



Where Automation Connects.



inRAX[®]
MVI56-AFC
ControlLogix Platform
Liquid and Gas Flow Computer

February 25, 2011

USER MANUAL

Your Feedback Please

We always want you to feel that you made the right decision to use our products. If you have suggestions, comments, compliments or complaints about our products, documentation, or support, please write or call us.

How to Contact Us

ProSoft Technology

5201 Truxtun Ave., 3rd Floor

Bakersfield, CA 93309

+1 (661) 716-5100

+1 (661) 716-5101 (Fax)

www.prosoft-technology.com

support@prosoft-technology.com

Copyright © 2011 ProSoft Technology, Inc., all rights reserved.

MVI56-AFC User Manual

February 25, 2011

ProSoft Technology[®], ProLinx[®], inRAx[®], ProTalk[®], and RadioLinx[®] are Registered Trademarks of ProSoft Technology, Inc. All other brand or product names are or may be trademarks of, and are used to identify products and services of, their respective owners.

ProSoft Technology[®] Product Documentation

In an effort to conserve paper, ProSoft Technology no longer includes printed manuals with our product shipments. User Manuals, Datasheets, Sample Ladder Files, and Configuration Files are provided on the enclosed CD-ROM, and are available at no charge from our web site: www.prosoft-technology.com

Important Installation Instructions

Power, Input, and Output (I/O) wiring must be in accordance with Class I, Division 2 wiring methods, Article 501-4 (b) of the National Electrical Code, NFPA 70 for installation in the U.S., or as specified in Section 18-1J2 of the Canadian Electrical Code for installations in Canada, and in accordance with the authority having jurisdiction. The following warnings must be heeded:

- A** WARNING - EXPLOSION HAZARD - SUBSTITUTION OF COMPONENTS MAY IMPAIR SUITABILITY FOR CLASS I, DIV. 2;
- B** WARNING - EXPLOSION HAZARD - WHEN IN HAZARDOUS LOCATIONS, TURN OFF POWER BEFORE REPLACING OR WIRING MODULES
- C** WARNING - EXPLOSION HAZARD - DO NOT DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NON-HAZARDOUS.
- D** THIS DEVICE SHALL BE POWERED BY CLASS 2 OUTPUTS ONLY.

MVI (Multi Vendor Interface) Modules

WARNING - EXPLOSION HAZARD - DO NOT DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NON-HAZARDOUS.

AVERTISSEMENT - RISQUE D'EXPLOSION - AVANT DE DÉCONNECTER L'ÉQUIPEMENT, COUPER LE COURANT OU S'ASSURER QUE L'EMPLACEMENT EST DÉSIGNÉ NON DANGEREUX.

Warnings

North America Warnings

Power, Input, and Output (I/O) wiring must be in accordance with Class I, Division 2 wiring methods, Article 501-4 (b) of the National Electrical Code, NFPA 70 for installation in the U.S., or as specified in Section 18-1J2 of the Canadian Electrical Code for installations in Canada, and in accordance with the authority having jurisdiction. The following warnings must be heeded:

- A** Warning - Explosion Hazard - Substitution of components may impair suitability for Class I, Division 2.
- B** Warning - Explosion Hazard - When in hazardous locations, turn off power before replacing or rewiring modules.
- C** Warning - Explosion Hazard - Do not disconnect equipment unless power has been switched off or the area is known to be non-hazardous.

Avertissement - Risque d'explosion - Avant de déconnecter l'équipement, couper le courant ou s'assurer que l'emplacement est désigné non dangereux.

- D** Suitable for use in Class I, Division 2 Groups A, B, C and D Hazardous Locations or Non-Hazardous Locations.

ATEX Warnings and Conditions of Safe Usage

Power, Input, and Output (I/O) wiring must be in accordance with the authority having jurisdiction.

- A** Warning - Explosion Hazard - When in hazardous locations, turn off power before replacing or wiring modules.
- B** Warning - Explosion Hazard - Do not disconnect equipment unless power has been switched off or the area is known to be non-hazardous.
- C** These products are intended to be mounted in an IP54 enclosure. The devices shall provide external means to prevent the rated voltage being exceeded by transient disturbances of more than 40%. This device must be used only with ATEX certified backplanes.
- D** DO NOT OPEN WHEN ENERGIZED.

Battery Life Advisory

The MVI46, MVI56, MVI56E, MVI69, and MVI71 modules use a rechargeable Lithium Vanadium Pentoxide battery to backup the real-time clock and CMOS. The battery should last for the life of the module. The module must be powered for approximately twenty hours before the battery becomes fully charged. After it is fully charged, the battery provides backup power for the CMOS setup and the real-time clock for approximately 21 days. When the battery is fully discharged, the module will revert to the default BIOS and clock settings.

Note: The battery is not user replaceable.

Markings

Electrical Ratings

- Backplane Current Load: 800 mA @ 5.1 Vdc; 3 mA @ 24 Vdc
- Operating Temperature: 0°C to 60°C (32°F to 140°F)
- Storage Temperature: -40°C to 85°C (-40°F to 185°F)
- Shock: 30 g, operational; 50 g, non-operational; Vibration: 5 g from 10 Hz to 150 Hz
- Relative Humidity: 5% to 95% with no condensation
- All phase conductor sizes must be at least 1.3 mm(squared) and all earth ground conductors must be at least 4mm(squared).

Label Markings

ATEX

II 3 G

EEx nA IIC T6

0°C <= Ta <= 60°C

cULus

E183151

Class I Div 2 Groups A,B,C,D

T6

-30°C <= Ta <= 60°C

Agency Approvals and Certifications

Agency	Applicable Standard
RoHS	
CE	EMC-EN61326-1:2006; EN61000-6-4:2007
ATEX	EN60079-15:2003
cULus	UL508; UL1604; CSA 22.2 No. 142 & 213
CB Safety	CA/10533/CSA IEC 61010-1 Ed.2; CB 243333-2056722 (2090408)
GOST-R	EN 61010
CSA	EN 61010

RoHS



243333



ME06



E183151

Contents

Your Feedback Please.....	2
How to Contact Us.....	2
ProSoft Technology® Product Documentation.....	2
Important Installation Instructions.....	3
MVI (Multi Vendor Interface) Modules.....	3
Warnings.....	3
Battery Life Advisory.....	3
Markings.....	4
1 Introduction	11
1.1 Update Notice.....	12
1.2 MVI56-AFC Module.....	14
1.3 Configuration Modification Lockout and Seal.....	15
2 Quick Start	17
2.1 Install AFC Manager.....	18
2.1.1 System Requirements.....	18
2.2 Starting AFC Manager.....	19
2.3 Using AFC Manager.....	20
2.3.1 Starting a New Project.....	20
2.3.2 Loading an Existing project.....	21
2.3.3 Printing the Configuration Report.....	21
2.3.4 Converting a Project.....	22
2.3.5 Resetting Configuration Parameters.....	23
2.3.6 Downloading the Project to the Module.....	24
2.3.7 Verifying Correct Operation.....	25
2.4 Ladder Logic Implementation.....	26
2.5 Setting the Wallclock.....	28
2.6 Module Initialization.....	29
3 Meter Channel Functionality	31
3.1 Meter Channels.....	32
3.2 Linear (Pulse) Meter Overview.....	33
3.2.1 Primary Input = Pulse Count.....	33
3.2.2 Primary Input = Pulse Frequency.....	33
3.3 Differential (Orifice) Meter Overview.....	34
3.3.1 Primary Input = Differential Pressure.....	34
3.3.2 Primary Input = Flow Rate.....	34
3.4 Gas Product Overview.....	35
3.5 Liquid Product Overview.....	36
3.5.1 To use a densitometer.....	36
3.5.2 Module Configuration.....	36
3.5.3 Density Units.....	36
3.5.4 Measuring Water Diluent.....	36
3.6 General Features.....	37
3.6.1 Process Variable Interface.....	37

3.6.2	Meter Scan Time.....	37
3.6.3	Multiple Meter Accumulators	37
3.6.4	Product Batching	37
3.6.5	Data Archiving.....	38
3.6.6	Event Log Function.....	38
3.6.7	Measurement Units.....	38
3.6.8	Process Input Scaling	39
4	Meter Proving	41
4.1	Prover Configuration.....	42
4.1.1	Prover Type	42
4.1.2	Prover Options.....	46
4.1.3	Run Counts.....	47
4.1.4	Run Input Setup.....	47
4.1.5	Prover Characteristics	49
4.2	Setting up the AFC module for Meter Proving.....	51
4.2.1	Initial Requirements	53
4.2.2	Meter Proving Alarms	54
4.2.3	Prover Operation (How to do a Prove)	57
4.3	Meter Proving Reports.....	63
4.4	Protected Meter Proving Data in the AFC's Input Register Bank.....	64
4.4.1	Latest Prove Results.....	64
4.4.2	Meter Previous Prove Summary.....	67
5	Modbus Database	69
5.1	AFC Modbus Address Space	70
5.1.1	Accessing the Data.....	70
5.2	Primary Slave.....	71
5.2.1	Modbus Address References	71
5.2.2	Modbus Address Examples	71
5.2.3	Meter-relative Data	72
5.2.4	Scratchpad.....	73
5.3	Virtual Slave.....	74
5.3.1	Virtual Slave Example Application	75
6	Modbus Communication	77
6.1	Communication Parameters	78
6.2	Port Options.....	79
6.3	Modbus Master	80
6.3.1	Example	81
6.4	Modbus Pass-Through	82
7	Accumulators	83
7.1	Accumulator Totalizer and Residue.....	84
7.2	Accumulator Types	85
7.2.1	Non-Resettable Accumulators	85
7.2.2	Resettable Accumulators.....	85
7.2.3	Archive Accumulators	88

7.3	Net Accumulator Calculation	89
7.4	Frequently Asked Questions	90
8	Archives	91
8.1	Archive Overview	92
8.2	Archive Generation	93
8.3	Archive Types	94
8.4	Archive Order	95
8.5	Archive Options	97
8.6	Archive Locations	98
8.7	Editing the Archive Structure	100
8.8	Extended Archives	102
8.8.1	Retrieving Extended Archives	102
8.9	Archive Reports	105
8.10	Archive Monitor	107
9	Events	113
9.1	The Event Log	114
9.2	Event Log Structures	115
9.3	Event Id Tag	116
9.4	Event-triggered Archives and Accumulator Resets	117
9.5	Downloading the Event Log in Firmware Version 2.07 and Later	118
9.5.1	Basic Principles of Implementation	125
9.5.2	Data Elements	127
9.5.3	Virtual Slave Precedence Relations	129
9.5.4	Security and Optimization	130
9.5.5	The Log-Download Window (LDW)	131
9.5.6	Modbus Transaction Sequencing and Constraints	132
9.5.7	Access by Multiple Hosts	137
9.5.8	Other Considerations	138
9.6	Period-end Events	139
9.7	Loggable Events	140
9.8	Special Events	141
9.9	Site Data Point Events	142
9.10	Meter Data Point Events	143
9.11	Stream Data Point Events	146
9.12	Prover Data Point Events	148
9.13	"Rkv" Notes	151
9.14	Downloading the Event Log in Firmware Version 2.05 and Earlier	152
10	Security (Passwords)	155
10.1	Hard Password	157
11	MVI56-AFC Backplane Communication	159
11.1	MVI56-AFC Module Data Transfer	160
11.1.1	Input/Output Blocks for Data Transfer	160
11.1.2	Input/Output Transactions	162
11.2	MVI56-AFC Function Block Interface	167

11.2.1	Function Block Structure.....	168
11.2.2	Function Block Definition - 0: Null.....	171
11.2.3	Function Block Definition - 1: Wall Clock.....	172
11.2.4	Function Block Definition - 4, 5, 6 & 7: Modbus Pass Through.....	173
11.2.5	Function Block Definition - 8: Meter Process Variables.....	175
11.2.6	Function Block Definition - 9: Meter Analysis, 16-bit.....	184
11.2.7	Function Block Definition - 10: Meter Type Fetch.....	186
11.2.8	Function Block Definition - 11: Meter Analysis, 32-bit.....	187
11.2.9	Function Block Definition - 12: Site/Meter Signals.....	190
11.2.10	Function Block Definition - 14: Meter Archive Fetch.....	192
11.2.11	Function Block Definition - 16/17/18/19: Modbus Gateway Read.....	193
11.2.12	Function Block Definition - 20, 21: Modbus Gateway Write.....	196
11.2.13	Function Block Definition - 24, 25, 26: Modbus Master.....	199
11.2.14	Function Block Definition - 28, 29: Disable/Enable Meters.....	203
12	MVI56-AFC Sample Logic	205
<hr/>		
12.1	Sample Logic Overview.....	206
12.1.1	Process Block (uses Transaction Numbers from 1 to 16).....	208
12.1.2	Modbus Gateway Block (uses Transaction Numbers from 17 to 25).....	209
12.1.3	Wallclock Block (uses Transaction Number =99).....	209
12.1.4	Sample MVI56-AFC Logic Tasks.....	210
12.2	Using the Sample Add-On Instruction.....	211
12.2.1	Import Procedure.....	211
12.3	ControlLogix Sample Logic Details.....	220
12.3.1	Enable/Disable Status.....	220
12.3.2	Disable Meter.....	220
12.3.3	Enable Meter.....	221
12.3.4	Wallclock.....	222
12.3.5	Meter Profile.....	223
12.3.6	Meter Process Variables.....	224
12.3.7	Meter Calculation Results.....	227
12.3.8	Meter Signals.....	229
12.3.9	Molar Analysis (For Gas Product Only).....	231
12.3.10	Set the Processor Time.....	236
12.3.11	Checking Meter Alarms.....	237
12.3.12	Site Status.....	239
12.3.13	Modbus Master.....	239
12.3.14	Modbus pass-through.....	244
12.3.15	Modbus Gateway.....	245
13	Diagnostics and Troubleshooting	255
<hr/>		
13.1	User LEDs.....	256
13.1.1	App Status LED.....	256
13.1.2	BP Act and P1, P2, or P3.....	256
13.2	BBRAM LEDs.....	257
13.3	Meter Alarms.....	258
13.4	Checksum Alarms.....	262
13.5	Events.....	263
13.6	Audit Scan.....	264

14 Reference 269

14.1	General Specifications	270
14.1.1	On-line Communication & Configuration.....	271
14.1.2	Reports	271
14.1.3	Modbus Interface.....	271
14.1.4	Configurable Options.....	272
14.1.5	Supported Meters.....	272
14.1.6	Hardware Specifications.....	273
14.2	Measurement Standards	274
14.2.1	Basic Metering According to Meter type	275
14.2.2	Liquid Correction Factor Details.....	277
14.3	Wedge Meter Applications	279
14.4	Configurable Archive Registers.....	280
14.4.1	Information for Users of AFC Manager Versions Older Than 2.01.000.....	284
14.5	Archive Data Format	286
14.5.1	Timestamp Date and Time Format	286
14.5.2	Pre-defined Header	287
14.5.3	Orifice (Differential) Meter with Gas Product.....	288
14.5.4	Pulse (Linear) Meter with Gas Product	289
14.5.5	Orifice (Differential) Meter with Liquid Product.....	289
14.5.6	Pulse (Linear) Meter with Liquid Product	290
14.5.7	Flow Rate Integration with Gas Product.....	290
14.5.8	Pulse Frequency Integration with Gas Product.....	291
14.5.9	Flow Rate Integration with Liquid Product.....	291
14.5.10	Pulse Frequency Integration with Liquid Product.....	292
14.6	Modbus Addressing Common to Both Primary and Virtual Slaves.....	293
14.7	Modbus Port configuration	296
14.8	Startup Basics and Frequently Asked Questions.....	298
14.9	Cable Connections	299
14.9.1	RS-232 Configuration/Debug Port	299
14.9.2	RS-232 Application Port(s).....	299
14.9.3	RS-422	302
14.9.4	RS-485 Application Port(s).....	302
14.9.5	DB9 to RJ45 Adaptor (Cable 14)	303

15 Support, Service & Warranty 305

	Contacting Technical Support.....	305
15.1	Return Material Authorization (RMA) Policies and Conditions.....	307
15.1.1	Returning Any Product	307
15.1.2	Returning Units Under Warranty	308
15.1.3	Returning Units Out of Warranty	308
15.2	LIMITED WARRANTY.....	309
15.2.1	What Is Covered By This Warranty.....	309
15.2.2	What Is Not Covered By This Warranty	310
15.2.3	Disclaimer Regarding High Risk Activities	310
15.2.4	Intellectual Property Indemnity.....	311
15.2.5	Disclaimer of all Other Warranties	311
15.2.6	Limitation of Remedies **	312
15.2.7	Time Limit for Bringing Suit	312
15.2.8	No Other Warranties	312
15.2.9	Allocation of Risks.....	312

15.2.10 Controlling Law and Severability 312

Index **313**

1 Introduction

In This Chapter

- ❖ Update Notice..... 12
- ❖ MVI56-AFC Module 14
- ❖ Configuration Modification Lockout and Seal 15

The MVI56-AFC Flow Computer module performs measurement of hydrocarbon gases and liquids using currently accepted industry measurement standards. The module consists of a single-slot solution for Rockwell Automation chassis. To obtain its process inputs for calculations, the module uses the process data collected by analog and pulse I/O modules. The processor transfers this data to the AFC module, which then calculates flow rates, accumulated volumes, and accumulated mass. The results of the calculations are transferred back to the processor for use in the application logic, or for transfer to a SCADA host.

The module has two communication ports for Modbus communication allowing easy access to a remote Modbus device. The module works as a Modbus slave or master device.

As discussed later in this manual, the internal Modbus database can be accessed by a Modbus Master device and by the processor (using the Modbus Gateway Function).

The AFC Manager software can be used for easy meter configuration and application monitoring. Refer to the *AFC Manager User Manual* for complete information about this tool.

The following section provides a sample application where input data is transferred from the transmitters to analog input cards on the Rockwell Automation rack and the values are transferred from the processor to the module (the module supports floating-point, scaled integer, or 4 to 20 mA format).

For Pulse meter applications, the pulse count and pulse frequency values are typically transmitted through high-speed counter modules in the rack.

The module performs the flow calculation based on the values transferred through the backplane. The calculation results can be read to the processor or polled from a remote Modbus master unit connected to one of the communication ports.

1.1 Update Notice

If your module measures liquids, please read this notice before upgrading from version 2.04 (or earlier) to 2.05 (or later).

For compliance with new measurement standards, the AFC version 2.05 has introduced several new liquid product groups. In particular, the two non-refined liquid product groups of version 2.04, which covered the entire density range of crudes and NGLs, have each been split into two separate product groups, one for the higher density range of crudes and the other for the lower density range of NGLs. If your module has meter channels configured for either "Crude, NGL" or "Oil-water emulsion", you should decide **before upgrading the firmware** the new product group (light or heavy) to which each such channel should be assigned. This assignment will be performed during the upgrade process and will preserve all other configuration and historical records including accumulator values and archives, in contrast to changing a product group after the upgrade which resets the meter configuration and erases all historical records. Meter channels configured for "Gas" or "Refined products" are not affected.

AFC Manager exhibits the same behavior when converting a project between versions 2.04 (or earlier) and 2.05 (or later).

The criterion for assigning the new product group depends on the density units and the Default Reference Density, as described in the following tables:

Density Units = kg/m3

Version 2.04 Product Group	Default Reference Density	Version 2.05 Product Group
Crude, NGL	= 0 OR ≥ 610.0	Crude oils, JP4
Crude, NGL	> 0 AND < 610.0	NGLs, LPGs
Oil Water Emulsion	= 0 OR ≥ 610.0	Oil-water emulsion (Crd)
Oil Water Emulsion	> 0 AND < 610.0	Oil-water emulsion (NGL)

Density Units = Rd/60

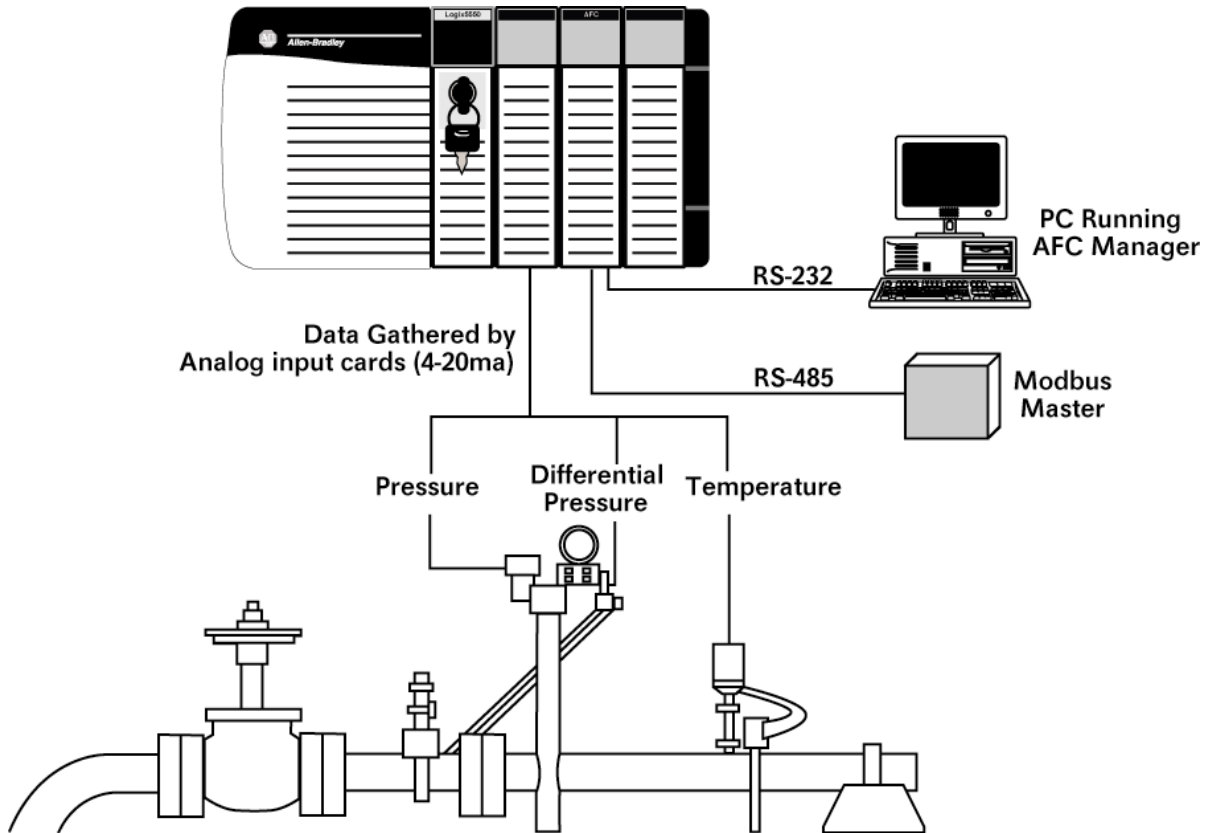
Version 2.04 Product Group	Default Reference Density	Version 2.05 Product Group
Crude, NGL	= 0 OR ≥ 0.6100	Crude oils, JP4
Crude, NGL	> 0 AND < 0.6100	NGLs, LPGs
Oil Water Emulsion	= 0 OR ≥ 0.6100	Oil-water emulsion (Crd)
Oil Water Emulsion	> 0 AND < 0.6100	Oil-water emulsion (NGL)

Due to roundoff error of numeric conversions, a Relative Density very close to the cutoff value of 0.6100 may cause the module to assign the new product group opposite to the one that was intended. Before upgrading, change the Default Reference Density to a number significantly different from 0.6100, such as 0.6110 (to target Crude) or 0.6090 (to target NGLs). You may change it back to the correct value after the upgrade.

Density Units = API Gravity

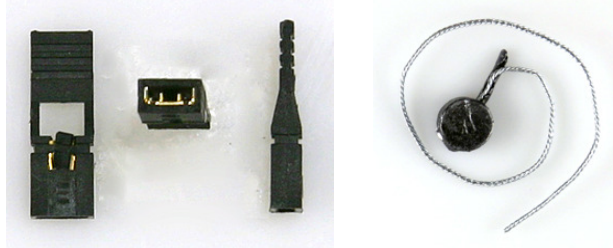
Version 2.04 Product Group	Default Reference Density	Version 2.05 Product Group
Crude, NGL	= 0 OR ≤ 100.0	Crude oils, JP4
Crude, NGL	> 0 AND > 100.0	NGLs, LPGs
Oil Water Emulsion	= 0 OR ≤ 100.0	Oil-water emulsion (Crd)
Oil Water Emulsion	> 0 AND > 100.0	Oil-water emulsion (NGL)

1.2 MVI56-AFC Module



1.3 Configuration Modification Lockout and Seal

The MVI56-AFC application configuration can be certified and sealed with a user-installable Lockout jumper and a tamper-evident lead seal. The Lockout jumper and seal are commonly required for Weights & Measures certification, or custody transfer applications.



Important: When the jumper is installed, the module will not accept configuration changes to Sealable Parameters, which are parameters that affect the accuracy of flow calculation. Before breaking the seal to remove the jumper, you should verify the steps required to recertify the module with the appropriate regulatory agency.

For more information on sealing procedures, refer to "Sealing Provisions", on page 8 of the *MVI56-AFC Custody Transfer Certification* document, which is available from the ProSoft Technology web site at <http://www.prosoft-technology.com/content/view/full/4613>

To install the Lockout jumper and seal, follow these steps.



- 1 Locate the Lockout Jumper pins and Lockout Block, labeled "W & M Lock" inside the module door, and below the BBRAM ERR and OK LEDs.



- 1 Install the provided Lockout Jumper to connect the two pins.



- 1 Carefully slide the Lockout Block up through the hole in the Lockout Jumper. Be careful not to bend or break the jumper block pins.



- 1 When the Lockout Block is positioned correctly, it will expose a hole in the block, through which you may pass the seal wire.



- 1 Slide the seal wire through the hole in the Lockout Block. Pass the wire through the slot in the lead seal, and then crimp the lead seal around the wire.

Once sealed, the Lockout block and jumper cannot be removed without damaging the seal, the block, or the jumper.

2 Quick Start

In This Chapter

❖ Install AFC Manager.....	18
❖ Starting AFC Manager.....	19
❖ Using AFC Manager.....	20
❖ Ladder Logic Implementation	26
❖ Setting the Wallclock.....	28
❖ Module Initialization.....	29

This section provides a general overview of the steps required to install and configure the module. You should read the *AFC Manager User Manual* to obtain a clear understanding of the steps outlined in this section.

2.1 Install AFC Manager

The AFC Manager application is included on the CD-ROM shipped with your module. Before you can use the application, you must install it on your computer.

2.1.1 System Requirements

The following system requirements are the recommended minimum specifications to successfully install and run AFC Manager:

- Microsoft Windows compatible PC
- Windows 2000 with Service Pack 2 or higher, or Windows XP Professional with Service Pack 2 or higher, or Windows 2003 or Windows Vista, or Windows 7.
- 300 MHz Pentium processor (or equivalent)
- 128 megabytes of RAM
- 20 megabytes of free disk space
- Available serial port (COM port) or USB to Serial adapter cable with necessary drivers, required for communication between AFC Manager software and the AFC module.
- DB9 adapter cable (included with module), required for connection between PC serial port and AFC module (PTQ-AFC module does not require an adapter).

To install the AFC Manager application

- 1 Insert the ProSoft Solutions CD in your CD-ROM drive. On most computers, a menu screen will open automatically. If you do not see a menu within a few seconds, follow these steps:
 - a Click the Start button, and then choose Run.
 - b In the Run dialog box, click the Browse button.
 - c In the Browse dialog box, click "My Computer". In the list of drives, choose the CD-ROM drive where you inserted the ProSoft Solutions CD.
 - d Select the file **prosoft.exe**, and then click Open.
 - e On the Run dialog box, click OK.
- 2 On the CD-ROM menu, click Documentation and Tools. This action opens a Windows Explorer dialog box.
- 3 Open the Utilities folder, and then open the AFCManager folder.
- 4 Double-click the file Setup.exe. If you are prompted to restart your computer so that files can be updated, close all open applications, and then click OK. When your computer has finished restarting, begin again at Step 1.
- 5 Click OK or Yes to dismiss any confirmation dialog boxes.
- 6 It may take a few seconds for the installation wizard to start. Click OK on the AFC Manager Setup dialog box to begin installing AFC Manager.
- 7 Follow the instructions on the installation wizard to install the program with its default location and settings.
- 8 When the installation finishes, you may be prompted to restart your computer if certain files were in use during installation. The updated files will be installed during the restart process.

2.2 Starting AFC Manager

To start AFC Manager

- 1 Click the **START** button, and then choose **PROGRAMS**.
- 2 In the Programs menu, choose ProSoft Technology.
- 3 In the ProSoft Technology menu, choose AFC Manager.

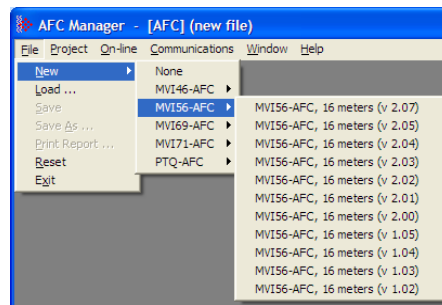
2.3 Using AFC Manager

The AFC module is configured with configuration files that you create using AFC Manager. A configuration file is called a Project.

2.3.1 Starting a New Project

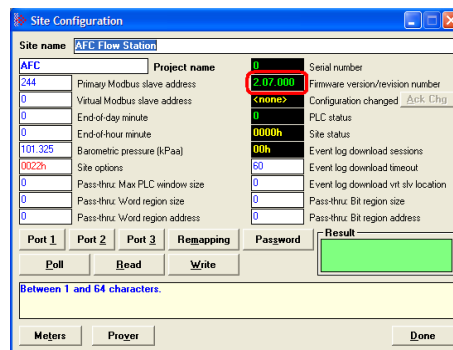
To start a new project

- 1 Start **AFC MANAGER**, and then open the *File* Menu.
- 2 On the *File* Menu, choose **NEW**, and then select your module and firmware version number.



The version number refers to the firmware version of your module. If you do not know the firmware version number, follow these steps:

- a) Open the *Project* menu.
- b) Choose **SITE CONFIGURATION**. This action opens the *Site Configuration dialog box*.
- c) Click the **READ** button. The firmware version is listed below the serial number, in the upper right part of the dialog box.



Important: You must be connected to the module and "online" to read data from the module.

- 3 Follow the steps in the remainder of this User Guide to configure your module and your AFC device.
- 4 Before closing the program, open the *File* menu and choose **SAVE AS**, to save your project so you can open it again later.

2.3.2 Loading an Existing project

You can open and edit a project you have previously saved. Do this if you have started, but not completed, the configuration of your project, or if you need to modify the settings for a project that has already been downloaded to the module.

To load an existing project

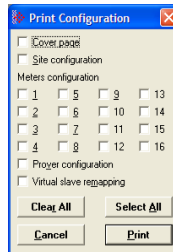
- 1 Start **AFC MANAGER**, and then open the *File* menu.
- 2 On the *File* menu, choose **LOAD**. This action opens a dialog box that shows a list of AFC Manager project files (AFC files) in the current folder.
- 3 Choose the project to load, and then click **OPEN**.

2.3.3 Printing the Configuration Report

You can print a report of your configuration for future reference, or for archival purposes.

To print the configuration report

- 1 Open the *File* menu, and then select **PRINT REPORT**. This action opens the *Print Configuration* dialog box.



- 2 On the *Print Configuration* dialog box, select (check) the items to include in the printed report.
- 3 Click **PRINT** to send the report to your default printer.

Note: The size of the report depends on items you choose to include, and may require 75 pages or more. Consider this before printing.

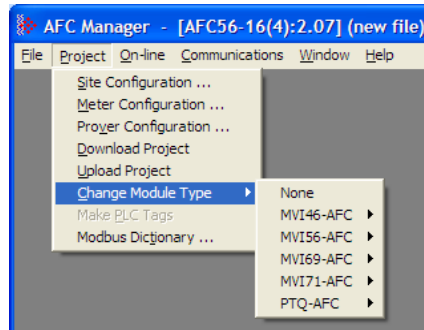
2.3.4 Converting a Project

You can convert an existing project (configuration file) to use it with a different module or firmware version. Do this if:

- You want to reuse an application created for a different AFC module, for example a project that was created for a PTQ-AFC that you want to use for an MVI69-AFC.
- You apply a firmware upgrade to a module.

To convert a project:

- 1 Open the *File* menu, and then choose **OPEN**.
- 2 Open the project (configuration file) to convert.
- 3 Open the *Project* menu, and then choose **CHANGE MODULE TYPE**.



- 4 Choose the module type and firmware version from the menu.
- 5 Save your project.

Note: AFC Manager will save your updated configuration file with the same name as the file you loaded. If you need to keep your original configuration, change the file name of your updated configuration before saving.

2.3.5 *Resetting Configuration Parameters*

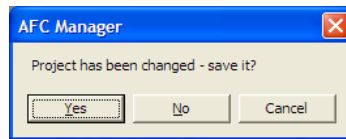
If you have modified your project (configuration file), or if you have loaded a configuration file from disk, but you want to start a new project, you can reset the configuration parameters back to their defaults without having to close and reopen the AFC Manager.

To reset configuration parameters

- 1 Close any dialog boxes that are open.
- 2 Save the configuration file you were working on, if you would like to load it again later.
- 3 On the *File* menu, choose **RESET**.

Note: This procedure has the same effect as choosing **File / New / None**.

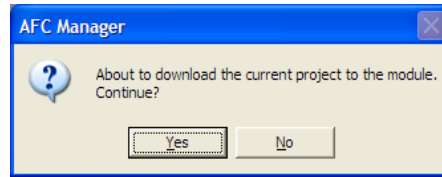
If you have made changes to the configuration that have not yet been saved, a confirmation dialog box will open.



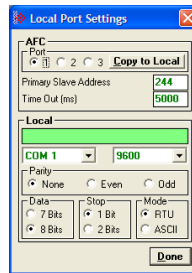
Answer Yes to save your changes, or No to discard your changes and begin working on a new configuration. Click Cancel to abandon the attempted action that caused this message.

2.3.6 Downloading the Project to the Module

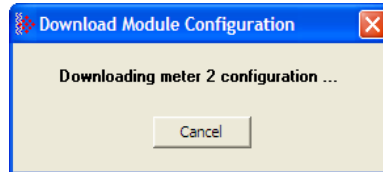
- 1 Click **PROJECT / DOWNLOAD PROJECT**.



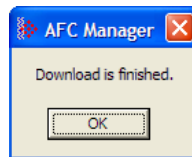
- 2 This action opens the Local Port Settings window. Enter the port parameters to use, and then click **DONE**.



- 3 During the download operation, the following progress window is displayed:



- 4 When the file transfer is complete, the following window is displayed:



Note: The virtual slave remapping data (page 74) is not downloaded during the procedure because it requires a separate download operation.

Troubleshooting Tip: If the AFC Manager displays an "Illegal Data Value" message, it typically indicates an invalid meter type or product group configuration. The module does not accept a configuration file that attempts to change a meter type or product group for a meter that is currently enabled. Disable all meters, change the meter types and product groups, and then enable the meters again.

2.3.7 Verifying Correct Operation

When all of the configuration steps have been completed, the module should be ready to perform measurement calculations. To verify that the module is configured correctly, follow these steps:

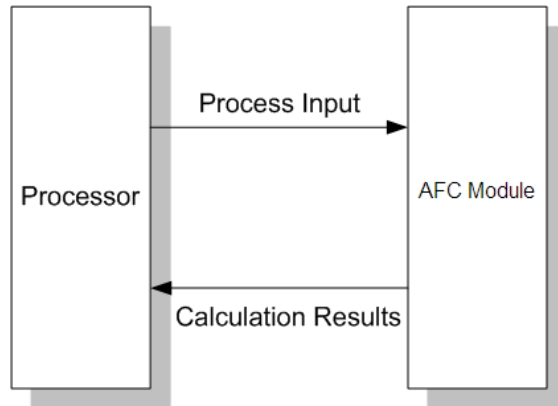
- 1** Enable all meters that will be used, as any meter will only perform calculations if it is enabled. Any meter can be enabled either with ladder logic (MVI56-AFC modules), function blocks (PTQ modules) or with AFC Manager.
- 2** Make sure that the wallclock is running, and that it has valid date and time information. After power-up, the wallclock will be stopped, therefore the module will not perform any time-scheduled operations, such as writing period-end archives, and will not timestamp records written to the event log until it receives a wallclock command from the ladder logic.

The sample ladder logic programs the wallclock update command upon detecting "power-up" status from the AFC. The date/time information used is the same as the processor, therefore you should use the configuration tool for your processor to verify that the processor has valid date/time data. If the processor wallclock is not valid (for example if the year = 1900), the module will not accept the command. You may easily determine if the wallclock is running by performing two consecutive read operations in the Meter Monitor.

- 3** Make sure that the meter does not have any alarms. A meter alarm may affect flow calculation. Look at the Meter Monitor dialog box for alarms.
- 4** Make sure that the input parameters transferred from the processor are correct. You can look at these values in the Meter Monitor dialog box.
- 5** When using a pulse meter, make sure that the pulse input rollover parameter in Meter Configuration matches the actual input rollover value used in the high speed counter module.

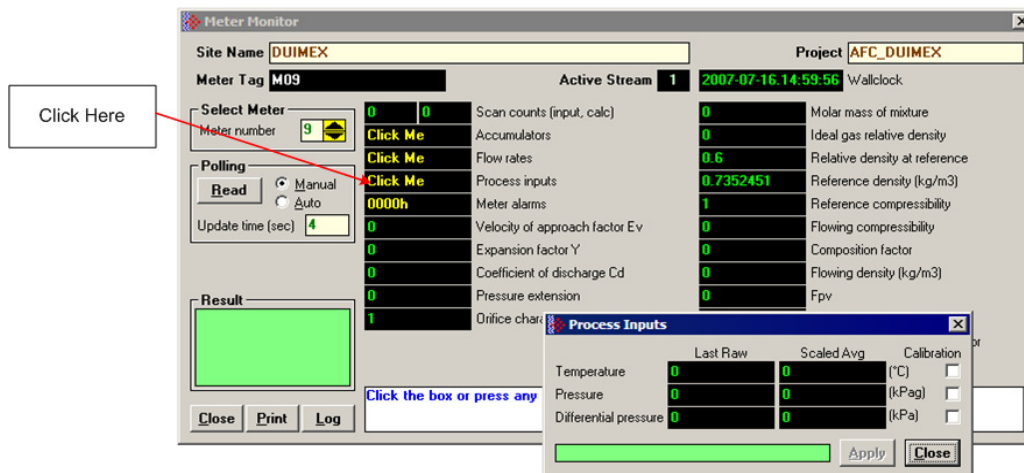
2.4 Ladder Logic Implementation

The sample ladder logic performs tasks that are covered in the Ladder Logic sections of this manual. The most important task is to continuously write meter process input variables from the processor to the module, and read calculation results from the module to the processor.

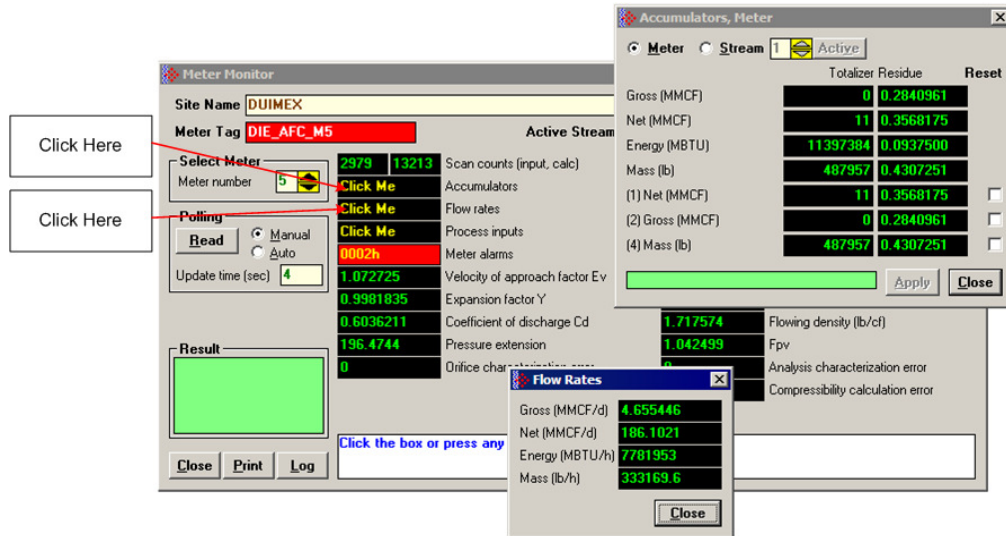


Refer to the Ladder Logic sections for instructions on how to transfer the meter process variables from the processor to the module. Ladder logic is required to move the process variables to the correct data file or controller tag in the processor.

The **Meter Monitor** window (*Process Inputs* field) displays the values that are transferred from the processor.



The values calculated by the module are continuously transferred to the processor. You can refer to the **Meter Monitor** window to verify results calculated by the module.



Refer to the Ladder Logic section for more information regarding the data files and controller tags that store the calculation results transferred from the module (for example, accumulator, flow rate, and so on).

2.5 Setting the Wallclock

After power-up, the module must receive valid wallclock data from the ladder logic to perform time-scheduled operations and to properly timestamp historical records. The sample ladder logic automatically writes the wallclock upon detecting power-up status from the AFC using the processor's date and time information. You should ensure that the processor contains valid date and time information. If it does not, the module may not accept the wallclock block.

You can verify the wallclock information using the Meter Monitor section as shown in the following example:



2007-07-16 14:59:28 Wallclock

Refer to the Sample Ladder Logic section for more information on this topic.

2.6 Module Initialization

When the module is powered up for the first time, both the **OK** and **ERR** BBRAM LEDs are illuminated. This indicates that the module is in the *Cold Start* state and is not yet ready to perform calculations. The following steps initialize the module:

- Enable at least one meter
- Set the processor to RUN mode

After these two steps are accomplished, the state is changed from *Cold Start* to *Released*. This indicates that that module is ready to perform flow calculations. When in the *Released* state, the **OK** LED is ON and the **ERR** LED is off.

When the module is ready, you will use AFC Manager to monitor meter operation, archives, and events. The *AFC Manager User Manual* contains detailed information on these tasks.

3 Meter Channel Functionality

In This Chapter

❖ Meter Channels	32
❖ Linear (Pulse) Meter Overview	33
❖ Differential (Orifice) Meter Overview	34
❖ Gas Product Overview.....	35
❖ Liquid Product Overview.....	36
❖ General Features	37

3.1 Meter Channels

Each meter channel can be assigned as a linear meter (e.g. a *pulse meter*) input or as a differential meter (e.g. an *orifice meter*) input for flow measurement using either SI (metric) or US units. Selecting the differential meter causes the module to use the AGA 3 standard for flow calculation (for a gas orifice meter you may optionally choose ISO 5167-2 instead). Selecting the linear meter causes the module to use the AGA 7 standard for gas flow calculation.

Each meter channel can be configured for gas or liquid (*crude or refined*) product. The Product Group essentially selects the API/AGA Standards to be used in calculating flow rates/increments.

Selecting "Gas" causes use of AGA8 and either AGA3 or AGA7 Standards.

Selecting any liquid group causes use of API MPMS Chapter 11 (tables 23/24/53/54) and related Standards. "Crude oils, JP4" and "Oil-water emulsion Crd)" use the, "A", tables. "NGLs, LPG's" and "Oil-water emulsion (NGL) use the "E" tables. "Refined Products" use the "B" tables. "Lubricating oils" use the "D" tables, and "Special applications" use the "C" tables. "Crude oils, JP4" and "NGLs/LPG" are used for propane, butane, NGLs (natural gas liquids), and crude oils which are relatively water-free (less than 5 percent). The two "Oil-water emulsion" groups are used for crude and NGL/LPG that might have a high concentration of water for which API MPMS Chapter 20.1 is applicable. "Refined products (xJP4)" is used for lighter refined liquids such as gasolines, jet fuels (except JP4), and fuel oils. "Lubricating oils" is used for heavier refined liquids. "Special applications" is used for those liquids that cannot reasonably be assigned to one of the other groups; for this product group an explicit coefficient of thermal expansion must be supplied.

The following table provides a brief overview of the standards used according to the Meter Type and Product Group:

Meter Type	Product Group	Standards
Differential	Gas	AGA8, AGA3/ISO5167
Differential	Liquid	MPMS ch 11 AGA3/ISO5167
Linear	Gas	AGA8, AGA7
Linear	Liquid	MPMS ch 11, MPMS ch12.2



Note: The meter channel must be disabled in order to change its meter type and product group.

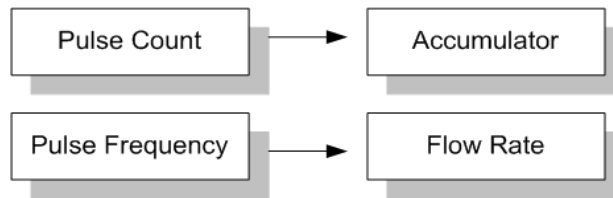
3.2 Linear (Pulse) Meter Overview

The module typically receives the pulse count and pulse frequency values from a high-speed counter module. The module uses these values to perform calculations.

You can configure the primary input to be used for volume calculation. You can configure it as Pulse Count or Pulse Frequency.

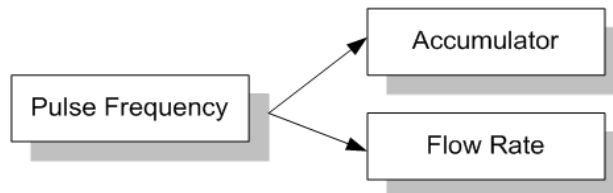
3.2.1 Primary Input = Pulse Count

If you select Pulse Count as the primary input, the module uses the pulse count value transferred through the backplane as the primary input for volume calculation. In this case, the pulse frequency will be used for flow rate calculation only.



3.2.2 Primary Input = Pulse Frequency

If you select Pulse Frequency as the primary input, the module uses the pulse frequency value transferred through the backplane as the primary input for both flow accumulation and flow rate calculation. The pulse count value is ignored by the module.



3.3 Differential (Orifice) Meter Overview

The static pressure of the gas stream can be measured either upstream of the meter (before the differential pressure drop), or downstream of the meter (after the pressure drop). Both AGA3 and AGA8 require the upstream static pressure for their calculations, where:

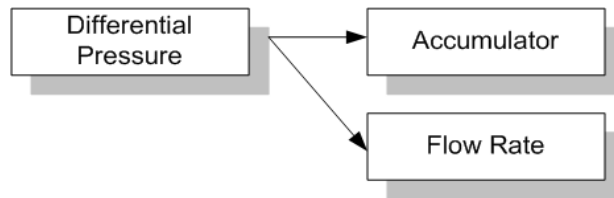
$$\text{upstream pressure} = \text{downstream pressure} + \text{differential pressure}$$

If the pressure is measured from a downstream tap (typical), the *Downstream Static Pressure* option should be set through the AFC Manager.

The module also supports the V-Cone device. You can configure V-Cone meters and downstream selections in AFC Manager, on the **Meter Configuration / Calculation Options** dialog box.

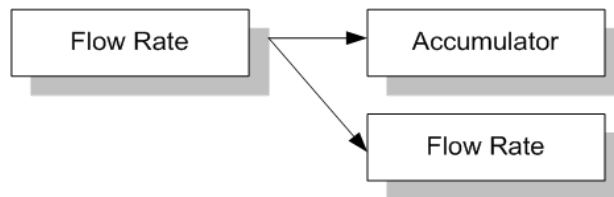
3.3.1 Primary Input = Differential Pressure

The primary input parameter configures the value used as source for the accumulator calculation. If the parameter is set to Differential Pressure, the module uses the differential pressure value transferred through the backplane for accumulator calculation.



3.3.2 Primary Input = Flow Rate

You can configure the primary input parameter as flow rate in order to use this value for the accumulator calculation.



Note: The flow rate can be converted to a different unit.

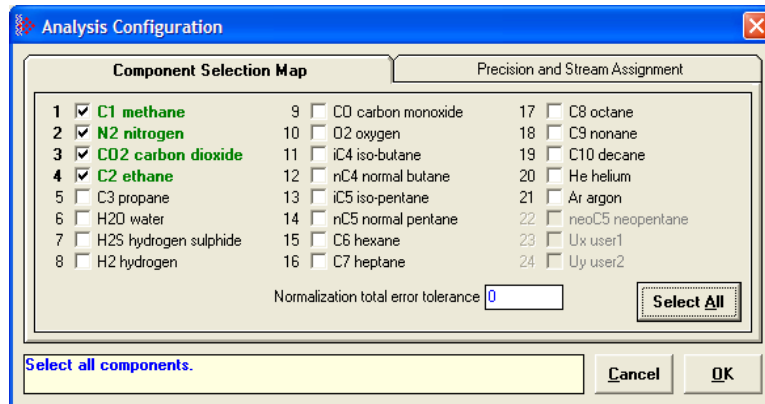
The AFC Manager software supports the following parameters:

- Orifice Plate and Meter Tube Measured Diameter
- Orifice Plate and Meter Tube Measurement Temperature
- Orifice Plate and Meter Tube, Coefficient of Thermal Expansion
- DP Flow Threshold (kPa)
- DP Alarm Threshold (kPa)

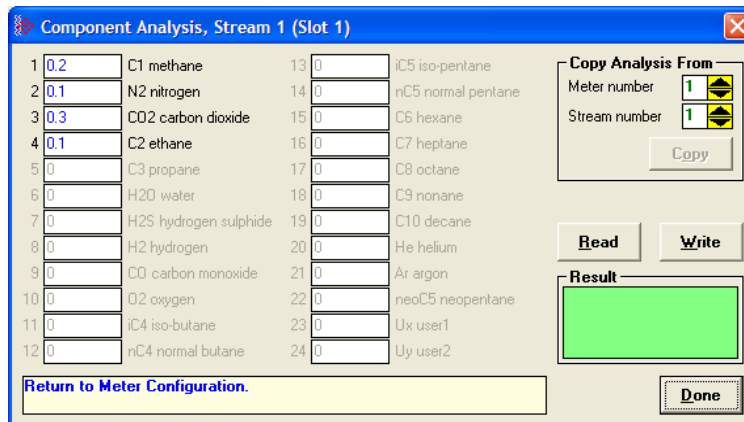
3.4 Gas Product Overview

The gas compressibility calculations are based on molar analysis concentrations of up to 21 components, using the Detail Characterization Method of AGA8 (1992). The module automatically generates alarms if the sum of the molar concentrations is not 100%

Configure the analysis settings using the AFC Manager (**Meter Configuration / Analysis Config**) as follows. This window allows the selection of the components(Component Selection Map) and analysis precision (Precision and Stream Assignment – version 2.06.000 or higher). The sample ladder logic assumes that all components are selected so check all components at the Component Selection Map window.



Enter the gas analysis concentrations by clicking the Analysis button. You can also update the concentrations through the backplane as will be later shown in this User Manual.



The module records events every time a molar concentration value changes. For applications that involve gas chromatograph devices, this feature might not be desirable because it is expected that the values should frequently change. You can disable this feature using AFC Manager (**Meter Configuration / Control Options / Treat Analysis as Process Input**).

3.5 Liquid Product Overview

The module supports applications involving crude or refined oil such as crude oil, oil/water emulsion, propane, butane, NGLs, LPGs, gasoline, jet fuels and lubricating oils.

When measuring liquids with density correction, density at flowing conditions is required. This value may be provided directly as a process input, or the module can calculate a density from the frequency provided by a densitometer device.

3.5.1 To use a densitometer

Follow the steps below to use a densitometer.

- 1 Configure it, entering all configuration parameters directly from the calibration data sheet supplied by the densitometer manufacturer.
- 2 Supply the frequency output from the densitometer in Hz as a floating-point value in the "Flowing density" process-input location over the backplane (refer to the Backplane Communication section for your platform in the MVI56-AFC manual to determine the correct location). The AFC then calculates a flowing density value, which is then validated by the range check mandated by the "Density" values of "Process Input Scaling" of the meter configuration. The "Scaling" sub-selection is not used against the frequency input, however; the frequency is always input as floating-point.

Note: If using the Densitometer feature, select the Density Process Input Scaling for 4 to 20mA and enter the densitometer frequency as a floating-point value.

3.5.2 Module Configuration

3.5.3 Density Units

The liquid density units can be expressed as:

- Density is in kg/m^3
- Relative density 60°F/60°F
- API gravity

3.5.4 Measuring Water Diluent

For liquid measurement applications, the optional automatic calculation of Net Oil Volume and mass based on the Sediment and Water (S&W) percent input is supported. Only provide the S&W percent value in the specified controller register. The module puts the gross standard (or gross clean oil), net oil and water accumulations in separate accumulators. Refer to Net Accumulator Calculation (page 89).

3.6 General Features

3.6.1 Process Variable Interface

Process variables for each of the meter runs must be produced by the controller for consumption by the AFC module. A versatile architecture for backplane transfer of process variables and other data and signals allow you to easily implement the data transfer. The sample ladder logic automatically transfers the process variables to the module and reads the calculation results to the processor.

3.6.2 Meter Scan Time

For good measurement, the process I/O must be sampled, and the flow calculations completed quickly in order to avoid losing process information and measurement accuracy. The process I/O scan time for the module is under one second for all meter runs.

Note: This is time-dependent on design of the ladder logic implemented to support the two-way data transfer between the AFC module and the controller. The meter calculation scan independent of the process I/O scan may take longer.

3.6.3 Multiple Meter Accumulators

Each meter channel supports the following set of full 32-bit accumulators that may be configured in binary or split decimal format with user-defined rollover values:

- Gross Volume
- Gross Standard Volume (liquid only)
- Net Volume
- Mass
- Water (liquid only)
- Energy (gas only)

Access to the above accumulators is available directly from the Modbus Slave communications ports.

3.6.4 Product Batching

Any or all of the available meter runs may be configured for field installation that requires shipping and/or receiving product batches of predetermined size. The configuration utility option of selecting resettable accumulators provides a simple way to use the power of ladder logic to design product batching, monitoring, and control tailored to suit specific field requirements.

The Meter Signals feature can be used to create an archive or reset an accumulator after the batch is concluded. Refer to the Ladder Logic section for your platform for more information on using this feature.

3.6.5 Data Archiving

The module supports the archiving of data for each meter channel. Each time, one record consisting of all the associated data is date and time stamped and archived. This option allows for archiving each hour for 2 days (48 records per meter run) and every day for 35 days (35 daily records per meter run) for each meter channel. Each record consists of up to 40 process and other variables. Archives are mapped to the local Modbus Table. Refer to Archives (page 91) for more information about this topic.

3.6.6 Event Log Function

The module can log up to 1999 critical events in an Event Log File stored as a set of easily accessible Modbus registers in non-volatile RAM. Changing critical parameters, such as orifice plate size, Meter Base K factors, and Meter Correction Factors, are time stamped and logged. Refer to Events for more information about this topic.

3.6.7 Measurement Units

This option is provided for each meter channel to be configured with SI or US units of measurement. Units for flow totalization (*volumetric* and *mass*) and flow rate monitoring are configurable for each meter channel separately if the default configuration is not applicable. Each meter channel may be configured to use any of the standard units from liters/gallons to thousand cubic meters/barrels. The flow rate period of each meter channel may be selected from flow rate per second, per minute, per hour, or per day.

3.6.8 Process Input Scaling

The module allows you to either pre-scale the process inputs via ladder logic for use in the measurement calculations, or provide unscaled values from the analog input modules directly. In the second case, the scaling is done internally. You can directly enter the zero-scale, the full-scale, and the default values for each of the process variable inputs through the configuration window. Pre-scaled values may be transferred as floating point or as scaled integer.

When selecting scaled integer, the integer values are scaled as follows:

Scaled Integer		
Variable	Format	Example
Temperature	Two decimal places implied	A value of 1342 would be equivalent to 13.42°C
Pressure	No decimal places implied for SI units (kPa) and one decimal place implied for U.S. units (psi).	A value of 200 would be equivalent to 200kPag
Differential Pressure	Two decimal places implied for inches of H ₂ O and 3 places for kPa	A value of 35142 would be equivalent to 35.142kPa
Density (kg/m ³)	One implied decimal place	A value of 5137 would be equivalent to 513.7 kg/m ³
Density (Relative Density)	Four implied decimal places	A value of 10023 would be equivalent to 1.0023 60F/60F.
Density (API)	Two implied decimal places	A value of 8045 would be equivalent to 80.45 °API.

When selecting the 4 to 20mA process input scaling, the module uses the following ranges:

4 to 20mA			
Processor	Module	0%	100%
SLC	MVI46-AFC	3277	16384
ControlLogix	MVI56-AFC	13107	65535
CompactLogix	MVI69-AFC	6241	31206
PLC	MVI71-AFC	819	4095
Quantum	PTQ-AFC	4000	20000

The module uses the configured values for zero and full scale to interpret the process input scaling.

In the **Meter Monitor** window, the raw values as transferred from the processor are shown at the "Last Raw" column and the converted values are shown at the "Scaled Avg" column.

4 Meter Proving

In This Chapter

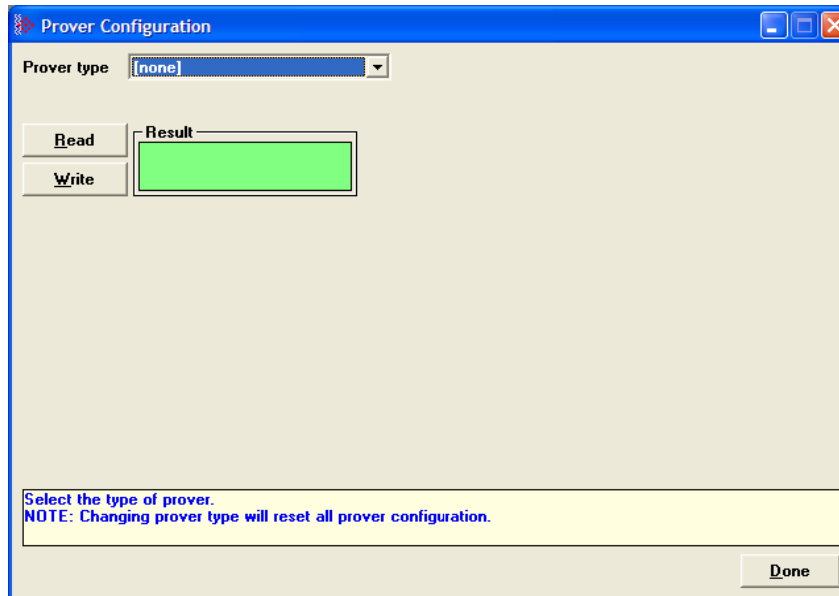
❖ Prover Configuration	42
❖ Setting up the AFC module for Meter Proving	51
❖ Meter Proving Reports.....	63
❖ Protected Meter Proving Data in the AFC's Input Register Bank	64

As meters continue to be used over time, the meter's measurement accuracy deteriorates. Many things can cause the flow sensor bearings to wear down beyond specified limits so that meters are measuring lower volume levels causing producers to pump more oil than the consumer is buying. Meter Provers have a "Known Traceable Volume" which allows using actual flowing and operating conditions to establish a meter correction factor to restore measurement accuracy.

There are 4 types of provers. This chapter will give a basic overview for each type, its options, and configuration.

- The Unidirectional Pipe Prover
- The Bidirectional Pipe Prover
- The Compact Prover
- The Master Meter

4.1 Prover Configuration



Prover type is a parameter that identifies the basic type of the prover. It's values are:

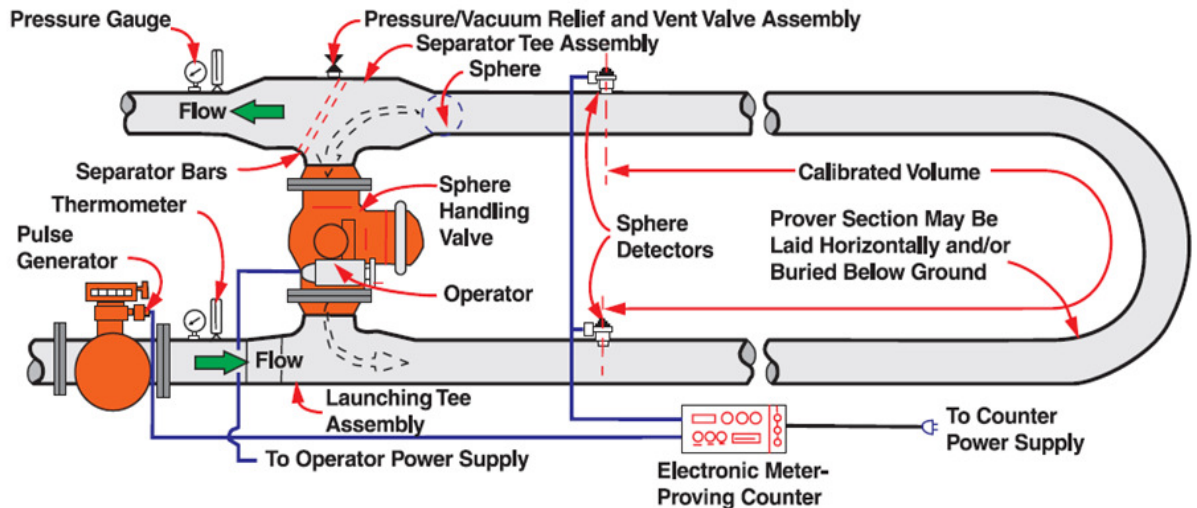
- **NO PROVER CONFIGURED**
- **UNIDIRECTIONAL PIPE PROVER** (You may also choose this selection for an atmospheric tank prover.)
- **BIDIRECTIONAL PIPE PROVER**
- **COMPACT (SHORT, SMALL VOLUME) PROVER**
- **MASTER METER**

4.1.1 Prover Type

Prover characteristics and configurations will vary based on the type of prover and options you select. The following topics describe each type of prover.

