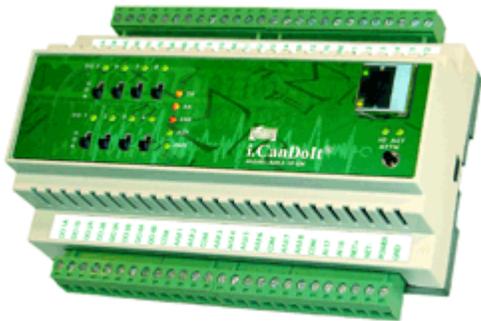


i.CanDoIt[®] Guided Tour BACnet IP Servers



Advanced i.CanDoIt[®]
*For flexible, powerful
Remote Monitoring & Control*

*Embedded Web Server
Data Logging & Trending
Time & Date Scheduling
Astronomical Clock
Email Event Notifications
Fill-in-the-blank Alarm Templates
Field Programmable
User Web Pages*



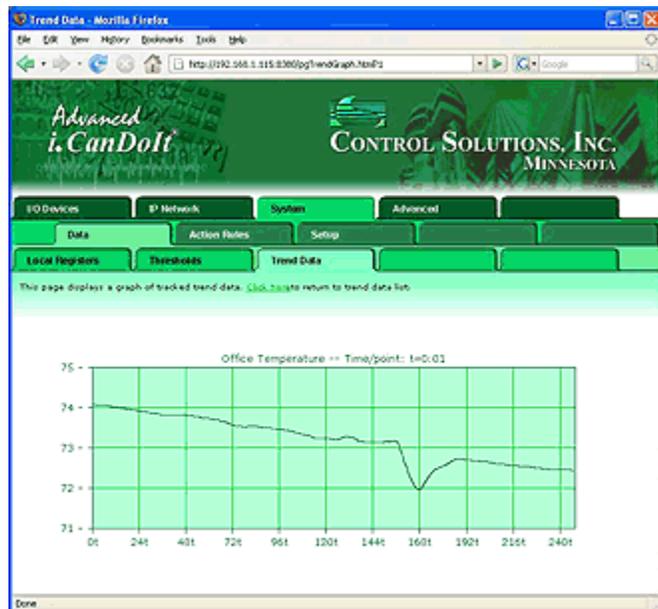
i.CanDoIt[®] Guided Tour Table of Contents

- [1. Background](#)
- [2. Device Overview](#)
- [3. I/O Configuration](#)

- [4. I/O Data and BACnet Objects](#)
- [5. Calculations and I/O Cascade](#)
- [6. Data Trending and Plotting](#)
- [7. Thresholds or Event Rules](#)
- [8. Data Logging](#)
- [9. Event Logging](#)
- [10. Email notifications](#)
- [11. Time & Date Scheduling](#)
- [12. XML Configuration Files](#)
- [13. PL/i Programming](#)
- [14. Internet Network Configuration](#)
- [15. BACnet Port Configuration](#)
- [16. BACnet IP Client](#)
- [17. Modbus/RTU Gateway](#)
- [18. User HTML/JavaScript and CGI](#)

i.CanDoIt® Guided Tour, part 1 Background

[\(return to index\)](#)



i.CanDoIt® Background

The goal of **i.CanDoIt®** is to provide a simple and cost effective facility management and remote monitoring solution suitable for use in small sites. The **i.CanDoIt®** has no site licenses or installation fees. The only software tool needed is a web browser. Multi-protocol support for BACnet, Modbus, and LonWorks are included. The **i.CanDoIt®** is programmable, even though it is rather powerful without programming.

Control Solutions pondered one particular question while doing conceptual design for **i.CanDoIt®**. Why, with the availability of advanced protocols like LonWorks, is Modbus still so popular? The answer is simple: Modbus is simple. While providing support for interface of LonWorks devices, we took a lesson from Modbus and decided that we should keep **i.CanDoIt®** simple.

Conceptually, **i.CanDoIt®** is a big spread sheet. It is register oriented just like Modbus, even though the registers are accessible in ways other than Modbus. When data is obtained from LonWorks or BACnet devices, or from local I/O, the data is placed into generic registers that are universally accessible to features of the system.

i.CanDoIt® includes data logging, time & date scheduling, and template or rule based alarm processing with email notification and/or I/O activation in response to events. Data logging simply records the contents of registers on a user specified schedule. Scheduling causes register values to be set according to a schedule. The resulting register manipulation may generate network communication with remote devices, control local I/O, or change setpoints for example.

i.CanDoIt® has two levels of programmability. It runs user programs at the server level on a 32-bit ARM processor, and (in some models) at the I/O level on a high performance 8-bit RISC processor directly connected to I/O. The programming language is **PL/i**, patterned after PL/1 at the source level, and executing byte code like Java at the execution level. The goal of PL/i was to create a compact compiler with a reasonably safe execution environment, and a syntax with some of the power of C while being simpler than C.

The register oriented spread sheet model of **i.CanDoIt®** makes it possible for users to write PL/i programs that are independent of protocol. The same program can run on a LonWorks, BACnet or Modbus network with no changes when only register references are used in the program.

The **PL/i** compiler tools are full self contained within i.CanDoIt and accessible via a browser. Programs can be compiled and sent over the BACnet, Modbus, or LonWorks network to other AddMe III devices that are not web enabled. PC based support tools for working with larger user programs, including external compiler, are free (download from this web site).

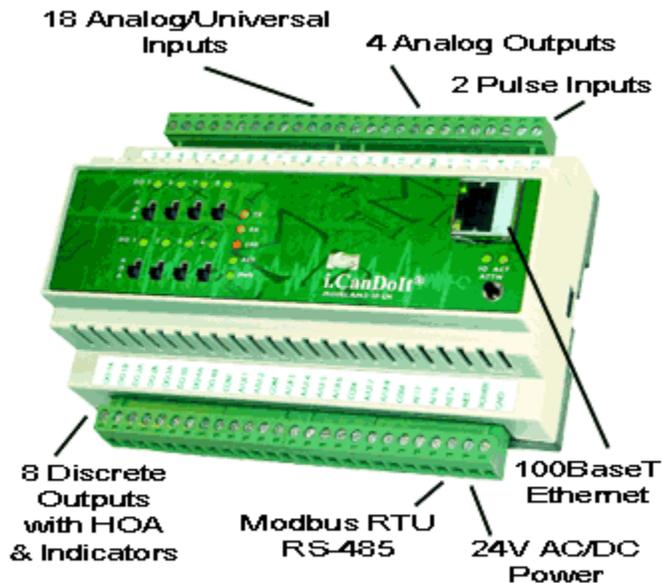
The **i.CanDoIt®** web server includes a large number of predefined web pages for configuration and monitoring. Support is also provided for user defined web pages using HTML and JavaScript. Any of the data registers are accessible to user defined web pages via internal CGI processing of user tags. This allows a fully customized look and feel for the system.

