Application Note

AN_415

FT90x Ethernet to GPIO Bridge

Version 1.1

Issue Date: 2017-04-04

This application note describes an Ethernet to GPIO Bridge implemented on the FT900. It uses lwIP for transferring data between GPIO and an HTTP configuration interface.

Bridgetek Pte Ltd (BRT Chip)

178 Paya Lebar Road, #07-03 Singapore 409030
Tel : +65 6547 4827 Fax : +65 6841 6071
Web Site: http://brtchip.com

Table of Contents

1 Introduction

The FT900/FT901 and FT905/FT906 devices include an Ethernet interface. This can be used with a suitable networking stack to allow the device to be networked. This application note shows how the Ethernet interface can be used to bridge GPIO to a configuration webpage. In addition, the network information such the MAC and IP address are retrieved and stored on the external I2C EEPROM, giving the advantage that subsequent application software updates will not overwrite this information.

1.1 Overview

The user can control GPIO output levels and read GPIO input status. Friendly names can be given to the GPIO ports for readability.

The network interface supports DHCP to obtain configuration information or it can be configured using a web page served by the FT90x firmware. GPIO control is performed using the same web page. The configuration is stored either on FT90x Flash or external EEPROM, so that they are not lost after power off:

- FT90x FlashROM: Hostname, GPIO names / numbers
- External EEPROM: MAC address, IP/ gateway/mask address, DHCP setting

If the EEPROM key is not valid, the software enables DHCP and resets other configuration settings to the default values.

Third-party open source code is used to implement a TCP/IP stack in this application note:

- TCP/IP - lwIP (LightWeight IP) library which has been ported to the FT90x.
- Printf – tinyprintf.

Links to resources for these libraries are in Appendix A – References.

1.2 Scope

GPIO can be used to control lots of things in the real world like motors, lights and fans.

This simple example demonstrates how to control such GPIO from a web interface where the PC and FT90x exist on the same private network.

The external I^2C EEPROM used in this example is the 24AA02E48T-I/OT which is a 2K serial EEPROM device. This memory is present on the MM900EVxA , excluding the MM900EV-LITE.

1.2.1 Features

The application note highlights the use of lwIP to provide a TCP/IP stack. The "arch" folder of lwIP in the source code for the application note contains the architecture-specific parts of lwIP required for the FT90x.

1.2.2 Enhancement

Enhancements to this application might include:

- Provide a configurable number of GPIO interfaces (default is 4 x input and 4 x output)
- Allow only permitted IP addresses to connect to the device.
- Reduce the overall size of the application code.

This Ethernet to GPIO bridge example application should be treated as an example. Full source code is provided allowing users to build and modify if required:

http://brtchip.com/ft90x/#FT90x Ethernet to GPIO Bridge

See section 5 Importing into the FT9xx Toolchain for more information.

2 Project Overview

The project files for this application note are divided into the following folders.

Folder	Description
Source	Application source code and abstraction files.
httpdfs	File system for web server.
lib	Library files.
lib\tinyprintf	tinyprintf library.
lib\lwip	lwIP library.

Table 2.1 Project Files Overview

The application source code is contained within the "Sources" folder. The main() function and the high-level functions of the application are in the "main.c" file. The network interface is within "net.c".

2.1 Main Program

The main program is responsible for handling configuration and control by the web interface. The code generating data for the simple server side includes (SSI) functions in the lwIP httpd server. Data is returned through HTTP GET requests to update the configurations. The httpd server uses callbacks for the server size and includes the HTTP GET handler.

The dlog features described in AN_365 FT9xx API Programmers Manual are used for persistent storage and retrieval of configuration data.

2.2 Network Code

The networking layer is provided by the lwIP library which has been ported to support the FT90x. The lwIP code is available separately from FTDI with the FT90x architecture extensions.

2.2.1 Network Abstraction

The network interface is abstracted from the main project in the file "net.c". All lwIP features which refer to an interface (netif) are passed through this layer.

The lwIP network interface is initialized and configured using data supplied from the main application. The function which processes incoming packets for the network interface within lwIP is called from the net_tick() function.

