

April 2, 2001

TMI 700A

Product Description

Embedded Ethernet Controller



Telemonitor, Inc.
Columbia, MD 21046

This product description is provided solely for evaluating the applicability of this product for possible use in a new application and is not intended to be a complete specification of the described product. Please contact Telemonitor, Inc. for complete information on this product.

Telemonitor, Inc.
9055F Guilford Road
Columbia, MD 21046
www.telemonitor.com

GENERAL.....	3
Identification	3
Scope.....	3
Purpose	3
DESCRIPTION.....	3
System Configuration.....	3
General Configuration	3
Primary Subsystems	5
Communications.....	5
Data-Channel Model.....	6
Reference Design.....	6
Processor	6
Memory	7
Operating System.....	7
Physical	7
Environment.....	7
Shock and Vibration	7
Electromagnetic.....	7
Power.....	8
Network Interface	8
Status Indicators	8
Real-time Clock.....	8
Network Configuration	8
MAC Address.....	9
IP Configuration	9
Static IP Assignment.....	9
Dynamic IP Address Assignment.....	10
Security.....	10
Security Overview.....	10
HTTP User Authentication.....	11
Implicit Login.....	11
Explicit Login.....	11
Security Modes	11
HTTP Permission Settings	12
FTP Permission Settings	12
Setting up and Using Passwords.....	13
File System.....	13
FTP Server	14
Software Updates	14

HTTP Server	15
Standard Web Pages	15
Custom Web Pages	15
SSI Support.....	15
URL Access	16
Modbus Support.....	16
Modbus RTU	16
Modbus/TCP.....	17
Data Logger.....	18
Overview	18
Operation.....	18
Power-Fail Log Entry.....	19
Binary Log File Contents	19
ASCII Log File Contents.....	19
Returned Data.....	19
Standard Web Pages	20
Administrative Web Pages	20
Data Logger Web Pages	20
Data Access Web Pages	21
Modbus RTU Web Pages.....	21
Standard URL-based API.....	21
Administrative API.....	21
Data Access API	22
Modbus RTU API	22
Data Logger API.....	22

General

Identification

This product description is for the Telemonitor embedded Ethernet controller (TMI EEC) to be used as an embedded control device where Ethernet is available. A major characteristic of the TMI EEC is that it contains a thin web server to provide a browser-based interface for remote configuration and control of the target system over an internal network or, to the degree that external access is permitted, over the Internet and the worldwide web.

Note that the TMI EEC is not a single finished product but rather is an enabling technology, including both software and a reference design of the hardware, for use in multiple end products. The TMI EEC as described by this product description will be configured, customized, and extended for individual applications.

Scope

This document describes only the root characteristics and capabilities of the TMI EEC that exist in most or all applications. Specific configuration, customization, and extension for specific applications will be described for those individual applications.

Purpose

The purpose of the TMI EEC is to “web-enable” OEM products so as to bring the advantages of the Internet age to as broad a market as possible. This technology can be used in industrial control, enterprise networking, and remote monitoring applications.

For some applications, the TMI EEC will be used to add a remote configuration and diagnostic capability to standalone systems or to devices that have other, perhaps legacy, network capability. Other applications will use the communications capability of the TMI EEC as the primary or only communications capability.

Description

System Configuration

General Configuration

The general configuration of the TMI EEC is shown in figure 1. This configuration will be customized and extended for individual applications. Note that the CPU and Web Server, the FTP Server, and the File Server are the only items that are included in the TMI EEC. The other items are application specific.