***i.CanDoIt®* Guided Tour, part 2**

Device Overview

[**\(return to index\)**](#)

The top of the line AddMe III with Advanced *i.CanDoIt®* includes 32 I/O points. AddMe Junior has 14 I/O points. The Model AM3-IP-BN is shown here, and features a BACnet IP client and server.



***i.CanDoIt®* Features a Dual Processor Architecture**

An ARM7 (32-bit RISC) processor is dedicated to server and supervisory functions, including web server, data logging, scheduling, email notification, etc. A PIC18 processor (high performance 8-bit RISC) is dedicated to hardware interface and control.

Hardware Details

- **18 Analog/universal inputs**
 - 0-10VDC, Thermistor, dry contact
 - Software selectable input type
 - 0.1% reference, up to 16-bit resolution
 - Continuously self-calibrating sigma-delta converter
- **4 Analog outputs**
 - 4-20mA (0-20mA)
 - 8-bit resolution
- **8 Discrete outputs**

- Form A relay, 5A @ 120VAC
- HOA switches & LED status indicators
- **2 Pulse/Discrete inputs**
 - TTL to 24VDC
 - Rate or totalizing count input
- **Modbus RTU Gateway Built In**
- 10/100BaseT Ethernet
- Data and event logger, email & trap notifications
- Real time scheduler including astronomical clock
- Web interface including user HTML wrapper
- User programmable using PL/i
- Powered by 18-30VDC or 24VAC 50/60 Hz
- Power Consumption: 0.3A @ 24VDC max.
- DIN rail mounting, 100mm H x 155mm W x 60mm D
- LED indicators for power, status, Tx/Rx, comm error
- Pluggable screw terminal blocks
- Operating temperature -40°C to +70°C
- Humidity 5% to 90%

i.CanDoIt® Guided Tour, part 3 I/O Configuration

[\(return to index\)](#)

There are no jumpers or physical settings required inside the device. Any configuration switching that needs to be done is handled by solid state switches. Configuration of all I/O is done through web pages found under the I/O Devices setup tabs illustrated above. Configuration of universal analog inputs is illustrated below.

This page displays analog input hardware configuration.

[Show data](#) [Show calibration helper](#) Showing 1 to 18 of 18 [Initial COV](#)

Object	Type	Scaling	Slope	Intercept	Threshold (%) / Filter	Object Name
AI 1	Analog	10k III (F)	1.000000	0.000000	0	Analog Input 1
AI 2	Analog	0-10V	1.000000	0.000000	0	Analog Input 2
AI 3	Analog	0-10V	1.000000	0.000000	0	Analog Input 3
AI 4	Analog	0-10V	1.000000	0.000000	0	Analog Input 4
AI 5	Analog	0-10V	1.000000	0.000000	0	Analog Input 5
AI 6	Analog	0-10V	1.000000	0.000000	0	Analog Input 6
AI 7	Analog	0-10V	1.000000	0.000000	0	Analog Input 7
AI 8	Analog	0-10V	1.000000	0.000000	0	Analog Input 8
AI 9	Analog	0-10V	1.000000	0.000000	0	Analog Input 9
AI 10	Fast analog	10k III (F)	1.000000	0.000000	0	Analog Input 10
AI 11	Discrete	10k II (F)	1.000000	0.000000	0	Analog Input 11
AI 12	Dry contact	3k II (F)	1.000000	0.000000	0	Analog Input 12
AI 13	Analog	20k IV (F)	1.000000	0.000000	0	Analog Input 13
AI 14	Analog	10k III (C)	1.000000	0.000000	0	Analog Input 14
AI 15	Analog	10k II (C)	1.000000	0.000000	0	Analog Input 15
AI 16	Analog	3k II (C)	1.000000	0.000000	0	Analog Input 16
AI 17	Analog	20k IV (C)	1.000000	0.000000	0	Analog Input 17
AI 18	Analog	Normal	1.000000	0.000000	0	Analog Input 18
AI 19	Analog	Inverted	1.000000	0.000000	0	Analog Input 18

Configure an **Analog Input** for temperature by selecting the appropriate thermistor type. Configure 0-10V input by selecting 0-10V, and entering a slope and intercept that corresponds to your sensor. If using a 4-20mA sensor, you will need a dropping resistor connected in parallel across the sensor input. If you wish to use an input as a simple on/off input, select discrete or dry contact.

A calibration helper page is built in. Simply enter data from readings you have taken, or data from the sensor spec sheet, then enter the desired display range. The helper will calculate slope and intercept for you. It will even take resistor values into account when a dropping resistor is used. Simply click on the calibration helper link at the top

of the Analog Inputs configuration page.

The difference between discrete and dry contact is this: With a discrete input, it is expecting you to provide external voltage excitation (up to 24VDC). With a dry contact input, it provides its own excitation, and you simply provide a contact closure to ground.

The first 16 inputs of AddMe III are fully configurable as shown above. Inputs 17 and 18 are restricted to 0-10V only (or 4-20mA with dropping resistor).

Analog Outputs only require a slope and intercept setting. This allows the output to be scaled to engineering units.

Discrete Inputs may be set to function as (1) state input, (2) frequency input, or (3) total count input. When configured for frequency, the slope and intercept will be used to scale an input to engineering units. A common requirement for this is converting tachometer input to RPM. Total count mode creates a totalizing input, and the counts are maintained through power loss.

Discrete Outputs have an optional timer associated with them. The timer can do one of two things: (1) generate a pulsed output, or (2) impose a minimum on time qualification to prevent rapid cycling.

This page displays analog input hardware configuration.

[Show data](#) [Show calibration helper](#) Showing 1 to 18 of 18 [Initial COV](#)

Object	Type	Scaling	Slope	Intercept	Threshold (%) / Filter	Object Name
BI 1	Dry contact	Inverted	1.000000	0.000000	50	Binary Input 1
BI 2	Dry contact	Inverted	1.000000	0.000000	50	Binary Input 2
BI 3	Dry contact	Inverted	1.000000	0.000000	50	Binary Input 3
BI 4	Dry contact	Inverted	1.000000	0.000000	50	Binary Input 4
AI 5	Analog	0-10V	1.000000	0.000000	0	Analog Input 5
AI 6	Analog	0-10V	1.000000	0.000000	0	Analog Input 6

Analog inputs by default are assigned to Analog Input BACnet objects. However, if reconfigured as dry contact or discrete inputs, they are automatically reassigned to Binary Input objects.

