Application Note

BRT_AN_001

ESD 3.0 Exported Project Porting Guide

Version 1.0

Issue Date: 2017-02-09

This application note is intended as a guide for porting an **E**VE **S**creen **D**esigner (**ESD**) 3.0 exported project to a non-FT9xx based MCU platform. Users are expected to have knowledge of ESD 3.0 as well as FT81x and STM32L4XX MCU

Use of Bridgetek Pte devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold Bridgetek Pte harmless from any and all damages, claims, suits or expense resulting from such use.

Bridgetek Pte Limited (BRTChip)

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

Table of Contents

1 Introduction 4
1.1 Overview 4
1.2 Scope 4
2 ESD 3.0 Exported Project - Introduction 5
3 Porting principles 7
3.1 Hardware 7
3.2 Software 7
4 Example 8
4.1 Connection 9
4.1.1 Power and Ground Connection 9
4.1.2 MCU and EVE Connection 10
4.2 Software Setup 10
4.2.1 Toolchain and Utility 10
4.2.2 Platform Configuration 11
4.2.3 System Clock Configuration 11
4.2.4 GPIO Initialize 12
4.2.5 SPI Initialize 13
4.2.6 Storage Media Configuration and Access 14
4.2.7 Redefining Data Type 15
4.2.8 Project Files 15
4.3 Linker Script 16
4.4 MCU Hardware Abstraction Layer and Drivers 16
4.5 APIs Re-Implemenation 17
4.6 Change Summary 19
5 Import & Build the Project 20
6 Contact Information 23
Appendix A– References 24
Document References 24
Acronyms and Abbreviations 24

Copyright © Bridgetek Pte Limited

Appendix B – List of Tables & Figures 25
List of Figures 25
List of Tables.....25
Appendix C– Revision History 26

Copyright © Bridgetek Pte Limited

1 Introduction

This application note is intended as a guide for porting an **E**VE **S**creen **D**esigner (**ESD**) 3.0 exported project to a non-FT9xx based MCU platform. Users are expected to have knowledge of ESD 3.0 as well as FT81x and STM32L4XX MCU.

1.1 Overview

This guide covers the following topics:

- ESD 3.0 exported project introduction
- Principles of porting
- Example

1.2 Scope

This document is intended to guide users to port an ESD 3.0 exported project to another MCU platform. It covers hardware changes and software modification as well as some debugging tips. It also provides some basic principles to successfully port a project.

Copyright © Bridgetek Pte Limited

2 ESD 3.0 Exported Project - Introduction

ESD 3.0 enables users to design an EVE based GUI application with minimum effort. Upon completing the design and successfully simulating it on a PC, users can choose to export the currently opened project by selecting **"File → Export → Export As...."**, as shown Figure 1.

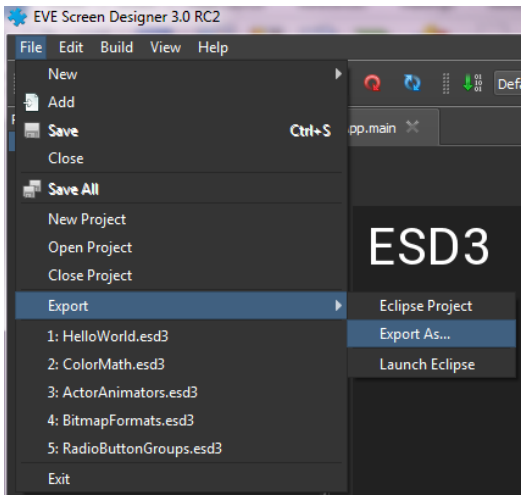
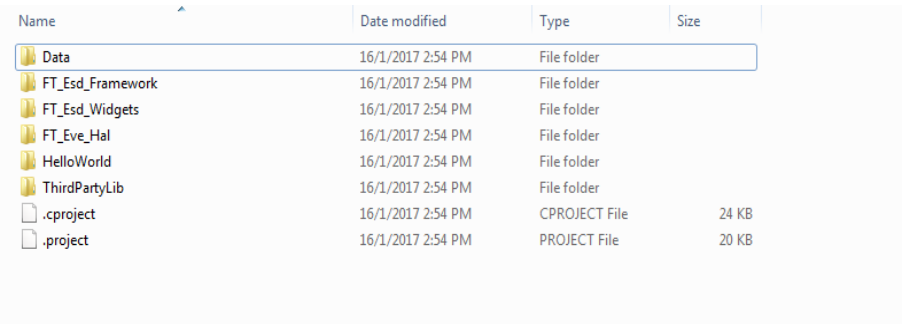


Figure 1 Export HelloWorld Project in ESD 3.0

Users are prompted to select a new folder as the destination folder to export. Once exporting is done, the destination folder shall be as below:



Name	Date modified	Type	Size
Data	16/1/2017 2:54 PM	File folder	
FT_Esd_Framework	16/1/2017 2:54 PM	File folder	
FT_Esd_Widgets	16/1/2017 2:54 PM	File folder	
FT_Eve_Hal	16/1/2017 2:54 PM	File folder	
HelloWorld	16/1/2017 2:54 PM	File folder	
ThirdPartyLib	16/1/2017 2:54 PM	File folder	
.cproject	16/1/2017 2:54 PM	CPROJECT File	24 KB
.project	16/1/2017 2:54 PM	PROJECT File	20 KB

Figure 2: Folder Structure of HelloWorld Project

Folder / file Name	Description	Remarks
Data	Contains the converted bitmap assets	Do not need to change
FT_Esd_Framework	Contains application framework source code	Reusable and common module
FT_Esd_Widgets	Contains the widgets-related source code	Reusable and common module
FT_Eve_Hal	Contains hardware abstraction layer	The major folder to be changed

Copyright © Bridgetek Pte Limited

Folder / file Name	Description	Remarks
HelloWorld	The project file contains the screen logic and users design	The folder name shall be same as project name
ThirdPartyLib	Contains the third part library	Currently only FatFs Library source code inside
.cproject	Eclipse CDT project file	Build configurations, tool chains, individual tools etc.
.project	Eclipse CDT project file	Build specification and build commands

Table 1: Folder Contents

By default, the exported ESD 3.0 project supports FT90X series platform only. Therefore, the default project file works only in "Eclipse for FT90X" IDE which is part of the FT90x Toolchain.

