# Application Note

# BRT_AN_073

# ESD Exported Project Porting Guide for STM32L4 Discovery Board

**Version 1.1**

**Issue Date: 2023-08-16**



This application note is intended as a guide for porting an **E**VE **S**creen **D**esigner (**ESD**) exported project to a non-FT9xx based MCU platform. Users are expected to have knowledge of ESD as well as BT81x and STM32L4 MCU.



**Bridgetek Pte Limited (BRTChip)**

Address: 1 Tai Seng Avenue, Tower A #03-05 Singapore 536464
Tel No : (+65) 6547 4827
Web Site: http://brtchip.com

# Table of Contents

# 1 Introduction

This application note is intended as a guide for porting an **E**VE **S**creen **D**esigner (**ESD**) exported project to an ARM Cortex-M4 based MCU platform, i.e., STM32L476 Discovery board. An example project "EvChargePoint" exported from **ESD** and the **STM32CubeIDE** are used to showcase the porting procedure. Readers are expected to have the knowledge of ESD as well as **STM32CubeIDE**.

This application note provides a step-by-step example using EVE module VM816C with a BT816 and LCD 800x480 resistive touch, below is the list of tools used:

| Hardware | Information |
|---|---|
| VA800A-SPI | brtchip.com/wp-content/uploads/Support/Documentation/Datasheets/ICs/EVE/DS_VA800A-SPI_MPSSE_Module.pdf |
| VM816C | brtchip.com/wp-content/uploads/sites/3/2021/07/DS_VM816C.pdf |
| 5.0 inch Premium Resistive TFT Display | Newheaven display NHD-5.0-800480TF-ATXL#-T<br><br>https://newhavendisplay.com/5-0-inch-premium-resistive-tft-display/ |
| 32L476GDISCOVERY | https://www.st.com/en/evaluation-tools/32l476gdiscovery.html |

**Table 1: Hardware list**

| Tool | Version | License | Information |
|---|---|---|---|
| **E**VE **S**creen **D**esigner (**ESD**) | 4.16 | Free | https://brtchip.com/esd/ |
| EVE Asset Builder | 2.6.1 | Free | https://brtchip.com/eab |
| STM32CubeIDE | 1.13.0 | Free | https://www.st.com/en/development-tools/stm32cubeide.html |
| STM32CubeMX | 6.9.1 | Free | https://www.st.com/en/development-tools/stm32cubemx.html |

**Table 2: Software list**

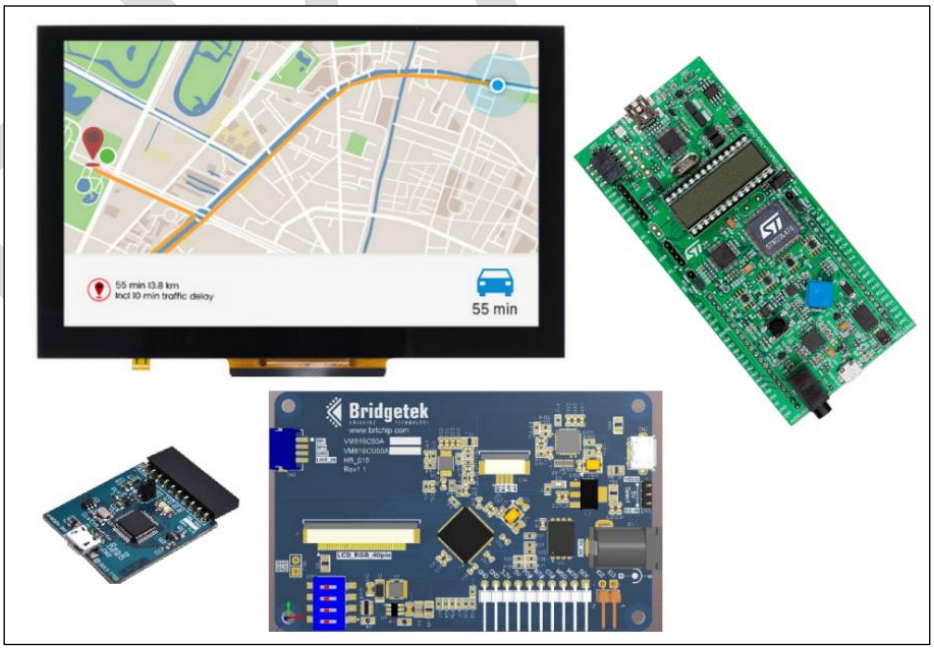



**Figure 1 Hardware list**

## Overview

This guide covers the following topics:

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

## Scope

This document covers hardware changes and software modification as well as some debugging tips while porting the exported project. It also provides some basic principles to successfully port a project.

# 2 ESD Exported Project - Introduction

ESD 4.16 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 Project"**, as shown figure below:

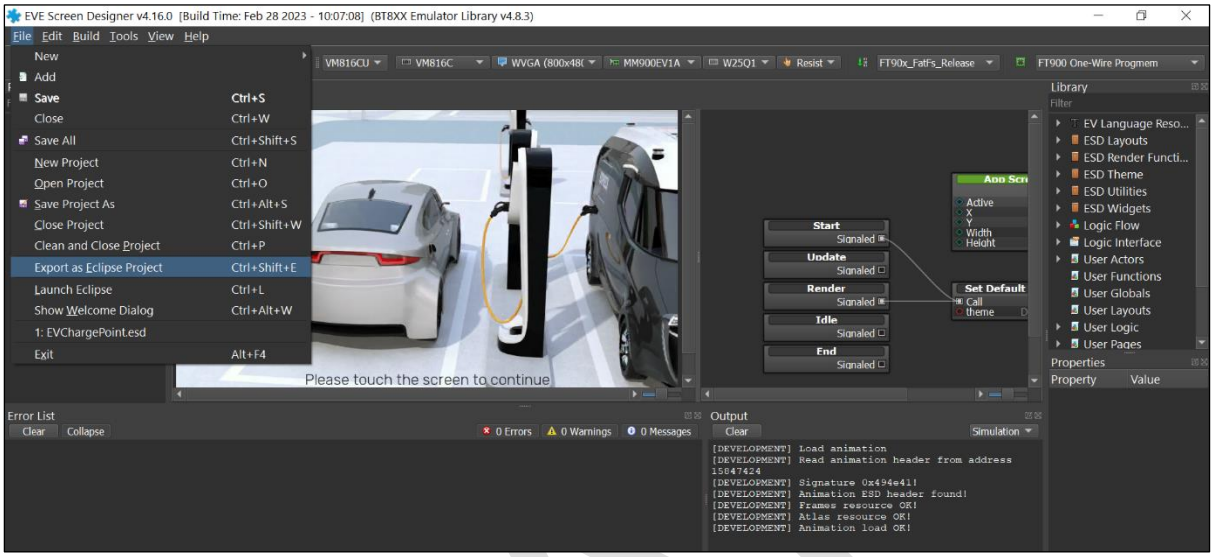



**Figure 2 Export EvChargePoint project in ESD 4.16**

"EvChargePoint**Error! Reference source not found.**" project is located at "$(ESD4.16Installation Path)\Examples\Advanced" folder and user can open it in ESD4.16directly.

The "$(ESD4.16Installation Path)" is set to "C:\Users\Public\Documents\EVE Screen Designer" by default.

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:

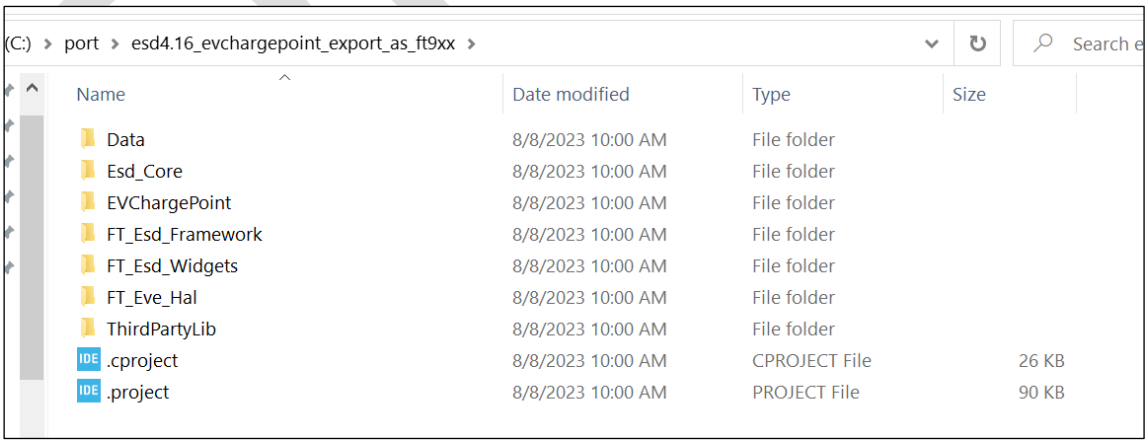



**Figure 3: Folder Structure of** Error! Reference source not found. **Project**

| Folder / file Name | Description | Remarks |
|---|---|---|
| Data | Contains the converted bitmap assets | Do not need to change. |
| ESD_Core | Contains application framework source code | Reusable and common module |
| **Error! Reference source not found.** | The project file contains the screen logic and users design | The folder name shall be same as project name |
| 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 |
| 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 3: Folder Contents**

