Application Note

BRT_AN_024

FT9xx with ESP32

Version 1.0

Issue Date: 2017-10-31

This Application Note describes an FT9xx device which implements simple network server functionality using an ESP32 to provide access to a Wi-Fi network. An FT81x display is used to configure the ESP32 network interface and display short text messages received from a network client.

Bridgetek Pte Ltd (BRTChip)
178 Paya Lebar Road, #07-03, Singapore 409030
Tel: +65 6547 4827 Fax: +65 6841 6071
Web Site: http://www.brtchip.com

Table of Contents

1 Introduction 3
1.1 Overview 3
1.2 Scope 4
1.2.1 Features 4
1.2.2 Possible Enhancements 4
2 Project Overview 5
2.1 Sources Folder 5
2.2 eve and eve_arch_ft9xx Folder 6
2.3 eve_ui Folder 6
2.4 esp32 Folder 6
3 Configuring Hardware 7
3.1 Required Hardware 7
3.2 Installing ESP32 Firmware 8
4 Use of Application Note Software 9
4.1 Buttons 9
4.2 Joining an Access Point 9
4.3 Settings 10
4.4 Receiving Messages 11
4.5 Responding to Messages 12
5 Contact Information 13
Appendix A– References 14
Document References 14
Acronyms and Abbreviations 14
Appendix B – List of Tables & Figures 15
List of Tables 15
List of Figures 15
Appendix C– Revision History 16

1 Introduction

This Application Note describes interfacing the FT9xx to Espressif System's ESP32 device. The AT Instruction Set Firmware is used on the ESP32 to abstract the configuration and operation of Wi-Fi networks.

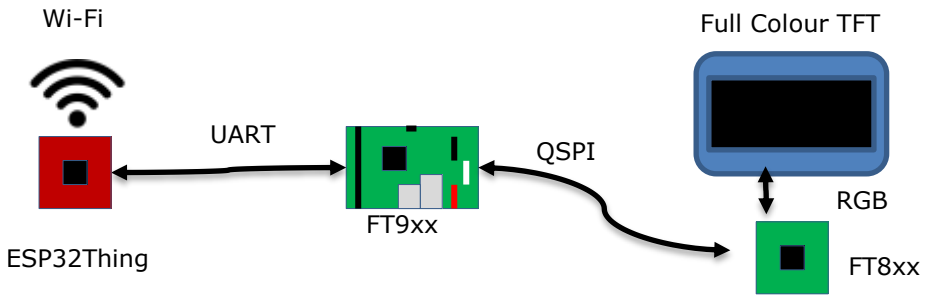

Figure 1 Block Diagram

The FT9xx and the ESP32 communicate with a UART interface. AT Commands are sent from the FT9xx to the ESP32 and the responses received. A library is provided to manage the AT Command generation and parse the responses.

The included code is able to use the station mode of the ESP32 and as such be able to list and connect to Wi-Fi networks; send and receive data over TCP, UDP or SSL; and manage the configuration of the ESP32.

A re-usable graphical interface for configuring the ESP32 is provided for the FT9xx/FT81x allowing a Wi-Fi Access Point to be selected, Wi-Fi password to be entered and IP addresses to be entered.

The document should be read in association with the example code provided in the references section.

1.1 Overview

This document describes the design and implementation of the FT9xx to ESP32 interface. The FT9xx code provides library code to:

- Implement a high speed UART interface to the ESP32 using ring buffering and flow control.
- Generate AT commands and keep track of the AT firmware state on the ESP32.
- Receive and parse the responses to AT commands from the ESP32.
- Receive and parse asynchronous status messages from the ESP32.
- Provide a graphical user interface to allow configuration of the ESP32 Wi-Fi.

This document is intended to demonstrate the capabilities of the FT9xx family of microcontrollers and FT81xx display controllers when leveraging the capabilities of the ESP32.