Copyright © Bridgetek Pte Limited

3 Porting principles

3.1 Hardware

An ESD 3.0 exported project usually needs access to the following hardware resources of the MCU:

- SPI interface : Read/Write EVE 2 module
- Clock : Provide delay and timing control functionality
- Storage media: To store bitmap assets, usually it is flash, SD/MMC card, USB disk etc.

Different MCU's have different hardware configurations; therefore, users need to ensure that the hardware components above work well. Users are assumed to be familiar with FT81X series ICs as well as EVE 2 series modules before starting the porting work. Refer to brtchip.com for more details.

3.2 Software

Apart from hardware, users are required to modify or add the following software modules:

- **FT_Eve_Hal:** Initialize the targeting MCU platform and re-implement the transportation layer API for the EVE chip interface.
- **Project files:** Configuration and instructions for building the project. MCU tool chain specific.
- **Linker script:** Instruct the linker software to generate an MCU platform specific binary.

Copyright © Bridgetek Pte Limited

4 Example

This example illustrates how to accomplish the porting activity, concurring to the above stated principles.

The selected target MCU platform is an STM32L476 Discovery board. It shall be connected to the development PC via a USB cable for downloading, debugging and power supply.

Figure 3 STM32L4XX Discovery Board

Selected EVE 2 Module is ME813A-WH50C:

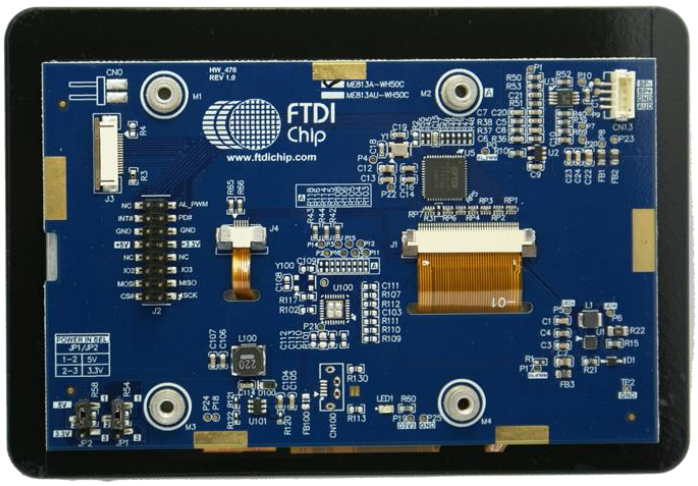


Figure 4 EVE 2 Module (ME813A-WH50C)

Copyright © Bridgetek Pte Limited

Here is the connection between two boards:

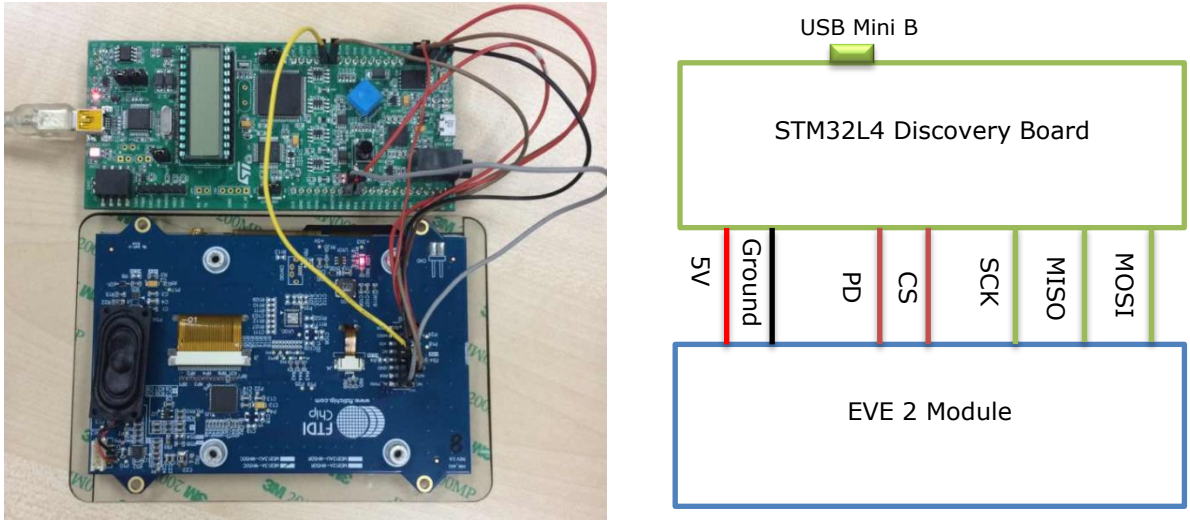


Figure 5 Hardware Setup

The "HelloWorld" example project from the ESD 3.0 samples is used to demonstrate how the porting will be done. The "HelloWorld" example is located at *"$(ESD3.0 Installation Folder)\\Examples\\Basic"* folder. Here is a sample screenshot:

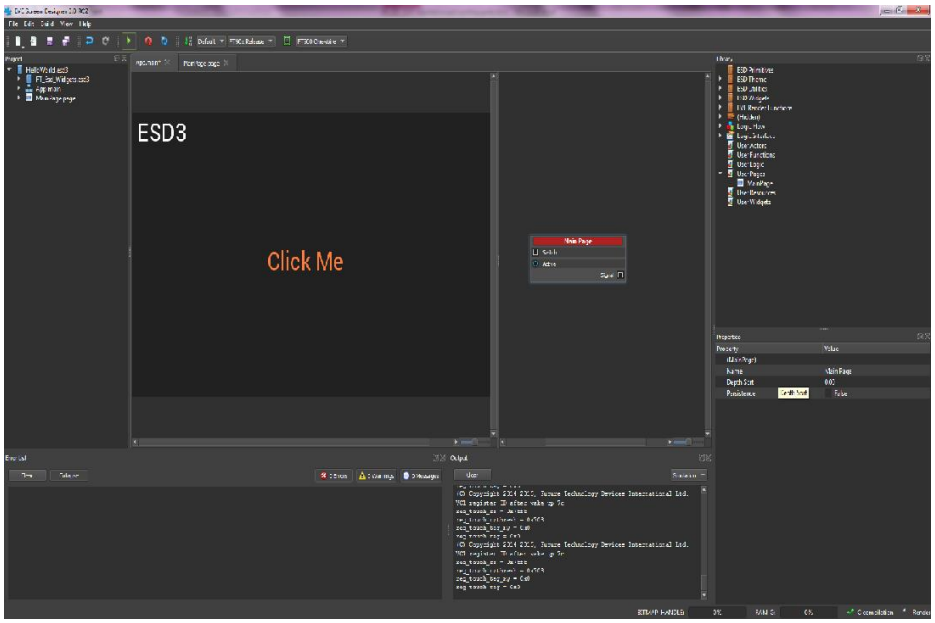


Figure 6 ESD 3.0 HelloWorld Project Screenshot

4.1 Connection

4.1.1 Power and Ground Connection

EVE 2 Module is powered by a 5V supply from the STM32L4 Discovery board and a ground pin is also connected.

Copyright © Bridgetek Pte Limited

4.1.2 MCU and EVE Connection

In this example, STM32L4 MCU's SPI 1 interface is selected and the following connection is set up:

MCU Pin Name	MCU Function	EVE Pin name
PB2	GPIO	#PD
PE8	GPIO	#CS
PE13	SPI1_SCK	SCK
PE14	SPI1_MISO	MISO
PE15	SPI1_MOSI	MOSI

Table 2 MCU and EVE Connection

Care must be taken if connecting the two boards with simple jumper wires. It may be necessary to lower the frequency of the SPI clock by setting the BR bits of the SPIx_CR1 register to ensure a stable signal quality.

4.2 Software Setup

4.2.1 Toolchain and Utility

For this example porting exercise, the Atollic TrueSTUDIO® for ARM® free version is selected as the compiler and linker for the STM32L4 MCU. In this example, version 7.0.1 is used.

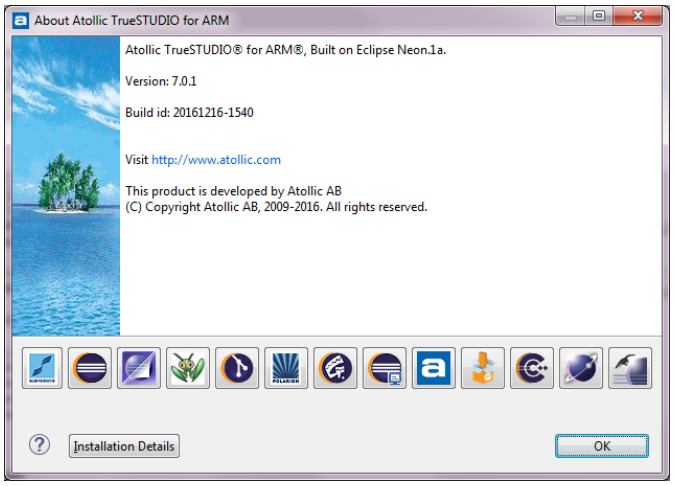


Figure 7 Atollic TrueSTUDIO Tool - About Window

To download and debug the firmware, users need to install the necessary driver on the PC from the STM32 link utility. It can be downloaded from **STSW-LINK004 - STM32 ST-LINK utility.**

Another very helpful tool is called "STM32CubeMX", which is downloadable from here. This tool can help users configure pin functionality easily. In addition, it automatically generates the source code to configure hardware resource.

Copyright © Bridgetek Pte Limited

The following file is the project file of the STMCubeMX tool which is used by the example project specified in this document.

stm32l4_eveconfiguration.ioc

The following screenshot shows how the pin configuration looks like, once the project is opened using the STMCubeMX tool.

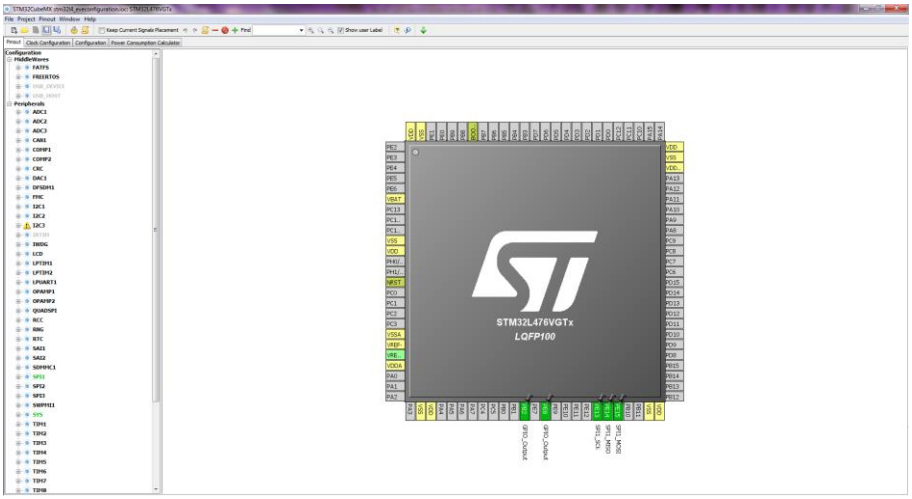


Figure 8 STMCubeMX Snapshot

4.2.2 Platform Configuration

In the "FT_Eve_Hal\FT_Platform.h" file, the default platform in the ESD 3.0 example project is FT90x and it is assigned to macro "**FT900_PLATFORM**". In order to support the STM32L4 platform, it is necessary to disable this macro and introduce a new macro for the STM32L4. The following file can be reused to make the definition of macro "**STM32L4_DISCOVERY**". Users shall use this macro in other source files to differentiate the STM32L4 specific implementation.

A good practice is to search the string "FT900_PLATFORM" within the exported project and check to see if the original function definition is necessary to be modified or re-implemented for the STM32L4 Discovery board.