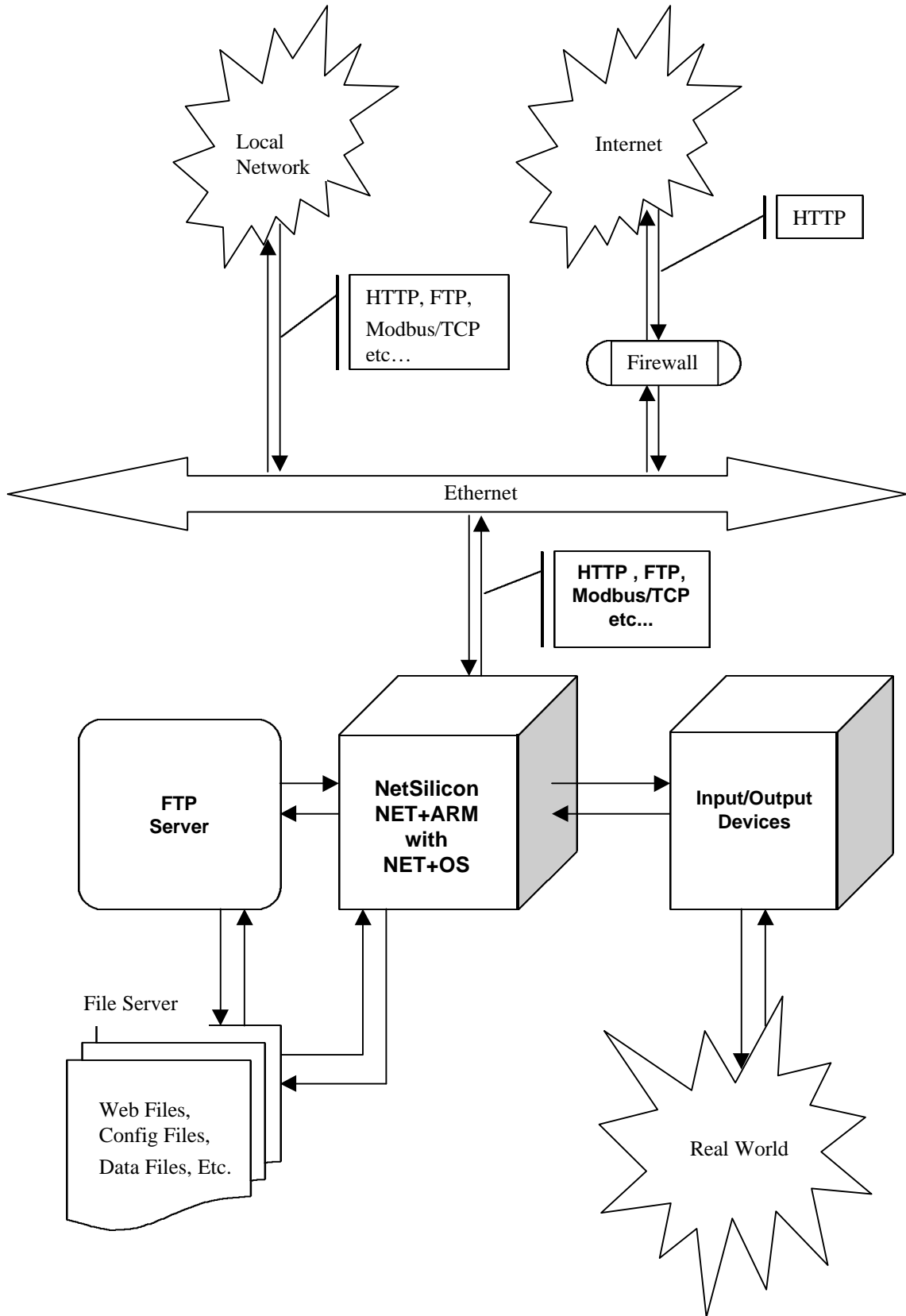


Figure 1. TMI EEC configuration

In particular, note that the Input/Output Devices and the physical connection and protocol must be defined for each application. The only I/O connection and protocol included as part of the basic TMI EEC is Modbus RTU. Modbus RTU to data channel mapping and Modbus/TCP to Modbus RTU passthrough capabilities are provided. Note also that the data channel and Modbus/TCP interfaces may be mapped to other input/output devices besides Modbus RTU, so the Modbus RTU and/or Modbus passthrough capability will not be present in all applications of the TMI EEC.

Optional I/O device capabilities include other asynchronous serial protocols such as proprietary RS-232/RS-485 interfaces, synchronous serial protocols such as IEEE 1451.2, and parallel interface protocols. These may be present singly or in combination and will be described specifically for those applications.

Primary Subsystems

A brief description of the primary subsystems of the TMI EEC is below. The detailed characteristics of the subsystems are described in the appropriate sections of this product description.

1. CPU and Web Server—The main embedded controller of the system, including the thin web server, standard applications like the Modbus passthrough and the data logger, and any customized applications. Responsible for communications with the other subsystems and with the network. The degree of communications possible through the Internet will depend on the presence and configuration of any firewalls.
2. File Server—Support for the nonvolatile (EEPROM-based) flash file system used for storing web pages, other web files such as graphics and Java applets, system configuration files such as user account files, and data files such as from the logger. End user access to the file system will be through the FTP server.
3. FTP Server—Password-protected communications interface between the end user and the flash file system. The FTP server will provide read and write access to individual files but will not support creating or deleting directories.

Communications

Setup, configuration, control, and input/output device data can be read and written over the Ethernet connection in any of the following ways:

1. Through web pages, either the standard ones or custom pages written and downloaded by the end user. Web pages can display dynamic data in addition to static HTML text and graphical content.
2. Through the HTTP application programming interface (API). This capability is useful for writing custom interfaces using downloadable applets written in browser-based languages such as Java, for interfaces to standard PC-based

programs such as Access, Excel, and Visual Basic which have a built-in HTTP capability, and for custom access and control programs that operate over the Internet or otherwise through routers, switches, and/or firewalls.

3. Through other TCP-based interfaces such as Modbus/TCP. For other than Modbus/TCP, the exact protocols and packet contents will be application-dependent. Due to protocol limitations and security concerns, the full range of setup and control functions are not available through Modbus/TCP, and are not expected to be available through other application-dependent TCP-based protocols.