Unidirectional Pipe Prover

This is a long pipe, with a ball or piston that fills the pipe and moves with the fluid flow. At each end of the pipe is a switch that is tripped when the ball passes it. A proving run counts the pulses occurring between the switch trips. A run is prepared by positioning the ball in a *cul-de-sac* upstream of the first switch, ready to be injected into the stream. At the end of the run, the ball is extracted from the stream and returned via another path to the upstream end. In order to calculate a meter factor with sufficient precision, the prover volume must be large enough to count sufficient pulses. Therefore, unidirectional provers can be quite large.



Prover Configuration

Prover type: **Unidirectional pipe, or tank** System units: US SI

Prover tag: **Prover** Density units: kg/m3 Rd/60 *API

Read **Result** **Identification** **Options**

Write **Process Input** **Variation Limits**

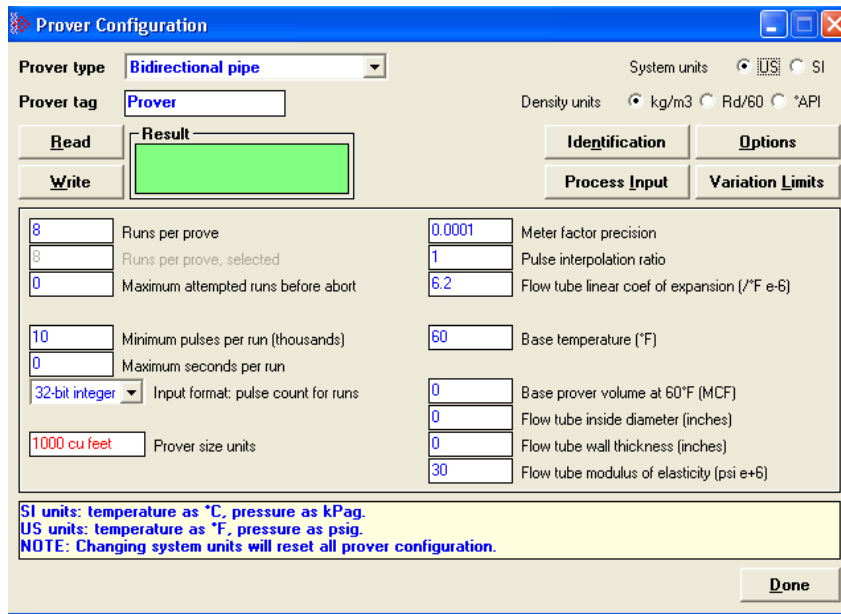
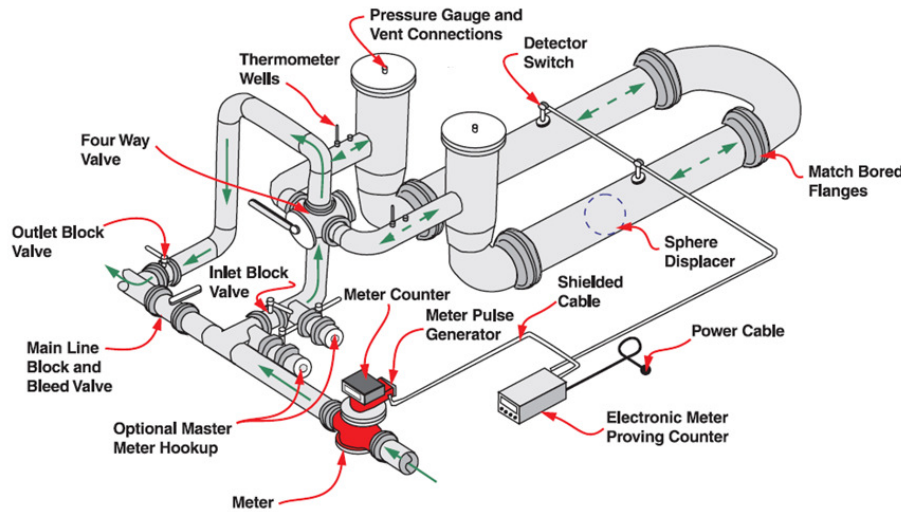
<input type="text" value="8"/>	Runs per prove	<input type="text" value="0.0001"/>	Meter factor precision
<input type="text" value="8"/>	Runs per prove, selected	<input type="text" value="1"/>	Pulse interpolation ratio
<input type="text" value="0"/>	Maximum attempted runs before abort	<input type="text" value="11.16"/>	Flow tube linear coef of expansion (1/C e-6)
<input type="text" value="10"/>	Minimum pulses per run (thousands)	<input type="text" value="15"/>	Base temperature (°C)
<input type="text" value="0"/>	Maximum seconds per run	<input type="text" value="0"/>	Base prover volume at 15°C (m3)
<input type="text" value="32-bit integer"/>	Input format: pulse count for runs	<input type="text" value="0"/>	Flow tube inside diameter (mm)
<input type="text" value="cubic metres"/>	Prover size units	<input type="text" value="0"/>	Flow tube wall thickness (mm)
		<input type="text" value="206.8"/>	Flow tube modulus of elasticity (kPa e+6)

[Read the prover configuration from the Module.](#)

Done

Bidirectional Pipe Prover

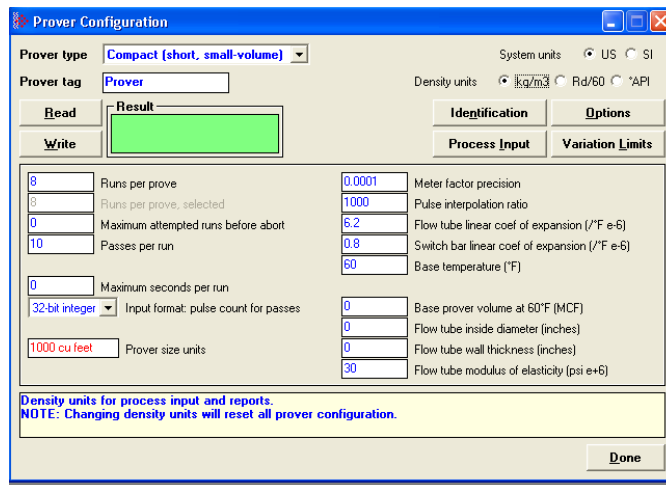
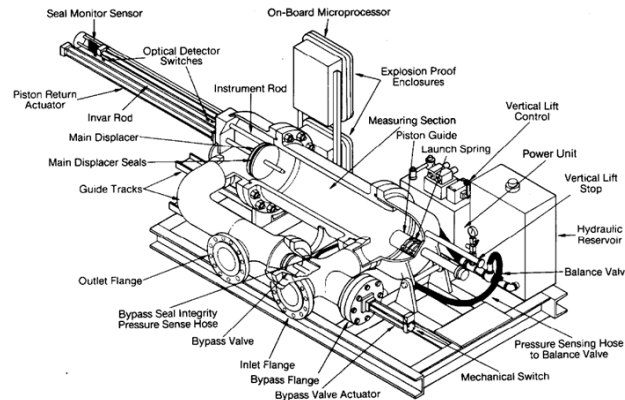
This is similar to a unidirectional prover, except that use is made of the *deadhead* transfer of the ball back to its starting point. Instead of returning the ball via a separate path, valves are swung to reverse the direction of flow in the prover and the ball is returned along its original path to trip the switches a second time in the opposite order. The first pass of the ball is called the *forward leg* and the second is called the *backward* or *return leg*. The pulse count for the run is then the sum of the counts for the two legs. Because the run's pulse count arises from two passes between the switches, a bidirectional prover need be only half the volume of its unidirectional counterpart and can be correspondingly smaller.



Compact (short, small volume) Prover

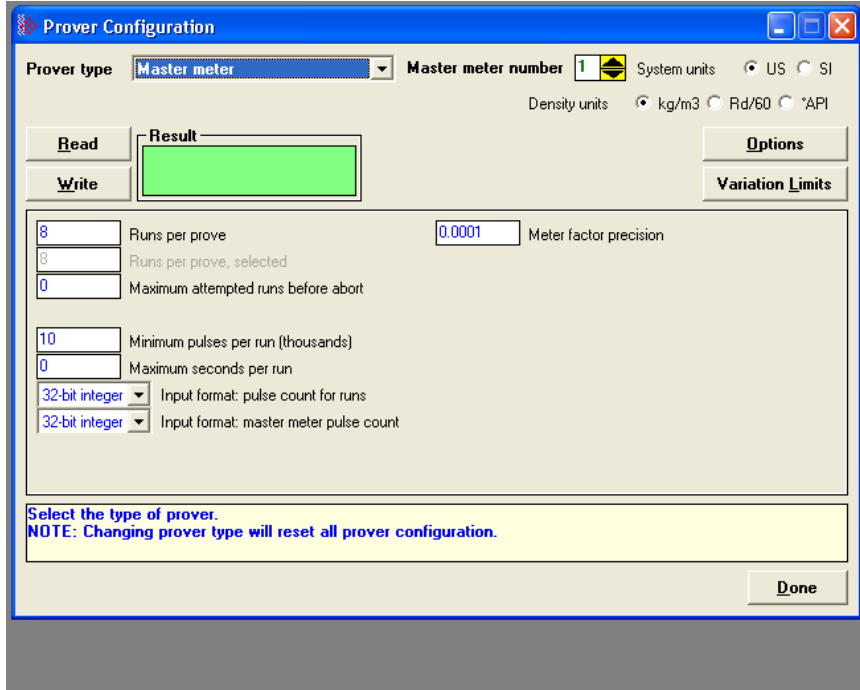
Attention: While the MVI56-AFC module provides the capability to do small volume proving, an I/O platform for ControlLogix that provides Double Chronometry pulse input signals does not exist at this time. These pulse signals are needed for accuracy that is recommended per API standards. Use of this module with small volume provers is not recommended as it may provide inaccurate results.

A compact prover, or small volume prover (SVP), has a short barrel or tube with a piston that travels the length of the tube. The piston has a valve that is opened to allow it to return to its starting point without stopping the flow in the tube. Most SVPs do not mount the switches to be tripped inside the tube. They mount the switches externally on a bar that moves with the piston outside the tube and the switches trip when they move past a fixed point. Each forth and back passage is called a *pass*. SVPs can be much less expensive than LVPs, so they are often preferred. Due to their small size they can collect at most a few hundred pulses during a pass. The number of pulses in a single pass is a number too small for calculating a meter factor with sufficient precision. The technique of double chronometry is then used to determine a fractional pulse count of sufficient precision. Even though a single pass in a SVP with double chronometry can yield a pulse count similar in precision to that from a single run of a LVP, it is often the practice to accumulate several passes into a single run so that the pulses totalized for all passes of the run yield a number large enough for calculating the required meter factor with sufficiently high precision.



Master Meter

This proving technique proves a meter by comparing its behavior to that of another *master* meter whose behavior is deemed to be accurate. A master meter itself must be proved to a high precision by using a conventional prover.



4.1.2 Prover Options

There are several options affecting the handling and representation of data, as well as affecting the relevance and availability of some configuration items. Not all options are available for all prover types. If an option does not apply to a particular prover type, it cannot be selected. For a description of each option listed below see the corresponding Modbus dictionary address in parenthesis below.

- Dual transmitters, temperature (65011.0)
- Dual transmitters, pressure (65011.1)
- Input meter density (65011.2)
- Return leg pulse count is round –trip count (65011.4)
- Prover is double-walled (65011.5)
- External switch bar (65011.6)
- Calculation method: Average Meter Factor (else Average Data) (65011.8)

4.1.3 Run Counts

Runs per prove (65012)

The total number of completed runs that constitute a single prove. This value must be at least 2 and must not exceed 8. If *Maximum attempted runs before abort* (register 65014) is non-zero, this value must not exceed that value.

Runs per prove, selected

The total number of completed runs to be selected for contribution to the prove calculations. This value must be at least 2 and not exceed *Runs per prove*, (register 65012). This value is automatically updated when you edit the *Runs per prove* field.

Maximum attempted runs before abort (65014)

The total number of runs to be attempted before abandoning a prove as incomplete, which permits an automatic proving sequence to automatically terminate itself under exceptionally variable conditions. If this value is zero, no limit is imposed. Otherwise, the value must be at least as large as *Runs per prove*, (register 65012) and must not exceed 65535.

4.1.4 Run Input Setup

Minimum pulses per run (thousands) (65016)

The minimum number of pulses required for a run to be considered for contribution to the prove, represented in thousands. This value must lie between 10 (representing 10,000 pulses) and 1000 (representing 1,000,000 pulses). Runs counting 10,000 pulses or more have sufficient precision to enable calculation of 4-digit meter factors. For all prover types except compact SVPs, the AFC rejects any LVP run that does not meet this condition. Since SVPs can deliver fractional pulse counts that provide sufficient precision with only a small number of pulses, the AFC does not impose this limitation on prover calculation using SVPs.

Maximum seconds per run (65017)

This parameter is a timeout for the duration of a run. A timer is started when the run is started, and if the timer value equals or exceeds this value before the run is completed, then the AFC automatically cancels the run. This allows an automatic prove to recover from conditions that put the AFC and the proving hardware out of step, such as a missed switch signal. This value must lie between 0 and 10000, where zero means that no timeout is imposed.

Input format: line meter pulse count (65020)

This parameter is a code that specifies the format in which pulse counts for the line meter are delivered to the AFC at the ends of runs or passes. These values are:

Value	Format	Description
0	None	No pulse counts are delivered. Used only when no prover is configured
1	32-bit	Pulse counts are delivered as 32-bit (double) integers
2	Split-double	Pulse counts are delivered as split-double values, in which the actual value is (MSW * 10,000 + LSW)
3	Floating point	Pulse counts are delivered as IEEE 32-bit floating point values

When a prover is configured, the default setting is 1 (32-bit), except for compact provers, for which it is 3 (floating point).

Input format: master meter pulse count (65021)

This parameter is a code that specifies the format in which pulse counts for the master meter are delivered to the AFC at the ends of runs or passes. These values are:

Value	Format	Description
0	None	No pulse counts are delivered. Used when the prover is not a master meter.
1	32-bit	Pulse counts are delivered as 32-bit (double) integers.
2	Split-double	Pulse counts are delivered as split-double values, in which the actual value is (MSW * 10,000 + LSW).
3	Floating point	Pulse counts are delivered as IEEE 32-bit floating point values.

When a master meter is configured, the default setting is 1 (32-bit). This parameter is meaningful only when using master meter provers.

4.1.5 Prover Characteristics

Prover Characteristics will vary based on the type of prover and options you select. The following topics describe each field and its operating range.

Prover size units (65018.L)

This parameter sets the units in which the prover's base volume is represented. This parameter is not meaningful for master meter provers.

Meter factor precision (65028+)

This parameter is a number between 0.00000001 and 0.0001. The default setting is 0.0001

Pulse interpolation ratio (65030+)

Meter-proving pulse counts delivered to the AFC may be fractional, such as when double chronometry is used with a SVP. This value is the number of delivered counts that constitute a single actual pulse, so that the actual pulse count is determined by dividing the delivered count by this. The default value is 1000.0 for compact provers and 1.0 for other types. This parameter is meaningful only for non-master meter provers.

Flow tube linear coefficient of thermal expansion (65032+)

Holds the coefficient of thermal expansion of the prover barrel material, meaningful only for non-master-meter provers. Here are some typical materials and their expansion coefficients.

▪ Stainless steel 304 or 316	9.3e-6/°F	16.7e-6/°C
▪ Monel	7.9e-6/°F	14.3e-6/°C
▪ Carbon steel	6.2e-6/°F	11.2e-6/°C
▪ Invar	.8e-6/°F	1.4e-6/°C

The default value is that of carbon steel, 6.2e-6/°F 11.2e-6/°C.

Switch bar linear coefficient of thermal expansion (65034+)

Holds the coefficient of thermal expansion of the external switch bar material, meaningful only for non-master-meter provers with option *External switch bar* (register 65011 bit 6) set. Here are some typical materials and their expansion coefficients.

▪ Stainless steel 304 or 316	9.3e-6/°F	16.7e-6/°C
▪ Monel	7.9e-6/°F	14.3e-6/°C
▪ Carbon steel	6.2e-6/°F	11.2e-6/°C
▪ Invar	.8e-6/°F	1.4e-6/°C

The default value is that of invar .8e-6/°F 1.4e-6/°C.

Base prover volume (65036+)

Holds the base volume of the prover barrel as determined by the water-draw method, in the units specified by *Prover size units* (register 65018.L). This parameter is meaningful only for non-master meter provers.

The accepted standards mandate that the base volume of a bidirectional prover be that registered by a round trip of the displacer.

Flow tube inside diameter (mm) (65038+)

This parameter is the measured inside diameter of the prover barrel at standard (base) conditions and is meaningful only for non-master meter provers with the option *Prover is double-walled* (register 65011 bit 5) clear.

Flow tube wall thickness (mm) (65040+)

This parameter is the measured thickness of the prover barrel wall, and is meaningful only for non-master meter provers with the option *Prover is double-walled* (register 65011 bit 5) clear.

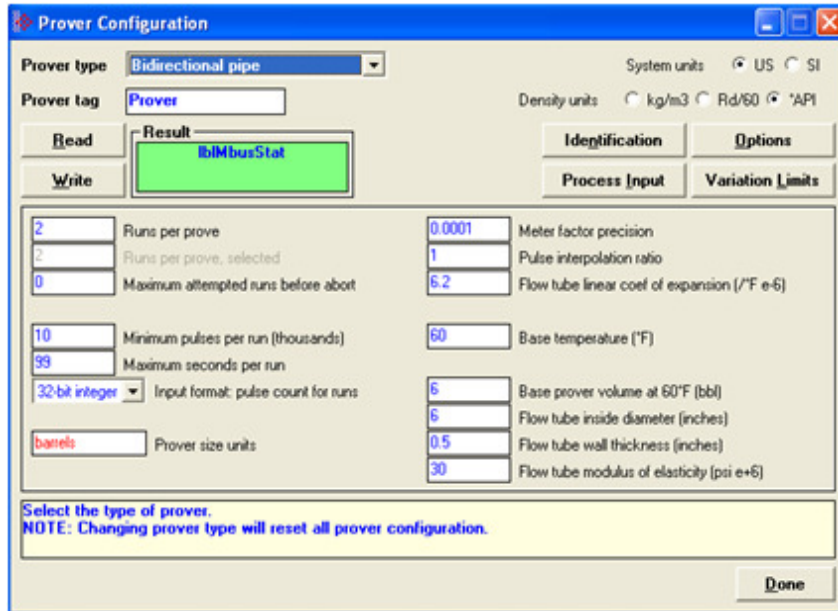
Flow tube modulus of elasticity (65042+)

This parameter is the prover barrel material modulus of elasticity, and is meaningful only for non-master meter provers with the option *Prover is double-walled* (register 65011 bit 5) clear. The default value is that of carbon steel, 206.8e+6 kPa or 30.00e+6 psi.

4.2 Setting up the AFC module for Meter Proving

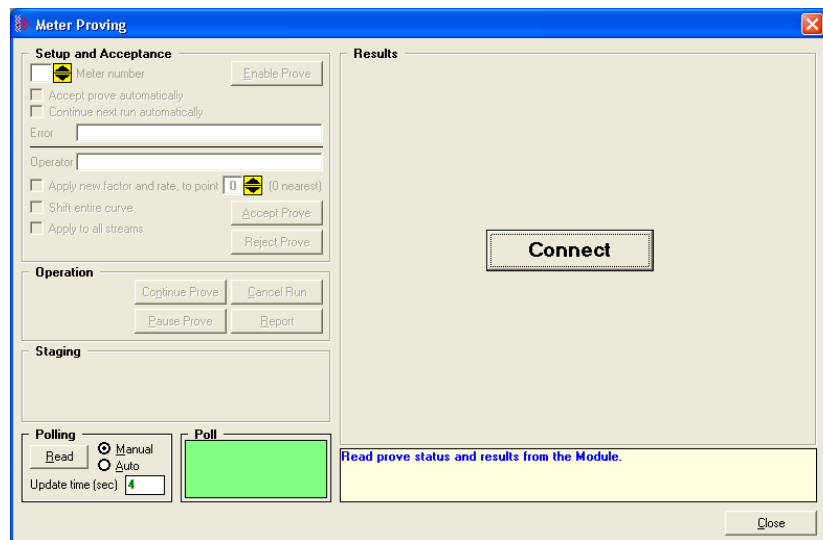
First, configure the parameters in the *Prover Configuration* dialog box. A Bidirectional Pipe Prover is shown in this example.

Note: Changing prover type will reset all prover configuration

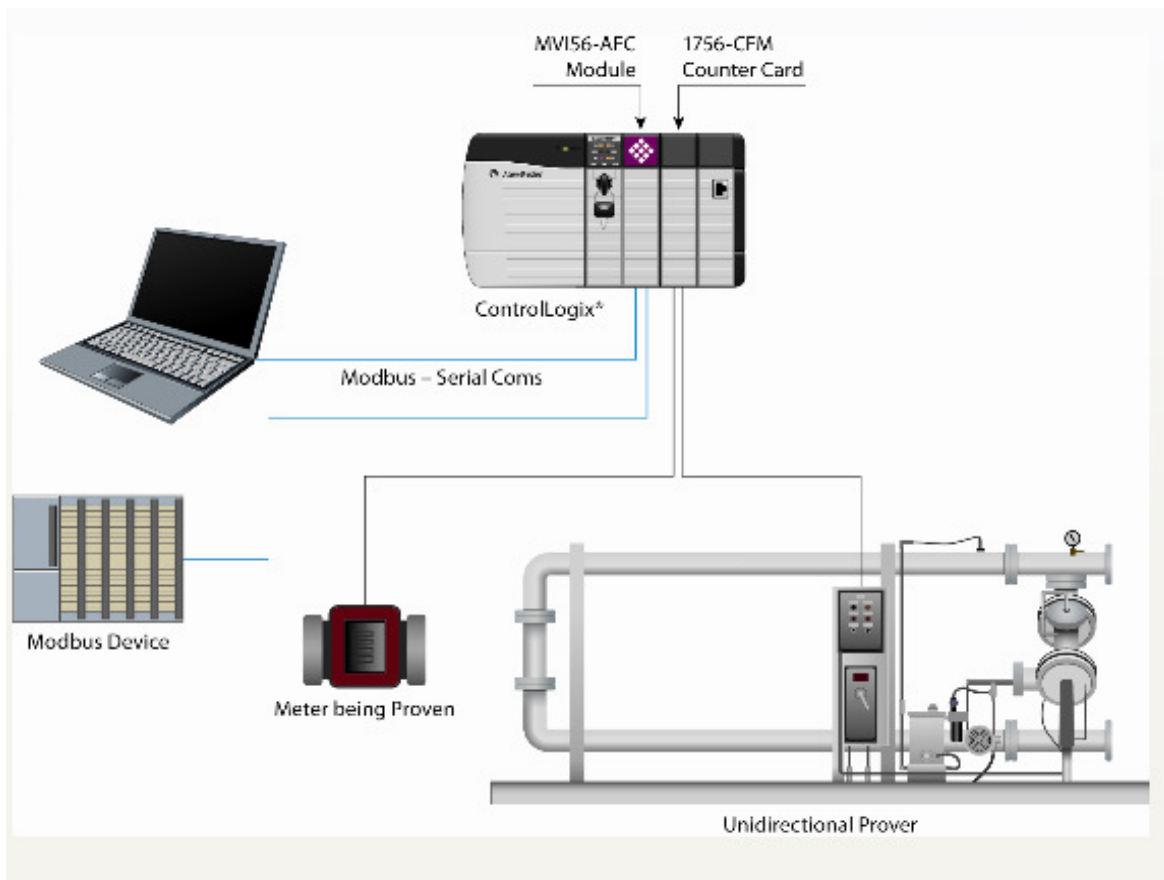


Meter Proving dialog box

This window is used to connect to the module to manage the prove and/or monitor prove status and results from the Modbus database.

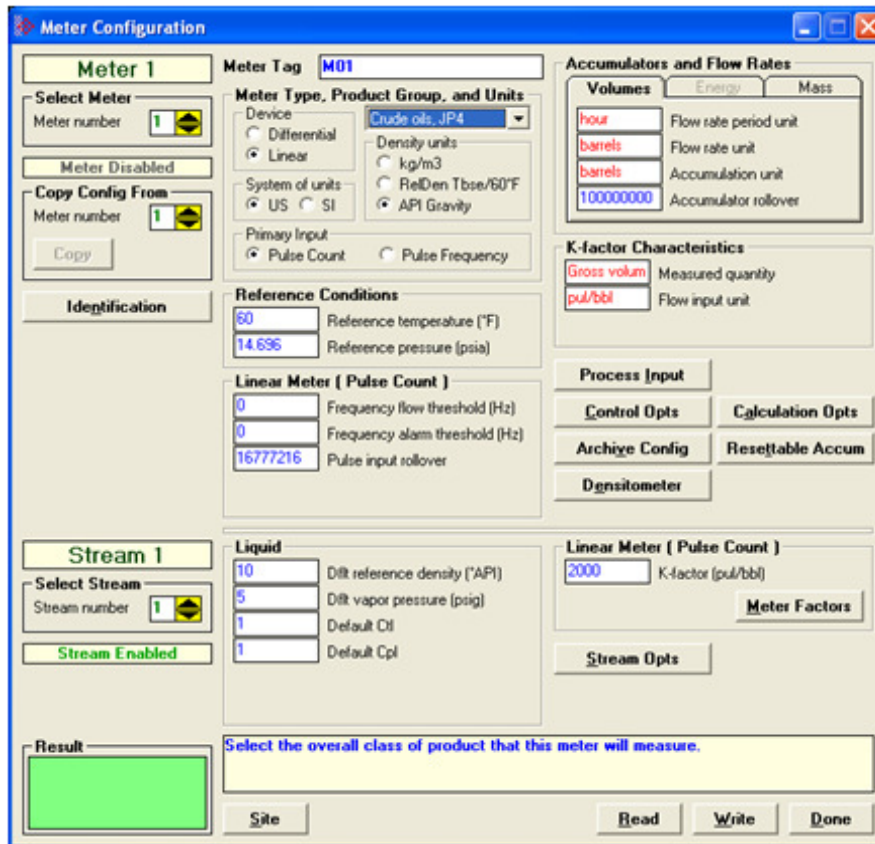


This is a typical configuration for a meter proving setup. Your application may vary from the example shown.

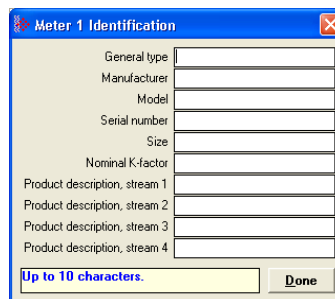


4.2.1 Initial Requirements

In its current version, the AFC supports proving of only liquid products, measured with linear devices that use pulse counts as the primary input variable, where each pulse represents a specific liquid volume.



In the *Meter Configuration* dialog box above, Meter 1 is used in this example as the meter selected to be proved. It can be proved using any one of the four provers that the AFC supports. These provers are described in the *Prover Configuration* section. There is an Identification button which opens an editable options window, shown below. Text entered here appears on the proving report.



4.2.2 Meter Proving Alarms

These alarms are transient and any one might exist only for a single scan, so they might be missed when viewing this register directly. However, alarms are also accumulated into the results database, so alarms that have occurred during any run may be viewed by inspecting that database.

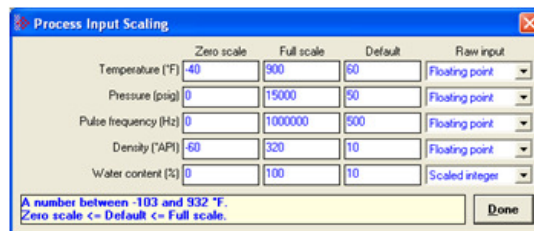
To Check for Alarms

- 1 Activate *Meter Monitor* dialog box
- 2 Select **METER** to be proved
- 3 Click on the **[READ]** button

Note: Verify that the meter is not generating any alarms. Meter proving cannot proceed while any alarm is displayed.



This is accomplished by providing **PROCESS PARAMETER** values that are within the range of the *Process Input Scaling* Dialog box.



There are two sources of *alarms*:

- 1 From the meter, which occur whether or not a prove is in progress. These are illustrated above.
- 2 From the prove, and there are 2 kinds:
 - a) Variation Limit Alarms
 - b) Prove Calculation Alarms

Note: Any alarm will always make a run not able to be selected.

Variation Limit Alarms

These alarms are due to variation outside the configured limits:

Bit/Byte	Description	Modbus Dictionary Address
01	Prover inlet temperature	65050
02	Prover outlet temperature	65052
03	Prover inlet-outlet temperature	65054
04	Prover temperature	65056
05	Prover-meter temperature	65058
06	Switch bar temperature	65060
07	Meter pressure	65062
08	Prover inlet pressure	65064
09	Prover outlet pressure	65066
10	Prover inlet-outlet pressure	65068
11	Prover pressure	65070
12	Prover-meter pressure	65072
13	Meter density	65074
14	Prover density	65076
15	Prover-meter density	65078
16	Water content	65080
17	Meter flow rate	65082
18	Prover flow rate	65084
19	Pulses over runs	65086
20	Pulses over passes	65088
21	Not enough pulses in run	N/A
22-31	[Reserved]	N/A

Prove Calculation Alarms

These alarms arise from prove calculations (e.g. outside API limits):

Bit/Byte	Description
00	[Reserved]
01	CTS prover
02	CPS prover
03	[Reserved]
04	High water
05	CTW
06	CPW
07	Density correction
08	CTL prover
09	CPL prover
10	CSW prover
11	Vapor pressure prover
12	CTL meter
13	CPL meter
14	CSW meter
15	Vapor pressure meter
16	Repeatability
17	Change in factor
18-22	[Reserved]
23	Divide by zero
24-31	[Reserved]

4.2.3 Prover Operation (How to do a Prove)

You must first configure a prover, and configure the channel of a Configurable Flow Meter (CFM) or High Speed Counter (HSC) module for proving.

Note: CFM modules are available for the 1756 platform from Rockwell Automation, and the Quantum platform via Spectrum. Any HSC card will work for the other modules, but if you use an HSC, you will need extra ladder logic in the PLC to implement proving.

Once the parameters for the proving session have been configured, (pipe diameter, water-draw volume, wall thickness, tolerances and limits on the variation of temperature, flow rate, and other process variables), and the prove setup has been completed, the entire proving session can be completely automated within the PLC ladder logic.

Steps for proving a meter

- a** Enter the prover parameters and variation limits (configuration)
- b** Enter the number of the meter to be proved (setup)
- c** Set the *enable prove* signal bit. This function verifies that the selected meter is provable (a liquid pulse meter), and clears the proving results for a new proving session.
- d** Enable the counter card channel for proving, and launch the ball. When the first switch is tripped, set the *run start* signal bit. During the run, continuously copy the prover temperature, pressure, density, etc, to the AFC, so that it may monitor their variation and accumulate them for final averaging. For the same purpose, the AFC module itself retrieves meter process variables directly from the meter input from the PLC without PLC intervention.
- e** When the second switch is tripped, copy the final pulse count from the counter card channel to the proper location and set the *run stop* signal bit. This function computes results for the completed run (averages of process variables, variation limit alarms, etc.), and also computes results for the entire prove over all completed runs (averages of run averages, variation limit alarms, API calculations and calculation alarms, final meter factor and change in meter factor, and number of completed runs). Upon a *run start* or *accept prove* signal, any bad runs are deleted from the prove before continuing with the remainder of the signaled function.
- f** When a sufficient number of runs have been completed, set either the *accept prove* or the *reject prove* signal, which function marks the data in the prover results accordingly.

Missed Switch

It is possible that the tripping of the second switch to end a run is not seen by the PLC (due to a broken wire or poorly lubricated switch), leaving the AFC and the physical prover in inconsistent states. You may recover from this condition with the *Run Cancel* signal, which clears any active run and resets the AFC to be ready to start a new run. Data from any bad run will also be deleted by the *Run Cancel*.

Proving Controls

These bits supply parameter information to the *Enable prove* and *Accept prove* signals (register 65308 bits 1 and 2 respectively). Control bits 0 through 7 parameterize the *Enable* and bits 8 through 15 parameterize the *Accept*. Controls are latched into the results database upon receipt of a signal. Changes thereafter have no effect on the state of these control bits.

Proving Signals

A prover signal instructs the AFC to immediately perform a particular function once. A signal bit is latched by the process issuing the signal (for example, the PLC) and is unlatched by the AFC when the function has been performed. Prover signals are discharged upon the next proving scan, before which several Modbus transactions may be completed. Modbus transactions to read the status of these signal bits may, therefore, show uncleared bits for functions that have already been scheduled but not discharged yet.

Prover Sequencing

This parameter reports the state of the proving hardware, making it available to the prove-management software for display of prove status and possible control of the prove. The prove-management feature of AFC Manager uses it only for display. This value usually comes from the proving hardware integrated into the PLC platform, therefore it is normally supplied by the PLC.

Prover Phase

These bits report the state of the run as known by the proving hardware. These values are chosen specifically for compatibility with several kinds of proving hardware, so that the work necessary for the PLC to translate hardware register values into these values required by the AFC is minimized and in many cases can be reduced to a simple mask-and-copy. There are 8 values ranging from 0-7. These values are:

Value	Name	Description
0	Prover not selected (not ready)	This is the normal value when no proving run is in progress.
1	Prover active, not yet counting	The counter card has been initialized for a proving run, but the ball or piston has not yet passed the first switch. Counting of the pulses for the run has not yet begun.
2	Prover active, past first switch and counting	The ball or piston has passed the first switch but not yet passed the second switch, and the run counter is counting pulses. For bidirectional provers, this is the forward leg.
3	Prover active, past second switch	This state is for bidirectional provers only. The ball or piston has passed the second switch of the forward leg, the run counter has been stopped, and the intermediate count for the forward leg is available. During this state the proving hardware should be swinging valves to reverse the stream's direction of flow through the prover, preparing it for the return leg.
4	Prover active, past first switch return leg	This state is for bidirectional provers only. The ball or piston has passed the first switch on the return leg but not yet passed the second switch, and the run counter is counting pulses.
5	Run Complete	The ball or piston has passed the second switch (for bidirectional provers, the second switch of the return leg), the run counter has been stopped, and the count for the run is available. For a bidirectional prover, this count may be either the count for only the return leg or the count for the entire run; use prover option "Return leg pulse count is round-trip count" (register 65011 bit 4) to specify which.
6	Prover not selected (not ready)	Some kinds of proving hardware report this value for a counting mode unrelated to proving. The AFC treats this value the same as value 0.
7	Prover not selected (not ready)	Some kinds of proving hardware report this value for a counting mode unrelated to proving. The AFC treats this value the same as value 0.

Prover Position: Ready for Launch

The prover's ball or piston is ready for launching into the stream. For a bidirectional prover, this is the launch of the forward leg.

Prover Position: Ready for Return

For bidirectional provers only, the prover's ball or piston is ready for launching into the stream for the return leg.

Prover Position: Valve Sealed Behind Ball

The prover's ball or piston has been launched into the stream and the sealing valve has been closed behind it. For a bidirectional prover, this is the start of the forward leg.

Prover Position: Valve Sealed Behind Ball, Return Leg

For bidirectional provers only, the prover's ball or piston has been launched into the stream for the return leg and the sealing valve has been closed behind it.

Prover Temperature

Absolute

This value is the process input temperature of the prover (traditional or master meter) in units relative to absolute zero, and is required for some calculations. This value is meaningful only while a prove is active.

Conventional

This value is the process input temperature of the prover (traditional or master meter) in conventional units. For a traditional prover with dual transmitters, this is the average of the two inputs. This value is meaningful only while a prove is active.

Prover Pressure

Absolute

This value is the process input pressure of the prover (traditional or master meter) in absolute units. This value is calculated as (gauge pressure) + (barometric pressure). This value is meaningful only while a prove is active.

Gauge

This value is the process input pressure of the prover (traditional or master meter) in gauge units. For a traditional prover with dual transmitters, this is the average of the two inputs. This value is meaningful only while a prove is active.

Prove-enable Error Code

This code reports the result of the most recent attempt to enable a prove. If the code is zero, the prove was successfully enabled; a non-zero code reports the reason for failure. The values are:

Value	Name	Description
0	The new prove has been enabled	The new prove has been enabled
21	<i>Requested meter number</i>	The <i>Requested meter number</i> (register 65300) is out of range, or, for a master meter prover, is the same as that of the master meter (an attempt to self-prove the master meter)
22	Line meter not liquid pulse	At the present time, the meter to be proved may only be a liquid pulse meter.
23	Incompatible measurement standard	At the present time, the configuration of both the prover and the line meter to be proved must specify the same system of measurement units (US, SI) and the same liquid density units selection (kg/m ³ , Rd/60, °API).
24	Unimplemented product group	Because of the nature of the proving calculations at the present time, not all product groups are provable. Meters configured for these product groups are provable: <ul style="list-style-type: none"> ▪ Liquid (crude oils and JP4) ▪ Liquid (refined products: gasolines, jet fuels, fuel oils, except JP4) ▪ Liquid (NGLs and LPGs) ▪ Liquid (lubricating oils) ▪ Liquid (special applications) Meters configured for these product groups are not provable: <ul style="list-style-type: none"> ▪ Gas ▪ Liquid (oil-water emulsion of crudes) ▪ Liquid (oil-water emulsion of NGLs)
25	Unimplemented measured quantity	At the present time, only pulse meters whose pulse train represents gross volume can be proved.
28	Line meter in calibration	The meter to be proved has at least one process input in calibration mode. Ensure that all process inputs are <i>live</i> before attempting to prove the meter.
29	Line meter not enabled	The meter to be proved is not enabled.
32	Master meter not liquid pulse	At the present time, a master meter prover must be a liquid pulse meter.

33	Master meter incompatible configuration	<p>For a master meter prover, both the line meter and the master meter must be compatibly configured, including identical settings of:</p> <ul style="list-style-type: none"> ▪ System of measurement units (US, SI) ▪ Liquid density units (kg/m³, Rd/60, °API) ▪ Product group ▪ Measured quantity (gross volume pulses) ▪ Reference conditions (base temperature and pressure) ▪ API calculation options (selection of density, temperature, and pressure corrections) ▪ For product group 8, <i>Special applications</i>, the coefficient of thermal expansion <i>Alpha</i>
38	Master meter in calibration	<p>The master meter has at least one process input in calibration mode. Ensure that all process inputs are <i>live</i> before attempting to use the master meter for proving.</p>
39	Master meter not enabled	<p>The master meter is not enabled.</p>
51	Invalid prover parameter	<p>For a traditional (non-master-meter) prover, the base prover volume (register 65036) must be greater than zero, and, if the prover is single-walled, the inside diameter, wall thickness, and modulus of elasticity (registers 65038, 65040, and 65042) must all be greater than zero.</p>
52	Invalid prover controls	<p>Some undefined bits in the <i>at-enable</i> controls (register 65306 bits 0 through 7) have been set.</p>

4.3 Meter Proving Reports

Clicking on the **REPORT** button generates a report with such information as:

- Manufacturer
- Model Number
- Serial Number
- Material Type
- Prover Tag
- Results of the prove will appear in this report, along with the static data entered in the text window during setup. For more information, see Initial Requirements (page 53).

Setup and Acceptance

12 Meter number

Accept prove automatically

Continue next run automatically

Error: _____

Operator: _____

Apply new factor and rate, to point 0 (0 nearest)

Shift entire curve

Apply to all streams

Operation

Staging

Meter flow rate, Gross (MCF/h) **2.1708**

Run 2 Ready Running Complete

Polling

Manual Auto

Update time (sec) 4

Results

Meter 12 Stream 1 MMstrm 1 Runs: 2 completed 2 selected 2 attempted
 Begun 1998-01-02.23:06 Updated 1998-01-02.23:08 Accepted
 Prove **Enabled** Factor application by stream ...

	1: Not applied	2: Not applied	3: Not applied	4: Not applied
Readings				
Number of samples	136	68	68	
Process input alarms	00000000h	00000000h	00000000h	
Readings alarms	00000000h	00000000h	00000000h	
Meter temperature (°F)	60.5	60.4	60.5	
Prover temperature (°F)	60.5	60.5	60.6	
Meter pressure (psi)	45	45	45	
Prover pressure (psi)	45	45	45	
Meter density (kg/m3)	820.1	819.6	820.6	
Water content (%)	0.04	0.04	0.04	
Meter flow rate (MCF/h)	2.10352	2.103674	2.103966	
Prover flow rate (MCF/h)	2.103406	2.102204	2.104608	
Pulse counts				
Pulse count for run	14637.5	14639	14636	
Master meter pulse count	14637.5	14639	14636	
Calculations				

[Show the prove report.](#)

The *Meter Proving* window above shows the system during a prove using a Master Meter. Notice the differences in the example of the information that is available before and after connecting to the module.

4.4 Protected Meter Proving Data in the AFC's Input Register Bank

The data concerned with Meter Proving is maintained in the Input Register Bank, (Modbus 3xxxxx read-only Input Register Addresses), protected from change from outside. There are two areas:

- a Latest Prove Results (3x63400 to 3x63709)
- b Meter Previous Prove Summary (3x61600 to 3x62399, 50 registers per meter)

These two areas are described in better detail in the following two topics.

4.4.1 Latest Prove Results

This area contains complete details of the latest prove that has been enabled, including

- Prove setup
- Prover and proved-meter configuration summary
- Prove state
- Prove-level calculations
- Run-level input and calculations for each run of the prove

This area supplies almost all the information presented on the proving report (the remaining info comes from the proved meter's Previous Prove Summary; see next). The contents of this area persist until a new prove is enabled, so a proving report may be regenerated at any time after the prove has been completed and before the next one is started. There is only one such area for all meters on the AFC module; therefore enabling a new prove for any meter resets the Prove Results from the last completed prove, regardless of which meters were involved.

The Latest Prove Results is a block of 1310 registers, starting at input register 62400 and proceeding through register 63709. The table below explains these sub-areas.

Name	Module Memory Address	Description
Prove Status	62400 to 62409	Occupies 10 registers
Prove Setup	62410 to 62553	Occupies 140 registers and protects meter configuration and prove setup information for use by proving calculations and report generation; this information remains unchanged from the moment of enable, regardless of how the original source information might be altered during or after the prove
Prove Acceptance	62554 to 62575	Occupies 22 registers and records timestamps associated with the prove, accumulator totalizer values, and details of the disposition of the new meter factor upon acceptance of the prove.
Prover Configuration	62576 to 62655	Occupies 80 registers and has the same purpose as Prove Setup, to protect the prover configuration against subsequent changes so that proving can proceed under reliably constant parameters, and so that the proving report can be generated and regenerated according to the original conditions of the prove.
Prove Only Calculations	62656 to 62665	Occupies 10 registers and contains a few calculated values that are applicable only for the prove as a whole.
Reading and Calculations for Prove	62666 to 62781	Occupies 116 registers and the "readings" part contains the averages of the corresponding readings for all runs of the prove. The "calculations" part contains calculations performed upon the prove-level readings if calculation method "average data" was chosen.
Reading and Calculations for Runs	62782 to 63709	Occupies 166 registers for each of up to 8 runs of the prove. The layout of each block of 116 registers is identical to that of the Readings and Calculations for Prove block. The "readings" part contains the weighted averages or snapshots of all process input and counter card input for the duration of the run. The "calculations" part contains calculations performed upon the run-level readings if calculation method "average meter factor" was chosen.

The Latest Prove Results area has a fixed layout so that any point can always be found at the same location regardless of setup, and with a collection of points intended to be sufficient for a variety of setups. Consequently, many points will be irrelevant for a given combination of prover configuration, meter configuration, and prove setup. Those irrelevant points will have zero values in the Results area and can be ignored. AFC Manager's Meter Proving window does not show irrelevant points.

4.4.2 Meter Previous Prove Summary

This area contains summary data for the previous prove of each of the AFC's meter runs. Each time a new prove is enabled and before the Prove Results area is reset, summary prove information for the meter previously proved (if any) is copied to the meter's Previous Prove Summary block, overwriting the old information. This area supplies a small amount of the information presented in the proving report.

The Previous Prove Summary block for each meter occupies 50 registers. Meter #1's block begins at input register 61600, so that Meter #2's block is at 61650, and so on; registers 61600 to 62399 are allocated to the Previous Prove Summary blocks for up to 16 meter runs.

5 Modbus Database

In This Chapter

❖ AFC Modbus Address Space	70
❖ Primary Slave	71
❖ Virtual Slave	74

The module supports two individual Modbus slaves (Primary and Virtual) to optimize the polling of data from the remote SCADA system, or from the processor (through the backplane). Refer to the Modbus Dictionary dialog box in AFC Manager for information about Modbus addressing.

5.1 AFC Modbus Address Space

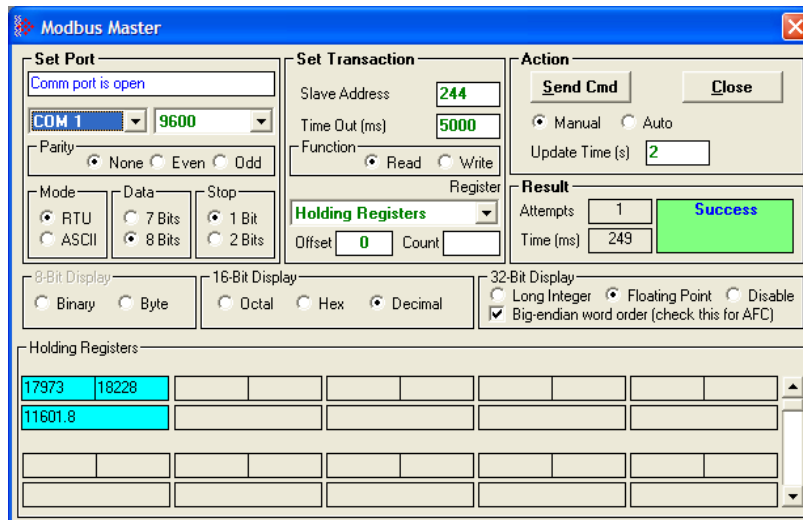
Addressable Modbus registers are divided into four banks as shown in the following table.

MODBUS Address Space Allocation: Total Modbus Registers: 131,072			
Primary Slave Banks (131072 registers)		Virtual Slave Banks (20,000 registers)	
Holding Registers	Input Registers	Holding Registers	Input Registers
From: 0	From: 0	From: 0	From: 0
To: 65535	To: 65535	To: 9999	To: 9999

The first 100 registers of the virtual slave (registers 0 through 99) are predefined to map to the first 100 registers of the primary slave. This mapping cannot be changed. Also, the Virtual Slave Input Registers can be accessed as Virtual Slave Holding Registers by adding 10000 to the Modbus register address; for example, Input Register 2386 is the same as Holding Register 12386.

5.1.1 Accessing the Data

The AFC Manager provides an easy way to read and write data from both slaves through the Modbus Master Interface.



5.2 Primary Slave

The Primary Slave contains the main AFC database that consists of 131,072 Modbus registers. The Site and Meter configuration, as well as all live process data and ongoing calculations are kept in the Primary Slave address space. This address space is divided equally between the Input Register Bank (65,536 registers) and the Holding Register Bank (65,536).

The register addressing is shown in the Modbus Dictionary dialog box in AFC Manager.

5.2.1 Modbus Address References

In these documents (the AFC Manager User's Guide and the User's Guide for your platform) you will occasionally see Modbus address references like *Ph00018* or *Mh00162*. The first two characters of such references indicate how to convert the following number into an absolute Modbus address in the module.

This table shows the possible values for the first identification character:

Address Translation ID	Description
P	Absolute Modbus address, Primary Slave
M	Meter-relative Modbus address, Primary Slave
V	Absolute Modbus address, Virtual Slave

This table shows the possible values for the second identification character:

Register Bank ID	Description
h	Holding register
i	Input register

5.2.2 Modbus Address Examples

Ph02000 = holding register located at address 2000 in the primary slave

Pi02000 = input register located at address 2000 in the primary slave

Mh00100 = Meter-relative holding register located at offset 100 in the block of the primary slave that contains the data for the meter

5.2.3 Meter-relative Data

Meter-relative data starts at absolute holding register address 8000 and occupies 2000 words of data for each meter channel.

Meter 1 Data	8000
Meter 2 Data	10000
Meter 3 Data	12000
Meter 4 Data	14000
Meter 5 Data	16000
Meter 6 Data	18000
Meter 7 Data	20000
Meter 8 Data	22000
	24000

The meter-relative addresses are offsets within each meter data area. The correct absolute address is calculated by the following formula (assumes meters are numbered starting with 1):

$$(\text{absolute address}) = (2000 * (\text{meter number}-1)) + 8000 + (\text{meter relative address})$$

In the Modbus Dictionary dialog box, addresses listed for the selected meter are absolute addresses, so you should subtract the appropriate multiple of 8000 to calculate the meter-relative address.

Example: Find the orifice diameter address for the first 5 meter channels.
 The meter 1 orifice diameter registers are located at the holding register address 8162 and 8163 as follows:

8160	8161	Float	Parameter: orifice plate: measurement temperature
8162	8163	Float	Parameter: orifice plate: measured diameter
8164	8165	Float	Parameter: orifice plate: coef of thermal expansion
8166	8167	Float	Parameter: meter tube: measurement temperature
8168	8169	Float	Parameter: meter tube: measured diameter
8170	8171	Float	Parameter: meter tube: coef of thermal expansion
8172	8173	Float	Parameter: differential pressure flow threshold

The meter-relative addresses are Mh00162 and Mh00163
 The addresses for meters 1 to 5 are listed on the following table.

Meter	Registers
1	8162 and 8163
2	10162 and 10163
3	12162 and 12163
4	14162 and 14163
5	16162 and 16163

5.2.4 Scratchpad

The Primary Modbus Slave contains a scratchpad area that can be used to store any data required by each application. This area is "empty" by default and contains 6000 words of data starting at holding register 2000 in the Primary Modbus Slave.

5.3 Virtual Slave

The module also provides a Virtual Address Space of 20,000 Modbus registers. This address space is divided equally between the Input Register Bank (10,000 registers) and the Holding Register Bank Holding Register Bank (10,000). This is where you can create a virtual re-map by cross-referencing any of the 130,072 Primary Slave Modbus registers to the 20,000 Modbus registers in the Virtual Slave Banks, thereby making it easy for a SCADA Master to poll only the necessary Modbus addresses in contiguous blocks. The virtual slave can also be used for data polling from the processor through the backplane.

Modbus access to the Virtual Modbus Slave is disabled by default since its Modbus address is originally set as 0. To use the Virtual Modbus Slave, you must initially configure a Modbus address greater than zero in order to enable it. Refer to Site Configuration for more information about enabling the Virtual Slave and using the remapping feature. The PLC may always access the Virtual Slave, whether or not it has a non-zero slave address and thus is available via Modbus. A download operation will not transfer the Virtual Slave Remapping configuration. You must click on the **Write** button on the **Indirect Address Remapping** dialog box to transfer the data.

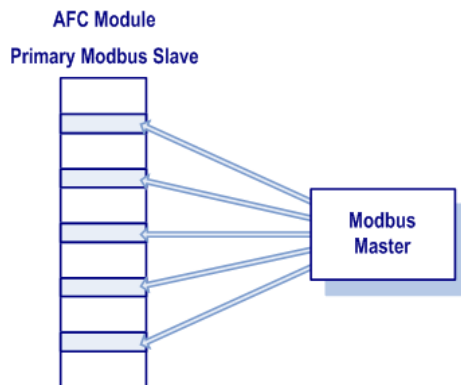
Note: The first 100 registers in the Virtual Slave Holding Register Bank have been pre-assigned and cannot be remapped. They map directly to the first 100 holding registers of the Primary Slave.

5.3.1 Virtual Slave Example Application

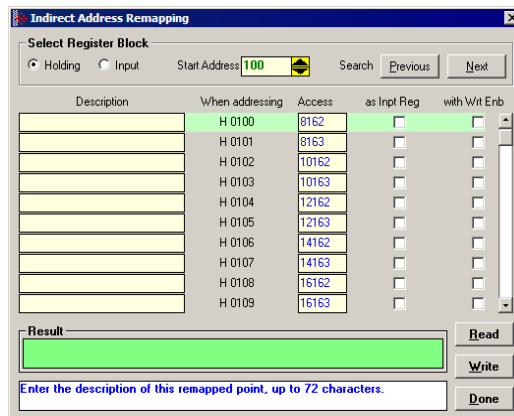
Assume that an application requires a remote Modbus Master to poll the orifice diameters for the first 5 channels. Continuing the previous example, the holding register addresses are listed again the following table.

Meter	Registers
1	8162 and 8163
2	10162 and 10163
3	12162 and 12163
4	14162 and 14163
5	16162 and 16163

Because these addresses are not contiguous, the Modbus Master would have to use five commands to poll all the data directly from the Primary Modbus Slave as follows:



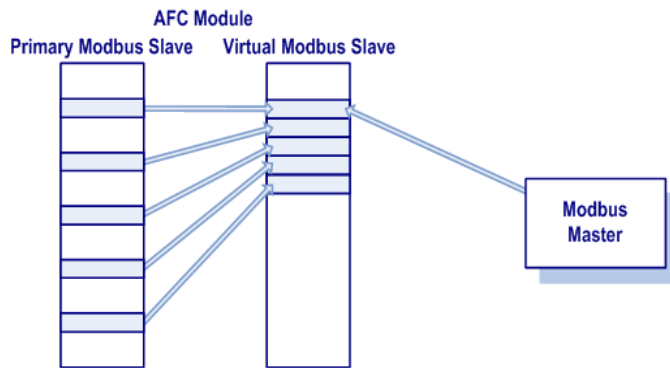
However, using the Virtual Modbus Slave optimizes the polling of data because the registers can be remapped in any order using the AFC Manager (Site Configuration window). The following illustration shows how the orifice diameter registers could be remapped to the Virtual Slave starting at address Vh00100:



The following table shows how the addresses would be remapped between both slaves:

Primary Modbus Slave Addresses	Virtual Modbus Slave Addresses
8162 and 8163	100 and 101
10162 and 10163	102 and 103
12162 and 12163	104 and 105
14162 and 14163	106 and 107
16162 and 16163	108 and 109

Therefore, instead of sending five Modbus commands (2 words each) to the Primary Modbus Slave, the Modbus Master device can now send one single Modbus command (10 words) to the Virtual Modbus Slave in order to poll the same data from the module:



This example demonstrates the benefits of using the Virtual Slave instead of accessing the data directly from the Primary Modbus Slave. The same procedure can be used when polling data from the processor (through the backplane) because the Modbus Gateway block also requires the data to be listed in a contiguous order.

6 Modbus Communication

In This Chapter

- ❖ Communication Parameters 78
- ❖ Port Options 79
- ❖ Modbus Master 80
- ❖ Modbus Pass-Through 82

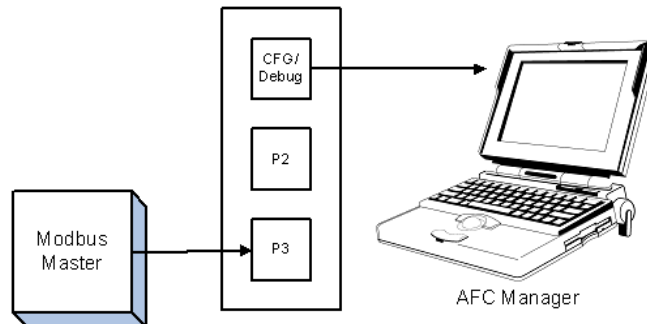
A remote Modbus Master device can be connected to any one of the communication ports for data polling. The module accepts the following Modbus command functions according to the Modbus protocol specification:

Modbus Function Code	Description
3	Read Holding Registers
4	Read Input Registers
6	Preset (Write) Single Register
16	Preset (Write) Multiple Registers

Ports 2 and 3 support RS-232, RS-422, or RS-485 communications. The Configuration/Debug port (Port 1) supports RS-232 only.

Refer to Cable Connections (page 299) for wiring instructions.

The Modbus Master command can be sent to either the Primary or Virtual Modbus Slaves in the module. Each slave has individual Modbus addresses that you can configure (**Project / Site Configuration**). The Primary Slave address is configured as 244 by default.



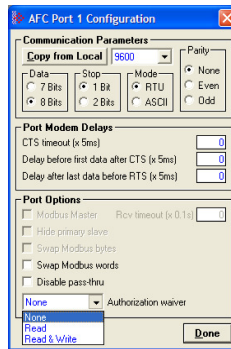
6.1 Communication Parameters

The module supports the following communication parameters for each communication port:

Parameter	Values
Baud Rate	300, 600, 1200, 2400, 4800, 9600 or 19200
Data Bits	7 or 8
Stop Bits	1 or 2 Bits
Mode	RTU or ASCII
Parity	None, Even or Odd

Note: Do not configure a port for both RTU mode and 7 data bits as this combination is not supported by the Modbus protocol.

You must configure the communication parameters for each communication port using the AFC Manager software (Site Configuration):



6.2 Port Options

The following options can be configured:

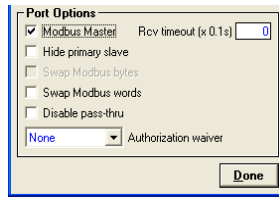
Port Options	Description
Hide Primary Slave	Protects the Primary Slave from any read or write operation from a remote master. Only the virtual slave is visible on this port.
Swap Modbus Bytes	Swap the Modbus bytes transferred through this port (Not implemented)
Swap Modbus Words	Swap the Modbus words transferred through this port. This parameter is only applicable to those data points that hold 32-bit quantities (long integers, floats, totalizers),
Disable Pass-Thru	Disables the pass-thru feature on this port
Modbus Master	Enables the Modbus Master for the port (Port 3 only)
Authorization waiver	Each port can be individually configured to waive the authorization requirement. This feature allows each port to have a different access level.

Not all options are available on every port:

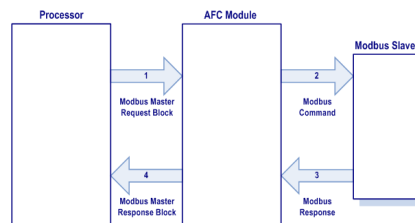
- Port 1 is restricted, so that AFC Manager can always communicate with the Primary Slave using this port.
- Modbus Master option is available only on Port 3.

6.3 Modbus Master

Port 3 can be configured for Modbus Master operation (**Project / Site Configuration / Port 3**).



The Modbus Master command is generated from the processor using ladder logic (Modbus master block). After the Modbus Master transaction is completed the module is ready to receive another Modbus Master request from the ladder logic:



The following Modbus functions are supported for Modbus Master operation:

Modbus Function Code	Description
1	Read Coil Status
2	Read Input Status
3	Read Holding Registers
4	Read Input Registers
15	Force (Write) Multiple Coils
16	Preset (Write) Multiple Registers

The module offers considerable flexibility for Modbus Master operation, allowing the ladder logic to select one of the following data types:

- Bit (packed 16 to a word)
- Word (16-bit register)
- Long (32-bit items as register pairs)
- Long Remote (32-bit items as single registers)

Note: Long data type implements each data unit as one pair of 16-bit registers (words). Each register contains two bytes. Long remote data type implements each data unit as one 32-bit register. Each register contains four bytes. The proper choice depends on the remote slave's Modbus implementation.

6.3.1 Example

The following table shows how the data types are implemented if a **write** function is selected and the item count is configured with a value of 10 (decimal):

Data Type	Register Type	Modbus Function	Number of Coils	Number of Bytes	Number of Registers	Number of words (16-bits) transferred
Bit	Coil	15	10	2	-	1
Word	Holding	16	-	20	10	10
Long	Holding	16	-	40	20	20
Long Remote	Holding	16	-	40	10	20

Note: The number of coils, bytes, and registers are part of the Modbus request (functions 15 and 16) according to the Modbus specification.

The following table shows how the data types are implemented if a **read** function is selected and the item count is configured with a value of 10 (decimal):

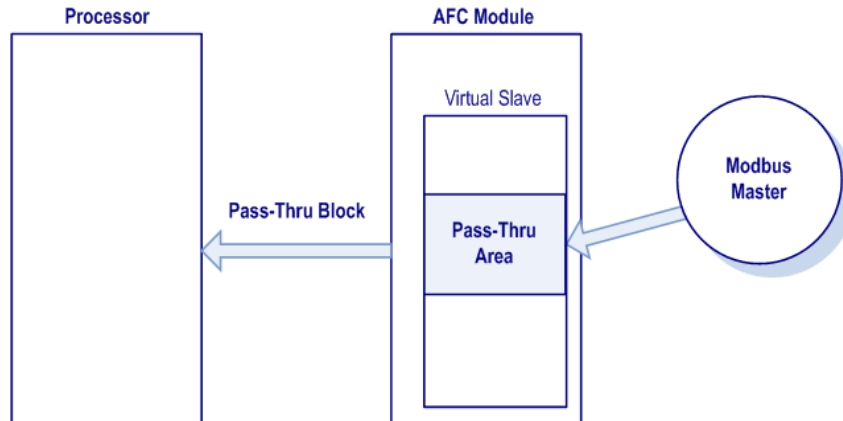
Data Type	Register Type	Modbus Function	Number of Registers
Bit	Coil	1	10
Bit	Input	2	10
Word	Holding	3	10
Word	Input	4	10
Long	Holding	3	20
Long	Input	4	20
Long Remote	Holding	3	10
Long Remote	Input	4	10

Note: The number of registers is part of the Modbus request according to the Modbus specification.

Refer to the ladder logic section for your module for more information about the Modbus Master block.

6.4 Modbus Pass-Through

The Modbus pass-through feature allows you to configure a Modbus pass-through region in the Virtual Slave (**Project / Site Configuration**). After the module receives a holding register write command (Modbus functions 6 or 16) or a bit write command (Modbus functions 5 or 15) to this region, it will generate a pass-through block to be sent to the processor containing the Modbus command data. You may define a word pass-through region (for Modbus functions 6 and 16) and a bit pass-through region (for Modbus functions 5 and 15).



