

Mapping Modbus Registers to BACnet® Objects



History

Revision	Date	Description	Author
1.00	September 2014	First release	SDa

Contents

1	Introduction	3
2	Modbus Device Addressing	3
2.1	Modbus Data Addressing	4
2.2	Modbus Function Codes	5
3	Assigning Modbus Registers to BACnet Objects	5
3.1	BACnet Object Properties	6
3.2	Creating Device Objects	7
4	Developing a Modbus Device Profile	7
4.1.1	Example Modbus Register Table for an Instrument	8
5	Virtual Routing	10
6	A Modbus Device Profile as Viewed in a Spreadsheet	11
7	Supported BACnet Units	15
7.1	Editing Modbus Device Profiles in a Spreadsheet	15
8	Using the Anybus BACnet to Modbus Gateway	15
8.1	Configure Settings	16
8.1.1	System	16
8.1.2	BACnet	16
8.1.3	Modbus Serial	16
8.2	Mapping Configuration	17
8.2.1	Authentication	18
8.2.2	Mapping Status	18
9	Adding Device Profiles	19
10	Firmware Updates	19
11	Modbus Utility	20

1 Introduction

Although Modbus is a popular protocol, being a relatively simple protocol to use and understand, it nonetheless is still not BACnet-compliant. Making a Modbus device BACnet-compliant requires a gateway, such as the Anybus BACnet to Modbus Gateway. This application note explains how to use this gateway to bring a Modbus device up to BACnet/IP.

This requires the development of a device profile for the Modbus device being accessed. Once the device profile is developed, it is uploaded to the Anybus BACnet to Modbus Gateway and a scan list is created by selecting the points of interest in the device profile. Only enabled Modbus points on the scan list will be accessible from a BACnet client.

Various ready-to-use Modbus device profiles are provided by HMS, which can however, be modified by customers for their own unique requirements. It is also possible to develop a completely new device profile. This application note provides information on how to do so.

One important feature of the Anybus BACnet to Modbus Gateway is that it supports virtual routing, meaning that each connected Modbus device is treated by BACnet as a BACnet device with a unique device instance. This makes it easy to add multiple Modbus devices with identical device profiles to the same gateway. Furthermore, multiple Modbus devices with different device profiles are just as easily supported. The only restriction is that each Anybus BACnet to Modbus Gateway supports up to 30 Modbus devices and a total of 1000 Modbus scanned points.

Modbus devices can be any of the variants Modbus ASCII, Modbus RTU and Modbus TCP. Modbus ASCII is an older serial protocol that is seldom used. Modbus RTU is a very popular serial line interface, whilst Modbus TCP operates over Ethernet. The Anybus BACnet to Modbus Gateway supports all of these Modbus variants.

One advantage of BACnet devices over Modbus devices is that BACnet devices allow their objects to be “discovered”, while Modbus devices lack this capability. This speeds up commissioning, whereby BACnet devices are modelled as a collection of objects that are “network visible.” By being able to view these BACnet objects and by understanding their pre-defined properties, much can be learned about the device. For Modbus devices it will be necessary to consult the user’s manual to understand the meaning of the Modbus registers. There is however, a way to create BACnet objects from Modbus registers using the Anybus BACnet to Modbus Gateway. Because Modbus equipment is popular, being able to integrate these devices into a “single-seat BACnet system” is important. In order to understand how Modbus device profiles are created, a short review of Modbus is provided below.

2 Modbus Device Addressing

Modbus slaves are addressed from 1-247, with address 0 reserved as the broadcast address – a message directed to all devices. The Anybus BACnet to Modbus Gateway functions as the Modbus master and thus has no address assignment. It reserves the Modbus slave address 247 for future use. Up to 30 Modbus serial devices can connect to the RS-485 port on the Anybus BACnet to Modbus Gateway, with each assigned a unique Modbus address in the allowable range. For Modbus TCP devices, each device must connect to a port on an Ethernet switch located on the same subnet as the Anybus BACnet to Modbus Gateway. As mentioned previously, the Anybus BACnet to Modbus Gateway supports virtual routing, meaning that each attached Modbus slave (serial or TCP) is treated as a unique BACnet device. Therefore, at a minimum, one BACnet device object must be created for each



connected Modbus device. Other BACnet objects must be created to represent Modbus data in the form of 1-bit and 16-bit registers.

2.1 Modbus Data Addressing

Modbus data is considered to be divided into four memory blocks — coils, discrete inputs, input registers and holding registers. Each memory location in each block can be accessed by a 16-bit address. Discrete inputs and coils are considered to be 1-bit registers, while input registers and holding registers are 16-bit. A 32-bit register would require two memory locations. Traditional addressing practice uses 5-digit decimal references with the leading digit signifying a particular memory block, as shown in Table 1. This approach is easier to understand, but it only addresses 10,000 points or registers, instead of the 65,536 that are possible for each block.

As the complexity of Modbus devices increases, so does the need to use more memory locations. Modern Modbus addressing incorporates 6-digit references, with the first digit signifying the memory block address (0, 1, 3, and 4), and the remaining 5 digits representing the complete 16 bits of address space for that block. Contemporary Controls utilizes the 6-digit addressing scheme as shown in Table 2. Note that the first register location within a block begins at 1. This is called PLC addressing, which was used by the original Modicon PLC. The data is actually stored in computer memory beginning with 0, but Contemporary Controls uses PLC addressing references to avoid confusion.

Memory Block	Bits	Access	Address Range
Coils	1	Read/Write	00001-09999
Discrete inputs	1	Read-Only	10001-19999
Input registers	16	Read-Only	30001-39999
Holding registers	16	Read/Write	40001-49999
Table 1 — Traditional decimal representation of Modbus registers using a 5-digit addressing scheme.			
Memory Block	Bits	Access	Address Range
Coils	1	Read/Write	000001-065536
Discrete inputs	1	Read-Only	100001-165536
Input registers	16	Read-Only	300001-365536
Holding registers	16	Read/Write	400001-465536
Table 2 — Modern decimal representation of Modbus registers using a 6-digit addressing scheme.			

