Application Note

BRT_AN_031

FT90x UART to USB BOMs Memory Bridge

Version 1.0

Issue Date: 2019-06-14

This Application Note demonstrates an FT90x device bridging its UART interface to a USB BOMs Memory device like a flash stick. The flash stick can be controlled via the UART interface through a series of commands.

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

This Application Note demonstrates an FT90x device bridging its UART interface to a USB BOMs Memory device like a flash stick. The flash stick can be controlled via the UART interface through a series of commands.

1.1 Overview

This document describes the design and implementation of the FT90x UART to USB BOMs Memory Bridge code. This code allows a user to:

- Control a flash stick connected to the USB Host port via UART0.
- Use various file, directory and other commands to create, edit and read to the flash stick.
- Enable a USB DFU interface to allow for firmware upgrade over the USB peripheral interface.

This document is intended to demonstrate the capabilities of the FT9xx family of microcontrollers by hosting a flash stick and bridging it to the UART interface.

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

- Printf – tinyprintf.
- FatFs – generic FAT Filesystem Module

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

1.2 Scope

This application note implements a USB Host and UART interface. The FT90x program detects a USB flash stick connected to the USB Host and controls it on the FT90x's UART interface.

The device connected to the UART interface can control the USB flash stick via a series of commands.

This has been tested with FAT32, ExFAT and 4GB, 8GB and 64GB flash sticks.

FatFs supports LFN (Long File Name).

1.2.1 Features

This application note shows how to implement a USB Host with BOMs class device connected.

The USB Host interface is used to control the flash stick with information received via the UART interface.

1.2.2 Possible Enhancements

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

- Implement a custom FAT filesystem since FatFS has some limitations.
- Extend the UART interface to include SPI.
- Add dedicated LEDs for status rather than using the Ethernet Port LEDs.
- Extend to allow for multiple flash sticks to be connected via USB Hub.

2 Project Overview

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

Folder	Description
Sources	Application source code.
Includes	Application specific header files.
lib	Library files.
lib\tinyprintf	tinyprintf library.
lib\fatfs	FatFs library.

Table 2.1 - Project Files Overview

2.1 Sources Folder

The main part of the application is found in the "Sources" folder.

The "main.c" file is generally responsible for the FT9xx setup including USB Host, UART, timers, interrupts, etc. FatFs is also interfaced here.

This uses the "Includes" folder for application specific header files.

3 System Block Diagram

The overall system block diagram is shown in Figure 3.1.

The software interacts with the following peripherals:

- USB Host to communicate with the flash disk
- UART0 to communicate with the external UART interface
- RTC for FatFs filestamp data
- Timers for USB Host controller
- Interrupt Controller for receive UART data and USB Host
- GPIO Controller for status LEDs

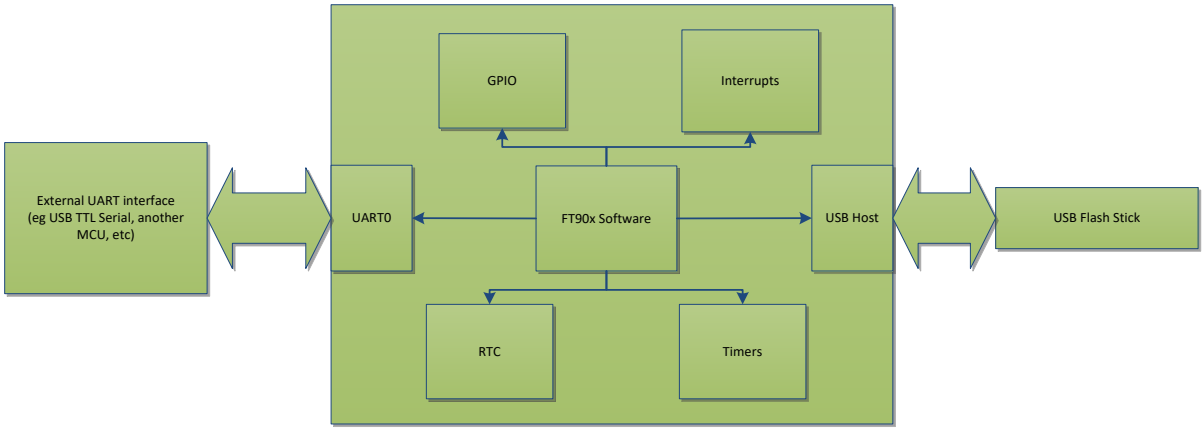

Figure 3.1 –Block Diagram

4 Software Flow Chart

The main function of the software flow chart is shown in Figure 4.1.

