

---

# **Software Communications Specification**

**For**

## **Open .Net Micro Mid-level Power Meter Communications Protocol**

**Version 1.01 approved**

**Prepared by Open Source Meter**

**Open Source Meter Inc**

**Jan10, 2010**

*Software specification*

# Table of Contents

**Table of Contents** ..... **ii**

**Revision History** ..... **ii**

**1. Introduction**.....**1**

    1.1 Purpose ..... 1

    1.2 Intended Audience and Reading Suggestions ..... 1

    1.3 Project Scope ..... 1

    1.4 References ..... 1

**2. Overall Description**.....**2**

    2.1 Product Perspective ..... 2

    2.2 Product Features ..... 2

    2.3 Operating Environment ..... 2

    2.4 Design and Implementation Constraints ..... 2

    2.5 User Documentation ..... 3

    2.6 Assumptions and Dependencies ..... 3

**3. System Features**.....**3**

**4. External Interface Requirements** .....**3**

    4.1 User Interfaces ..... 3

    4.2 Hardware Interfaces ..... 3

    4.3 Software Interfaces ..... 3

        4.3.1 Addressing ..... 3

        4.3.2 Data Point Addressing ..... 3

    4.4 Register Address List ..... 4

**5. Other Nonfunctional Requirements**.....**10**

    5.1 Performance Requirements ..... 10

    5.2 Security Requirements ..... 10

# Revision History

Name	Date	Reason For Changes	Version
	4/7/10	First Draft	V1_00

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to define a Modbus communications protocol for interface to a power meter. The communications protocol shall be designed to support a common communications protocol type that is easily configurable to conform to a very flexible metering application.

## 1.2 Intended Audience and Reading Suggestions

The intended audience of this document is for developers, project managers and integrators involved in the development and utility of this product. It is expected that developers use this document as the basis for the development of detailed software specifications of the product hence forward defined in this document.

## 1.3 Project Scope

The project name is “Open .Net Mid-level Power Meter Communication Protocol Stack”. The purpose of the software is to develop or integrate a Communication stack in the .net micro mid-level Power Meter for use with a user Application of the Open .net Power meter.

The goal of this software is to provide a fixable well accepted communications interface that can be managed though a large pool of developers that support .net platforms.

The long term strategy of this software project is to produce an interim communications protocol stack that can be used for initial projects and as a stepping stone to more sophisticated protocol stacks.

## 1.4 References

Free Modbus source code provider

<http://code.google.com/p/nmodbus/>

Microsoft Reference list:

<http://www.microsoft.com/netmf/resources/default.aspx>

GHI Electronics

<http://www.ghielectronics.com/product/116>

## 2. Overall Description

### 2.1 Product Perspective

The software product is the first product of its kind. This software product is a new design. The product is intended to interface with an application layer for control, configuration and data management of a power meter. The protocol stack will interface serial to a TCP/IP tunnel.

### 2.2 Product Features

The product shall incorporate a common communication protocol stack. The protocol stack shall be a Modbus slave. The Modbus slave shall be a serial interface to a TCP/IP tunnel. The communications stack can be purchased from a third party as libraries or as source code. The product shall have an interface to a dual port data source. The product shall have an interface to the metering application to perform control and configuration.

The end product shall provide a set of pseudo code as examples to for control interfaces and data retrieval. The product shall be license free.

### 2.3 Operating Environment

The software will be developed for .Net Micro using the .Net Micro Framework. The coding environment will be C#. The Hardware platform will be the Power-Meter platform designed by Just Nodes Inc. The platform is based on the LPC2387 from NXP. The SDK is provided by GHI Electronics.

This application must be designed to interface with following board level peripherals:

- Serial IP Module from Lantronix

In the grandest sense the operating environment also refers to the operating system. It is assumed that the aspect of the managed code shall provide many of the features of an operating system including memory resource management, exception handling and garbage collection among other functions.

### 2.4 Design and Implementation Constraints

The product currently has a set of know limitations and constraints that are particularly centered on the operation environment with respect to the hardware available. The hardware had been designed with the assumption that a small memory footprint OS would be used such as Free RTOS or equivalent. The current requirement now requires that .Net Micro be used requiring a much larger footprint of 256K or half the available 512K. This large memory footprint makes it likely that a hardware upgrade will be required to allow for potentially more customer requirement and more customer options. The product has a clean migration path to upgrade to a larger memory platform.

