OPTOEMU SENSOR COMMUNICATION GUIDE

Form 1958-120809—August 2012

OPTO 22

43044 Business Park Drive • Temecula • CA 92590-3614
Phone: 800-321-OPTO (6786) or 951-695-3000
Fax: 800-832-OPTO (6786) or 951-695-2712
www.opto22.com

Product Support Services
800-TEK-OPTO (835-6786) or 951-695-3080
Fax: 951-695-3017
Email: support@opto22.com
Web: support.opto22.com

OptoEMU Sensor Communication Guide
Form 1958-120809—August 2012

The information in this manual has been checked carefully and is believed to be accurate; however, Opto 22 assumes no responsibility for possible inaccuracies or omissions. Specifications are subject to change without notice.

Opto 22 warrants all of its products to be free from defects in material or workmanship for 30 months from the manufacturing date code. This warranty is limited to the original cost of the unit only and does not cover installation, labor, or any other contingent costs. Opto 22 I/O modules and solid-state relays with date codes of 1/96 or later are guaranteed for life. This lifetime warranty excludes reed relay, SNAP serial communication modules, SNAP PID modules, and modules that contain mechanical contacts or switches. Opto 22 does not warrant any product, components, or parts not manufactured by Opto 22; for these items, the warranty from the original manufacturer applies. These products include, but are not limited to, OptoTerminal-G70, OptoTerminal-G75, and Sony Ericsson GT-48; see the product data sheet for specific warranty information. Refer to Opto 22 form number 1042 for complete warranty information.

Wired+Wireless controllers and brains and N-TRON wireless access points are licensed under one or more of the following patents: U.S. Patent No(s). 5282222, RE37802, 6963617; Canadian Patent No. 2064975; European Patent No. 1142245; French Patent No. 1142245; British Patent No. 1142245; Japanese Patent No. 2002535925A; German Patent No. 60011224.

Table of Contents

1: Introduction

About the OptoEMU Sensor

OptoEMU Sensor™ Energy Monitoring Units monitor the electrical energy used in your facility and then deliver that data to applications so you can see it, analyze it, and take steps to reduce energy costs. The OptoEMU Sensor gives you the detailed, real-time data you need.

OptoEMU Sensor can provide data to web-based or local software services; to control systems based on Opto 22's PAC Project™ software and Modbus/TCP; to OPC clients; and to standard SQL databases. Incorporating energy data from OptoEMU Sensor in systems you currently use for industrial control or building management can provide a more complete solution than online monitoring alone.

Based on standard Ethernet networking, OptoEMU Sensor is simple to install and use.

- The *OptoEMU Sensor User's Guide* (form 1932) shows you how to install all models of the Sensor, configure the Sensor's inputs and outputs, and set it up for use with online software services and networks. The user's guide also includes maintenance and troubleshooting information.
- Once you have the Sensor installed and configured, this *Communication Guide* shows you how to provide energy data from the Sensor to other applications and systems. This guide assumes that you already know how to use these other systems and how to program in the language of your choice.

Three models are available: OPTOEMU-SNR-3V, OPTOEMU-SNR-DR1, and OPTOEMU-SNR-DR2.

OPTOEMU-SNR-3V—Use the Sensor 3V to connect directly to electrical circuits using current transformers (CTs) with a 0.333 VAC secondary. Direct connections can be made to main electrical panels, subpanels, and individual pieces of equipment such as chillers, boilers, and compressors. The unit can also connect to pulsing devices and Modbus devices (serial or Ethernet). This unit monitors usage and provides energy data but does not control equipment.

OPTOEMU-SNR-DR1 and **OPTOEMU-SNR-DR2**—These small-footprint, less expensive units are ideal for monitoring pulsing meters and Modbus devices of all kinds. You can also use Sensor DRs to respond to a demand response (DR) event by signaling equipment to turn on or off or signaling energy or building management systems to shed loads. Both units connect to pulsing devices or Modbus devices, but not directly to panels or equipment using CTs. The OPTOEMU-SNR-DR1 communicates over both standard wired Ethernet and a wireless LAN. The OPTOEMU-SNR-DR2 offers wired Ethernet but not wireless.

Data In and Data Out: OptoEMU Sensor 3V

OptoEMU Sensor 3V monitors:

- Pulsed outputs from pulsing meter or submeter
- Energy devices that communicate over a serial network using the Modbus protocol
- Ethernet energy devices that communicate using Modbus/TCP
- Direct voltage and current from:
  - Main electrical panel
  - Electrical subpanels
  - Electrical equipment such as chillers, boilers, compressors, elevators, and air handling units (AHUs)

OptoEMU Sensor 3V provides data for:

- Online energy monitoring applications
- PAC Control strategies
- PAC Display HMIs
- OPC 2.0-compliant clients
- Standard databases (SQL Server, MySQL, Microsoft Access)
- Modbus/TCP systems
- Custom systems you've developed, for example using our .NET Toolkit

Data In and Data Out: OptoEMU Sensor DR

OptoEMU Sensor DR monitors:

- Pulsed outputs from pulsing meter or submeter
- Energy devices that communicate over a serial network using the Modbus protocol
- Ethernet energy devices that communicate using Modbus/TCP

OptoEMU Sensor DR provides data for:

- Online energy monitoring applications
- PAC Control strategies
- PAC Display HMIs
- OPC 2.0-compliant clients
- Standard databases (SQL Server, MySQL, Microsoft Access)
- Modbus/TCP systems
- Custom systems you've developed, for example using our .NET Toolkit

In This Guide

This guide includes the following chapters:

- **Chapter 2: OptoEMU Sensor and PAC Project Software**—Use data from the Sensor in PAC Control™ strategies and PAC Display™ HMIs. Share Sensor data with OPC clients and SQL databases.
- **Chapter 3: OptoEMU Sensor and Modbus Systems**—Get data from Modbus devices and share OptoEMU Sensor data with Modbus/TCP systems.
- **Chapter 4: OptoEMU Sensor and Custom Programs**—Develop your own program for OptoEMU Sensor data using .NET.
- **Chapter 5: OptoEMU Sensor and XML**—Create your own data connector to receive and present energy data from the OptoEMU Sensor in XML format.
- **Chapter A: Scratch Pad Tables and Variables**—Reference tables

- **Chapter B: EMU Sensor Tag Database Tables and Variables**—Reference tables

For installing, configuring inputs, setting up an online energy monitoring service, maintaining, and troubleshooting the OptoEMU Sensor, see Opto 22 form 1932, the *OptoEMU Sensor User's Guide*.

For Help

If you have problems using the OptoEMU Sensor and cannot find the help you need in this guide or in the User's Guide, contact Opto 22 Product Support.

Phone:	800-TEK-OPTO (835-6786) 951-695-3080 (Hours are Monday through Friday, 7 a.m. to 5 p.m. Pacific Time)
Fax:	951-695-3017
Email:	support@opto22.com
Opto 22 website:	www.opto22.com

NOTE: Email messages and phone calls to Opto 22 Product Support are grouped together and answered in the order received.

Note for Experienced Opto 22 Customers

Since the OptoEMU Sensor is an appliance designed for a specific purpose, its construction and operation differ from the Opto 22 PACs and I/O you normally use. While some specifications and behaviors are similar, others are quite different. Please read the information in this guide carefully.

2: OptoEMU Sensor and PAC Project Software

Introduction

If you are using Opto 22's PAC Project Software Suite with a SNAP PAC programmable automation controller, you can incorporate energy data from OptoEMU Sensor in your SNAP PAC System.

OptoEMU Sensor data can also be sent to OPC 2.0-compliant clients (for example, Wonderware or ICONICS) or to common databases, such as MySql, Microsoft SQL Server, and Microsoft Access. You can send data to OPC clients or databases whether or not you use the SNAP PAC System.

If you are using an OptoEMU Sensor DR, any of these methods can also be used to control the DR's four signal relay outputs.

About the SNAP PAC System

If you are new to the SNAP PAC System, you'll find it an easy-to-use automation system with flowchart-based control development software, plain-language commands, compact and industrially hardened programmable automation controllers, distributed intelligence, and high-quality input and output (I/O) modules.

Most I/O modules are guaranteed for life. SNAP PAC System products are manufactured and supported in the U.S.A., and Product Support is free.

Using the SNAP PAC System, you can monitor and control equipment, devices, and processes. For more information, visit www.opto22.com/lp/snappacsystem.aspx.

In This Chapter

This chapter explains how to communicate with the Sensor in the following ways:

Communicate with...	Required Software	Method	See
PAC Control strategy	PAC Project Basic (free download) or PAC Project Professional (purchase)	Sensor's Scratch Pad (peer-to-peer)	page 6
PAC Display HMI	PAC Project Basic (free download) or PAC Project Professional (purchase)	Public .idb file	page 7
OPC 2.0-compliant clients, including third-party HMIs like Wonderware® and ICONICS	OptoOPCServer (as separate purchase or as part of PAC Project Professional)	Public .idb file	page 8

Communicate with...	Required Software	Method	See
Common databases (MySql, Microsoft SQL Server, Access)	OptoDataLink (as separate purchase or as part of PAC Project Professional)	Public .idb file	page 12

For Current Opto 22 Customers

OptoEMU Sensor is an appliance—a box built for a specific purpose—and although wiring and some features may seem familiar to Opto 22 customers, it is not a collection of standard Opto parts. That's why it must not be removed from its enclosure, and that's why, even though it contains a PAC Control strategy, you cannot open or change the strategy.

Instead, we've provided two ways to communicate with the Sensor using the SNAP PAC System:

- The Sensor's **Scratch Pad**, which other controllers can read and write to, peer-to-peer, using I/O Unit—Scratch Pad commands in PAC Control (see below)
- A **public .idb file**, which contains energy data tables and variables for use with PAC Display, OptoOPCServer, and OptoDataLink (see page 7)

Using the Sensor's Scratch Pad with PAC Control Strategies

This section assumes that you are already familiar with PAC Control. (If you are not, see the *PAC Control User's Guide*, form 1700, and the *PAC Control Command Reference*, form 1701.)

The OptoEMU Sensor continually stores energy data in its Scratch Pad, so you can read the most recent data at any time using commands in PAC Control Basic or Professional. Then you can use the data in your PAC Control strategy. On a Sensor DR, the control strategy can also write to the DR's outputs using the Scratch Pad.

Except for outputs on a Sensor DR, these OptoEMU Scratch Pad addresses are for reading only; if you write to them, nothing changes, and the values you wrote are overwritten on the next data update.

Here's how to communicate with an OptoEMU Sensor in your strategy:

1. Add the OptoEMU Sensor as an I/O unit in your strategy. Configure it as a Generic Memory Map Device with the IP address the Sensor uses to communicate with PAC Control.

   The IP address should be written on a label on the Sensor's box.

   For instructions to add an I/O unit, see the *PAC Control User's Guide* (form 1700).

2. To read or write, use the I/O Unit—Scratch Pad commands in PAC Control together with the Scratch Pad indexes listed in Appendix A: Scratch Pad Tables and Variables. Remember that data from the Sensor's inputs can be read only; you cannot write to Scratch Pad input indexes.

3. Because these are I/O unit commands, check all return values and errors to make sure the command was successful. (If not successful, the Sensor may be disabled in the strategy; the Sensor will keep running and collecting data, but the data won't be read into your strategy.)

   For more information on using I/O Unit—Scratch Pad commands, see Chapter 10 in the *PAC Control User's Guide* and the individual commands in PAC Control Command Help or the *PAC Control Command Reference* (Form 1701). Both guides can be downloaded from our website, www.opto22.com; the easiest way to find one is to search on its form number.

Communicating with PAC Display, OPC Clients, and Databases

Included with the Sensor is a public file named OptoEMU Sensor.idb, which is a database of accessible tags for the OptoEMU Sensor. This file contains data tables and variables from the Sensor that you can use in PAC Display, OptoOPCServer, and OptoDataLink. The OptoEMU Sensor.idb file is located on the CD that came with the OptoEMU Sensor and can also be downloaded from our website at http://op22.co/optoemu-snr-3v (click the Downloads tab and locate the file in the list).

In order to reduce data complexity and improve system performance, we've kept most data values in tables rather than treating them as individual named variables.

Except for outputs on a Sensor DR, the tables and variables in the OptoEMU Sensor.idb file are for reading only; if you try to change them, the changes do not affect the Sensor's strategy. Some configuration values are also available through this file, but they are also read-only; all configuration must be done through the OptoEMU Sensor Manager software (see the *OptoEMU Sensor User's Guide*, form 1932).

See Appendix B: EMU Sensor Tag Database Tables and Variables for all tags.

PAC Display HMI

These instructions assume you are already familiar with PAC Display. Use either PAC Display Basic or PAC Display Professional.

1. Locate the OptoEMU Sensor.idb file on the CD that came with the OptoEMU Sensor, or download it from our website at http://op22.co/optoemu-snr-3v (click the Downloads tab and locate the file in the list). Copy the .idb file to a permanent location on your PC.

   While you're in the Sensor's Download tab, you can also download the sample PAC Display project.

2. Open your project (or the sample project) in PAC Display Configurator.
3. Configure the OptoEMU Sensor as a control engine: choose Configure > Control Engine and click Add. Navigate to the OptoEMU Sensor.idb file, and then add the OptoEMU Sensor as a control engine.

   NOTE: If you are also going to use OptoEMU Sensor data with OptoOPCServer or OptoDataLink, for the control engine name, you must use `EMU_Sensor_1`

   Additional Sensors can use these control engine names: `EMU_Sensor_2`, `EMU_Sensor_3`, *and* `EMU_Sensor_4`

4. To run the sample PAC Display project, choose File > Save Project and Load Runtime. Click OK if you get a message about regenerating IO Scanner tag names.
5. To add Sensor data to your own project or to change the sample project, use the Sensor's tags, listed in Appendix B: EMU Sensor Tag Database Tables and Variables, in the HMI as usual.