[Data Objects](#)
[IP Network](#)
[System](#)
[Advanced](#)

[Server Setup](#)
[Action Rules](#)
[I/O Setup](#)
[I/O Data](#)
[Modbus RTU](#)

[Analog Inputs](#)
[Analog Outputs](#)
[Discrete Inputs](#)
[Discrete Outputs](#)

This page displays analog output hardware configuration.

[Show data](#)
Showing 1 to 4 of 4 [Setup](#) [Update](#)

Object	Object Name	Initial COV Increment	Initial COV Period	Init. Relinq. Default	Units
AO 1	Analog Output 1	0.0	0	5.000000	no_units
AO 2	Analog Output 2	0.0	0	0.0	no_units
AO 3	Analog Output 3	0.0	0	0.0	no_units
AO 4	Analog Output 4	0.0	0	0.0	no_units

Configuration of I/O points includes additional optional BACnet IP object settings. All analog objects may have an initial COV increment. This is the COV increment that will apply by default if none is specified in the subscription by the requesting client. The initial COV increment (in seconds) is also used by default if not specified by the requesting client. The initial relinquish default value, applicable only to commandable output objects, allows setting a non-zero value for initial default for that object. Units are a setting returned to any client requesting units information for the object. The units setting does not result in any implicit data scaling (scaling may be set in I/O configuration above).

i.CanDoIt® Guided Tour, part 4 ***I/O Data and BACnet Objects***

[\(return to index\)](#)

Data may generally be viewed relative to its source in addition to a global system wide view. Data from I/O points may be viewed under the I/O Data tab.

This page displays the input levels on analog inputs.

[Show setup](#) Showing 1 to 18 of 18 [Update](#)

Object	Present Value	Object Name
AI 1	77.09000	Analog Input 1
AI 2	0.021489	Analog Input 2
AI 3	0.017342	Analog Input 3
AI 4	0.017719	Analog Input 4
AI 5	0.018473	Analog Input 5

The global view of BACnet Object data is found under the Data Objects tab. This shows data from I/O points, and includes data retrieved from remote devices via the BACnet IP client or other gateway features. It also includes data generated by user programs, or written into the register by a remote client (e.g. BACnet IP client).

The object pages provide the ability to locally set an object out of service (this can also be done via the BACnet IP network). Data values may be forced, although any forced value is potentially overwritten by the next client or I/O update. Reliability and status are indicated for each object, and these are also accessible as standard BACnet object properties via the network.

This page displays data as presently found in the local registers maintained by this device.

Showing objects from [Update](#) [< Prev](#) [Next >](#)

Object #	Object Name	Out of Service	Force	Present Value	Priority Array	Reliab	Status Flags	Device Link
AO 1	Analog Output 1	<input type="checkbox"/>	<input type="checkbox"/>	5.000000	rq> 5.000000	0	0,0,0,0	---
AO 2	Analog Output 2	<input type="checkbox"/>	<input type="checkbox"/>	0.0	rq> 0.0	0	0,0,0,0	---
AO 3	Analog Output 3	<input type="checkbox"/>	<input type="checkbox"/>	0.0	rq> 0.0	0	0,0,0,0	---
AO 4	Analog Output 4	<input type="checkbox"/>	<input type="checkbox"/>	0.0	rq> 0.0	0	0,0,0,0	---
AO 5	Analog Output 5	<input type="checkbox"/>	<input type="checkbox"/>	0.0	rq> 0.0	0	0,0,0,0	---

i.CanDoIt® Guided Tour, part 5 ***Calculations, I/O Cascade***

[\(return to index\)](#)

There is a tab called "Action Rules" under System. Some of these actions provide very simply programming without any programming. For example, you can do simple logic using the Calculate rules.

Suppose you want to turn on an output when any of several switches are tripped. The following Calculate rule will logically OR Binary Inputs 1 through 4 and place the result in Binary Output 1. We would need to configure analog inputs 1 through 4 to be discrete or dry contact, with the invert bit set appropriately so that we get a "1" when the switches are closed. Writing "1" to BO 1 (on AddMe III) will turn on relay #1.

This page allows setting up simple calculations on object data.

Showing 1 to 1 of 1 Update < Prev Next >

Rule #	Perform Operation	Using Object	And/Through	This Object	Place Result in Object
1	logic OR	BI 1	thru	BI 4	BO 1

Insert Delete

Another simply but useful action rule is the Cascade rule. Suppose you want other outputs to turn on as a result of one particular output coming on. The following example will copy BO 1 to outputs BO 2 through BO 4 any time BO 1 changes. In this example, it means when relay #1 turns on, relays 2, 3 and 4 will also come on automatically.

This page allows configuring the cascading of objects. Data from the source object is copied to the destination object(s).

Showing 1 to 1 of 1 Update < Prev Next >

Rule #	Source Object	Destination Object	Last Destination Object in Range
1	BO 1	BO 2	BO 4

There is also a rule named Constant. This rule simply writes a fixed value to a specified object one time at system startup (or re-initialization). This can be used for establishing initial setpoints, etc.

i.CanDoIt® Guided Tour, part 6

Data Trending & Plotting

(return to index)

Data trending provides two main functions: (1) Track minimum, maximum, and average, and (2) graph the trend over some recent period.

The min-max-average trending is meaningful when trending is being logged either to a data file or via HTTP Get to a remote server. At the end of each tracking period, the data is logged, and then trend values are reset. If no logging is being done, the min-max-average values will have little meaning since they are essentially "average forever" values. However, the graph is still useful without logging because it is always a sliding window of actual data samples.

Trending is set up under the System->Action Rules->Trending tab. You simply select which data point you wish to track and identify it by register number. Then allocate destination registers for the trend results. You can use the auto-allocate button for this.

This page allows setup of min/max/average tracking of selected registers.

Showing 1 to 2 of 2 Update < Prev Next >

Tracked Object	Object Name	Average Dest.	Min. Dest.	Max. Dest.	Reset	Period HH:MM	Slices	Flash
AI1	Analog Input 1	AV1	AV2	AV3	When logged	0:10	10	<input type="checkbox"/>
---	---	---	---	---	When logged	0:00	0	<input type="checkbox"/>

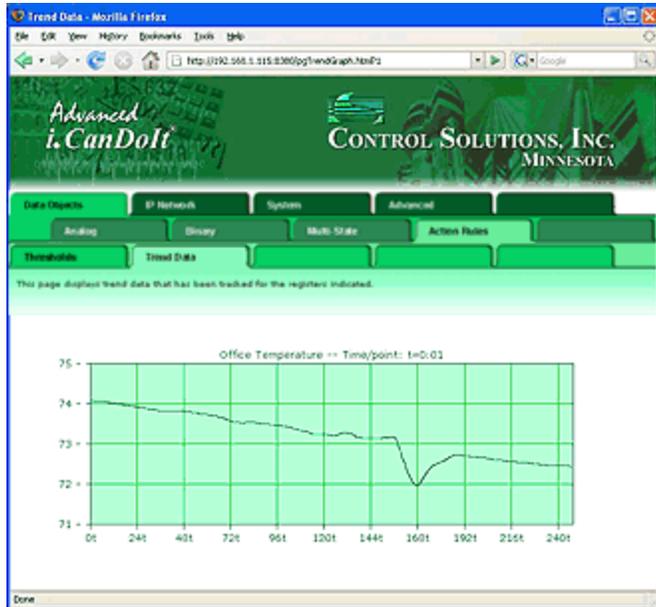
Trend data may be viewed under the Data Objects->Action Rules->Trend Data tab. The values that will be logged next if logging is enabled will appear here. You have the option of resetting the values. The illustration below shows that the trend has just been reset, in which case minimum, maximum, and average are the same value.