FT_Platform.h

4.2.3 System Clock Configuration

To control the timing and necessary delays, the system clock of MCU is enabled as below:

```
/** System Clock Configuration
*/
static void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct;
  RCC_ClkInitTypeDef RCC_ClkInitStruct;
```


Copyright © Bridgetek Pte Limited

```
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_MSI;
  RCC_OscInitStruct.MSIState = RCC_MSI_ON;
  RCC_OscInitStruct.MSICalibrationValue = 0;
  RCC_OscInitStruct.MSIClockRange = RCC_MSIRANGE_6;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_MSI;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }

  __HAL_RCC_PWR_CLK_ENABLE();

  if (HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VOLTAGE_SCALE1) != HAL_OK)
  {
    Error_Handler();
  }

  HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

  HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

  /* SysTick_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}
```

This will enable the MCU HAL API “HAL_Delay” and “HAL_GetTick”. It shall be called before GPU.

4.2.4 GPIO Initialize

MCU pin PB2 for EVE #PD pin and PE8 for EVE #CS are configured as outputs.

```
static void MX_GPIO_Init(void)
{

  GPIO_InitTypeDef GPIO_InitStruct;

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOB_CLK_ENABLE();
  __HAL_RCC_GPIOE_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOE, GPIO_PIN_8, GPIO_PIN_SET);

  /*Configure GPIO pin : PB2 for EVE #PD pin*/
  GPIO_InitStruct.Pin = GPIO_PIN_2;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

  /*Configure GPIO pin : PE8 for EVE #CS pin*/
  GPIO_InitStruct.Pin = GPIO_PIN_8;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
}
```


Copyright © Bridgetek Pte Limited

4.2.5 SPI Initialize

In this example, SPI1 is enabled as master mode in duplex mode. The data size per frame is 8 bits and the most significant bit is shifted in or out first on the SPI bus. NSS pulse mode is disabled and the NSS pin is under software control. Both values of CPOL and CPHA of the SPI1 clock are zero. The following sample code shows the configuration:

```
static void MX_SPI_Init(void)
{
        hspi1.Instance = SPI1;
        hspi1.Init.Mode = SPI_MODE_MASTER;
        hspi1.Init.Direction = SPI_DIRECTION_2LINES; //Full duplex
        hspi1.Init.DataSize = SPI_DATASIZE_8BIT;

        hspi1.Init.CLKPolarity = SPI_POLARITY_LOW; //SPI mode 0
        hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;//SPI mode 0

        hspi1.Init.NSS = SPI_NSS_SOFT;
        hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_8;//Make the baudrate lower if
the jumping wire is too long
        hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB; //Most Significant Bit

        hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
        hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_ENABLE;
        hspi1.Init.CRCPolynomial = 7;
        hspi1.Init.CRCLength = SPI_CRC_LENGTH_DATASIZE;
        hspi1.Init.NSSPMode = SPI_NSS_PULSE_DISABLE; // No pulse generated
        if (HAL_SPI_Init(&hspi1) != HAL_OK)
        {
                Error_Handler();
        }
}
```

In the "FT_Gpu_Hal.c" file, functions "SystemClock_Config", "MX_GPIO_Init", "MX_SPI_Init" are called by function "FT_Gpu_Hal_Init" as below:

```
ft_bool_t  Ft_Gpu_Hal_Init(Ft_Gpu_HalInit_t *halinit)
{
#ifdef STM32L4_DISCOVERY
          /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
          HAL_Init();

          /* Configure the system clock */
          SystemClock_Config();
          /* Initialize all configured peripherals */
          MX_GPIO_Init();

          MX_SPI_Init();
#endif

        return TRUE;
}
```


Copyright © Bridgetek Pte Limited

4.2.6 Storage Media Configuration and Access

In this example, the storage media is not enabled to simplify the procedure. The configuration of the storage media, such as USB mass storage or Quad SPI flash shall be done properly inside the function "FT_Gpu_Hal_Init".

To access bitmap assets, users need to re-implement the function "int load_inflate_image(ft_uint32_t address, const char *filename)" from file "Ft_Esd_BitmapInfo.c" ', which is under the "FT_Esd_Framework" folder. This function is to be called when the "filename" specified bitmap file needs to be loaded into the address in the EVE RAM_G. The bitmap file is normally in an EVE supported format after conversion. Users need to read the file from the storage media and load it into EVE. Here is the example for FT90X platform:

```
int load_inflate_image(ft_uint32_t address, const char *filename)
{
        Ft_Gpu_Hal_Context_t *phost = Ft_Esd_Host;
#if defined (FT900_PLATFORM)

        FRESULT fResult;
        FIL InfSrc;

        ft_int32_t blocklen, filesize;
        ft_uint32_t g_random_buffer[512L];

        fResult = f_open(&InfSrc, filename, FA_READ | FA_OPEN_EXISTING);
        if (fResult == FR_OK)
        {
                Ft_Gpu_Hal_WrCmd32(phost, CMD_INFLATE);
                Ft_Gpu_Hal_WrCmd32(phost, address);
                filesize = f_size(&InfSrc);
                while (filesize>0)
                {
                        fResult = f_read(&InfSrc, g_random_buffer, 512, &blocklen); // read a
chunk of src file
                        filesize -= blocklen;
                        Ft_Gpu_Hal_WrCmdBuf(phost, (char *)g_random_buffer, blocklen);

                }
                f_close(&InfSrc);
                return 1;
        }
        else
        {
                printf("Unable to open file: \"%s\"\n", filename);
                return 0;
        }
#endif
}
```


Copyright © Bridgetek Pte Limited

4.2.7 Redefining Data Type

In the "FT_DataType.h" file, the customized data type is defined for different platforms. For ARM platforms, to access the data stored in flash, there is no special compiler keyword required.

