# Contents

I. Preface .................................................................................................................. 4
   A. Purpose ................................................................................................................. 4
   B. Related documents ................................................................................................. 4
   C. Acronyms and Abbreviations ............................................................................... 4

II. Overview .................................................................................................................... 5
   A. Introduction .......................................................................................................... 5
   B. Key Features ......................................................................................................... 5
   C. Known Issues & Limitations .............................................................................. 5
   D. Credits .................................................................................................................. 5

III. Setup and Installation .............................................................................................. 6
   E. System Requirements ......................................................................................... 6
   A. Installing EVE Asset Builder .............................................................................. 6

IV. Getting Started ......................................................................................................... 9
   A. Session .................................................................................................................. 9
      1. Start Session ...................................................................................................... 10
      2. Load Session ..................................................................................................... 10
      3. Stop Session ....................................................................................................... 11
   B. Image Utilities ..................................................................................................... 11
   C. Video Converter .................................................................................................. 13
   D. Audio Converter .................................................................................................. 14
   E. Generate Font ....................................................................................................... 15
      1. Legacy Format ..................................................................................................... 16
      2. Extended format [BT81X] ................................................................................ 17
   F. Generate Animation ............................................................................................. 18
   G. Flash Utilities ........................................................................................................ 20
      1. Generated Flash Image ....................................................................................... 20
      2. Detect Flash ........................................................................................................ 21
      3. Program Flash .................................................................................................... 22
      4. Read Flash .......................................................................................................... 22
      5. INSTALL BLOB (for advanced user) ................................................................. 22
      6. Sample code ...................................................................................................... 23
   H. Asset Compressor ................................................................................................. 24
   I. Custom Touch ........................................................................................................ 25
   J. Bin2C ..................................................................................................................... 26
   K. SampleApp ........................................................................................................... 27

V. Example ..................................................................................................................... 28
   A. Image converter .................................................................................................... 28
   B. Video Converter .................................................................................................. 30
   C. Audio Converter .................................................................................................. 32
   D. Generate Font ...................................................................................................... 34
   E. Generate Animation and Generate Flash ............................................................ 36

VI. Contact Information ............................................................................................... 39
Appendix A – List of Figures ................................................................. 40
Appendix B – Revision History .......................................................... 42
I. Preface

A. Purpose

This document describes the features and procedures involved in using the EVE Asset Builder (EAB).

B. Related documents

- AN_391 EVE Platform Guide
- BT8XX Series Programming Guide

C. Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>Terms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAB</td>
<td>EVE Asset Builder</td>
</tr>
<tr>
<td>EVE</td>
<td>Embedded Video Engine</td>
</tr>
<tr>
<td>HAL</td>
<td>Hardware Abstraction Layer</td>
</tr>
<tr>
<td>MSVC</td>
<td>Microsoft Visual Studio C++</td>
</tr>
<tr>
<td>SPI</td>
<td>Serial Peripheral Interface</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
</tbody>
</table>
II. Overview

A. Introduction

EVE Asset Builder (EAB) is a Windows based tool that contains useful utilities to generate resources for EVE series devices.

This document provides a guideline on how to use EAB tools.

B. Key Features

The following are some of the key features of EVE Asset Builder:

- Image Utilities
- Video Converter
- Audio Converter
- Generate Font
- Generate Animation
- Flash Utilities
- Asset Compressor
- Custom Touch Compiler
- Binary File To C Array Converter

C. Known Issues & Limitations

There is a warning from Qt Framework "qt.qpa.fonts: Unable to open default EUDC font: "EUDC.TTE". This warning only occurs when Qt cannot open file “EUTC.TTE” in the local PC. It does not affect the EAB.

D. Credits

Open Source Software

- Python: https://www.python.org/
- FFmpeg: https://www.ffmpeg.org/
- OptiPNG: http://optipng.sourceforge.net/
- Jpegtran: http://jpegclub.org/jpegtran/

Icons Copyright

Icons made by "Freepik", "Smashicons" and "Dave Gandy" from "www.flaticon.com"
III. Setup and Installation

E. System Requirements

To install the application, ensure that your PC meets the requirements recommended below:

- Ideally Windows 10; alternatively Windows 8 or 7 with the latest windows updates
- 64-bit platform
- 1.6GHz or faster processor
- 1GB of RAM (1.5GB if running on a virtual machine)
- Multi-Core CPU is highly recommended
- At least 512MB hard disk space
- Display resolution 1080 x 800 pixels or higher
- "Write" permission to the installation folder

A. Installing EVE Asset Builder

The following steps will guide you through the EVE Asset Builder Setup/Installation process.


ii. When prompted with a download dialog box. Click on Save.

iii. Navigate to the folder under which the package files are downloaded.

iv. Extract the zip file contents. Double click on the file EVE-Asset-Builder-setup.exe

v. The EVE Asset Builder Setup Wizard is displayed along with a Welcome message.