This page displays trend data that has been tracked for the registers indicated.

Showing 1 to 2 of 2 Update < Prev Next >

Tracked Object	Object Name	Average Value	Min. Value	Max. Value	Reset	Graph Status
AI 1	Analog Input 1	77.09000	77.09000	77.09000	<input type="checkbox"/>	0 Graph

The most useful link on the Trend Data page is the Graph link on the far right. Click on this link, and the respective register's sliding window of data samples will be displayed graphically. This virtual log file is maintained independent of any other log files you are maintaining via the Data Logging features. If you wish to send this data as a csv file attached to an email, you need to set up a true log file under the Data Logging tab.



i.CanDoIt® Guided Tour, part 7 ***Thresholds or Event Rules***

[\(return to index\)](#)

What are Threshold Rules? What are Events? Where are the Alarms?

These questions all point to essentially the same answer. You build an alarm by defining a threshold rule. When the threshold rule is triggered or activated, we call this an event. There are several things you can do with an event, and you may treat any of them as an alarm. An event may result in turning specific I/O on or off. An event may result in a notification message being sent (e.g., via email to your cell phone). An event may send a message to a central server which then decides what to do. An event may simply be logged in an event file to simply record the fact that it happened sometime.

How do I set up an Alarm?

I/O point data is placed in "registers" by easily configured I/O. Threshold "rules" determine what constitutes an event. The image below is a screen shot of a threshold rule that will result in an event when the sensor on Analog Input #1 exceeds a level of 90. Data values are scaled to any units you decide. You find the threshold rules in the System->Action Rules->Thresholds page.

Data Objects IP Network **System** Advanced

Server Setup **Action Rules** I/O Setup I/O Data Modbus RTU

Thresholds **Trending** Cascade Calculate Constants

This page displays thresholds, or rules, for defining events and assigning responses to events. Thresholds can create output based on conditional input.

Rule # Rule presently tests FALSE

Read local source object for this event named

Event is TRUE if the value is this value: this local object:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

Set local destination object as follows below while logging on-time to object

(true) To a value which is same as the source this value: from local object:

(false) Otherwise to a value which is same as the source this value: from local object:

How do I prevent spurious events when the test value is hovering around the threshold?

Hysteresis will prevent spurious events near the threshold. In the following example, the event will occur when Analog Input #1 reaches a level of 90. If AI 1 drops below 90 by a margin *less than 2*, then returns to above 90, the event will not repeat. The

value of AI 1 must drop below 88 and return to over 90 before the event will repeat.

Read local source object for this event named

Event is TRUE if the value is this value: this local object:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

How do I change event thresholds based on time of day?

Start by using a value object for the test threshold rather than fixed value as shown below:

Read local source object for this event named

Event is TRUE if the value is this value: this local object:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

Then alter the contents of this object based on schedule in the Advanced->Scheduler->Weekly Schedule page as shown below.

Data Objects | **IP Network** | **System** | **Advanced**

Scheduler | **Data Logger** | **Email Alerts** | **PL/I Programming**

Weekly Schedule | **On Demand** | **Holidays**

This page allows you to manage weekly schedules.

Showing to 1 of 1

#	S...M...T...W...T...F...S	Holidays	On Time	Off Time	Object	"On" Value	"Off" Value	Register Name
1	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>	Holidays	8:00:00	16:00:00	AV 5	85.000000	95.000000	Analog Value 5

Using the example setup shown above, the event will be generated any time Analog Input #1 exceeds a level of 85 between 8AM and 4PM, Monday through Friday. The rest of the time, Analog Input #1 needs to exceed a level of 95 before an event will be generated.

How do I control outputs based on threshold rules?

Use all of the same criteria for setting up the rule. Now add a destination object and value to write to that object.

Read local source object for this event named

Event is TRUE if the value is this value: this local object:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

Set local destination object as follows below while logging on-time to object

(true) To a value which is same as the source this value: from local object

(false) Otherwise to a value which is same as the source this value: from local object

The above example says that Binary Output #1 will turn on any time Analog Input #1

is above 90.

i.CanDoIt® Guided Tour, part 8 ***Data Logging***

[\(return to index\)](#)

Data may be logged to files in the internal memory based file system. These files may be retrieved via FTP, or delivered to you as an attachment to an email. The setup below illustrates sending a daily log file with an email, and also says that after 2 days, the old files should be discarded (to make way for new data).

This page allows you to manage data log files and logging schedules.

Data Log 1 File size: 1K Free space: 1406K/1906K Status: Error 0
File name: amlog200703241F0.txt Update < Prev Next >

File Name: FLASH0 Enable Logging Reset Error

Log Rate: 1:00:00 Log only when register # 0 is greater than zero.

New File: Daily End Time: 7:30:00 Size: 10 KB Life: 2 Days
Upon resources exhausted: Discard oldest Stop logging

Recipient: support@csimn.com
Subject: AddMelll Data Log
Message: Daily log file from Jim's test system.

[Click here for log file format information](#)

Once a file is set up, you need to tell the system what data points to log. Any object in the system may be selected for logging, and multiple points may be logged to a single file. They are recorded in CSV format for easy import into spread sheet software. Up to three different log files may be active at the same time allowing different log rates for different points. Simply click the check box in the file column for the objects that are to be logged in that file. The Report box refers to reporting this point by HTTP Get to a remote server set up to receive data logging via Get requests.

Object	Object Name	Log to File 1	Log to File 2	Log to File 3	Report
AI1	Analog Input 1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

i.CanDoIt® Guided Tour, part 9 ***Event Logging***

[\(return to index\)](#)

The event log file is set up in the same manner as data log files, and it may also be delivered via email. The two main differences with the event log is the file format (still CSV but with different data), and the source of the data logged. Data logging allows you to check off a list of data objects to log. Event logging allows you to check off any of the defined threshold rules for logging.