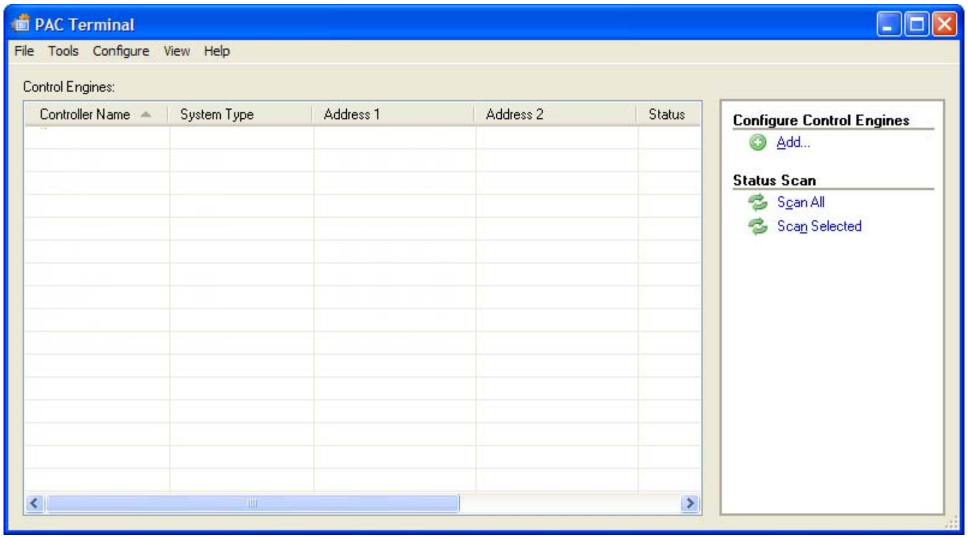
For help, see form 1702, the *PAC Display User's Guide*, available on our website.

OPC Clients

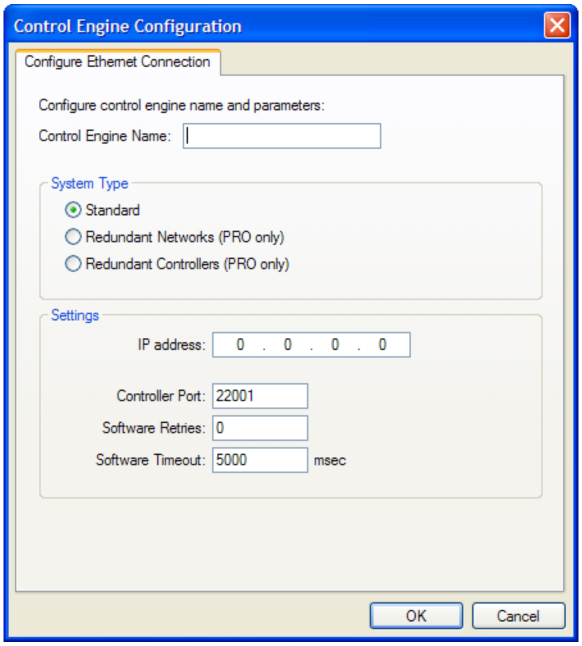
You can use OptoOPCServer software for communication between OptoEMU Sensor and OPC 2.0-compliant clients, whether or not you use the SNAP PAC System. You must have PAC Project Professional (or purchase OptoOPCServer separately).

Before you begin: If you purchased OptoOPCServer separately (not as part of PAC Project Professional), when you install OptoOPCServer, make sure you check the box to install PAC Utilities as well.

1. Locate the OptoEMU Sensor.idb file on the CD that came with the OptoEMU Sensor, or download it from our website at http://op22.co/optoemu-snr-3v (click the Downloads tab and locate the file in the list). Copy the .idb file to a permanent location on your PC.
2. If you have already configured the OptoEMU Sensor as a control engine, skip to step 3. If you have not, do the following:
   a. Choose Start > Opto 22 > PAC Project > Tools > OptoTerminal.



b. In the upper right, click Add to open the Control Engine Configuration dialog box:



Configure control engine name and parameters:

Control Engine Name:

● Standard
○ Redundant Networks (PRO only)
○ Redundant Controllers (PRO only)

IP address: 0 . 0 . 0 . 0
Controller Port: 22001
Software Retries: 0
Software Timeout: 5000 msec

c. In the Control Engine Name field, type `EMU_Sensor_1`

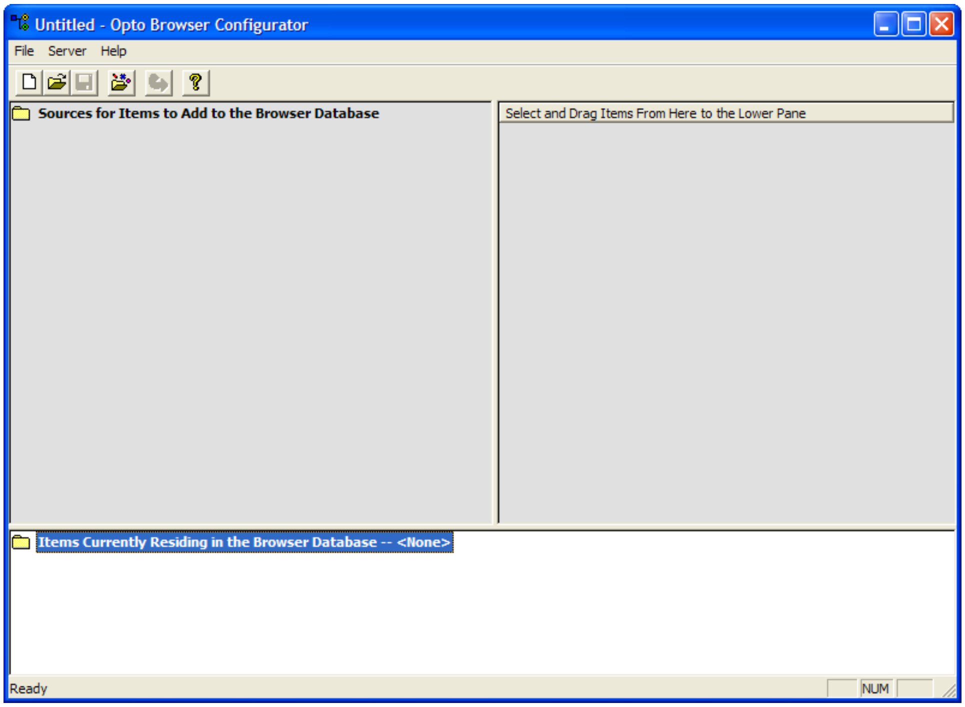
IMPORTANT: You must use this exact name.

d. In the IP address field, enter the OptoEMU Sensor's IP address that is used to communicate with OptoOPCServer software. Leave all other values as shown above. Click OK.

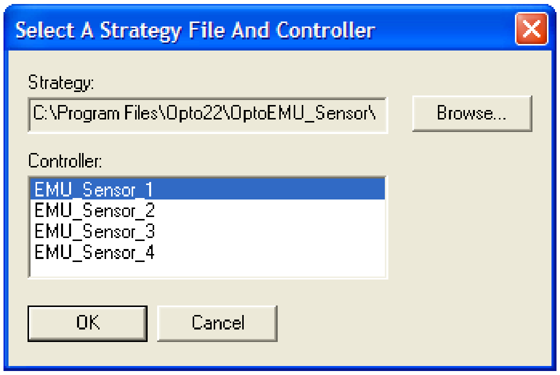
The dialog box closes and the Sensor appears in the Control Engines list.

NOTE: If you have additional OptoEMU Sensors, you can configure them using the following names:
EMU_Sensor_2
EMU_Sensor_3
EMU_Sensor_4

3. Choose Start > Opto 22 > PAC Project > OptoOPCServer > Browser Configurator.


Untitled - Opto Browser Configurator
File
Server
Help
Sources for Items to Add to the Browser Database
Select and Drag Items From Here to the Lower Pane
Items Currently Residing in the Browser Database -- <None>
Ready
NUM

4. Click the Open Browser Items File button (or choose File > Browser Items > Open Browser Items File). Navigate to the OptoEMU Sensor.idb file. Highlight the filename and click Open.

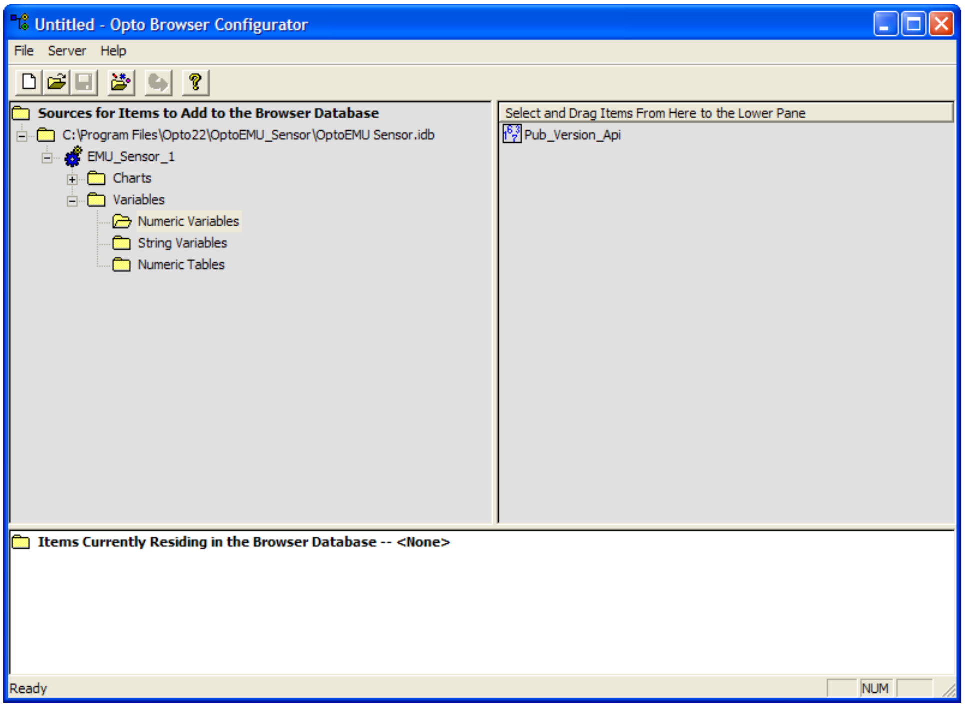


Strategy: C:\Program Files\Opto22\OptoEMU_Sensor\

Controller:
- EMU_Sensor_1
- EMU_Sensor_2
- EMU_Sensor_3
- EMU_Sensor_4

5. Make sure the correct control engine name (for example, EMU_Sensor_1) is highlighted in the list. Click OK.

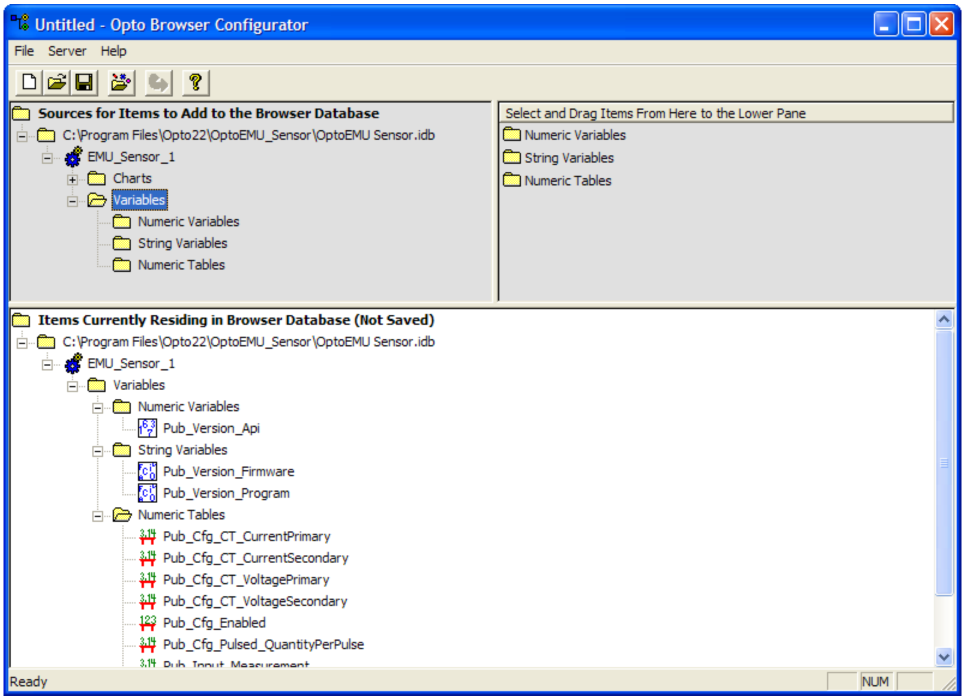
   Contents of the .idb file appear in the upper-left pane of the window. Notice that the Numeric variables folder is open, and the variables in it (just one, in this case) appear in the upper-right pane.


Untitled - Opto Browser Configurator
File Server Help
Sources for Items to Add to the Browser Database
C:\Program Files\Opto22\OptoEMU_Sensor\OptoEMU Sensor.idb
EMU_Sensor_1
Charts
Variables
Numeric Variables
String Variables
Numeric Tables
Select and Drag Items From Here to the Lower Pane
Pub_Version_Api
Items Currently Residing in the Browser Database -- <None>
Ready
NUM

6. Click the Variables folder to highlight it.

   Its contents, the three folders within it, appear in the upper-right pane.