![Figure 1 EVE Asset Builder Setup Wizard](image-url)
vi. Click **Next** to select a “Destination Folder” for installing the files. Accept the default folder or click **Browse** to specify a different location. Click **Next** to confirm the destination folder and continue.

![Figure 2 EVE Asset Builder Setup – Select destination folder](image)

vii. In the **Select Additional Tasks** window, check “**Create a desktop / Create Quick Launch shortcut**” boxes, to have the EVE Asset Builder icon and Quick Launch shortcut displayed on the desktop if required. Click **Next** to prepare for the installation.

The initial setup is completed and the application is ready for installation.

![Figure 3 EVE Asset Builder Setup – Ready for Installation](image)
viii. Click **Install** to start the installation. A progress bar indicates that the installation is in progress.

![Figure 4 EVE Asset Builder Setup – Installing in Progress](image)

ix. Upon successful installation, click **Finish**. The EVE Asset Builder application UI is displayed.

![Figure 5 EVE Asset Builder Setup - Finish](image)
IV. Getting Started

A. Session

This feature saves user actions and loads them later. Users can start a new session, load and stop a session via 3 buttons on the left edge:

**START SESSION**: To start recording actions from user, all information are stored in a session file.

**STOP SESSION**: To stop recording.

**LOAD SESSION**: To load a session file and restore all inputs.

![Figure 6 Asset Building Session](image-url)
1. Start Session

Users have to select an empty folder, which will become the output folder for all utilities in EAB. Sessions should be named in order to find easily at the time of loading. Once a session has started, all inputs from users will be collected then saved to a file which has name <session name>.eab

![Session Dialog]

**Figure 7 Session dialog**

2. Load Session

Load a session which has previously been saved. Every user input will be filled again.

![Select Session File]

**Figure 8 Select session file**
3. Stop Session

All inputs from the user are saved to the session file. From the time of stopping a session, EAB will not collect user inputs anymore. The “Stop Session” button is enabled only when a session is started or loaded.

B. Image Utilities

The first tab is used to convert PNG/JPEG/BMP images to EVE compatible image format. The second tab is targeted to check the compatibility of PNG/JPEG file for CMD_LOADIMAGE.

![Image Converter Tab]

Figure 9 Image converter tab
Figure 10 PNG/JPEG validator tab

**Input Files:** PNG/JPG/BMP files to be converted. Shortcut key Ctrl+Up and Ctrl+Down can be used to change the order of images.

**Output folder:** Folder contains converted files.

**EVE Product Range:** Supported BT81X, FT81X, FT80X.

**Output Format:** Output format of images, it is one of the following: ARGB1555, L1, L2, L4, L8, RGB232, ARGB2, ARGB4, RGB565, PALETTED565, PALETTED4444, PALETTED8, ASTC format from 4x4 to 12x12, DXT1

**ASTC Preset:** Only available for BT81X. Available options are: veryfast, fast, medium, thorough and exhaustive. Quality increases from veryfast to exhaustive while encoding speed decreases as well.

**ASTC Options:** More options of ASTC.
C. Video Converter

Convert video file into EVE-compatible AVI file which is in MJPEG format and mono audio channel. Output is a converted AVI file and a sample C file that shows how to playback video on EVE.

**Input:** The original video to be converted.

**Output Folder:** Folder contains converted files.

**Width And Height:** Must be specified, the video will be converted to this size.

**Output Video Codec:** Only supported codec is MJPEG.

**Output Audio Sample Rate:** Audio sample rate of the output video, can be same as input video (default) or any value selected from the list. They are: 8000Hz, 11025Hz, 16000 Hz, 22050 Hz, 32000 Hz, 32780 Hz, 44056 Hz, 44100 Hz, and 48000 Hz.
D. Audio Converter

Convert WAV/MP3 into EVE-compatible file.