Rule #	Event Name	Log on True	Log on False	Report
1	High Temperature	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Verbose Log

The "High Temperature" example checked off for event logging above was set up with a threshold rule like shown below. It is not necessary to include a destination register in the threshold rule when the rule's only purpose is logging or notification.

Rule # 1 Rule presently tests FALSE

Read local source object for this event named

Event is TRUE if the value is this value: this local object:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

Set local destination object as follows below while logging on-time to object

(true) To a value which is same as the source this value: from local object:

(false) Otherwise to a value which is same as the source this value: from local object:

i.CanDoIt® Guided Tour, part 10 ***Email Notifications***

[\(return to index\)](#)

One of the useful things you can do with an event defined by a "threshold rule" is send an email to notify someone that something has happened. You may define "recipients" which can be one or more email addresses. You select or define the message, and then pick which events cause this email to be sent.

This page allows you to identify email Recipients and select the message format they will receive when Events trigger an email alert message.

Showing 1 of 3 Update < Prev Next >

Recipient: Send Test

Subject:

Default Email Message:
 This is an automated email alert originating from <system name> at <system location> at <time/date>. The event named <event name> with a test criteria of <rule> a value of <test value> tested <true/false> for register <n> <register name> with a value of <value>.

Include standard message (default or cell phone)

Cell Phone Message (160 char. max.):
 <system name>/<system location>/<register name>/<value>=<event name>

Send Status: (0 = No Error)

Recv Status:

Reset Errors

Each threshold rule that has been defined under System->Action Rules->Thresholds is listed on the email events page. Each event may be checked off for delivery to any of the recipients. You have the option of sending the email when the event first occurs (notify on true) and when the event clears (notify on false).

The email handling capability of i.CanDoIt includes the ability to receive email as well. If you would like email messages to be repeated until somebody acknowledges the event, check the "Expect Ack" box and enter a repeat time. To acknowledge the event, you only need to "Reply To" the email you received. The mail will contain a coded acknowledge string generated automatically when the email is sent. If the reply contains this expected reply string, the event is considered acknowledged.

Rule #	Event Name	Notify on True	Notify on False	Email Recip 1	Email Recip 2	Email Recip 3	Expect Ack	Repeat Time DD,HH:MM
1	High Temperature	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="text" value="0,0:00"/>

Use of email requires that you have an email account available to i.CanDoIt. You must have a user name and password for the account. Most email servers will require that

you have an email address that matches the account information. You need to configure the SMTP and POP3 server names (or IP addresses if fixed). You should also be aware that many email servers do not allow sending email from outside of their own domain. You will need to work with your ISP, or network or IT representative, to get email configured properly.

Recipients	Events	Email Server		
This page tells the system where to log in to send and receive emails.				
<hr/>				
<hr/>				
SMTP IP Address: <input type="text" value="0.0.0.0"/> <input type="checkbox"/> Use Authentication <input type="button" value="Update"/>				
SMTP Host Name: <input type="text" value="smtp.comcast.net"/> (send email) Network				
SMTP Port: <input type="text" value="25"/> SMTP Domain: <input type="text"/>				
POP3 IP Address: <input type="text" value="0.0.0.0"/>				
POP3 Host Name: <input type="text" value="mail.comcast.net"/> (receive email)				
User Name: <input type="text" value="icandoitonline"/>				
Password: <input type="password" value="XXXXXXXXXX"/>				
<hr/>				
System Name: <input type="text" value="AddMe III"/> <input type="button" value="Update"/>				
System Location: <input type="text" value="White Bear Lake, Minnesota"/>				
System Contact: <input type="text" value="icandoitonline@comcast.net"/>				
Note: System contact must be email account name to use email notification.				

***i.CanDoIt®* Guided Tour, part 11** **Time & Date Scheduling**

[\(return to index\)](#)

Weekly Scheduling using Time/Date or Astronomical Clocks

Suppose you want the lights in the parking lot to come on half hour before sunset, and go off half hour after sunrise. One way to do that is use a time clock and reset it a few times a year. The better way to do that with ***i.CanDoIt®*** is use the astronomical clock. Instead of on and off times, you simply say "dawn" and "dusk" with an optional offset.

#	S...M...T...W...T...F...S	Holidays	On Time	Off Time	Object	"On" Value	"Off" Value	Register Name
1	<input checked="" type="checkbox"/>	Holidays	DUSK-30	DAWN+30	BO 3	1.00000	0.00000	Relay 3
2	<input checked="" type="checkbox"/>	Holidays	21:00:00	7:30:00	BO 4	1.00000	0.00000	Relay 4

In this example, we set BO #3 half hour before sunset, and clear it half hour after sunrise. Our hardware configuration determined that BO #3 is allocated to Relay 3, which we have wired to control the parking lot lights. BO #4, Relay 4, shows how to do it the old way using time of day.

The above example shows two scheduled events that will happen every day of the week because all days are checked. If you wish the weekly event to only occur on Saturday, for example, you would only check the "S" box at the end of the string of boxes. For weekdays only, you would check M through F (Monday through Friday).

The astronomical clock requires knowing where in the world you are. The clock setting page found at System->Setup->Time & Date includes a section for setting location.

Astronomical Clock

Latitude: Longitude: Time Zone:

Sunrise: **07:09** Sunset: **19:29**

Once the internal battery backed clock and calendar have been set, and you have entered latitude, longitude, and time zone, the clock setting page will show you when the system has calculated sunrise and sunset to be at this location.

On Demand Scheduling using Time/Date or Astronomical Clocks

Sometimes it is desirable to schedule something to happen just once at a specific time and date. This is done with "on demand" scheduling. You simply specify a starting time and date, and ending time and date, and the given register will be set accordingly during that time.

#	On Time	On Date	Off Time	Off Date	Object	"On" Value	"Off" Value	Register Name
1	8:00:00	03/24/2007	17:00:00	03/24/2007	BO 6	1.00000	0.0	Parking Lot Gate
2	8:00:00	03/31/2007	17:00:00	03/31/2007	BO 6	1.00000	0.0	Parking Lot Gate

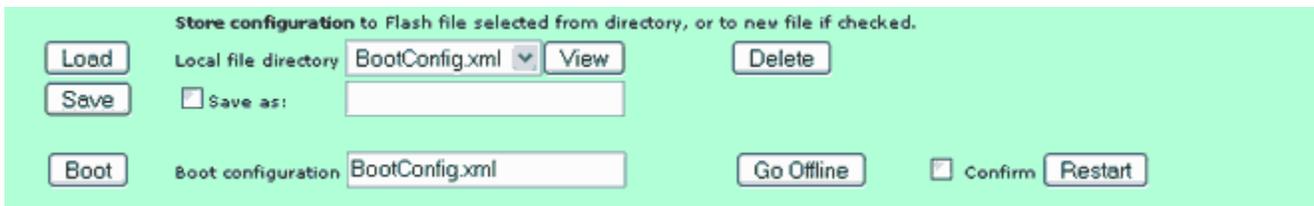
In this example, we are opening the parking lot gate on two consecutive dates. Our parking lot is normally closed on weekends, but we are sponsoring a special event requiring the use of our parking lot on two consecutive Saturdays. Fill in your own favorite example here.

i.CanDoIt® Guided Tour, part 12 ***XML Configuration Files***

[\(return to index\)](#)

Most configuration information is stored in an XML file in the device's internal Flash file system. The only exceptions are things like the device's IP address, the name of the XML file from which to load configuration at boot-up, and things needed to get as far as loading that file in the first place.