A simple text messaging server is provided to demonstrate an application for this projection. Once configured and connected to a Wi-Fi network the code will wait for a TCP connection from a remote system and display text received from it on a 'console'. A virtual keyboard can be used to send a reply to the remote system.

Third-party open source code is used to implement this application note:

- Printf – tinyprintf.

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

1.2 Scope

The AT command library supplied for the FT9xx demonstrates the use of the ESP32 AT command set to provide Wi-Fi connectivity to projects. The Wi-Fi configuration interface for the FT81x provides a set of expandable features for both Wi-Fi configuration and additional user interfaces for a project.

1.2.1 Features

The application note shows how to implement a UART ring buffer with flow control between the FT9xx and the ESP32. It also demonstrates the use of an FT81x display to provide configuration data and a user interface.

FreeRTOS is used with message queues to abstract the FT81x interface from the ESP32 interface allowing flexible user input and output in response to networking events.

1.2.2 Possible Enhancements

This application note can be seen as a start for customisation or extension. Some example enhancements could be:

- Tailor user interface for a particular application. E.g. Point of Sale or Remote Messaging.
- Add multimedia display information from remote network client.
- Implementing both a client and a server version or a peer network of devices to support messaging.

2 Project Overview

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

Folder	Description
Source	Application source code and abstraction files.
Includes	Application specific header files.
Images	JPEG images for this application.
Scripts	FT9xx build script which includes DLOG partition.
lib	Library files.
lib\esp32	ESP32 AT Command library.
lib\eve lib\eve_arch_ft9xx	EVE API libraries.
lib\eve_ui	EVE User Interface libraries.
lib\tinyprintf	tinyprintf library.
lib\uartrb	Ring buffer implementation for FT9xx.
lib\FreeRTOS	Library files for FT9xx port of FreeRTOS.

Table 1 - Project Files Overview

2.1 Sources Folder

The main part of the application is found in the "Sources" folder. This is split into 3 main sections and has 3 source code files.

- The "main.c" file is generally responsible for setting up FreeRTOS and starting tasks for the FT81x and ESP32 interfaces;
- The ESP32 FreeRTOS task is implemented in the file "at_monitor.c";
- The FT81x FreeRTOS task is implemented in the file "eve_monitor.c";

The other 2 files in this folder are:

- "images.c" contains JPEG images encoded into C array declarations. These images are for the FT81x user interface;
- "fonts_extended.c" which has custom characters required by the keyboard component in the FT81x user interface;
- "freertos_d2xx_dlog_crt0.S" a modified startup file (in FT9xx assembly language) to allow the application to write to a protected section of FlashROM on the device.

Files in these folders use the "Includes" folder for application specific header files.

2.2 eve and eve_arch_ft9xx Folder

This folder holds the FT8xx API code which abstracts the FT8xx register and processing list writes into C functions. The API code is based on the "BRT AN 008 FT81x Creating a Simple Library For PIC MCU" application note.

2.3 eve_ui Folder

The source files in "lib\eve_ui" ("eve_ui_main.c", "eve_ui_keyboard.c", "eve_ui_choice.c", "eve_ui_multiline.c") implement a re-usable user interface for the FT81x comprising a keyboard, a menu system and a textual console, this is called by the FT81x FreeRTOS task.

2.4 esp32 Folder

An AT command library is provided in the file "at.c", this uses a ring buffer library which is coded in "uartrb.c".

3 Configuring Hardware

In this section the method for connecting the FT81x, FT9xx and ESP32 required to use the application code is discussed.

3.1 Required Hardware

The FT81x device used for this application will typically be on an ME812A-WH50R or ME813A-WH50C EVE development module with an MM900EV2A or MM900EV3A MCU module. The application note is written to work on an 800 by 600 display. Changes to the EVE module or FT9xx device can be made in the "EVE_config.h" file in the "eve" folder of the source code.

The MM900EVxA module connects directly to the EVE development module with a set of pin headers. The QSPI interface on the FT9xx device is taken through the pin headers to the FT81x on the EVE module. The host PC connects via USB to the MM900EVxA module.