Data-Channel Model

The TMI EEC uses a data-channel model for managing and reporting data from the input/output devices. Since multiple I/O devices can be connected to a single node, the TMI EEC also supports the use of device numbers. A unique data channel comprises a device number and a device channel number. For systems with only one device, a default device number may be set. Data read from channels can be real-time (most recent readings), or can be historical readings from the data logger. Data from these channels can be accessed using any of the specified communications techniques (web pages, the HTML API, TCP, etc.).

The data channel configuration (number of devices, default device number, number of channels, range of each channel, units to display, etc.) may be fixed for a given application or it may be programmable through web pages, the HTML API, or configuration files downloaded into the file system. The specific configuration options will be application specific.

Note that the Modbus/TCP to Modbus RTU passthrough does not explicitly use the data-channel model, but rather passes data to and from the specific registers and coils specified in the Modbus/TCP packets.

Reference Design

The reference design is provided as a starting point for development of a complete system, as well as for use as a test bed and development platform. The base TMI EEC design will be customized and extended for specific applications.

The hardware-oriented requirements, such as for physical size and environments, are representative of what is anticipated to be required of a specific application.

Processor

The TMI EEC is based on the NetSilicon NET+ARM processor. The NET+ARM processor has a 32-bit RISC core and includes an Ethernet media access controller (MAC), a memory controller, serial ports, and parallel ports in a single package.

Memory

The TMI ECC reference design has 16 MB of SDRAM and 2 MB of non-volatile flash EEPROM. The EEPROM stores both code and data, with 1 MB partitioned for code and 1 MB partitioned for data. The size of both the SDRAM and EEPROM can be changed to suit the needs of specific applications.

Most of the data partition is used for the file system and is available to the user through the FTP interface. The code partition is not directly accessible by the user, but there is provision for updating the software through the FTP interface using special directory and access provisions.

The TMI EEC reference design uses Intel flash memory and the Intel Virtual File Manager (VFM). Currently, Intel provides a royalty-free source-code license for use of the VFM with Intel flash memory hardware.

Operating System

The TMI EEC is based on the NetSilicon NET+OS operating system 2.x, with custom extensions to support end-user files. These include adding the file server itself, as well as FTP support for reading and writing files, and HTTP server support for serving user files.

Physical

The TMI-ECC reference design will fit on a single printed wiring board not to exceed 12.4 square inches (80 x 100 mm nominal size) with a maximum component height of less than 0.5 inch.

Environment

-40° to +85° C (storage)

-40° to +85° C (operate)

Humidity 0 to 95%, non-condensing.

Shock and Vibration

The TMI EEC reference design will use rugged components and mounting techniques consistent with qualification of a specific end-item design to typical industrial, military, or transportation shock and vibration requirements, when properly mounted and packaged in an appropriate housing.

Electromagnetic

The TMI ECC reference design will support qualification of a specific end item design to the requirements for FCC Part 15 Class B digital devices, as well as the applicable EC EMC directives, when packaged in an appropriate housing.

Power

The TMI ECC requires an external DC power supply that provides a minimum of 5 V and a maximum of 24 V at 3 W. Internally, the TMI EEC operates at 3.3 V (required by the NET+ARM processor). The external I/O pins operate at 5 V.

A standby battery is included which provides operation of the real-time clock for a minimum of five years. Longer standby operation can be provided for in specific applications by using a larger battery.

Network Interface

The network interface is Ethernet, with support for both 10 and 100 Base-T. The physical connection is made using an RJ-45 socket mounted on the printed wiring board. Other connection arrangements can be used for specific applications.

Status Indicators

The TMI EEC has a system status LED mounted on the printed wiring board that is green when the unit is operating normally and red when the CPU detects an error condition. Errors indicated by the status LED are limited to faults that might interfere with the ability to communicate over the network, such as a CPU failure or an EEPROM code segment checksum failure. Other faults, such as a low battery for the real-time clock, are reported through the network communications.

The Ethernet physical interface driver indicates the status of the network connection with one LED for link status and one LED for receive/transmit activity.

Real-time Clock

A hardware real-time clock (RTC) keeps the current time of day to the nearest second when external power is not applied. On power-up or reset, the TMI EEC reads the real-time clock and sets the operating system clock to the current time. The TMI EEC monitors the condition of the RTC battery and sets the appropriate error status register bit when a low battery is detected. This results in an error indication on the standard web pages.

Network Configuration

The TMI EEC communicates using TCP/IP over 10/100 Base-T Ethernet. In order to do this, the Ethernet and IP addresses must be set. The Ethernet address is unique and is set by the manufacturer of the interface. In this case, that may be either the manufacturer of the TMI ECC or of the end item that is based on the TMI ECC. The IP address may or may not be unique and is set by the end user. It is the responsibility of the manufacturer to provide means for the end user to set the IP address.

