi-directional discrete mode communications (Class 1 Messaging) of weight or display increments, status, and control data between the PLC and the IND570.

2.4.1. Definition of Terms

Some terms (such as Target) used by the ControlNet PLC application have a different meaning from their use by the IND570 terminal. Table 2-1 offers definitions specific to ControlNet.

Table 2-1: ControlNet Definition of Terms

Term	Definition
Adapter Class	An Adapter Class product emulates functions provided by traditional rack-adapter products. This type of node exchanges real-time I/O data with a Scanner Class product. It does not initiate connections on its own.
Class 1 Messaging	In ControlNet communication protocol scheduled (cyclic) message transfer between a PLC and CIP Adapter Class device.
Class 3 Messaging	In ControlNet communication protocol unscheduled message transfer between a PLC and CIP Adapter Class device. This is used by the IND570 for explicit messaging.
Connected Messaging	A connection is a relationship between two or more application objects on different nodes. The connection establishes a virtual circuit between end points for transfer of data. Node resources are reserved in advance of data transfer and are dedicated and always available. Connected messaging reduces data handling of messages in the node. Connected messages can be Implicit or Explicit. See also Unconnected Messaging .
Connection Originator	Source for I/O connection or message requests. Initiates an I/O connection or explicit message connection.
Explicit Messaging	Explicit Messages can be sent as a connected or unconnected message. CIP defines an Explicit Messaging protocol that states the meaning of the message. This messaging protocol is contained in the message data. Explicit Messages are a one-time transport of a data item that provides the means by which typical request/response oriented functions are performed (e.g. module configuration). These messages are typically point-to-point. IND570

Term	Definition
Implicit Messaging	Implicit Messages are exchanged across I/O Connections with an associated Connection ID. The Connection ID defines the meaning of the data and establishes the regular/repeated transport rate and the transport class. No messaging protocol is contained within the message data as with Explicit Messaging. Implicit Messages can be point-to-point or multicast and are used to transmit application-specific I/O data. This term is used interchangeably with the term I/O Messaging .
I/O Client	Function that uses the I/O messaging services of another (I/O Server) device to perform a task. Initiates a request for an I/O message to the server module. The I/O Client is a Connection Originator.
I/O Messaging	Used interchangeably with the term Implicit Messaging .
I/O Server	Function that provides I/O messaging services to another (I/O Client) device. Responds to a request from the I/O Client. I/O Server is the target of the connection request.
Message Client	Function that uses the Explicit messaging services of another (Message Server) device to perform a task. It initiates an Explicit message request to the server device.IND570
Message Server	Function that provides Explicit messaging services to another (Message Client) device. It responds to an Explicit message request from the Message Client.IND570
Scanner Class	A Scanner Class product exchanges real-time I/O data with Adapter Class and Scanner Class products. This type of node can respond to connection requests and can also initiate connections on its own.
Target	Destination for I/O connection or message requests. Can only respond to a request, cannot initiate an I/O connection or message.
Unconnected Messaging	Provides a means for a node to send message requests without establishing a connection prior to data transfer. More overhead is contained within each message and the message is not guaranteed destination node resources. Unconnected Messaging is used for non-periodic requests (e.g. network "Who" function). Explicit messages only. See also Connected Messaging .IND570

2.4.2. Communications

The IND570 terminal uses component parts that ensure complete compatibility with the Allen-Bradley ControlNet network. An IND570 terminal is recognized as a generic ControlNet device by the PLC.

Each ControlNet option connected to the ControlNet network represents a physical node. The connection is made using BNC connectors on the option card.

The wiring between the PLC and the IND570 ControlNet connection uses RG-6 CATV cable and 75 ohm impedance matching transformer tap for each node. The cable is commonly referred to as coaxial cable. The cable installation procedures and specification including distance and termination requirements are the same as recommended by Allen-Bradley for the ControlNet network. The normal connection is to the channel A connector. The channel B connector is only used for redundant connection networks.

The IND570 terminal's communication update rate is set up by the use of Allen Bradley software **RSNetWorx** for ControlNet.

The IND570 uses Class 1 cyclic data for discrete data transfer and explicit messages for access to the IND570 Shared Data Variables. Explicit message blocks may be connected or unconnected; the PLC programmer must make this choice.

2.4.2.1. Node Address

Each ControlNet option represents one physical node. This address is chosen by the system designer, and then programmed into the IND570 terminal and PLC. The IND570 terminal's address is programmed in setup at **Communication > PLC Interface > ControlNet**. The IND570 address entry is in decimal.

2.4.3. Data Formats

The ControlNet option provides discrete data transfer, Class 1 messaging. Discrete data is continuously available. The ControlNet option has its own logical node address to send and receive information to and from the PLC. There are three data formats: Integer, Divisions, and Floating Point. Appendix A and B provide detailed information on data formats.

2.5. Data Definition

The ControlNet Kit option uses discrete data for its communication with PLCs. Data transfer is accomplished via the PLC's cyclic messaging.

2.5.1. Data Integrity

The IND570 has specific bits to allow the PLC to confirm that data was received without interrupt and the IND570 is not in an error condition. It is important to monitor these bits. Any PLC code should use them to confirm the integrity of the data received by the IND570. Refer to the data charts for specific information regarding the Data OK, Update in Progress, Data Integrity bits and their usage.

2.5.2. Assembly Instances of Class 1 Cyclic Communications

Class 1 cyclic communications is used for transfer of Discrete Data between the PLC and the IND570.

The PLC Input Assembly Instance is 100 (decimal). This instance is used for all Data Formats and data size requirements.

The PLC Output Assembly Instance is 150 (decimal). This instance is used for all Data Formats and data size requirements.

The IND570 uses data only; no configuration data is used or required. Within the PLC ControlNet Interface setup set the Configuration Instance to 1 and the data size to zero.

The EDS file provided on the Documentation CD has no Assembly Instance or data size limitations. The IND570 programming controls the Assembly Instance and data size limitations.

- **NOTE:** Version 20 and higher of ControlLogix has a feature that allows loading of an EDS file for use as a communications module in the PLC program itself. The IND570's EDS file is not designed for this purpose. The programmer should select the Generic Communication Modules instead, and only use the EDS file for programs such as RSLinx and RSNetWorx for ControlNet.

2.5.3. Data Formats

For a general account of Data Format types, please refer to Appendix C, **Common Data Features**.

Changing the Data Format to be used by the IND570 will clear all Message Slots. Data format is selected in the **Communication > PLC > Data Format** setup block – see Figure 2-4.

2.5.4. Byte Order

For a general account of Byte Order, please refer to Appendix C, **Common Data Features**.

2.5.5. Message Slots

There may be up to 4 message slots for discrete data transfer, Class 1 messaging, in Integer, Divisions and Floating Point Data Formats. Each message slot represents the scale but may be controlled by the PLC to present different data in each message slot. The number of Message Slots is selected in the terminal's setup menu at **Communication > PLC > Data Format**.

The integer and division formats provide two 16-bit words of input and two 16-bit words of output data per Slot. Each Message Slot's first input word provides scale weight data. The type of data displayed, such as Gross, Tare, etc., is selected by the PLC using the Message Slot's second output word bits 0, bit 1 and bit 2. Table 2-2 and Table 2-3 provide input and output usage information.

Table 2-2: ControlNet PLC Integer and Division I/O Data

Input Data to PLC				Output Data from PLC			
Word Offset	Description		Input Size	Output Size	Description	Word Offset	
0	4 Bytes Reserved		4 Words (8 Bytes)	2 Words (4 Bytes)	Msg Slot 1 Load Integer Value	0	
1						Command	1
2	Integer Value	Msg Slot 1	4 Words (8 Bytes)	4 Words (8 Bytes)	Msg Slot 2 Load Integer Value	2	
3	Scale Status					Command	3
4	Integer Value	Msg Slot 2	6 Words (12 Bytes)	6 Words (12 Bytes)	Msg Slot 3 Load Integer Value	4	
5	Scale Status					Command	5
6	Integer Value	Msg Slot 3	8 Words (16 Bytes)	8 Words (16 Bytes)	Msg Slot 4 Load Integer Value	6	
7	Scale Status					Command	7
8	Integer Value	Msg Slot 4	10 Words (20 Bytes)				
9	Scale Status						

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	4	2	8	4

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
2	6	4	12	8
3	8	6	16	12
4	10	8	20	16

The floating point format provides four 16-bit words of input data and three 16-bit words of output data) per Message Slot. Refer to Table 2-3 for details.

Table 2-3: ControlNet PLC Floating Point I/O Data

Input Data to PLC			Output Data from PLC		
Word Offset	Description	Input Size	Output Size	Description	Word Offset
0	4 Bytes Reserved	4 Words (8 Bytes)	4 Words (8 Bytes)	Reserved	0
1					1
2	Command Response			4-Byte Floating Point Load Value	2
3	4-Byte Floating Point Value				3
4			4		
5	Scale Status		7 Words (14 Bytes)	Command	4
6	Command Response				5
7	4-Byte Floating Point Value			4-Byte Floating Point Load Value	6
8			7		
9	Scale Status		10 Words (20 Bytes)	Command	7
10	Command Response				8
11	4-Byte Floating Point Value			4-Byte Floating Point Load Value	9
12			10		
13	Scale Status		13 Words (26 Bytes)	Command	10
14	Command Response				11
15	4-Byte Floating Point Value			4-Byte Floating Point Load Value	12
16			11		
17	Scale Status	18 Words (36 Bytes)			

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	4	4	8	8
2	8	7	16	14
3	12	10	24	20
4	16	13	32	26

2.5.6. Floating Point

For a general account of Floating Point operation, data format and compatibility, please refer to Appendix B, **Floating Point Format**.

2.5.7. Shared Data Mode

The Shared Data mode PLC communications is provided using CIP explicit messages.

The IND570 Shared Data Reference manual lists the Shared Data Variables available to ControlNet. This document also includes the hex Class Code, Instance and Attribute for the shared data. The PLC must use Get Attribute Single (0e hex) to read a Shared Data Variable and Set Attribute Single (10 hex) to write a Shared Data Variable.

The **IND570 Shared Data Reference** manual is available on the terminal's documentation CD.

2.6. Controlling the Discrete I/O Using a PLC Interface

The IND570 terminal provides the ability to directly control its discrete outputs and read its discrete inputs via the (digital) PLC interface options. System integrators should be aware that the IND570 terminal's discrete I/O updates are synchronized with the terminal's interface update rate and not with the PLC I/O scan rate. This may cause a noticeable delay in reading inputs or updating outputs as observed from the PLC to real world signals.

Consult the **IND570 Terminal Installation** for discrete I/O wiring. Also note that the outputs must be unassigned in the IND570 terminal at **Setup > Application > Discrete I/O** in order to be controlled by the PLC.

2.7. Hardware Setup

2.7.1. Wiring

The ControlNet PLC Module connects to the ControlNet network via a tap and drop cable from the original trunk cable. The option supports one BNC coaxial connection or two (for redundancy).

Channel A is the normal connection and Channel B (redundant with Channel A) can be used if ControlNet detects no signal on Channel A. Refer to Figure 2-2.

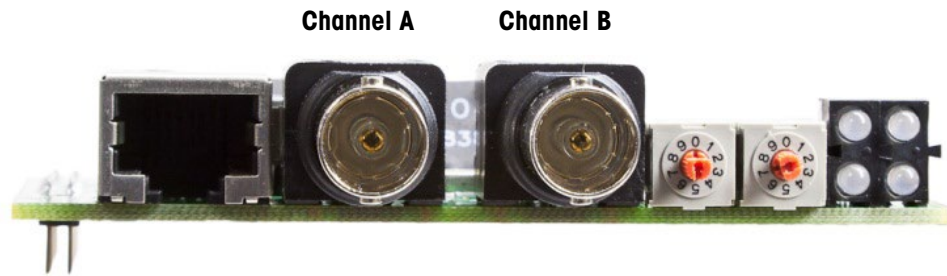


Figure 2-2: Coaxial Connector for ControlNet

Figure 2-3 shows examples of two different ControlNet tap and drop cables. Note that the connector to the ControlNet option may be straight or right-angled, as seen here. The IND570 panel-mount enclosures can use either type of drop cables. This drop cable is not supplied by METTLER TOLEDO.

Cable distance, type, and termination are specified by Allen-Bradley. Refer to Allen-Bradley supplied documentation for cable design guidelines for the various PLCs.

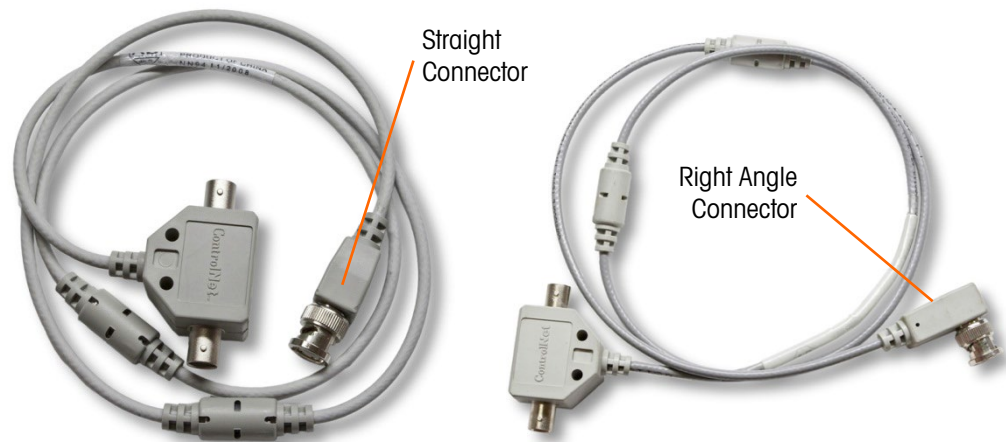


Figure 2-3: ControlNet Tap and Drop Cables

2.8. Software Setup

The IND570 terminal automatically detects the presence of a ControlNet option board if one is installed. When detected, the IND570 terminal adds the ControlNet parameters in a program block under **Communication > PLC**. Figure 2-4 shows the ControlNet program block.

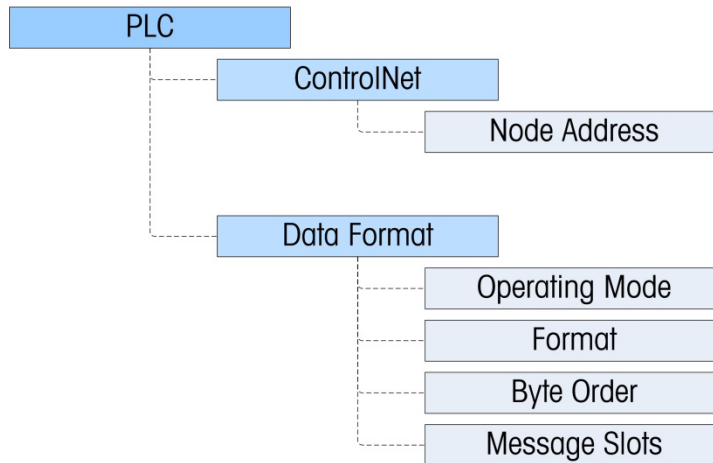


Figure 2-4: Setup Menu for ControlNet

2.8.1. ControlNet and Data Format Setup Blocks

This block lets you specify how the ControlNet interface is used. Several options are available to correspond with your system setup.

2.8.1.1. ControlNet Setup

2.8.1.1.1. Node Address

The ControlNet setup block at **Communication > PLC Interface > ControlNet** allows the user to set the node address. Each IND570 terminal connected to the network represents one physical node. This node address is determined by the system designer, then configured in the terminal by entering the appropriate number, **0** to **99** (default), in the Node Address field.

2.8.1.2. Data Format Setup

In Setup, navigate to **Communication > PLC Interface > Data Format**. The following fields are available for ControlNet.

2.8.1.2.1. Operating Mode

Operating Mode may be selected from a drop-down list. Choices are:

Compatibility Mode [default], IND560 Emulation

Depending on the Byte Order selection (refer to section 3.8.9.6.3, **Byte Order**, below), **Compatibility Mode** will provide the same discrete mode byte order arrangements as the METTLER TOLEDO IND131/331 and IND780 terminals. If **IND560 Emulation** is selected, the transmitted bytes in discrete mode will match the existing IND560 byte order determined by the Byte Order selection. Byte order arrangement in the IND560 terminals does not match that of IND131/331 and IND780. IND560 Emulation mode should be chosen only when replacing an IND560 **and** the programming within the PLC will not be modified.

2.8.1.2.2. Format

Select the Format: Integer (default), Divisions or Floating Point. Changing the Format will delete any existing Message Slots.

2.8.1.2.3.

Byte Order

Available selections are Standard, Byte Swap, Word Swap (default), and Double Word Swap. See Table 2-2 for definitions.

2.8.1.2.4.

Message Slots

Select 1, 2, 3 or 4 slots.

2.9. Troubleshooting

If the IND570 does not communicate with PLC, do the following:

- Check wiring and network termination.
- Confirm that the IND570 settings for data type, **I/O size**, and node assignment match those in the PLC and that each IND570 has a unique node assignment.
- Confirm that the EDS file has been loaded into the Network Configuration Tool (RSNetWorx for ControlNet on Allen-Bradley/Rockwell systems), the nodes added to the network configuration, and the configuration downloaded to the keeper module.
- Confirm that the updated network configuration has been optimized and scheduled.
- If the PLC interface pcb was changed from another type, like EtherNet/IP or DeviceNet, a master reset of the IND570 should be performed. Contact Mettler Toledo service for assistance.
- Contact METTLER TOLEDO service for replacement of the ControlNet interface.

2.9.1.

Status LEDs

The ControlNet option board has a four LED array that indicates the state of the communication. Figure 2-5 shows the array of the status indicator LEDs with each LED labeled.

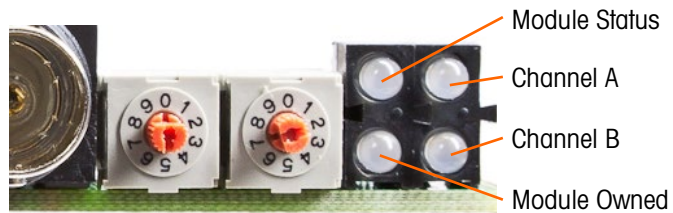


Figure 2-5: ControlNet Status Indicator LEDs

Table 2-4 describes the different conditions of the LEDs.

Table 2-4: ControlNet Status Indications

LED	LED State	Description
Module Status	Green	Connection in Run state
	Green, flashing	Connecting Connection Idle
	Red	Major fault
	Red, flashing	Minor fault

LED	LED State	Description
Channel A <i>And</i> Channel B	Off	Module not initialized
	Red	Major fault
	Alternating red/green	Self test
	Red, flashing	Node configuration error; duplicate MAC ID, etc.
Channel A <i>or</i> Channel B	Off	Channel disabled
	Green	Normal operation of channel
	Green, flashing	Temporary error (node will self-correct) or not configured
	Red, flashing	No other nodes, or media fault
Module Owned	Off	No connection has been opened
	Green	A connection has been opened towards the module

2.10. Programming Examples

The following Figures show screen images of ladder logic programming examples for RSLogix 5000 software (version 20).

The sample program demonstrates logic to interface to an IND570 set up for Floating Point Data or Integer Data Formats. The logic also includes routines that access Shared Data over the ControlNet interface in both Floating Point and Integer Data Formats.

This sample program is subject to change without notice. Please visit www.mt.com to download the most recent version of PLC sample code.

- **Note:** Complete versions of the examples are available for download at www.mt.com/IND570. These screen images are provided for illustrative purposes only.

Figure 2-6 shows the configuration of the ControlNet Scanner Module (1756-CNB/D)

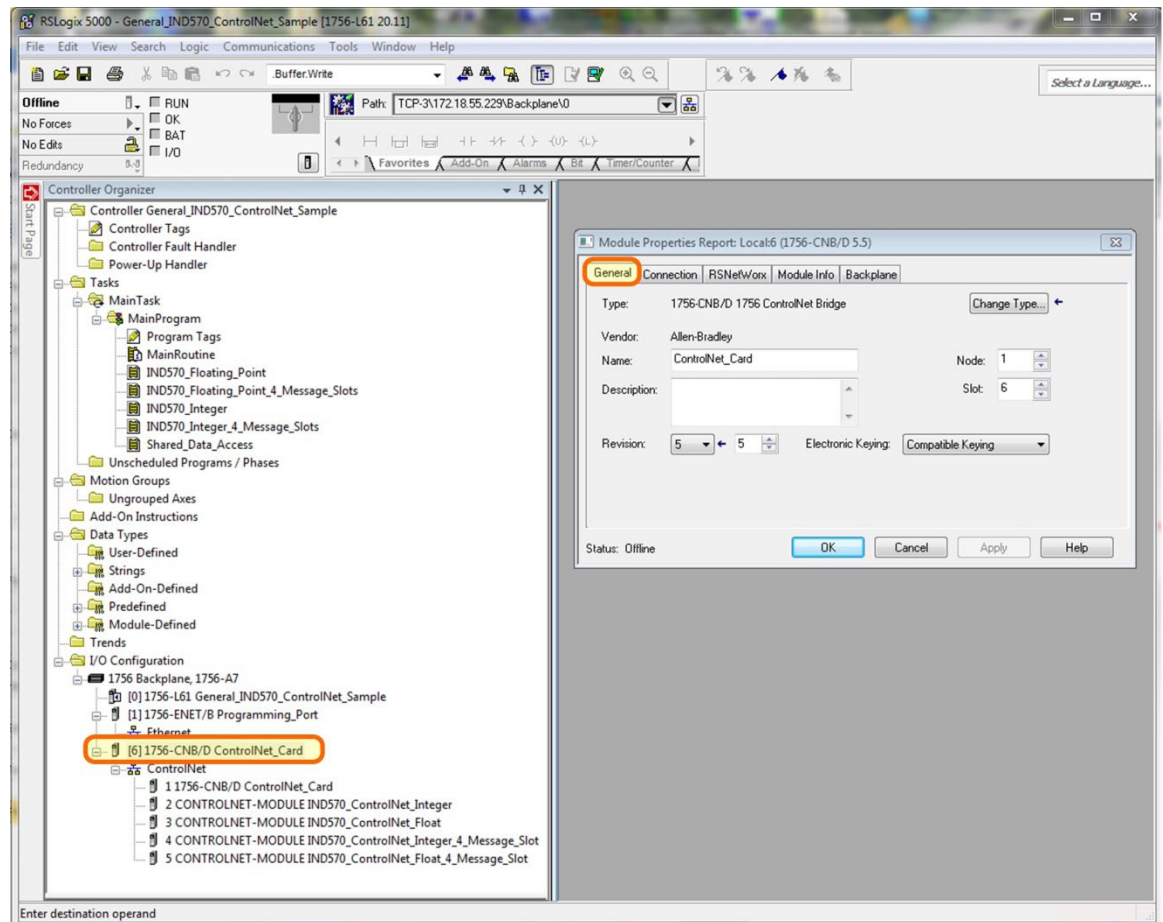


Figure 2-6: 1756 CNB Setup

At this time, an Add On Profile does not exist for the IND570's ControlNet Module. To define an IND570 on the PLC's ControlNet configuration, use the Generic ControlNet Module, selectable from the "Select Module Type" form's "Catalog" tab.

The sample program includes the following example Module definitions for 1 and 4 message slot Terminals using the Integer (or Division) Format.

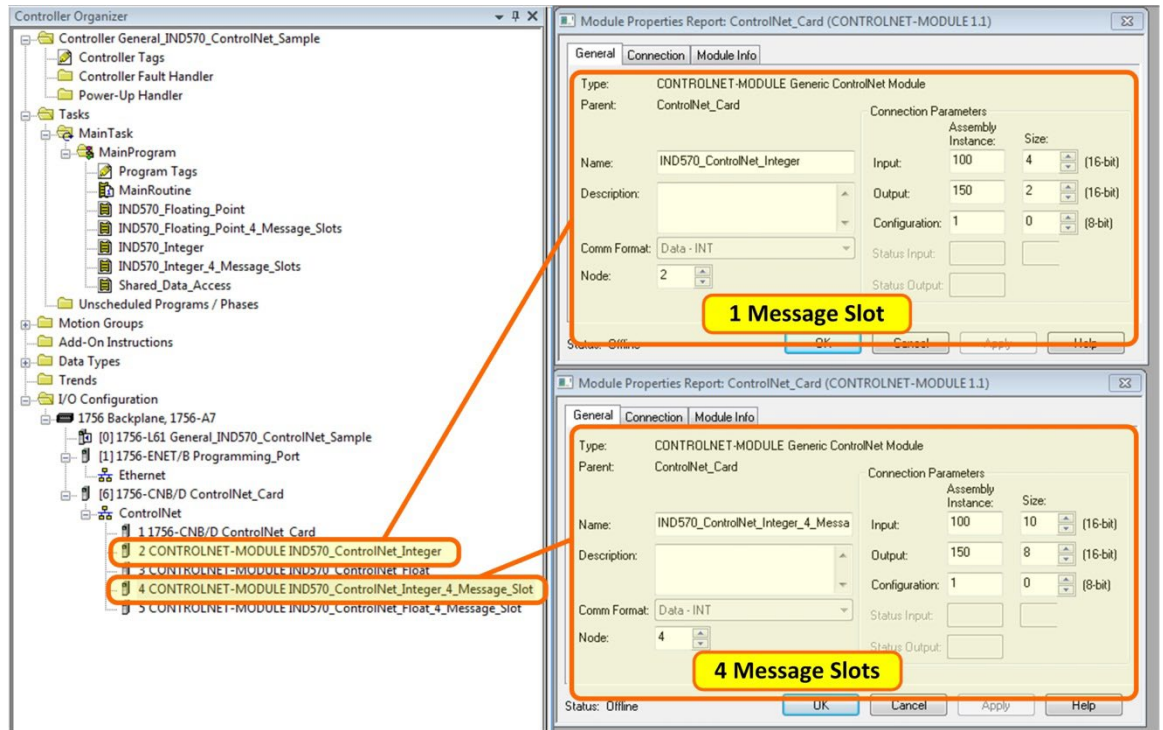


Figure 2-7: Module Definitions for 1 and 4 Message Slot Terminals, using Integer or Division Format

The sample program also includes the following example Module definitions for 1 and 4 message slot Terminals using the Floating Point Format.

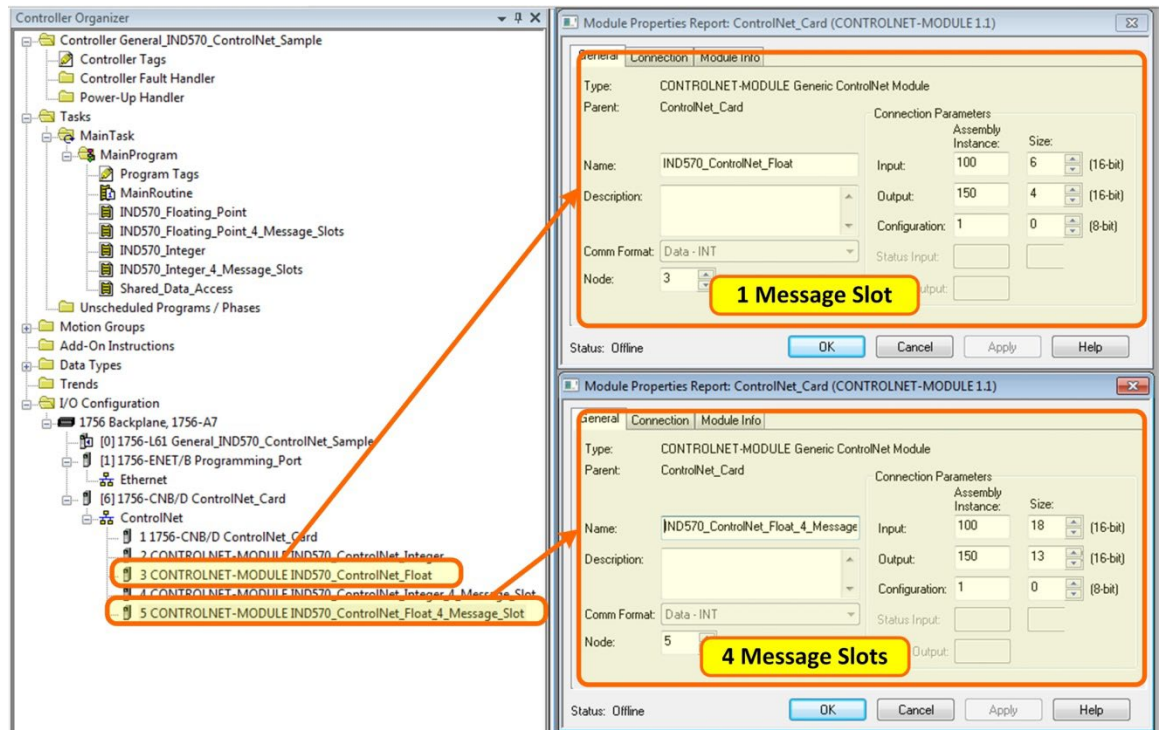


Figure 2-8: Module Definitions for 1 and 4 Message Slot Terminals, using Floating Point Format

The sample program also includes a RSNetwork configuration file (for ControlNet) named IND570.xc, that has all 4 nodes defined, as shown in Figure 2-9.

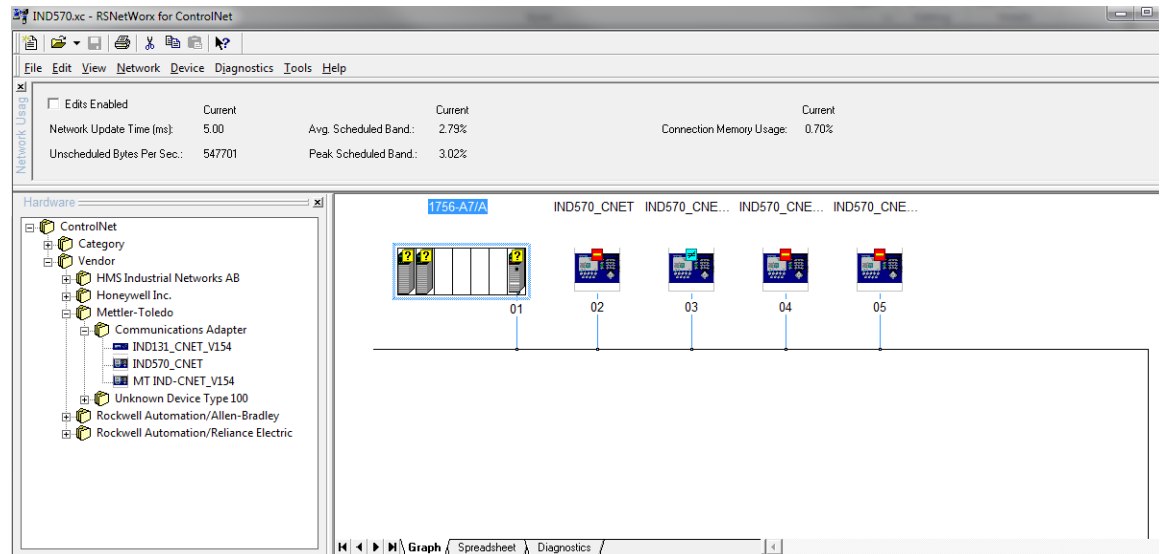


Figure 2-9: RSNetwork Configuration File with 4 Nodes Defined

This configuration must be downloaded and scheduled to the ControlNet scanner module for the sample program to work correctly.

The following RSLogix 5000 screens for Integer and Floating Point data formats only show an example of a particular Input and Output size configuration. The Connection Parameters I/O sizes must be appropriately configured with reference to the number of slots assigned in the IND570 PLC Data Format Message Slots settings. Table 2-5 and Table 2-6 show the relationship between the IND570 message slots and the RSLogix 5000 I/O sizing for Integer, Division and Floating Point data formats.

Table 2-5: Message Slot and PLC I/O Sizes (Integer/ Division)

IND570 Integer/ Division Data			RSLogix 5000 Comm Format	
Message Slots	Bytes (8 Bit)		INT (16 Bit)	
	IND570 >> PLC Input	PLC Output >> IND570	Input	Output
1	8	4	4	2
2	12	8	6	4
3	16	12	8	6
4	20	16	10	8

Table 2-6: Message Slot and PLC I/O Sizes (Floating Point)

IND570 Floating Point Data			RSLogix 5000 Comm Format	
Message Slots	Bytes (8 Bit)		INT (16 Bit)	
	IND570 >> PLC Input	PLC Output >> IND570	Input	Output
1	12	8	6	4
2	20	14	10	7
3	28	20	14	10
4	36	26	18	13

2.10.1. General Programming notes

The following principles should always be applied to guarantee the validity of the data before using it in a process. Note that there are different principles for the different modes (Floating Point versus Integer or Divisions).

For Floating Point Mode, data being read from the Terminal should be filtered with the Data_OK bit and the two Data Integrity bits as shown in Figure 2-10.

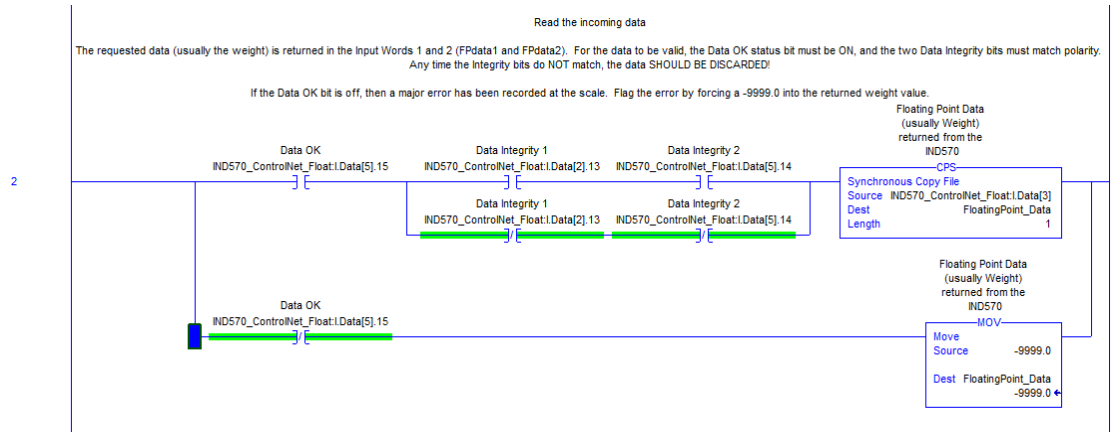


Figure 2-10: Filtering with Data_OK Bit and Two Data Integrity Bits, Floating Point Mode

Filtering the data in this way makes sure that the Terminal is in a valid operational state (Data_OK = 1) and that the Analog Update from the Load Cell has properly completed before the data was read (Integrity_1 = Integrity_2). Failing to perform these checks can result in invalid data being used by the PLC program.

For Integer or Division Mode, a similar filter should be applied as shown in Figure 2-11.

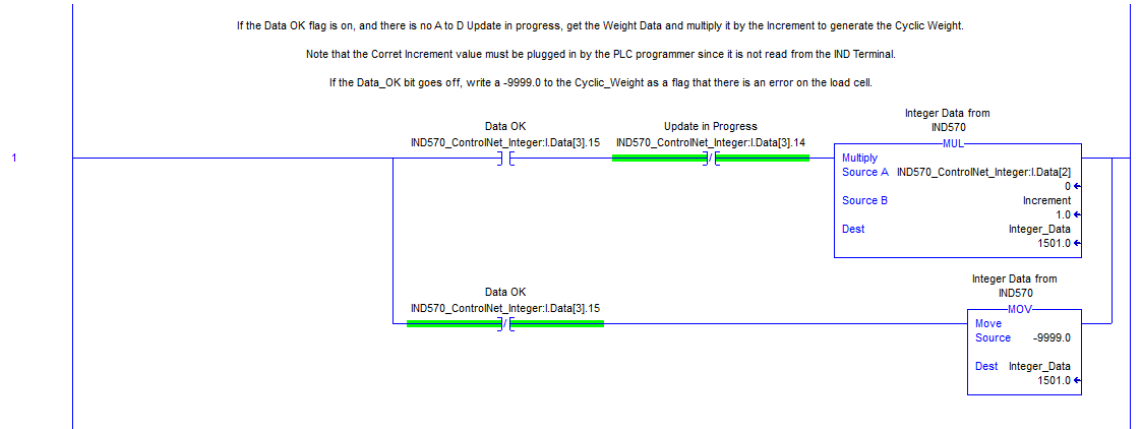


Figure 2-11: Filtering with Data_OK Bit and Two Data Integrity Bits, Integer or Division Mode

In this case, the data is filtered with the Data_OK bit and the Update_In_Progress bit to make sure that the data coming back from the terminal is valid. From there, it is converted into a Floating Point value by multiplying it by the hard-coded Increment size for the terminal to properly place the decimal point.

2.10.2. Shared Data Access Overview

Shared Data is a memory area in the terminal that contains many different kinds of information ranging from standard weight data to system variables and Task Expert application data. Providing access to this information to the PLC can be an immense help when coordinating the process with functions occurring in the Terminal.

For ControlNet, access to Shared Data is accomplished using discrete messages (otherwise known as explicit, asynchronous, or Class 3 messages).

In order to access Shared Data, a program must provide the following information to the Read and Write message instructions:

- Class Code
- Instance Number
- Attribute Number
- Length

This information can be found in the **Shared Data Reference Manual** (part number 30205337) for each Shared Data variable. For example, here is how you would find that information for a 'WT' type Shared Data variable:

Scale Functionality
Dynamic Scale Weight (WT)

Access: "Read Only." Access level is not customizable.
Class Code: wt Data Type: D

ControlNet Class Code: 68 hex

Instances: 5
Instance 1 - 4 = Scale platforms 1 - 4
Instance 5 = Sum scale.

Attributes:
Note: The last two digits of each shared data variable is its attribute.

wt-00	Composite wt block	Struct	na	Composite of entire block
wt-01	Displayed Gross Weight	S13	rt	
wt-02	Displayed Net Weight	S13	rt	When user has enabled MinWeigh, the first character contains an "*" when the MinWeigh conditions are not met.
wt-03	Weight Units	S4	rt	lb pounds, kg kilograms, grams, oz ounces, oztroy, dwf pennyweights, metric tons, ton, or custom units name
wt-04	Displayed Aux Gross Weight	S13	rt	
wt-05	Displayed Aux Net Weight	S13	rt	
wt-06	Aux Weight Units	S7	rt	lb pounds, kg kilograms, grams, oz ounces, lb-oz pounds & ounces, oztroy, ounces, dwf pennyweights, metric tons, ton, or custom units name
wt-07	Rate Period	S2	rt	No, Sec, Min, Hour
wt-08	Displayed Rate	S13	rt	

Figure 2-12: Shared Data Class, Instance, Attribute and Length

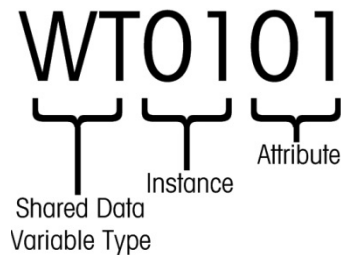


Figure 2-13: Construction of a Shared Data Variable Name

This information can help you in setting up your program to read or write the Shared Data variables that you need to access.

2.10.2.1. Shared Data Variable Name Instance Number

The 'Instance' is used in other METTLER TOLEDO terminals to refer to multiple instruments (scales or flowmeters) that may be serviced by the terminal. In the case of the IND570, there will only be one instrument (one scale), so most of the time the Instance number will be "01" when used in the Shared Data Variable name. There are exceptions to this rule, so attention must be paid to the details of the variable spelled out in the **Shared Data Reference**.

2.10.3. Shared Data Access Program Details

Since the type of data sent to, and read back from the IND570 is **not** defined by the communications mode selected (Integer, Divisions, or Floating Point), the method to access Shared Data in the IND570 Terminal is identical for both the Floating Point and Integer Modes using ControlNet.

Figure 2-14 shows a rung of logic that sends a trigger to the IND570 to Tare the scale. The configuration of the message instruction is shown below the rung.

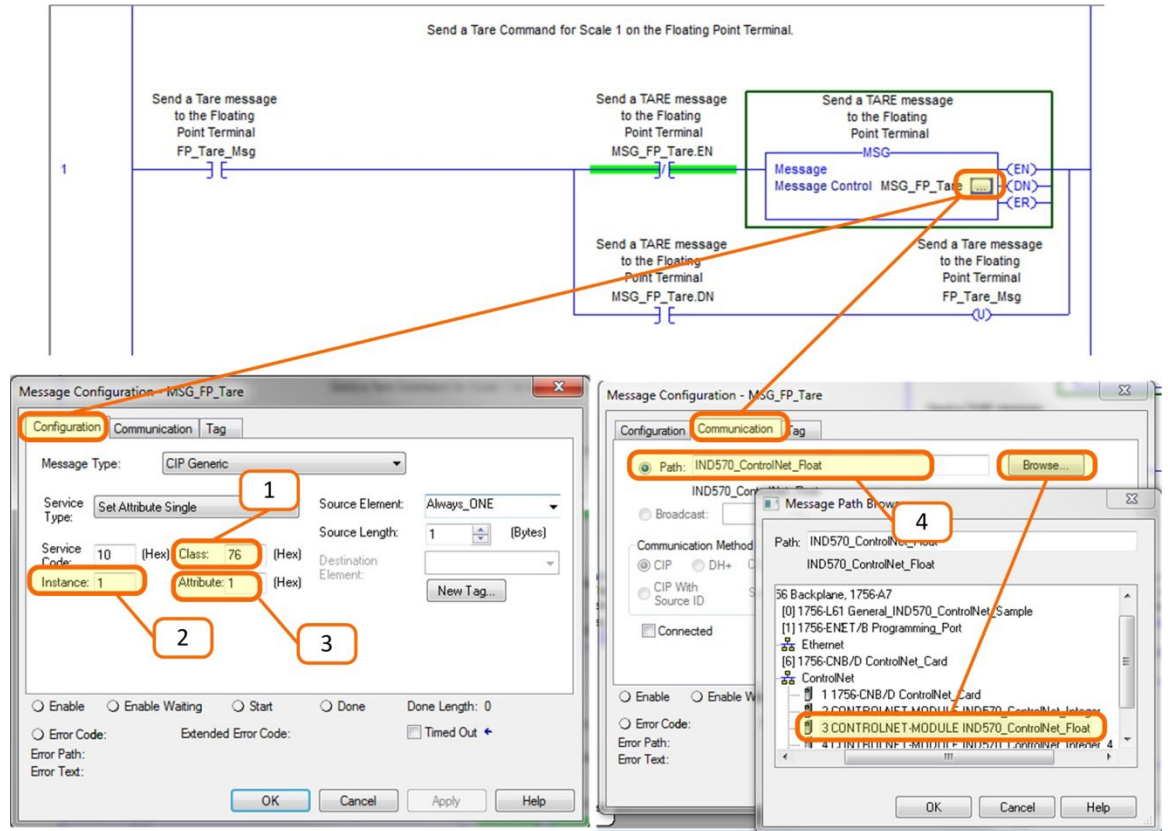


Figure 2-14: Ladder Logic – Taring the Scale

Note that the message type is a CIP Generic, with a Service Type “Set Attribute Single.”

Notes for Figure 2-14

1. Class Code in Hexadecimal for the Shared Data Variable WC0101, found in the Shared Data Reference Manual.
2. Instance number for Shared Data Variable WC0101, found in the Shared Data Reference Manual.
3. Attribute in Hexadecimal for the Shared Data Variable WC0101, found in the Shared Data Reference Manual.
4. The Path to the ControlNet Node that the message will be sent. The path can be selected by clicking the Browse button and selecting it from the list.

Other commands such as Clear, Zero, and Print would be sent in an identical way.

Figure 2-15 shows a rung of logic that triggers a read of the rounded gross weight on the terminal, which maps to Shared Data Variable WT0110. The configuration of the message instruction, along with the data area where the response will be stored, is shown below the rung.

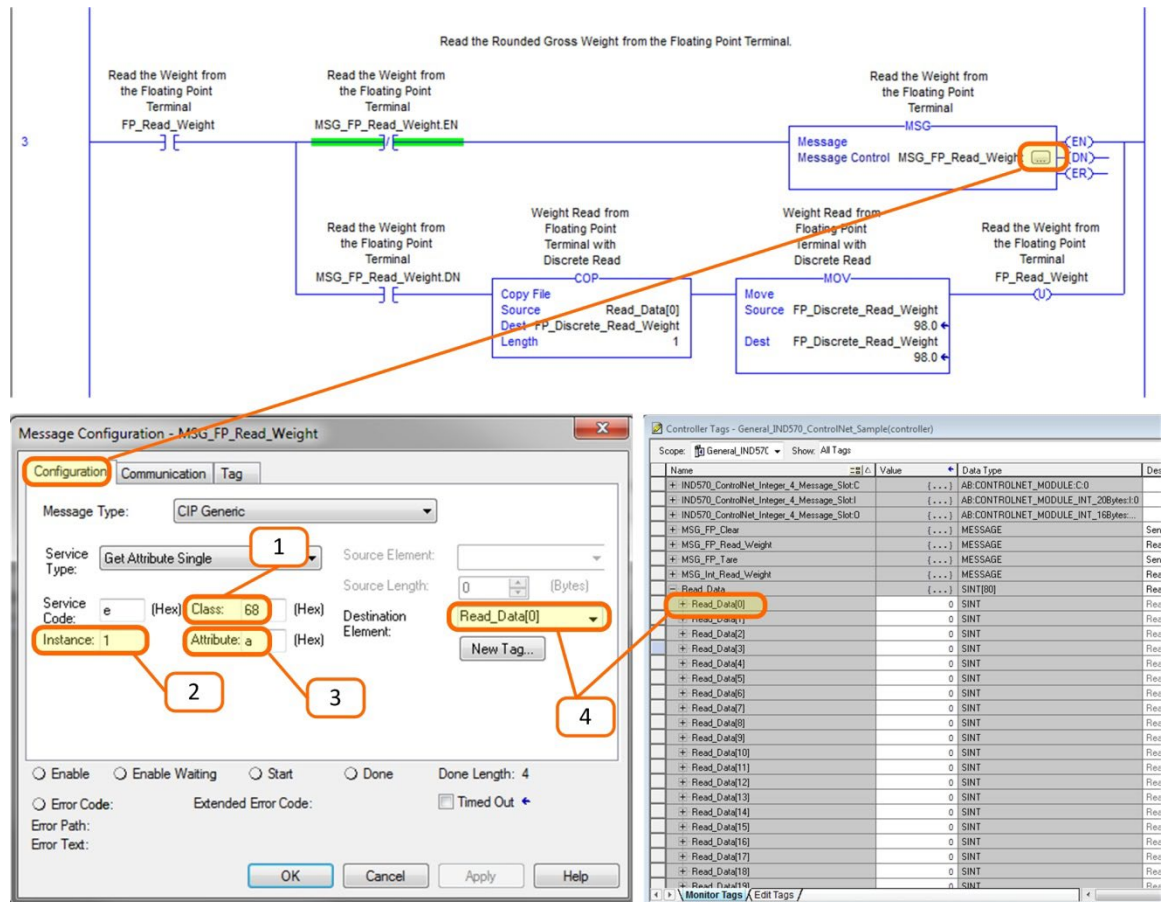


Figure 2-15: Ladder Logic – Reading Gross Weight and Saving to Shared Data

The message type is a CIP Generic, with a Service Type “Get Attribute Single.”

Notes for Figure 2-15

1. Class Code in Hexadecimal for the Shared Data Variable WT0110, found in the Shared Data Reference Manual.
2. Instance number for Shared Data Variable WT0110, found in the Shared Data Reference Manual.
3. Attribute in Hexadecimal for the Shared Data Variable WT0110, found in the Shared Data Reference Manual.
4. The Variable Tag to be used to store the data returned from the IND570. Note that the destination element must reference the array index [0] in order to correctly place the data in its destination.

The IND570 returns 4 bytes of data into the Read_Data array, which represents an IEEE 754 type Single Floating Point number. The program then converts those 4 bytes into a ‘REAL’ type number

by copying them into the tag `FP_Discrete_Read_Weight`. Note that the `MOV` instruction immediately after the copy is for convenience only so that the programmer can easily see the value that was returned. Note that the message instruction could have easily returned the value directly into the `REAL` typed variable instead of the `Byte Array`. Returning the data into the `Byte Array` gives the programmer some flexibility for swapping bytes and words around as needed, and also provides some useful troubleshooting information should the process fail for some reason.

3 DeviceNet™

3.1. Preface

Users should note that the DeviceNet option used in the IND570 terminal is also used in both the METTLER TOLEDO IND780 terminals. These terminals share the same EDS file and Icon file for use in a DeviceNet network configuration tool. However, there are minor differences in the Floating Point polled data between the terminals, so care should be taken to use the appropriate PLC data format guide for each terminal. This chapter describes connections and setup specific to the DeviceNet option for IND570. The formats of data transferred between the IND570 and the PLC are described in Appendix A and Appendix B.

3.2. Overview

DeviceNet is an RS-485 based network utilizing CAN chip technology. This network was created for bit and byte-level devices. The network can be configured to run up to 500Kbits per second depending on cabling and distances. Messages are limited to 8 un-fragmented bytes. Any larger message must be broken up and sent in multiples. The IND570 implementation of DeviceNet does not support fragmented messages – all messages are 8 bytes or shorter. The network is capable of 64 nodes including the master, commonly called the scanner.

3.3. DeviceNet Interface

Figure 3-1 shows a view of the DeviceNet interface option board, with its connector and status lights indicated.

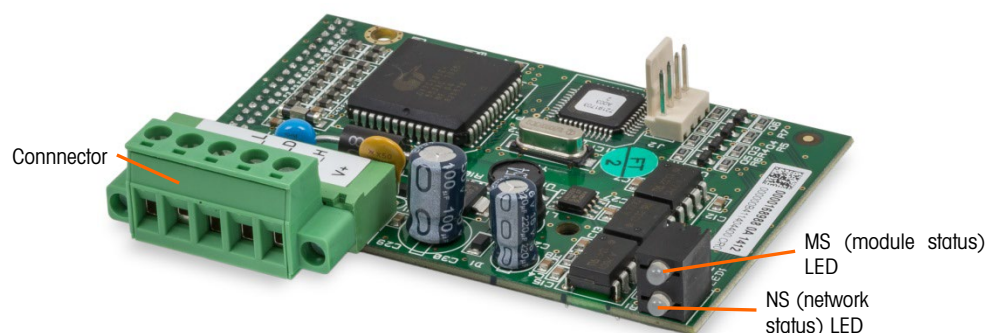


Figure 3-1: DeviceNet Option Board Components

3.3.1. Communications

The IND570 utilizes polled messages. This type of message may be referred to as scheduled or cyclic messages. It does not support explicit or unscheduled messaging.

3.3.2. Node Address

The IND570 can be assigned any valid DeviceNet node address. Typically 0 is reserved for scanner cards and address 63 is reserved for new devices “out of the box”.

3.3.3. Supported Data Formats

For a general account of Data Format types, please refer to Appendix C, **Common Data Features**.

- Note that DeviceNet **cannot** access Shared Data. Appendix A and B provide detailed information on data formats.

3.3.4. Network Power and Current

Table 3-1 and Table 3-2 detail power and current characteristics of the DeviceNet network.

Table 3-1: Network Power Consumption

Voltage	Current
11 V	50 mA
25 V	30 mA

Table 3-2: Network Inrush Current

Voltage	Current	Duration
24 V	0.7 A	6 ms

3.4. Data Definition

3.4.1. Data Integrity

The IND570 Terminals have specific bits to allow the PLC to confirm that data was received without interrupt and the IND570 Terminal is not in an error condition. It is important to monitor these bits. Any PLC code should use them to confirm the integrity of the data received for the IND570 Terminal. Refer to the data charts for specific information regarding the Data OK, Update in Progress, Data Integrity bits and their usage.

3.4.2. Discrete Data

Please refer to Appendix C, **Common Data Features** for a description of discrete data, and to Appendix A and Appendix B for a detailed description of the data available in each format, in order to determine which is most suitable.

There are three formats of discrete data available with the DeviceNet interface option: integer, division, and floating point. Only one type of data format may be selected and used by IND570 terminals sharing the same DeviceNet logical node address.

The integer and division formats allow bi-directional communication of discrete bit encoded information or 16 bit binary word numerical values. The IND570 terminal provides four bytes per message slot. Two slots are available in integer and division mode while only one eight byte slot is available via floating point mode.

The floating-point format allows bi-directional communication of discrete bit encoded information or numeric data encoded in IEEE 754, single precision floating point format. The floating-point format requires more space per IND570 terminal because floating point data uses two 16-bit words of data to represent just the numeric data alone. Selection of the appropriate format depends on issues such as the range or capacity of the scale used in the application. The integer format can represent a numerical value up to 32,767. The division format can represent a value up to 32,767 scale divisions or increments. The floating-point format can represent a value encoded in IEEE 754, single precision floating point format.

3.4.3. Byte Order

For a general account of byte ordering, please refer to Appendix C, **Common Data Features**.

3.4.4. Message Slots

There may be up to 2 message slots for discrete data transfer in the integer or divisions data formats and one message slot for the Floating point data format. Each message slot represents the scale but may be controlled by the PLC to present different data in each message slot. The number of Message Slots is selected in the terminal's setup menu at **Communication > PLC > Data Format**. The data format for the slots are described in Appendix A and B.

The integer and division formats provide two 16-bit words of input and two 16-bit words of output data per slot. Each message slot's first input word provides scale weight data. The type of data displayed, such as gross, tare, etc., is selected by the PLC using the message slot's second output word bits 0, bit 1 and bit 2. Table 3-3 and Table 3-4 provide input and output usage information.

Table 3-3: I/O

Input Data to PLC				Output Data from PLC			
Word Offset	Description		Input Size	Output Size	Description		Word Offset
0	Integer Value	Msg Slot 1	2 Words (4 Bytes)	2 Words (4 Bytes)	Load Integer Value	Command	0
1	Scale Status				1		
2	Integer Value	Msg Slot 2	4 Words (8 Bytes)	4 Words (8 Bytes)	Load Integer Value	Command	2
3	Scale Status				3		

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	2	2	4	4

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
2	4	4	8	8

The floating point format provides four 16-bit words of input data and four 16-bit words of output data as shown in Table 3-4.

Table 3-4: DeviceNet PLC Floating Point I/O Words

Input Data to PLC			Output Data from PLC			
Word Offset	Description	Input Size	Output Size	Description	Word Offset	
0	Integer Value	4 Words (8 Bytes)	4 Words (8 Bytes)	Reserved	0	
1	4 Byte Floating Point Value			Message Slot 1	Command	1
2					4 Byte Floating Point Load Value	2
3	Scale Status					3

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	4	4	4	4

3.5. Floating Point

For a general account of Floating Point operation, data format and compatibility, please refer to Appendix B, Floating Point Format.

3.6. Controlling Discrete I/O Using a PLC Interface

Please refer to Appendix C, Common Data Features.

3.7. Hardware Setup

3.7.1. Wiring

The IND570 wiring is shown in Figure 3-2 and Table 3-5. Consult <http://www.odva.org/> for additional DeviceNet wiring information.