The exported ESD project supports FT4222, RP2040 and FT90X platforms. User must select the FT9XX in this dropdown list before exporting:

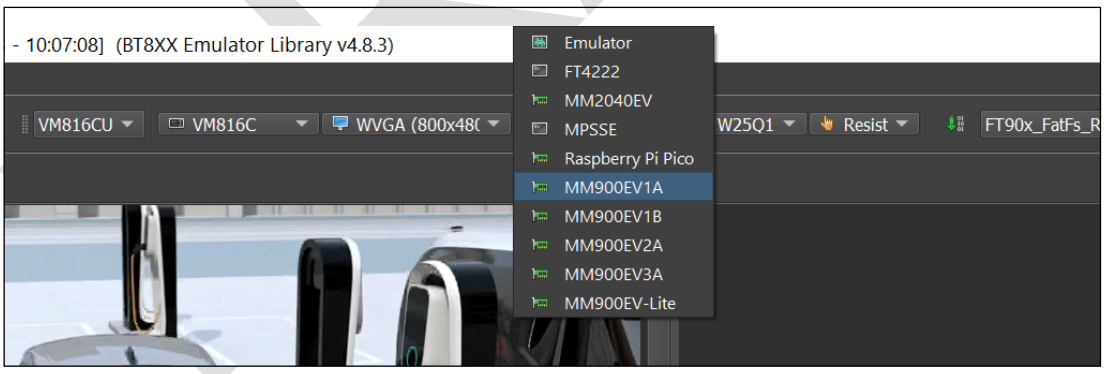



**Figure 4: Select host platform as FT9xx**

# 3 Porting principles

## Hardware

An ESD 4.16 exported project usually needs access to the following hardware resources of the MCU:

- SPI interface: Read/Write the EVE 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 BT81x series ICs as well as EVE3 series modules before starting the porting work. Refer to brtchip.com for more details.

## Software

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

- **Additional EVE_Hal implementation for the target host platform:** Initialize the targeting MCU platform and re-implement the transportation layer API for the EVE chip interface. This must be modified manually by comparing the reference project.
- **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.

# 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 5 STM32L4 Discovery Board**

Selected EVE**Error! Reference source not found.** module VM816C (included BT816 and LCD 800x480):

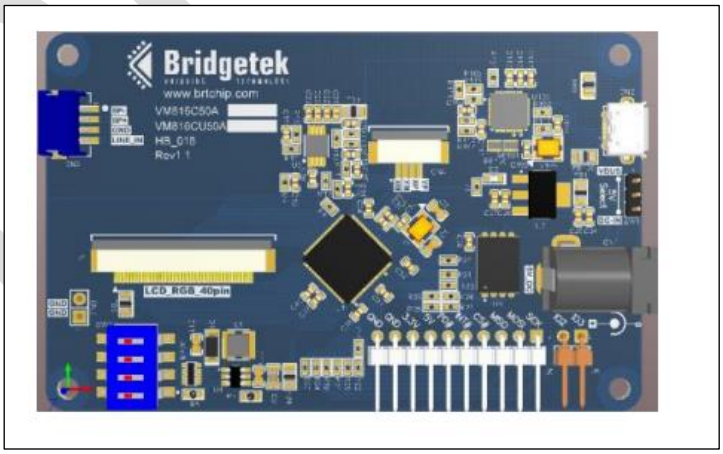



**Figure 6 VM816C Module**

Here is the connection between two boards:

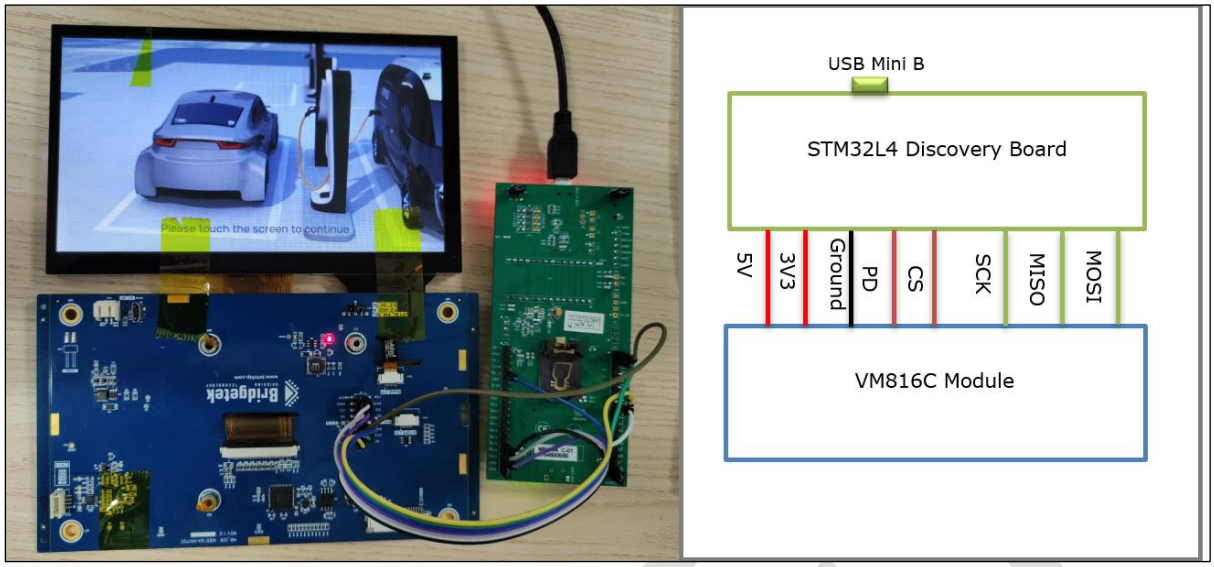



**Figure 7 Hardware Setup**

The "EvChargePoint" example project from the ESD 4.16examples folder is used to demonstrate how the porting will be done. The "EvChargePoint" example is located at *"$(ESD*4.16*Installation Folder)\\Examples\\Advanced"* folder. Here is a sample screenshot after it is opened in ESD 4.16:

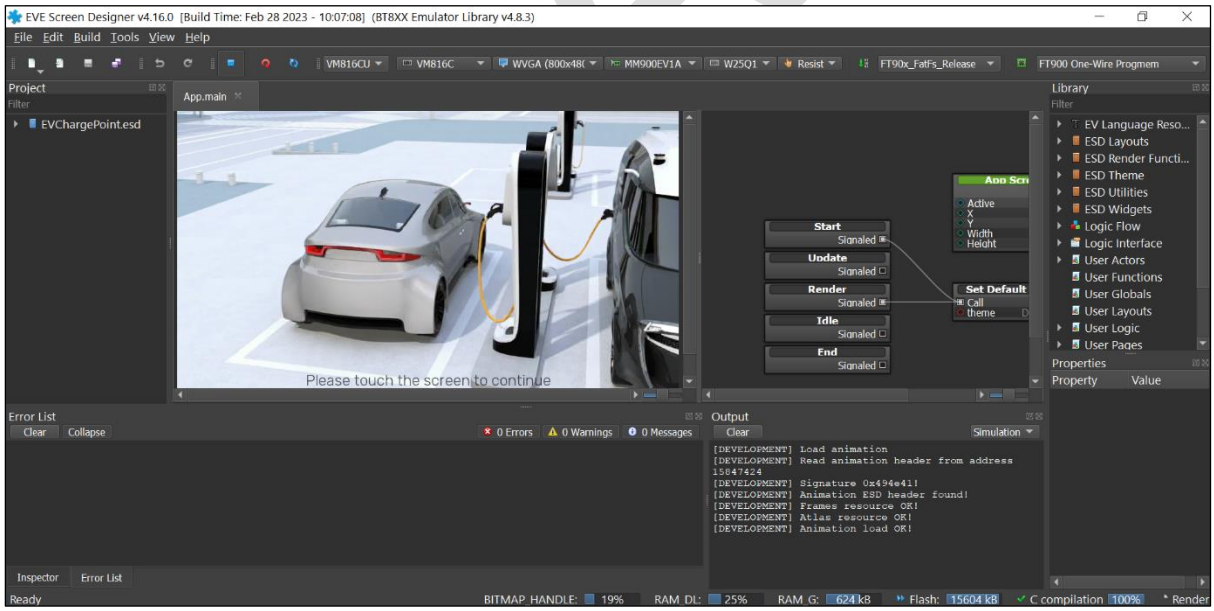



**Figure 8 ESD 4.16 EvChargePoint Project Screenshot**

# Hardware Connection

## 4.1.1 Power and Ground Connection

VM816C module is powered by both pin 5V and 3.3V, supply from the STM32L4 Discovery board and a ground pin is also connected.