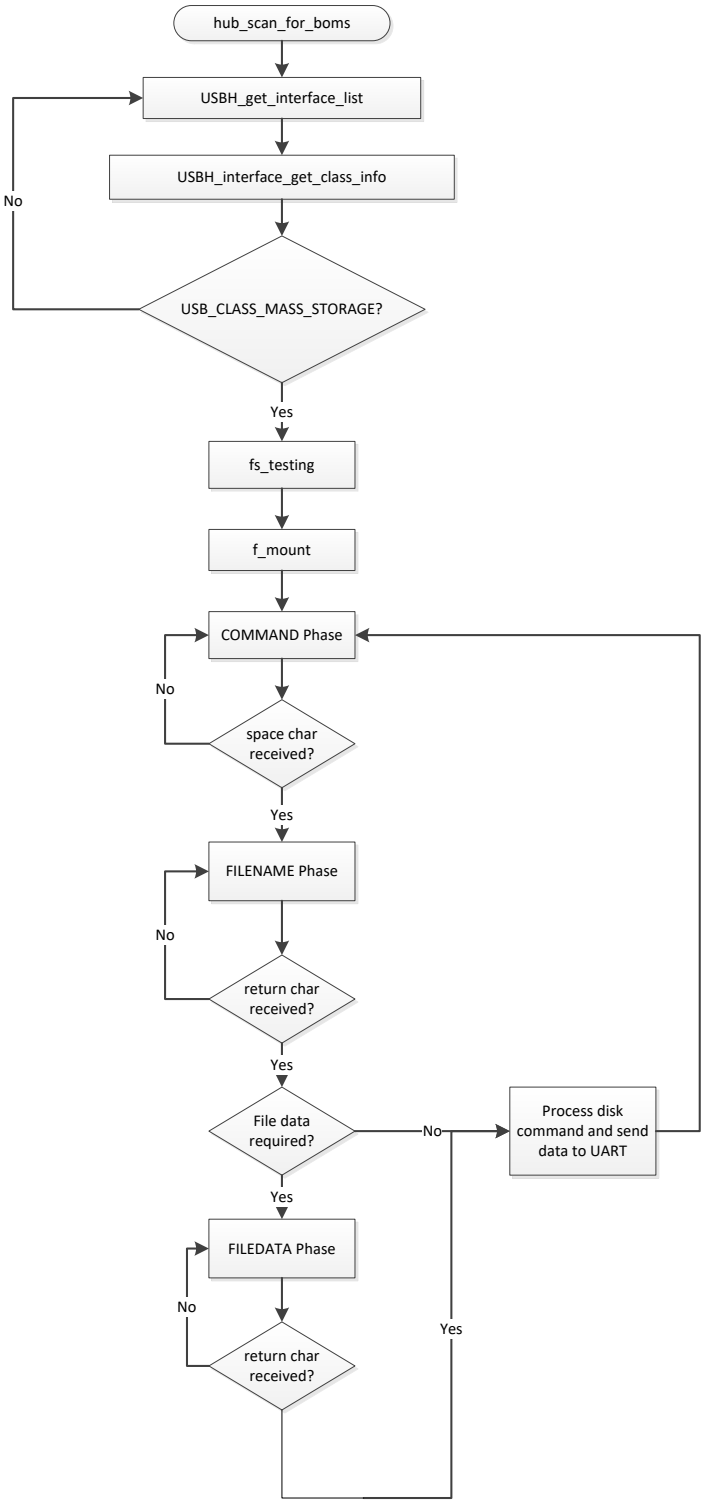

Figure 4.1 –Software Flow Chart

5 FatFs Library

FatFs is a generic FAT/exFAT filesystem module for small embedded systems. The FatFs module is written in compliance with ANSI C (C89) and completely separated from the disk I/O layer. Therefore it is independent of the platform.

It has been incorporated into the FT9xx MCU in this example code.

Key Features are:

- DOS/Windows compatible FAT/exFAT filesystem.
- Very small footprint for program code and work area.
- Various configuration options to support for: Long file name in ANSI/OEM, UTF-16 or UTF-8.
- exFAT filesystem.
- Thread safe for RTOS.
- Multiple volumes (physical drives and partitions).
- Variable sector size.
- Multiple code pages including DBCS.
- Read-only, optional API, I/O buffer and etc...

More information can be found at the following link:

http://elm-chan.org/fsw/ff/00index_e.html

The version implemented in this software is R0.13a.

Users could write their own filesystem or use another which can be incorporated into an embedded MCU.

6 Using the FT90x UART to USB BOMs Memory Bridge

6.1 Required Hardware

This application note is intended to be used on an MM900EVxA or MM900EVxB MCU module.

The UMFTPD2A debugger/programmer module is used for programming and debugging, however Port C can also be used as a USB to UART interface, allowing for ease of testing and evaluation. Otherwise USB TTL Serial or TTL 234 Serial could also be used. The required UART connections when using the UMFTPD2A are shown in Table 6.1.

UMFTPD2A Pin	Signal	MM900EVxA / MM900EVxB Pin	Signal
J2-2	CTS#	CN3-8	RTS#
J2-4	TXD	CN3-6	RXD
J2-5	RXD	CN3-4	TXD
J2-6	RTS#	CN3-10	CTS#

Table 6.1 – UMFTPD2A UART Connections

The MM900EVxA or MM900EVxB module connects to the UMFTPD2A via the supplied Micro-MaTch connector. Both modules are connected to the PC via Micro USB connectors.

The FT90x UART interface can be connected to the UMFTPD2A Port C interface by jumper wires.

The USB BOMS device (flash stick) is connected to CN6 of the MM900EVxA or MM900EVxB.

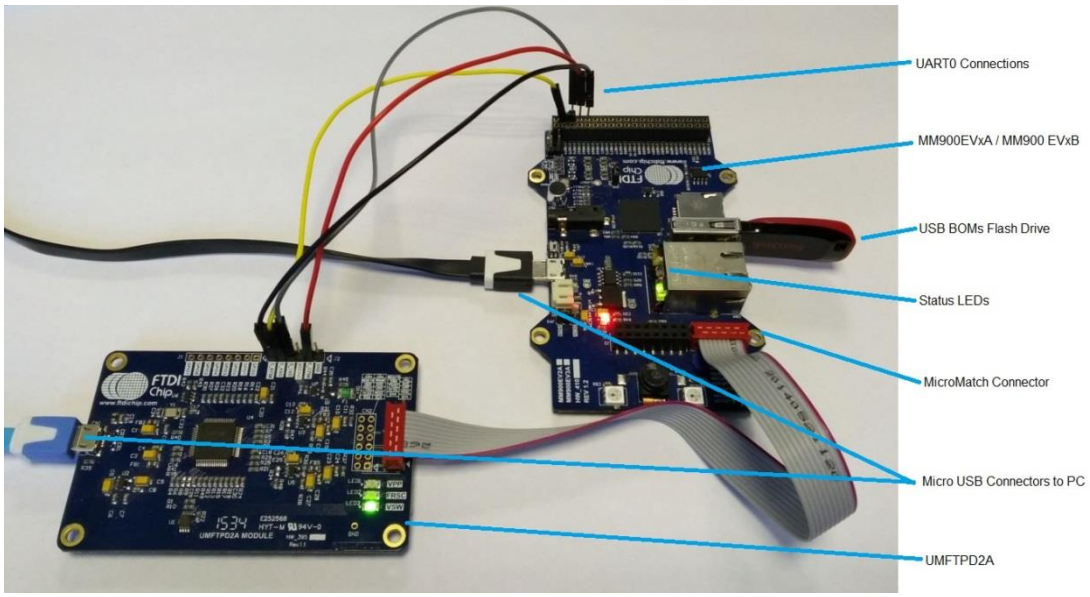

Figure 6.1 -MM900EVxA or MM900EVxB and UMFTPD2A Setup

6.2 Debug and Release Builds

There are two build options available as shown in Figure 6.8.

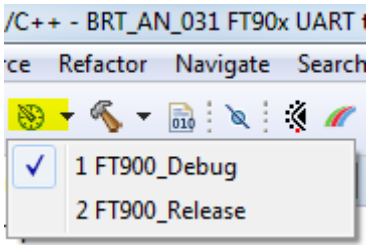

Figure 6.2 –Build Options

The behaviour of each build affects the data which is sent to the UART interface. The debug build contains a lot more information, where the release version has a cut down version making it easier for connected hardware to decode the information returned.