**Figure 12 Audio converter tab**

**Input File:** The original audio file to be converted.

**Output folder:** Folder contains converted files.

**Output format:** Audio format of the output

- 8 bit signed PCM
- 8 bit u-Law
- 4 bit IMA ADPCM

**Output Sample Rate:** Same as “Output Audio Sample Rate” of the video converter. Output also contains a sample C file that shows how to playback video on EVE.
E. Generate Font

Extract characters from the OpenType or TrueType font file into FT80X / FT81X / BT81X specific font metric block as well as the raw bitmap data.

![Generate font tab](image)

**Figure 13 Generate font tab**

- **Font size:** Sets output font size.

- **Manual Kerning:** This check box lets users decide to convert each glyph as its original shape in the font file. Then each glyph must be kerned manually.

- **Escape Line Break:** BT817/8 chip support to overwrite line break. The ASCII 0x0A is interpreted as the index, other than the line break.

- **Input Font File:** Font file to be converted.

- **Output Folder:** Folder contains converted files.
1. Legacy Format

![Figure 14 Generate Font: Legacy Format]

**Bitmap Format**: Select All, or a combination of L1, L2, L4, L8

**EVE Command support**: Select command SETFONT or SETFONT2 to setup font file.

- **SETFONT**: Generate FT80X compatible font metric table, the default option.
- **SETFONT2**: Generate BT81X/FT81X compatible font metric table for Cmd_SetFont2 command. The generated files with this option are not compatible with FT80X.

**Address of Font Data(Metrics Table + Glyph)**: Address to put font data in RAM_G

**User Defined Character Set**: If not selected (default), all characters in the font will be converted. If selected, only a list of characters in a text file would be converted, characters in this list must be defined in the input font file.

**Printable ASCII Characters [32-126]**: If selected, ASCII characters which have code point from 32 to 126 will be converted.

**Output**:

- Generate the metric block file as well as L1, L2, L4, L8 format bitmap data.
- The output is one 148 bytes metric block followed by the raw bitmap data.
- This tool also generates sample C code to demonstrate the usage.
- The output data is prepared for 1 bitmap handle.
2. Extended format [BT81X]

Handle fonts with a full range of Unicode code points. This feature only supports for BT81X EVE series.

**Figure 15 Generate Font: Extended Format**

**Bitmap Format:** Can be one of ASTC/L1/L2/L4/L8. This option will decide the format of each generated glyph.

**ASTC Block Footprints:** Select one of following: Auto, 4x4, 5x4, 5x5, 6x5, 6x6, 8x5, 8x6, 8x8, 10x5, 10x6, 10x8, 10x10, 12x10, 12x12

**Compression Speed:** veryfast, fast, medium, thorough and exhaustive. Quality increases from veryfast to exhaustive while encoding speed decreases as well.

**Address Of Glyph Data:** Address to put font data in, it can be RAM_G or FLASH.

**Characters To Be Converted:** If not selected (default), all characters in the font will be converted. If selected, only a list of characters in a text file would be converted, characters in this list must be defined in the input font file. By default, it converts full range of Unicode code points. Users can specify which characters need to be converted.

**Full Unicode Character:** All Unicode characters from 0 to 65535 will be converted.

**Output:** .glyph file contains graphic data and .xfont file contains the font block, as well as the sample C code demonstrating the usage.
F. Generate Animation

Convert GIF file or a list of PNG/JPEG/BMP files into EVE compatible animation file. Animation is supported by BT81X chip and above.

![Generate Animation](image)

**Figure 16 Generate Animation**

**Input File Or Folder**: Users can select GIF file or image folder. Image folder must contain PNG/JPEG/BMP files which have the sequential name as regular expression “.*[0-9]+\.(png|jpeg|jpg|bmp)” defined. The number denotes the sequence of animated frames. For example: “001.png”, “002.png”.

**Output Folder**: Folder contains converted files.

**ASTC Block Footprints**: Select one of following: 4x4, 5x4, 5x5, 6x5, 6x6, 8x5, 8x6, 8x8, 10x5, 10x6, 10x8, 10x10, 12x10, 12x12

**Compression Speed**: veryfast, fast, medium, thorough and exhaustive. Quality increases from veryfast to exhaustive while encoding speed decreases as well.
Frame Count: Display the number of frames which will be generated.