Since configuration is stored in XML format, it is easily ported from one system to another. It is also easy to review the configuration - simply click on the file name, and your browser will display XML. You load a configuration file from your PC, or save it back to your PC.



Store configuration to Flash file selected from directory, or to new file if checked.

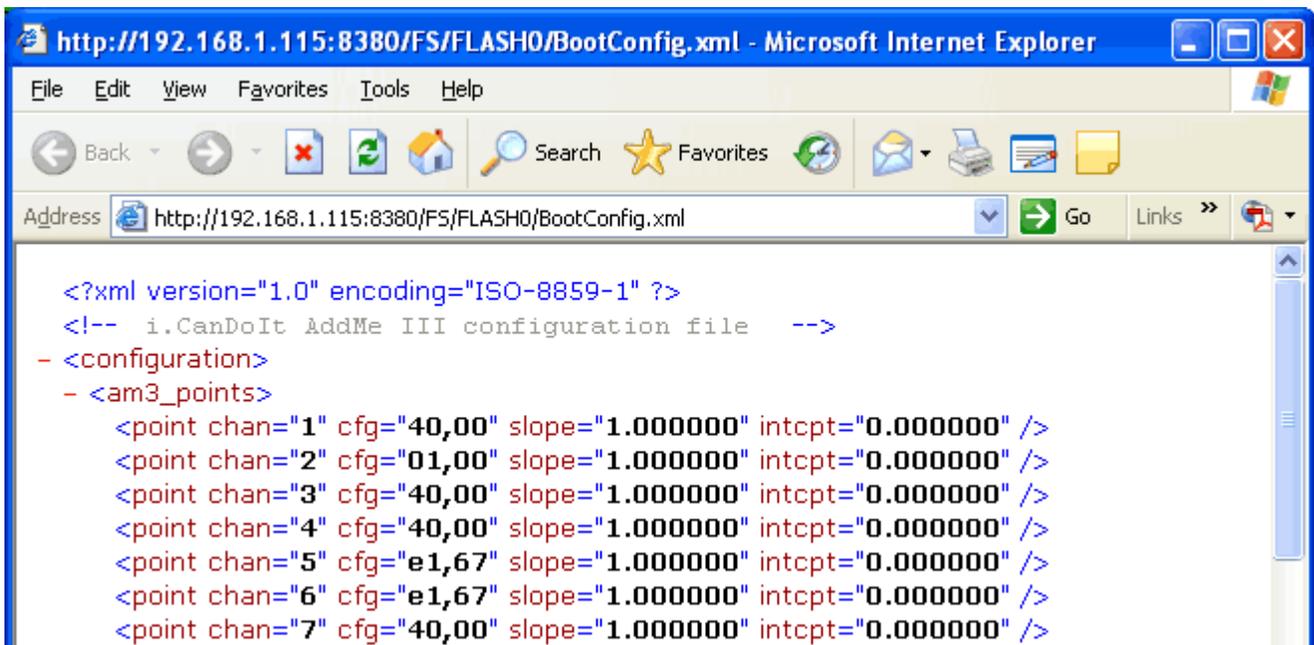
Local file directory:

Save as:

Boot configuration: Confirm

Buttons: Load, Save, Boot

The first few lines of XML viewed in the browser look like this...



http://192.168.1.115:8380/FS/FLASH0/BootConfig.xml - Microsoft Internet Explorer

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- i.CanDoIt AddMe III configuration file -->
- <configuration>
- <am3_points>
  <point chan="1" cfg="40,00" slope="1.000000" intcpt="0.000000" />
  <point chan="2" cfg="01,00" slope="1.000000" intcpt="0.000000" />
  <point chan="3" cfg="40,00" slope="1.000000" intcpt="0.000000" />
  <point chan="4" cfg="40,00" slope="1.000000" intcpt="0.000000" />
  <point chan="5" cfg="e1,67" slope="1.000000" intcpt="0.000000" />
  <point chan="6" cfg="e1,67" slope="1.000000" intcpt="0.000000" />
  <point chan="7" cfg="40,00" slope="1.000000" intcpt="0.000000" />
```

i.CanDoIt® Guided Tour, part 13 ***PL/i Programming***

[\(return to index\)](#)

The PL/i programming language is simpler than C but more structured than Basic at the source code level. At the execution level, AddMe III runs on byte code similar to Java. The goal of PL/i was to create a compiler that generated code which was compact, efficient, and safer than C.

The AddMe III architecture is register based. Therefore a carefully written program will be portable to systems running on Modbus, LonWorks or BACnet networks with equivalent results.

A summary of PL/i syntax and grammar can be found in the help pages in the device's internal web site under the Advanced->PL/i Programming tabs. [A more in depth tutorial is available here.](#)

What does PL/i look like?

Most of any given program will look a lot like Basic. Referring to subroutines as "procedures" is borrowed from older languages like Pascal. The block structure is common to many languages, and setting the blocks apart with "begin..end" instead of {} is also borrowed from Pascal. In fact, if you are a Pascal programmer, you may recognize several similarities. We hope you won't miss ":=".

The simple test program shown here will ramp an analog output on AddMe III up and down from zero to near full scale. It got named "dimmer" because if we connect an LED across the output, we have an LED dimmer. This makes a good "my first program" exercise.