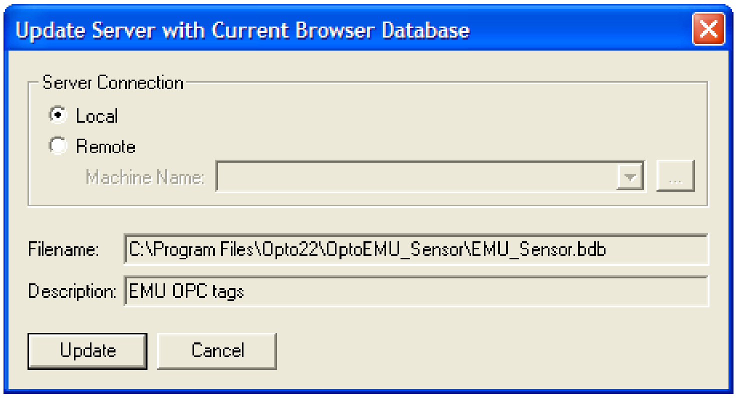
7. In the upper-right pane, hold down the SHIFT or CTRL key and highlight all three folders. Drag them into the bottom pane of the window. In the bottom pane, open all three folders to see the data items inside:


Untitled - Opto Browser Configurator
File Server Help
Sources for Items to Add to the Browser Database
C:\Program Files\Opto22\OptoEMU_Sensor\OptoEMU Sensor.idb
EMU_Sensor_1
Charts
Variables
Numeric Variables
String Variables
Numeric Tables
Select and Drag Items From Here to the Lower Pane
Numeric Variables
String Variables
Numeric Tables
Items Currently Residing in Browser Database (Not Saved)
C:\Program Files\Opto22\OptoEMU_Sensor\OptoEMU Sensor.idb
EMU_Sensor_1
Variables
Numeric Variables
Pub_Version_Api
String Variables
Pub_Version_Firmware
Pub_Version_Program
Numeric Tables
Pub_Cfg_CT_CurrentPrimary
Pub_Cfg_CT_CurrentSecondary
Pub_Cfg_CT_VoltagePrimary
Pub_Cfg_CT_VoltageSecondary
Pub_Cfg_Enabled
Pub_Cfg_Pulsed_QuantityPerPulse
Pub_Input_Measurement
Ready
NUM

These are all the OptoEMU Sensor data items available; details about them are in Appendix B: EMU Sensor Tag Database Tables and Variables.

NOTE: Do not use items within the Charts folder; they could cause problems within the OptoEMU.

8. Click the Save button and save the browser file in a permanent place on your computer. When prompted, enter a Description for the file.
9. From the Server menu, choose Update Server With Current Browser Database.



Update Server with Current Browser Database

Server Connection

● Local

○ Remote

Machine Name:

Filename: C:\Program Files\Opto22\OptoEMU_Sensor\EMU_Sensor.bdb

Description: EMU OPC tags

10. Choose Local unless OptoOPCServer resides on a different computer on the network. If you choose Remote, enter the name of the machine running OptoOPCServer.

    The machine name is the network identification of the computer running OptoOPCServer. You can find its machine name by viewing the computer's System Properties.
11. Click Update.

    The tags are now ready to be used by the OPC client.

    For more information on using OptoOPCServer, see form 1439, the *OptoOPCServer User's Guide*, which came with your purchase of the software. (To open the PDF file for the guide, choose Start > Opto 22 > PAC Project > Manuals > OptoOPCServer User's Guide.)

Databases or Text Files

You can use OptoDataLink software to exchange bi-directional data between the OptoEMU Sensor and a common database such as MySql, Microsoft SQL Server, or Microsoft Access. You can also use OptoDataLink to send data from the Sensor to a text file. You can use OptoDataLink either with a SNAP PAC System or on its own. You must have PAC Project Professional (or purchase OptoDataLink separately).

Setting Up Your Database

1. Using the tables in Appendix B: EMU Sensor Tag Database Tables and Variables, determine the data you need.
2. In your database, create a table to hold the data. In the table, set up one column for each data item.

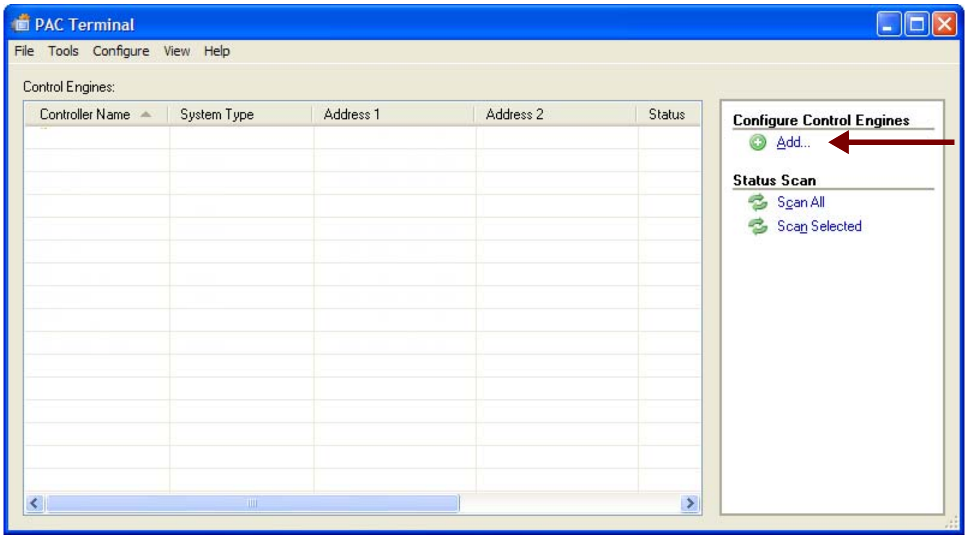
   For example, suppose you want to read the following three pieces of data. You would create a table named, for example, Utilities. You would set up three columns in the database table, one for each item. You can name the columns whatever you wish; you will use those names later in OptoDataLink.You might set up the columns as shown:

Data	Column Name
Live measurement for CT Total True Power/Energy	Live Energy
Accumulated quantity for CT Total True Power/Energy	Total Energy
Live measurement for CT Total Power Factor	Power Factor

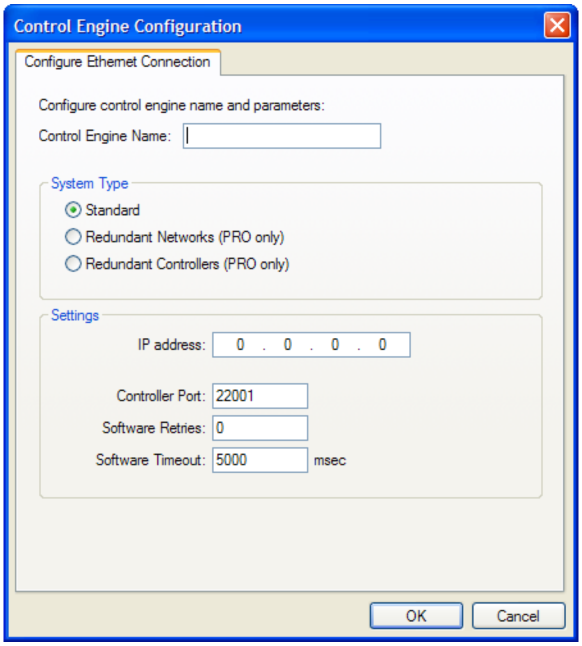
Setting Up OptoDataLink

Before you begin: If you purchased OptoDataLink separately (not as part of PAC Project Professional), when you install OptoDataLink, make sure you check the box to install PAC Utilities as well.

1. Locate the OptoEMU Sensor.idb file on the CD that came with the OptoEMU Sensor, or download it from our website at http://op22.co/optoemu-snr-3v (click the Downloads tab and locate the file in the list). Copy the .idb file to a permanent location on your PC.
2. If you have already configured the OptoEMU Sensor as a control engine, skip to step 3. If you have not, do the following:
   a. Choose Start > Opto 22 > PAC Project > Tools > OptoTerminal.



b. In the upper right, click Add to open the Control Engine Configuration dialog box:



Control Engine Configuration

Configure Ethernet Connection

Configure control engine name and parameters:

Control Engine Name:

System Type

● Standard

○ Redundant Networks (PRO only)

○ Redundant Controllers (PRO only)

Settings

IP address: 0 . 0 . 0 . 0

Controller Port: 22001

Software Retries: 0

Software Timeout: 5000 msec

c. In the Control Engine Name field, type `EMU_Sensor_1`

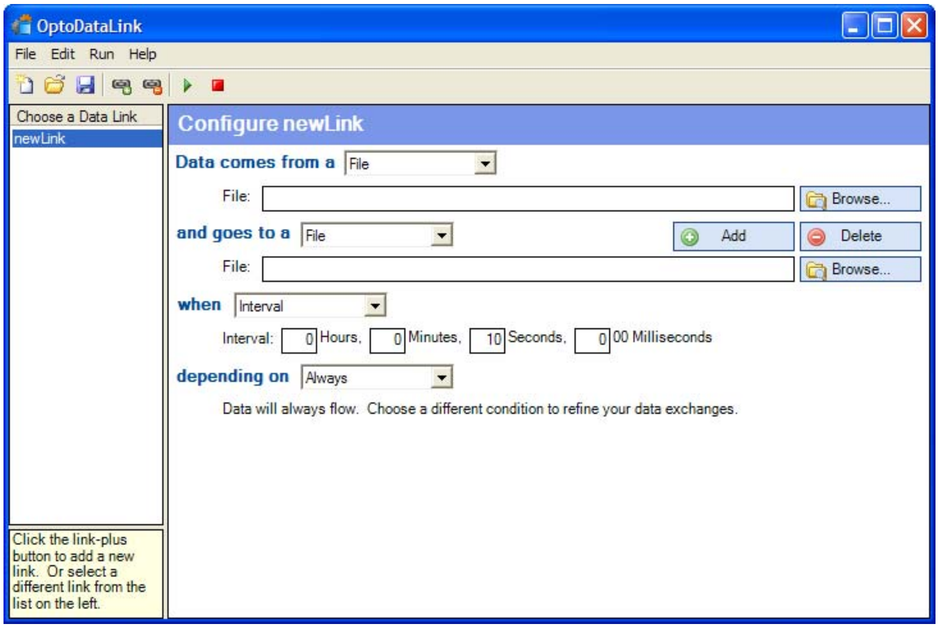
IMPORTANT: You must use this exact name.

d. In the IP address field, enter the OptoEMU Sensor's IP address that is used to communicate with OptoDataLink. Leave all other values as shown above. Click OK.

The dialog box closes and the Sensor appears in the Control Engines list.

NOTE: If you have additional OptoEMU Sensors, you can configure them using the following names:
EMU_Sensor_2
EMU_Sensor_3
EMU_Sensor_4

3. Choose Start > Opto 22 > PAC Project > OptoDataLink.



Choose a Data Link

newLink

Click the link-plus button to add a new link. Or select a different link from the list on the left.

Configure newLink

Data comes from a: File

File:

and goes to a: File

File:

when: Interval

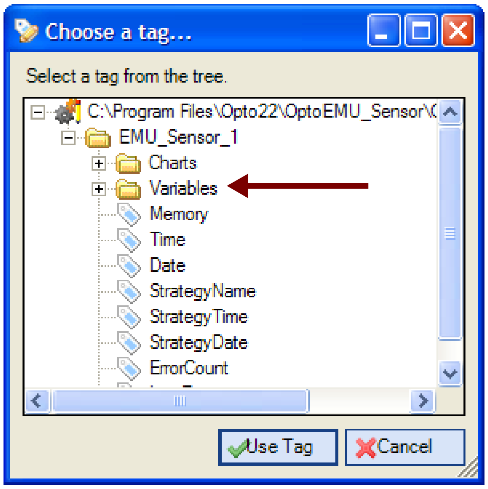
Interval: 0 Hours, 0 Minutes, 10 Seconds, 0 00 Milliseconds

depending on: Always

Data will always flow. Choose a different condition to refine your data exchanges.

4. From the “Data comes from a” dropdown list, choose Opto 22 Device.

5. Click Add Tag and then click Edit. Navigate to the OptoEMU Sensor.idb file you saved on your computer, highlight it, and click Open.



Select a tag from the tree.

C:\Program Files\Opto22\OptoEMU_Sensor\(
EMU_Sensor_1
Charts
Variables
Memory
Time
Date
StrategyName
StrategyTime
StrategyDate
ErrorCount

Several tags are shown at the top level (Memory, Time, and so on), but these are status for the Sensor, not the Sensor data.

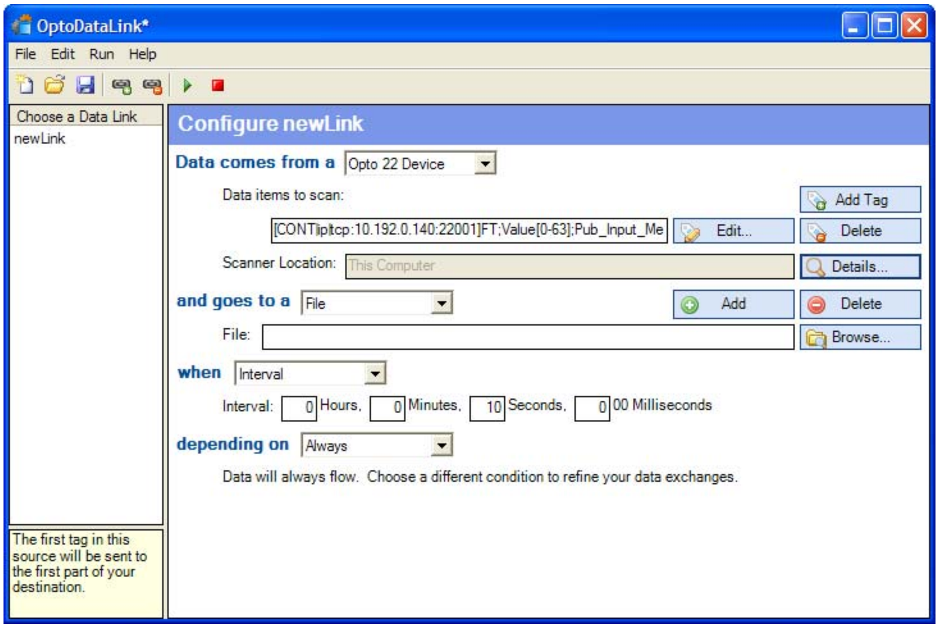
The tags you want to use are in the Variables folder.