2.2 Modbus Function Codes

In order to act upon the data within the Modbus memory blocks, the Modbus protocol defines a set of function codes, although not all function codes are supported by all Modbus devices. The Anybus BACnet to Modbus Gateway supports the function codes listed in Table 3, which are adequate for accessing common Modbus devices. If a Modbus slave device does not support multiple register reads, the Anybus BACnet to Modbus Gateway will communicate with the device using single register reads.

Function Code	Description
1	Read coils
2	Read discrete inputs
3	Read holding registers
4	Read input registers
5	Write single coil (available but not used)
6	Write single register (available but not used)
15	Write multiple coils
16	Write multiple registers

Table 3 — The Anybus BACnet to Modbus Gateway supports the most common Modbus

3 Assigning Modbus Registers to BACnet Objects

When considering the four types of Modbus memory blocks and understanding the available BACnet objects, it becomes obvious that Modbus discrete inputs should be assigned to BACnet binary inputs (BI) and Modbus input registers should be assigned to BACnet analog inputs (AI). The current value of both types of objects can only be read.

Modbus coils should be assigned to BACnet binary outputs (BO). Modbus coils, as well as BACnet binary outputs, can be read or written. If a Modbus coil is to be blocked from being written via the Modbus interface, it is best to treat this Modbus point as a BACnet BI, and not as a BO. Holding registers can be inputs or outputs, so assigning them as analog outputs (AO) makes sense as they can be read or written. However, if it is obvious that a holding register is to be treated as an input only, it should be assigned to a BACnet AI instead of an AO.

When assigning Modbus registers or pointing to a BACnet object, the significance of the variable needs to be understood. HMS Industrial Networks does its best to interpret the Modbus register tables provided by the equipment supplier. If a different interpretation is required, it is possible to modify the Modbus device profile accordingly.

3.1 BACnet Object Properties

From the above discussion, only four BACnet object types have been selected (apart from the required device object). The next step is to determine the required properties for these four objects. The abbreviated conformance code table (below) shows which properties are required, and which are optional.

BACnet Property	BI	BO	AI	AO
Object Identifier	R	R	R	R
Object Name	R	R	R	R
Object Type	R	R	R	R
Present Value	R	W	R	W
Status Flags	R	R	R	R
Event State	R	R	R	R
Out of Service	R	R	R	R
Polarity	R	R	NA	NA
Units	NA	NA	R	R
Priority Array	NA	R	NA	R
Relinquish Default	NA	R	NA	R
COV Increment	NA	NA	O	O

Table 4 — Abbreviated Conformance Code table

O = optional

R = required to be read

W = required being read and written

NA = not applicable to this object

Within a single BACnet device there can be several object types, such as BI, BO; AI; etc. However, the Object Identifier for any of these types within the device must be unique. Likewise, the Object Names within the device must be unique. The Object Type will be set accordingly, to represent the Modbus register as BI, BO, AI or AO. The Present Value represents the actual value of the point being read or written to by the Anybus BACnet to Modbus Gateway. For BIs and BOs, the BACnet variable type BOOLEAN is supported. For AIs and AOs, the BACnet variable type REAL is supported, which follows IEEE floating point convention.

Status Flags, Event State and Out of Service do not require configuration, as they are handled by the Anybus BACnet to Modbus Gateway. Polarity is a required property for binary points. All polarities are preset to NORMAL, so as to not invert the state of a Modbus coil or discrete input. Units of Measure for analog points need to be identified, but it is possible to indicate “no units”. An attempt will be made to assign a BACnet unit to an equivalent Modbus unit; otherwise “no units” will be inserted.

For output points, the Priority Array and Relinquish Default must be set. Priority Array is set when the Present Value is written. The Present Value assumes the value of the Relinquish Default when there is a null Priority Array. In other words, the Anybus BACnet to Modbus Gateway follows the BACnet rules for prioritizing the writing of outputs. A Modbus output will only be changed when receiving a command with a higher priority than that already written for a particular output.

The Anybus BACnet to Modbus Gateway supports BACnet Change of Value (COV), so the COV increment must be set on analog points that will be subscribed. It is not necessary to set an increment on a binary point. The Anybus BACnet to Modbus Gateway can support up to 100 binary COV subscriptions and 100 analog COV subscriptions.

The first issue is to create a unique Object Instance for each Modbus register in the device profile. It was decided to simply assign sequential numbers beginning with 1. This is fine for 1-bit registers and single 16-bit word registers, but if the data is represented as a double-word (32-bits) then the first register is assigned a BACnet object instance and the second register is skipped. Sequential numbering will then continue with the register following the skipped location. If the user decides to modify an object instance, care should be exercised to ensure that object instance uniqueness is maintained.

There is a special case called ‘bit-picking’ where a single 16-bit word register represents 1-16 discrete points. This register must then be mapped to individual BI and BO points. Therefore, instead of having a single object instance for one Modbus 16-bit register, up to 16 sequential object instances are created, depending upon the number of discrete points.

The next issue is the Object Name, which must be unique. Any unique name will work, but it is recommended to use the register description for the corresponding Modbus register. For example, if register 400001 means the “Voltage phase A to neutral,” then that will be the name. However, this exact name cannot be used anywhere within the device profile. Loading two or more identical device profiles into the same Anybus BACnet to Modbus Gateway is not an issue, as each device profile corresponds to different devices, thus meeting the uniqueness constraint. However, to guard against the user modifying an object name already in use within a device, the Anybus BACnet to Modbus Gateway will save the change with an appended number, to maintain uniqueness.

3.2 Creating Device Objects

Each BACnet device (virtual or real) must have one, and only one device object. The Anybus BACnet to Modbus Gateway is a real BACnet device and must therefore be represented as a device object. Each attached Modbus device is considered a virtual device and must therefore also be represented by a device object. The properties for a device object are much different than for a BI, BO, AI or AO and most of the required properties are pre-configured by Contemporary Controls and not accessible for the user. However, there are two properties that must be assigned for each Modbus device.

The Device Object Identifier must be unique BACnet internetwork-wide, therefore the convention used was to take Contemporary Controls’ BACnet vendor ID (245), append a “0” and then append the 3-digit Modbus slave address. For example, if the Modbus slave address is 99, then the Device Object Identifier would be 2450099. This does not guarantee uniqueness BACnet internetwork-wide so it must be checked against all other devices in the network.