Important: You must enable the virtual slave by configuring a Modbus address greater than 0 (**Project / Site Configuration**).

You can control which communication ports will support the pass-through (**Project / Site Configuration / Port X button**).

This feature requires ladder logic to read the pass-through block from the module to the processor. Refer to the Ladder Logic section for more information about the pass-through feature.

7 Accumulators

In This Chapter

❖ Accumulator Totalizer and Residue.....	84
❖ Accumulator Types.....	85
❖ Net Accumulator Calculation	89
❖ Frequently Asked Questions	90

The accumulators store the current amount of measured quantity for a meter channel. This section provides detailed information about the accumulators.

7.1 Accumulator Totalizer and Residue

The accumulators are expressed as the totalizer and residue parts. This implementation allows the accumulation of a wide range of increments, while keeping a high precision of fractional part with an approximately constant and small round off error.

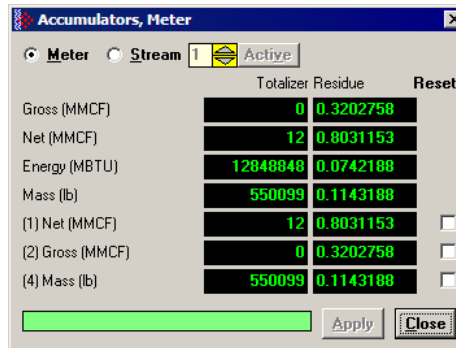
The totalizer stores the integral part of an accumulator as a 32-bit (or split) integer. The residue is the fractional part (always less than 1.0) expressed as a 32-bit IEEE floating point.

The Total Accumulator is given by the formula:

$$\text{ACCUMULATOR} = \text{TOTALIZER} + \text{RESIDUE}$$

Example

If the meter monitor window shows the following values for the accumulators:



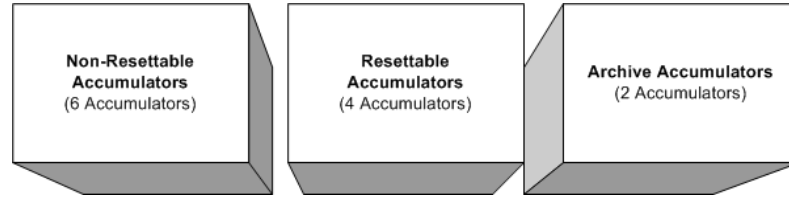
The total resettable accumulator 1 value (net) is 12.8031153.

The accumulator totalizer values can be configured to "split" with the low-order word rolling over from 9999 to 0000 at which time the high-order word is incremented. Refer to the AFC Manager (AFC Manager / Meter Configuration / Split Double Accumulators) to select this feature.

A 32-bit value is more suited to computation and has a greater range than a split value, whereas a split value is easier to read when it is represented as a pair of 16-bit numbers, as in a processor data file.

7.2 Accumulator Types

The module supports a total of 12 accumulators per meter channel divided into the following categories:



These 3 accumulator types are independent. For example, resetting a resettable accumulator does not affect the other accumulators.

For multiple-stream firmware (version 2.05 and later), each stream also has its own set of ten accumulators (six non-resettable and four resettable). Increments are applied both to the meter accumulators and to the accumulators for the active stream.

7.2.1 Non-Resetable Accumulators

The non-resettable accumulators are only reset when the accumulator rollover value is reached. The accumulator rollover value, and the accumulator unit must be configured using the AFC Manager. Refer to the AFC Manager User Manual for more information about this topic.

The module supports six non-resettable accumulators in order to show the measured quantity to be totalized:

- Non-resettable accumulator mass
- Non-resettable accumulator energy (Gas applications only)
- Non-resettable accumulator net
- Non-resettable accumulator gross
- Non-resettable accumulator gross standard (Liquid applications only). For Oil-Water Emulsion, this is non-resettable accumulator gross clean oil.
- Non-resettable accumulator water (Liquid applications only)

Refer to the Modbus Dictionary dialog box in AFC Manager for more information about the Modbus addresses for these registers.

7.2.2 Resetable Accumulators

The resettable accumulators are referred to as:

- Resetable Accumulator 1
- Resetable Accumulator 2
- Resetable Accumulator 3
- Resetable Accumulator 4

Configuring Resettable Accumulators

Resettable Accumulators are configured from the Resettable Accumulator Select dialog box. To open this dialog box, click the Resettable Accum button on the Meter Configuration dialog box.

Each Resettable Accumulator can be configured to represent a different quantity as follows:

Accumulator	Modbus address for accumulator select (Meter-relative)	Default Value
Resettable accumulator 1	136	Net (code 3)
Resettable accumulator 2	137	Gross (code 4)
Resettable accumulator 3	138	Gross Standard (code 5)
Resettable accumulator 4	139	Mass (code 1)

Valid Configuration Codes

The valid codes are:

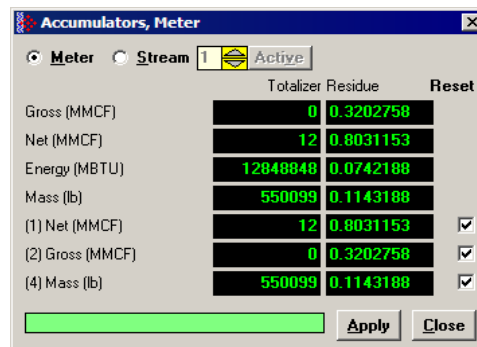
Code	Quantity
0	None
1	Mass
2	Energy (Gas Only)
3	Net
4	Gross
5	Gross Standard (Liquid Only)
6	Water (Liquid Applications Only).

For example, moving a value of 4 to holding register 8136 will configure Meter 1's resettable accumulator 1 as "Gross Volume". Moving "0" to holding register 10138 configures Meter 2's Resettable Accumulator 3 to accumulate nothing (takes it out of service).

The resettable accumulators are reset when one of the following situations occur.

Reset from AFC Manager

You may reset any of the resettable accumulators using the AFC Manager (Meter Monitor):

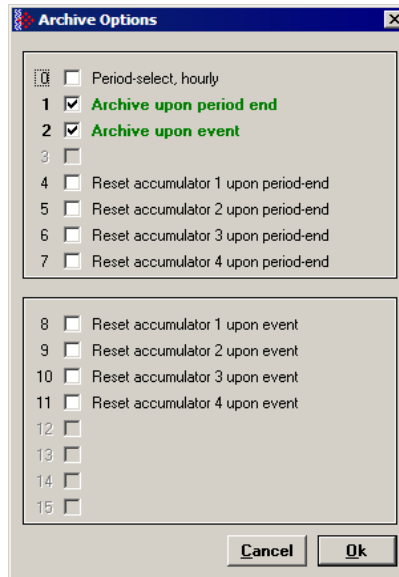


Reset from Ladder Logic

The ladder logic may send a meter signals block to command one or more resettable accumulators to be reset. This feature is especially important for applications involving field installations that require shipping and/or receiving product batches of predetermined size. Refer to the Ladder Logic section for your module type for more information.

Reset Upon Archive Period End or Reset Upon Event

Use AFC Manager to configure the resettable accumulator to be reset when the archive period ends or when an event occurs. Refer to **Event Log** in the *AFC Manager User Guide* for more information on configuring and monitoring events.



Refer to Archives (page 91) for more information.

Reset When the Accumulator Rollover Value is Reached

The resettable accumulator is reset when the accumulator rollover value is reached. You must configure the accumulator rollover value using the AFC Manager software (Meter Configuration). Refer to the AFC Manager User Manual for more information about this subject.

For multiple-stream firmware (version 2.05 or later), resetting a resettable accumulator resets that accumulator for both the meter and for all its streams.

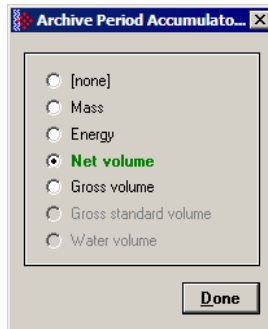
7.2.3 Archive Accumulators

The archive accumulators are part of the current archive (archive 0) data. These accumulators are automatically reset when a new archive is generated. The following Modbus holding registers are used:

Meter	Daily Archive		Hourly Archive	
	Accumulator: Totalizer	Accumulator: Residue	Accumulator: Totalizer	Accumulator: Residue
1	8890 to 8891	8892 to 8893	8894 to 8895	8896 to 8897
2	10890 to 10891	10892 to 10893	10894 to 10895	10896 to 10897
3	12890 to 12891	12892 to 12893	12894 to 12895	12896 to 12897
4	14890 to 14891	14892 to 14893	14894 to 14895	14896 to 14897
5	16890 to 16891	16892 to 16893	16894 to 16895	16896 to 16897
6	18890 to 18891	18892 to 18893	18894 to 18895	18896 to 18897
7	20890 to 20891	20892 to 20893	20894 to 20895	20896 to 20897
8	22890 to 22891	22892 to 22893	22894 to 22895	22896 to 22897

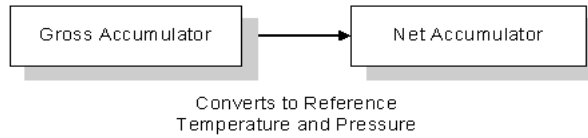
You can view the addresses, datum types and descriptions in the Modbus Dictionary dialog box.

You may configure the accumulator quantity to be used for each archive accumulator using the AFC Manager (**Meter Configuration / Archive Config / Accumulator Select**):

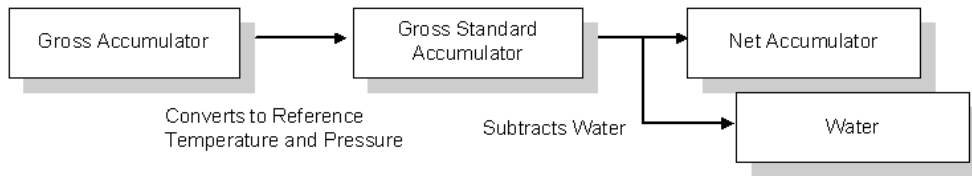


7.3 Net Accumulator Calculation

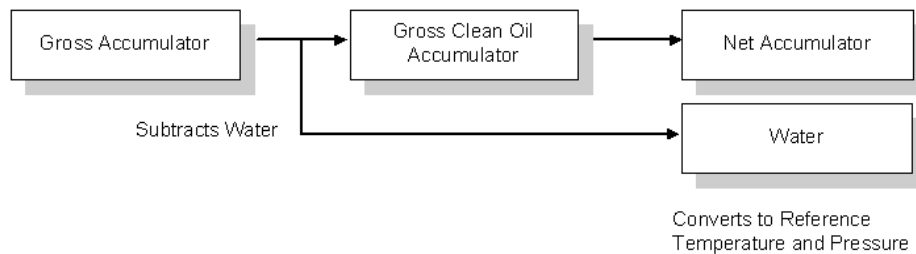
The Net Accumulator Calculation depends on the product group (gas or liquid).
For gas applications, the Net Accumulator is calculated as follows:



For liquid applications (all except Emulsion), the Net Accumulator is calculated as follows:



For liquid applications (Oil-Water Emulsion), the net accumulator is calculated as follows, using API ch 20.1:



7.4 Frequently Asked Questions

I need the accumulators to be reset upon period end. Which accumulator should my application use? Resettable Accumulator or Archive Accumulator?

You can use either one. The Archive Accumulators are reset every time a new archive is created and you configure whether the archive should be created upon period end and/or upon events.

There are some applications that may require the archives to be generated upon period end and upon event while the accumulators should be reset only upon period end. For these applications, you should consider the Resettable Accumulator (configured to be reset upon period end only) because the Archive Accumulators will also be reset when an event occurs.

8 Archives

In This Chapter

❖ Archive Overview	92
❖ Archive Generation.....	93
❖ Archive Types.....	94
❖ Archive Order	95
❖ Archive Options	97
❖ Archive Locations	98
❖ Editing the Archive Structure	100
❖ Extended Archives	102
❖ Archive Reports.....	105
❖ Archive Monitor	107

8.1 Archive Overview

An archive is a set of data that records relevant process values that occurred during a certain period of time (per meter channel). The archives are automatically generated by the module and no further action is required. The process values can include:

- Net flow rate (average)
- Total accumulator
- Temperature (average)
- Alarms occurred during the period

The process values will depend on the meter type and product group as listed later in this section.

Each archive contains two values that exhibit the period of time about that archive:

- opening timestamp = starting date and time for archive
- closing timestamp = ending date and time for archive

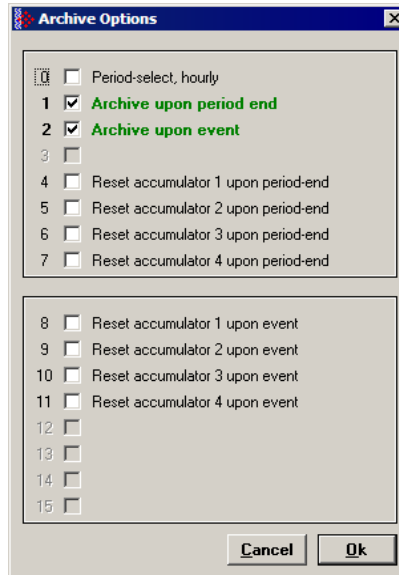
The example described in this chapter is of the default archive configuration as is present for a newly allocated meter. Version 2.01 of the firmware and AFC Manager allows the default configuration to be changed. Refer to Editing the Archive Structure.

8.2 Archive Generation

The archives can be generated during one of the following situations:

- Upon period end
- Upon event
- Upon processor command

You can configure if the archives should be generated upon period end and/or event using the AFC Manager (**Meter Configuration / Archive Config / Options**)



Refer to the AFC Manager User Manual for more information about this topic. By default the archives are generated upon period end and event.

If the archive is configured to be created upon period end, it will be periodically (daily or hourly) generated at the time configured by the End-of-day minute and End-of-hour minute parameters (**Project / Site Configuration**).

If the archive is configured to be created upon event, it will be generated every time an event occurs. For example, if an operator changes the orifice diameter for Meter 1, the module would automatically generate a new archive to save the relevant data to this point. Refer to this User Manual for the Events section for more information about events.

Note: Changing a meter type, product group, system of units, or primary input parameter will erase all archives for that meter.

8.3 Archive Types

The module supports two types of archives: hourly archives and daily archives:

Archive Type	Period	Period End	Number of 30-Word Archives Stored Locally
Hourly	60 minutes (1 hour)	Set by <i>End-of-Hour Minute</i> parameter	48
Daily	1440 minutes (1 day)	Set by <i>End-of-Day Minute</i> parameter	35

The Period End parameters must be set using the AFC Manager (Site Configuration). The default value is zero for both archive types which means that:

- Daily Archives are generated every day at midnight (00:00)
- Hourly Archives are generated every hour on the hour (1:00, 2:00, 3:00, 4:00)

For example, if the parameters are configured as follows:

End-of-day minute = 480

The daily archives would be created every day at 08:00.

End-of-hour minute = 30

The hourly archives would be created every hour at 1:30, 2:30, 3:30, 4:30, and so on.

8.4 Archive Order

An important concept regarding this topic is the archive order. Understanding this simple concept is essential when reading archive data (through the backplane or Modbus Master). Each archive has a number (its "age") that labels its position in the archive queue. The following table shows the archive numbering scheme (both daily and hourly archives):

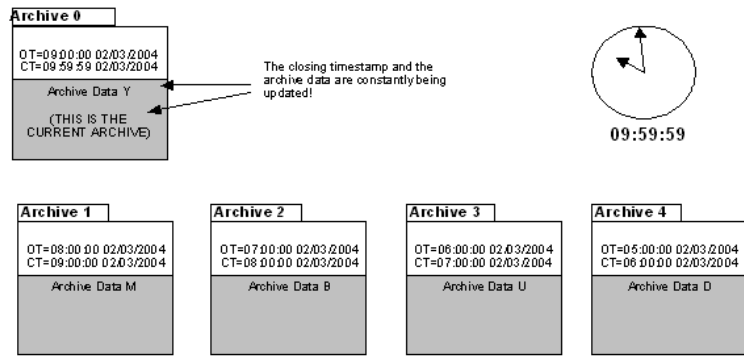
Archive Age	Register Types	Description
0	Holding Register	Current archive.
1	Input Register	Most recent archive
2	Input Register	Second most recent archive
3	Input Register	Third most recent archive
4	Input Register	Fourth most recent archive

(and so forth)

The archive 0 is the current archive. Because its period has not been concluded its closing timestamp and values (such as accumulator, average temperature, etc...) will be continuously updated. After the period is over (or an event occurs depending on the archive configuration) the data in archive 0 will be saved as the "new" archive 1. The data in the "old" archive 1 will be saved as the new archive 2 and so forth.

The current archive is stored in the primary slave's holding register bank. The past archives are stored in the primary slave's input register bank.

The following illustration shows an example for hourly archives:



Where:

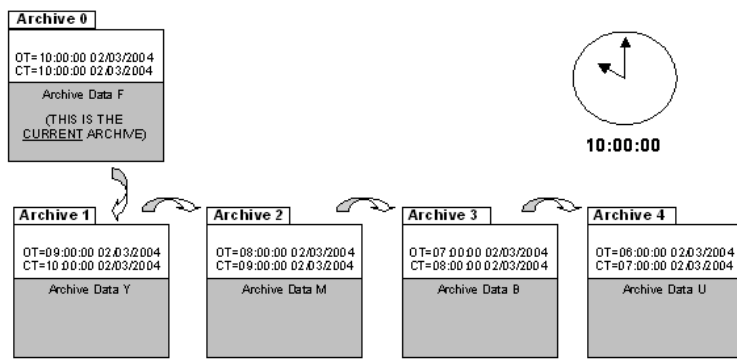
OT = Opening Time Stamp

CT = Closing Time Stamp

The previous figure shows an example where the hourly archives are configured to be generated upon period-end at the minute "0" (1:00, 2:00, 3:00, etc...).

Therefore, at 09:59:59 the archive 0 (current archive) is just about to be saved as the "new" archive 1.

When the clock changes to 10:00:00 the following illustration shows how the latest four archives are modified:



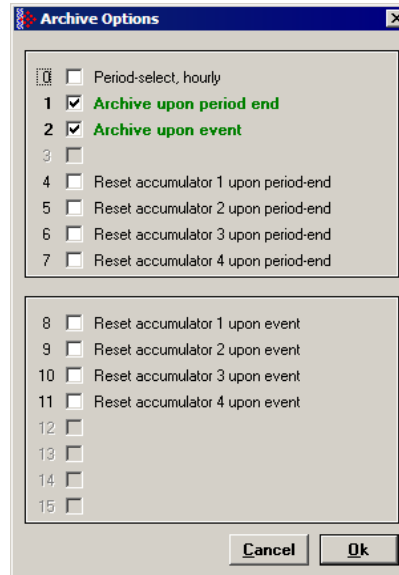
Where:

OT = Opening Time Stamp

CT = Closing Time Stamp

8.5 Archive Options

The module also allows you to configure whether the resettable accumulator should be reset upon period end and/or event. Most applications will require the resettable accumulators to be reset just after the archive is generated. The AFC Manager (version 2.01.000 or later) supports this feature through the archive options window as shown in the following example:



By default, the module is configured to generate archives upon period end and event. The module is not configured by default to reset the resettable accumulators upon period end.

8.6 Archive Locations

Click the Modbus Addresses button on the Archive Configuration dialog box to learn how to fetch an archive record of a specific age (procedure and Modbus location), and even the actual Modbus address of a specific file archived datum point (if you have highlighted the item in the archive record template).

The following table shows the current archive (Archive 0) location in the Primary Modbus Slave for each of the first 8 meters. These addresses refer to the holding register bank.

Archive 0 - Current Archives

Meter	Start Daily Archive	End Daily Archive	Start Hourly Archive	End Hourly Archive
1	9900	9939	9950	9989
2	11900	11939	11950	11989
3	13900	13939	13950	13989
4	15900	15939	15950	15989
5	17900	17939	17950	17989
6	19900	19939	19950	19989
7	21900	21939	21950	21989
8	23900	23939	23950	23989

Refer to the Modbus Dictionary dialog box for the current archive addressing.

The following table shows the past archives location in the Primary Modbus Slave for each of the first 8 meters. These addresses refer to the input register bank.

Archives 1 to n - Past Archives

Meter	Start Daily Archive	End Daily Archive	Start Hourly Archive	End Hourly Archive
1	0	1059	1060	2499
2	2500	3559	3560	4999
3	5000	6059	6060	7499
4	7500	8559	8560	9999
5	10000	11059	11060	12499
6	12500	13559	13560	14999
7	15000	16059	16060	17499
8	17500	18559	18560	19999

The default configuration sets 30 words per meter archive. For example, the Meter 1 daily archives are addressed as follows:

Daily Archive Number	Start Address	End Address
1	0	29
2	30	59
3	60	89
4	90	119
...
35	1020	1049

The Meter 1 hourly archives are addressed as follows:

Hourly Archive Number	Start Address	End Address
1	1060	1089
2	1090	1119
3	1120	1149
4	1150	1179
...
48	2470	2499

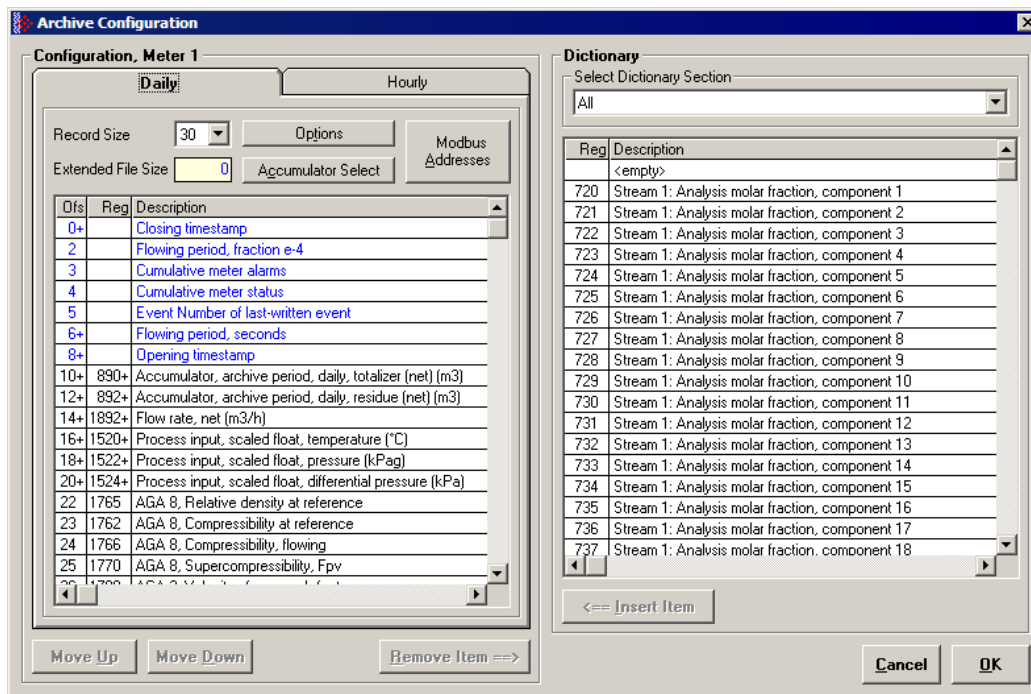
8.7 Editing the Archive Structure

Note: The features presented on this section are only available for AFC firmware version 2.01.000 or later. Please contact the tech support team for more information about the module upgrade.

For advanced applications, you can edit the archive contents, the record size, the order of the registers in the archive, and the archive accumulator quantity.

The Archive Configuration window (**Meter Configuration / Archive Config**) allows you to fully configure the meter archive (daily or hourly). The data to be inserted in the archive must be copied from the Dictionary Section on the right half of the window.

Refer to the AFC Manager User Manual for more information about this topic.



The module reserves 1060 words for daily archives and 1440 words for hourly archives. Because the default configuration sets the record size for 30 words, it means that the maximum (default) number of archives per meter channel is 35 daily archives and 48 hourly archives. However, because you can change the number of words per archive, the actual maximum number of archives per meter channel will depend on the configured number of words per archive as follows:

Number of Words per Archive	Number of Daily Archives	Number of Hourly Archives
10	106 daily archives	144 hourly archives
20	53 daily archives	72 hourly archives
30	35 daily archives	48 hourly archives
40	26 daily archives	36 hourly archives

You may also configure the accumulator type for each archive. You must configure one of the following options:

- Mass
- Energy (Gas product only)
- Net Volume
- Gross Volume
- Gross Standard
- Water Volume (Liquid product only)

The following topics show the default archive structure when you configure a new meter. You can edit this structure according to your own requirements.

8.8 Extended Archives

This feature is only supported on firmware versions 2.01.000 or newer, and requires a Compact Flash card to be installed.

The module supports the extended archive feature that allows you to configure more archives than the regular 35 daily archives and 48 hourly archives. The module supports the following number of extended archives:

	Daily Archives	Hourly Archives
Max Number of Archives	350 (version 2.04 and earlier) 1440 (version 2.05 and newer)	1260 (version 2.04 and earlier) 1440 (version 2.05 and newer)

Refer to Extended File Size entry on the **Archive Configuration** window for more information.

Note: The maximum number of extended archives is not dependent on the number of words per archive. Extended archives are stored on a Compact Flash card which must be installed for Extended Archive configuration to be effective.

8.8.1 Retrieving Extended Archives

The module implements an easy way to retrieve extended archives from the Modbus database. To learn how to retrieve extended archives, click Archive Config on the Meter Configuration dialog box, and then click Modbus Addresses.

For each archive file the module reserves a block of 50 Input registers to hold the "selected Archive", as listed in the following table.

Meter	Daily Archive Start (Input Register)	Daily Archive End (Input Register)	Hourly Archive Start (Input Register)	Hourly Archive End (Input Register)
1	60000	60049	60050	60099
2	60100	60149	60150	60199
3	60200	60249	60250	60299
4	60300	60349	60350	60399
5	60400	60449	60450	60499
6	60500	60549	60550	60599
7	60600	60649	60650	60699
8	60700	60749	60750	60799
9	60800	60849	60850	60899
10	60900	60949	60950	60999
11	61000	61049	61050	61099
12	61100	61149	61150	61199
13	61200	61249	61250	61299
14	61300	61349	61350	61399
15	61400	61449	61450	61499
16	61500	61549	61550	61599

Note: Meters 9 through 16 are only available for the PTQ-AFC and MVI56-AFC modules.

The Selected Archive start address can be calculated as (assumes meters are numbered starting at 1):

Daily Archive Start Address = $60000 + ((\text{Meter Number} - 1) * 100)$

Hourly Archive Start Address = $60000 + (((\text{Meter Number} - 1) * 100) + 50)$

Note: When using processor logic to retrieve extended archives, when possible, use unsigned 16-bit integer data type variables to hold archive addresses. Unsigned 16-bit integers display data in the range 0 to 65535.

If your programming software (such as Rockwell Automation® RSLogix™ 5000) does not support unsigned integer data types, there are a couple of possible alternatives. If your programming software supports signed 32-bit double integer data types, you may use that type of variable to hold the addresses above.

If you must use signed 16-bit integer data type variables to contain addresses (such as in the case of Rockwell Automation RSLogix5 or RSLogix500), you will not be able to enter the values in the previous table as positive numbers. This is because 16-bit signed integers display values only in the range -32768 to +32767. But, it is the underlying bit pattern and not the displayed decimal value that is important to the AFC module.

To enter the correct bit pattern for these addresses into a signed 16-bit integer, you will need to enter them as negative numbers. To determine the correct negative number, simply subtract 65536 from the address in the table, which will result in a negative number being displayed in the signed integer variable. This negative number (a binary twos-complement form of the archive address) will contain the equivalent bit pattern for the value in the chart if it were held in an unsigned integer variable.

Use these modified versions of the above formulas to calculate the address values for signed 16-bit integer variables:

Daily Archive Start Address = $((60000 + ((\text{Meter Number} - 1) * 100)) - 65536)$

Hourly Archive Start Address = $((60000 + (((\text{Meter Number} - 1) * 100) + 50)) - 65536)$

The Selected Archive is continuously maintained to be a copy of the archive record having the age given in the corresponding "Archive Select" holding register, as listed in the following table. This means that the Selected Archive changes whenever either (a) the age in the Open Archive Select register is changed or (b) when the posting of a new archive causes the ages of all archives to be increased by 1.

Meter	Open Daily Archive Select Address	Open Hourly Archive Select Address
1	8300	8301
2	10300	10301
3	12300	12301
4	14300	14301
5	16300	16301
6	18300	18301
7	20300	20301
8	22300	22301
9	24300	24301
10	26300	26301
11	28300	28301
12	30300	30301
13	32300	32301
14	34300	34301
15	36300	36301
16	38300	38301

Note: Meters 9 through 16 are only available for the PTQ-AFC and MVI56-AFC modules.

Use the following procedure to retrieve extended archives:

- 1 Copy the archive age to the correct Open Archive Select register.
- 2 Read the archive data from the 60000-range input addresses.

Example

To read Meter 2 Hourly Archive Number 277:

- 1 Write a value of 277 to Modbus Holding Register 10301.
- 2 Read the archive record data starting at input register 60150.

Note: This procedure can also be used to retrieve regular archives.

8.9 Archive Reports

Use the Archive Monitor in AFC Manager to generate an archive report or print it to a local printer. You can also save the archive report in two formats:

- Text
- Comma Separated

A report saved in **text format** (.log) contains a complete archive description. The following illustration shows an example of a text format report.

```

AFC-56(16) [2.02] Daily Archive                               Date: 4/15/2004 9:23:52 AM
Site Name: MVI Flow Station
Project: AFC
File: \\C:\AFC-56(16)
    
```

```

Meter 16:
Tag                               M01
Archive                            33
Closing timestamp                  2004-04-17.01:49:42
Flowing period, fraction e-4      1
Cumulative meter alarms           0000h
Cumulative site status            00h
Event Number of last-written event 160
Flowing period, seconds           16
Opening timestamp                  2004-04-17.01:49:26
Accumulator, archive period, daily, totalizer (m3) 0
Accumulator, archive period, daily, residue (m3) 0.4645103
Flow rate, net (m3/h)              101.4091
Process input, scaled float, temperature (°C) 20
Process input, scaled float, pressure (kPag) 50
Process input, scaled float, dif prs / flow rate / freq (kPa) 70
AGA 8, Relative density at reference 0.5548
AGA 8, Compressibility at reference 0.998
AGA 8, Compressibility, flowing 0.9959
AGA 8, Supercompressibility, Fpv 1.001
AGA 3, Velocity of approach factor 1
AGA 3, Expansion factor            0.9017
AGA 3, Coefficient of discharge    0.5975
<not used>                         0

Alarm Bits
bit 0 Temperature input out of range -
bit 1 Pressure input out of range -
bit 2 Differential pressure input out of range -
bit 3 Flowing density input out of range -
bit 4 Water content input out of range -
bit 5 Differential pressure low -
bit 7 Accumulator overflow -
bit 8 Orifice characterization error -
bit 9 Analysis total zero -
bit 10 Analysis total not normalized -
bit 11 Compressibility calculation error -
bit 12 API calc error - density correction -
bit 13 API calc error - CT1 -
bit 14 API calc error - vapour pressure -
bit 15 API calc error - Cpl -

Status Bits
bit 11 Meter was enabled -
bit 12 Backplane communication fault -
bit 13 Measurement configuration changed -
bit 14 Power up -
bit 15 Cold start -
    
```

Saving the archive report in **comma-separated** (.csv) format allows it to be imported to an Excel® spreadsheet. The following example shows a portion of the .CSV report imported into Excel:

	A	B	C	D
1	AFC-71(8) [2.02] Daily Archive			
2	Date:	3/30/2004 11:21		
3	Site Name:	MVI Flow Station		
4	Project:	AFC		
5	Meter 2:			
6	Tag	M01		
7				
8	Archive	Current	1	2
9				
10	Closing timestamp	2004-03-30.08:36:54	2004-03-30.00:00:00	2004-03-29.00:00:00
11	Flowing period, fraction e-4	1	1	1
12	Cumulative meter alarms	0002h	0002h	0002h
13	Cumulative site status	00h	00h	00h
14	Event Number of last-written event	474	474	474
15	Flowing period, seconds	31014	86400	86400
16	Opening timestamp	2004-03-30.00:00:00	2004-03-29.00:00:00	2004-03-28.00:00:00
17	Accumulator, archive period, daily, totalizer (m3)	7	20	20
18	Accumulator, archive period, daily, residue (m3)	0.3965147	0.6051551	0.6052574
19	Flow rate, net (m3/h)	0.8572201	0.8571935	0.8571963
20	Process input, scaled float, temperature (°C)	49.99487	50.03679	50.03685
21	Process input, scaled float, pressure (kPag)	1	1	1
22	Process input, scaled float, dif prs / flow rate / freq (kPa)	11.00041	11.00247	11.00249
23	Process input, scaled float, flowing density (kg/m3)	700.3123	700.6372	700.6348
24	API 2540, Density at reference (kg/m3)	730.3	730.7	730.7
25	API 2540, Temperature correction factor, CTL	0.9592	0.9596	0.9596
26	API 2540, Pressure correction factor, CPL	0.9999	1.0001	1.0001
27	AGA 3, Velocity of approach factor	1	1.0003	1.0003
28	AGA 3, Expansion factor	0.9999	1.0001	1.0001
29	AGA 3, Coefficient of discharge	0.5964	0.5966	0.5966
30				

8.10 Archive Monitor

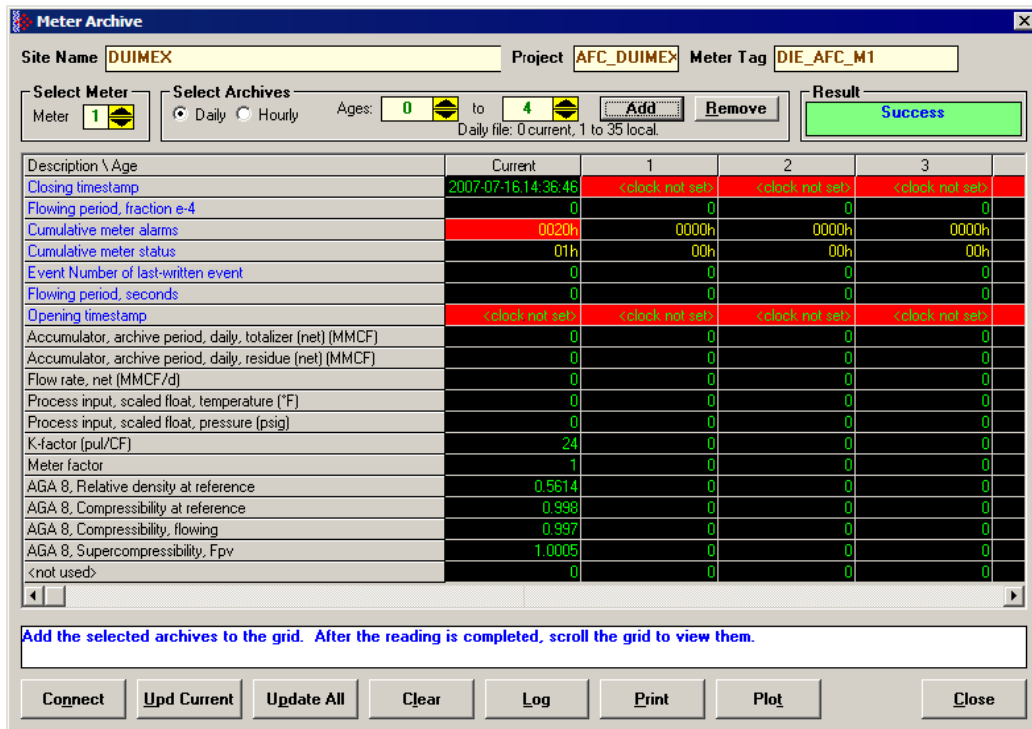
The Archive Monitor dialog box opens when you open the Monitor menu, and then choose Archive.

The module can archive data for each meter channel. The archives are periodically generated according to the period end defined in the Site Configuration.

There are hourly archives (48 archives) and daily archives (35 archives).

For example the daily archives will be stored as:

- Archive 0 = current archive
 - Archive 1 = Archive created yesterday
 - Archive 2 = Archive created 2 days ago
 - Archive 3 = Archive created 3 days ago
- And so on.



Control	Description
Select Meter	Select the meter number
Select Archives	Select the archive type
Ages	Select the first archive to be added or removed
To	Select the last archive to be added or removed
Add	Add the selected archives to the grid, fetching as necessary
Remove	Remove the selected archives from the grid
Connect	Connect to the module, if necessary
Upd Current	Update the current archive
Update All	Update all archives in the grid
Clear	Clear the grid
Log	Create a log file containing the archived data
Print	Print the archives to the local printer
Plot	Display a plot of two datum points from archives in the grid

The following shows an example of an archive report generated by the AFC Manager:

AFC-56(16) Daily Archive Date: 16-09-2002 16:26:41
 Site Name: MVI Flow Station
 Project: AFC

```

Meter 1:
Tag                M01
Archive            0

Closing timestamp of archive    2002-04-27.23:59:08
Opening timestamp of archive    2002-04-27.00:00:02
Status bitmap (details below)   00h
Alarms bitmap (details below)   0000h
Flowing period                  86346
Event counter                   53
Net accumulator (x f3)          604
Net accumulator residue (x f3)  0,6703186
Net flow rate (x f3/h)          40247,93
Temperature (°F)                14,99997
Pressure (psig)                 999,9995
Differential pressure (hw)       21,99997
Relative density (60°F/60°F)    0,7404
Reference compressibility        0,9989
Flowing compressibility          0,9051
Fpv                             1,0505
Velocity of approach factor Ev  1,0328
Expansion factor Y               0,9997
Discharge coefficient            0,6043

Alarm Bits
bit 0 Temperature input out of range -
bit 1 Pressure input out of range   -
    
```

```

bit 2 Diff. pressure input out of range -
bit 3 Flowing density input out of range -
bit 4 Water content input out of range -
bit 5 Diff. pressure low -
bit 8 Orifice characterization error -
bit 9 Analysis total zero -
bit 10 Analysis total not normalized -
bit 11 AGA8 calculation error -
bit 12 API calculation error, density correctio -
bit 13 API calculation error, Ctl -
bit 14 API calculation error, vapor pressure -
bit 15 API calculation error, Cpl -
  
```

Status Bits

```

bit 11 Meter was enabled -
bit 12 Backplane communication fault -
bit 13 Measurement configuration changed -
bit 14 Power up -
bit 15 Cold start -
  
```

AFC-56(16) Daily Archive
 Site Name: MVI Flow Station
 Project: AFC

Date: 16-09-2002 16:26:41

Meter 1:

```

Tag M01
Archive 1

Closing timestamp of archive 2002-04-27.00:00:02
Opening timestamp of archive 2002-04-26.23:59:42
Status bitmap (details below) 00h
Alarms bitmap (details below) 0000h
Flowing period 20
Event counter 53
Net accumulator (x f3) 234
Net accumulator residue (x f3) 0,1092186
Net flow rate (x f3/h) 40248,01
Temperature (°F) 15
Pressure (psig) 1000
Differential pressure (hw) 22
Relative density (60°F/60°F) 0,7404
Reference compressibility 0,9989
Flowing compressibility 0,9051
Fpv 1,0505
Velocity of approach factor Ev 1,0328
Expansion factor Y 0,9997
Discharge coefficient 0,6043
  
```

Alarm Bits

```

bit 0 Temperature input out of range -
bit 1 Pressure input out of range -
bit 2 Diff. pressure input out of range -
bit 3 Flowing density input out of range -
  
```

```

bit 4 Water content input out of range -
bit 5 Diff. pressure low -
bit 8 Orifice characterization error -
bit 9 Analysis total zero -
bit 10 Analysis total not normalized -
bit 11 AGA8 calculation error -
bit 12 API calculation error, density correctio -
bit 13 API calculation error, Ctl -
bit 14 API calculation error, vapor pressure -
bit 15 API calculation error, Cpl -

```

Status Bits

```

bit 11 Meter was enabled -
bit 12 Backplane communication fault -
bit 13 Measurement configuration changed -
bit 14 Power up -
bit 15 Cold start -

```

AFC-56(16) Daily Archive
Site Name: MVI Flow Station
Project: AFC

Date: 16-09-2002 16:26:44

Meter 1:

```

Tag M01
Archive 2

```

```

Closing timestamp of archive 2002-04-26.23:59:42
Opening timestamp of archive 2002-04-26.06:16:34
Status bitmap (details below) 60h
Alarms bitmap (details below) 0000h
Flowing period 1019877652
Event counter 53
Net accumulator (x f3) 174811
Net accumulator residue (x f3) 0,9399567
Net flow rate (x f3/h) 40247,88
Temperature (°F) 15,00736
Pressure (psig) 1000,416
Differential pressure (hw) 22,00479
Relative density (60°F/60°F) 0,7404
Reference compressibility 0,9989
Flowing compressibility 0,9053
Fpv 1,0506
Velocity of approach factor Ev 1,0331
Expansion factor Y 1,0001
Discharge coefficient 0,6045

```

Alarm Bits

```

bit 0 Temperature input out of range -
bit 1 Pressure input out of range -
bit 2 Diff. pressure input out of range -
bit 3 Flowing density input out of range -
bit 4 Water content input out of range -
bit 5 Diff. pressure low -

```

```

bit 8 Orifice characterization error -
bit 9 Analysis total zero -
bit 10 Analysis total not normalized -
bit 11 AGA8 calculation error -
bit 12 API calculation error, density correctio -
bit 13 API calculation error, Ctl -
bit 14 API calculation error, vapor pressure -
bit 15 API calculation error, Cpl -
  
```

Status Bits

```

bit 11 Meter was enabled -
bit 12 Backplane communication fault -
bit 13 Measurement configuration changed yes
bit 14 Power up yes
bit 15 Cold start -
  
```

AFC-56(16) Daily Archive
 Site Name: MVI Flow Station
 Project: AFC

Date: 16-09-2002 16:26:51

Meter 1:

```

Tag M01
Archive 3

Closing timestamp of archive 2002-04-26.06:16:34
Opening timestamp of archive 2002-04-26.06:14:08
Status bitmap (details below) 20h
Alarms bitmap (details below) 0000h
Flowing period 146
Event counter 50
Net accumulator (x f3) 1633
Net accumulator residue (x f3) 6,271362E-02
Net flow rate (x f3/h) 40248,02
Temperature (°F) 14,99999
Pressure (psig) 1000,002
Differential pressure (hw) 22,00003
Relative density (60°F/60°F) 0,7404
Reference compressibility 0,9989
Flowing compressibility 0,9051
Fpv 1,0505
Velocity of approach factor Ev 1,0328
Expansion factor Y 0,9997
Discharge coefficient 0,6043
  
```

Alarm Bits

```

bit 0 Temperature input out of range -
bit 1 Pressure input out of range -
bit 2 Diff. pressure input out of range -
bit 3 Flowing density input out of range -
bit 4 Water content input out of range -
bit 5 Diff. pressure low -
bit 8 Orifice characterization error -
bit 9 Analysis total zero -
  
```

bit 10	Analysis total not normalized	-
bit 11	AGA8 calculation error	-
bit 12	API calculation error, density correctio	-
bit 13	API calculation error, Ctl	-
bit 14	API calculation error, vapor pressure	-
bit 15	API calculation error, Cpl	-
Status Bits		
bit 11	Meter was enabled	-
bit 12	Backplane communication fault	-
bit 13	Measurement configuration changed	yes
bit 14	Power up	-
bit 15	Cold start	-

9 Events

In This Chapter

❖ The Event Log	114
❖ Event Log Structures	115
❖ Event Id Tag	116
❖ Event-triggered Archives and Accumulator Resets	117
❖ Event Log Download	Error! Bookmark not defined.
❖ Period-end Events	139
❖ Loggable Events	140
❖ Special Events	141
❖ Site Data Point Events	142
❖ Meter Data Point Events	143
❖ Stream Data Point Events	146
❖ Stream Data Point Events	148
❖ "Rkv" Notes	151
❖ Event Numbers	152

9.1 The Event Log

An "event" is any occurrence that may affect the manner in which, or whether, measurement is performed. Events include, for example:

- Any change to a sealable parameter.
- Power-up (product may have been lost during the power-down period).
- A change in PLC operating mode (programming changes may alter measurement).
- A download of the event log (for audit trail purposes).

The Event Log occupies a block of 16000 Input registers in the Modbus table starting at address 40000 and proceeding through address 55999. It consists of a 5-register "header" at address 40000 followed by 1999 8-register "event" records starting at address 40008. As they are Input registers (read with Modbus function code 4), no part of the Event Log can be written from outside the module, but it is maintained exclusively by the AFC firmware.

As events occur they are recorded in the Log, which acts as a circular file. Each new event record overwrites the oldest one, hence the log stores up to 1999 of the most recent events. As each record is written the values in the header are updated to reflect the new status of the log.

Auditors may require the Log to be "downloaded" from time to time; events are read from the module and stored in a more permanent database, and the events so copied and archived are marked in the module as "downloaded".

If all record positions contain events that have not yet been downloaded, the log is full. In this case, the handling of a new event depends on the value of the "Event log unlocked" site option:

- If the option is set, then the log-full condition is ignored and the new event overwrites the oldest one. Since the overwritten event was never downloaded, it is permanently lost.
- If the option is clear, then the Event Log is "locked", and the new event is rejected if possible and otherwise ignored. Controllable events, that is, changes to sealable parameters, are not allowed to occur; such datum points remain unchanged retaining their current values and a Modbus command that attempts such a change receives an "illegal data" exception response. Uncontrollable events, such as PLC mode change, are simply not recorded. The Log must be downloaded in order to unlock it for further events.

9.2 Event Log Structures

The Event Log header contains housekeeping information for maintaining the Log. Its layout is:

Address	Description
40000	Number of records maximum (== 1999)
40001	Next new record position (0 thru maximum-1)
40002	Next new event number (0 thru 65535, wrapping to 0)
40003	Oldest event number on file
40004	Oldest event number on file not yet downloaded
40005-40007	[reserved]

Each event record is an 8-register quantity laid out as four 32-bit items (big-endian):

Registers	Contents
0 to 1	Event Id Tag (page 116)
2 to 3	Timestamp of event In our standard "packed bit-field" format.
4 to 5	Old item value For a Datum Point event, format depends on the "datum type" field of the Event Id Tag.
6 to 7	New item value For a Datum Point event, format depends on the "datum type" field of the Event Id Tag.

Each value is right-justified in its field and sign-extended or padded with zeros (0) if necessary, according to the source data type.

9.3 Event Id Tag

This 32-bit field has the following structure:

Bits	N	Meaning																																																			
31	1	0 Special, 1 Datum Point (e.g. sealable parameter) If this bit is clear, then bits 19-00 contain a value from the Special event tag list below; if the bit is set, then bits 19-00 have the interpretation given here.																																																			
30	1	PLC offline; timestamp may not be accurate This bit may also be set for a Special event.																																																			
29	1	[reserved]																																																			
28 to 24	5	Meter number, or 0 for Site This field may also be set for a Special event.																																																			
23 to 20	4	[Meter] Stream number or 0; [Site] 0 This field may also be set for a Special event.																																																			
19 to 16	4	Data type: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Mnemonic</th> <th>Format</th> </tr> </thead> <tbody> <tr><td>0</td><td>Ubyt</td><td>Unsigned byte</td></tr> <tr><td>1</td><td>Usht</td><td>Unsigned short integer</td></tr> <tr><td>2</td><td></td><td>[reserved]</td></tr> <tr><td>3</td><td>Ulng</td><td>Unsigned long integer</td></tr> <tr><td>4</td><td>Sbyt</td><td>Signed byte</td></tr> <tr><td>5</td><td>Ssht</td><td>Signed short integer</td></tr> <tr><td>6</td><td></td><td>[reserved]</td></tr> <tr><td>7</td><td>Slng</td><td>Signed long integer</td></tr> <tr><td>8</td><td>Bbyt</td><td>Bitmap (up to 8 bits)</td></tr> <tr><td>9</td><td>Bsht</td><td>Bitmap (up to 16 bits)</td></tr> <tr><td>10</td><td>Bm24</td><td>Bitmap (up to 24 bits)</td></tr> <tr><td>11</td><td>Blng</td><td>Bitmap (up to 32 bits)</td></tr> <tr><td>12</td><td>Bool</td><td>Boolean (value 0 or 1)</td></tr> <tr><td>13</td><td>DiBy</td><td>Dibyte (both high and low)</td></tr> <tr><td>14</td><td>B448</td><td>Bitfield nybble/nybble/byte</td></tr> <tr><td>15</td><td>Flot</td><td>Floating point</td></tr> </tbody> </table>	Value	Mnemonic	Format	0	Ubyt	Unsigned byte	1	Usht	Unsigned short integer	2		[reserved]	3	Ulng	Unsigned long integer	4	Sbyt	Signed byte	5	Ssht	Signed short integer	6		[reserved]	7	Slng	Signed long integer	8	Bbyt	Bitmap (up to 8 bits)	9	Bsht	Bitmap (up to 16 bits)	10	Bm24	Bitmap (up to 24 bits)	11	Blng	Bitmap (up to 32 bits)	12	Bool	Boolean (value 0 or 1)	13	DiBy	Dibyte (both high and low)	14	B448	Bitfield nybble/nybble/byte	15	Flot	Floating point
Value	Mnemonic	Format																																																			
0	Ubyt	Unsigned byte																																																			
1	Usht	Unsigned short integer																																																			
2		[reserved]																																																			
3	Ulng	Unsigned long integer																																																			
4	Sbyt	Signed byte																																																			
5	Ssht	Signed short integer																																																			
6		[reserved]																																																			
7	Slng	Signed long integer																																																			
8	Bbyt	Bitmap (up to 8 bits)																																																			
9	Bsht	Bitmap (up to 16 bits)																																																			
10	Bm24	Bitmap (up to 24 bits)																																																			
11	Blng	Bitmap (up to 32 bits)																																																			
12	Bool	Boolean (value 0 or 1)																																																			
13	DiBy	Dibyte (both high and low)																																																			
14	B448	Bitfield nybble/nybble/byte																																																			
15	Flot	Floating point																																																			
15 to 12	4	[reserved]																																																			
11 to 08	4	Group code This value is one of the "measurement configuration changed" bit numbers.																																																			
07 to 04	4	Subgroup code This value is the ordinal number (starting at 0) of the subgroup of parameters in the specified group.																																																			
03 to 00	4	Subgroup item code Since a parameter subgroup may contain more than one item, this value identifies the particular item; items are numbered from 0.																																																			

9.4 Event-triggered Archives and Accumulator Resets

Each archive file (two for each meter) contains an Archive Options bitmap whose configuration specifies the actions to be scheduled (write archive and/or reset resettable accumulator(s)) when an event occurs (daily or hourly period-end, or most loggable events). Archives and/or resets are scheduled only for enabled meters (with one important clarification; see "Rkv" notes (page 151)). The actions to be taken upon period-end and those to be taken upon loggable events are configured separately.

Several archive/reset-triggering events can occur simultaneously. In such cases the archive or reset occurs only once (an archive is written only when archivable data has been accumulated for at least one meter scan; additional resets of already-reset accumulators have no effect).

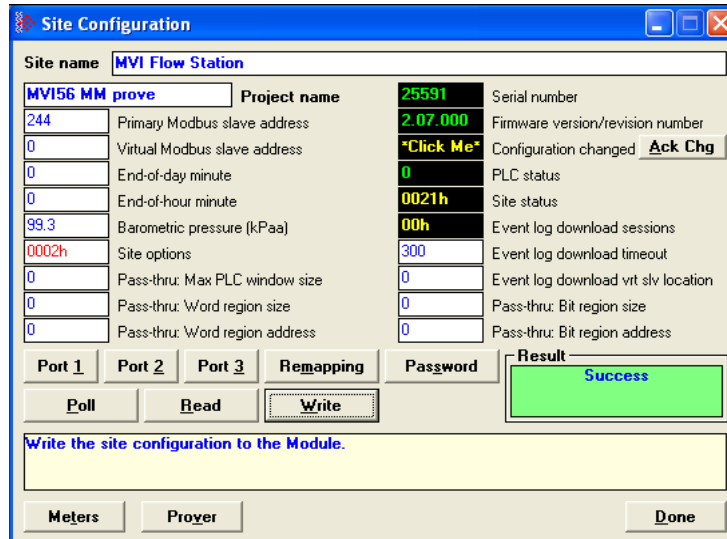
Scheduled accumulator resets are performed at the top of the meter scan. This permits their final values to be inspected/fetched/archived while the AFC rotates its scan among the other meters.

Scheduled archives are written at the top of the meter scan, at its bottom, or between successive scans, depending on the nature of the triggering event. Archives written at the top of the scan are written before any accumulator resets.

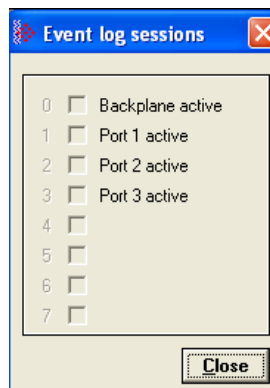
9.5 Downloading the Event Log in Firmware Version 2.07 and Later

The following is an example of the Event Log Download.

In the *Site Configuration* window, the *Event log download timeout* has been changed from the default of 60 seconds, to 300 seconds (5 minutes).



In the *Site Configuration* window above there are no active download sessions. This is indicated by the value of zero in *Event log download sessions field*. Click the box to show the *Event log sessions* window.



After opening the *Event log sessions* window from the **ON-LINE/EVENT LOG** menu; you will see instructions in green in the lower right area of the window.

Click **POLL** to fetch status and to prepare the session:

The screenshot shows the 'Event Log' window with the following details:

- Site name:** MVI Flow Station
- Project:** MVI56 MM prove
- Buttons:** Poll (highlighted), Download
- Set Up Session:** First event (0), Last event + 1 (0), Acknowledgement method (Brief selected, Verbose unselected), Manual selection (unchecked)
- Event Log Status:** Wallclock (blacked out), Next event to be written (blacked out), Next event to be downloaded (blacked out), Events not yet downloaded (blacked out)
- Table:**

Event	ID tag	Description	Timestamp	Old value	New value
- Result:** (Green box)
- Message:** Learn how many events have not been logged yet. After this, click Download to read all of those events.
- Save Session Data:** Log, Print
- Exit Session:** Commit, Abandon
- Instructions:** Step 1: "Poll" to fetch event log status and prepare to set up session.
- Buttons:** Close

After polling, you are prompted to download:

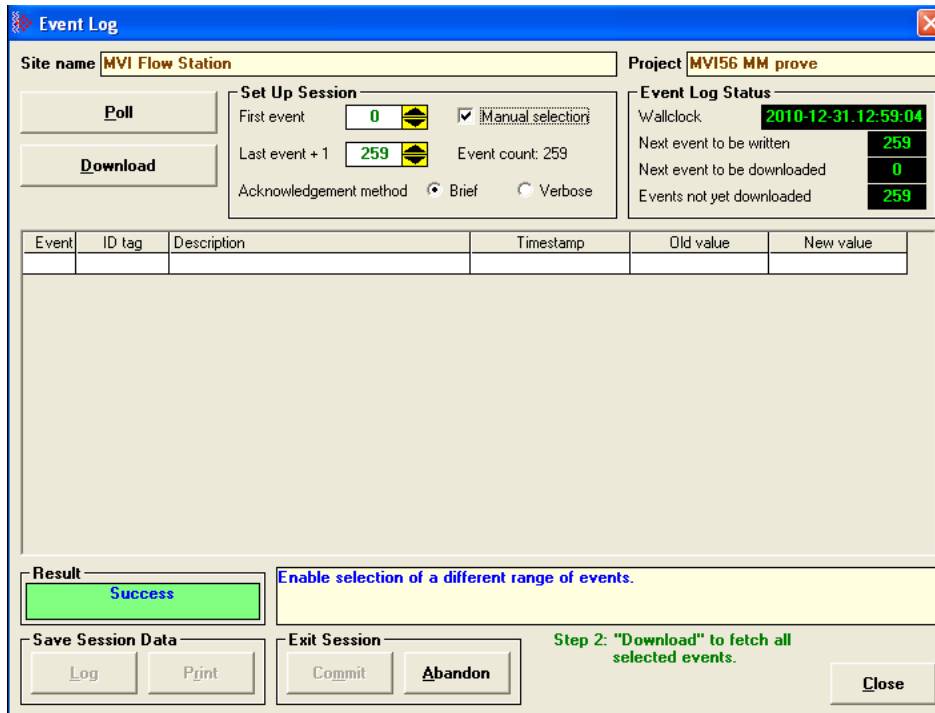
The screenshot shows the 'Event Log' window after the polling step:

- Site name:** MVI Flow Station
- Project:** MVI56 MM prove
- Buttons:** Poll, Download (highlighted)
- Set Up Session:** First event (0), Last event + 1 (259), Event count: 259, Acknowledgement method (Brief selected, Verbose unselected), Manual selection (unchecked)
- Event Log Status:** Wallclock (2010-12-31.12:59:04), Next event to be written (259), Next event to be downloaded (0), Events not yet downloaded (259)
- Table:**

Event	ID tag	Description	Timestamp	Old value	New value
- Result:** Success (Green box)
- Message:** Learn how many events have not been logged yet. After this, click Download to read all of those events.
- Save Session Data:** Log, Print
- Exit Session:** Commit, Abandon
- Instructions:** Step 2: "Download" to fetch all selected events.
- Buttons:** Close

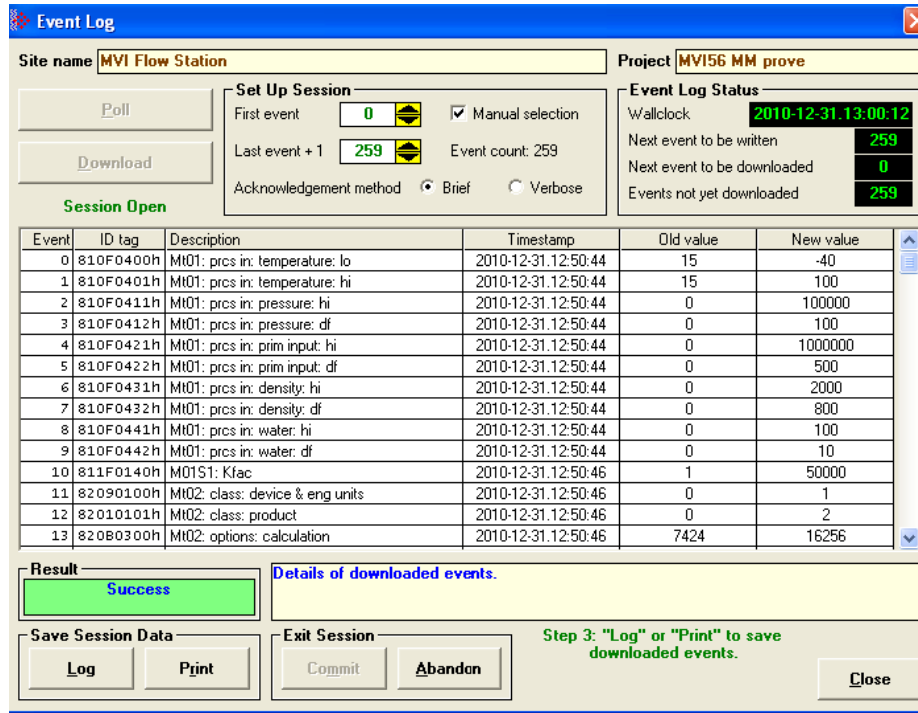
Before downloading you might want to modify the parameters of your download. Reasons for modification include such things as a need to re-fetch recent events that were already downloaded and committed in a previous session, or to limit the amount of download.

To change the download parameters, check the box **MANUAL SELECTION** in the *Event Log* window, and make any desired changes.

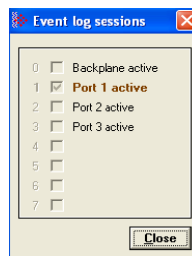


After the parameters are set to the chosen values click **DOWNLOAD**.