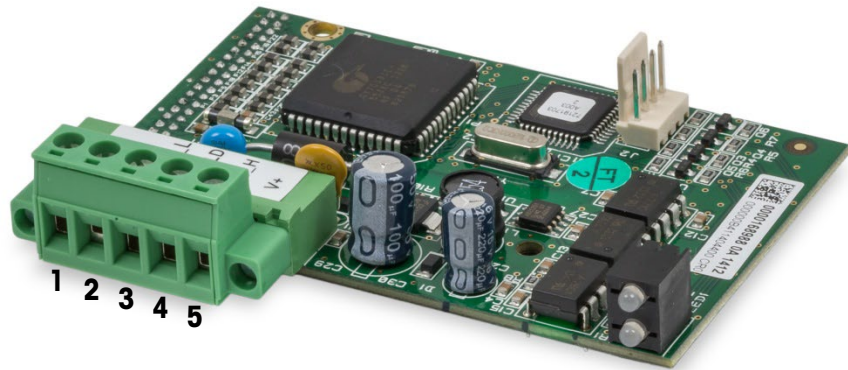


Figure 3-2: DeviceNet Connector Pin Numbering

Table 3-5: DeviceNet Pin Number and Corresponding Wiring

Pin Number	Description	Wire Color
1	V –	Black
2	CAN L	Green
3	Drain	
4	CAN H	White
5	V +	Red

3.8. Software Setup

The IND570 terminal automatically detects the presence of a DeviceNet option board if one is installed. When the option is detected, the IND570 terminal adds the DeviceNet parameters in a program block under **Communications > PLC**. Figure 3-3 graphs the DeviceNet and PLC Data Format program blocks.

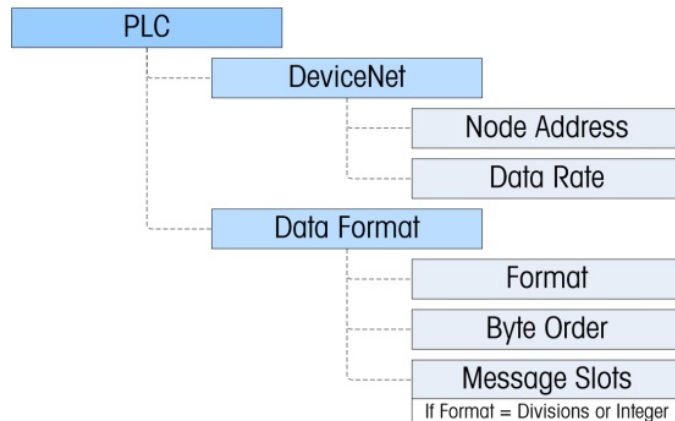


Figure 3-3: The DeviceNet Program Block and Data Format Setup Block

3.8.1. DeviceNet Setup Sub-Block

3.8.1.1. DeviceNet Setup

In Setup, access **Communication > PLC Interface > DeviceNet**. Enter the Node Address field and set an address, from 0 to 63. The address should be unique in the DeviceNet network. Choose the desired Data Rate: 125Kb, 250Kb or 500Kb

3.8.1.2. Data Format Setup

In Setup, access **Communication > PLC Interface > Data Format**. The following fields are available for DeviceNet.

3.8.1.2.1. Format

Format may be selected from a drop-down list. Choose Divisions, Integer (default) or Floating Point.

3.8.1.2.2. Byte Order

Available selections are Standard, Byte Swap, Word Swap (default), and Double Word Swap. See Table 3-3 for definitions.

3.8.1.2.3. Message Slots

If Divisions or Integer is selected for Data Format, the Message Slots option appears. Choose 1 or 2 slots.

3.9. Troubleshooting

If the IND570 does not communicate with PLC, do the following:

- Check wiring and network termination.
- Confirm that the IND570 settings for Node Address and Data Rate match those in the PLC and that each IND570 has a unique address.
- Confirm that the EDS file has been loaded into the Network Configuration Tool (RSNetWorx for DeviceNet or equivalent) and that the node is recognized by the tool.
- Confirm that the node has been properly configured in the Network Configuration Tool (Node is in the scan list with the correct I/O Sizes and location in the scan buffer memory), and that the configuration has been downloaded to the scanner module in the PLC's rack.
- Make sure that the Power Supply for the Network power is not shared other devices that are overloading the Power Supply or causing noise on the lines (devices such as relays, solenoids, motor starters, etc. should never share this power supply with the Network).
- Make sure that Network errors such as "Bus Off Detected" are cleared at the PLC (reset of the scanner module is probably needed).
- If the PLC interface pcb was changed from another type, like EtherNet/IP, a master reset of the IND570 should be performed. Contact Mettler Toledo service for assistance.
- Contact METTLER TOLEDO service for replacement of the DeviceNet interface.

3.9.1. Module status LED (MS)

All DeviceNet nodes are required to have 2 status LED's. These LED's (labeled in Figure 3-1) indicate module and network status. Refer to the definitions below.

This LED displays the status of the IND570 Terminal DeviceNet board.

Table 3-6: Module Status LED Indications

LED State	Meaning
Solid Green	Normal operation
Flashing Green	DeviceNet board fault
OFF	No power to the DeviceNet board
Solid Red	Unrecoverable board fault
Flashing Red	Recoverable fault
Flashing Orange	Board performing self-test

- **Note:** If the module status LED indicates anything other than normal operation, after powering up the unit and attaching the DeviceNet cable, the IND570 Terminal must be powered down and restarted. If the LED continues to show a condition other than solid green, replace the board.

3.9.2. Network status LED (NS)

This LED displays network status.

Table 3-7: Network Status LED Indications

LED State	Meaning
Solid Green	Node is communicating to scanner
Flashing Green	Device is connected to the network but not being scanned The most common reason for this is the device has not been added to the scan list. Consult DeviceNet configuration tool's help in order to commission the node and put it in the scan list.
OFF	No DeviceNet power
Solid Red	Critical Link error This error typically indicates a cable problem
Blinking Red	Connection Timeout

3.10. DeviceNet Option Kit

No spare parts are associated with the DeviceNet option kit. The kit part number is 30116110. Table 3-8 shows what comes in the kit.

Table 3-8: DeviceNet Option Kit

Description	Qty.
Installation Instructions	1
PCB Package	1
Installation Kit	1
Gland Kit	1

3.11. DeviceNet Commissioning and Configuration Examples

The user must register the EDS, connect the device and add the IND570 Terminal to the DeviceNet master's scanlist. Note that every vendor's software is different. Depending upon master and software, the user may have to cycle power on the master in order to complete the commissioning of any device added to the network. Consult the master's documentation for more information. The following example is for Rockwell software and Logix5000 processor.

3.11.1. Configuring the IND570 Terminal with RSNetWorx for DeviceNet

The EDS file located on the CD-ROM supplied with the IND570 Terminal contains configuration information to allow RSNetWorx for DeviceNet to set up a single polled I/O connection between a METTLER TOLEDO IND570 Terminal and DeviceNet master/scanner. Note that this order of operations isn't the only way of configuring the IND570 terminal.

3.11.1.1. Registering the EDS File

The EDS file must first be registered into RSNetWorx for DeviceNet. This is accomplished using the EDS Wizard.

3.11.1.1.1. To access the EDS Wizard

1. Click **T**ools then **EDS Wizard...** to begin the registration process.

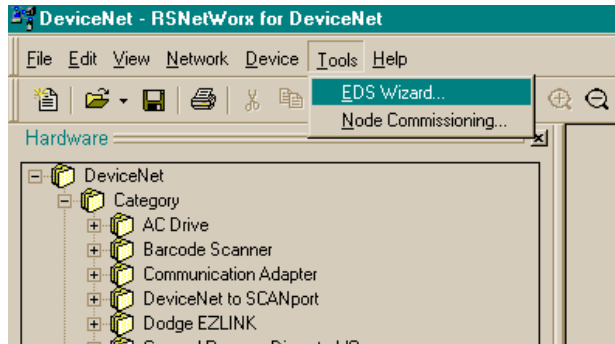


Figure 3-4: Accessing the EDS Wizard

2. The EDS Wizard Welcome screen appears.

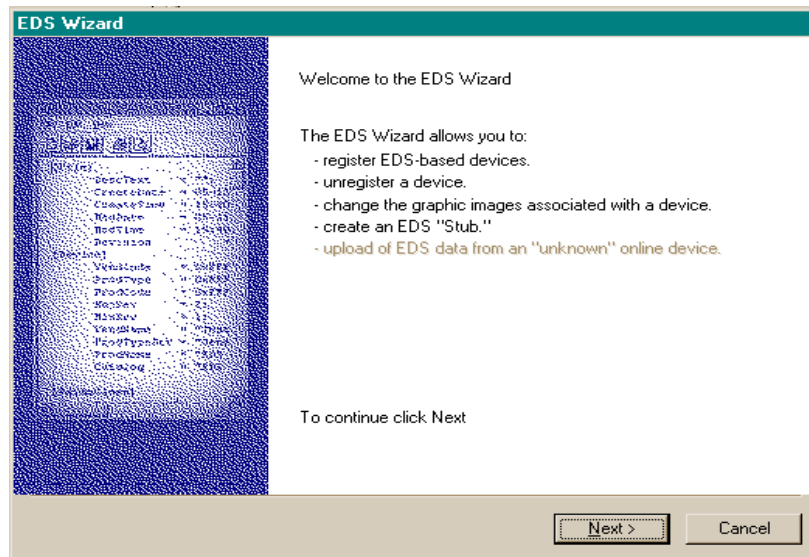


Figure 3-5: EDS Wizard Welcome

3. Click **N**ext to begin the registration process.

- In the Options screen, make sure the **Register an EDS file(s)** radio button is selected.

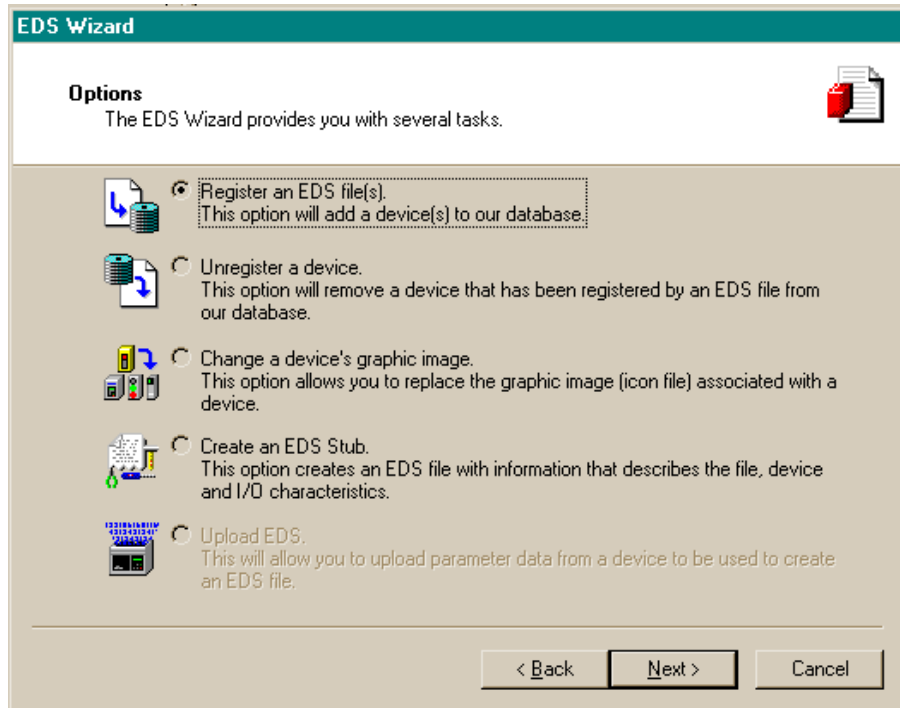


Figure 3-6: EDS Wizard Options Screen

- Click **Next**, then **Browse** to select a file to register.
- Browse to the appropriate location and select the file **MT_IND-DNET.eds**. (The EDS file is located on the CD-ROM.) Click the **Open** button.
- Confirm that the correct file is showing in the **Named:** field, then click **Next**.

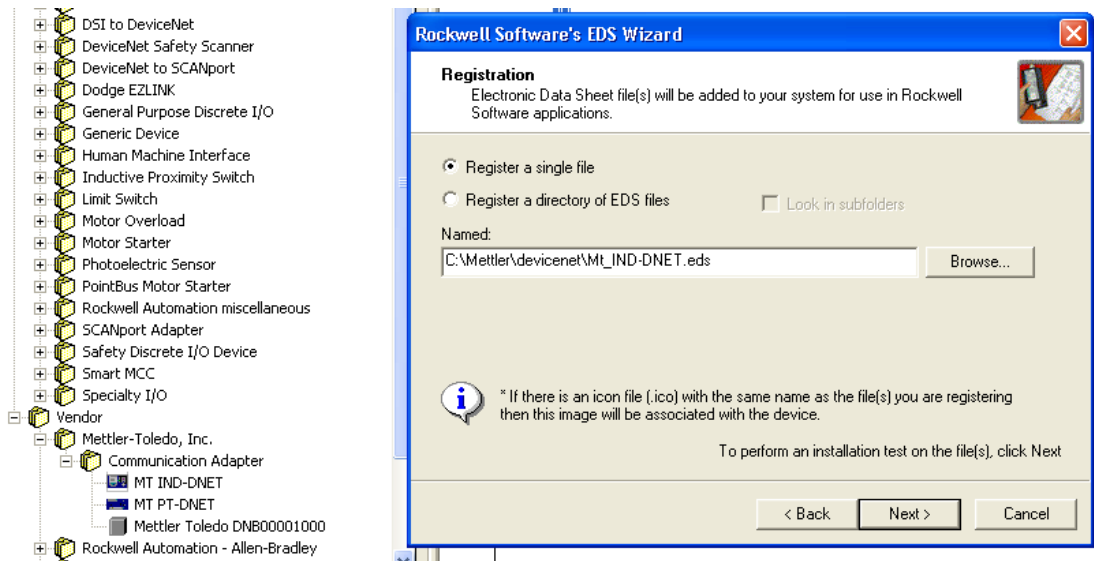


Figure 3-7: File Selected, Ready to Register

- Acknowledge the error applet. This error is generated due to the EDS file being generic for other MT devices. The IO sizes will be configured later in the process.

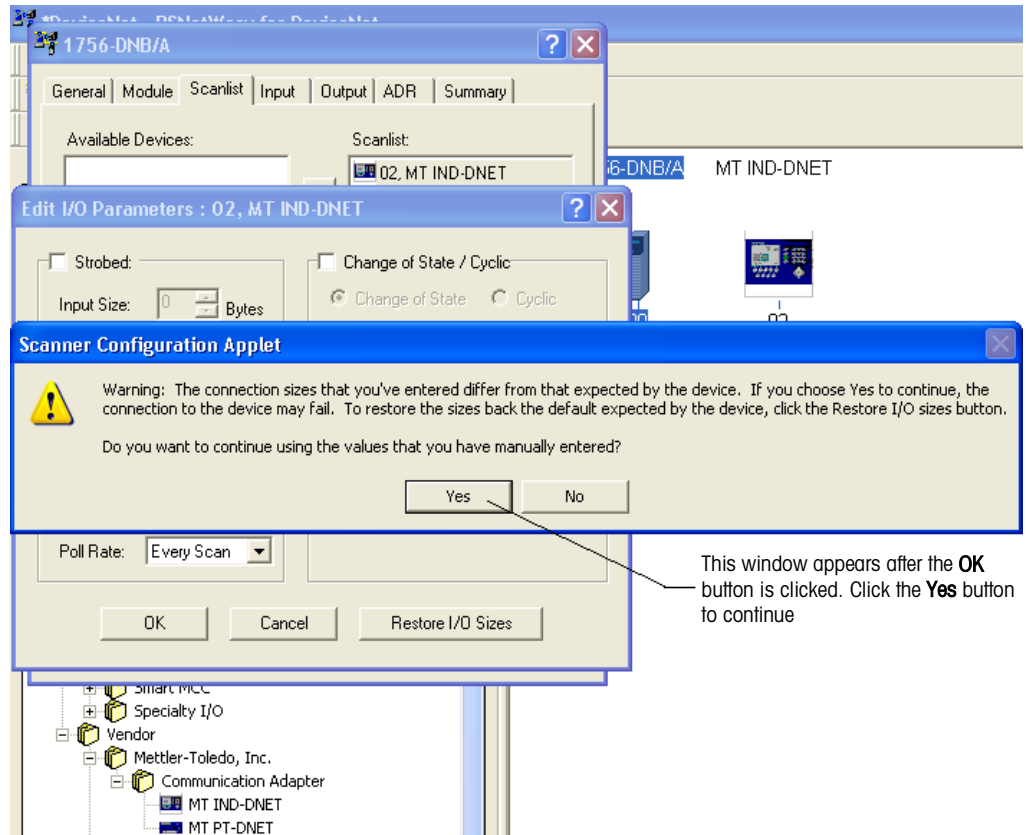


Figure 3-8: File Validity Confirmed

9. Ensure that the **MT_IND-DNET.ico** icon is selected.
- **Note:** RSNetWorx for DeviceNet will not be able to find the icon unless it is in the same directory as the EDS file.

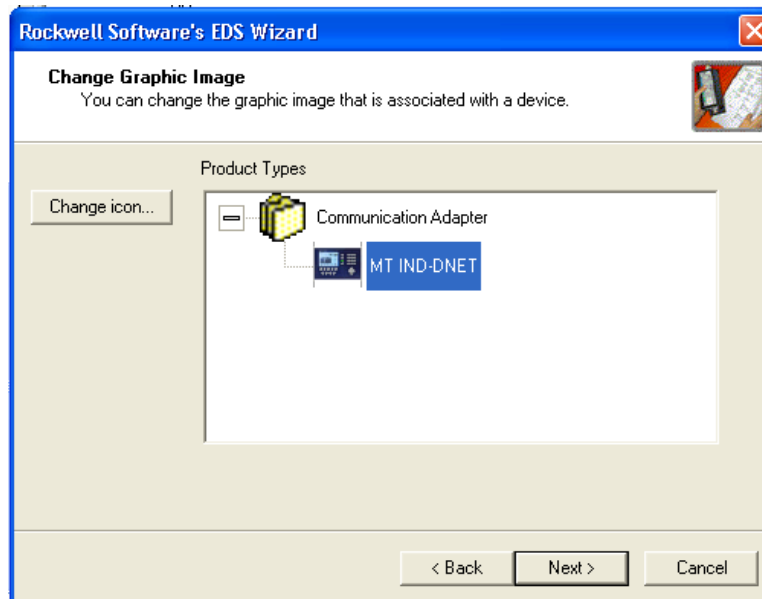


Figure 3-9: EDS Wizard Graphic Image (Icon) Selection

10. The Final Task Summary screen (Figure 3-10) will appear. Click **Next**.

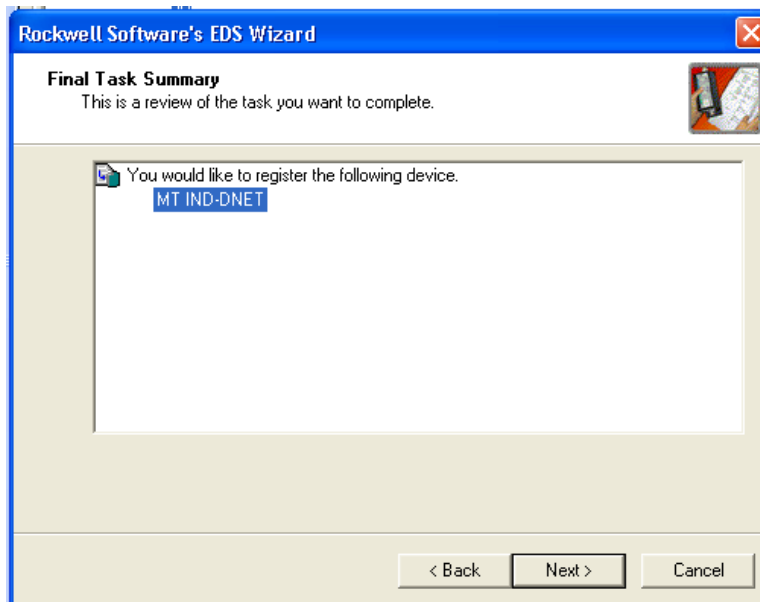


Figure 3-10: Final Task Summary Screen

3.11.1.1.2. Setting up an I/O Connection

After the EDS file has been registered, RSNetWorx is used to set up a polled connection between the METTLER TOLEDO IND Terminal and the DeviceNet master/scanner.

■ **Note:** You must add the DeviceNet scanner card and choose the proper revision before going online.

1. Select **Network** then **Online** to browse the DeviceNet network.

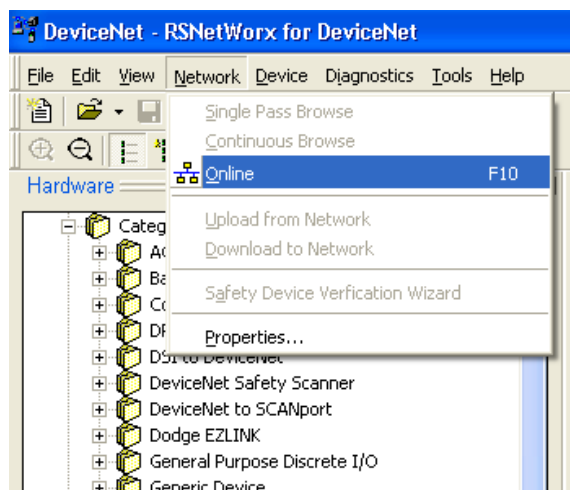


Figure 3-11: RSNetWorx Online Browse

2. Select the appropriate network path. In this case (Figure 3-12), 1756-DNB/A DeviceNet Scanner is selected.

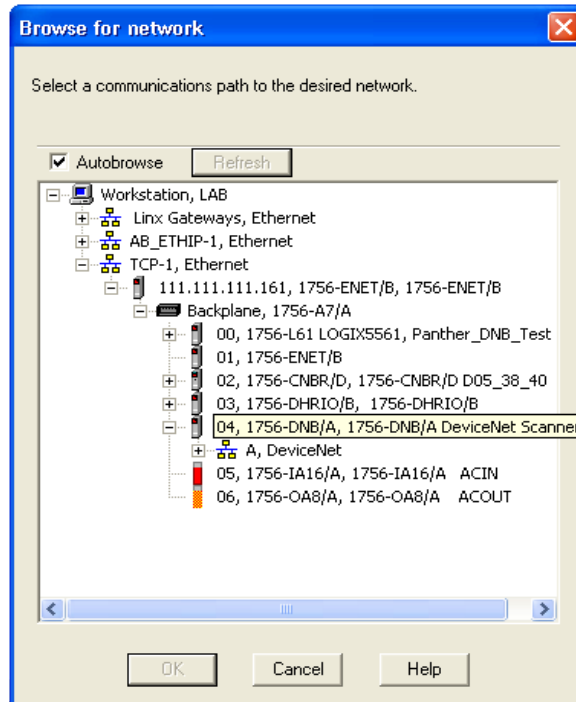


Figure 3-12: Browse for network

3. Click **OK** to continue. A dialog box like the one shown in Figure 3-13 will appear. Note that you may be asked to upload or download, depending on the version of software used.

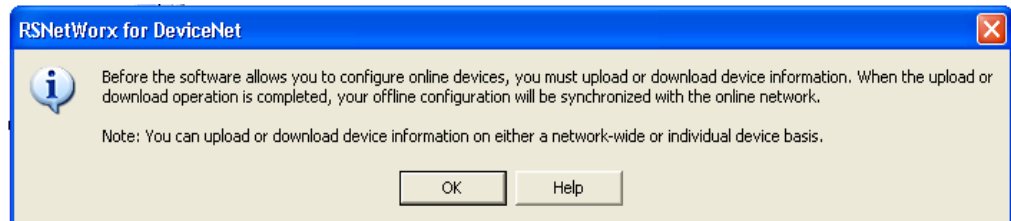


Figure 3-13: Confirmation Dialog Box

4. Click **OK** in the dialog box. A **Browsing network...** box will display with a progress meter indicating that the process is underway.

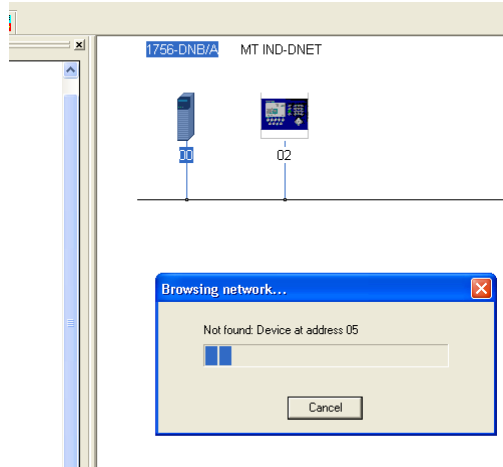


Figure 3-14: Browsing Network Underway

5. Once the scanner has browsed the entire network, add IND Terminal to the 1756-DNB/A's scanlist by right-clicking on the scanner icon in the Graph window (note the tab at the bottom of Figure 3-15), and choosing **Properties...**



Figure 3-15: Accessing Scanner Properties

6. The scanner's properties dialog box will appear, as in Figure 3-16.

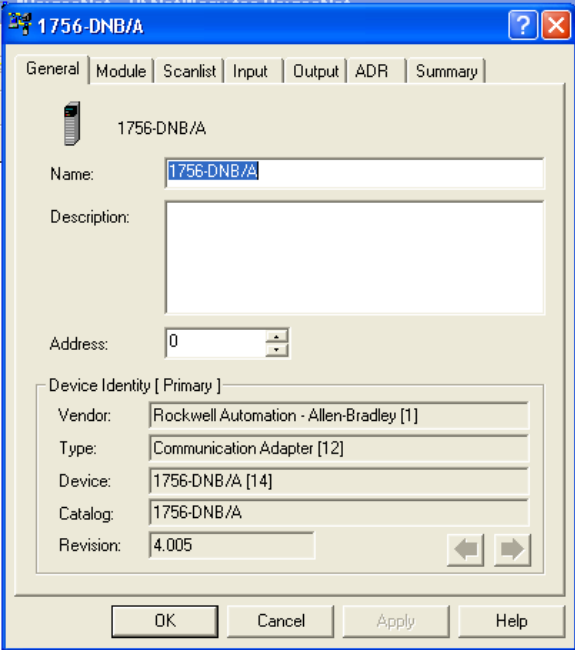


Figure 3-16: Scanner Properties Dialog: Initial View

7. Click the Scanlist tab in the properties dialog box. The view shown in Figure 3-17 will appear.

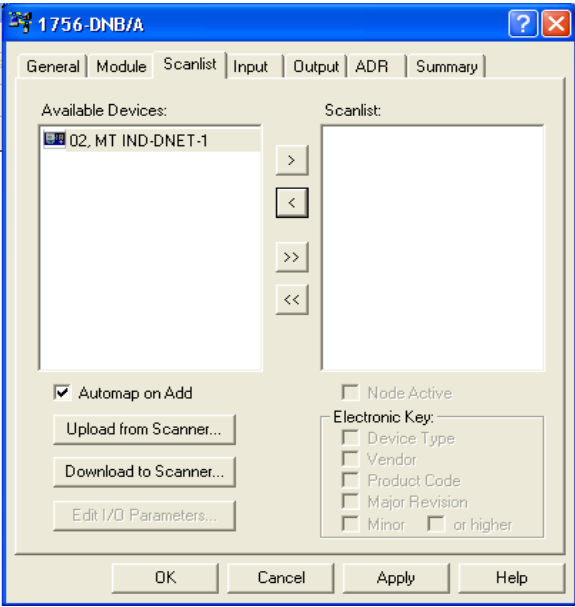


Figure 3-17: Scanner Properties Dialog: Scanlist Tab Showing

8. Highlight the IND Terminal (MT IND-DNET) and left-click to add it to the Scanlist. Once the IND Terminal is added, it will appear in the right pane (Figure 3-18). Click **OK**.

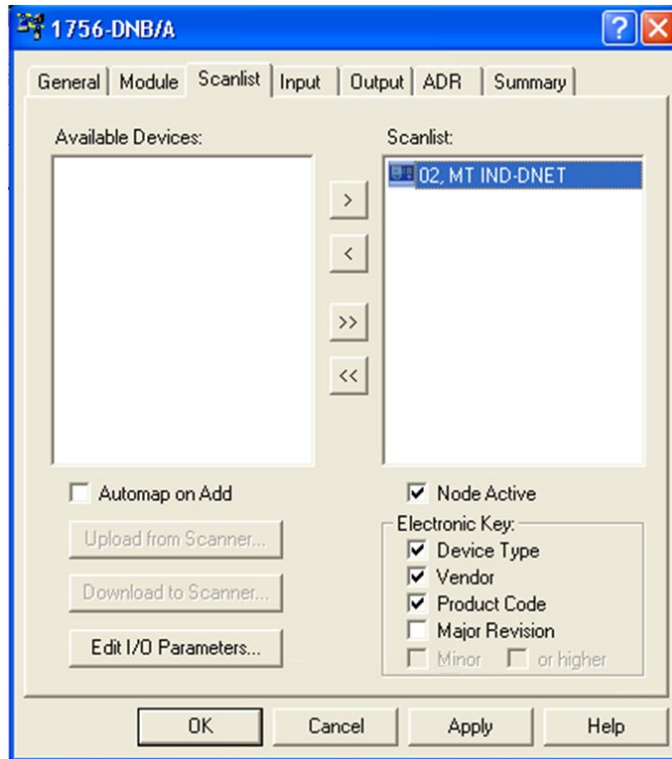


Figure 3-18: IND Terminal Added to Scanlist

9. The next step is to edit the I/O parameters of the IND terminal, by clicking on the **Edit I/O Parameters** button visible in Figure 3-18. The I/O size depends on the data type and the number of slots selected in the terminal. Note that slots is a terminal phrase that isn't used in typical DeviceNet terms; it was derived from previous PLC memory mapping. It is used in the terminal setup to remain consistent across the METTLER TOLEDO terminal line. Integer or Divisions with one slot will be 4 bytes/in 4 bytes out. Integer or Division with 2 slots is 8 bytes in/8 bytes out. Float is always 8 bytes in and out.

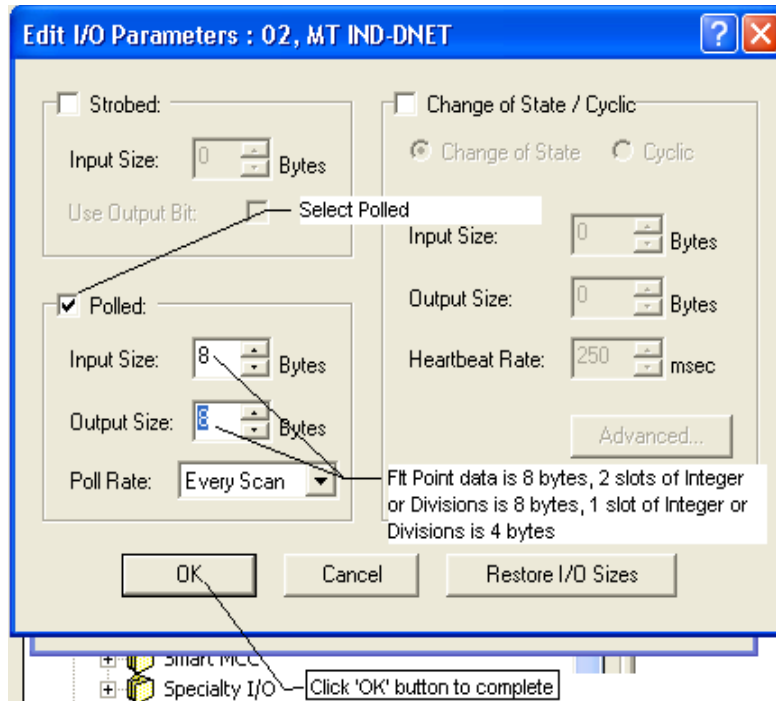


Figure 3-19: Editing I/O Parameters

- Download the configuration to the scanner card, in order to commission the network. In the prompt that appears (Figure 3-20), click **Yes** to continue. Note that some scanner cards may require power down for changes to take effect.

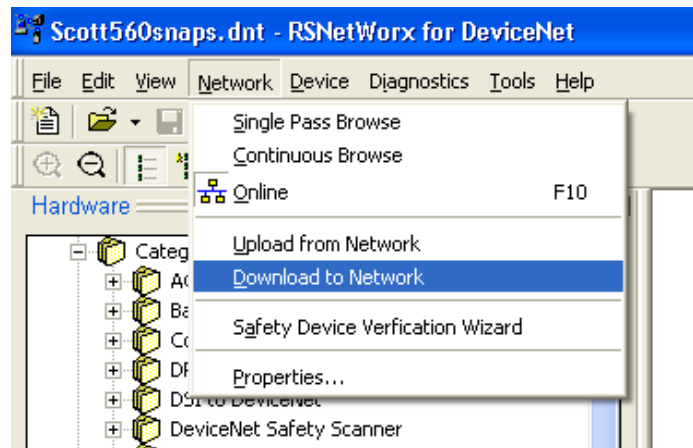


Figure 3-20: Network Download Confirmation Prompt

- Once the IND Terminal has been added to the scanlist, access the Properties dialog to verify its I/O mapping (Figure 3-21 and Figure 3-22) within the scanner card. Note that auto or manual

mapping can be used. Consult the master’s documentation for advance I/O mapping options. If manual mapping is used. Be sure to start with an unused section of memory for offset.

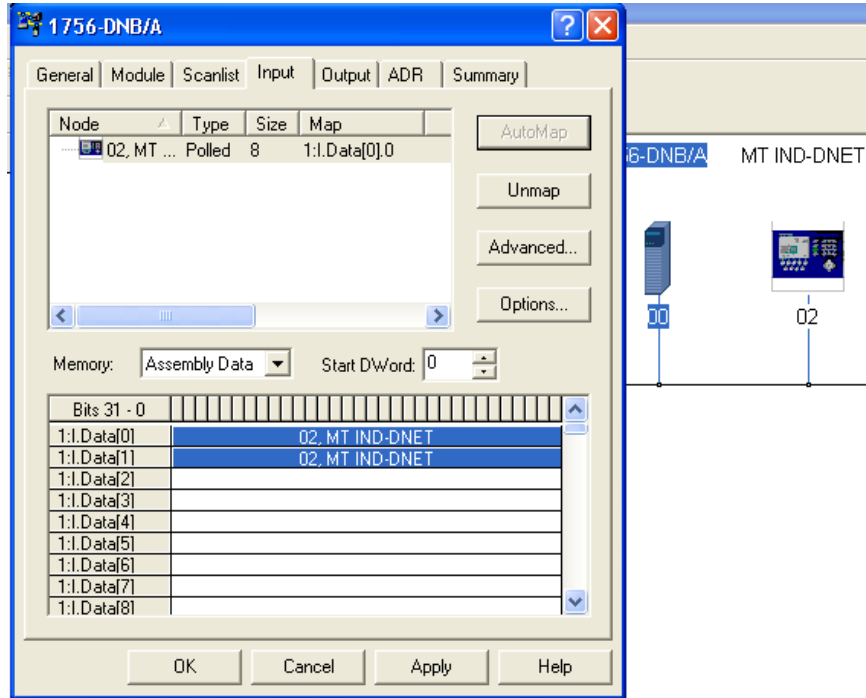


Figure 3-21: IND Terminal Mapping

12. Verify that I/O mapping is complete by choosing the summary tab. Note that the Mapped columns show **Yes** for the IND Terminal.

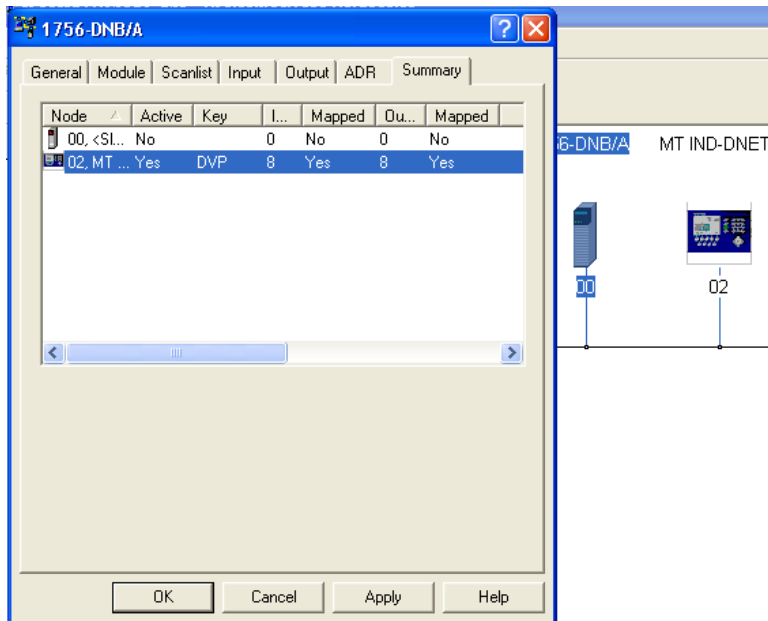


Figure 3-22: Summary Tab

3.11.2. PLC Programming

The following Figures show screen images of ladder logic programming examples for RSLogix 5000 software (version 20).

The sample program demonstrates logic to interface to an IND570 set up for Floating Point Data or Integer Data Formats. The logic also includes routines that access Shared Data over the ControlNet interface in both Floating Point and Integer Data Formats.

This sample program is subject to change without notice. Please visit www.mt.com to download the most recent version of PLC sample code.

- **Note:** Complete versions of the examples can be downloaded from www.mt.com/IND570. These screen images are provided for illustrative purposes only.

Figure 3-23 shows the configuration of the DeviceNet Scanner Module (1756-DNB)

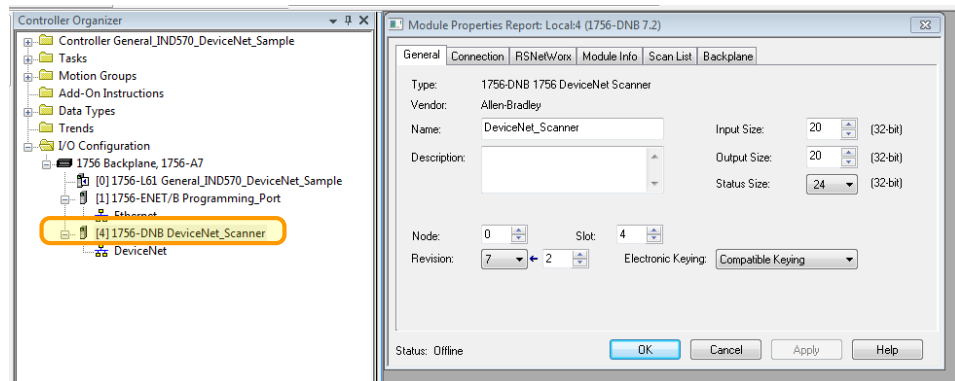


Figure 3-23: DeviceNet Scanner Module Configuration

3.11.3. General Programming Notes

The sample program provides User-Defined types that can be used to help with your program's documentation. For Floating Point, the User-Defined types **Cmd_Response** and **FP_Scale_Status** are shown in Figure 3-24.

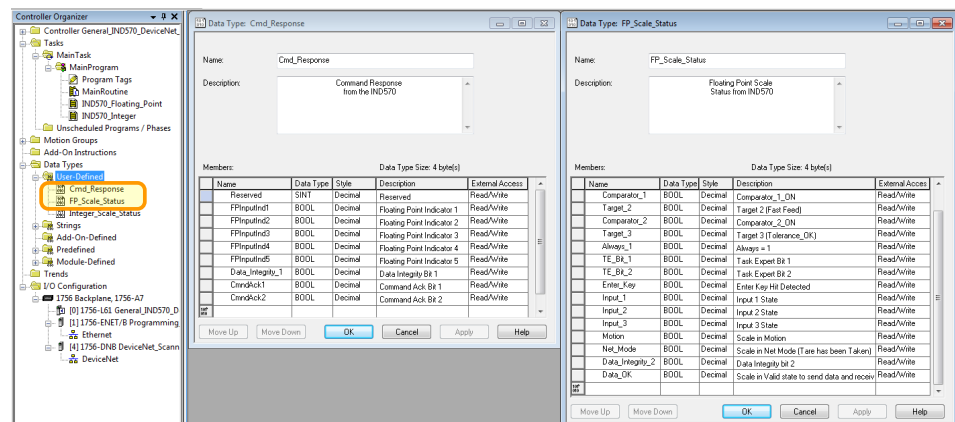


Figure 3-24: User-Defined Floating Point Data Types

For Integer and Divisions mode, the User-Defined type **Integer_Scale_Status** is shown in Figure 3-25.

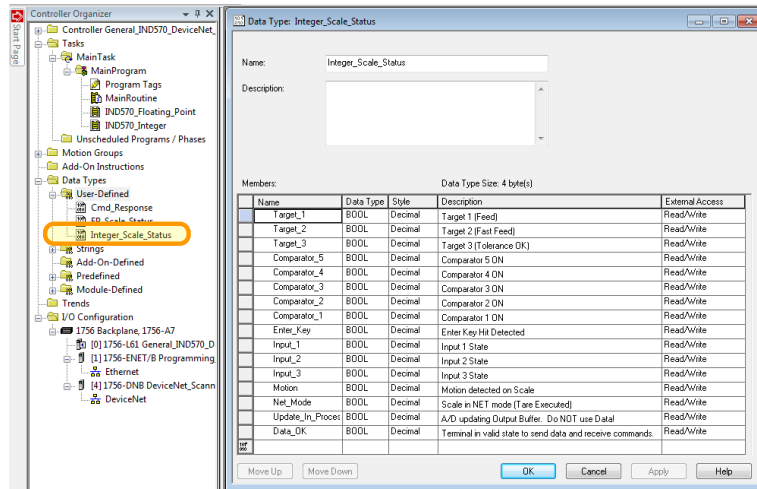


Figure 3-25: User-Defined Integer-Divisions Data Types

Making use of these User-Defined types in your own program can help to simplify the overall programming effort.

The following principles should always be applied to guarantee the validity of the data before using it in a process. Note that there are different principles for the different modes (Floating Point versus Integer or Divisions).

For Floating Point Mode, data being read from the Terminal should be filtered with the **Data_OK** bit and the two Data Integrity bits as shown in Figure 3-26.

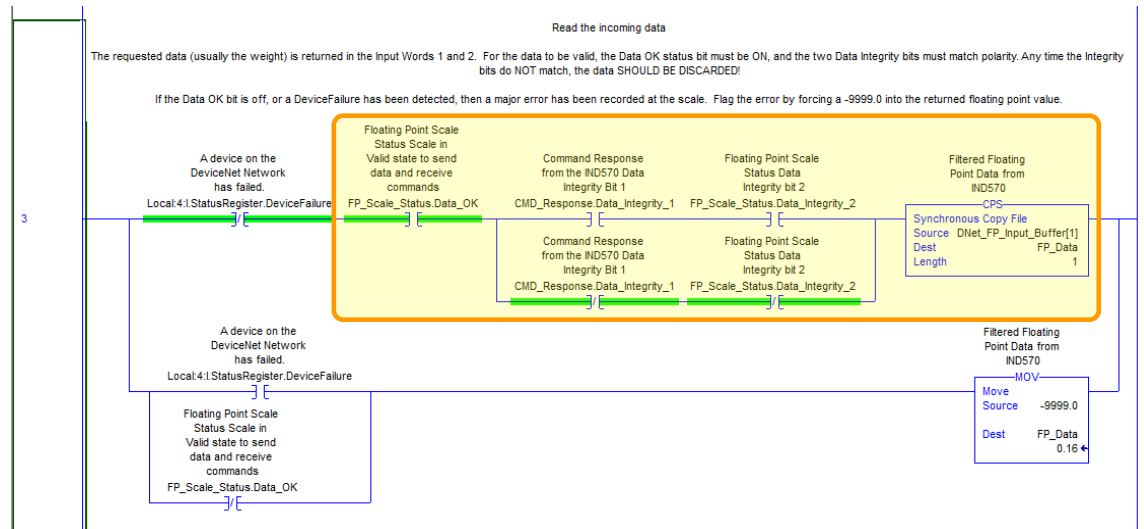


Figure 3-26: Filtering of Floating Point Data into PLC

Filtering the data in this way makes sure that the Terminal is in a valid operational state (**Data_OK** = 1) and that the Analog Update from the Load Cell has properly completed before the data was

read (**Integrity_1 = Integrity_2**). Failing to perform these checks can result in the PLC program using invalid data.

For Integer or Division Mode, a similar filter should be applied as shown in Figure 3-27.

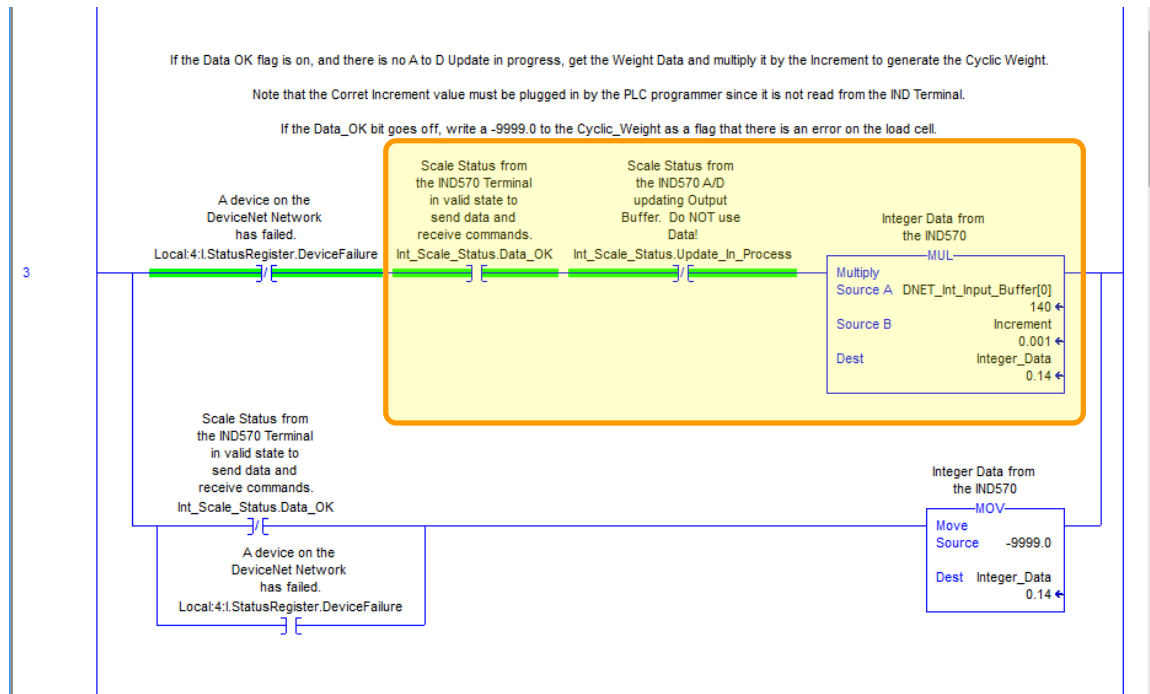


Figure 3-27: Filtering Integer/Divisions Data into the PLC

In this case, the data is filtered with the **Data_OK** and **Update_In_Progress** bits to make sure that the data coming back from the terminal is valid. From there, it is converted into a Floating Point value by multiplying it by the hard-coded Increment size for the terminal to properly place the decimal point.

- **Note:** Due to the fact that I/O Data may not be zeroed by the PLC when communications to the DeviceNet Node fails, it is recommended that the Input data be further filtered by the communications state flag available from the DeviceNet Scanner Module.

4 EtherNet/IP™

4.1. Preface

Users should note that the Ethernet/IP option used in the IND570 terminal is also used in the METTLER TOLEDO IND131, IND331, IND570 and IND780 terminals.

There are minor differences in the Floating Point polled data between the terminals, so care should be taken to use the appropriate PLC data format guide for each terminal family. This chapter describes connections and setup that are specific to the EtherNet/IP option for IND570. The formats of the data that is transferred between the IND570 and the PLC are described in Appendix A and Appendix B.

4.2. EtherNet/IP Interface Board

Figure 4-1 shows an EtherNet/IP module and its components. Note that the module's address is set in software (see Figure 4-2), and the DIP switches indicated in Figure 4-1 must all be set to OFF.

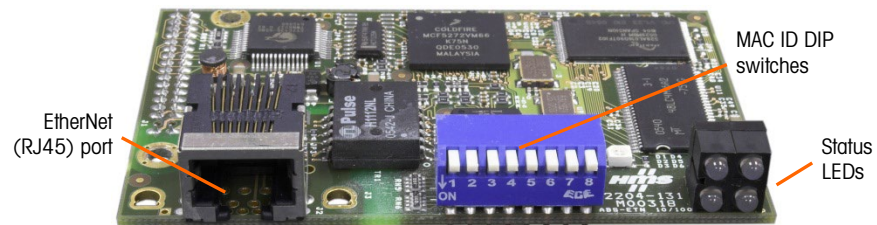


Figure 4-1: EtherNet/IP Module Components

4.3. Overview

EtherNet/IP, short for "EtherNet Industrial Protocol," is an open industrial networking standard that takes advantage of commercial, off-the-shelf EtherNet communication chips and physical media. This networking standard supports both implicit messaging (real-time I/O messaging) and explicit messaging (message exchange). The protocol is supported by the Open DeviceNet Vendor Association (ODVA).

EtherNet/IP utilizes commercial, off-the-shelf EtherNet hardware (for example, switches and routers) and is fully compatible with the Ethernet TCP/IP protocol suite. It uses the proven Control and Information Protocol (CIP) to provide control, configuration, and data collection capability.

The kit enables the IND570 terminal to communicate to Programmable Logic Controllers (PLCs) through direct connection to the EtherNet/IP network at either 10 or 100 MBPS speed. The kit

consists of a backplane-compatible I/O module, mounting hardware, and a ferrite. Software to implement the data exchange resides in the IND570 terminal.

4.4. EtherNet/IP Characteristics

The EtherNet/IP Kit option has the following features:

- User-programmable IP addressing.
- Capability for bi-directional discrete mode communications (Class 1 Messaging) of weight or display increments, status, and control data between the PLC and the IND570.

4.4.1. Definition of Terms

Some terms (such as Target) used by the EtherNet/IP PLC application have a different sense from their use by the IND570 terminal. Table 4-1 provides definitions specific to EtherNet/IP.

Table 4-1: EtherNet/IP Definition of Terms

Term	Definition
Adapter Class	An Adapter Class product emulates functions provided by traditional rack-adapter products. This type of node exchanges real-time I/O data with a Scanner Class product. It does not initiate connections on its own.
Class 1 Messaging	In EtherNet/IP communication protocol scheduled (cyclic) message transfer between a PLC and CIP Adapter Class device.
Class 3 Messaging	In EtherNet/IP communication protocol unscheduled message transfer between a PLC and CIP Adapter Class device. This is used by the IND570 for explicit messaging.
Connected Messaging	A connection is a relationship between two or more application objects on different nodes. The connection establishes a virtual circuit between end points for transfer of data. Node resources are reserved in advance of data transfer and are dedicated and always available. Connected messaging reduces data handling of messages in the node. Connected messages can be Implicit or Explicit . See also Unconnected Messaging .
Connection Originator	Source for I/O connection or message requests. Initiates an I/O connection or explicit message connection.
Explicit Messaging	Explicit Messages (also known as Discrete, or Class 3, or Acyclic messages) can be sent as a connected or unconnected message. CIP defines an Explicit Messaging protocol that states the meaning of the message. This messaging protocol is contained in the message data. Explicit Messages provide a one-time transport of a data item. Explicit Messaging provide the means by which typical request/response oriented functions are performed (e.g. module configuration). These messages are typically point-to-point.

Term	Definition
Implicit Messaging	Implicit Messages (also known as Class 1, or cyclic messages) are exchanged across I/O Connections with an associated Connection ID. The Connection ID defines the meaning of the data and establishes the regular/repeated transport rate and the transport class. No messaging protocol is contained within the message data as with Explicit Messaging. Implicit Messages can be point-to-point or multicast and are used to transmit application-specific I/O data. This term is used interchangeably with the term I/O Messaging .
I/O Client	Function that uses the I/O messaging services of another (I/O Server) device to perform a task. Initiates a request for an I/O message to the server module. The I/O Client is a Connection Originator .
I/O Messaging	Used interchangeably with the term Implicit Messaging .
I/O Server	Function that provides I/O messaging services to another (I/O Client) device. Responds to a request from the I/O Client. I/O Server is the target of the connection request.
Message Client	Function that uses the Explicit messaging services of another (Message Server) device to perform a task. Initiates an Explicit message request to the server device.
Message Server	Function that provides Explicit messaging services to another (Message Client) device. Responds to an Explicit message request from the Message Client.
Scanner Class	A Scanner Class product exchanges real-time I/O data with Adapter Class and Scanner Class products. This type of node can respond to connection requests and can also initiate connections on its own.
Target	Destination for I/O connection or message requests. Can only respond to a request, cannot initiate an I/O connection or message.
Unconnected Messaging	Provides a means for a node to send message requests without establishing a connection prior to data transfer. More overhead is contained within each message and the message is not guaranteed destination node resources. Unconnected Messaging is used for non-periodic requests (e.g. network "Who" function). Explicit messages only. See also Connected Messaging .

4.4.2. Communications

The IND570 terminal utilizes component parts to ensure complete compatibility with the EtherNet/IP network. An IND570 terminal is recognized as a generic EtherNet/IP device by the PLC.

Each EtherNet/IP option connected to the EtherNet/IP network represents a physical IP Address. The connection is made via a RJ-45 connector on the option card (see Figure 4-1).

The wiring between the PLC and the IND570 EtherNet/IP connection uses EtherNet twisted pair cable. The cable installation procedures and specification including distance and termination requirements are the same as recommended by Allen-Bradley for the EtherNet/IP network.

The IND570 only uses Class 1 cyclic data for discrete data and Class 3 explicit messages for access to the IND570 Shared Data Variables. Explicit message blocks may be connected or unconnected; the PLC programmer must make this choice.

4.4.3. IP Address

Each EtherNet/IP option represents one physical IP Address. This address is chosen by the system designer, and then programmed into the IND570 terminal and PLC. The IND570 terminal's address is programmed at **Communication > PLC Interface > EtherNet/IP-Modbus TCP** in the terminal's setup menu. The IND570 IP Address entry must be unique for each IND570 terminal, and must not conflict with other devices on the network.

4.4.4. Supported Data Formats

The terminal's EtherNet/IP interface provides discrete data transfer and Class 1 messaging. Data transfer is accomplished via the PLC's cyclic messaging. The EtherNet/IP interface has its own logical IP address to send and receive information to and from the PLC continuously. The EtherNet/IP interface uses discrete data for its communication with PLCs.

Three formats of discrete data are available with the EtherNet/IP interface option: integer (the default), divisions and floating point.

Appendix A and B provide detailed information on data formats.

4.5. Data Definition

4.5.1. Assembly Instances of Class 1 Cyclic Communications

Class 1 cyclic communications is used for transfer of Discrete Data between the PLC and the IND570.

The PLC Input Assembly Instance is 100 (decimal). This instance is used for all Data Formats and data size requirements.

The PLC Output Assembly Instance is 150 (decimal). This instance is used for all Data Formats and data size requirements.

The IND570 uses data only. Configuration data is not used or required. Within the PLC EtherNet/IP Interface setup set the Configuration Instance to 1 and the data size to zero.

The EDS file provided on the IND570 documentation CD has no Assembly Instance or data size limitations. The IND570 programming controls the Assembly Instance and data size limitations.

4.5.2. Discrete Data

Please refer to Appendix C, **Common Data Features** for a description of discrete data, and to Appendix A and Appendix B for a detailed description of the data available in each format, in order to determine which is most suitable.

4.5.3. Byte Order

For a general account of byte ordering, please refer to Appendix C, **Common Data Features**.

