ELKOR Technologies Inc.
6 Bainard Street
London, Ontario N6P 1A8 CANADA
tel(519)652-9959 fax(519)652-1057

Application Note AN1701

ETnet/W2-E3 HTTP Server & XML Query Structure

Associated Product: ETnet (including WattsOn-Mark II with E3 Module)

Foreward

The ETnet features a built in web server. The web server can server local pages which are stored in the /http folder on the ETnet. This folder is accessible via FTP.
The structure of the HTTP files is not within the scope of this document, however it is worth noting that the /config folder stores all public access web pages, and the /protected folder contains the pages which are protected by the username and password. Note that the web page structure uses JavaScript and AJAX to dynamically update the web pages and the corresponding information.

XML Queries

The web server includes a special interface protocol which can be queried to serve dynamic information. This is done by sending a query to the HTTP server to the `/relay` address. The queries must be in XML format, and sent via a POST to the server.

The header should include the content type "`text/xml`"

Some valid XML queries include:

---

find

Request: `<find />`
Sample Response:

```
<device-configuration mac="00:80:a3:91:4d:27" dynamic="true">
   <device>ETnet</device>
   <name>ETnet</name>
   <firmware release-date="13-02-04 1214">1.2.1</firmware>
   <ip>192.168.138.46</ip>
   <subnet-mask>255.255.255.0</subnet-mask>
   <default-gateway>192.168.138.1</default-gateway>
   <primary-dns-server>192.168.138.1</primary-dns-server>
   <secondary-dns-server></secondary-dns-server>
</device-configuration>
```

The "find" request may also be broadcast via UDP on port 30139. All ETnet devices will respond.

6 Bainard Street
London, Ontario N6P 1A8 CANADA
tel(519)652-9959 fax(519)652-1057

get-status

Request: `<get-status />`

Sample Response:

```
<status>
   <connected-device serial="0" address="1" detecting="false">
      <name>WattsOn-Mark II</name>
      <firmware>10.84</firmware>
   </connected-device>
   <posting enabled="true" time-since="7" timestamp="2017/03/07 15:38:17">
      <result type="success">Successful</result>
      <status-code>200</status-code>
      <reason-phrase>OK</reason-phrase>
      <response>HTTP/1.0 200 OK&#x0D;&#x0A;Content-Type: text/html;
            charset=utf-8&#x0D;&#x0A;Cache-Control: no-cache&#x0D;&#x0A;X-Cloud-Trace-
            Context: 3b4c02c39857ff0623a96f1ec7cceb51&#x0D;&#x0A;Date: Tue, 07 Mar
            2017 15:38:17 GMT&#x0D;&#x0A;Server: Google Frontend&#x0D;&#x0A;Content-
            Length: 56&#x0D;&#x0A;&#x0D;&#x0A;S2C.io ACK Message: OK!. Request from
            IP: 99.226.133.177
      </response>
   </posting>
</status>
```

6 Bainard Street
London, Ontario N6P 1A8 CANADA
tel(519)652-9959 fax(519)652-1057

read-modbus

Request: `<read-modbus />`

Query Structure:

```
<read-modbus>
   <Block   address=ADDRESS
            offset=START
            length=LENGTH
            decimals=DEC
            type=TYPE />
   <Block .. />
   <Block .. />
</read-modbus>
```

Parameters:

ADDRESS: Modbus Address of the device to query.

START: Address of the start of the block to be queried. The address may be specified in hex (ie: 0x200) or decimal (512) format).

LENGTH: Number of Modbus registers (16-bit) to be queried. The address may be specified in hex (ie: 0x56) or decimal (86) format. Note: 32-bit register types require TWO 16-bit register reads. For instance: if it is desired to read "2" floating point registers, the length should be "4", etc.

DEC: Number of decimal places to present in the response (optional parameter for 32-bit floating point types). "5" decimal points is assumed if the value is omitted.

TYPE: The variable data type to reply with. Valid data types include:

U: 32-bit unsigned integer (big endian – MSW followed by LSW)
S: 32-bit signed integer (big endian – MSW followed by LSW)
F: 32-bit floating point (big endian – MSW followed by LSW)
u: 16-bit unsigned integer
s: 16-bit signed integer

6 Bainard Street
London, Ontario N6P 1A8 CANADA
tel(519)652-9959 fax(519)652-1057

Sample request:

```
<read-modbus>
   <block address=1 offset=0x200 length=0x8 type=F />
   <block address=1 offset=0x1100 length=0x8 type=F />
   <block address=1 offset=0x515 length=0x02 type=U />
   <block address=1 offset=0x502 length=0x06 type=u />
</read-modbus>
```

This request reads:

4 * 32-bit Floats from register bank 0x200 (*WattsOn-Mark II instantaneous registers*)
4 * 32-bit Floats from register bank 0x1100 (*WattsOn-Mark II energy registers*)
1 * 32-bit unsigned integer from register 0x515 (*WattsOn-Mark II uptime registers*)
6 * 16-bit unsigned integer from register 0x502 (*WattsOn-Mark II CT Ratios*)

Sample Response:

```
<read-modbus time="2017/03/07 16:43:30 UTC">
   <block offset="512" length="8" type="F" address="1" decimals="5" success="1">
      <register offset="F_200">0.00000</register>
      <register offset="F_202">0.00000</register>
      <register offset="F_204">0.00000</register>
      <register offset="F_206">0.00000</register>
   </block>
   <block offset="4352" length="8" type="F" address="1" decimals="5" success="1">
      <register offset="F_1100">0.00000</register>
      <register offset="F_1102">0.00000</register>
      <register offset="F_1104">0.00000</register>
      <register offset="F_1106">0.00000</register>
   </block>
   <block offset="1301" length="2" type="U" address="1" success="1">
      <register offset="U_515">7066</register>
   </block>
   <block offset="1282" length="6" type="u" address="1" success="1">
      <register offset="u_502">1</register>
      <register offset="u_503">1</register>
      <register offset="u_504">1</register>
      <register offset="u_505">1</register>
      <register offset="u_506">1</register>
      <register offset="u_507">1</register>
   </block>
</read-modbus>
```

The “offset” attribute of the register is prefaced with the type (ie: “u_502”, where the number represents the register offset in HEX format). This makes it easy to iterate through all of the “register” elements and find the required queried register based on the requested type.

Each “block” element is the result of a separate Modbus query. If the query fails, then the success attribute is set to “0”, and the text “`Error receiving modbus response`.” is provided as the element text. The details of the failure are not currently provided. Possible failure causes include:

1. Device wiring problems / device not present.
2. Device address does not match query address
3. Query offset does not exist on device
4. Query length exceeds available registers on device

6 Bainard Street
London, Ontario N6P 1A8 CANADA
tel(519)652-9959 fax(519)652-1057

Example using command line (wget utility) –

NOTE: *line feeds used for clarity, the command line should be one string.*

```
>wget 192.168.138.69/relay
   --header="Content-Type: text/xml"
   --post-data "<read-modbus>
   <block address=1 offset=0x200 length=0x8 type=F />
   <block address=1 offset=0x1100 length=0x8 type=F />
   <block address=1 offset=0x515 length=0x02 type=U />
   <block address=1 offset=0x502 length=0x06 type=U /></read-modbus>"
   -Ooutput.txt
```

References:

wget software:

https://en.wikipedia.org/wiki/Wget
https://www.gnu.org/software/wget/

wget for windows:

http://gnuwin32.sourceforge.net/packages/wget.htm