2.2.2 lwIP Library

The library is available with a BSD-style license. Version 2.0.0 of the code is used. No changes have been made to the code from the lwIP project page.

The FT90x architecture code is responsible for obtaining packets from the Ethernet interface and transmitting packets to the Ethernet interface.

The application makes extensive use of the callback methods in lwIP to permit sending and receiving of data.

The httpd app is used to provide an interface for configuration. This makes use of server side includes (SSI) and CGI scripts (HTTP GET requests) to present configuration data to the user and change it when requested to do so.

2.3 Persistent Storage

Networking parameters such as MAC address, IP/ gateway/mask address and DHCP setting are stored on external EEPROM accessible via I2C reads and writes.

The following variables in the application code show the location of each of the parameters which reside in the I2C EEPROM:

```
struct eeprom_net_config {
        uint16_t key;
        uint8_t dhcp;
        ip_addr_t ip;
        ip_addr_t gw;
        ip_addr_t mask;
};
```

The permanent MAC address is stored in the last 6 bytes of the EEPROM memory (0xFA to 0xFF. Note that this is permanently protected and cannot be overwritten.

Other configuration parameters including the network hostname and some GPIO parameters are stored in FlashROM on the device. The 4 kB long datalogger partition is used to keep this data between power cycles. The data is updated each time the data is changed by the configuration interface.

When the application starts it checks the data stored for the network and GPIO configuration in the persistent storage. Two 32-bit keys are used (one for the network and one for the GPIO sections) to ensure that the data read is valid. If one value is incorrect then default values are loaded for that function.

When the FlashROM is reprogrammed by the FT90x Programming Utility it is normal for this data to be overwritten. This will result in default parameters being used until the data is reprogrammed.

2.4 Web Pages

The configuration web pages are stored in a folder called httpdfs on the top level of the directory structure for the project. The raw html and image files are compiled into a single C file (fsdata.c) which is included in the source code of the application. An internal library in the lwIP httpd code reads this data and extracts files for the web service.

A batch file (makefs.bat) and a compilation utility (makefsdata.exe) for Windows platforms are included to perform the compilation and place the fsdata.c file in the correct place. Changes to the web pages have to be compiled before the application is recompiled.

To make changes to the webpage, edit the htm or html files using a text editor. Knowledge of HTML code is required. The page must be rebuilt by browsing to the project's top level directory in a windows console and typing makefs.bat as shown in Figure 2.1.