The 3.3v power must be provided for the touch functionality of the VM816C.

### 4.1.2 Signal lines Connection

In this example, STM32L4'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 |
| 5V | 5V | Main power |
| 3.3V | 3.3V | Touch power |
| GND | GND | Ground |

**Table 4 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.

## Software Setup

### 4.1.3 Toolchain and Utility

For this example porting exercise, the STM32CubeIDE is selected as the compiler and linker for the STM32L4, it is a free software.

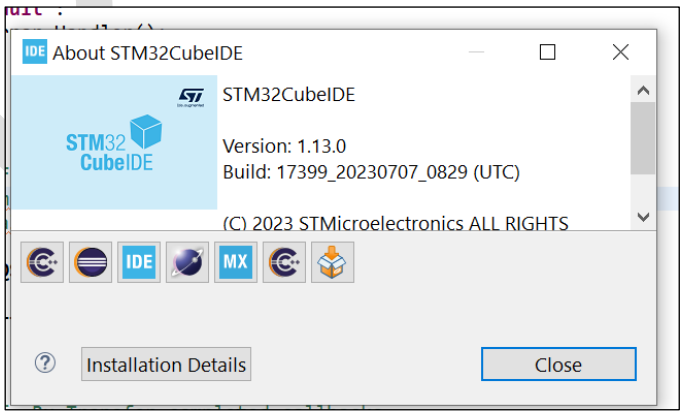


**Figure 9 STM32CubeIDE Tool - About Window**

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.

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

The following screenshot shows how the pin configuration looks like once the project is opened using the STMCubeMX tool. User can open file STM32L476G_With_EVE.ioc with STMCubeMX for more detail.

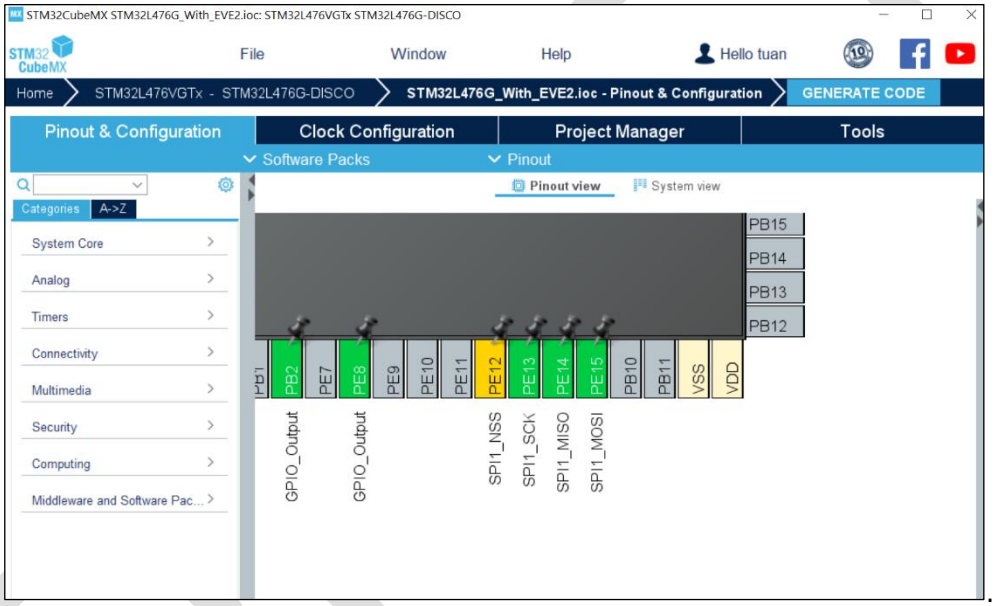



**Figure 10 STMCubeMX Snapshot**

## Project porting procedure

This section describes how to use STMCubeMX and STM32CubeIDE to port an ESD project to STM32L4 Discovery board.

Basically, we generate project for STM32L4 Discovery board by STMCubeMX at first, then use this project to build the ESD generated source code on STM32CubeIDE.

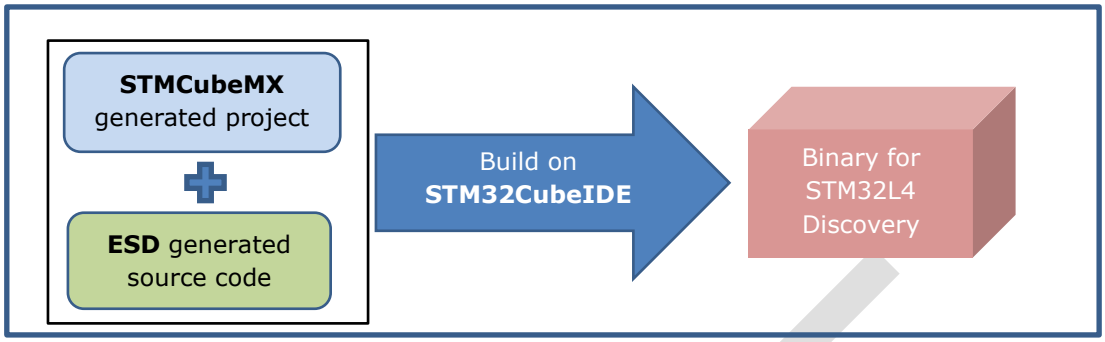



**Figure 11 Project porting procedure**

### 4.1.4 Create project on ESD and generate source code for eclipse IDE

1. On ESD, open "EvChargePoint" project
It is located at "$(ESD4.16Installation Path)\Examples\Advanced" folder.

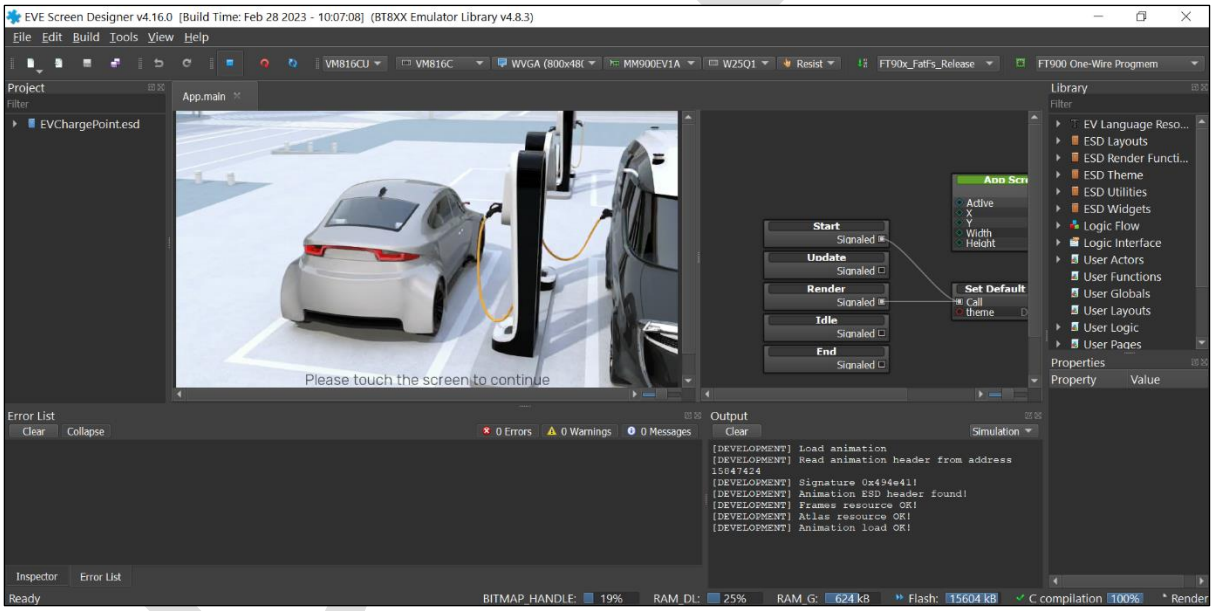



**Figure 12 The EvChargePoint project on ESD**

2. Configure the ESD to use the right Eve platform and LCD size.
In this example, we use VM816C platform, with an LCD 800x480 in size.

3. Export the project to local folder
   Select File -> "Export as Eclipse Project" and choose a local folder to store export files.