6. Click the plus sign to open the Variables folder, then open the Numeric Tables folder.

Pub_Input_Measurement, Pub_Input_TotalQuantity, and Pub_Input_TotalQuantity64 are the tables that contain OptoEMU Sensor data. Several other tags contain read-only configuration data. See Appendix B: EMU Sensor Tag Database Tables and Variables for a complete list of these tables and their elements.

7. Click to highlight a tag you want to use, for example, Pub_Input_Measurement. Click Use Tag.

The table tag appears in the OptoDataLink main window.



Choose a Data Link

newLink

The first tag in this source will be sent to the first part of your destination.

Configure newLink

Data comes from a: Opto 22 Device

Data items to scan:

[CONT|ip|tcp:10.192.0.140:22001]FT;Value[0-63];Pub_Input_Me

Scanner Location: This Computer

and goes to a: File

File:

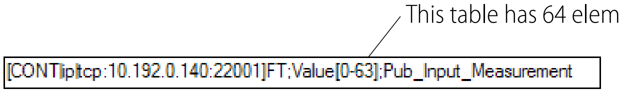
when: Interval

Interval: 0 Hours, 0 Minutes, 10 Seconds, 0 00 Milliseconds

depending on: Always

Data will always flow. Choose a different condition to refine your data exchanges.

The tag specifies the indexes for all of the elements it contains. For example, this table tag has 64 elements (0-63):



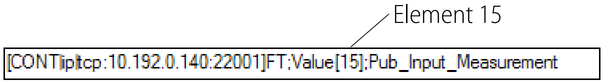
This table has 64 elements with indexes 0-63

[CONT|ip|tcp:10.192.0.140:22001]FT;Value[0-63];Pub_Input_Measurement

But in OptoDataLink you must enter a separate tag for each table element you want to use.

8. Edit the tag to specify the index for the first element you want to use.

For example, looking at the available Sensor data in Appendix B, you see that the CT Total True Power/Energy is element [15] in the Pub_Input_Measurement table. Since this is one piece of data you want, you would edit the tag to indicate only element 15:



Element 15

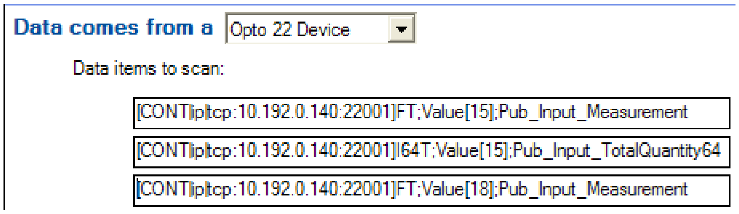
[CONT|ip|tcp:10.192.0.140:22001]FT;Value[15];Pub_Input_Measurement

9. For each additional element you want to use, click Add Tag, copy and paste the tag information to the new text box, and edit the index for the specific element.

In our example, you want the OptoEMU Sensor to send the following data to your database. You consult Appendix B and decide to use the more accurate *Pub_Input_TotalQuantity64* values (see "Data Key" on page 41). You locate the three tags you need:

- Live measurement for CT Total True Power/Energy—Pub_Input_Measurement[15]
- Accumulated quantity for CT Total True Power/Energy—Pub_Input_TotalQuantity64[15]
- Live measurement for CT Total Power Factor— Pub_Input_Measurement[18]

Your tags in OptoDataLink would look like this:



Data comes from a: Opto 22 Device

Data items to scan:

```
[CONT|ip|tcp:10.192.0.140:22001]FT;Value[15];Pub_Input_Measurement
[CONT|ip|tcp:10.192.0.140:22001]I64T;Value[15];Pub_Input_TotalQuantity64
[CONT|ip|tcp:10.192.0.140:22001]FT;Value[18];Pub_Input_Measurement
```

10. When you have finished adding tags, from the "and goes to a" dropdown list, choose Database. (For a text file, choose File, browse to the filename and location, and skip to step 16.)

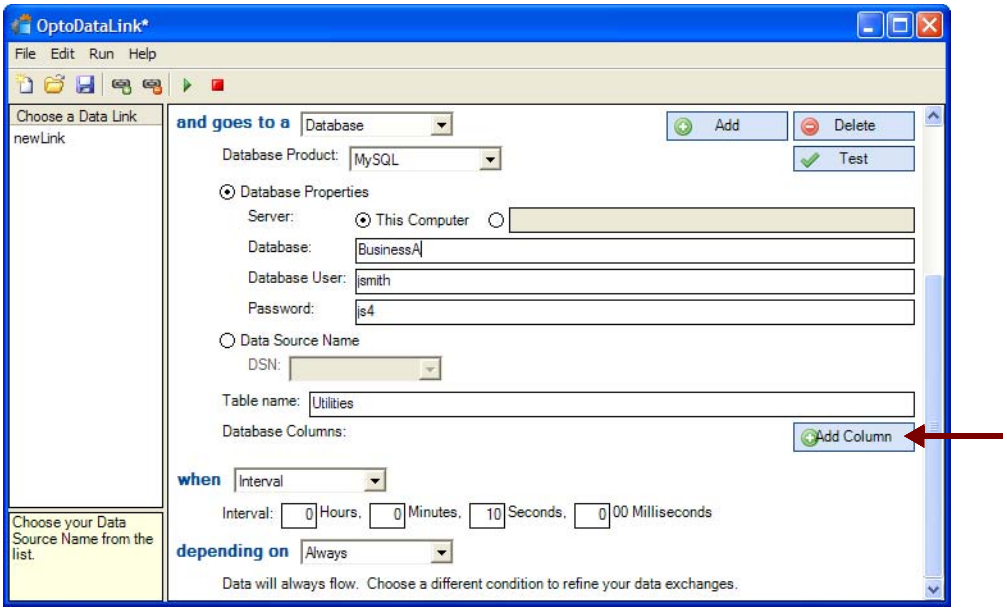
11. From the Database Product dropdown list, choose the database you are using.

12. If you have a Data Source Name, skip down and click that field. Choose the DSN from the dropdown list and enter the table name and columns, if applicable.

13. If you don't have a Data Source Name, enter the database server location, username, and password (if applicable).

Once you have entered database information, you can click the Test button to see if OptoDataLink can connect to the database.

14. In the Table Name field, type the name of the table you created in your database to hold the OptoEMU Sensor data.



and goes to a: Database

Database Product: MySQL

● Database Properties

Server: ● This Computer ○

Database: BusinessA

Database User: jsmith

Password: js4

○ Data Source Name

DSN:

Table name: Utilities

Database Columns:

when: Interval

Interval: 0 Hours, 0 Minutes, 10 Seconds, 0 00 Milliseconds

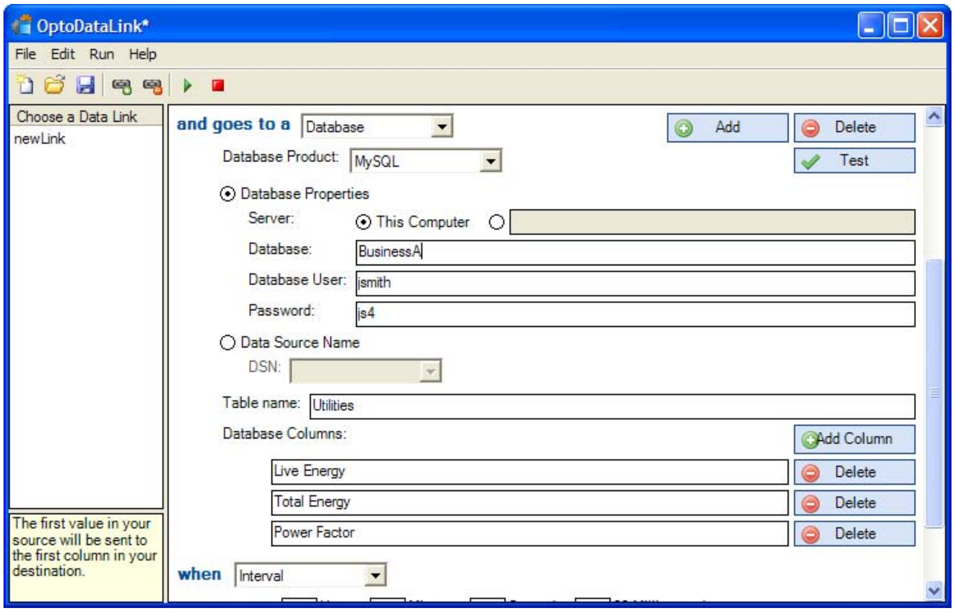
depending on: Always

Data will always flow. Choose a different condition to refine your data exchanges.

Choose your Data Source Name from the list.

15. Click Add Column and enter the name of a database table column in the blank field. Do the same thing for each column in the destination database table.

In our example, you created a database table named Utilities with three columns to receive data (see "Setting Up Your Database" on page 12): Live Energy, Total Energy, and Power Factor. So these are the names you would use for the table name and the database column names, as shown below.



newLink

and goes to a Database

Database Product: MySQL

● Database Properties

Server: ● This Computer ○

Database: BusinessA

Database User: jsmith

Password: js4

○ Data Source Name

DSN:

Table name: Utilities

Database Columns:

Live Energy

Total Energy

Power Factor

The first value in your source will be sent to the first column in your destination.

when Interval

16. Scroll down to the "when" dropdown list. Choose to send the data from the Sensor at a specific interval or at a specific time of day. Complete the necessary information for the method you choose.

Interval: Minutes and seconds have a maximum of 59; milliseconds are in increments of 100. Minimum interval is 100 milliseconds.

Time of Day: Noon is 12:00:00 P.M. Midnight is 12:00:00 A.M.

17. Scroll down to the "depending on" dropdown list. Choose the condition under which you want data to be sent from the OptoEMU Sensor to the database and complete the necessary fields. (Do not choose Opto 22 Device, as you will not be able to set a flag in the Sensor.)

18. Click the Run button ▶ . Give the OptoDataLink file a name and save it in the location you want on your PC.

OptoDataLink begins running and starts sending OptoEMU Sensor data to the database (or text file).

For more information on using OptoDataLink, see form 1705, the *OptoDataLink User's Guide*, which came with your purchase of the software. (To open the PDF file for the guide, choose Start > Opto 22 > PAC Project > Manuals > OptoDataLink User's Guide.)

3: OptoEMU Sensor and Modbus Systems

Introduction

Modbus protocol support is built into the OptoEMU Sensor, both for inputs and outputs:

- The OptoEMU Sensor can get data from Modbus devices over serial (RTU or ASCII) and Ethernet. The Sensor acts as a Modbus master for inputs.
- The Sensor can also send data to Modbus/TCP systems over Ethernet. The Sensor acts as a Modbus slave for output.

This ability can be especially useful for energy control and management. This chapter shows you where energy data from Modbus inputs is stored in the Sensor and how Modbus/TCP devices or systems can read or write to the Sensor.

This chapter assumes that you already know how to use Modbus and Modbus/TCP.

Modbus Input (Serial and Ethernet)

Modbus inputs, both serial and Ethernet, bring data into the EMU. That data—up to 64 pieces of Modbus data—can be read through the public interface or Scratch Pad just like data the EMU gets from pulsing meters and electrical panels.

This data is not limited to energy usage. For example, suppose you need to know the temperature of a server room or the pressure and level of material in a tank. You could use an Opto 22 SNAP PAC EB brain and I/O to acquire that data and deliver it to the EMU via Modbus/TCP.

Modbus Output (Ethernet only)

In addition to acquiring data from Modbus devices, the OptoEMU Sensor can communicate with a Modbus/TCP system over Ethernet. The Modbus/TCP system can read data from the Sensor and write to a Sensor DR's signal relay outputs.

In this way you can easily integrate data from the EMU into a Modbus/TCP building or control system you already use. You can automatically control equipment and processes so your system can be responsive to energy use and conditions that affect it. For example, the system could alter a process when energy use exceeds a threshold or a temperature rises above a specified value.

Modbus Communication Basics

The OptoEMU Sensor continually stores energy data from all inputs, including Modbus inputs, in the Scratch Pad area of its memory map. A Modbus/TCP system can read the most recent data at any time and then use that data.

For a Sensor DR, a Modbus/TCP system can write to the Scratch Pad to control the DR's signal relay outputs.

OptoEMU Sensor uses a Modbus packet inside TCP/IP. The Modbus checksum is not used; instead, the Ethernet TCP/IP link layer checksum guarantees data. The size of the packet is limited to 256 bytes.

Communicating with OptoEMU Sensor using Modbus/TCP requires three basic steps: connect, read, and disconnect.

The Modbus master needs to support Class 0 commands, function code 03 (Read holding registers), and use of the Unit ID. When opening a TCP/IP connection to the system, use port 502. Up to two Modbus masters can be simultaneously handled by the Sensor.

Read or write values by using the Unit ID and Register numbers in the tables on the following pages. Except for the signal relay outputs on a Sensor DR, these OptoEMU Scratch Pad addresses are for reading only; if you write to them, nothing changes, and the values you wrote are overwritten on the next data update.

Data Update Frequency

CT and Pulsed Input measurements are updated when a new measurement is made:

- For CTs, once per second
- For Pulsed Inputs, the next pulse after the minimum sample period, which is set when the input is configured

Configuration values are written every 5 seconds. Modbus inputs are read sequentially, so frequency depends on network conditions and number of inputs.