The Device Object Name must also be unique within the BACnet internetwork. The user can assign a name that has significance to the Modbus device being connected. The Anybus BACnet to Modbus Gateway provides the default name “Modbus device name”, and to ensure uniqueness appends the Device Object Identifier as shown above. For example, the default Device Object Name for the Device Object Identifier above would be “Default Modbus Name 2450099”.

4 Developing a Modbus Device Profile

The development of a Modbus Device Profile begins with the manufacturer’s product information in the form of a Modbus Register Table. Note that in the following example the registers in Table 5 are all holding registers with the start location 40001. Although these are 5-digit references this is not a problem. Modbus uses offset base+1 addressing, but the actual data is stored in a base+0 address as shown. Although read/ write holding registers are indicated by a 4xxxx reference, most of the registers are actually inputs, so they will be mapped as AIs in BACnet. Please note register 40016, which must be expanded as it actually represents a collection of status bits that will be mapped to BIs in BACnet. The detailed data can be found in Table 6.

4.1.1 Example Modbus Register Table for an Instrument

Register Address	Modbus Address	Data Type	Scaling	Comment
0x00	40001	Flow in Eng unit (low)	No	Mass flow in selected
0x01	40002	Flow in Eng unit (high)	No	
0x02	40003	Total (low)	No	Total in selected unit
0x03	40004	Total (High)	No	
0x04	40005	Temperature (low)	*10	Temperature in selected
0x05	40006	Temperature (high)	*10	
0x06	40007	Elapsed time (low)	*10	Elapsed time in hours *
0x07	40008	Elapsed time (high)	*10	
0x08	40009	Velocity (Low)	No	Velocity in nm/hr
0x09	40010	Velocity (high)	No	
0x0A	40011	Flow in Eng unit * 10	10	Mass flow in selected
0x0B	40012	Flow in Eng unit *100	100	Mass flow in selected
0x0C	40013	Total *100	100	Total in selected unit *
0x0D	40014	Total2 (low, 2 gas curves only)	No	Total #2 for 2 gas
0x0E	40015	Total2 (high, 2 gas curves only)	No	Total #2 for 2 gas
0x0F	40016	Status	No	Status
0x10	40017	Spare/ Not used		
0x11	40018	Spare/ Not used		
0x12	40019	Spare/ Not used		
0x13	40020	Flow in Eng Unit (float, upper 16 bits)	No	Mass flow in selected
0x14	40021	Flow in Eng Unit (float , lower 16 bits)	No	Mass flow in selected
0x15	40022	Total in Eng Unit (float, upper 16 bits)	No	Total in selected unit
0x16	40023	Total in Eng Unit (float, lower 16 bits)	No	Total in selected unit
0x17	40024	Total#2 for 2 gas curve (float, upper 16 bits)	No	Total in selected unit
0x18	40025	Total#2 for 2 gas curve (float, lower 16 bits)	No	Total in selected unit
0x19	40026	Temp in selected unit (float, upper 16 bits)	No	Temperature in selected
0x1A	40027	Temp in selected unit (float, lower 16 bits)	No	Temperature in selected
0x1B	40028	Elapsed time in hours (float, upper 16 bits)	No	Elapsed time in hours
0x1C	40029	Elapsed time in hours (float, lower 16 bits)	No	Elapsed time in hours
0x1D	40030	Velocity in selected unit (float, upper 16 bits)	No	Velocity in selected unit
0x1E	40031	Velocity in selected unit (float, lower 16 bits)	No	Velocity in selected unit
0x1F	40032	Spare/ Not used		
0x20	40033	Spare/ Not used		
0x21	40034	Spare/ Not used		
0x22	40035	Spare/ Not used		
0x23	40036	Spare/ Not used		

Table 5 — Modbus register table from an equipment vendor.

The following is an expansion of register 40016. The complete register must be represented as 16 binary inputs.

Bit	Definition	Comment
0	Power up indication	Reset when out of the power up sequence
1	Flow rate reached high limit threshold	Set limit to zero to disable
2	Flow rate reached low limit threshold	Set limit to zero to disable
3	Temperature reached high limit threshold	Set limit to zero to disable
4	Temperature reached low limit threshold	Set limit to zero to disable
5	Sensor reading is out of range	Check sensor wiring
6	Velocity flow rate outside of calibration table	Check sensor wiring
7	Incorrect Settings	Check settings
8	In simulation mode	Set simulation value to 0 to disable
9	Frequency output is out of range	Check frequency output settings
10	Analog 4-20 mA for flow is out of range	Check analog output settings
11	Analog 4-20 mA for temperature is out of range	Check analog output settings
12	Anybus error	Check wiring from RS485 to Anybus IC
13	RTC error (only for FT2 with RTC)	Check RTC
14	CRC error	Check parameters and reset CRC
15	Tot Error	Reset total

Table 6 — Details of register 40016 as a collection of status bits.

Once all the Modbus points are determined and there is an understanding of how they are to be mapped to BACnet objects, the Modbus Device Profile can be generated using the Anybus Profile Builder (an Excel spreadsheet).

The Modbus Device Profile is actually a Comma Delimited Variable (CSV) file supported by Excel. It can also be viewed and edited with the **Anybus Profile Builder**, or with a simple text editor such as Microsoft Notepad. Each record ends with a line feed (LF). There is no start-of-record character. Care must be taken to not corrupt the file by inserting invalid characters.

See section 6 to see a Modbus Device Profile for the above device as viewed in a spreadsheet.