The requested events are fetched and displayed in the scrollable grid. The events displayed in this screenshot are of changes to configuration (*sealable parameters*):

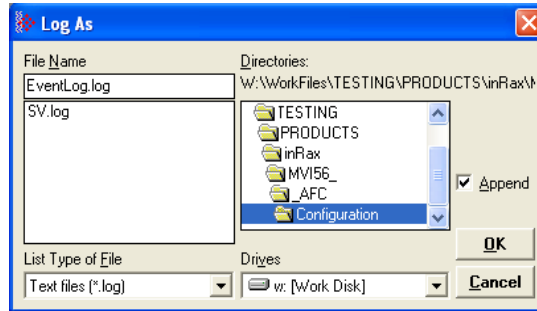


The *Download* action opens an *Event log sessions* window, which will be closed only when either (a) it is *Committed*, or (b) it is *Abandoned*, or (c) it times out due to no activity. this can be seen by viewing again the *Event log download sessions* information from *Site Configuration* window (you will have to re-*Poll* the site information to see the update):

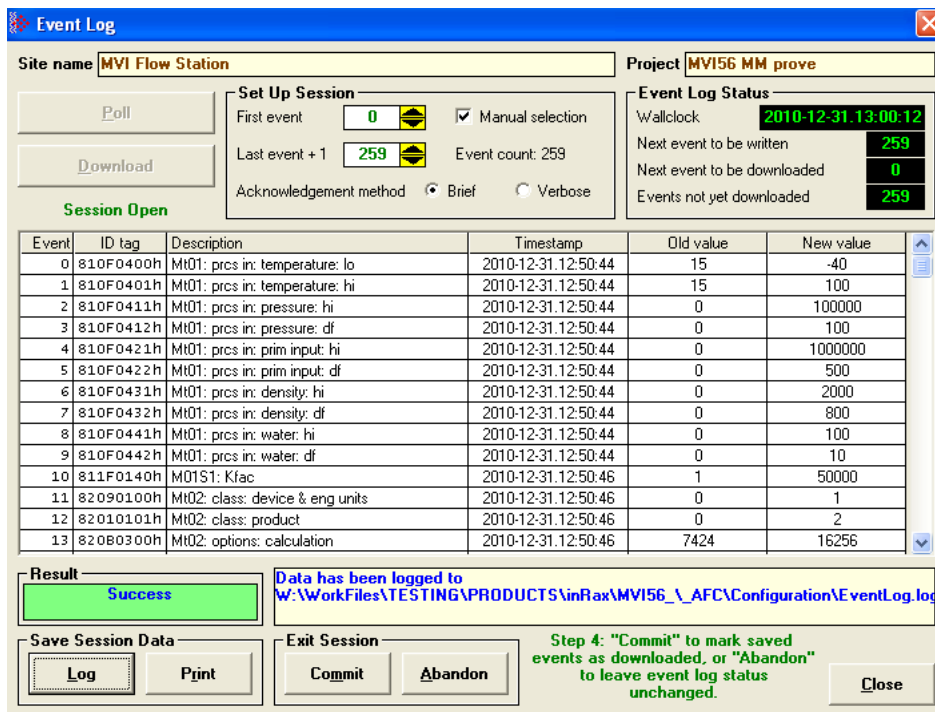


Committing a download tells the AFC module to *purge* the downloaded events, that is, to move its download pointer past the downloaded events and allow those events to be overwritten. Event log management standards mandate that downloaded events must be saved to more permanent storage before they can be purged from the module. In the earlier screenshot you can see that the **COMMIT** button is grayed out because the saving action has not yet been accomplished. AFC Manager considers the events to have been saved when they have been either logged to a file or printed.

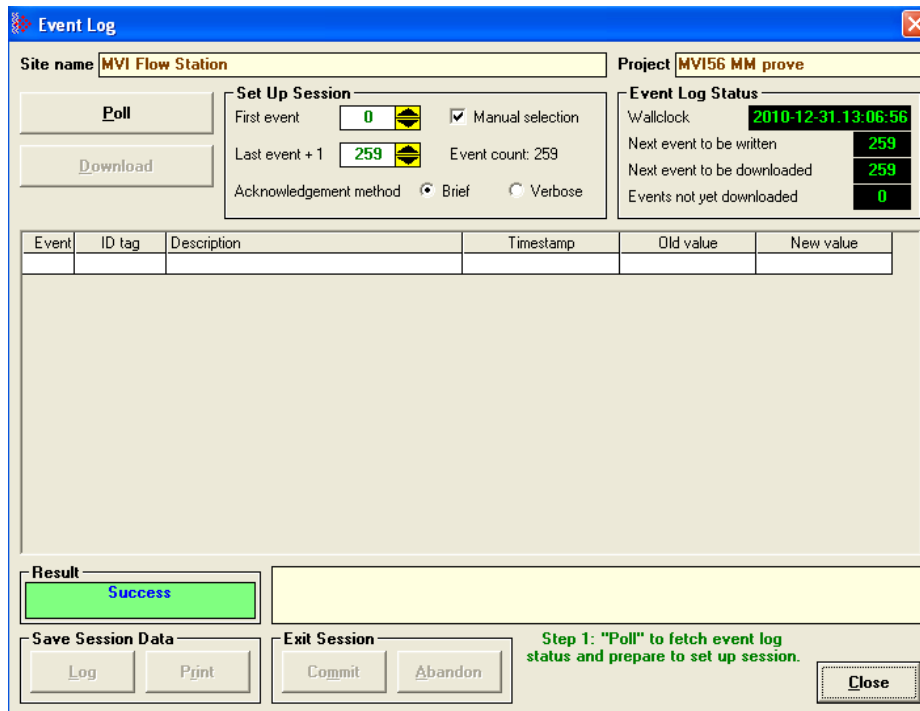
Click the **LOG** button and save the downloaded events to a text file of type ".log":



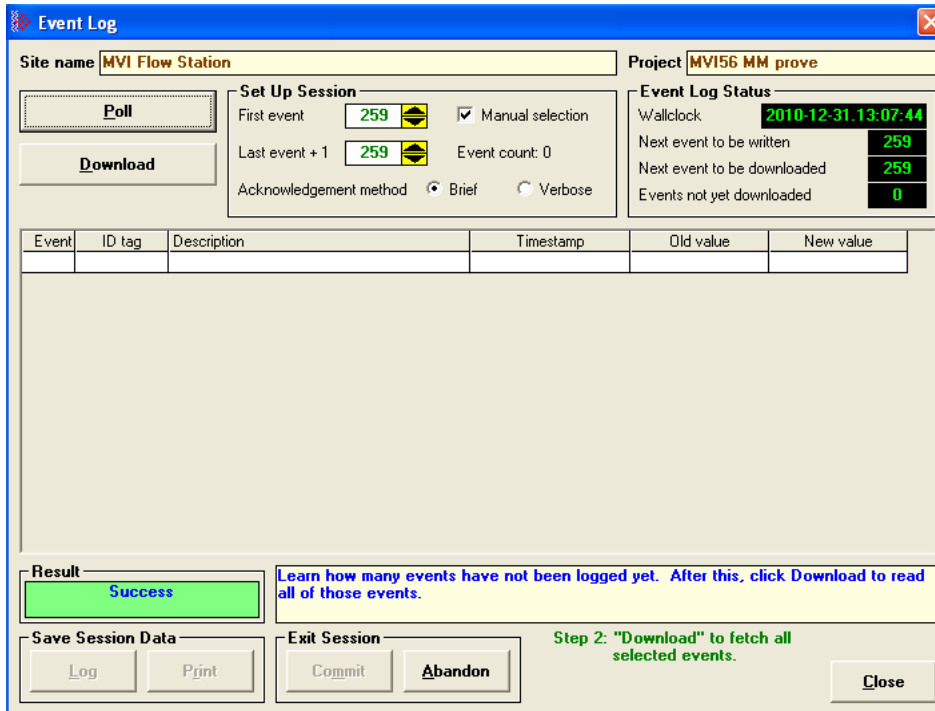
The **COMMIT** button is now enabled.



Click **COMMIT**, and the AFC then moves its download pointer (*Next event to be downloaded* in the *Event Log Status* panel):



To download again, click **POLL**. This will prompt you to start the next download starting from the new download pointer (*First event* in the *Set Up Session* panel). Once downloading all outstanding events is completed, there will be no more events to be downloaded as shown here by a 0 in the *Events not yet downloaded* field



For AFC firmware versions prior to 2.07, *Event Log Download* uses this same interface, but because the new firmware implementation is not present some features are not available or behave slightly differently. However, the same basic sequence of *poll*, *download*, and *commit/abandon* is the same.

9.5.1 Basic Principles of Implementation

A properly committed log-download session has three phases:

Phases

Setup Phase

This phase opens the session by a single Modbus transaction to the AFC that specifies the first event to be downloaded in the session. This event may be identified explicitly by event number, or it may be conveniently identified as the earliest event not yet previously downloaded.

Download Phase

This phase transfers event data during an open session by a sequence of zero or more fetch-and-acknowledge (F&A) cycles. A first Modbus transaction reads a short block of consecutive chronologically-ordered events from the AFC, and a second transaction writes sufficient data to inform the AFC that the events just read have been received without error; this pair of Modbus transactions constitutes a single F&A cycle. The starting event of the first F&A cycle is the event identified in the Setup phase. Each successful F&A cycle causes the subsequent cycle, if any, to begin with the earliest event not included in the previous cycle. The block of data read from the AFC includes, besides the events themselves, information that identifies the first event in the block, the number of events in the block, and the number of not-yet-downloaded events that remain in the AFC following those in the block. A failed F&A cycle does not cause the starting event to change, so it may be repeated as many times as is necessary to accomplish success. All F&A Modbus transactions access a block of registers beginning at the same address.

Completion Phase

This phase closes the session by a single Modbus transaction that updates the download point in the event log header with the earliest non-downloaded event and posts the Download event.

Dynamic Context

In order for the AFC to manage the log-download session, it maintains a "dynamic context" for the session, which includes in particular the number of the earliest event to be retrieved by the next F&A cycle. The dynamic context exists only while the session is open; the Setup phase opens the session and creates the context, and the Completion phase closes the session and discards the context. The dynamic context must not persist (the session remain open) indefinitely. Therefore, in case that a session is abandoned before completion, or otherwise fails due to problems such as loss of communication, a timeout is implemented that automatically abandons the session (closes it without completion) after a short period of inactivity; all Modbus transactions for the session must occur at a rate faster than this timeout for the session to be completed successfully.

One dynamic context is maintained for each port, permitting multiple hosts to perform download sessions simultaneously; see Section 8 for more on this. The backplane is deemed to be "port #0", so that a session may also be performed by the PLC via the Modbus Gateway feature.

To ensure maximum possible security, strict constraints are placed on the sequencing, addressing, and lengths of all Modbus transactions for the session. Violation of these constraints causes the offending transaction, depending on its nature, to be either rejected with a Modbus exception code or accepted but ignored; in no case does an offending transaction cause change of state in the AFC.

9.5.2 Data Elements

Modbus Points

Point in Modbus Table	Mandatory or Optional	Value	Description
slave selection	Mandatory	P	Primary Slave
		V	Virtual Slave
register bank	Mandatory	h	holding registers
		i	input registers
register address	Mandatory	nnnnn or, +nnn	A register address has the form "nnnnn" which is the 5-digit offset (0-based) of the register in the selected bank; a register address may also have the form "+nnn" where "nnn" is the 3-digit offset (0-based) of the register in a block of registers which block is located more globally by accompanying text.
byte selection	Optional	L	low-order byte
		H	high-order byte
bit number	Optional	/nn	Bit number, if present, is "/nn" where "nn" is the number of the bit in the 16-bit register or 8-bit byte, with bits numbered from 0 at the low-order end of the register or byte.

- "Ph00146" identifies the 16-bit register at offset 146 in the Primary Slave's holding register bank.
- "Ph00006H/01" identifies bit 1 of the high-order byte of Primary Slave holding register 6; this byte is already allocated in the AFC as the Site Extended Status byte. This bit can be alternately addressed as "Ph00006/09", but the AFC internally treats the two bytes as distinct.
- "Ph00200/05" identifies bit 5 of Primary Slave holding register 200; this bit is an as-yet unallocated bit in the Site Signals register.
- "LDW+000" identifies the first of a block of registers in the Primary Slave holding register bank and optionally in the Virtual Slave holding register bank, which block's absolute location is determined according to accompanying text.

Primary Slave Elements

These elements comprise configuration and status items required for the implementation, as well as the Log-Download Window (LDW) itself. If a host must access only the Virtual Slave for a log-download session, these points can be mapped to the Virtual Slave using the existing mapping functionality. As accesses to the LDW are severely restricted (every Modbus transaction to the LDW must be addressed to offset 0 of the LDW), a shortcut mapping is available that maps the entire LDW with a single data point.

Log-Download Window (LDW) Allocation

This is a block of 116 registers beginning at Primary Slave holding register 65400 and having a predefined layout. See Section 6 for the layout of this window. See Section 4 for how this window, when mapped to the Virtual Slave, is coordinated with other components of the Virtual Slave.

Site Configuration Items

- 1 Event Log Download Session Timeout Configuration word at Ph00146. This timeout is in seconds, which must be at least 5 and may not exceed 300 (5 minutes). Default value is 60 (1 minute).
- 2 Virtual Slave LDW Location Configuration word at Ph00147. This is the address in the Virtual Slave's holding register bank of the first register of the LDW. This value need not be limited to the 9900-register range of the mappable Virtual Slave but may be located anywhere in the Virtual Slave's address space from register 100 up to 65420 (which address places the last register of the LDW, at offset 115, at Virtual Slave register 65535). If this value is 0, the LDW is not available in the Virtual Slave (a conventionally mapped LDW is not visible).

Status

Event Log Download Active Session Map

Byte at Ph00007H (same word as Ph00007L, "PLC Offline Code").

This byte, previously unallocated, is a bitmap of the ports that currently have event-log download sessions active. Bits are numbered by the 1-based port number (where "Port #0" is the backplane), so that, for example, if a session is active on Port #2, the middle of the three front-panel ports, bit 2 of Ph00007H is set. The relevant bit is set when a download session is opened by the Setup phase, and is cleared when the session is committed by the Completion phase or when it is abandoned by timeout or explicit command. This byte Ph00007H is already available in the premapped portion of the Virtual Slave at Vh00007H.

9.5.3 Virtual Slave Precedence Relations

Addressing conflicts (collisions) can arise among three distinct regions of the Virtual Slave holding register bank. These regions are:

- The defined range of 9900 registers to which Primary Slave registers may be mapped.
- The word region of the pass-thru configuration, significant for Modbus write commands only.
- The LDW, specified in this document, when mapped to the Virtual Slave via Ph00147.

An addressing collision arises when the address of a holding register falls within more than one region. The AFC resolves such collisions as follows:

Each address is deemed to fall into one of (i) the pass-thru word region, (ii) the LDW, or (iii) the 9900-register defined range, whichever occurs first.

All Virtual Slave registers addressed by a single Modbus transaction must reside in the same region; no region-spanning is permitted. A region-spanning transaction is rejected with Modbus exception code 2, Illegal Address.

9.5.4 Security and Optimization

Two features are available that can improve security and throughput of a log-download session. These are:

- 1 **Session ID** - This is a value between 0 and 255 that is chosen by the host at the opening of the session and must be used in all transactions of the session. While a session is open, transactions that do not supply the correct Session ID are rejected. The AFC never displays the ID of the open session, so only the initiating host knows its value.
- 2 **Collapsed Acknowledgement** - This allows the Acknowledgement transaction of one F&A cycle to be embedded in the Fetch transaction of the next, reducing by almost half the number of transactions required for the session.

Use of either of these features, excepting only the use of Session ID 0, requires non-standard tweaking of the Modbus command packets of Fetch transactions, hence applications that cannot perform such tweaking are limited to the use of Session ID 0 and the non-Collapsed acknowledgement methods.

All session transactions except Fetch are Modbus writes; in those, the Session ID is included as an element of the written data. Fetch transactions are Modbus reads, which contain no data in their command packets; in those, the Session ID and (if used) the Collapsed Acknowledgement are encoded in the high-order 9 bits of the "number of registers" field, so that:

Bits 0 thru 6 contain the true "number of registers", which 7 bits are, for holding register access, sufficient to carry any value that is valid according to the standard Modbus protocol.

Bit 7 is used for the Collapsed Acknowledgement.

Bits 8 thru 15 contain the Session ID.

The tweaking of the Modbus read command packet is performed before calculation of the CRC or LRC. See Sections 6 and 7 for more detail.

All Modbus transactions of a log-download session, whether read or write, address offset 0 of the LDW, either to its Primary Slave location of 65400 or to its Virtual Slave mapped location configured by Ph00147, hence validation of Session ID, and recognition of tweaked Modbus command packets, are performed only for commands that address that location. Any attempt to address any other offset into the LDW is rejected with a Modbus exception code.

9.5.5 The Log-Download Window (LDW)

The LDW, located at Primary Slave holding register 65400 and optionally mapped to the Virtual Slave by register Ph00147 consists of a header of 4 registers followed by up to fourteen 8-register event records. Accordingly, the size of the block may be as large as $(4+14*8)$ registers, that is, 116. The block is returned as the data in the Fetch transaction of a F&A cycle. Certain subsets of the block may be read or written at other times.

Layout

The header has this layout:

1 LDW+000: Function and Session ID

When written, this register is interpreted as two bytes, where:

The low-order byte contains a function code:

0: Acknowledge Fetch

1: Open Session

2: Commit Session

3: Abandon Session

All other function codes are rejected with Modbus exception 3, "Illegal Data".

The high-order byte contains the Session ID. This ID is specified by the host upon opening the session and it must be matched by all subsequent accesses during the session.

When read, this register is always zero.

LDW+001: Starting Event Number

This value is initialized to the number of the event selected by the Setup phase. After every successful F&A cycle, it is advanced by the number of events fetched by that cycle.

LDW+002: Number of Events This Block

This is the number of events returned by the Fetch transaction of a F&A cycle. It never exceeds the number of events that can fit inside the size of the data block requested, but it may be smaller, such as when fewer events are available than the number requested. In the latter case, the extra unreturned event records are all zero.

LDW+003: Number of Non-Downloaded Events After This Block

This is the number of events remaining in the log that have not yet been downloaded, not counting the events in the current Fetch.

9.5.6 Modbus Transaction Sequencing and Constraints

This section describes the details of the Modbus transactions that manage a log-download session, including full specifications of transaction contents and the conditions under which they are permitted. Also considered is smooth recovery from failed transactions, as might happen over an intermittently failing communication link; the general principle is that an otherwise valid transaction may be repeated as many times as is necessary to ensure success.

Setup Phase

This phase, which opens a download session, may be accomplished in one of two ways:

The Detailed Method

Issue a Modbus write of two registers to offset 0 of the LDW, specifying function *Open Session* (1) and the desired Session ID followed by the number of the desired starting event.

The Quick Method

Issue a Modbus write of a single register to offset 0 of the LDW, specifying function *Open Session* (1) and the desired Session ID. This is equivalent to the Detailed method in which the starting event is copied from the download pointer in the event log header.

The dynamic context for the session includes two event numbers: one is the *Session Download Pointer* (SDP), which is the number of the first event to be retrieved by the next F&A cycle, and the other is the *File Download Pointer* (FDP), which is the number of the earliest event on file that has not yet been downloaded. The Setup phase specifies, explicitly or implicitly, the SDP, and initializes the FDP from the download pointer in the event log header, so the SDP and FDP need not be the same number.

Download Phase

This phase performs the actual retrieval of logged events as a sequence of F&A cycles. All transactions must place the Session ID into the first register of data (writes), or into the "number of registers" field of the Modbus command packet (reads), as described in Section 5.

Fetch Transaction

Issue a Modbus holding-register read (function 3) addressed to offset 0 of the LDW and with length calculated as $(4+n*8)$, where n is the number of events to be read and must lie between 0 and 14 inclusive; any other length constitutes an error. Note that a read of 0 events is permitted, so that the LDW header can be inspected without changing the session's dynamic context; however, such a fetch must still be acknowledged in the manner described next. The response is a block of the requested length formatted as described in Section 6, whose second register at offset 1 is the current value of the SDP.

Acknowledge Transaction

There are four methods of acknowledgement, three of which may be used at any time, and a session need not use any one consistently, even when repeating an acknowledgement that has apparently failed:

- 1 Collapsed method** - This method embeds the acknowledgement of the previous Fetch transaction into the next Fetch transaction, as described in Section 5.A Collapsed Acknowledgement bit value of 1 acknowledges the previous Fetch; a 0, if the previous Fetch has not been explicitly acknowledged by one of the other methods, elicits a repeat of the previous block of events. The Collapsed Acknowledgement bit of the first Fetch of a session must be 0.
- 2 Brief method** - Issue a Modbus write of a single register to offset 0 of the LDW, specifying function "Acknowledge Fetch" (0) with the correct Session ID. Use this method to conserve bandwidth when use of the Collapsed method is not possible.
- 3 Verbose method** - Issue a Modbus write of $(4+n*8)$ registers to offset 0 of the LDW, that echoes the complete data block read by the Fetch transaction except for insertion of the correct Session ID. The AFC verifies that all register values, except those at offsets 0 and 3 of the LDW header, are the same as were transmitted. Use this method for greater confidence of acknowledgement when bandwidth is less of a concern.
- 4 Implicit method** - The final Fetch transaction of a session can be implicitly acknowledged by the Completion phase (7.3, next). Because of the potential for undetected data corruption with the LRC of ASCII mode, only the Verbose method is recommended for an ASCII-mode Modbus channel.

A successful F&A cycle adjusts the session's dynamic context as follows:

- A** The SDP is advanced by the number of events returned by the fetch transaction.
- B** If at any time the SDP reaches the FDP, the FDP becomes "locked" to the SDP, thereafter tracking the SDP so that it keeps the same value, until the end of the session.

This ensures that any update of the download pointer in the event log header during Completion (7.3, next) is done only when it is guaranteed that all newly downloaded events have been retrieved by the host.

Completion Phase

This phase commits a session by determining a final FDP for the event log header, closing the session, and discarding the dynamic context. If the value of the final FDP differs from its original value in the event log header, then the header is updated with the new value and, provided that Site Option "Event Log Locked" is set, the Download event is written. If the FDP has not changed, no Download event is written, so the download state of the log remains unchanged. The session is then closed and its dynamic context discarded. The final FDP is determined by one of two methods:

- 1 Implicit Completion:** Issue a Modbus write of a single register to offset 0 of the LDW, specifying function "Commit Session" (2) with the correct Session ID. This method takes the final FDP from the dynamic context at the moment of completion.
- 2 Explicit Completion:** Issue a Modbus write of two registers to offset 0 of the LDW, the same as Implicit Completion but passing in the second register the desired final FDP as an explicit value. If at any time during the session the FDP of the dynamic context reached this value, this value becomes the final FDP for the session completion, in preference to that of the dynamic context. If the explicit FDP was not reached, this phase is equivalent to session abandonment (next). An Explicit Completion never marks more events as "downloaded" than an Implicit Completion.

A Completion of either method can implicitly acknowledge the final Fetch transaction.

Abandonment

This action explicitly abandons a session by closing it and discarding the dynamic context, without updating the log header or writing a Download event. In this case, therefore, adjustment of the log's download state implied by a changed FDP is not performed. It is equivalent to waiting for the session timeout to occur except that its effect is immediate. Issue a Modbus write of a single register to offset 0 of the LDW, specifying function "Abandon Session" (3) with the correct Session ID.

Error Recovery

During a session, the AFC maintains sufficient context information to accept not only the next expected Modbus transaction of the sequence but also a repeat of the previous one, except for the *Commit Session* and *Abandon Session* actions (see below). This is to permit simple repetition of a transaction that has succeeded from the point of view of the AFC but has failed from that of the host, which would occur when the Modbus response transmitted by the AFC is not received by the host. (If the host uses Collapsed Acknowledgement, it must take care to set the acknowledgement bit correctly in a repeated Fetch.) Other than in such a case, any Modbus transaction that does not comply strictly with the conditions and sequencing described above is rejected with a Modbus exception code (typically code 2, Illegal Address, if not addressing LDW offset 0, otherwise code 3, Illegal Data).

Because both the *Commit Session* and the *Abandon Session* actions close the session, there is afterwards no session context to allow recognition of a repeated *Commit* or *Abandon*. In this case, a repeated *Commit* elicits a Modbus exception of 3, *Illegal Data*, and a repeated *Abandon* is accepted without error regardless of its Session ID.

Session Timeout

All transactions of a session must succeed frequently enough so that the duration between successful transactions does not exceed the session timeout. Each successful transaction restarts the timeout. If the timeout expires, the session is automatically abandoned. Other Modbus activity, unrelated to the log-download session, is not considered, and its only effect upon the timeout would be delays of session transactions imposed by bandwidth usage.

9.5.7 Access by Multiple Hosts

The functionality specified in this document can permit complete event-log retrieval by multiple hosts, provided that these conditions are satisfied:

- 1** As the session's dynamic context is local to the accessed port, multiple hosts may perform sessions simultaneously provided that they access separate ports. The Session ID is part of the dynamic context, so separate-port sessions may use the same Session ID without ambiguity.
- 2** For multiple hosts that access the same port (using Modbus Master arbitration or a similar scheme), all must perform their sessions at times sufficiently separated so that one host does not interfere by disturbing the dynamic context of another host's session in an unpredictable manner. The Session ID can provide significant protection against inadvertent infringement of this condition.
- 3** One host must be the Active host, performing the Completion phase that updates the AFC's event log state (download pointer). All other hosts must be Passive, failing to Complete their sessions but instead Abandoning them. If this condition is disregarded, so that multiple Active hosts perform simultaneous sessions each ending with the Completion phase, the AFC's Event Log, which is global, manages any updating of the download pointer and posting of the Download event in a globally consistent manner, but each host cannot be sure that the Download event written upon Completion, if any, is exactly what it expected.
- 4** Each host must, in one way or another, have access to its own long-term download context, which is the number of the earliest event not yet downloaded by that host. All Passive hosts must maintain this context locally. The Active host may let the AFC maintain its long-term context, using the download pointer in the event log header for this purpose; in such a case the same host must always be the Active one. If, however, each host regardless of role maintains its own long-term context, the role of Active host may be passed around among hosts.
- 5** All hosts must perform download sessions sufficiently often so that events are not lost by being overwritten by newer ones before those events have been downloaded by that host.

9.5.8 Other Considerations

Expired Events

If the starting event number is sufficiently small relative to the events on file (in particular, if it is 1999 or more before the number of the next event to be recorded), then that event is no longer on file and has already been overwritten. In this case, the event returned by a Fetch is all zero. This is not an issue for the Active host, especially when the event log is configured to be *locked*, as the Active host is interested only in non-downloaded events and those always remain on file with a *locked* log. But for Passive hosts, and when the event log is configured as *unlocked*, download sessions must be performed frequently enough so that the requested events still remain on file. Because event numbers wrap from 65535 to 0, and because events that have not yet been written do not in fact exist and have never existed, an event number that is numerically equal to or greater than the number of the next event to be recorded is deemed to be the number of an event from the previous wrap cycle.

Persistence

A log-download session does not survive a reset of the module (e.g. power cycle).

9.6 Period-end Events

A "period-end" event is detected by the wallclock. There are two such:

- a) "End-of-hour" occurs when the minute of the hour steps into the "End-of-hour minute" of Site Configuration.
- b) "End-of-day" occurs when the minute of the day steps into the "End-of-day minute" of Site Configuration.

A wallclock change that skips forward over an end-of-period minute will cause that period-end to be missed, and a change that skips backward over that minute will cause that period-end to be repeated, so wallclock adjustments should be performed at times well-removed from either end-of-period minute.

Though a period-end event is not recorded in the event log, it does cause archives and resets to be scheduled for all enabled meters according to their configured "period-end" Archive Options. Archives and resets scheduled by period-end are delayed in their action until at least one meter scan has occurred after the event (the archive data accumulation that takes place at the end of the meter scan also records the latest timestamp, so the written archive then reflects the fact that the period-end has occurred).

9.7 Loggable Events

The tables below give full details of all events that are recorded in the Event Log. For the Special events (page 141), columns are:

Tag	Numeric value that identifies the event.
Rkv	Effect on archives and accumulator resets (see next).
Description	Lists: The event name, identifying its triggering condition. Contents and meaning of the old and new value fields. Relevant additional information.

For the Data Point (page 143, page 142, page 146) events, columns are:

Group	Group code.
Sbgp	Subgroup code.
Item	Item code.
Dtyp	Datum type code (mnemonic).
Rkv	Effect on archives and accumulator resets (see next).
Datum point	The corresponding writable Modbus point.

In these tables, the "Rkv" columns specify how archives and accumulator resets are scheduled upon occurrence of the corresponding loggable events.

Column values are:

Value	Meaning
*	Upon this event archives and resets are scheduled according to the configured "event" Archive Options, provided that the applicable meter(s) is(are) enabled. Applicable meters depend upon the event class: (a) Special (non-meter-specific) and Site Datum Point events: All meters. (b) Meter events (including meter-specific Specials): The addressed meter. (c) Stream events: The addressed meter, provided that the addressed stream is active. Scheduled archives are always written before completing any change to data or module state implied by the event; this ensures that the data contributing to an archive is limited to that which was available before the event.
-	This event has no effect on archives and resets.
(n)	Upon this event archives and resets are scheduled as for "*", modified by the conditions and actions given in "Note (n)" in "Rkv" notes (page 151).

9.8 Special Events

Tag	Rkv	Description
0	-	<p>Never Used</p> <p>Value: Always 0.</p> <p>Notes: This entry in the Event Log has never been written.</p> <p>The number of such entries starts at 1999 upon cold start and decreases as events are written until none remain, after which oldest events are overwritten with new ones.</p>
1	-	<p>Event Log Download</p> <p>Value: Number of last-downloaded event.</p> <p>Notes: Triggered by a purge of the Event Log, which marks older events as available to be overwritten by new ones.</p>
2	-	<p>Cold Start</p> <p>Value: Always 0.</p> <p>Notes: This event is obsolete and is never written.</p>
3	(1)	<p>Power-Up</p> <p>Value: "Old" value is the last-saved wallclock from the previous session; "new" value is always 0 (clock not yet set).</p> <p>Notes: The last event written upon restart of the application and before entering the meter scan. This event may be preceded by Checksum Alarm and/or PLC Mode Change events.</p>
4	-	<p>PLC Mode Change</p> <p>Value: PLC mode (0 on line, 1 off line).</p> <p>Notes: Logs changes to PLC connectivity as reported by the backplane procedures. Typically caused by switching the PLC between "run" and "program" modes.</p>
5	-	<p>Checksum Alarm</p> <p>Value: Checksum alarm word (datum type "Bsht").</p> <p>Notes: Logs changes to the checksum alarm bitmaps.</p> <p>Includes site/meter identification (bits 28-24).</p> <p>Upon power-up: Written automatically upon power up when a checksum failure is detected. In this case the event is written even if the bitmap does not change, such as when an affected bit is already set from a previous failure that was never cleared.</p> <p>Upon Modbus write to the bitmap: Records changes to the bitmap only, typically when clearing bits, though setting bits is also permitted.</p>
6	(2)	<p>Wallclock Change</p> <p>Value: Wallclock (packed bitfields).</p> <p>Notes: Triggered when the wallclock is set for the first time, or when it is reset to a value that differs from its current value by five minutes or more. These two cases can be distinguished by the "old value" in the event entry: for the initial setting this value is zero ("clock not set").</p>
7	*	<p>Stream Select</p> <p>Value: Stream number.</p> <p>Notes: Triggered by a "select active stream" meter signal.</p> <p>Includes meter identification (bits 28-24).</p>

9.9 Site Data Point Events

Group	Sbgp	Item	DTyp	Rkv	Data point
0	0	0	Bsht	(3)	Site options
1					Site parameter value
	0	0	Flot	*	Barometric pressure
8	n	0	Usht	-	Arbitrary event-logged value "n" ("n" = 0 thru 9)
15					PLC image address (Quantum platform only)
	0	0	Usht	*	Supervisory, get
	1	0	Usht	*	Supervisory, put
	2	0	Usht	*	Wallclock, get & put
	3	0	Usht	*	Modbus gateway, get & put
	4	0	Usht	*	Modbus pass-thru, put
	5	0	Usht	*	Modbus master, get & put

9.10 Meter Data Point Events

Group	Sbgp	Item	DTyp	Rkv	Data point
0	0				Process input calibration
		0	Flot	*	Temperature
		1	Flot	*	Pressure
		2	Flot	*	Primary input
		3	Flot	*	Flowing density
		4	Flot	*	Water content
0	1				Process input alarm
		0	Flot	-	Temperature range
		1	Flot	-	Pressure range
		2	Flot	-	Primary input range
		3	Flot	-	Flowing density range
		4	Flot	-	Water content range
1	0				Meter classification
		0	Bsht	*	Meter device and engineering units
		1	Usht	*	Product group
2					Reference conditions
	0	0	Flot	*	Temperature
	1	0	Flot	*	Pressure
3					Meter options
	0	0	BIng	*	Calculation options
	1	0	BIng	(4)	Control options
4					Input scaling
	0				Temperature
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module id code
	1				Pressure
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module id code
	2				Primary input
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module id code
	3				Flowing density
		0	Flot	*	Range low end

Group	Sbgp	Item	DTyp	Rkv	Data point
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module id code
	4				Water content
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module id code
5	0	0	Bm24	*	Analysis component selection map
6	0	0	Ulng	*	Pulse input rollover
7					Units
	0	0	B448	*	Primary input (period, quantity, units)
	1	0	Ubyt	*	Mass flow rate period
	2	0	Ubyt	*	Mass flow rate units
	3	0	Ubyt	*	Mass accumulator units
	4	0	Ubyt	*	Energy flow rate period
	5	0	Ubyt	*	Energy flow rate units
	6	0	Ubyt	*	Energy accumulator units
	7	0	Ubyt	*	Volume flow rates period
	8	0	Ubyt	*	Volume flow rates units
	9	0	Ubyt	*	Volume accumulators units
8					Accumulator rollovers
	0	0	Ulng	*	Mass
	1	0	Ulng	*	Energy
	2	0	Ulng	*	Volumes
9					Meter parameter value
	0	0	Flot	*	Orifice plate measurement temperature
	1	0	Flot	*	Orifice plate measured diameter
	2	0	Flot	*	Orifice plate coefficient of thermal expansion
	3	0	Flot	*	Meter tube measurement temperature
	4	0	Flot	*	Meter tube measured diameter
	5	0	Flot	*	Meter tube coefficient of thermal expansion
	6	0	Flot	*	Primary input flow threshold
	7	0	Flot	*	Primary input alarm threshold
	8	0	Flot	*	V-cone/Wedge coefficient of discharge
10					[reserved]
11	0				Densitometer
		0	Usht	*	Densitometer type
		1	Flot	*	Calibration temperature
		2	Flot	*	Calibration pressure

Group	Sbgp	Item	DTyp	Rkv	Data point
		3	Flot	*	Calibration constant K0
		4	Flot	*	Calibration constant K1
		5	Flot	*	Calibration constant K2
		6	Flot	*	Calibration constant 6
		7	Flot	*	Calibration constant 7
		8	Flot	*	Calibration constant 8
		9	Flot	*	Calibration constant 9
		10	Flot	*	Calibration constant 10
		11	Flot	*	Calibration constant 11
		12	Flot	*	Calibration constant 12
		13	Flot	*	Calibration constant 13
		14	Flot	*	Calibration constant 14
		15			PLC image address (Quantum platform only)
	0	0	Usht	*	Meter process input &c, get
	1	0	Usht	*	Meter results, put
	2	0	Usht	*	Meter archive fetch, put

9.11 Stream Data Point Events

Group	Sbgp	Item	DTyp	Rkv	Data point
0	0	0	Bsht	*	Stream options
1					Stream parameter value
	0	0	Flot	*	Default relative density (gas) at reference
	1	0	Flot	*	Viscosity
	2	0	Flot	*	Isentropic exponent
	3	0	Flot	*	Default Fpv
	4	0	Flot	*	K/meter factor
	5	0	Flot	*	Default energy content
	6	0	Flot	*	Default reference density (liquid)
	7	0	Flot	*	Default vapor pressure
	8	0	Flot	*	Water density at API reference
	9	0	Flot	*	Default Ctl
	10	0	Flot	*	Default Cpl
	11	0	Flot	*	Shrinkage factor
	12	0	Flot	*	Precalculated alpha
2	0				Meter factor curve
		0	Flot	*	Datum point 1, meter factor
		1	Flot	*	Datum point 1, flow rate
		2	Flot	*	Datum point 2, meter factor
		3	Flot	*	Datum point 2, flow rate
		4	Flot	*	Datum point 3, meter factor
		5	Flot	*	Datum point 3, flow rate
		6	Flot	*	Datum point 4, meter factor
		7	Flot	*	Datum point 4, flow rate
		8	Flot	*	Datum point 5, meter factor
		9	Flot	*	Datum point 5, flow rate
3	0				Analysis mole fraction
					** Because the item code extends into the subgroup field, this can be the only subgroup of group 3 ! (Pending any future reformat of the Event Id Tag)
		0	Usht	(5)	Component 1, scaled molar fraction
		1	Usht	(5)	Component 2, scaled molar fraction
		2	Usht	(5)	Component 3, scaled molar fraction
		3	Usht	(5)	Component 4, scaled molar fraction
		4	Usht	(5)	Component 5, scaled molar fraction
		5	Usht	(5)	Component 6, scaled molar fraction
		6	Usht	(5)	Component 7, scaled molar fraction
		7	Usht	(5)	Component 8, scaled molar fraction
		8	Usht	(5)	Component 9, scaled molar fraction
		9	Usht	(5)	Component 10, scaled molar fraction

Group	Sbgp	Item	DTyp	Rkv	Data point
		10	Usht	(5)	Component 11, scaled molar fraction
		11	Usht	(5)	Component 12, scaled molar fraction
		12	Usht	(5)	Component 13, scaled molar fraction
		13	Usht	(5)	Component 14, scaled molar fraction
		14	Usht	(5)	Component 15, scaled molar fraction
		15	Usht	(5)	Component 16, scaled molar fraction
		16	Usht	(5)	Component 17, scaled molar fraction
		17	Usht	(5)	Component 18, scaled molar fraction
		18	Usht	(5)	Component 19, scaled molar fraction
		19	Usht	(5)	Component 20, scaled molar fraction
		20	Usht	(5)	Component 21, scaled molar fraction
		21	Usht	(5)	Component 22, scaled molar fraction
		22	Usht	(5)	Component 23, scaled molar fraction
		23	Usht	(5)	Component 24, scaled molar fraction

9.12 Prover Data Point Events

Note: Currently, this function is only available on the MVI56- AFC. For all other platforms continue to use AFC Manager 2.05 or earlier.

Group	Sbgp	Item	DTyp	Rkv	Data point
0	0				Process input calibration ** not implemented
		0	Flot	-	(Inlet) temperature
		1	Flot	-	Outlet temperature
		2	Flot	-	Switch bar temperature
		3	Flot	-	(Inlet) pressure
		4	Flot	-	Outlet pressure
0	1				Process input alarm ** not implemented
		0	Flot	-	(Inlet) temperature range
		1	Flot	-	Outlet temperature range
		2	Flot	-	Switch bar temperature range
		3	Flot	-	(Inlet) pressure range
		4	Flot	-	Outlet pressure range
1	0				Prover classification
		0	Diby	-	Prover type, master meter number
		1	Diby	-	Measurement system, density selection
2	0	0	Bsht	-	Prover options
3	0				Prover run counts
		0	Usht	-	Runs per prove, total
		1	Usht	-	Runs per prove, selected
		2	Usht	-	Max total runs before abort
		3	Usht	-	Passes per run (short prover)
		4	Usht	-	Min pulses per run * 1000
		5	Usht	-	Max seconds per run
4					Prover input format codes
	0	0	Usht	-	Pulse count for run/pass
	1	0	Usht	-	Master meter pulse count
5					Prover reference conditions
	0	0	Flot	*	Temperature
	1	0	Flot	*	Pressure
6					Prover parameter value
	0	0	Flot	-	Meter factor precision
	1	0	Flot	-	Pulse interpolation ratio
	2	0	Flot	-	Flow tube linear thermal expansion coefficient
	3	0	Flot	-	Switch bar linear thermal expansion coefficient
	4	0	Flot	-	Calibrated prover volume

Group	Sbgp	Item	DTyp	Rkv	Data point
	5	0	Flot	-	Flow tube inside diameter
	6	0	Flot	-	Flow tube wall thickness
	7	0	Flot	-	Flow tube modulus of elasticity
7	0				Prover variation limits
		0	Flot	-	Meter temperature
		1	Flot	-	Prover inlet temperature
		2	Flot	-	Prover outlet temperature
		3	Flot	-	Prover inlet-outlet temperature
		4	Flot	-	Prover temperature
		5	Flot	-	Prover-Meter temperature
		6	Flot	-	Switch bar temperature
		7	Flot	-	Meter pressure
		8	Flot	-	Prover inlet pressure
		9	Flot	-	Prover outlet pressure
		10	Flot	-	Prover inlet-outlet pressure
		11	Flot	-	Prover pressure
		12	Flot	-	Prover-meter pressure
		13	Flot	-	Density
		14	Flot	-	Water content
		15	Flot	-	Meter flow rate
		16	Flot	-	Prover flow rate
		17	Flot	-	Pulses over runs
		18	Flot	-	Pulses over passes
		19	Flot	-	Repeatability
		20	Flot	-	Change in factor
8					Prover process input scaling
	0				(Inlet) temperature
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module ID code
	1				Outlet temperature
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module ID codes
	2				Switch bar temperature
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default

Group	Sbgp	Item	DTyp	Rkv	Data point
		3	Sbyt	*	Module ID codes
3					(Inlet) pressure
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module ID codes
4					Outlet pressure
		0	Flot	*	Range low end
		1	Flot	*	Range high end
		2	Flot	*	Default
		3	Sbyt	*	Module ID codes
15					PLC image address (Quantum/Unity platform only)
	0	0	Usht	-	Prover, get
	1	0	Usht	-	Prover, put

9.13 "Rkv" Notes

- 1 Archives (only, not resets) are forced regardless of configuration, capturing any unarchived data from the previous session.
- 2 Archives and resets are scheduled (immediately, without a "period-end" delay) only for the initial setting of the wallclock; a "five-minute" event causes no scheduling. This ensures capture of any flow that has occurred prior to the initial clock-set.
- 3 Event occurs only when one or more of the following bits are changed:
 - Bit 2, "Barometric pressure units"
 - Bit 5, "Process input out of range use last good"
 - Bit 12, "Analysis is packed in module"
 - Bit 13, "Analysis is packed over backplane" (1756 and 1769 platforms only)
- 4 A change to Meter Control Options bit 15, "Meter enable", imposes these adjustments to the normally-scheduled archives/resets:
 - Upon meter enable, cancel any scheduled archives (no data yet to be archived), but leave in place any scheduled resets.
 - Upon meter disable, cancel any resets (for inspection and so on.; reset will be rescheduled upon subsequent enable), and force archiving of both files regardless of configuration (so that a disabled meter never has any pending unarchived data).
- 5 Events occur only if Meter Control Options bit 10, "Treat analysis as process input", is clear.

9.14 Downloading the Event Log in Firmware Version 2.05 and Earlier

For auditing purposes, each event has a "number" assigned sequentially, starting at 0 for the first event written and increasing up through 65535 then wrapping to 0 again.

An event record properly includes its event number along with the information listed in the preceding sections. To conserve space, and to make transmittal more efficient, the event number is not stored as part of the event record. Instead, the Event Log header contains sufficient information to calculate for any event its event number from the position of its record in the Log and vice versa.

The following procedures use these terms:

Term	Meaning
my_record	Known record position. Input to procedures (A) and (C)
event_number	Desired event number. Output from procedure (A).
Modbus_address	Desired Modbus address. Output from procedure (C).
my_event	Known event number. Input to procedure (B).
record_position	Desired record position. Output from procedure (B).
number_of_records	Maximum number of records. Contents of register 40000. In this version of the AFC "number_of_records" is 1999; however, to be compatible with future versions that may store a different number of events, an application should use the value from the header instead of a constant 1999.
next_record	Next new record position. Contents of register 40001.
next_event	Next new event number. Contents of register 40002.
oldest_event	Oldest event number on file. Contents of register 40003.
oldest_not_downloaded	Oldest event number not yet downloaded. Contents of register 40004.
events_on_file	Total number of events on file. Calculated. This value starts at 0 upon cold start, then, as events are logged, it rises to a maximum of "number_of_records" and stays there.
downloadable_event	Event number of event being downloaded. Calculated.
event_age	The age of the event in question. Calculated. The next event to be written (which of course is not yet on file) has age 0; the newest event already on file has age 1, the next older event has age 2, and so on up to age "number_of_records".

Also in these procedures:

- a) The expression "AND 0x0000FFFF" means "take the low-order 16 bits of the result, discarding all other higher-order bits"; it is equivalent to "(non-negative) remainder upon dividing by 65536" (A traditionally negative remainder that would result from dividing a negative dividend by 65536 must be made positive by subtracting its absolute value from 65536)
- b) The operator ":= " means "assignment"; that is, "assign" the expression on the right to the object on the left by calculating the value of the expression on the right and making the object on the left assume that value. The operator "==" means "is equal to".
- c) Words in all caps and the other arithmetic operators have their expected meanings.
- d) Text enclosed in brackets ("["]") are comments only.

Procedure (A): Calculate event number from record position.

- 1 Calculate number of events on file.**

```
events_on_file := ( next_event - oldest_event ) AND 0x0000FFFF
```

- 2 Determine whether desired record is on file.**

```
IF ( my_record < 0 OR my_record ≥ events_on_file ) THEN  
  [record is not on file]  
  EXIT this procedure
```

- 3 Calculate age of desired record.**

```
event_age := ( next_record - my_record )  
IF ( event_age ≤ 0 ) THEN  
  event_age := event_age + number_of_records
```

- 4 Calculate event number of desired record.**

```
event_number := ( next_event - event_age ) AND 0x0000FFFF
```

Procedure (B): Calculate record position from event number.

- 1 Calculate number of events on file.**

```
events_on_file := ( next_event - oldest_event ) AND 0x0000FFFF
```

- 2 Calculate age of desired event.**

```
event_age := ( next_event - my_event ) AND 0x0000FFFF
```

- 3 Determine whether desired event is on file.**

```
IF ( event_age == 0 OR event_age > events_on_file ) THEN  
  [event is not on file]  
  EXIT this procedure
```

- 4 Calculate record position of desired event.**

```
record_position := ( next_position - event_age )  
IF ( record_position < 0 ) THEN  
  record_position := record_position + number_of_records
```

Procedure (C): Calculate Modbus address of record from record position.

- 1 Calculate number of events on file.**

```
events_on_file := ( next_event - oldest_event ) AND 0x0000FFFF
```

- 2 Determine whether desired record is on file.**

```
IF ( my_record < 0 OR my_record ≥ events_on_file ) THEN  
  [record is not on file]  
  EXIT this procedure
```

3 Calculate Modbus address.

```
Modbus_address := ( my_record * 8 ) + 40008
```

Procedure (D): Download all events not yet downloaded.

The downloading application should download the entire Log, starting at the oldest event not yet downloaded and extending through all newer events.

1 Fetch event number of oldest event not yet downloaded.

```
downloadable_event := oldest_not_downloaded
```

2 Determine whether any more events remain to be downloaded.

```
IF ( downloadable_event == next_event ) THEN  
  [all events have been downloaded]  
  EXIT this procedure
```

3 Download this event.

a) Calculate record number.

```
my_event := downloadable_event  
record_position := { via Procedure (B) }
```

b) Calculate Modbus address.

```
my_record := record_position  
Modbus_address := { via Procedure (C) }
```

c) Download the event with Modbus.

```
Set Modbus Function Code := 4, Read Input Registers  
Set Modbus Number of Registers := 8  
Set Modbus Register Address := Modbus_address  
Execute  
Copy the returned data to permanent storage
```

4 Step to next event and loop.

```
downloadable_event := ( downloadable_event + 1 ) AND 0x0000FFFF  
GOTO step 2.
```

When the download is complete, and the downloaded events have been logged to disk, the AFC should be told of this fact by issuing the "download complete" Site Signal. This signal updates the header to show that all records have been downloaded, unlocking the Log for further events, and (if "Event log unlocked" is clear) posts a "download" event. A download may be performed at any time; it is not necessary to wait for the log-full condition in order to download.

An application that downloads the event log should explicitly include the event number in any copy of the event that it stores in its own database.

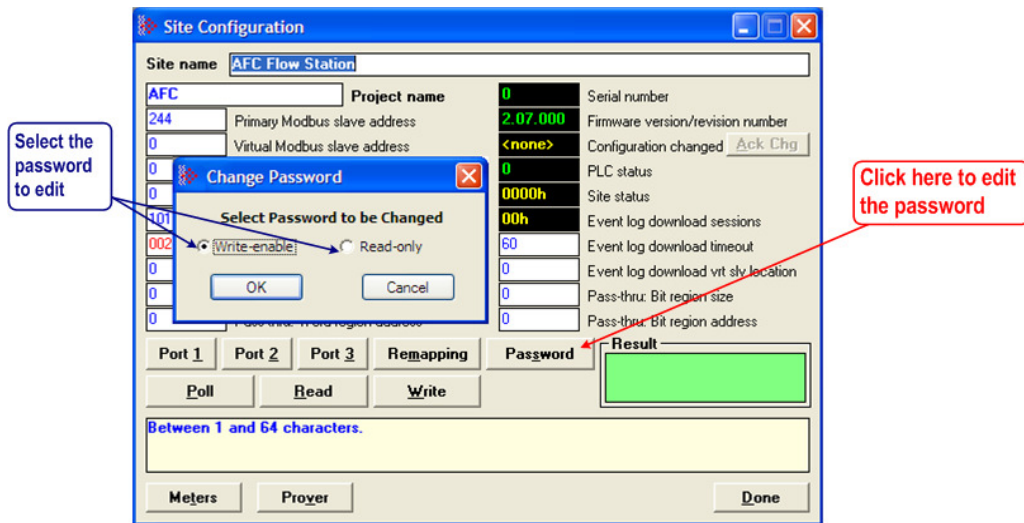
10 Security (Passwords)

In This Chapter

- ❖ Hard Password..... 157

The passwords are intended for interrogation by application software in order to verify an operator's authorization to make configuration changes and to view measurement results. The passwords are resident in the module so that different operators using different copies of the application software must use the same password. Passwords cannot be retrieved in "Hard Password" mode. The password protection is not used by default.

Passwords can be numbers between -32768 and 32767. For example, 1234. A password of 0 (zero) is interpreted as "No password present".



The module supports two passwords: Write-Enable and Read-Only. Each password is enabled when you write a non-zero value to the corresponding register.

Password	Holding Register Address	Description
Write-Enable	9	Protects the module from write operations from the AFC Manager
Read-Only	19	Protects the module from read or write operations from the AFC Manager

The following table shows how the passwords affect the AFC Manager operation depending on the values that you configure:

Protection Level	Read-Only Password	Write-Enable Password	Read Operation: Requires Authorization?	Write Operation: Requires Authorization?
No protection	Zero	Zero	No	No
Write Protection	Zero	Non-zero	No	Yes (Use Write-Enable password)
Read and Write Protection	Non-zero	Zero	Yes (Use Read-Only password)	Yes (Use Read Only password)
Read and Write Protection	Non-zero	Non-zero	Yes (Use Read-Only or Write-Enable password)	Yes (Use Write-Enable password)

Each port can be assigned to different password protection levels. Refer to the Port Configuration Section for more information about this topic.

10.1 Hard Password

The hard password feature offers further protection against unauthorized access to the module.

If the Hard Password option is cleared, these registers can be read either from an external Modbus device, from the processor or using the Modbus Master interface in the AFC Manager. This operation mode is called "Soft Password" mode. It is then the responsibility of a compatible application (such as AFC Manager) to verify the password given by the operator against those fetched from the module in order to determine the access granted.

If the Hard Password option is selected, a read of a password register will return zero regardless of the password's actual value. In this case, read or write access is obtained by writing a candidate password to the Password Test register (register 18), the module itself verifies the password, and the access granted is determined by reading back that same register 18 (called the Accessed Port and Authorization register when read) and examining its contents. The access is granted to the port over which the request was made; other ports remain unaffected. If the port remains idle with no Modbus activity for two minutes, then the granted access is removed and can be regained only by writing a new password to the test register. For highest security, you can explicitly revoke your own password-obtained authorization before it times out by writing zero to the Password Test register.

Access granted by password, whether Soft or Hard, is to the module as a whole, including the password registers themselves. That is, in order to change a stored Hard password you must first obtain write access to the module by giving the correct Write-Enable password. However, some registers are exempt from authorization. There are a very few registers that are exempt from write authorization and are always writable; the Password Test register 18 is one such for the obvious reason. Similarly, some registers are exempt from read authorization and are always readable; they include most of the first 20 holding registers, including the Firmware Product and Group codes in registers 0 and 1 (so an application like AFC Manager can learn whether it is talking to an AFC without being trapped in a catch-22), the Site Status in register 6 (so the application can learn whether the password mode is Soft or Hard and verify the operator's password entry using the proper method), and the Accessed Port and Authorization register 18 (so the application can learn whether access was granted in Hard-password mode even if the wrong read password was entered).

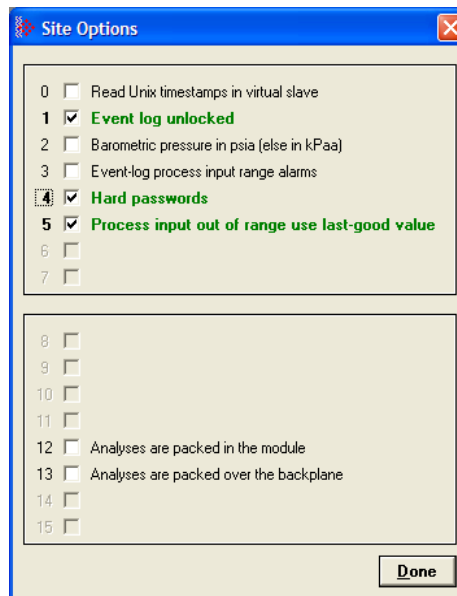
The Accessed Port and Authorization register is a bit-mapped word defined as follows:

Bits	Description
0 to 3	The number of the accessing port (0 for Modbus Gateway)
4	Read Authorization Waived
5	Write Authorization Waived
6	Read Access Granted
7	Write Access Granted
8 to 15	Reserved

A waived authorization means that password entry is not required for this action even if a non-zero password has been configured. Authorization waivers are configured separately for each port, so, for example, a SCADA system connected to port 2 can be allowed to read measurement results without having to supply a password while an operator connecting AFC Manager to port 1 still must enter the correct password. The backplane is always given both waivers, so the PLC never has to supply a password.

To set a hard password in AFC Manager:

- 1 Open the Site Configuration Dialog box
- 2 Click in the Site Options field. This action opens the Site Options dialog box
- 3 Select (check) option 4, Hard Passwords



When this option is selected, any authorization granted using Hard Passwords times out after two minutes of inactivity, and the user will be required to re-enter the password to continue.

11 MVI56-AFC Backplane Communication

In This Chapter

- ❖ MVI56-AFC Module Data Transfer 160
- ❖ MVI56-AFC Function Block Interface 167

11.1 MVI56-AFC Module Data Transfer

11.1.1 Input/Output Blocks for Data Transfer

When you configure the MVI56-AFC module to the I/O configuration of your ControlLogix controller, the tags of the Output Block Array (OBA) and the Input Block Array (IBA) for the module are automatically created as controller tags by the RSLogix5000 programming software.

Output Block Array

Size: 248 elements

Data type: Integer (2 bytes)

The OBA is used for transferring process variables from individual meter runs and other data that are available in the ControlLogix processor memory to your AFC module via the backplane. This array is made up of a block of 248 contiguous integer elements.

Element	Attribute
Local 3.O (slot number)	Local.O.Data[0]
	Local.O.Data[1]
	Local.O.Data[2]

	Local.O.Data[n]

	Local.O.Data[247]

The meter run process variables such as temperature, pressure, differential pressure, meter pulses, and so on. may come from other I/O modules in the rack or from other sources like an HMI and so on. These variables must be copied to predefined structured function blocks which transfer them to the AFC module as part of one or more OBAs.

Input Block Array

Size: 250 elements Data type: Integer (2 bytes)

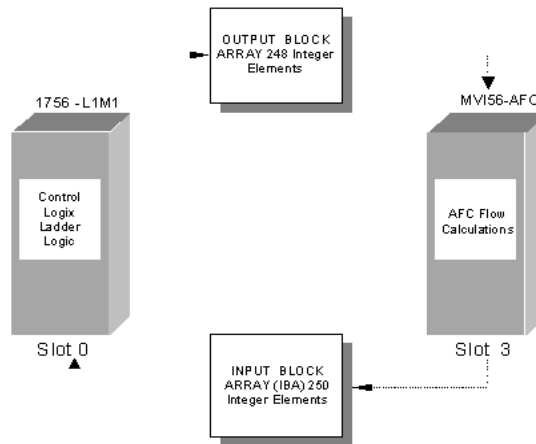
The AFC consumes the process variable inputs received in the OBAs and returns calculated variables such as process flowrate, accumulated product volume, and so on. by populating the IBA and returning it to the ControlLogix processor via the backplane. This array is made up of a block of 250 contiguous integer elements.

Element	Attribute
Local 3.I (slot number)	Local.I.Data[0]
	Local.I.Data[1]
	Local.I.Data[2]

	Local.I.Data[n]

	Local.I.Data[249]

The following section describes in detail how the MVI56-AFC module facilitates the two-way data transfer by conveniently partitioning the Input/Output blocks into smaller function blocks to allow flexibility in designing and implementing the logic to support the AFC module.



11.1.2 Input/Output Transactions

The MVI56-AFC module runs automatically on power up. It receives the process variables from the ControlLogix controller through the associated output block array and returns the calculated results back through the Input Block Array.

The data transfer to and from the MVI56-AFC module is implemented through processor logic, using the concept of Transactions to establish an interface between the AFC module and the ControlLogix Controller. A transaction is a transfer of a pre-assembled block of data words between the controller and the AFC module. The AFC module supports a block multiplexing scheme whereby you may set up a number of Output Block Arrays (OBAs) with corresponding Input Block Arrays (IBAs) for two way data transfer. This is done by giving each OBA a unique non-zero Transaction Identifier number. The AFC module will not process an OBA unless the Transaction Identifier number of the current OBA is different from the previous one.

Output (Transaction) Block Array Definition

The 248 integer elements of the OBA that must be formatted in the controller ladder logic program are defined as follows:

Element #	Attribute
0	Sentinel (Transaction Number)
1	Output Block Array Length (≤ 245)
2	
3	
4	Space for User Allocated Sub Blocks
.	(Total Available Elements = 245)
.	
.	
.	
244	
245	
246	
247	Anchor (Transaction Number)

The AFC module recognizes an OBA as valid only if all three of the following conditions are true:

- 1 The values of the sentinel and the anchor are the same (this is called the "transaction number").
- 2 The transaction number (sentinel & anchor) is non-zero.
- 3 The transaction number is different from the preceding one.

AFC Response to an OBA

Each time the AFC module receives an OBA, it processes this array and returns a corresponding Input Block Array (IBA) with the same transaction number so that the processor logic can act upon the AFC's response to the data it received in the OBA. The OBA description follows.

Sentinel & Anchor (Transaction Number)

In order to allow the AFC to tell each Output Block Array apart, a unique Transaction Identifier number is assigned to each transmit block. This Transaction Identifier number is assigned to the top and bottom elements of the transaction block to ensure data transfer integrity. The top transaction identifier is called the *Transaction Number Sentinel* and the bottom transaction identifier is called the *Transaction Number Anchor*.

Output Block Length

The element immediately following the sentinel contains the length of the data portion, not including the sentinel, the anchor, or the Output Block Length element itself; hence its maximum value is 245. If the output data block you configured is less than 245 elements, unused elements between the end of the data block and the anchor are ignored on output (to the AFC) and returned as zero on input (from the AFC).

Output Function Blocks (OFB)

The architecture of the Output Block Array provides you with a flexible and powerful way to interface the process and other data to the AFC module. A number of pre-defined Output Function Blocks are at your disposal to implement as required. Each Output Function Block type carries out a specific function. For example, the Meter Process Variable Output Function Block transfers the process variables (temperature, pressure, meter pulses, and so on) to a specific target meter channel in the AFC module. The calculated flow rates, product accumulators, and so on are returned to the controller by the AFC in the associated Input Function Block (IFB) as part of the Input Block Array (IBA). In the same way a Wall Clock Function Block allows you to synchronize the AFC Wall Clock to the processor Wall Clock. Like Lego blocks of various sizes, these function blocks can be stacked together in the data portion of the Output Block Array as shown in the following diagram:

	Element #	Attribute
	0	Sentinel (Transaction Number)
	1	Output Block Length (OBL<=245)
	2	
Global Wall Clock Function Block: 7 Elements	3	
Meter 2 Process Variable Function Block: 12 Elements	.	Space for User Allocated Function Blocks
Meter 3 Process Variable Function Block: 12 Elements	.	(Total Available Elements = 245)
Meter 3 Analysis Function Block: 25 Elements	57	
	.	
	.	
	244	
	245	
	246	
	247	Anchor (Transaction Number)

Input (Transaction) Block Array Definition

The Input Block Array is where the AFC returns the (processed and calculated data) responses to all function blocks received in the Output Block Array from the controller. It returns Input Function Blocks in locations and sizes matching those of the OFBs in the OBA. The AFC completes the actions implied by all function blocks before responding with the IBA to the controller. If the AFC is unable to determine the size of any function block, or the implied size overlaps the block array's anchor, then that and all following function blocks are not processed and a format alarm is raised.

The 250 integer elements of the Input (Transaction) Block Array (IBA) that are returned to the controller to the AFC module are defined as follows:

Element	Attribute																												
0	Sentinel Echo (Transaction Number)																												
1	Block length echo (IBL) (Negative if formatting error present)																												
2																													
3																													
4	Space for Input Image of User Allocated Function Blocks																												
.	(Total Available Elements = 245)																												
.																													
.																													
244																													
245																													
246																													
247	Site Status																												
	<table border="1"> <thead> <tr> <th>Bit #</th> <th>Site Status</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>MVI56-AFC Released and Active</td> </tr> <tr> <td>1</td> <td>Checksum Alarm</td> </tr> <tr> <td>2</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td>Reserved</td> </tr> <tr> <td>4</td> <td>Processor Halted, Offline or Missing</td> </tr> <tr> <td>5</td> <td>Measurement Configuration Changed</td> </tr> <tr> <td>6</td> <td>Set upon Power-Up & cleared when the Wall Clock is set for the first time</td> </tr> <tr> <td>7</td> <td>Cold Start: remains set until first enabled meter is detected</td> </tr> <tr> <td>8</td> <td>Modbus Master poll pending</td> </tr> <tr> <td>9</td> <td>Modbus Master poll complete, results waiting</td> </tr> <tr> <td>10</td> <td>Pass-Thru input pending</td> </tr> <tr> <td>11</td> <td>version 2.06 or later</td> </tr> <tr> <td>12 to 15</td> <td>Reserved</td> </tr> </tbody> </table>	Bit #	Site Status	0	MVI56-AFC Released and Active	1	Checksum Alarm	2	Reserved	3	Reserved	4	Processor Halted, Offline or Missing	5	Measurement Configuration Changed	6	Set upon Power-Up & cleared when the Wall Clock is set for the first time	7	Cold Start: remains set until first enabled meter is detected	8	Modbus Master poll pending	9	Modbus Master poll complete, results waiting	10	Pass-Thru input pending	11	version 2.06 or later	12 to 15	Reserved
Bit #	Site Status																												
0	MVI56-AFC Released and Active																												
1	Checksum Alarm																												
2	Reserved																												
3	Reserved																												
4	Processor Halted, Offline or Missing																												
5	Measurement Configuration Changed																												
6	Set upon Power-Up & cleared when the Wall Clock is set for the first time																												
7	Cold Start: remains set until first enabled meter is detected																												
8	Modbus Master poll pending																												
9	Modbus Master poll complete, results waiting																												
10	Pass-Thru input pending																												
11	version 2.06 or later																												
12 to 15	Reserved																												
248	Meter Alarm Map: Bit <n> set=Meter <n+1> in Alarm																												
249	Anchor Echo (Transaction Number)																												

The input block returned by the AFC module has the sentinel echo and the anchor echo in fixed locations.

