Technical Note

BRT_TN_002

Modifying FT9xx API Functions

Version 1.0

Issue Date: 2017-07-13

This technical note demonstrates how to modify API functions provided with the FT9xx toolchain.

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

The FT9xx Peripheral Driver Library is a collection of 'C' language based functions that are intended to ease the development of applications running on the FT9xx Microcontroller. See Figure 1.1. Details of the API functions can be found in AN_365 FT9xx API Programmers Manual.

Sometimes the functionality of the API driver functions provided might not meet the needs of the particular operation for a custom design. However, the user has a couple of options:

1. Write your own functions to access the IC at register level. See AN_324 FT900 User Manual for more information.
2. Modify the API driver functions to change the functionality.
3. Reference the API driver source code to create your own functions.

This technical note demonstrates the 2nd option, modifying the API driver functions provided with the FT9xx toolchain.

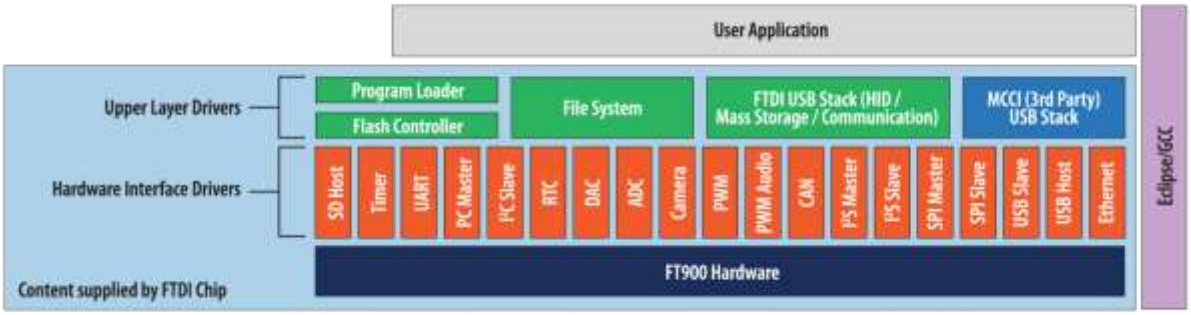

Figure 1.1 –FT9xx Software Stack

2 Modification Process

To change the way any of the source code in the FT9xx API driver library works, copying the source code of the relevant module into the Eclipse project ignores the version in the library and uses the locally copied version, allowing users to change the code.

An example could be with the I^2C master library. At time of writing, the I^2C read and write functions only allow for 8-bit register addressing. The API driver functions can be modified to allow for 16-bit register addressing. This example scenario is used in the next sections to help demonstrate the process.

2.1 Create Project

Users can create new projects in Eclipse via File → New Project → C/C++ Project as shown in Figure 2.1.