```
program dimmer

declare
  myVar: int;

procedure turnon (setting: int)
declare
  level: int;
begin
  level = setting * 10;
  seti (1001, level);
  delay (5);
end;

begin
  while TRUE do
  begin
```

```
    for myVar = 0 to 20 turnon(myVar);  
    for myVar = 20 down to 0 turnon(myVar);  
end;  
end
```

i.CanDoIt® Guided Tour, part 14 ***Network Configuration***

[\(return to index\)](#)

If you spend much time looking at this page, you're a network administrator - either by training or by default because you had to. If you're not the network administrator, this is the one page you do want that person to look at on your behalf if you are going to use any of the more advanced network features.

The only thing most people need to think about on this page is simply setting the IP address of the device. It comes defaulted to 10.0.0.101. The two most common domains for DSL and cable routers to be shipped with by default are 10.0.0.0 and 192.168.1.0, so we had only half a chance of picking the right one for you. Refer to the quick start guide for tips on getting up and running for the first time. (Additional help is shown at the bottom of each page in the server.)

This page allows you to change this device's IP address, and optionally set other network parameters including DNS.

IP Address	<input type="text" value="192.168.1.151"/>	192.168.1.151	<input type="button" value="Review IP"/>	Email Server
Subnet Mask	<input type="text" value="255.255.255.0"/>	255.255.255.0	<input type="button" value="Change IP"/>	
Gateway	<input type="text" value="192.168.1.1"/>	192.168.1.1		
Static DNS1	<input type="text" value="209.193.72.2"/>	209.193.72.2	<input type="button" value="Apply DNS"/>	<input type="button" value="Reset DNS"/>
Static DNS2	<input type="text" value="209.193.68.2"/>	209.193.68.2		
Dynamic DNS Service	<input type="text" value="Custom DynDNS"/>	71.195.57.89	DDNS status: No update needed.	
Host Name	<input type="text" value="icandoit.deviceisp.net"/>			
DDNS User Name	<input type="text" value="icandoit"/>	Password	<input type="password" value="AAAAAAAA"/>	
HTTP Port	<input type="text" value="8151"/> (default 80)	<input type="checkbox"/> Disallow HTTP Query	<input type="button" value="Set Ports"/>	
Telnet Port	<input type="text" value="23"/> (default 23)			
Client Update Host	<input type="text"/>	<input type="checkbox"/> Enable	<input type="button" value="Update"/>	
Log Page	<input type="text"/>			
Log Parameters	<input type="text"/> (optional)			
Configuration Page	<input type="text"/>			
Update Timeout	<input type="text" value="0"/>	Client ID	<input type="text" value="0"/>	

i.CanDoIt[®] Guided Tour, part 15 ***BACnet Port Configuration***

[**\(return to index\)**](#)

The Local Host settings such as IP address of this device are set on the Local Host page. Additional settings are required for BACnet IP. Those settings are shown on this page. Device description is returned as a BACnet device property. Port number and device instance are used to access this device along with IP address.

This page displays BACnet IP port settings.

[Refresh](#)

BACnet IP Settings:

Device Instance	<input type="text" value="151"/>	Save	Local Network Settings
Port (default 0xBAC0 = 47808)	<input type="text" value="47808"/>		
Device Description	<input type="text" value="AddMe III i.CanDoIt BACnet Server"/>		
Device Location	<input type="text" value="White Bear Lake, Minnesota"/>		
Local Command Priority (default 10)	<input type="text" value="10"/>		

i.CanDoIt® Guided Tour, part 16 ***BACnet IP Client***

[\(return to index\)](#)

The BACnet IP Client allows this device to interact with other BACnet IP devices. Any other client may read and write any objects in the local device. However, for the local device to read and write object properties in remote devices, it is necessary to make that remote device known in the BACnet IP Client device table, and to set up a map that results in reading or writing that remote object. Once data is read from a remote device and placed into a local object for holding, any of the action rules such as event thresholds may be applied to the data obtained from the remote device. This effectively means one AddMe III or AddMe Jr may function as an alarm monitoring or scheduling device for multiple BACnet IP devices.

To identify a remote device, you need only know its device instance. Enter the device instance along with a local name that will be referenced in the maps that follow. Once the remote device has replied to a "who is", its IP address will be indicated here.

The screenshot shows a web-based configuration interface for the BACnet IP Client. The interface has a dark green header with several tabs: 'Data Objects', 'IP Network' (selected), 'System', and 'Advanced'. Under 'IP Network', there are sub-tabs: 'BACnet IP Client' (selected), 'Diagnostics', and others. Below these, there are more sub-tabs: 'Devices', 'Client Read Map' (selected), and 'Client Write Map'. A text box explains: "This page sets up the device list for remote BACnet IP devices that will be accessed for remote input and/or output (via the client read and client write maps). The local device acts as a BACnet client to the remote servers listed below." Below this is a form for configuring a device. It includes a 'Device #' field with the value '1', and buttons for 'Update', '< Prev', and 'Next >'. The main configuration area has several fields: 'Device Instance' (117), 'Local Name' (Monitoring System), 'Default Poll Period' (1.0 Seconds), 'Default Write Priority' (10), 'Reply Timeout' (1.0 Seconds), 'Timeouts' (0), and 'Who Is Reply' (192.168.1.117). There is also a 'Clear' button.

Device #	1	Update	< Prev	Next >
Device Instance	117	Local Name:	Monitoring System	
Default Poll Period	1.0 Seconds	Default Write Priority:	10	
Reply Timeout:	1.0 Seconds	Timeouts:	0	Clear