Note: The anchor echo is returned as the 250th element in the IBA and not as the 248th element as sent in the OBA.

The controller may then verify the validity of the returned data by checking that the received sentinel echo is the same as the anchor echo. If more than one OBA is being processed, the transaction number must be used to match the IBA (AFC output) with the originating OBA (AFC input, controller output).

Input Block Length and Format Alarm

The "length" element returned by the module in the IBA has the high-order (sign) bit set if any formatting error is detected in the data portion of the OBA you formatted in the logic. This is referred to as a "format alarm". The remainder of the input length element contains the number of data words successfully processed before any formatting inconsistencies were detected by the AFC module and the alarm condition annunciated. If no formatting error exists, the length element will echo the output length you entered. If the output length element itself raises an alarm (for example, value > 245), then the returned length value is -1.

11.2 MVI56-AFC Function Block Interface

The following table defines the various function blocks that that PLC programmer may use to set up two-way data transfer between logic and the AFC module:

Function Block Type	Function Code	Overall Block Length	Meter-Specific
Null	0	User defined	No
Wall Clock	1	7 elements	No
Modbus Pass-Through	6	User defined	No
Meter Process Variable	8	12 elements	Yes
Meter Analysis	9	25 elements	Yes
Meter Type Fetch	10	2 elements	Yes
Meter & Site Signals	12	2 elements	Yes
Meter Archive Fetch	14	42 elements	Yes
Gateway Read	16, 17, 18, 19	User defined	No
Gateway Write	20, 21	User defined	No
Modbus Master	24, 25, 26	User defined	No
Meter Disable/Enable	28, 29	2 elements	No

Any of the above function blocks may be allocated to the 245 elements of the data portion of the OBA in the order and quantity required by the end-user application at the discretion of the logic programmer.

11.2.1 Function Block Structure

The first word of each output and input function block, called the "block ID", specifies the block type, control flags, addressing, and additional information depending on type. For each function block type, the matching input function block (returned by the AFC) also contains a block ID. Portions of this word echo the output verbatim, while the remainder may contain status information about the result of the action.

Each output function block ID is verified by the AFC to be unambiguous; any value that is out of range, or any undefined bit that is set to 1, causes a format alarm.

The typical structure of a function block is shown in the following diagram:

Typical Output Function Block (OFB)

Element	Attribute																																																			
0	Block ID: Function Code, Processing Controls, Length or Meter/Stream Selection <table border="1"> <thead> <tr> <th>Bit</th> <th>15</th> <th>14</th> <th>13</th> <th>12</th> <th>11</th> <th>10</th> <th>9</th> <th>8</th> <th>7</th> <th>6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td>Attrib</td> <td colspan="6">Function Code</td> <td>inp</td> <td>out</td> <td colspan="8">Data Block Length or Meter and Stream Numbers</td> </tr> <tr> <td></td> <td colspan="6">When bit 9 is set to 1, the AFC module skips returning the input</td> <td></td> <td></td> <td colspan="8">When bit 8 is set to 1, the AFC module ignores the received output</td> </tr> </tbody> </table>	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Attrib	Function Code						inp	out	Data Block Length or Meter and Stream Numbers									When bit 9 is set to 1, the AFC module skips returning the input								When bit 8 is set to 1, the AFC module ignores the received output							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																				
Attrib	Function Code						inp	out	Data Block Length or Meter and Stream Numbers																																											
	When bit 9 is set to 1, the AFC module skips returning the input								When bit 8 is set to 1, the AFC module ignores the received output																																											
1	Data Block Length																																																			
2																																																				
...																																																				
...																																																				
N																																																				

Typical Input Function Block (IFB)

Element	Attribute																																																			
0	Block ID: Function Code Echo, Status and Alarms, Meter and Active Stream Numbers <table border="1"> <thead> <tr> <th>Bit</th> <th>15</th> <th>14</th> <th>13</th> <th>12</th> <th>11</th> <th>10</th> <th>9</th> <th>8</th> <th>7</th> <th>6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td>Attrib</td> <td colspan="6">Function Code Echo</td> <td>Alarm</td> <td>0</td> <td colspan="3">Active Stream</td> <td colspan="5">Meter Number or Function Status</td> </tr> <tr> <td></td> <td colspan="6">For meter-specific functions, bit 9 reports the meter's alarm status</td> <td></td> <td></td> <td colspan="8">Meter-specific functions echo the numbers of the meter and the active stream; other functions may return result status here</td> </tr> </tbody> </table>	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Attrib	Function Code Echo						Alarm	0	Active Stream			Meter Number or Function Status						For meter-specific functions, bit 9 reports the meter's alarm status								Meter-specific functions echo the numbers of the meter and the active stream; other functions may return result status here							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																				
Attrib	Function Code Echo						Alarm	0	Active Stream			Meter Number or Function Status																																								
	For meter-specific functions, bit 9 reports the meter's alarm status								Meter-specific functions echo the numbers of the meter and the active stream; other functions may return result status here																																											
1	Data Block Length																																																			
2																																																				
...																																																				
...																																																				
N																																																				

Fixed and Variable Length Function Blocks

As shown in the Function Block Type table above, some function blocks have variable lengths (Overall Block Length "User defined") while the others have lengths that are fixed. Also, some of the fixed-length blocks are specific to individual meters while the others, and all variable-length blocks, are not. Bits 7 to 0 of the output block ID provide this information.

For a variable-length block, bits 7 to 0 of the output block ID contain the length in elements of the data portion of the block that follows the block ID itself; that is, the overall length of the block is one element greater than the number coded into those bits. For example, a block that performs a Gateway Read of 10 holding registers from the primary slave will have this Block ID:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 16				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0

In this case, the overall block length is 11 elements (You may specify a longer block length if you wish, because the number of registers actually read is specified separately, but the data length must be at least large enough to accommodate the registers to be transferred; see the detailed function description for more information).

For a fixed-length meter-specific block, bits 7 to 5 of the output block ID give the stream number (if applicable) or zero, and bits 4 to 0 give the meter number. For example, if the Meter Process Variable Block (Function Code 8) is required to service meter number 3, you must format the function block ID element as shown in the following diagram:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 8					inp	out	Stream No. N/A			Meter Number 3					
Value	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1

For a fixed length non-meter-specific block, bits 7 to 0 of the output block ID may implement options to modify the behavior of the function, depending on the function.

When the function has been performed, bits 15 to 10 of the input block ID in the IBA always echo the function code, and bits 9 to 0 contain information depending on the block type. For meter-specific blocks, bit 9 reports whether the meter is in alarm, bit 8 is zero, bits 7 to 5 contain the number of the active stream and bits 4 to 0 echo the meter number. For example, the above Meter Process Variable Block may return this input block ID:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 8						err	n/a	Active Stream			Meter Number 3				
Value	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	1

This block id reports that Meter 3 is measuring flow on its Stream 2 and is in alarm (For firmware version 2.05 and later, the returned active stream number is always non-zero; for earlier versions, having a single stream which was always active, the returned active stream number is always zero. The ladder may use this information to distinguish between multiple-stream and single-stream firmwares). For non-meter-specific blocks, bits 9 to 0 may return status indicators from the performance of the function, depending on the function.

Special Notes

Note that you can control the processing of the Output Function Blocks by the AFC module through simple ladder logic implementation, as described next.

Managing Output Function Block(s) by manipulating Bit 8

For every Output Function Block (OFB) whether it is meter-specific or not, bit 8 allows you to control and manage its execution by the AFC Module in the following way:

- To disable processing of the OFB, set bit 8 to value 1
- For normal processing of the OFB, set bit 8 to value 0

Managing Input Function Block(s) by Manipulating Bit 9


For every Input Function Block (IFB), which is a response by the AFC to the associated OFB, whether it is meter-specific or not, bit 9 allows you to control input to the controller from the AFC module in the following way:

- To disable input to the controller via the IFB, set bit 9 to value 1
- For normal data/response from the AFC module, set bit 9 to value 0

Note: The input block ID always returns the information described above, regardless of the settings of bits 8 and 9 in the output block ID, unless otherwise stated in the individual function description.

11.2.2 Function Block Definition - 0: Null

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 0						inp	out	Data Block Length: User Defined							
Value	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
Example: A block length of 23 (10111 binary) shown. 																

Function Block Name:	Null
Function Block ID:	0 decimal, 000000 binary, 0 dec.
Target:	Global
Data Block Length:	User Defined
Total Block Length:	(User Defined + 1) Elements

Description

This function block is used as a space filler only. It implies no action by the AFC.

Null: Output Function Block (OFB)		Null: Input Function Block (IFB)	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length	0	Block ID - Function Code
1	Null Data	1	
2		2	
...		...	
...		...	
n		n	

Note: A Block ID with all 16 bits set to 0 is interpreted as a Null FB of length 1 (the Block ID itself). Note also that because the Null FB implies no action, the settings of the "out" and "inp" bits are irrelevant.

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.3 Function Block Definition - 1: Wall Clock

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 1						inp	out	Data Block Length: Set to 0							
Value	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 0						inp	out							N	E
Value	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bit = 1 while WallClock not set ↑

Bit = 1 when "Set WallClock" fails with error ↑

Function Block Name:	Wall Clock
Function Block ID:	1 decimal, 000001 binary, 1024 dec.
Target:	Global
Data Block Length:	6 Elements
Total Block Length:	7 Elements

Description

This block sets the AFC Module Wall Clock. It may be used to synchronize the AFC Wall Clock to the Controller Wall Clock. The Wall Clock IFB returns the wall clock as known by the AFC modules.

If the output function block fails to set the clock, bit 0 of the input function block ID is set. If the input returns a clock that has not been set, bit 1 of the input block ID is set and the clock value returned is all zeros.

Null: Output Function Block (OFB)		Null: Input Function Block (IFB)	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length	0	Block ID - Function Code & Length
1	Year	1	Year
2	Month	2	Month
3	Day	3	Day
4	Hours	4	Hours
5	Minutes	5	Minutes
6	Seconds	6	Seconds

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.4 Function Block Definition - 4, 5, 6 & 7: Modbus Pass Through

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 6						inp	out	Data Block Length: User Defined							
Value	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0
<p>Example: A block length of 48 (00110000 binary) shown.</p>																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 6						inp	out	Reserved – Not used					W	B	R
Value	0	0	0	1	1	0	0	0	0	0	0	0	0	0/1	0/1	0/1
<p>See the Special Notes below</p>																

Function Block Name:	Modbus pass-through
Function Block ID:	6 decimal, 000110 binary, 6144 dec.
Target:	Global
Data Block Length:	User Defined - from 3 to 127
Total Block Length:	(User Defined + 1) Elements

Description

This block fetches any pass-through Modbus write command sent by an external Modbus host, which is returned to the Processor essentially verbatim (see following illustration). The AFC module buffers any such command until it is returned to the Processor via this input function block, at which time the buffer is made available for the next command.

Because a Modbus write command must write at least one word (bit) and cannot write more than 125 words (2000 bits), the data length of this function block must lie between 3 and 127 (register address + number of registers + data).

MB pass-through: Output Function Block		MB pass-through: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length	0	Block ID - Function Code & Length
1		1	Modbus Register Address
2		2	Number of Registers
3		3	Data Element 1
4		4	Data Element 2
5		5	Data Element 3
...	
...	
n		n	Data Element (n-2)

A Modbus pass-through function block formatted as in the previous example with data block length of 48 elements (total block size of 49 elements) when executed returns up to 46 Modbus integer data elements written to the AFC module in Modbus protocol format. The first data block element contains the number of Modbus data elements to follow.

Note: If the value in element 2 of the IFB is 0, no Modbus command was pending. If bit 15 in this element is high, it indicates Modbus data overflow; that is, the SCADA host has written more data elements than the pass-through allows. In this case, the Pass-Through IFB is filled with as much data as will fit and the high order bit of "Number of Registers" (element 2 of the IFB) is set indicating this overflow condition. The number of registers (bits 14-0 of element 2 of the IFB) is equal to that of the MB message received. On data overflow, the last command stays pending and may be flushed by setting bit 9 in the Output Function Block ID (skip input to controller).

When the AFC completes processing the OFB, the IFB ID is returned with the Modbus Message Block as described above. The AFC provides a number of indicators to allow you to design the Processor logic to take appropriate action once the controller receives the message in the returned IFB.

Special Notes

Monitoring Bit 0 (R) of the Block ID Element of the Input Function Block

- This bit is set to 1 if a Pass-Through Modbus message from a SCADA host was present.
- This bit is set to 0 if no message was present.

Monitoring Bit 1 (B) of the Block ID Element of the Input Function Block

- This bit is set to 1 if the Modbus message from a SCADA host was a bit write - Modbus function code 5 or 15.
- This bit is set to 0 if the Modbus message from a SCADA host was a word write - Modbus function codes 1 or 16.

Monitoring Bit 2 (W) of the Block ID Element of the Input Function Block


- This bit is set to 1 if the Modbus message from the SCADA host was longer than the data block size you defined - Overflow Status.
- This bit is set to 0 if the Modbus message from a SCADA host was successfully transferred to the controller (or flushed).

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

Function 4 is the same as function 6, except that the received command remains pending even if no overflow occurred; this allows you to "pre-inspect" the command before completing its processing with function 6. Functions 5 and 7 are the same as functions 4 and 6 respectively, except that for word-write commands (Modbus function codes 6 and 16) pairs of words in the data are swapped.

11.2.5 Function Block Definition - 8: Meter Process Variables

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 8						inp	out	n/a			Meter Number				
Value	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
Example: Meter Number 6 (00110 binary) shown. 																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 8						err	n/a	Active Stream			Meter Number				
Value	0	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0
<p>Bit 9 is set to 1 if any meter alarm bit is on</p> <p>Stream 1 active</p> <p>Meter number 6 echoed</p>																

Function Block Name:	Meter Process Variables
Function Block ID:	8 decimal, 001000 binary, 8192 dec.
Target:	Selected Meter Number
Data Block Length:	11 Elements
Total Block Length:	12 Elements

Description

This block provides fresh values of the process inputs to and returns results from the latest meter calculation scan. The meter is scanned continuously; therefore several calculations may occur between applications of this block. Process inputs in the output block are meaningful depending on configuration. For example, the fourth input is seen as differential pressure, pulse count, flow rate, or ignored depending on the meter type; the water content is meaningful only for liquids.

The input block always returns an alarm indicator as part of the block ID; the remaining 11 words are freely selectable from the entire meter database, with the default selection including detailed alarms, net volume accumulator, net volume flow rate, correction factors, and intermediate values depending on configuration.

The Meter Process Variable Block must be set up for one of the following meter types:

- Orifice (Differential) Meter with Gas Product
- Pulse (Linear) Meter with Gas Product
- Orifice (Differential) Meter with Liquid Product
- Pulse (Linear) Meter with Liquid Product
- Flow Rate Integration with Gas Product
- Pulse Frequency Integration with Gas Product
- Flow Rate Integration with Liquid Product
- Pulse Frequency Integration with Liquid Product

Orifice Meter with Gas Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Reserved	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Differential Pressure LS (see Note)	6	Gross Flow Rate LS (float)
7	Differential Pressure MS	7	Gross Flow Rate MS (float)
8	Reserved	8	Fpv LS (float)
9	Reserved	9	Fpv MS (float)
10	Reserved	10	Cprime LS (float)
11	Reserved	11	Cprime MS (float)

Pulse Meter with Gas Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Reserved	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Meter Pulses LS (double integer)	6	Gross Flow Rate LS (float)
7	Meter Pulses MS (double integer)	7	Gross Flow Rate MS (float)
8	Reserved	8	Fpv LS (float)
9	Reserved	9	Fpv MS (float)
10	Meter Pulse Freq: Hz LS (float)	10	Cprime LS (float)
11	Meter Pulse Freq: Hz MS (float)	11	Cprime MS (float)

Orifice Meter with Liquid Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Water % (see Note)	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Differential Pressure LS (see Note)	6	Gross Accumulator LS (double integer)
7	Differential Pressure MS	7	Gross Accumulator MS (double integer)
8	Density LS (see Note)	8	Gross Standard Accumulator LS (double integer)
9	Density MS	9	Gross Standard Accumulator MS (double integer)
10	Reserved	10	Mass Accumulator LS (double integer)
11	Reserved	11	Mass Accumulator MS (double integer)

Pulse Meter with Liquid Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Water % (see Note)	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Meter Pulses LS (double integer)	6	Gross Accumulator LS (double integer)
7	Meter Pulses MS (double integer)	7	Gross Accumulator MS (double integer)
8	Density LS (see Note)	8	Gross Standard Accumulator LS (double integer)
9	Density MS	9	Gross Standard Accumulator MS (double integer)
10	Meter Pulse Freq: Hz LS (float)	10	Mass Accumulator LS (double integer)
11	Meter Pulse Freq: Hz MS (float)	11	Mass Accumulator MS (double integer)

Flow Rate Integration with Gas Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Reserved	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Flow Rate LS (see Note)	6	Gross Flow Rate LS (float)
7	Flow Rate MS	7	Gross Flow Rate MS (float)
8	Reserved	8	Fpv LS (float)
9	Reserved	9	Fpv MS (float)
10	Reserved	10	Cprime LS (float)
11	Reserved	11	Cprime MS (float)

Pulse Frequency Integration with Gas Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Reserved	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Reserved	6	Gross Flow Rate LS (float)
7	Reserved	7	Gross Flow Rate MS (float)
8	Reserved	8	Fpv LS (float)
9	Reserved	9	Fpv MS (float)
10	Meter Pulse Freq: Hz LS (float)	10	Cprime LS (float)
11	Meter Pulse Freq: Hz MS (float)	11	Cprime MS (float)

Flow Rate Integration with Liquid Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Water % (see Note)	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Flow Rate LS (see Note)	6	Gross Accumulator LS (double integer)
7	Flow Rate MS	7	Gross Accumulator MS (double integer)
8	Density LS (see Note)	8	Gross Standard Accumulator LS (double integer)
9	Density MS	9	Gross Standard Accumulator MS (double integer)
10	Reserved	10	Mass Accumulator LS (double integer)
11	Reserved	11	Mass Accumulator MS (double integer)

Pulse Frequency Integration with Liquid Product

Meter PV: Output Function Block		Meter PV: Input Function Block	
Element	Attribute	Element	Attribute
0	Block ID: Function Code & Meter Number	0	Block ID: Function Code, Meter Number, Active Stream Number
1	Water % (see Note)	1	Meter Alarms (Bitmap)
2	Temperature LS (see Note)	2	Net Accumulator LS (double integer)
3	Temperature MS	3	Net Accumulator MS (double integer)
4	Pressure LS (see Note)	4	Net Flow Rate LS (float)
5	Pressure MS	5	Net Flow Rate MS (float)
6	Reserved	6	Gross Accumulator LS (double integer)
7	Reserved	7	Gross Accumulator MS (double integer)
8	Density LS (see Note)	8	Gross Standard Accumulator LS (double integer)
9	Density MS	9	Gross Standard Accumulator MS (double integer)
10	Meter Pulse Freq: Hz LS (float)	10	Mass Accumulator LS (double integer)
11	Meter Pulse Freq: Hz MS (float)	11	Mass Accumulator MS (double integer)

Note: During meter configuration you can select from up to 3 data format options (floating point, scaled integer, 4 to 20 mA) for supplying in the OFB the process input values noted. See the Special Notes section below for details.

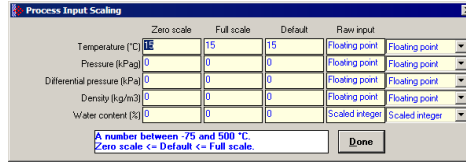
Returned Alarm Codes for Meter Data

The following table provides Alarm codes for meter data:

Alarm Code	Bit
Input out of range: Temperature	Bit 0
Input out of range: Pressure	Bit 1
Input out of range: Differential Pressure (or Flow Rate, or Frequency)	Bit 2
Input out of range: Flowing Density	Bit 3
Input out of range: Water Content	Bit 4
Differential pressure low (or Flow Rate, or Frequency)	Bit 5
Orifice Pressure Exception	Bit 6
Accumulation Overflow Error	Bit 7
Orifice Characterization Error	Bit 8
Analysis Total Zero (version 2.04 or earlier) Reserved (version 2.05 or later)	Bit 9
Analysis Total Not Normalized (version 2.04 or earlier) Analysis Characterization Error (version 2.05 or later)	Bit 10
Compressibility Calculation Error (gas) High Water Error (liquids)	Bit 11
Reference Density Error	Bit 12
Temperature Correction Error	Bit 13
Vapor Pressure Error	Bit 14
Pressure Correction Error	Bit 15

Special Notes

The Meter Process Variable Block must provide all process variables in the format you configured through the AFC Manager for each meter run that is implemented.



This OFB can handle process variables in the following formats:

- Floating Point - Recommended format, used by sample ladder logic
- Scaled Integer
- 4 to 20 mA (raw A/D count)

Note 1: For water % (liquids only) floating point is not available. As a Scaled Integer it must be copied to element 1 as an INT with two decimal places implied. For example, water % value of 7.23 must be entered as 723. The AFC divides this value by 100.

Note 2: The floating point format takes up two elements (32 bits) for each process variable. For example, the Temperature must be copied from a floating point tag in the controller to elements 2 and 3 of the OFB.

Note 3a: Temperature as a scaled integer must be copied to elements 2 and 3 as a DINT with 2 decimal places implied (1/100th of a degree). For example, a temperature of 24.97°F must be copied as 2497.

Note 3b: Flowing pressure as a scaled integer must be copied to elements 4 and 5 as a DINT with no decimal places implied for the SI units (kPa) and one decimal place implied for the U.S. units (psi). For example, a pressure of 5000 kPag must be copied as 5000 and a pressure of 259.7 psi must be copied as 2597.

Note 3c: Differential Pressure as a scaled integer must be copied to elements 6 and 7 as a DINT with 2 decimal places implied for inches of H₂O (hw) and 3 places for kPa (1/100th & 1/1000th of the selected unit). For example, a DP of 37.52 in H₂O must be copied as 3752.

Note 3d: Flow Rate as a scaled integer must be copied as a DINT to elements 4 and 5 with zero decimal places implied. To obtain a desired precision, choose an appropriate Flow Input Unit (Meter Configuration window, Primary Input Characteristics panel)

Note 3e: Pulse Frequency may be supplied in scaled integer or 4 to 20 mA formats only in version 2.05 or later; in version 2.04 or earlier only floating point format is available. As a scaled integer it must be copied to elements 10 and 11 as a DINT in units of Hz with no decimal places implied. For example, a frequency of 2574 Hz must be copied as 2574.

Note 4: Note that three options for the product density for liquid meters are available, and if the Scaled Integer option is selected then density must be copied as a DINT to elements 8 and 9 as follows:

- Kg/m³ - One implied decimal place (513.7 kg/m³ must be entered as 5137)
- Relative Density - Four implied decimal places (1.0023 60F/60F must be entered as 10023)
- API - Two implied decimal places (80.45 API must be entered as 8045)

Note 5: For the 4 to 20 mA format, the raw A/D count from the analog input module must be copied as a DINT to the proper pair of elements of the OFB (or for Water %, as an INT to element 1).

Note 6: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.6 Function Block Definition - 9: Meter Analysis, 16-bit

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 9						err	n/a	Stream Number			Meter Number				
Value	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0
<p>Example: Stream Number 0 (000 binary) shown; selects the Active Stream</p> <p>Example: Meter number 6 (00110 binary) shown.</p>																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 9						err	n/a	Active Stream			Meter Number				
Value	0	0	1	0	0	1	0	0	0	0	1	0	0	1	1	0
<p>Bit 9 is set to 1 if any meter alarm bit is on</p> <p>Stream 1 active</p> <p>Meter number 6 echoed</p>																

Function Block Name:	Meter Analysis
Function Block ID:	9 decimal, 001001 binary, 9216 dec.
Target:	Selected Meter Number and Stream Number
Data Block Length:	24 Elements
Total Block Length:	25 Elements

Description

This block supplies the analysis for AGA8 and GPA8173 calculations, in low-precision form as integers scaled to 4 decimal places. Upon change of analysis the meter undergoes a "characterization" calculation, which is performed before this block is returned; if the analysis has not changed, no characterization is required, and this block is returned immediately.

Element	Attribute	Element	Attribute
0	Block ID - Function Code & Meter Number	0	Block ID - Function Code, Meter Number, Active Stream Number
1	Propane - C1		Reserved
2	Nitrogen - N2		Reserved
3	Carbon dioxide -CO2		Reserved
4	Methane - C2		Reserved
5	Propane - C3		Reserved
6	Water - H2O		Reserved
7	Hydrogen Sulfide - H2S		Reserved
8	Hydrogen - H2		Reserved
9	Carbon Monoxide - CO		Reserved
10	Oxygen - O2		Reserved
11	Iso Butane - IC4		Reserved
12	Butane - C4		Reserved
13	Iso Pentane - IC5		Reserved
14	Pentane - C5		Reserved
15	Hexane - C6		Reserved
16	Heptane - C7		Reserved
17	Octane - C8		Reserved
18	Nonane - C9		Reserved
19	Decane - C10		Reserved
20	Helium - He		Reserved
21	Argon - Ar		Reserved
22	Neo Pentane - C5		Reserved
23	Ux User 1		Reserved
24	Ux User 2		Reserved

The input block always returns an alarm indicator as part of the block id; the remaining 24 words are freely selectable from the entire meter database, with the default selection being nothing (all zero).

An application should not implement both 16-bit and 32-bit analysis blocks for the same stream. The AFC will convert the analysis, whether 16-bit or 32-bit, to the precision configured for the analysis slot assigned to the targeted stream, though best practice would be to use the analysis block that matches the configured precision.

Special Notes

Note 1: The address of each component will depend on which components are selected during the configuration of the meter (using the AFC Manager). If less than 24 components are selected, they are packed to remove unused component elements in the Analysis Block. The relative order of the selected components is maintained and the overall size of the Analysis FB remains unchanged (With firmware version 2.05 and later, you may choose to receive the analysis unpacked regardless of the component selection; see the descriptions of Site Option bits 13 and 14 for more information about the packing of analyses)

Note 2: The component mole fractions are entered as scaled integers. A component mole fraction of .0753 (7.53 mole percent) is entered as the integer value 753. This number is internally divided in the AFC module by 10,000.

Note 3: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.7 Function Block Definition - 10: Meter Type Fetch

Note: This function is available only in versions 2.04 and later.

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 10						inp	out	n/a			Meter Number				
Value	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0
Example: Meter Number 6 (00110 binary) shown.																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 10						err	n/a	Active Stream			Meter Number				
Value	0	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0
Bit 9 is set to 1 if any meter alarm bit is on																
Stream 1 active																
Meter number 6 echoed																

Function Block Name:	Meter Process Variables
Function Block ID:	10 decimal, 001010 binary, 10240 dec.
Target:	Selected Meter Number
Data Block Length:	1 Element
Total Block Length:	2 Elements

Description

This block fetches a summary of the meter type and product group, which may be used by the PLC to tailor its logic for providing process input values to the AFC. Since there is no significant output, the "ignore output" bit (bit 8) has no function. For firmware version 2.05 and later, non-zero stream information is returned; for firmware version 2.04, the one stream is always enabled and active which renders stream information redundant, and returned stream information is all zero. The summary is returned as a single element of data, a bitmap having this layout:

Bit	Description
0	meter is in alarm
1	meter is enabled
2	[spare]
3	[spare]
4	metering device is linear (pulse)
5	product phase is liquid
6	primary input is flow rate / frequency
7	[spare]
8	stream 1 enabled
9:	stream 2 enabled
10	stream 3 enabled
11	stream 4 enabled
12 to 13:	number of active stream (0-based: 0 thru 3)
14	[spare]
15	[spare]

11.2.8 Function Block Definition - 11: Meter Analysis, 32-bit

Note: Currently, this function is only available on the MVI56- AFC. For all other platforms continue to use AFC Manager 2.05 or earlier.

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Attrib	Function Code: 11						err	n/a	Stream Number				Meter Number				
Value	0	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	0
<p>Example: Stream Number 0 (000 binary) shown; selects the Active Stream</p> <p>Example: Meter number 6 (00110 binary) shown.</p>																	

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Attrib	Function Code: 11						err	n/a	Active Stream			Meter Number					
Value	0	0	1	0	1	1	0	0	0	0	1	0	0	1	1	0	

Bit 9 is set to 1 if any meter alarm bit is on

Stream 1 active

Meter number 6 echoed

Function Block Name:	Meter Analysis
Function Block ID:	11 decimal, 001011 binary, 11264 dec.
Target:	Selected Meter Number and Stream Number
Data Block Length:	24 Elements
Total Block Length:	25 Elements

Analysis Precision

This block supplies the analysis for AGA8 and GPA8173 calculations, in high-precision form as IEEE 32-bit floating point numbers. Upon change of analysis the meter undergoes a "characterization" calculation, which is performed before this block is returned; if the analysis has not changed, no characterization is required, and this block is returned immediately.

The input block always returns an alarm indicator as part of the block id; the remaining 48 words contain the following information:

Word	Description																				
1	Firmware major and minor version numbers: Low-order byte: minor version number High-order byte: major version number																				
2	Firmware revision number																				
3	Analysis precision and stream assignment for meter 1 -- Contents of meter-relative holding register 133 (for meter 1 this is holding register 8133), which is laid out as follows (also see AFC Manager's Modbus Dictionary): Low-order byte: analysis precision: 4 dibits, one for each analysis "slot": where each dibit has one of these values: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Bits</th> <th>Contents</th> </tr> </thead> <tbody> <tr> <td>0, 1</td> <td>Precision of analysis in slot 0 (Mh00720++)</td> </tr> <tr> <td>2, 3</td> <td>Precision of analysis in slot 1 (Mh00750++)</td> </tr> <tr> <td>4, 5</td> <td>Precision of analysis in slot 2 (Mh00780++)</td> </tr> <tr> <td>6, 7</td> <td>Precision of analysis in slot 3 (Mh00810++)</td> </tr> </tbody> </table> High-order byte: stream analysis assignment: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Precision</th> </tr> </thead> <tbody> <tr> <td>0 (00b)</td> <td>(second slot of preceding high-precision)</td> </tr> <tr> <td>1 (01b)</td> <td>Low-precision, one slot only</td> </tr> <tr> <td>2 (10b)</td> <td>High-precision, two consecutive slots</td> </tr> <tr> <td>3 (11b)</td> <td>(illegal value; never occurs)</td> </tr> </tbody> </table> 4 dibits, one for each stream:	Bits	Contents	0, 1	Precision of analysis in slot 0 (Mh00720++)	2, 3	Precision of analysis in slot 1 (Mh00750++)	4, 5	Precision of analysis in slot 2 (Mh00780++)	6, 7	Precision of analysis in slot 3 (Mh00810++)	Value	Precision	0 (00b)	(second slot of preceding high-precision)	1 (01b)	Low-precision, one slot only	2 (10b)	High-precision, two consecutive slots	3 (11b)	(illegal value; never occurs)
Bits	Contents																				
0, 1	Precision of analysis in slot 0 (Mh00720++)																				
2, 3	Precision of analysis in slot 1 (Mh00750++)																				
4, 5	Precision of analysis in slot 2 (Mh00780++)																				
6, 7	Precision of analysis in slot 3 (Mh00810++)																				
Value	Precision																				
0 (00b)	(second slot of preceding high-precision)																				
1 (01b)	Low-precision, one slot only																				
2 (10b)	High-precision, two consecutive slots																				
3 (11b)	(illegal value; never occurs)																				

Word	Description										
4	Analysis precision and stream assignment for meter 2										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Contents</th> </tr> </thead> <tbody> <tr> <td>0, 1</td> <td>Analysis slot number (0-based) for stream 1</td> </tr> <tr> <td>2, 3</td> <td>Analysis slot number (0-based) for stream 2</td> </tr> <tr> <td>4, 5</td> <td>Analysis slot number (0-based) for stream 3</td> </tr> <tr> <td>6, 7</td> <td>Analysis slot number (0-based) for stream 4</td> </tr> </tbody> </table>	Bits	Contents	0, 1	Analysis slot number (0-based) for stream 1	2, 3	Analysis slot number (0-based) for stream 2	4, 5	Analysis slot number (0-based) for stream 3	6, 7	Analysis slot number (0-based) for stream 4
Bits	Contents										
0, 1	Analysis slot number (0-based) for stream 1										
2, 3	Analysis slot number (0-based) for stream 2										
4, 5	Analysis slot number (0-based) for stream 3										
6, 7	Analysis slot number (0-based) for stream 4										
5	Analysis precision and stream assignment for meter 3										
6	Analysis precision and stream assignment for meter 4										
7	Analysis precision and stream assignment for meter 5										
8	Analysis precision and stream assignment for meter 6										
9	Analysis precision and stream assignment for meter 7										
10	Analysis precision and stream assignment for meter 8										
11	Analysis precision and stream assignment for meter 9										
12	Analysis precision and stream assignment for meter 10										
13	Analysis precision and stream assignment for meter 11										
14	Analysis precision and stream assignment for meter 12										
15	Analysis precision and stream assignment for meter 13										
16	Analysis precision and stream assignment for meter 14										
17	Analysis precision and stream assignment for meter 15										
18	Analysis precision and stream assignment for meter 16										
9 to 48	Zero										

An application should not implement both 16-bit and 32-bit analysis blocks for the same stream. The AFC will convert the analysis, whether 16-bit or 32-bit, to the precision configured for the analysis slot assigned to the targeted stream, though best practice would be to use the analysis block that matches the configured precision.

The AFC clips component concentrations received in this block to the range 0.0000 through 6.5535, without raising a range alarm for any value that is actually clipped. However, if clipping is performed then the analysis itself will raise a normalization alarm.

11.2.9 Function Block Definition - 12: Site/Meter Signals

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 12						inp	out	n/a			Meter Number or 0				
Value	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0
Example: Meter Number 6 (00110 binary) shown.																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 12						err	n/a	Active Stream			Meter Number or 0				
Value	0	0	1	1	0	0	0	0	0	0	1	0	0	1	1	0
Bit 9 is set to 1 if any meter alarm bit is on																
Stream 1 active																
Meter number 6 echoed																

Function Block Name:	Site/Meter Signals
Function Block ID:	12 decimal, 001100 binary, 12288 dec.
Target:	Selected Meter Number, or 0 for Site
Data Block Length:	1 Element
Total Block Length:	2 Elements

Description

This block issues signals to the meter logic in the AFC. A signal is an instruction for the AFC to perform a function once. Signals are latched in the AFC and remain pending until they are discharged. Typical signals are:

- Reset Resettable Accumulator
- Write Ad-hoc Archive Record

The AFC returns this block immediately, without delaying it until the signal is discharged. However, it is still possible for the signal to be discharged before the block is returned, so it is not guaranteed that output signal bits will be echoed in the input. To schedule a signal, the Processor should latch the appropriate bit in the output signal word (Note that AFC acts upon a signal bit transition from 0 to 1 only). The maps of available signal bits are provided in the following table.

Signals: Output Function Block OFB		Signals: Input Function Block IFB	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Mtr No.	0	Block ID - Function Code & Mtr No.
1	Signal Bits	1	Pending Signals (see description above)
	Bit # Meter Signal Bits		
	0 Select Stream 1 (version 2.05 and later)		
	1 Select Stream 2 (version 2.05 and later)		
	2 Select Stream 3 (version 2.05 and later)		
	3 Select Stream 4 (version 2.05 and later)		
	4 Reset Resettable Accumulator 1		
	5 Reset Resettable Accumulator 2		
	6 Reset Resettable Accumulator 3		
	7 Reset Resettable Accumulator 4		
	8 Write Daily Archive		
	9 Write Hourly Archive		
	10 to 15 Reserved		
	Bit # Site Signal Bits		
	0 Event log download complete (purge event log)		
	1 Clear all checksum alarms		
	2 to 15 Reserved		

Special Note: Using Meter Number Value of 0 in OFB ID to write Site Signal Bits

- Set the Meter number to 0 to write to the Site Signal bits.
- Set the Meter number (1 through 16) to write to the selected Meter Signal bits.

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.10 Function Block Definition - 14: Meter Archive Fetch

Note: This function is available only in versions 2.04 and later.

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 14						inp	out	n/a			Meter Number				
Value	0	0	1	1	1	0	0	0	0	0	0	0	0	1	1	0
Example: Meter Number 6 (00110 binary) shown.																

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 14						err	n/a	Active Stream			Meter Number				
Value	0	0	1	1	1	0	0	0	0	0	1	0	0	1	1	0
Bit 9 is set to 1 if any meter alarm bit is on																
Stream 1 active																
Meter number 6 echoed																

Function Block Name:	Meter Process Variables
Function Block ID:	14 decimal, 001110 binary, 14336 dec.
Target:	Selected Meter Number
Data Block Length:	41 Element
Total Block Length:	42 Elements

Description

This block returns an archive record to the PLC, selected according to the criteria in the Output block. Archive fetch may be controlled through the use of the "ignore output" bit (bit 8) of the block id; the "skip input" bit (bit 9) has no function and is ignored.

Output block format:

Word	Description
1	Archive file select; 0 daily, 1 hourly
2	Age of requested archive
3 to 41	[ignored]

Input block format:

Word	Description
1	Error bitmap: Bit 0: Archive file selection invalid Bit 1: Age outside configured file size
2 to 41	Archive record data (up to 40 words, or 0 on error)

11.2.11 Function Block Definition - 16/17/18/19: Modbus Gateway Read

Output Function Block ID

Function Code 16: Read Modbus Registers from the PRIMARY Slave HOLDING register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 16				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Example: A block length of 48 (00110000 binary) shown.																

Function Code 17: Read Modbus Registers from the VIRTUAL Slave HOLDING register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 17				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	0	0	1	0	0	0	0	0	1	1	0	0	0

Function Code 18: Read Modbus Registers from the PRIMARY Slave INPUT register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 18				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0

Function Code 19: Read Modbus Registers from the VIRTUAL Slave INPUT register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 19				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	0	1	1	0	0	0	0	0	1	1	0	0	0

Input Function Block ID

When the AFC completes processing the output function block, the input Function Block ID is returned with the Modbus Message Block as described in more detail below. The AFC provides a number of indicators to allow you to design the Processor logic to take appropriate action once the controller receives the message in the returned IFB.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 19								Reserved – Not used					V	A	F
Value	0	1	0	1	1	1	0	0	0	0	0	0	0	0/1	0/1	0/1
														See the Special Notes below		
														↑ ↑ ↑		

Function Block Name:	Modbus Gateway Read
Function Block ID:	16 through 19
Target:	Global
Data Block Length:	User defined - from 2 to 244
Total Block Length:	(User defined +1) Elements

Description

These four function codes allow you to fetch data from the vast address space of the AFC module for consumption by the controller. The four functions are designed to access the four separate register banks by assigning unique function codes for each type of access. These four function codes are tabulated below in this section.

This block performs an arbitrary data transfer between the Processor and the AFC data table. Any data transfer that can be performed with a Modbus command to either AFC slave through any of the module's ports may be implemented using the gateway.

Any data words not relevant to the command are ignored upon output (to the AFC) and zero upon input (from the AFC). Gateway data transfer may be controlled through the use of the "ignore output" bit (bit 8) of the block ID; the "skip input" bit (bit 9) has no function and is ignored.

MB Gateway Read: OFB		MB Gateway Read: IFB	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length	0	Block ID - Function Code & Status
1	Modbus Register Address	1	Data (returned) Element 1
2	Read Number of Registers (n)	2	Data (returned) Element 2
3		3	Data (returned) Element 3
4		4
5		5
...	
...	
n		n	Data (returned) Element n

The number of registers given in the output block must be such that the implied Modbus data fits entirely within the block as specified by its data length, else a format alarm is raised. This means that for a Modbus read function, the number of registers must not exceed the data length, and for a Modbus write function, the number of registers must not exceed the data length minus 2.

Modbus Gateway Read Function Codes		
Function Code	To Access Register Bank	To address AFC Slave
16	Holding	Primary
17	Holding	Virtual
18	Input	Primary
19	Input	Virtual

Special Notes

- If Bit 0 (F) of the Block ID element of the input Function Block is set:
 - Indicates Modbus exception: Illegal function (requested number of registers is 0).
- If Bit 1 (A) of the Block ID element of the Input Function Block is set:
 - Indicates Modbus exception: Illegal address (an attempt to access one or more non-existent Modbus registers).
- If Bit 2 (V) of the Block ID element of the Input Function Block is set:
 - Indicates Modbus exception: Illegal data value (never occurs for a "read" function).

Note: If any of the above three bits are set, it indicates that no action on the OFB has been taken by the AFC module.

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.12 Function Block Definition - 20, 21: Modbus Gateway Write

Output Function Block ID

Function Code 20: Write to Modbus Registers in the PRIMARY Slave HOLDING register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Func. Code: 20				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0	0
Example: A block length of 48 (00110000 binary) shown.																

Function Code 21: Write to Modbus Registers in the VIRTUAL Slave HOLDING register bank.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Func. Code: 21				bnk	slv	inp	out	Data Block Length: User defined							
Value	0	1	0	1	0	1	0	0	0	0	1	1	0	0	0	0

Input Function Block ID

When the AFC completes processing the output function block, the input function block ID is returned with the Modbus Message block as described above. The AFC provides a number of indicators to allow you to design the PLC logic to take appropriate action once the controller receives the message in the returned IFB.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 21								Reserved – Not used					V	A	F
Value	0	1	0	1	0	1	0	0	0	0	0	0	0	0/1	0/1	0/1
														See the Special Notes below		
														↑ ↑ ↑		

Function Block Name:	Modbus Gateway Write
Function Block ID:	20 and 21
Target:	Global
Data Block Length:	User defined - from 3 to 244
Total Block Length:	(User defined +1) Elements

Description

These two function codes allow you to write data from the controller to the vast address space of the AFC module for its consumption. The two functions are designed to access the two separate holding register banks by assigning unique function codes for each type of access. These two function codes are tabulated below in this section.

This block performs an arbitrary data transfer between the PLC and the AFC data table. Any data transfer that can be performed with a Modbus command to either AFC slave through any of the module's ports may be implemented using the gateway.

Any data words not relevant to the command are ignored upon output (to the AFC) and zero upon input (from the AFC). Gateway data transfer may be controlled through the use of the "ignore output" bit of the block ID. The "skip input" bit has no function and is ignored.

MB Gateway Write: OFB		MB Gateway Write: IFB	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length	0	Block ID - Function Code & Status
1	Modbus Register Address	1	Reserved
2	Write Number of Registers (m)	2	Reserved
3	Data (write) Element 1	3	Reserved
4	Data (write) Element 2	4	Reserved
5	Data (write) Element 3	5	Reserved
...	Reserved
...	Reserved
n	Data (write) Element (n-2)	n	Reserved

Note: The number of registers for a Gateway Write Function block must not exceed two less than the data block length declared in the OFB ID.

Modbus Gateway Write Function Codes

Function Code	To Write to Register Bank	To address AFC Slave
20	Holding	Primary
21	Holding	Virtual

Special Notes

- If Bit 0 (F) of the Block ID element of the Input Function Block is set:
 - Indicates Modbus exception: Illegal function (either the requested number of registers is 0 or all the registers addressed are read-only).
- If Bit 1 (A) of the Block ID element of the Input Function Block is set:
 - Indicates Modbus exception: Illegal address (an attempt to access one or more non-existent Modbus registers).
- If Bit 2 (V) of the Block ID element of the Input Function Block is set:
 - Indicates Modbus exception: Illegal data value (an attempt to write a value to a register that is out of range or otherwise invalid for its intended target OR an attempt to change a sealable parameter while the Event Log is full).

Note: If any of the above three bits are set, it indicates that no action on the OFB has been taken by the AFC module.

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

11.2.13 Function Block Definition - 24, 25, 26: Modbus Master

Output Function Block ID

Function Code 24: Modbus Master READ from slave's HOLDING register or COIL bank

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 24				d	r	inp	out	Length							
Value	0	1	1	0	0	0	0	0	0	0	1	1	0	0	0	0
Example: A block length of 48 (00110000 binary) shown.																


Function Code 25: Modbus Master READ from slave's INPUT register or INPUT status bank

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 25				d	r	inp	out	Length							
Value	0	1	1	0	0	1	0	0	0	0	1	1	0	0	0	0

Function Code 26: Modbus Master WRITE to slave's HOLDING register or COIL bank

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 26				d	r	inp	out	Length							
Value	0	1	1	0	1	0	0	0	0	0	1	1	0	0	0	0

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 24				d	r										p
Value	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0/1
Bit = 1 if a previous transaction is still pending 																

Function Block Name:	Modbus Master
Function Block ID:	24, 25, 26
Target:	Global
Data Block Length:	User defined - from 6 to 130
Total Block Length:	(User defined +1) Element

Description

This block performs an arbitrary data transfer between the ControlLogix and external Modbus slaves connected to AFC Port 3, provided that port 3 is configured as a Modbus master. Any data transfer to or from a slave's holding registers, input registers, output coils, or input status may be implemented using this function; equivalent Modbus function codes are 1, 2, 3, 4, 15, and 16. In addition, capability is provided for access to a slave's "long remote" (32-bit) registers where the slave implements them; in particular, Enron-style long integer (5000 series) and floating-point (7000 series) registers are accessible. Any data words not relevant to the command are ignored upon output (to the AFC) and zero upon input (from the AFC). The "transaction number" is provided as a resource to the PLC programmer in implementing multiplexing, if required; the AFC copies it verbatim from output to input and does not use it in any other manner.

Use the "ignore output" bit of the block ID to control the scheduling of master mode transactions and the "skip input" bit to control the retrieval of the result. The output block is processed and a master transaction scheduled when "ignore output" is clear and there is no transaction currently pending; in all other cases the output is ignored. While "skip input" is clear, a pending transaction is indicated by the "pending" bit, if no transaction is pending then the results of the latest transaction are returned, and in all cases the "transaction number" of the latest or current transaction is returned.

Modbus Master OFB		Modbus Master IFB	
Element	Attribute	Element	Attribute
0	Block ID - Function Code & Length.	0	Block ID - Function Code & Length
1	Transaction Number	1	Transaction Number Echo
2	Data Item Size & Swap Options (see below)	2	Error Code (see below)
3	Slave Address	3	Data Read Element 1 (if used)
4	Modbus Register Address	4	Data Read Element 2 (if used)
5	Number of Data Items	5	Data Read Element 3 (if used)
6	Data Write Element 1 (if used)	6	Data Read Element 4 (if used)
...
...

Data Item Size and Swap Options

The data item size is given by one of the following values:

- 0 - Bit (In the AFC, packed 16 to a word)
- 1 - Word (16-bit registers)
- 2 - long (32-bit items as register pairs)
- 3 - long remote (32-bit items as single registers)

To configure byte or word swapping, add 10 for byte swap (except size 0) and/or add 20 for word swap (sizes 2 and 3 only).

Error Codes

Code	Description
=0 -	No Error
>0	Modbus Exception Code or Communication Error Modbus Exception codes are issued by the responding slave and listed in commonly available Modbus protocol manuals; they lie between 1 and 127 and include: 1 - Illegal Function 2 - Illegal Address 3 - Illegal Data Value Communication Errors are issued by the AFC: 500 - CTS Timeout 501 - Receive Timeout 502 - Bad Framing 503 - Buffer Overrun 504 - Bad Checksum/CRC 505 - Wrong Slave 506 - Wrong Function Code 507 - Wrong Length
<0	Configuration, Parameter, or Logic Error: -1 - Master Port not configured -2 - Master Port never used -3 - Bad Slave Address -4 - Bad Direction/ Target -5 - Bad Datum Size / Swap Options -6 - Bad Number of Data Items

Because the output block always contains the five words described above, the block's data length must be at least 5, else a format alarm is raised.

The number of data items given in the output block must be such that the implied Modbus data fits entirely within the block as specified by its data length, else a format alarm is raised. This means that for a Modbus read function, the number of words occupied by the data must not exceed the block data length minus 2, and for a Modbus write function, the number of words occupied by the data must not exceed the block data length minus 5.

11.2.14 Function Block Definition - 28, 29: Disable/Enable Meters

Output Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 28						inp	out	n/a							
Value	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Input Function Block ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Attrib	Function Code: 28						V		n/a							
Value	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Function Block Name:	Disable/Enable Meters
Function Block ID:	Disable 28 28672 decimal 7000 Hex Enable 29 29696 decimal, 7400 Hex
Target:	Global
Data Block Length:	1 Element
Total Block Length:	2 Elements

Description

This block issues Meter Disable (28) or Meter Enable (29) signals to the meter logic in the AFC. A signal is an instruction for the AFC to perform a function once. Signals are latched in the AFC and remain pending until they are discharged. The OFB data element must be loaded with the bits set to correspond to the meters to be disabled or enabled. The AFC returns the data element with the bitmap of the enabled meters.

To schedule a signal, the Processor should latch the appropriate bit in the output signal word. To prevent the accidental reissuing of signals, the Processor should clear to zero the output signal word immediately after writing it to the module.

Signals: Output Function Block OFB		Signals: Input Function Block IFB	
Element	Attribute	Element	Attribute
0	Block ID - Function Code	0	Block ID - Function Code
1	Signal Bits	1	Bitmap of enabled meters
	Bit #		Meter Signal Bits
	0		Disable/Enable Meter 1
	1		Disable/Enable Meter 2
	2		Disable/Enable Meter 3
	3		Disable/Enable Meter 4
	4		Disable/Enable Meter 5
	5		Disable/Enable Meter 6
	6		Disable/Enable Meter 7
	7		Disable/Enable Meter 8
	8		Disable/Enable Meter 9
	9		Disable/Enable Meter 10
	10		Disable/Enable Meter 11
	11		Disable/Enable Meter 12
	12		Disable/Enable Meter 13
	13		Disable/Enable Meter 14
	14		Disable/Enable Meter 15
	15		Disable/Enable Meter 16

Special Notes

- If bit 9 (V) of the block ID element of the Input Function Block is 0, this indicates that the action was performed and event logged, or no action was required.
- If bit 9 (V) of the block ID element of the Input Function Block is 1, this indicates that the event log is full - action was not performed.

Note: Refer to Function Block Structure (page 168) for information on Bit 8 and 9 settings.

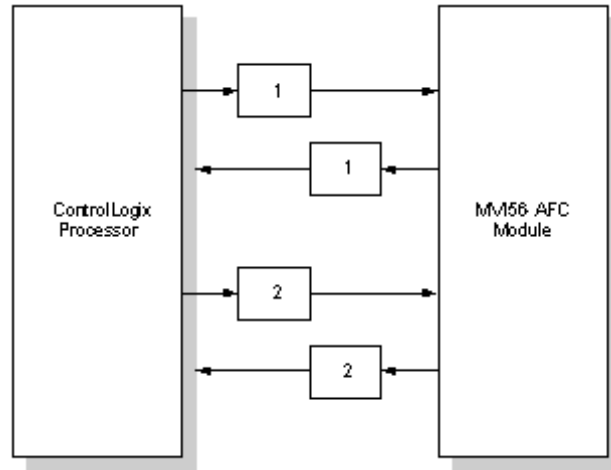
12 MVI56-AFC Sample Logic

In This Chapter

- ❖ Sample Logic Overview 206
- ❖ Using the Sample Add-On Instruction 211
- ❖ ControlLogix Sample Logic Details..... 220

12.1 Sample Logic Overview

The logic basically sends output blocks to the MVI56-AFC and receives the input blocks from the module. Each block contains a Block Sequence number that identifies the block. The input response block sent by the module will also contain the same Block Sequence Number as the previous output block.



The ladder logic performs the following sequence:

- 1 Receives the input block from the MVI56-AFC
- 2 Copies the input block content from the input buffer to the controller tags based on the Block Sequence Number
- 3 Increments the next Block Sequence Number
- 4 Builds the next output block with the new Sequence Number
- 5 Sends the new output block to the module.

The sample ladder uses the following Block Sequence Number to generate output blocks:

Block Sequence Number	Description
1	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 1) and read profile (Meter 1)
2	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 2) and read profile (Meter 2)
3	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 3) and read profile (Meter 3)
4	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 4) and read profile (Meter 4)
5	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 5) and read profile (Meter 5)
6	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 6) and read profile (Meter 6)
7	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 7) and read profile (Meter 7)
8	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 8) and read profile (Meter 8)
9	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 9) and read profile (Meter 9)
10	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 10) and read profile (Meter 10)
11	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 11) and read profile (Meter 11)
12	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 12) and read profile (Meter 12)
13	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 13) and read profile (Meter 13)
14	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 14) and read profile (Meter 14)
15	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 15) and read profile (Meter 15)
16	Process variables, enable/disable (all Meters). Write Molar Analysis (Meter 16) and read profile (Meter 16)
17	Modbus Gateway Block 0 (if configured)
18	Modbus Gateway Block 1 (if configured)
19	Modbus Gateway Block 2 (if configured)
20	Modbus Gateway Block 3 (if configured)
21	Modbus Gateway Block 4 (if configured)
22	Modbus Gateway Block 5 (if configured)
23	Modbus Gateway Block 6 (if configured)
24	Modbus Gateway Block 7 (if configured)
25	Modbus Gateway Block 8 (if configured)
25	Modbus Gateway Block 9 (if configured)

Each Process Block (Block Sequence Number 1 to 16) is organized as follows:

12.1.1 Process Block (uses Transaction Numbers from 1 to 16)

Controller Tag Begin	Controller Tag End	Description
AFC56.outputbuffer[0]		Transaction Number [x]
AFC56.outputbuffer[1]		Block Length (226)
AFC56.outputbuffer[2]	AFC56.outputbuffer[13]	Meter 1 Process Variables
AFC56.outputbuffer[14]	AFC56.outputbuffer[25]	Meter 2 Process Variables
AFC56.outputbuffer[26]	AFC56.outputbuffer[37]	Meter 3 Process Variables
AFC56.outputbuffer[38]	AFC56.outputbuffer[49]	Meter 4 Process Variables
AFC56.outputbuffer[50]	AFC56.outputbuffer[61]	Meter 5 Process Variables
AFC56.outputbuffer[62]	AFC56.outputbuffer[73]	Meter 6 Process Variables
AFC56.outputbuffer[74]	AFC56.outputbuffer[85]	Meter 7 Process Variables
AFC56.outputbuffer[86]	AFC56.outputbuffer[97]	Meter 8 Process Variables
AFC56.outputbuffer[98]	AFC56.outputbuffer[109]	Meter 9 Process Variables
AFC56.outputbuffer[110]	AFC56.outputbuffer[121]	Meter 10 Process Variables
AFC56.outputbuffer[122]	AFC56.outputbuffer[133]	Meter 11 Process Variables
AFC56.outputbuffer[134]	AFC56.outputbuffer[145]	Meter 12 Process Variables
AFC56.outputbuffer[146]	AFC56.outputbuffer[157]	Meter 13 Process Variables
AFC56.outputbuffer[158]	AFC56.outputbuffer[169]	Meter 14 Process Variables
AFC56.outputbuffer[170]	AFC56.outputbuffer[181]	Meter 15 Process Variables
AFC56.outputbuffer[182]	AFC56.outputbuffer[193]	Meter 16 Process Variables
AFC56.outputbuffer[194]	AFC56.outputbuffer[195]	Enable Meters
AFC56.outputbuffer[196]	AFC56.outputbuffer[197]	Disable Meters
AFC56.outputbuffer[198]	AFC56.outputbuffer[199]	Reset Acc, Write Archives
AFC56.outputbuffer[200]	AFC56.outputbuffer[224]	Molar Analysis for Meter [x]
AFC56.outputbuffer[225]	AFC56.outputbuffer[227]	Read Meter [x] Profile
AFC56.outputbuffer[226]	AFC56.outputbuffer[246]	Not Used
AFC56.outputbuffer[247]		Transaction Number [x]

After the module receives this output block, it performs the required calculations and then builds the input response block to be sent to the processor. The module uses the same offsets as the output block. For example: the calculation results for Meter 9 will be copied at offset 98 in the input block.

12.1.2 Modbus Gateway Block (uses Transaction Numbers from 17 to 25)

Each Modbus Gateway block has the following structure:

Controller Tag Begin	Controller Tag End	Description
AFC56.outputbuffer[0]		Transaction Number [y]
AFC56.outputbuffer[1]		Block Length (see note)
AFC56.outputbuffer[2]		Modbus Gateway BOD
AFC56.outputbuffer[3]		Start Register
AFC56.outputbuffer[4]		Register Count
AFC56.outputbuffer[5]	AFC56.outputbuffer[204]	Reserved for Modbus Gateway
AFC56.outputbuffer[205]	AFC56.outputbuffer[246]	Not Used
AFC56.outputbuffer[247]		Transaction Number [y]

Note: The block length will depend on the Register Count parameter you entered.

12.1.3 Wallclock Block (uses Transaction Number =99)

The wallclock block has the following structure:

Controller Tag Begin	Controller Tag End	Description
AFC56.outputbuffer[0]		Transaction Number = 99
AFC56.outputbuffer[1]		Block Length = 7
AFC56.outputbuffer[2]		Wallclock BOD
AFC56.outputbuffer[3]		Year
AFC56.outputbuffer[4]		Month
AFC56.outputbuffer[5]		Day
AFC56.outputbuffer[6]		Hour
AFC56.outputbuffer[7]		Minute
AFC56.outputbuffer[8]		Seconds
AFC56.outputbuffer[9]		Transaction Number = 99

12.1.4 Sample MVI56-AFC Logic Tasks

The sample ladder performs several important tasks for the MVI56-AFC operation. For most of the applications, you should not have to perform any modifications to the sample ladder logic.

The sample ladder logic performs the following tasks:

- 1** Transfer the process variables from the processor to the MVI56-AFC for all 16 meters.
- 2** Transfer the calculation results from the MVI56-AFC to the processor for all 16 meters.
- 3** Write the Wallclock from the processor to the MVI56-AFC
- 4** Enable all 16 meters
- 5** Disable all 16 meters
- 6** Displays the current enable/disable status of all 16 meters
- 7** Transfer the molar analysis data (gas only) for each meter
- 8** Reset all 4 Resettable Accumulators for each meter
- 9** Write a Daily or Hourly Archive
- 10** Select a meter stream
- 11** Read each meter profile (meter type, product group and selected stream)
- 12** Transfer up to 2000 words of data between the processor and the Primary or Virtual Modbus Slaves.
- 13** Set the processor date and time information using a controller tag as source
- 14** Read the meter alarms
- 15** Read the site status
- 16** Request Modbus master commands to remote connected Modbus slaves
- 17** Read Modbus pass-through messages from the module

12.2 Using the Sample Add-On Instruction

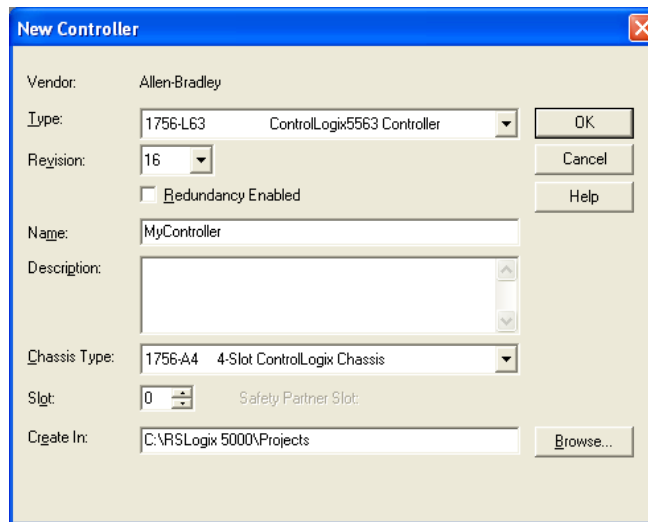
Before You Start

Make sure to check the following

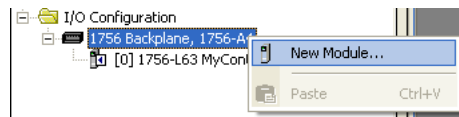
- 1 Download the sample program file (PS56AFC.L5X) from the web site or refer to the ProSoft Solutions CD-ROM
- 2 Make sure you have RSLogix5000 version 16 installed on your PC
- 3 Make sure that your ControlLogix processor has firmware version 16

12.2.1 Import Procedure

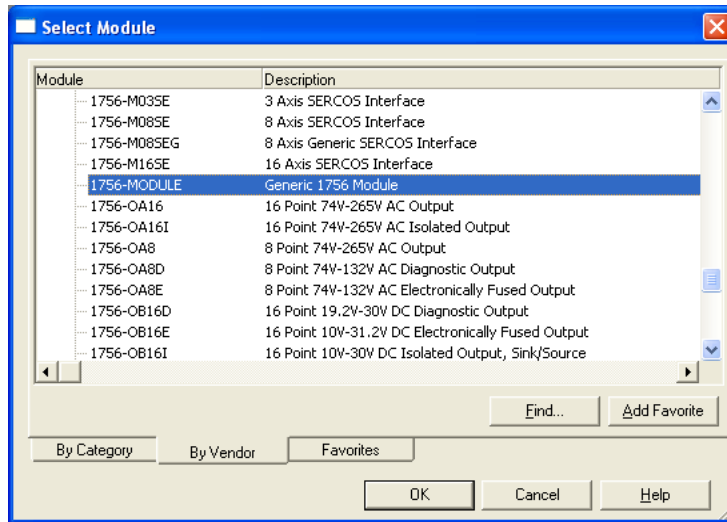
- 1 Create a new RSLogix 5000 project. Select revision 16 as follows.