The .Net Micro platform is a managed code platform and is not RTOS and thus not intended for a deterministic system. Expected delays as a result of the managed code could be a long as 10ms.

The application needs to be designed as an object oriented design. The development environment is Visual Studio and the development language needs to be C#.

The development methodology should be of a type that is a type of waterfall modified to include feedback loops. The methodology requires up front research and development of a specification.

The software development organization will be responsible for defining and maintaining a document control standard and application that can be share online. Typical applications might include Source Gear: Vault and others.

## **2.5 User Documentation**

It is expected that a detailed specification be written that defines each of the objects that are to be designed. A complete description of each of the methods of the objects (Class) shall be provided.

## **2.6 Assumptions and Dependencies**

It is intended that the developer produce the interface to the communications protocol based on the a short power meter test application that will support all control of the principal meter APIs. The final product will be provided with source code as a DLL although this is product is intended to be Open source.

# **3. System Features**

# **4. External Interface Requirements**

## **4.1 User Interfaces**

The Modbus user interface shall be based on the following tables.

## **4.2 Hardware Interfaces**

The Modbus RTU interface shall use a TTL serial interface to connect to a TTL to Modbus TCP over Ethernet Bridge.

## **4.3 Software Interfaces**

### **4.3.1 Addressing**

Device Serial Slave Modbus RTU address range 1-247

### **4.3.2 Data Point Addressing**

Serial Slave Modbus RTU address registers range 0001-FFFF (Not used by the customer)

### **4.3.3 Data Point Addressing**

Functions supported 4 and 16

## 4.4 Register Address List

### MODBUS Register List

#### Basic Registers – Floating Point 32bit

Registers	Name	Units	Access	Description
Energy Registers				
1001, 1002	EnergySum	kWh	Read-Only	Total net (bidirectional) energy of Phase A,B,C (The Sum of all E1 outputs that conform to the pulse output equation)
Power Registers				
1009, 1010	PowerSum	W	Read-Only	Apparent power, sum of active phases
1011, 1012	PowerA	W	Read-Only	Real power, phase A (Apparent Power)
1013, 1014	PowerB	W	Read-Only	Real power, phase B (Apparent Power)
1015, 1016	PowerC	W	Read-Only	Real power, phase C (Apparent Power)
Voltage Registers				
1019, 1020	VoltA	V	Read-Only	RMS voltage, phase A to neutral
1021, 1022	VoltB	V	Read-Only	RMS voltage, phase B to neutral
1023, 1024	VoltC	V	Read-Only	RMS voltage, phase C to neutral
Frequency Register				
1033, 1034	Freq	Hz	Read-Only	(Not Implemented) Power line frequency

#### Basic Registers – Integer 16bit

Registers	Name	Units	Access	Description
Energy Registers				
1201,1202	EnergySum	0.1 kWh	Read-Only	Total net (bidirectional) energy of Phase A,B,C (The Sum of all E1 outputs that

				conform to the pulse output equation) represented in 0.1kWh
<b>Power Registers</b>				
1209, 1210	PowerSum	0.1W	Read-Only	(Apparent Power), sum of active phases
1211,1212	PowerA	0.1W	Read-Only	(Apparent Power) phase A
1213,1214	PowerB	0.1W	Read-Only	(Apparent Power) phase B
1215, 1216	PowerC	0.1W	Read-Only	(Apparent Power) phase C
<b>Voltage Registers</b>				
1217	VoltA	0.1 V	Read-Only	RMS voltage, phase A to neutral
1218	VoltB	0.1 V	Read-Only	RMS voltage, phase B to neutral
1219	VoltC	0.1 V	Read-Only	RMS voltage, phase C to neutral
<b>Frequency Register</b>				
1221	Freq	0.1 Hz	Read-Only	(Not Implemented) Power line frequency