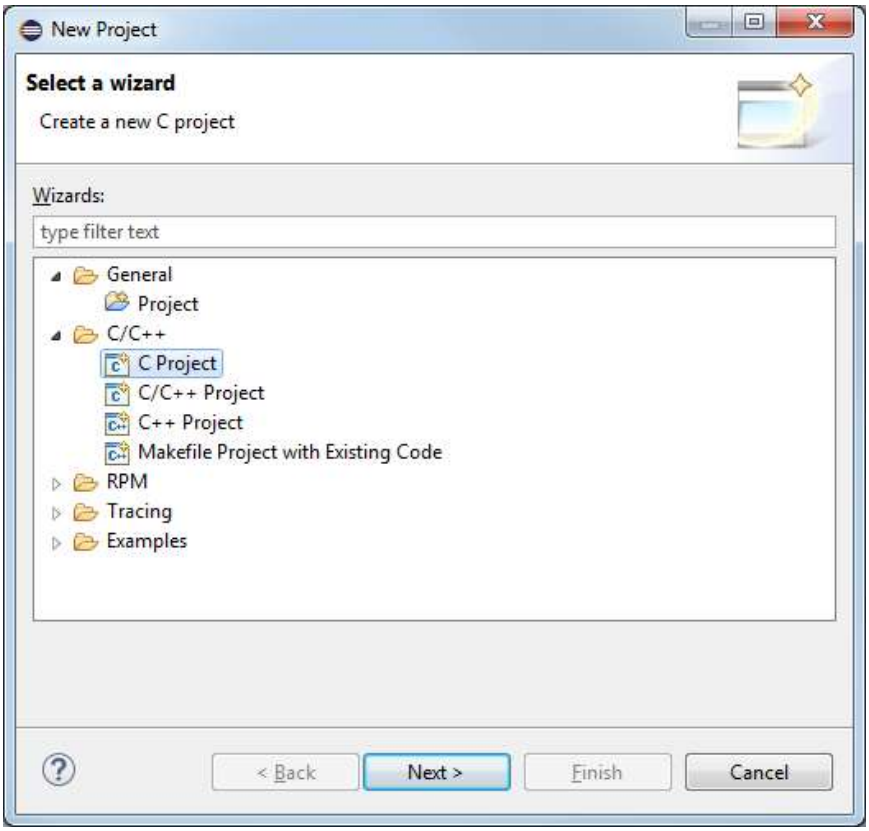

Figure 2.1 –New C Project

Enter the project name, select Executable → Empty Project, and select Bridgetek FT9xx GCC under Toolchains as shown in Figure 2.2.

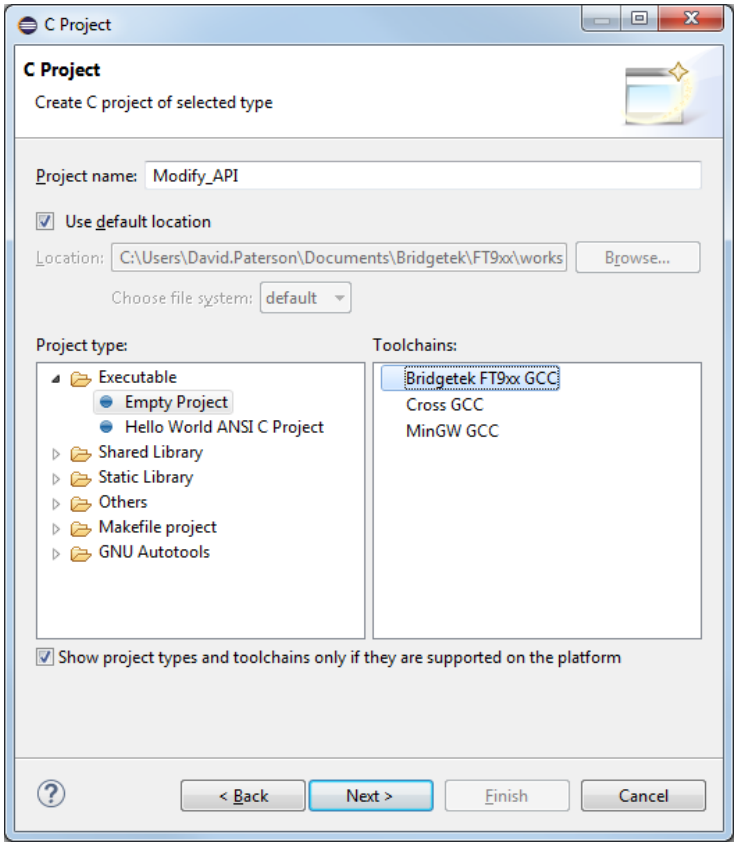

Figure 2.2 –New Project Type

Click Next and select the MCU types that you require (FT900 / FT930) as shown in Figure 2.3.