4.5.4. Message Slots

There may be up to 4 message slots for discrete data transfer, Class 1 messaging, in Integer, Divisions and Floating Point Data Formats. Each message slot represents the scale but may be controlled by the PLC to present different data in each message slot. The number of Message Slots is selected in the terminal's setup menu at **Communication > PLC > Data Format** (Figure 4-2).

The integer and division formats provide two 16-bit words of input and two 16-bit words of output data per Slot. Each Message Slot's first input word provides scale weight data. The type of data displayed, such as Gross, Tare, etc., is selected by the PLC using the Message Slot's second output word bits 0, bit 1 and bit 2. Table 4-2 and Table 4-3 provide input and output usage information.

Table 4-2: EtherNet/IP PLC Integer and Division Input Data and Data Usage

Input Data to PLC				Output Data from PLC			
Word Offset	Description		Input Size	Output Size	Description	Word Offset	
0	Integer Value	Msg Slot 1	2 Words (4 Bytes)	2 Words (4 Bytes)	Load Integer Value	0	
1	Scale Status				Command	1	
2	Integer Value	Msg Slot 2	4 Words (8 Bytes)	4 Words (8 Bytes)	Load Integer Value	2	
3	Scale Status				Command	3	
4	Integer Value	Msg Slot 3	6 Words (12 Bytes)	6 Words (12 Bytes)	Load Integer Value	4	
5	Scale Status				Command	5	
6	Integer Value	Msg Slot 4	8 Words (16 Bytes)	8 Words (16 Bytes)	Load Integer Value	6	
7	Scale Status				Command	7	

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	2	2	4	4
2	4	4	8	8
3	6	6	12	12
4	8	8	16	16

The floating point format provides four 16-bit words of input data and three 16-bit words of output data) per Message Slot. Table 4-3 provides details.

Table 4-3: EtherNet/IP PLC Floating Point Input Words

Input Data to PLC				Output Data from PLC			
Word Offset	Description		Input Size	Output Size	Description	Word Offset	
0	Command Response	Message Slot 1	4 Words (8 Bytes)	4 Words (8 Bytes)	Reserved	0	
1	4-Byte Floating Point Value				Message Slot 1	Command	1
2						4-Byte Floating Point Load Value	2
3	Scale Status				Message Slot 1	4-Byte Floating Point Load Value	3
4	Command Response	Message Slot 2	8 Words (16 Bytes)	7 Words (14 Bytes)	Command	4	
5	4-Byte Floating Point Value				Message Slot 2	4-Byte Floating Point Load Value	5
6						4-Byte Floating Point Load Value	6
7	Scale Status				Message Slot 2	4-Byte Floating Point Load Value	6
8	Command Response	Message Slot 3	12 Words (24 Bytes)	10 Words (20 Bytes)	Command	7	
9	4-Byte Floating Point Value				Message Slot 3	4-Byte Floating Point Load Value	8
10						4-Byte Floating Point Load Value	9
11	Scale Status				Message Slot 3	4-Byte Floating Point Load Value	9
12	Command Response	Message Slot 4	16 Words (32 Bytes)	13 Words (26 Bytes)	Command	10	
13	4-Byte Floating Point Value				Message Slot 4	4-Byte Floating Point Load Value	11
14						4-Byte Floating Point Load Value	12
15	Scale Status				Message Slot 4	4-Byte Floating Point Load Value	12

I/O Size Summary				
Message Slot	Words		Bytes	
	Input	Output	Input	Output
1	4	4	8	8
2	8	7	16	14
3	12	10	24	20
4	16	13	32	26

4.5.5. Floating Point

For a general account of Floating Point operation, data format and compatibility, please refer to Appendix B, **Floating Point Format**.

4.5.5.1. Data Integrity

The IND570 uses two data integrity bits to maintain data integrity when communicating with the PLC. One bit is in the beginning word of the data; the second is in the ending byte of the data for a scale slot. The PLC program must verify that both data integrity bits have the same polarity for the data in the scale slot to be valid. There is a possibility that the PLC program will see several consecutive invalid reads when the terminal is freely sending weigh updates to the PLC, if the PLC program detects this condition, it should send a new command to the terminal.

The method of handling string and floating point data varies between Allen-Bradley PLC generations.

4.6. Shared Data Mode

The Shared Data mode PLC communications is provided using CIP explicit (Class 3) messages.

The IND570 Shared Data Reference manual lists the Shared Data Variables available to EtherNet/IP. This document also includes the hex Class Code, Instance and Attribute for the shared data. The PLC must use Get Attribute Single (service code e) to read a Shared Data Variable and Set Attribute Single (service code 10) to write a Shared Data Variable.

The **IND570 Shared Data Reference** is available on the terminal's documentation CD.

4.7. Controlling Discrete I/O Using a PLC Interface

Please refer to Appendix C, **Common Data Features**.

4.8. Software Setup

When the IND570 terminal detects the presence of a EtherNet/IP Kit option board, the EtherNet/IP parameters are enabled in a Setup program block at **Communication > PLC > EtherNet/IP**. Figure 4-2 shows the EtherNet/IP setup block.

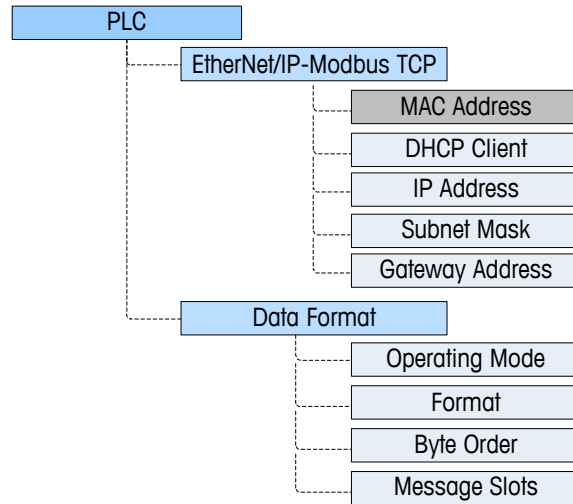


Figure 4-2: EtherNet/IP Setup Block

4.8.1. EtherNet/IP and Data Format Setup Blocks

4.8.1.1. EtherNet/IP Setup

The EtherNet/IP setup block at **Communication > PLC Interface > EtherNet/IP** is where the IP Address, Subnet Mask and Gateway Address of the EtherNet/IP interface are specified. If DHCP is selected, the IND570 will carry out a power cycle and will automatically populate the IP Address, Subnet Mask and Gateway Address fields with information received from the network's DHCP router.

The MAC address is displayed, but cannot be modified.

4.8.1.2. Data Format Setup

In Setup, navigate to **Communication > PLC Interface > Data Format**. The following fields are available for EtherNet/IP.

4.8.1.2.1. Operating Mode

Operating Mode may be selected from a drop-down list. Choices are:

Compatibility Mode [default], IND560 Emulation

Depending on the Byte Order selection (refer to section 3.8.9.6.3, **Byte Order**, below), **Compatibility Mode** will provide the same discrete mode byte order arrangements as the METTLER TOLEDO IND131/331 and IND780 terminals. If **IND560 Emulation** is selected, the transmitted bytes in discrete mode will match the existing IND560 byte order determined by the Byte Order selection. Byte order arrangement in the IND560 terminals does not match that of IND131/331 and IND780.

IND560 Emulation mode should be chosen only when replacing an IND560 **and** the programming within the PLC will not be modified.

4.8.1.2.2. Format

Select the Format: Integer (default), Divisions or Floating Point. Changing the Format will delete any existing Message Slots.

■ **Note:** METTLER TOLEDO recommends using Floating Point format whenever possible since this is the data format maintained within the terminal. Use of Integer / Division mode requires that the internal data be converted from Floating Point, which can give small errors during conversion.

4.8.1.2.3. Byte Order

Available selections are Standard, Byte Swap, Word Swap (default), and Double Word Swap.

4.8.1.2.4. Message Slots

Select 1, 2, 3 or 4 slots.

4.9. Troubleshooting

■ **Note:** Some PLCs, such as Micrologix and SLC PLCs cannot exchange cyclic (class 1) messages. If these PLCs are used, they must use Explicit (class 3) Messaging to communicate with the IND570.

If the IND570 does not communicate with PLC, do the following:

- Confirm that the IND570 can respond to a Ping on the Network. If it doesn't, then check the wiring and network connections.
- Use the Status LED's (described below) to diagnose and correct specific Network error conditions such as IP address conflicts.
- Confirm that the IND570 settings for data type, I/O size and IP address assignment match those in the PLC, and that each IND570 has a unique IP address.
- Use the provided Add On Profile (AOP) when possible to simplify the setup in the PLC.
- Check the Electronic Keying in the Add On Profile (AOP) to confirm that the firmware revision specified matches the firmware of the Ethernet/IP module installed in the IND570. If necessary, change the firmware version specified in the AOP, or change the Electronic Keying designation from "Exact Match" to "Compatible Module" or "Disable Keying".
- Confirm that the PLC's Ethernet/IP module firmware is up to date: the IND570's module contains the latest protocol updates, which means that it may not connect to PLCs using older firmware.
- If the PLC interface PCB was changed from another type, like DeviceNet or ControlNet, a master reset of the IND570 should be performed. Contact Mettler Toledo service for assistance.
- Contact METTLER TOLEDO service for replacement of the EtherNet/IP interface.

4.9.1. Status LEDs

The EtherNet/ IP interface card has four status LEDs indicators to indicate communication and fault status of the card. Figure 4-3 indicates the location and functions of these LEDs. Table 4-4 explains the meaning of the indicators.

The Primary status is the condition most probably indicated by the LED. The Secondary status represents other conditions which may occur in the EtherNet/IP module or EtherNet/IP network, in conjunction with the Primary status. Both Primary and Secondary statuses are provided, to better support troubleshooting efforts.

For example, when the Link Activity LED (1) is solid green, the primary indication is that IND570 EtherNet/IP module is plugged into the network. However, it is possible that the PLC is physically connected to the network, but not communicating with it.

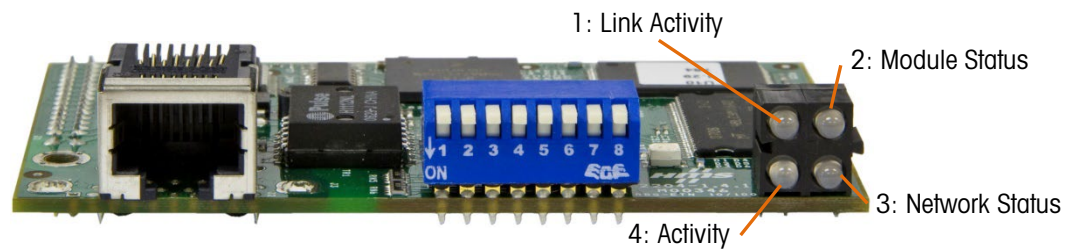


Figure 4-3: EtherNet/IP Status Indicator LEDs

Table 4-4: EtherNet/ IP LEDs Status Indications

LED #	State	IND570 EIP Primary Status	Secondary Status
1 - Link Activity	Off	Cable disconnected at terminal	No link (or no power)
	Solid green	Cable connected at terminal	Disconnected cable at PLC PLC in Program mode Terminal in setup mode
2 - Module Status	Off	No power	-
	Solid green	Connected and operating normally	Terminal in setup mode
	Flashing green	Cable disconnected at PLC Cable disconnected at terminal PLC in Program mode	-
3 - Network Status	Off	No IP address No power	-
	Solid green	EtherNet/ IP network connection properly established	PLC in Program mode Terminal in setup mode
	Flashing red	Cable disconnected at PLC Cable disconnected at terminal	-
4 - Activity	Off	Cable disconnected at terminal	No Ethernet activity, or no power
	Blinking green	Cable connected at terminal Cable disconnected at PLC PLC in program mode Terminal in setup mode	-

4.10. Programming Examples

The following Figures show sample screen images of ladder logic programming examples for RSLogix 5000 software (version 20).

- **Note:** The Utilities folder of the IND570 documentation CD contains complete versions of the examples. These screen images are provided for illustrative purposes only.

The Ethernet/IP module for the IND570 is capable of communicating at either 10 or 100 Mbps. It communicates using Full Duplex mode, and is compatible with the Auto Negotiate setting used by many PLC's and Ethernet Switches.

The sample program includes the following example Module definitions for 1 and 4 message slot Terminals using the Integer (or Division) and Floating Point format.

For convenience, the sample modules shown below can be copied into your own program, which will in turn copy their associated descriptors. The 4 Message Slot versions of each module can then be reduced to other message slot sizes so that the programming software will maintain the descriptors for the updated modules.

4.10.1. Communication Module Profiles

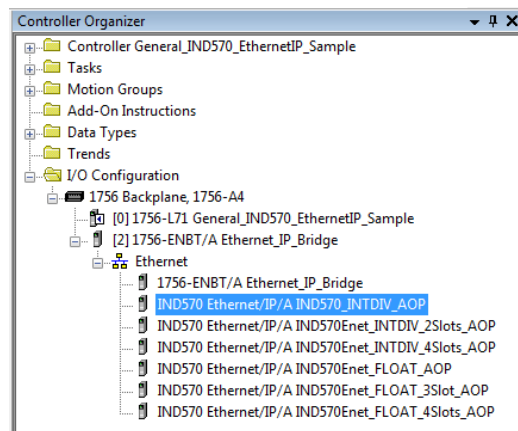


Figure 4-4: Integer Mode Sample Communication Module

In Figure 4-4, the sample Integer Mode modules are circled. Note that all of them are using the IND570 AOP, where the second two are for 2 and 4 slot IND570 configurations.

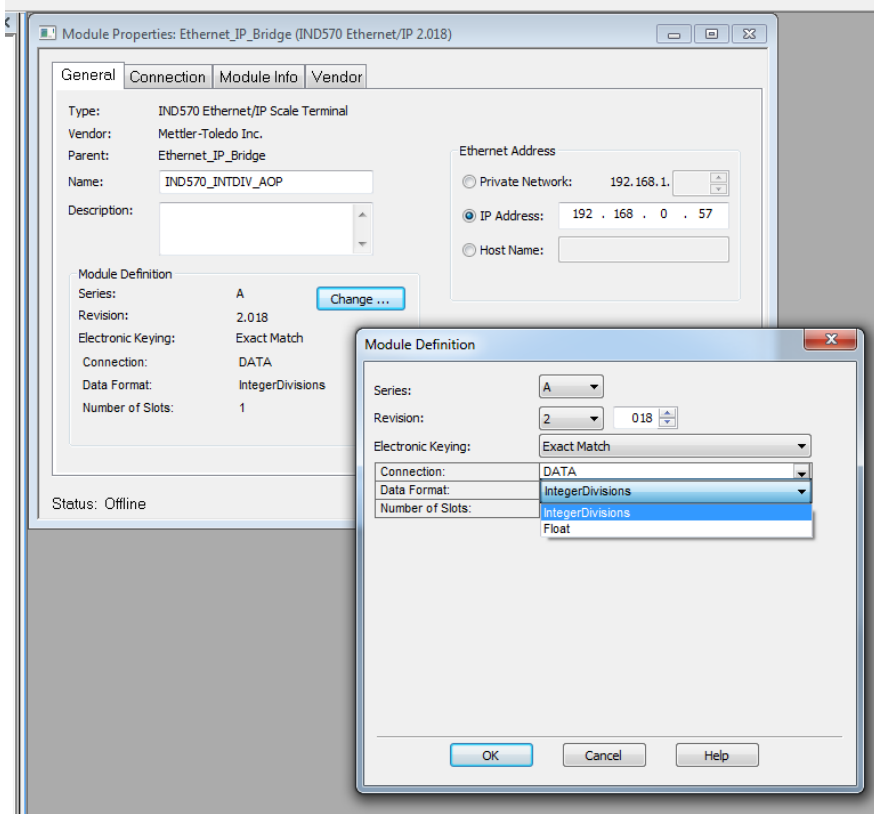


Figure 4-5: Integer Mode Communication Module Configurations

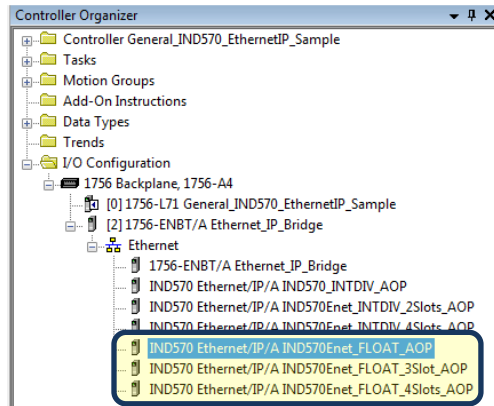


Figure 4-6: Floating Point Mode Sample Communication Module

In Figure 4-6, the sample Floating Point Mode modules are circled. Note that the first one uses 1 message slot, second is 3 message slots, and the last one is for 4 message slots.

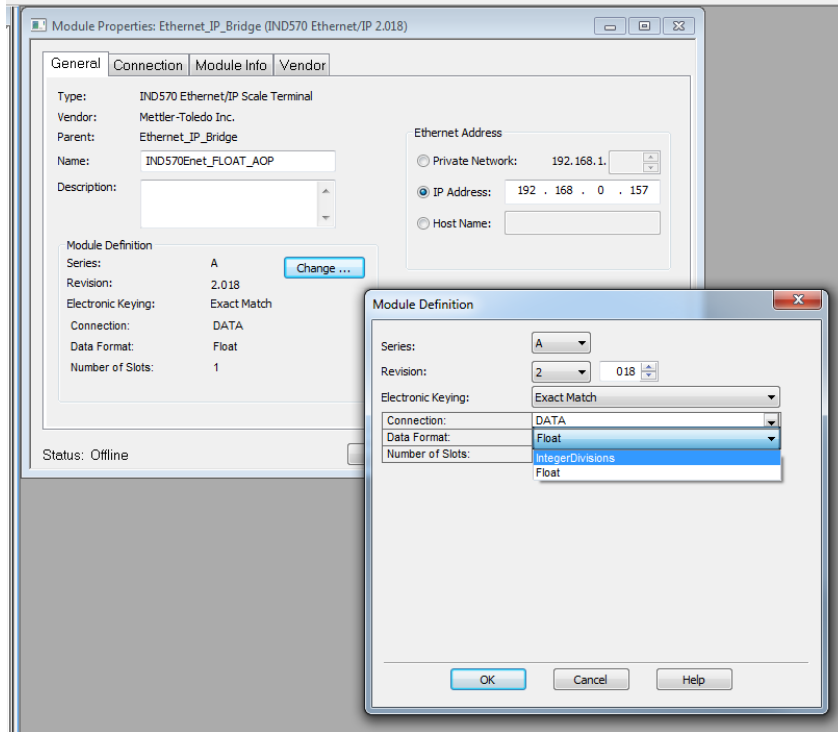


Figure 4-7: Floating Point Mode Communication Module Configuration

4.10.2. General Programming Notes

The following principles should always be applied to guarantee the validity of the data before using it in a process. Note that there are different principles for the different modes (Floating Point versus Integer or Divisions).

For Floating Point Mode, data being read from the Terminal should be filtered with the Data_OK bit and the two Data Integrity bits as shown in Figure 4-8.

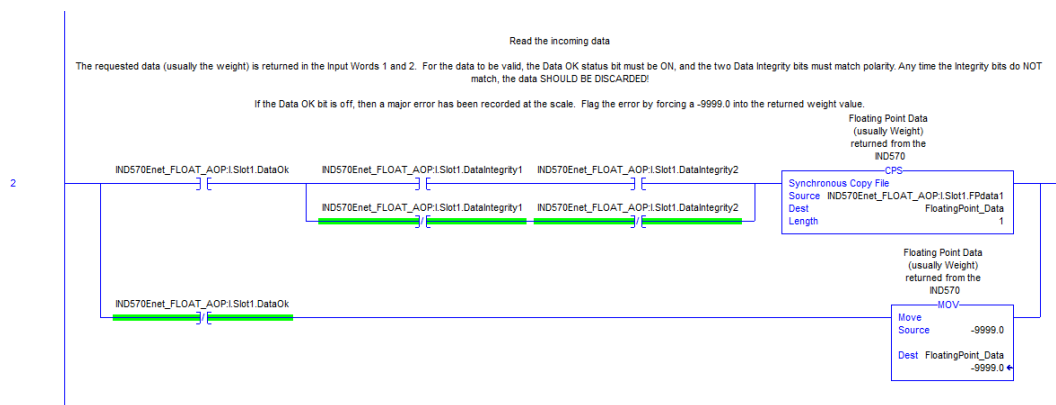


Figure 4-8: Floating Point Mode Cyclic Data Read

Filtering the data in this way makes sure that the Terminal is in a valid operational state (Data_OK = 1) and that the Analog Update from the Load Cell has properly completed before the data was read (Integrity 1 = Integrity 2). Failing to perform these checks can result in invalid data being used by the PLC program.

For Integer or Division Mode, a similar filter should be applied as shown in Figure 4-9.

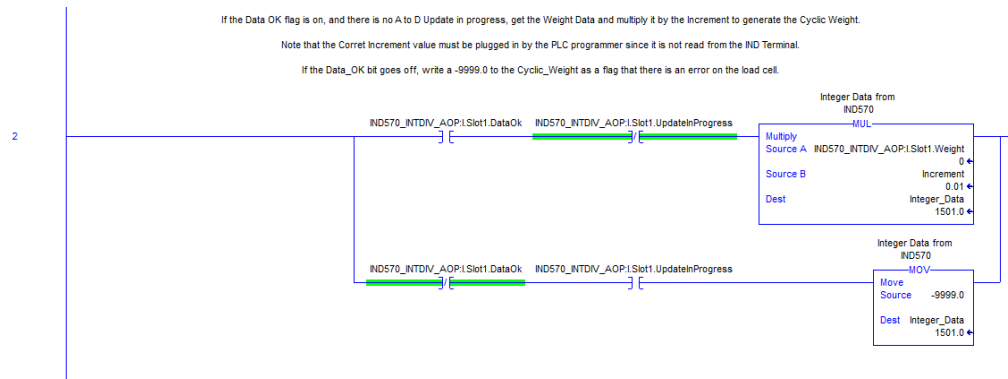


Figure 4-9: Integer Mode Cyclic Data Read

In this case, the data is filtered with the Data_OK bit and the Update_In_Progress bit to make sure that the data coming back from the terminal is valid. From there, it is converted into a Floating Point value by multiplying it by the hard-coded Increment size for the terminal to properly place the decimal point.

4.10.3. Shared Data Access Overview

Shared Data is a memory area in the terminal that contains many different kinds of information ranging from standard weight data to system variables and Task Expert application data. Providing access to this information to the PLC can be an immense help when coordinating the process with functions occurring in the Terminal.

For Ethernet/IP, access to Shared Data is accomplished using discrete messages (otherwise known as explicit, asynchronous, or Class 3 messages).

In order to access Shared Data, a program must provide the following information to the Read and Write message instructions:

- Class Code
- Instance Number
- Attribute Number
- Length

This information can be found in the **IND570 Shared Data Reference** (part number 30205337) for each Shared Data variable. The example in Figure 4-10 shows how to find this information for a 'WT' type Shared Data variable.

Scale Functionality
Dynamic Scale Weight (WT)

Access: "Read Only." Access level is not customizable.
Class Code: wt Data Type: D

ControlNet Class Code: 68 hex

Instances: 5
Instance 1 - 4 = Scale platforms 1 - 4
Instance 5 = Sum scale.

Attributes:
Note: The last two digits of each shared data variable is its attribute.

wt--00	Composite wt block	Struct	na	Composite of entire block
wt--01	Displayed Gross Weight	S13	rt	
wt--02	Displayed Net Weight	S13	rt	When user has enabled MinWeigh, the first character contains an "*" when the MinWeigh conditions are not met.
wt--03	Weight Units	S4	rt	lb pounds, kg kilograms, grams, oz ounces, oztroy, dwt pennyweights, metric tons, ton, or custom units name
wt--04	Displayed Aux Gross Weight	S13	rt	
wt--05	Displayed Aux Net Weight	S13	rt	
wt--06	Aux Weight Units	S7	rt	lb pounds, kg kilograms, grams, oz ounces, lb-oz pounds & ounces, oztroy, ounces, dwt pennyweights, metric tons, ton, or custom units name
wt--07	Rate Period	S2	rt	No, Sec, Min, Hour
wt--08	Displayed Rate	S13	rt	

Figure 4-10: Shared Data Class, Instance, Attribute and Length

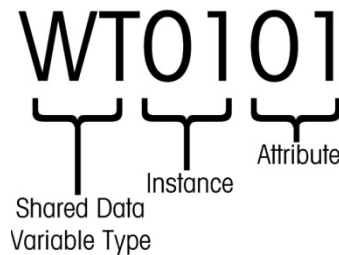


Figure 4-11: Construction of a Shared Data Variable Name

This information can help you in setting up your program to read or write the Shared Data variables that you need to access.

4.10.3.1. Shared Data Variable Name Instance Number

The 'Instance' is used in other METTLER TOLEDO terminals to refer to multiple instruments (scales or flowmeters) that may be serviced by the terminal. In the case of the IND570, there will only be one instrument (one scale), so most of the time the Instance number will be "01" when used in the Shared Data Variable name. There are exceptions to this rule, so attention must be paid to the details of the variable spelled out in the **Shared Data Reference**.

4.10.4. Shared Data Access Program Details

Since the type of data sent to, and read back from the IND570 is **not** defined by the communications mode selected (Integer, Divisions, or Floating Point), the method to access Shared Data in the IND570 Terminal is identical for both the Floating Point and Integer Modes using ControlNet.

Figure 4-12 shows a rung of logic that sends a trigger to the IND570 to tare the scale. The configuration of the message instruction is shown below the rung.

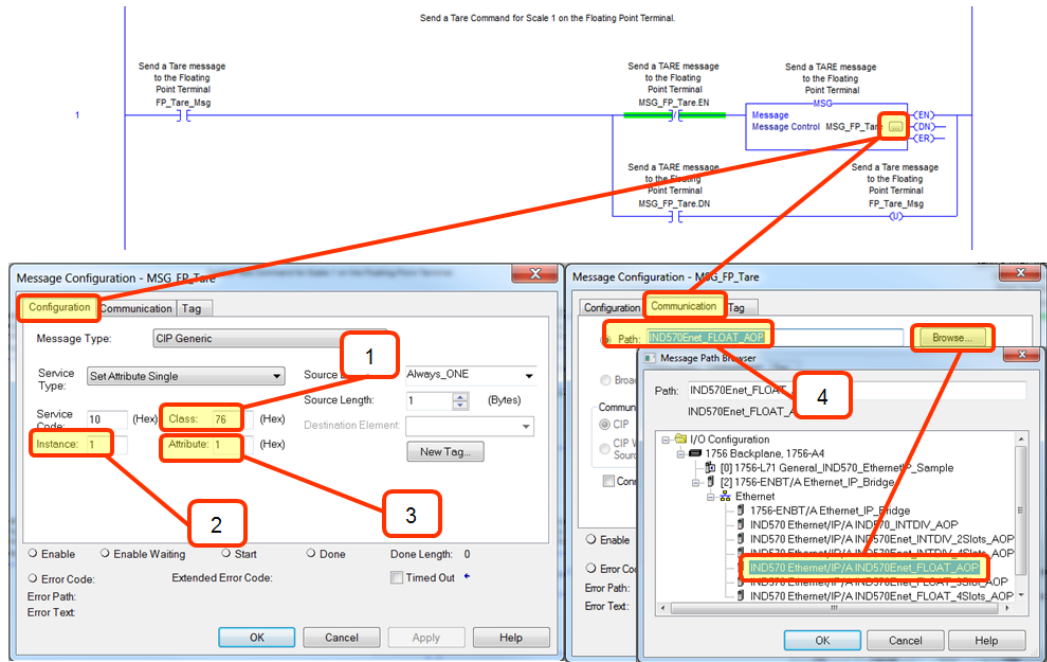


Figure 4-12: Shared Data Write to wc0101

Note that the message type is a CIP Generic, with a Service Type “Set Attribute Single” (service code 10).

Key to Figure 4-12

1. Class Code in Hexadecimal for the Shared Data Variable WC0101, found in the Shared Data Reference Manual.
2. Instance number for Shared Data Variable WC0101, found in the Shared Data Reference Manual.
3. Attribute in Hexadecimal for the Shared Data Variable WC0101, found in the Shared Data Reference Manual.
4. The Path to the ControlNet Node that the message will be sent. The path can be selected by clicking the Browse button and selecting it from the list.

Other commands such as Clear, Zero, and Print would be sent in an identical way, changing the attribute to match the desired command.

Figure 4-13 shows a rung of logic that triggers a read of the rounded gross weight on the terminal, which maps to Shared Data Variable WT0101. The configuration of the message instruction, along with the data area where the response will be stored, is shown below the rung.

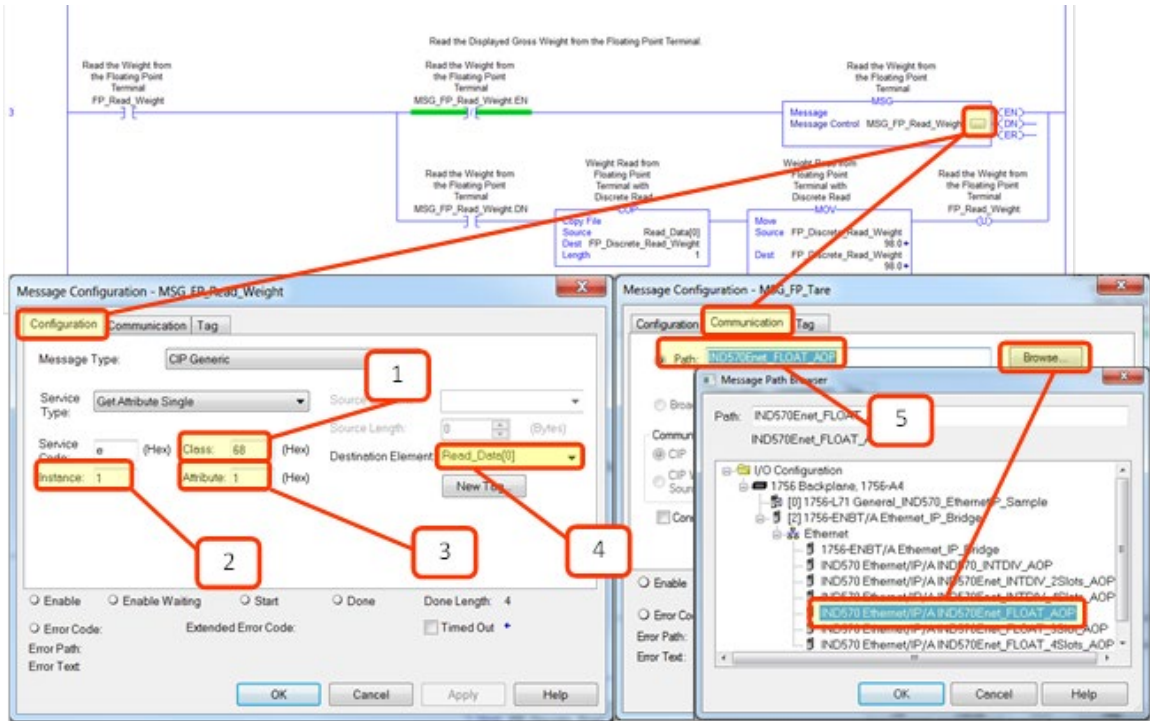


Figure 4-13: Shared Data Read from wt0101

The message type is a CIP Generic, with a Service Type “Get Attribute Single” (service code e).

Key to Figure 4-13

1. Class Code in Hexadecimal for the Shared Data Variable WT0101, found in the Shared Data Reference Manual.
2. Instance number for Shared Data Variable WT0101, found in the Shared Data Reference Manual.
3. Attribute in Hexadecimal for the Shared Data Variable WT0101, found in the Shared Data Reference Manual.
4. The Variable Tag to be used to store the data returned from the IND570. Note that the destination element must reference the array index [0] in order to correctly place the data in its destination.
5. The Message Path is then needed to connect the MSG instruction to the proper device.

The IND570 returns 4 bytes of data into the Read_Data array, which represents an IEEE 754 type Single Floating Point number. The program then converts those 4 bytes into a ‘REAL’ type number by copying them into the tag FP_Discrete_Read_Weight. Note that the MOV instruction immediately after the copy is for convenience only so that the programmer can easily see the value that was returned. Note that the message instruction could have easily returned the value directly into the REAL typed variable instead of the Byte Array. Returning the data into the Byte Array gives the

programmer some flexibility for swapping bytes and words around as needed, and also provides some useful troubleshooting information should the process fail for some reason.

5 Modbus TCP

5.1. Modbus TCP Interface

Figure 5-1 shows the EtherNet/IP – Modbus TCP Option board, with its port, DIP switches and status lights indicated. Note that the module's address is set in software, and the DIP switches must all be set to OFF.

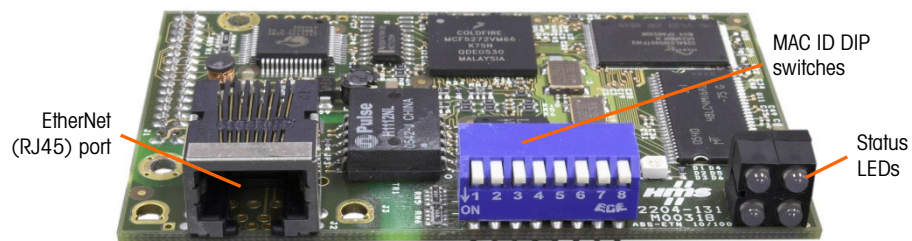


Figure 5-1: EtherNet/IP - Modbus TCP Option Board Components

5.2. Overview

Modbus protocol is a messaging structure developed by Modicon in 1979. It is used to establish master-slave/client-server communication between intelligent devices. It is an open standard network protocol, widely used in the industrial manufacturing environment. Modbus can be used in multiple master-slave applications to monitor and program devices; to communicate between intelligent devices and sensors and instruments; and to monitor field devices using PCs and HMIs. Modbus is also an ideal protocol for RTU applications where wireless communication is required.

TCP/IP is an Internet transport protocol of that consists of a set of layered protocols, providing a reliable data transport mechanism between machines. The open Modbus TCP/IP specification was developed in 1999. The ModbusTCP protocol takes the Modbus instruction set and wraps TCP/IP around it.

5.3. Modbus TCP Characteristics

- User-programmable IP addressing
- Capability for bi-directional discrete mode communications (Cyclic Messaging) of weight or display increments, status, and control data between the PLC and the IND570.

5.3.1. Specifications

Network Type	Ethernet-TCP/IP based simple Client/Server network.
Topology	Star, tree or line structures; all topologies that can be implemented with standard Ethernet technology, including switched networks, are applicable.
Installation	Standard 10, 100 Mbit/s Ethernet technology based on copper cables, fiber optic or wireless standards can be used. The IND570 Modbus TCP option provides an RJ-45 Ethernet port connection.
Speed	10, 100 Mbit/s.
Max. stations	Nearly unlimited.
Network features	Client/Server network based on standard Ethernet technology and TCP/UDP/IP protocols in Layer 3-4.
User Organization	Modbus-IDA user Group.

5.3.2. Communications

The IND570 terminal uses component parts to ensure complete compatibility with the Modbus TCP network. The PLC recognizes the IND570 terminal as a generic Modbus TCP device.

Each Modbus TCP option connected to the network represents a physical IP Address. The connection is made via the board's RJ-45 connector – see Figure 5-1.

The wiring between the PLC and the IND570 Modbus TCP connection uses Ethernet twisted pair cable. The cable installation procedures and specification, including distance and termination requirements, are the same as recommended by Schneider Electric (Modicon) for the Modbus TCP network.

5.3.3. IP Address

Each Modbus TCP interface option represents one physical IP Address. This address is chosen by the system designer, and then programmed into the IND570 terminal and PLC. The IND570 terminal's address is programmed at **Communication > PLC Interface > EtherNet/IP-Modbus TCP**. The IND570 IP Address entry must be unique for each IND570.

5.3.4. Supported Data Formats

For a general account of Data Format types, please refer to Appendix C, **Common Data Features**.

5.4. Data Definition

5.4.1. Data Integrity

The IND570 has specific bits to allow the PLC to confirm that data was received without interruption and that the IND570 is not in an error condition. It is important to monitor these bits. Any PLC code should use them to confirm the integrity of the data received by the IND570. Refer to the data charts in Appendix A and B for specific information regarding the Data OK, Update in Progress and Data Integrity bits and their usage.

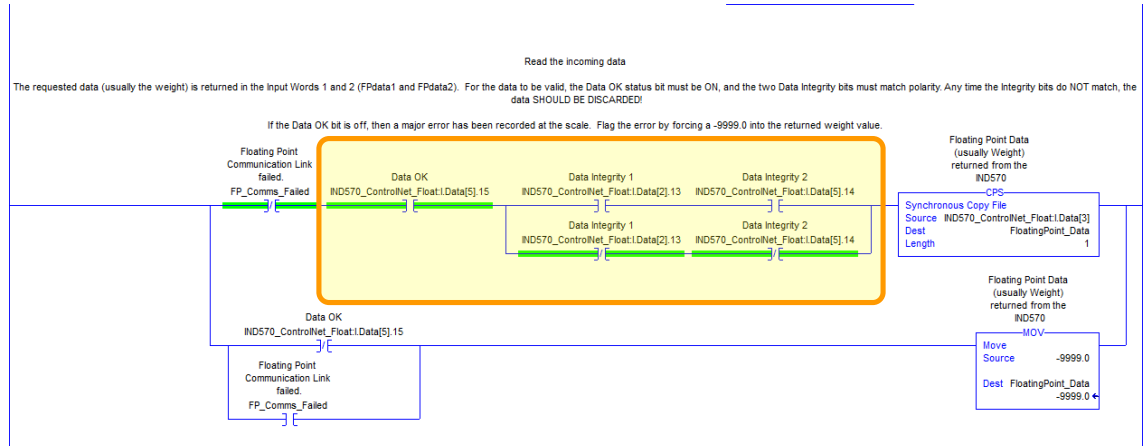


Figure 5-2: Correct Filter Ladder Logic for Floating Point Data

5.4.2. Discrete Data

Please refer to Appendix C, **Common Data Features** for a description of discrete data, and to Appendix A and Appendix B for a detailed description of the data available in each format, in order to determine which is most suitable.

5.4.3. Byte Order

For a general account of byte ordering, please refer to Appendix C, **Common Data Features**.

5.4.4. Register Mapping

The memory of the Modbus TCP Kit option board is mapped as shown in Table 5-1. The read and write areas of memory are offset by 1024. In a Quantum PLC, the PLC would read data from the IND570 starting at 400001 and would write data to the IND570 starting at register 401025.

Table 5-1: Modbus TCP-IP PLC Input and Output data map

Register #	Area	Offset In Area
1	Read Data (from IND570)	0000h...0001h
2		0002h...0003h
3		0004h...0005h
4		0006h...0007h
.....	
1024	Write Data (To IND570)	0000h...0001h
1025		0002h...0003h
1026		0004h...0005h
1027		0006h...0007h
.....	

5.4.5. Message Slots

The IND570 can be configured for up to 4 message slots for discrete data transfer, in Integer, Divisions and Floating Point Data Formats. Each message slot is assigned to an internal local or remote scale.

The number of Message Slots is set up in **Communication > PLC > Data Format** setup menu.

The integer and division formats provide (two 16-bit words of input and two 16-bit words of output data) per Message Slot. Each Message Slot's first input word provides scale weight data and the input weight data may be selected by the PLC using the Message Slot's second output word bit 0, bit 1 and bit 2. Table 5-2 and Table 5-3 provide input and output usage information.

Table 5-2: Modbus TCP PLC I/O Data and Data Usage (Integer and Division)

Input Data to PLC				Output Data from PLC			
Register Address	Description		Input Size	Output Size	Description	Word Offset	
400001*	Integer Value	Msg Slot 1	2 Words (4 Bytes)	2 Words (4 Bytes)	Load Integer Value	401025	
400002	Scale Status				Command	401026	
400003	Integer Value	Msg Slot 2	4 Words (8 Bytes)	4 Words (8 Bytes)	Load Integer Value	401027	
400004	Scale Status				Command	401028	
400005	Integer Value	Msg Slot 3	6 Words (12 Bytes)	6 Words (12 Bytes)	Load Integer Value	401029	
400006	Scale Status				Command	401030	
400007	Integer Value	Msg Slot 4	8 Words (16 Bytes)	8 Words (16 Bytes)	Load Integer Value	401031	
400008	Scale Status				Command	401032	

* 4000, 40001 and 400001 are PLC processor memory-dependent. Refer to the PLC documentation for I/O mapping.

The floating point format provides four 16-bit words of input data and three 16-bit words of output data per Message Slot. Table 5-3 provides details.

Table 5-3: Modbus TCP PLC Floating Point I/O Words

Input Data to PLC				Output Data from PLC			
Register Address	Description		Input Size	Output Size	Description	Register Address	
400001	Command Response	Message Slot 1	4 Words (8 Bytes)	4 Words (8 Bytes)	Reserved	401025	
400002	4-Byte Floating Point Value				Command	401026	
400003	Scale Status				4-Byte Floating Point Load Value	401027	
400004	Command Response				Command	401028	
400005	4-Byte Floating Point Value	Message Slot 2	8 Words (16 Bytes)	7 Words (14 Bytes)	Command	401029	
400006	Scale Status				4-Byte Floating Point Load Value	401030	
400007	Command				4-Byte Floating Point Load Value	401031	
400008	Command	Message Slot 3	12 Words	10 Words (20 Bytes)	Command	401032	
400009	4-Byte Floating Point Value				4-Byte Floating Point Load Value	401033	

Input Data to PLC			
Register Address	Description		Input Size
	Response		(24 Bytes)
400010	4-Byte Floating Point Value		
400011	4-Byte Floating Point Value		
400012	Scale Status		
400013	Command Response	Message Slot 4	16 Words (32 Bytes)
400014	4-Byte Floating Point Value		
400015	4-Byte Floating Point Value		
400016	Scale Status		

Output Data from PLC			
Output Size	Description		Register Address
		Point Load Value	401034
		Command	401035
13 Words (26 Bytes)	Message Slot 4	4-Byte Floating Point Load Value	401036
			401037

5.4.6. Integer and Division

When one of these formats is selected, the IND570 will have two 16-bit words for input data and two 16-bit words for output data in each Message Slot. The PLC's input data will contain one 16-bit word for the scale's weight information and one 16-bit word for bit encoded status information for each Message Slot. The IND570 will send specific weight data to the PLC input data based on the data it receives from the PLC's output data. The PLC's output words consist of one 16-bit integer value, which may be used to download a tare or target, and one 16-bit word for bit encoded command information.

Appendix A provides detailed information on the integer and division data formats.

5.4.7. Floating Point

For a general account of Floating Point operation, data format and compatibility, please refer to Appendix B, **Floating Point Format**.

5.5. Controlling Discrete I/O Using a PLC Interface

Please refer to Appendix C, **Common Data Features**.

5.6. Software Setup

The IND570 terminal automatically detects the presence of a Modbus TCP Kit option board if one is installed. When the option is detected, the IND570 terminal enables the Modbus TCP parameters in a program block under **Communications > PLC Interface > Ethernet/IP**. Figure 5-3 shows the Modbus TCP setup block.

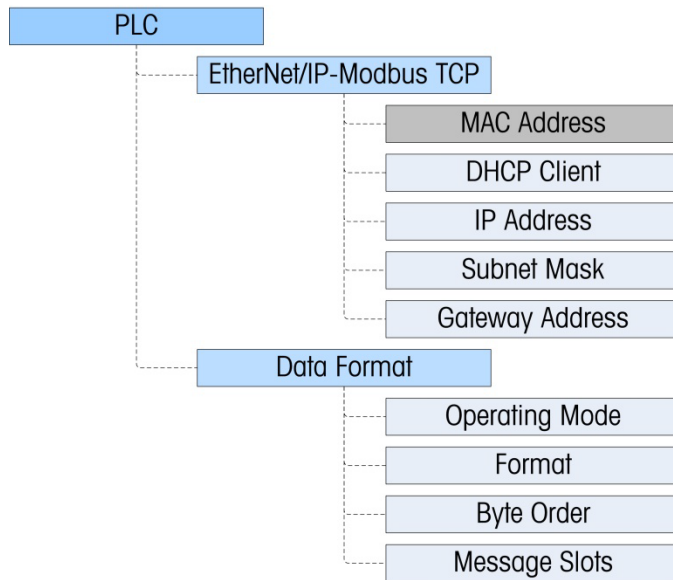


Figure 5-3: The Modbus TCP Setup Block

5.6.1. Modbus TCP and Data Format Setup Blocks

5.6.1.1. Modbus TCP Setup

Setup for Modbus TCP is the same setup block used for EtherNet/IP setup. The Modbus setup, found at **Communication > PLC > Ethernet/IP-Modbus TCP**, allows the Modbus TCP interface's IP address, subnet mask and gateway address to be specified. If DHCP is selected, the IND570 will carry out a power cycle and will automatically populate the IP Address, Subnet Mask and Gateway Address fields with information received from the network.

The MAC address is displayed, but cannot be modified.

5.6.1.2. Data Format Setup

In Setup, navigate to **Communication > PLC Interface > Data Format**. The following fields are available for Modbus TCP.

5.6.1.2.1. Operating Mode

Operating Mode may be selected from a drop-down list. Choices are:

Compatibility Mode [default], IND560 Emulation

Depending on the Byte Order selection (refer to section 3.8.9.6.3, **Byte Order**, below), **Compatibility Mode** will provide the same discrete mode byte order arrangements as the METTLER TOLEDO IND131/331 and IND780 terminals. If **IND560 Emulation** is selected, the transmitted bytes in discrete mode will match the existing IND560 byte order determined by the Byte Order selection. Byte order arrangement in the IND560 terminals does not match that of IND131/331 and IND780. IND560 Emulation mode should be chosen only when replacing an IND560 **and** the programming within the PLC will not be modified.

- 5.6.1.2.2. Format
Select the Format (Integer [the default], Divisions, Floating Point or Application). Changing the Format will delete any existing Message Slots.
- 5.6.1.2.3. Byte Order
Available selections are Standard, Byte Swap, Word Swap (default), and Double Word Swap.
- 5.6.1.2.4. Message Slots
Select 1, 2, 3 or 4 slots.

5.7. Troubleshooting

If the IND570 does not communicate with PLC, do the following:

- Check wiring and network termination (see LED Indications below for Network and Modbus TCP Module status).
- Confirm that the IND570 settings for data type and IP Address assignment match those in the PLC and that each IND570 has a unique IP address.
- Confirm that the proper PLC address offsets are being used for reads and writes.
- If the PLC interface PCB was changed from another type, like ControlNet or DeviceNet, a master reset of the IND570 should be performed. Contact METTER TOLEDO service for assistance.
- Contact METTLER TOLEDO service for replacement of the ControlNet interface.

5.7.1. Status LEDs

The EtherNet/IP – Modbus TCP interface card has four status LEDs indicators to indicate communication and fault status of the card. Figure 5-1 indicates the location of these LEDs, and Figure 5-4 shows the array of the LEDs on the card. Table 5-4 explains the meaning

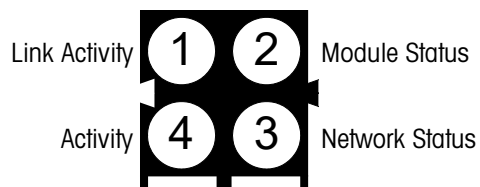


Figure 5-4:– Modbus TCP Board Status LED Array

Table 5-4: EtherNet/IP-Modbus TCP LEDs Status Indications

Network Status	LED State			
	1	2	3	4
Connected	Solid Green	Solid Green	Solid Green	Blinking Green
Disconnected Cable at PLC	Solid Green	Flashing Green	Flashing Red	Blinking Green
Disconnected Cable at Terminal	Off	Flashing Green	Flashing Red	Off
PLC in Program Mode	Solid Green	Flashing Green	Solid Green	Blinking Green
Terminal in Setup Mode	Solid Green	Solid Green	Solid Green	Blinking Green

5.8. Modbus TCP Configuration Example

This demo was set up using Concept Version 2.6 XL, SR1, b (Figure 5-5).

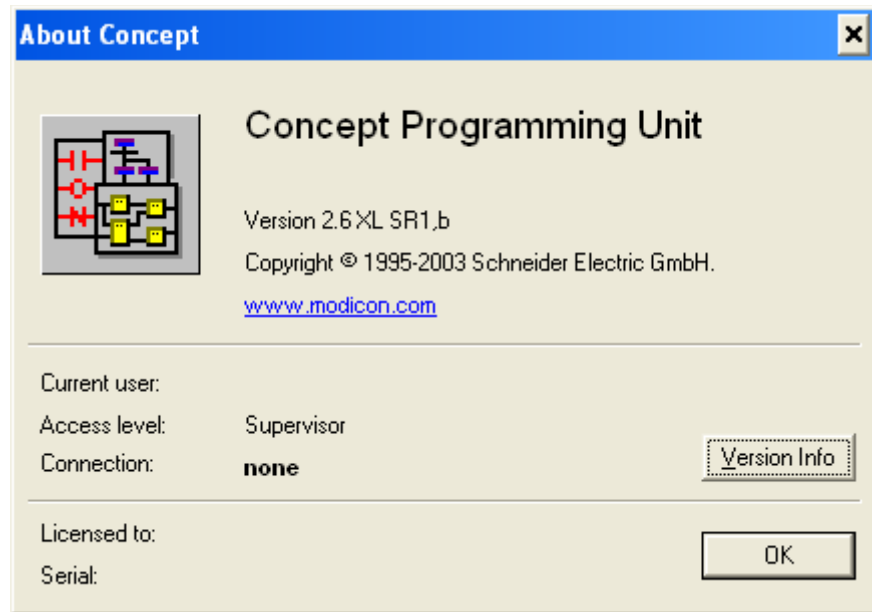


Figure 5-5: Concept Programming Unit Welcome Screen

1. Open a project by accessing the file menu and selecting OPEN, then selecting the project. In this example, the project is named MT_INT.PRJ (Figure 5-6).

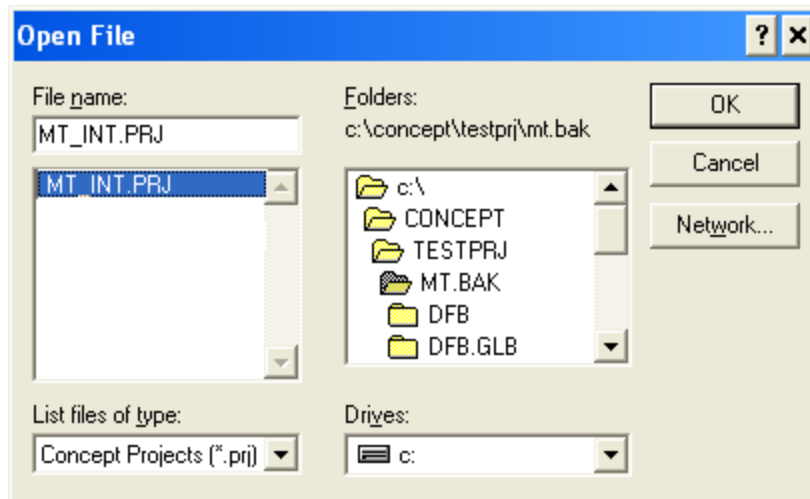



Figure 5-6: Project Selection Dialog

2. Once the project is open, the project browser should appear; if it does not appear, click on  to display it.
3. Next the Network card must be configured. Double click on your project in the project browser. In this example, click on the blue highlighted (Figure 5-7) item to open the PLC Configuration window.

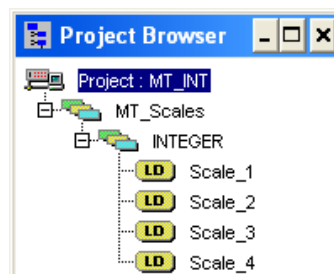


Figure 5-7: Project Viewed in Project Browser

4. The PLC Configuration window (Figure 5-8) will open.

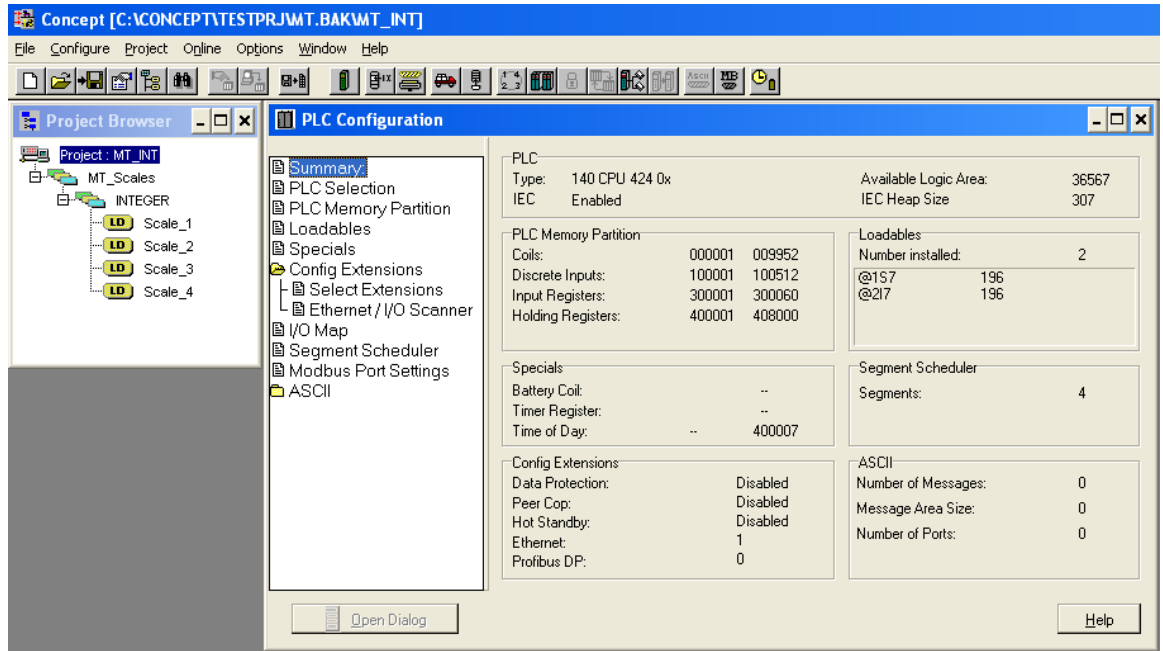


Figure 5-8: PLC Configuration Window

5. Click on the Config Extensions Folder in the center pane, above. The branch will expand to show Ethernet / I/O Scanner. Double click on the Ethernet / I/O Scanner to bring up the details of the Ethernet card (Figure 5-9).
6. Here, the IP addresses must be configured – the PLC’s, that of the IND570 with which it communicates. The data communicated to and from the IND570 is also configured in this window.