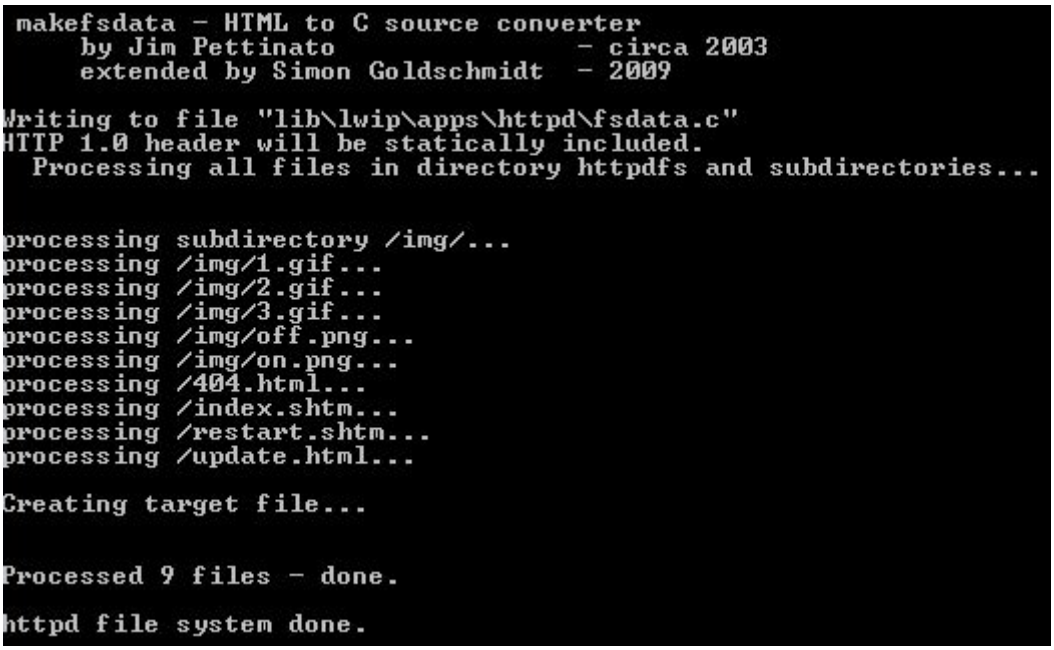

Figure 2.1 Makefs.bat Output

Further details on how lwIP performs Server Side Include (SSI) and acts on the HTTP GET requests used to update the configuration are to be found in the lwIP documentation.

2.5 Other Features

The DFU-C facility is enabled for this application. This is called from the macro STARTUP_DFU() in the main() function. It will briefly enable the USB device on the FT90x and allow a DFU utility to update the application code. This can be removed entirely or configured to alter the number of milliseconds it will wait before closing the USB device and continuing with the application.

3 Code Structure

Data sent and received from the network are handled by the lwIP library. Simple FT90x API functions used in the HTTPD SSI and CGI functions:

- gpio_dir. Sets the direction of the GPIO port to input or output.
- gpio_read. Reads the specified GPIO port.
- gpio_write. Writes to the specified GPIO port.

More information on these FT90x API functions can be found in AN_365 FT9xx API Programmers Manual.

Data received from the network like the GPIO number is used in each of the functions above and is handled in the HTTPD SSI and CGI functions.

The main function "gpio_bridge()" shows the main network setup and the lwIP library (Figure 3.1).