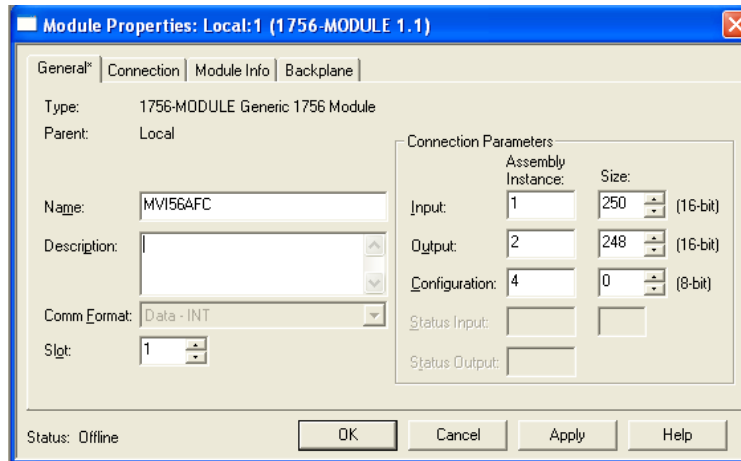
- 2 In the Controller Organization window, expand the I/O Configuration folder.
- 3 Select 1756 Backplane, and then click the right mouse button to open a shortcut menu.
- 4 On the shortcut menu, choose New Module...



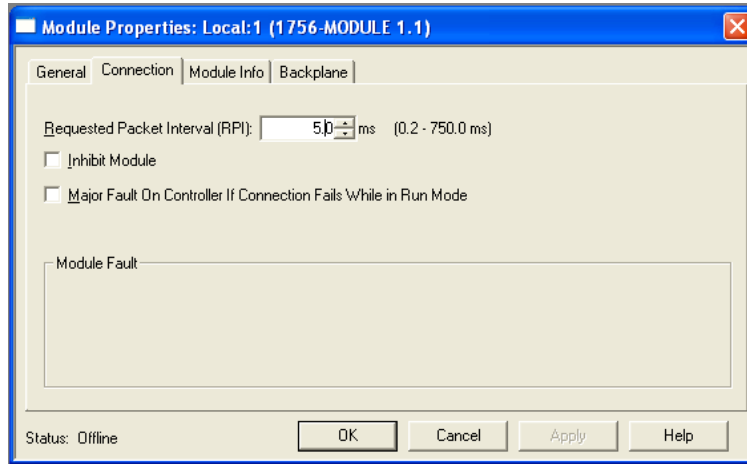
5 On the Select Module dialog box, select 1756-MODULE



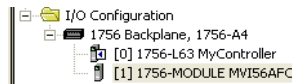
6 Configure the module using the settings in the following illustration.



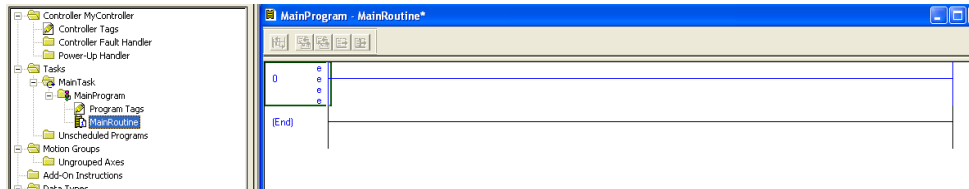
Select the RPI time (min 5ms). Click OK to confirm.



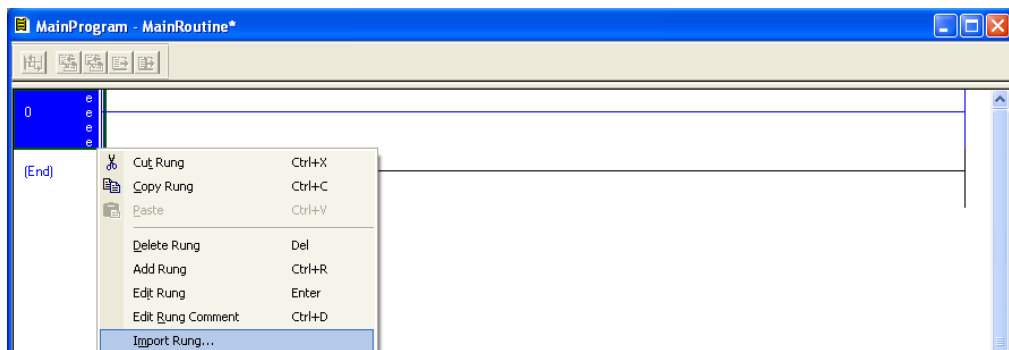
The MVI56-AFC module will now be visible in the I/O Configuration folder.



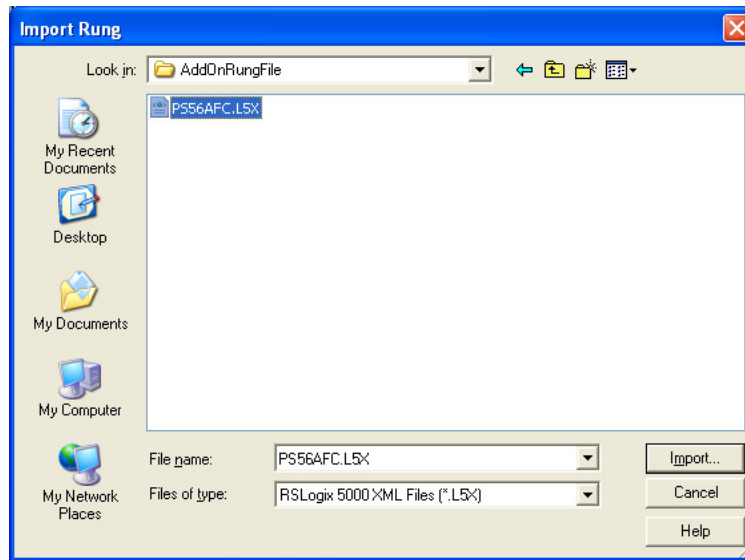
7 Expand the Tasks folder, and then select MainProgram.



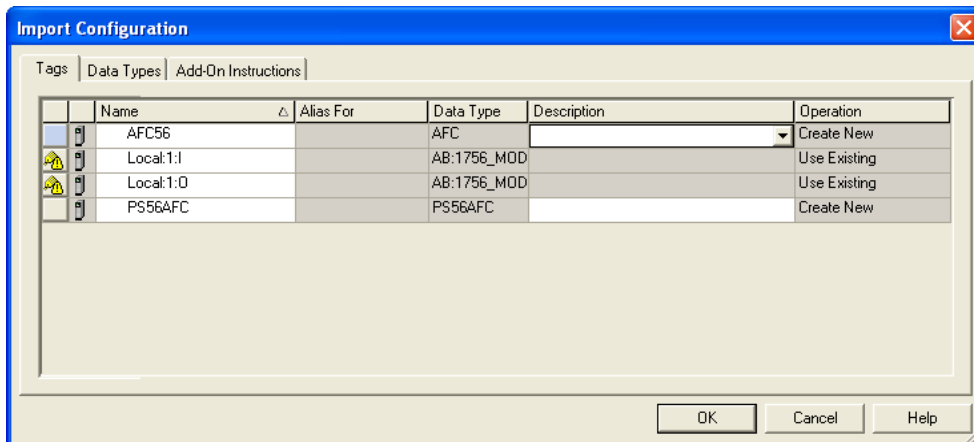
8 Click the right mouse button to open a shortcut menu, and then choose Import Rung. This action opens the Import Rung dialog box.



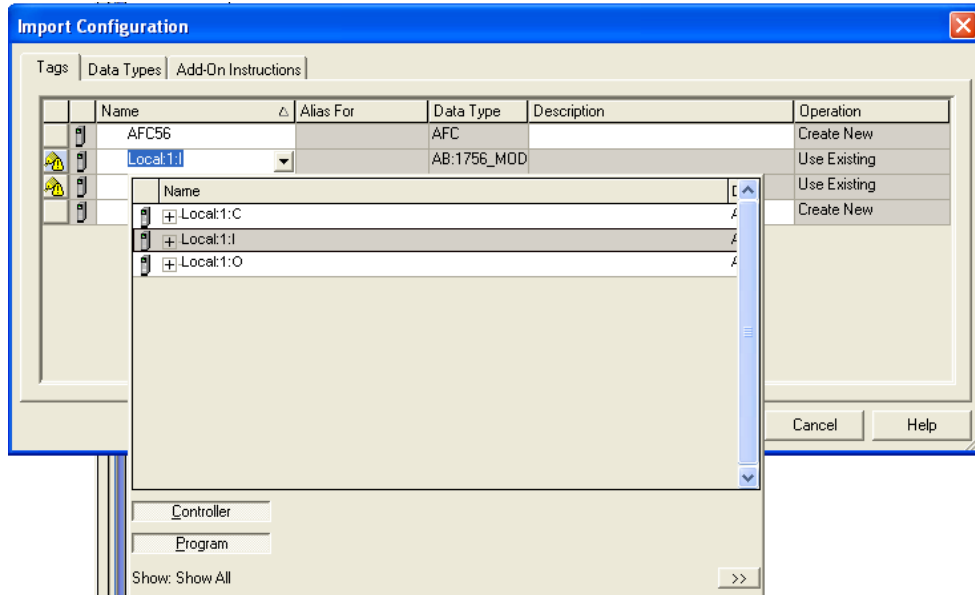
- 9 In the Import Rung dialog box, select PS56AFC.L5X, and then click Import.



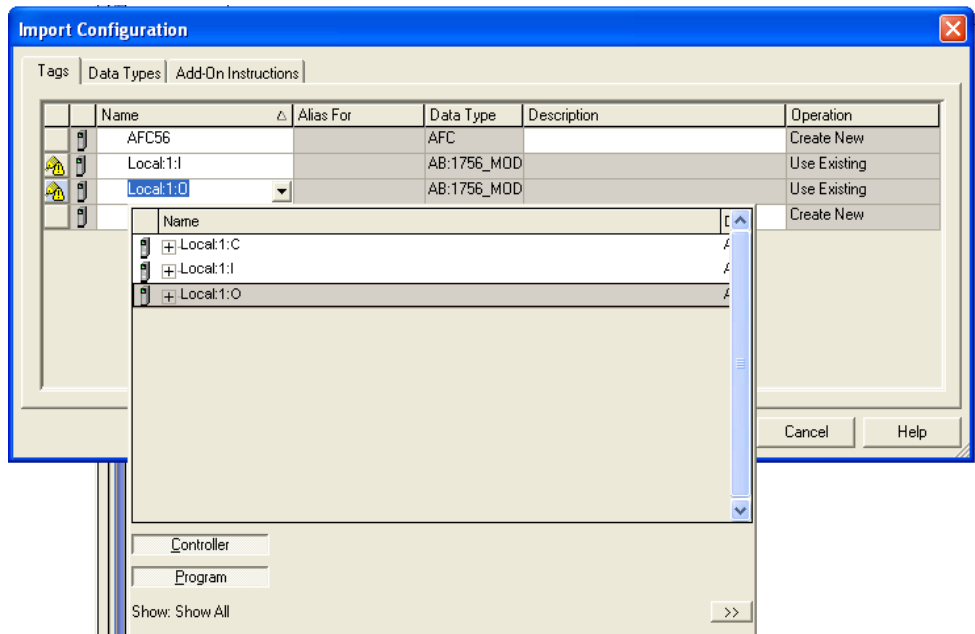
- 10 In the Import Configuration dialog box, set up the tags as shown in the following illustration.



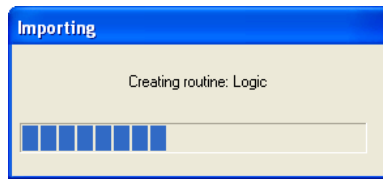
- Select the input image file to the file associated to the module that you previously created (this example is for a module located in slot 1 of the local rack).



- Select the output image file to the file associated to the module that you previously created (this example is for a module located in slot 1 of the local rack).



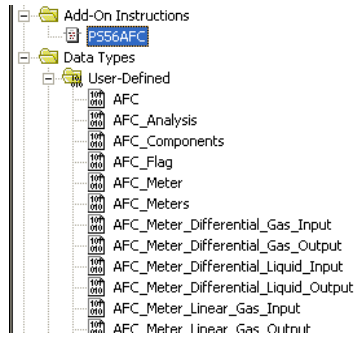
13 Click OK to import the rung. The following message box shows that the import is in progress.



When the import is completed, the Add-On instruction will now be visible in the newly imported ladder rung



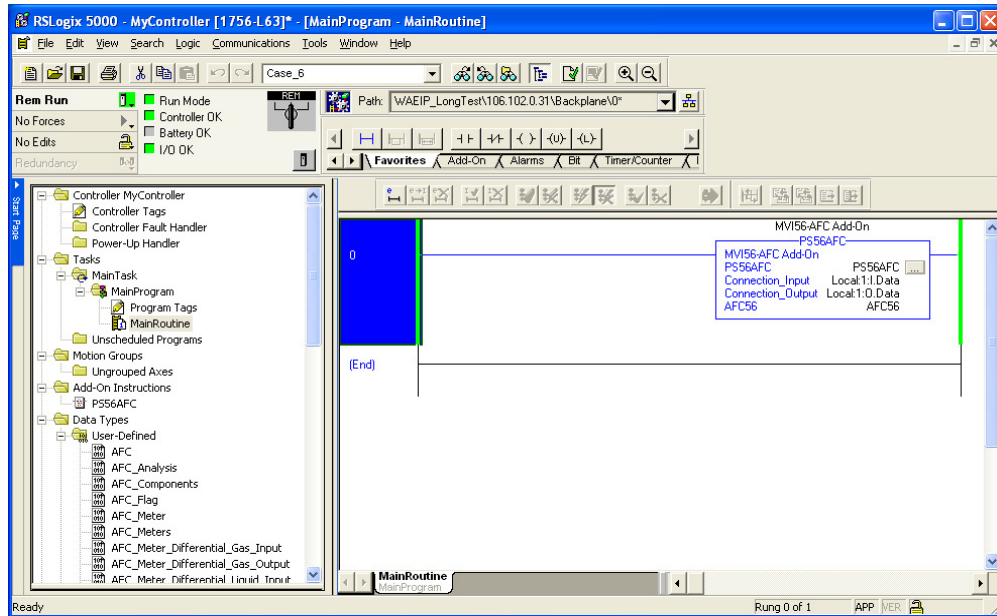
The procedure has also imported User-Defined data types that will be used by the sample program.



The procedure has also imported controller tags that will be used by the sample program

Scope: MyController		Shgw...	Show All			
Name	Value	Force Mask	Style	Data Type		
⊕ AFC56		{ ... }	{ ... }	AFC		
⊕ Local:1:C		{ ... }	{ ... }	AB:1756_MODULE:C:0		
⊕ Local:1:I		{ ... }	{ ... }	AB:1756_MODULE_INT_500Bytes:I:0		
⊕ Local:1:O		{ ... }	{ ... }	AB:1756_MODULE_INT_496Bytes:O:0		
⊕ PS56AFC		{ ... }	{ ... }	PS56AFC		

- 14 The import procedure is now completed. Save your project and download it to the ControlLogix processor.



12.3 ControlLogix Sample Logic Details

This section shows how you can extract important information from the sample logic without requiring you to know the details of how the sample logic actually works. For most applications it will be enough to refer only to the controller tags in order to perform the tasks.

12.3.1 Enable/Disable Status

Each meter run will only perform flow calculation while it is enabled. However, you cannot change a meter type, product group or units while the meter is enabled. In order to accomplish this, you have to disable the meter, change meter type, product or units and then enable the meter again. The meters can be enabled or disabled from logic or AFC Manager.

The logic constantly reads each meter enable/disable status from the MVI56-AFC.

Refer to *AFC56.EnableStatus* data structure for each meter status. Each variable should be interpreted as:

AFC56.EnableStatus.Meterx = 0 => Meter x is Disabled

AFC56.EnableStatus.Meterx = 1 => Meter x is Enabled

AFC56.EnableStatus		{...}
-AFC56.EnableStatus.Meter1		1
-AFC56.EnableStatus.Meter2		1
-AFC56.EnableStatus.Meter3		0
-AFC56.EnableStatus.Meter4		0
-AFC56.EnableStatus.Meter5		0
-AFC56.EnableStatus.Meter6		0
-AFC56.EnableStatus.Meter7		0
-AFC56.EnableStatus.Meter8		0
-AFC56.EnableStatus.Meter9		0
-AFC56.EnableStatus.Meter10		0
-AFC56.EnableStatus.Meter11		0
-AFC56.EnableStatus.Meter12		0
-AFC56.EnableStatus.Meter13		0
-AFC56.EnableStatus.Meter14		0
-AFC56.EnableStatus.Meter15		0
-AFC56.EnableStatus.Meter16		0

In the previous example, Meter 1 and Meter 2 are enabled. All other meters are disabled.

12.3.2 Disable Meter

Each meter can be disabled through logic. Refer to the *AFC56.DisableMeter* data structure. Toggle each *AFC56.DisableMeter.Meterx* controller tag in order to command the meter to be enabled.

The logic will continuously analyze each meter enable status. After the logic determines that a specific meter is disabled (*AFC56.DisableStatus.Meterx = 1*) the command bit (*AFC56.DisableMeter.Meterx*) will be unlatched.

[-] AFC56.DisableMeter		{...}
-AFC56.DisableMeter.Meter1		0
-AFC56.DisableMeter.Meter2		0
-AFC56.DisableMeter.Meter3		0
-AFC56.DisableMeter.Meter4		0
-AFC56.DisableMeter.Meter5		0
-AFC56.DisableMeter.Meter6		0
-AFC56.DisableMeter.Meter7		0
-AFC56.DisableMeter.Meter8		0
-AFC56.DisableMeter.Meter9		0
-AFC56.DisableMeter.Meter10		0
-AFC56.DisableMeter.Meter11		0
-AFC56.DisableMeter.Meter12		0
-AFC56.DisableMeter.Meter13		0
-AFC56.DisableMeter.Meter14		0
-AFC56.DisableMeter.Meter15		0
-AFC56.DisableMeter.Meter16		0

Note: DO NOT create a rung in logic to constantly disable the meter. The command bit should be toggled only once in order to disable the meter.

12.3.3 Enable Meter

Each meter can be enabled through logic. Refer to the *AFC56.EnableMeter* data structure. Toggle each *AFC56.EnableMeter.Meterx* controller tag in order to command the meter to be enabled.

The logic will continuously analyze each meter enable status. After the logic determines that a specific meter is enabled (*AFC56.EnableStatus.Meterx = 1*) the command bit (*AFC56.EnableMeter.Meterx*) will be unlatched.

[-] AFC56.EnableMeter		{...}
-AFC56.EnableMeter.Meter1		0
-AFC56.EnableMeter.Meter2		0
-AFC56.EnableMeter.Meter3		0
-AFC56.EnableMeter.Meter4		0
-AFC56.EnableMeter.Meter5		0
-AFC56.EnableMeter.Meter6		0
-AFC56.EnableMeter.Meter7		0
-AFC56.EnableMeter.Meter8		0
-AFC56.EnableMeter.Meter9		0
-AFC56.EnableMeter.Meter10		0
-AFC56.EnableMeter.Meter11		0
-AFC56.EnableMeter.Meter12		0
-AFC56.EnableMeter.Meter13		0
-AFC56.EnableMeter.Meter14		0
-AFC56.EnableMeter.Meter15		0
-AFC56.EnableMeter.Meter16		0

Note: DO NOT create a rung in logic to constantly enable the meter. The command bit should be toggled only once in order to enable the meter.

12.3.4 Wallclock

After the module powers up, it will not perform flow calculation until it receives valid wallclock information from the processor. The sample ladder logic uses the processor internal clock as the source of the wallclock information.

Configure the processor time and date information:

- 1 Right-Click on Controller MVI56-AFC folder
- 2 Click on Properties
- 3 Select the Date/Time tab
- 4 Enter a valid date and time information.

After the *AFC56.Flags.AFC_Set_Clock* bit is toggled, the logic will move the date and time information from the processor to the MVI56-AFC module.

[-] AFC56.Flags	{...}
- AFC56.Flags.AFC_Stopped	0
- AFC56.Flags.AFC_Set_Clock	0
- AFC56.Flags.AFC_OutputError	0
+ AFC56.Flags.AFC_InputSequenceNum	0
- AFC56.Flags.plc_set_clock	0

The *AFC56.Flags.AFC_Set_Clock* bit is latched in the power up routine, in order to guarantee that the module will be up and running after power up.



After the ladder logic receives the input block back from the module it unlatches the *AFC56.Flags.AFC_Set_Clock* bit.

You may want to periodically synchronize the processor and the module's wallclock, especially when the date and time information is received from a remote station. In this case, further ladder logic is required from you to periodically toggle the *AFC56.Flags.AFC_Set_Clock* bit.

12.3.5 Meter Profile

The logic constantly reads each meter profile. The meter profile informs each meter type (linear or differential), product group (gas or liquid) and currently selected stream (1 through 4).

The *AFC56.Meters[x].Profile* controller tag stores the profile information for each meter run:

AFC56.Meters[0].Profile - Meter 1 Profile

AFC56.Meters[1].Profile - Meter 2 Profile

...

AFC56.Meters[15].Profile - Meter 16 Profile

The controller tags are interpreted as follows:

Controller Tag	Value	Description
<i>AFC56.Meters[x].Profile.MeterType</i>	0	Meter x is Differential Meter
<i>AFC56.Meters[x].Profile.MeterType</i>	1	Meter x is a Linear Meter
<i>AFC56.Meters[x].Profile.ProductGroup</i>	0	Meter x uses a Gas product
<i>AFC56.Meters[x].Profile.ProductGroup</i>	1	Meter x uses a Liquid product
<i>AFC56.Meters[x].Profile.SelectedStream</i>	1	Stream 1 is currently selected (active)
<i>AFC56.Meters[x].Profile.SelectedStream</i>	2	Stream 2 is currently selected (active)
<i>AFC56.Meters[x].Profile.SelectedStream</i>	3	Stream 3 is currently selected (active)
<i>AFC56.Meters[x].Profile.SelectedStream</i>	4	Stream 4 is currently selected (active)

The example below shows a situation where Meter 0 is configured as a linear meter and uses a gas product.

<i>AFC56.Meters[0].Profile.MeterType</i>	1	Decimal	BOOL
<i>AFC56.Meters[0].Profile.ProductGroup</i>	0	Decimal	BOOL
<i>AFC56.Meters[0].Profile.SelectedStream</i>	1	Decimal	INT

12.3.6 Meter Process Variables

In order to perform flow calculation the module must receive the meter process variables from the processor.

The process variables will depend on the meter type and product group. So the profile information (discussed before) is used to decide from which controller tag the variables will be copied from. The following options are used:

Meter Type	Product Group	Use this Controller Tag
Differential	Gas	<i>AFC56.Meters[x].Variables.DifferentialGas</i>
Differential	Liquid	<i>AFC56.Meters[x].Variables.DifferentialLiquid</i>
Linear	Gas	<i>AFC56.Meters[x].Variables.LinearGas</i>
Linear	Liquid	<i>AFC56.Meters[x].Variables.LinearLiquid</i>

The following illustration shows an example controller tag from RSLogix5000:

- AFC56.Meters[0].Variables	{...}
+ AFC56.Meters[0].Variables.BID	2#0010_0000_0000_0001
+ AFC56.Meters[0].Variables.DifferentialGas	{...}
+ AFC56.Meters[0].Variables.DifferentialLiquid	{...}
+ AFC56.Meters[0].Variables.LinearGas	{...}
+ AFC56.Meters[0].Variables.LinearLiquid	{...}

This means that all you have to do is identify where the variables will be copied from (based on the meter type and product group). The logic will automatically select the correct controller tags.

Note: In order to configure each meter type and product group, refer to the AFC Manager software tool.

For each possible combination, the following variables are used:

1) Meter Type = Differential & Product Group = Gas

Process Input	Controller Tag	Data Type
Temperature	AFC56.Meters[x].Variables.DifferentialGas.Temperature	REAL
Pressure	AFC56.Meters[x].Variables.DifferentialGas.Pressure	REAL
Differential Pressure	AFC56.Meters[x].Variables.DifferentialGas.DifferentialPressure	REAL

Where x can assume values between 0 and 15.

The following illustration shows an example for Meter 1:

- AFC56.Meters[0].Variables.DifferentialGas	{...}
- AFC56.Meters[0].Variables.DifferentialGas.Temperature	10.0
- AFC56.Meters[0].Variables.DifferentialGas.Pressure	20.0
- AFC56.Meters[0].Variables.DifferentialGas.Differential_Pressure	70.0

2) Meter Type = Differential & Product Group = Liquid

Process Input	Controller Tag	Data Type
Water %	AFC56.Meters[x].Variables. DifferentialLiquid.Water_Percent	INT
Temperature	AFC56.Meters[x].Variables. DifferentialLiquid.Temperature	REAL
Pressure	AFC56.Meters[x].Variables. DifferentialLiquid.Pressure	REAL
Diff Pressure	AFC56.Meters[x].Variables. DifferentialLiquid.DifferentialPressure	REAL
Density	AFC56.Meters[x].Variables. DifferentialLiquid.Density	REAL

Where x can assume values between 0 and 15.

- AFC56.Meters[0].Variables.DifferentialLiquid	{...}
+ AFC56.Meters[0].Variables.DifferentialLiquid.Water_Percent	7
- AFC56.Meters[0].Variables.DifferentialLiquid.Temperature	42.3
- AFC56.Meters[0].Variables.DifferentialLiquid.Pressure	23.1
- AFC56.Meters[0].Variables.DifferentialLiquid.Differential_Pressure	52.21
- AFC56.Meters[0].Variables.DifferentialLiquid.Density	442.0

3) Meter Type = Linear & Product Group = Gas

Process Input	Controller Tag	Data Type
Temperature	AFC56.Meters[x].Variables.LinearGas.Temperature	REAL
Pressure	AFC56.Meters[x].Variables.LinearGas.Pressure	REAL
Pulse Count	AFC56.Meters[x].Variables.LinearGas.Meter_Pulses	DINT
Pulse Frequency	AFC56.Meters[x].Variables.LinearGas.Frequency	REAL

Where x can assume values between 0 and 15.

The following illustration shows an example for Meter 1:

- AFC56.Meters[0].Variables.LinearGas	{...}
- AFC56.Meters[0].Variables.LinearGas.Temperature	12.0
- AFC56.Meters[0].Variables.LinearGas.Pressure	41.0
+ AFC56.Meters[0].Variables.LinearGas.Meter_Pulses	111
- AFC56.Meters[0].Variables.LinearGas.Pulse_Frequency	122.0

4) Meter Type = Linear & Product Group = Liquid

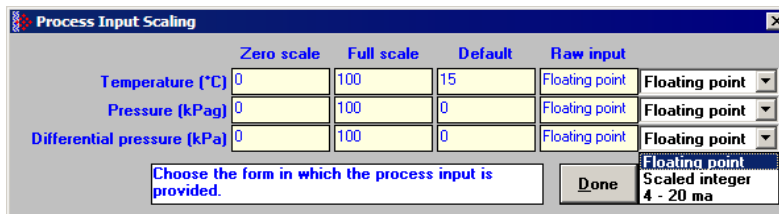
Process Input	Controller Tag	Data Type
Water Percent	AFC56.Meters[x].Variables.LinearLiquid.Water_Percent	INT
Temperature	AFC56.Meters[x].Variables.LinearLiquid.Temperature	REAL
Pressure	AFC56.Meters[x].Variables.LinearLiquid.Pressure	REAL
Pulse Count	AFC56.Meters[x].Variables.LinearLiquid.Meter_Pulses	DINT
Density	AFC56.Meters[x].Variables.LinearLiquid.Density	REAL
Pulse Frequency	AFC56.Meters[x].Variables.LinearLiquid.Pulse_Frequency	REAL

Where x can assume values between 0 and 15.

The following illustration shows an example for Meter 1:

- AFC56.Meters[0].Variables.LinearLiquid	{...}
+ AFC56.Meters[0].Variables.LinearLiquid.Water_Percent	12
- AFC56.Meters[0].Variables.LinearLiquid.Temperature	12.0
- AFC56.Meters[0].Variables.LinearLiquid.Pressure	31.0
+ AFC56.Meters[0].Variables.LinearLiquid.Meter_Pulses	33
- AFC56.Meters[0].Variables.LinearLiquid.Density	222.0
- AFC56.Meters[0].Variables.LinearLiquid.Pulse_Frequency	12.0

Important: The sample logic is configured considering the input variables with floating point format (default from AFC Manager). However, if Scaled Integer or 4 to 20 mA formats are used, change the meter variables format from floating point (REAL) to 32-bit long integer (DINT) in the logic.



The *AFC56.Meters[0].Variables.BID* is the function block ID for each meter. This value is automatically calculated by the logic, so do not force any value to this controller tag.

12.3.7 Meter Calculation Results

After the module has performed the AGA/API calculation, all results are moved to the processor. The logic will move the calculation results to the controller tags described in this section.

The calculation results will depend on the meter type and product group. So the profile information (discussed before) is used to decide to which controller tag the results will be copied to. The following options are used:

Meter Type	Product Group	Use this Controller Tag
Orifice	Gas	<i>AFC56.Meters[x].Results.DifferentialGas</i>
Orifice	Liquid	<i>AFC56.Meters[x].Results.DifferentialLiquid</i>
Pulse	Gas	<i>AFC56.Meters[x].Results.PulseGas</i>
Pulse	Liquid	<i>AFC56.Meters[x].Results.PulseLiquid</i>

The following shows a controller tag illustration from RSLogix5000:

- AFC56.Meters[0].Results	{...}
+ AFC56.Meters[0].Results.BID	2#0010_0000_0010_0001
+ AFC56.Meters[0].Results.Alarms	2#0000_0000_0000_0000
+ AFC56.Meters[0].Results.DifferentialGas	{...}
+ AFC56.Meters[0].Results.DifferentialLiquid	{...}
+ AFC56.Meters[0].Results.LinearGas	{...}
+ AFC56.Meters[0].Results.LinearLiquid	{...}

This means that all you have to do is to identify where to copy the results (based on the meter type and product group). The logic will automatically select the correct controller tags.

Note: In order to configure each meter type and product group, refer to the AFC manager software tool.

The following variables are used for each possible combination:

1) Meter Type = Differential & Product Group = Gas

Calculation Result	Controller Tag	Data Type
Net Accumulator	<i>AFC56.Meters[x].Results.DifferentialGas.Net_Accumulator</i>	DINT
Net Flow Rate	<i>AFC56.Meters[x].Results.DifferentialGas.Net_Flowrate</i>	REAL
Gross Flow Rate	<i>AFC56.Meters[x].Results.DifferentialGas.Gross_Flowrate</i>	REAL
Fpv	<i>AFC56.Meters[x].Results.DifferentialGas.Fpv</i>	REAL
Cprime	<i>AFC56.Meters[x].Results.DifferentialGas.Cprime</i>	REAL

Where x can assume values between 0 and 15.

The following is an example for Meter 1:

+ AFC56.Meters[0].Results.DifferentialGas.Net_Accumulator	1098
- AFC56.Meters[0].Results.DifferentialGas.Net_Flowrate	94.267624
- AFC56.Meters[0].Results.DifferentialGas.Gross_Flowrate	48.95955
- AFC56.Meters[0].Results.DifferentialGas.Fpv	1.0009998
- AFC56.Meters[0].Results.DifferentialGas.Cprime	1.9254185

2) Meter Type = Differential & Product Group = Liquid

Calculation Result	Controller Tag	Data Type
Net Accumulator	AFC56.Meters[x].Results. <i>DifferentialLiquid</i> .Net_Accumulator	DINT
Net Flow Rate	AFC56.Meters[x].Results. <i>DifferentialLiquid</i> .Net_Flowrate	REAL
Gross Accumulator	AFC56.Meters[x].Results. <i>DifferentialLiquid</i> .Gross_Accumulator	DINT
Gross Standard Accumulator	AFC56.Meters[x].Results. <i>DifferentialLiquid</i> .Standard_Accumulator	DINT
Mass Accumulator	AFC56.Meters[x].Results. <i>DifferentialLiquid</i> .Mass_Accumulator	DINT

Where x can assume values between 0 and 15.

The following shows an example for Meter 1:

+ AFC56.Meters[0].Results. <i>DifferentialLiquid</i> .Net_Accumulator	422
- AFC56.Meters[0].Results. <i>DifferentialLiquid</i> .Net_Flowrate	12.0
+ AFC56.Meters[0].Results. <i>DifferentialLiquid</i> .Gross_Accumulator	122
+ AFC56.Meters[0].Results. <i>DifferentialLiquid</i> .Gross_Standard_Accumulator	561
+ AFC56.Meters[0].Results. <i>DifferentialLiquid</i> .Mass_Accumulator	235

3) Meter Type = Linear & Product Group = Gas

Calculation Result	Controller Tag	Data Type
Net Accumulator	AFC56.Meters[x].Results. <i>LinearGas</i> .Net_Accumulator	DINT
Net Flow Rate	AFC56.Meters[x].Results. <i>LinearGas</i> .Net_Flowrate	REAL
Gross Flow Rate	AFC56.Meters[x].Results. <i>LinearGas</i> .Gross_Flowrate	REAL
Fpv	AFC56.Meters[x].Results. <i>LinearGas</i> .Fpv	REAL
Cprime	AFC56.Meters[x].Results. <i>LinearGas</i> .Cprime	REAL

Where x can assume values between 0 and 15.

The following shows an example for Meter 1:

+ AFC56.Meters[0].Results. <i>LinearGas</i> .Net_Accumulator	543
- AFC56.Meters[0].Results. <i>LinearGas</i> .Net_Flowrate	32.79
- AFC56.Meters[0].Results. <i>LinearGas</i> .Gross_Flowrate	42.64
- AFC56.Meters[0].Results. <i>LinearGas</i> .Fpv	1.0
- AFC56.Meters[0].Results. <i>LinearGas</i> .CPrime	1.0

4) Meter Type = Pulse & Product Group = Liquid

Calculation Result	Controller Tag	Data Type
Net Accumulator	AFC56.Meters[x].Results. <i>LinearLiquid</i> .Net_Accumulator	DINT
Net Flow Rate	AFC56.Meters[x].Results. <i>LinearLiquid</i> .Net_Flowrate	REAL
Gross Accumulator	AFC56.Meters[x].Results. <i>LinearLiquid</i> .Gross_Accumulator	DINT
Gross Standard Accumulator	AFC56.Meters[x].Results. <i>LinearLiquid</i> .Gross_Standard_Accumulator	DINT
Mass Accumulator	AFC56.Meters[x].Results. <i>LinearLiquid</i> .Mass_Accumulator	DINT

Where x can assume values between 0 and 15.

The following is an example for Meter 1:

+ AFC56.Meters[0].Results.LinearLiquid.Net_Accumulator	623
- AFC56.Meters[0].Results.LinearLiquid.Net_Flowrate	12.0
+ AFC56.Meters[0].Results.LinearLiquid.Gross_Accumulator	123
+ AFC56.Meters[0].Results.LinearLiquid.Gross_Standard_Accumulator	233
+ AFC56.Meters[0].Results.LinearLiquid.Mass_Accumulator	434

12.3.8 Meter Signals

Command	Use this Controller Tag	Description
Enable	<i>AFC56.MeterSignals.Enable</i>	This value should be set as 1 in order to enable the command. If this value is 0 the module will ignore the reset commands from the ladder
Select Meter	<i>AFC56.MeterSignals.Meter</i>	Select the meter number (1 to 16) for the reset command
Select Action	<i>AFC56.MeterSignals.Action</i>	Select the action to execute (Select Stream or Reset Resettable Accumulator)
Select Signals	<i>AFC56.MeterSignals.Signals</i>	
	<i>AFC56.MeterSignals.ResetOnce</i>	
	<i>AFC56.MeterSignals.LastBlockID</i>	

Note: The *AFC56.MeterSignals.BID* and *AFC56.MeterSignals.Action* tags are automatically updated by the ladder logic so you do not have to enter any value for this tag.

The following illustration shows the meter signals data structure.

[-] AFC56.MeterSignals	{ ... }
[-] AFC56.MeterSignals.Enable	1
[+] AFC56.MeterSignals.BID	0
[+] AFC56.MeterSignals.Meter	0
[+] AFC56.MeterSignals.Action	0
[-] AFC56.MeterSignals.Signals	{ ... }
[-] AFC56.MeterSignals.Signals.Sel_Stream1	0
[-] AFC56.MeterSignals.Signals.Sel_Stream2	0
[-] AFC56.MeterSignals.Signals.Sel_Stream3	0
[-] AFC56.MeterSignals.Signals.Sel_Stream4	0
[-] AFC56.MeterSignals.Signals.Res_Acc1	0
[-] AFC56.MeterSignals.Signals.Res_Acc2	0
[-] AFC56.MeterSignals.Signals.Res_Acc3	0
[-] AFC56.MeterSignals.Signals.Res_Acc4	0
[-] AFC56.MeterSignals.Signals.Wr_Daily_Archive	0
[-] AFC56.MeterSignals.Signals.Wr_Hrly_Archive	0
[-] AFC56.MeterSignals.ResetOnce	0
[+] AFC56.MeterSignals.LastBlockID	0

Select Stream

Note: Currently, this function is only available on the MVI56- AFC. For all other platforms continue to use AFC Manager 2.05 or earlier.

The ladder logic can select which stream (1 to 4) that will be currently performing flow calculation. Each meter has up to 4 associated streams

Controller Tag	Description
<i>AFC56.MeterSignals.Signals.Sel_Stream1</i>	Select stream 1
<i>AFC56.MeterSignals.Signals.Sel_Stream2</i>	Select stream 2
<i>AFC56.MeterSignals.Signals.Sel_Stream3</i>	Select stream 3
<i>AFC56.MeterSignals.Signals.Sel_Stream4</i>	Select stream 4

Reset Resettable Accumulators

The sample ladder logic allows the reset off all meter resettable accumulators. The module also allows the option to automatically reset the resettable accumulators when the archives are created. The module can be configured to automatically reset the resettable accumulators upon period end. Refer to Accumulators (page 83) for more information about this topic.

Applications that involve batch operations may require the reset of these accumulator from ladder In this case you should toggle the following bit commands:

Command	Use this Controller Tag	Description
Reset Resettable Accumulator 1	<i>AFC56.MeterSignals.Signals.Res_Acc1</i>	If this bit is set to 1 the module will reset Acc1 for the selected meter. The ladder logic will reset this command.
Reset Resettable Accumulator 2	<i>AFC56.MeterSignals.Signals.Res_Acc2</i>	If this bit is set to 1 the module will reset Acc2 for the selected meter. The ladder logic will reset this command.
Reset Resettable Accumulator 3	<i>AFC56.MeterSignals.Signals.Res_Acc3</i>	If this bit is set to 1 the module will reset Acc3 for the selected meter. The ladder logic will reset this command.
Reset Resettable Accumulator 4	<i>AFC56.MeterSignals.Signals.Res_Acc4</i>	If this bit is set to 1 the module will reset Acc4 for the selected meter. The ladder logic will reset this command.

Write Hourly/Daily Archive

The sample ladder logic shows how to create hourly or daily archive. It is very important to notice that the sample ladder automatically creates both archives, depending on the configured **End-of-day minute** and **End-of-hour minute** parameters entered in the AFC Manager. Most users will not need to generate the archives from ladder logic.

In order to create archives from ladder logic, refer to the following data tags:

Command	Use this Controller Tag	Description
Enable	<i>AFC56.MeterSignals.Enable</i>	This value should be set as 1 in order to enable the command. If this value is 0 the module will ignore the commands from the ladder
Select Meter	<i>AFC56.MeterSignals.Meter</i>	Select the meter number (1 to 16) for the archive command
Write Daily Archive	<i>AFC56.MeterSignals.Signals.Wr_Daily_Archive</i>	If this bit is set to 1 the module will generate a daily archive
Write Hourly Archive	<i>AFC56.MeterSignals.Signals.Wr_Hrly_Archive</i>	If this bit is set to 1 the module will generate an hourly archive

12.3.9 Molar Analysis (For Gas Product Only)

If the application uses a chromatograph device to send the molar concentrations to the module, the sample ladder may dynamically supply all molar concentrations to the MVI56-AFC.

Initially, you should select (check) the "Selected" check boxes for all elements using the AFC Manager (clicking on the Components button (version 2.05.000 or later) or Analysis button (version 2.04.000 or older) at the Meter Configuration window).

In order to write the molar concentration values from the ladder logic, you should set the *AFC56.Meters[x].Analysis.Enable* bit to 1. Also select the stream to be updated through *AFC56.Meters[x].Analysis.Stream* controller tag (select 0 for the current active stream).

If you select other stream than the one selected, the new molar analysis will only be visible through AFC Manager once that stream is later selected as the new active one.

[-] AFC56.Meters[0].Analysis.Low_Precision	{...}
[-] AFC56.Meters[0].Analysis.Low_Precision.Enable	1
[+] AFC56.Meters[0].Analysis.Low_Precision.Stream	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components	{...}
[-] AFC56.Meters[0].Analysis.High_Precision	{...}
[-] AFC56.Meters[0].Analysis.High_Precision.Enable	0
[+] AFC56.Meters[0].Analysis.High_Precision.Stream	0
[+] AFC56.Meters[0].Analysis.High_Precision.Components	{...}
[-] AFC56.Meters[0].Analysis.Status	{...}
[+] AFC56.Meters[0].Analysis.Status.Slot_In_Stream	{...}
[+] AFC56.Meters[0].Analysis.Status.Slot_Precision	{...}

After that, any molar concentration configuration performed through AFC Manager will be overwritten by the ladder logic.

Refer to the Meters[x].Analysis controller tag in order to move the concentrations for each meter (x assumes values between 0 and 15).

[-] AFC56.Meters[0].Analysis	{...}
[-] AFC56.Meters[0].Analysis.Low_Precision	{...}
[-] AFC56.Meters[0].Analysis.Low_Precision.Enable	1
[+] AFC56.Meters[0].Analysis.Low_Precision.Stream	0
[-] AFC56.Meters[0].Analysis.Low_Precision.Components	{...}
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C1	2500
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.N2	7500
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.CO2	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C2	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C3	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.H2O	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.H2S	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.H2	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.CO	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.O2	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.IC4	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.NC4	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.IC5	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.NC5	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C6	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C7	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C8	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C9	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.C10	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.He	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.Ar	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.NeoC5	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.Ux_User1	0
[+] AFC56.Meters[0].Analysis.Low_Precision.Components.Uy_User2	0

The concentrations are entered as scaled integer format where 10000 = 100%.

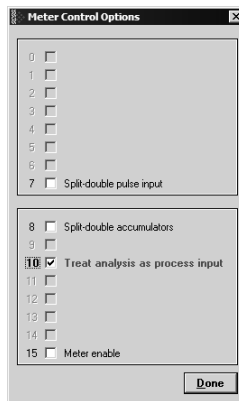
For example:

$$C1 = 9168 \geq 91.68\%$$

The sum of all concentration should be 100%. The chromatograph can measure values which total is slightly less (or more) than 100%. In this case you should configure the Normalization Error Tolerance parameter in the AFC Manager in order to make sure that the module will not generate any alarms.

When the module detects that a molar concentration value has changed it will generate an event. However, when the values are updated from ladder logic using a chromatograph device it is not convenient to generate an alarm every time a concentration value changes. In this case you may configure the module to not generate any alarms when a molar value is modified.

So you should select (check) the "Meter Configuration-> Ctrl Options->Treat Analysis as Process Input" check box:



AFC56.Meters[].Analysis.Low Precision

Stores the analysis configuration to be used for the streams configured for low precision analysis. The sub-elements are defined as follows:

Tag	Description
.Enable	Set to 1 to update the stream analysis with the data at .Components. The user application must reset the value to interrupt the analysis update through the backplane.
.Stream	Select the stream number to be updated with the molar data
.Components	Contains the molar components in low precision (16-bit integer values)

AFC56.Meters[].Analysis.High Precision

Stores the analysis configuration to be used for the streams configured for high precision analysis. The sub-elements are defined as follows:

Note: Currently, this function is only available on the MVI56- AFC. For all other platforms continue to use AFC Manager 2.05 or earlier.

Tag	Description
.Enable	Set to 1 to update the stream analysis with the data at .Components. The sample logic will automatically reset the bit after requesting the update once.
.Stream	Select the stream number to be updated with the molar data
.Components	Contains the molar components in low precision (32-bit integer values)

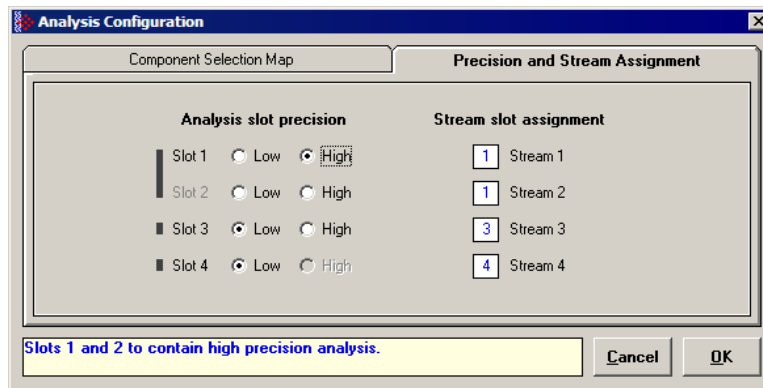
It is the user responsibility to update the correct tag (high or low precision) according to the stream configuration). The sample logic will not check if the update request will match the current stream configuration.

AFC56.Meters[].Analysis.Status

Shows the molar analysis configuration information.

- 1 Stream Analysis Assignment
(AFC56.Meters[].Analysis.Status.Slot_In_Stream)
Shows the slot assigned for each stream (0-based)
- 2 Analysis Precision(AFC56.Meters[0].Analysis.Status.Slot_Precision)
Shows the stream precision for each slot. The following codes are used:
 - 0 (00b) (second slot of preceding high-precision)
 - 1 (01b) Low-precision, one slot only
 - 2 (10b) High-precision, two consecutive slots
 - 3 (11b) (illegal value; never occurs)

For example, for the following configuration:



The analysis data will be updated as follows:

[-] AFC56.Meters[0].Analysis.Status	{ ... }	{ ... }		AFC_Analysis_Status
[-] AFC56.Meters[0].Analysis.Status.Slot_In_Stream	{ ... }	{ ... }	Decimal	SINT[4]
[+] AFC56.Meters[0].Analysis.Status.Slot_In_Stream[0]	0		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_In_Stream[1]	0		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_In_Stream[2]	2		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_In_Stream[3]	3		Decimal	SINT
[-] AFC56.Meters[0].Analysis.Status.Slot_Precision	{ ... }	{ ... }	Decimal	SINT[4]
[+] AFC56.Meters[0].Analysis.Status.Slot_Precision[0]	2		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_Precision[1]	0		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_Precision[2]	1		Decimal	SINT
[+] AFC56.Meters[0].Analysis.Status.Slot_Precision[3]	1		Decimal	SINT

The status information is automatically updated by the sample logic upon the following events:

- 1 After module power up
- 2 After double-precision analysis request block

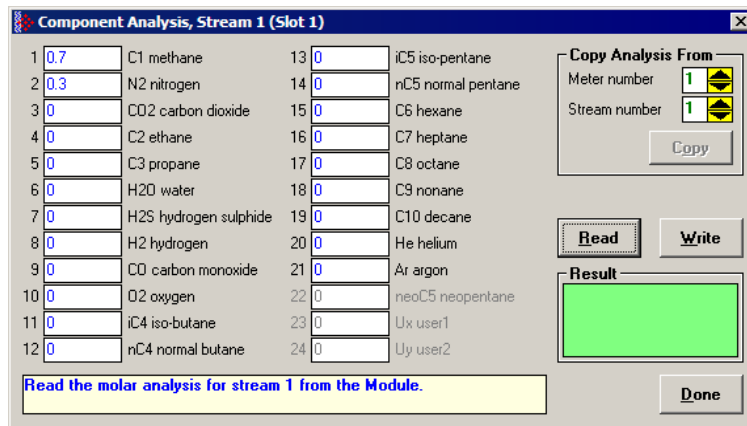
Updating the High Precision Molar Analysis

To update the molar analysis with high precision data, follow these steps.

- 1 Set the stream number through
AFC56.Meters[].Analysis.High_Precision.Stream
- 2 Enter the molar analysis through
AFC56.Meters[0].Analysis.High_Precision.Components. For the example below the components are set as C1 = 0.7 (70%) and N2 = 0.3 (30%)
- 3 Set the AFC56.Meters[0].Analysis.High_Precision.Enable value as 1
- 4 Observe AFC56.Meters[0].Analysis.High_Precision.Enable bit automatically reset to 0

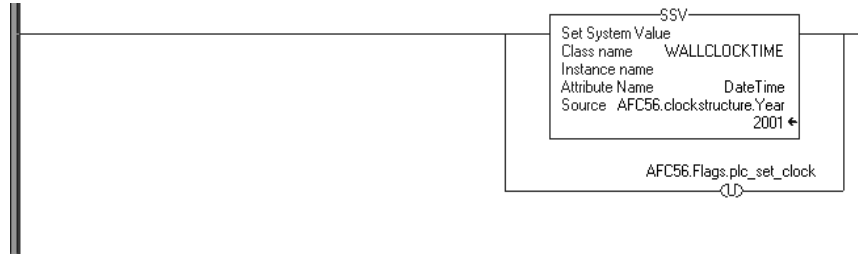
- AFC56.Meters[0].Analysis	{...}	{...}		AFC_Meter_Analysis
+ AFC56.Meters[0].Analysis.Low_Precision	{...}	{...}		AFC_Analysis_Low
- AFC56.Meters[0].Analysis.High_Precision	{...}	{...}		AFC_Analysis_High
- AFC56.Meters[0].Analysis.High_Precision.Enable	0		Decimal	BOOL
+ AFC56.Meters[0].Analysis.High_Precision.Stream	1		Decimal	INT
- AFC56.Meters[0].Analysis.High_Precision.Components	{...}	{...}		AFC_Components_High
- AFC56.Meters[0].Analysis.High_Precision.Components.C1	0.7		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.N2	0.3		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.CO2	0.0		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.C2	0.0		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.C3	0.0		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.H2O	0.0		Float	REAL
- AFC56.Meters[0].Analysis.High_Precision.Components.H2S	0.0		Float	REAL

- 5 Refer to the AFC Manager to observe the updated molar analysis:



12.3.10 Set the Processor Time

This logic was added for convenience purposes only. It shows how to set the processor time and date information using a controller tag as the source:



The source of the information is:

Value	Controller Tag
Year	AFC56.clockstructure.Year
Month	AFC56.clockstructure.Month
Day	AFC56.clockstructure.Day
Hour	AFC56.clockstructure.Hour
Minute	AFC56.clockstructure.Minute
Seconds	AFC56.clockstructure.Second
Milliseconds	AFC56.clockstructure.msec

[-] AFC56.clockstructure	{...}
[+] AFC56.clockstructure.Year	2001
[+] AFC56.clockstructure.Month	3
[+] AFC56.clockstructure.Day	30
[+] AFC56.clockstructure.Hour	8
[+] AFC56.clockstructure.Minute	8
[+] AFC56.clockstructure.second	8
[+] AFC56.clockstructure.msec	0

The ladder logic should toggle the following bit in order to transfer the date and time information to the processor.

Command	Controller Tag
Write Clock	AFC56.Flags.plc_set_clock

12.3.11 Checking Meter Alarms

The logic continuously informs if a meter has an alarm or not. Refer to the following controller tags for the meter alarm status:

Information	Controller Tag	Values
Meter 1 Alarm Status	AFC56.Site_Alarms.Meter1	0 = Meter 1 does not have alarm 1 = Meter 1 has alarm
Meter 2 Alarm Status	AFC56.Site_Alarms.Meter2	0 = Meter 2 does not have alarm 1 = Meter 2 has alarm
Meter 3 Alarm Status	AFC56.Site_Alarms.Meter3	0 = Meter 3 does not have alarm 1 = Meter 3 has alarm
Meter 4 Alarm Status	AFC56.Site_Alarms.Meter4	0 = Meter 4 does not have alarm 1 = Meter 4 has alarm
...
Meter 16 Alarm Status	AFC56.Site_Alarms.Meter16	0 = Meter 16 does not have alarm 1 = Meter 16 has alarm

In the following example, Meters 1 and 2 have alarms. The other meters do not:

AFC56.Site_Alarms	{...}
AFC56.Site_Alarms.Meter1	1
AFC56.Site_Alarms.Meter2	1
AFC56.Site_Alarms.Meter3	0
AFC56.Site_Alarms.Meter4	0
AFC56.Site_Alarms.Meter5	0
AFC56.Site_Alarms.Meter6	0
AFC56.Site_Alarms.Meter7	0
AFC56.Site_Alarms.Meter8	0
AFC56.Site_Alarms.Meter9	0
AFC56.Site_Alarms.Meter10	0
AFC56.Site_Alarms.Meter11	0
AFC56.Site_Alarms.Meter12	0
AFC56.Site_Alarms.Meter13	0
AFC56.Site_Alarms.Meter14	0
AFC56.Site_Alarms.Meter15	0
AFC56.Site_Alarms.Meter16	0

For each meter, the logic also indicates which alarm was generated. Refer to the following controller tags for meter alarm information:

Information	Controller Tag	Values
Meter 1 Alarm	AFC56.Meters[0].Results.Alarms	Please see following table
Meter 2 Alarm	AFC56.Meters[1].Results.Alarms	Please see following table
Meter 3 Alarm	AFC56.Meters[2].Results.Alarms	Please see following table
Meter 4 Alarm	AFC56.Meters[3].Results.Alarms	Please see following table
...
Meter 16 Alarm	AFC56.Meters[15].Results.Alarms	Please see following table

Each Alarm word is interpreted as follows:

Bit Number	Description
0	Input out of range: Temperature
1	Input out of range: Pressure
2	Input out of range: Differential Pressure
3	Input out of range: Flowing Density
4	Input out of range: Water Content
5	Differential Pressure Low
6	Orifice Pressure Exception
7	Accumulation Overflow
8	Orifice Characterization Error
9	Analysis Total Zero
10	Analysis Total Not Normalized
11	AGA 8 Calculation Error
12	API Calculation Error: Density Correction
13	API Calculation Error: Ctl
14	API Calculation Error: Vapor Pressure
15	API Calculation Error: Cpl

The following illustration shows an example where the Meter 1 has an "Input **Out of Range: Temperature**" alarm:

[-] AFC56.Meters[0].Results	{...}
[+] AFC56.Meters[0].Results.BID	2#0010_0010_0000_0001
[+] AFC56.Meters[0].Results.Alarms	2#0000_0000_0000_0001
[+] AFC56.Meters[0].Results.OrificeGas	{...}
[+] AFC56.Meters[0].Results.OrificeLiquid	{...}
[+] AFC56.Meters[0].Results.PulseGas	{...}
[+] AFC56.Meters[0].Results.PulseLiquid	{...}

12.3.12 Site Status

The ladder logic continuously reads the site status from the MVI56-AFC. The following controller tags are used:

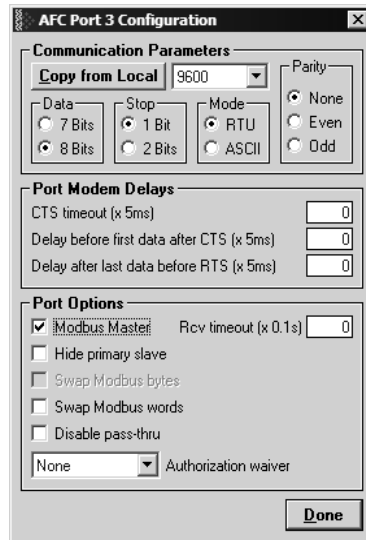
Information	Controller Tag
AFC56(16) Released	AFC56.Site_Status.AFC_ACTIVE
Checksum Alarms	AFC56.Site_Status.Checksum_Alarms
PLC Halted	AFC56.Site_Status.PLC_Halted
Cfg Changed	AFC56.Site_Status.Cfg_changed
Power Up	AFC56.Site_Status.Powerup
Cold Start	AFC56.Site_Status.ColdStart

AFC56.Site_Status	{...}
-AFC56.Site_Status.AFC_ACTIVE	1
-AFC56.Site_Status.Checksum_Alarms	0
-AFC56.Site_Status.reserved2	0
-AFC56.Site_Status.reserved3	0
-AFC56.Site_Status.PLC_HALTED	0
-AFC56.Site_Status.Cfg_changed	0
-AFC56.Site_Status.Powerup	0
AFC56.Site_Status.ColdStart	0
-AFC56.Site_Status.reserved8	0
-AFC56.Site_Status.reserved9	0
-AFC56.Site_Status.reserved10	0
-AFC56.Site_Status.reserved11	0
-AFC56.Site_Status.reserved12	0
-AFC56.Site_Status.reserved13	0
-AFC56.Site_Status.reserved14	0
-AFC56.Site_Status.reserved15	0

12.3.13 Modbus Master

This block performs an arbitrary data transfer between the PLC and external Modbus slaves connected to AFC port 3, provided that port 3 is configured as a Modbus master. Any data transfer to or from a slave's holding registers, input registers, output coils, or input status may be implemented using this function; equivalent Modbus function codes are 1, 2, 3, 4, 15, and 16. In addition, capability is provided for access to a slave's "long remote" (32-bit) registers where the slave implements them; in particular, Daniel-style long integer (5000 series) and floating point (7000 series) registers are accessible. Any data words not relevant to the command are ignored upon output (to the AFC) and zero upon input (from the AFC).

This feature requires port 3 to be configured as a Modbus master through AFC Manager as follows:



In order to enable this functionality set AFC56.ModbusMaster.Enable controller tag bit to 1. Select the command index to be executed (0 to 9) through AFC56.ModbusMaster.CommandIndex.

- AFC56.ModbusMaster.Enable	1
+ AFC56.ModbusMaster.CommandIndex	0
- AFC56.ModbusMaster.Command	{...}
+ AFC56.ModbusMaster.Command[0]	{...}
+ AFC56.ModbusMaster.Command[1]	{...}
+ AFC56.ModbusMaster.Command[2]	{...}
+ AFC56.ModbusMaster.Command[3]	{...}
+ AFC56.ModbusMaster.Command[4]	{...}
+ AFC56.ModbusMaster.Command[5]	{...}
+ AFC56.ModbusMaster.Command[6]	{...}
+ AFC56.ModbusMaster.Command[7]	{...}
+ AFC56.ModbusMaster.Command[8]	{...}
+ AFC56.ModbusMaster.Command[9]	{...}

The user should configure the following parameters for the command to be sent:

Parameter	Use this Controller Tag	Description
Enable	<i>AFC56.ModbusMaster.Command[0].Output.Config.Enable</i>	This value should be set as 1 in order to enable the command. If this value is 0 the module will ignore the Modbus master command
Function Type	<i>AFC56.ModbusMaster.Command[0].Output.Config.FunctionType_Write</i>	0 = Read from Modbus slave 1 = Write to Modbus slave
Register Bank	<i>AFC56.ModbusMaster.Command[0].Output.Config.RegisterBank_Input</i>	0 = Holding Register 1 = Input Register
Data Type = Bit	<i>AFC56.ModbusMaster.Command[0].Output.Config.DataType.Bit</i>	0 = Don't use bit 1 = Use bit
Data Type = Word	<i>AFC56.ModbusMaster.Command[0].Output.Config.DataType.Word</i>	0 = Don't use word 1 = Use word
Data Type = Long	<i>AFC56.ModbusMaster.Command[0].Output.Config.DataType.Long</i>	0 = Don't use long 1 = Use long
Data Type = Long Remote	<i>AFC56.ModbusMaster.Command[0].Output.Config.DataType.LongRemote</i>	0 = Don't use long remote 1 = Use long remote
Swap Bytes	<i>AFC56.ModbusMaster.Command[0].Output.Config.SwapOptions.SwapBytes</i>	0 = Don't swap bytes 1 = Swap bytes (Not valid for bits)
Swap Words	<i>AFC56.ModbusMaster.Command[0].Output.Config.SwapOptions.SwapWords</i>	0 = Don't swap words 1 = Swap words (Not valid for bits or words)
Slave Address	<i>AFC56.ModbusMaster.Command[0].Output.Config.SlaveAddress</i>	Modbus slave address
Modbus Address	<i>AFC56.ModbusMaster.Command[0].Output.Config.ModbusAddress</i>	Register address in the Modbus slave
Element Count	<i>AFC56.ModbusMaster.Command[0].Output.Config.ElementCounts</i>	Number of elements to be transferred
Transaction Number	<i>AFC56.ModbusMaster.Command[0].Output.TransactionNumber</i>	This number will be echoed in the input block. It allows to multiplex more than one command in the same logic.