MAC Address

The Ethernet address, also called a media access controller (MAC) address, is a globally unique 48-bit number that is permanently assigned to the MAC hardware in the factory. Ethernet addresses must be purchased from the IEEE either as a block of 500 complete addresses or as a 24-bit organizationally unique identifier (OUI). A complete Ethernet address is 48 bits (6 bytes). The OUI is the top 24-bits of the Ethernet address. This allows manufacturers to assign the lower 24-bits, allowing for 16 million unique addresses. For more information about purchasing Ethernet addresses, see <http://standards.ieee.org/regauth/oui/index.shtml>.

For the TMI EEC, the MAC address is programmed at the time the initial operating software is loaded into the EEPROM, and is stored in a special secure area of the EEPROM. The MAC address is memory-mapped for the initial software loading so standard device programmers with a serialization capability can assign the MAC address. Software updates will not change the MAC address.

IP Configuration

The TMI EEC must also be assigned an IP address, subnet mask, and default gateway for use on a TCP/IP network. There are two methods of doing this: static assignment and dynamic assignment. Static versus dynamic assignment can be selected on the administrative web pages, through the HTTP API, or by use of the static assignment utility. If a user chooses to use static assignment, then dynamic assignment will not work until the setting is changed. The factory default will be dynamic assignment so that users with that capability will not need to use the static assignment utility at all.

Static IP Assignment

The TMI EEC static IP settings may be configured in three different ways:

1. Using the IP configuration pages of the standard administration web pages.
2. Using the IP configuration URLs of the administrative HTTP API.
3. Using a special client-side utility program (Windows/98 or Windows/NT) that searches a local subnet and identifies the attached TMI EEC units.

Any of these methods can be used for a unit that is correctly configured and communicating properly, but the utility program is the only way to set the IP configuration of a new unit, or of one that is improperly configured or previously configured for a different subnet.

Changes to the static IP configuration are effective after a system reset.

Dynamic IP Address Assignment

The TMI EEC may be assigned a dynamic IP address using the Dynamic Host Configuration Protocol (DHCP) in accordance with RFC2131. DHCP is a standard protocol and is available in many network environments.

If the IP configuration mode is changed from static to dynamic using one of the methods described for changing the static IP configuration, then the unit will search for a DHCP server the next time it is powered up or reset.

Security

Security Overview

The TMI EEC implements a straightforward multi-level protection mechanism. This mechanism is based on a combination of security modes for the node, permission levels for the users, and permission level settings on each URL to determine if a transaction from a user is permitted to execute the target URL. This combination of node security mode, user permission levels, and URL permission settings allows the tailoring of access to any URL in the TMI EEC.

The TMI EEC will support conventional environments where there are a large number of individual users, each with their own username and password, and each with the ability to change their password. However, the TMI EEC also can be used in applications where the administrator wants to control access to the device by classes of users. A user class such as "operator" for example may have many individuals who log in with the same username and password and have the same permission levels on the node. It would not be appropriate to allow one individual operator to change the password used by many other operators. In this case, the administrator can deny individual users the ability to change their own passwords.

The security mode (OPEN or CLOSED) determines whether or not passwords are required to access URLs on the node. The permission levels determine which users can access a given web page or API URL in CLOSED mode.

The default factory configuration is to ship with all access control turned off (i.e. in OPEN mode) and with root and guest accounts. The default usernames and passwords and permission settings for these accounts are described in the paragraph on setting up and using passwords.

This security scheme is not intended to provide or replace stronger security mechanisms for authentication or encryption such as Secure Sockets Layer (SSL), but rather to provide protection against casual misuse. For example, the TMI EEC does not currently implement encryption of information passed in HTTP requests.

The web server supports controlling different users' accesses to different HTTP API URLs. Individual users may also be given read or write permission for the user file system

through the FTP server. Only the root-level administrative user has read and write permission to the system directory in the file system.

HTTP User Authentication

The web server supports user authentication with the concepts of a user and sessions associated with that user (See RFC2617). A session is a series of HTTP transactions by a single user defined by the use of a common Session ID (SID). If a SID has not been used for fifteen minutes, the server will automatically log the user out. The web server supports up to fifty simultaneous sessions with unique active SIDs at one time.

If security is turned on, users will need to first log in with their user name and password before they are allowed access to the server. There are two ways to login: implicitly and explicitly.

Implicit Login

An implicit login request is made when an HTTP transaction request is received with non-NULL values for the parameters user and password. If the user name and password are valid, a SID is created and associated with this transaction. For URLs pointing to server-parsed HTML files, the resulting file will contain the SID if the file contains a SID tag. If a SID is not used for fifteen minutes, the session times out and the user must log in again and obtain a new SID.

Explicit Login

The user may explicitly log in using the standard web page or the login URL of the HTTP API. If the user name and password are valid, the SID string will be returned in the entity body of the response. For subsequent requests, users can simply supply the SID as proof of their credentials, rather than having to supply a user name and a password each time. Normally this is done directly by the URL links on generated pages, but it can also be done programmatically. The permission levels associated with the SID are those of the account used in the login to generate the returned SID. If a SID is not used for fifteen minutes, the session times out and the user must log in again and obtain a new SID.