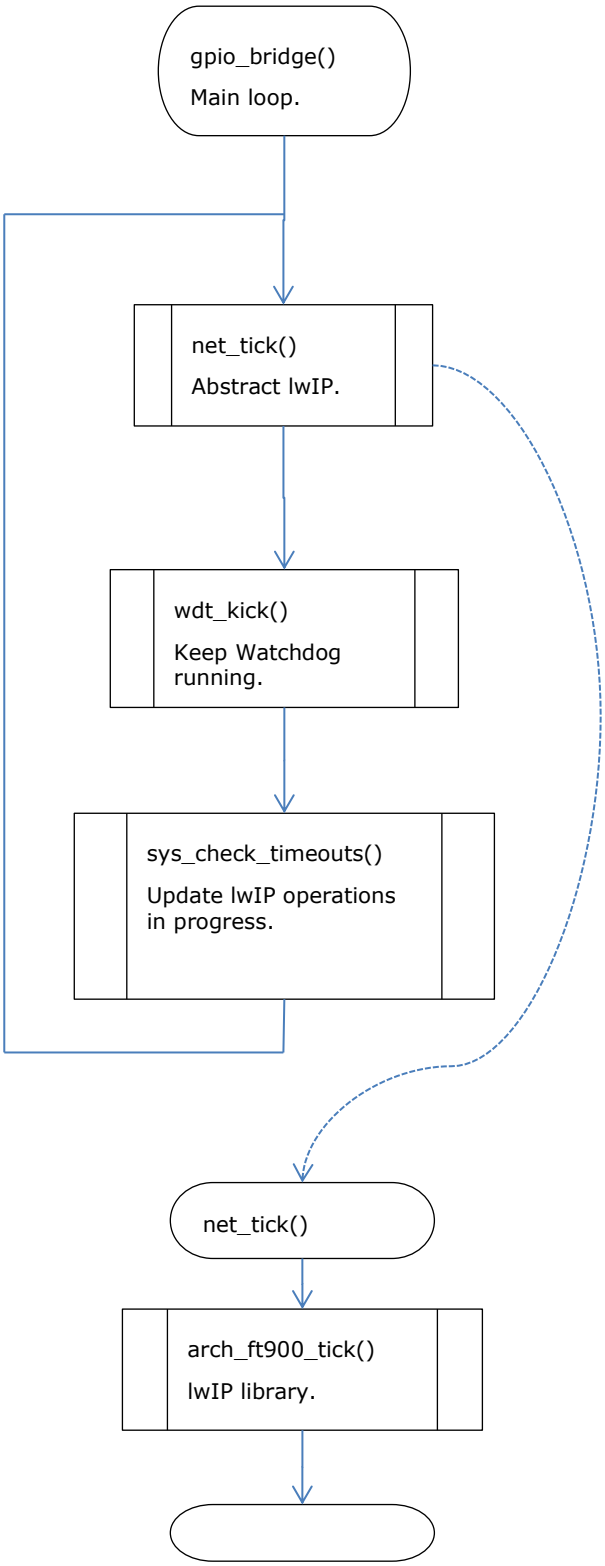

Figure 3.1 Main Function Flowchart

4 Configuration

This section describes the default configuration used by the application and how to change the configuration. Figure 4.1 shows a screen capture of the default configuration.