Data Key

Data may indicate live measurement or accumulated quantity. Live measurement (usually expressed as power) is somewhat like your car's speedometer, which shows you your current rate of speed in miles per hour. Accumulated quantity (expressed as energy, which is power over time) is like the car's odometer, which shows the total number of miles driven since the car was first put on the road.

For pulse devices, you can get the following data:

- Power in kW (live measurement)
- Energy in kWh (accumulated quantity)

For Modbus inputs (serial or Ethernet):

Each Live Measurement is the last read value in one of the following formats:

- Live Measurement (floats)
- Live Measurement (UINT16, INT16, INT32)
- Live Measurement (UINT32)—To avoid sign-extension problems, Int64 values are used for the UINT32 type.

To determine whether the value is fresh enough to be used, also read the quality fields for the corresponding Live Measurement:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads.
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read sequentially If one or more inputs is currently offline, it can affect the freshness of the points that are online.

For direct (CT) connections to panels and devices, you can get the following data:

For each phase of a three-phase device, or for each single-phase device:

- Voltage (live measurement) in volts
- Current (live measurement) in amps
- True Power (RMS voltage multiplied by RMS current and cosine of the phase angle)—Power in kW and energy in kWh
- Apparent Power (RMS voltage multiplied by RMS current; greater than or equal to True Power)—Power in kVA and energy in kVAh
- Power Factor (true power divided by apparent power)—a ratio

Totals for a three-phase device:

- Total True Power (true power for all three phases added together)—Power in kW and energy in kWh
- Total Abs True Power (true power for all three phases added together as positive values)—Power in kW and energy in kWh
- Total Apparent Power (apparent power for all three phases added together)—Power in kVA and energy in kVAh
- Total Power Factor (total true power divided by total apparent power)—a ratio

The data you need depends on what you want to track, which usually depends on how your utility company structures billing. Many utilities bill on the basis of True Power usage, but some bill based on kVA (Apparent Power). In addition, the utility may penalize facilities with low power factor, since they draw more current. Check your utility bill for details, and contact your local utility company with any questions.

Note on Energy Data

For Energy, two sets of data are provided: 32-bit floats and 64-bit integers. While both represent the total accumulated quantity, 64-bit integer values are more accurate. Choose the one that works better for you:

- *32-bit floats* are a convenient representation of the total accumulated quantity. However, because 32-bit floats do not have a high level of precision, they are not well suited for accumulating. 32-bit floats take up two registers.
- *64-bit integers* represent the same value, but as an integer multiplied by 1,000,000. This value is less convenient to use but has the precision required for accumulating values. It is effectively milliwatt hours (mWh) or millivoltamp hours (mVAh). 64-bit integers take up four registers. To convert mWh or mVAh to kWh or kVAh, convert the 64-bit integer value to a double precision float and divide by 1,000,000.

Data Tables

Pulse Inputs

Four pulse inputs are available on a Sensor 3V. Two pulse inputs (Pulse A and Pulse B) are available on a Sensor DR.

Input	Power (32-bit Floats)			Energy (32-bit Floats)			Energy (64-bit Integers)		
	Units	Unit ID	Register	Units	Unit ID	Register	Units	Unit ID	Register
Pulse A	kW	110	4135	kWh	110	4335	mWh	113	77
Pulse B	kW	110	4137	kWh	110	4337	mWh	113	81
Pulse C	kW	110	4139	kWh	110	4339	mWh	113	85
Pulse D	kW	110	4141	kWh	110	4341	mWh	113	89

Modbus Inputs

Up to 64 pieces of Modbus data can be inputs. They can be virtually any kind of data, from one or more devices, connected over serial and/or over Ethernet.

Input	Item*	Live Measurement (Floats)		Live Measurement (UINT16, INT16, INT32)		Live Measurement (UINT32) **	
		Unit ID	Register	Unit ID	Register	Unit ID	Register
Remote 0	Measurement	110	5497	110	3449	113	2801
	Quality (Status)	110	5499	110	3451	113	2805
	Quality (Freshness)	110	5501	110	3453	113	2809
Remote 1	Measurement	110	5503	110	3455	113	2813
	Quality (Status)	110	5505	110	3457	113	2817
	Quality (Freshness)	110	5507	110	3459	113	2821
Remote *N*	Measurement	110	5497 + (*N* * *6*)	110	3449 + (*N* * *6*)	113	2801 + (*N* * *12*)
	Quality (Status)	110	5499 + (*N* * *6*)	110	3451 + (*N* * *6*)	113	2805 + (*N* * *12*)
	Quality (Freshness)	110	5501 + (*N* * *6*)	110	3453 + (*N* * *6*)	113	2809 + (*N* * *12*)
Remote 63	Measurement	110	5875	110	3827	113	3557
	Quality (Status)	110	5877	110	3829	113	3561
	Quality (Freshness)	110	5879	110	3831	113	3565

* Measurement = Last read value. Use the quality fields to determine if the value is fresh enough to be used:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read in a round-robin fashion. If one or more inputs is currently offline, it can affect the freshness of the points that are online.

** Int64 values are used for the UINT32 type to avoid sign-extension problems. It's possible to read only the lower 32 bits.

CT Inputs—Power Data

Input	Power Data (32-bit Floats)		
	Units	Unit ID	Register
CT Phase A - Voltage	V	110	4097
CT Phase A - Current	A	110	4099
CT Phase A - True Power	kW	110	4101
CT Phase A - Apparent Power	kVA	110	4103
CT Phase A - Power Factor	Ratio	110	4105
CT Phase B - Voltage	V	110	4107
CT Phase B - Current	A	110	4109
CT Phase B - True Power	kW	110	4111
CT Phase B - Apparent Power	kVA	110	4113
CT Phase B - Power Factor	Ratio	110	4115
CT Phase C - Voltage	V	110	4117
CT Phase C - Current	A	110	4119
CT Phase C - True Power	kW	110	4121
CT Phase C - Apparent Power	kVA	110	4123
CT Phase C - Power Factor	Ratio	110	4125
CT Total True Power	kW	110	4127
CT Total Abs True Power	kW	110	4129
CT Total Apparent Power	kVA	110	4131
CT Total Power Factor	Ratio	110	4133

CT Inputs—Energy

Input	Energy (32-bit Floats)			Energy (64-bit Integers)		
	Units	Unit ID	Register	Units	Unit ID	Register
CT Phase A - True Energy	kWh	110	4301	mWh	113	9
CT Phase A - Apparent Energy	kVAh	110	4303	mVAh	113	13
CT Phase B - True Energy	kWh	110	4311	mWh	113	29
CT Phase B - Apparent Energy	kVAh	110	4313	mVAh	113	33
CT Phase C - True Energy	kWh	110	4321	mWh	113	49
CT Phase C - Apparent Energy	kVAh	110	4323	mVAh	113	53
CT Total True Energy	kWh	110	4327	mWh	113	61
CT Total Abs True Energy	kWh	110	4329	mWh	113	65
CT Total Apparent Energy	kVAh	110	4331	mVAh	113	69

Signal Relay Outputs

Applies only to an OptoEMU Sensor DR. Write a 1 to activate the channel, or a 0 to deactivate. Remember that each output can be wired for normally open or normally closed; make sure you know the end result of the value you write to each channel.

IMPORTANT: *The Sensor DR must already be configured correctly for receiving writes to its output channels. In OptoEMU Sensor Manager, connect to the Sensor DR, choose Relay Outputs in the left navigation panel, and make sure that Relay Control Mode is set to Public Interface - Scratch Pad.*

Output Channel	Unit ID	Register
0	110	2649
1	110	2651
2	110	2653
3	110	2655

Configuration Tables

Configuration values are written every 5 seconds. You can read configuration values but cannot write to them. All configuration changes must be made using the OptoEMU Sensor Manager software, as described in the *OptoEMU User's Guide* (form 1932).

Inputs—Enabled or Disabled

Shows whether the input is enabled. All are 32-bit integers.
Zero = disabled; non-zero = enabled.

Input	Enabled Flag	
	Unit ID	Register
CT Phase A	110	2253
CT Phase B	110	2263
CT Phase C	110	2273
CT Total	110	2279
Pulse A	110	2287
Pulse B	110	2289
Pulse C	110	2291
Pulse D	110	2293

Configuration—Pulsed Inputs

All are 32-bit floats.

Input	Unit	Unit ID	Register
Pulse A	kWh per pulse	110	5335
Pulse B	kWh per pulse	110	5337
Pulse C	kWh per pulse	110	5339
Pulse D	kWh per pulse	110	5341

Configuration—CT Inputs

All are 32-bit floats.

Input	Voltage Primary		Voltage Secondary		Current Primary		Current Secondary	
	Unit ID	Register	Unit ID	Register	Unit ID	Register	Unit ID	Register
CT Phase A	110	4501	110	4701	110	4901	110	5101
CT Phase B	110	4511	110	4711	110	4911	110	5111
CT Phase C	110	4521	110	4721	110	4921	110	5121
CT Total	110	4527	110	4727	110	4927	110	5127

Miscellaneous

Item	Unit ID	Register	Description
Public Interface Version	110	2049	Currently set to 2. Incremented whenever the documented public interface changes.
Device Type - 3V Flag	110	2051	1 for 3V, 0 for other
Device Type - DR Flag	110	2053	1 for DR1 or DR2, 0 for other
Device Type - DR1 Flag	110	2055	1 for DR1, 0 for other
Device Type - DR2 Flag	110	2057	1 for DR2, 0 for other

4: OptoEMU Sensor and Custom Programs

Introduction

You can also communicate with the OptoEMU Sensor using custom programs you develop. You can develop these programs using XML (see Chapter 5: OptoEMU Sensor and XML), using the OptoMMP Communication Toolkit (ActiveX or C++) available on our website, or using our .NET Toolkit.

This chapter introduces the free OptoEMU Sensor .NET Toolkit, a 100% managed toolkit that provides an easy-to-use interface for a .NET developer to rapidly create an OptoEMU Sensor-based Windows application. This chapter assumes that you already know how to program in .NET.

OptoEMU Sensor .NET Toolkit

The OptoEMU Sensor .NET Toolkit is built using Microsoft's .NET 3.5 Framework. It can be used for Visual Studio 2005, 2008, and 2010 versions.

The Toolkit supports Windows 7 Professional (64- and 32-bit editions), Windows Vista Business (64- and 32-bit editions), and Windows XP Professional (IPV4 only).

The Toolkit includes a sample program, source code, and documentation. Documentation is provided in HTML format.

Obtaining the Toolkit

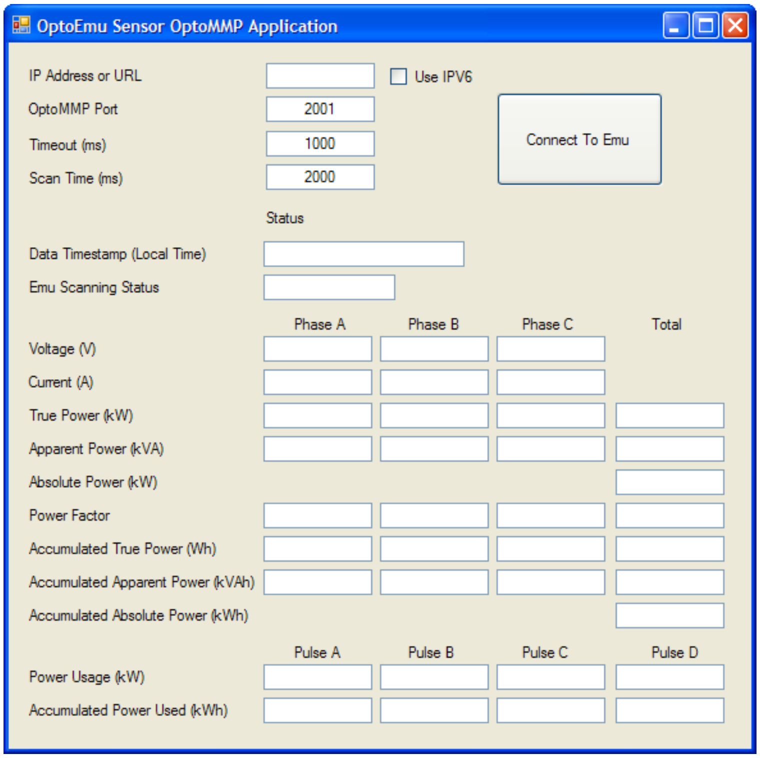
The OptoEMU Sensor .NET Toolkit is a free download from the Opto 22 website.

1. Go to www.opto22.com/site/downloads/
2. Search downloads for `.net` and download the OptoEMU Sensor .NET Toolkit to your PC.
3. Extract the .zip file to a permanent location on your PC. Extract the Documentation .zip file as well; to see documentation, double-click index.html.

Running the Sample Program

The sample program presents data from the OptoEMU Sensor by reading values in the Sensor's memory map. Source code for the sample program is included in the Toolkit.