The following variables are automatically built by the ladder logic and the user does not have to modify these values:

AFC56.ModbusMaster[0].Output.Config.Type_Swap

AFC56.ModbusMaster[0].Output.BOD

- AFC56.ModbusMaster.Command[0]	{...}
- AFC56.ModbusMaster.Command[0].Output	{...}
- AFC56.ModbusMaster.Command[0].Output.Config	{...}
- AFC56.ModbusMaster.Command[0].Output.Config.Enable	1
- AFC56.ModbusMaster.Command[0].Output.Config.FunctionType_Write	1
- AFC56.ModbusMaster.Command[0].Output.Config.RegisterBank_Input	0
- AFC56.ModbusMaster.Command[0].Output.Config.DataType	{...}
- AFC56.ModbusMaster.Command[0].Output.Config.DataType.Bit	0
- AFC56.ModbusMaster.Command[0].Output.Config.DataType.Word	1
- AFC56.ModbusMaster.Command[0].Output.Config.DataType.Long	0
- AFC56.ModbusMaster.Command[0].Output.Config.DataType.LongRemote	0
- AFC56.ModbusMaster.Command[0].Output.Config.SwapOption	{...}
- AFC56.ModbusMaster.Command[0].Output.Config.SwapOption.Swap_Words	0
- AFC56.ModbusMaster.Command[0].Output.Config.SwapOption.Swap_Bytes	0
+ AFC56.ModbusMaster.Command[0].Output.Config.Type_Swap	1
+ AFC56.ModbusMaster.Command[0].Output.Config.SlaveAddress	244
+ AFC56.ModbusMaster.Command[0].Output.Config.ModbusAddress	2000
+ AFC56.ModbusMaster.Command[0].Output.Config.ElementCount	20
+ AFC56.ModbusMaster.Command[0].Output.TransactionNumber	0

- If the Modbus Master command is set as a READ function type the data will be read to the following controller tag:

AFC56.ModbusMaster.Command[0].Input.ReadData[]

- If the Modbus Gateway command is set as a **WRITE** function type the data will be written from the following controller tag:

AFC56.ModbusMaster.Command[0].Output.WriteData[]

- If any Modbus error occurs it will be copied to the following data tag:

AFC56.ModbusMaster[0].Input.ErrorCode

The valid Modbus error codes are listed below:

Error	Description
=0	No error
>0	Modbus exception code or communication error Modbus exception codes are issued by the responding slave and listed in commonly available Modbus protocol manuals; they lie between 1 and 127, and include: <ul style="list-style-type: none"> 1 Illegal function 2 Illegal address 3 Illegal data value

Communication errors are issued by the AFC:

Error	Description
500	CTS timeout
501	Receive timeout
502	Bad framing
503	Buffer overrun
504	Bad checksum/CRC
505	Wrong slave
506	Wrong function code
507	Wrong length
<0	Configuration, parameter, or logic error: <ul style="list-style-type: none"> -1 Master Port not configured -2 Master Port never used -3 Bad Slave Address -4 Bad Direction/Target -5 Bad Datum Size/Swap Options -6 Bad number of data items

12.3.14 Modbus pass-through

In order to use pass-through, enable the following controller tag:

AFC56.ModbusPassThru.Output.Enable (BOOL)

Once the module receives the Modbus write command the data will be copied to the following controller tag:

AFC56.ModbusPassThru.Input.ReadData_Word[] (word commands)

AFC56.ModbusPassThru.Input.ReadData_Bit[] (bit commands)

[- AFC56.ModbusPassThru	{...}
[- AFC56.ModbusPassThru.Output	{...}
+ AFC56.ModbusPassThru.Output.BOD	2#0001_1000_0110_0110
- AFC56.ModbusPassThru.Output.Enable	1
[- AFC56.ModbusPassThru.Input	{...}
+ AFC56.ModbusPassThru.Input.BID	2#0001_1000_0000_0000
- AFC56.ModbusPassThru.Input.MessagePresent	0
- AFC56.ModbusPassThru.Input.Function_BitWrite	0
- AFC56.ModbusPassThru.Input.OverflowError	0
+ AFC56.ModbusPassThru.Input.ModbusAddress	2200
+ AFC56.ModbusPassThru.Input.RegisterCount	5
+ AFC56.ModbusPassThru.Input.ReadData_Word	{...}
+ AFC56.ModbusPassThru.Input.ReadData_Bit	{...}

12.3.15 Modbus Gateway

The ladder logic can be used to read or write data from one of the internal Modbus Slaves (primary or virtual). Also, any data that is not part of the `AFC56.Meters[]`. Results must be read through the Modbus Gateway blocks.

Each block can transfer up to 200 words of data and uses a specific `AFC56.Modbus.Gateway[]` controller tag. Each one of these tags must be configurable in order to read or write data between the module and the processor.

Perform the following steps to use the Modbus Gateway blocks:

- 1 Identify how many words (total) will be transferred. The sample ladder supports up to 2000 words.
- 2 Based on the number of registers to be transferred, calculate how many Modbus Gateway blocks will be necessary. Each block contains up to 200 registers. For example: if 700 registers will be used to transfer data, 4 Modbus Gateway blocks will be required.
- 3 Based on the number of Modbus Gateway blocks, configure the `AFC56.ModbusGateway.CommandCount` controller tag.

- AFC56.ModbusGateway	{ ... }
+ AFC56.ModbusGateway.CommandCount	4
+ AFC56.ModbusGateway.Command	{ ... }

For example, if you configure the number of blocks as 4, the ladder logic automatically sends the following Modbus Gateway blocks to the module:

`AFC56.ModbusGateway.Command[0]`
`AFC56.ModbusGateway.Command [1]`
`AFC56.ModbusGateway.Command [2]`
`AFC56.ModbusGateway.Command [3]`

The maximum number of supported commands by the sample ladder is 10.

If the `AFC56.ModbusGateway.CommandCount` controller tag is configured as 0 the module will not send any Modbus Gateway blocks.

- 1 Refer to the Modbus Dictionary dialog box in AFC Manager and identify the addresses of all registers in the Primary Slave.
- 2 Using the AFC Manager, re-map the registers from the Primary Slave to the Virtual Slave (refer to AFC Manager User Manual for more information about this subject).
- 3 You must also set a Virtual Slave Address greater than 0 in order to activate the Virtual Slave.

- In the sample ladder logic, configure each Modbus Gateway Block using the following controller tags:

Parameter	Controller Tag	Values
Enable Transaction	AFC56.ModbusGateway.Command [0].Config.Enable	0 = The module will ignore this request 1 = The module will process this request.
Start Register	AFC56.ModbusGateway.Command [0].Config.StartRegister	Start register in the Modbus Slave to be written or read from
Register Count	AFC56.ModbusGateway.Command [0].Config.RegisterCount	Number of words to be written or read between the module and the processor. Maximum of 200 words is supported per command.
Function Type	AFC56.ModbusGateway.Command [0].Config.FunctionType_Write	0 = Read from MVI56-AFC 1 = Write to MVI56-AFC
Register Type	AFC56.ModbusGateway.Command [0].Config.RegisterType_Input	0 = Holding Register 1 = Input Register
Slave Type	AFC56.ModbusGateway.Command [0].Config.SlaveType_Virtual	0 = Primary Slave 1 = Virtual Slave

Note: It is strongly suggested that you first configure all parameters having the Enable bit set to 0. After the configuration is finished than the Enable bit can be set to 1.

Example 1 - Modbus Gateway Function

In order to write 200 words from the processor to the Primary Modbus Slave starting at holding register address 2000, the *AFC56.Modbus.Gateway[0]* block should be configured as follows:

[-] AFC56.ModbusGateway.Command[0].Config	{...}
- AFC56.ModbusGateway.Command[0].Config.Enable	1
[+] AFC56.ModbusGateway.Command[0].Config.StartRegister	2000
[+] AFC56.ModbusGateway.Command[0].Config.RegisterCount	200
- AFC56.ModbusGateway.Command[0].Config.FunctionType_Write	1
- AFC56.ModbusGateway.Command[0].Config.RegisterType_Input	0
- AFC56.ModbusGateway.Command[0].Config.SlaveType_Virtual	0

- Refer to the *AFC56.ModbusGateway.Command[x].WriteData* or *AFC56.ModbusGateway.Command[x].ReadData* controller tags depending on the configured function type:
- If the Modbus Gateway block uses a **READ** function type:
The data will be read to the *AFC56.ModbusGateway.Command[x].ReadData[]* array.
If the Modbus Gateway block uses a **WRITE** function type:
The data will be written from the *AFC56.Modbus.Gateway[x].WriteData[]* array.

Note: The BID and BOD controller tags are automatically generated by the logic so you do not have to write any value to these controller tags.

Example 2 - Read Net Accumulator Totalizer From Yesterday's Archive (Meters 1 to 4) (assume orifice meters, gas product)

Perform the following steps:

This application only transfers 8 words (each totalizer occupies 2 words). Because the number of words is less than 200, it means that only one Modbus Gateway block will be used (AFC56.Modbus.Gateway[0])

- 1 Configure the *AFC56.Modbus.BlockCount* tag with a value of 1.
- 2 In order to identify the register addresses in the Primary Modbus Slave, refer to the following spreadsheet that shows the addresses for the hourly and daily archives:

Input Registers

Meter	Start Daily Archive	End Daily Archive	Start Hourly Archive	End Hourly Archive
1	0	1059	1060	2499
2	2500	3559	3560	4999
3	5000	6059	6060	7499
4	7500	8559	8560	9999
5	10000	11059	11060	12499
6	12500	13559	13560	14999
7	15000	16059	16060	17499
8	17500	18559	18560	19999
9	20000	21059	21060	22499
10	22500	23559	23560	24999
11	25000	26059	26060	27499
12	27500	28559	28560	29999
13	30000	31059	31060	32499
14	32500	33559	33560	34999
15	35000	36059	36060	37499
16	37500	38559	38560	39999

Each archive occupies a 30-word block. For example, for meter 1:

Archive	Start Register	End Register
Yesterday	0	29
2 days ago	30	59
3 days ago	60	89
4 days ago	90	119
5 days ago	120	149
...

The following shows the structure of each archive. The first 10 words are common for all archives. The rest of the archive structure will depend on the meter type or product group:

Pre-defined Overhead

Start Offset	End Offset	Data Type	Description
00	01	Dt	Closing timestamp of archive
02		Wd	Flowing period
03		Bm	Cumulative meter alarms
04		Bm	Cumulative status
05		Wd	Event counter
06	07	Dw	Flowing period, seconds
08	09	Dt	Opening timestamp of archive

Orifice Meter with Gas Product

Start Offset	End Offset	Data Type	Description
10	11	Acc	Accumulator totalizer, net/liqeqv
12	13	Fp	Accumulator residue, net/liqeqv
14	15	Fp	Flow rate, net/liqeqv
16	17	Fp	Temperature, CU
18	19	Fp	Pressure, CUG
20	21	Fp	Differential pressure, CU
22		Wd	Relative density 15°C/15°C, e-4
23		Wd	Compressibility, reference, e-4
24		Wd	Compressibility, flowing, e-4
25		Wd	Fpv, e-4
26		Wd	Velocity of approach factor, Ev, e-4
27		Wd	Expansion factor, Y, e-4
28		Wd	Coefficient of discharge, Cd, e-4
29		Wd	Reserved

Pulse Meter with Gas Product

Start Offset	End Offset	Data Type	Description
10	11	Acc	Accumulator totalizer, net/liqeqv
12	13	Fp	Accumulator residue, net/liqeqv
14	15	Fp	Flow rate, net/liqeqv
16	17	Fp	Temperature, CU
18	19	Fp	Pressure, CUG
20	21	Fp	K-Factor
22	23	Fp	Meter Factor
24		Wd	Relative density 15°C/15°C, e-4
25		Wd	Compressibility, reference, e-4
26		Wd	Compressibility, flowing, e-4
27		Wd	Fpv, e-4
28	29	Wd	Reserved

Orifice Meter with Liquid Product

Start Offset	End Offset	Data Type	Description
10	11	Acc	Accumulator totalizer, net/liqeqv
12	13	Fp	Accumulator residue, net/liqeqv
14	15	Fp	Flow rate, net/liqeqv
16	17	Fp	Temperature, CU
18	19	Fp	Pressure, CUg
20	21	Fp	Differential pressure, CU
22	23	Fp	Flowing density, CU
24		Wd	Corrected density, kg/m ³ e-1
25		Wd	Ctl e-4
26		Wd	Cpl e-4
27		Wd	Velocity of approach factor, Ev, e-4
28		Wd	Expansion factor, Y, e-4
29		Wd	Coefficient of discharge, Cd, e-4

Pulse Meter with Liquid Product

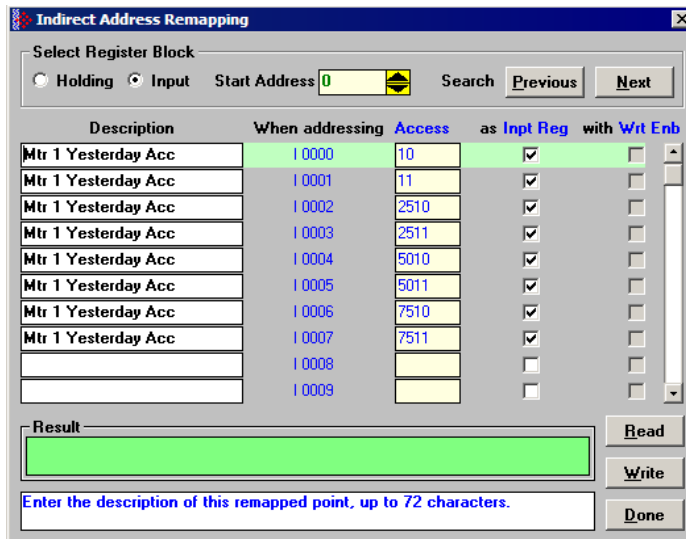
Start Offset	End Offset	Data Type	Description
10	11	Acc	Accumulator totalizer, net/liqeqv
12	13	Fp	Accumulator residue, net/liqeqv
14	15	Fp	Flow rate, net/liqeqv
16	17	Fp	Temperature, CU
18	19	Fp	Pressure, CUg
20	21	Fp	K-Factor
22	23	Fp	Meter Factor
24	25	Fp	Flowing density, CU
26		Wd	Water content, % e-2
27		Wd	Corrected density, kg/m ³ e-1
28		Wd	Ctl e-4
29		Wd	Cpl e-4

The following input registers are important:

Primary Modbus Slave Input Register Address	Description
10	Net Acc Totalizer from Yesterday archive - Meter 1
11	Net Acc Totalizer from Yesterday archive - Meter 1
2510	Net Acc Totalizer from Yesterday archive - Meter 2
2511	Net Acc Totalizer from Yesterday archive - Meter 2
5010	Net Acc Totalizer from Yesterday archive - Meter 3
5011	Net Acc Totalizer from Yesterday archive - Meter 3
7510	Net Acc Totalizer from Yesterday archive - Meter 4
7511	Net Acc Totalizer from Yesterday archive - Meter 4

Using the AFC Manager you should re-map the registers above to the Virtual Slave. In this example we will re-map the addresses as follows:

Virtual Modbus Slave Input Register Address	Primary Modbus Slave Input Register Address
0	10
1	11
2	2510
3	2511
4	5010
5	5011
6	7510
7	7511



Note: The past archives and events are always input registers. All other data should be re-mapped as holding registers.

In the sample ladder we can now configure the Modbus Gateway block to read 6 input registers words from the Virtual slave starting at address 0:

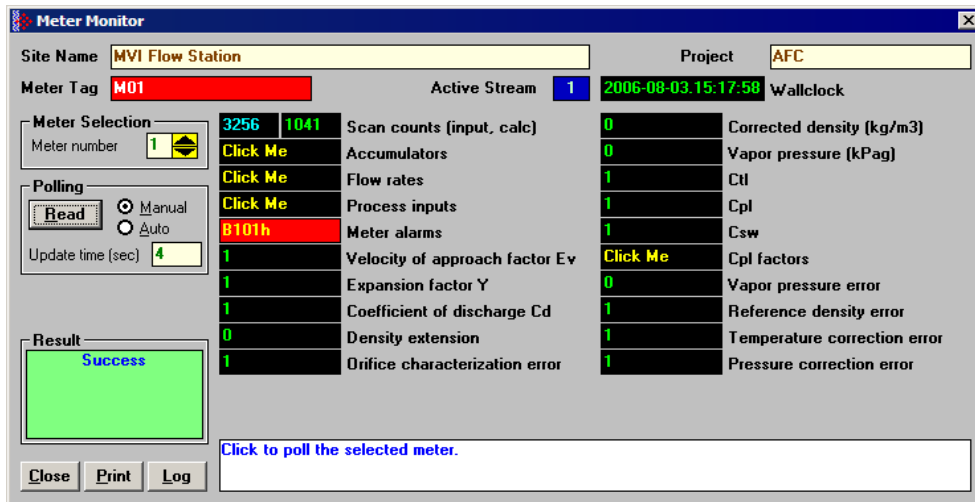
[-] AFC56.ModbusGateway.Command[0]	{ ... }
[-] AFC56.ModbusGateway.Command[0].Config	{ ... }
[-] AFC56.ModbusGateway.Command[0].Config.Enable	1
[+] AFC56.ModbusGateway.Command[0].Config.StartRegister	0
[+] AFC56.ModbusGateway.Command[0].Config.RegisterCount	0
[-] AFC56.ModbusGateway.Command[0].Config.FunctionType_Write	0
[-] AFC56.ModbusGateway.Command[0].Config.RegisterType_Input	1
[-] AFC56.ModbusGateway.Command[0].Config.SlaveType_Virtual	1

The data will be available in the
AFC56.ModbusGateway.Command[0].ReadData[0] controller tag.

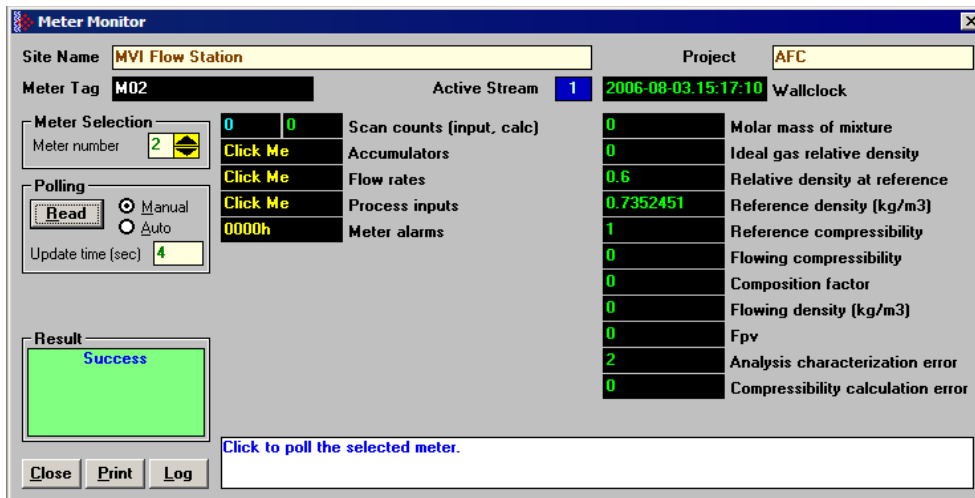
The following illustrations show that, for this example, the totalizer values are:

Totalizer for yesterday's archive	
Meter 1	5
Meter 2	7
Meter 3	11
Meter 4	14

Meter 1: Yesterday's Archive



Meter 2: Yesterday's Archive



Meter 3: Yesterday's Archive

Meter Monitor

Site Name: **MVI Flow Station** Project: **AFC**

Meter Tag: **M03** Active Stream: **1** 2006-08-03.15:18:28 Wallclock

Meter Selection: Meter number **3**

Polling: Manual Auto Update time (sec): **4**

Result: **Success**

Click to poll the selected meter.

7025	1536	Scan counts (input, calc)	0	Molar mass of mixture
Click Me		Accumulators	0	Ideal gas relative density
Click Me		Flow rates	0.6	Relative density at reference
Click Me		Process inputs	0.7352451	Reference density (kg/m3)
0000h		Meter alarms	1	Reference compressibility
1.149037		Velocity of approach factor Ev	1	Flowing compressibility
1		Expansion factor Y	0.1189177	Composition factor
6.524647		Coefficient of discharge Cd	14.16015	Flowing density (kg/m3)
0		Pressure extension	1	Fpv
1		Orifice characterization error	0	Analysis characterization error
			0	Compressibility calculation error

Meter 4: Yesterday's Archive

Meter Monitor

Site Name: **MVI Flow Station** Project: **AFC**

Meter Tag: **M04** Active Stream: **1** 2006-08-03.15:18:58 Wallclock

Meter Selection: Meter number **4**

Polling: Manual Auto Update time (sec): **4**

Result: **Success**

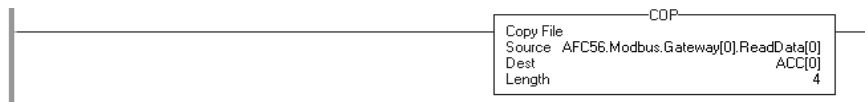
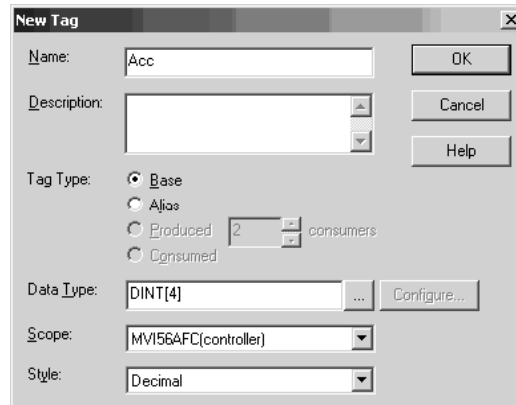
Click to poll the selected meter.

7038	1538	Scan counts (input, calc)	0	Molar mass of mixture
Click Me		Accumulators	0	Ideal gas relative density
Click Me		Flow rates	0.6	Relative density at reference
Click Me		Process inputs	0.7352451	Reference density (kg/m3)
0000h		Meter alarms	1	Reference compressibility
1.030235		Velocity of approach factor Ev	1	Flowing compressibility
0.9787242		Expansion factor Y	0.1281473	Composition factor
0.6041108		Coefficient of discharge Cd	1.663933	Flowing density (kg/m3)
25.88556		Pressure extension	1	Fpv
1		Orifice characterization error	0	Analysis characterization error
			0	Compressibility calculation error

The controller tags show:

- AFC56.ModbusGateway.Command[0].ReadData	{...}
+ AFC56.ModbusGateway.Command[0].ReadData[0]	5
+ AFC56.ModbusGateway.Command[0].ReadData[1]	0
+ AFC56.ModbusGateway.Command[0].ReadData[2]	7
+ AFC56.ModbusGateway.Command[0].ReadData[3]	0
+ AFC56.ModbusGateway.Command[0].ReadData[4]	11
+ AFC56.ModbusGateway.Command[0].ReadData[5]	0
+ AFC56.ModbusGateway.Command[0].ReadData[6]	14
+ AFC56.ModbusGateway.Command[0].ReadData[7]	0

The ReadData[] tag is an array of integers. Because the totalizer is displayed as 32-bit long integer, you may create an array of DINT elements and then add a rung to copy the values to this new array.



The totalizers will be copied to the new ACC[] array:

ACC	{...}
ACC[0]	5
ACC[1]	7
ACC[2]	11
ACC[3]	14

13 Diagnostics and Troubleshooting

In This Chapter

❖ User LEDs	256
❖ BBRAM LEDs	257
❖ Meter Alarms	258
❖ Checksum Alarms	262
❖ Events	263
❖ Audit Scan	264

MVI56-AFC modules have the following communication connections on the module:

- Two RS-232/422/485 Application ports
- One RS-232 Configuration/Debug port

This section provides information that will assist you during the module operation on troubleshooting issues. This section describes the following topics:

- LEDs
- Meter Alarms
- Checksum Alarms
- Events
- Audit Scan

13.1 User LEDs

There are two "user" LEDs used to indicate overall module status; App Status and BP Act (with P1, P2, or P3).

13.1.1 App Status LED

State	Description
Rapid Blinking	The processor is offline (probably in program mode).
Steady On	Some meter is indicating an alarm or no meters are enabled.
Off	The module is functioning properly.

13.1.2 BP Act and P1, P2, or P3

These LEDs indicate current Modbus traffic on any port.

State	Description
On	A Modbus command for the module is recognized. On Port 3, this LED may also indicate that a Modbus Master command was sent.
Off	No Activity

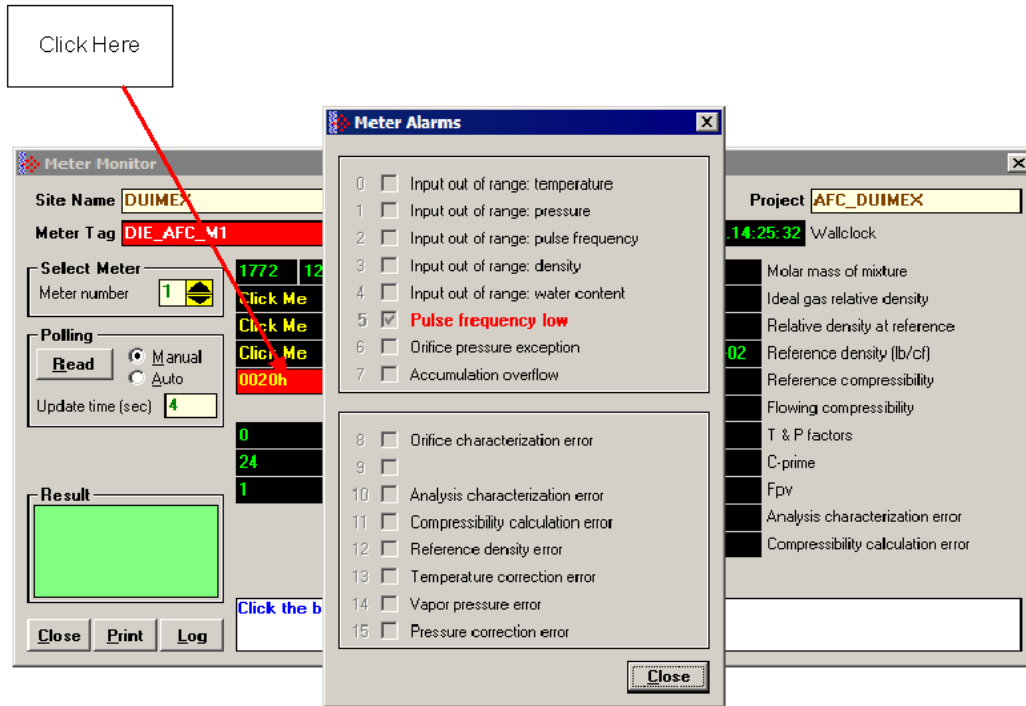
13.2 BBRAM LEDs

The BBRAM (Battery Backed RAM) LEDs inform you about the condition of the BBRAM hardware used for data storage. The following table lists the possible situations that might occur during normal operation.

OK (Green)	ERR (Red)	Description
ON	ON	The module is in a Cold Start condition that typically occurs when you power up the module for the first time. After at least one meter is enabled and the processor is in RUN mode the module starts operating.
ON	OFF	Normal Operation
Blinking	OFF	This condition is warning that a checksum flag was raised after a power cycle. If this alarm issue occurs, refer to the AFC Manager (On-line Monitor / Checksum Alarms) in order to determine the data section in which the alarm issue has occurred. After verifying that the checksum error has not affected the referred memory area you may clear the checksum alarm using the same AFC Manager interface. After the alarm is cleared the OK LED will be ON

13.3 Meter Alarms

If the module is generating unexpected data, you should verify if the meter has any alarms. Some alarms may be caused by an issue that could potentially affect the calculation results. Each archive also keeps track of the alarms that have occurred during the period (refer to the Archive section). The Meter Monitor dialog box allows you to monitor the meter alarms.



The above image shows the Meter Alarms bitmap, which gives you a quick overview of active alarms. Associated with many of these bits are Alarm Code registers which supply specific reasons for the alarms, most of which appear in the lower right corner of the main Meter Monitor window. For complete information, including which Code registers are associated with which alarm bits, use the Modbus Dictionary feature of AFC Manager.

The possible alarms are listed in the following table. Of the Alarm Codes listed, the values that can actually appear depend on both the selected Product Group and the firmware version.

Alarm Message	Description	Solution
Accumulation Overflow	The module ignores an accumulator increment of less than zero or greater than 1.000.000.000 occurring in a single meter scan.	Check your meter configuration to verify if your project is generating reasonable values.
Analysis Total Not Normalized ($v \leq 2.04$)	Absolute difference between analysis total and 1.0000 (100%) is greater than the error tolerance	Make sure that the sum of all molar concentrations is within the error tolerance of 1.0000 (100%).

Alarm Message	Description	Solution
Analysis Total Zero ($v \leq 2.04$)	The molar concentration sum is zero.	Make sure that the sum of all molar concentrations is within the error tolerance of 1.0000 (100%).
Analysis Characterization error ($v \geq 2.05$)	Absolute difference between analysis total and 1.0000 (100%) is greater than the error tolerance, OR the molar concentration sum is zero.	Make sure that the sum of all molar concentrations is within the error tolerance of 1.0000 (100%). Alarm Code values: 0 = No alarm 1 = Analysis total not normalized 2 = Analysis total zero
Compressibility calculation error	The compressibility calculation resulted in error based on the input values and configuration parameters used.	Check the input values and meter configuration parameters. Alarm Code values: 0 = No alarm 1 = Density exceeded reasonable maximum (warning only) 2 = Pressure maximum found 3 = Non-convergence of procedure "braket" 4 = Non-convergence of procedure "ddetail"
Differential Pressure Low	The differential pressure value transferred to the module is below the DP Alarm Threshold parameter configured in the Meter Configuration.	Check the input differential pressure value transferred to the module. If the value is correct, change the DP Alarm Threshold parameter for your project.
Flow Rate Low	The flow rate value transferred to the module is below the FR Alarm Threshold parameter configured in the Meter Configuration.	Check the input flow rate value transferred to the module. If the value is correct, change the FR Alarm Threshold parameter for your project.
Pulse Frequency Low	The pulse frequency value transferred to the module is below the Frequency Alarm Threshold parameter configured in the Meter Configuration.	Check the input pulse frequency value transferred to the module. If the value is correct, change the Frequency Alarm Threshold parameter for your project.
High Water error	Set if input water content is greater than 99% (less than 1% oil). For this condition, the emulsion is deemed to be all water. Both volume and mass fractions are set to zero. The module does not perform any density correction calculation, so the "default standard density" value is assumed. This alarm is applied for emulsion liquids only.	Check that the value of process input "Water %" is reasonable Alarm Code values: 0 = No alarm 1 = Emulsion is more than 99% water
Input Out of Range	The input value is not within the range specified in the meter configuration window. Applies to temperature, pressure, differential pressure, flowing density, water content, pulse frequency ($v \geq 2.05$).	Check that the input variable's ranges (Meter Configuration / Process Input button) and the process input itself have reasonable values.

Alarm Message	Description	Solution
Orifice Characterization error	The orifice parameters (Meter Configuration / Orifice button) are invalid.	<p>Check the orifice and meter parameters. The following conditions should be true:</p> <ul style="list-style-type: none"> ▪ Orifice diameter > 0 ▪ Tube diameter > 0 ▪ Orifice diameter < Tube diameter <p>The beta ratio between the orifice and tube diameters should follow the AGA Standard.</p> <p>Alarm Code values:</p> <ul style="list-style-type: none"> ▪ 0 = No alarm ▪ 1 = Orifice diameter non-positive ▪ 2 = Orifice not narrower than pipe ▪ 3 = Beta ratio less than 0.10 (adjusted by tolerance) ▪ 4 = Beta ratio greater than 0.75 (adjusted by tolerance) ▪ 5 = Pipe diameter less than 2.0 inches (adjusted by tolerance) ▪ 6 = Orifice diameter less than 0.45 inches (adjusted by tolerance) <p>The "tolerance", fixed by the AFC firmware, allows the AGA limits to be exceeded by up to 75% towards the physical limit. For example, while AGA restricts pipe diameter to 2.0 inches or greater, the AFC allows it to be as small as 0.5 inch.</p>
Orifice Pressure Exception	Configuration and process input for an Orifice Meter are such that the effective downstream pressure is less than vacuum. For calculation, upstream pressure is raised by the amount necessary to raise absolute downstream pressure to zero.	Check the process inputs for Gauge Pressure and Differential Pressure, and the configured Barometric Pressure and Static Pressure Tap Location. Also check any performed vapor pressure calculations to ensure that all are reasonable.
Pressure correction error	The pressure correction calculation resulted in an error according to the standard.	<p>Alarm Code values:</p> <ul style="list-style-type: none"> 0 = No alarm 1 = Density outside range of API Chapter 11.2 2 = Temperature above near critical limit 3 = Temperature outside range of API Chapter 11.2.1 4 = Temperature outside range of API Chapter 11.2.2 5 = Non-convergence of Cpl-density iteration

Alarm Message	Description	Solution
Reference density error	The density correction calculation resulted in an error according to the standard.	Alarm Code values: 0 = No alarm 1 = Low density (NGLs), input outside API range 2 = High density (crudes & refined), input outside API range 3 = Non-convergence 4 = Zero VCF 5 = Temperature above critical point 6 = Input density outside reference fluid adjusted range 7 = Corrected density out of range 8 = Standard density input outside API range 9 = Alpha input outside API range Also check the input values and calculation parameters for your project.
Temperature Correction error	The temperature correction calculation OR the water temperature correction calculation resulted in an error according to the standard.	Alarm Code values: 0 = No alarm 1 = Low density (NGLs), input outside API range 2 = High density (crudes & refined), input outside API range 5 = Temperature above critical point 9 = Alpha input outside API range Also see the Alarm Code for Water Temperature Correction error.
Vapor pressure error	The vapor pressure calculation resulted in an error according to the standard.	Alarm Code values: 0 = No alarm 1 = Expected vapor pressure above range of TP-15 (stream's "Default Vapor Pressure" is substituted) 2 = Vapor pressure > measured static absolute pressure (vapor pressure assumed to equal static pressure) 3 = Both 1 and 2
Water Temperature error (Alarm Code only)	The water temperature correction calculation resulted in an error according to the standard. This Alarm Code sets the "Temperature Correction error" alarm bit.	Alarm Code values: 0 = No alarm 1 = Temperature < 0°C (32°F) or > 138°C (280°F)

13.4 Checksum Alarms

A checksum alarm indicates a checksum verification failure during power-up. Non-volatile information is kept in battery-backed RAM. It is partitioned into several blocks, each of which contains a checksum, and when the information is changed the checksum is updated also. During power-up, the checksum is verified, and upon failure the alarm bit is latched and the checksum corrected.

The alarm bit remains latched, even through subsequent power cycles, until it is explicitly cleared from an external source such as the AFC Manager. Refer to the AFC Manager User Manual for more information about this feature.

13.5 Events

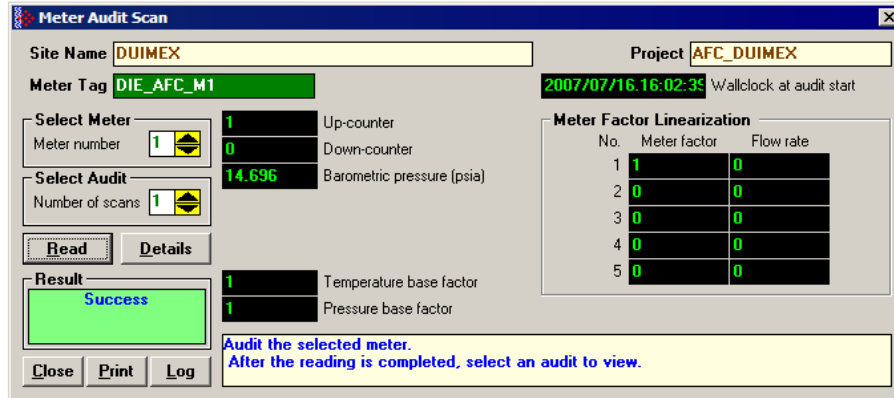
The module records up to 1999 events that have occurred during the module operation.

Important Note: Events are occurrences that may affect the results calculated by the module. This is an essential tool for troubleshooting the module.

Refer to the Events section for more information about event monitor.

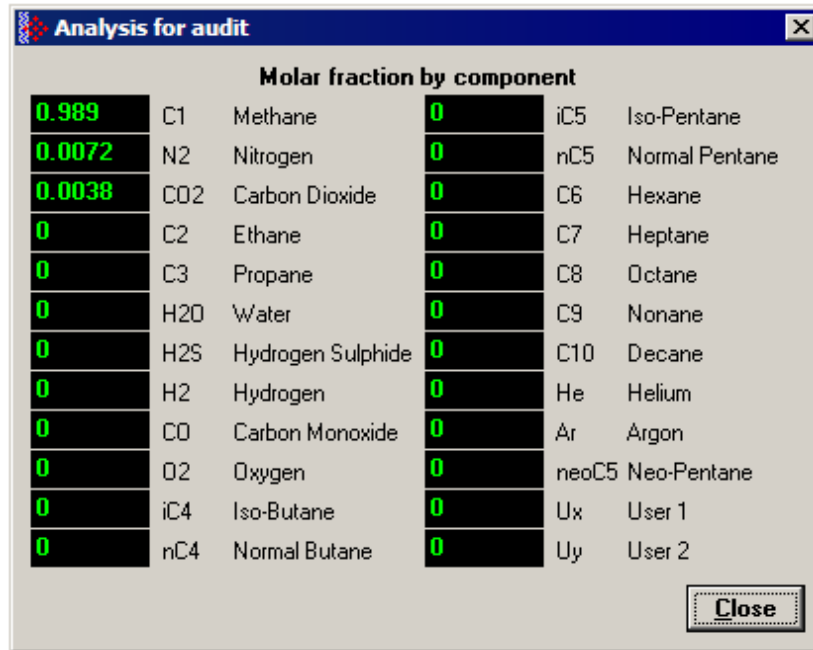
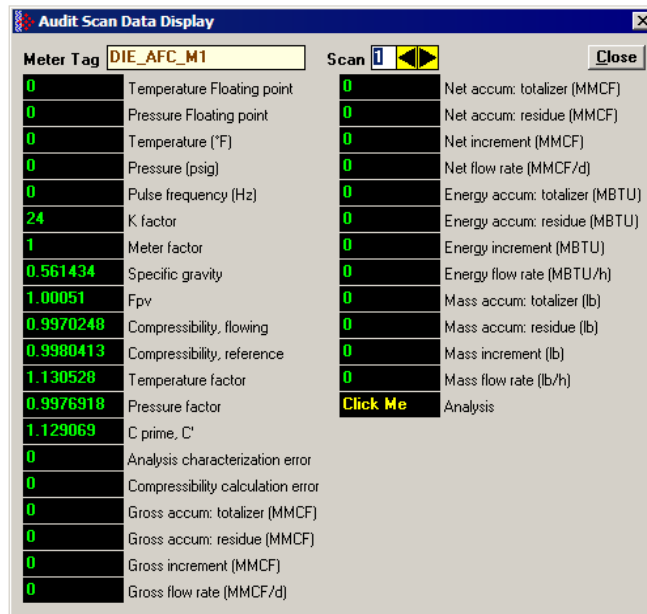
13.6 Audit Scan

An Audit Scan captures a "snapshot" of input values, intermediate calculated values, and output results for each of a short series of calculation scans for a single meter. This allows an auditor to rigorously verify the calculations performed by the AFC on live in-service production meters. The module supports eight consecutive audit scans at a time.



- 1 Select the Meter Number for the audit
- 2 Select the number of scans for the audit
- 3 Click the Read Button to begin the audit
- 4 Look at the operation result. Success = audit has been successfully completed

- When the Audit Scan is complete, click the Details Button to view the calculation and the input variables.



The following shows an example of an audit scan file report generated by the AFC Manager for 2 scans:

AFC-56(16) Audit
Site Name: MVI Flow Station
Project: AFC

Date: 16-09-2002 16:18:07

Meter 1:		
Tag		M01
Wallclock		0000/00/00.00:00:00
Barometric pressurekPaa		101,325
Viscosity		0,010268
Orifice/pipe geometric parameters		
	Orifice plate	Meter tube
Temperature	68	68
Diameter	1	2
Coefficient	9,25E-06	0,0000062

Scan		1
Temperature (Floating point)		15
Pressure (Floating point)		1000
Dif. pressure (Floating point)		22
Temperature (°F)		15
Pressure (psig)		1000
Dif. pressure (hw)		22
Scan period (second)		0,48
Specific gravity		0,7404104
Fpv		0
Compressibility flowing		0,9051347
Compressibility reference		0,9989105
Diameter at T tube		1,999343
Diameter at T orifice		0,9995098
Velocity of approach factor ev		1,032773
Pressure extension xt		149,4683
Coefficient of discharge cd		0,6042569
Expansion factor y		0,9997441
Composition factor		0,2728558
Mass flow Qh		2280,571
Orifice characterization error		0
Analysis characterization error		0
AGA8 calculation error		0
Gross accu. - totalizer (x f3)		3408
Gross accu. - residue (x f3)		0,2047686
Gross increment (x f3)		6,442598E-02
Gross flow rate (x f3/h)		483,1948
Net accu. - totalizer (x f3)		390113
Net accu. - residue (x f3)		0,8464546
Net increment (x f3)		5,3664
Net flow rate (x f3/h)		40248
Mass accu. - totalizer (x lb)		22094
Mass accu. - residue (x lb)		0,5677222
Mass increment (x lb)		0,3040761
Mass flow rate (x lb/h)		2280,571
Analysis components		
C1 methane		0,55
N2 nitrogen		0,45
CO2 carbon dioxide		0
C2 ethane		0
C3 propane		0
H2O water		0
H2S hydrogen sulphide		0
H2 hydrogen		0
CO carbon monoxide		0

O2 oxygen	0
iC4 iso-butane	0
nC4 normal butane	0
iC5 iso-pentane	0
nC5 normal pentane	0
C6 hexane	0
C7 heptane	0
C8 octane	0
C9 nonane	0
C10 decane	0
He helium	0
Ar argon	0
neoC5 neopentane	0
Ux user1	0
Uy user2	0

AFC-56(16) Audit
 Site Name: MVI Flow Station
 Project: AFC

Date: 16-09-2002 16:18:08

Meter 1:

Tag		M01
Wallclock		0000/00/00.00:00:00
Barometric pressurekPaa		101,325
Viscosity		0,010268
Orifice/pipe geometric parameters		
	Orifice plate	Meter tube
Temperature	68	68
Diameter	1	2
Coefficient	9,25E-06	0,0000062

Scan	2
Temperature (Floating point)	15
Pressure (Floating point)	1000
Dif. pressure (Floating point)	22
Temperature (°F)	15
Pressure (psig)	1000
Dif. pressure (hw)	22
Scan period (second)	0,495
Specific gravity	0,7404104
Fpv	0
Compressibility flowing	0,9051347
Compressibility reference	0,9989105
Diameter at T tube	1,999343
Diameter at T orifice	0,9995098
Velocity of approach factor ev	1,032773
Pressure extension xt	149,4683
Coefficient of discharge cd	0,6042569
Expansion factor y	0,9997441
Composition factor	0,2728558
Mass flow Qh	2280,571
Orifice characterization error	0
Analysis characterization error	0
AGA8 calculation error	0

Gross accu.- totalizer (x f3)	3408
Gross accu. - residue (x f3)	0,2712079
Gross increment (x f3)	6,643929E-02
Gross flow rate (x f3/h)	483,1948
Net accu. - totalizer (x f3)	390119
Net accu. - residue (x f3)	0,3805552
Net increment (x f3)	5,534101
Net flow rate (x f3/h)	40248
Mass accu. - totalizer (x lb)	22094
Mass accu. - residue (x lb)	0,8813007
Mass increment (x lb)	0,3135785
Mass flow rate (x lb/h)	2280,571
Analysis components	
C1 methane	0
N2 nitrogen	0
CO2 carbon dioxide	0
C2 ethane	0
C3 propane	0
H2O water	0
H2S hydrogen sulphide	0
H2 hydrogen	0
CO carbon monoxide	0
O2 oxygen	0
iC4 iso-butane	0
nC4 normal butane	0
iC5 iso-pentane	0
nC5 normal pentane	0
C6 hexane	0
C7 heptane	0
C8 octane	0
C9 nonane	0
C10 decane	0
He helium	0
Ar argon	0
neoC5 neopentane	0
Ux user1	0
Uy user2	0

14 Reference

In This Chapter

❖ General Specifications.....	270
❖ Measurement Standards	274
❖ Wedge Meter Applications.....	279
❖ Configurable Archive Registers	280
❖ Archive Data Format	286
❖ Modbus Addressing Common to Both Primary and Virtual Slaves.....	293
❖ Modbus Port configuration.....	296
❖ Startup Basics and Frequently Asked Questions	298
❖ Cable Connections	299

14.1 General Specifications

- Process I/O: analog inputs (pressure, temperature, differential pressure density) from analog modules and pulse inputs from pulse/frequency input modules in rack
- Number of meter channels: 16 meters: differential (AGA3 or ISO5167) or linear (AGA7) Gas; (MPMS Ch 12.2) Liquid.

Calculation Methods

- AGA3 (1992)
- AGA7
- AGA8 (1992) Detail Characterization Method
- API MPMS Ch 21.1, 21.2
- API Tables (API MPMS Ch 11.1) 23/53 and 24/54 for Hydrocarbon Liquids
- GPA TP-27 for Hydrocarbon Liquids (Tables 23E/24E)
- API MPMS Ch 11.2
- GPA TP-15 for Vapor Pressure Correlation
- Energy (heating value) for gases according to AGA 8 Appendix C-4
- API MPMS Ch 20.1
- ISO 5167

Supports energy measurement for gas applications

Meter I/O Scan Time: Less than one second for all channels.

Product Measurement: Hydrocarbon gases and liquids including refined products

Process I/O Calibration Mode: Allows the calibration of transmitters without interfering with the process update for the module or impacting measurement.

Data Archiving

- Hourly for 2 days for each meter run (48 records per channel)
- Daily for 35 days

Note: The number of archives depends on the archive size you have configured. The default values for a 30 word archive are 48 hourly archives and 35 daily archives.

- Extended Archive feature supports up to 1440 daily archives and 1440 hourly archives stored on Compact Flash
- Each record consists of nearly 20 process and other variables. All archived data is available in the onboard Modbus memory map.
- User may configure when archives are generated
- User may configure archive content (from pre-defined list)
- Archives can be exported to an Excel spreadsheet or printed to a local printer.

Other Features

- Event Log with 1999-event buffer and timestamp.
- Virtual Slave with 20,000 re-mappable Modbus registers for contiguous SCADA polling.
- Password protection

14.1.1 On-line Communication & Configuration

The module is designed for online configuration via the configuration port. A user-friendly Windows 95/98/2000/NT/XP-based Module Configuration and Reporting/Monitoring Manager allows easy access to all configuration data for editing and saving on your computer.

Project configurations may be uploaded, downloaded, and saved to disk under user-selectable filenames. The module takes just minutes to configure using the MS Windows-based AFC Manager.

14.1.2 Reports

- **Event Log Report:** All security-sensitive configuration data (for example, orifice diameter) is date and time stamped and mapped to the local Modbus memory map. This data can be imported into any spreadsheet program and saved to disk or printed to a local printer.
- **Hourly and Daily Archive Reports:** Mapped to local Modbus memory. This data can be imported into any spreadsheet program and saved to disk, or printed as hard copy.
- **System Configuration:** May be transferred to or from the module. The configuration file can also be printed for hard reference or archiving.
- **Audit Scan:** A report can be saved to disk or printed to the local printer.

14.1.3 Modbus Interface

The two Modbus Slave ports allow the unit to be used as a SCADA interface and to broaden access to the AFC module's data table.

- Ports 2 and 3 support RS-232, RS-422 and RS-485 modes
- Supports baud rates of up to 19200 baud
- All ports may be configured for RTU or ASCII Modbus mode.
- All Modbus Slave ports provide access to all configuration and measurement data mapped to the Modbus table.
- Module contains two internal slaves (Primary and Virtual)
- Over 130,000 Modbus registers of the Primary Slave table may be re-mapped to up to 20,000 Modbus registers of the Virtual Slave for contiguous polling from a SCADA master.
- Port 3 can be configured as a Modbus master node
- Supports Modbus functions 3, 4, 5, 6, 15, and 16 as a slave (5 and 15 only on pass-thru operation)
- Supports Modbus functions 1, 2, 3, 4, 15, and 16 as a master
- Scratch Pad Modbus block of 6000 words for transfer of arbitrary data between the processor and the SCADA host via the module.

14.1.4 Configurable Options

Configurable options include:

- Gas analysis concentrations for up to 21 components
- Accumulator Rollover
- Reference temperature and pressure for both gases and liquids
- Orifice and pipe diameters, selection of type of taps, and tap locations, and so on.
- Meter K Factor and Meter Factors with 5-point linearization curve
- Temperature, Pressure, and Density Correction for liquids
- Local Atmospheric (barometric) pressure
- Default process and operating parameters such as DP Threshold for flow cutoff, and so on.
- Metric or US units
- User-selectable units for totalizers and flow rates on a per channel basis
- Resettable and non-resettable totalizers for every meter channel.

14.1.5 Supported Meters

The following meter types have been used with the MVI56-AFC module. Because of the broad range of meters available in today's market, refer to the meter's specifications and the contents of this manual to evaluate the use of the AFC modules (even if the meter is listed here). If you have questions, please contact ProSoft Technology Technical Support Group.

Meter Type	Configured As (Differential or Linear)
Turbine	Linear
Orifice	Differential
V-Cone	Differential. You must configure the meter as V-Cone type in the AFC Manager (Meter Configuration / Calculation Options)
Wedge	Differential. Refer to Wedge Meter Applications (page 279) for information about using the wedge meters.
Vortex	Linear or Differential
Ultrasonic	Linear or Differential
Coriolis	Linear or Differential

Note: For Vortex, Ultrasonic or Coriolis meters, the selection depends on the output generated by the meter.

If the meter provides a pulse train representing the volume increment, the AFC meter should be configured as Linear with Primary Input selected as Pulse Count.

If the meter provides the instantaneous flow rate, then the AFC meter should be configured as Differential with Primary Input selected as Flow Rate.

Note: The module does not support applications to measure water, because the implemented standards are applicable to hydrocarbon fluids only.

14.1.6 Hardware Specifications

Specification	Description
Backplane Current Load	800 mA @ 5 Vdc 3 mA @ 24 Vdc
Operating Temperature	0°C to 60°C (32°F to 140°F)
Storage Temperature	-40°C to 85°C (-40°F to 185°F)
Shock	30 g operational, 50 g non-operational Vibration: 5 g from 10 Hz to 150 Hz
Relative Humidity	5% to 95% (with no condensation)
LED Indicators	Module Status, Backplane Transfer Status, Application Status, Serial Activity
Debug/Configuration port (CFG)	
CFG Port (CFG)	RJ45 (DB-9M with supplied cable), RS-232 only
Application ports (MODBUS 2 & MODBUS 3)	
Full hardware handshaking control, providing radio, modem and multi-drop support	
App Ports (MODBUS 2 & MODBUS 3)	RJ45 (DB-9M with supplied cable) RS-232, RS-485, RS-422 jumper selectable RS-232 handshaking configurable 500 V Optical isolation from backplane
Shipped with Unit	RJ45 to DB-9M cables for each port 6-foot RS-232 configuration cable

14.2 Measurement Standards

The module supports the following hydrocarbon (gases and liquids) measurement standards currently employed in the oil and gas measurement industry:

American Petroleum Institute (API) Manual of Petroleum Measurement Standards (MPMS)

- a. Density Correction to Reference Temperature
Chapter 11.1.53, 11.1.23
Equations, Tables 53, 23 - Generalized Crude Oils, Refined Products, Lubricating Oils, Special Applications

 - b. Correction of Volume to Reference Temperature and Thermal Expansion: Ctl.
Chapter 11.1.54, 11.1.24
Equations, Tables 54, 24 - Generalized Crude Oils, Refined Products, Lubricating Oils, Special Applications

 - c. Compressibility Factors for Hydrocarbons: Cpl.
Chapter 11.2.1/Chapter 11.2.2 (Chapter 11.2.1M and 11.2.2M for SI units.)

 - d. Orifice Metering of NGLs & Crude Oils
Chapter 14.3 (AGA3)

 - e. Calculation of Liquid Petroleum Quantities Measured by Turbine or Displacement Meters
Chapter 12.2

 - f. Allocation Measurement
Chapter 20.1 (high-water-content calculations used for emulsions)

 - g. Flow Measurement Using Electronic Metering Systems
Chapter 21.1, 21.2

 - h. Proving reports (firmware version 2.07 and later // Chapter 12.3)
-

American Gas Association (AGA)

- a. Orifice Metering of Natural Gas & Other Hydrocarbon Fluids
AGA Report No. 3 (1992) (MPMS Ch 14.3)

 - b. Compressibility Factors of Natural Gas and Other Related Hydrocarbon Gases
AGA Report No. 8 (1992) - Detail Characterization Method
-

International Standards Organization (ISO)

- a. Measurement of fluid flow by means of pressure differential devices inserted in circular cross-section conduits running full - Part 2: Orifice plates
ISO 5167-2 (2003)
-

Gas Processors Association (GPA)

- a. Temperature Correction for the Volume of Light Hydrocarbons - TP-25

 - b. A Simplified Vapor Pressure Completion for Commercial NGLs
GPA Document TP-15
-

14.2.1 Basic Metering According to Meter type

Orifice (Include V-cone): Uses AGA3 1992 / ISO 5167.

A V-cone meter is like an orifice meter, except that the V-cone is an obstruction in the center of the pipe while an orifice is an aperture. V-cone calculation differs from orifice calculation in the following respects:

- 1 The orifice Beta ratio is actually the square root of the ratio of aperture cross-section to pipe cross-section hence for the V-cone it is calculated differently from the two diameters.
- 2 The V-cone Coefficient of Discharge is entered as configuration and not calculated. Expansion Factor (Y) is calculated differently.

Output of the calculation is mass flow rate, which is divided by density to get volume and then integrated over time for accumulation.

Pulse: Both Gas and Liquid

Gross Volume is (pulses) / (K-factor) * (meter factor), according to API MPMS Ch 12 sec 2 1981 and 1995. Output of the standard calculation is volume flow increment, which is then multiplied by density to get mass increment. Flow rate is calculated in parallel to flow increment by applying to (pulse frequency) process input the same calculation as is applied to (pulses); this technique is employed instead of flow increment differentiation because the pulse frequency available from the counter card in the processor is not subject to variations of timing caused by scheduling delays in processor backplane transfer and in the firmware of the module, thus yielding a smoother flow rate.

Correction Factors According to Product Phase

Gas

Compressibility is calculated according to the Detail Characterization Method of AGA8 (1992). Gas density is a byproduct of this calculation. Essential input for this calculation is molar analysis. The compressibility Z is a factor in the gas equation $PV=ZNRT$, which is the rule by which gas volumes are corrected to reference conditions.

Liquid

Temperature and pressure correction factors are calculated according to API MPMS Ch 11 and applied according to the rules given in MPMS Ch 12. Essential input for this calculation is Liquid Density (page 36) at either standard or flowing conditions.

Gas Pulse Measurement

The standard applied is AGA7, which is merely a combination of the gross volume calculation (page 275) and the gas law ($PV=ZNRT$) which includes compressibility. It also specifies calculation of some intermediate factors, which are now idiosyncratic and vestigial, having been imported from an earlier AGA3 (1985 and before) which used the "factor" method to calculate gas flow and which has been superseded by the completely overhauled 1990/1992 AGA3.

Water Content of Liquids

The handling of water content in crude and NGL products depends upon whether an "emulsion" Product Group is chosen.

For emulsions, water content is removed from the mixture according to the calculations of API MPMS Chapter 20.1 before calculating and applying correction factors. In this case the volumetric quantity intermediate between "Gross" and "Net" is "Gross Clean Oil", which is the hydrocarbon component of the mixture at flowing conditions. This method is recommended for mixtures containing more than 5% water.

For non-emulsions, water content is removed from the mixture according to the rules of API MPMS Chapter 12.2 after calculating and applying correction factors. In this case the volumetric quantity intermediate between "Gross" and "Net" is "Gross Standard", which is the entire mixture including its water content corrected to standard conditions under the assumption that it is pure hydrocarbon. Because the presence of water skews the correction calculations, this method should be used only when the water content is very low.

Non-Standard Reference Conditions

For both liquids and gases, the AFC permits a range of reference conditions for volume measurement which may vary from the API/AGA standard of 15°C/101.325kPaa (SI) or 60°F/14.696psia (US) (US pressure base for gases is 14.73psia). The allowed ranges for SI units are temperature between 0°C and 25°C and pressure between 50kPaa and 110kPaa, with the allowed ranges for US units approximately equivalent.

For gases, this flexibility of reference conditions is handled automatically by the implementation of the AGA 8 (1992) standard for compressibility Z together with the "real" gas law $PV=ZNRT$.

For liquids, correction factors for non-standard reference conditions are calculated differently depending on the firmware version. For version 2.05 and later, correction factors and corrected density are calculated according to the 2004 edition of API MPMS Chapter 11.1, except for the "NGL" product groups for which the CTL and density calculations of GPA TP-27 are extended with the CPL calculations of (old) MPMS Chapter 11.2 in a manner analogous to that of the 2004 Chapter 11.1. For version 2.04 and earlier, correction factors and corrected density are calculated as described in the following paragraphs, using the calculations of the 1980 edition of MPMS Chapter 11.1. In all cases, the density input to the calculations is the density at standard API base conditions.

Temperature Correction Factor, CTL

First, the "standard" factor, $CTL(\text{Flowing} / \text{ApiBase})$, is calculated, except that the final rounding step is not performed. Then, $CTL(\text{UserBase} / \text{ApiBase})$ is calculated, also unrounded. The $CTL(\text{Flowing} / \text{UserBase})$ is then calculated as $(CTL(\text{Flowing} / \text{ApiBase}) / CTL(\text{UserBase} / \text{ApiBase}))$, to which result is applied the final rounding step of the standard CTL calculation.

Pressure Correction Factor, CPL

The CPL(Flowing / UserBase) is calculated according to the method given in MPMS Ch 12.2 1995. In order to correct "density at reference" to User Base conditions, and also when iteratively calculating corrected density for the effect of elevated pressure, the CPL(Flowing / ApiBase) (unrounded) is also calculated according to the same method.

Density Correction

The density at API Base is determined according to relevant standards, which density is used as input to the CTL and CPL calculations. The density at User Base is determined by multiplying den(ApiBase) by the term $(CTL(UserBase / ApiBase) * CPL(Flowing / ApiBase) / CPL(Flowing / UserBase))$, all unrounded factors; this density is reported only and is not used in any calculations. When density correction is not selected, or an alarm causes a default to be assumed, any default "density at reference conditions" is deemed to be at User Base, and is also corrected to API Base for input to the CTL and CPL calculations.

Archiving and Event Log

- a) Accumulation and data recording for gas-phase archives conform to the requirements of API MPMS Ch 21 sec 1, 1993. Liquid-phase archives conform to API MPMS Ch 21 sec 2.
- b) Event-logging conforms to the requirements given in the Industry Canada Weights and Measures Board Draft Specification "Metrological Audit Trails" of 1995-03-01

14.2.2 Liquid Correction Factor Details

For firmware version 2.05 and later, correction factors for most liquids are calculated according to the 2004 edition of API MPMS Chapter 11.1, enhanced with additional CPL calculations if required in order to allow selection of a non-standard base (reference) pressure. For lighter liquids (NGLs and LPGs), to which the 2004 Chapter 11.1 does not apply, the CTL and density correction calculations of GPA TP-27 are enhanced with the incorporation of the CPL calculations of MPMS Chapters 11.2.1 and 11.2.2 in a manner analogous to the method of the 2004 Chapter 11.1, to permit density correction to account for the effect of pressure and to yield the combined correction factor CTPL. For all liquids the option is available to use the vapor pressure correlation of GPA TP-15 June 1988.

For firmware version 2.04 and earlier, correction factors are calculated as described in the following paragraphs.

Temperature Correction Factor CTL

(According to Several "Tables" of MPMS Ch 11.1 (1980, except E Tables 1998 = GPA TP-25) and Other Standards)

Calculation of CTL (= VCF, Volume Correction Factor) from flowing temperature and density at standard temperature depends on the measurement system (SI or US), the product type (crude or refined), and the density range (high or low).

SI units:

$D \geq 610$ kg/m³ Table 54A (Crude&NGL) or 54B (Refined Products)

$500 \leq D < 610$ (LPG) ASTM-IP-API Petroleum Measurement Tables for Light Hydrocarbon Liquids 500-653 kg/m³ 1986 ISBN 0 471 90961 0

US units:

$D \geq 0.610$ RD60 Table 24A (Crude&NGL) or 24B (Refined Products),

$0.350 \leq D < 0.610$ (LPG) Table 24E - TP25

The low density range of 0.350 RD60 in US units is considerably lower than the 500 kg/m³ in SI units, because the E Tables are available only for US units.

Correction of density from flowing temperature to standard temperature is a converging iteration which includes the calculation of the VCF (Volume Correction Factor). Standards applied are those listed above except that Tables n3x are used instead of Tables n4x.

An option is available to iteratively correct the density calculation for elevated flowing pressure according to the condition given in bold type in MPMS Ch12.2 1995 Part 1 Appendix B Section B.1 (page 21).

Compressibility Factor F

(According to MPMS Ch 11.2 (US) or 11.2M (SI) 1986)

- Vapor pressure correlation according to GPA TP-15 June 1988.
- Pressure Correction Factor (CPL) is calculated from F and pressure above equilibrium according to MPMS ch12.2 1995, where "atmospheric pressure" is read as "base pressure" and "gage pressure" is read as "pressure above base". The module considers:

Pressure process input + barometric pressure = absolute pressure

14.3 Wedge Meter Applications

For Wedge Meter applications you must convert some parameters from the meter manufacturer's data sheet before entering these values to the AFC Manager. The following spreadsheets can be used to calculate the AFC Manager parameters according to the meter manufacturer as follows:

Filename	Application
WEDGE_ABB.xls	ABB Wedge Meter
WEDGE_PRESO.xls	PRESO Wedge Meter

You must initially configure the meter as a differential type. Then you must configure it as a V-Cone Device (**Meter Configuration / Calculation Options**). Refer to the spreadsheet for instructions on how to enter the correct values into AFC Manager.