A comparison is shown in Figure 6.3 when a file is accessed but does not exist.

```
>

rdf BRT1.TXT

Opening for reading

ERROR RDF 1 4

Error opening for reading

Error Code 4: FR_NO_FILE



>

rdf BRT1.TXT

ERROR RDF 1 4
```

Figure 6.3 –Debug vs Release UART Output

6.3 Status LEDs

On the MM900EVxA or MM900EVxB module there are no user specific LEDs that can be used.

However, inside the Ethernet connector there are two LEDs, orange and green which can be controlled by software. The following functions are used in the code to do this:

```
/* Toggle LEDs */
toggle_LEDs();

/* Set LEDs */
gpio_write(4, 1);
gpio_write(5, 0);
```

A summary of the LED behaviour is shown in Table 6.2.

USB Host State	Description
No Device Connected	Green/Orange LEDs toggle
Device Inserted	Green LED is ON
Device Unmounted	Orange LED is ON. Green LED is off
Device Disconnected	Green/Orange LEDs toggle

Table 6.2 –Status LEDs

6.4 Use of Application Note Software

A terminal program like PuTTY can be easily used to send and receive the UART data from the FT90x. The settings are shown in Figure 6.5.

The COM Port number can be found via device manager on the PC.

Note: There are four ports associated with the UMFTPD2A since this includes an FT4232H IC, so Port C would be the third port number for UART communication. This is shown in Figure 6.4.