1. Navigate to the extracted Toolkit files and double-click OptoEmu Sensor MMP.exe to open the sample program.



IP Address or URL	
□ Use IPV6	
OptoMMP Port	2001
Timeout (ms)	1000
Scan Time (ms)	2000

Status

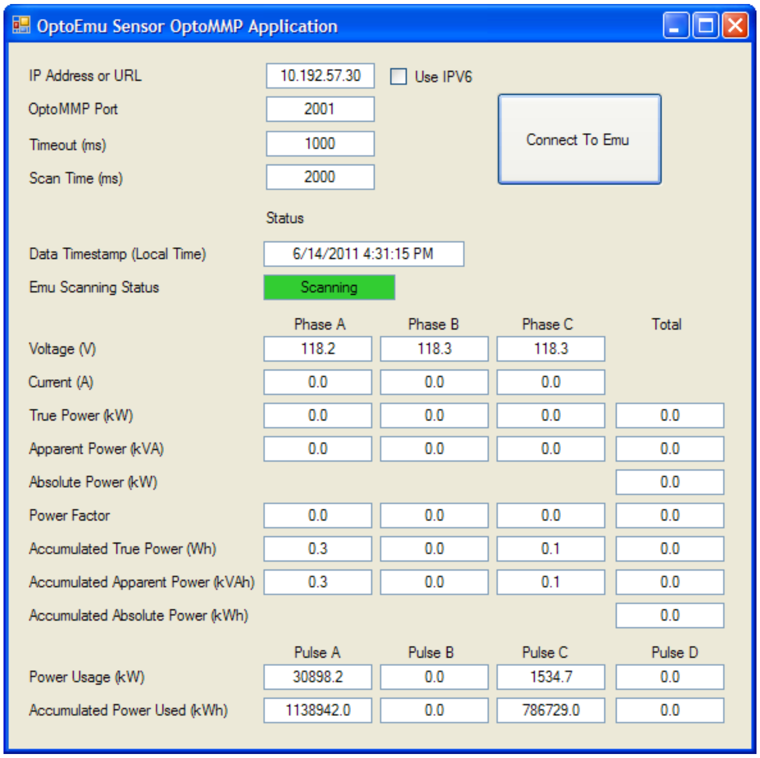
Data Timestamp (Local Time)	
Emu Scanning Status	

	Phase A	Phase B	Phase C	Total
Voltage (V)				
Current (A)				
True Power (kW)				
Apparent Power (kVA)				
Absolute Power (kW)				
Power Factor				
Accumulated True Power (Wh)				
Accumulated Apparent Power (kVAh)				
Accumulated Absolute Power (kWh)				

	Pulse A	Pulse B	Pulse C	Pulse D
Power Usage (kW)				
Accumulated Power Used (kWh)				

2. In the IP address field, enter your Sensor's IP address.

3. Click the big Connect To Emu button.



IP Address or URL	10.192.57.30
□ Use IPV6	
OptoMMP Port	2001
Timeout (ms)	1000
Scan Time (ms)	2000

Status

Data Timestamp (Local Time)	6/14/2011 4:31:15 PM
Emu Scanning Status	🟩 Scanning

	Phase A	Phase B	Phase C	Total
Voltage (V)	118.2	118.3	118.3	
Current (A)	0.0	0.0	0.0	
True Power (kW)	0.0	0.0	0.0	0.0
Apparent Power (kVA)	0.0	0.0	0.0	0.0
Absolute Power (kW)				0.0
Power Factor	0.0	0.0	0.0	0.0
Accumulated True Power (Wh)	0.3	0.0	0.1	0.0
Accumulated Apparent Power (kVAh)	0.3	0.0	0.1	0.0
Accumulated Absolute Power (kWh)				0.0

	Pulse A	Pulse B	Pulse C	Pulse D
Power Usage (kW)	30898.2	0.0	1534.7	0.0
Accumulated Power Used (kWh)	1138942.0	0.0	786729.0	0.0

Values reflect live data from your OptoEMU Sensor.

5: OptoEMU Sensor and XML

Introduction

OptoEMU Sensor is designed to send data at intervals to a software service that parses the data and presents it so that users can see and analyze it for the purpose of energy management. We call these services *data connectors*.

Pulse Energy and eSight Energy are two such software services. The OptoEMU Sensor sends them data, and they present it for viewing and analysis. Both offer web-based data services; eSight Energy also offers a software product with the same features that you can purchase and run on your local computer network.

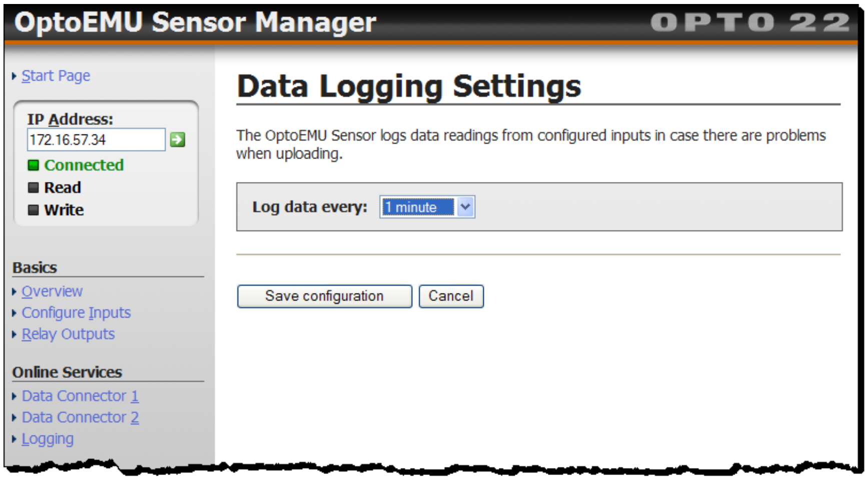
If you are familiar with XML, you can program your own software service, either web-based or local, to receive data from the OptoEMU Sensor and present the data as you wish.

This chaper gives you the XML format and schema of Sensor data. This chapter assumes that you already understand XML and know how to program in your preferred language.

Configure Your Connector

Configure your data connector using the OptoEMU Sensor Manager. For more information on Sensor Manager software, see the *OptoEMU Sensor User's Guide*, form 1932.

1. Connect to the Sensor. Under Online Services in the left nav bar, choose Logging.



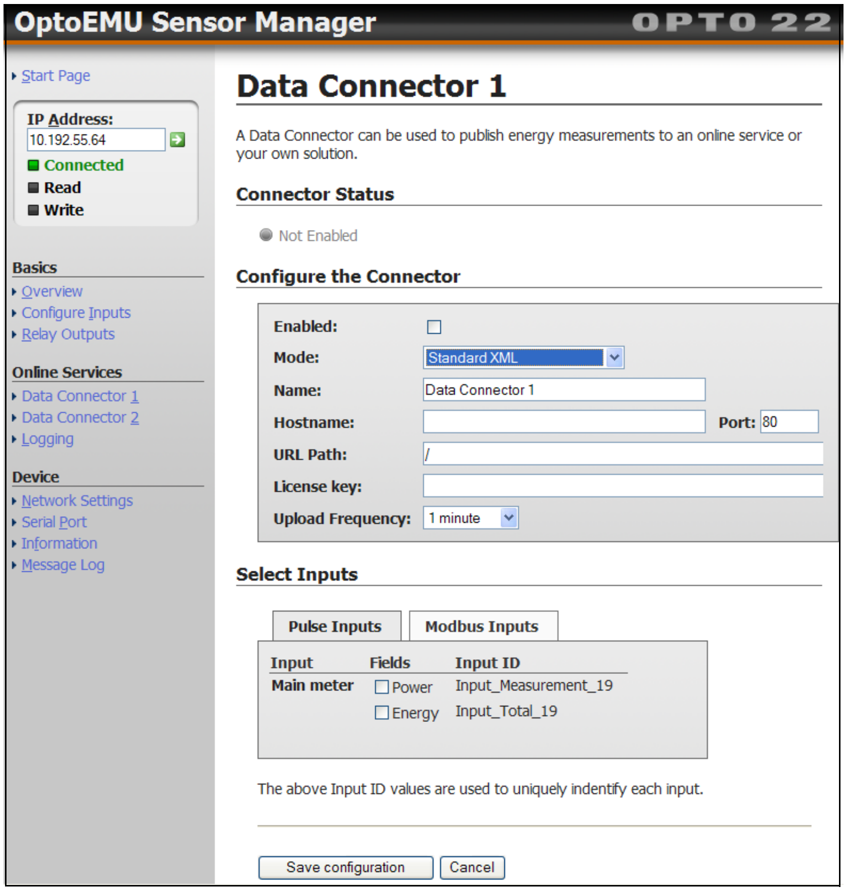
Data Logging Settings

The OptoEMU Sensor logs data readings from configured inputs in case there are problems when uploading.

Log data every: 1 minute

IP Address: 172.16.57.34

2. In the dropdown, choose how often you want the Sensor to log data from its inputs. (Default is one minute.) Click the Save configuration button.
3. In the left nav bar, choose Data Connector 1 (or 2, if 1 is already used).



IP Address: 10.192.55.64
🟩 Connected
■ Read
■ Write

Data Connector 1

A Data Connector can be used to publish energy measurements to an online service or your own solution.

Connector Status

● Not Enabled

Configure the Connector

Enabled: □
Mode: Standard XML
Name: Data Connector 1
Hostname: Port: 80
URL Path: /
License key:
Upload Frequency: 1 minute

Select Inputs

Input	Fields	Input ID
Main meter	□ Power	Input_Measurement_19
	□ Energy	Input_Total_19

The above Input ID values are used to uniquely indentify each input.

4. Put a check mark in the Enabled box.
5. In the Mode dropdown, choose Standard XML.
6. Enter the following:

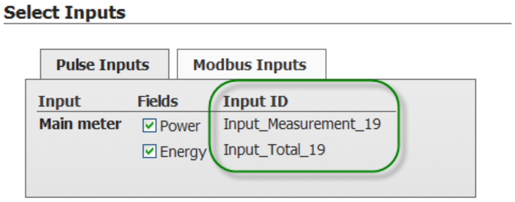
Name	A name for your data connector
Hostname	Domain name or IP address
Port	Default port is 80.
URL Path	Portion of the URL after the hostname; starts with the first slash
License key	Optional
Upload Frequency	(Minutes) How often the Sensor or Sensor DR should send its logged data to the connector. Default is 1 minute. Example: If you set Logging to 1 minute and Upload Frequency to 3 minutes, the Sensor will send three sets of data to the connector every 3 minutes.

7. Scroll down to select the inputs to send to your data connector. Click the tab for the type of input you want to send: Pulse Inputs, CT Inputs, or Modbus Inputs.

NOTE: The CT Inputs tab does not appear for a Sensor DR.

8. In the tab, put a checkmark in the box for each piece of data you want to send to your data connector.

9. Note the codes for each item of data, shown in the right-hand column. Use the Input ID to identify the data when you're parsing the XML file. Even though the name of the input may change, the Input ID does not. See "Input ID Codes" on page 32 for a complete list of codes.



Select Inputs

Input	Fields	Input ID
Main meter	☑ Power	Input_Measurement_19
	☑ Energy	Input_Total_19

10. Scroll down and click the Save configuration button.

When communication is established with the connector, the Connector Status area near the top of the screen shows "OK" in green. If you can't establish communication, make sure you have entered Gateway and DNS address information under Device > Network Settings.

XML Data from the OptoEMU Sensor

Here is a sample of energy data from the Sensor. It comes as one long string; here it's been formatted to be more humanly readable:

```
<?xml version="1.0" encoding="UTF-8" ?>
<connector xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:noNamespaceSchemaLocation="ConnectorStandardXml.xsd"
           id="OptoEMU_01C204"
           licenseKey="4-8-15-16-23-42"
           part="OPTOEMU-SNR-3V"
           MAC="00-A0-3D-01-C2-04"
           timeZone="UTC">
  <input id="Input_Total_19" label="Pulse Input A (CH 0/1) (Energy)" units="kWh">
    <reading time="2012-03-06T22:43:00-00:00" value="5114.000004" />
    <reading time="2012-03-06T22:44:00-00:00" value="5118.800004" />
    <reading time="2012-03-06T22:45:00-00:00" value="5123.000005" />
    <reading time="2012-03-06T22:46:00-00:00" value="5127.200006" />
  </input>
  <input id="Input_Measurement_19" label="Pulse Input A (CH 0/1) (Power)" units="kW">
    <reading time="2012-03-06T22:43:00-00:00" value="273.052" />
    <reading time="2012-03-06T22:44:00-00:00" value="277.363" />
    <reading time="2012-03-06T22:45:00-00:00" value="257.686" />
    <reading time="2012-03-06T22:46:00-00:00" value="253.793" />
  </input>
</connector>
```

XML Schema

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="connector">
    <xs:complexType>
      <xs:sequence minOccurs="0">
        <xs:element name="input" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence minOccurs="0">
              <xs:element name="reading " maxOccurs="unbounded">
                <xs:complexType>
                  <xs:attribute name="time" type="xs:dateTime" use="required" />
                  <xs:attribute name="value" type="xs:double" use="required" />
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="id" type="xs:ID" use="required" />
            <xs:attribute name="label" type="xs:string" use="required" />
            <xs:attribute name="units" type="xs:string" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="licenseKey" type="xs:string" use="required" />
      <xs:attribute name="part" type="xs:string" use="required" />
      <xs:attribute name="MAC" type="xs:string" use="required" />
      <xs:attribute name="id" type="xs:string" use="required" />
      <xs:attribute name="timeZone" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Input ID Codes

Each data item has an Input ID that distinguishes it from other Sensor data. Individual IDs are shown in the OptoEMU Sensor Manager when you configure the data connector (see step 9 on page 31). The following tables list all inputs, the data you can get from each, and the input IDs.