Security Modes

There are two parts to the protection mechanism for any given web page or HTTP API URL: security mode and permission level. For a user to be able to access a given URL, both the security mode and the permission level constraints must be satisfied.

Security mode is a server-level policy that enforces user authentication. A TMI EEC is always in one of two security modes:

OPEN—All of the URLs are open to the public. No user authentication is required.

CLOSED—None of the URLs except for the standard login page are open to the public. User authentication is required for all other access.

The current HTTP security mode can be set and queried via the security web page or API URLs.

Note that login is always required for FTP access to the file system.

HTTP Permission Settings

Each HTTP API URL has a fixed multi-part permission level mask associated with it. Individual web pages also may have permission levels associated with them. Server-parsed HTML pages (those that have the extension of SHTML) can have tags included in the header of the page to set the permission levels for the individual web pages.

Pages with the extension of HTML are not parsed by the server and will be served regardless of the security mode and permission level. This permits public access to a login page such as /www/index.html prior to login and authentication. Since these pages are totally open without regard to any security features, the HTML extension should be used infrequently.

There are two permission levels that can be set for each page or API URL: read access and write access. The normal range of the permission levels is zero to five. Lower numbers grant more access. The root user has permission levels of zero for both read and write access. The permission level of the user must be equal to or lower than that set for the page or API URL in order to execute the request.

Read access to a page or API URL means that the page will be displayed or the requested data will be returned. Write access to a page or API URL means that submitted data will be accepted and can change system settings or I/O output settings.

This multi-level control allows powerful tailoring of access to systems. For example, guests may be given read access to an equipment virtual front panel but not be allowed to change any settings. Operators may be allowed to change the system set point within preset limits, but not change those limits or the system operating parameters. Engineering can change the operating limits and parameters, but not the network settings or who has access to the node. The root user can view or change anything.

FTP Permission Settings

The FTP server recognizes two permission bits for individual users: read and write. The read bit may be set individually for each user. The read bit must be set to permit access to the FTP server. The read bit also allows the user to download files from the server. To grant the user the additional permission required for writing or deleting files the write bit must be set. However, the write bit should be set only if the read bit is also set. If the write bit is set without the read bit being set, the FTP server will not accept the user login.

Only the root user has access to the system directory. All users with FTP read permission can list the contents of the system directory but only the root user can download or write files in this directory.

Setting up and Using Passwords

The user account information (usernames, passwords, and permission levels) is stored in an encrypted file in the system directory of the file system. The encryption is unique to the particular server, so a password file from one TMI EEC will not work on another. The root user may delete the password file through the FTP server.

At power-up or reset, the system reads the password file. If no valid password file exists, the system sets the security mode to OPEN and creates a password file that contains the default users “root” and “guest.” The root account has full access to everything and can change its own password. The guest account has lower level permissions for HTTP services, no access to the FTP server, and only the root user can change its password.

Accounts may be edited or deleted using the administrative web pages or the admin HTTP API. The root account password may be changed, but the root account may not be deleted and its permission levels may not be changed. This is to ensure that there is at least one account with full access to the system. Up to ten user accounts may be created.

Usernames and passwords cannot exceed 31 characters in length and may contain any printable ASCII character except for <sp>.

Recovery from forgotten root user passwords or corrupt password files is provided by a hardware access means. If the unit is powered up or reset with a specified test pin shorted to ground, then the system deletes the existing password file and goes into default mode as described above. This procedure assumes that anyone who has access to the internal hardware of the unit can be permitted to reset the password file. This recovery capability can be disabled for higher security installations. In that case, any provision for recovery from a forgotten root password or corrupt password file will be described in the System Requirements Specification for that specific application.

File System

The TMI EEC includes a flash EEPROM file system for storing web pages, user data files, and system files. Users have access to the file system through the FTP system. The standard directories are:

- / Used for common access files such as /system.ini.
- /log Used for user data files such as those created by the data logger.
- /system Used for protected system files such as the password file
- /www Used for web server files such as pages and graphics.

Other directories may be included for specific applications. In that event, they will be described in the System Requirements Specification for that application.

Only the root user has access to /system. Access to the other directories is controlled by the FTP permission settings for the individual users.

Filenames cannot exceed 31 characters in length and may contain any printable ASCII character except for <sp> and ”/”.

FTP Server

The TMI EEC includes a restricted File Transfer Protocol (FTP) server in accordance with RFC0959 and RFC1579 to provide access to the flash file system. This supports replacing web pages, adding new web pages, and retrieving data from the system, in addition to other uses.

The FTP server does not permit anonymous FTP. All FTP sessions require a valid username and password with at least FTP read permission. Users without FTP write permission will not be allowed to perform any operation that modifies any data in the file system.

The FTP server does not support creating or deleting directories.

The FTP server only supports ASCII (AN) and IMAGE (binary) transfer TYPE and only supports stream MODE.