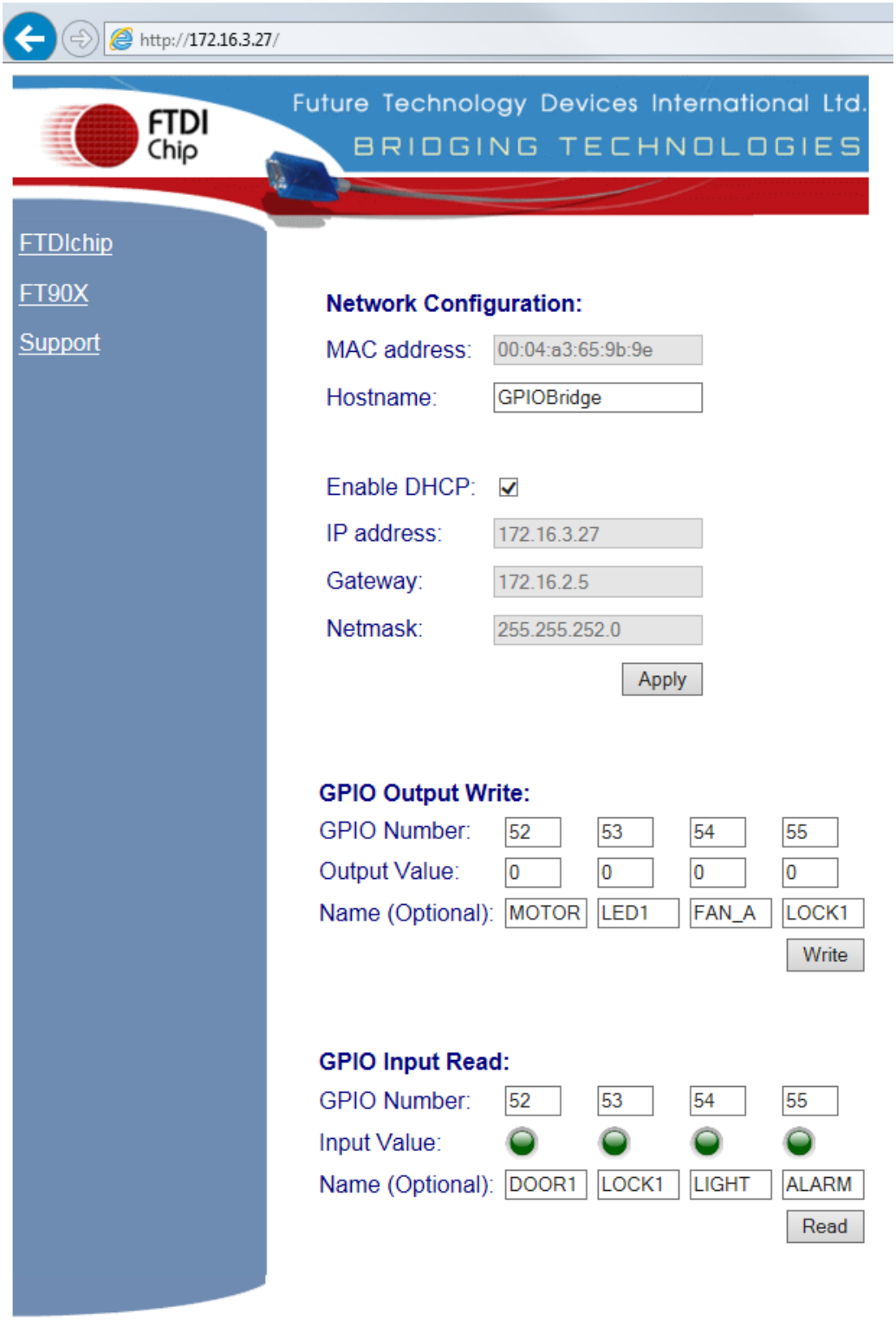

Figure 4.1 Web Configuration and Control Interface

A web configuration page is accessible when the device connects to the network. This allows both networking settings and GPIO control to be altered.

In order to serve the configuration web page the device must be connected to the network and have obtained (or have been configured with) an IP address. DHCP is enabled by default allowing the application to request an IP address when it is first started.

In many cases it is possible to query the DHCP server on a network to find out the IP address allocated to a particular device. In this application the device will send the DHCP server the hostname "GPIOBridge" which some DHCP servers may make available to allow users to identify the allocated address more easily.

Another way to obtain the IP address to access the web configuration page is to check the UART debug text, while NET_DEBUG is defined in net.c, for example:

```
IP config:
ipaddr = 172.16.3.115
gwaddr = 172.16.2.5
netmask = 255.255.252.0
```

4.1 Default Configuration

The MAC address is obtained from the onboard 24AA02E48T-I/OT EEPROM during initialization (see 'net_init' in net.c). This memory is present on the MM900EVxA , excluding the MM900EV-LITE. The pre-programmed permanently protected MAC address resides at the last 6 bytes of memory in the EEPROM.

Note: The code should be changed if you are using hardware that does not have this external EEPROM.

4.1.1 Network

DHCP is turned on by default. The application will ask for an IP address, gateway and netmask from any available DHCP server and use these values to configure the network interface. It will also provide a hostname of "GPIOBridge" to the DHCP server.

4.1.2 UART

The following settings are used by default for the UART which displays debug messages:

- 19200 baud
- 8 bits data
- 1 stop bit
- no parity

4.2 Modifying the Configuration

The configuration web page is split into 3 sections:

- a 'Network Configuration' section.
- a 'GPIO Output Write' section.
- a 'GPIO Input Read' section.

There are also shortcut links on the left to visit the FTDI web site, the FT90x section and the support information within the FTDI website.

4.2.1 Network Configuration

The network configuration section has 2 sub-sections.

The first is the MAC address and hostname. These are required for configuring the device. As has been mentioned earlier the MAC address is required by Ethernet to be unique on a network. This is the hardware address used for all low-level communications on Ethernet. A network will act in an unspecified manner if this is duplicated.

It is recommended that bit 2 in the most significant byte (first octet transmitted) be set to 1 to indicate that the MAC address is locally administered.

The hostname field is sent to the DHCP server during the DHCP negotiation and may be used by the server to identify a device. In some DHCP setups it can be passed onto DNS to allow the hostname to be used seamlessly within a local network.

The third network option is to enable or disable DHCP. When enabled, this will instruct the application to request an IP address, gateway address and netmask from a DHCP server. Therefore the fields that configure these values are disabled. The values obtained from the server are shown in the fields for information only.

If DHCP is disabled then the 3 fields are enabled allowing specific addresses to be configured.

There is no error checking that the values in the form are correctly formatted. The IP address, gateway and netmask are 4 decimal values separated by dots.

4.2.1.1 Programming the Network Configuration

To program the Network Configuration via the configuration webpage, make edits as required then click on the 'Apply' button, then click on restart on the next page. Note that the FT90x IC will be rebooted.