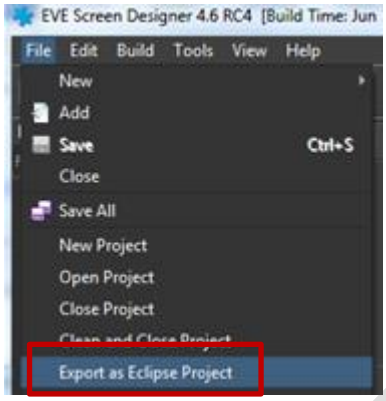



**Figure 13 Export as Eclipse Project**

The exported project has below files and folders:

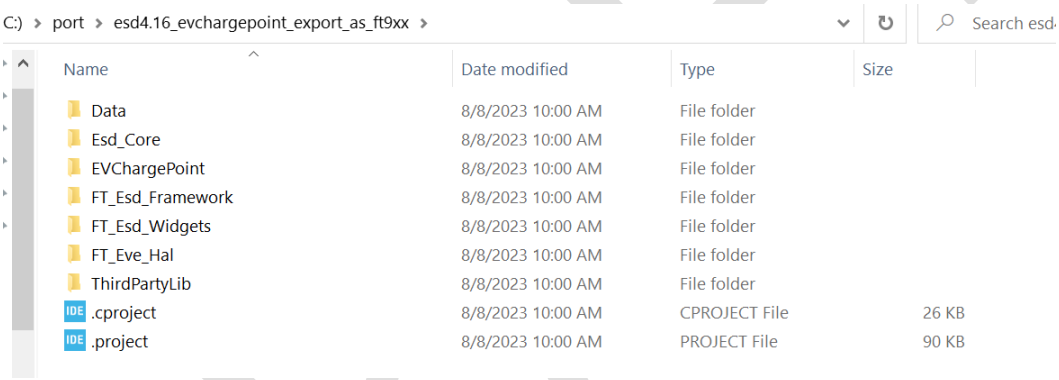
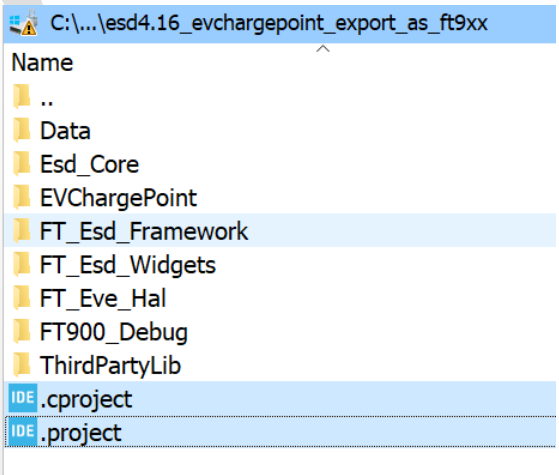



**Figure 14 ESD exported project files and folders**

Remove 2 project file from the generated project file:



### 4.1.5 Generate project for STM32L4 Discovery board

1. Open STMCubeMX and select "ASSESS TO BOARD SELECTOR"

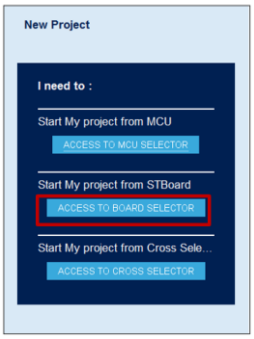



**Figure 15 ASSESS TO BOARD SELECTOR**

2. Select 32L476GDISCOVERY board

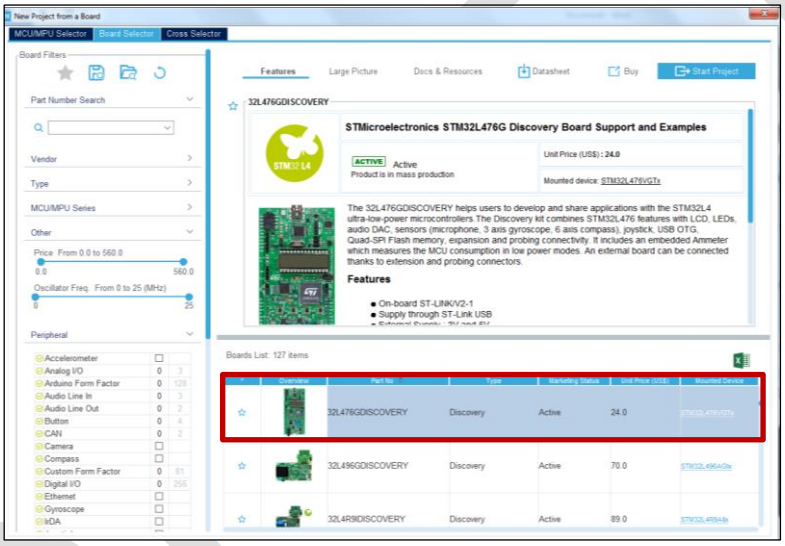



**Figure 16 Select 32L476GDISCOVERY board**

3. Select default setting for all peripherals

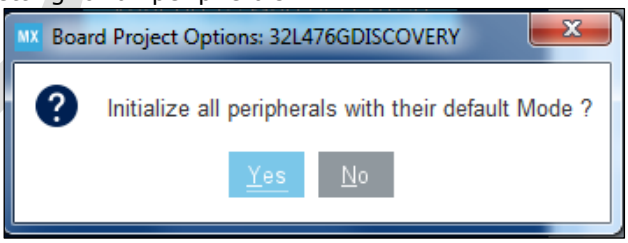



**Figure 17 select default mode**

4. The Pinout and configuration screen appear

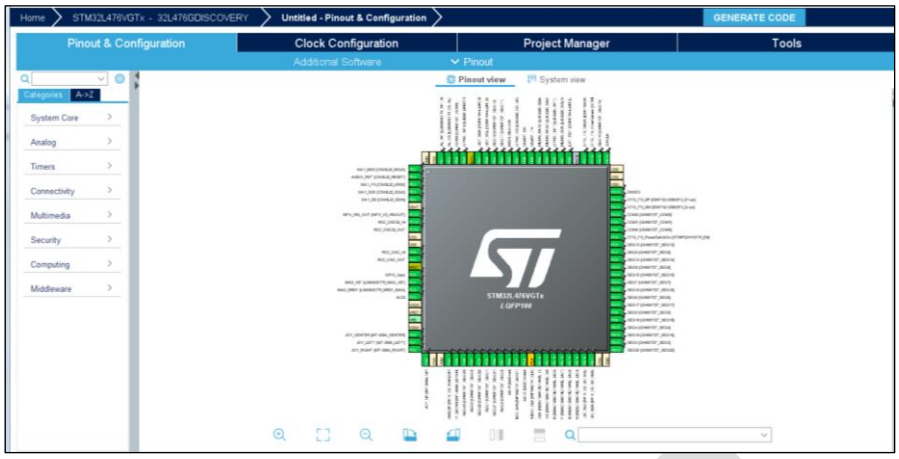



**Figure 18 Pinout and configuration screen**

5. Setting PINs for SPI1 on the STM32L4 board
Click on the PE12 pin and select SPI1_NSS.

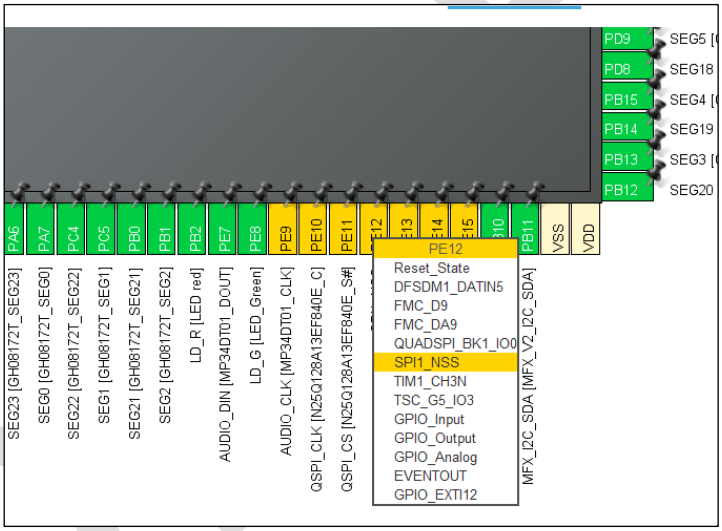



**Figure 19 Select SPI1_NSS on PE12**

Select SPI1_SCK, SPI1_MISO and SPI1_MOSI for PE13, PE14, PE15. Set PE8 and PB2 as GPIO_Output