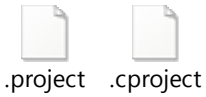
Therefore, the following modification is required:

```
#define TRUE     (1)

#ifdef STM32L4_DISCOVERY
#define ft_prog_uchar8_t  ft_uchar8_t  const
#define ft_prog_char8_t   ft_char8_t   const
#define ft_prog_uint16_t  ft_uint16_t  const

#define FT_PROGMEM
#define ft_pgm_read_byte_near(x)   (*((ft_prog_uchar8_t *)(x)))
#define ft_pgm_read_byte(x)        (*((ft_prog_uchar8_t *)(x)))
#define ft_pgm_read_word(addr)   (*((ft_prog_uint16_t *)(addr)))

#else
#define ft_prog_uchar8_t  ft_uchar8_t __flash__ const
#define ft_prog_char8_t   ft_char8_t __flash__ const
#define ft_prog_uint16_t  ft_uint16_t __flash__ const
```

Figure 9 Disable Flash Access Keyword

4.2.8 Project Files

Since Atollic TrueSTUDIO® for ARM® IDE is used to compile and build the source code; users need to rewrite the .cproject file and .project file. The easiest way is to overwrite the old ones with the following two files and change the configuration accordingly for different projects.



4.2.8.1 Project Name

Open the .cproject file in a text editor and replace all the "HelloWorld" strings with the new project name. Do the same things at .project file.

4.2.8.2 Source and Header File

Open the .project file in text editor and add the necessary files under <linkedResources> element as below:

```
<link>
  <name>HelloWorld/MainPage__Generated.c</name>
  <type>1</type>
   <locationURI>PROJECT_LOC/HelloWorld/MainPage__Generated.c</locationURI>
</link>
<link>
  <name>HelloWorld/MainPage.h</name>
  <type>1</type>
```


Copyright © Bridgetek Pte Limited

```
    <locationURI>PROJECT_LOC/HelloWorld/MainPage.h</locationURI>
  </link>
```

The example above adds one header file and one source file under the project folder.

4.3 Linker Script

To define the memory and flash address space as well as stack and heap size, a linker script is required to be placed under the root of the project folder. Normally, it is MCU specific and does not require any change from project to project. The file below is the file for STM32L476VG MCU on the STM32L4 Discovery board. Users can copy it directly to the exported project folder.

STM32L476VG_FLASH.ld

4.4 MCU Hardware Abstraction Layer and Drivers

The STM32L476XX MCU provides a reusable hardware abstraction layer which ensures that the source code can be reused across other STM32 MCU. All the related files are listed in the picture below.

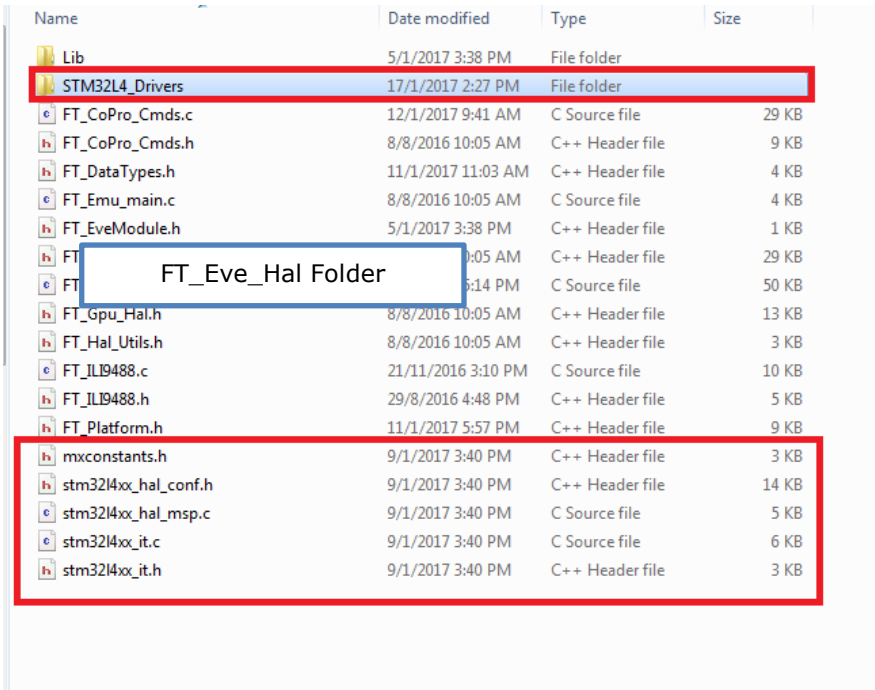


Figure 10 Extra Files for MCU Configuration

<u>Note:</u> Those files whose name starts with "stm" and "mxconstant.h" are generated from STM32CubeMX tool.

Copyright © Bridgetek Pte Limited

Users shall copy the files above into the "FT_Eve_Hal" folder and unzip the STM32L4_Drivers.zip file. These files shall be kept intact because they expose the APIs such as API "*HAL_SPI_Init*" to the user's code.

4.5 APIs Re-Implemenation

To make the MCU boot up and communicate with its peripheral, the following APIs are required to be re-implemented. In this example, the implementation is valid only when the macro "STM32L4_DISCOVERY" is switched on in "FT_Platform.h" to ensure its changes will not affect other platforms.

API name	File	Remarks	Note
Ft_Gpu_Hal_Init	FT_Gpu_Hal.c	Initialize the MCU and its peripheral (SPI, GPIO)	NA
Ft_Gpu_Hal_Open	FT_Gpu_Hal.c	Prepare software structure	NA
Ft_Gpu_Hal_Close	FT_Gpu_Hal.c	Close software structure handle	NA
Ft_Gpu_Hal_DeInit	FT_Gpu_Hal.c	De-initialize system	NA
Ft_Gpu_Hal_StartTransfer	FT_Gpu_Hal.c	Initiate address phase by transmitting 3 bytes address code and assert CS	For SPI reading, there are 3 dummy bytes to be returned and discarded.
Ft_Gpu_Hal_Transfer8	FT_Gpu_Hal.c	Send or receive one byte	NA
Ft_Gpu_Hal_EndTransfer	FT_Gpu_Hal.c	De-assert the CS to end the SPI transferring	NA
Ft_Gpu_HostCommand	FT_Gpu_Hal.c	Send host commands to EVE	NA
Ft_Gpu_HostCommand_Ext3	FT_Gpu_Hal.c	Send 3 bytes host commands to EVE	NA
Ft_Gpu_Hal_WrCmdBuf	FT_Gpu_Hal.c	Send the data in the buffer to EVE	NA