In Section 2.3 it is states that the configuration parameters are overwritten when programming new code to the FlashROM. This is because the FT90x Programming Utility will write the entire FlashROM when a new image is programmed. This will overwrite the datalogger partition with blank formatting.

However, when using the FT90x Programming Utility to configure one or many devices, a "Config File" can be specified and used to store preconfigured parameters into the datalogger partition.

The address of the partition will be 8 kB from the top of FlashROM (0x3E000).

An existing datalogger partition can be copied from an FT90x device using the "Data Log" tab in the FT90x Programming Utility. The binary file obtained can be modified to change any of the configurable parameters using the structs "net_config" and "gpio_config" defined in the source code.

The network config is at an offset of 0x100 in the datalogger file and the GPIO config is at offset 0x200. Both structures have a 32-bit signature field to validate the values stored. The first and last 256 bytes of the datalogger file have required validation data, do not modify this.

4.2.2 GPIO Output Write

Four GPIO output ports can be controlled at the same time. This amount was set as an example and can be changed in the application code.

Default port numbers are set to 52, 53, 54 and 55 since these are UART0/UART1 pins which are not used in this example.

An optional name can be given to the port number so that it is readable and user friendly. The maximum number of characters is 5 but this can be changed in the source code if required.

To control GPIO ports as output, enter the port number between 0 and 66 and click on 'Write'. If an invalid number is entered and Write is clicked, the process will not complete as shown in Figure 4.2.

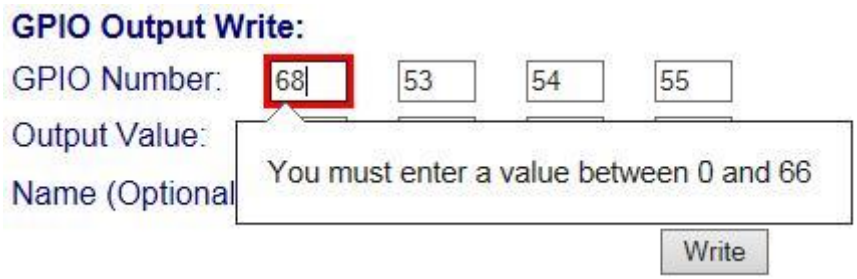

Figure 4.2 GPIO Output Port Error

On a successful write, the GPIO port will be set as shown in Figure 4.3, and the GPIO number and name will be stored in flash memory so they are not lost after power off.

Note that all four ports will be set when clicking 'Write'.