Playback FPS: For previewing the speed of animation

Address of Animation Data: Select FLASH or RAM_G

Tile Size: Auto, Manual or Disabled

- Auto: EAB will select the best tile size
- Manual: user select Tile Width and Tile Height

**Note:** Result of Tile Width x Tile Height must be divisible by 4, due to a constraint of ASTC rendering from flash.

- Disabled: Do not apply Tile

Output:

- A C code sample to demonstrate how to display animation.
- An animation file, which is used to generate a flash file, so that animation can be rendered from flash. From BT817/8, animation can be rendered from RAM_G for faster speed.
G. Flash Utilities

Including Generate Flash Image, Detect, Program and Read Flash.

1. Generated Flash Image

Generate flash image and its map file from input files.

**Input Files**: Select files to be generated.

**EVE Chip**: Select BT815/6 or BT817/8, a corresponding flash .blob file will be used.

**Data Alignment**: Each asset is 64-byte alignment, the flash file is 256-byte alignment. Alignment will not be applied if unchecked.

**Output Folder**: Folder containing generated files.

**Output Name**: Name of generated flash file.

**Flash’s Map**: Map of files in the generated flash file, include address and size of each file.

**Output**: BIN file to write into BT81X’s flash and MAP file contains all the resources’ address and size with format: `<resource> : <address> : <size>`
2. Detect Flash

**Detect:** Detect flash information.

**Programmer Module:** It can be **FT4222** or **MPSSE**.

Make sure **FT4222** or **MPSSE** module is present on your design and all required drivers are installed. If your design has no FT4222 or MPSSE module present, you may need purchase these modules and connect it with your design properly.

**Limitation:** Only one Programmer Module can be used at a time.

**EVE Chip:** Select BT815/6 or BT817/8 based on the board.
3. Program Flash

![Program Flash](image)

**Figure 19 Program flash**

**Binary File:** Select file to write to flash chip.

**ERASE:** Erase all data in flash chip.

**UPDATE:** Write .bin into Flash, erasing if necessary. Flash is not cleared out completely, only updates the partition that's being updated.

**UPDATE&VERIFY:** Same as Update, then verifying flash data after updating done.

**PROGRAM:** Write into factory state flash chip. It is two or three times faster because it does not compare data before writing. This feature is supported for BT817/8 and above.

4. Read Flash

Read all content of flash chip into .bin file.

![Read Flash](image)

**Figure 20 Read flash**

**Output folder:** Select a folder to save output file

**Output name:** Select name for output file

5. INSTALL BLOB (for advanced user)

This feature is hidden by default. The following steps below will enable it:

1. Go to EAB installation folder
2. Open file `ui_config.json`
3. Change value of “install_blob” from `false` to `true`
4. Reopen EAB
Blob File: Select .blob file to install.

6. Sample code

```c
// The pseudo code to read flash content to a file

// Assume the flash is in detach mode and now attach it
cmd_flashattach();

// Now check if the flash is in basic mode after attaching
while (FLASH_STATUS_BASIC != rd8(REG_FLASH_STATUS));

// Get flash size
flash_size = rd32(REG_FLASH_SIZE)

// Open file to write
FILE * fp = fopen(file_name, "wb");

// Read flash content
loop while flash_size > 0
    cmd_flashread(buffer, src_flash, buffer_size);

    // Write buffer to file
    fwrite(buffer, l, buffer_size, fp);

    // Update flash address and flash size
    src_flash += buffer_size;
    flash_size -= buffer_size;
```

Figure 22 Flash Sample Code

Copy Source: Copy source code in current tab to clipboard.
H. Asset Compressor

Add: Add asset files to compress

Remove: Remove asset file from list

Output Folder: Folder to write compressed files to

Library: Select library to compress. They are zlib and zopfli.
I. Custom Touch

Compile a small program in a tiny C-like language and produces a loadable firmware image.

**Figure 24 Compile Custom Touch Firmware**

**Input File:** Set a tiny C-like language file to compile

**Output Folder:** Set a folder to save output files

**Read Me Tab:** Introduction to the Custom Touch Firmware compiler
J. Bin2C

Bin2C is a tool to convert a binary file into C array.

**Figure 25 Bin2C Converter**

**Input Files:** List of binary files to be converted

**Output Folder:** A folder which will contain C array files

**Type:** Data type of C array, one of byte/word/long

**Format:** Set number format in decimal/hexadecimal

**Item Per Line:** Number of array items in one line

**Endian:** Set endian little/big when reading binary file
K. SampleApp

SampleApp Project is located at: <Installation Folder>/Assets For Test/SampleApp
After converting EVE resources, EAB will generate a .c file containing sample code. Users can use this sample code with SampleApp project for testing.