Copyright © Bridgetek Pte Limited

API name	File	Remarks	Note
Ft_Gpu_Hal_WrCmdBuf_nowait	FT_Gpu_Hal.c	Send the data in the buffer to EVE	NA
Ft_Gpu_Hal_Powercycle	FT_Gpu_Hal.c	Toggle PD pin to wake up EVE	The delay after each toggle is mandatory
Ft_Gpu_Hal_WrMemFromFlash	FT_Gpu_Hal.c	Add the macro for STM32L4	NA
Ft_Gpu_Hal_WrMem	FT_Gpu_Hal.c	Send the data in the buffer to EVE	NA
Ft_Gpu_Hal_RdMem	FT_Gpu_Hal.c	Read the data from Eve and fill the buffer	NA
Ft_Gpu_Hal_Sleep	FT_Gpu_Hal.c	Delay the specified amount of time	NA
ft_millis	FT_Gpu_Hal.c	Get the current system tick	NA
Ft_Gpu_Copro_SendCmd	FT_CoPro_Cmds.c	Add the macro for STM32L4	NA
Ft_Gpu_CoCmd_SendStr	FT_CoPro_Cmds.c	Add the macro for STM32L4	NA
Ft_Gpu_CoCmd_StartFunc	FT_CoPro_Cmds.c	Add the macro for STM32L4	NA
Ft_Gpu_CoCmd_EndFunc	FT_CoPro_Cmds.c	Add the macro for STM32L4	NA
load_inflate_image	Ft_Esd_BitmapInfo.c	Load the image file from storage into EVE RAM_G	Keep it as empty for users to define its own implementation
FT800_BootupConfig	Ft_Esd_App.c	Add the delay after wakeup command	Bug fix
main	Ft_Esd_App.c	Enable main function for STM32L4	

Table 3 APIs to be changed

The changes above may not be optimal for the SPI transfer performance because the primary target is to keep the code structure working on different platforms. Users are encouraged to read the code thoroughly and optimize the transfer performance by sending more data for each SPI transaction.

Copyright © Bridgetek Pte Limited

4.6 Change Summary

The figure below shows the list of files that are added or modified during porting. Users are encouraged to review porting activity.

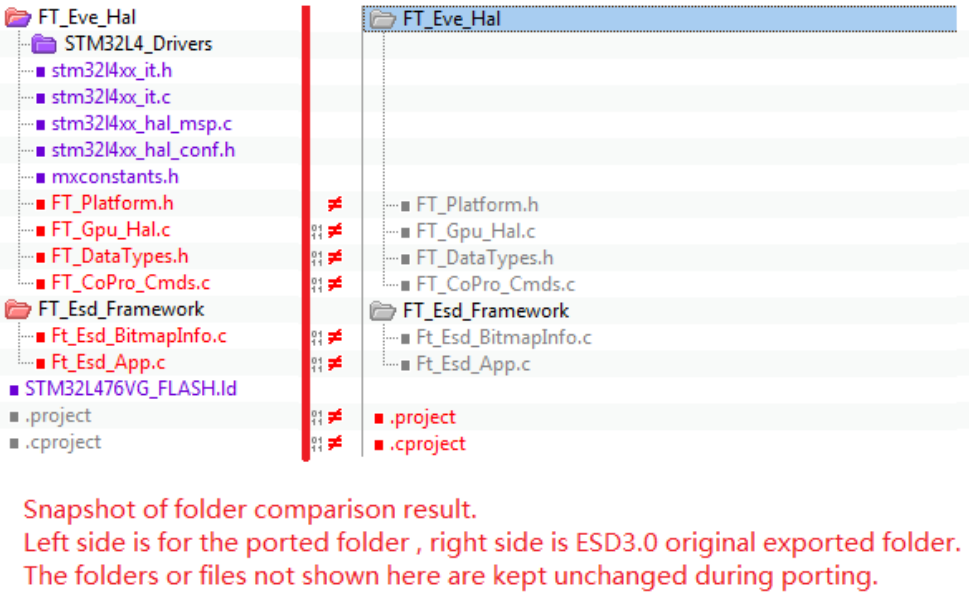


Figure 11 Comparison Result of Ported VS Original Project

The complete ported example is enclosed as below:

HelloWorld_Export.zip

Copyright © Bridgetek Pte Limited

5 Import & Build the Project

Upon completing all the necessary changes for the STM32L4 platform, it is time to import and build the project.

Atollic TrueSTUDIO IDE shall be launched to import the project. Refer to the screenshots below:

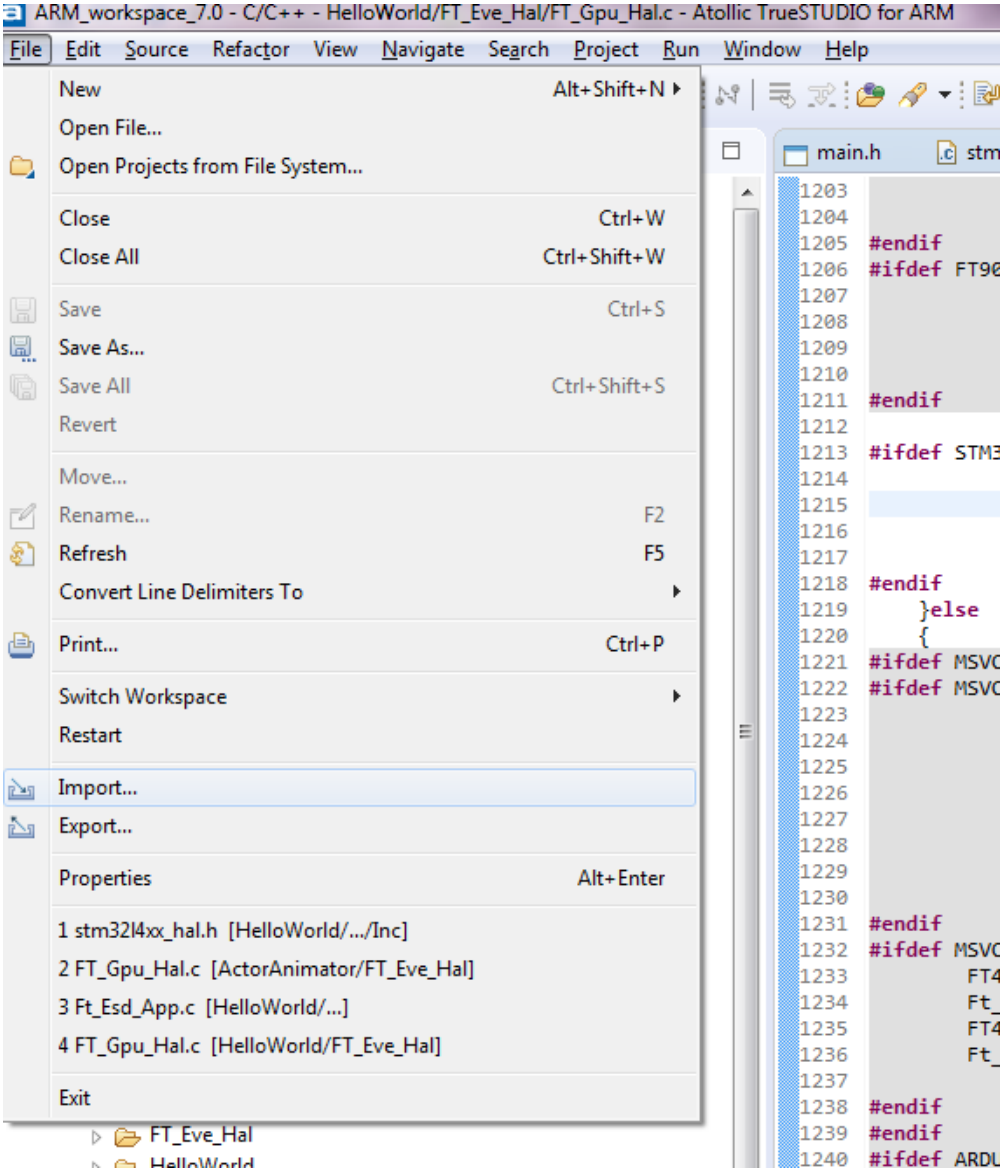


Figure 12 Import Ported Project - I

Copyright © Bridgetek Pte Limited

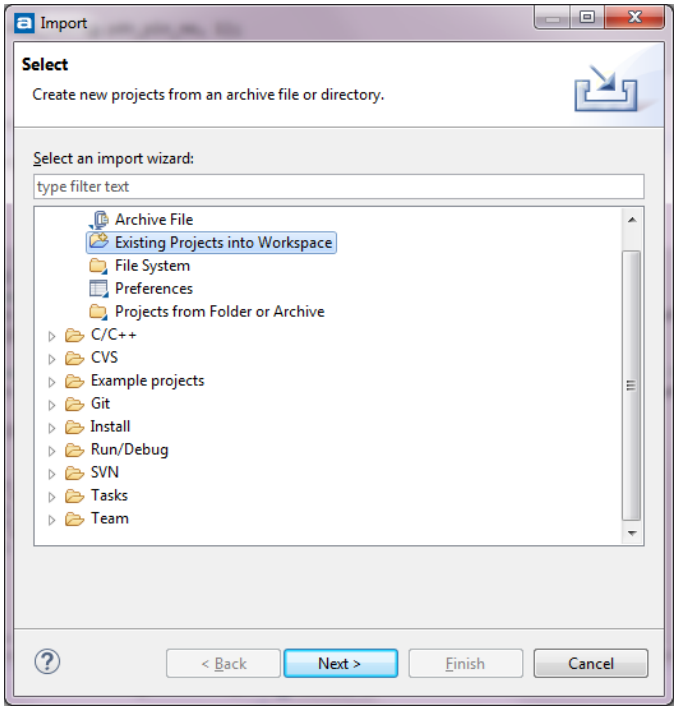


Figure 13 Import Ported Project - II

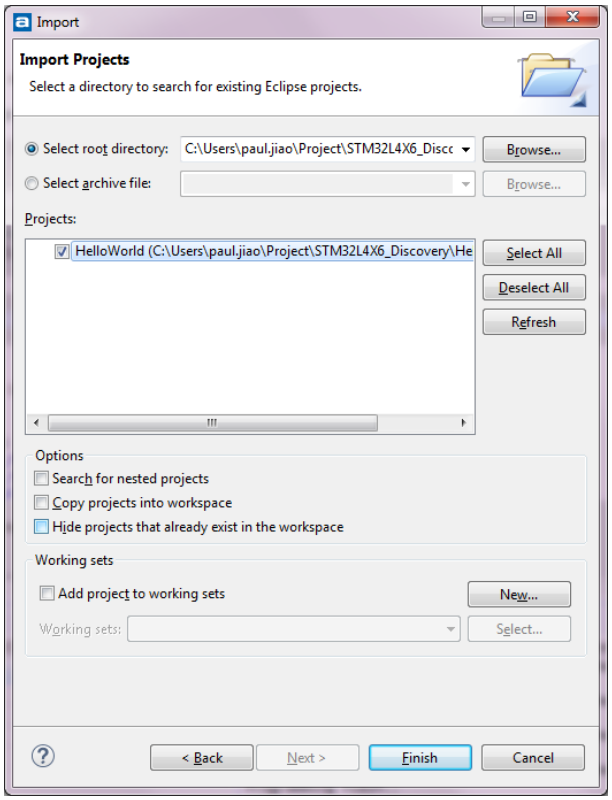


Figure 14 Import Ported Project - III

After importing the project, users can build and download the project from the Eclipse IDE if a USB cable is connected.