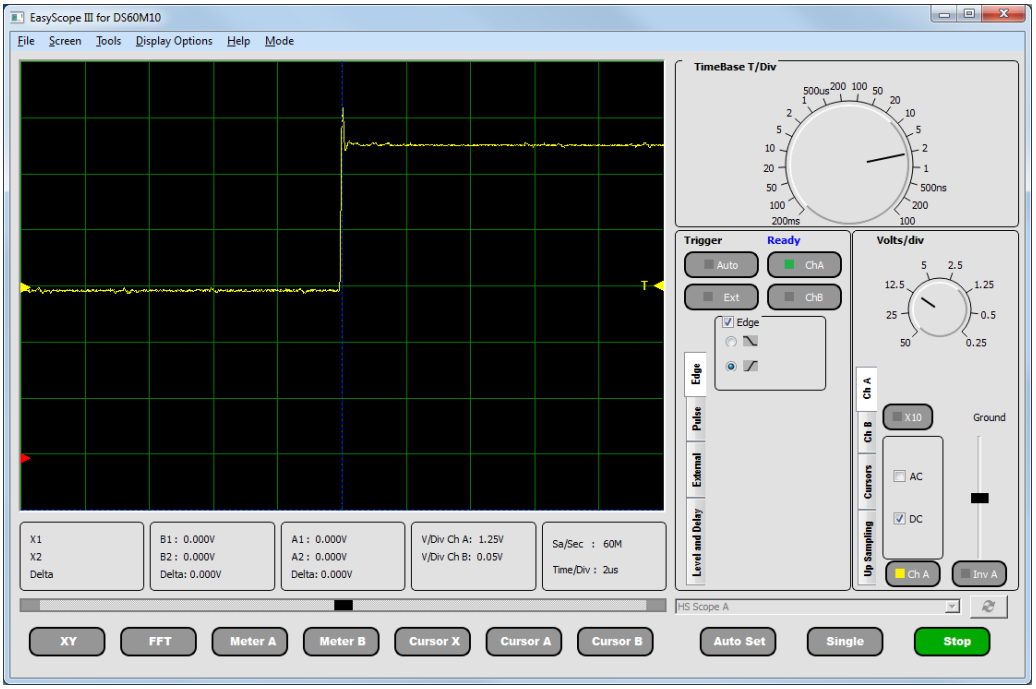

Figure 4.3 GPIO Output toggle

This oscilloscope capture was taken using an [EasySYNC USB Oscilloscope].

The FT90x API function gpio_dir is used to set the entered GPIO Number to be an output, and gpio_write is used to write the entered output value to the GPIO port. These are fairly simple

functions to use and are integrated within the HTTPD SSI and CGI functions in the application code.

More information on these FT90x API functions can be found in AN_365 FT9xx API Programmers Manual.

4.2.3 GPIO Input Read

Four GPIO input ports can be read at the same time. This amount was set as an example and can be changed in the application code.

Default port numbers are set to 52, 53, 54 and 55 since these are UART0/UART1 pins which are not used in this example.

An optional name can be given to the port number so that it is readable and user friendly. The maximum number of characters is 5 but this can be changed in the source code if required.

To read GPIO input ports, enter the port number between 0 and 66 and click on 'Read'. If an invalid number is entered and Read is clicked, the process will not complete similar to Figure 4.2.

Figure 4.4 shows an example GPIO read showing ports 52 and 55 as low or OFF, and ports 53 and 54 as high or ON.

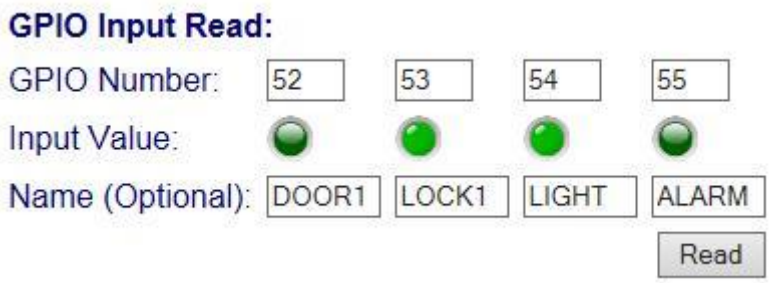

Figure 4.4 GPIO Input Read

The FT90x API function gpio_dir is used to set the entered GPIO Number to be an input, and gpio_read is used to decide on which LED image to display, either OFF or ON. These are fairly simple functions to use and are integrated within the HTTPD SSI and CGI functions in the application code.

More information on these FT90x API functions can be found in AN_365 FT9xx API Programmers Manual.

5 Importing into the FT9xx Toolchain

The AN_415 Firmware found at the following link can be easily imported into the FT9xx Toolchain:

http://brtchip.com/ft90x/#FT90x Ethernet to GPIO Bridge

Once installed, select File --> Import --> General --> Existing Projects into Eclipse, and point to the downloaded and extracted project directory.

The project will appear in Eclipse Project Explorer as shown in Figure 5.1.

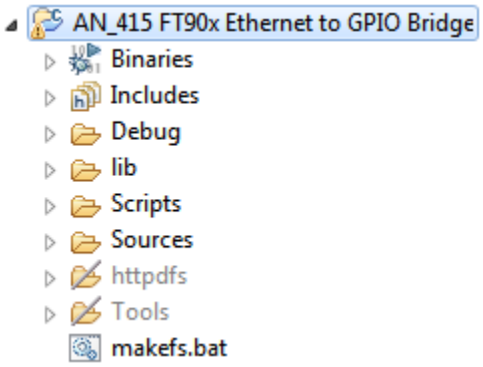

Figure 5.1 Eclipse Project Structure

5.1 Changing the Application Software

The application software provided can be altered and changed if required. The FT9xx Toolchain is a free tool to enable code development and debug for the FT90x series and is based on plug-ins for the free popular IDE using the GCC compiler.