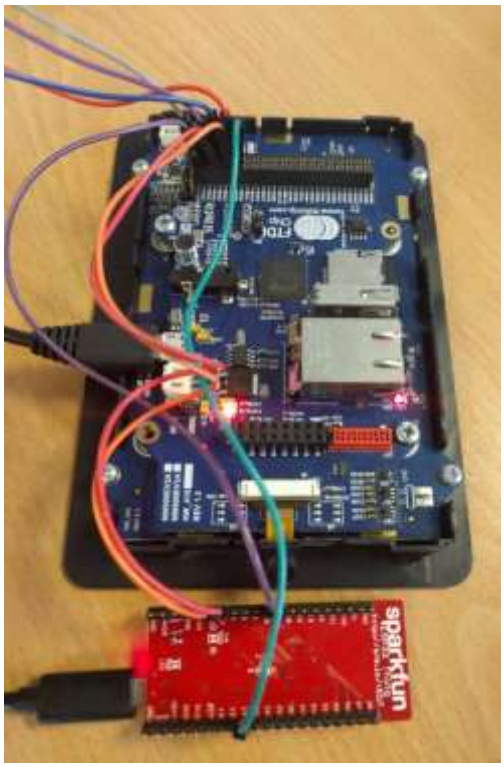

Figure 2 MM900EV2A and ME812A-WH50R module with ESP32

A SparkFun ESP32Thing module is used although other ESP32 modules are available. This is connected to the UART1 interface on the FT9xx. The UART0 interface is used for simple debugging as in other FT9xx application notes.

Six jumper wires are required to connect the ESP32 to the FT9xx if power is derived from the FT9xx. Four connectors are required if the FT9xx and ESP32 are both self-powered via their USB connectors.

The photograph in Figure 2 shows the connector wires from CN3 on the MM900EV2A board to the ESP32Thing module. Both modules are powered from USB. The required connections are listed in Table 2. The pin names are used for the ESP32 module as various versions are available.

ESP32	MM900EV2A	Description
GPIO16	CN3 9	ESP32 RXD FT9xx UART1 TXD
GPIO17	CN3 7	ESP32 TXD FT9xx UART1 RXD
GPIO14	CN3 11	ESP32 RTS FT9xx UART1 CTS
GPIO15	CN3 3	ESP32 CTS FT9xx UART1 RTS

Table 2 – Connections from MM900EV2A to ESP32 Module

3.2 Installing ESP32 Firmware

The ESP32 must have the ESP-AT firmware installed. This firmware is available from the Espressif web pages and may come pre-installed on some ESP32 modules.

Refer to the instructions from Espressif Systems for installation instructions and updates to the firmware.

4 Use of Application Note Software

4.1 Buttons

The active buttons in the application are as follows:

-  Settings
-  Cancel
-  Refresh/Reload
-  OK
-  Keypad
-  Keyboard

4.2 Joining an Access Point

When started, the application will ask to join a Wi-Fi access point.

The ESP32 will be queried to find available access points. The FT9xx application will present a list of access points. A typical list of access points are shown in Figure 3. When a suitable access point is selected then it will progress to ask for the password for the access point as shown in Figure 4. If there is no password required then this should be left blank.

Figure 3 Choose Access Point Screen

Wi-Fi access points and IP configuration are stored in non-volatile memory on the ESP32. Therefore the configuration cycle will only be entered if the previous Wi-Fi access point cannot be found.

Figure 4 Access Point Password Screen

4.3 Settings

The settings button will allow the current network configuration to be modified. Settings will be entered whenever there is no connection to an access point. The settings screen is shown in Figure 5.

The first option is to Connect or Disconnect from an Access Point. When connecting the menu structure used in Section 4.2 will be used.

When DHCP is enabled the "Address", "Mask", and "Gateway" options will be disabled. When DHCP is disabled then these may be modified to enter IPv4 configuration data. The values will be stored in non-volatile memory by the ESP32.