### Advanced Registers - Floating Point

Registers	Name	Units	Access	Description
<b>Energy Registers</b>				
1101, 1102	EnergyA	kWh	Read-Only	Net (bidirectional) energy, phase A (E1 output)
1103, 1104	EnergyB	kWh	Read-Only	Net (bidirectional) energy, phase B (E1 output)
1105, 1106	EnergyC	kWh	Read-Only	Net (bidirectional) energy, phase C (E1 output)
<b>Power Factor Registers</b>				
1139, 1140	PowerFactorAvg		Read-Only	Power factor, average of active phases
1141, 1142	PowerFactorA		Read-Only	Power factor, phase A
1143, 1144	PowerFactorB		Read-Only	Power factor, phase B
1145, 1146	PowerFactorC		Read-Only	Power factor, phase C
<b>Current Registers</b>				
1163, 1164	CurrentA	A	Read-Only	RMS current, phase

				A
1165, 1166	CurrentB	A	Read-Only	RMS current, phase B
1167, 1168	CurrentC	A	Read-Only	RMS current, phase C

**Advanced Registers – Integer 16bit**

Registers	Name	Units	Access	Description
<b>Energy Registers</b>				
1301, 1302	EnergyA	0.1 kWh	Read-Only	Net energy, phase A
1303, 1304	EnergyB	0.1 kWh	Read-Only	Net energy, phase B
1305, 1306	EnergyC	0.1 kWh	Read-Only	Net energy, phase C
<b>Power Factor Registers</b>				
1339	PowerFactorAvg	0.01	Read-Only	Power factor, average of active phases (returns value from -1 to 1)
1340	PowerFactorA	0.01	Read-Only	Power factor, phase A (returns value from -1 to 1)
1341	PowerFactorB	0.01	Read-Only	Power factor, phase B (returns value from -1 to 1)
1342	PowerFactorC	0.01	Read-Only	Power factor, phase C (returns value from -1 to 1)
<b>Current Registers</b>				
1351	CurrentA	0.1A	Read-Only	RMS current, phase A
1352	CurrentB	0.1A	Read-Only	RMS current, phase B
1353	CurrentC	0.1A	Read-Only	RMS current, phase C

**Configuration Registers - Integer**

Registers	Name	Units	Access	Description
1604	CtAmpsA	1 A	Read/Write	CT rated current for phase A stored in NVM and retrieved on reset from NVM
1605	CtAmpsB	1 A	Read/Write	CT rated current override for phase B stored in NVM and retrieved on reset from NVM

1606	CtAmpsC	1 A	Read/Write	CT rated current override for phase C stored in NVM and retrieved on reset from NVM
1609	PowerIntScale	1 W	Read/Write	(Not Implemented) Scale factor for integer power registers
1610	VrmsGainPhaseA		Read/Write	Set Vrms Gain register and stored in NVM and retrieved on reset from NVM
1611	VrmsGainPhaseB		Read/Write	Set Vrms Gain register and stored in NVM and retrieved on reset from NVM
1612	VrmsGainPhaseC		Read/Write	Set Vrms Gain register and stored in NVM and retrieved on reset from NVM
1613	IrmsGainPhaseA		Read/Write	Set Irms Gain register and stored in NVM and retrieved on reset from NVM
1614	IrmsGainPhaseB		Read/Write	Set Irms Gain register and stored in NVM and retrieved on reset from NVM
1615	IrmsGainPhaseC		Read/Write	Set Irms Gain register and stored in NVM and retrieved on reset from NVM
1616	VoffsetPhaseA		Read/Write	Set Voffset Gain register and stored in NVM and retrieved on reset from NVM
1617	VoffsetPhaseB		Read/Write	Set Voffset Gain register and stored in NVM and retrieved on reset from NVM
1618	VoffsetPhaseC		Read/Write	Set Voffset Gain register and stored in NVM and

1619	loffsetPhaseA		Read/Write	Set IOffset Gain register and stored in NVM and retrieved on reset from NVM
1620	loffsetPhaseA		Read/Write	Set IOffset Gain register and stored in NVM and
1621	loffsetPhaseA		Read/Write	Set IOffset Gain register and stored in NVM and retrieved on reset from NVM