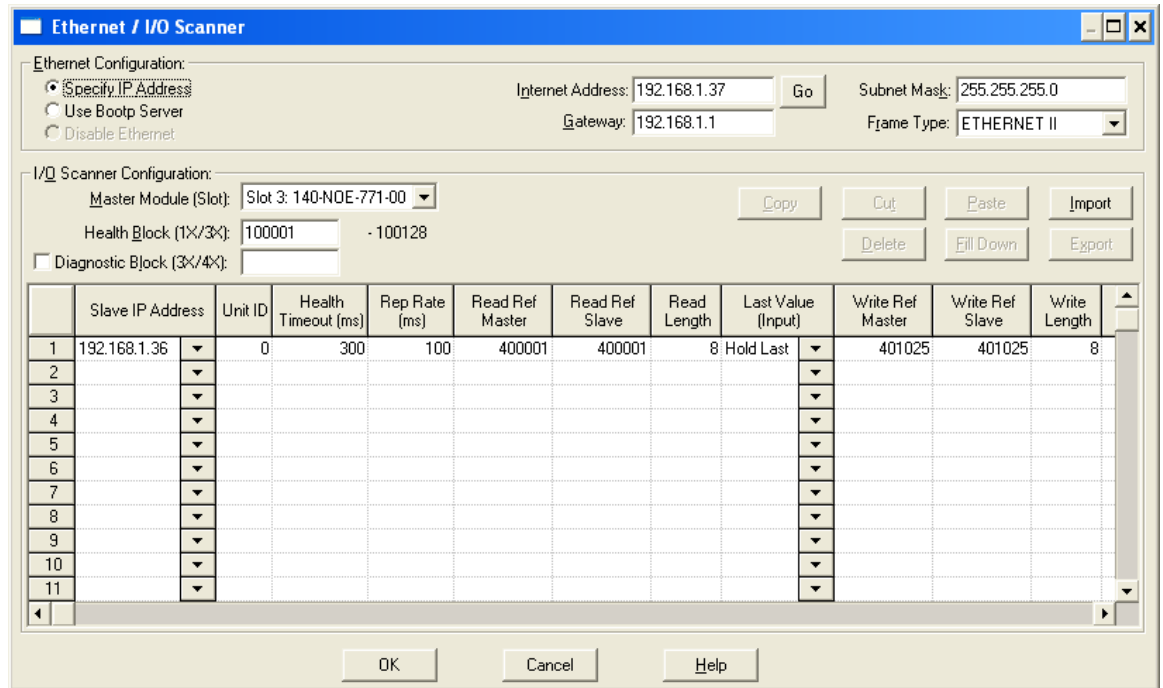


Figure 5-9: Ethernet / I/O Scanner Window

For a more detailed description of each column in the configuration window, click on the Help button (at lower right in Figure 5-9). The following elements must be configured:

Slave IP Address: IP Address of the IND570 terminal's Modbus TCP interface. This value is configured in the IND570 Setup tree at Communication > PLC Interface > EtherNet/IP.

Unit ID: This value is typically 0

Health Timeout:

Rep Rate:

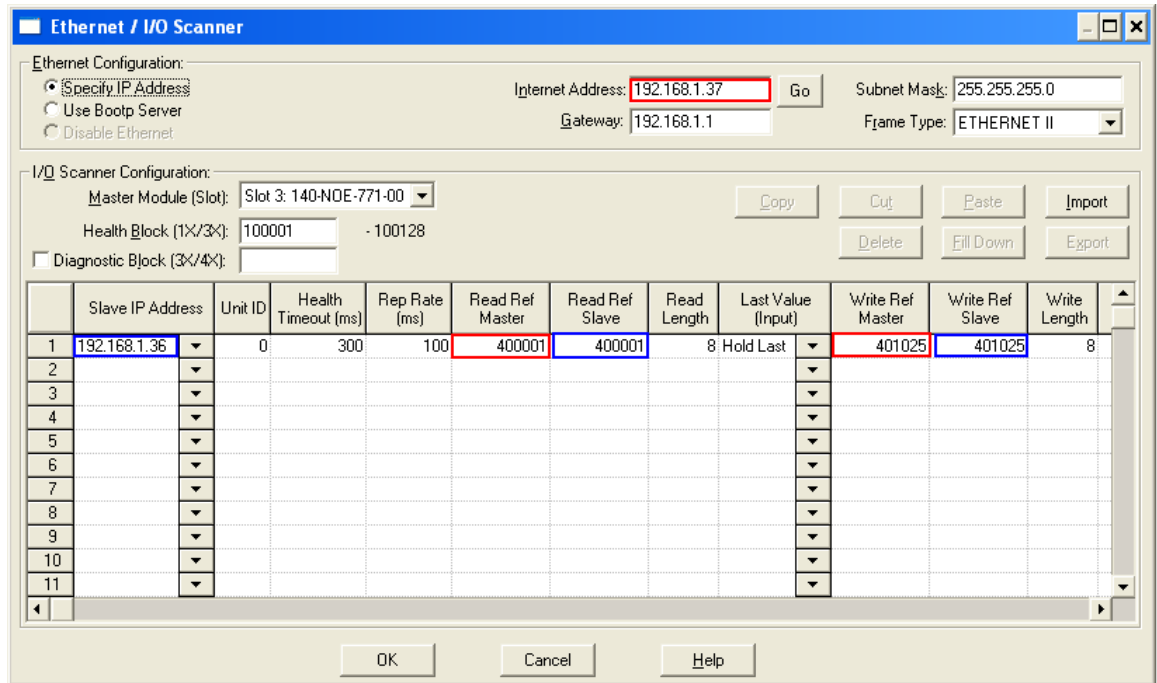
Read Ref Master: The start of PLC registers to which the IND570's information is written. This address ALWAYS is 400001

Read Ref Slave: The start of IND570 register where the scale data is stored. This address can be any value 4XXXXX PLC address.

■ NOTE, data in the Read Ref Slave is read and then stored in the Read Ref Master.

Read Length & Write Length: This is determined by the IND570 settings, and is determined by # of scales, Mode of operation etc. In our example we are using 4 slots in INTEGER Mode. In the IND570 we are reading 16 bytes and writing 16 bytes. When configuring the PLC each 4XXXX register address word consists of 2 bytes of information. This gives a total of 16 bytes / 2 bytes per word, or 8 for Read Length and 8 for Write Length.

7. Both the PLC and the IND570 IP address and address settings must be configured – refer to Figure 5-10. The Ethernet card used on the configuration shown is the 140-NOE-771-00



PLC
IND570 IND780

Figure 5-10: PLC and IND570 Values for Ethernet / I/O Scanner

Examples of how to configure the Modicon Ethernet I/O scanner for various scale configurations are provided below.

5.8.1. Integer and Division Mode Configuration

The IND570 Configured for 4 slots in either integer or division mode. 8 Words are Read into the PLC and 8 words are written to the IND570. Table 5-5 indicates the values for each scale.

	Slave IP Address	Unit ID	Health Timeout (ms)	Rep Rate (ms)	Read Ref Master	Read Ref Slave	Read Length	Last Value (Input)	Write Ref Master	Write Ref Slave	Write Length
1	192.168.1.36	0	300	100	400001	400001	8	Hold Last	401025	401025	8

Figure 5-11: Integer or Division Mode Configuration

Table 5-5: Configuration of Integer or Division Mode

Description	Slot / Scale*	Address in IND570	Format
Read by PLC from 570:			
Weight Data	Slot 1	400001	Int
Status Data	Slot 1	400002	Int
Weight Data	Slot 2	400003	Int
Status Data	Slot 2	400004	Int
Weight Data	Slot 3	400005	Int
Status Data	Slot 3	400006	Int
Weight Data	Slot 4	400007	Int
Status Data	Slot 4	400008	Int
The PLC will write to:			
Data Value to be written	Slot 1	401025	Int
Command Word	Slot 1	401026	Int
Data Value to be written	Slot 2	401027	Int
Command Word	Slot 2	401028	Int
Data Value to be written	Slot 3	401029	Int
Command Word	Slot 3	401030	Int
Data Value to be written	Slot 4	401031	Int
Command Word	Slot 4	401032	Int

* 4001, 40001, 400001 are PLC Memory Dependent.

5.8.2. Floating Point Mode Configuration

The IND570 Configured for 4 slots in Floating Point mode FP. 16 Words are Read into the PLC and 13 words are written to the IND570. Table 5-6 indicates the values for each scale.

	Slave IP Address	Unit ID	Health Timeout (ms)	Rep Rate (ms)	Read Ref Master	Read Ref Slave	Read Length	Last Value (Input)	Write Ref Master	Write Ref Slave	Write Length
1	192.168.1.36	0	300	100	400001	400001	16	Hold Last	401025	401025	13

Figure 5-12: FLP Mode Configuration

Table 5-6: Configuration Floating Point Mode

Description	Slot / Scale*	Address in IND570	Format
Read by PLC from 570:			
Weight Data	Slot 1	400002-400003	Float
Command Ack Register	Slot 1	400001	Int
Status Register	Slot 1	400004	Int
Weight Data	Slot 2	400006-400007	Float
Command Ack Register	Slot 2	400005	Int
Status Register	Slot 2	400008	Int
Weight Data	Slot 3	400010-400011	Float
Command Ack Register	Slot 3	400009	Int
Status Register	Slot 3	400012	Int
Weight Data	Slot 4	400014-400015	Float
Command Ack Register	Slot 4	400013	Int
Status Data	Slot 4	400016	Int
The PLC will write to:			
Reserved	Slot 1	401025	Int
Command Word	Slot 1	401026	Int
Data Value to be Written	Slot 1	401027-401028	Float
Command Word	Slot 2	401029	Int
Data Value to be Written	Slot 2	401030-401031	Float
Command Word	Slot 3	401032	Int
Data Value to be Written	Slot 3	401033-401034	Float
Command Word	Slot 4	401035	Int
Data Value to be Written	Slot 4	401036-401037	Float

* Note that any scale data can be configured to correspond with any slot number. 4001, 40001, 400001 PLC Memory Dependent.

5.8.3. Integer Logic Examples

2 Words of Data are associated with a scale when in integer mode.

- Weight Data for scale 1 is stored in the IND570 in register 400001.

- Status Data for this weight and the IND570 is in register 400002.

5.8.3.1. Read Logic

The 400001 weight data can be read directly by the PLC. However, to understand the 400002 Status data fully some basic logic is needed to break the data Word into Bits.

In concept the use of an INT_TO_WORD instruction will first read the integer value from the IND570 in a form that can be broken into bits. Then once the data is in a word format, a WORD_TO_BIT instruction will complete the process of extracting the individual bits. Figure 5-13 and Figure 5-14 show an example of logic that can be used to read the status word.

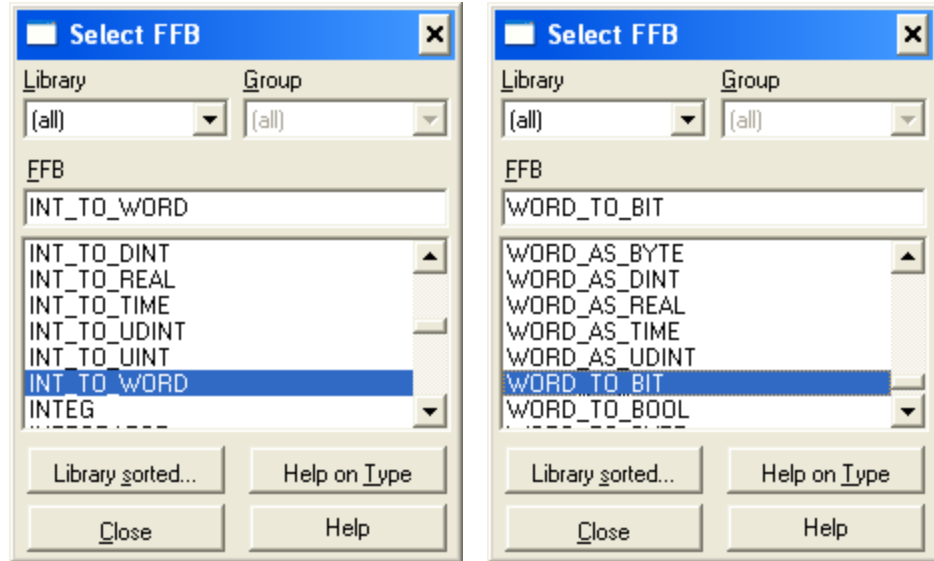


Figure 5-13: Selecting Integer-to-Word (left) and Word-to-Bit (right) Conversions

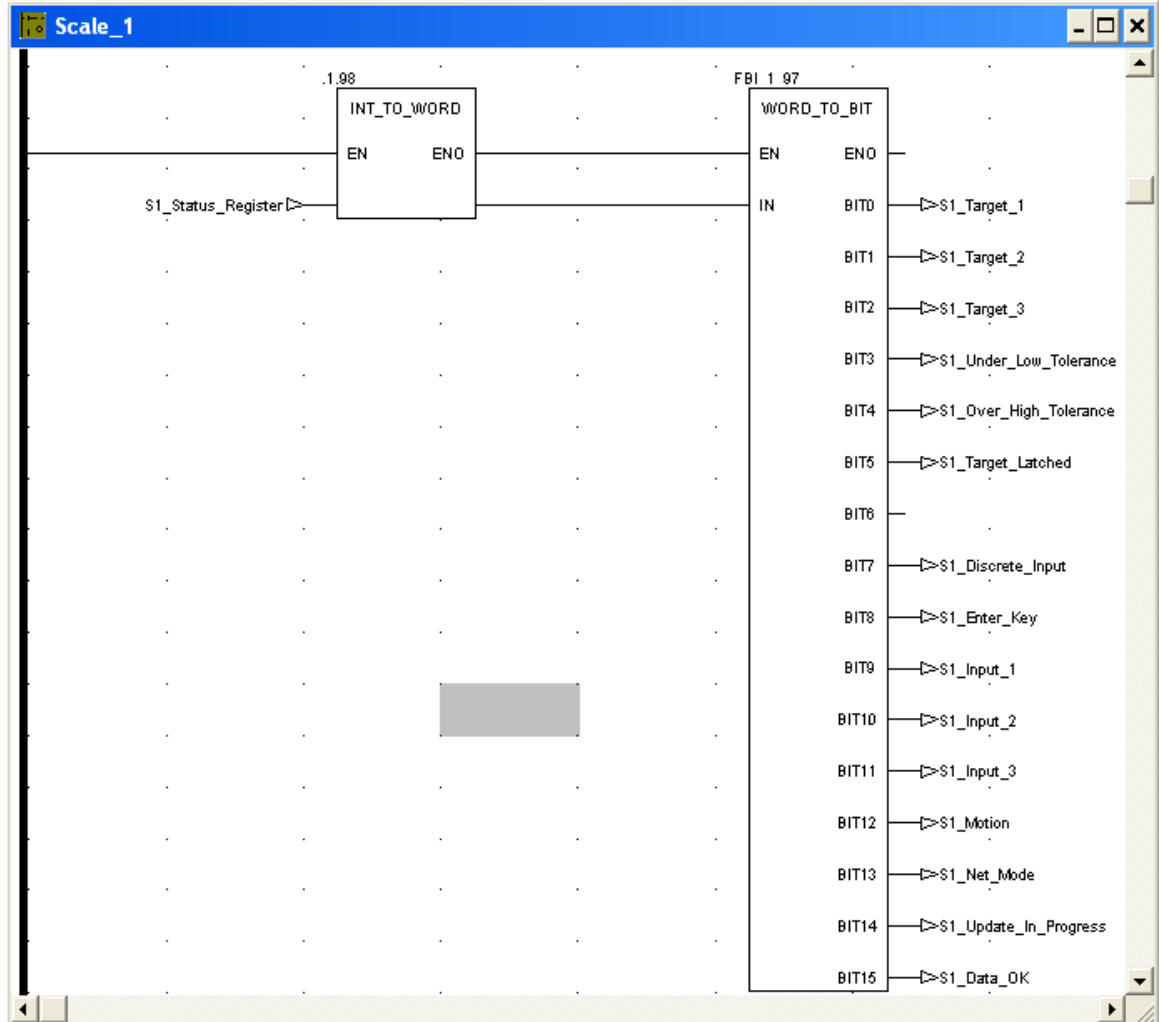


Figure 5-14: Integer-to-Word and Word-to-Bit Logic

5.8.3.2. Write Logic

The 401025 Data Value can be written directly by the PLC. However, to utilize the 401026 command word fully some basic logic is needed to convert the command Bits into a data Word.

In concept, the use of a BIT_TO_WORD instruction will first get the command bits into a WORD value. Next the use of a WORD_TO_INT instruction will complete the process of packing the individual command bits into an integer format that can be written to the IND570. Figure 5-15 shows an example of logic that can be used control the command word.

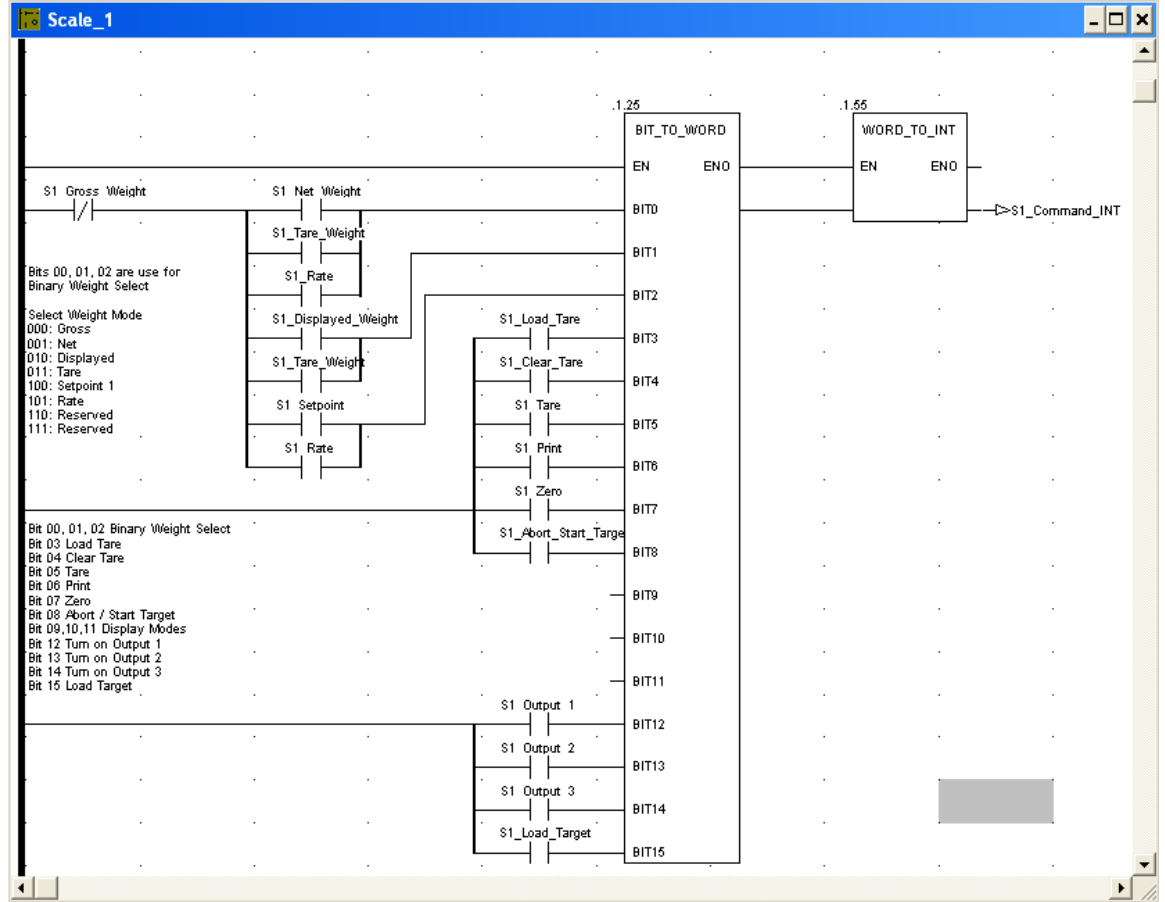


Figure 5-15: Bit to Word and Word to Integer Logic

6 PROFIBUS

The PROFIBUS option card enables the IND570 terminal to communicate to a PROFIBUS DP master according to DIN 19 245. It consists of an IND570 terminal backplane-compatible module and software that resides in the terminal, which implements the data exchange.

The card interfaces to Programmable Logic Controllers (PLCs) and Digital Control Systems (DCSs) that adhere to the PROFIBUS-DP/0 specification. The PROFIBUS appears as a block of I/O on the PROFIBUS network. The size and mapping of the I/O depends on the setup of the PROFIBUS interface within the software of the IND570.

The data mapped within the I/O block is defined as Discrete or Shared Data Variables. Discrete data can be set as Integer, Division, or Floating Point.

Discrete data is sent in groups defined as message slots. The number of message blocks (1 to 4) is setup within the IND570. While the format of each message block is the same, the data received depends on the commands in the block.

In the newer Siemens PLC S7, the PROFIBUS option is integrated into the main PCB.

6.1.1. PROFIBUS Option Kit

There are two different PROFIBUS interface PCBs. The type of option board used depends on the IND570 enclosure in which it is to be used. The two boards differ in the orientation of their connectors. Figure 6-1 shows the harsh version of the option board, Figure 6-2 the panel mount version. Both connectors are active on the harsh version of the option board.

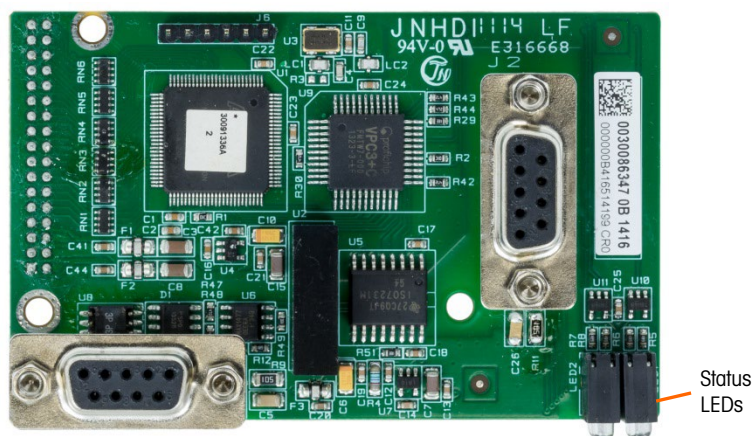


Figure 6-1: PROFIBUS Kit Option Board, Harsh Enclosure Version

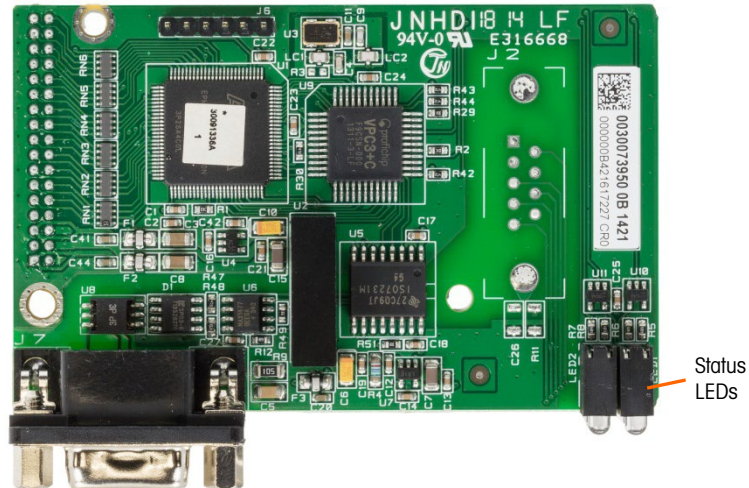


Figure 6-2: PROFIBUS Kit Option Board, Panel Mount Version

6.2. Communications

PROFIBUS is based on a variety of existing national and international standards. The protocol architecture is based on the Open Systems Interconnection (OSI) reference model in accordance with the international standard ISO 7498.

The IND570 terminal supports PROFIBUS-DPVO which is designed for high-speed data transfer at the sensor actuator level – DP means Distributed Peripherals. At this level, controllers such as PLCs exchange data with their distributed peripherals via a fast serial link. The data exchange with these distributed devices is cyclic. The central controller reads the input information from the slaves and sends the output information back to the peripherals.

6.2.1. Node/Rack Address

Each IND570 PROFIBUS option card represents one physical node. The node address is chosen by the system designer and then programmed into the IND570 and the PLC. The IND570's node address is programmed in setup at **Communication > PLC**. The node address and number of input and output words used to communicate between the terminal and the PLC are programmed into the PLC by using its PROFIBUS network configuration software and the IND570's PROFIBUS .GSD files.

The IND570 setup menus allow the selection of the logical rack (node) address, the data format (Integer/Floating Point/Divisions), the number of message slots assigned to the node, and the option of sending and receiving Shared Data. The number of input and output words required and the mapping of the I/O data depends on these selections.

The IND570 PROFIBUS GSD has a block of I/O defined for each of the 14 possible IND570 PROFIBUS combinations. The IND570 terminal will determine the number of input and output words needed for the number of configured message slots and the chosen data format. The PLC must be configured for the same amount of space.

6.2.2. Supported Data Formats

The terminal's PROFIBUS interface has two types of data exchanges: discrete data and shared data. The locations for each of these types of data are predefined by the IND570.

Each message slot selected to pass data through the terminal's PROFIBUS option has its own assigned input and output words for continuous information to and from the PLC. Floating point, integer, and divisions are supported.

Shared data access is only available when the **Setup > Communications > PLC > PROFIBUS > Shared Data** option is **Enabled**. This data is used to pass information that cannot be sent in the discrete data because of its size, or because of process speed limitations. It uses additional input and output word space. The length of shared data value and data type depends on the type of shared data field requested. In no case does it exceed 10 words (20 bytes).

6.3. Data Definition

6.3.1. Data Integrity

The IND570 has specific bits to allow the PLC to confirm that data was received without interruption and that the IND570 is not in an error condition. It is important to monitor these bits. Any PLC code should use them to confirm the integrity of the data received by the IND570. Refer to the data charts in Appendix A and B for specific information regarding the Data OK, Update in Progress and Data Integrity bits and their usage.

6.3.2. Discrete Data

Please refer to Appendix C, **Common Data Features** for a description of discrete data, and to Appendix A and Appendix B for a detailed description of the data available in each format, in order to determine which is most suitable.

6.3.3. Byte Order

For a general account of byte ordering, please refer to Appendix C, **Common Data Features**.

6.3.4. Floating Point

For a general account of Floating Point operation, data format and compatibility, please refer to Appendix B, **Floating Point Format**.

Siemens S7 and Rockwell/Allen-Bradley PLCs support the IEEE Standard floating point numbers.

6.4. Shared Data

6.4.1. Operational Overview

If Shared Data is **Enabled** in the PLC's PROFIBUS Setup, the PLC can access Shared Data on an IND570 over the PROFIBUS, using an extension of the cyclic I/O.

The PLC must specify the Shared Data command and variable name in the PLC output message. If the command is a write command, then the PLC output message must also contain the write field value. The maximum length of the value is 20 bytes.

When the Shared Data command is a read command, the PLC input message will have a read field containing the data from the Shared Data variable specified in the output message. The maximum length of the data reported in the read field is 20 bytes.

The Shared Data variables are self-typing. The IND570 terminal determines the type of any valid data field in the message from the variable's name and definition in Shared Data. The terminal will not allow string data to be written in a floating point variable or vice versa.

6.4.1.1. Shared Data Input

The input information for the shared data consists of two sections: the shared data status and the shared data read field value (if requested by the shared data output command). The shared data status information is a word that contains an integer value. This integer value represents one of the following status values:

- 0 Null status
- 1 Command completed successfully
- 2 Invalid shared data name
- 3 Invalid shared data command
- 4 Cannot write because field is write-protected (legal for trade)

The shared data read field value contains the value of the shared data variable specified in the shared data output (from the PLC to the terminal). It is only present when the command from the shared data output requests read shared data. This value is self-typing; for example, it could be a floating point number or a string variable. The length is determined by the variable selected but will not exceed 20 bytes. See the tables following the Shared Data Output for a list of possible variables and their contents.

6.4.1.2. Shared Data Output

The output information for the shared data consists of four sections: the shared data command, the shared data name, the shared data variable name, and the shared data write value (if required by the shared data output command). The shared data command information is a word that contains an integer value. This integer value represents one of the following status values:

- 0 Null command
- 1 Read shared data
- 2 Write shared data

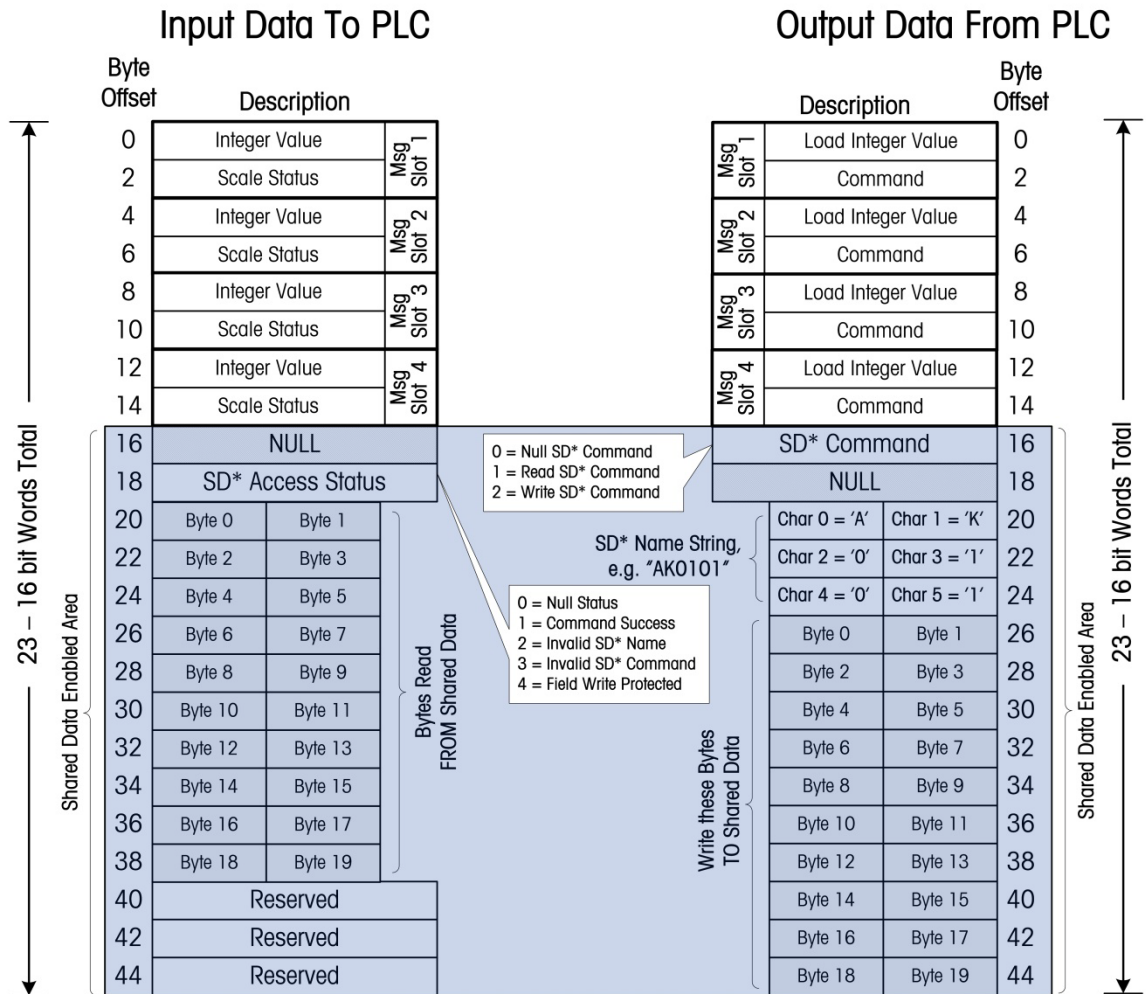
The terminal processes a shared data command "on demand" by the PLC. When a new value is placed in the shared data command word, the terminal will perform the command issued. The terminal does not provide "real time" information to the PLC; it supplies a "snapshot" of the data not an automatic update of new values of the same shared data command. Instead, the PLC must request the information again by setting a new value in the shared data command word.

To do successive reads, for example, the PLC must alternate between a “null” command and a “read” command in the shared data command word. For the most efficient processing, the PLC should set up the terminal name, the variable name, and the write value (if any) while it is setting the “null” command. Once that is completed, the PLC can then set the shared data command to “read” or “write”.

Refer to the [IND570 Shared Data Reference](#) for a complete listing of Shared Data Fields.

6.5. IND570 PROFIBUS I/O Mapping

Integer or Division Mode With Shared Data Access



* SD refers to Shared Data

Figure 6-3: Integer/Division I/O Mapping

Figure 6-3 shows the overall I/O mapping for IND570 terminals configured for Integer or Division PLC Communication mode. For both the Input and Output mappings, each message slot occupies 4 bytes or 2 Integer words. When configuring the PLC’s communications, the appropriate GSD I/O selection should be made as follows:

- 1 Message Slot = I/O 2 Wrd
- 2 Message Slots = I/O 4 Wrd
- 3 Message Slots = I/O 6 Wrd
- 4 Message Slots = I/O 8 Wrd

■ Note that if Shared Data has been Enabled, then regardless of the number of message slots configured, the I/O size selected from the GSD configuration should always be "I/O 23 Wrd" (for Integer and Division mode only!).

Floating Point Mode With Shared Data Access

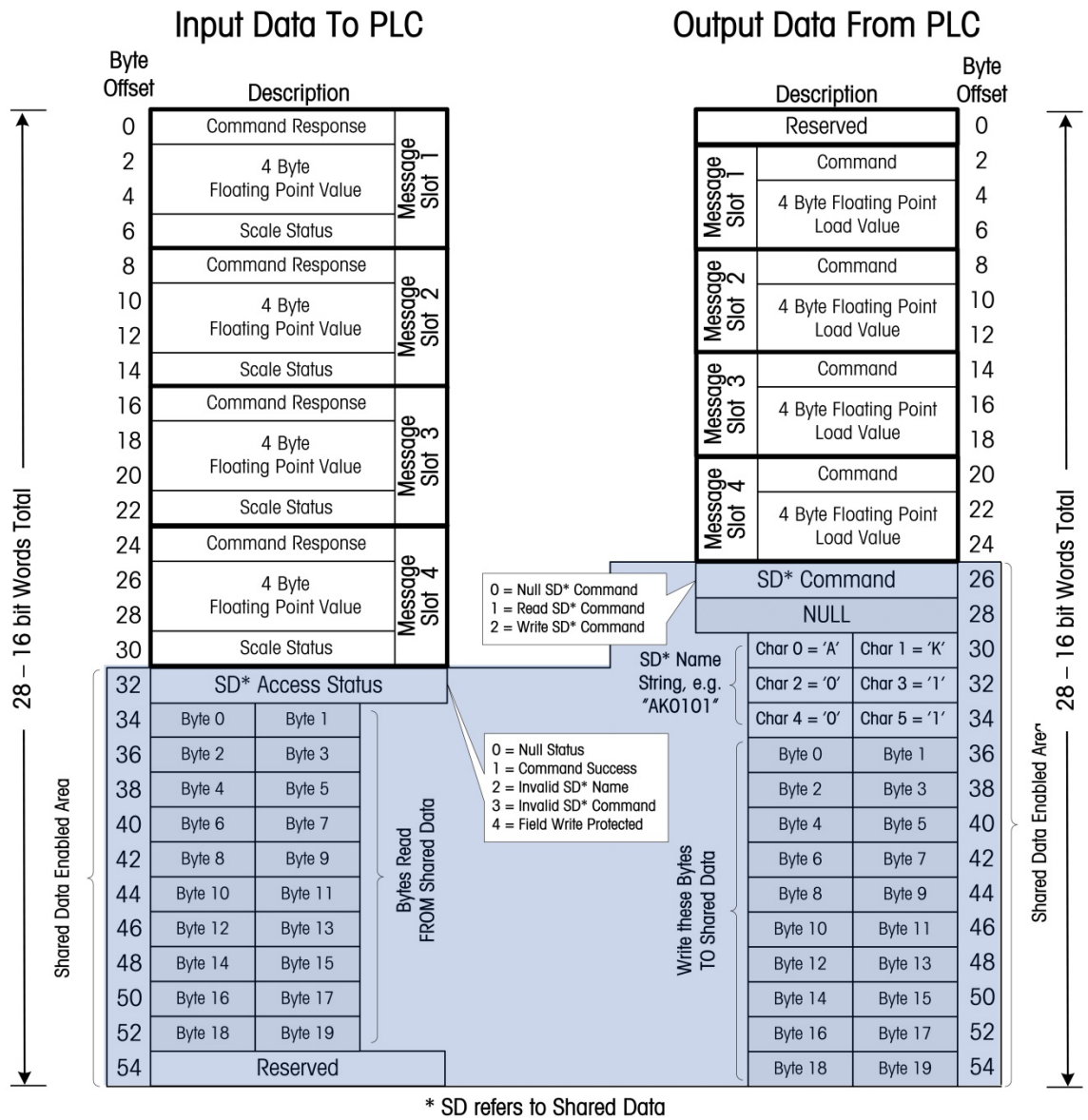


Figure 6-4: Floating Point I/O Mapping

The Floating Point I/O Mapping table above shows the overall I/O mapping for IND570 terminals configured for Floating Point PLC Communications mode. Each message slot occupies 8 bytes of Input and 6 bytes of Output memory, with the Output offset by 2 bytes. When configuring the PLC's communications, the appropriate GSD I/O selections should be made as follows:

- 1 Message Slot = I/O 4 Wrd
- 2 Message Slots = I/O 8 Wrd
- 3 Message Slots = I/O 12 Wrd
- 4 Message Slots = I/O 16 Wrd

■ Note that if Shared Data has been Enabled, then regardless of the number of message slots configured, the I/O size selected from the GSD file should always be "I/O 28 Wrd" (for Floating Point mode only!).

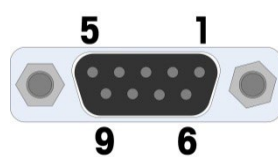
6.6. Controlling Discrete I/O Using a PLC Interface

Please refer to Appendix C, Common Data Features.

6.7. Hardware Setup

6.7.1. Wiring

The IND570 terminal's PROFIBUS option card has a DB-9 connector to connect to the PROFIBUS network interface (Figure 6-5). Cable distance, type, and termination are specified by PROFIBUS. (Refer to the PLC documentation for cable design guidelines for the various PLCs.)



Pin	Signal
1	Not used
2	Not used
3	RxD/TxD +
4	RTS
5	GND bus
6	+5V bus
7	Not used
8	RxD/TxD -
9	Not used

NOTES:

1. USE MATING CONNECTORS AND CABLE RECOMMENDED FOR PROFIBUS CONNECTIONS.
2. REFER TO PROFIBUS INTERNATIONAL DOCUMENTATION FOR OTHER CONSIDERATIONS.

Figure 6-5: PROFIBUS Option Card DB-9 Connector

The IND570 harsh unit requires a right angle connector Siemens part number 6ES7 972-0BA41-OXA0. The panel mount can use either the right angle or the straight connector, METTLER TOLEDO part number 64054361. Figure 6-6 shows the PROFIBUS option installed in an IND570 enclosure for harsh environments.



Figure 6-6: PROFIBUS Option Board Installed - Harsh Enclosure

6.8. Software Setup

The IND570 terminal automatically detects the presence of a PROFIBUS option card if one is installed, and adds the setup parameters to the options block. To configure the terminal for PROFIBUS, enter Setup and advance to the **Communications > PLC > PROFIBUS** sub-block (Figure 6-7).

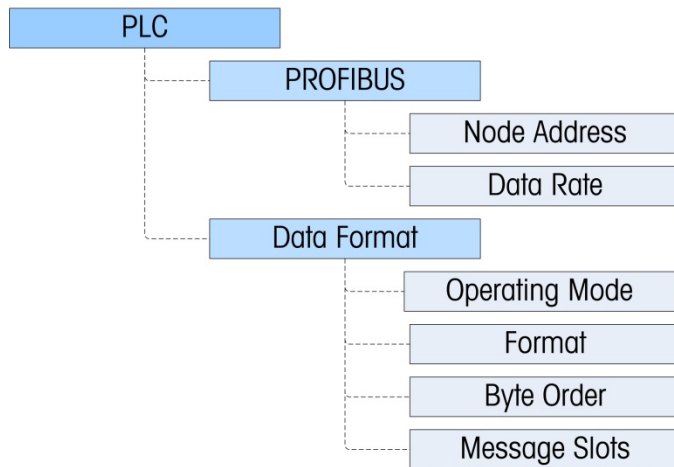


Figure 6-7: PROFIBUS Configuration Options in IND570 Setup

6.8.1. PROFIBUS and Data Format Setup Blocks

6.8.1.1. PROFIBUS Setup

The PROFIBUS setup block at **Communication > PLC > PROFIBUS** allows the user to specify how the PROFIBUS interface is used.

6.8.1.1.1. Node Address

Enter a unique Node Address of 0–125.

6.8.1.1.2. Shared Data
Set Shared Data as Enabled or Disabled

6.8.1.2. Data Format Setup

In Setup, navigate to **Communication > PLC Interface > Data Format**. The following fields are available for PROFIBUS.

6.8.1.2.1. Operating Mode

Operating Mode may be selected from a drop-down list. Choices are:

Compatibility Mode [default], IND560 Emulation

Depending on the Byte Order selection (refer to section 3.8.9.6.3, **Byte Order**, below), **Compatibility Mode** will provide the same discrete mode byte order arrangements as the METTLER TOLEDO IND131/331 and IND780 terminals. If **IND560 Emulation** is selected, the transmitted bytes in discrete mode will match the existing IND560 byte order determined by the Byte Order selection. Byte order arrangement in the IND560 terminals does not match that of IND131/331 and IND780. IND560 Emulation mode should be chosen only when replacing an IND560 **and** the programming within the PLC will not be modified.

6.8.1.2.2. Format

Select the Format from a drop-down list. Select Divisions, Integer (default) or Floating Point.

6.8.1.2.3. Byte Order

Available selections are Standard, Byte Swap, Word Swap (default), and Double Word Swap. Refer to Table 4-2 for definitions.

6.8.1.2.4. Message Slots

Select 1, 2, 3 or 4 slots.

6.9. Troubleshooting

If the IND570 does not communicate with PLC, do the following:

- Check wiring and network termination.
- Confirm that the IND570's GSD file has been loaded into the PLC's network configuration (even if using IND560 Emulation Mode), and that the IND570's network node was defined using it.
- Confirm that the IND570 settings for data type and node address match those in the PLC and that each IND570 has a unique node address.
- Confirm that the I/O Word size selected in the PLC's network configuration matches the configuration setup in the IND570 (see section on PROFIBUS I/O Mapping), paying special attention to the Shared Data Enabled Setting in the IND570's configuration.
- If the PLC interface PCB was changed from another type, like EtherNet/IP or DeviceNet, a master reset of the IND570 should be performed. Contact Mettler Toledo service for assistance.
- Contact METTLER TOLEDO service for replacement of the PROFIBUS interface.

6.9.1. Status LEDs

The PROFIBUS interface card has four status LEDs indicators to indicate communication and fault status of the card. Figure 6-1 and Figure 6-2 indicate the location of these LEDs, and Figure 6-8 shows the array of the LEDs on the card. Table 6-1 explains the meaning of the indicators.

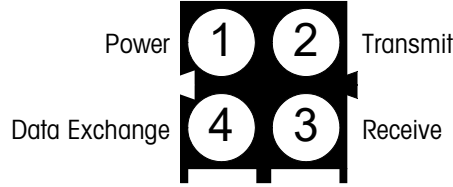


Figure 6-8: PROFIBUS Status Indicator LEDs

Table 6-1: PROFIBUS LEDs Status Indications

LED #	State	Status
1 - Power	Off	Power OFF
	Green	Power ON
2 – Transmit Status	Solid Green	Sending data
	Off	No data being sent; no power
3 - Receive Status	Solid Green	Data is being received
	Off	No data being received; no power
4 – Data Exchange	Solid Green	Data is being exchanged
	Off	No data being exchanged; no power

6.10. Interfacing Examples

Figure 6-9 shows a sample screen of IND570 hardware setup and I/O monitoring in the Siemens Step 7 software. Complete versions of these examples can be downloaded at www.mt.com/IND570.

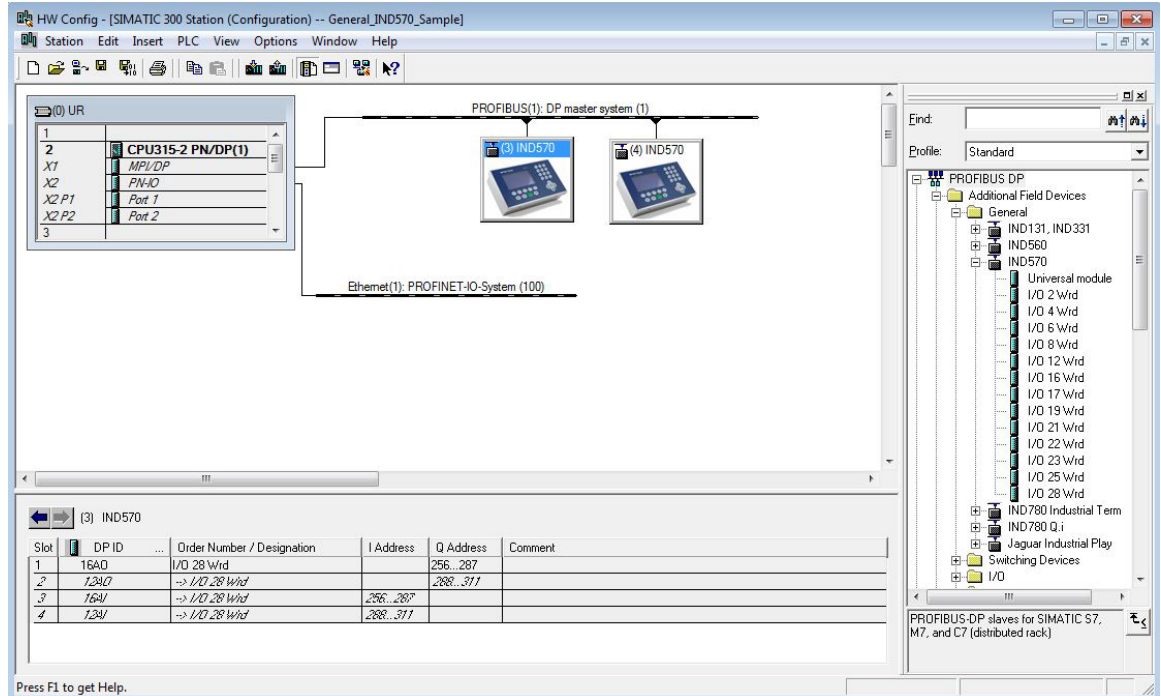


Figure 6-9: Hardware Setup

The Hardware setup shown is for the included sample program. In this sample, an IND570 is configured for Floating Point mode (GSD configuration “I/O 28 Wrd”) at PROFIBUS node 3, and another IND570 configured for Integer mode (GSD configuration “I/O 23 Wrd”) at PROFIBUS node 4.

Both nodes have the same configuration details:

- Shared Data = Enabled
- Operating Mode = Compatibility Mode
- Byte Order = Byte Swap
- Message Slots = 4

6.11. Sample PLC Program

Two General sample PLC programs are included on the documentation CD – one for an S7-315 PN/DP using Step 7 version 5 (SP3) and the other for an S7-1200 using TIA Portal. The two samples are essentially the same, and both programs include numerous comments that explain the

details of their operation. For the sake of brevity, only the main points of the S7-315's program are discussed here.

The sample program demonstrates the logic used to interface to an IND570 set up for Floating Point Data or Integer Data Formats. The logic also includes routines that access Shared Data over the PROFIBUS interface in both Floating Point and Integer Data Formats.

This sample program is subject to change without notice. Please visit www.mt.com/IND5xx to download the most recent version of PLC sample code.

6.11.1. PLC Hardware Configuration

For proper operation, the PLC Processor's configuration must be set up so that the size of the of the Process Image Input and Output areas is 512, as shown in Figure 6-10.

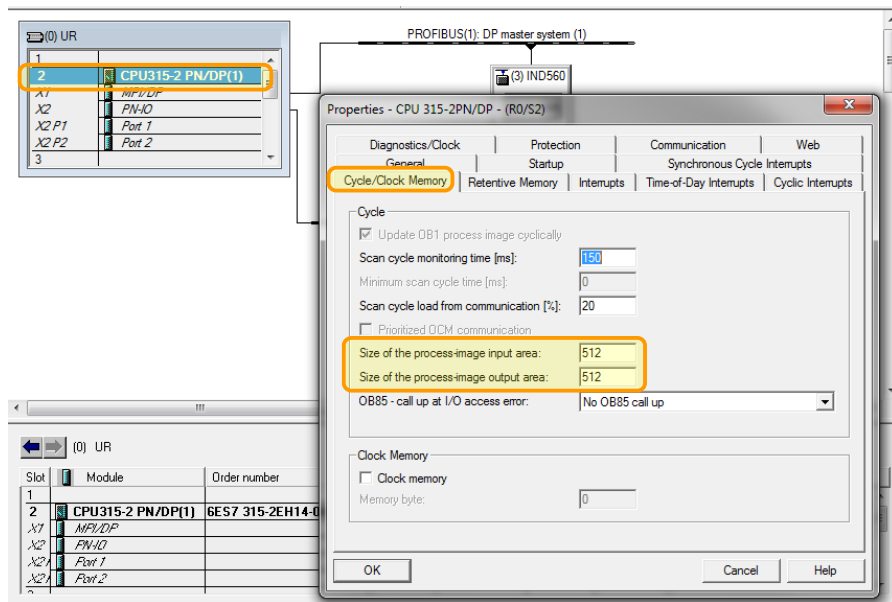


Figure 6-10: CPU315-2 PN/DP Object Properties

6.11.2. General Programming Notes

The following principles should always be applied to guarantee the validity of the data before using it in a process. Note that there are different principles for the different modes (Floating Point versus Integer or Divisions).

For Floating Point Mode, data being read from the Terminal should be filtered with the Data_OK bit and the two Data Integrity bits as shown in Figure 6-11.

□ Network 3 : Update the data read from the ProfiBus IND

Only update the data if the Data OK bit is on, and the Data Integrity bits match (indicating that the data inbetween the bits is valid). Otherwise, throw the data away.

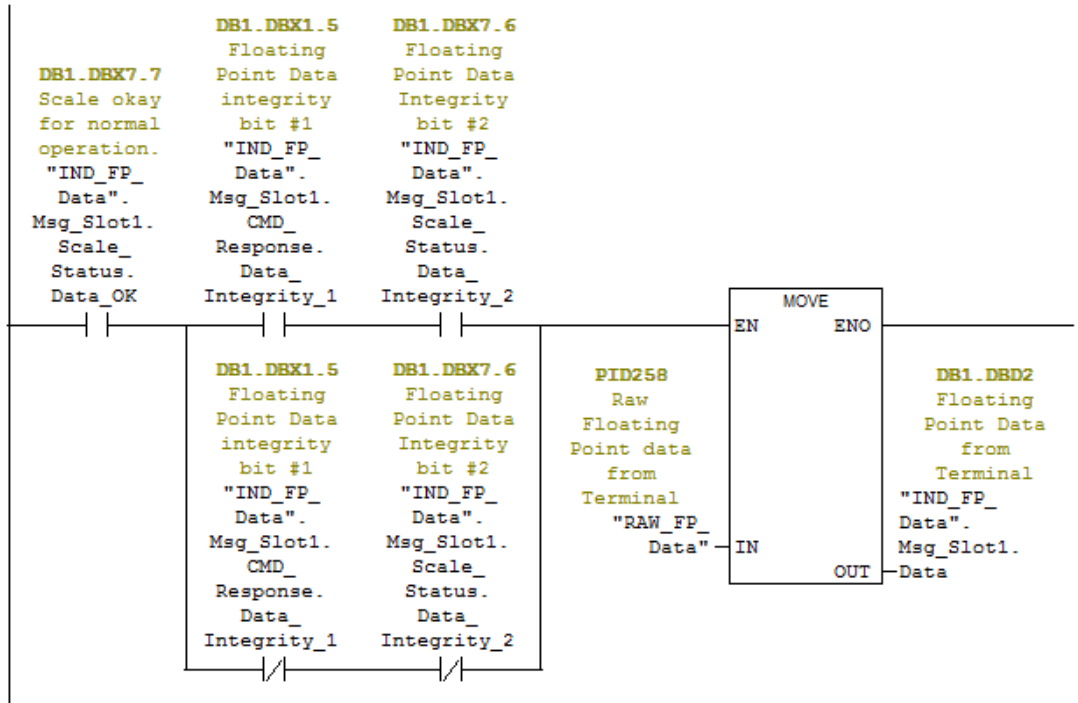


Figure 6-11: Filtering for Valid Data, Floating Point Mode

Filtering the data in this way makes sure that the Terminal is in a valid operational state (Data_OK = 1) and that the Analog Update from the Load Cell has properly completed before the data was read (Integrity_1 = Integrity_2). Failing to perform these checks can result in invalid data being used by the PLC program.

For Integer or Division Mode, a similar filter should be applied as shown in Figure 6-12.

□ Network 3 : Get the data from the Terminal and Normalize it to the Increment

We need to 'Normalize' the incoming Integer Data by multiplying it by the multiplier that is defined for the scale (that value is Hard Coded in Network 1 above, and is based on the resolution defined by the terminal's increment setting contained in the scale setup that is in the IND terminal itself). But first, we have to convert the Integer data coming back to a Double Integer, and then convert the Double Integer into a Real data type. After all of that, we can finally multiply the data by the Increment and store the result in the Message Slot data area.

Note that the Data OK bit MUST be ON, and the Update In Progress bit MUST be OFF, or the data coming back should be ignored.

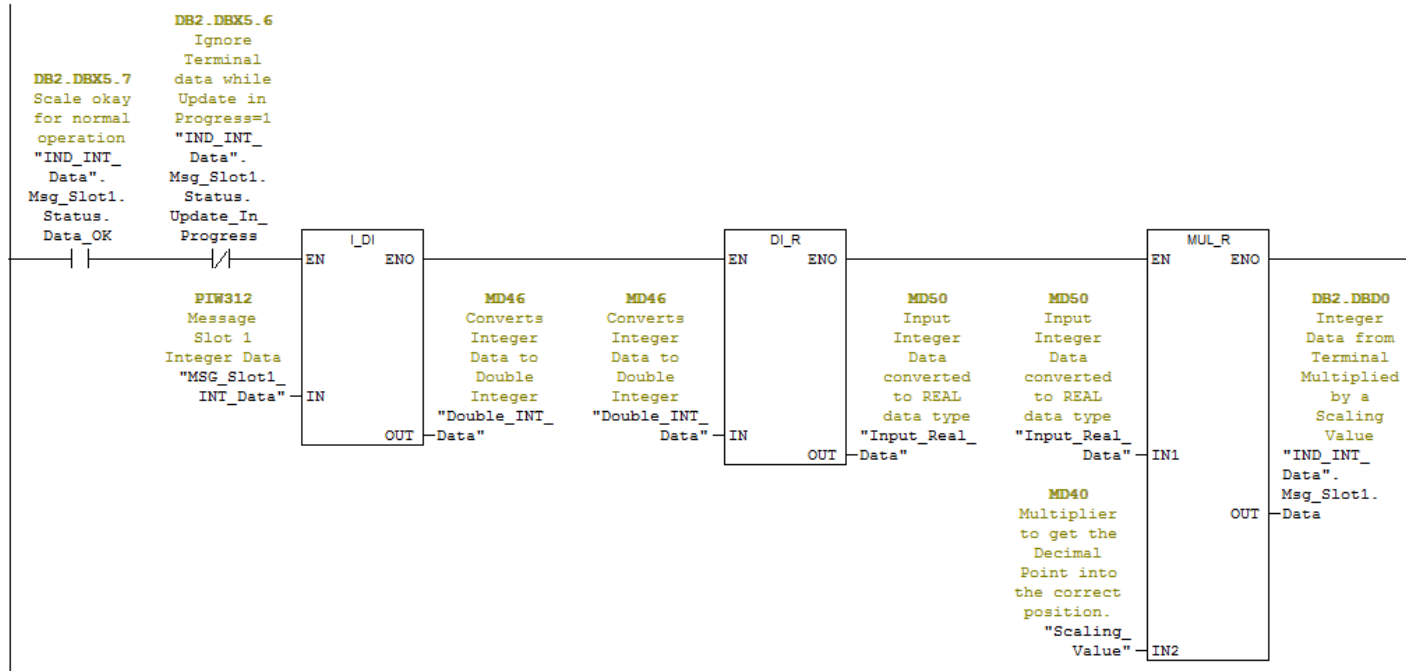


Figure 6-12: Filtering for Valid Data, Integer or Division Mode

In this case, the data is filtered with the Data_OK bit and the Update_In_Progress bit to make sure that the data coming back from the terminal is valid. From there, it is converted into a Floating Point value by first converting it to a 4 byte Integer, then multiplying it by the increment size to place the decimal point correctly.