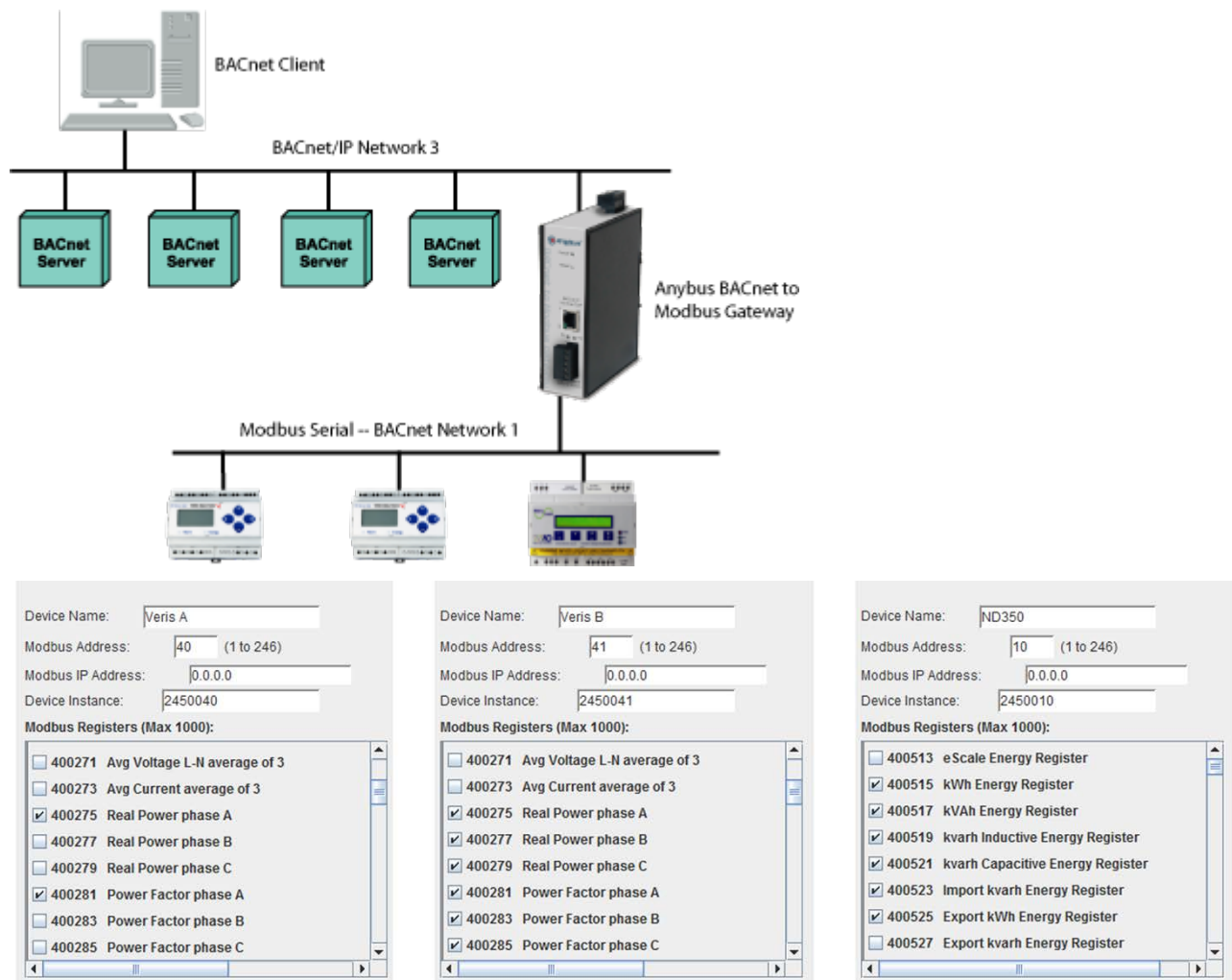
5 Virtual Routing

The Anybus BACnet to Modbus Gateway incorporates a feature called virtual routing, which allows each attached Modbus device to appear as a BACnet device. In the figure below, there is one BACnet client communicating with BACnet/IP server devices and an Anybus BACnet to Modbus Gateway. Each of these devices, including the Gateway, will have a unique device instance. The Anybus BACnet to Modbus Gateway complies with the BACnet device profile B-ASC, which means that it is an application-specific controller functioning as a server.

Connected to the RS-485 port on the Anybus BACnet to Modbus Gateway are three Modbus RTU energy meters. Two of these are identical Veris meters, and the third is a Northern Design meter. Virtual routing makes each of these meters appear as BACnet devices with unique device instances.

Each will inherit the same characteristics as the Anybus BACnet to Modbus Gateway, including the B-ASC device profile. Note that these three devices are assigned to BACnet network 1 because they are considered to be residing on a separate network, in this case a virtual network. The Anybus BACnet to Modbus Gateway resides on network 3.

Network 1 contains three device profiles representing three Modbus devices. The first two are the same, as they represent the same device, although not all points have been enabled for scanning. The third device profile is for a completely different meter. Note that the device names and device instances are different and must be unique for the entire BACnet internetwork. The Modbus slave addresses are shown. As these devices are not Modbus TCP devices, there will be no Modbus IP address assignment.



6 A Modbus Device Profile as Viewed in a Spreadsheet

	A	B	C	D	E	F
1			Fox_Thermal_Instuments_Model_FT2			
2	YES	0	Flow in Eng unit	10	X400001	32 Bit Unsigned Integer LO/HI
3	YES	0	Total	10	X400003	32 Bit Unsigned Integer LO/HI
4	YES	0	Temperature	10	X400005	32 Bit Unsigned Integer LO/HI
5	YES	0	Elapsed time	10	X400007	32 Bit Unsigned Integer LO/HI
6	YES	0	Velocity	10	X400009	32 Bit Unsigned Integer LO/HI
7	YES	0	Flow in Eng unit * 10	10	X400011	16 Bit Unsigned Integer
8	YES	0	Flow in Eng unit *100	10	X400012	16 Bit Unsigned Integer
9	YES	0	Total *100	10	X400013	16 Bit Unsigned Integer
10	YES	0	Total2	10	X400014	32 Bit Unsigned Integer LO/HI
11	YES	0	Status Power up indication	10	X400016	Bit16
12	YES	0	Status Flow rate reached high limit threshold	10	X400016	Bit16
13	YES	0	Status Flow rate reached high limit threshold	10	X400016	Bit16
14	YES	0	Status Temperature reached high limit threshold	10	X400016	Bit16
15	YES	0	Status Temperature reached low limit threshold	10	X400016	Bit16
16	YES	0	Status Sensor reading is out of range	10	X400016	Bit16
17	YES	0	Status Velocity flow rate outside of calibration table	10	X400016	Bit16
18	YES	0	Status Incorrect Settings	10	X400016	Bit16
19	YES	0	Status In simulation mode	10	X400016	Bit16
20	YES	0	Status Frequency output is out of range	10	X400016	Bit16
21	YES	0	Status Analog 4-20 mA for flow is out of range	10	X400016	Bit16
22	YES	0	Status Analog 4-20 mA for temperature is out of range	10	X400016	Bit16
23	YES	0	Status Anybus error	10	X400016	Bit16
24	YES	0	Status RTC error (only for FT2 with RTC)	10	X400016	Bit16
25	YES	0	Status CRC error	10	X400016	Bit16
26	YES	0	Status Tot Error	10	X400016	Bit16
27	YES	0	Flow in Eng Unit	10	X400020	Floating Point HI/LO
28	YES	0	Total in Eng Unit	10	X400022	Floating Point HI/LO
29	YES	0	Total#2 for 2 gas curve	10	X400024	Floating Point HI/LO
30	YES	0	Temperature in selected unit	10	X400026	Floating Point HI/LO
31	YES	0	Elapsed time in hours	10	X400028	Floating Point HI/LO
32	YES	0	Velocity in selected unit	10	X400030	Floating Point HI/LO