The FTP server supports the following commands: CDUP, CWD, DELE, HELP, LIST, MODE (Stream only), NLIST, NOOP, PASS, PASV, PORT, PWD, QUIT, RETR, STOR, SYST, TYPE (ASCII and IMAGE only), and USER. See the referenced RFCs for a description of these commands.

The FTP server and the TMI EEC file system can be accessed from a standard web browser by entering the following in the URL field: ftp://username:password@system where appropriate substitutions are made for “username,” “password,” and “system.” If :password” is omitted, the browser will prompt for one.

Software Updates

The FTP system can be used for field software updates. If a file is written to the special hidden directory /update the system will load the contents of that file into data memory. After closing the FTP session, the system verifies the integrity of the file and begins to program the flash memory. The status light of the TMI EEC will alternate between the normal and error indications once per second during this transfer. When the transfer is complete, the unit will execute a software reset and begin execution of the new code.

A field software update does not affect any other data on the unit, including the MAC address, IP configuration, web pages, user accounts, or any logger data files.

Note that only the root user has access to /update so only the root user can perform a field software update.

If the software transfer is interrupted before it is complete, then the unit will not perform the firmware update. During a firmware update the unit will not respond to FTP or web accesses. Do not remove power to the unit during the firmware update since this will leave the system in an unstable state. Recovery from this condition will require repeating the factory load of the software using the JTAG port or a device programmer on the EEPROM.

HTTP Server

The TMI EEC includes a thin web server based on HTTP/1.0 (see RFC1945). This server can serve standard web files (HTML, CLASS, JPG, JIF, etc.), either the standard pages furnished with the unit or custom pages developed by the user. It also supports server-parsed HTML to permit displaying live data in the web pages.

Standard Web Pages

The TMI EEC is furnished with several standard web pages that can be used to log in, for administration and configuration of the system, and for display of the real-world interface provided by the I/O devices. These pages are loaded at the time of manufacture into the /www directory and may be read by any user having FTP read privileges. They may be deleted or replaced (overwritten) by any user having FTP write privileges.

These standard pages permit the user to immediately use and configure the unit and they can be read, modified, and replaced to provide a customized version of the web interface with minimum effort by the user.

The standard web pages are described in the Web Pages sections of this document.

Custom Web Pages

For a more custom appearance and functionality, the user may develop fully custom pages from scratch and load them into /www. In addition to HTML and SHTML pages, other standard web site files (CLASS, JPG, JIF, etc.) can also be used.

Note that the default page for initial access of the node is /www/index.html, which is normally used as a login page to establish a SID and control access to the other pages.

SSI Support

Web pages with the SHTML extension are Server-Parsed HTML and will be parsed by the web server. SHTML pages use Server Side Includes (SSI) directives (tags) to direct the server to insert current data into the page before serving it to the browser. The format of an SSI directive is:

```
<!--#command attribute="value" -->
```

SSI directives look like HTML comments so that they are hidden in files without the SHTML extension, or if presented to a server that does not have SSI capability.

Server-Parsed HTML by Netscape and *Server Side Includes (SSI)* by Purdue University give the basic requirements for SHTML and SSI use.

URL Access

The TMI EEC provides special URLs for programmatic access to the unit configuration and I/O device data. This capability is useful for writing custom interfaces using downloadable applets written in browser-based languages such as Java, for interfaces to standard PC-based programs such as Access, Excel, and Visual Basic which have a built-in HTTP capability, and for custom access and control programs that operate over the Internet or otherwise through routers, switches, and/or firewalls.

The client application submits or requests data in the form of an HTTP transaction request to the specific URL. The server will respond with text as specified for the individual URLs. All HTTP API URLs are in the standard form of:

```
http://<server>/<path>/<file name>?<parameter 1>=<value 1>&<parameter 2>=<value 2>...&<parameter n>=<value n>
```

The HTTP API URLs are described in the API sections of this document.

Modbus Support

The basic TMI EEC provides support for Modbus RTU I/O devices, both as data channel devices and via the Modbus/TCP passthrough function. These services will not be present in specific applications that define other protocols for the data channel to I/O device interface, or which use the Modbus/TCP capability for access to other than standard Modbus RTU devices.

Modbus RTU

Modbus RTU is an asynchronous serial protocol usually carried on an RS-232 or RS-485 physical layer. When used with RS-485, the protocol is multi-drop in that messages include a unit number. Up to 255 Modbus devices can communicate on a single RS-485 network. *Chapter 1 Modbus Protocol* by Modicon provides more information about the details of the Modbus RTU protocol.

The TMI EEC Modbus RTU interface maps Modbus coils and registers to data channels. Reads from and writes to those channels are translated to read and writes for the appropriate register or coil. The interface has the following limitations:

- It can use either the RS-232 or the RS-485 physical layer, but not both.

- The communications parameters for all RS-485 devices must be identical except for the device addresses, which must be unique.