Pulse Inputs

Input	Data	Input ID
Pulse Input A (CH 0/1)	Power	Input_Measurement_19
	Energy	Input_Total_19
Pulse Input B (CH 2/3)	Power	Input_Measurement_20
	Energy	Input_Total_20
Pulse Input C (CH 4/5)	Power	Input_Measurement_21
	Energy	Input_Total_21
Pulse Input D (CH 6/7)	Power	Input_Measurement_22
	Energy	Input_Total_22

CT Inputs (Sensor 3V only)

Input	Data	Input ID
CT 3-Phase Input - Phase A	Power	Input_Measurement_2
	Energy	Input_Total_2
	Apparent Power	Input_Measurement_3
	Apparent Energy	Input_Total_3
	Power Factor	Input_Measurement_4
	Voltage	Input_Measurement_0
	Current	Input_Measurement_1
CT 3-Phase Input - Phase B	Power	Input_Measurement_7
	Energy	Input_Total_7
	Apparent Power	Input_Measurement_8
	Apparent Energy	Input_Total_8
	Power Factor	Input_Measurement_9
	Voltage	Input_Measurement_5
	Current	Input_Measurement_6
CT 3-Phase Input - Phase C	Power	Input_Measurement_12
	Energy	Input_Total_12
	Apparent Power	Input_Measurement_13
	Apparent Energy	Input_Total_13
	Power Factor	Input_Measurement_14
	Voltage	Input_Measurement_10
	Current	Input_Measurement_11
CT 3-Phase Input - Total	Power	Input_Measurement_15
	Energy	Input_Total_15
	Apparent Power	Input_Measurement_17
	Apparent Energy	Input_Total_17
	Power Factor	Input_Measurement_18

Modbus Inputs

IMPORTANT: *The Input ID reflects the units you've configured in the OptoEMU Sensor Manager. For example, if Modbus Input 0 was configured with degrees as the units, the Input ID shows degrees:*

Input	Input ID
Modbus Input 0	Input_Measurement_23_*degrees*
Modbus Input 1	Input_Measurement_24_*units*
Modbus Input 2	Input_Measurement_25_*units*
Modbus Input 3	Input_Measurement_26_*units*
Modbus Input 4	Input_Measurement_27_*units*
Modbus Input 5	Input_Measurement_28_*units*
Modbus Input 6	Input_Measurement_29_*units*
Modbus Input 7	Input_Measurement_30_*units*
Modbus Input 8	Input_Measurement_31_*units*
Modbus Input 9	Input_Measurement_32_*units*
Modbus Input 10	Input_Measurement_33_*units*
Modbus Input 11	Input_Measurement_34_*units*
Modbus Input 12	Input_Measurement_35_*units*
Modbus Input 13	Input_Measurement_36_*units*
Modbus Input 14	Input_Measurement_37_*units*
Modbus Input 15	Input_Measurement_38_*units*
Modbus Input 16	Input_Measurement_39_*units*
Modbus Input 17	Input_Measurement_40_*units*
Modbus Input 18	Input_Measurement_41_*units*
Modbus Input 19	Input_Measurement_42_*units*
Modbus Input 20	Input_Measurement_43_*units*
Modbus Input 21	Input_Measurement_44_*units*
Modbus Input 22	Input_Measurement_45_*units*
Modbus Input 23	Input_Measurement_46_*units*
Modbus Input 24	Input_Measurement_47_*units*
Modbus Input 25	Input_Measurement_48_*units*
Modbus Input 26	Input_Measurement_49_*units*
Modbus Input 27	Input_Measurement_50_*units*
Modbus Input 28	Input_Measurement_51_*units*
Modbus Input 29	Input_Measurement_52_*units*
Modbus Input 30	Input_Measurement_53_*units*
Modbus Input 31	Input_Measurement_54_*units*

Input	Input ID
Modbus Input 32	Input_Measurement_55_*units*
Modbus Input 33	Input_Measurement_56_*units*
Modbus Input 34	Input_Measurement_57_*units*
Modbus Input 35	Input_Measurement_58_*units*
Modbus Input 36	Input_Measurement_59_*units*
Modbus Input 37	Input_Measurement_60_*units*
Modbus Input 38	Input_Measurement_61_*units*
Modbus Input 39	Input_Measurement_62_*units*
Modbus Input 40	Input_Measurement_63_*units*
Modbus Input 41	Input_Measurement_64_*units*
Modbus Input 42	Input_Measurement_65_*units*
Modbus Input 43	Input_Measurement_66_*units*
Modbus Input 44	Input_Measurement_67_*units*
Modbus Input 45	Input_Measurement_68_*units*
Modbus Input 46	Input_Measurement_69_*units*
Modbus Input 47	Input_Measurement_70_*units*
Modbus Input 48	Input_Measurement_71_*units*
Modbus Input 49	Input_Measurement_72_*units*
Modbus Input 50	Input_Measurement_73_*units*
Modbus Input 51	Input_Measurement_74_*units*
Modbus Input 52	Input_Measurement_75_*units*
Modbus Input 53	Input_Measurement_76_*units*
Modbus Input 54	Input_Measurement_77_*units*
Modbus Input 55	Input_Measurement_78_*units*
Modbus Input 56	Input_Measurement_79_*units*
Modbus Input 57	Input_Measurement_80_*units*
Modbus Input 58	Input_Measurement_81_*units*
Modbus Input 59	Input_Measurement_82_*units*
Modbus Input 60	Input_Measurement_83_*units*
Modbus Input 61	Input_Measurement_84_*units*
Modbus Input 62	Input_Measurement_85_*units*
Modbus Input 63	Input_Measurement_86_*units*

A: Scratch Pad Tables and Variables

Introduction

Use these tables together with Chapter 2 to incorporate energy data from OptoEMU Sensor into a PAC Control strategy, or to control the signal relay outputs on a Sensor DR from PAC Control.

Except for outputs on a Sensor DR, all data is read-only. If you write to any tag other than the DR's relay outputs, it will have no effect on the control program running in the OptoEMU Sensor, and the value you wrote will be overwritten at the next update. Configuration data is also read-only. All configuration must be done through the OptoEMU Sensor Manager software, as described in the *OptoEMU Sensor User's Guide* (form 1932).

Data Update Frequency

CT and Pulsed Input measurements are updated when a new measurement is made:

- For CTs, once per second
- For Pulsed Inputs, the next pulse after the minimum sample period, which is set when the input is configured

Configuration values are written every 5 seconds. Modbus inputs are read sequentially, so frequency depends on network conditions and number of inputs.

Data Key

Data may indicate live measurement or accumulated quantity. Live measurement (usually expressed as power) is somewhat like your car's speedometer, which shows you your current rate of speed in miles per hour. Accumulated quantity (expressed as energy, which is power over time) is like the car's odometer, which shows the total number of miles driven since the car was first put on the road.

For pulse devices, you can get the following data:

- Power in kW (live measurement)
- Energy in kWh (accumulated quantity)

For Modbus inputs (serial or Ethernet):

Each Live Measurement is the last read value in one of the following formats:

- Live Measurement (floats)
- Live Measurement (UINT16, INT16, INT32)

- Live Measurement (UINT32)—To avoid sign-extension problems, Int64 values are used for the UINT32 type.

To determine whether the value is fresh enough to be used, also read the quality fields for the corresponding Live Measurement:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads.
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read in a round-robin fashion. If one or more inputs is currently offline, it can affect the freshness of the points that are online.

For direct (CT) connections to panels and equipment, you can get the following data:

For each phase of a three-phase device, or for each single-phase device:

- Voltage (live measurement) in volts
- Current (live measurement) in amps
- True Power (RMS voltage multiplied by RMS current and cosine of the phase angle)—Power in kW and energy in kWh
- Apparent Power (RMS voltage multiplied by RMS current; greater than or equal to True Power)—Power in kVA and energy in kVAh
- Power Factor (true power divided by apparent power)—a ratio

Totals for a three-phase device:

- Total True Power (true power for all three phases added together)—Power in kW and energy in kWh
- Total Abs True Power (true power for all three phases added together as positive values)—Power in kW and energy in kWh
- Total Apparent Power (apparent power for all three phases added together)—Power in kVA and energy in kVAh
- Total Power Factor (total true power divided by total apparent power)—a ratio

The data you need depends on what you want to track, which usually depends on how your utility company structures billing. Many utilities bill on the basis of True Power usage, but some bill based on kVA (Apparent Power). In addition, the utility may penalize facilities with low power factor, since they draw more current. Check your utility bill for details, and contact your local utility company with any questions.

Note on Energy Data

For Energy, two sets of data are provided: 32-bit floats and 64-bit integers. While both represent the total accumulated quantity, 64-bit integer values are more accurate. Choose the one that works better for you:

- *32-bit floats* are a convenient representation of the total accumulated quantity. However, because 32-bit floats do not have a high level of precision, they are not well suited for accumulating.
- *64-bit integers* represent the same value, but as an integer multiplied by 1,000,000. This value is less convenient to use but has the precision required for accumulating values. It is effectively milliwatt hours (mWh) or millivoltamp hours (mVAh). To convert mWh or mVAh to kWh or kVAh, convert the 64-bit integer value to a double precision float and divide by 1,000,000.

Data Tables

Pulse Inputs

Input	Power		Energy (32-bit Floats)		Energy (64-bit Integers)	
	Units	FloatIndex	Units	Float Index	Units	Int64 Index
Pulse A	kW	19	kWh	119	mWh	19
Pulse B	kW	20	kWh	120	mWh	20
Pulse C	kW	21	kWh	121	mWh	21
Pulse D	kW	22	kWh	122	mWh	22

Modbus Inputs

Up to 64 pieces of Modbus data can be inputs. They can be virtually any kind of data, from one or more devices, connected over serial and/or over Ethernet.

Input	Item *	Live Measurement (Floats)	Live Measurement (UINT16, INT16, INT32)	Live Measurement (UINT32) **
		Float Index	Int32 Index	Int64 Index
Remote 0	Measurement	700	700	700
	Quality (Status)	701	701	701
	Quality (Freshness)	702	702	702
Remote 1	Measurement	703	703	703
	Quality (Status)	704	704	704
	Quality (Freshness)	705	705	705
Remote *N*	Measurement	700 + (*N**3)	700 + (*N**3)	700 + (*N**3)
	Quality (Status)	701 + (*N**3)	701 + (*N**3)	701 + (*N**3)
	Quality (Freshness)	702 + (*N**3)	702 + (*N**3)	702 + (*N**3)
Remote 63	Measurement	889	889	889
	Quality (Status)	890	890	890
	Quality(Freshness)	891	891	891

* Measurement = Last read value. Use the quality fields to determine if the value is fresh enough to be used:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read in a round-robin fashion. If one or more inputs is currently offline, it can affect the freshness of the points that are online.

** Int64 values are used for the UINT32 type to avoid sign-extension problems.

CT Inputs—Power Data

Input	Power Data	
	Units	FloatIndex
CT Phase A - Voltage	V	0
CT Phase A - Current	A	1
CT Phase A - True Power	kW	2
CT Phase A - Apparent Power	kVA	3
CT Phase A - Power Factor	Ratio	4
CT Phase B - Voltage	V	5
CT Phase B - Current	A	6
CT Phase B - True Power	kW	7
CT Phase B - Apparent Power	kVA	8
CT Phase B - Power Factor	Ratio	9
CT Phase C - Voltage	V	10
CT Phase C - Current	A	11
CT Phase C - True Power	kW	12
CT Phase C - Apparent Power	kVA	13
CT Phase C - Power Factor	Ratio	14
CT Total True Power	kW	15
CT Total Abs True Power	kW	16
CT Total Apparent Power	kVA	17
CT Total Power Factor	Ratio	18

CT Inputs—Energy

Input	Energy (32-bit Floats)		Energy (64-bit Integers)	
	Units	Float Index	Units	Int64 Index
CT Phase A - True Energy	kWh	102	mWh	2
CT Phase A - Apparent Energy	kVAh	103	mVAh	3
CT Phase B - True Energy	kWh	107	mWh	7
CT Phase B - Apparent Energy	kVAh	108	mVAh	8
CT Phase C - True Energy	kWh	112	mWh	12
CT Phase C - Apparent Energy	kVAh	113	mVAh	13
CT Total True Energy	kWh	115	mWh	15
CT Total Abs True Energy	kWh	116	mWh	16
CT Total Apparent Energy	kVAh	117	mVAh	17

Signal Relay Outputs

Applies only to an OptoEMU Sensor DR. Write a 1 to activate the channel, or a 0 to deactivate. Remember that each output can be wired for normally open or normally closed; make sure you know the end result of the value you write to each channel.

IMPORTANT: *The Sensor DR must already be configured correctly for receiving writes to its output channels. In OptoEMU Sensor Manager, connect to the Sensor DR, choose Relay Outputs in the left navigation panel, and make sure that Relay Control Mode is set to Public Interface - Scratch Pad.*

Output Channel	Int 32 Index
0	300
1	301
2	302
3	303

Configuration Tables

Configuration data is for information only. All configuration changes must be made using the OptoEMU Sensor Manager software, as described in the *OptoEMU Sensor User's Guide* (form 1932).

Inputs—Enabled or Disabled

Shows whether the input is enabled. Zero = disabled; non-zero = enabled.

Input	Int 32 Index
CT Phase A	102
CT Phase B	107
CT Phase C	112
CT Total	115
Pulse A	119
Pulse B	120
Pulse C	121
Pulse D	122

Configuration—Pulsed Inputs

Input	Unit	Float Index
Pulse A	kWh per pulse	619
Pulse B	kWh per pulse	620
Pulse C	kWh per pulse	621
Pulse D	kWh per pulse	622

Configuration—CT Inputs

Input	Voltage Primary	Voltage Secondary	Current Primary	Current Secondary
	Float Index	Float Index	Float Index	Float Index
CT Phase A	202	302	402	502
CT Phase B	207	307	407	507
CT Phase C	212	312	412	512
CT Total	215	315	415	515

Miscellaneous

Item	Type	Index	Description
Public Interface Version	Int 32	0	Currently set to 2. Incremented whenever the documented public interface changes.
Device Type - 3V Flag	Int 32	1	1 for 3V, 0 for other
Device Type - DR Flag	Int 32	2	1 for DR1 or DR2, 0 for other
Device Type - DR1 Flag	Int 32	3	1 for DR1, 0 for other
Device Type - DR2 Flag	Int 32	4	1 for DR2, 0 for other

B: EMU Sensor Tag Database Tables and Variables

Introduction

Use these tables with Chapter 2 to show energy data from OptoEMU Sensor in a PAC Display HMI, send data to an OPC client using OptoOPCServer, or send data to a database using OptoDataLink. You can also control the signal relay outputs on a Sensor DR using any of these methods.