6.11.3. Shared Data Access

The sample program demonstrates two slightly different methods of accessing Shared Data in the Terminal. The first method uses mainly a Variable Access Table, and one network of logic (in routine OB1) as shown in Figure 6-13.

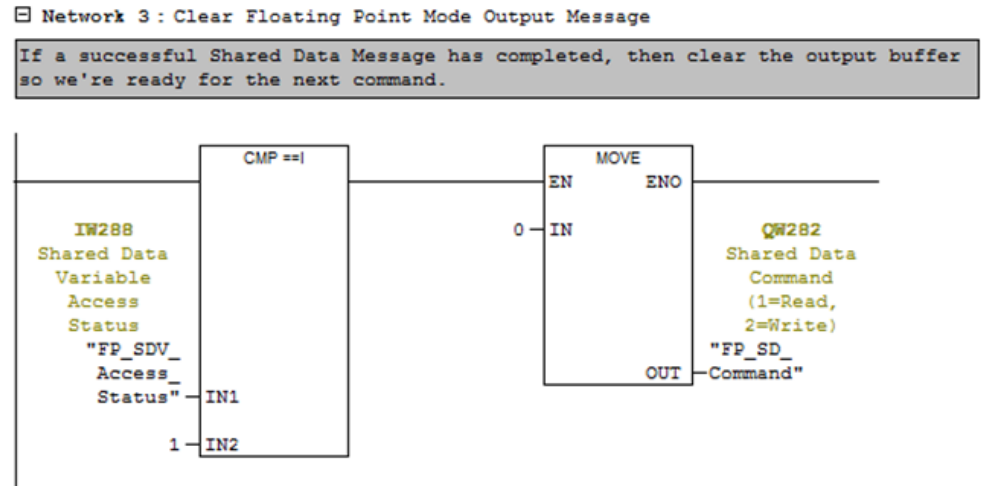


Figure 6-13: Shared Data Access, Method 1

The network logic looks for the Shared Data Access status to indicate a successful transaction, and then zeroes the Shared Data Command being output to the terminal. This is necessary to reset the sequence between commands so that the terminal will recognize that a new command has been sent when the time comes. This logic makes the Variable Access table function more smoothly because the user does not have to remember to zero the command in-between commands, since this rung does it automatically. If an error occurs (Shared Data Access Status > 1), the command does not get zeroed, making it easier for the user to identify the problem.

This is the recommended general practice with Shared Data Access commands, and it is also recommended that the command output remain zeroed for at least 100 milliseconds before a new command is allowed to be issued.

Any PLC logic that performs a Shared Data Access should monitor the Shared Data Access Status to determine:

1. That the command has completed successfully (status = 1).
2. Or that an error has occurred (any status > 1) and corrective action is required.

In the included Program sample, the Variable Access Table that can be used to manually read or write Shared Data is "FP_SDV_Access" for Floating Point mode or "INT_SDV_Access" for Integer mode. Since both tables are essentially identical (except for their memory mapping), only the Floating Point table is considered here.

6.11.4. Issuing a Shared Data Read Command

To Read a Shared Data variable, perform the sequence shown in Figure 6-14 and described below.

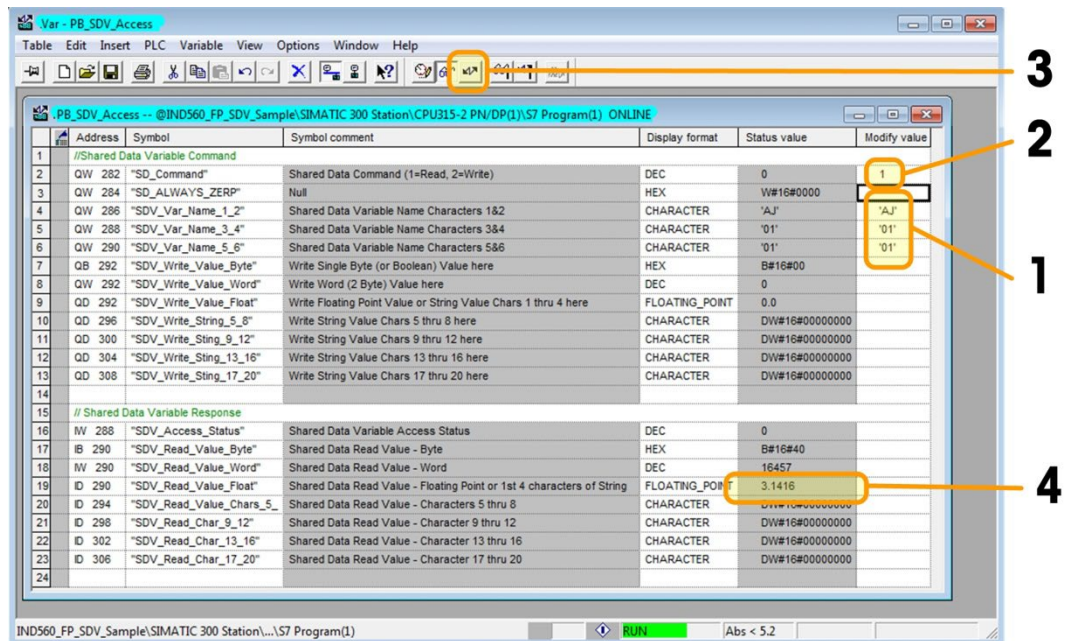


Figure 6-14: Sequence for Reading a Shared Data Variable

1. Enter the Shared Data variable name string as shown in the consecutive addresses starting at QW286.
2. Enter the "Read Shared Data" command (1) into the address QW 282.
3. Click the "Modify Variables" button to write the changes to the PLC, which in turn sends the command to the IND570.
4. The result (contents of AJ0101) is reported in the Input area starting at ID 290 (for floating Point values). In this case, the value read back from the terminal was 3.1416.

When this sequence is complete, the Shared Data Variable Access Status will briefly turn to a '1' if the command is successful. In this case, the logic described for the OB1 routine (Figure 6-11) will immediately zero the Write command from the buffer, which in turn will cause the IND570 terminal to zero the Shared Data Variable Access state. If an error occurs, the command is **not** zeroed, and the Shared Data Status will retain the error returned from the IND570.

6.11.5. Issuing a Shared Data Write Command

To Write to a Shared Data variable, perform the sequence shown in Figure 6-15 and described below.

Address	Symbol	Symbol comment	Display format	Status value	Modify value
//Shared Data Variable Command					
QW 282	"SD_Command"	Shared Data Command (1=Read, 2=Write)	DEC	0	2
QW 284	"SD_ALWAYS_ZERP"	Null	HEX	W#16#0000	
QW 286	"SDV_Var_Name_1_2"	Shared Data Variable Name Characters 1&2	CHARACTER	'AJ'	'AJ'
QW 288	"SDV_Var_Name_3_4"	Shared Data Variable Name Characters 3&4	CHARACTER	'01'	'01'
QW 290	"SDV_Var_Name_5_6"	Shared Data Variable Name Characters 5&6	CHARACTER	'01'	'01'
QB 292	"SDV_Write_Value_Byte"	Write Single Byte (or Boolean) Value here	HEX	B#16#40	
QW 292	"SDV_Write_Value_Word"	Write Word (2 Byte) Value here	DEC	16432	
QD 292	"SDV_Write_Value_Float"	Write Floating Point Value or String Value Chars 1 thru 4 here	FLOATING_POINT	2.76	2.76
QD 296	"SDV_Write_String_5_8"	Write String Value Chars 5 thru 8 here	CHARACTER	DW#16#00000000	
QD 300	"SDV_Write_Sting_9_12"	Write String Value Chars 9 thru 12 here	CHARACTER	DW#16#00000000	
QD 304	"SDV_Write_Sting_13_16"	Write String Value Chars 13 thru 16 here	CHARACTER	DW#16#00000000	
QD 308	"SDV_Write_Sting_17_20"	Write String Value Chars 17 thru 20 here	CHARACTER	DW#16#00000000	
// Shared Data Variable Response					
IW 288	"SDV_Access_Status"	Shared Data Variable Access Status	DEC	0	
IB 290	"SDV_Read_Value_Byte"	Shared Data Read Value - Byte	HEX	B#16#40	
IW 290	"SDV_Read_Value_Word"	Shared Data Read Value - Word	DEC	16457	
ID 290	"SDV_Read_Value_Float"	Shared Data Read Value - Floating Point or 1st 4 characters of String	FLOATING_POINT	3.1416	
ID 294	"SDV_Read_Value_Chars_5_8"	Shared Data Read Value - Characters 5 thru 8	CHARACTER	DW#16#00000000	
ID 298	"SDV_Read_Char_9_12"	Shared Data Read Value - Character 9 thru 12	CHARACTER	DW#16#00000000	
ID 302	"SDV_Read_Char_13_16"	Shared Data Read Value - Character 13 thru 16	CHARACTER	DW#16#00000000	
ID 306	"SDV_Read_Char_17_20"	Shared Data Read Value - Character 17 thru 20	CHARACTER	DW#16#00000000	

Figure 6-15: Sequence for Issuing a Shared Data Write Command

1. Enter the Shared Data variable name string as shown in the consecutive addresses starting at QW286.
2. Enter the "Write Shared Data" command (2) into the address QW 282.
3. Enter the new value to be written to AJ0101 into QD 292.
4. Click the "Modify Variables" button to write the changes to the PLC, which in turn sends the command to the IND570.

To confirm that the variable was written properly, the same read routine performed previously can be repeated.

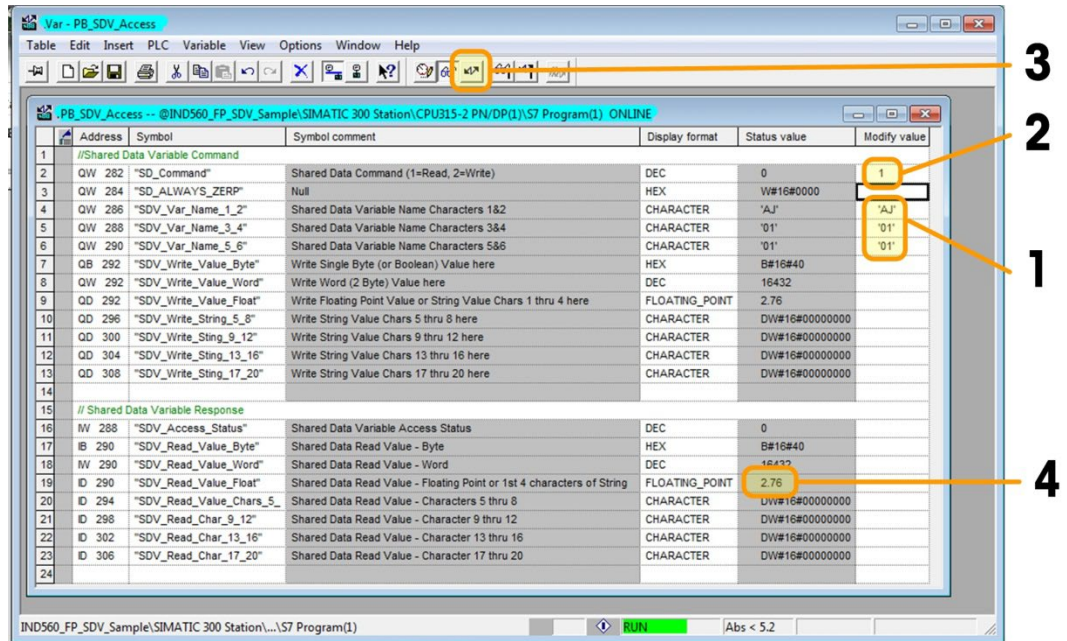


Figure 6-16: Read Sequence, Repeated to Confirm Written Data

1. Enter the Shared Data variable name string as shown in the consecutive addresses starting at QW286.
2. Enter the "Read Shared Data" command (1) into the address QW 282.
3. Click the "Modify Variables" button to write the changes to the PLC, which in turn sends the command to the IND570.
4. The result (contents of AJ0101) is reported in the Input area starting at ID 290 (for floating Point values). In this case, the value read back from the terminal was 2.76 – the same value that was written in the previous example.

6.11.6. Shared Data Access via PLC Code

The sample programs include ladder logic code to read and write Shared Data from the program. The code for the Floating Point Shared Data memory mapping is in FC2, while the code for the Integer Shared Data memory mapping is in FC4.

Both routines allow the program to specify the Shared Data Variable name by using a character array in their associated Data Block files. For Floating Point the Data Block is DB1 (Integer is DB2), which is shown below.

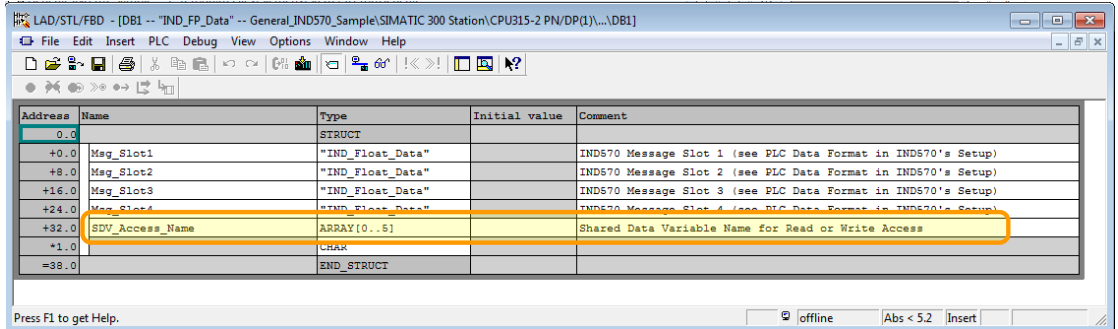


Figure 6-17: Shared Data Access Using PLC Code

To change the variable being accessed, the program should write a new Shared Data variable name (in ASCII Characters) to this array.

A separate Variable Access Table exists to drive FC2 called FP_SDV_Access_Program (Figure 6-18). Another Variable Access Table to drive FC4 (INT_SDV_Access_Program) exist to drive the Integer Mode process.

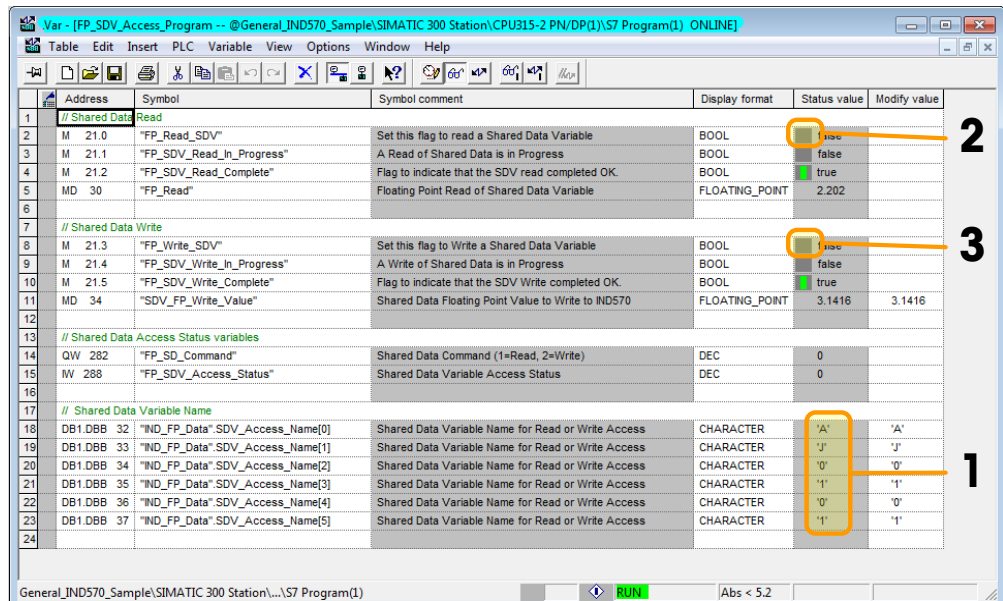


Figure 6-18: Shared Data Variable Access Table

1. Update the DB1 Shared Data Name Array here.
2. Set this bit to 1 to trigger a Read.
3. Set this bit to 1 to trigger a Write.

The main thing to note in Shared Data Access routines within the sample program are the timers used to control when the last issued command is to get zeroed. Having these timers setup forces the Shared Data command back to zero for at least 100 milliseconds to allow the terminal to recognize when the next new command is issued.

6.11.7. Shared Data Access: Conclusion

Almost any Shared Data Variable can be read and written as shown in the previous examples. In this respect, the PROFIBUS interface has fewer restrictions than most of the other Fieldbus interfaces supported by the IND570.

However, there is one restriction that is important to note – the maximum number of bytes that can be written by the PLC to the IND570, or read back to the PLC from the IND570, is 20. That limit applies to all strings and arrays.

7 PROFINET

7.1. Overview

PROFINET is an open industrial networking standard that was developed by Siemens as an Ethernet replacement for its widely popular PROFIBUS Network. The network supports Cyclic and Acyclic messaging, both of which have been implemented in the IND570. PROFINET utilizes commercial, off-the-shelf Ethernet hardware (for example, switches and routers) and is fully compatible with the Ethernet TCP/IP protocol suite.

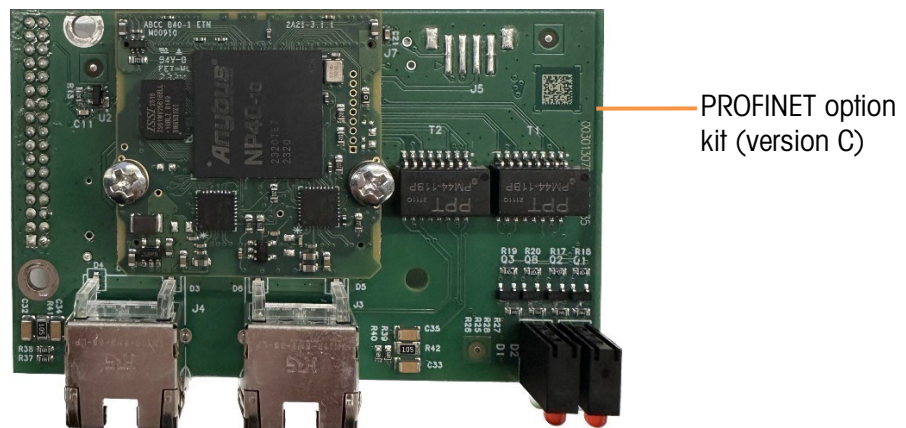
The IND570 PROFINET option implements PROFINET IO for cyclic data exchange with the PLC, and uses acyclic messages for Shared Data Access by the PLC.

The PROFINET option enables the IND570 terminal to communicate to PROFINET enabled Programmable Logic Controllers (PLCs) through direct connection to the PROFINET network at 100 MBPS speed. This solution consists of an internal module and internal software to implement the data exchange.

7.2. PROFINET Interface

The part # of the IND570 PROFINET option kit (version D) is 30260484 and the part # of the IND570 PROFINET option kit (version C) is 30861289. Figure 7-1 and Figure 7-2 show a PROFINET module and its components.

The PROFINET option board (version D) comes with a label to indicate the PCBAC number, version, as well as the production year and week (e.g., **30687088** represents the upgraded PROFINET option board - version D).



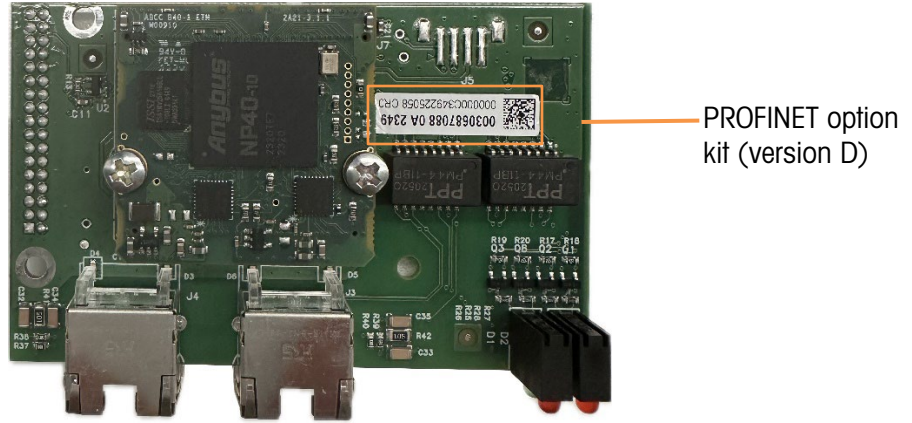


Figure 7-1: PROFINET Module

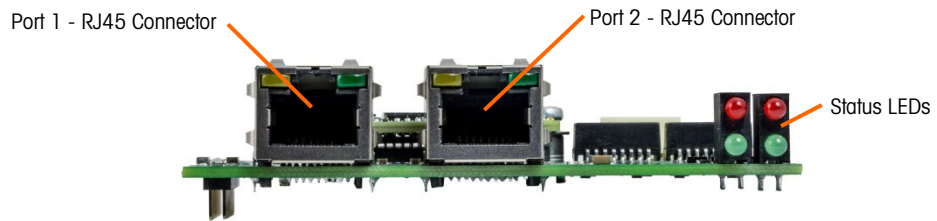


Figure 7-2: IND570 PROFINET Module

7.2.1. Definition of Terms

The following terms are used in this chapter.

Table 7-1: PROFINET Definition of Terms

Term	Definition
DAP	Device Access Point
DCP	Discovery and basic Configuration Protocol. Used for IP configuration over PROFINET.
DHCP	De-facto standard for dynamic IP address management
GSDML	XML-based descriptive language for GSD-files
Initial Record Data	Record Data write-requests destined for a sub-module. Comparable to PROFIBUS-DP User Parameter Data.
IOCS	IO Consumer Status
IOPS	IO Provider Status
IO Controller	Controlling device which acts as a client for several IO devices. Usually a PLC. Comparable to a PROFIBUS-DP Class 1 master.
IO Device	Field device assigned to an IO Controller. Comparable to a PROFIBUS DPV1 slave.
IO Supervisor	Programming device with commissioning and diagnostic functions. Comparable to a PROFIBUS-DP Class 2 master.
Module	Hardware or logical component of a network device.

Term	Definition
MRP	Media Redundancy Protocol. An Ethernet Ring Topology used with PROFINET IO to provide media redundant communications. Messages are sent out one Ethernet port of the PLC and come back in the other. If the PLC detects a media break in the ring then it reconfigures the network within 200 milliseconds so that messages will be sent out both ports of the PLC. Requires PLC's and devices that are MRP enabled. Any switches on the network must also be MRP enabled. Non-MRP enabled devices may be connected to the loop by using MRP enabled switches.
Submodule	Hardware or logical component of a module
PDEV	Physical DEvice. From specification version 2.0 it is possible to describe the physical Ethernet interface and its ports (PDEV, or Physical Device) with a special mechanism. This is done with special sub-modules at slot 0 (the module at slot 0 is the access point for the device).
PNIO	Short for PROFINET IO.
PROFINET IO	PROFINET IO is a communication concept for the implementation of modular, decentralized applications. Comparable to PROFIBUS-DP, where I/O data of field devices are cyclically transmitted to the process image of a PLC. The real time capabilities of PROFINET IO are further divided into RT and IRT (see below).
PROFINET IO RT	PROFINET IO with Real Time capabilities. Optimized real time communication channel for time critical I/O data and Alarms. Implemented in software.
PROFINET IRT	PROFINET IO with Isochronous Real Time capabilities. Necessary for motion control application which require an update rate of 1ms, or less, with no jitter. Implemented in hardware.
PROFINET CBA	PROFINET Component Based Automation. Comparable to PROFIBUS FMS.
Record Data	Comparable to PROFIBUS DPV1 acyclic Read/Write.

7.2.2. Communications

The IND570 terminal uses component parts to ensure complete compatibility with the Siemens PROFINET network. An IND570 terminal is recognized as a generic PROFINET device by the PLC.

7.2.3. IP Address

Each PROFINET option represents one physical IP Address. This address can be chosen by the system designer, and then programmed into the IND570 terminal and PLC, or the address can be automatically assigned by the PLC. Each IND570 within a system must have a unique PROFINET IP Address.

The IND570 terminal's PROFINET IP address is programmed in the terminal's setup menu at **Communication > PLC > PROFINET**.

7.2.4. Supported Data Transfer

The PROFINET interface provides both discrete data transfer and an acyclic messaging capability that is used for Shared Data access. Access to Shared Data is done in a manner that is very similar to the method used by the ControlNet and Ethernet/IP modules.

7.2.5. Connection Methods

The dual ports on the PROFINET Interface module provide several possible methods for connecting the IND570 to the control Network. Those methods are described in this section. It is important that in both the Daisy Chain and MRP Redundant Loop configurations, the physical network wiring matches the network topology defined on the PLC as it relates to Port 1 and Port 2. If the wiring does not match the defined topology, errors will be reported.

7.2.5.1. Star Network

A star network consists of multiple devices being attached to one or more Ethernet switches.

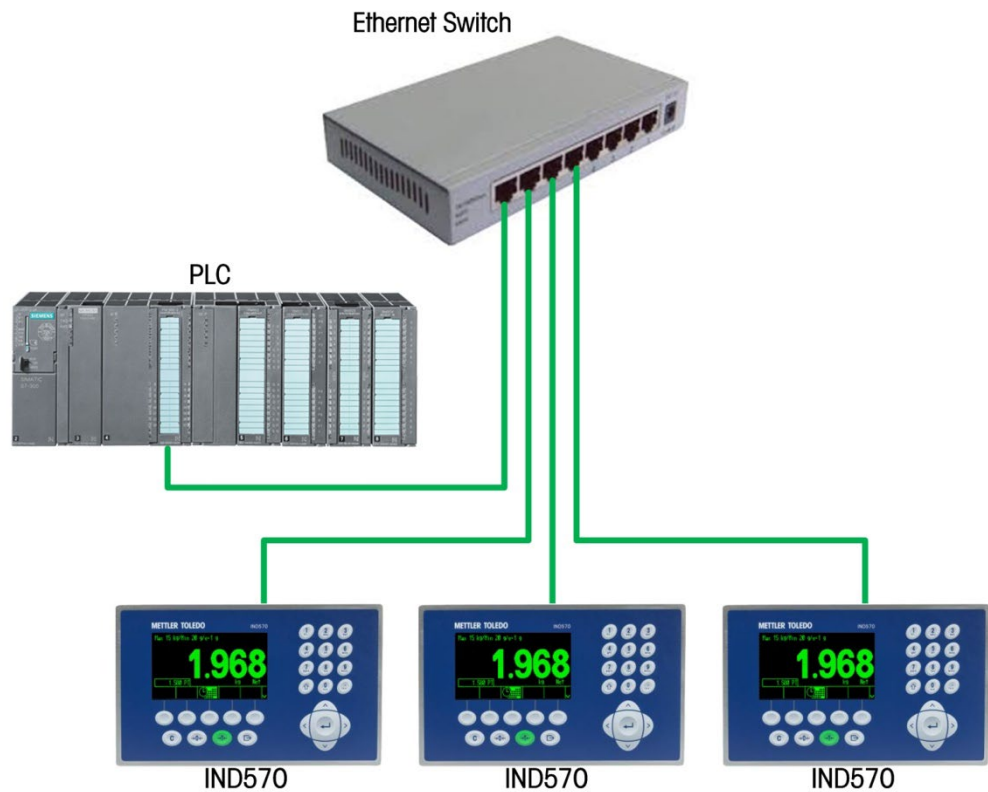


Figure 7-3: Star Network Example

7.2.5.2. Daisy Chain

A Daisy Chain network has the advantage of not requiring switches for multiple devices to be connected to the Controller. This has advantages in a cabinet or tight space where there may not be sufficient area to run individual cables all of the way back to a central point such as a switch.

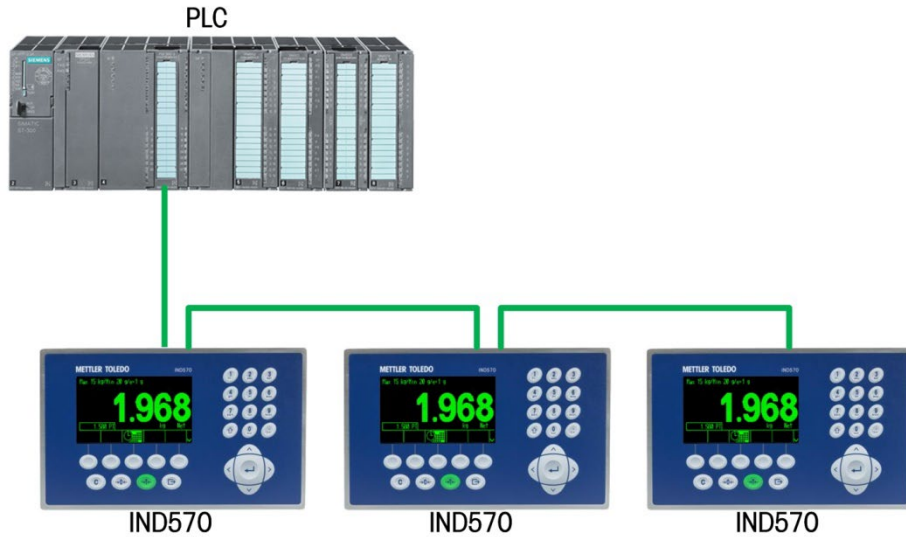


Figure 7-4: Daisy Chain Example

7.2.5.3. MRP Redundant Loop

An MRP redundant loop is very similar to the Daisy Chain topology, where the PLC is connected on one end of the loop, and devices are daisy chained along the loop until the loop is terminated back at the same PLC on a second Ethernet port. This provides a 'Ring' topology where messages can be routed either direction around the ring, and has the advantage of not requiring any switches as long as the PLC and the devices are MRP capable. If a break in the Ring occurs, the PLC will quickly detect it by noticing that messages are no longer making it back to the PLC on the opposite end of the ring that is attached to it. Under those conditions the PLC will then start transmitting the messages out both ports so that all devices on the ring can still get the messages. The result is a network of daisy chains out each port that continue to function regardless of the break. PROFINET MRP is designed to make the break detection and switch over in less than 200 milliseconds. NOTE that your process must be able to tolerate a loss of communications for up to 200 ms.

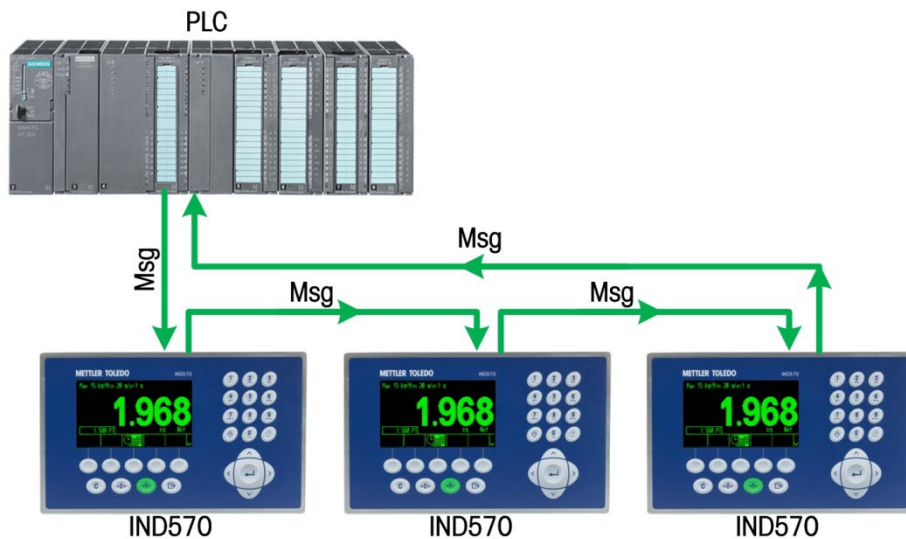


Figure 7-5: Intact MRP Ring

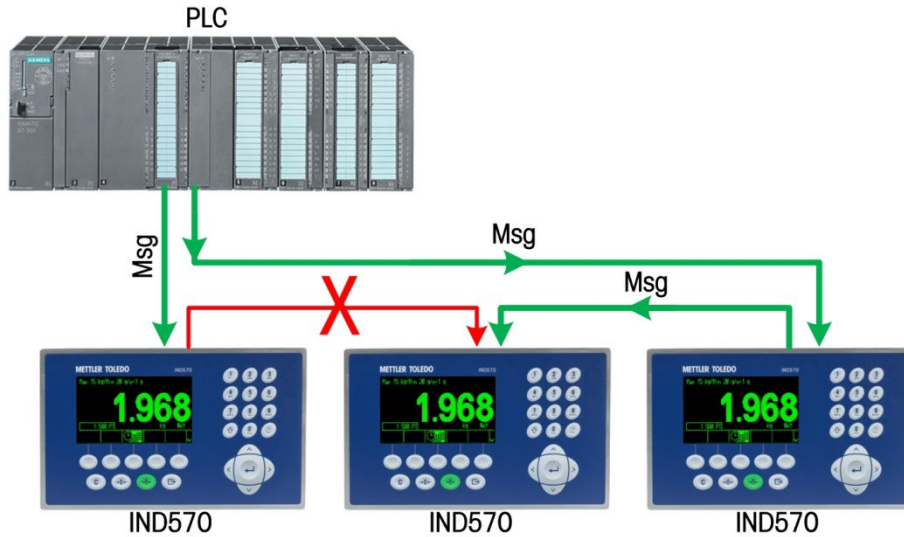


Figure 7-6: MRP Ring with Break

- Note that messages still reach all devices, because the network is self-healing.

7.3. Data Definition

7.3.1. Data Integrity

The IND570 has specific bits to allow the PLC to confirm that data was received without interruption and that the IND570 is not in an error condition. It is important to monitor these bits. Any PLC code should use them to confirm the integrity of the data received by the IND570.

Refer to the data charts in Appendix A and Appendix B for specific information regarding the Data OK, Update in Progress and Data Integrity bits and their usage.

7.3.2. Discrete Data

The terminal's PROFINET interface has three discrete data formats that may be selected. The data types are: Integer, Divisions and Floating Point.

Please refer to Appendix C, **Common Data Features** for a description of discrete data, and to Appendix A and Appendix B for a detailed description of the information available in each data format.

7.3.3. Byte Order

For a general account of byte ordering, please refer to Appendix C, **Common Data Features**.

7.3.4. Message Slots

There may be up to 4 message slots for discrete data transfer of cyclic messages in Integer, Divisions and Floating Point Data Formats. Each message slot is assigned to a local or remote scale and scales may be repeated in additional message slots. The integer and division formats provide two 16-bit words of input and two 16-bit words of output data per Message Slot. Each Message Slot's first input word provides scale weight data and the input weight data may be

selected by the PLC using the Message Slot's second output word bit 0, bit 1 and bit 2. The following two Tables provide input and output usage information.

The floating point format provides four 16-bit words of input data and three 16-bit words of output data) per Message Slot. Refer to Table 7-2 and Table 7-3 for details.

The number of Message Slots is selected in the terminal's setup menu at **Communication > PLC Interface > Data Format**.

Table 7-2: Message Slot and PLC I/O Sizes (Integer/ Division)

IND570 Integer/ Division Data		
Message Slots	Bytes (8 Bit)	
	IND570 >> PLC Input	PLC Output >> IND570
1	4	4
2	8	8
3	12	12
4	16	16

Integer or Division Mode

Input Data To PLC

Output Data From PLC

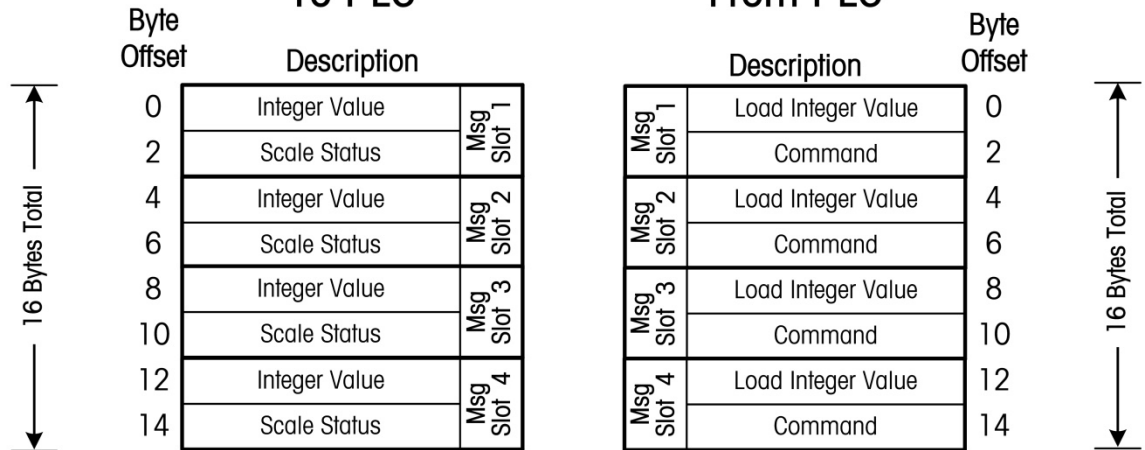


Figure 7-7 Integer/Divisions Message Slot I/O Mapping

Table 7-3: Message Slot and PLC I/O Sizes (Floating Point)

IND570 Floating Point Data		
Message Slots	Bytes (8 Bit)	
	IND570 >> PLC Input	PLC Output >> IND570
1	8	8
2	16	14

3	24	20
4	32	26

Floating Point Mode

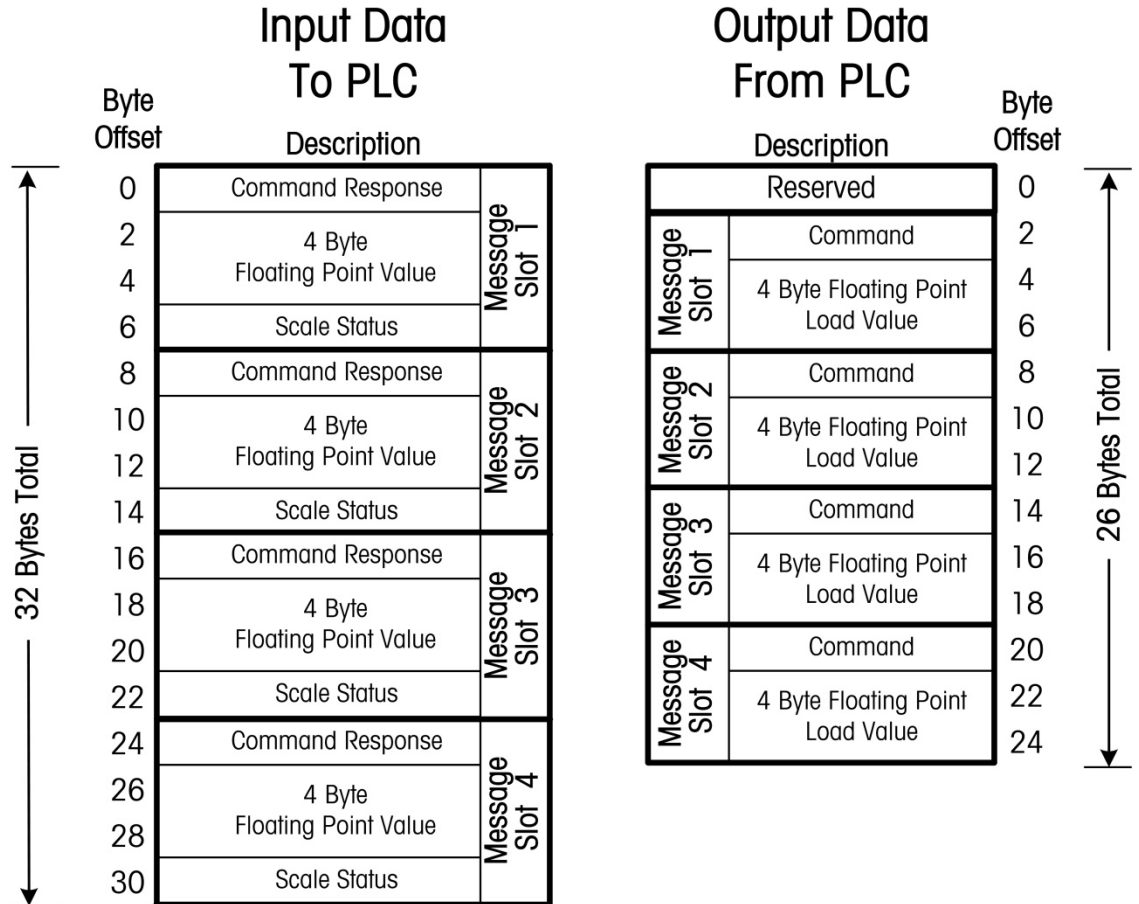


Figure 7-8 Floating Point Message Slot I/O Mapping

7.4. Controlling the Discrete I/O Using a PLC Interface

The IND570 terminal provides the ability to directly control its discrete outputs and read its discrete inputs via the (digital) PLC interface options. System integrators should be aware that the IND570 terminal's discrete I/O updates are synchronized with the terminal's interface update rate and not with the PLC I/O scan rate. This may cause a noticeable delay in reading inputs or updating outputs as observed from the PLC to real world signals. Consult the **IND570 Terminal Technical Manual** for discrete I/O wiring.

7.5. Shared Data Access

The Shared Data mode PLC communications is provided using Acyclic messaging to the IND570 terminal.

The IND570 Shared Data document lists the Shared Data Variables available to Ethernet/IP, ControlNet, and PROFINET. This document also includes the hex Class Code, Instance and Attribute for the shared data. The PLC must use a combination of RDREC (SFB52) and WRREC (SFB53) to read a Shared Data Variable and WRREC (SFB53) to write a Shared Data Variable.

7.6. Software Setup

The IND570 terminal automatically detects the presence of a PROFINET option board when installed and makes the PROFINET configuration parameters available within the setup menu. To configure the terminal for PROFINET communication, enter setup and navigate to the **Communication > PLC > PROFINET** sub-block (Figure 7-9).

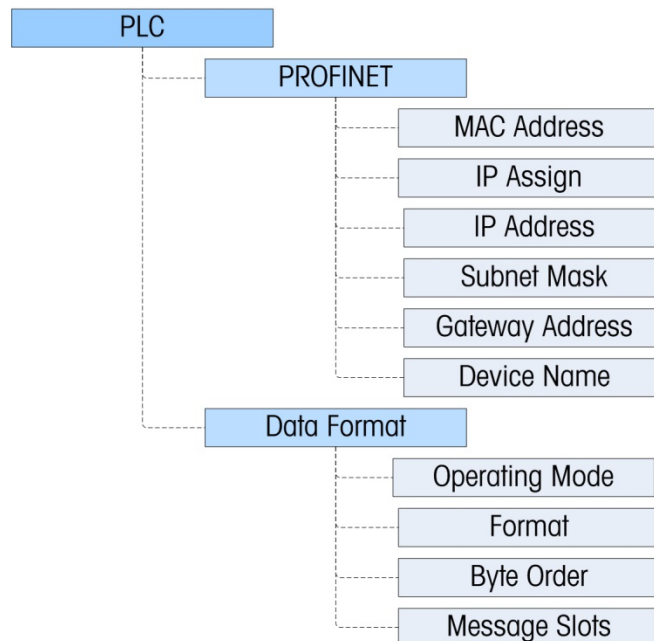


Figure 7-9: PROFINET Setup Block

7.6.1. PROFINET and Data Format Setup Blocks

7.6.1.1. PROFINET setup

The PROFINET setup block at **Communication > PLC > PROFINET** allows the user to specify how the PROFINET interface is used.

7.6.1.1.1. MAC Address

The MAC address is displayed, but cannot be modified.

7.6.1.1.2.

IP Assign

By default, the **IP Assign** field is set to **DCP** so that the PLC programming software can assign the IP Address, Subnet Mask and Gateway Address fields with information received from the network.

Where the IP Address, Subnet Mask and Gateway Address of the PROFINET interface are specified by the customer, the IP Assign field should be set as **Manual**. With this setting, the IP Address, Subnet Mask and Gateway Address fields must be manually populated by the installer.

The **DHCP** selection allows the general network (non-PLC network) to assign the IP Address, Subnet Mask and Gateway Address fields. This will be the least common circumstance.

Note that in all cases, the Device Name (a setting that is not programmable in the IND570) must be determined from the PLC programming software before communications to the PLC will be established.

7.6.1.2.

Data Format setup

In Setup, navigate to **Communication > PLC Interface > Data Format**. The following must be configured for PROFINET.

7.6.1.2.1.

Operating Mode

The default setting is Compatibility Mode. This cannot be changed. Compatibility Mode will provide the same discrete mode byte order arrangements as the METTLER TOLEDO IND131/331 and IND780 terminals.

NOTE: Because PROFINET was not available in the IND560 terminal, there is no need for the IND560 Emulation selection that is found in the setup of other IND570 PLC interfaces.

7.6.1.2.2.

Format

Select the data format: Integer (default), Divisions, or Floating Point. Changing the Format will delete any existing Message Slots.

7.6.1.3.

Byte Order

Available selections are Standard, Byte Swap, Word Swap (default) and Double Word Swap.

7.6.1.3.1.

Message Slots

Select 1, 2, 3 or 4 slots.

7.7. PROFINET GSDML File

The PROFINET GSDML file contains eight (8) Input configurations, and eight (8) Output configurations. It is very important that the sizes selected for the input and output configurations match each other. For example, if "FLOAT 1 Slot" is the Input selection, then "FLOAT 1 Slot" must also be the Output selection.

The number of slots designated in each configuration references the number of Message Slots configured in the IND570 itself.

- Note: The PROFINET GSDML file for the IND570 and complete versions of the programming examples can be downloaded from www.mt.com/IND570. The following screen images are provided for illustrative purposes only.

Figure 7-10 shows two IND570's placed on the PROFINET I/O Network. Node 1 (IND570) is configured as a Floating Point device, while node 2 (IND570-1) is configured as an Integer/Divisions type device.

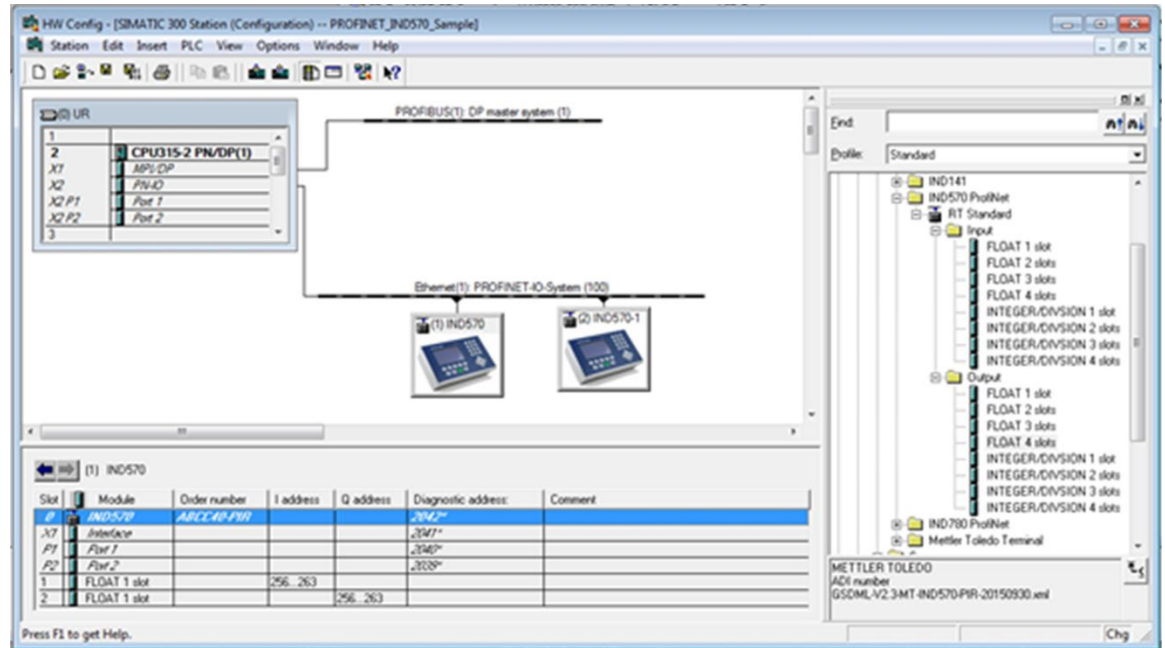


Figure 7-10: Hardware Network Setup

7.8. Assigning the IP Address and Device Name

By default, assigning the IND570's IP address and Device Name takes place via the DCP (Discovery and basic Configuration Protocol). This function is accessed via the PLC Engineering Software as shown below.

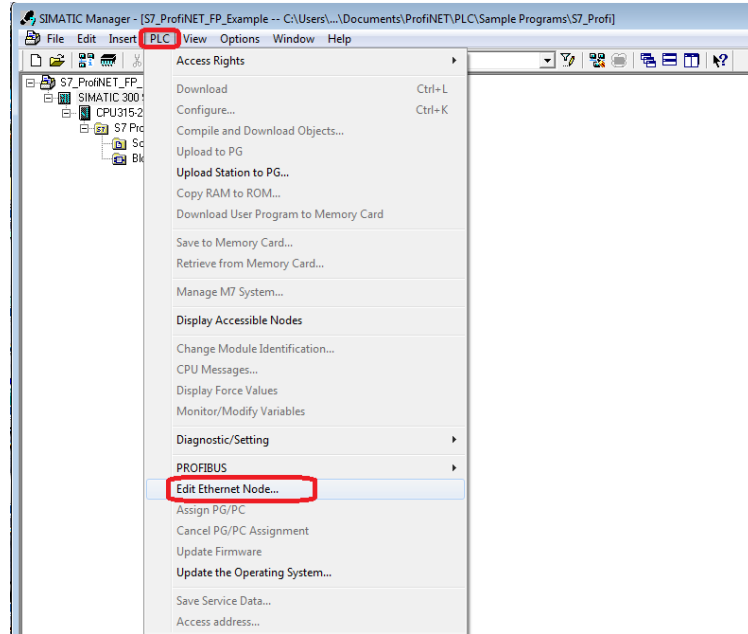


Figure 7-11: Configuration via DCP

Using the Browse function initiates the discovery of MAC addresses on the network. Select the MAC address you wish to work with (shown in the terminal at **Setup > Communications > PLC Interface > PROFINET**) by clicking on it, then clicking the **OK** button to continue.



Figure 7-12: PROFINET MAC Address in IND570 Setup Screen

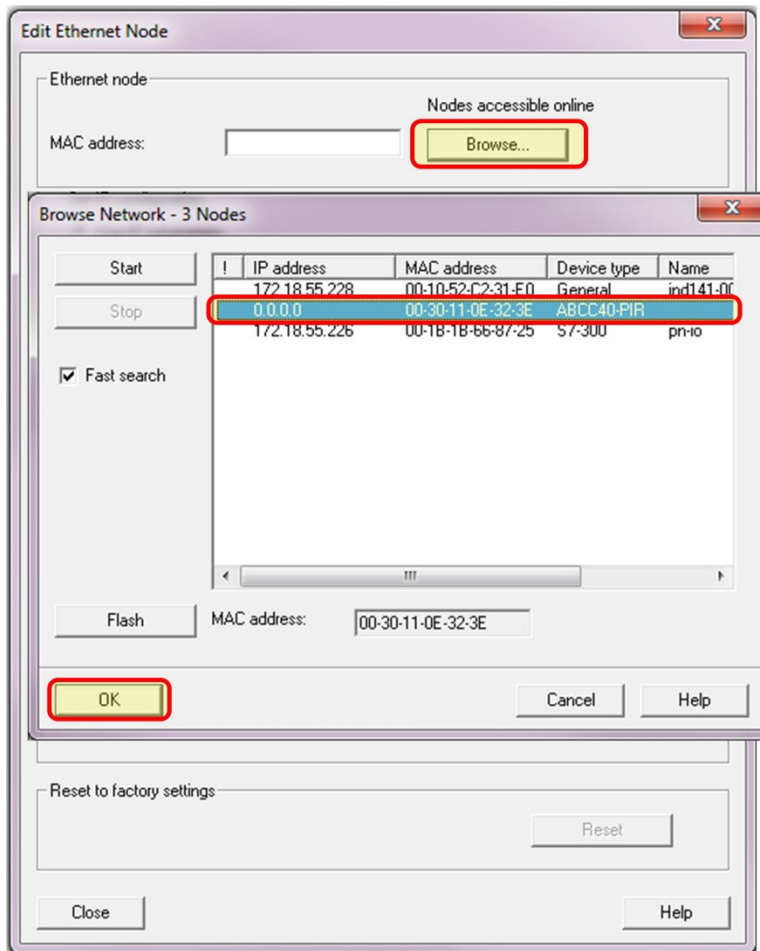


Figure 7-13: DCP Browse Function, Showing Network Node MAC Addresses

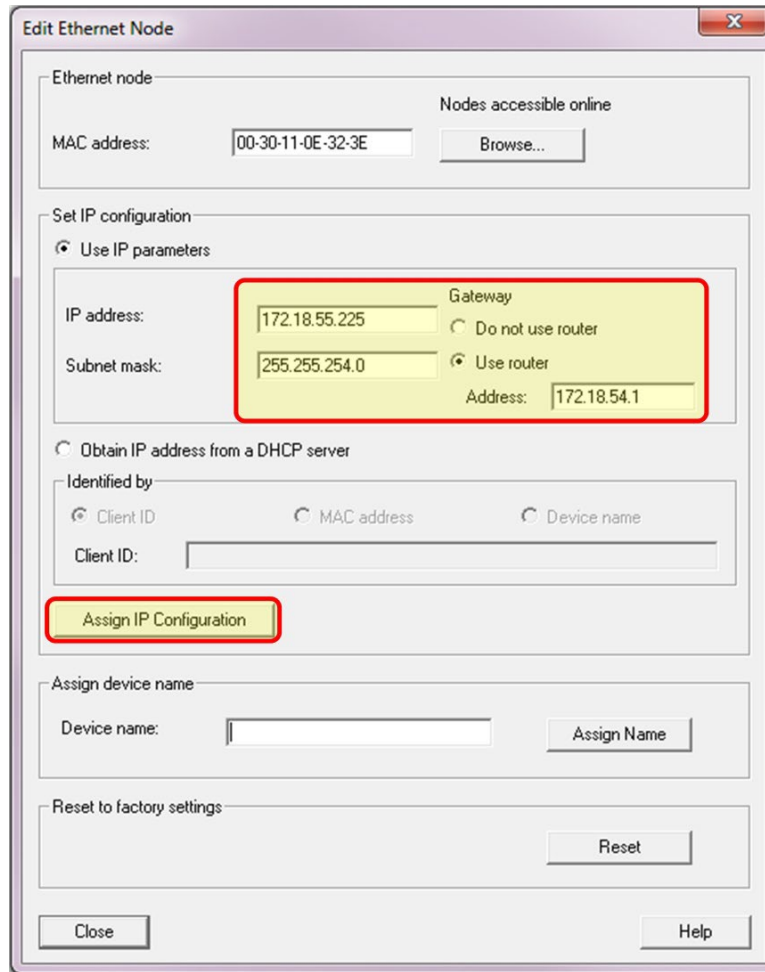


Figure 7-14: DCP Ethernet Node Editing

Once the other settings are configured, assign the device a name.