14.4 Configurable Archive Registers

The following table shows the possible registers that can be included in the archive definition. Use the Insert and Remove buttons on the Archive Configuration dialog box to customize the list of registers for each meter archive.

Description	Meter-Relative Address	Length
Analysis molar fraction, component 1	720	1 word
Analysis molar fraction, component 2	721	1 word
Analysis molar fraction, component 3	722	1 word
Analysis molar fraction, component 4	723	1 word
Analysis molar fraction, component 5	724	1 word
Analysis molar fraction, component 6	725	1 word
Analysis molar fraction, component 7	726	1 word
Analysis molar fraction, component 8	727	1 word
Analysis molar fraction, component 9	728	1 word
Analysis molar fraction, component 10	729	1 word
Analysis molar fraction, component 11	730	1 word
Analysis molar fraction, component 12	731	1 word
Analysis molar fraction, component 13	732	1 word
Analysis molar fraction, component 14	733	1 word
Analysis molar fraction, component 15	734	1 word
Analysis molar fraction, component 16	735	1 word
Analysis molar fraction, component 17	736	1 word
Analysis molar fraction, component 18	737	1 word
Analysis molar fraction, component 19	738	1 word
Analysis molar fraction, component 20	739	1 word
Analysis molar fraction, component 21	740	1 word
Analysis molar fraction, component 22	741	1 word
Analysis molar fraction, component 23	742	1 word
Analysis molar fraction, component 24	743	1 word
Input pulse count, archive reset, daily	840	2 words
Input pulse count, archive reset, hourly	842	2 words
Previous input pulse count	846	2 words
Current master pulse count	848	2 words
Non-resettable accumulator, mass, totalizer	850	2 words
Non-resettable accumulator, mass, residue	852	2 words
Non-resettable accumulator, energy, totalizer	854	2 words
Non-resettable accumulator, energy, residue	856	2 words
Non-resettable accumulator, net, totalizer	858	2 words
Non-resettable accumulator, net, residue	860	2 words
Non-resettable accumulator, gross, totalizer	862	2 words
Non-resettable accumulator, gross, residue	864	2 words

Description	Meter-Relative Address	Length
Non-resettable accumulator, gross standard, totalizer	866	2 words
Non-resettable accumulator, gross standard, residue	868	2 words
Non-resettable accumulator, water, totalizer	870	2 words
Non-resettable accumulator, water, residue	872	2 words
Resettable accumulator 1, totalizer	874	2 words
Resettable accumulator 1, residue	876	2 words
Resettable accumulator 2, totalizer	878	2 words
Resettable accumulator 2, residue	880	2 words
Resettable accumulator 3, totalizer	882	2 words
Resettable accumulator 3, residue	884	2 words
Resettable accumulator 4, totalizer	886	2 words
Resettable accumulator 4, residue	888	2 words
Accumulator, archive period, daily, totalizer	890	2 words
Accumulator, archive period, daily, residue	892	2 words
Accumulator, archive period, hourly, totalizer	894	2 words
Accumulator, archive period, hourly, residue	896	2 words
Process input, scaled float, temperature	1520	2 words
Process input, scaled float, pressure	1522	2 words
Process input, scaled float, dif prs / flow rate / freq	1524	2 words
Process input, scaled float, flowing density	1526	2 words
Process input, scaled float, water and sediment	1528	2 words
Process input, scaled integer, temperature	1540	1 word
Process input, scaled integer, pressure	1541	1 word
Process input, scaled integer, dif prs / flow rate / freq	1542	1 word
Process input, scaled integer, flowing density	1543	1 word
Process input, scaled integer, water and sediment	1544	1 word
Temperature, absolute	1570	2 words
Upstream pressure, absolute	1572	2 words
Densitometer frequency	1574	2 words
AGA 7 temperature base factor, Ftb	1594	2 words
AGA 7 pressure base factor, Fpb	1596	2 words
Meter alarms	1601	1 word
Orifice characterization error	1602	1 word
Analysis characterization error	1603	1 word
AGA 8 calculation error	1604	1 word
Density correction error	1605	1 word
Temperature correction error	1606	1 word
Vapor pressure error	1607	1 word
Pressure correction error	1608	1 word
Scan count, process input	1618	1 word

Description	Meter-Relative Address	Length
Scan count, calculation	1619	1 word
AGA 8, Molar mass of mixture	1620	2 words
AGA 8, Ideal gas relative density	1622	2 words
AGA 8, Compressibility at reference	1624	2 words
AGA 8, Molar density at reference	1626	2 words
AGA 8, Density at reference	1628	2 words
AGA 8, Relative density at reference	1630	2 words
AGA 8, Compressibility, flowing	1632	2 words
AGA 8, Molar density, flowing	1634	2 words
AGA 8, Density, flowing	1636	2 words
AGA 8, Supercompressibility, Fpv	1640	2 words
Previous timer tick count	1661	1 word
Scan period (seconds)	1662	2 words
AGA 3, Pressure extension	1664	2 words
AGA 3, Differential pressure in static pressure units	1666	2 words
AGA 3, Orifice bore diameter at temperature	1668	2 words
AGA 3, Meter tube internal diameter at temperature	1670	2 words
Reserved	1672	2 words
AGA 3, Density, flowing	1674	2 words
AGA 3, Mass flow rate, Qm	1678	2 words
AGA 3, Velocity of approach factor, Ev	1680	2 words
AGA 3, Expansion factor, Y	1682	2 words
AGA 3, Coefficient of discharge, Cd	1684	2 words
AGA 3, Composition factor	1686	2 words
AGA 7, Temperature factor, Ftm	1694	2 words
AGA 7, Pressure factor, Fpm	1696	2 words
AGA 7, C-prime	1698	2 words
Molar heating value, MJ/kmol	1700	2 words
Mass heating value	1702	2 words
Volumetric heating value	1704	2 words
MPMS Ch 11, Density at API base temperature	1738	2 words
MPMS Ch 11, Hydrometer correction factor	1740	2 words
MPMS Ch 11, Density at reference	1742	2 words
MPMS Ch 11, Vapor pressure	1744	2 words
MPMS Ch 11, CPL low density factor A	1746	2 words
MPMS Ch 11, CPL low density factor B	1748	2 words
MPMS Ch 11, CPL factor F	1750	2 words
MPMS Ch 11, Temperature correction factor, CTL	1752	2 words
MPMS Ch 11, Pressure correction factor, CPL	1754	2 words
MPMS Ch 11, Sediment and water correction factor, CSW	1756	2 words

Description	Meter-Relative Address	Length
Density calculation select	1759	1 word
AGA 8, Ideal gas relative density - scaled integer	1761	1 word
AGA 8, Compressibility at reference - scaled integer	1762	1 word
AGA 8, Relative density at reference - scaled integer	1765	1 word
AGA 8, Compressibility, flowing - scaled integer	1766	1 word
AGA 8, Supercompressibility, Fpv - scaled integer	1770	1 word
Reserved	1786	1 word
AGA 3, Velocity of approach factor - scaled integer	1790	1 word
AGA 3, Expansion factor - scaled integer	1791	1 word
AGA 3, Coefficient of discharge - scaled integer	1792	1 word
MPMS Ch 11, Density at reference	1821	1 word
MPMS Ch 11, Vapor pressure	1822	1 word
MPMS Ch 11, Temperature correction factor, CTL	1826	1 word
MPMS Ch 11, Pressure correction factor, CPL	1827	1 word
MPMS Ch 11, Sediment and water correction factor, CSW	1828	1 word
Startup input pulse count	1840	2 words
Current input pulse count	1842	2 words
Pulse increment	1844	2 words
Pulse frequency	1846	2 words
Interpolated/static K-factor	1848	2 words
Interpolated/static meter factor	1850	2 words
Multiplier, mass flow rate	1864	2 words
Multiplier, energy flow rate	1866	2 words
Multiplier, volume flow rate	1868	2 words
Multiplier, mass accumulator	1870	2 words
Multiplier, energy accumulator	1872	2 words
Multiplier, volume accumulator	1874	2 words
Accumulator increment, mass	1876	2 words
Accumulator increment, energy	1878	2 words
Accumulator increment, net	1880	2 words
Accumulator increment, gross	1882	2 words
Accumulator increment, gross standard	1884	2 words
Accumulator increment, water	1886	2 words
Flow rate, mass	1888	2 words
Flow rate, energy	1890	2 words
Flow rate, net	1892	2 words
Flow rate, gross	1894	2 words
Flow rate, gross standard	1896	2 words
Flow rate, water	1898	2 words

14.4.1 Information for Users of AFC Manager Versions Older Than 2.01.000

If you are using AFC Manager versions older than 2.01.000, you must set these bits using the Modbus master interface in the AFC Manager. Please refer to the AFC Manager User Manual for further information about the Modbus Master interface feature.

Refer to the following words to configure the archive options directly to the Modbus database:

Address	Description
8341	Meter 1 daily archive configuration word
8421	Meter 1 hourly archive configuration word
10341	Meter 2 daily archive configuration word
10421	Meter 2 hourly archive configuration word
12341	Meter 3 daily archive configuration word
12421	Meter 3 hourly archive configuration word
14341	Meter 4 daily archive configuration word
14421	Meter 4 hourly archive configuration word
16341	Meter 5 daily archive configuration word
16421	Meter 5 hourly archive configuration word
18341	Meter 6 daily archive configuration word
18421	Meter 6 hourly archive configuration word
20341	Meter 7 daily archive configuration word
20421	Meter 7 hourly archive configuration word
22341	Meter 8 daily archive configuration word
22421	Meter 8 hourly archive configuration word

Each archive configuration word has the following bitmap structure:

Bit	Description
0	Period select, hourly
1	Archive upon period end
2	Archive upon event
3	Reserved
4	Reset resettable accumulator 1 upon period end
5	Reset resettable accumulator 2 upon period end
6	Reset resettable accumulator 3 upon period end
7	Reset resettable accumulator 4 upon period end
8	Reset resettable accumulator 1 upon event
9	Reset resettable accumulator 2 upon event
10	Reset resettable accumulator 3 upon event
11	Reset resettable accumulator 4 upon event
12	Reserved
13	Reserved
14	Reserved
15	Reserved

Note: Bit 0 must be set only for the hourly archives.

Changes made directly to the Modbus table in this manner are not automatically made to your open AFC configuration. To incorporate these changes into your configuration so that they may be saved in the AFC file on your hard disk, you must read back the meter configuration from the module after making the change by using the "Read Configuration" button on the Meter Configuration window.

14.5 Archive Data Format

There are 3 columns associated with each archive data:

Column	Description
Ofs	Shows the offset location of the data in each archive. The maximum offset value will depend on the <i>Record Size</i> value you configured. If the value has a "+" value (for example 0+) it means that the data occupies 2 words of data.
Reg	Shows the Primary Modbus Slave Address of the data. This is a meter-relative address. For example: a Reg value of 890+ for meter 1 would be equivalent to Modbus addresses 8890 and 8891.
Description	Data Description.

14.5.1 Timestamp Date and Time Format

The date and time format used in the archives is stored in a highly compressed form in order to represent the date and time using only 2 words of data:

Word	Description
0	Date
1	Time

In order to extract the information from the date format use the following arithmetic:

Date Word

Year = ([Bits 15 thru 9] from Word 0) + 1996

Month = ([Bits 8 thru 5] from Word 0) + 1

Day = ([Bits 4 thru 0] from Word 0) + 1

Time Word

Hour = ([Bits 15 thru 11] from Word 1)

Minute = ([Bits 10 thru 5] from Word 1)

Second = ([Bits 4 thru 0] from Word 1) * 2

The first 10 words of data (archive header) are common for all archives:

14.5.2 Pre-defined Header

These archive areas are included in the default archive data, and cannot be reconfigured by the user.

Start Offset	End Offset	Data Format	Type	Description
0	1	Timestamp	Snapshot	Closing timestamp of archive
2		Word	Calculated	Flowing period
3		Bitmap	Calculated	Cumulative meter alarms
4		Bitmap	Calculated	Cumulative status
5		Word	Snapshot	Event counter
6	7	Double word	Calculated	Flowing period, seconds
8	9	Timestamp	snapshot	Opening timestamp of archive

Additional areas are also included in the default archive data, according to the meter type and product group associated with the meter.

The cumulative meter alarms are defined as follows:

Offset	Description
0	Current archive, daily, cumulative meter alarm: Input out of range, temperature
1	Current archive, daily, cumulative meter alarm: Input out of range: pressure
2	Current archive, daily, cumulative meter alarm: Input out of range: differential pressure
3	Current archive, daily, cumulative meter alarm: Input out of range: flowing density
4	Current archive, daily, cumulative meter alarm: Input out of range: water content
5	Current archive, daily, cumulative meter alarm: Differential Pressure Low
6	Current archive, daily, cumulative meter alarm: Orifice Pressure Exception
7	Current archive, daily, cumulative meter alarm: Accumulation overflow
8	Current archive, daily, cumulative meter alarm: Orifice characterization error
9	Not Used
10	Current archive, daily, cumulative meter alarm: Current archive, daily, cumulative meter alarm: Analysis characterization error
11	Current archive, daily, cumulative meter alarm: Compressibility calculation error
12	Current archive, daily, cumulative meter alarm: Reference density error
13	Current archive, daily, cumulative meter alarm: Temperature correction error
14	Current archive, daily, cumulative meter alarm: Vapor pressure error
15	Current archive, daily, cumulative meter alarm: Pressure correction error

The cumulative status bits are defined as follows:

Offset	End Offset
00	Stream 1 active
01	Stream 2 active
02	Stream 3 active
03	Stream 4 active
11	Meter enabled
12	Backplane Communication Fault
13	Measurement Configuration Changed
14	Power up
15	Cold Start

The following 20 words (default configuration) will depend on the meter type and product group as follows:

14.5.3 Orifice (Differential) Meter with Gas Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	Differential pressure
22		Word	Flow weighted average	Relative density, e-4
23		Word	Flow weighted average	Compressibility, reference, e-4
24		Word	Flow weighted average	Compressibility, flowing, e-4
25		Word	Flow weighted average	Supercompressibility, e-4
26		Word	Flow weighted average	Velocity of approach factor, Ev, e-4
27		Word	Flow weighted average	Expansion factor, Y, e-4
28		Word	Flow weighted average	Coefficient of discharge, Cd, e-4
29		Word		(available)

14.5.4 Pulse (Linear) Meter with Gas Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	K-Factor
22	23	Floating point	Flow weighted average	Meter Factor
24		Word	Flow weighted average	Relative density, e-4
25		Word	Flow weighted average	Compressibility, reference, e-4
26		Word	Flow weighted average	Compressibility, flowing, e-4
27		Word	Flow weighted average	Supercompressibility, e-4
28	29	Double Word	Snapshot	Pulse Count

14.5.5 Orifice (Differential) Meter with Liquid Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	Differential pressure
22	23	Floating point	Flow weighted average	Density input
24		Word	Flow weighted average	Corrected density (scaled integer)
25		Word	Flow weighted average	CTL e-4
26		Word	Flow weighted average	CPL e-4
27		Word	Flow weighted average	Velocity of approach factor, Ev, e-4
28		Word	Flow weighted average	Expansion factor, Y, e-4
29		Word	Flow weighted average	Coefficient of discharge, Cd, e-4

14.5.6 Pulse (Linear) Meter with Liquid Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	K-Factor
22	23	Floating point	Flow weighted average	Meter Factor
24	25	Floating point	Flow weighted average	Density Input
26		Word	Flow weighted average	Water content, % e-2
27		Word	Flow weighted average	Corrected density (scaled integer)
28		Word	Flow weighted average	CTL e-4
29		Word	Flow weighted average	CPL e-4

14.5.7 Flow Rate Integration with Gas Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	Flow Rate Input
22		Word	Flow weighted average	Relative density, e-4
23		Word	Flow weighted average	Compressibility, reference, e-4
24		Word	Flow weighted average	Compressibility, flowing, e-4
25		Word	Flow weighted average	Supercompressibility, e-4
26		Word		(available)
27		Word		(available)
28		Word		(available)
29		Word		(available)

14.5.8 Pulse Frequency Integration with Gas Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	K-Factor
22	23	Floating point	Flow weighted average	Meter Factor
24		Word	Flow weighted average	Relative density e-4
25		Word	Flow weighted average	Compressibility, reference, e-4
26		Word	Flow weighted average	Compressibility, flowing, e-4
27		Word	Flow weighted average	Supercompressibility, e-4
28	29	Floating point	Flow weighted average	Pulse Frequency

14.5.9 Flow Rate Integration with Liquid Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	Flow Rate Input
22	23	Floating point	Flow weighted average	Density Input
24		Word	Flow weighted average	Corrected density (scaled integer)
25		Word	Flow weighted average	CTL e-4
26		Word	Flow weighted average	CPL e-4
27		Word		(available)
28		Word		(available)
29		Word		(available)

14.5.10 Pulse Frequency Integration with Liquid Product

Start Offset	End Offset	Data Format	Type	Description
10	11	Accumulator	Snapshot	Accumulator totalizer, net
12	13	Floating point	Snapshot	Accumulator residue, net
14	15	Floating point	Flow weighted average	Flow rate, net
16	17	Floating point	Flow weighted average	Temperature
18	19	Floating point	Flow weighted average	Pressure
20	21	Floating point	Flow weighted average	K-Factor
22	23	Floating point	Flow weighted average	Meter Factor
24	25	Floating point	Flow weighted average	Density Input
26		Word	Flow weighted average	Water content, % e-2
27		Word	Flow weighted average	Corrected density (scaled integer)
28	29	Floating point	Flow weighted average	Pulse Frequency

Example 1

Find the Net Accumulator addresses at archive 1 (latest daily archive) for the first 4 meters.

Primary Modbus Slave Register Address	Description
10 and 11	Net Accumulator Totalizer from archive 1 - Meter 1
2510 and 2511	Net Accumulator Totalizer from archive 1 - Meter 2
5010 and 5011	Net Accumulator Totalizer from archive 1 - Meter 3
7510 and 7511	Net Accumulator Totalizer from archive 1 - Meter 4

Example 2

Find the Net Accumulator addresses at archive 0 (current daily archive) for the first 4 meters.

Primary Modbus Slave Holding Register Address	Description
9910 and 9911	Net Accumulator Totalizer from archive 0 - Meter 1
11910 and 11911	Net Accumulator Totalizer from archive 0 - Meter 2
13910 and 13911	Net Accumulator Totalizer from archive 0 - Meter 3
15910 and 15911	Net Accumulator Totalizer from archive 0 - Meter 4

14.6 Modbus Addressing Common to Both Primary and Virtual Slaves

Address	Type	Description
Ch00000	Char	Firmware product code, group Low byte: platform High byte: application class
Ch00001	Char	Firmware product code, item Low byte: number of streams High byte: number of meters
Ch00002	Int	Firmware version number Low byte: minor version number High byte: major version number
Ch00003	Int	Firmware revision number
Ch00004 to Ch00005	Int	Serial number

Address	Type	Description
Ch00006	Bm	<p>Site status</p> <p>bit 0 - AFC released</p> <p>Latched when both bit 15 (cold start) and bit 12 (Processor offline) first become clear, remaining so until any subsequent cold start. While this bit remains clear events are not logged, allowing an initial configuration to be fully completed without filling up the event log.</p> <p>bit 1 - Checksum alarm</p> <p>Set when any bit in the "Checksum Alarms" registers, for site and each meter, is set; clear when all such bits are clear.</p> <p>bit 2 - [reserved]</p> <p>bit 3 - [reserved]</p> <p>bit 4 - Processor halted, offline, or missing</p> <p>Set while backplane communication is faulty, which typically occurs when the Processor is switched to program mode. While set, measurement continues using the latest process input values obtained from the processor. Upon resumption of backplane communication, the AFC compensates for the downtime by computing an accumulator increment in a manner that depends on the meter type. For differential (orifice) meters, the first measurement scan acquires a scan period equal to the period of downtime as computed from the system timer, hence periods of processor downtime shorter than the rollover period of the system timer cause no loss of product. For linear (pulse) meters, the first measurement scan acquires a pulse increment equal to the difference between the processor-supplied pulse count of the current scan and that of the last scan before communication loss, hence periods of processor downtime shorter than the rollover period of the counter module cause no loss of product.</p> <p>bit 5 - Measurement configuration changed</p> <p>Set when any bit in the "Measurement Configuration Changed" registers is set; clear when all such bits are clear.</p> <p>bit 6 - Power up</p> <p>Set upon power-up, and cleared upon setting the wallclock for the first time..</p> <p>bit 7 - Cold start</p> <p>Upon power-up, AFC's non-volatile storage is checked for validity, by verifying a checksum and confirming that certain known values are present in their proper locations. If the storage is invalid, then it is initialized with a default configuration, and this bit is set. The bit remains set, even through subsequent power cycles, until at least one meter is enabled at which time the bit is cleared.</p> <p>bit 8 - A copy of the "Hard Passwords" site option, made available here so that an external application such as AFC Manager can learn all it needs to know in order to connect to the module by reading the first 20 holding registers from the Modbus table.</p> <p>bit 9 - [reserved]</p> <p>bit 10 - [reserved]</p> <p>bit 11 - [reserved]</p> <p>bit 12 - [reserved]</p> <p>bit 13 - [reserved]</p> <p>bit 14 - [reserved]</p> <p>bit 15 - [reserved]</p>
Ch00007	By	Processor offline code: 0 online, 1 offline
Ch00008	By	<p>Zero / primary slave address</p> <p>This value distinguishes the two slaves. When read from the primary slave this value is zero; when read from the virtual slave this value is the primary slave address.</p>
Ch00009	Wd	Password, write-enable

Address	Type	Description
Ch00010 to Ch00015	Wd	Wallclock (Y,M,D,h,m,s) The wallclock has a resolution of 1 second.
Ch00016 to Ch00017	Bm	Wallclock (packed) The packed wallclock has a resolution of 2 seconds.
Ch00018	Bm	accessed port and authorization bits 0- 3 - Accessed port; 0 = gateway bit 4 - Password authorization waived for read bit 5 - Password authorization waived for write bit 6 - Password authorization granted for read bit 7 - Password authorization granted for write
Ch00019	Wd	Password, read-enable
Ch00020 to Ch00089	--	[reserved] Reserved for use by diagnostic and similar procedures.
Ch00090 to Ch00099	Wd	Arbitrary event-logged registers. A Modbus master (such as the processor using Modbus Gateway) can use these to record in the Event Log changes to values unrelated to flow measurement.

14.7 Modbus Port configuration

Configuration of the serial ports is stored in these blocks of the Modbus table:

Address	Type	Description
Ph00102 to Ph00105	Bm	Port 1 configuration
Ph00106 to Ph00109	Bm	Port 2 configuration
Ph00110 to Ph00113	Bm	Port 3 configuration

Each group of registers specifies configuration of the corresponding serial port. The four registers of each block are interpreted as follows:

Ofs	Type	Tag	Contents
+0	Bm	Uart	UART parameters and port options
+1.L	By	TmoC	LSB: Timeout for CTS
+1.H	By	TmoR	MSB: Master mode receive timeout
+2	By	Dly1	Delay before first data after CTS
+3	By	Dly0	Delay after last data before ~RTS

The CTS timeout and both delays are in units of 5ms (200Hz system clock), with valid values from 0 thru 255, and are significant only for transmission of outgoing Modbus messages. The receive timeout is in units of 0.1 second, with valid values from 0 thru 255 (where 0 implies the default of 5, that is, one-half second), and is significant only for the last port when configured as a Modbus master. The UART parameters and port options word is a bitmap:

Bit	Parameter	Value
bits 0 to 2	Baud	000: none; see below
		001: 300 baud
		010: 600 baud
		011: 1200 baud
		100: 2400 baud
		101: 4800 baud
		110: 9600 baud
		111: 19200 baud
bits 3 to 4	Parity	00: no parity
		01: odd parity
		10: even parity
		11: no parity (should not be used)
bit 5	Data bits	0: 8 data bits
		1: 7 data bits
bit 6	Stop bits	0: 1 stop bit
		1: 2 stop bits

Bit	Parameter	Value
bit 7	Modbus mode	0: RTU mode 1: ASCII mode
bit 8	Modbus orientation	0: slave 1: master (permitted only for last port)
bit 9	Primary slave accessibility (not meaningful for master port)	0: primary slave accessible through this port 1: primary slave not accessible (not permitted for Port 1)
bit 10		Swap Modbus bytes
bit 11		Swap Modbus words
bit 12		Disable pass-thru (not meaningful for master port)
bits 13 to 15		[reserved]

A change in configuration takes effect after transmission of the response to the Modbus command that causes the change; the response is sent using the old configuration, but subsequent Modbus commands to the reconfigured port must use the new one. Writing a baud code of 0 means that the current configuration is not to be changed, and all other items are ignored. Default values are 6 for the bitmap (9600,N,8,1,RTU,slave,primary,noswap,passthru) and 0 for the timeout and both delays. The message transmission procedure is:

- Raise RTS.
- If TmoC is zero ignore CTS, else wait up to TmoC clock ticks for CTS.
- Delay for Dly1 clock ticks.
- Transmit message.
- Delay for Dly0 clock ticks.
- Drop RTS.

14.8 Startup Basics and Frequently Asked Questions

The Automatic Flow Computer (AFC) is a powerful rack flow computer solution for PLC platforms. The design intent of the module is to simplify the setup and maintenance of a meter installation. With this in mind, the sample ladder logic was created to accomplish the following:

- Pass meter run variables to the module.
- Return meter results to the processor.
- Allow individual meters to be enabled or disabled.
- Allow resets of individual meter runs.
- Allow transfer of a new gas analysis to an individual meter run.

Actual meter setup includes units of measure setup, range checking for input variables, and the type of meter being used. This setup is handled by the AFC Manager software. The intended design is to have the processor only handle the variables of an actual process and the AFC Manager handle the setup and configuration of necessary meter variables.

The sample ladder logic included with the system is intended to fulfill this requirement and works for many applications. Should you feel that your application requires more than this, then a very intimate knowledge of the operations of the module are required to be successful in the implementation of the application. It is highly recommended that the sample be used as a starting point for any application.

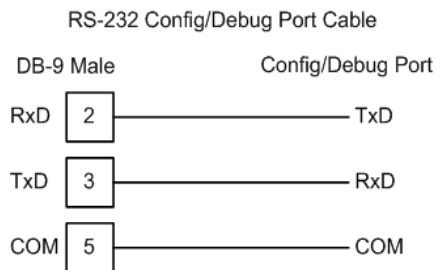
14.9 Cable Connections

The application ports on the MVI56-AFC module support RS-232, RS-422, and RS-485 interfaces. Please inspect the module to ensure that the jumpers are set correctly to correspond with the type of interface you are using.

Note: When using RS-232 with radio modem applications, some radios or modems require hardware handshaking (control and monitoring of modem signal lines). Enable this in the configuration of the module by setting the UseCTS parameter to 1.

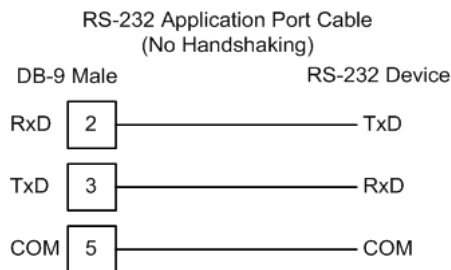
14.9.1 RS-232 Configuration/Debug Port

This port is physically an RJ45 connection. An RJ45 to DB-9 adapter cable is included with the module. This port permits a PC based terminal emulation program to view configuration and status data in the module and to control the module. The cable for communications on this port is shown in the following diagram:



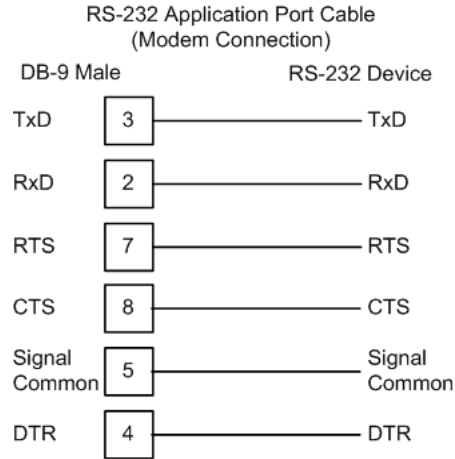
14.9.2 RS-232 Application Port(s)

When the RS-232 interface is selected, the use of hardware handshaking (control and monitoring of modem signal lines) is user definable. If no hardware handshaking will be used, here are the cable pinouts to connect to the port.



RS-232: Modem Connection (Hardware Handshaking Required)

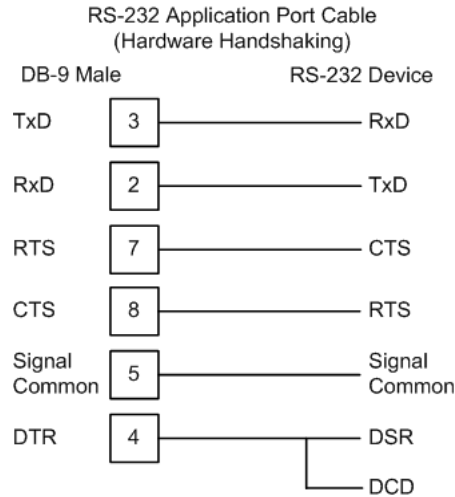
This type of connection is required between the module and a modem or other communication device.



The "Use CTS Line" parameter for the port configuration should be set to 'Y' for most modem applications.

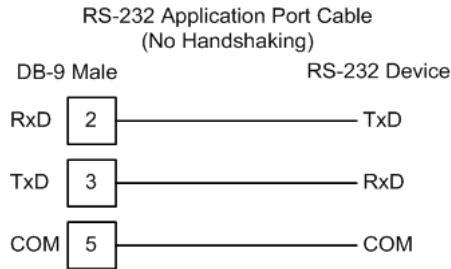
RS-232: Null Modem Connection (Hardware Handshaking)

This type of connection is used when the device connected to the module requires hardware handshaking (control and monitoring of modem signal lines).

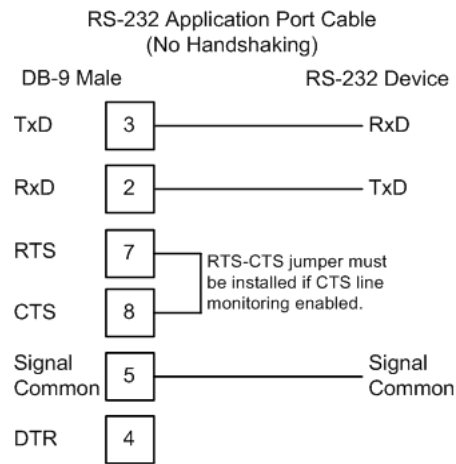


RS-232: Null Modem Connection (No Hardware Handshaking)

This type of connection can be used to connect the module to a computer or field device communication port.

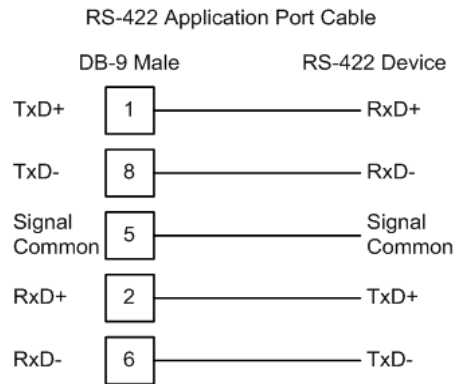


Note: For most null modem connections where hardware handshaking is not required, the *Use CTS Line* parameter should be set to **N** and no jumper will be required between Pins 7 (RTS) and 8 (CTS) on the connector. If the port is configured with the *Use CTS Line* set to **Y**, then a jumper is required between the RTS and the CTS lines on the port connection.



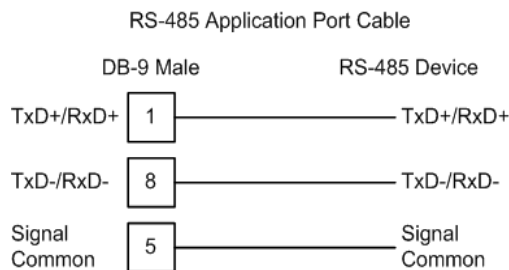
14.9.3 RS-422

The RS-422 interface requires a single four or five wire cable. The Common connection is optional, depending on the RS-422 network devices used. The cable required for this interface is shown below:



14.9.4 RS-485 Application Port(s)

The RS-485 interface requires a single two or three wire cable. The Common connection is optional, depending on the RS-485 network devices used. The cable required for this interface is shown below:

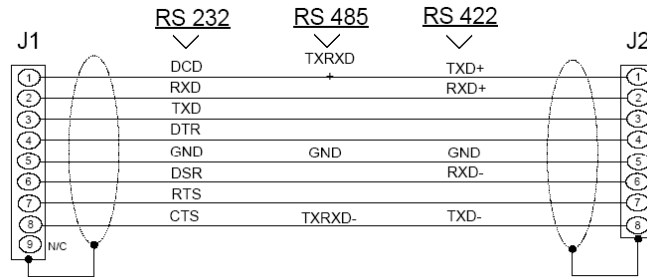
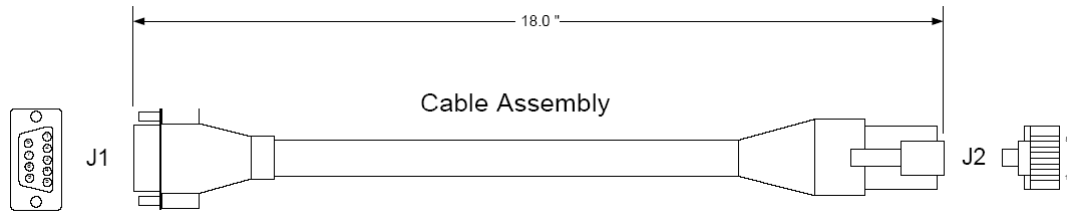


Note: Terminating resistors are generally not required on the RS-485 network, unless you are experiencing communication problems that can be attributed to signal echoes or reflections. In these cases, installing a 120-ohm terminating resistor between pins 1 and 8 on the module connector end of the RS-485 line may improve communication quality.

RS-485 and RS-422 Tip

If communication in the RS-422 or RS-485 mode does not work at first, despite all attempts, try switching termination polarities. Some manufacturers interpret + and -, or A and B, polarities differently.

14.9.5 DB9 to RJ45 Adaptor (Cable 14)



Wiring Diagram

15 Support, Service & Warranty

In This Chapter

- ❖ Contacting Technical Support 305
- ❖ Return Material Authorization (RMA) Policies and Conditions..... 307
- ❖ LIMITED WARRANTY..... 309

Contacting Technical Support

ProSoft Technology, Inc. (ProSoft) is committed to providing the most efficient and effective support possible. Before calling, please gather the following information to assist in expediting this process:

- 1 Product Version Number
- 2 System architecture
- 3 Network details

If the issue is hardware related, we will also need information regarding:

- 1 Module configuration and associated ladder files, if any
- 2 Module operation and any unusual behavior
- 3 Configuration/Debug status information
- 4 LED patterns
- 5 Details about the serial, Ethernet or fieldbus devices interfaced to the module, if any.

Note: For technical support calls within the United States, an after-hours answering system allows 24-hour/7-days-a-week pager access to one of our qualified Technical and/or Application Support Engineers. Detailed contact information for all our worldwide locations is available on the following page.

Internet	Web Site: www.prosoft-technology.com/support E-mail address: support@prosoft-technology.com
Asia Pacific (location in Malaysia)	Tel: +603.7724.2080, E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Asia Pacific (location in China)	Tel: +86.21.5187.7337 x888, E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Europe (location in Toulouse, France)	Tel: +33 (0) 5.34.36.87.20, E-mail: support.EMEA@prosoft-technology.com Languages spoken include: French, English
Europe (location in Dubai, UAE)	Tel: +971-4-214-6911, E-mail: mea@prosoft-technology.com Languages spoken include: English, Hindi
North America (location in California)	Tel: +1.661.716.5100, E-mail: support@prosoft-technology.com Languages spoken include: English, Spanish
Latin America (Oficina Regional)	Tel: +1-281-2989109, E-Mail: latinam@prosoft-technology.com Languages spoken include: Spanish, English
Latin America (location in Puebla, Mexico)	Tel: +52-222-3-99-6565, E-mail: soporte@prosoft-technology.com Languages spoken include: Spanish
Brasil (location in Sao Paulo)	Tel: +55-11-5083-3776, E-mail: brasil@prosoft-technology.com Languages spoken include: Portuguese, English

15.1 Return Material Authorization (RMA) Policies and Conditions

The following Return Material Authorization (RMA) Policies and Conditions (collectively, "RMA Policies") apply to any returned product. These RMA Policies are subject to change by ProSoft Technology, Inc., without notice. For warranty information, see Limited Warranty (page 309). In the event of any inconsistency between the RMA Policies and the Warranty, the Warranty shall govern.

15.1.1 Returning Any Product

- a) In order to return a Product for repair, exchange, or otherwise, the Customer must obtain a Return Material Authorization (RMA) number from ProSoft Technology and comply with ProSoft Technology shipping instructions.
- b) In the event that the Customer experiences a problem with the Product for any reason, Customer should contact ProSoft Technical Support at one of the telephone numbers listed above (page 305). A Technical Support Engineer will request that you perform several tests in an attempt to isolate the problem. If after completing these tests, the Product is found to be the source of the problem, we will issue an RMA.
- c) All returned Products must be shipped freight prepaid, in the original shipping container or equivalent, to the location specified by ProSoft Technology, and be accompanied by proof of purchase and receipt date. The RMA number is to be prominently marked on the outside of the shipping box. Customer agrees to insure the Product or assume the risk of loss or damage in transit. Products shipped to ProSoft Technology using a shipment method other than that specified by ProSoft Technology, or shipped without an RMA number will be returned to the Customer, freight collect. Contact ProSoft Technical Support for further information.
- d) A 10% restocking fee applies to all warranty credit returns, whereby a Customer has an application change, ordered too many, does not need, etc. Returns for credit require that all accessory parts included in the original box (i.e.; antennas, cables) be returned. Failure to return these items will result in a deduction from the total credit due for each missing item.

15.1.2 Returning Units Under Warranty

A Technical Support Engineer must approve the return of Product under ProSoft Technology's Warranty:

- a) A replacement module will be shipped and invoiced. A purchase order will be required.
- b) Credit for a product under warranty will be issued upon receipt of authorized product by ProSoft Technology at designated location referenced on the Return Material Authorization
 - i. If a defect is found and is determined to be customer generated, or if the defect is otherwise not covered by ProSoft Technology's warranty, there will be no credit given. Customer will be contacted and can request module be returned at their expense;
 - ii. If defect is customer generated and is repairable, customer can authorize ProSoft Technology to repair the unit by providing a purchase order for 30% of the current list price plus freight charges, duties and taxes as applicable.

15.1.3 Returning Units Out of Warranty

- a) Customer sends unit in for evaluation to location specified by ProSoft Technology, freight prepaid.
- b) If no defect is found, Customer will be charged the equivalent of \$100 USD, plus freight charges, duties and taxes as applicable. A new purchase order will be required.
- c) If unit is repaired, charge to Customer will be 30% of current list price (USD) plus freight charges, duties and taxes as applicable. A new purchase order will be required or authorization to use the purchase order submitted for evaluation fee.

The following is a list of non-repairable units:

- 3150 - All
- 3750
- 3600 - All
- 3700
- 3170 - All
- 3250
- 1560 - Can be repaired, only if defect is the power supply
- 1550 - Can be repaired, only if defect is the power supply
- 3350
- 3300
- 1500 - All

15.2 LIMITED WARRANTY

This Limited Warranty ("Warranty") governs all sales of hardware, software, and other products (collectively, "Product") manufactured and/or offered for sale by ProSoft Technology, Incorporated (ProSoft), and all related services provided by ProSoft, including maintenance, repair, warranty exchange, and service programs (collectively, "Services"). By purchasing or using the Product or Services, the individual or entity purchasing or using the Product or Services ("Customer") agrees to all of the terms and provisions (collectively, the "Terms") of this Limited Warranty. All sales of software or other intellectual property are, in addition, subject to any license agreement accompanying such software or other intellectual property.

15.2.1 What Is Covered By This Warranty

- a) *Warranty On New Products:* ProSoft warrants, to the original purchaser, that the Product that is the subject of the sale will (1) conform to and perform in accordance with published specifications prepared, approved and issued by ProSoft, and (2) will be free from defects in material or workmanship; provided these warranties only cover Product that is sold as new. This Warranty expires three (3) years from the date of shipment for Product purchased **on or after** January 1st, 2008, or one (1) year from the date of shipment for Product purchased **before** January 1st, 2008 (the "Warranty Period"). If the Customer discovers within the Warranty Period a failure of the Product to conform to specifications, or a defect in material or workmanship of the Product, the Customer must promptly notify ProSoft by fax, email or telephone. In no event may that notification be received by ProSoft later than 39 months from date of original shipment. Within a reasonable time after notification, ProSoft will correct any failure of the Product to conform to specifications or any defect in material or workmanship of the Product, with either new or remanufactured replacement parts. ProSoft reserves the right, and at its sole discretion, may replace unrepairable units with new or remanufactured equipment. All replacement units will be covered under warranty for the 3 year period commencing from the date of original equipment purchase, not the date of shipment of the replacement unit. Such repair, including both parts and labor, will be performed at ProSoft's expense. All warranty service will be performed at service centers designated by ProSoft.
- b) *Warranty On Services:* Materials and labor performed by ProSoft to repair a verified malfunction or defect are warranted in the terms specified above for new Product, provided said warranty will be for the period remaining on the original new equipment warranty or, if the original warranty is no longer in effect, for a period of 90 days from the date of repair.

15.2.2 What Is Not Covered By This Warranty

- a) ProSoft makes no representation or warranty, expressed or implied, that the operation of software purchased from ProSoft will be uninterrupted or error free or that the functions contained in the software will meet or satisfy the purchaser's intended use or requirements; the Customer assumes complete responsibility for decisions made or actions taken based on information obtained using ProSoft software.
- b) This Warranty does not cover the failure of the Product to perform specified functions, or any other non-conformance, defects, losses or damages caused by or attributable to any of the following: (i) shipping; (ii) improper installation or other failure of Customer to adhere to ProSoft's specifications or instructions; (iii) unauthorized repair or maintenance; (iv) attachments, equipment, options, parts, software, or user-created programming (including, but not limited to, programs developed with any IEC 61131-3, "C" or any variant of "C" programming languages) not furnished by ProSoft; (v) use of the Product for purposes other than those for which it was designed; (vi) any other abuse, misapplication, neglect or misuse by the Customer; (vii) accident, improper testing or causes external to the Product such as, but not limited to, exposure to extremes of temperature or humidity, power failure or power surges; or (viii) disasters such as fire, flood, earthquake, wind and lightning.
- c) The information in this Agreement is subject to change without notice. ProSoft shall not be liable for technical or editorial errors or omissions made herein; nor for incidental or consequential damages resulting from the furnishing, performance or use of this material. The user guide included with your original product purchase from ProSoft contains information protected by copyright. No part of the guide may be duplicated or reproduced in any form without prior written consent from ProSoft.

15.2.3 Disclaimer Regarding High Risk Activities

Product manufactured or supplied by ProSoft is not fault tolerant and is not designed, manufactured or intended for use in hazardous environments requiring fail-safe performance including and without limitation: the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly or indirectly to death, personal injury or severe physical or environmental damage (collectively, "high risk activities"). ProSoft specifically disclaims any express or implied warranty of fitness for high risk activities.

15.2.4 Intellectual Property Indemnity

Buyer shall indemnify and hold harmless ProSoft and its employees from and against all liabilities, losses, claims, costs and expenses (including attorney's fees and expenses) related to any claim, investigation, litigation or proceeding (whether or not ProSoft is a party) which arises or is alleged to arise from Buyer's acts or omissions under these Terms or in any way with respect to the Products. Without limiting the foregoing, Buyer (at its own expense) shall indemnify and hold harmless ProSoft and defend or settle any action brought against such Companies to the extent based on a claim that any Product made to Buyer specifications infringed intellectual property rights of another party. ProSoft makes no warranty that the product is or will be delivered free of any person's claiming of patent, trademark, or similar infringement. The Buyer assumes all risks (including the risk of suit) that the product or any use of the product will infringe existing or subsequently issued patents, trademarks, or copyrights.

- a) Any documentation included with Product purchased from ProSoft is protected by copyright and may not be duplicated or reproduced in any form without prior written consent from ProSoft.
- b) ProSoft's technical specifications and documentation that are included with the Product are subject to editing and modification without notice.
- c) Transfer of title shall not operate to convey to Customer any right to make, or have made, any Product supplied by ProSoft.
- d) Customer is granted no right or license to use any software or other intellectual property in any manner or for any purpose not expressly permitted by any license agreement accompanying such software or other intellectual property.
- e) Customer agrees that it shall not, and shall not authorize others to, copy software provided by ProSoft (except as expressly permitted in any license agreement accompanying such software); transfer software to a third party separately from the Product; modify, alter, translate, decode, decompile, disassemble, reverse-engineer or otherwise attempt to derive the source code of the software or create derivative works based on the software; export the software or underlying technology in contravention of applicable US and international export laws and regulations; or use the software other than as authorized in connection with use of Product.
- f) **Additional Restrictions Relating To Software And Other Intellectual Property**

In addition to compliance with the Terms of this Warranty, Customers purchasing software or other intellectual property shall comply with any license agreement accompanying such software or other intellectual property. Failure to do so may void this Warranty with respect to such software and/or other intellectual property.

15.2.5 Disclaimer of all Other Warranties

The Warranty set forth in What Is Covered By This Warranty (page 309) are in lieu of all other warranties, express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

15.2.6 Limitation of Remedies **

In no event will ProSoft or its Dealer be liable for any special, incidental or consequential damages based on breach of warranty, breach of contract, negligence, strict tort or any other legal theory. Damages that ProSoft or its Dealer will not be responsible for include, but are not limited to: Loss of profits; loss of savings or revenue; loss of use of the product or any associated equipment; loss of data; cost of capital; cost of any substitute equipment, facilities, or services; downtime; the claims of third parties including, customers of the Purchaser; and, injury to property.

** Some areas do not allow time limitations on an implied warranty, or allow the exclusion or limitation of incidental or consequential damages. In such areas, the above limitations may not apply. This Warranty gives you specific legal rights, and you may also have other rights which vary from place to place.

15.2.7 Time Limit for Bringing Suit

Any action for breach of warranty must be commenced within 39 months following shipment of the Product.

15.2.8 No Other Warranties

Unless modified in writing and signed by both parties, this Warranty is understood to be the complete and exclusive agreement between the parties, suspending all oral or written prior agreements and all other communications between the parties relating to the subject matter of this Warranty, including statements made by salesperson. No employee of ProSoft or any other party is authorized to make any warranty in addition to those made in this Warranty. The Customer is warned, therefore, to check this Warranty carefully to see that it correctly reflects those terms that are important to the Customer.

15.2.9 Allocation of Risks

This Warranty allocates the risk of product failure between ProSoft and the Customer. This allocation is recognized by both parties and is reflected in the price of the goods. The Customer acknowledges that it has read this Warranty, understands it, and is bound by its Terms.

15.2.10 Controlling Law and Severability

This Warranty shall be governed by and construed in accordance with the laws of the United States and the domestic laws of the State of California, without reference to its conflicts of law provisions. If for any reason a court of competent jurisdiction finds any provisions of this Warranty, or a portion thereof, to be unenforceable, that provision shall be enforced to the maximum extent permissible and the remainder of this Warranty shall remain in full force and effect. Any cause of action with respect to the Product or Services must be instituted in a court of competent jurisdiction in the State of California.

Index

1

- 1) Meter Type = Differential & Product Group = Gas • 224, 227

2

- 2) Meter Type = Differential & Product Group = Liquid • 225, 228

3

- 3) Meter Type = Linear & Product Group = Gas • 225, 228

4

- 4) Meter Type = Linear & Product Group = Liquid • 226
- 4) Meter Type = Pulse & Product Group = Liquid • 228

A

- Abandonment • 135
- Access by Multiple Hosts • 137
- Accessing the Data • 70
- Accumulator Totalizer and Residue • 84
- Accumulator Types • 85
- Accumulators • 83, 230
- Acknowledge Transaction • 134
- Adjust the Input and Output Array Sizes (Optional) • 217
- AFC Modbus Address Space • 70
- AFC Response to an OBA • 162
- AFC56.Meters[].Analysis.High_Precision • 233
- AFC56.Meters[].Analysis.Low_Precision • 233
- AFC56.Meters[].Analysis.Status • 234
- Allocation of Risks • 312
- Analysis Precision • 188
- App Status LED • 256
- Archive Accumulators • 88
- Archive Data Format • 286
- Archive Generation • 93
- Archive Locations • 98
- Archive Monitor • 107
- Archive Options • 97
- Archive Order • 95
- Archive Overview • 92
- Archive Reports • 105
- Archive Types • 94
- Archives • 38, 87, 91
- Archiving and Event Log • 277
- Audit Scan • 264

B

- Base prover volume (65036+) • 50

- Basic Metering According to Meter type • 275
- Basic Principles of Implementation • 125
- Battery Life Advisory • 3
- BBRAM LEDs • 257
- Bidirectional Pipe Prover • 44
- BP Act and P1, P2, or P3 • 256

C

- Cable Connections • 77, 299
- Checking Meter Alarms • 237
- Checksum Alarms • 262
- Communication Parameters • 78
- Compact (short, small volume) Prover • 45
- Completion Phase • 125, 135
- Compressibility Factor F • 278
- Configurable Archive Registers • 280
- Configurable Options • 272
- Configuration Modification Lockout and Seal • 15
- Contacting Technical Support • 305, 307
- Controlling Law and Severability • 312
- ControlLogix Sample Logic Details • 220
- Converting a Project • 22
- Correction Factors According to Product Phase • 275

D

- Data Archiving • 38
- Data Elements • 127
- Data Item Size and Swap Options • 201
- Date Word • 286
- DB9 to RJ45 Adaptor (Cable 14) • 303
- Density Correction • 277
- Density Units • 36
- Description • 171, 172, 174, 176, 185, 187, 191, 193, 195, 198, 200, 204
- Diagnostics and Troubleshooting • 255
- Differential (Orifice) Meter Overview • 34
- Disable Meter • 220
- Disclaimer of all Other Warranties • 311
- Disclaimer Regarding High Risk Activities • 310
- Download Phase • 125, 133
- Downloading the Project to the Module • 24
- Dynamic Context • 126

E

- Editing the Archive Structure • 100
- Enable Meter • 221
- Enable/Disable Status • 220
- Error Codes • 202
- Error Recovery • 136
- Event Id Tag • 115, 116
- Event Log Function • 38
- Event Log Structures • 115
- Event Numbers • 118, 152
- Events • 113, 263
- Event-triggered Archives and Accumulator Resets • 117
- Example • 81, 84
- Example 1 • 292
- Example 1 - Modbus Gateway Function • 246

Example 2 • 292
Example 2 - Read Net Accumulator Totalizer From
Yesterday's Archive (Meters 1 to 4) (assume orifice
meters, gas product) • 247
Expired Events • 138
Extended Archives • 102

F

Fetch Transaction • 133
Fixed and Variable Length Function Blocks • 169
Flow Rate Integration with Gas Product • 179, 290
Flow Rate Integration with Liquid Product • 180, 291
Flow tube inside diameter (mm) (65038+) • 50
Flow tube linear coefficient of thermal expansion
(65032+) • 49
Flow tube modulus of elasticity (65042+) • 50
Flow tube wall thickness (mm) (65040+) • 50
Frequently Asked Questions • 90
Function Block Definition - 0
Null • 171
Function Block Definition - 1
Wall Clock • 172
Function Block Definition - 10
Meter Type Fetch • 186
Function Block Definition - 11
Meter Analysis, 32-bit • 187
Function Block Definition - 12
Site/Meter Signals • 190
Function Block Definition - 14
Meter Archive Fetch • 192
Function Block Definition - 16/17/18/19
Modbus Gateway Read • 193
Function Block Definition - 20, 21
Modbus Gateway Write • 196
Function Block Definition - 24, 25, 26
Modbus Master • 199
Function Block Definition - 28, 29
Disable/Enable Meters • 203
Function Block Definition - 4, 5, 6 & 7
Modbus Pass Through • 173
Function Block Definition - 8
Meter Process Variables • 175
Function Block Definition - 9
Meter Analysis, 16-bit • 184
Function Block Structure • 168, 171, 172, 175, 183,
186, 191, 196, 199, 204

G

Gas • 275
Gas Product Overview • 35
Gas Pulse Measurement • 275
General Features • 37
General Specifications • 270

H

Hard Password • 157
Hardware Specifications • 273
How to Contact Us • 2

I

Import Procedure • 211
Important Installation Instructions • 3
Information for Users of AFC Manager Versions Older
Than 2.01.000 • 284
Initial Requirements • 53, 63
Input (Transaction) Block Array Definition • 165
Input Block Array • 161
Input Block Length and Format Alarm • 166
Input format
line meter pulse count (65020) • 48
master meter pulse count (65021) • 48
Input Function Block ID • 172, 173, 176, 184, 186, 188,
190, 192, 194, 197, 200, 203
Input Registers • 247
Input/Output Blocks for Data Transfer • 160
Input/Output Transactions • 162
Install AFC Manager • 18
Intellectual Property Indemnity • 311
Introduction • 11

L

Ladder Logic Implementation • 26
Latest Prove Results • 64
Layout • 131
Limitation of Remedies ** • 312
LIMITED WARRANTY • 307, 309
Linear (Pulse) Meter Overview • 33
Liquid • 275
Liquid Correction Factor Details • 277
Liquid Product Overview • 36, 275
Loading an Existing project • 21
Log-Download Window (LDW) Allocation • 128
Loggable Events • 140

M

Managing Input Function Block(s) by Manipulating Bit
9 • 170
Managing Output Function Block(s) by manipulating
Bit 8 • 170
Markings • 4
Master Meter • 46
Maximum attempted runs before abort (65014) • 47
Maximum seconds per run (65017) • 47
Measurement Standards • 274
Measurement Units • 38
Measuring Water Diluent • 36
Meter 1
Yesterday's Archive • 251
Meter 2
Yesterday's Archive • 251
Meter 3
Yesterday's Archive • 252
Meter 4
Yesterday's Archive • 252
Meter Alarms • 258
Meter Calculation Results • 227
Meter Channel Functionality • 31

- Meter Channels • 32
- Meter Data Point Events • 140, 143
- Meter factor precision (65028+) • 49
- Meter Previous Prove Summary • 67
- Meter Process Variables • 224
- Meter Profile • 223
- Meter Proving • 41
- Meter Proving Alarms • 54
- Meter Proving Reports • 63
- Meter Scan Time • 37
- Meter Signals • 229
- Meter-relative Data • 72
- Minimum pulses per run (thousands) (65016) • 47
- Modbus Address Examples • 71
- Modbus Address References • 71
- Modbus Addressing Common to Both Primary and Virtual Slaves • 293
- Modbus Communication • 77
- Modbus Database • 69
- Modbus Gateway • 245
- Modbus Gateway Block (uses Transaction Numbers from 17 to 25) • 209
- Modbus Interface • 271
- Modbus Master • 80, 239
- Modbus pass-through • 244
- Modbus Pass-Through • 82
- Modbus Points • 127
- Modbus Port configuration • 296
- Modbus Transaction Sequencing and Constraints • 132
- Module Configuration • 36
- Module Initialization • 29
- Molar Analysis (For Gas Product Only) • 231
- Multiple Meter Accumulators • 37
- MVI (Multi Vendor Interface) Modules • 3
- MVI56-AFC Backplane Communication • 159
- MVI56-AFC Function Block Interface • 167
- MVI56-AFC Module • 14
- MVI56-AFC Module Data Transfer • 160
- MVI56-AFC Sample Logic • 205

N

- Net Accumulator Calculation • 36, 89
- No Other Warranties • 312
- Non-Resettable Accumulators • 85
- Non-Standard Reference Conditions • 276

O

- On-line Communication & Configuration • 271
- Orifice (Differential) Meter with Gas Product • 288
- Orifice (Differential) Meter with Liquid Product • 289
- Orifice (Include V-cone)
 - Uses AGA3 1992 / ISO 5167. • 275
- Orifice Meter with Gas Product • 177, 248
- Orifice Meter with Liquid Product • 178, 249
- Other Considerations • 138
- Output (Transaction) Block Array Definition • 162
- Output Block Array • 160
- Output Block Length • 163

- Output Function Block ID • 171, 172, 173, 175, 184, 186, 187, 190, 192, 193, 196, 199, 203
- Output Function Blocks (OFB) • 164

P

- Period-end Events • 139
- Persistence • 138
- Phases • 125
- Pinouts • 3, 77, 299, 303
- Port Options • 79
- Pre-defined Header • 287
- Pre-defined Overhead • 248
- Pressure Correction Factor, CPL • 277
- Primary Input = Differential Pressure • 34
- Primary Input = Flow Rate • 34
- Primary Input = Pulse Count • 33
- Primary Input = Pulse Frequency • 33
- Primary Slave • 71
- Primary Slave Elements • 128
- Printing the Configuration Report • 21
- Process Block (uses Transaction Numbers from 1 to 16) • 208
- Process Input Scaling • 39
- Process Variable Interface • 37
- Product Batching • 37
- ProSoft Technology® Product Documentation • 2
- Protected Meter Proving Data in the AFC's Input Register Bank • 64
- Prove Calculation Alarms • 56
- Prove-enable Error Code • 61
- Prover Characteristics • 49
- Prover Configuration • 42
- Prover Operation (How to do a Prove) • 57
- Prover Options • 46
- Prover Phase • 59
- Prover Position
 - Ready for Launch • 60
 - Ready for Return • 60
 - Valve Sealed Behind Ball • 60
 - Valve Sealed Behind Ball, Return Leg • 60
- Prover Pressure • 60
- Prover Sequencing • 58
- Prover size units (65018.L) • 49
- Prover Temperature • 60
- Prover Type • 42
- Proving Controls • 58
- Proving Signals • 58
- Pulse
 - Both Gas and Liquid • 275
- Pulse (Linear) Meter with Gas Product • 289
- Pulse (Linear) Meter with Liquid Product • 290
- Pulse Frequency Integration with Gas Product • 180, 291
- Pulse Frequency Integration with Liquid Product • 181, 292
- Pulse interpolation ratio (65030+) • 49
- Pulse Meter with Gas Product • 177, 248
- Pulse Meter with Liquid Product • 179, 249

Q

Quick Start • 17

R

Reference • 269
Reports • 271
Reset from AFC Manager • 86
Reset from Ladder Logic • 87
Reset Resetable Accumulators • 230
Reset Upon Archive Period End or Reset Upon Event • 87
Reset When the Accumulator Rollover Value is Reached • 87
Resetable Accumulators • 85
Resetting Configuration Parameters • 23
Retrieving Extended Archives • 102
Return Material Authorization (RMA) Policies and Conditions • 307
Returned Alarm Codes for Meter Data • 182
Returning Any Product • 307
Returning Units Out of Warranty • 308
Returning Units Under Warranty • 308
RS-232
 Modem Connection (Hardware Handshaking Required) • 300
 Null Modem Connection (Hardware Handshaking) • 300
 Null Modem Connection (No Hardware Handshaking) • 301
RS-232 Application Port(s) • 299
RS-232 Configuration/Debug Port • 299
RS-422 • 302
RS-485 and RS-422 Tip • 302
RS-485 Application Port(s) • 302
Run Counts • 47
Run Input Setup • 47
Runs per prove (65012) • 47
Runs per prove, selected • 47

S

Sample Logic Overview • 206
Sample MVI56-AFC Logic Tasks • 210
Scratchpad • 73
Security (Passwords) • 155
Security and Optimization • 130
Select Stream • 230
Sentinel & Anchor (Transaction Number) • 163
Session Timeout • 136
Set the Processor Time • 236
Setting the Wallclock • 28
Setting up the AFC module for Meter Proving • 51
Setup Phase • 125, 133
SI units: • 278
Site Configuration Items • 128
Site Data Point Events • 140, 142
Site Status • 239
Special Events • 140, 141
Special Notes • 170, 175, 183, 186, 196, 199, 204

Starting a New Project • 20
Starting AFC Manager • 19
Startup Basics and Frequently Asked Questions • 298
Status • 128
Stream Data Point Events • 140, 146, 148
Support, Service & Warranty • 305
Supported Meters • 272
Switch bar linear coefficient of thermal expansion (65034+) • 49
System Requirements • 18

T

Temperature Correction Factor CTL • 277
Temperature Correction Factor, CTL • 276
The Detailed Method • 133
The Event Log • 114
The Log-Download Window (LDW) • 131
The Quick Method • 133
Time Limit for Bringing Suit • 312
Time Word • 286
Timestamp Date and Time Format • 286
To use a densitometer • 36

U

Unidirectional Pipe Prover • 43
Update Notice • 12
Updating the High Precision Molar Analysis • 235
US units: • 278
User LEDs • 256
Using AFC Manager • 20
Using the Sample Add-On Instruction • 211

V

Variation Limit Alarms • 55
Verifying Correct Operation • 25
Virtual Slave • 24, 74
Virtual Slave Example Application • 75
Virtual Slave Precedence Relations • 129

W

Wallclock • 222
Wallclock Block (uses Transaction Number =99) • 209
Warnings • 3
Water Content of Liquids • 276
Wedge Meter Applications • 272, 279
What Is Covered By This Warranty • 309, 311
What Is Not Covered By This Warranty • 310
Write Hourly/Daily Archive • 231

Y

Your Feedback Please • 2