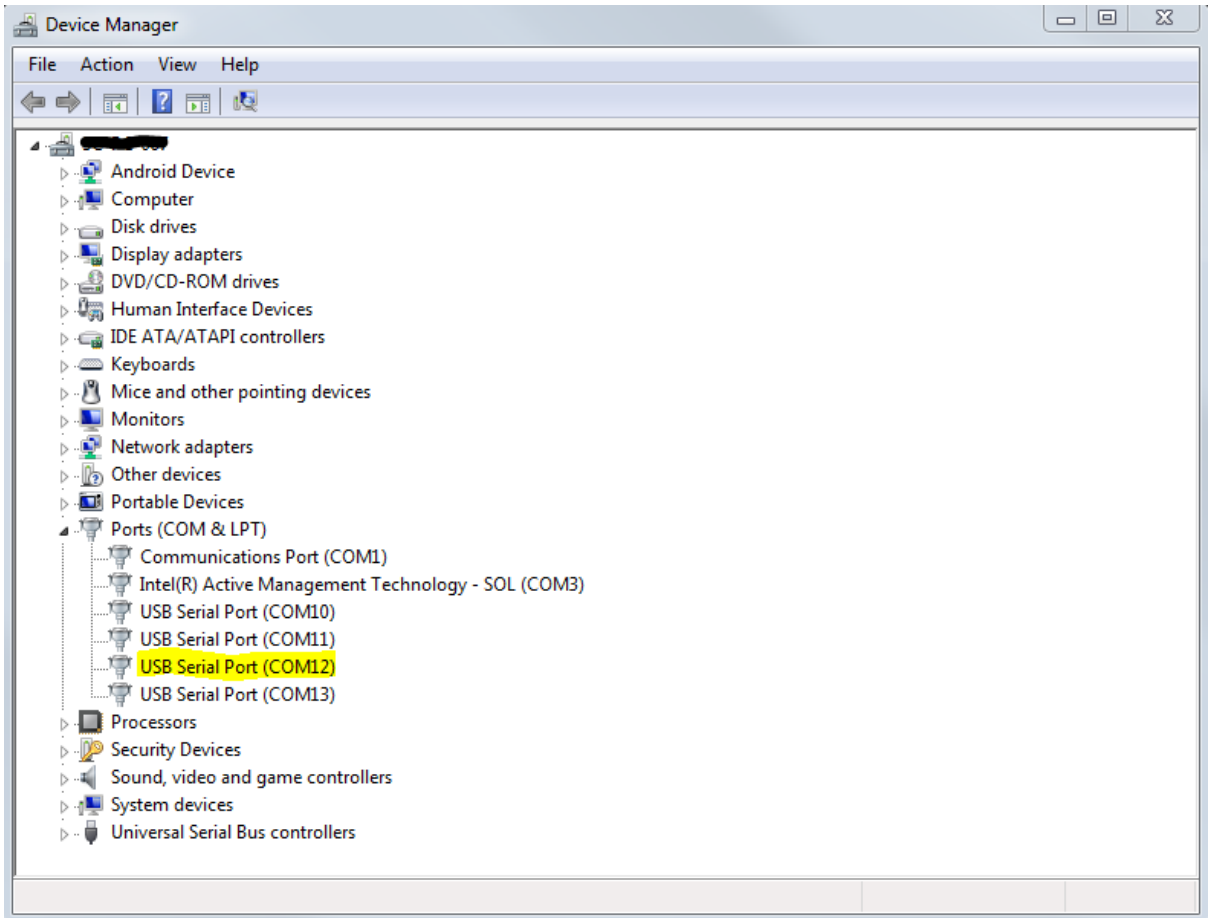

Figure 6.4 –Device Manager

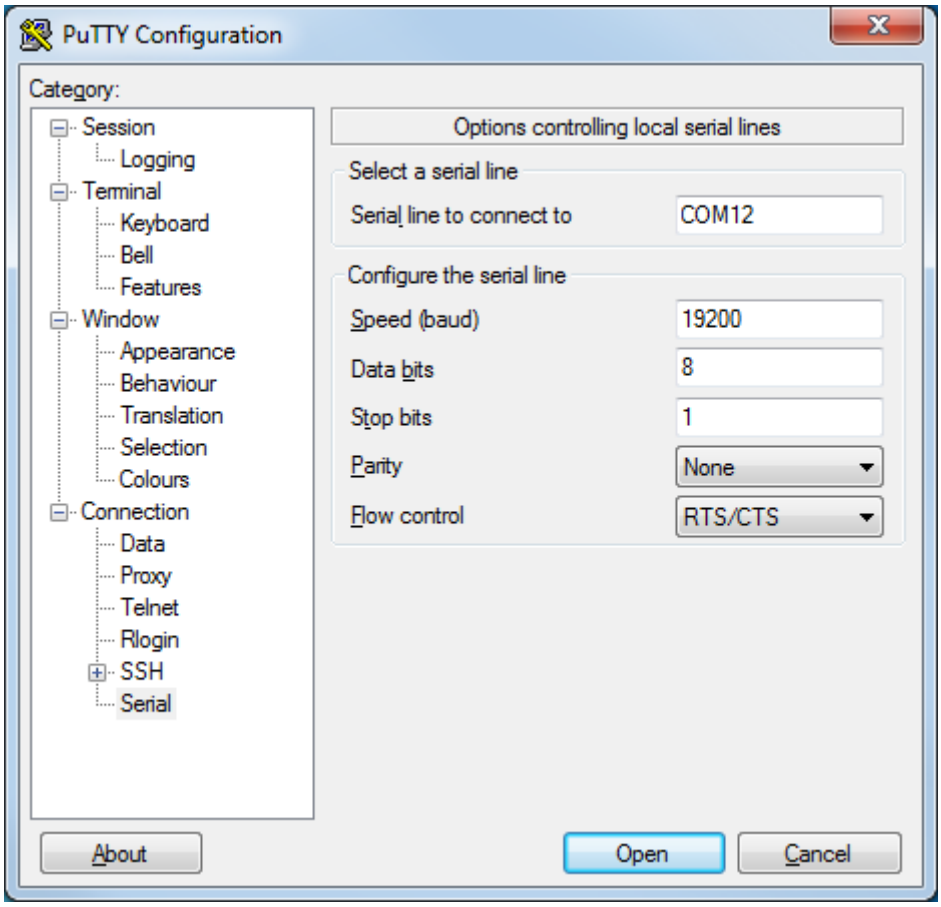

Figure 6.5 –PuTTY Configuration

When the MM900EVxA or MM900EVxB is powered up and the UART interface is connected to a PC (eg via the UMFTPD2A Port C), the following message in Figure 6.6 is displayed.

```
Copyright (C) Bridgetek Pte Ltd

------------------------------------------------------------------

Welcome to the UART to BOMs example...


Find Flash Disk devices connected to the USB

Host Port and allow control via UART0.

------------------------------------------------------------------


Please plug in a USB Device
```

Figure 6.6 –Welcome Message

Once the flash stick is inserted, the following message in Figure 6.7 is displayed.

```
USB Device Detected

USB Device Enumerated

BOMS device found at level 1



>

Mount File System OK
```

Figure 6.7 –Device Enumerated

6.5 UART Commands

There are a lot of file, directory and other commands available and these are detailed in this section. The commands are case sensitive. If '?' is sent to the UART interface, the commands are listed as shown in Figure 6.8.

Note: The debug build is shown in this section which provides more information.

```
?

HELP MENU.

General Format is:

<command><space><rtn>

<command><space><filename/directory name><rtn>

<command><space><filename/directory name><rtn><data><rtn>



Directory Commands:

dir <directory><ret>                -List directory contents ('.' is current)

mkd <directory><ret>                -Make directory

cd <directory><ret>                 -Change directory

get <ret>                           -Get current directory (N/A on exFAT)



File Commands:

crf <filename><ret>                 -Create a new file

wtf <filename><ret><data><ret>      -Write data to file. Max 1024 chars.

rdf <filename><ret>                 -Read file contents

siz <filename><ret>                 -Get file size in bytes
```

```
File/Directory Commands:

del <file/dir><ret>                  -Delete file or EMPTY directory

rnm <file/dir><rtn><file/dir><rtn>   -Rename/move file or directory

tim <file/dir><ret>

<YYYYMMDDHHMMSS><rtn>                -Set timestamp of file or directory


Disk Commands:

unm <ret>                            -Unmount drive

gel <ret>                            -Get drive label

sel <name><ret>                      -Set drive label

fre <ret>                            -Get free clusters on the volume

rtc <YYYYMMDDHHMMSS><ret>            -Set the FT90x RTC for timestamp info


Other Commands:

? <ret>                              -Help menu

dfu <value><ret>                     -Start the DFU programming interface in ms

ver <ret>                            -Get the version of this firmware.

PRESS ESCAPE KEY to abandon any command or filename/directory name entry
```

Figure 6.8 –Command List

6.5.1 Directory Commands

6.5.2 dir Command

The dir command is used to list the contents of the current or specified directory. Some examples are shown below.

To list the current directory, use ‘dir ’, as shown in Figure 6.9.

```
>

dipath =

DD/MM/YYYY HH:MM               Size Filename

05/01/2018 11:40    <DIR>          0 BRT

05/01/2018 00:07                   0 BRT12345.TXT

           1 File(s)               0 bytes
```

Figure 6.9 –dir current

To list a specified directory use 'dir <path/dir name >' as shown in Figure 6.10.

```
>

dir BRT/BRT2



path = BRT/BRT2

DD/MM/YYYY HH:MM               Size Filename

05/01/2018 00:02                  0 BRT3.TXT

           1 File(s)              0 bytes
```

Figure 6.10 –dir specified

Error codes are returned for example if a particular directory does not exist as shown in Figure 6.11.

```
>

dir TEST

ERROR DIR 5

DIR Error

Error Code 5: FR_NO_PATH
```

Figure 6.11 –dir error

6.5.2.1 mkd Command

The mkd command is used to create a new directory. Some examples are shown below.

To create a new directory in the current directory, use 'mkd <dir name> ', as shown in Figure 6.12.

```
>

mkd BRT2

Directory created OK
```

Figure 6.12 –mkd current

To create a new directory in a specified directory, use 'mkd <path/dir name> ', as shown in Figure 6.13.

```
>

mkd BRT2/BRT3

Directory created OK
```

Figure 6.13 –mkd specified

Error codes are returned for example if a particular directory path does not exist as shown in Figure 6.14.

```
>

mkd BRT4/BRT5

ERROR MKD 5

Error creating directory

Error Code 5: FR_NO_PATH
```

Figure 6.14 –mkd error

6.5.2.2 cd Command

The cd command is used to change the current directory. Some examples are shown below.

To go to the top level home directory, use 'cd /', as shown in Figure 6.15.

```
>

cd /

Directory changed OK
```

Figure 6.15 –cd home

To change the directory, use 'cd <dir name>', as shown in Figure 6.16.

```
>

cd BRT

Directory changed OK
```

Figure 6.16 –cd change

To go back one directory, use 'cd ..', as shown in Figure 6.17.

Note: This does not work with ExFAT since ExFAT . and .. directories do not exist.

```
>

cd ..

Directory changed OK
```

Figure 6.17 –cd back

To change the directory with a path, use 'cd <path/dir name>', as shown in Figure 6.18.

```
>
cd BRT/BRT2

Directory changed OK
```

Figure 6.18 –cd specified

Error codes are returned for example if a particular directory path does not exist as shown in Figure 6.19.

```
>

cd BRT/BRT3

ERROR CD 5

Error changing directory

Error Code 5: FR_NO_PATH
```

Figure 6.19 –cd error

6.5.2.3 get Command

The get command retrieves the current directory.