Except for outputs on a Sensor DR, all data is read-only. If you write to any tag other than the DR's relay outputs, it will have no effect on the control program running in the OptoEMU Sensor, and the value you wrote will be overwritten at the next update. Configuration data is also read-only. All configuration must be done using the OptoEMU Sensor Manager software, as described in the *OptoEMU Sensor User's Guide* (form 1932).

Data Update Frequency

CT and Pulsed Input measurements are updated when a new measurement is made:

- For CTs, once per second
- For Pulsed Inputs, the next pulse after the minimum sample period, which is set when the input is configured

Configuration values are written every 5 seconds. Modbus inputs are read sequentially, so frequency depends on network conditions and number of inputs.

Data Key

Data may indicate live measurement or accumulated quantity. Live measurement is somewhat like your car's speedometer, which shows you your current rate of speed in miles per hour, while accumulated quantity is like the car's odometer, which shows the total number of miles driven since the car was first put on the road.

For pulse devices, you can get the following data:

- Power in kW (live measurement)
- Energy in kWh (accumulated quantity)

For Modbus inputs (serial or Ethernet):

Each Live Measurement is the last read value in one of the following formats:

- Live Measurement (floats)
- Live Measurement (UINT16, INT16, INT32)

- Live Measurement (UINT32)—To avoid sign-extension problems, Int64 values are used for the UINT32 type.

To determine whether the value is fresh enough to be used, also read the quality fields for the corresponding Live Measurement:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads.
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read in a round-robin fashion. If one or more inputs is currently offline, it can affect the freshness of the points that are online.

For direct (CT) connections to panels and devices, you can get the following data:

For each phase of a three-phase device, or for each single-phase device:

- Voltage (live measurement) in volts
- Current (live measurement) in amps
- True Power (RMS voltage multiplied by RMS current and cosine of the phase angle)—Power in kW and energy in kWh
- Apparent Power (RMS voltage multiplied by RMS current; greater than or equal to True Power)—Power in kVA and energy in kVAh
- Power Factor—True power divided by apparent power (a ratio)

Totals for a three-phase device:

- Total True Power (true power for all three phases added together)—Power in kW and energy in kWh
- Total Abs True Power (true power for all three phases added together as positive values)—Power in kW and energy in kWh
- Total Apparent Power (apparent power for all three phases added together)—Power in kVA and energy in kVAh
- Total Power Factor (total true power divided by total apparent power)—a ratio

The data you need depends on what you want to track, which usually depends on how your utility company structures billing. Many utilities bill on the basis of True Power usage, but some bill based on kVA (Apparent Power). In addition, the utility may penalize facilities with low power factor, since they draw more current. Check your utility bill for details, and contact your local utility company with any questions.

Note on Energy Data

For Energy data, two tables are provided: Pub_Input_TotalQuantity and Pub_Input_TotalQuantity64. While both represent the total accumulated quantity, TotalQuantityInt64 values are more accurate. Choose the one that works better for you:

- *Pub_Input_TotalQuantity* values are 32-bit floats, which are a convenient representation of the total accumulated quantity. However, because 32-bit floats do not have a high level of precision, they are not well suited for accumulating.
- *Pub_Input_TotalQuantity64* values represent the same value, but as an integer multiplied by 1,000,000. This value is less convenient to use but has the precision required for accumulating values. It is effectively milliwatt hours (mWh) or millivoltamp hours (mVAh). To convert mWh or mVAh to kWh or kVAh, convert the 64-bit integer value to a double precision float and divide by 1,000,000.

Data Tables

Pulse Inputs

Input	Power (32-bit floats)		Energy (32-bit Floats or 64-bit Integers)*	
	Units	Tag	Units	Tag
Pulse A	kW	Pub_Input_Measurement[19]	kWh mWh	Pub_Input_TotalQuantity[19] Pub_Input_TotalQuantity64[19]
Pulse B	kW	Pub_Input_Measurement[20]	kWh mWh	Pub_Input_TotalQuantity[20] Pub_Input_TotalQuantity64[20]
Pulse C	kW	Pub_Input_Measurement[21]	kWh mWh	Pub_Input_TotalQuantity[21] Pub_Input_TotalQuantity64[21]
Pulse D	kW	Pub_Input_Measurement[22]	kWh mWh	Pub_Input_TotalQuantity[22] Pub_Input_TotalQuantity64[22]

* Use either Pub_Input_TotalQuantity or Pub_Input_TotalQuantity64. Table elements remain the same.

Modbus Inputs

Up to 64 pieces of Modbus data can be inputs. They can be virtually any kind of data, from one or more devices, connected over serial and/or over Ethernet.

Input	Item *	Live Measurement (Floats)	Live Measurement (UINT16, INT16, INT32)	Live Measurement (UINT32) **
Remote 0	Measurement	Pub_Remote_Float[0]	Pub_Remote_Int32[0]	Pub_Remote_Int64[0]
	Quality (Status)	Pub_Remote_Float[1]	Pub_Remote_Int32[1]	Pub_Remote_Int64[1]
	Quality (Freshness)	Pub_Remote_Float[2]	Pub_Remote_Int32[2]	Pub_Remote_Int64[2]
Remote 1	Measurement	Pub_Remote_Float[3]	Pub_Remote_Int32[3]	Pub_Remote_Int64[3]
	Quality (Status)	Pub_Remote_Float[4]	Pub_Remote_Int32[4]	Pub_Remote_Int64[4]
	Quality (Freshness)	Pub_Remote_Float[5]	Pub_Remote_Int32[5]	Pub_Remote_Int64[5]
Remote *N*	Measurement	Pub_Remote_Float[0+(*N**3)]	Pub_Remote_Int32[0+(*N**3)]	Pub_Remote_Int64[0+(*N**3)]
	Quality (Status)	Pub_Remote_Float[1+(*N**3)]	Pub_Remote_Int32[1+(*N**3)]	Pub_Remote_Int64[1+(*N**3)]
	Quality (Freshness)	Pub_Remote_Float[2+(*N**3)]	Pub_Remote_Int32[2+(*N**3)]	Pub_Remote_Int64[2+(*N**3)]
Remote 63	Measurement	Pub_Remote_Float[189]	Pub_Remote_Int32[189]	Pub_Remote_Int64[189]
	Quality (Status)	Pub_Remote_Float[190]	Pub_Remote_Int32[190]	Pub_Remote_Int64[190]
	Quality (Freshness)	Pub_Remote_Float[191]	Pub_Remote_Int32[191]	Pub_Remote_Int64[191]

* Measurement = Last read value. Use the quality fields to determine if the value is fresh enough to be used:

- Quality (Status): 0 = Value has not been read yet. -1 or less = Number of consecutive failed reads. 1 or greater = Number of consecutive successful reads
- Quality (Freshness): Number of seconds since last successful read. Modbus inputs are read in a round-robin fashion. If one or more inputs is currently offline, it can affect the freshness of the points that are online.

** Int64 values are used for the UINT32 type to avoid sign-extension problems.

CT Inputs—Power Data

Input	Power Data (32-bit Floats)	
	Units	Tag
CT Phase A - Voltage	V	Pub_Input_Measurement[0]
CT Phase A - Current	A	Pub_Input_Measurement[1]
CT Phase A - True Power	kW	Pub_Input_Measurement[2]
CT Phase A - Apparent Power	kVA	Pub_Input_Measurement[3]
CT Phase A - Power Factor	Ratio	Pub_Input_Measurement[4]
CT Phase B - Voltage	V	Pub_Input_Measurement[5]
CT Phase B - Current	A	Pub_Input_Measurement[6]
CT Phase B - True Power	kW	Pub_Input_Measurement[7]
CT Phase B - Apparent Power	kVA	Pub_Input_Measurement[8]
CT Phase B - Power Factor	Ratio	Pub_Input_Measurement[9]
CT Phase C - Voltage	V	Pub_Input_Measurement[10]
CT Phase C - Current	A	Pub_Input_Measurement[11]
CT Phase C - True Power	kW	Pub_Input_Measurement[12]
CT Phase C - Apparent Power	kVA	Pub_Input_Measurement[13]
CT Phase C - Power Factor	Ratio	Pub_Input_Measurement[14]
CT Total True Power	kW	Pub_Input_Measurement[15]
CT Total Abs True Power	kW	Pub_Input_Measurement[16]
CT Total Apparent Power	kVA	Pub_Input_Measurement[17]
CT Total Power Factor	Ratio	Pub_Input_Measurement[18]

CT Inputs—Energy

Input	Energy (32-bit Floats)		Energy (64-bit Integers)	
	Units	Tag	Units	Tag
CT Phase A - True Energy	kWh	Pub_Input_TotalQuantity[2]	mWh	Pub_Input_TotalQuantity64[2]
CT Phase A - Apparent Energy	kVAh	Pub_Input_TotalQuantity[3]	mVAh	Pub_Input_TotalQuantity64[3]
CT Phase B - True Energy	kWh	Pub_Input_TotalQuantity[7]	mWh	Pub_Input_TotalQuantity64[7]
CT Phase B - Apparent Energy	kVAh	Pub_Input_TotalQuantity[8]	mVAh	Pub_Input_TotalQuantity64[8]
CT Phase C - True Energy	kWh	Pub_Input_TotalQuantity[12]	mWh	Pub_Input_TotalQuantity64[12]
CT Phase C - Apparent Energy	kVAh	Pub_Input_TotalQuantity[13]	mVAh	Pub_Input_TotalQuantity64[13]
CT Total True Energy	kWh	Pub_Input_TotalQuantity[15]	mWh	Pub_Input_TotalQuantity64[15]
CT Total Abs True Energy	kWh	Pub_Input_TotalQuantity[16]	mWh	Pub_Input_TotalQuantity64[16]
CT Total Apparent Energy	kVAh	Pub_Input_TotalQuantity[17]	mVAh	Pub_Input_TotalQuantity64[17]

Signal Relay Outputs

Applies only to an OptoEMU Sensor DR. Write a 1 to activate the channel, or a 0 to deactivate. Remember that each output can be wired for normally open or normally closed; make sure you know the end result of the value you write to each channel.

IMPORTANT: *The Sensor DR must already be configured correctly for receiving writes to its output channels. In OptoEMU Sensor Manager, connect to the Sensor DR, choose Relay Outputs in the left navigation panel, and make sure that Relay Control Mode is set to Public Interface - Strategy Tags.*

Output Channel	Tag
0	Pub_Output_State[0]
1	Pub_Output_State[1]
2	Pub_Output_State[2]
3	Pub_Output_State[3]

Configuration Tables

Configuration data is for information only. All configuration changes must be made using the OptoEMU Sensor Manager software, as described in the *OptoEMU Sensor User's Guide* (form 1932).

Inputs—Enabled or Disabled

Shows whether the input is enabled. All are 32-bit integers.
Zero = disabled; non-zero = enabled.

Input Enabled	Tag
CT Phase A	Pub_Cfg_Enabled[2]
CT Phase B	Pub_Cfg_Enabled[7]
CT Phase C	Pub_Cfg_Enabled[12]
CT Total	Pub_Cfg_Enabled[15]
Pulse A	Pub_Cfg_Enabled[19]
Pulse B	Pub_Cfg_Enabled[20]
Pulse C	Pub_Cfg_Enabled[21]
Pulse D	Pub_Cfg_Enabled[22]

Configuration—Pulsed Inputs

All are 32-bit floats.

Input	Unit	Tag
Pulse A	kWh per pulse	Pub_Cfg_Pulsed_QuantityPerPulse[19]
Pulse B	kWh per pulse	Pub_Cfg_Pulsed_QuantityPerPulse[20]
Pulse C	kWh per pulse	Pub_Cfg_Pulsed_QuantityPerPulse[21]
Pulse D	kWh per pulse	Pub_Cfg_Pulsed_QuantityPerPulse[22]

Configuration—CT Inputs

All are 32-bit floats.

Input	Voltage Primary	Voltage Secondary
CT Phase A	Pub_Cfg_CT_VoltagePrimary[2]	Pub_Cfg_CT_VoltageSecondary[2]
CT Phase B	Pub_Cfg_CT_VoltagePrimary[7]	Pub_Cfg_CT_VoltageSecondary[7]
CT Phase C	Pub_Cfg_CT_VoltagePrimary[12]	Pub_Cfg_CT_VoltageSecondary[12]
CT Total	Pub_Cfg_CT_VoltagePrimary[15]	Pub_Cfg_CT_VoltageSecondary[15]

Input	Current Primary	Current Secondary
CT Phase A	Pub_Cfg_CT_CurrentPrimary[2]	Pub_Cfg_CT_CurrentSecondary[2]
CT Phase B	Pub_Cfg_CT_CurrentPrimary[7]	Pub_Cfg_CT_CurrentSecondary[7]
CT Phase C	Pub_Cfg_CT_CurrentPrimary[12]	Pub_Cfg_CT_CurrentSecondary[12]
CT Total	Pub_Cfg_CT_CurrentPrimary[15]	Pub_Cfg_CT_CurrentSecondary[15]

Miscellaneous

Item	Type	Tag	Description
Public Interface Version	Int 32	Pub_Version_Api	Currently set to 2. Incremented whenever the documented public interface changes.
Firmware Version	String	Pub_Version_Firmware	Example: "R9.1a"
Program Version	String	Pub_Version_Program	Example: "R1.1a"
Device Type - 3V Flag	Int 32	Pub_Device_Type_3V	1 for 3V, 0 for other
Device Type - DR Flag	Int 32	Pub_Device_Type_DR	1 for DR1 or DR2, 0 for other
Device Type - DR1 Flag	Int 32	Pub_Device_Type_DR1	1 for DR1, 0 for other
Device Type - DR2 Flag	Int 32	Pub_Device_Type_DR2	1 for DR2, 0 for other