	G	H	I	J	K	L	M	N	O	P	Q
1											
2	ANALOG_INPUT	1	NO_UNITS	Not Defined	None	0	1	0	1	1	0
3	ANALOG_INPUT	2	NO_UNITS	Not Defined	None	0	1	0	1	1	0
4	ANALOG_INPUT	3	NO_UNITS	Not Defined	None	0	1	0	0.1	1	0
5	ANALOG_INPUT	4	NO_UNITS	Not Defined	None	0	1	0	0.1	1	0
6	ANALOG_INPUT	5	NO_UNITS	Not Defined	None	0	1	0	1	1	0
7	ANALOG_INPUT	6	NO_UNITS	Not Defined	None	0	1	0	1	1	0
8	ANALOG_INPUT	7	NO_UNITS	Not Defined	None	0	1	0	1	1	0
9	ANALOG_INPUT	8	NO_UNITS	Not Defined	None	0	1	0	1	1	0
10	ANALOG_INPUT	9	NO_UNITS	Not Defined	None	0	1	0	1	1	0
11	BINARY_INPUT	10	NO_UNITS	Not Defined	0	0	1	0	1	1	0
12	BINARY_INPUT	11	NO_UNITS	Not Defined	1	0	1	0	1	1	0
13	BINARY_INPUT	12	NO_UNITS	Not Defined	2	0	1	0	1	1	0
14	BINARY_INPUT	13	NO_UNITS	Not Defined	3	0	1	0	1	1	0
15	BINARY_INPUT	14	NO_UNITS	Not Defined	4	0	1	0	1	1	0
16	BINARY_INPUT	15	NO_UNITS	Not Defined	5	0	1	0	1	1	0
17	BINARY_INPUT	16	NO_UNITS	Not Defined	6	0	1	0	1	1	0
18	BINARY_INPUT	17	NO_UNITS	Not Defined	7	0	1	0	1	1	0
19	BINARY_INPUT	18	NO_UNITS	Not Defined	8	0	1	0	1	1	0
20	BINARY_INPUT	19	NO_UNITS	Not Defined	9	0	1	0	1	1	0
21	BINARY_INPUT	20	NO_UNITS	Not Defined	10	0	1	0	1	1	0
22	BINARY_INPUT	21	NO_UNITS	Not Defined	11	0	1	0	1	1	0
23	BINARY_INPUT	22	NO_UNITS	Not Defined	12	0	1	0	1	1	0
24	BINARY_INPUT	23	NO_UNITS	Not Defined	13	0	1	0	1	1	0
25	BINARY_INPUT	24	NO_UNITS	Not Defined	14	0	1	0	1	1	0
26	BINARY_INPUT	25	NO_UNITS	Not Defined	15	0	1	0	1	1	0
27	ANALOG_INPUT	26	NO_UNITS	Not Defined	None	0	1	0	1	1	0
28	ANALOG_INPUT	27	NO_UNITS	Not Defined	None	0	1	0	1	1	0
29	ANALOG_INPUT	28	NO_UNITS	Not Defined	None	0	1	0	1	1	0
30	ANALOG_INPUT	29	NO_UNITS	Not Defined	None	0	1	0	1	1	0
31	ANALOG_INPUT	30	NO_UNITS	Not Defined	None	0	1	0	1	1	0
32	ANALOG_INPUT	31	NO_UNITS	Not Defined	None	0	1	0	1	1	0

Cell A1 contains the file name that will appear in the Anybus BACnet to Modbus Gateway drop list box. A maximum of 40 characters can be used, which can be alphanumeric character and the _ (underscore) character. Spaces are not allowed. All other cells in row 1 must be blank.

Example: Anybus_Test

The remainder of the fields in row A should be left blank, as they are reserved for future use.

Rows 2 and beyond contain information in columns A through R. Data fields may not contain spaces or non-printable characters. Additionally, the following characters are not allowed: & : ‘ “

Column A (Poll YES/NO)

This value represents the polling flag and contains the word YES or NO, which are default conditions. The YES will be interpreted by the Anybus BACnet to Modbus Gateway as a checked box to poll for the variable, while NO is an unchecked box indicating that polling will not occur.

Column B (BACnet Object Name)

This must be unique and contain 1-64 characters, with no commas. Note that the name will not be checked in the profile builder, but any duplicates will be detected when running the verification tool.

Column C (Modbus Variable Type)

This contains the Modbus data type, which is defined by an underscore followed by the Modbus data type (0, 1, 3 or 4), followed by a second underscore and a description. For example: _4_Holding_Register.