Both the Modbus RTU communications interface and the individual data devices and channels must be set up before use. Both are set up using the Modbus API that is described in detail later. The setup parameters are straightforward and are described in the API section. Once properly set up, the specified Modbus RTU registers can be read and written using the data channel access web pages, SSI tags, and HTTP API.

The Modbus RTU interface supports four types of registers:

1. 0x registers (coils)—These are one-bit input/output devices. A zero will be written to the coil if the data is zero and a one will be written to the coil if the data is non-zero. A one or zero, as appropriate, will be returned from reading a coil. The data type and length are ignored for these registers.
2. 1x registers (discrete inputs)—These are one-bit input devices. A one or zero, as appropriate, will be returned from reading a discrete input. The data type and length are ignored for these registers.
3. 3x registers (input registers)—These are read in 16-bit increments. If the data type is float, then the data length parameter is ignored and two registers (32 bits) are always read or written, in reverse order if specified. If the data type is integer, then the data length will be interpreted as the number of 16-bit registers to be read or written, in the order specified in the Modbus RTU protocol.
4. 4x registers (holding registers)—These are read or written in 16-bit increments. If the data type is float, then the data length parameter is ignored and two registers (32 bits) are always read or written, in reverse order if specified. If the data type is integer, then the data length will be interpreted as the number of 16-bit registers to be read or written, in the order specified in the Modbus RTU protocol.

Modbus/TCP

Modbus/TCP is a passthrough protocol that encapsulates Modbus register read and writes in TCP/IP data packets. Modicon defines the Modbus/TCP packets in *Open Modbus/TCP Specification Release 1.0, 29 March 1999*. The TMI EEC converts those Modbus/TCP packets to Modbus RTU reads and writes and executes them on the Modbus RTU I/O device interface.

The Modbus RTU communications parameters are used for the Modbus/TCP passthrough, so they must be properly configured before use. The Modbus RTU to data channel mappings are not affected by the Modbus/TCP operations and need not be configured. If configured, however, they are not incompatible with the Modbus/TCP usage and both forms of access to the Modbus RTU data may be used simultaneously.

Note that other specific applications may map the Modbus/TCP registers to other usage than a straight translation to Modbus RTU registers. All that is required is to assign Modbus register numbers to new usage, keeping in mind the implied data lengths for the different blocks of register numbers, and the balance of the Modbus/TCP protocol can be used as defined. In that event, the Modbus RTU support will not be available.

The specific usage of the I/O device interface and of the Modbus/TCP packets will be defined for any such specific application.

Data Logger

Overview

The basic data logger supplied with the TMI EEC reads user-designated data channels at a user-specified sampling period and stores the readings in a log file in the EEPROM flash file system. The stored readings are available for retrieval later by an external client such as a Java applet, Excel spreadsheet, or Visual Basic program. The logger does not delete stored data unless explicitly directed. Only one logger session can be running at a time, and the parameters cannot be changed while the logger is running.

The log file can be binary or ASCII and is stored in the /log directory of the file system. The file can be read, copied, or deleted by any user with the appropriate FTP permission. The file can only be accessed by FTP when the logger is idle. The logger creates a new data file each time it is started using a unique file name.

Operation

The logger may be in the stopped, running, or waiting to run state. If stopped, no readings are being stored in the log file, but the logger will respond to all HTTP transaction requests and the user can change the list of devices and channels to log and the sampling period. A single sampling period applies to all channels.

When running, the only transactions that the logger will respond to are requests for data or status (device and channel list, sampling period, etc.), and the command to stop sampling and return to the idle state.

When waiting to start, the logger cannot be configured. The waiting to start state is entered when a non-zero start time is specified. When the start time occurs, the logger will automatically enter the running state.

The logger can report several error conditions including memory full, file write error, power-fail detected (if auto-start is enabled), and file system full. Stopping and then restarting the logger will clear the error condition.

Configuration and operation of the logger are conceptually simple: select the devices and channels to log, set the sample period, set the starting time (if desired), and issue the start command. The logger will append new data to the log file until the stop command is

issued or the data record limit is reached. Setting the starting time can be used to synchronize the sample time-stamps to an even time reading (exactly on even seconds for example), rather than the random time that otherwise would result. If the starting time is set to zero, or left set to the previous value which has already passed, the logger will start immediately when the status is changed to running.

Power-Fail Log Entry

If the logger configuration is set to auto-start at power-on and the logger was running when the power was removed, the logger will record a special entry in the last log file that was in use. This log entry will have a current time and date stamp with all remaining fields set to zero. In addition, the logger will report a power-fail error to the logger web pages and API functions. The logger will then create a new log file and start running normally. To avoid creating power-fail log entries, stop the data logger before removing power from the EEC. The EEC will automatically stop the data logger when a system reset is issued using the reset API call.

Binary Log File Contents

Each binary log file contains a header and several data records. Data are stored in big endian order. Each file has a header followed by one or more data records. The data type of all of the fields in the file is 32-bit unsigned integer, except for the sample data value. The data type of the sample value is a 64-bit double.