### Communication Registers

Registers	Name	Units	Access	Description
1651	ApplyComConfig		Read/Write	Writing 1234 applies the configuration settings below. Reads 1 if changes not applied yet.
1652	Address		Read/Write	MODBUS address has default address
1653	BaudRate		Read/Write	0 = 9600 baud (default) 1 = 19200 baud 2 = 115200 baud
1654	ParityMode		Read/Write	0 = N81 (no parity, one stop bit)
1655	ModbusMode		Read/Write	0=RTU 1=TCP-RTU

### Customer Diagnostic Registers - Integer

Registers	Name	Units	Access	Description
1701, 1702	Serial Number		Read/Write	The unique OpenSourceMeter serial number
1707	Model		Read/Write	Encoded OpenSource model
1708	Version		Read/Write	Firmware version
1709	Options		Read/Write	OpenSource options (Not used)
1710	ErrorStatus	n.a.	Read/Write	List of recent errors and events

1716	ErrorStatus1	n.a.	Read/Write	Newest error or event (0 = no errors) tbd
1717	ErrorStatus2	n.a.	Read/Write	Next oldest error or event tbd
1718	ErrorStatus3	n.a.	Read/Write	Next oldest error or event tbd
1719	ErrorStatus4	n.a.	Read/Write	Next oldest error or event tbd
1720	ErrorStatus5	n.a.	Read/Write	Next oldest error or event tbd
1721	ErrorStatus6	n.a.	Read/Write	Next oldest error or event tbd
1722	ErrorStatus7	n.a.	Read/Write	Next oldest error or event tbd
1723	ErrorStatus8	n.a.	Read/Write	Oldest error or event tbd
1724,1725,	<b>Command RegisterPhase A</b>		<b>Read/Write</b>	<b>Set Command register phase A stored in NVM and retrieved on reset from NVM</b>
1726, 1727	<b>Status Register A</b>		<b>Read-Only</b>	<b>Read status register phase A</b>
1728, 1729	<b>Command RegisterPhase B</b>		<b>Read/Write</b>	<b>Set Command register phase B stored in NVM and retrieved on reset from NVM</b>
1730, 1731	<b>Status Register B</b>		<b>Read-Only</b>	<b>Read status register phase B</b>
1732, 1733	<b>Command RegisterPhase C</b>		<b>Read/Write</b>	<b>Set Command register phase C stored in NVM and retrieved on reset from NVM</b>
1734, 1735	<b>Status Register C</b>		<b>Read-Only</b>	<b>Read status register phase C</b>
1800	<b>RPC Mode</b>		<b>Read/Write</b>	<p><b>Run Program and Configuration Mode Can be set when logged in: Default = Logged In</b></p> <p><b>Register set to:</b>  <b>RUN Mode = 0</b>  <b>Program Mode = 1</b>  <b>Configure Mode = 2</b>  <b>Value &gt; 2 = RUN Mode</b></p> <p><b>RUN Mode = Normal operation</b>  <b>Program Mode =</b></p>

				<p>State of limited operation mode for programming reverts back to Run mode after 30-seconds</p> <p>Configuration Mode = I, Vrms Gain, and I,V Offset, and Command register configurable</p>
--	--	--	--	--

**Notes:**

Integer values are in 1/10 of a KWh, W, A or V.

Any Item that is **BOLD** and has highlight is required to be applied

**Example Configuration Mode:****Configuration**

- Write register 1800 = 2 configuration mode
- Write register 1612 – 1618 with new values
- Write register 1651 = 1234 to apply changes
- Read resister 1651 = 1 completed anything else = not completed
- Write register 1800 = 0 run mode
- Done

**Example Program Mode:****Program**

- Write register 1800 = 1 program mode ( Interrupts turned off and looping stopped for 30 seconds defaults back to RUN Mode after the timeout )
- Program with debugger
- Defaults back to Run Mode after reprogramming

## 5. Other Nonfunctional Requirements

### 5.1 Performance Requirements

The current implementation will be limited the maximum best serial speed of 115200 Baud or by the 10ms garbage collection cycle of the .net micro frame work.

### 5.2 Security Requirements

Security to be implemented with a simple user ID and Password