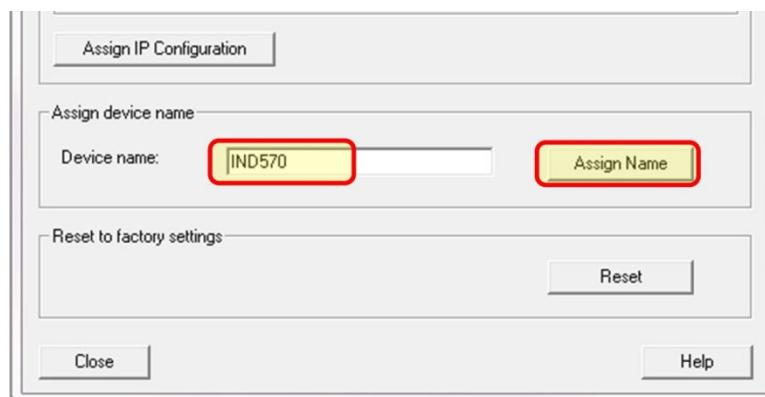


Figure 7-15: Assigning a Name to the New Device

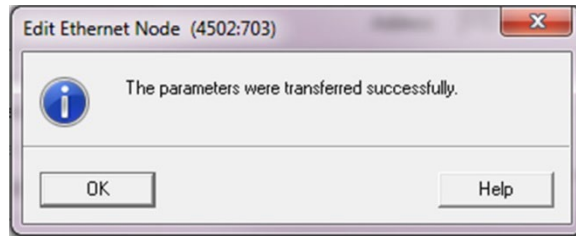


Figure 7-16: Confirmation of Transfer of Parameters

After the message is received that the parameters have been successfully transferred, you may close the DCP form. If the Data Format in the IND570 has been set up properly and the Network properly configured in the Siemens "HW Config." window, the terminal's PROFINET module should start communicating with the PLC.

7.9. Troubleshooting

If the IND570 does **not** communicate with the PLC, do the following:

- Confirm that both the IP Address configuration and the Device Name configuration have been assigned in the PLC (note that the Device Name must always be assigned using DCP). Cycle power on the IND570 to ensure that any updated settings take effect.
- Check for IP Address conflicts. Use a Ping command from a PC to verify addresses.
- Check physical wiring and network connections.
- Confirm that the IND570 settings for data type and IP Address assignment match those in the PLC and that each IND570 on the network has a unique address.
- Confirm that the number of message slots assigned within the IND570's setup menu match both the Input and Output assignment in the Siemen's HW Configuration Tool.
- If the communication interface in the IND570 was changed from another type (e.g. Ethernet/IP or ControlNet), a master reset of the IND570 may need to be performed.
- Replace the PROFINET interface kit if communication problems persist.

7.9.1. Status LEDs

The PROFINET interface card has a link activity LED on each of the two RJ45 connectors and four status LEDs indicators to indicate communication and fault status of the card. Figure 7-17 shows

the layout of the LED module, and Figure 7-18 shows the array of the LEDs on the card. Table 7-4 and Table 7-5 explain the meaning of the indicators.

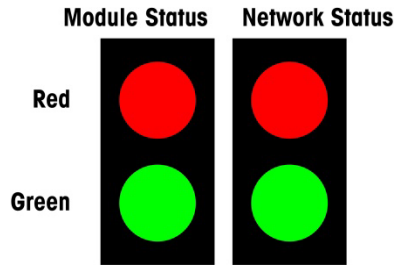


Figure 7-17: PROFINET Status Indicator LEDs

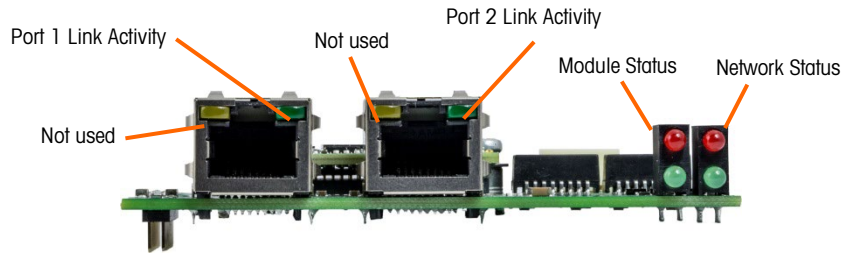


Figure 7-18: PROFINET Status Indicator LEDs

Table 7-4: Module Status LEDs

LED State	Description	Comments
Off	Not Initialized	<ul style="list-style-type: none"> No power Module in 'SETUP' or 'NW_INIT' state
Green	Normal Operation	<ul style="list-style-type: none"> Module has shifted from the 'NW_INIT' state
Green, 1 flash	Diagnostic Event(s)	<ul style="list-style-type: none"> Diagnostic event(s) present
Red	Exception error	<ul style="list-style-type: none"> Device in state EXCEPTION
	Fatal event	<ul style="list-style-type: none"> Major internal error (this indication is combined with a red network status LED)
Alternating Red/Green	Firmware update	<ul style="list-style-type: none"> Do NOT power off the module. Turning the module off during this phase could cause permanent damage.

Table 7-5: Network Status LEDs

LED State	Description	Comments
Off	Offline	<ul style="list-style-type: none"> No Power No connection with IO Controller
Green	Online (RUN)	<ul style="list-style-type: none"> Connection with IO Controller established
Green, 1 flash	Online (STOP)	<ul style="list-style-type: none"> IO Controller in STOP state IO data bad IRT synchronization not finished

LED State	Description	Comments
Green, blinking	Blink	<ul style="list-style-type: none"> Used by engineering tools to identify the node on the network
Red	Fatal event	<ul style="list-style-type: none"> Major internal error (this indication is combined with a red module status LED)
Red, 1 flash	Station Name error	<ul style="list-style-type: none"> Station Name not set
Red, 2 flashes	IP address error	<ul style="list-style-type: none"> IP address not set
Red, 3 flashes	Configuration error	<ul style="list-style-type: none"> Expected Identification differs from Real Identification

7.10. Siemens S7-300 Programming Examples

The following Figures show sample screen images of ladder logic programming examples for SIMATIC Step 7 software (version V5.5 + SP3).

■ **Note:** Complete versions of the examples can be downloaded at www.mt.com/IND570. The following screen images are provided for illustrative purposes only.

The following SIMATIC Step 7 screens for Integer, Division and Floating Point data formats only show an example of a particular Input and Output size configuration. The Connection Parameters I/O sizes must be appropriately configured with reference to the number of slots assigned in the IND570 PLC Data Format Message Slots settings. Table 7-2 and Table 7-3 show the relationship between the IND570 message slots and the SIMATIC Step 7 I/O sizing for Integer, Division and Floating Point data formats.

7.10.1. Floating Point Mode Program Example

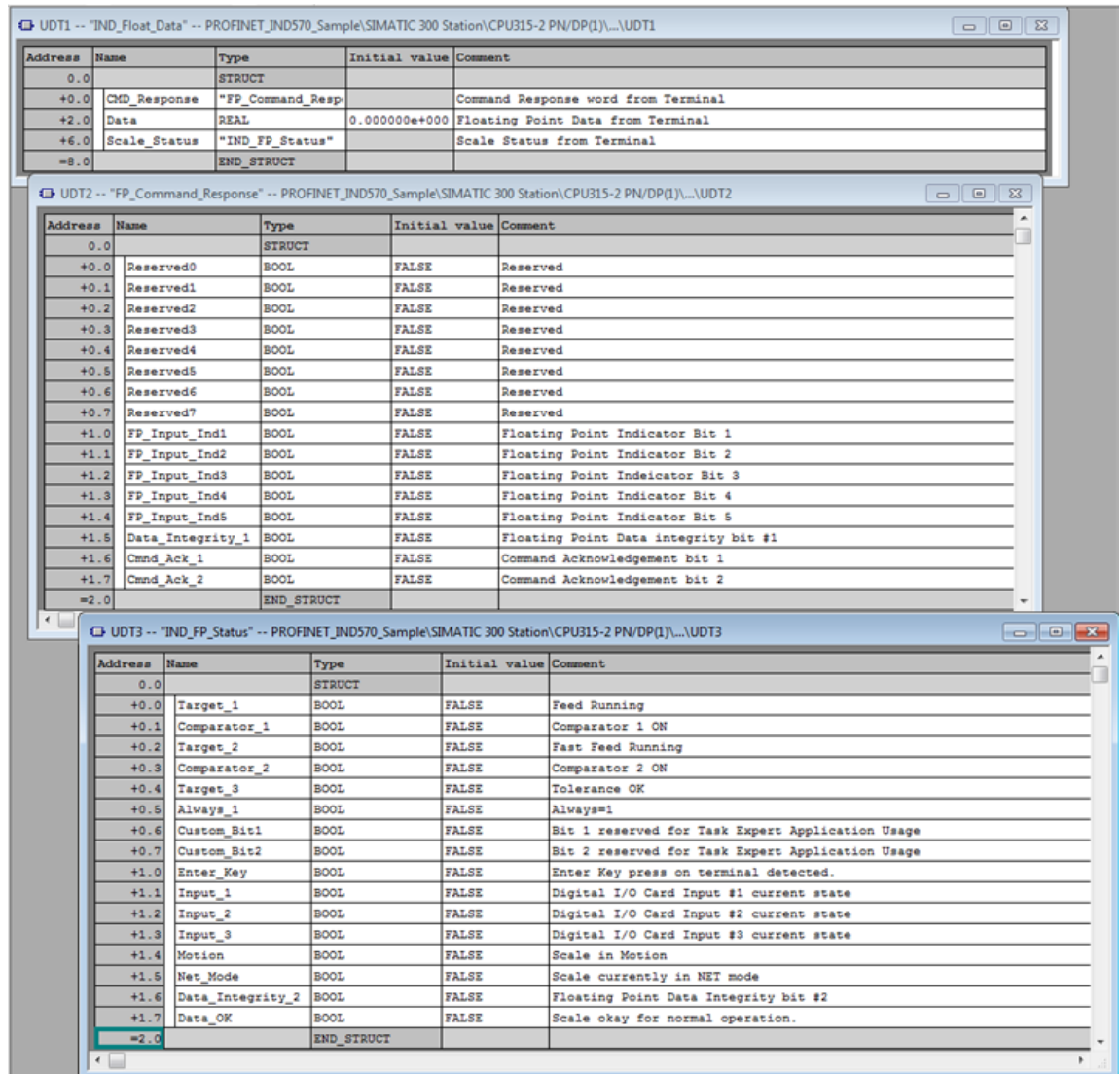


Figure 7-19: UDT Definitions used in the Floating Point sample program

The UDT's (User Defined Types) used in the sample program are:

UDT1 = IND570 Float Data. Format of the Floating Point data that comes back from the terminal, including the status registers with their supporting UDT's.

UDT2 = Command Response. Status register indicating the response of the IND570 Terminal to a command sent to it over the Field Bus.

UDT3 = Indicator Floating Point Status: Indicates the state of the measuring device (Scale or Flowmeter).

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Msg_Slot1	"IND_Float_Data"		IND570 Message Slot 1 (see PLC Data Format in IND570's Setup)
+8.0	Msg_Slot2	"IND_Float_Data"		IND570 Message Slot 2 (see PLC Data Format in IND570's Setup)
+16.0	Msg_Slot3	"IND_Float_Data"		IND570 Message Slot 3 (see PLC Data Format in IND570's Setup)
+24.0	Msg_Slot4	"IND_Float_Data"		IND570 Message Slot 4 (see PLC Data Format in IND570's Setup)
=32.0		END_STRUCT		

Figure 7-20: Data Block 1 (DB1) used in the Floating Point sample program

Data Block 1 (DB1) has defined all four (4) possible message slots for the IND570. This is NOT necessary for proper operation of the program, but does allow data to be stored in the data block if the program is designed to collect data from all 4 slots.

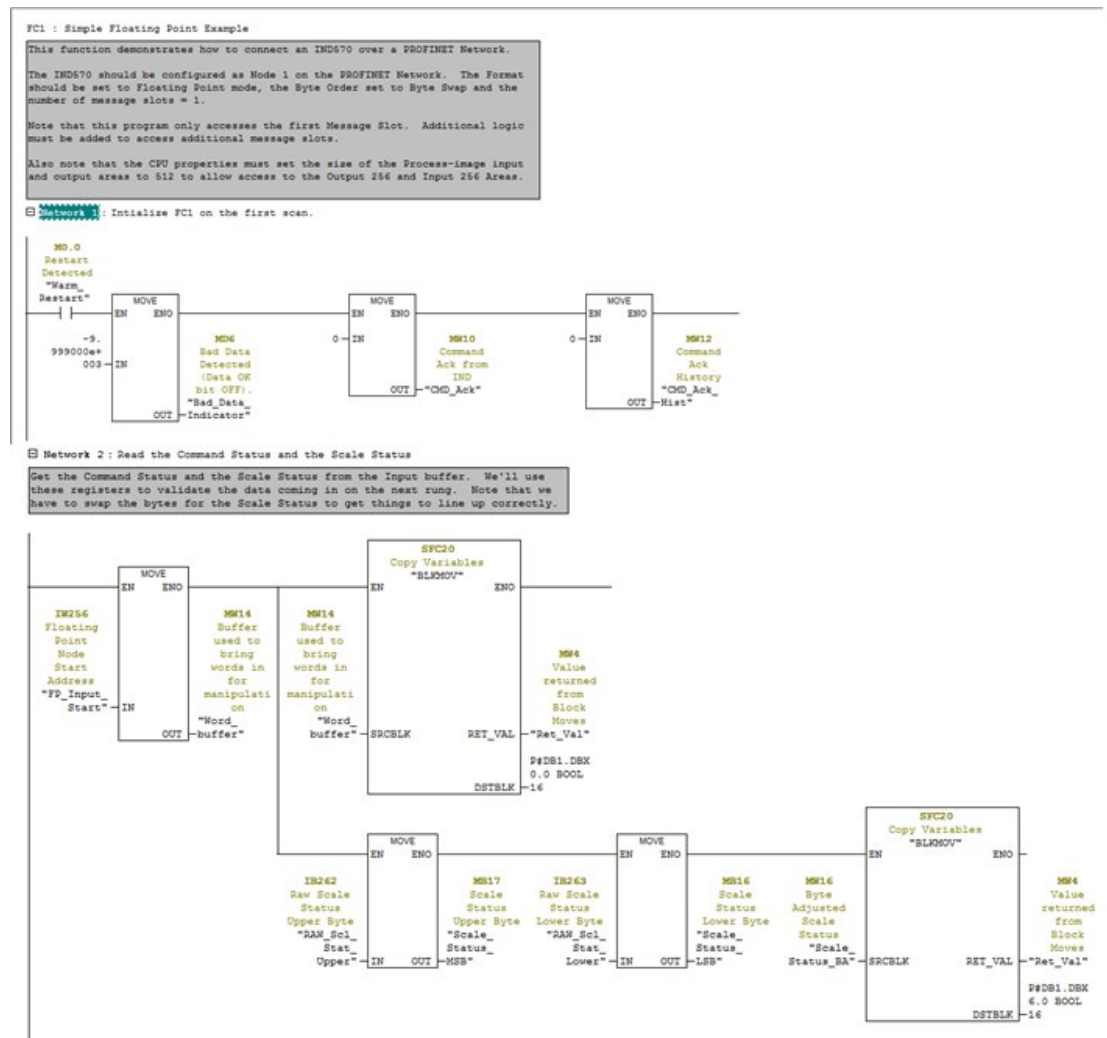


Figure 7-21: Networks 1 & 2 of FC1

When the program first starts up, Network 1 initializes it with its starting values.

Network 2 reads in the Command Response and the Floating Point Status, does any necessary byte swapping, and then stores them in their associated message slot positions in DB1. The bits will be used later.

□ Network 3: Update the data read from the Profibus IND

Only update the data if the Data OK bit is on, and the Data Integrity bits match (indicating that the data inbetween the bits is valid). Otherwise, throw the data away.

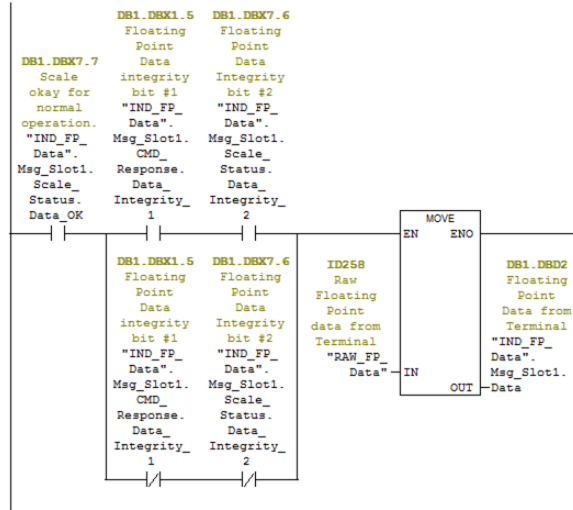


Figure 7-22: Network 3 of FC1

Network 3 utilizes the Data OK and the Data Integrity status bits read in Network 2 to determine if the Floating Point data is valid. If it is, then copy the data to its associated position in DB1.

□ Network 4 : Data OK bit is OFF

If the Data OK bit is off, signal the process by setting the data to a known bad value that flags everyone that something is wrong at the scale. Also, set the Data_Displayed variable to -1 to flag all other routines that there is a problem.

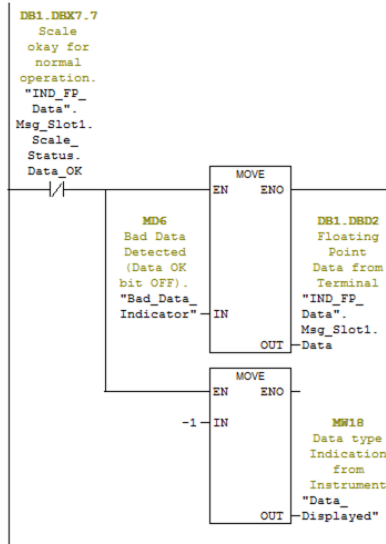


Figure 7-23: Network 4 of FC1

Network 4 acts on the Data OK bit if it is off, indicating that the instrument is not in a valid mode to send data. In that case, set the output to -9999.0 ("Bad_Data_Indicator") to flag the operator that something is wrong. Set the Data Displayed indicator to -1 to flag that something is wrong with the data coming back.

Network 5: Read the Command Acknowledgement status

Grab the two Command Acknowledgement bits out of the status and stuff them into a word (MW10) that we can use in the following rung to determine if a command has been acknowledged.

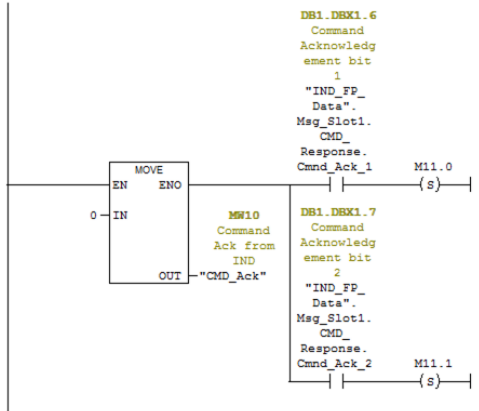


Figure 7-24: Network 5 of FC1

Network 5 reads the Command Acknowledge bits from the IND570 and turns the result into an Integer word from the returned bits. The integer will be used in the following networks to act on any commands that are sent.

Network 6: Handle Command Acknowledged bit

When a Command Ack is detected (CMD_Ack is NOT Equal to CMD_Ack_Hist) the turn on the Command_Acked bit and run a timer. When the Timer expires, set the history to the new CMD_Ack state, which shuts everything off.

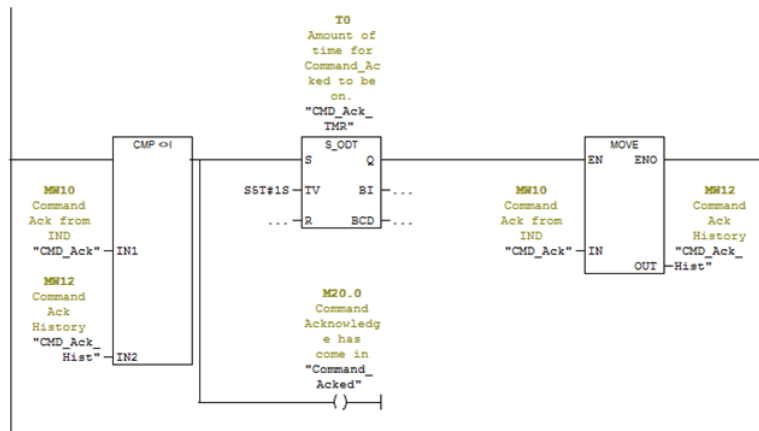


Figure 7-25: Network 6 of FC1

Network 6 looks for a change in the Command Acknowledge state. If one is found, it turns on the Command_Acked bit and runs a timer. When the timer expires, the history is updated, which in turn causes the Command_Acked bit to turn off. The Command_Acked bit is used later when processing commands to the IND570.

Network 7: Update the Command to the IND

The IND only acts on a command if it sees a change from the previous command. That means if you need to send another Tare after the first Tare, there must be another command between them - such as Display Net Weight.

This rung waits until the comand changes, then it writes the Command, and the Command Data to the PLC's output buffer to the IND. It then updates the history.

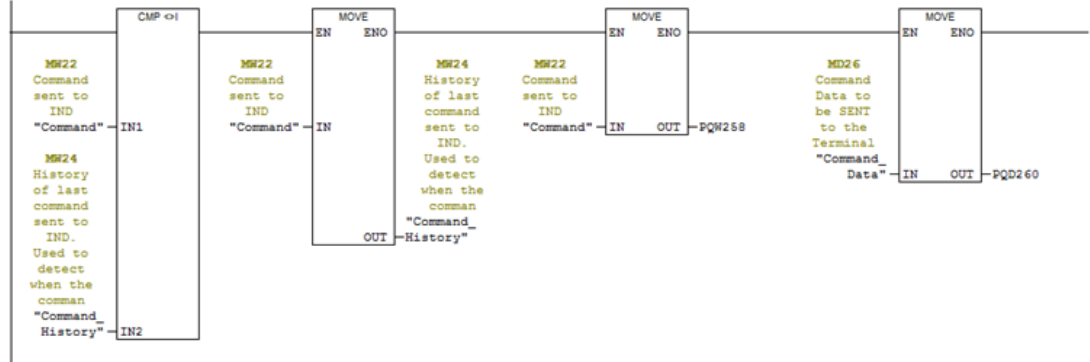


Figure 7-26: Network 7 of FC1

Network 7 looks for a change in the Command being sent to the IND570. If a new command has been issued then move it to the output buffer and update the history. Programming the command in this manner allows the freedom of manually updating the command output buffer if desired for troubleshooting purposes.

□ Network 8 : Returned floating Point value type indicator.

Map the Floating Point Input Indicator bits into a word so we can decode what type of data the Indicator is currently returning to us. See table in the PLC Interface Manual for a mapping of these values.

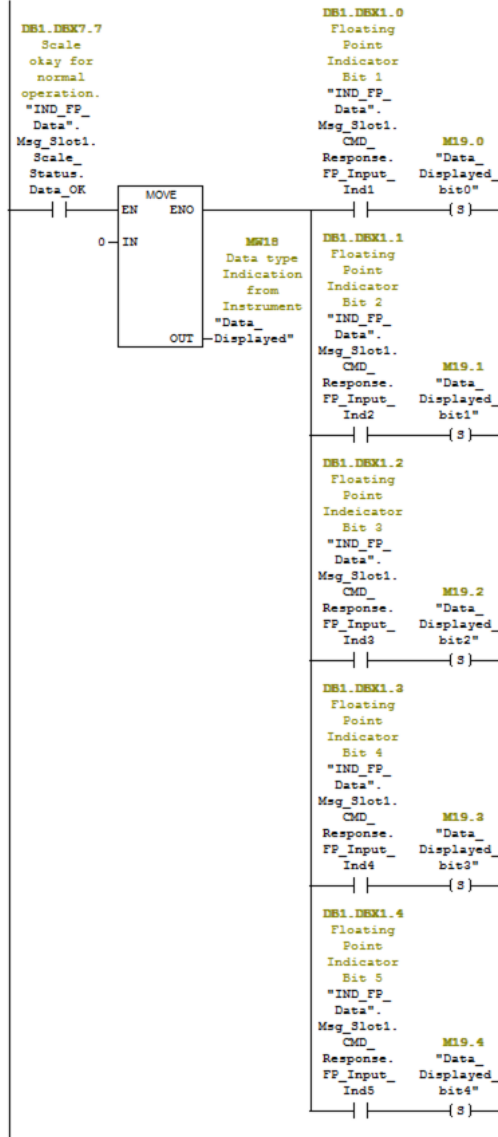


Figure 7-27: Network 8 of FC1

Network 8 reads the Floating Point Indicator bits in the Command Status Word and creates an Integer from the result that is used later to determine what kind of data is being returned from the IND570.

□ Network 9 : Various example commands

This rung allows triggering of some sample commands to the Indicator. Your own logic does not need to function this way. It only needs to write a value to "Command", and if necessary, to "Command_Data."

The MOV instruction for IND_FP_Data is only so that the returned data is easy to read after one of the command bits has been set.

Other commands that can be sent to the Indicator can be found in the PLC Interface Manual.

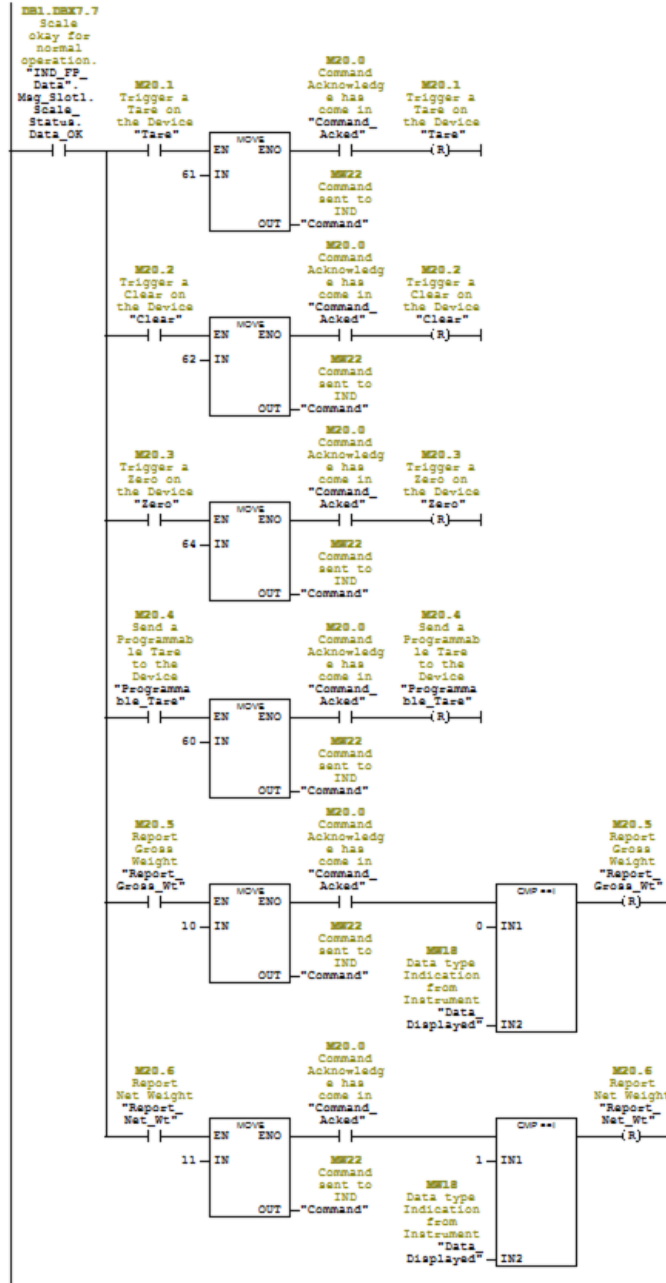


Figure 7-28: Network 9 of FC1

Network 9 issues commands to the IND570 as the user sets the desired command bits. It then waits for the commands to be acknowledged before clearing the bits that were set. If a different type of data for display is being requested, then wait until the IND570 indicates that the correct data is being displayed before clearing the command bit.

7.10.1.1. Running the Sample Program

The sample program can be run from the Variable Access Table as shown in Figure 7-29 and Figure 7-30 below.

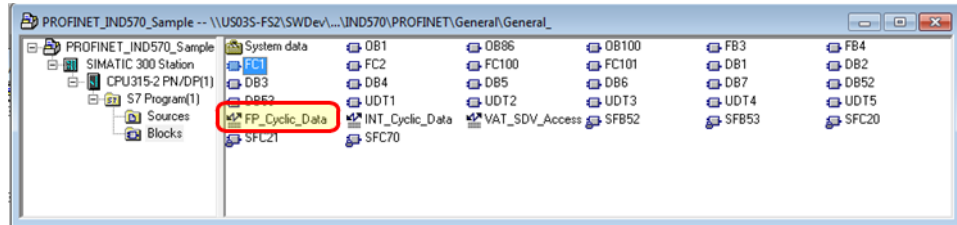


Figure 7-29: FP_Cyclic_Data

VAT_Float_Cyclic_Data is a Variable Access Table that allows the user to monitor the processed cyclic data and issue commands to the IND570 as shown below:

Address	Symbol	Display format	Status value	Modify value
1	# Cyclic Data being returned.			
2	DB1.DB0 2	"IND_FP_Data".Msg_Slot1.Data	FLOATING_POINT	3.83
3	# Cyclic Commands			
4	M 20.1	"Tare"	BOOL	false
5	M 20.2	"Clear"	BOOL	false
6	M 20.3	"Zero"	BOOL	false
7	M 20.4	"Programmable Tare"	BOOL	false
8	MD 26	"Command_Data"	FLOATING_POINT	0.0 2.0
9	M 20.5	"Report_Gross_Wt"	BOOL	false
10	M 20.6	"Report_Net_Wt"	BOOL	false
11	# Command Status Flags			
12	MW 18	"Data_Displayed"	DEC	0
13	MW 10	"CMD_Ack"	DEC	0
14	DB1.DBX 1.5	"IND_FP_Data".Msg_Slot1.CMD_Response.Data_Integrity_1	BOOL	false
15	DB1.DBX 1.6	"IND_FP_Data".Msg_Slot1.CMD_Response.Cmd_Ack_1	BOOL	false
16	DB1.DBX 1.7	"IND_FP_Data".Msg_Slot1.CMD_Response.Cmd_Ack_2	BOOL	false
17	# Scale Status Flags			
18	DB1.DBX 6.0	"IND_FP_Data".Msg_Slot1.Scale_Status.Target_1	BOOL	false
19	DB1.DBX 6.1	"IND_FP_Data".Msg_Slot1.Scale_Status.Comparator_1	BOOL	false
20	DB1.DBX 6.2	"IND_FP_Data".Msg_Slot1.Scale_Status.Target_2	BOOL	false
21	DB1.DBX 6.3	"IND_FP_Data".Msg_Slot1.Scale_Status.Comparator_2	BOOL	false
22	DB1.DBX 6.4	"IND_FP_Data".Msg_Slot1.Scale_Status.Target_3	BOOL	false
23	DB1.DBX 6.5	"IND_FP_Data".Msg_Slot1.Scale_Status.Always_1	BOOL	true
24	DB1.DBX 6.6	"IND_FP_Data".Msg_Slot1.Scale_Status.Custom_Bit1	BOOL	false
25	DB1.DBX 6.7	"IND_FP_Data".Msg_Slot1.Scale_Status.Custom_Bit2	BOOL	false
26	DB1.DBX 7.0	"IND_FP_Data".Msg_Slot1.Scale_Status.Enter_Key	BOOL	true
27	DB1.DBX 7.1	"IND_FP_Data".Msg_Slot1.Scale_Status.Input_1	BOOL	false
28	DB1.DBX 7.2	"IND_FP_Data".Msg_Slot1.Scale_Status.Input_2	BOOL	false
29	DB1.DBX 7.3	"IND_FP_Data".Msg_Slot1.Scale_Status.Input_3	BOOL	false
30	DB1.DBX 7.4	"IND_FP_Data".Msg_Slot1.Scale_Status.Motion	BOOL	false
31	DB1.DBX 7.5	"IND_FP_Data".Msg_Slot1.Scale_Status.Net_Mode	BOOL	false
32	DB1.DBX 7.6	"IND_FP_Data".Msg_Slot1.Scale_Status.Data_Integrity_2	BOOL	false
33	DB1.DBX 7.7	"IND_FP_Data".Msg_Slot1.Scale_Status.Data_OK	BOOL	true

Commands. Set these bits to trigger their associated commands. Bits will be cleared when the IND570 acknowledges the command.

Filtered Floating Point Data from IND570.

Command Data. Sent with commands that require Data, such as the Programmable Tare

Floating Point Data Filter bits: Data_OK bit must be on and Data Integrity bits 1 & 2 must be the same polarity for the current Floating Point data sample to be valid.

Figure 7-30: FP_Cyclic_Data Description

7.10.2. Integer Mode Program Example

The image shows two screenshots of the SIMATIC Manager's Variable Declaration Table (UDT) editor. The top screenshot shows UDT4, and the bottom screenshot shows UDT5.

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Data	REAL	0.000000e+000	Integer Data from Terminal Multiplied by a Scaling Value
+4.0	Status	"IND_INT_Status"		Scale Status from the Terminal
=6.0		END_STRUCT		

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Target_1	BOOL	FALSE	Feed
+0.1	Target_2	BOOL	FALSE	Fast_Feed
+0.2	Target_3	BOOL	FALSE	Tolerance_OK
+0.3	Comparator5	BOOL	FALSE	Comparator 5 ON
+0.4	Comparator4	BOOL	FALSE	Comparator 4 ON
+0.5	Comparator3	BOOL	FALSE	Comparator 3 ON
+0.6	Comparator2	BOOL	FALSE	Comparator 2 ON
+0.7	Comparator1	BOOL	FALSE	Comparator 1 ON
+1.0	Enter_Key	BOOL	FALSE	Enter Key press on terminal detected.
+1.1	Input_1	BOOL	FALSE	Digital I/O Card Input #1 current State
+1.2	Input_2	BOOL	FALSE	Digital I/O Card Input #2 current State
+1.3	Input_3	BOOL	FALSE	Digital I/O Card Input #3 current State
+1.4	Motion	BOOL	FALSE	Scale in Motion
+1.5	Net_Mode	BOOL	FALSE	Scale currently in NET mode
+1.6	Update_In_Progress	BOOL	FALSE	Ignore Terminal data while Update in Progress=1
+1.7	Data_OK	BOOL	FALSE	Scale okay for normal operation
=2.0		END_STRUCT		

Figure 7-31: UDT Definitions used in the Integer sample program

The UDT's (User Defined Types) used in the sample program are:

UDT4 = IND570 Integer Data. Format of the Floating Point data that comes back from the terminal, including the status registers with their supporting UDT's.

UDT5 = Indicator Integer Status: Indicates the state of the measuring device (Scale or Flowmeter).

The image shows a screenshot of the SIMATIC Manager's Data Block (DB) editor for Data Block 2 (DB2).

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Msg_Slot1	"IND_Integer_Data"		IND570 Message Slot 1
+6.0	Msg_Slot2	"IND_Integer_Data"		IND570 Message Slot 2
+12.0	Msg_Slot3	"IND_Integer_Data"		IND570 Message Slot 3
+18.0	Msg_Slot4	"IND_Integer_Data"		IND570 Message Slot 4
=24.0		END_STRUCT		

Figure 7-32: Data Block 2 (DB2) used in the Integer sample program

Data Block 2 (DB2) has defined all four (4) possible message slots for the IND570. This is NOT necessary for proper operation of the program, but does allow data to be stored in the data block if the program is designed to collect data from all 4 slots.

FC2 : Simple Integer Example

This function demonstrates how to connect to an IND570 over a ProfiBus Network using the Integer Mode format.

The IND570 should be configured as Node 4 on the ProfiBus Network with Shared Data Enabled. The Format should be set to Integer Mode, the Byte Order set to Byte Swap and the number of message slots = 4.

Note that this program only accesses the first Message Slot. Additional logic must be added to access the additional message slots.

Also note that the CPU properties must set the size of the Process-image input and output areas to 512 to allow access to the Output 256 and Input 256 Areas/ "I/O 23 Wrds" should be selected from the GSD file for the IND570's I/O configuration.

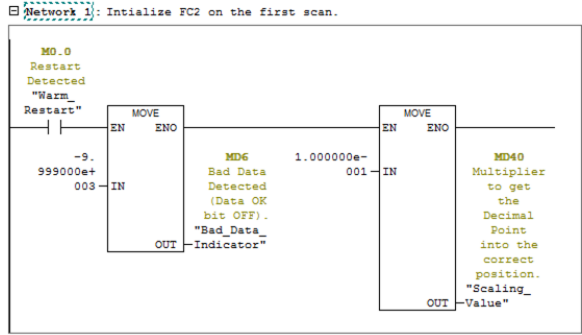


Figure 7-33: Network 1 of FC2

Network 1 initializes the program when it first starts up with its starting values.

Network 2 : Copy the Input Status to the Scale Status Bits in DB2

First, swap the bytes of the status word to get them to line up properly. The BLKMOV function can't copy directly from an Input Buffer, so we have to first move it to MW44. Then we can use the BLKMOV, which will map the bits to their proper locations in the DB2 Scale Status area.

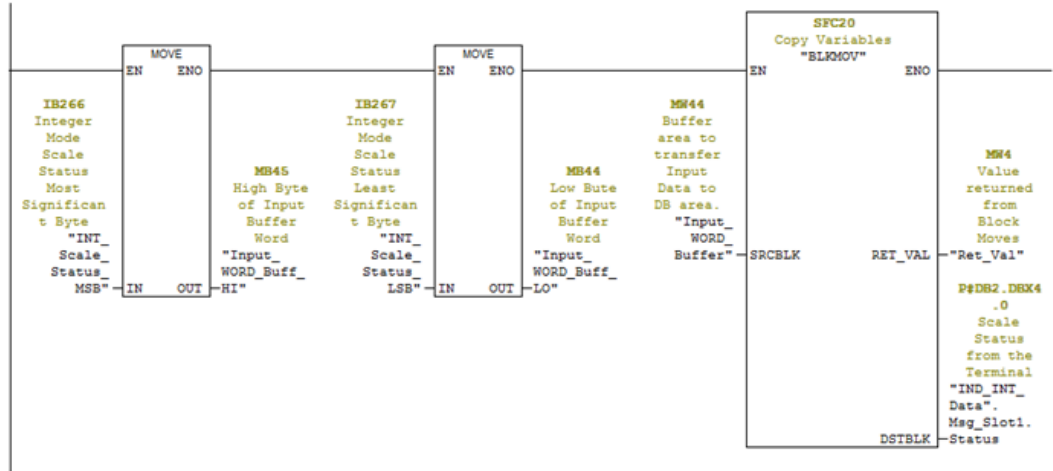


Figure 7-34: Network 2 of FC2

Network 2 Swaps the bytes of the Scale Status Word, then uses a BLKMOV to map the Status Bits to their proper locations for later use.

□ Network 3 : Get the data from the Terminal and Normalize it to the Increment

We need to 'Normalize' the incoming Integer Data by multiplying it by the multiplier that is defined for the scale (that value is Hard Coded in Network 1 above, and is based on the resolution defined by the terminal's increment setting contained in the scale setup that is in the IND terminal itself). But first, we have to convert the Integer data coming back to a Double Integer, and then convert the Double Integer into a Real data type. After all of that, we can finally multiply the data by the Increment and store the result in the Message Slot data area.

Note that the Data OK bit MUST be ON, and the Update In Progress bit MUST be OFF, or the data coming back should be ignored.

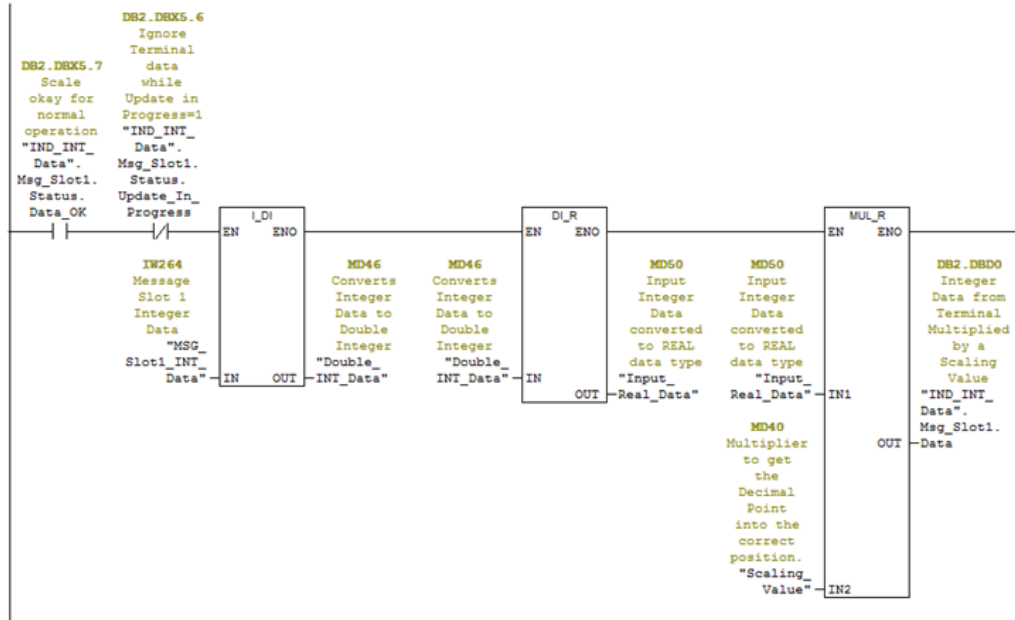


Figure 7-35: Network 3 of FC2

Network 3 filters the Data Input with the Data OK and the Update In Progress bits from the Status word. If the Data OK bit is off, or the Update in Progress bit is set, then the data may be invalid and should be discarded. If the filter bits are okay, then the Integer value needs to be converted to a Real data type for later use. To do that, the program must first convert the Integer to a Double Integer, and then convert the Double Integer to Real value. Finally, the Real value is multiplied by a Scaling value that puts the decimal point into the same place that the Terminal Display uses. The value is then transferred to DB2.

□ Network 4: Data OK bit is OFF

If the Data OK bit is off, signal the process by setting the data to a known bad value that flags everyone that something is wrong at the scale.

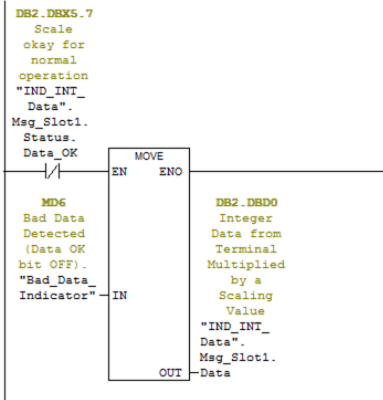


Figure 7-36: Network 4 of FC2

Network 4 looks at the Data OK bit. If the Data OK bit is off, then the terminal is not in a proper state to deliver valid data. In that case, the "Bad Data" indicator value is written to DB2 instead of the data.

Network 5 : Sample Commands to the Terminal.

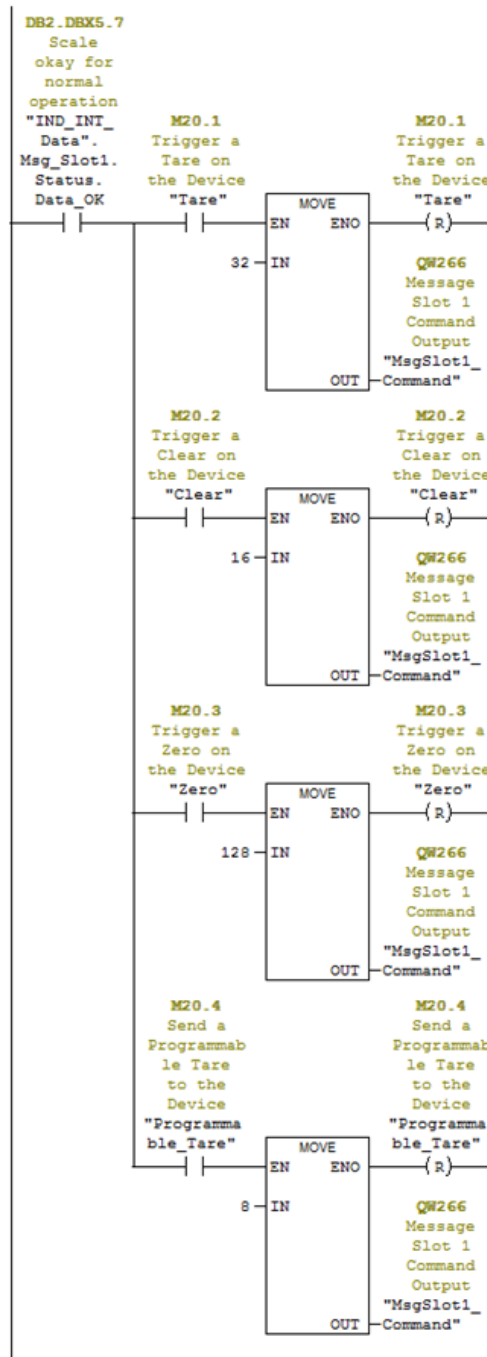


Figure 7-37: Network 5 of FC2

If the terminal is in a state to receive commands (the Data OK bit is on) then Network 5 looks at the command trigger bits which could be set either by other networks in the program, or directly by a user from a VAT Table. If a command trigger bit is found to be on, write the command value to the Command Output and clear the trigger bit. This network is intended as a sample for how commands may be sent to the Terminal.

Network 6 : More sample commands to the terminal

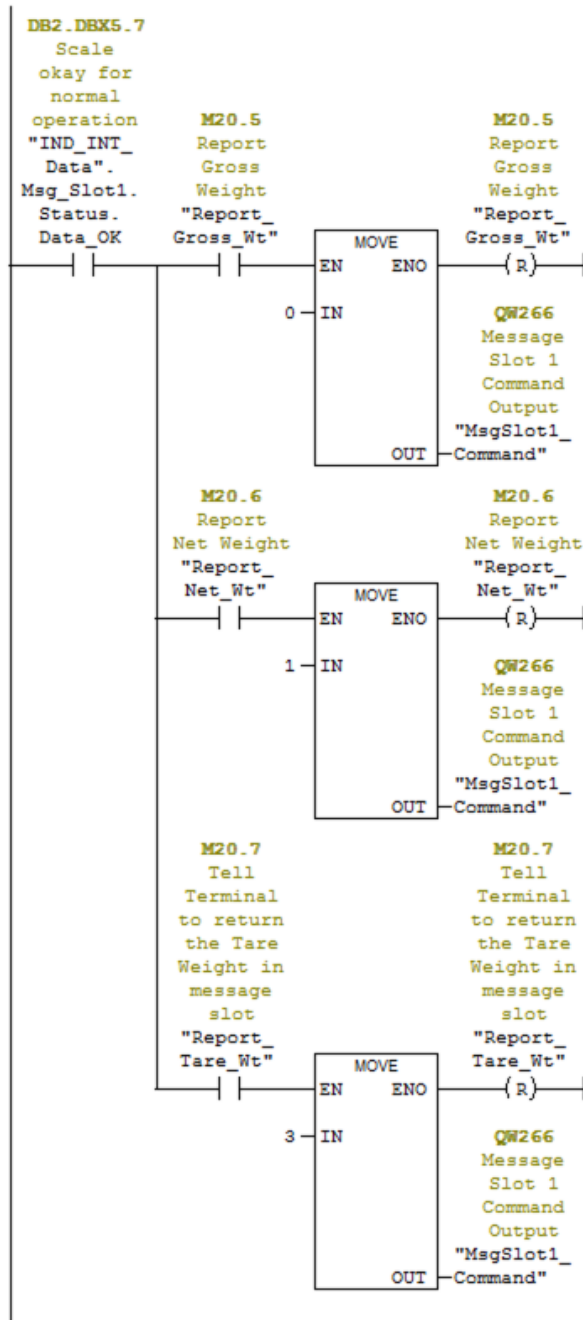


Figure 7-38: Network 6 of FC2

Network 6 shows some additional commands that could be sent to the terminal. Note that these commands are also filtered with the Data OK bit.

Network 7: Output the Normalized Data to the Terminal.

Take the Floating Point Data that was written to the Normalized_Data_Out variable and divide it by the Increment value to generate an Integer Value that can be output to the Terminal.

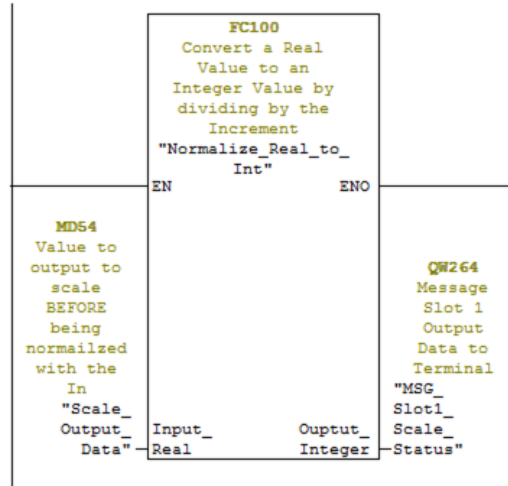


Figure 7-39: Network 7 of FC2

Network 7 shows how Real Typed data can be converted and sent as an Integer value to the terminal to supplement commands. This data can be used to pre-load a Tare value, set a Target, set up Tolerances, etc. Note that FC100 is provided with the sample program, but is not covered here.

7.10.2.1. Running the Sample Program

The sample program can be run from the Variable Access Table as shown in Figure 7-40 and Figure 7-41 below.

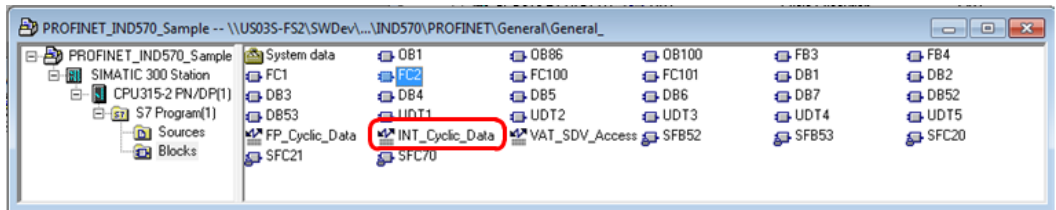


Figure 7-40: INT_Cyclic_Data

INT_Cyclic_Data is a Variable Access Table that allows the user to monitor the processed cyclic data and issue commands to the IND570 as shown below:

Address	Symbol	Display format	Status value	Modify value
// Cyclic Data being returned				
DB2.DBD 0	"IND_INT_Data".Msg_Slot1.Data	FLOATING_POINT	3.83	
MD 40	"Scaling_Value"	FLOATING_POINT	0.001	0.001
// Cyclic Commands				
M 20.1	"Tare"	BOOL	false	
M 20.2	"Clear"	BOOL	false	
M 20.3	"Zero"	BOOL	false	
M 20.4	"Programmable_Tare"	BOOL	false	
MD 54	"Scale_Output_Data"	FLOATING_POINT	2.0	2.0
M 20.5	"Report_Gross_Wt"	BOOL	false	
M 20.6	"Report_Net_Wt"	BOOL	false	
// Scale Status Flags				
DB2.DBX 4.0	"IND_INT_Data".Msg_Slot1.Status.Target_1	BOOL	false	
DB2.DBX 4.1	"IND_INT_Data".Msg_Slot1.Status.Target_2	BOOL	false	
DB2.DBX 4.2	"IND_INT_Data".Msg_Slot1.Status.Target_3	BOOL	false	
DB2.DBX 4.3	"IND_INT_Data".Msg_Slot1.Status.Comparator5	BOOL	false	
DB2.DBX 4.4	"IND_INT_Data".Msg_Slot1.Status.Comparator4	BOOL	false	
DB2.DBX 4.5	"IND_INT_Data".Msg_Slot1.Status.Comparator3	BOOL	false	
DB2.DBX 4.6	"IND_INT_Data".Msg_Slot1.Status.Comparator2	BOOL	false	
DB2.DBX 4.7	"IND_INT_Data".Msg_Slot1.Status.Comparator1	BOOL	false	
DB2.DBX 5.0	"IND_INT_Data".Msg_Slot1.Status.Enter_Key	BOOL	true	
DB2.DBX 5.1	"IND_INT_Data".Msg_Slot1.Status.Input_1	BOOL	false	
DB2.DBX 5.2	"IND_INT_Data".Msg_Slot1.Status.Input_2	BOOL	false	
DB2.DBX 5.3	"IND_INT_Data".Msg_Slot1.Status.Input_3	BOOL	false	
DB2.DBX 5.4	"IND_INT_Data".Msg_Slot1.Status.Motion	BOOL	false	
DB2.DBX 5.5	"IND_INT_Data".Msg_Slot1.Status.Net_Mode	BOOL	false	
DB2.DBX 5.6	"IND_INT_Data".Msg_Slot1.Status.Update_In_Progress	BOOL	false	
DB2.DBX 5.7	"IND_INT_Data".Msg_Slot1.Status.Data_OK	BOOL	true	

Figure 7-41: INT_Cyclic_Data Description

7.10.3. Shared Data Access Overview

In order to access Shared Data, a program must provide the following information to the Read and Write routines:

- Class Code
- Instance Number
- Attribute Number
- Length

This information can be found in the IND570 Shared Data Reference Manual (part number 3025337) for each Shared Data variable. For example, here is how you would find that information for a 'WT' type Shared Data variable:

2 Scale Data

2.1. Scale Functionality

2.1.1. Dynamic Scale Weight (WT)

Class

Access: "Read Only" Access.	
Class Code: 0x68	Data Type: D
Instances: 1	Instance 1 = Scale platforms 1

Instance

Length

2.1.1.1. Attributes

wt0100	Composite wt block	Struct	Na	Composite of entire block
wt0101	Displayed Gross Weight	S13	rt	Rounded Gross Weight shown in selected increment size.
wt0102	Displayed Net Weight	S13	rt	Rounded Net Weight shown in selected increment size.
wt0103	Weight Units	S4	rt	lb pounds, kg kilograms, grams , oz ounces, oztroy , dwt pennyweights, metric tons , ton , or custom units name
wt0104	3 rd Weight Unit Gross Weight	S13	rt	Shows the current displayed gross weight converted to 3 rd units
wt0105	3 rd Weight Unit Net Weight	S13	rt	Shows the current displayed net weight converted to 3 rd units
wt0106	Third Weight Unit	S7	rt	lb pounds, kg kilograms, grams , oz ounces, lb-oz pounds & ounces, oztroy , ounces, dwt pennyweights, metric tons, ton , or custom units name
wt0108	Displayed Rate	S13	rt	

Attribute

Figure 7-42: Identifying Shared Data Class, Instance, Attribute, and Length

If you have used Shared Data Variable names before, then you are already familiar with using the Instance and Attribute in the name definition as shown in Figure 7-43:

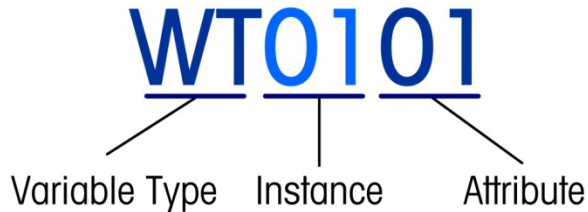


Figure 7-43: Construction of Shared Data Names

This information can help you in setting up your program to read or write the Shared Data variables that you need to access.