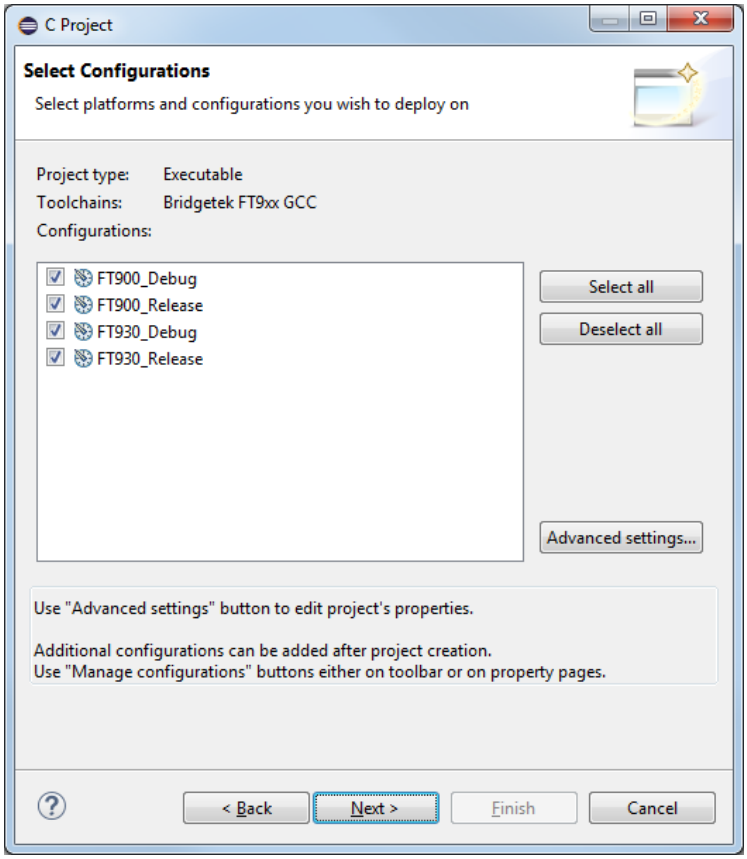

Figure 2.3 –New Project Configurations

Click Next then Finish.

The project will appear in Project Explorer as shown in Figure 2.4

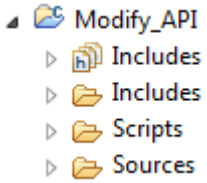

Figure 2.4 –New Project Structure

Some code has been written which utilizes the *i2cm_read* API function as shown in Figure 2.5.

main.c

```
72      uart_open(UART0,                    /* Device */
73                1,                        /* Prescaler = 1 */
74                UART_DIVIDER_19200_BAUD,  /* Divider = 1302 */
75                uart_data_bits_8,         /* No. Data Bits */
76                uart_parity_none,         /* Parity */
77                uart_stop_bits_1);        /* No. Stop Bits */
78
79      /* Print out a welcome message... */
80      uart_puts(UART0,
81          "-------------------------------------------------------------------- \r\n"
82          "Modify API Project \r\n"
83          "-------------------------------------------------------------------- \r\n"
84          );
85
86      /* Enable the I2C Slave device... */
87          sys_enable(sys_device_i2c_master);
88
89      #if defined(__FT900__)
90      #if I2C_CHANNEL == 0
91          /* Set GPIO44 to I2CM_SCL and GPIO45 to I2CM_SDA... */
92          gpio_function(44, pad_i2c0_scl);
93          gpio_pull(44, pad_pull_none);
94          gpio_function(45, pad_i2c0_sda);
95          gpio_pull(45, pad_pull_none);
96      #elif I2C_CHANNEL == 1
97          /* Set GPIO46 to I2CM_SCL and GPIO47 to I2CM_SDA... */
98          gpio_function(44, pad_i2c0_scl);
99          gpio_pull(46, pad_pull_none);
100         gpio_function(45, pad_i2c0_sda);
101         gpio_pull(47, pad_pull_none);
102
103         /* Set the I2C Master pins to channel 1 */
104         sys_i2c_swop(1);
105     #else
106     #   error Please set I2C_CHANNEL to either 0 or 1
107     #endif /* I2C_CHANNEL */
108
109     #elif defined(__FT930__)
110
111         /* Set GPIO12 to I2CM_SCL and GPIO13 to I2CM_SDA... */
112         gpio_function(12, pad_i2cm_scl);
113         gpio_pull(12, pad_pull_none);
114         gpio_function(13, pad_i2cm_sda);
115         gpio_pull(13, pad_pull_none);
116
117     #endif
118
119         /* Set up I2C */
120         i2cm_init(I2CM_NORMAL_SPEED, 10000);
121
122
123         i2cm_read(EEPROM_ADDR, location, eeprom_buffer, EEPROM_SIZE);
124
125
126     /* Now keep looping */
127     while (1);
128
129     return 0;
130 }
```

Figure 2.5 –New Project Code

2.2 Copy Source

In order to change the API source code, copy the *i2cm.c* file into the project as shown in Figure 2.6.

The API source files can be found here:

C:\Users*Username*\Documents\Bridgetek\FT9xx*version*\Source\src

The file can be copied at windows explorer level then right-click on the project in Eclipse and select Refresh to see the added file. They can also be copied from Windows explorer and pasted into Eclipse where no refresh is required within Eclipse.