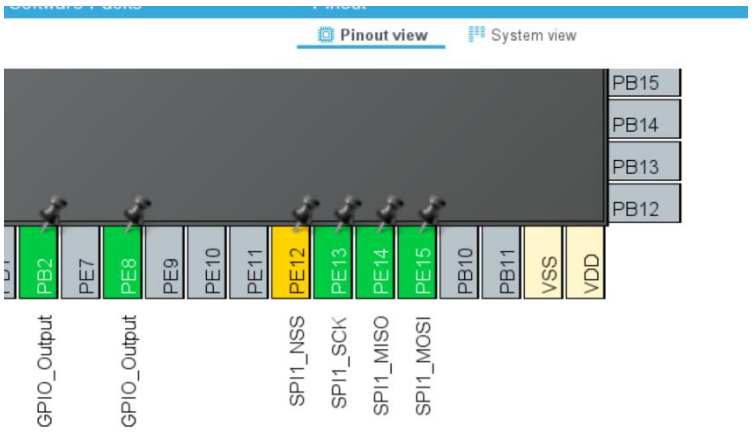



**Figure 20 Select SPI1_SCK, SPI1_MISO and SPI1_MOSI**

6. Enable SPI1
   Set SPI1 to "Full-Duplex master" mode

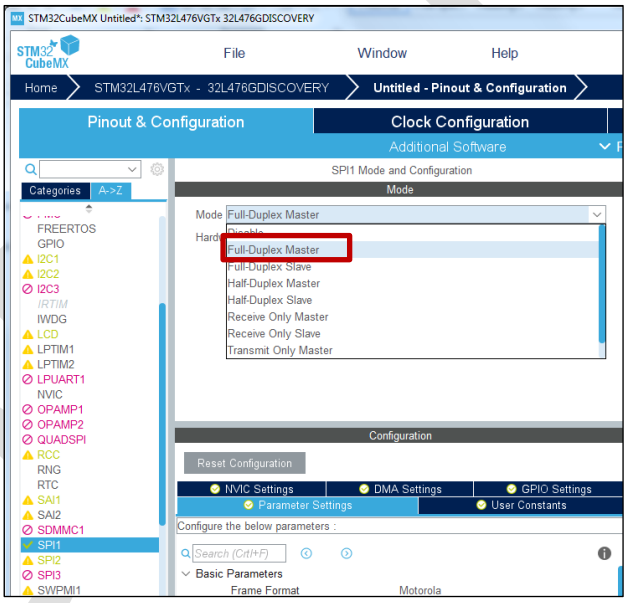



**Figure 21 Set SPI1 to Full-Duplex master**

Set Data size = 8 for SPI1

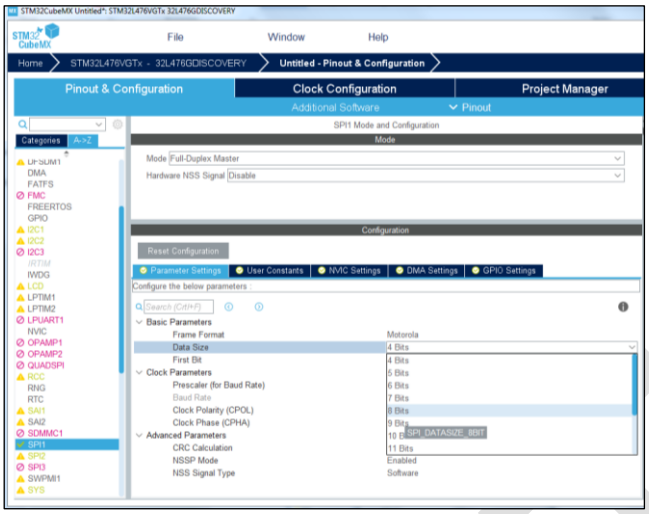



**Figure 22 SPI1 - Data size**

7. Export project for 32L476GDISCOVERY board
   - Select tab "Project manager"
   - Input the project name, such as "STM32L476G_With_EVE"
   - Input the project path
   - Select "toolchain/IDE" is "STM32CubeIDE".

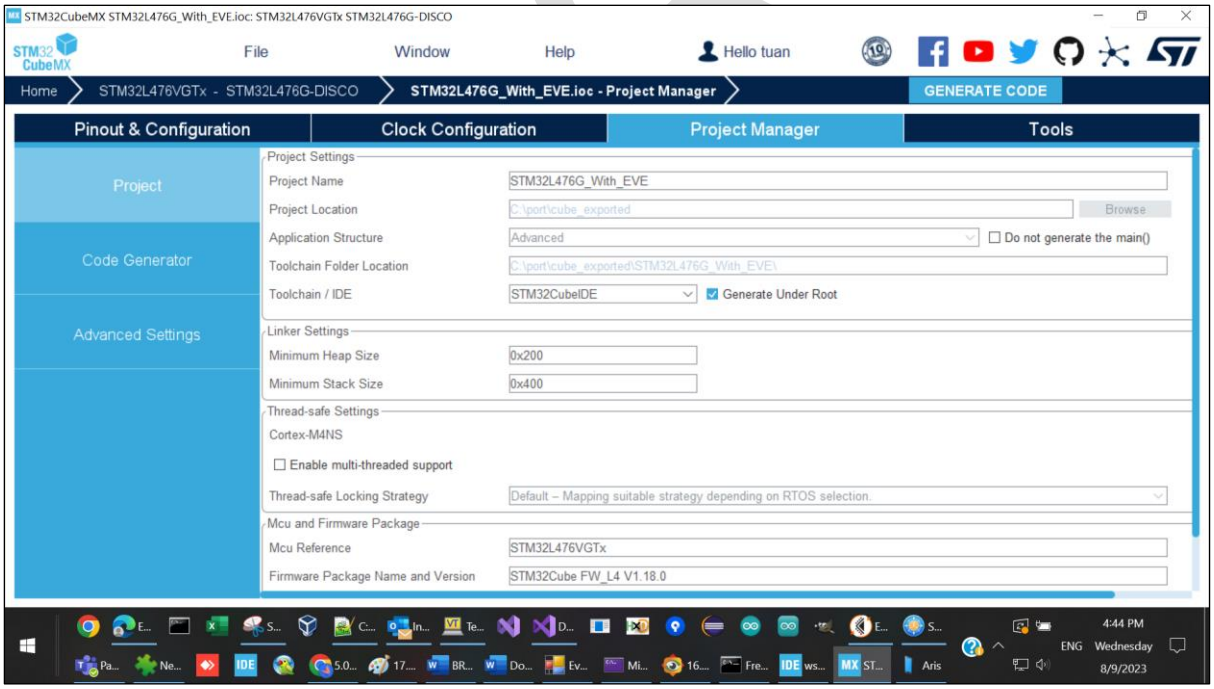



**Figure 23 Export project for 32L476GDISCOVERY board**

Click button "GENERATE CODE", wait for STMCubeMX complete this action

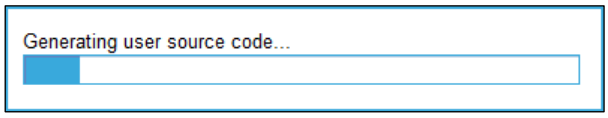



**Figure 24 Generate code in progress**

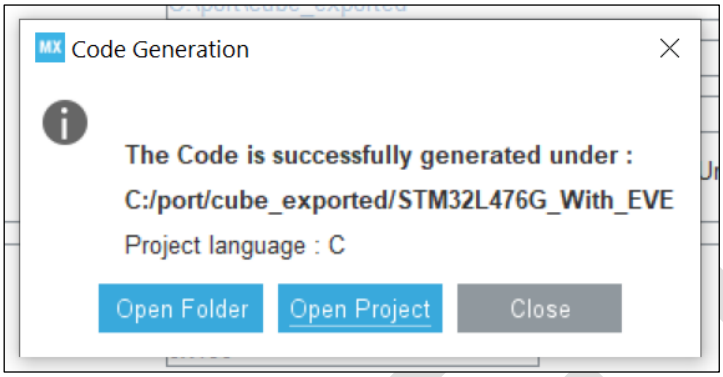



**Figure 25 Generate code done**

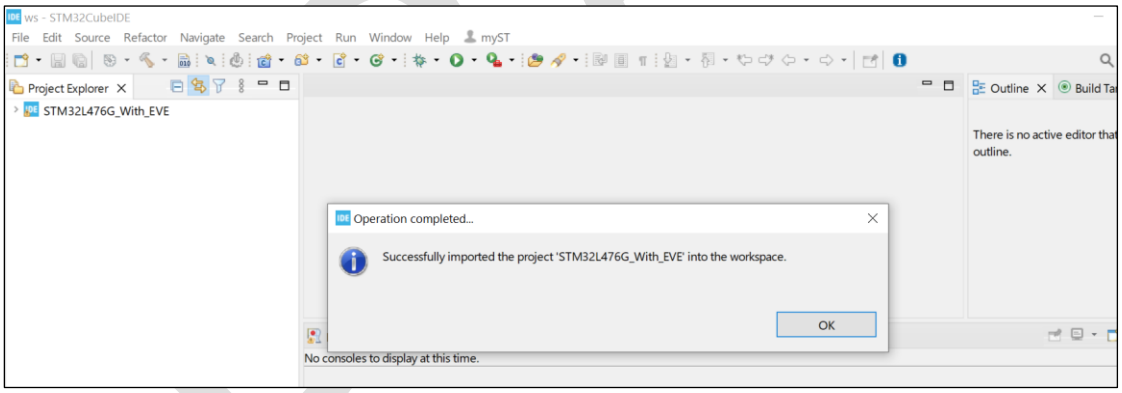



**Figure 26 Import code**

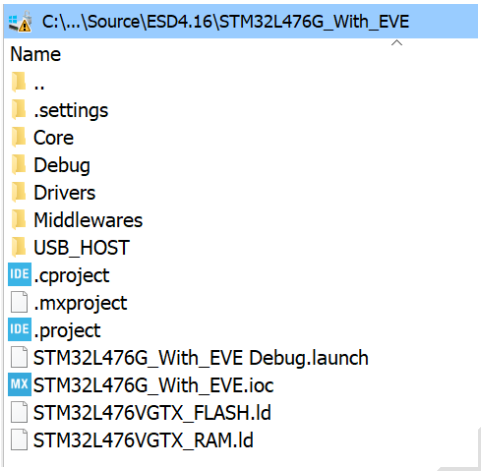



**Figure 27 the generated project files for 32L476GDISCOVERY board**

### 4.1.6 Port ESD generated source code to STM32L4 Discovery's project

#### 4.1.6.1 Load generated STM32L4 Discovery's project to STM32CubeIDE

1. Copy ESD generated files to the generated STM32L4 Discovery's project
Create subfolder "EVE_project" in the generated STM32L4 Discovery's project

Copy the whole folder "esd4.16_evchargepoint_export_as_ft9xx" into "EVE_project"

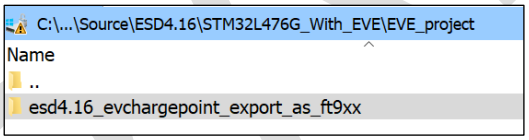



**Figure 28 Copy ESD generated source code into STM32 Cube MX generated folder**

2. Create subfolder "EVE_porting" in the generated STM32L4 Discovery's project
In this folder, add your implementation of the STM32 host platform.
Please refer to the github for an example implementation.