**Example:** Convert image *Lena.png* into COMPRESSED_RGBA_ASTC_5x5_KHR format.

```c
void Load_Image(Gpu_Hal_Context_t *phost)
{
    Gpu_Hal_WaitCmdfifo_empty(phost);

    Gpu_Cmd_Dlstart(phost);
    App_WrCoCmd_Buffer(phost, CLEAR(1, 1, 1));
    App_WrCoCmd_Buffer(phost, COLOR_RGB(255, 255, 255));

    uint16_t iw = 35;
    uint16_t ih = 35;
    uint16_t format = COMPRESSED_RGBA_ASTC_5x5_KHR;

    Figure 26 Sample code: load ASTC image
```

Copy Load_Image() function to SampleApp.c file, set correct value for EVE Resources’ address, and call Load_Image() in main() function as below:

```c
int32_t main(int32_t argc, char8_t *argv[])
{
    phost = &host;
    /* Init HW Hal */
    App_Common_Init(phost);
    /* Show Logo, do calibration and display welcome screen */
    App_Common_Start(phost, NULL);
    /* Add display code here */

    Load_Image(phost);

    /* Close all the opened handles */
    Gpu_Hal_Close(phost);
    Gpu_Hal_DeInit();

    Figure 27 Sample code: call Load_Image()
```

**Note:** The SampleApp is prepared for the Bridgetek modules VM816CU50A or ME817EV which have built-in FT4222H.
V. Example

A. Image converter

Below is an example to covert a PNG file into ASTC format 4x4 and how the output file is to be utilized. Other image formats have the same usage.

Step 1: Convert PNG file into ASTC

![Image converter example on EAB](image.png)

**Figure 28 Image converter example on EAB**
Output folder:

<table>
<thead>
<tr>
<th>Name</th>
<th>Ext</th>
<th>Size</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>photo_800x480_COMPRESSED_RGB_A_.c</td>
<td>.c</td>
<td>1455</td>
<td>25/05/2020 11:10</td>
</tr>
<tr>
<td>photo_800x480_COMPRESSED_RGB_A_.json</td>
<td>.json</td>
<td>234</td>
<td>25/05/2020 11:10</td>
</tr>
<tr>
<td>photo_800x480_COMPRESSED_RGB_A_.raw</td>
<td>.raw</td>
<td>245,760</td>
<td>25/05/2020 11:10</td>
</tr>
<tr>
<td>photo_800x480_COMPRESSED_RGB_A_.raw</td>
<td>.raw</td>
<td>878,293</td>
<td>25/05/2020 11:10</td>
</tr>
<tr>
<td>photo_800x480_COMPRESSED_RGB_A_.png</td>
<td>.png</td>
<td>389,331</td>
<td>25/05/2020 11:10</td>
</tr>
</tbody>
</table>

*Figure 29 Image converter example: Output folder*

The generated `photo_800x480_COMPRESSED_RGB_A_.c` contains a function `Load_Image()` to load the generated .raw file:

**Step 2:** Call that function inside `sampleApp` MSVC project, images will be displayed on LCD:
B. Video Converter

Below is an example to covert an AVI file and how the output file is to be utilized.

**Step 1:** Convert video file into EVE-compatible file by EAB

![Video converter example on EAB](image)

**Output folder:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Ext</th>
<th>Size</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>chickens-4</td>
<td>avi</td>
<td>5,072,354</td>
<td>25/05/2020 11:04</td>
</tr>
<tr>
<td>videoplayback</td>
<td>c</td>
<td>9,736</td>
<td>03/07/2019 09:51</td>
</tr>
</tbody>
</table>

![Video converter example: Output folder](image)
The generated videoplayback.c contains a C function to load the generated .raw file:

```c
void VideoPlayback()
{
    #if USE_BT81X_FLASH
        /* Switch Flash to FULL Mode */
        Gpu_CoCmd_FlashHelper_SwitchFullMode(ghost);
        Gpu_Hal_Wr32(ghost, REG_PLAY_CONTROL, 1);
        Gpu_CoCmd_DStart(ghost);
        App_WrCoCmd_Buffer(ghost, CLEAR(1, 1, 1));
        App_WrCoCmd_Buffer(ghost, COLOR_RGB(255, 255, 255));

        Gpu_CoCmd_FlashSource(ghost, ADDR_VIDEO);
        App_WrCoCmd_Buffer(ghost, CMD_PLAYVIDEO);
        App_WrCoCmd_Buffer(ghost, OPT_FLASH | OPT_SOUND | OPT_NOTEAR);
        App_Flush_Co_Buffer(ghost);
        Gpu_Hal_WaitCmdfifo_empty(ghost);
    
    Figure 33 Video converter example: sample code

    Step 2: Call function VideoPlayback from sampleApp MSVC project, video will be played on LCD.
```