Copyright © Bridgetek Pte Limited

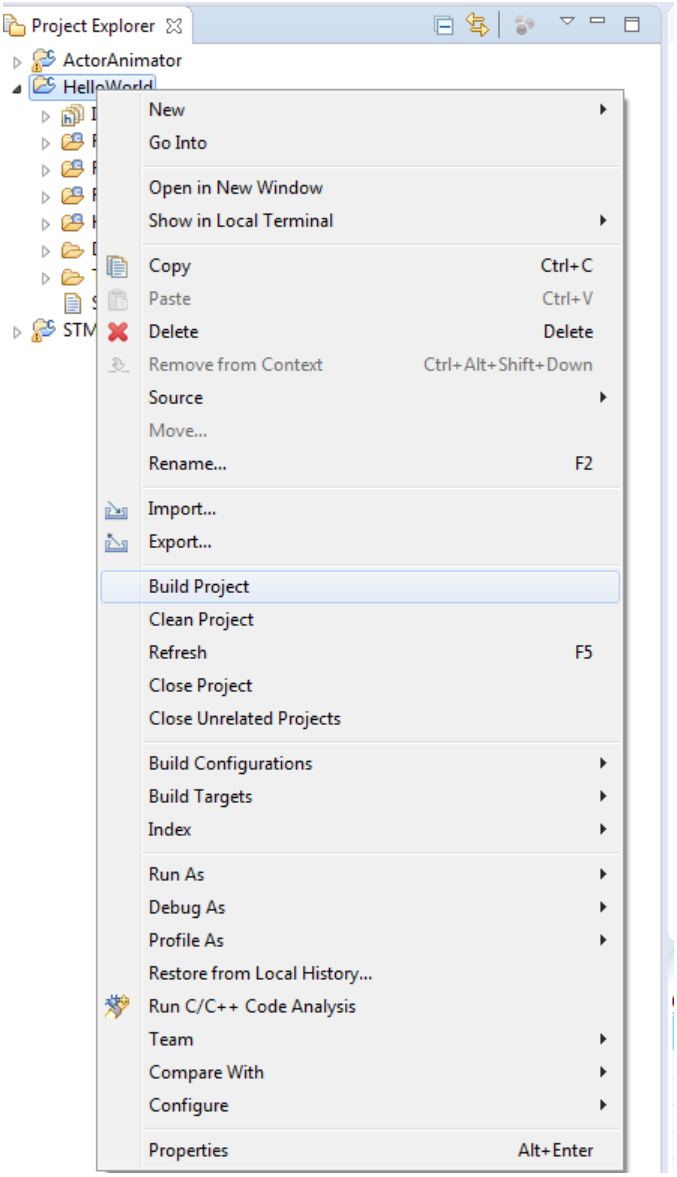


Figure 15 Build Ported Project

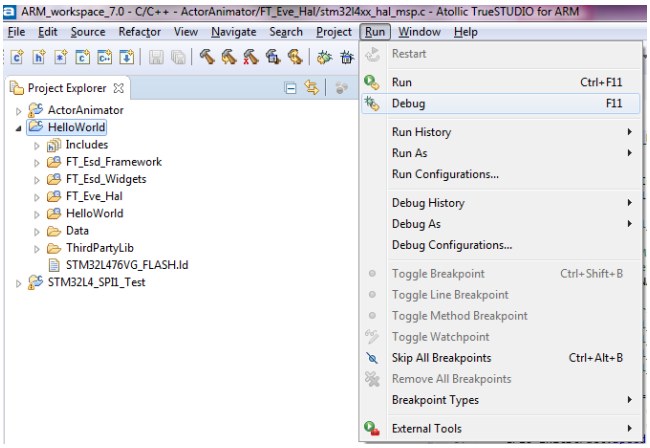


Figure 16 Debug & Download Ported Project

Copyright © Bridgetek Pte Limited

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.

System and equipment manufacturers and designers are responsible to ensure that their systems, and any Bridgetek Pte Limited (BRTChip) devices incorporated in their systems, meet all applicable safety, regulatory and system-level performance requirements. All application-related information in this document (including application descriptions, suggested Bridgetek devices and other materials) is provided for reference only. While Bridgetek has taken care to assure it is accurate, this information is subject to customer confirmation, and Bridgetek disclaims all liability for system designs and for any applications assistance provided by Bridgetek. Use of Bridgetek devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold harmless Bridgetek from any and all damages, claims, suits or expense resulting from such use. This document is subject to change without notice. No freedom to use patents or other intellectual property rights is implied by the publication of this document. Neither the whole nor any part of the information contained in, or the product described in this document, may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. Bridgetek Pte Limited, 178 Paya Lebar Road, #07-03, Singapore 409030. Singapore Registered Company Number: 201542387H.

Copyright © Bridgetek Pte Limited

Appendix A– References

Document References

STM32L4 Reference Manual

STM32L476xx datasheet

User Manual of STM32L4 Discovery board

FT81x Programmer Guide

FT81x Datasheet

Acronyms and Abbreviations

Terms	Description
EVE	Embedded Video Engine
EVE 2 Module	FT81X series based display module
FT900	FT900 Microcontroller from FTDI
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
ESD 3.0	EVE Screen Designer 3.0

Copyright © Bridgetek Pte Limited

Appendix B – List of Tables & Figures

List of Figures

Figure 1 Export HelloWorld Project in ESD 3.0 5
Figure 2: Folder Structure of HelloWorld Project 5
Figure 3 STM32L4XX Discovery Board 8
Figure 4 EVE 2 Module (ME813A-WH50C) 8
Figure 5 Hardware Setup 9
Figure 6 ESD 3.0 HelloWorld Project Screenshot 9
Figure 7 Atollic TrueSTUDIO Tool - About Window 10
Figure 8 STMCubeMX Snapshot 11
Figure 9 Disable Flash Access Keyword 15
Figure 10 Extra Files for MCU Configuration 16
Figure 11 Comparison Result of Ported VS Original Project 19
Figure 12 Import Ported Project - I 20
Figure 13 Import Ported Project - II 21
Figure 14 Import Ported Project - III 21
Figure 15 Build Ported Project 22
Figure 16 Debug & Download Ported Project 22

List of Tables

Table 1: Folder Contents 6
Table 2 MCU and EVE Connection 10
Table 3 APIs to be changed 18

Copyright © Bridgetek Pte Limited

Appendix C– Revision History

Document Title: BRT_AN_001 ESD 3.0 Exported Project Porting Guide
Document Reference No.: BRT_000071
Clearance No.: BRT#058
Product Page: http://brtchip.com/i-ft8/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial release	2017-02-09

Copyright © Bridgetek Pte Limited