Who Is Reply	192.168.1.117			

Reading data from a remote BACnet IP device is invoked periodically by setting up a read map as illustrated below. Data is periodically read from the remote object indicated, and that data is placed in the local object indicated.

Data Objects IP Network System Advanced

BACnet IP Client Diagnostics

Devices Client Read Map Client Write Map

This page creates a map entry that reads data from a remote BACnet IP server for processing here.

Map #

Read instance # of object type at using index

Then apply scale: and offset:

Save in local object named Repeat this process every seconds.

Apply this default value: after read failure(s).

Writing remote objects via BACnet IP is invoked by a write map as illustrated here. Data is periodically (or upon sufficient delta) taken from the local object indicated, and written to the remote object indicated.

Data Objects IP Network System Advanced

BACnet IP Client Diagnostics

Devices Client Read Map Client Write Map

This page creates a map entry that writes data to one or more remote BACnet IP servers from data contained here.

Map #

Read local object named

Apply default value of at power-up and/or when seconds have elapsed with no host update.

Write remote register any time local object has changed by or when seconds have elapsed with no change.

Otherwise write remote register unconditionally. In any event, when writing remote register, apply local object data as follows:

Apply scale: and offset: Then, using index and priority proceed to

Write instance # of object type at

Repeat this process at least no more than every seconds.

i.CanDoIt® Guided Tour, part 17 ***Modbus RTU Gateway***

[\(return to index\)](#)

The Modbus RTU gateway is a Modbus master, and can poll up to 50 slave devices. The remote device "read rule" mainly says what remote register to read and where to put the data. But you have the option of scaling data as it comes in. You also have the option of applying a default value if the read fails.

The screenshot shows the configuration page for the Read Map. The navigation tabs at the top include Data Objects, IP Network, System, and Advanced. Under System, there are sub-tabs for Server Setup, Action Rules, I/O Setup, I/O Data, and Modbus RTU. The Modbus RTU sub-tab is active, and within it, the RTU Read Map sub-tab is selected. The page title is "Read remote registers into local objects. This page creates a map entry that reads data from one or more remote Modbus RTU serial devices for processing here. Click on map number to see more detail and insert/delete rules." Below the title, there is a "Map # 1" input field, an "Update" button, and navigation buttons "< Prev" and "Next >". The main configuration area includes: "Read" dropdown set to "Holding Register", "as" dropdown set to "Integer", "from register #" set to "5", "at Unit #" set to "2", and a "with doubles swapped" checkbox. Below this, there are fields for "Apply bit mask if applicable: 0000", "then apply scale: 0.0", and "and offset: 0.0". The "Save in local object" section shows "AI 150" named "Analog Input 150" with a "Repeat this process every 2.0 seconds" interval. There are also fields for "Apply this default value: 0.0 after 0 read failure(s)" and "Initial COV increment: 0.0 Initial COV Period: 0 (sec.) Units: no_units".

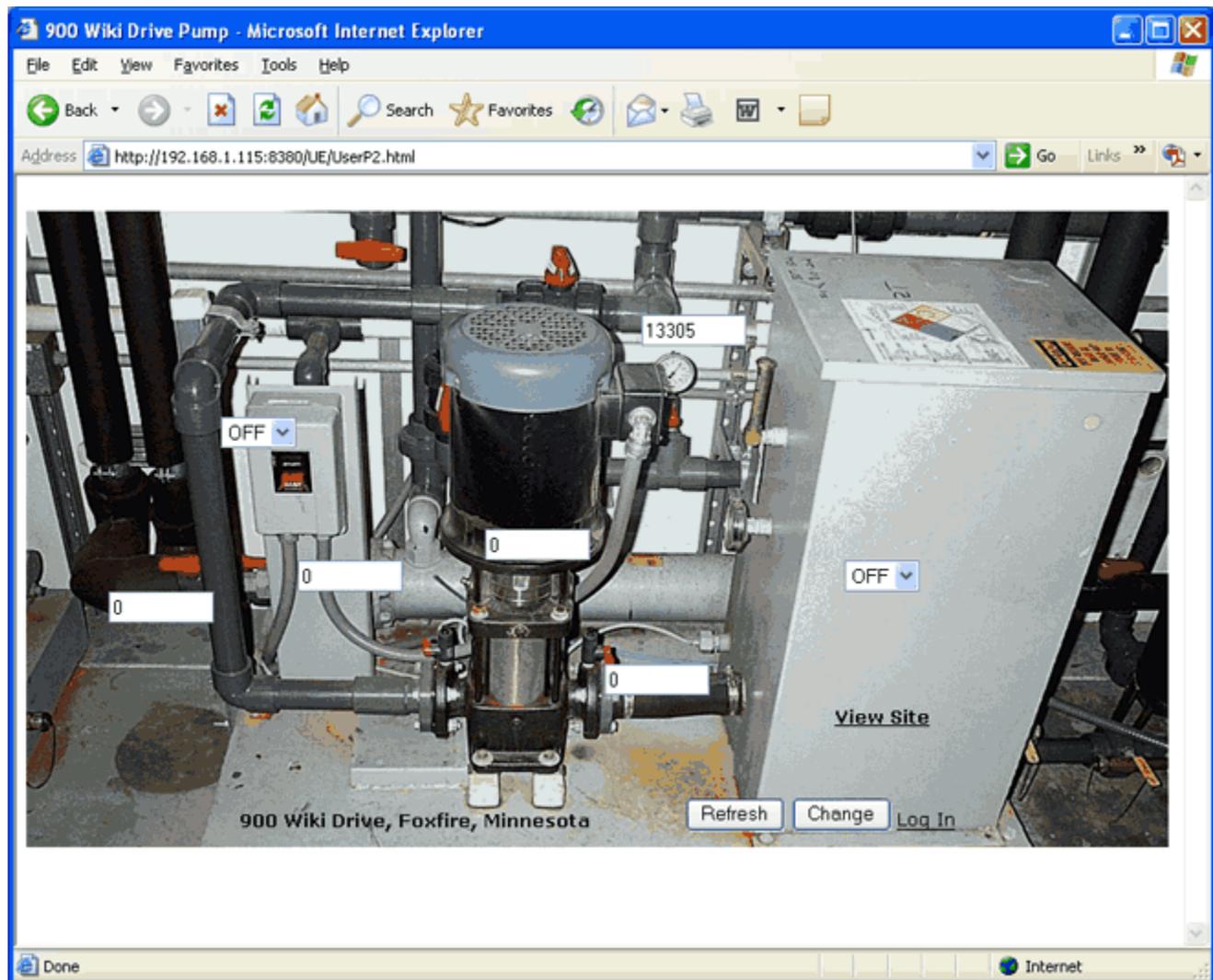
The remote device "write rule" allows setting up a "send on delta" to only write the register to the remote device when the local data has changed. You can also send periodically, with optional data scaling on the way out.

The screenshot shows the configuration page for the Write Map. The navigation tabs at the top include Data Objects, IP Network, System, and Advanced. Under System, there are sub-tabs for Server Setup, Action Rules, I/O Setup, I/O Data, and Modbus RTU. The Modbus RTU sub-tab is active, and within it, the RTU Write Map sub-tab is selected. The page title is "Write local objects out to remote registers. This page creates a map entry that writes data to one or more remote Modbus RTU serial devices from data contained here." Below the title, there is a "Map # 1" input field, an "Update" button, and navigation buttons "< Prev" and "Next >". The main configuration area includes: "Read local object" set to "AO 33" named "Analog Output 33". The "Write remote register" section has a checked checkbox for "any time local object has changed by 2.000000" and an unchecked checkbox for "when 0.0 seconds have elapsed with no change". Below this, there are fields for "Otherwise write remote register unconditionally, applying local register data as follows: Apply scale: 0.0 and offset: 0.0 Then if applicable, apply bit mask: 0000 and bit fill: 0000". The "Write" dropdown is set to "Holding Register", "as" dropdown is set to "Integer", "to register #" is set to "22", "at Unit #" is set to "1", and a "with doubles swapped" checkbox. The "Repeat this process" section has radio buttons for "at least" (selected) and "no more than every 2.0 seconds". There are also fields for "Initial COV increment: 0.0 Initial COV Period: 0 (sec.) Units: no_units" and "Initial Relinquish Default: 0.0".

i.CanDoIt® Guided Tour, part 18 ***User HTML***

[\(return to index\)](#)

You have the ability to put user friendly web pages in front of the configuration pages. If user HTML has been loaded, logging into the device will pull up the user pages instead of the default index page. There may be multiple linked user pages. An example user page is shown here.



Data types supported by the CGI processor include text and select option inputs. Dynamic, live access to data is by way of object number references used as the name of the input object. Input windows can be defined as read-only. If they are not restricted, any new data entered by the user will be written into data objects.