For example, the default GPIO configuration can be changed so different port numbers and names can be displayed:

```
static struct gpio_config default_gpio_config = {
                GPIO_CONFIG_KEY,
                {52, 53, 54, 55},
                {52, 53, 54, 55},
                {'M', 'O', 'T', 'O','R',0,},
                {'L', 'E', 'D', '1',0,0,},
                {'F', 'A', 'N', '_','A',0,},
                {'L', 'O', 'C', 'K','1',0,},
                {'D', 'O', 'O', 'R','1',0,},
                {'L', 'O', 'C', 'K','1',0,},
                {'L', 'I', 'G','H','T',0,},
                {'A', 'L', 'A', 'R','M',0,},
};
```

With each software change, the project should be rebuilt and reprogrammed into the FT90x IC. Please refer to AN_325 FT9xx Toolchain Installation Guide for further information.

6 Contact Information

Head Quarters – Singapore

Bridgetek Pte Ltd
178 Paya Lebar Road, #07-03
Singapore 409030
Tel: +65 6547 4827
Fax: +65 6841 6071

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Branch Office – Taipei, Taiwan

Bridgetek Pte Ltd, Taiwan Branch
2 Floor, No. 516, Sec. 1, Nei Hu Road, Nei Hu District
Taipei 114
Taiwan , R.O.C.
Tel: +886 (2) 8797 5691
Fax: +886 (2) 8751 9737

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Branch Office - Glasgow, United Kingdom

Bridgetek Pte. Ltd.
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

E-mail (Sales) sales.emea@brtichip.com
E-mail (Support) support.emea@brtchip.com

Branch Office – Vietnam

Bridgetek VietNam Company Limited
Lutaco Tower Building, 5th Floor, 173A Nguyen Van Troi,
Ward 11, Phu Nhuan District,
Ho Chi Minh City, Vietnam
Tel : 08 38453222
Fax : 08 38455222

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Web Site

http://brtchip.com/

Distributor and Sales Representatives

Please visit the Sales Network page of the Bridgetek Web site for the contact details of our distributor(s) and sales representative(s) in your country.

Appendix A – References

Document References

FT90x Datasheet: http://brtchip.com/m-ft9/

lwIP http://savannah.nongnu.org/projects/lwip/

tinyprintf http://www.sparetimelabs.com/tinyprintf/tinyprintf.php

AN_325 FT9xx Toolchain Installation Guide

AN_365 FT9xx API Programmers Manual

DFU AN_380 FT900 Bootloader DFU Usage

TN_157 Ethernet Explained

EasySYNC USB Oscilloscope

FT9xx Toolchain
AN_415 Firmware: http://brtchip.com/ft90x/#FT90x Ethernet to GPIO Bridge

24AA02E48T-I/OT EEPROM http://www.microchip.com/wwwproducts/en/24AA02E48

MM900EVxA http://brtchip.com/ft9xx-development-modules-landing/

Acronyms and Abbreviations

Terms	Description
CGI	Common Gateway Interface
DFU	Device Firmware Update
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
MAC	Media Access Controller
SSI	Server Side Include
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver and Transmitter (Serial Port)
GPIO	General Purpose Input Output
EEPROM	Electrically Erasable Programmable Read Only Memory
I2C	Inter-Integrated Circuit

Appendix B – List of Tables & Figures

List of Tables

List of Figures

Appendix C – Revision History

Document Title: AN_415 FT90x Ethernet to GPIO Bridge
Document Reference No.: BRT_000104
Clearance No.: BRT#062
Product Page: http://brtchip.com/m-ft9/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2017-02-17
1.1	Updated AN_415 firmware download links to be generic since they were broken in version 1.1.	2017-04-04