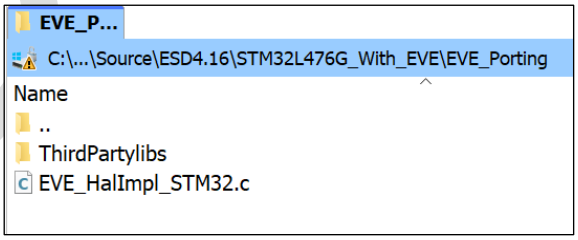



**Figure 29 A simple implementation of the STM32 host platform**

#### 4.1.6.2 Build configuration on STM32CubeIDE

1. Add include path to ESD generated header files

Right click on project name, select Properties->C/C++ General -> path and Symbols -> "includes" tab

Click "Add" button to add below include path:

- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/Esd_Core
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/EVChargePoint
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/FT_Esd_Framework
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/FT_Esd_Widgets
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/FT_Eve_Hal
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/ThirdPartyLib/fatfs
- ✓ EVE_project/esd4.16_evchargepoint_export_as_ft9xx/ThirdPartyLib/littlefs
- ✓ EVE_Porting

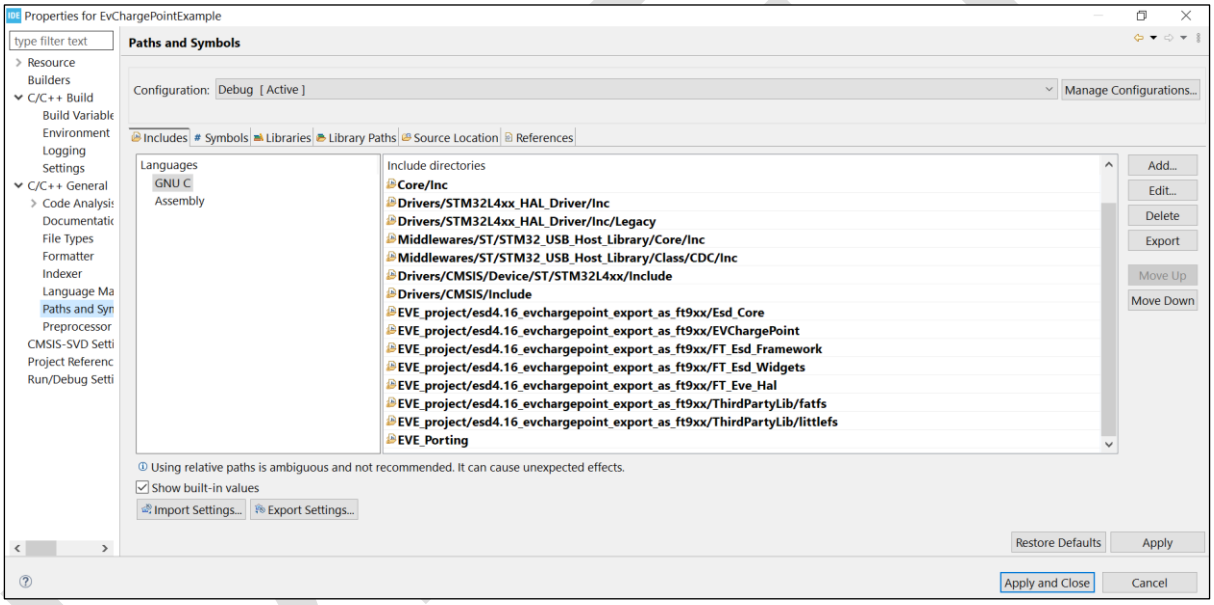



**Figure 30 Add include path to ESD generated header files**

2. Add platform macro for EVE platform and STM32L4 platform

Select tab "Symbols" and add below definitions:

- ✓ EVE_GRAPHICS_AVAILABLE
- ✓ BT816_ENABLE
- ✓ EVE_USE_INTERNAL_OSC
- ✓ EVE_TOUCH_RESISTIVE
- ✓ EVE_DISPLAY_WVGA
- ✓ EVE_HOST=99
- ✓ EVE_FLASH_W25Q16=1

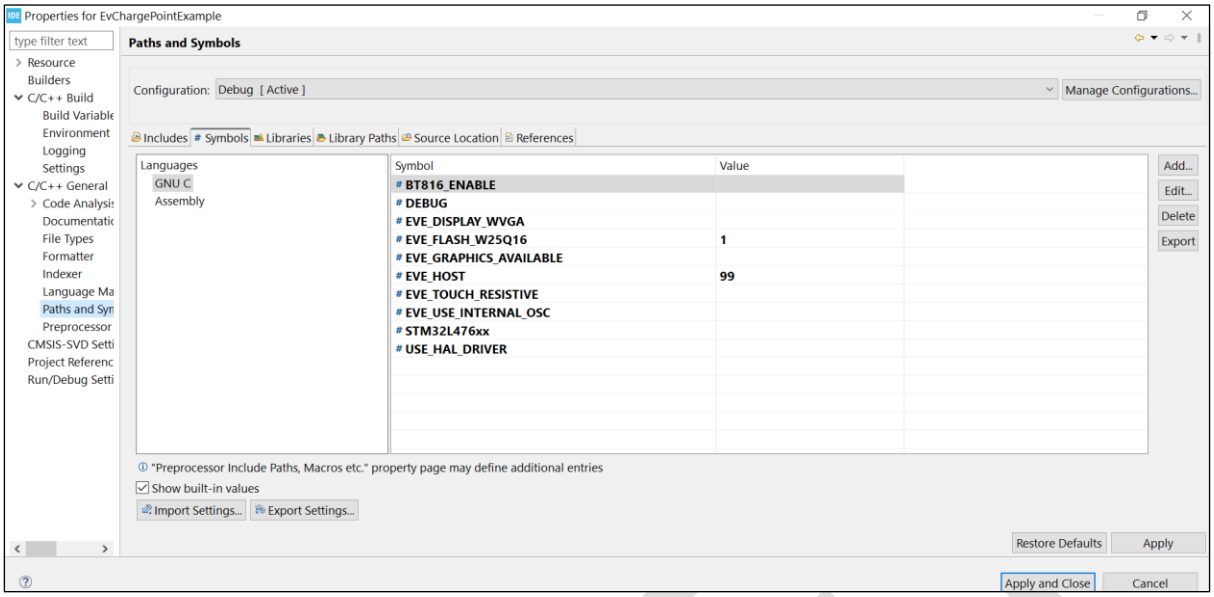



**Figure 31 Add platform macro**

#### 4.1.6.3 Source code modification

1. Open App__Generated.c file, rename "main" function to "ESD_Start"

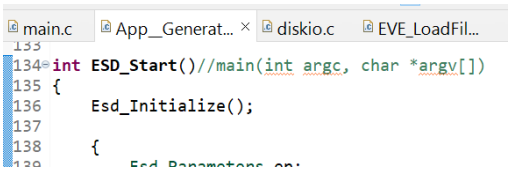



**Figure 32 Modify App__Generated.c**

1. Open main.c file, call to "ESD_Start" in the while loop

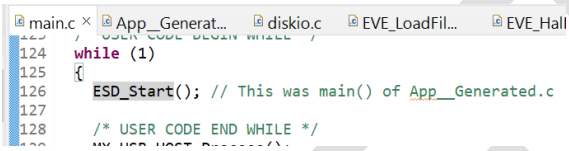



**Figure 33 Modify main.c**

2. Exclude some source files from the compilation
   Right click on below files/folder, select "Resource configuration" -> "Exclude from build":