Note that while the file format allows for sample periods and starting times in nanoseconds, the shortest sample period that can be set for the reference design is 10 ms. The period and starting time will be rounded off to the nearest 10 ms.

ASCII Log File Contents

The ASCII log file is a simple tab-delimited file suitable for importing into Excel and other applications. The first line of each file is a header line describing the contents of each column. Each remaining line is a numbered data record.

Returned Data

Sample data returned by an HTML request consists of tab-delimited ASCII records, one line per record, with a <CR><LF> terminator. The returned data is in the same format as the records in the ASCII log file.

The data may also be returned in XML format. The XML tag names are the same as the ASCII log file field names.

Standard Web Pages

Administrative Web Pages

The following standard web pages are included with the TMI EEC for use in configuring and controlling the system. These pages may be edited or replaced by the end item manufacturer or the end user to extend or customize the administrative interface.

Some page read and write permission levels are specified as “root only” and cannot be changed. Other levels can be changed by using the `httpSetPageSecurity` SSI tag.

- `addAccount.shtml`: Add a new user account and set password and permission levels.
- `deleteAccount.shtml`: Delete a user account.
- `editAccount.shtml`: Edit user account permission levels.
- `userAccount.shtml`: Tabular list of user accounts and permission levels.
- `networkConfig.shtml`: Set the IP configuration (address, subnet, gateway).
- `password.shtml`: Change the password of the currently logged-in account. Or, if accessed by an administrative-level account, change the password of any account.
- `securityMode.shtml`: Set the security mode (OPEN or SECURE).
- `time.shtml`: View and set the system time.
- `fileSystem.shtml`: View the file system status.
- `displayProperties.shtml`: View and change the data display properties.
- `serialConfig.shtml`: View and set the serial communication parameters.
- `logout.shtml`: End the current session.

Data Logger Web Pages

- `loggerConfig.shtml`: Configure and control the data logger.
- `loggerStatus.shtml`: View the status of the data logger.
- `loggerData.shtml`: View data from the current logger session.

Data Access Web Pages

- `chanConfig.shtml`: Examine and change data channel configuration settings (gain, lower, name, offset, units, upper).
- `chanIO.shtml`: Examine and change I/O device channels.

Modbus RTU Web Pages

- `rtuConfig.shtml`: Examine and change Modbus communications configuration settings (RS-232/485, baud, parity, stop bits, retry limit, timeout).
- `rtuDeleteChan.shtml`: Delete the mapping of a Modbus register to a data channel.
- `rtuDeleteDevice.shtml`: Delete the mapping of a Modbus address to a data device.
- `rtuMapChan.shtml`: Map a Modbus register to a data channel and set the configuration (device channel, register, data type, data length).
- `rtuMapDevice.shtml`: Map a Modbus address to a data device (device number, Modbus device address, word order for floats).

Standard URL-based API

Administrative API

Web pages or applets can use the following URLs to initiate actions within the unit as well as to access information from the device. Most of the URLs return values to be used by the calling page or applet.

The read and write levels of API URLs are fixed and cannot be changed.

- `addAccount`: Add a user account to the system.
- `blink`: Blink the status LED for identification purposes.
- `editAccount`: Change an existing user account.
- `deleteAccount`: Delete an existing user account.
- `login`: Log in an account
- `logout`: End the specified session.
- `networkConfig`: Set one or more network configuration items.
- `serverName`: Set or return the server name.

- serverLocation: Set or return the server location.
- password: Set the password for an account. Requires password change permission unless used by a administrator-level user.
- reset: Reset (reboot) the node.
- securityMode: Retrieve or set the security mode.
- time: Retrieve or set the current time. If set, also updates the real-time clock.
- firmwareVersion: Retrieve the current firmware version.

Data Access API

- chanConfig: Set one or more data channel parameters.
- chanQuery: Retrieve data channel configuration item. Only one item can be specified. If more than one item is specified, only the first is returned.
- read: Read channel data
- write: Write channel data. If data or rawData is not specified, nothing is done.

Modbus RTU API

- rtuConfig: Configure Modbus communications parameters.
- rtuMapChan: Map a Modbus register to a data channel and set the configuration.
- rtuMapDevice: Map a Modbus unit address to a data device.
- rtuQuery: Retrieve Modbus configuration item. Only one item can be specified. If more than one item is specified, only the first is returned.

Data Logger API

- loggerDataPoints: Set or retrieve the device and channel list for the logger.
- loggerSampleRate: Set or retrieve the logger sample rate.
- loggerStartTime: Set or retrieve the logger start time.
- loggerStatus: Set or retrieve the logger status.
- loggerFileType: Set or retrieve the file type (text or binary).

- `loggerRecordLimit`: Set or retrieve the maximum number of records to store in a single log file.
- `loggerPowerRecovery`: Set or retrieve the logger power recovery mode.
- `loggerData`: Retrieve logger data records.