The method to access Shared Data in the IND570 Terminal is identical for both the Floating Point and Integer modes using PROFINET. Both sample programs use identical routines and variables, and are covered here as a single process.

The Shared Data Access over PROFINET shown here makes use of the system SFB blocks SFB52 (RDREC DP) and SFB53 (WRREC DP) to read and write information to the IND570 using Acyclic messages over the PROFINET link.

Any Shared Data Variable access (both read and write) requires that the Class Code, Instance, Attribute, and Length all be embedded in a message written to the IND570. Note that this information is included in both message blocks below, and must be populated before the operation is started.

The top screenshot shows the Read Buffer (DB5) structure:

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Class_Code	INT	0	Shared Data Variable Class Code
+2.0	Instance	INT	0	Instance (usually a scale or flowmeter number)
+4.0	Attribute	INT	0	Attribute pointer into the Class Code - selects specific Shared Data Variable
+6.0	Length	INT	0	Length of message read back (Read Command)
+8.0	Data_Buffer	ARRAY[0..25]		Data sent back from Terminal
+4.0		DWORD		

The bottom screenshot shows the Write Buffer (DB6) structure:

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Class_Code	INT	0	Shared Data Variable Class Code
+2.0	Instance	INT	0	Instance (usually a scale or flowmeter number)
+4.0	Attribute	INT	0	Attribute pointer into the Class Code - selects specific Shared Data Variable
+6.0	Length	INT	0	Length of message to send to Terminal (Write Command)
+8.0	Data_Buffer	ARRAY[0..25]		Data to send to Terminal
+4.0		DWORD		

Figure 7-44: Read Buffer (DB5) and Write Buffer (DB6)

A Shared Data Write is the simplest function since it requires only a call to SFB53 (WRREC) with the data to be written to the specified Shared Data variable, to complete the action.

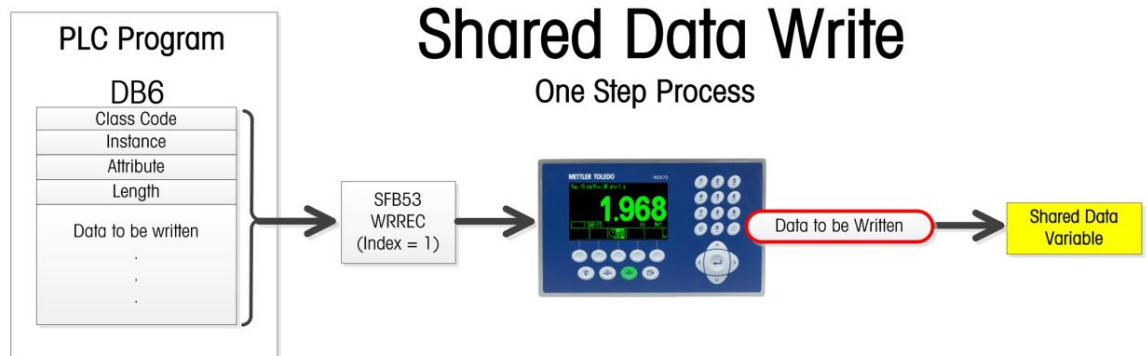


Figure 7-45: Shared Data Write – One Step Process

A Shared Data Read is a two-step process where a call to SFB53 (WRREC) must be done first to tell the IND570 what shared data variable to read. The write is then followed by a call to SFB52 (RDREC) to read the result back from the IND570.

Shared Data Read

Two Step Process

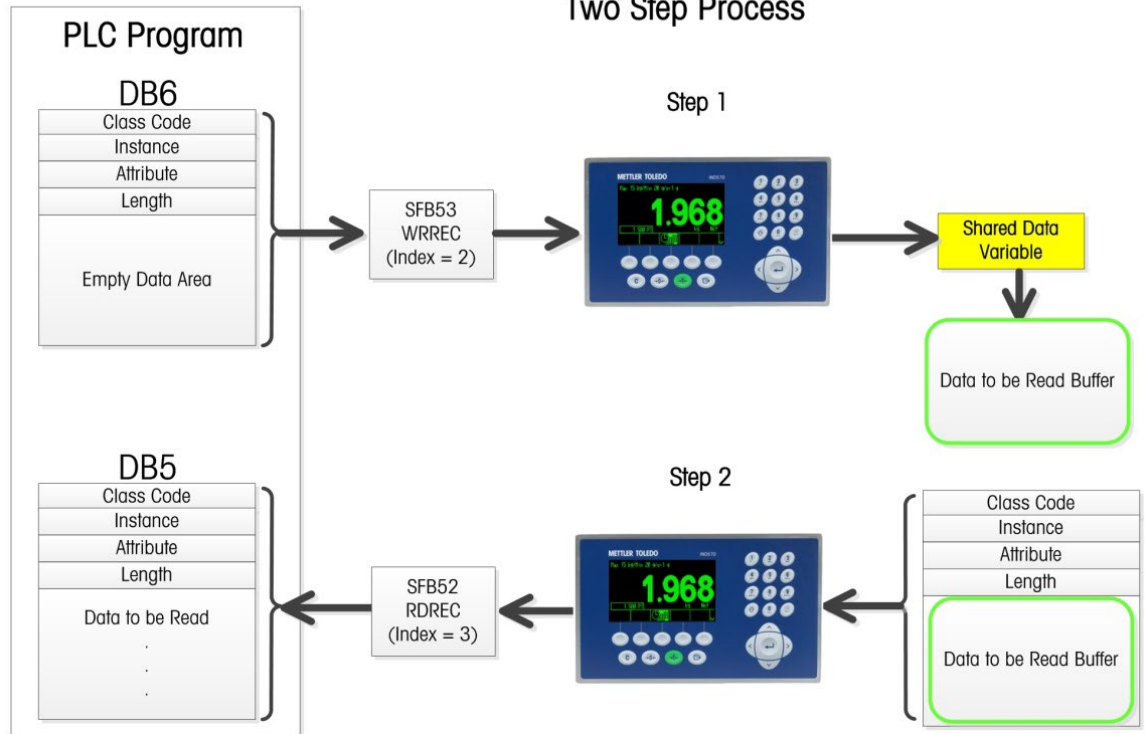


Figure 7-46: Shared Data Read – Two Step Process

When the Shared Data Read process is complete, the Class Code, Instance, and Attribute returned from the IND570 are compared to the requested values to make sure that the proper request was fulfilled.

7.10.4. Shared Data Access Program Details

7.10.4.1. OB1 Program details

In addition to the call to either the Floating Point or Integer cyclic data function, OB1 has the following logic added:

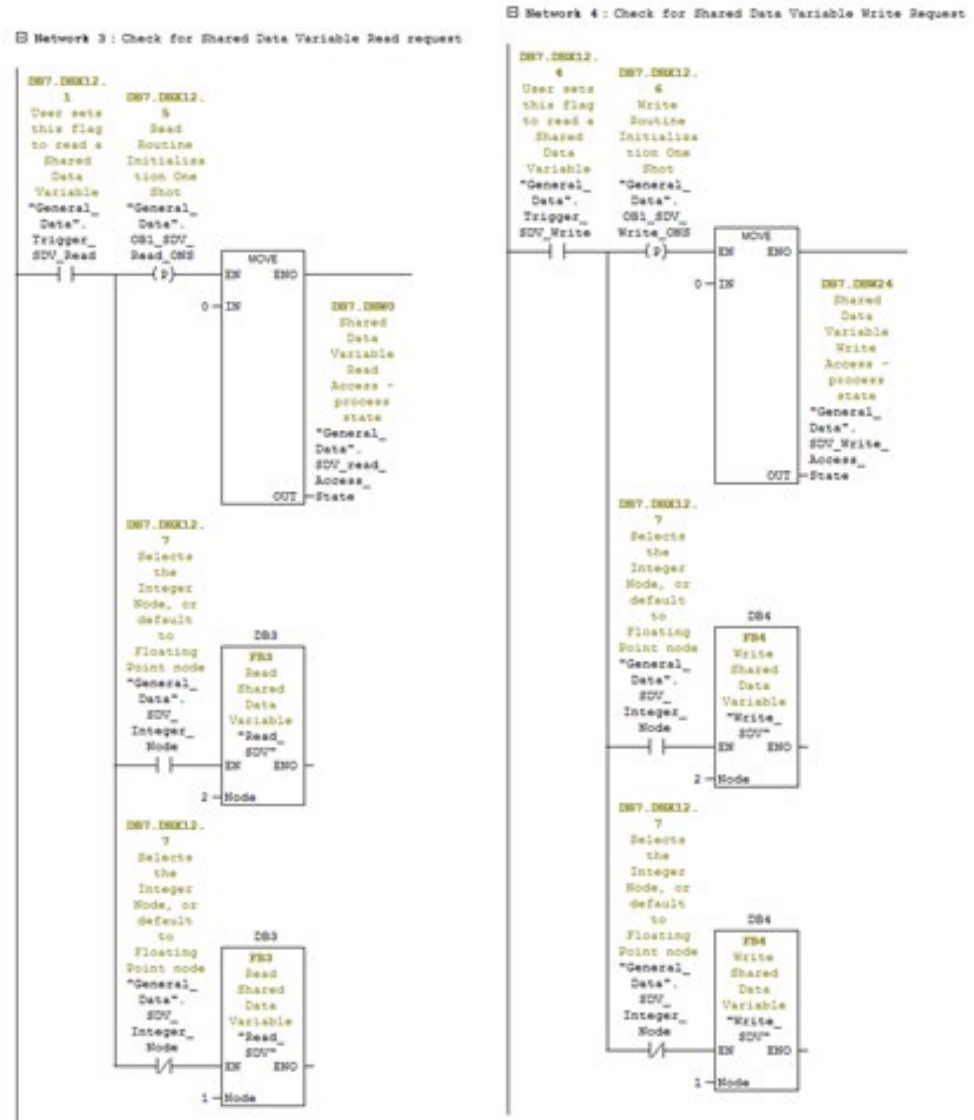


Figure 7-47: Networks 3 & 4 of OB1

Networks 3 & 4 of OB1 show how the functions that handle both the Read and Write to Shared Data in the IND570 can be called. In both cases, a bit can be set either by the program or by a user via a Variable Access Table (VAT) that triggers the function to run. Note that the only difference between the calls for the Integer device and the Floating Point device is the Node Number on the network. Otherwise, the Shared Data access works the same for both Integer and Floating Point.

7.10.4.2. FB4 (Write Shared Data) Program details

Since FB4 is the simpler routine, and contains a subset of the instructions included in FB3 (Read Shared Data), we will look at it first.

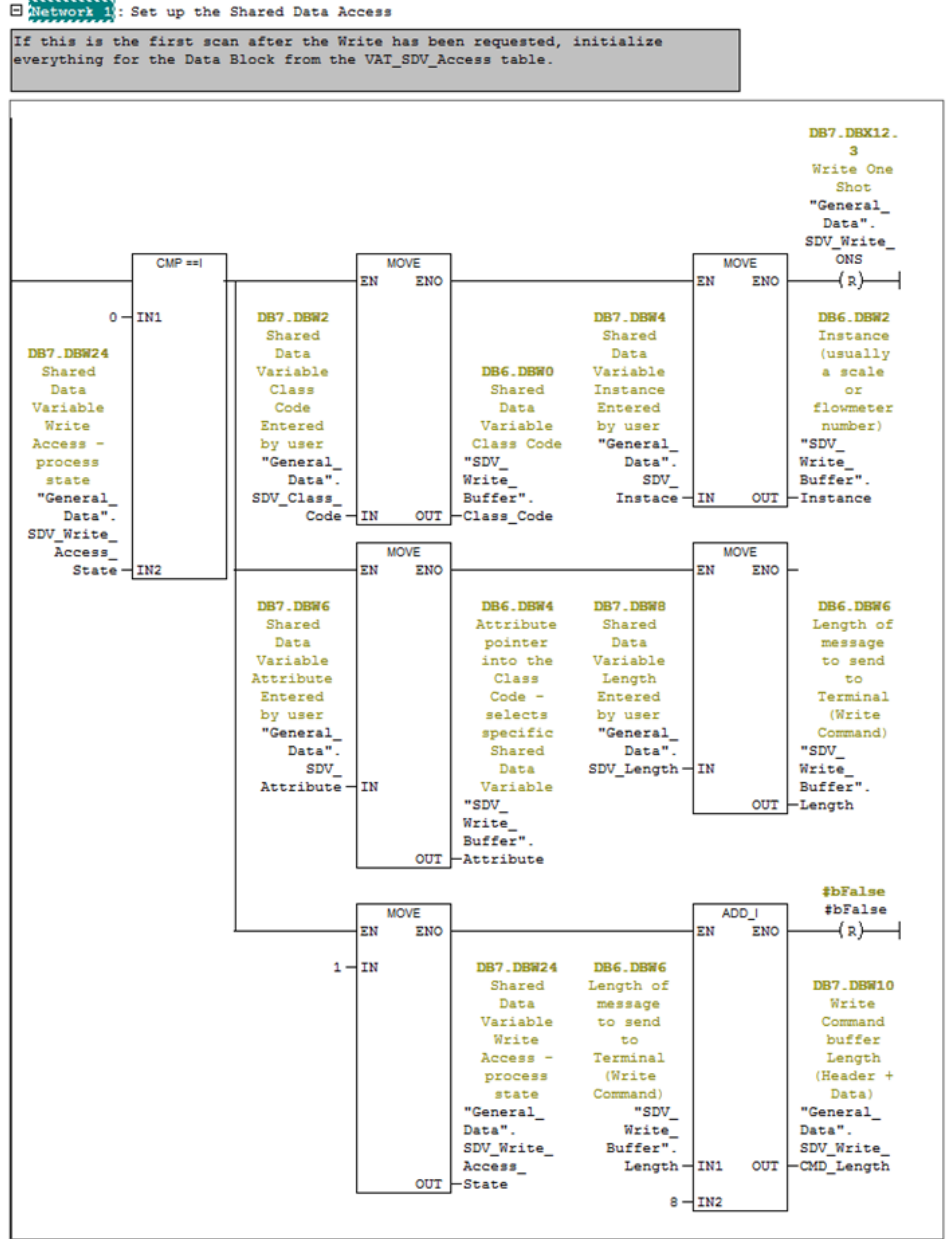


Figure 7-48: Network 1 of FB4

Network 1 Checks to see if this is the first scan since the Write flag was set. If it is, then get the Shared Data Class, Instance, Attribute, and data length from the requestor (in this case, the VAT table). **Note that in this case, the data to be written has already been populated into the Data Buffer of DB6.**

Kick the step variable to the next step in the sequence, and calculate how long the entire Write buffer will be by adding the header length to the data length provided by the caller.

Network 2 : Send Write Command to the Terminal.

Send the Write Command to the Terminal specifying the Class, Instance, Attribute, and the data to be written - all in DB6.

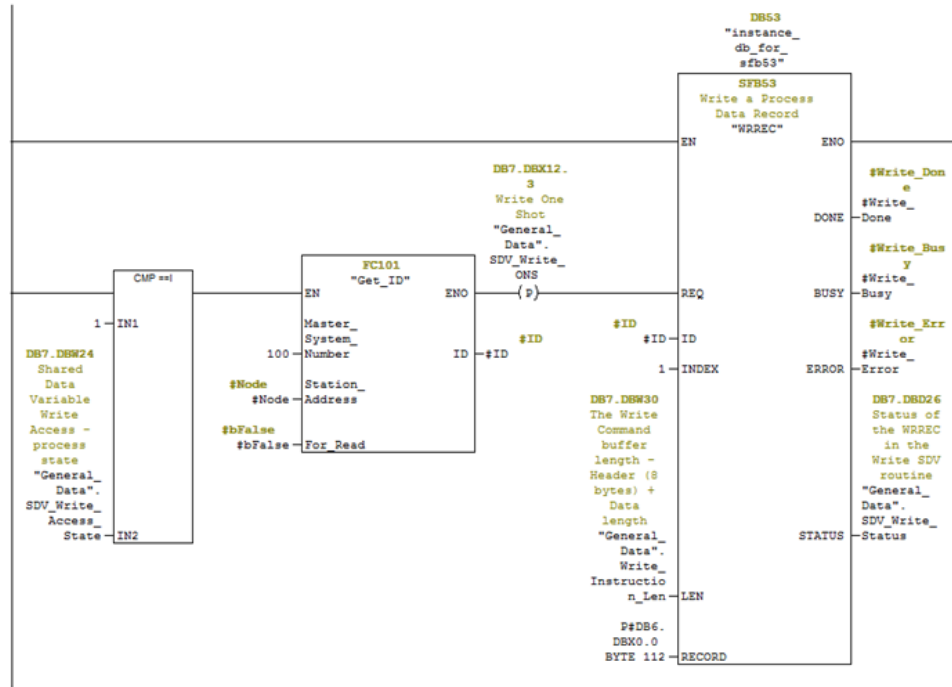


Figure 7-49: Network 2 of FB4

Network 2 calls FC101, which generates the Network ID for the specified Node. Then it calls SFB53 (WRREC) to send the data in DB6 to the IND570 specified by the ID code.

□ Network 3: Wait for the Write command to complete.

While the Busy Bit is set, wait for completion. If the Error bit is set then return a -98 in the step to indicate that the Write command failed that the user should check the WRREC returned status for more information. If the Done bit is set then move on to the next step in the Write sequence.

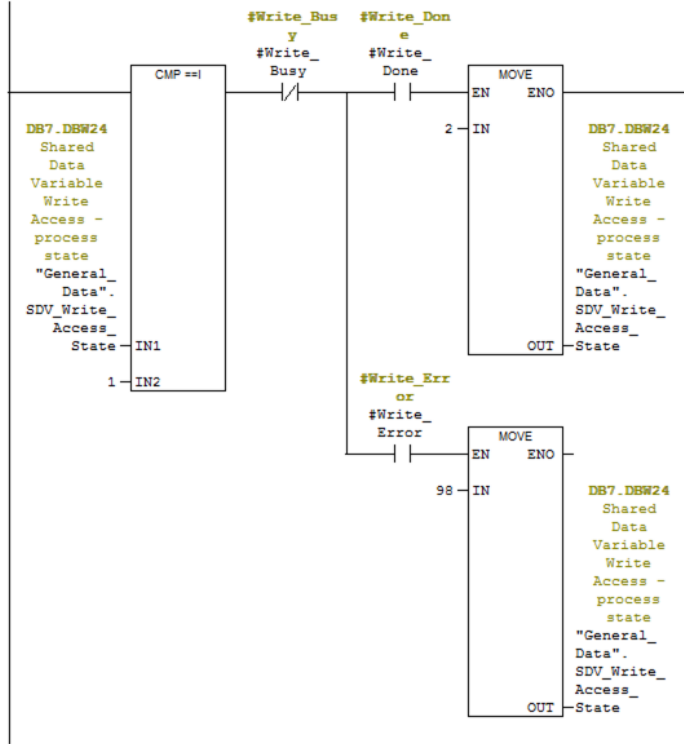


Figure 7-50: Network 3 of FB4

Network 3 confirms that the program is executing step 1 of the Shared Data Write routine, and then waits for the Data_Write Busy signal to go low. At that point, if the Data_Write Done flag is set then move on to the next step in the sequence. If instead the Data_Write Error flag is set, move the value of 98 into the sequence step to flag that an error has occurred.

□ Network 4 : Clear the Write Request Flag.

We're all done. Cancel the request and leave the step number as a positive value to indicate success.

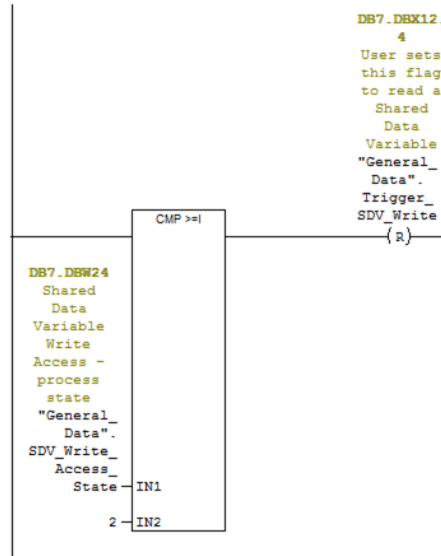


Figure 7-51: Network 4 of FB4

Network 4 terminates the sequence by clearing the request bit, at which point the calling routine can check the step number and status flags for the completion status.

7.10.4.3. FB3 (Read Shared Data) Program details

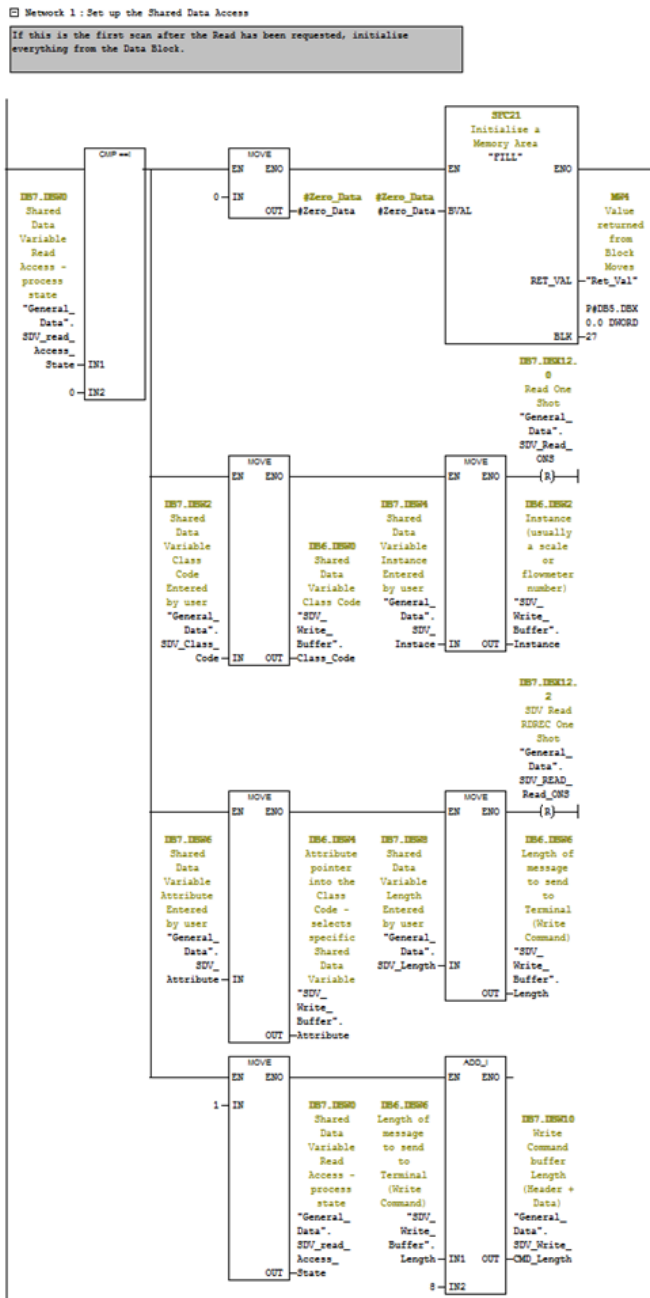


Figure 7-52: Network 1 of FB3

On the first scan of FB3 after the read flag has been set, network 1 initializes DB5 to all zeroes. Then the Class code, Instance, Attribute and data length are copied into DB5. The command length is calculated by adding the header length (8) to the data length. The step sequence is then incremented to the next step.

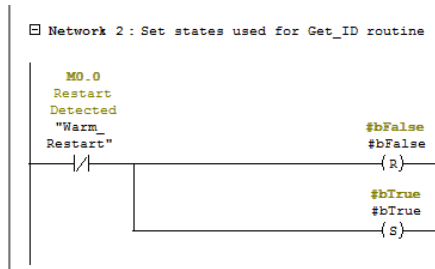


Figure 7-53 : Network 2 of FB3

Network 2 Sets the True and False bits for later use in the routine.

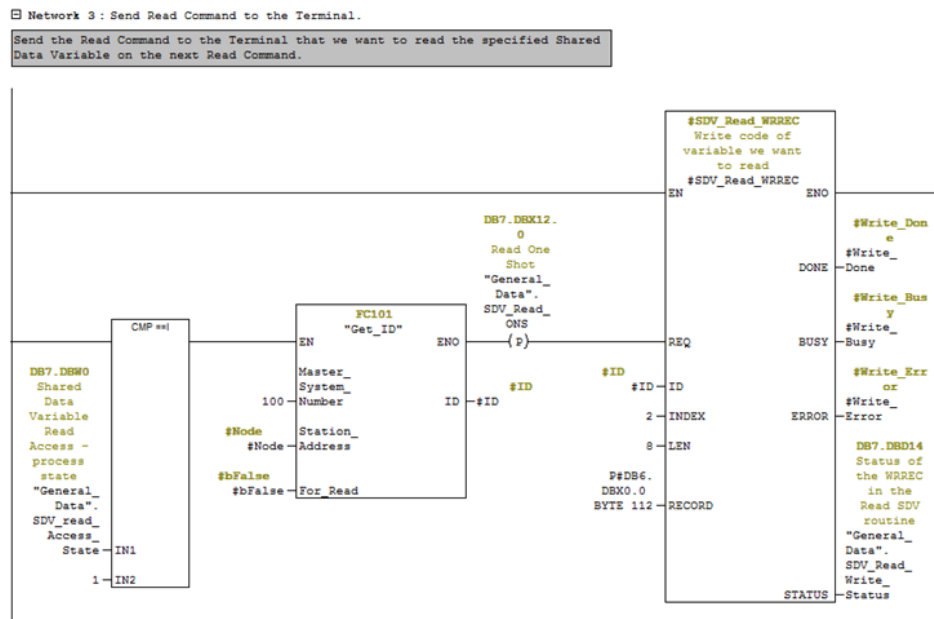


Figure 7-54: Network 3 of FB3

When the sequence step is 1, network 3 writes the contents of DB6 to the IND570 specified in the ID code (generated by FC101, which is not discussed here). Note that the Index is set to 2. This tells the IND570 that the WRREC routine is requesting that the data contained in the Shared Data variable specified by the Class, Instance, and Attribute, be returned to the PLC. The IND570 will buffer up the requested data and wait for the corresponding REREC routine call to be issued.

□ Network 4 : Wait for the command for the Read to complete

After the Write to the Terminal (which contains the request to read the specified Shared Data value) has executed, wait for either the Done or Error flags to be set. If the Error flag gets set, cancel the request and return a -98 to indicate that the Read Command failed and that the user should check the returned Write Status for more information.

If the done flag is set, then move on to the next step in the Shared Data Variable Read sequence.

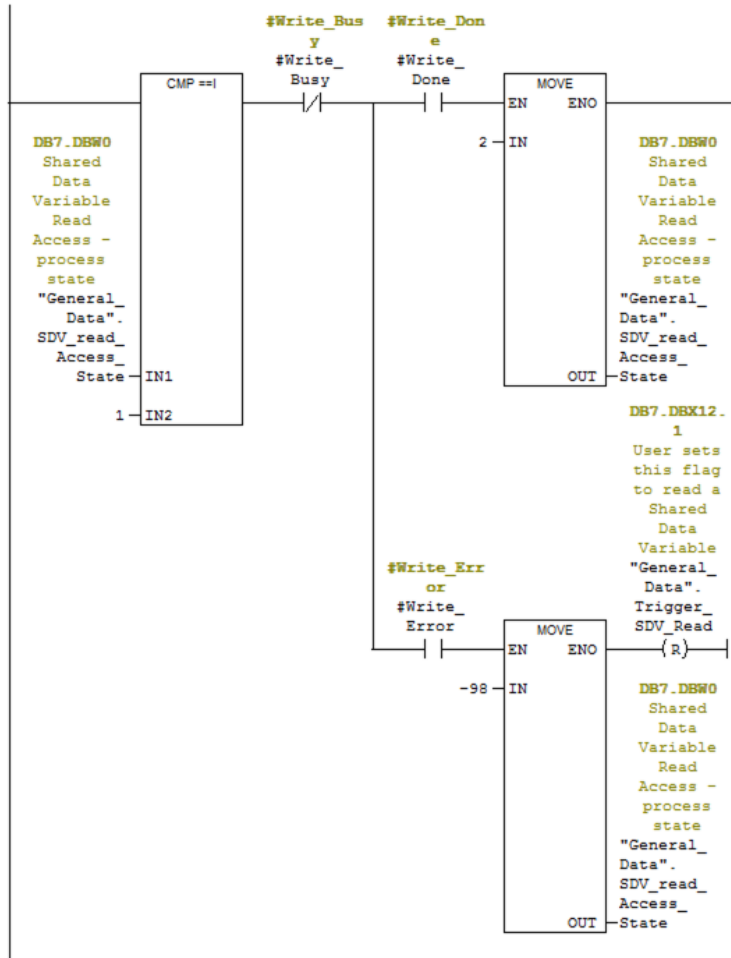


Figure 7-55: Network 4 of FB3

If the sequence step is still 1, then wait for the Data_Write busy flag to turn off. After that, if the Data_Write Done flag is set then increment the step counter to move on to the next step. If the Data_Write Error flag is set, then put a -98 into the step counter and clear the Read Request flag. The -98 will tell the caller that the read failed on the initial contact with the IND570.

Network 5: Read the Result

If the command write above was successful, send a command to read the requested data from the IND Instrument. Return the read data into DB5.

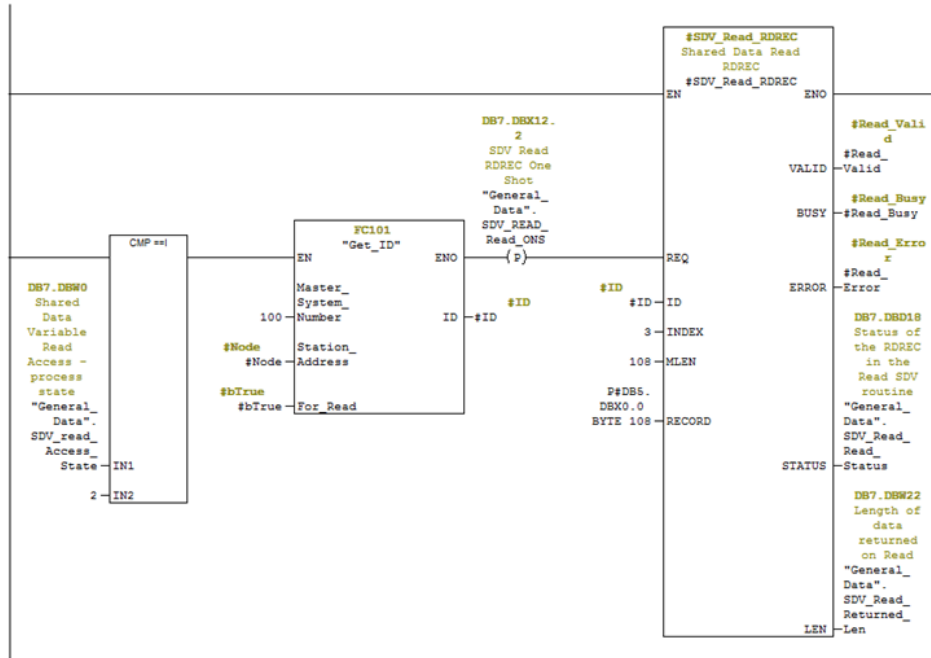


Figure 7-56: Network 5 of FB3

If the sequence step is 2, then trigger the Read of the requested data from the IND570. Note that the ID is generated by a call to the same FC101 routine that was used for the WRREC.

□ Network 6 : Wait for the Read to Complete

Wait for the Busy Flag to go away. Once it does, check the Valid Read flag and the Error flag. If the Error flag is set, shut down the routine and return a -99 to indicate that the Read failed (the user should check the returned status data from the above read command for more information).

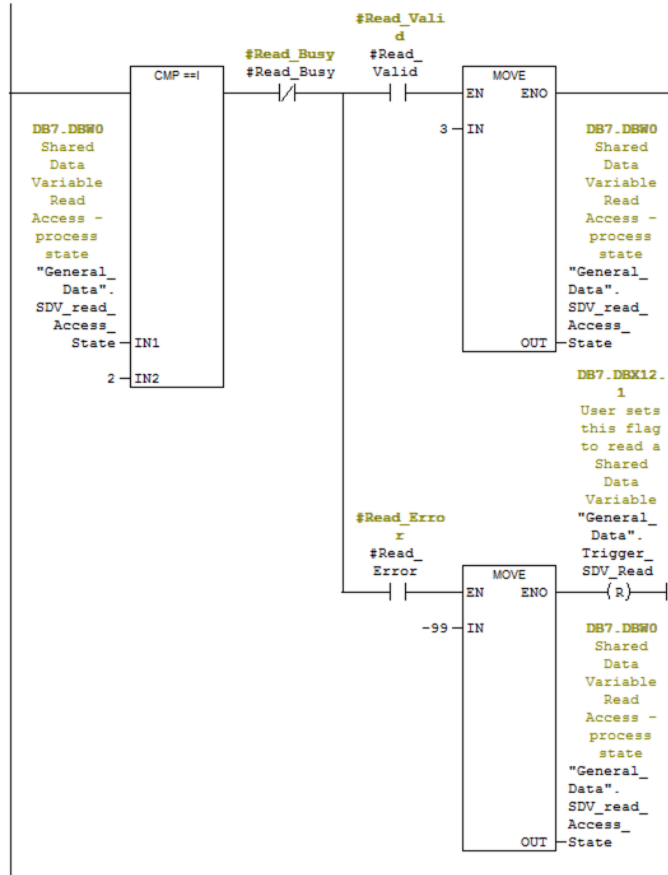


Figure 7-57: Network 6 of FB3

If the sequence step is 2, then wait for the Data_Read busy flag to turn off. Once it does, if the Data_Read Valid flag is set then increment the routine's step counter. If the Data_Read Error flag is set, the move a -99 into the step counter to flag that an error occurred in the second half of the Read routine, and clear the Read Request flag.

Network 7 : Confirm that we got what we wanted.
 First, assume that the Shared Data Read returned the WRONG Shared Data variable (because the Siemens compares seem to insist that we do things this way). Then, check to see if the Class, Instance, and Attribute in the returned Read data match what was sent in the Write command. If they all match, then make our step number a positive final value. Otherwise, return a -100 to flag that the READ failed.

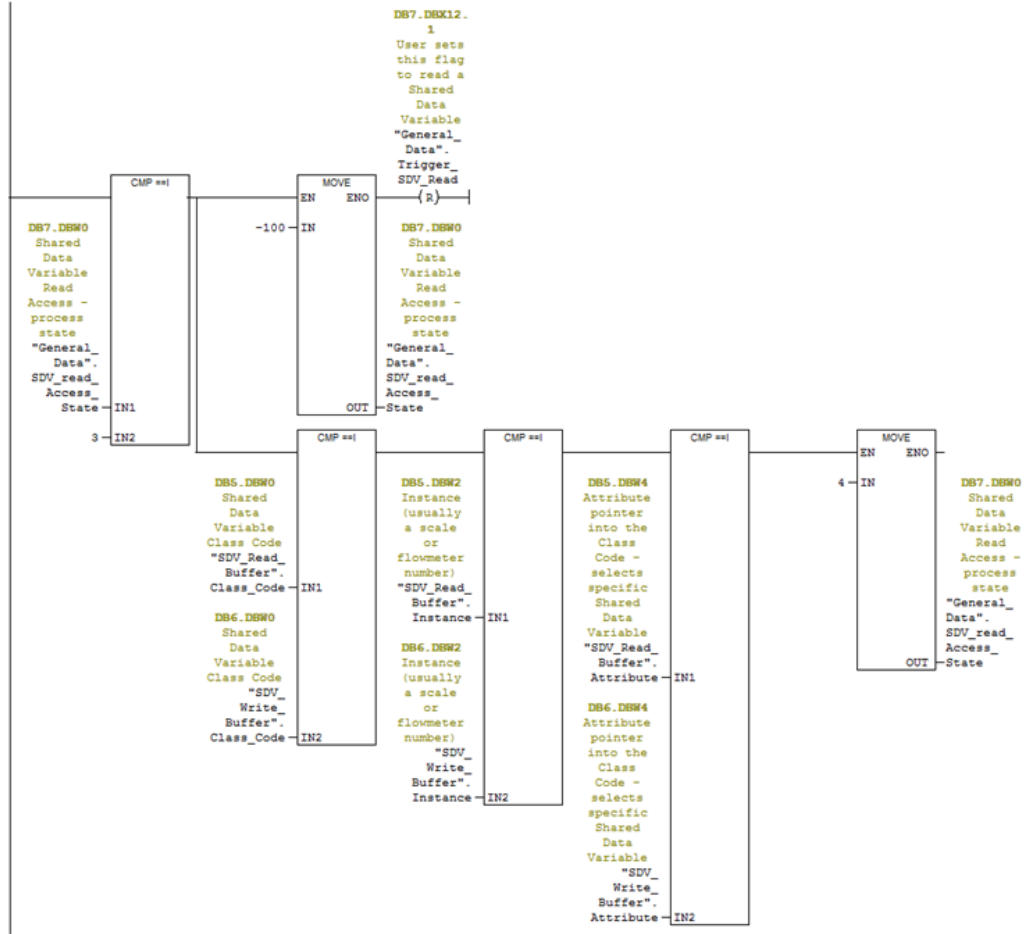


Figure 7-58: Network 7 of FB3

If the sequence step is 3, then process the final result by first assuming that the sequence failed and making sure that the Read Request bit is cleared to indicate that we're done. Next, compare the returned Class Code, Instance, and Attribute numbers with the ones that were originally sent. If they all match then increment the Step number and allow the routine to exit normally. If they do NOT match then the routine will automatically exit with the error codes set.

7.10.4.4. Running the Sample Program

The sample program can be run from the Variable Access Table as shown in the Figure 7-59 and Figure 7-60.

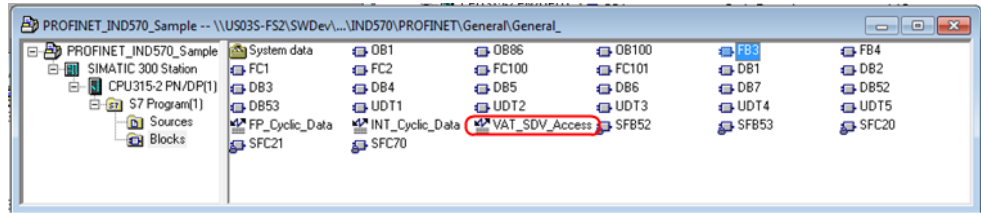


Figure 7-59: VAT_SDV_Access

VAT_SDV_Access is a Variable Access Table that allows the user to monitor the processed cyclic data and issue commands to the IND570 as shown below:

Address	Symbol	Display format	Status value	Modify value
1	// Note that this table will work for both Floating Point and Integer Mode, but be sure to set DB7.DBX 12.7 accordingly			
2	// Shared Data Access, specify Class, Instance, Attribute and length.			
3	DB7.DBX 12.7	"General_Data".SDV_Integer_Node	BOOL	false
4	DB7.DBW 2	"General_Data".SDV_Class_Code	HEX	W#16#006D
5	DB7.DBW 4	"General_Data".SDV_Instance	HEX	W#16#0001
6	DB7.DBW 6	"General_Data".SDV_Attribute	HEX	W#16#0001
7	DB7.DBW 8	"General_Data".SDV_Length	HEX	W#16#0004
9	// Shared Data Read Trigger and Status			
10	DB7.DBX 12.1	"General_Data".Trigger_SDV_Read	BOOL	false
11	DB7.DBW 0	"General_Data".SDV_read_Access_State	DEC	0
12	DB7.DBW 22	"General_Data".SDV_Read_Returned_Len	DEC	0
13	DB7.DBD 14	"General_Data".SDV_Read_Write_Status	HEX	DW#16#00000000
14	DB7.DBD 18	"General_Data".SDV_Read_Read_Status	HEX	DW#16#00000000
16	// Read Buffer Data Request Echo Back			
17	DB5.DBW 0	"SDV_Read_Buffer".Class_Code	HEX	W#16#0000
18	DB5.DBW 2	"SDV_Read_Buffer".Instance	HEX	W#16#0000
19	DB5.DBW 4	"SDV_Read_Buffer".Attribute	HEX	W#16#0000
20	DB5.DBW 6	"SDV_Read_Buffer".Length	HEX	W#16#0000
22	// First 20 bytes of the Read Buffer (from IND780) appears here. More is available if needed			
23	DB5.DBD 8	"SDV_Read_Buffer".Data_Buffer[0]	CHARACTER	DW#16#00000000
24	DB5.DBD 12	"SDV_Read_Buffer".Data_Buffer[1]	CHARACTER	DW#16#00000000
25	DB5.DBD 16	"SDV_Read_Buffer".Data_Buffer[2]	CHARACTER	DW#16#00000000
26	DB5.DBD 20	"SDV_Read_Buffer".Data_Buffer[3]	CHARACTER	DW#16#00000000
27	DB5.DBD 24	"SDV_Read_Buffer".Data_Buffer[4]	CHARACTER	DW#16#00000000
29	// Shared Data Write Trigger and Status			
30	DB7.DBX 12.4	"General_Data".Trigger_SDV_Write	BOOL	false
31	DB7.DBW 24	"General_Data".SDV_Write_Access_State	DEC	0
32	DB7.DBD 26	"General_Data".SDV_Write_Status	HEX	DW#16#00000000
33	DB7.DBW 30	"General_Data".Write_Instruction_Len	DEC	12
35	// Shared Data Write Output Buffer			
36	DB6.DBW 0	"SDV_Write_Buffer".Class_Code	HEX	W#16#0000
37	DB6.DBW 2	"SDV_Write_Buffer".Instance	HEX	W#16#0000
38	DB6.DBW 4	"SDV_Write_Buffer".Attribute	HEX	W#16#0000
39	DB6.DBW 6	"SDV_Write_Buffer".Length	HEX	W#16#0000
41	// first 20 bytes of the Write Buffer (to IND780) appears here. More is available if needed			
42	DB6.DBD 8	"SDV_Write_Buffer".Data_Buffer[0]	FLOATING_POINT	2.76
43	DB6.DBD 12	"SDV_Write_Buffer".Data_Buffer[1]	CHARACTER	DW#16#00000000
44	DB6.DBD 16	"SDV_Write_Buffer".Data_Buffer[2]	CHARACTER	DW#16#00000000
45	DB6.DBD 20	"SDV_Write_Buffer".Data_Buffer[3]	CHARACTER	DW#16#00000000
46	DB6.DBD 24	"SDV_Write_Buffer".Data_Buffer[4]	CHARACTER	DW#16#00000000

Figure 7-60: VAT_SDV_Access – Description

7.10.5. MRP Loop Example

Assigning the IP Addresses and Device Name information for an MRP Loop works as described in section 7.8.

MRP Redundant Loop communications requires no special programming. The only differences appear in the Hardware configuration, which will be covered here.

7.10.5.1. To configure an MRP loop with no other switches, using the PLC as the loop manager

The following steps must be performed in the Hardware configuration:

1. Confirm that the CPU firmware is capable of MRP communications – this information is found in the CPU object properties.

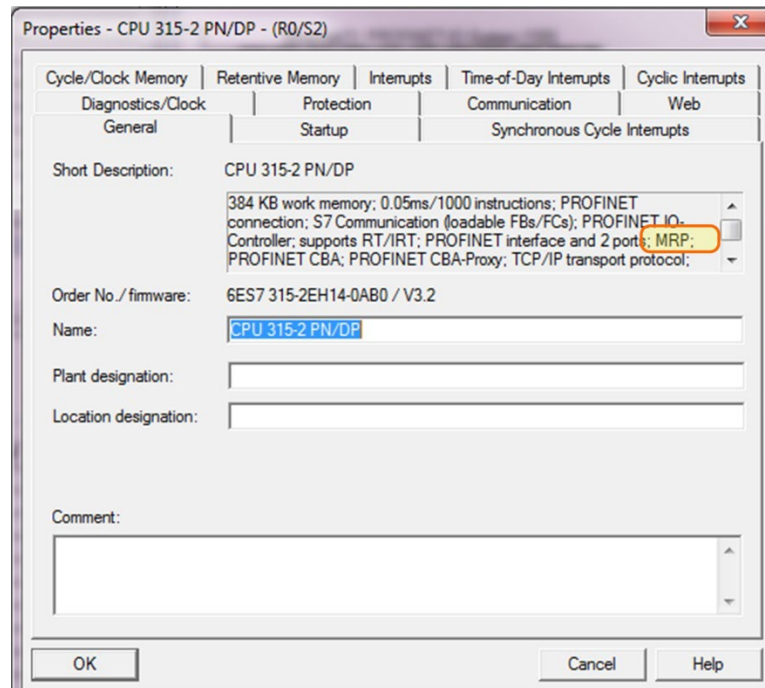


Figure 7-61: CPU Object Properties

2. Configure the PROFINET I/O Domain Management as follows (refer to section 7.10.5.1.1, below):
 - a. PLC role as the loop Manager (Auto)
 - b. All other device roles as Client
3. Configure all of the connections in the PROFINET I/O Topology (refer to section 7.10.5.1.2).
4. Finally, set up the Device Watchdog timers to be **greater** than the 200 millisecond network recovery time (refer to section 7.10.5.1.3).

The following sections cover each item in detail.

7.10.5.1.1. Configure the PROFINET I/O Domain Management

1. When all nodes have been added to the PROFINET link, open the **PROFINET IO Domain Management** form

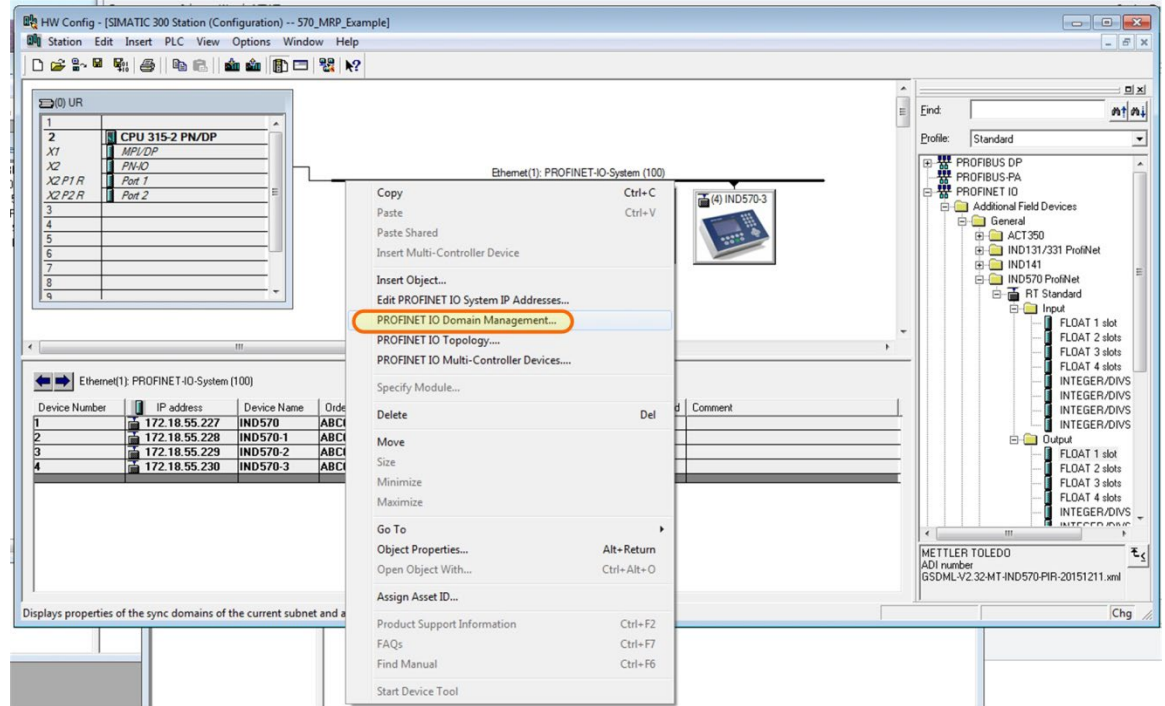


Figure 7-62: Domain Management

2. Edit the Properties of each device as shown in Figure 7-63

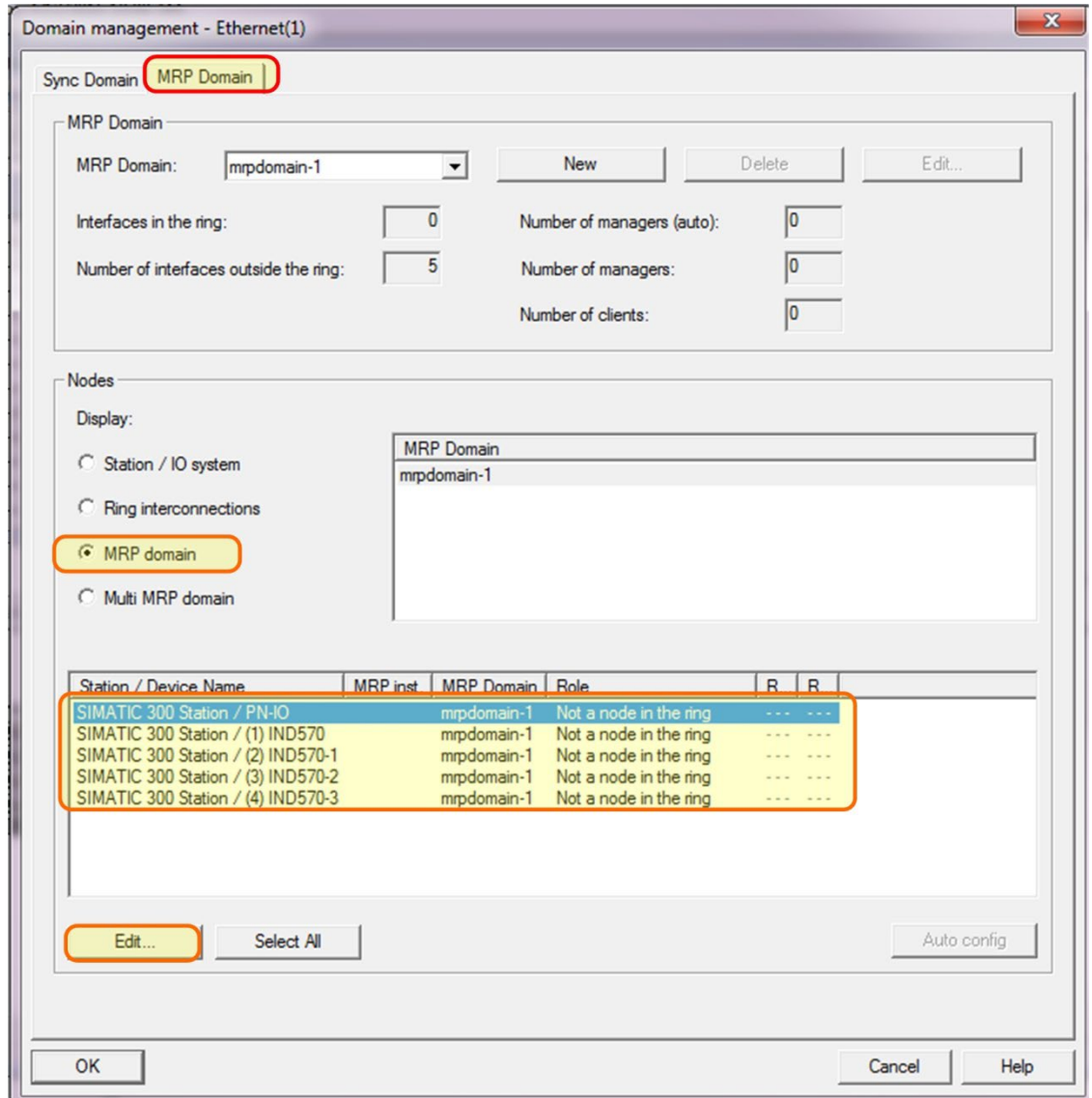


Figure 7-63: Editing Device Properties

- PLC Role = Manager (Auto)
- All other node Roles = Client

3. The result should be as shown in Figure 7-64.

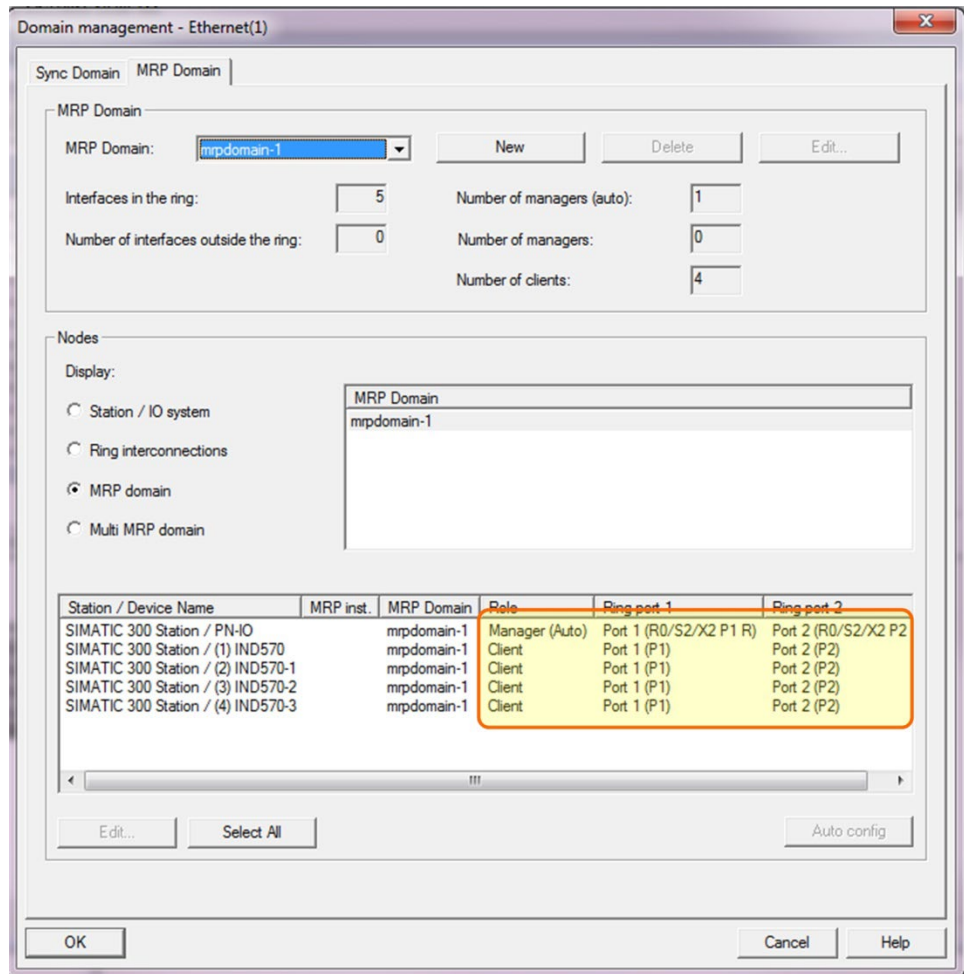


Figure 7-64: Device Properties Correctly Configured

- **Note:** Because the PLC is also the Loop Manager, be sure to leave the Diagnostic Interrupts OFF (unchecked) on all devices.