Note: This does not work on ExFAT due to a known issue with the FatFs version used in this build but could be fixed in future releases. Some examples are shown below.

To get the current directory, use 'get ', as shown in Figure 6.20.

```
>

get

Retrieving the current directory OK

cwd: /BRT/BRT2
```

Figure 6.20 –get current directory

Note: The home directory will be returned as '/'.

An Error will be returned if this is executed on ExFAT filesystem as shown in Figure 6.21.

```
>

get

ERROR GET 1

Get current directory currently not supported on ExFAT
```

Figure 6.21 –get error

6.5.3 File Commands

6.5.3.1 crf Command

The crf command is used to create a new file in the current or specified directory. Some examples are shown below.

To create a new file in the current directory, use 'cfr <filename>', as shown in Figure 6.22.

```
>

crf BRT1.TXT

Creating new file

Closing
```

Figure 6.22 –crf current directory

To create a new file in a specified directory, use 'cfr <path/filename>' as shown in Figure 6.23.

```
>

crf BRT1/BRT1.TXT

Creating new file

Closing
```

Figure 6.23 –crf specified directory

Error codes are returned for example if a particular directory path does not exist as shown in Figure 6.24.

```
>

crf BRT2/BRT.TXT

Creating new file

ERROR CRF 2 5

Problem creating file

Error Code 5: FR_NO_PATH
```

Figure 6.24 –crf error

6.5.3.2 wtf Command

The wtf command is used to write a new line of data to a specified file. The file is not written to until all the specified data is received as it is stored in a variable. The maximum number of characters allowed with this method is 1024 characters. If this is reached the write command stops. The user terminates the write by sending the <return> character. Some examples are shown below.

Note: A new line is added after each successful write command. This can be easily removed in software if desired.

To write some data to a file, use 'wtf <filename><return><data><return>', as shown in Figure 6.25.

Note: Trigger characters like <return> and ESC cannot be part of the data stream.

```
>

wtf BRT1.TXT

Opening for writing

This is a test

Wrote 14 bytes

Closing
```

Figure 6.25 –wtf file

If the maximum number of characters is entered, the command returns and the data is written to the file, as shown in Figure 6.26.

```
>

wtf BRT1.TXT

Opening for writing

bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbb

Wrote 1024 bytes

Closing
```

Figure 6.26 –wtf maximum characters

Error codes are returned for example if a particular file does not exist as shown in Figure 6.27.

```
>

wtf BRT2.TXT

ERROR WTF 1 4

File does not exist.

Error Code 4: FR_NO_FILE
```

Figure 6.27 –wtf error

6.5.3.3 rdf Command

The rdf command is used to read file data contents from a specified file. Some examples are shown below with data entered in the previous two write command examples.

To read data from a file, use 'rdf <filename><return>', as shown in Figure 6.28.

Note: The write data from the previous section is separated by a new line, which can be easily changed in software if desired.

```
>

rdf BRT1.TXT

Opening for reading

This is a test

bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbb

This is a test

Closing
```

Figure 6.28 –rdf file

Error codes are returned for example if a particular file does not exist as shown in Figure 6.29.

```
>

rdf BRT3.TXT

Opening for reading

ERROR RDF 1 4

Error opening for reading

Error Code 4: FR_NO_FILE
```

Figure 6.29 –rdf error

6.5.3.4 siz Command

The siz command is used to get a specified file size in bytes. Some examples are shown below.

To read data from a file, use 'siz <filename><ret>', as shown inFigure 6.30.

Note: The write data from the previous section is separated by a new line, which can be easily changed in software if desired.

```
>

siz BRT1.TXT

Opening file

File Size: 1058 bytes.

Closing
```

Figure 6.30 –siz file

Error codes are returned for example if a particular file does not exist as shown in Figure 6.31.

```
>

siz BRT3.TXT

ERROR SIZ 1 4

File does not exist.

Error Code 4: FR_NO_FILE
```

Figure 6.31 –siz error

6.5.4 File/Directory Commands

6.5.4.1 del Command

The del command is used to delete a file or empty directory.

Note: If a directory contains files, this command will be unable to delete the directory until all files are deleted. If the file or directory does not exist an error is returned. Some examples are shown below.

To delete a file in the current directory, use 'del <filename/dir name><return>', as shown in Figure 6.32.

```
>

del BRT2.TXT

File or directory exists.

Delete successful.
```

Figure 6.32 –del file

To delete a directory from the current directory, use 'del <filename/dir name><return>', as shown in Figure 6.33.

```
>

del BRT1

File or directory exists.

Delete successful.
```

Figure 6.33 –del directory

Error codes are returned for example if a particular file does not exist as shown in Figure 6.34.

```
>

del BRT3.TXT

ERROR DEL 1 4

Error file/directory does no exist

Error Code 4: FR_NO_FILE
```

Figure 6.34 –del file error

Error codes are returned for example if a directory contains files as shown in Figure 6.35.

```
> del BRT1

File or directory exists.

ERROR DEL 2 7

Delete Unsuccessful.

Error Code 7: FR_DENIED
```

Figure 6.35 –del directory error

6.5.4.2 rnm Command

The rnm command is used rename and/or move a file or directory. Some examples are shown below.

To rename a file in the current directory, use 'rnm <filename><return><filename ><return>', as shown in Figure 6.36.

```
>

rnm BRT1.TXT

BRT2.TXT

Rename OK.
```

Figure 6.36 –rnm file

To rename and move a file in the current directory, use 'rnm <filename><return><path/filename ><return>', as shown in Figure 6.37.

```
>

rnm BRT2.TXT

BRT1/BRT3.TXT

Rename OK.
```

Figure 6.37 –rnm and move file

Error codes are returned for example if a particular file does not exist as shown in Figure 6.38.

```
>

rnm BRT1.TXT

ERROR RNM 1 4

File does not exist.

Error Code 4: FR_NO_FILE
```

Figure 6.38 –rnm file error

Error codes are returned for example if a particular directory path does not exist when attempting to move a file as shown in Figure 6.39.

```
>

rnm BRT1.TXT

BRT3/BRT1.TXT

ERROR RNM 2 5

Error renaming

Error Code 5: FR_NO_PATH
```

Figure 6.39 –rnm move error

6.5.4.3 tim Command

The tim command is used to set the timestamp for a specified file or directory.

Note: This is not necessary if the rtc command has been used previously. There is no error checking on that values entered for YYYYMMDDHHMMSS. Some examples are shown below.