   - ✓ esd4.16_evchargepoint_export_as_ft9xx/FT_Eve_Hal/EVE_LoadFile_STDIO.c
   - ✓ esd4.16_evchargepoint_export_as_ft9xx/FT_Eve_Hal/EVE_LoadFile_FATFS.c
   - ✓ esd4.16_evchargepoint_export_as_ft9xx/ThirdPartyLib

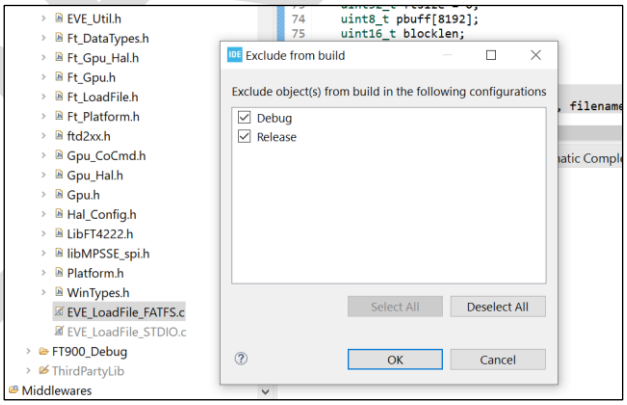



**Figure 34 Exclude from the compilation**

### 4.1.7 Build and run on STM32L4 Discovery board

1. Build project
Press Ctrl + B to compile

2. Run on STM32L4 Discovery board
Select Run->"Run Configuration"

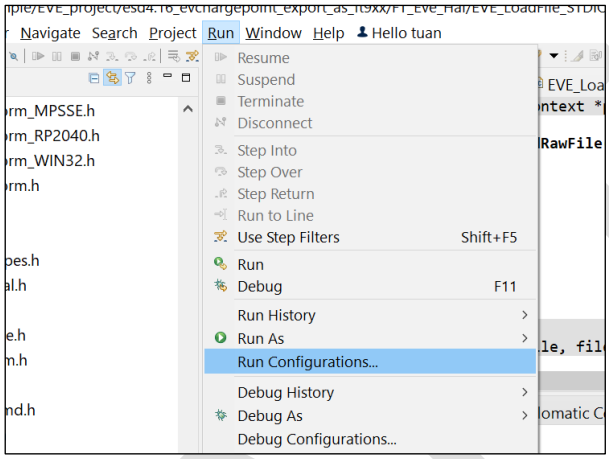



**Figure 35 Run Configuration**

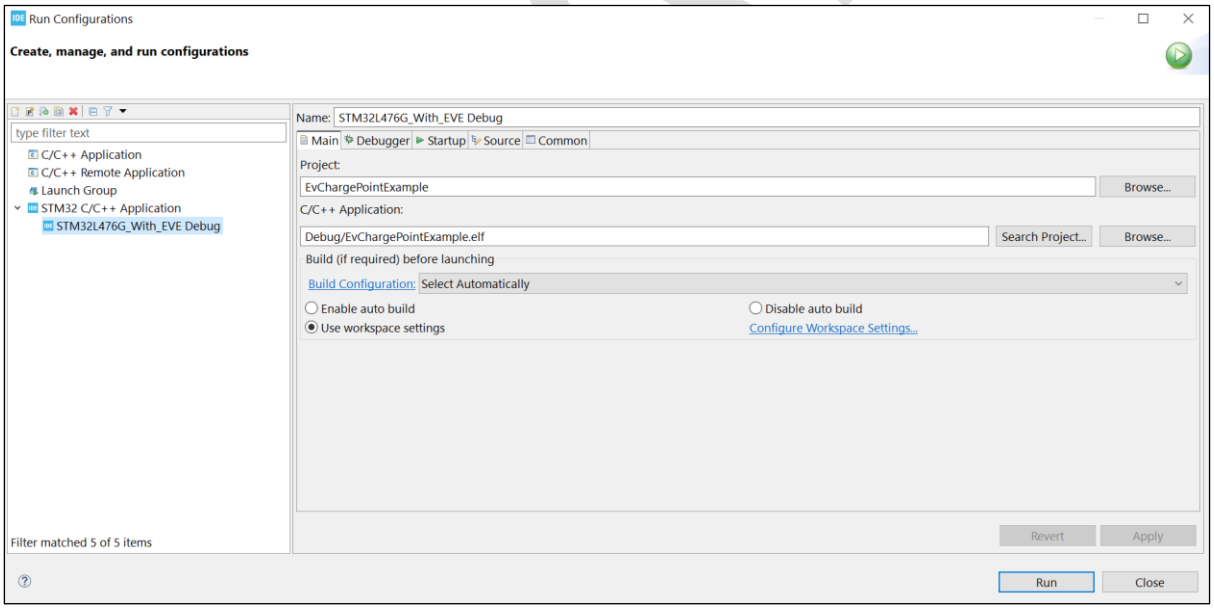



**Figure 36 Run Configuration**

Add a new run configuration for "STM32 application", click run button

The application shall be displayed on LCD

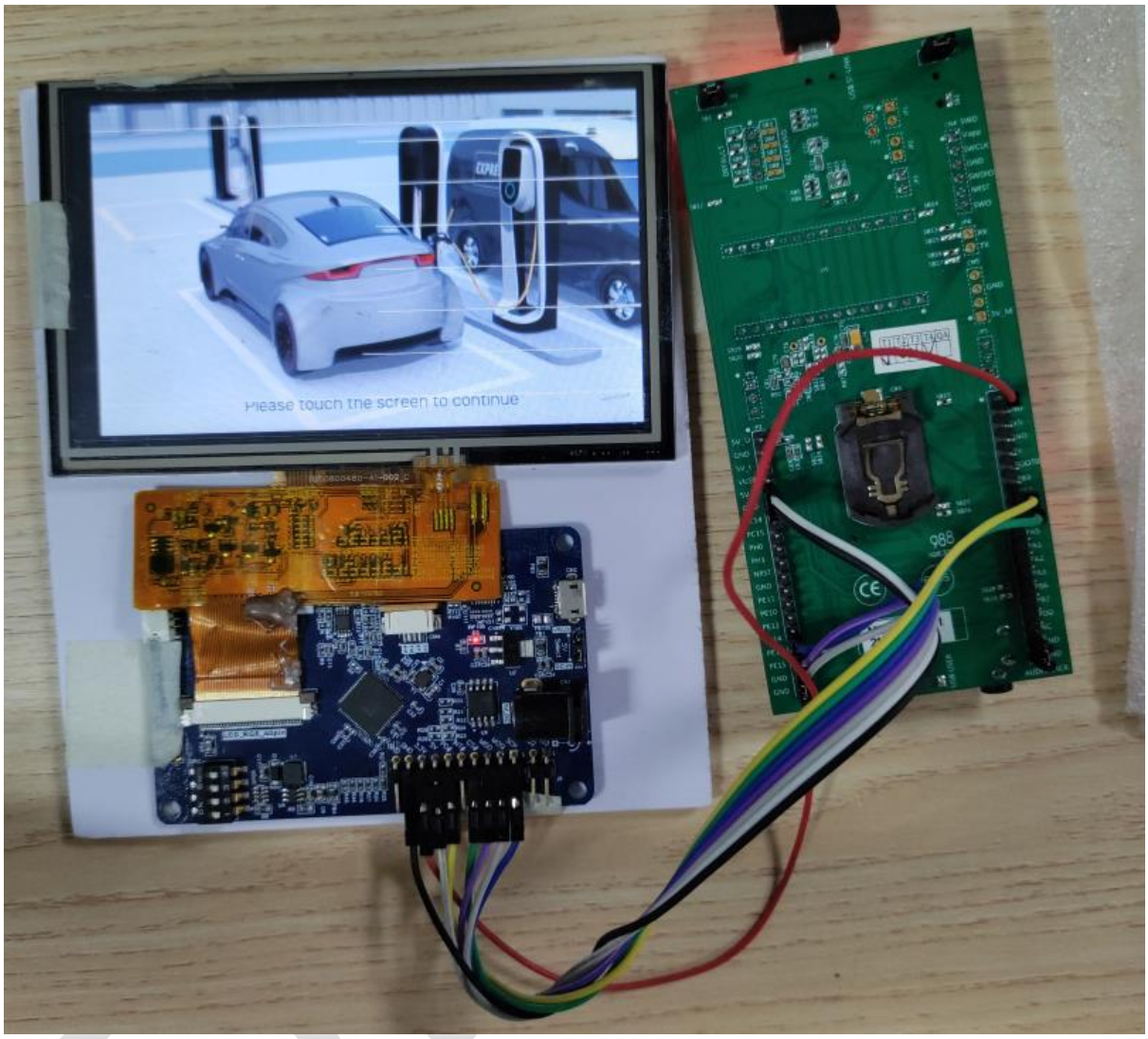



**Figure 37 The application**

### 4.1.8 Storage Media Configuration and Access

In this example, the storage media is not enabled to simplify the procedure.