Column D (1 to 65535)

This column contains the Modbus register address

Column E (Data format)

This is the data format for the Modbus register. The following formats are available:

- 16-Bit Unsigned Integer — 16 bits read as 0 to 65,535.
- 16-Bit Signed Integer — 16 bits read as -32,768 to 32,767.
- 32-Bit Unsigned Integer LO/HI — 32 bits read as 0 to 4,294,967,295, with the least significant word at the base address.
- 32-Bit Signed Integer LO/HI — 32 bits read as -2,147,483,648 to 2,147,483,647, with the least significant word at the base address.
- 32-Bit Unsigned Integer HI/LO — 32 bits read as 0 to 4,294,967,295, with the least significant word at the base address + 1.
- 32-Bit Signed Integer HI/LO — 32 bits read as -2,147,483,648 to 2,147,483,647, with the least significant word at the base address + 1.
- Floating Point LO/HI — 32 bits read as -3.4028235E+38 to 3.4028235E+38 with the least significant data at the base address.
- Floating Point HI/LO — 32 bits read as -3.4028235E+38 to 3.4028235E+38, with the least significant data at the base address +1.
- Bit16 — 16 bits read as 0 or 1 per bit, Bit number is specified in column K
- Bit32 LO/HI — 32 bits read as 0 or 1 per bit, Bit number is specified in column K with the least significant data at the base address.
- Bit32 HI/LO — 32 bits read as 0 or 1 per bit, Bit number is specified in column K with the least significant data at the base address +1.

Column F (Bit#).

Typically “None”, this is for references to 0aaaaa and 1aaaaa and for the bit picking of 3aaaaa or 4aaaaa references. A 6–digit addressing scheme is used, with “aaaaa” representing an address in the range 1-65536.

If a bit pick is to occur, a value 0-15 (16 Bit), or 0-31 (32 Bit), is specified, with bit 0 being the least significant.

Columns G, H, I and J

These are for scaling of the incoming Modbus value. G and H represent two points (X1 & X2) on the Modbus input scale, while I and J represent two equivalent points (Y1 & Y2) of the scaled result that will be presented to BACnet. A linear relationship is assumed between the input values and the output values.

For example; the range of values in the Modbus register is 0-4800, representing a voltage, but the BACnet device should read 0-480.0. Set the value in Column G to 0, Column H to 1, Column I to 0 and Column J to .1 to scale the value. Any two points along the Modbus scale can be used while entering the equivalent BACnet values.

Column K (Read-only or R_W)

Determines if the value is read-only or also writeable.

Column L (BACnet object type)

The type is automatically selected according to the Modbus Variable type selected in Column C.

- BINARY_INPUT — for use with all 1aaaaa, 0aaaaa that will be read-only, bit-pick of all 3aaaaa, and bit-pick of 4aaaaa that are read-only.
- BINARY_OUTPUT — for use with 0aaaaa that are read-write and bit-pick of 4aaaaa that are read/write.
- ANALOG_INPUT — for all 3aaaaa and 4aaaaa that are read-only.
- ANALOG_OUTPUT — for all 4aaaaa that are read/write.

Column M (BACnet Object description)

This can contain 64 characters. An optional field in BACnet, but it should not be left blank.

Column N (Change of value (COV) increment)

For a binary (on/off) variable, this is always a 0, as a change of state is always transmitted if the host has established a COV connection. For an analog variable, the value can be 0 (every change will be transmitted), or some other value, such as 10, 20.56; etc., which will represent the COV increment.

Column O (Unit group)

A pre-defined list of available units, e.g. area, currency, electrical, etc.

Column P (Unit Value)

Contains the BACnet valid unit of measure for the Unit Group. Enter **No_Units** in this column if the unit of measure is unknown, not required, or is not supported by BACnet.

Column Q (Grouping)

YES allows group reads of Modbus variables, assuming the variables are contiguous. NO disallows grouping.

Column R (Update on reconnect)

Contains YES or NO, and pertains to Modbus output registers. YES allows the last value written to a Modbus register by BACnet to be re-transmitted when a Modbus device returns from an offline to an online condition. For example, if the Modbus device was power-cycled and returned to service, the Anybus BACnet to Modbus Gateway will resume reading Modbus registers in the device. However, register outputs might not be updated to the same state when the Modbus device first went offline. If continuity of the output state of a particular output register is important after a re-connection, set this bit to YES. If the default state of the Modbus register is preferred when re-connecting the Modbus device to the Anybus BACnet to Modbus Gateway set this value to NO. There will be no attempt to resend the last value of the selected Modbus register after a re-connection. The default value for this column is NO.

7 Supported BACnet Units

The Anybus BACnet to Modbus Gateway supports all of the BACnet Engineering Units as described in ASHRAE 135-2010 clause 21.

7.1 Editing Modbus Device Profiles

Editing of Modbus Device Profiles can be done with the **Anybus Profile Builder**, available from HMS Industrial Networks. This can also be accomplished with any spreadsheet program, as long as the program can read and save a file with a .csv extension

If column widths are changed for better reading and it is desirable to retain these changes for later edits, it is best to save the work in the native spreadsheet format. However, for loading into the Anybus BACnet to Modbus Gateway, the file format **MUST** be comma-separated variable and must therefore be saved as such.

8 Using the Anybus BACnet to Modbus Gateway

The Anybus BACnet to Modbus Gateway web pages are accessible from any browser with Java version 6.0 or above. Access is password protected and instructions on installing the unit can be found in the installation guide. Once access is achieved, resident help screens are available to assist the user. These help screens are reprinted below to assist the reader in understanding how to load and use device profiles. Although the product is shipped with some device profiles, Contemporary Controls maintains a library of device profiles at www.anybus.com. Device profiles can be downloaded to the user's PC and then uploaded to the Anybus BACnet to Modbus Gateway. Once device profiles are installed, they can be configured for scanning by simply clicking boxes adjacent to the desired registers. If more significant changes need to be made to the device profile, it can be modified on the user's PC and then uploaded with a different file name.

What follows is information on how to use the Anybus BACnet to Modbus Gateway by examining the resident screens on the unit. However, the data on the screens will differ from that of the target device.

8.1 Configure Settings

Use this page to make the System, Modbus Serial and BACnet settings.

8.1.1 System

System Name: Provide a name for your system. The name is not critical.

IP Address: Changing the default value of 192.168.92.68 is recommended.

Subnet Mask: The default value of 255.255.255.0 is adequate for most users.

Gateway Address: If your Ethernet LAN has a gateway or IP router, enter its address here.