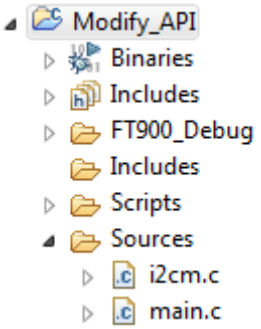

Figure 2.6 –Insert API source

2.3 Edit the Code

The Project code and API code can now be edited.

Figure 2.7 shows some new I^2C read code for 16-bit addressing.

Note that the function structure has changed from:

```
int8_t i2cm_read(const uint8_t addr, const uint8_t cmd, uint8_t *data, uint16_t number_to_read)
```

to

```
int8_t i2cm_read(const uint8_t addr, const uint16_t reg, uint8_t *data)
```

The *ft900_i2cm.h* header file at the following location needs to be edited to reflect the function attribute change:

C:\Program Files (x86)\Bridgetek\FT9xx Toolchain\Toolchain\hardware\include

Note: Administrator privileges are required to edit this file as it resides in the Program Files directory.

Note: If the function structure hasn't changed there is no need to edit the header file.

The project main code should also be updated to reflect this change:

```
#define I2C_ADDR (0x78)
#define EEPROM_SIZE (1024/8)
uint8_t eeprom_buffer[EEPROM_SIZE] = {0};
const uint8_t addr = I2C_ADDR;
```

```
i2cm_read(addr, location, eeprom_buffer);
```

main.c | i2cm.c

```
228 }
229
230 int8_t i2cm_read(const uint8_t addr, const uint16_t reg, uint8_t *data)
231 {
232     int8_t ret = 0;
233     uint8_t regl, regh;
234
235     regl = reg & 0xff;
236     regh = reg >> 8;
237
238     do
239     {
240         /* Write slave address to SA */
241         I2CM->I2CM_SLV_ADDR = addr;
242
243         /* Write register high address to BUF */
244         I2CM->I2CM_DATA = regh;
245         /* Write command to I2CMCR to send Start and command byte. */
246         I2CM->I2CM_CNTL_STATUS = I2C_FLAGS_START | I2C_FLAGS_RUN;
247         if(i2c_wait_for()) {
248             /* An Error Occurred */
249             ret = -1;
250             break;
251         }
252
253         /* Write register low address to BUF */
254         I2CM->I2CM_DATA = regl;
255         I2CM->I2CM_CNTL_STATUS = I2C_FLAGS_RUN | I2C_FLAGS_STOP;
256         if(i2c_wait_for()) {
257             /* An Error Occurred */
258             ret = -2;
259             break;
260         }
261
262         delayms(1);
263
264         /* Write slave address to SA with R operation */
265         I2CM->I2CM_SLV_ADDR = addr | 0x01;
266         /* Receive with a NACK */
267         I2CM->I2CM_CNTL_STATUS = I2C_FLAGS_START | I2C_FLAGS_STOP | I2C_FLAGS_RUN;
268         if(i2c_wait_for()) {
269             /* An Error Occurred */
270             ret = -4;
271             break;
272         }
273
274         /* Read data from device. */
275         *data = I2CM->I2CM_DATA;
276
277     } while (0);
278
279     return ret;
280 }
```

Figure 2.7 –New API Code

2.4 Rebuild

The project can now be rebuilt. This can be done by right-clicking on the project and selecting build, or by clicking on the build icon  .

The status is shown in the console window. See Figure 2.8 for a successful build.

Figure 2.8 –Console Output Success

Any warnings or errors will be shown in the console windows. Figure 2.9 shows an error case when the header file is not updated to reflect a function type change.

Figure 2.9 –Console Output Error

3 Conclusion

The FT9xx API functions can easily be edited to suit your needs, if the default functionality does not suit. Users can also reference this code to create their own functions.

4 Contact Information

Headquarters – 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

FT93x Product Page

AN_324 FT900 User Manual

FT90x Toolchain

Acronyms and Abbreviations

Terms	Description
API	Application Programming Interface
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit

Appendix B – List of Tables & Figures

List of Tables

NA

List of Figures

Appendix C – Revision History

Document Title: BRT_TN_002 Modifying FT9xx API Functions
Document Reference No.: BRT_000144
Clearance No.: BRT#086
Product Page: http://brtchip.com/product/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2017-07-13