To set the timestamp of a file or directory, use 'tim <file/dir><ret><YYYYMMDDHHMMSS><rtn>', as shown in Figure 6.40.

```
>

tim BRT1.TXT

20180123134000

Set filename/directory time OK.
```

Figure 6.40 –tim file

Error codes are returned for example if a particular file does not exist as shown in Figure 6.41.

```
>

tim BRT2.TXT



ERROR TIM 1 4

File does not exist.

Error Code 4: FR_NO_FILE
```

Figure 6.41 –tim error 1

Error codes are returned for example if the wrong number of characters is entered for <YYYYMMDDHHMMSS> as shown in Figure 6.42.

```
>

tim BRT1.TXT

2018012314230099

ERROR TIM 2

Wrong number of numbers.
```

Figure 6.42 –tim error 2

6.5.5 Disk Commands

6.5.5.1 unm Command

The unm command is used to unmount the flash disk for safe removal. The LEDs will also signal various states. See 6.3 Status LEDs for more information. Some examples are shown below.

To unmount the flash disk, use 'unm <return>', as shown in Figure 6.43. The orange status LED will be on until the device is removed.

```
>

unm

Unmount



Please remove the USB Device
```

Figure 6.43 –unm command

When the device is removed the status LEDs will toggle and the message will be shown in Figure 6.44.

```
Please plug in a USB Device
```

Figure 6.44 –unm command disconnected

6.5.5.2 gel Command

The gel command is used to get the drive label information. Some examples are shown below.

To get the drive label, use 'gel <return>', as shown in Figure 6.45.

```
>

gel

Drive label: BRTDSK1
```

Figure 6.45 –gel command

6.5.5.3 sel Command

The sel command is used to set the drive label information. Some examples are shown below.

To set the drive label, use 'sel <name><ret>', as shown in Figure 6.46.

```
>

sel BRTDSK2

Set Drive Label OK.
```

Figure 6.46 –sel command

6.5.5.4 fre Command

The fre command is used to get the total number of sectors and the number of free clusters on the disk. Some examples are shown below.

To get the total and free sectors on the disk, use 'fre <return>', as shown in figure. This disk connected is an 8GB flash stick.

```
>

fre

   7846912 KB total sectors.

   7776340 KB free sectors.
```

Figure 6.47 –fre command

6.5.5.5 rtc Command

The rtc command is used to set the time on the FT90x RTC which is powered by an external battery. This information is used by FatFs to provide file and directory timestamp information. For example, when a file is created, moved or written to, the timestamp data is appended to the file. This only has to be set once. There is no error checking on that values entered for YYYYMMDDHHMMSS. Some examples are shown below.

Note: The minimum YYYY for FT90x RevB is 1980 and for FT90x RevC is 2000 due to differences in the RTC.

Note: FT90x RevB and FT90x RevC RTC are significantly different, so the software implementations for both are also different. The software decides which IC is being used by the following code:

```
//Get Silicon ID
HIPID_value = SYS->HIPID
```

This is then used throughout the software to decide on which software to run:

```
if (HIPID_value == 0x9000001) //RevB
{
}
else if (HIPID_value == 0x9000002)//RevC
{
}
```

Note: FT90x RevC RTC calibration needs to be done for the long-term time keeping. See RTC auto calibration using a 32,768 Hz reference clock in the FT90x RevC User Manual.

To set the RTC, use 'rtc <YYYYMMDDHHMMSS><return>', as shown in Figure 6.48.

```
>

rtc 20180123142300

RTC Set OK
```

Figure 6.48 –rtc command

If a file is created and listed, the timestamp data is shown for the file. Further file updates such as a file write command will incur an incremented timestamp from the RTC, as shown in Figure 6.49.

```
>

crf BRT3.TXT

Creating new file

Closing

>

dir


path =

DD/MM/YYYY HH:MM               Size Filename

23/01/2018 14:23                  0 BRT3.TXT

          1 File(s)               0 bytes
```

Figure 6.49 –rtc example

Error codes are returned for example if the wrong number of characters is entered for <YYYYMMDDHHMMSS> as shown in Figure 6.50.

```
>

rtc 2018012314230099

ERROR RTC 1

Wrong number of numbers.
```

Figure 6.50 –rtc error

6.5.6 Other Commands

6.5.6.1 ? (help) Command

The ? (help) command is used to list all available commands as listed in 6.5 UART Commands.

6.5.6.2 dfu Command

The dfu command is used to enable the USB DFU interface to allow for programming the IC via the USB device port. The time that the USB DFU interface is available for is specified with the command. Some examples are shown below.

To start the USB DFU interface for 10 seconds, use 'dfu 10000', as shown in Figure 6.51.

```
>

dfu 10000

DFU Command Received

Interface available for 10000 ms
```

Figure 6.51 –dfu command

The USB DFU interface can be found in device manager as shown in Figure 6.52. The WinUsb drivers should be automatically loaded via Windows Update.