8.1.2 BACnet

Device Instance: Give the Anybus BACnet to Modbus Gateway a unique value (0–4,194,302). Default = 5000.

UDP Port: The default of 0xBAC0 (47808 in decimal) should normally not be changed.

BBMD IP Address: If the local subnet has no BBMD and the Anybus BACnet to Modbus Gateway must pass data to another subnet, it must register as a Foreign Device with a remote BBMD whose address is entered here.

BBMD Reg Time: Specify the time in seconds between successive foreign device registrations.

Virtual Network: Specify a unique network number for Modbus devices.

8.1.3 Modbus Serial

All devices on the EIA-485 bus must use the same Baud rate, Protocol and Parity.

Baudrate: Choose a value from 2,400 to 115,200. Default = 19,200.

Protocol: Choose RTU or ASCII. Default = RTU.

Parity: Specify NONE, ODD or EVEN. Default = EVEN.

Command Timeout (ms): Specify how long the Master will wait for a slave to response (50–3000). Default = 1,000ms. If a device fails to respond, it is put in a queue to be checked every Offline Poll Period.

Inter Scan Delay (ms): Specify the delay between each poll cycle (100-30,000). Default = 2000 ms.

System

System Name:	<input type="text" value="Default Configuration Name"/>
IP Address:	<input type="text" value="10.0.0.241"/>
Subnet Mask:	<input type="text" value="255.255.255.0"/>
Gateway Address:	<input type="text" value="0.0.0.0"/>

BACnet

Device Instance:	<input type="text" value="5000"/>
UDP Port:	<input type="text" value="0xBAC0"/>
BBMD IP Address:	<input type="text" value="0.0.0.0"/>
BBMD Reg Time:	<input type="text" value="100"/>
Virtual Network:	<input type="text" value="1100"/>

Modbus

Baudrate:	<input type="text" value="19200"/>
Protocol:	<input type="text" value="Modbus RTU"/>
Parity:	<input type="text" value="EVEN"/>
Command Timeout:	<input type="text" value="1000"/>
Inter Scan Delay:	<input type="text" value="100"/>
Offline Poll Period:	<input type="text" value="15000"/>
Consecutive RD Delay:	<input type="text" value="3"/>

NOTE
All Time Values in Milliseconds

After changing a System or Modbus setting, press Reboot.

Offline Poll Period (ms): Set how often (2000–30000) the Master checks to see if a slave device is back online. Default = 15000 ms.

Consecutive Access Delay (ms): Set the delay (0–1000) before back-to-back accesses of a slave. Default = 10 ms.

8.2 Mapping Configuration

Use this page to add Modbus devices along with device profiles. Each attached Modbus device must have its own device profile.

Use the **Add** button to add a device. Use the **Copy** button to clone an existing device. Use the **Modify** button to modify an existing device. Use **Delete** to remove a listed device. Double-clicking an item under Configured Devices opens a read-only Device Information window.

Configured Devices (Max 30):				1
Mapped Objects (Max 1000):				1
Modbus Address	Instance	Name	Type	
10	2450010	Watt-Node	Watt_Node_a	

[To view changes press Refresh](#)

Pressing **Add**, **Copy** or **Modify** opens a new screen in which you can name the device and set its Modbus Address. Leave the Modbus IP Address at 0.0.0.0 to disable Modbus TCP for this device. A non-zero value assumes a Modbus TCP device and not a Modbus Serial device.

Automatic Device Instance assignments: By default, 7-digit BACnet device instances are assigned automatically beginning with a '245' (Contemporary Controls' Vendor ID) followed by a region digit and a three-digit Modbus slave address (1 to 246). Modbus address 247 is reserved for Contemporary Controls. When using up to 10 Anybus BACnet to Modbus Gateway units on the same BACnet internetwork, assign a different Region Number (0–9) for each Anybus BACnet to Modbus Gateway to make auto-created Device Instances unique across the BACnet internetwork. Uncheck Device Instance Auto to manually enter a Device Instance. Any scheme may be used, as long as there are no duplicated BACnet device instance assignments. **NOTE:** If your Modbus TCP devices use duplicate Modbus slave addresses, then disable Device Instance Auto and manually assign a Device Instance.

Add Device

Name:

Modbus Address: (1 to 246) MAX Consecutive RD:

Modbus IP Address:

Device Instance Auto Region Number:

Device Instance:

Device Profile:

Modbus Registers (Max 1000):

<input checked="" type="checkbox"/>	400101	40101
<input checked="" type="checkbox"/>	400102	40102
<input checked="" type="checkbox"/>	400103	40103
<input checked="" type="checkbox"/>	400104	40104
<input checked="" type="checkbox"/>	400105	40105
<input checked="" type="checkbox"/>	400106	40106

Device Profiles: Choose a Device Profile from the drop-down menu that matches your device. If no match is found, visit www.anybus.com and download more profiles to your PC. Then upload profiles of interest to the Anybus BACnet to Modbus Gateway. After selecting a Device Profile, its Modbus Registers will fill the viewing window so that you can choose which registers to map to BACnet objects. Only those registers clicked will be scanned and then converted to BACnet objects. The maximum number of Modbus Registers per Anybus BACnet to Modbus Gateway is 1000 across all connected Modbus devices — TCP or serial. Once the configuration is completed, click **Submit** to go to the main configuration screen. **NOTE:** To see changes, press the **Refresh** button.

The **Device Profile button** is used to display or delete Device Profiles stored in the Anybus BACnet to Modbus Gateway.

Max Consecutive Read (ms): Set the number of registers (1–125) to be read as a group. Default = 1.

8.2.1 Authentication

A web browser can be used to view the web pages in the Anybus BACnet to Modbus Gateway, but authentication is required to gain access. The default username is admin and the default password is admin. It is recommended that both the username and password be changed. Only alphanumeric characters can be used for both and the length of each must be five characters or more. If it is necessary to reset the username or password, the device can be reset by depressing a recessed pushbutton on the front of the unit.

Change Username/Password

Username	<input type="text"/>
Password	<input type="password"/>
Confirm Password	<input type="password"/>
<input type="button" value="Submit"/>	

The Anybus BACnet to Modbus Gateway incorporates Java applets so Java Runtime environment (JRE) 6.0 or later must be installed on the device that hosts your web browser.