Figure 34 Video converter example: LCD screen
C. Audio Converter

Below is an example to covert a WAV file and how the output file is to be utilized.

**Step 1:** Convert audio file into EVE-compatible file

![Image of EVE Asset Builder](image.png)

**Figure 35 Audio converter example on EAB**

**Output folder:**

![Table of Files](table.png)

**Figure 36 Audio converter example: Output folder**
The generated "I Have A Dream.c" contains a C function to load the generated .raw file:

```c
void Audio_Playback() {
    Fifo_Init(&MediaFifo, MEDIA_FIFO_ADDR, MEDIA_FIFO_BUFFER, REG_MEDIAFIFO_READ, REG_MEDIAFIFO_WRITE);
    Gpu_Cmd_MediaFifo(phost, MEDIA_FIFO_ADDR, MEDIA_FIFO_BUFFER);
    printf("Mediafifo: Start address and length %d %d\n", MEDIA_FIFO_ADDR, MEDIA_FIFO_BUFFER);
    App_Flush_Co_Buffer(phost);
    Gpu_Hal_WaitCondfifo_empty(phost);
    MediaFile.pfile = fopen("path\to\"I Have A Dream.c", "rb");
    if (MediaFile.pfile != NULL) {
        fseek(MediaFile.pfile, 0, SEEK_END);
        MediaFile.FILE_SIZE = MediaFile.file_len = ftell(MediaFile.pfile);
        MediaFile.played_len = 0;
        fseek(MediaFile.pfile, 0, SEEK_SET);
        Gpu_Hal_Wr32(phost, REG_PLAYBACK_READPTR, MediaFifo.fifo_buff);
        MediaFifo.fifo_rp = 0;
        MediaFifo.fifo_wp = 0;
    }
}
```

**Figure 37 Audio converter example: Sample code**

**Step 2:** Call function Audio_Playback () from sampleApp MSVC project, audio will be played on LCD.
D. Generate Font

Below is an example to covert a True Font File and how the output file is to be utilized.

**Step 1:** Convert font file into EVE-compatible file

![Image of EVE Asset Builder interface with font conversion process]

**Figure 38 Generate font example on EAB**

**Output folder:**

![Image of EAB output folder]

**Figure 39 Generate font example: Output folder**
The generated simkai_39_ASTC.c contains a C function to load the generated .raw file:

```c
void LoadXfont()
{
    uint32_t fontAddr = RAM_G;

    // NOTE: Remember to write glyph file into BT815's flash at address 4096
    // Switch Flash to FULL Mode
    Gpu_CoCmd_FlashHelper_SwitchFullMode(phost);

    // Load xfont file into graphics RAM
    Gpu_Hal_LoadImageToMemory(phost, "path\to\simkai_39_ASTC.xfont", fontAddr, LOAD);
    Gpu_Hal_WaitCmdfifo_empty(phost);
}
```

**Figure 40 Generate font example: sample code**

**Step 2:** Call function VideoPlayback from sampleApp MSVC project, video will be played on LCD:

![Figure 41 Generate font example: LCD screen](image-url)
E. Generate Animation and Generate Flash

Below is an example to convert 10 PNG images into animation file, generate flash for the output and display animation on LCD

**Step 1:** Convert PNG images into EVE-compatible file

![Image of EVE Asset Builder software interface]

*Figure 42 Generate animation example on EAB*

**Output folder:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Ext</th>
<th>Size</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animation-John</td>
<td>anim</td>
<td>474,918</td>
<td>25/05/2020 13:36</td>
</tr>
<tr>
<td>Animation-John_anim</td>
<td>c</td>
<td>2,216</td>
<td>25/05/2020 13:36</td>
</tr>
</tbody>
</table>