The "Timeout" and "Port" settings are not persistent and will be reset after a power cycle.

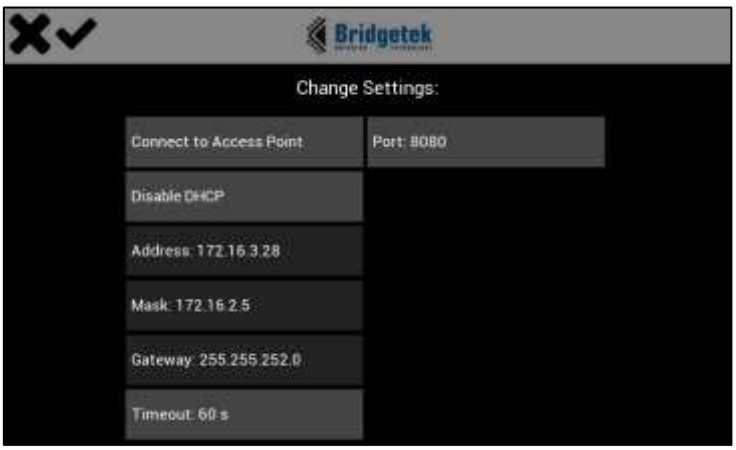

Figure 5 Settings Screen

4.4 Receiving Messages

Once the network is configured the application will listen on port 8080, by default, for remote connections. Once a connection is established then any data packets received will be added to the console screen. It is assumed that plain text is being sent from the remote client.

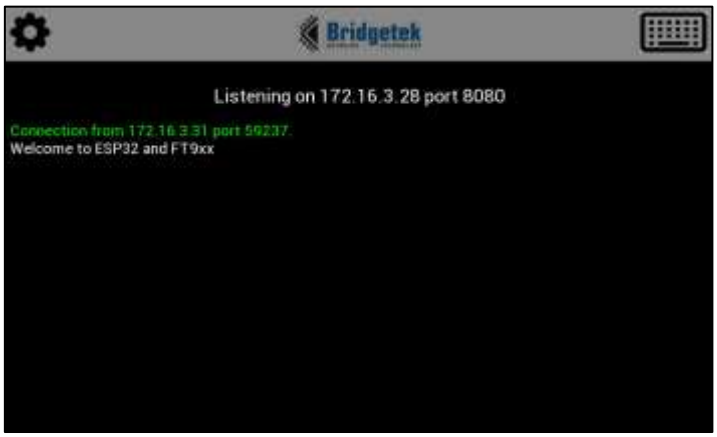

Figure 6 Receiving Messages Screen

A convenient program to use on a remote client is called "Packet Sender". This allows arbitrary TCP connections to be made to remote servers and is useful for testing purposes. A typical screen is shown in Figure 7.