7.10.5.1.2. Configure all of the connections in the PROFINET I/O Topology

1. Select the PROFINET IO Topology from the PROFINET Properties

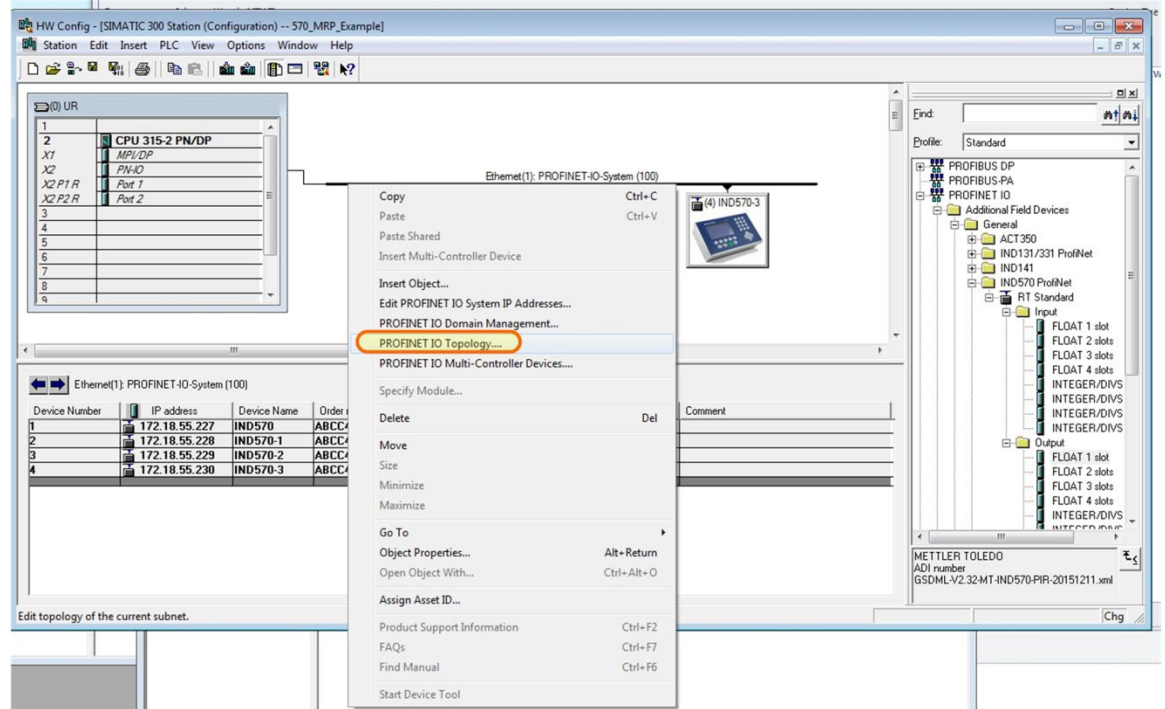


Figure 7-65: I/O Topology Selected from PROFINET Properties

2. In the Editor window that opens, click on the **Graphic View** tab and set up the links between the PLC and the network

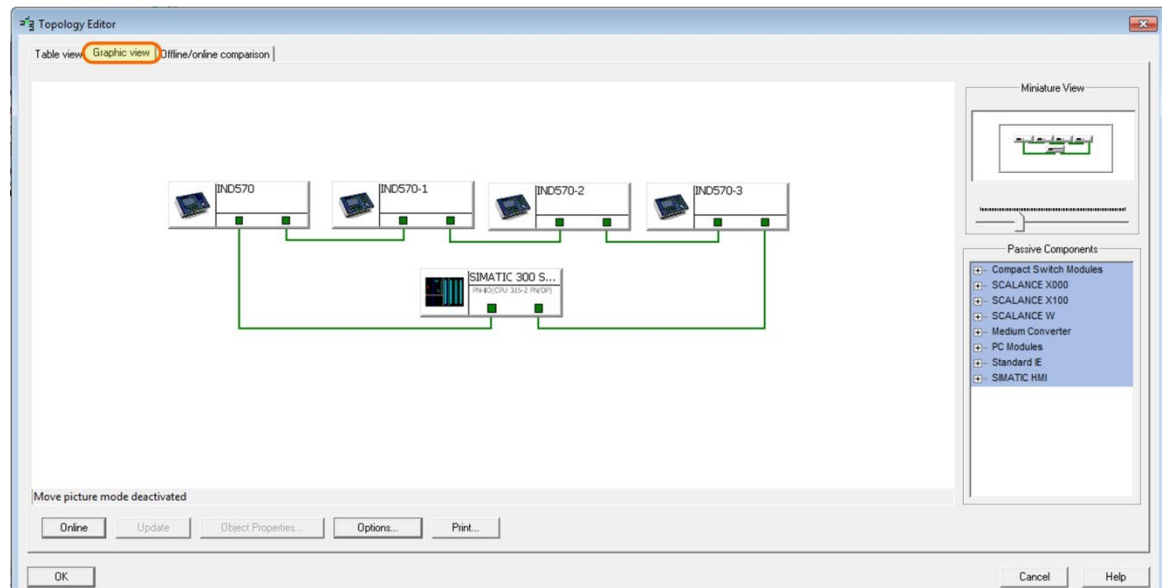


Figure 7-66: Configuring Network Links

7.10.5.1.3. Set up the Device Watchdog Timers

The "Watchdog time" is the amount of time that a device can go without seeing an I/O update before it faults. It is important to make sure that this time is configured to be **greater** than the MRP Loop recovery time, which is 200 milliseconds, to prevent the devices from faulting while the PLC is attempting to reconfigure the network after a break has occurred.

The watchdog timer is set by selecting the number of Update Intervals desired. The software then multiplies that number by the Update time. For example, if the update time is 2 ms, then select 128 intervals (256 ms) to set the Watch Dog timer above the 200 millisecond recovery time.

1. First, click on the device (1).
2. Then click on **Interface** (2).

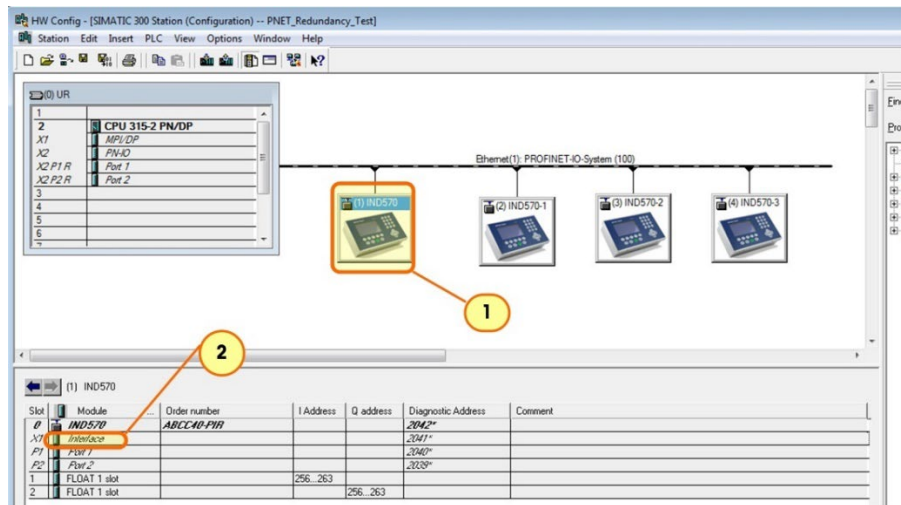


Figure 7-67: Device Watchdog Timer Setup, 1

3. In the dialog that opens, click on IO Cycle (3).

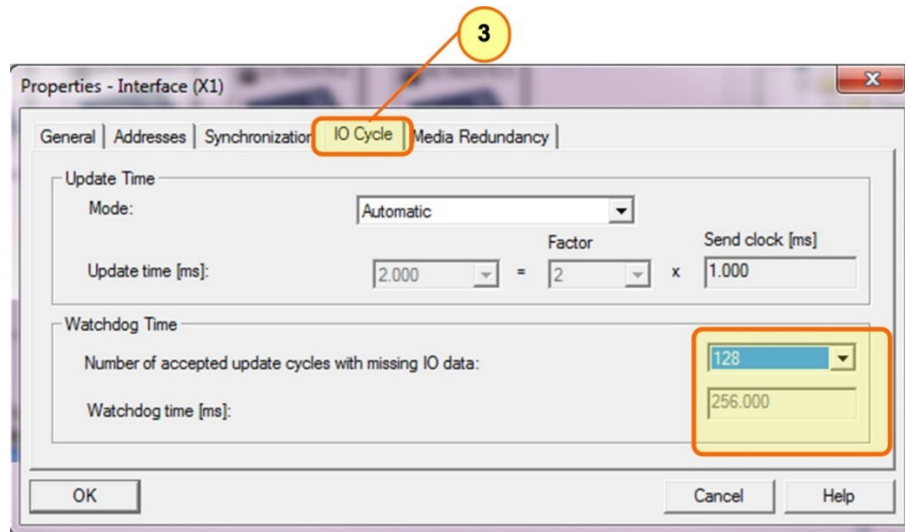


Figure 7-68: Watchdog Timer Setup, 2

4. Click **OK** to exit the dialog.

8 Modbus RTU

8.1. Overview

The Modbus protocol is a messaging structure developed by Modicon and supported today by the Modbus Organization. It is used to establish server/client communication between intelligent devices. It is an open standard network protocol, widely used in the industrial manufacturing environment.

The IND570 can be set up to communicate via Modbus RTU from COM1, COM2, or COM3. COM1 is provided through the standard COM1 serial port on the main board. COM2 and COM3 are provided by the COM2/COM3 option board, which has better anti-interference performance.

The IND570 Modbus RTU only supports the function codes for reading (03H) and writing (06H) a single register.

IND570 data contains numerical and status information and commands. Numerical data sent to and from the IND570 terminal can be in either Integer or Float format.

- **Integer Mode** Reports scale weight as a signed 16 bit integer (± 32767).
- **Float Mode** Displays weight in floating point data format.

8.2. Parameters

8.2.1. Port

Choose one of the following ports for Modbus RTU communication:

COM 1 [default], COM2, COM3

8.2.2. Assignment

Select Modbus RTU Server in the Assignment selection list.

8.2.3. Address

To prevent address conflicts in a network on the first setup, the terminal address is set to 0 by default. Please set the terminal address to a valid value from 1 to 15, and ensure that the address is exclusive for each client in that network.

8.2.4. Byte Order

Byte Order selections include:

Big Endian [default]	An order in which the "big end" (most significant value in the sequence) is stored first, at the lowest storage address
Little Endian	An order in which the "little end" (least significant value in the sequence) is stored first.
Byte Swap	Consists of masking each byte and shifting them to the correct location
Word Swap	Consists of masking each word and shifting them to the correct location

If the data received is erroneous, try changing the Byte Order selection.

8.3. Data Definition

8.3.1. Holding Registers Assignments

Table 8-1 shows the holding register assignment. Note that the register addresses are PLC-dependent. This table shows the register address as 5 digits. The addressing is based on the type of PLC used. In all cases, the IND570 registers are mapped to the twelve holding registers at most.

Table 8-1: Modbus RTU Holding Register Assignments

- It is important to note whether the byte order of the terminal (big-endian mode, little-endian mode, byte swapping, or word swapping) matches that of the communication master. To use the Modbus RTU communication function, a 'Modbus RTU Server' connection must be added under the Terminal menu -> Communication -> Connection. The byte order can be selected in Connection Edit -> Byte Order.



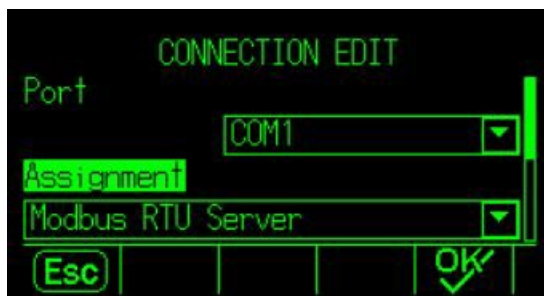


Figure 8-1: Connection Edit pages

Register Address	R/W	Type	Description	Note
40001	R	float	Default: Rounded gross weight	This value is the rounded gross weight.
40003	R	float	Rounded gross weight	
40005	R	float	Rounded tare weight	This value is the rounded tare weight.
40007	R	float	Rounded net weight	This value is the rounded net weight.
40009	R	float	Gross weight	This value is the non-rounded gross weight.
40011	R	float	Tare weight	This value is the non-rounded tare weight.
40013	R	float	Net weight	This value is the non-rounded net weight.
40015	R	float	Weight unit	Used to obtain the weight unit. When terminal is set to little-endian mode, the meaning of the value obtained is as follows: 1: lb 2: kg 3: g 4: t 5: ton
40017	R	float	Raw load cell counts	This value is the raw load cell counts.
40020	W	float	Preset tare	Used to set the floating-point preset tare weight. Due to the fact terminal doesn't support function code 16 (write multiple holding registers), it is necessary to use function code 06 to write to the corresponding holding register in two separate writes.
40022	W	short	Tare	Used to trigger taring action. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.
40023	R	short	Tare operation status	Used to obtain status of the tare operation. When terminal is set to little-endian mode, the meaning of the value read is as follows: 0 = Tare completed successfully 1 = Tare in progress

Register Address	R/W	Type	Description	Note
				2 = Scale in motion during tare 3 = Pushbutton tare not enabled 4 = Programmable tare not enabled 8 = Tare value too small 9 = Tare when power-up zero not captured 10 = Tare over capacity 11 = Tare under zero 12 = Tare value exceeds limit 14 = Scale in expanded mode
40024	W	short	Zero	Used to zero the scale after motion check. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.
40025	R	short	Zero operation status	Used to obtain status of the zero operation. When terminal is set to little-endian mode, the meaning of the value read is as follows: 0 = Zero completed successfully 1 = Zero in progress 2 = Scale in motion during zero 3 = Illegal scale mode during zero 4 = Scale out of zeroing range 98 = Invalid function parameter
40026	W	short	Clear tare	Used to trigger immediate clear tare action. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.
40027	W	short	Tare immediate	Used to trigger immediate tare action. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.
40029	W	short	Print	Used to trigger print action. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.
40204	R	float	Increment	This value is the increment.
40206	R	float	Scale capacity	This value is the scale capacity.
40803	.0 R	bit	Discrete input 1	

Register Address	R/W	Type	Description	Note	
	.1	R	bit	Discrete input 2	Used to read the status of discrete inputs or read/set the status of discrete outputs, which requires the configuration of discrete input/output interface board for the read/write to take effect. When terminal is set to little-endian mode, the corresponding relationships are as follows: bit0: discrete input 1 bit1: discrete input 2 bit2: discrete input 3 bit3: discrete input 4 bit4: discrete input 5 bit5-bit15: reserved bit0: discrete output 1 bit1: discrete output 2 bit2: discrete output 3 bit3: discrete output 4 bit4: discrete output 5 bit5: discrete output 6 bit6: discrete output 7 bit7: discrete output 8 bit8-bit15: reserved Reading a bit information returns 1 or setting a bit information to 1 indicates that the corresponding input/output point is valid.
	.2	R	bit	Discrete input 3	
	.3	R	bit	Discrete input 4	
	.4	R	bit	Discrete input 5	
40805	.0	R/W	bit	Discrete output 1	
	.1	R/W	bit	Discrete output 2	
	.2	R/W	bit	Discrete output 3	
	.3	R/W	bit	Discrete output 4	
	.4	R/W	bit	Discrete output 5	
	.5	R/W	bit	Discrete output 6	
	.6	R/W	bit	Discrete output 7	
	.7	R/W	bit	Discrete output 8	
40901	R/W	float	Rate value	Used to set the floating-point rate value. As terminal does not support function code 16 (write multiple holding registers), it is necessary to use function code 06 to write the corresponding holding registers in two separate writes.	
40998	W	short	Restart terminal	Used to trigger the terminal to restart. When terminal is set to little-endian mode, writing a value of 1 triggers the action, that is, setting bit 0 of the low byte to 1 triggers the action. In other words, in little-endian mode, write 1; in big-endian mode, write 256.	
41001	R	float	Rounded tare weight	This value is the rounded tare weight.	
41003	R	float	Display weight	This value is the display weight.	
41005	.0	R	bit	Scale Status	Used to read status of the scale. When terminal is set to little-endian mode, the corresponding relationships are as follows: bit0: Weighing data OK, 0=invalid, 1= valid bit1: Motion, 0=stability, 1=motion bit2: Net mode, 0=gross mode, 1=net mode
	.1	R	bit		
	.2	R	bit		
	.3	R	bit		
	.4	R	bit		
	.5	R	bit		

Register Address	R/W	Type	Description	Note
.6	R	bit		bit3: Center of Zero, 0=invalid, 1=valid bit4: X10, 0=invalid, 1=valid bit5: Print status, 0 = No operation, 1 = In process bit6: Zero status, 0 = No operation, 1 = In process bit7: Tare status, 0 = No operation, 1 = In process bit8-bit15: reserved
.7	R	bit		
.8	R	bit		
.9	R	bit		
.10	R	bit		
.11	R	bit		
.12	R	bit		
.13	R	bit		
.14	R	bit		
.15	R	bit		

A. Integer and Division Formats

When one of these formats is selected, the IND570 will have two 16-bit words for input data and two 16-bit words for output data in each Message Slot. There can be up to four slots and the number of slots is setup at the IND570. The PLC's input data will contain one 16-bit word for the scale's weight information and one 16-bit word for bit encoded status information for each Message Slot. The IND570 will send specific weight data to the PLC input based on the selections the IND570 receives from the PLC's output data. The PLC's output words consist of one 16-bit integer value, which may be used to download a tare or target logic value, and one 16-bit word for bit encoded command information.

The "Select 1, 2 or 3" commands in write word 1 select the type of data that will be returned in the scale data slot. While any type of data can be reported back from any Integer or Division slot, commands such as Tare, Clear and Zero can only be sent to slot 1. (This applies to Integer/Division mode only.)

Table A-1 and Table A-2 provide detailed information on the integer and division data formats. Note that the designation of "Read" or "Write" data is based on the PLC's viewpoint—"Read" data refers to the PLC's input data and "Write" data refers to the PLC's output data.

Table A-1: Discrete Read Integer or Division – IND570 > PLC, per Message Slot

Bit number	First Word	Second Word
0	See Note 1	Target 1 ²
1		Target 2 ²
2		Target 2 ²
3		Comparator 5 ³
4		Comparator 4 ³
5		Comparator 3 ³
6		Comparator 2 ³
7		Comparator 1 ³
8		Enter Key ⁴
9		Input 1 ⁵
10		Input 2 ⁵
11		Input 3 ⁵
12		Motion ⁶
13		Net Mode ⁷
14		Update in Process ⁸
15	Data OK ⁹	

Notes for Table A-1

- 1 The first word is a 16 bit, signed integer that may represent the indicator's gross weight, net weight, displayed weight, tare weight, or rate. The **bits 0 to 2** in the PLC 2nd output word designate the type of data that is being sent by the indicator.
- 2 The second word **bits 0, 1 and 2** indicate the state of the target comparison logic. When in the material transfer mode; **bit 0** is Feed, **bit 1** is Fast Feed and **bit 2** is Tolerance Ok (within range). When in the over/under mode; **bit 0** is Under, **bit 1** is OK and **bit 2** is Over. An 'ON' condition is indicated by the bit being set to '1'; an 'OFF' condition is indicated by the bit being set to '0'.
- 3 The second word Comparator bits indicate the state of the associated comparator logic; when the bit is set to '1' the comparator state is 'ON'; when the bit is set to '0' the comparator state is 'OFF'. The setup of each comparator will determine when the state is 'ON' or 'OFF'.
- 4 The second word **bit 8** is set to '1' when the Enter Key has been pressed on the indicator keypad. The bit can be reset to '0' by changing the state of the second output word **bits 9, 10 and 11**.
5. The second word **bits 9, 10, and 11** indicate the state of the associated hardware input internal to the indicator; these are 0.1.1, 0.1.2 and 0.1.3. When the input is 'ON' the associated bit is set to '1'.
- 6 The second word **bit 12**; The motion bit is set to '1' when the scale is in motion (unstable).
- 7 The second word **bit 13**; The net mode bit is set to '1' when scale is in the net mode (a tare has been taken).
- 8 The second word **bit 14** (update in process) is set to '1' when the indicator is in process of updating the data to the PLC communications adapter. The PLC should ignore all data while this bit is set to '1'.
- 9 The second word **bit 15**; The data ok bit is set to '1' when the indicator operating conditions are normal. The bit is set to '0' during power-up, during indicator setup, when the scale is over capacity or under zero, and when in the x10 display mode; additionally, the first word integer value is set to '0'. Note that, when in x10 mode, the data sent is 000000. The PLC should continuously monitor the data ok bit in the IND570 communication and also any PLC data connection fault bit that exists in the PLC (refer to the PLC manufacturer's documentation) to determine the validity of the data in the PLC.

Table A-2: Discrete Write Integer or Division –PLC > IND570, per Message Slot

Bit number	First Word	Second Word [Scale Command]
0	See Note 1	Select 1 ²
1		Select 2 ²
2		Select 3 ²
3		Load Tare 1 st message slot only ¹²
4		Clear Tare ⁴ 1 st message slot only ¹²
5		Tare ⁵ 1 st message slot only ¹²
6		Print ⁶ 1 st message slot only ¹²
7		Zero ⁷ 1 st message slot only ¹²

Bit number	First Word	Second Word [Scale Command]
8		Start/Abort Target ⁸ 1 st message slot only ¹²
9		Message Display Mode ⁹ 1 st message slot only
10		Message Display Mode ⁹ 1 st message slot only ¹²
11		Message Display Mode ⁹ 1 st message slot only ¹²
12		Output 1 ¹⁰ 1 st message slot only ¹²
13		Output 2 ¹⁰ 1 st message slot only ¹²
14		Output 3 ¹⁰ 1 st message slot only ¹²
15		Load Target ¹¹ 1 st message slot only ¹²

Notes for Table A-2

- 1 First word is a 16 bit, signed integer that represents a value to be downloaded to the indicator. The value represents a tare or target value. When using the divisions format, the data set must be in the number of divisions, not an integer weight value. A value must be loaded in this word before setting the **bits 3** or **15** in the second word. To load the target value ,first enter the value into the first word and then set bit 15 (Load Target) “On”
- 2 The select bits change the type of data being sent from the indicator in the first word. Use a decimal value in binary format within **bits 0, 1, and 2** to change the data reported by the indicator. ‘0’ = gross weight, ‘1’ = net weight, ‘2’ = displayed weight, ‘3’ = tare weight, ‘4’ = target, ‘5’ = rate; any value above 5 will equal gross weight.
- 3 A transition from ‘0’ to ‘1’ will cause the value in the first word to be loaded into the tare register of the indicator and set the indicator into the net mode. Set this bit to ‘1’ only **after** the first word has been loaded with the required value.
- 4 A transition from ‘0’ to ‘1’ will cause the indicator tare register to be set to ‘0’ and the indicator will be set to the gross weight mode.
- 5 A transition from ‘0’ to ‘1’ will cause the weight on the scale to be used as the tare value and set the indicator to the net mode (equivalent to a tare command). Note that the scale will not tare while the scale is “In Motion”. If the indicator has not tared within 3 seconds, the command must be resent. A good practice is to check for no motion –bit 12 of input word 1-“Off”
- 6 A transition from ‘0’ to ‘1’ will issue a print command.
- 7 A transition from ‘0’ to ‘1’ will cause the scale to re-zero, but only within the ranges established in scale setup.
- 8 A transition from ‘0’ to ‘1’ will cause the target logic to start. A transition from ‘1’ to ‘0’ will cause the target logic to abort. The use of the PLC in conjunction with the indicator console keypad and/or a remote input is not advised, as unexpected results may occur.

- 9 The message display mode bits will cause messages to be displayed on the indicator display above the soft key prompts; messages are limited to 20 characters. The use of the display mode bits will clear the Enter Key bit in the second word of the indicator output data. The message display mode bits cause a value to be written to shared data pd0119, which is available for use by Task Expert applications. The transition from '0' to a decimal value in binary form to the second word **bits 9, 10 and 11** will initiate the message events.
- Setting the message display bits to a value of '1' will cause the characters in shared data aw0101 to be displayed and pd0119 will be set to '1'.
- Setting to '2' = display aw0102 and pd0119 = '2'.
- Setting to '3' = display aw0103 and pd0119 = '3'.
- Setting to '4' display aw0104 and pd0119 = '4'.
- Setting to '5' = display aw0105 and pd0119 = '5'.
- Setting to '6' = start Prompt sequence, pd0119 = '6' and xc0134 = '1'.
- Setting to '7' = display pd0118 and pd0119 = '7'.
- The message display mode bits must return to '0' before a new message can be displayed.
- 10 The output bits will cause the associated hardware output to be turned 'ON' and 'OFF'. This is the indicator internal outputs only; 0.1.1, 0.1.2 and 0.1.3. The output bits will not override the hardware outputs being used by the indicator logic as setup within the indicator. Setting a bit to '1' will cause the output to turn 'ON'; setting the bit to '0' will cause the output to turn 'OFF'.
- 11 A transition from '0' to '1' will cause the value in the first word to be loaded into the target register of the indicator and will be used the next time the target logic is started. Set this bit to '1' only **after** the first word has been loaded with the required value.
- 12 These are bit commands to the indicator that function only in the first message slot.

B. Floating Point Format

B.1. Operational Overview

The IND570 uses integer commands from the PLC to select the floating point weight input data. The IND570 recognizes a command when it sees a new value in the Message Slot command word. If the command has an associated floating point value (for example: loading a target value), it must be loaded into the floating point value words before the command is issued. Once the IND570 recognizes a command, it acknowledges the command by setting a new value in the command acknowledge bits of the scale's command response word. The IND570 also tells the PLC what floating point value is being sent (via the floating point input indicator bits of the command response word). The PLC should wait until it receives the command acknowledgment from the IND570 before sending another command.

The IND570 can report two types of values to the PLC: real-time and static. When the PLC requests a real-time value, the IND570 acknowledges the command from the PLC once but sends and updates the value at every interface update cycle. If the PLC requests a static value, the IND570 acknowledges the command from the PLC once and updates the value once. The IND570 will continue to send this value until it receives a new command from the PLC. Gross weight and net weight are examples of real-time data. Tare weight, target, feed, and tolerance values are examples of static data.

The IND570 can send a rotation of up to nine different real-time values. The PLC sends commands to the IND570 to add a value to the rotation. Once the rotation is established, the PLC must instruct the IND570 to begin its rotation automatically, or the PLC may control the pace of rotation by instructing the IND570 to advance to the next value. If the IND570 is asked to automatically alternate its output data, it will switch to the next value in its rotation at the next interface update cycle. (The interface update cycle has an update rate of up to 20 Hz or 58 milliseconds.)

The PLC may control the rotation by sending alternate report next field commands (1 and 2). When the PLC changes to the next command, the IND570 switches to the next value in the rotation order. The IND570 stores the rotation in its shared data so the rotation does not have to be re-initialized after each power cycle. When the PLC does not set up an input rotation, the default input rotation consists of gross weight only. See the floating-point command examples in Table B-5 through Table B-8 for additional information. The method of handling string and floating point data varies between PLC types. The IND570 provides floating point data in the order entered in Data Format setup.

B.2. Floating Point Data Format and Compatibility

In Floating Point Data Format, the PLC and IND570 terminal exchange weight, target, and tare data in single-precision floating-point format. The IEEE Standard for Binary Floating-Point Arithmetic,

ANSI/IEEE Standard 754-1985, specifies the format for single-precision floating point numbers. It is a 32-bit number that has a 1-bit sign, an 8-bit signed exponent, and a 23-bit mantissa. The 8-bit signed exponent provides scaling of weight data. The 23-bit mantissa allows representation of 8 million unique counts.

Although the single-precision floating point number provides greater numerical precision and flexibility than integer weight representations, it has limitations. The weight representation may not be exact, particularly for the extended-resolution weight fields for high-precision bases.

There are two data integrity bits that the IND570 uses to maintain data integrity when communicating with the PLC. One bit is in the beginning word of the data; the second is in the ending byte of the data for a scale slot. The PLC program must verify that both data integrity bits have the same polarity for the data in the scale slot to be valid. There is a possibility that the PLC program will see several consecutive invalid reads when the terminal is freely sending weigh updates to the PLC, if the PLC program detects this condition, it should send a new command to the terminal.

The method of handling string and floating point data varies between Allen-Bradley PLC generations.

B.2.1. Notes: Floating Point Numbers in Various PLCs

The Simatic TI505 PLCs support the IEEE Standard floating point numbers. According to the **Simatic TI505 Programming Reference Manual**, real numbers are stored in the single-precision 32-bit format, according to ANSI/IEEE Standard 754-1985, in the range 5.42101070 E-20 to 9.22337177 E18.

Siemens S5 PLCs do not inherently support the IEEE-format floating point numbers. S5 PLCs do support floating point numbers in their own unique format. S software “function block” can be implemented in the S5 PLC, to convert between S5 floating point numbers and IEEE Standard floating point numbers.

B.3. Floating Point Data Format Definitions

The following tables provide detailed information on the floating-point data format. Read data refers to the PLC’s input data and write data refers to the PLC’s output data.

Table B-1: Discrete Read Floating Point – IND570 > PLC Input, per Message Slot

Bit number	1 st Word Command Response	2 nd Word FP value	3 rd Word FP value	4 th Word Scale Status
0	RESERVED	See Note 4	See Note 4	Target 1 ⁵
1				Comparator 1 ⁶
2				Target 2 ⁵
3				Comparator 2 ⁶
4				Target 3 ⁵
5				Always = 1

Bit number	1 st Word Command Response	2 nd Word FP value	3 rd Word FP value	4 th Word Scale Status
6				TE bit 1 ⁷
7				TE bit 2 ⁷
8	FP Input Indicator 1 ¹			Enter Key ⁸
9	FP Input Indicator 2 ¹			Input 1 ⁹
10	FP Input Indicator 3 ¹			Input 2 ⁹
11	FP Input Indicator 4 ¹			Input 3 ⁹
12	FP Input Indicator 5 ¹			Motion ¹⁰
13	Data integrity1 ²			Net Mode ¹¹
14	Command Ack 1 ³			Data Integrity 2 ²
15	Command Ack 2 ³			Data OK ¹²

Notes for Table B-1

- 1 The Floating Point Indicator bits (1st word bits 8-12) are used to determine what type of floating or other data is being sent in the second and third words. See the Floating Point Indicator Table for the information from these bits in decimal format.
- 2 The Data Integrity bits (1st word **bit 13** and 4th word **bit 14**) should be used to assure that communication is still valid and that data are valid. Both of these bits are set to '1' for one update from the indicator, then are set to '0' for the next update from the indicator and this change of state is on every update and is constant as long as the communications link is not disrupted.
- 3 The first word Command Acknowledge bits (**bits 14** and **15**) are used by the indicator to inform the PLC that a new command was received. The decimal values of these bits will rotate sequentially from 1 to 3 as long as a command other than '0' is being sent (3rd output word). The decimal value of these bits will be '0' when the 3rd output word (PLC output command word) is decimal '0'.
- 4 The second and third words are 32 bit, single precision floating point data. The data may represent the various scale weight data or setup configuration data. The PLC output command word determines what data will be sent.
- 5 The fourth word, **bits 0, 2** and **4** indicate the state of the Target comparison logic. When in the material transfer mode; **bit 0** is Feed, **bit 2** is Fast Feed and **bit 4** is Tolerance Ok (within range). When in the over/under mode; **bit 0** is Under, **bit 2** is OK and **bit 4** is Over. An 'ON' condition is indicated by the bit being set to '1'; an 'OFF' condition is indicated by the bit being set to '0'.
- 6 The fourth word; Comparator bits indicate the state of the associated comparator logic; when the bit is set to '1' the comparator state is 'ON'; when it is set to '0' the comparator state is 'OFF'. The setup on each comparator will determine when the state is 'ON' or 'OFF'.
- 7 The fourth word; TE **bit 1** is the state of shared data variable ac0101. TE **bit 2** is the state of shared data variable ac0102. A Task Expert (TE) application may use these bits to instruct the PLC to perform a procedure or function.
- 8 The fourth word **bit 8** is set to '1' when the Enter Key has been pressed on the keypad of the indicator. The bit can be reset to '0' by sending the command 75 (decimal) in the PLC output command word.

- 9 The fourth word **bits 9, 10, and 11** indicate the state of the associated hardware input internal to the indicator; these are 0.1.1, 0.1.2 and 0.1.3. When the input is 'ON' the associated bit is set to '1'.
- 10 The fourth word **bit 12**; The motion bit is set to '1' when the scale is in motion.
- 11 The fourth word **bit 13**; The net mode bit is set to '1' when scale is in the net mode (a tare has been taken).
- 12 The fourth word **bit 15**; The data ok bit is set to '1' when the indicator operating conditions are normal, and when in the x10 display mode. Note that when in x10 mode, the data sent is in the higher resolution. The bit is set to '0' during power-up, during indicator setup, and when the scale is over capacity or under zero. The PLC should continuously monitor the data ok bit in the IND570 communication and also any PLC data connection fault bit that exists in the PLC (refer to the PLC manufacturer documentation) to determine the validity of the data in the PLC.

Table B-2: Floating Point Input Indication

Dec	Data	Dec	Data	Dec	Data
0	Gross Weight ¹	11	Low-pass filter frequency	22	Weigh-in +tolerance value ³
1	Net Weight ¹	12	Notch filter frequency	23	Weigh-in -tolerance value ³
2	Tare Weight ¹	13	Target value ³	24	Weigh-out target value ³
3	Fine Gross Weight ¹	14	+ Tolerance value ³	25	Weigh-out fine feed value ³
4	Fine Net Weight ¹	15	Fine feed value ³	26	Weigh-out spill value ³
5	Fine Tare Weight ¹	16	- Tolerance value ³	27	Weigh-out +tolerance value ³
6	Rate ¹	17	Spill value ³	28	Weigh-out -tolerance value ³
7	Custom field #1	18	Primary units, low increment size	29	Last indicator error code
8	Custom field #2 ²	19	Weigh-in target value ³	30	Command received successfully, no response
9	Custom field #3	20	Weigh-in fine feed value ³	31	Invalid Command
10	Custom field #4 ²	21	Weigh-in spill value ³		

Notes for Table B-2

- 1 Data is refreshed on every indicator update
- 2 Data is ASCII characters and is limited to the first 4 characters
- 3 Value that is in the Target registers, may not be the active Target value

Table B-3: Discrete Write Floating Point – PLC >> IND570, per Message Slot

Bit Number	1 st Word [Scale command]	2 nd Word	3 rd Word
0	See Note 1	See Notes 2 and 3	
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

Notes for Table B-3

- 1 The first word is a 16 bit integer and is used to send commands to the indicator.
The commands are used to:
 - instruct the indicator to report a specific type of data in words 2 and 3. Examples are Gross Weight, Net Weight, + Tolerance Value, etc.
 - instruct the indicator to load the floating point data in the second and third words for a tare value, target value; or other value
 - instruct the indicator to turn on internal outputs or perform a functions, such as Clear Tare, Print, Tare, Start Weigh, etc
- 2 The second and third words represent a 32 bit single precision floating point value that will be used for downloading a tare, target or other value to the indicator.
- 3 Not all commands require a floating point value in the second and third words.

Table B-4: PLC Output Command Table (Floating Point Only)

Dec	Hex	Command	SDName
0	0	Report next rotation field @ next A/D update ¹	
1	1	Report next rotation field ^{1,3}	
2	2	Report next rotation field ^{1,3}	
3	3	Reset (cancel) rotation	
10	A	Report gross weight ²	
11	B	Report net weight ²	
12	C	Report tare weight ²	
13	D	Report fine gross weight ²	
14	E	Report fine net weight ²	
15	F	Report fine tare weight ²	
16	10	Report Rate ²	
17	11	Report custom float value #1 ^{2,5}	aj0101
18	12	Report custom string value #2 ^{2,4,5}	ak0101
19	13	Report low-pass filter frequency ^{2,5}	
20	14	Report notch filter frequency ²	
21	15	Report target value ^{2,5}	
22	16	Report (+) tolerance value ^{2,5}	
23	17	Report fine feed ^{2,5}	
24	18	Report (-) tolerance value ^{2,5}	
25	19	Report spill value ⁵	
27	1B	Report custom float value #3 ⁵	aj0102
28	1C	Report custom string value #4 ⁵	ak0102
30	1E	Report primary units ⁵	
40	28	Add gross weight to rotation ⁷	
41	29	Add net weight to rotation ⁷	

Dec	Hex	Command	SDName
42	2A	Add tare weight to rotation ⁷	
43	2B	Add fine gross weight to rotation ⁷	
44	2C	Add fine net weight to rotation ⁷	
45	2D	Add fine tare weight to rotation ⁷	
46	2E	Add rate to rotation ⁷	
47	2F	Add custom value #1 to rotation ⁷	aj0101
48	30	Add custom value #2 to rotation ⁷	ak0101
60	3C	Load programmable tare value ⁶	
61	3D	Pushbutton tare command ⁷	
62	3E	Clear command ⁷	
63	3F	Print command ⁷	
64	40	Zero command ⁷	
68	44	Trigger 1 command ⁷	
69	45	Trigger 2 command ⁷	
70	46	Trigger 3 command ⁷	
71	47	Trigger 4 command ⁷	
72	48	Trigger 5 command ⁷	
73	49	Set low-pass filter frequency ⁶	
74	4A	Set notch filter frequency ⁶	
75	4B	Reset (clear) ENTER key ⁷	
80	50	Clear display message ^{7,8}	
81	51	Display Message 1 ^{7,8}	
82	52	Display Message 2 ^{7,8}	
83	53	Display Message 3 ^{7,8}	
84	54	Display Message 4 ^{7,8}	
85	55	Display Message 5 ^{7,8}	

Dec	Hex	Command	SDName
86	56	Display Message 6 ^{7,8}	
87	57	Display Message 7 ^{7,8}	
88	58	Disable weight display ⁷	
89	59	Enable weight display ⁷	
90	5A	Set discrete output 0.1.1 "ON" ⁷	di0105
91	5B	Set discrete output 0.1.2 "ON" ⁷	di0106
92	5C	Set discrete output 0.1.3 "ON" ⁷	di0107
93	5D	Set discrete output 0.1.4 "ON" ⁷	di0108
100	64	Set discrete output 0.1.1 "OFF" ⁷	di0105
101	65	Set discrete output 0.1.2 "OFF" ⁷	di0106
102	66	Set discrete output 0.1.3 "OFF" ⁷	di0107
103	67	Set discrete output 0.1.4 "OFF" ⁷	di0108
110	6E	Set target value ^{6,13}	
111	6F	Set target fine feed value ^{6,13}	
112	70	Set - tolerance value ^{6,13}	
114	72	Start target comparison ^{7, 12,13}	
115	73	Abort target comparison ^{7, 12,13}	
116	74	Target use gross weight ⁷	
117	75	Target use net weight ⁷	
119	77	Weigh-In Start ^{7,10}	
120	78	Weigh-Out Start ^{7,10}	
121	79	Enable target latching ⁷	
122	7A	Disable target latching ⁷	
123	7B	Reset target latch ⁷	
124	7C	Set Spill Value ^{6,13}	
131	83	Set (+) tolerance value ^{6,13}	
160	A0	Apply scale setup (reinitialize) ^{7, 9}	

Dec	Hex	Command	SDName
162	A2	Disable indicator tare (IDNet only) ⁷	
163	A3	Enable indicator tare (IDNet only) ⁷	
170	AA	Set weigh-in target value ^{6,10,11}	af0161
171	AB	Set weigh-in fine feed value ^{6,10,11}	af0163
172	AC	Set weigh-in spill value ^{6,10, 11}	af0162
173	AD	Set weigh-in +tolerance value ^{6,10,11}	af0164
174	AE	Set weigh-in -tolerance value ^{6,10,11}	af0165
175	AF	Set weigh-out target value ^{6,10,11}	af0151
176	B0	Set weigh-out fine feed value ^{6,10,11}	af0153
177	B1	Set weigh-out spill value ^{6,10,11}	af0152
178	B2	Set weigh-out +tolerance value ^{6,10,11}	af0154
179	B3	Set weigh-out -tolerance value ^{6,10,11}	af0155
180	B4	Report weigh-in target value ^{6,10}	
181	B5	Report weigh-in fine feed value ^{6,10}	
182	B6	Report weigh-in spill value ^{6,10}	
183	B7	Report weigh-in +tolerance value ^{6,10}	
184	B8	Report weigh-in -tolerance value ^{6,10}	
185	B9	Report weigh-out target value ^{6,10}	
186	BA	Report weigh-out fine feed value ^{6,10}	
187	BB	Report weigh-out spill value ^{6,10}	
188	BC	Report weigh-out +tolerance value ^{6,10}	
189	BD	Report weigh-out -tolerance value ^{6,10}	
190	BE	Not used	
191	BF	Not used	
192	C0	Trigger OK key ⁷	ac0109
193	C1	Trigger ENTER key ⁷	xc0130
194	C2	Trigger weigh-in pause ^{7,10}	

Dec	Hex	Command	SDName
195	C3	Trigger weigh-in resume	ac0101
196	C4	Trigger weigh-in abort ^{7,10}	
197	C5	Trigger weigh-out pause ^{7,10}	
198	C6	Trigger weigh-out resume ^{7,10}	ac0102
199	C7	Trigger weigh-out abort ^{7,10}	
210	D2	Set Comparator 1 limit	
211	D3	Set Comparator 1 high limit	
212	D4	Set Comparator 2 limit	
213	D5	Set Comparator 2 high limit	
214	D6	Set Comparator 3 limit	
215	D7	Set Comparator 3 high limit	
216	D8	Set Comparator 4 limit	
217	D9	Set Comparator 4 high limit	
218	DA	Set Comparator 5 limit	
219	DB	Set Comparator 5 high limit	
220	DC	Disable Keypad	
221	DD	Enable Keypad	
222	DE	Report Comparator 1 limit	
223	DF	Report Comparator 1 high limit	
224	E0	Report Comparator 2 limit	
225	E1	Report Comparator 2 high limit	
226	E2	Report Comparator 3 limit	
227	E3	Report Comparator 3 high limit	
228	E4	Report Comparator 4 limit	
229	E5	Report Comparator 4 high limit	
230	E6	Report Comparator 5 limit	
231	E7	Report Comparator 5 high limit	

Dec	Hex	Command	SDName
232	E8	Apply Comparator Values	
233		Set weigh-in +tolerance value for tolerance type "% of target" ¹⁰	af0166
234		Set weigh-in -tolerance value for tolerance type "% of target" ¹⁰	af0167
235		Set weigh-out +tolerance value for tolerance type "% of target" ¹⁰	af0156
236		Set weigh-out -tolerance value for tolerance type "% of target" ¹⁰	af0157
237		Report weigh-in +tolerance value for tolerance type "% of target" ¹⁰	
238		Report weigh-in -tolerance value for tolerance type "% of target" ¹⁰	
239		Report weigh-out +tolerance value for tolerance type "% of target" ¹⁰	
240		Report weigh-out -tolerance value for tolerance type "% of target" ¹⁰	

Notes for Table B-4

- 1 Rotation is set up by commands 40 to 48 (dec). On each indicator update the next field of the rotation setup is reported in the second and third words of the floating point output from the indicator. The floating point indication date reports what the field data represents. To keep up with the rotation changes, the PLC program scan time should be 30 milliseconds or less. A command of '0' without rotation setup will report the scale gross weight. The commands acknowledge bits are set to the value of '0'.
- 2 A command that requests data that is refreshed on every indicator update.
- 3 Toggling between commands 1 and 2 will allow the PLC to control the rotation field change.

- 4 Only 4 characters of a string field are reported; the PLC must process the data as a string value.
- 5 A command that request a specific value; as long as the request is in the command word to the indicator no other data will be reported by the indicator.
- 6 A command that requires a floating point value be in the second and third word when the command is sent to the indicator. If the command is successful the returned floating point value will equal the value sent to the indicator.
- 7 A command that will not report back a value; the floating point data from the indicator will be zero.
- 8 The message display commands will cause messages to be displayed on the indicator display above the soft key prompts; this is limited to 20 characters. The message display commands cause a value to be written to shared data PDO119; PDO119 values can be use by Task Expert applications. The command 81 to 87 (dec) will initiate the message events. Command 81 will cause the characters in shared data AW0101 to be displayed and PDO119 will be set to '1'. Command 82 = display AW0102 and PDO119 = '2'. Command 83 = display AW0103 and PDO119 = '3'. Command 84 display AW0104 and PDO119 = '4'.
Command 85 = display AW0105 and PDO119 = '5'.
Command 86 = start Prompt sequence, PDO119 = '6' and XC0134 = '1'. Command = display PDO118 and PDO119 = '7'. Command 80 (dec) will remove the message display.
- 9 If shared data classes pl, ds, ll, nt, ce, zr, ct, cm, xs, cs, dp, wk, ao, rp, or dc are changed by the PLC this command (160 dec) will trigger the changes into effect. Shared data is not available with the AB-RIO, DeviceNet and Modbus TCP.
- 10 A command that can only be used with the IND570 Fill. When Fill-570 is installed, the following commands for standard target control cannot be used: 110-115, 124, 131
- 11 If Fill-570 is not installed in the terminal, this command can be used to access the corresponding Shared Data field incorporated into a custom TaskExpert program.
- 12 In the basic terminal (without Fill-570 installed), target control can be paused and resumed using the Abort and Start commands. Note that any changes made to the target values since the original START command was given will be loaded before target control is resumed.
- 13 This command does not function when Fill-570 is installed. Commands with a "10" footnote should be used for Fill-570 target control.

B.4. Floating Point Command Examples

Table B-5: Data Requirement: Only Net Weight Sent (continuously) for Scale 1

Step #	Scale Command (From PLC)	Scale Floating Point Value	Command Response From Terminal	Floating Point Value
1 (PLC sends command to IND570 terminal to report net weight)	11 (dec) loaded into command word 0	none required		
2 (IND570 terminal sees new command)			Command ack. = 1 F.P. ind. = 1 (net)	Net weight in floating point

As long as the PLC leaves the 11 (dec) in the command word, the IND570 terminal will update the net value every interface update cycle.

Table B-6: Data Requirement: Load Target Value = 21.75 for Scale 1

Step #	Scale command (from PLC)	Scale Floating Point Value	Command response from terminal	Floating Point Value
1 (PLC loads floating point value first)		floating point value = 21.75		
2 (PLC sends command to set target 1 cutoff value)	110 (dec) loaded into command word 0	floating point value = 21.75		
3 (IND570 terminal sees new command, loads the value into the target and ends a return message to indicate the new target value)			Command ack. = 1 F.P. ind = 30	Floating point value = 21.75
4 (PLC instructs IND570 terminal to start "using" new target value)	114 (dec) loaded into command word 0			
5 (IND570 terminal sees new command)			Command ack. = 2 F.P. ind = 30	0.0

The PLC should always wait to receive a command acknowledgment before sending the next command to the IND570 terminal. After the PLC finishes loading its target value, it can resume monitoring the weight information required by sending a command to report some type of weight or set up a rotation of reported data.

Table B-7: Data Requirement: Rotation of Gross Weight and Rate Updated on Interface Update Cycle

Step #	Scale Command (from PLC)	Scale Floating Point Value	Command Response from Terminal	Floating Point Value
1 (PLC clears out any previous rotation with reset)	3 (dec) loaded into command word 0			
2 (IND570 terminal sees new command)			Command ack. = 1 F.P. ind = 30	0.0
3 (PLC adds gross weight to rotation)	40 (dec) loaded into command word 0	(null value)		
4 (IND570 terminal sees new command)			Command ack. = 2 F.P. ind = 30	0.0
5 (PLC adds rate to the rotation)	46 (dec) loaded into command word 0			
6 (IND570 terminal sees new command)			Command ack. = 3 F.P. ind = 30	0.0

At this point, the rotation has been set up. Now the PLC needs to command the IND570 terminal to begin the rotation.

7 (PLC sends the command to begin the rotation at interface update cycle)	0 (dec) loaded into command word 0			
8 (IND570 terminal sends gross weight at interface update cycle ~ 60 msec)			Command ack. = 0 F.P. ind = 0	Floating point value = gross wt.
9 (PLC leaves 0 in its command word and the IND570 terminal sends the rate value at the next interface update cycle)	0 (dec) loaded into command word 0		Command ack. = 0 F.P. ind = 6	Floating point value = rate
10 (PLC leaves 0 in its command word and IND570 terminal sends the gross value at next interface update cycle)	0 (dec) loaded into command word 0		Command ack. = 0 F.P. ind = 0	Floating point value = gross wt.
11 (PLC leaves 0 in command word and IND570 terminal sends the rate value at the next interface update cycle)	0 (dec) loaded into command word 0	RESERVED for Future Use	Command ack. = 0 F.P. ind = 6	Floating point value = rate

This rotation continues until the PLC sends a different command. At approximately every 60 msec the IND570 terminal updates its data with the next field in its rotation. The PLC must check the floating point indication bits to determine which data is in the floating point value.

Table B-8: Data Requirement: Rotation of Net Weight and Rate Updated on PLC Command

Step #	Scale command (from PLC)	Scale Floating Point Value	Command response from terminal	Floating Point Value
1 (PLC clears out any previous rotation with reset)	3 (dec) loaded into command word 0			
2 (IND570 terminal sees new command)			Command ack. = 1 F.P. ind = 30	0.0
3 (PLC adds net weight to rotation)	41 (dec) loaded into command word 0	(null value)		
4 (IND570 terminal sees new command)			Command ack. = 2 F.P. ind = 30	0.0
5 (PLC adds rate to the rotation)	46 (dec) loaded into command word 0	RESERVED for Future Use		
6 (IND570 terminal sees new command)			Command ack. = 3 F.P. ind = 30	0.0

At this point, the rotation has been set up. Now the PLC needs to send commands to the IND570 terminal to begin the rotation and advance to the next value when required.

7 (PLC sends the command to report the first field in the rotation.)	1 (dec) loaded into command word 0			
8 (IND570 terminal acknowledges the command and sends net weight at every interface update cycle until the PLC gives the command to report the next rotation field.)			Command ack. = 1 F.P. ind = 1	Floating point value = net weight
9 (PLC sends the command to report the next field.) Note: if the PLC leaves the 1 (dec) in the command, the IND570 terminal does NOT see this as another command to report the next rotation field.	2 (dec) loaded into command word 0			
10 (IND570 terminal acknowledges the command and sends rate at every interface update cycle until the PLC gives the command to report the next rotation field.)		RESERVED for Future Use	Command ack. = 2 F.P. ind = 6	Floating point value = rate

Step #	Scale command (from PLC)	Scale Floating Point Value	Command response from terminal	Floating Point Value
11 (PLC sends the command to report the next field in the rotation.)	1 (dec) loaded into command word 0			
12 (IND570 terminal acknowledges the command and sends net weight at every interface update cycle until the PLC gives the command to report the next rotation field.)			Command ack. = 1 F.P. ind = 1	Floating point value = net wt.
13 (PLC sends the command to report the next field.)	2 (dec) loaded into command word 0			
14 (IND570 terminal acknowledges the command and sends rate at every interface update cycle until the PLC gives the command to report the next rotation field.)		RESERVED for Future Use	Command ack. = 2 F.P. ind = 6	Floating point value = rate

At approximately every 60 msec the IND570 terminal updates its data with new data, but it does not advance to the next field in the rotation until the PLC sends it the command to report the next field. The PLC should check the floating point indication bits to determine which data is in the floating point value

C. Common Data Features

C.1. Data Formats

C.1.1. Discrete Data

Three data formats are available: Integer (the default), Divisions and Floating Point.

- Integer** Reports scale weight as a signed 16 bit integer (± 32767).
- Divisions** Reports scale weight in display divisions (± 32767). The PLC multiplies the reported divisions by the increment size to calculate the weight in display units.
- Floating Point** Displays weight in floating point data format

The data format of discrete data will affect the data size required in the configuration of the PLC. The IND570 console PLC message slot setup screen provides data size requirements in bytes.

Selection of the appropriate format depends on issues such as the range or capacity of the scale used in the application. The integer format can represent a numerical value up to 32,767. The division format can represent a value up to 32,767 scale divisions or increments. The floating-point format can represent a value encoded in IEEE 754, single precision floating point format.

Floating point is the only data format that includes decimal point information. Integer and division formats ignore decimal points. Accommodation of decimal point location must occur in the PLC logic, when it is needed with these formats.

C.1.1.1.

Examples

250 x .01 scale					50,000 x 10 scale				
IND570 Displays:	0	2.00	51.67	250.00	IND570 Displays:	0	200	5160	50000
Format sent:					Format sent:				
Integer	0	200	5167	25000	Integer	0	200	5160	-(xxxxx)
Division	0	200	5167	25000	Division	0	20	516	5000
Floating Point	0	2.00	51.67	250.00	Floating Point	0	200	5160	50000

Any of the formats could be used in this case.

The integer format could not be used because it would send a negative or invalid value once the weight exceeded 32,760.

150 x .001 scale				
IND570 Displays:	0	2.100	51.607	150.000
Format sent:				

150 x .001 scale				
Integer	0	2100	-(xxxxx)	-(xxxxx)
Division	0	2100	-(xxxxx)	-(xxxxx)
Floating Point	0	2.100	51.607	150.000

The integer and division formats could not be used because they would send a negative value once the weight exceeded 32.767.

Please refer to Appendix A and Appendix B for each format's detailed description of data available to determine which is most suitable.

C.2. Byte Order

The byte order parameter sets the order in which the data bytes and words will be presented in the PLC data format. Available Byte Order selections are:

Word Swap	Makes the data format compatible with RSLogix 5000 processors.
Byte Swap	Makes the data format compatible with S7 Profibus.
Standard	Makes the data format compatible with PLC 5
Double Word Swap	Makes the data format compatible with the Modicon Quantum PLC for Modbus TCP networks.

Table C-1 provides examples of the various byte ordering.

Table C-1: PLC Data Byte Ordering

		Word Swap			Byte Swap			Double Word Swap			Standard		
Terminal Weight Value		1355											
PLC		15	Bit #	0	15	Bit #	0	15	Bit #	0	15	Bit #	0
Integer	Weight value word	0x054B Hex			0x4B05 Hex			0x4B05 Hex			0x054B Hex		
Floating Point	1st Weight value word	0x6000 Hex			0xA944 Hex			0x0060 Hex			0x44A9 Hex		
	2nd Weight value word	0x44A9 Hex			0x0060 Hex			0xA944 Hex			0x6000 Hex		

Please refer to Appendix A and Appendix B for each format's detailed description of data available to determine which is most suitable.

C.3. Controlling Discrete I/O Using a PLC Interface

The IND570 terminal provides the ability to directly control some of its discrete outputs and read some of its discrete inputs via the (digital) PLC interface options. System integrators should be aware that the terminal's discrete I/O updates are synchronized with the terminal's A/D rate and not with the PLC I/O scan rate. This may cause a noticeable delay in reading inputs or updating outputs as observed from the PLC to real world signals.

Consult the **IND570 Terminal Installation Manual** for discrete I/O wiring. Also note that the outputs must be unassigned in the IND570 terminal at **Setup > Application > Discrete I/O** in order to be controlled by the PLC.

To protect your product's future:

METTLER TOLEDO Service assures the quality, measuring accuracy and preservation of value of this product for years to come.

Please request full details about our attractive terms of service.

 www.mt.com/service

www.mt.com/IND570

For more information

Mettler-Toledo, LLC
1900 Polaris Parkway
Columbus, OH 43240

© 2024 Mettler-Toledo, LLC
30205335 Rev. M, 1/2024



30205335