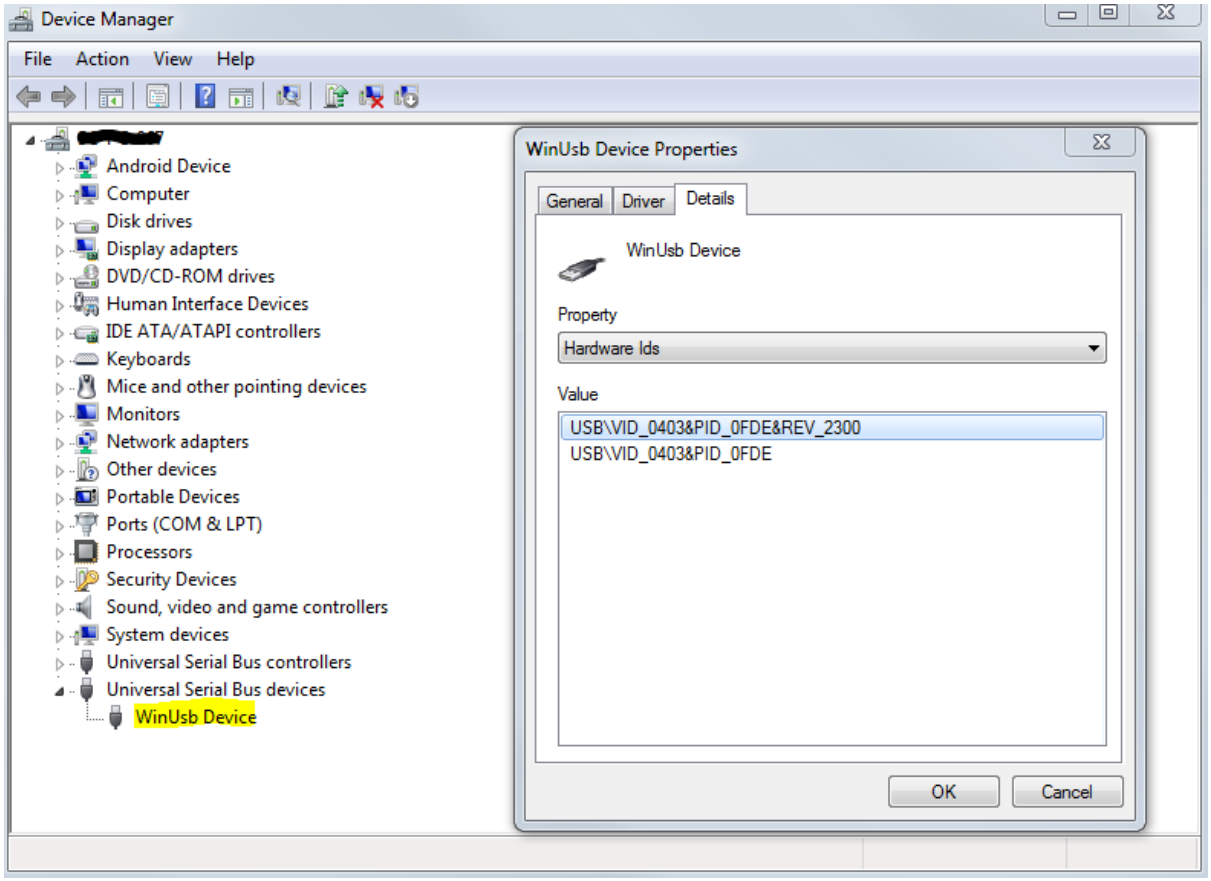

Figure 6.52 –dfu device manager

The FT9xx Programming Utility provided with the FT9xx Toolchain can be used to program the IC via the USB DFU interface, as shown in Figure 6.53 and Figure 6.54.

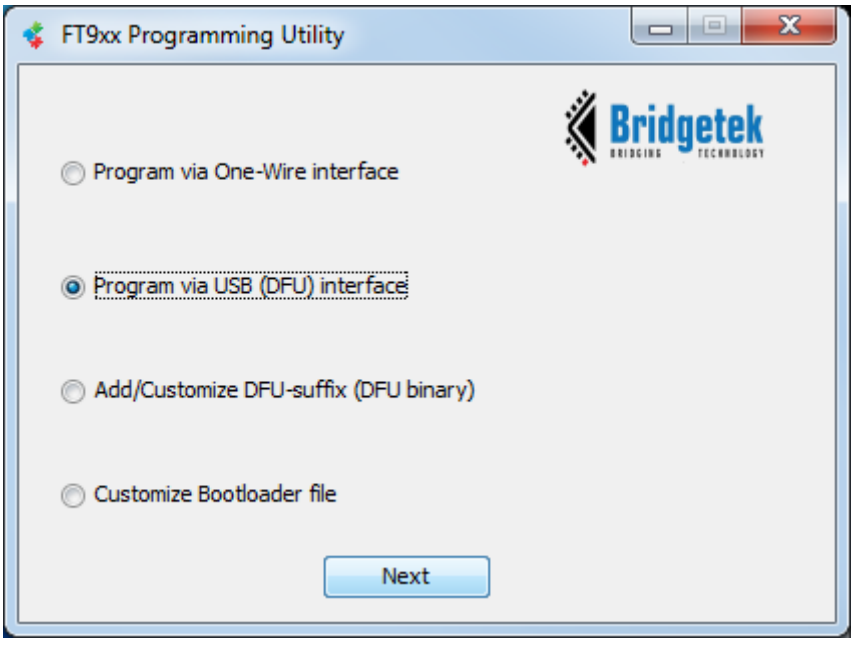

Figure 6.53 –dfu programming utility 1

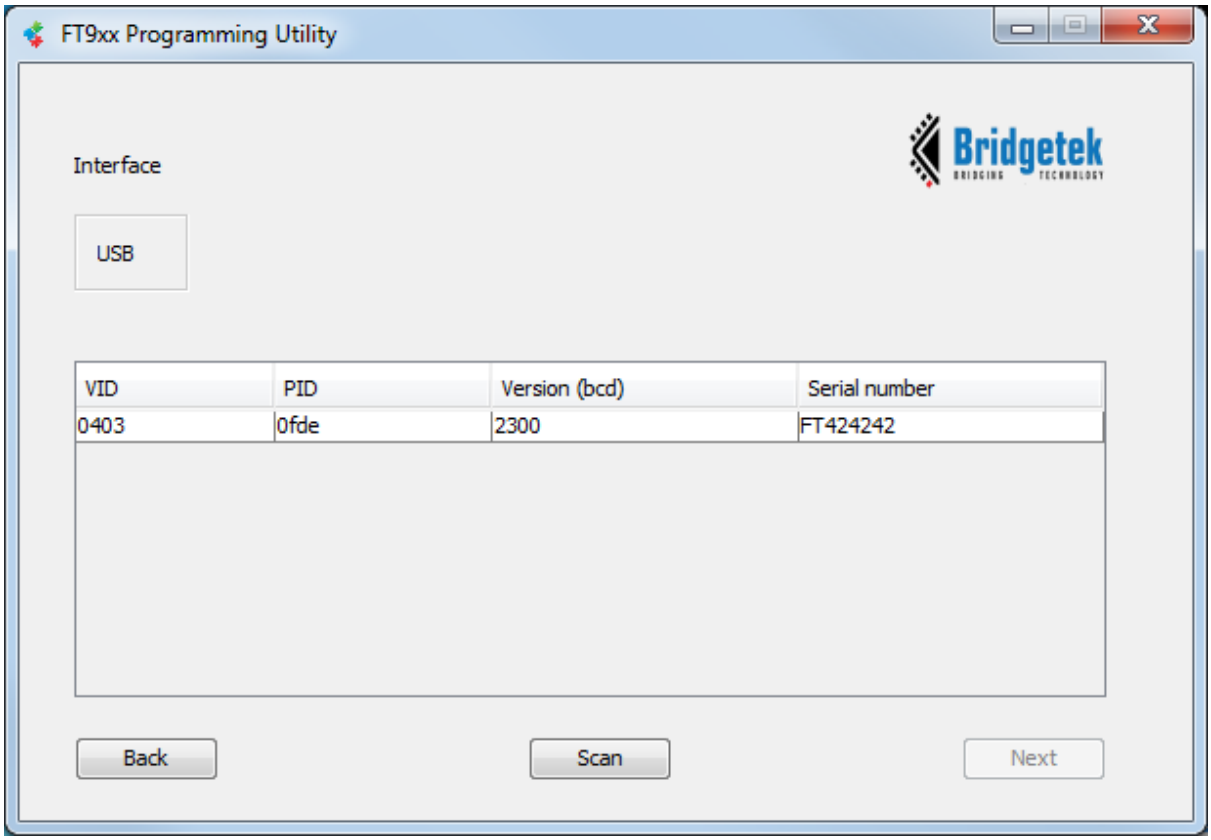

Figure 6.54 –dfu programming utility 2

6.5.6.3 ver Command

The ver command is used to get the version of this firmware. This can be easily modified in the code. Some examples are shown below.