8.2.2 Mapping Status

Use this page to view the status of mapped Modbus registers to BACnet objects.

Unit Status lists all the Modbus devices that are to be polled. Those listed in green indicate that the device is online and that every register marked for polling can be accessed. Those in black mean that device cannot be reached and therefore considered offline. If the color changes from green to black frequently, this means that the device is online but a register marked for access cannot be reached. This usually indicates a faulty device profile.

Use the **Device Instance** drop-down to select a device to view or change. Its Modbus Address appears to the right. For Modbus serial devices, only the slave address is shown. For Modbus TCP devices, its IP address will precede its slave address. After choosing an Object Instance and Object Property of the device, click Read to view the Property Value. If the Write button is undimmed, you can enter a Property Value to be written. Enter a value and click the **Write** button.

CAUTION: Understand the significance of making a change to an output before executing the command.

Device Instance	Modbus Address
2450011	11
Object Instance	
1 : AI : Flow in Eng unit	
Object Property	
Present Value	
Property Value	
<input type="button" value="Read"/>	<input type="button" value="Write"/>
Unit Status	
Address 11 Dev Instance 2450011 Online	
Address 12 Dev Instance 2450012 Online	
Address 10.0.0.248/1 Dev Instance 2450001 Online	
Address 10 Dev Instance 2450010 Online	
Address 40 Dev Instance 2450246 Offline	

9 Adding Device Profiles

A library of Modbus Device Profiles can be found at www.anybus.com, where further profiles are added as they become available. To use one of these device profiles, download the (zip) file to your PC and unzip it to reveal the profile in the proper csv format, with a .csv extension. At this point the device profile can be edited, or simply uploaded to the Anybus BACnet to Modbus Gateway.

While in the Anybus BACnet to Modbus Gateway, browse for the file on your computer and, once selected, click **Upload**.

After the file is uploaded, click **Update** to see the new device profile in the Configure Modbus window.

Up to 100 device profiles can be stored within the Anybus BACnet to Modbus Gateway — plus up to 30 more selected for scanning.

Upload Profile

Select File: No file selected.

Note: click Update after you load the last file.

10 Firmware Updates

The latest Anybus BACnet to Modbus Gateway firmware can be found and downloaded from www.anybus.com. The file extension is .tgz and this should not be changed.

While in the Anybus BACnet to Modbus Gateway, browse for the file and click **Upload**.

Upload Firmware

Select File: No file selected.

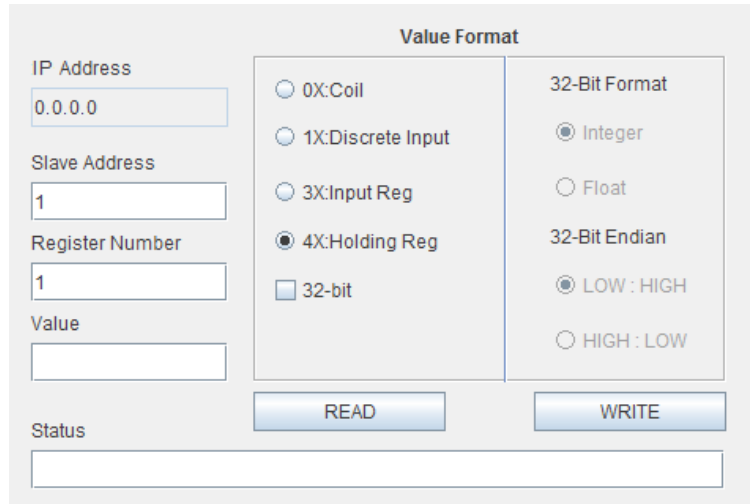
11 Modbus Utility

As a convenience to the installer, a resident Modbus Utility (Java applet) is available to verify Anybus BACnet to Modbus Gateway communication to attached Modbus serial devices. This utility is best used BEFORE entering device profiles. It is useful in confirming the data structure of Modbus registers within each device.

IP Address: (0.0.0.0 by default.) If the device is a Modbus TCP device, enter its IP address. For Modbus serial devices leave the 0.0.0.0 default address.

Slave Address: (1 by default.) Enter the slave address of the Modbus serial or Modbus TCP device to be accessed (1–246).

Register Number: (1 by default.) Enter only the 5-digit register address but not the leading 1-digit memory block address. This will be specified in the Value Format field. For example, to read register 400001, enter the register number 1 and click holding register 4X in the Value Format field. Note that PLC addressing is being used.



The screenshot shows the Modbus Utility interface. It has several input fields: IP Address (0.0.0.0), Slave Address (1), Register Number (1), and Value (empty). To the right is the 'Value Format' section with radio buttons for 0X:Coil, 1X:Discrete Input, 3X:Input Reg, and 4X:Holding Reg (selected). There is also a checkbox for 32-bit. Below these are two columns of options: '32-Bit Format' with Integer (selected) and Float, and '32-Bit Endian' with LOW:HIGH (selected) and HIGH:LOW. At the bottom are READ and WRITE buttons and a Status field.

Value Format: This reads or writes values for the register in the Register Number field. Select one of the memory blocks as follows:

- 0X: Coil: Check this if the addressed device is a Coil or Binary Output (0 or 1).
- 1X: Discrete Input: Use this to read a Binary Input (0 or 1).
- 3X: Input Reg: Use this to read a 16-bit register (raw-number format).
- 4X: Holding Reg: Use this to read or write a 16-bit register (raw-number format).
- 32-Bit: This displays a 32-bit value of a register pair (raw-number format) and enables the following four buttons:
 - Integer: Use this for a value with no fractional content.
 - Float: Use this for a single-precision floating-point value.
 - LOW: HIGH: Use this for Little Endian format (low word is in the base register and high word is in the next-higher register).
 - HIGH:LOW: Use this for Big Endian format (high word is in the base register and low word is in the next-higher register).

Click on **Read** and the result will appear in the Value field. For a write operation, enter the proper value in the Value field and click **Write**.

Status: When a register is successfully read or written, “Success” will appear here and Value shows the result. If a register access is unsuccessful, an error is reported in the Value field.