To access bitmap assets, users need to re-implement the functions below in file "Src\ESD\FT_Eve_Hal\EVE_LoadFile_FATFS.c" for STM32L4 platform:

```
bool EVE_Util_loadImageFile(EVE_HalContext *phost, uint32_t address, const char *filename, uint32_t *format)

bool EVE_Util_loadInflateFile(EVE_HalContext *phost, uint32_t address, const char *filename)

bool EVE_Util_loadRawFile(EVE_HalContext *phost, uint32_t address, const char *filename)

bool EVE_Util_loadSdCard(EVE_HalContext *phost)
```

User may need have a FATFS library to implement these functions.

| API name | Remarks |
|---|---|
| Ft_Hal_LoadImageFile | It loads the image file from storage media with specified "filename" to EVE RAM_G "address" and sends the data through coprocessor command "CMD_LOADIMAGE". |
| Ft_Hal_LoadInflateFile | It loads the deflated file from storage media with specified "filename" to EVE RAM_G "address" and sends the data through coprocessor command "CMD_INFLATE". |
| Ft_Hal_LoadRawFile | It loads the raw image file from storage media with specified "filename" to EVE RAM_G "address". |
| Ft_Hal_LoadSDCard | Prepare SD card if it is used in FATFS. Otherwise, just make it as empty. |

In our example, these four functions are kept Empty.

The STM32L476G Discovery doesn't have internal flash memory or SD card reader port, it can communicate with an external SD card reader to read or write files like image below:

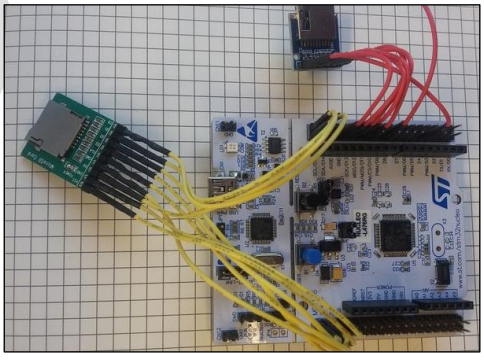


**Figure 38 External SD card reader**

### 4.1.9 APIs Re-Implementation

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 done in EVE_HalImpl_STM32L476GDISCOVERY.c.

| No | API name | Header file | Remarks |
|---|---|---|---|
| 1 | EVE_millis | EVE_HalImpl.h | Global counter of up time |
| 2 | EVE_Millis_initialize | EVE_HalImpl.h | Initialize the global counter |
| 3 | EVE_Millis_release | EVE_HalImpl.h | De- initialize the global counter |
| 4 | EVE_sleep | EVE_Hal.h | Sleep in miliseconds |
| 5 | EVE_Mcu_initialize | EVE_HalImpl.h | Initialize the MCU and its peripheral (SPI, GPIO) |
| 6 | EVE_Mcu_release | EVE_HalImpl.h | Initialize the MCU and its peripheral (SPI, GPIO) |
| 7 | EVE_HalImpl_initialize | EVE_HalImpl.h | Initialize system |
| 8 | EVE_HalImpl_release | EVE_HalImpl.h | De-initialize system |
| 9 | EVE_HalImpl_defaults | EVE_HalImpl.h | Get the default configuration parameters |
| 10 | EVE_HalImpl_open | EVE_HalImpl.h | Prepare software structure |
| 11 | EVE_HalImpl_close | EVE_HalImpl.h | Close software structure handle |
| 12 | EVE_HalImpl_idle | EVE_HalImpl.h | Idle state of the application, will be called regularly to update frequently changing internal state |
| 13 | EVE_Hal_startTransfer | EVE_HalImpl.h | Initiate address phase by transmitting 3 bytes address code and assert CS |
| 14 | EVE_Hal_endTransfer | EVE_HalImpl.h | De-assert the CS to end the SPI transferring |
| 15 | EVE_Hal_flush | EVE_HalImpl.h | Flush command buffer to EVE |
| 16 | EVE_Hal_transfer8 | EVE_HalImpl.h | Send or receive one byte |
| 17 | EVE_Hal_transfer16 | EVE_HalImpl.h | Send or receive two bytes |
| 18 | EVE_Hal_transfer32 | EVE_HalImpl.h | Send or receive four bytes |
| 19 | EVE_Hal_transferMem | EVE_Hal.h | Send or receive a buffer |
| 20 | EVE_Hal_transferProgmem | EVE_Hal.h | Send a PROGMEM buffer |

| No | API name | Header file | Remarks |
|---|---|---|---|
| 21 | EVE_Hal_transferString | EVE_Hal.h | Send a string |
| 22 | EVE_Hal_hostCommand | EVE_Hal.h | Send host commands to EVE |
| 23 | EVE_Hal_hostCommand Ext3 | EVE_Hal.h | Send 3 bytes host commands to EVE |
| 24 | EVE_Hal_powerCycle | EVE_Hal.h | Toggle PD pin to wake up EVE |
| 25 | EVE_Hal_setSPI | EVE_Hal.h | Switch SPI mode (Single, Double, Quad) |

**Table 5 APIs to be changed**

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

STM32L4 Reference Manual

STM32L476xx datasheet

User Manual of STM32L4 Discovery board

BT81x Programmer Guide

BT81x Datasheet

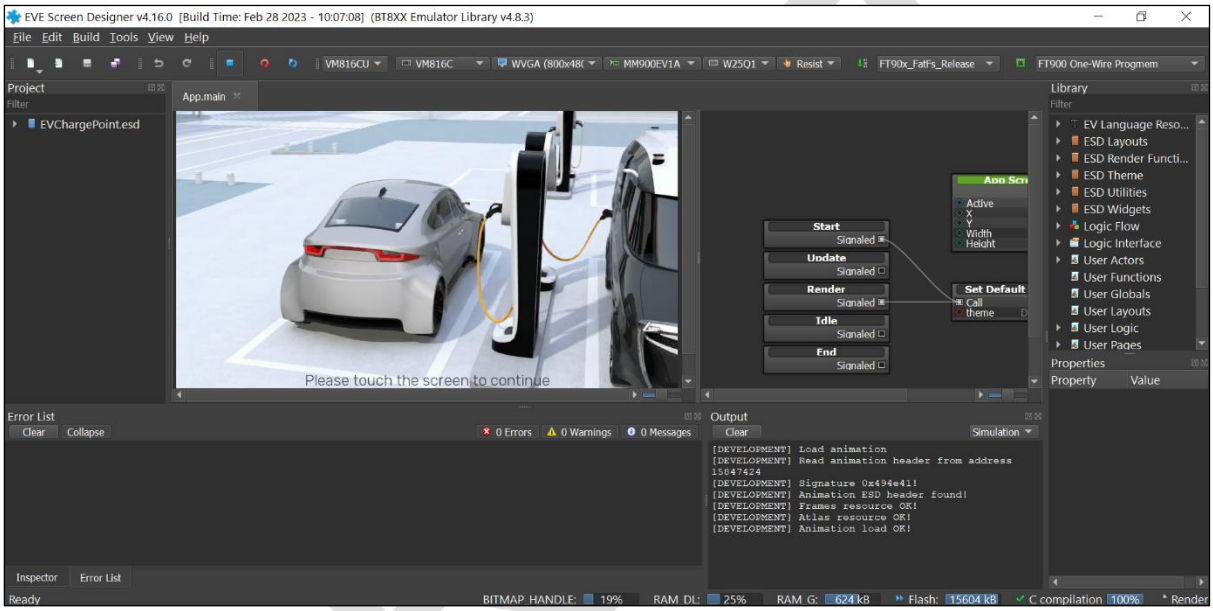
## Acronyms and Abbreviations

| Terms | Description |
|---|---|
| EVE | Embedded Video Engine |
| EVE3 Module | BT81X series based display module |
| FT900 | FT900 Microcontroller from FTDI |
| SPI | Serial Peripheral Interface |
| USB | Universal Serial Bus |
| ESD 4.16 | EVE Screen Designer 4.16 |

# Appendix B – List of Tables & Figures

## List of Figures





## List of Tables

# Appendix C– Revision History

| | |
|---|---|
| Document Title: | BRT_AN_073 ESD Exported Project Porting Guide for STM32L4 Discovery Board |
| Document Reference No.: | BRT_000206 |
| Clearance No.: | NA |
| Product Page: | http://brtchip.com/utilities/#ESD4 |
| Document Feedback: | Send Feedback |

| Revision | Changes | Date |
|---|---|---|
| 0.1 | First Draft | 2017-09-25 |

# Appendix D – Sample source code

Below is the source code of the application mentioned in (4.3), it can be imported to STM32CubeIDE to be built and ran (see 4.3.3.1, at 2)

Please check STM32L476G_With_EVE.zip

# Revision History

Revision history (internal use only, please clearly state all changes here before saving the file)

| Revision | Date YYYY-MM-DD | Changes | Editor |
|---|---|---|---|
| 0.1 | 2019-07-18 | Update to ESD 4.6, use STM32CubeIDE instead of TrueSTUDIO | Tuan Nguyen |
| 1.1 | 2023-08-16 | Update to ESD 4.16 | Tuan Nguyen |