To get the version of the firmware, use 'ver <return> ', as shown in Figure 6.55.

```
>

ver

Version 1.0
```

Figure 6.55 –ver command

6.5.6.4 Escape Key

The ESCAPE KEY can be entered to abandon any command or filename/directory name entry.

The file data entry phase cannot be abandoned. Some examples are shown below.

To abandon the current command if for example the wrong character was typed, press the escape key as shown in figure 6.59.

```
>

mkb



ESC

ESCAPE pressed. Resetting.
```

Figure 6.56 –escape example 1

To abandon the current filename entry if for example the wrong filename was typed, press the escape key as shown in.

```
>

rdf BRT323



ESC

ESCAPE pressed. Resetting.
```

Figure 6.57 –escape example 2

6.6 Error Checking

Some error checking is provided in the code and detailed in this section. The user can alter the code to provide further error checking if required.

6.6.1 Command

If an invalid command is entered, there is an error returned as shown in Figure 6.58.

```
>

rfd

Invalid Command.
```

Figure 6.58 –command error

6.6.2 FatFs

A complete list of all FatFs errors are shown in Figure 6.59.

```
Error Code 1: FR_DISK_ERR

Error Code 2: FR_INT_ERR

Error Code 3: FR_NOT_READY

Error Code 4: FR_NO_FILE

Error Code 5: FR_NO_PATH

Error Code 6: FR_INVALID_NAME
```

```
Error Code 7: FR_DENIED

Error Code 8: FR_EXIST

Error Code 9: FR_INVALID_OBJECT

Error Code 10: FR_WRITE_PROTECTED

Error Code 11: FR_INVALID_DRIVE

Error Code 12: FR_NOT_ENABLED

Error Code 13: FR_NO_FILESYSTEM

Error Code 14: FR_MKFS_ABORTED

Error Code 15: FR_TIMEOUT

Error Code 16: FR_LOCKED

Error Code 17: FR_NOT_ENOUGH_CORE

Error Code 18: FR_TOO_MANY_OPEN_FILES

Error Code 19: FR_INVALID_PARAMETER
```

Figure 6.59 –FatFs errors

For example, if an invalid filename or directory name is entered with one of the commands, FatFs returns an error as shown in Figure 6.60.

```
>

rdf BRT2.TXT

Opening for reading

ERROR RDF 1 4

Error opening for reading

Error Code 4: FR_NO_FILE
```

Figure 6.60 –filename error

6.6.3 rtc and tim Commands

Error codes are returned for example if the wrong number of characters is entered for <YYYYMMDDHHMMSS> as shown in Figure 6.50.

```
>

rtc 2018012314230099

ERROR RTC 1

Wrong number of numbers.
```

Figure 6.61 –rtc error

7 Importing into the FT9xx Toolchain

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

http://brtchip.com/ft90x/#FT90x UART to USB BOMs Memory 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 **Error! Reference source not found.**.

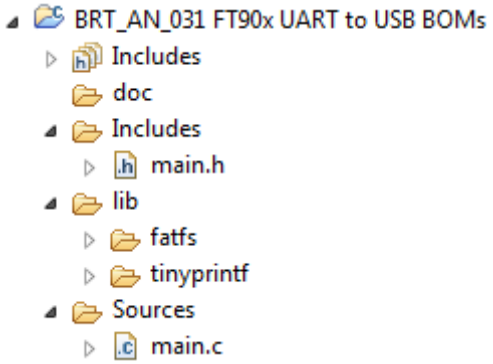

Figure 7.1 –Eclipse Project Structure

7.1 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 FT9xx series and is based on plug-ins for the free popular Eclipse IDE using the GCC compiler.

For example, the UART baud rate and other settings can be easily changed:

```
uart_open(UART0,                    /* Device */
              1,                        /* Prescaler = 1 */
              UART_DIVIDER_19200_BAUD,  /* Divider = 1302 */
              uart_data_bits_8,         /* No. Data Bits */
              uart_parity_none,         /* Parity */
              uart_stop_bits_1);        /* No. Stop Bits */
```

The RTC can also be disabled if it's not required:

```
/* RTC resources */
#define RTC_ENABLE
//#undef RTC_ENABLE
```

There are other configuration options available for FatFs. See the FatFs resources for more information.

8 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 1330
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 Product Page

FT900/FT901/FT902/FT903 Datasheet

FT905/FT906/FT907/FT908 Datasheet

FT9xx Toolchain

FT9xx Development Modules

MM900EVxA datasheet

UMFTPD2A datasheet

Fatfs http://elm-chan.org/fsw/ff/00index_e.html

USB TTL Serial or TTL 234 Serial cables.

App Note Software http://brtchip.com/ft90x/#FT90x UART to USB BOMs Memory Bridge

Acronyms and Abbreviations

Terms	Description
API	Application Programming Interface
BOMs	Bulk Only Mass Storage
DFU	Device Firmware Upgrade
ExFAT	Extended File Allocation Table
FAT32	File Allocation Table 32
GB	Giga Byte
GCC	GNU Compiler Collection
GPIO	General Purpose Input/Output
IC	Integrated Circuit
IDE	Integrated Development Environment
LEDs	Light Emitting Diodes
LFN	Long File Name
MCU	Microcontroller Unit

PC	Personal Computer
RTC	Real Time Clock
RTOS	Real Time Operating System
SPI	Serial Peripheral Interface
TTL	Transistor-Transistor Logic
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus

Appendix B – List of Tables & Figures

List of Tables

List of Figures

Appendix C– Revision History

Document Title: FT90x UART to USB BOMs Memory Bridge
Document Reference No.: BRT_000223
Clearance No.: BRT#139
Product Page: http://brtchip.com/i-ft9/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2019-06-14