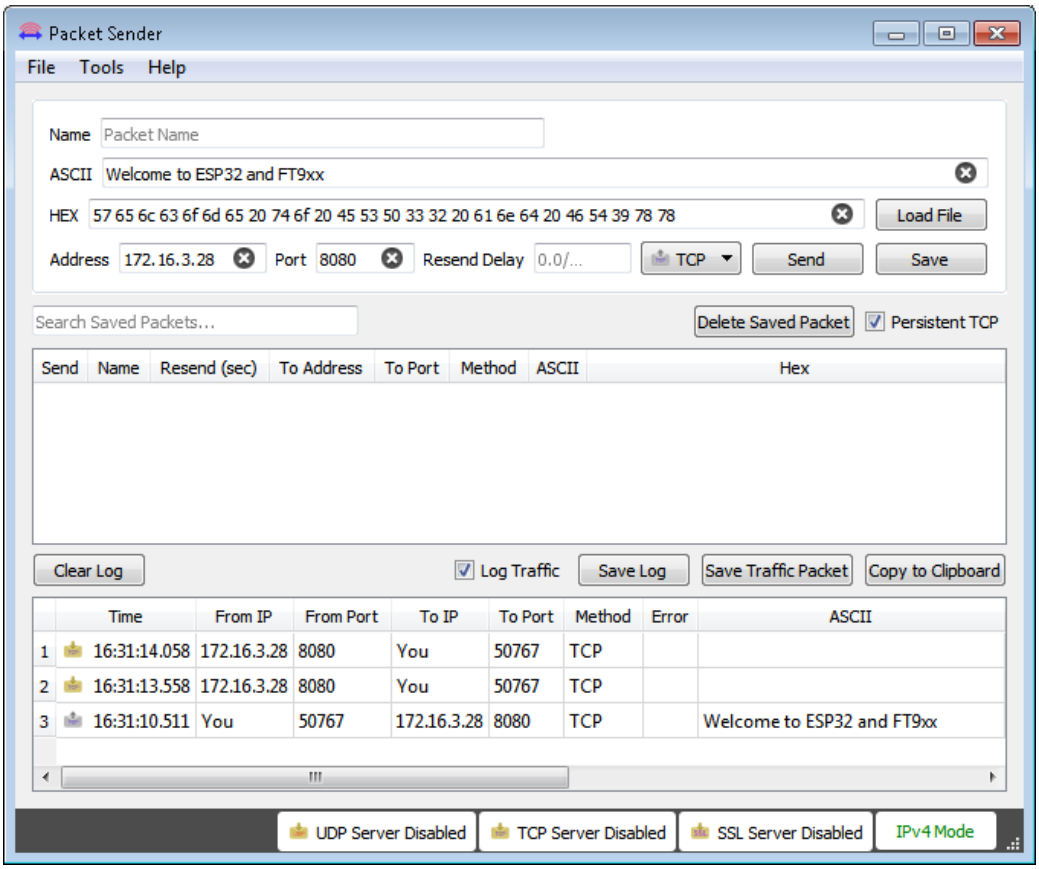

Figure 7 Packet Sender

4.5 Responding to Messages

Once a connection from a client has been established then the application can respond with a text message. Clicking on the keyboard button in the Receiving Messages screen will bring up a choice of which connected client to respond to. After one of the active clients is chosen the keyboard will appear allowing a response to be typed. This is shown in Figure 8.

Figure 8 Response Keyboard Screen

5 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

FT900/901/902/903 Datasheet

FT905/906/907/908 Datasheet

FT930/931/933 Datasheet

FT81x Datasheet

MM900EVxA datasheet

AN_324 FT9xx User Manual

BRT_AN_xxx_FT9xx_with_ESP32 Source Code V1.0

Espressif Systems ESP-AT firmware http://espressif.com/en/products/software/esp-at/resource

Packet Sender utility https://packetsender.com/

Acronyms and Abbreviations

Terms	Description
API	Application Programming Interface
EVE	Embedded Video Engine
IC	Integrated Circuit
JPEG	Joint Photographic Experts Group
MCU	Microcontroller Unit
QSPI	Quad Serial Peripheral Interface
RAM	Random Access Memory
RGB	Red Green Blue (Color Model)
TTF	True Type Font

Appendix B – List of Tables & Figures

List of Tables

Table 1 - Project Files Overview 5
Table 2 – Connections from MM900EV2A to ESP32 Module 8

List of Figures

Figure 1 Block Diagram 3
Figure 2 MM900EV2A and ME812A-WH50R module with ESP32 7
Figure 3 Choose Access Point Screen 9
Figure 4 Access Point Password Screen 10
Figure 5 Settings Screen 10
Figure 6 Receiving Messages Screen 11
Figure 7 Packet Sender 11
Figure 8 Response Keyboard Screen 12

Appendix C– Revision History

Document Title: BRT_AN_024 FT9xx with ESP32
Document Reference No.: BRT_000210
Clearance No.: BRT#099
Product Page: http://brtchip.com/product/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial version	2017-10-31