*Figure 43 Generate animation example: Output folder*
The generated "Animation-JohnAnim.c" contains a C function to load the generated .raw file:

```c
void LoadAnimation(Gpu_Hal_Context_t *phost, uint8_t frame_rate)
{
    double duration = 1000.0 / frame_rate;
    double remain = 0;
    double frame_time = 0;

    // Switch Flash to FULL Mode
    Gpu_Cmd_FlashHelper_SwitchFullScreen(phost);
}
```

**Figure 44 Generate animation example: Sample code**

**Step 2:** Generate Flash file containing the animation file:

**Figure 45 Generate animation example: Generate flash**

Content of map file of the generated animation:

```plaintext
unified.blob : 0  4096
Animation-John.anim.data : 4096  1202048
Animation-John.anim.object : 1206144  320
```

**Figure 46 Generate animation example: Map file**
Step 3: Flash the Flash file into EVE’s flash:

![Program Flash Interface](image)

**Figure 47 Generate animation example: Program flash**

Step 4: Call function LoadAnimation() from sampleApp MSVC project, update address of ANIM_ADDR with content of .map file in step 2, video will be played on LCD:

![LCD Screen](image)

**Figure 48 Generate animation example: LCD screen**
## VI. Contact Information

<table>
<thead>
<tr>
<th>Head Quarters – Singapore</th>
<th>Branch Office – Taipei, Taiwan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brintek Pte Ltd</td>
<td>Brintek Pte Ltd, Taiwan Branch</td>
</tr>
<tr>
<td>178 Paya Lebar Road, #07-03</td>
<td>2 Floor, No. 516, Sec. 1, Nei Hu Road, Nei Hu</td>
</tr>
<tr>
<td>Singapore 409030</td>
<td>District</td>
</tr>
<tr>
<td>Tel: +65 6547 4827</td>
<td>Taipei 114</td>
</tr>
<tr>
<td>Fax: +65 6841 6071</td>
<td>Taiwan, R.O.C.</td>
</tr>
</tbody>
</table>

| E-mail (Sales)            | sales.apac@brtchip.com        |
| E-mail (Support)          | support.apac@brtchip.com      |

<table>
<thead>
<tr>
<th>Branch Office – Glasgow, United Kingdom</th>
<th>Branch Office – Vietnam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brintek Pte. Ltd.</td>
<td>Brintek VietNam Company Limited</td>
</tr>
<tr>
<td>Unit 1, 2 Seaward Place, Centurion Business Park</td>
<td>Lutaco Tower Building, 5th Floor, 173A Nguyen</td>
</tr>
<tr>
<td>Glasgow G41 1HH</td>
<td>Van Troi</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>Ward 11, Phu Nhuan District,</td>
</tr>
<tr>
<td>Tel: +44 (0) 141 429 2777</td>
<td>Ho Chi Minh City, Vietnam</td>
</tr>
<tr>
<td>Fax: +44 (0) 141 429 2758</td>
<td>Tel : 08 38453222</td>
</tr>
<tr>
<td></td>
<td>Fax : 08 38455222</td>
</tr>
</tbody>
</table>

| E-mail (Sales)            | sales.emea@brtchip.com      |
| E-mail (Support)          | support.emea@brtchip.com    |

| E-mail (Sales)            | sales.apac@brtchip.com      |
| E-mail (Support)          | support.apac@brtchip.com    |

<table>
<thead>
<tr>
<th>Web Site</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://brtchip.com/">http://brtchip.com/</a></td>
</tr>
</tbody>
</table>

**Distributor and Sales Representatives**

Please visit the Sales Network page of the [Bridgetek Web site](http://brtchip.com) 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 Brintek Pte Ltd (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 Brintek devices and other materials) is provided for reference only. While Brintek has taken care to assure it is accurate, this information is subject to customer confirmation, and Brintek disclaims all liability for system designs and for any applications assistance provided by Brintek. Use of Brintek 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 Brintek 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. Brintek Pte Ltd, 178 Paya Lebar Road, #07-03, Singapore 409030. Singapore Registered Company Number: 201542387H.
Appendix A – List of Figures

Figure 1 EVE Asset Builder Setup Wizard................................................................. 6
Figure 2 EVE Asset Builder Setup – Select destination folder................................. 7
Figure 3 EVE Asset Builder Setup – Ready for Installation ......................................... 7
Figure 4 EVE Asset Builder Setup – Installing in Progress ........................................ 8
Figure 5 EVE Asset Builder Setup - Finish................................................................. 8
Figure 6 Asset Building Session.............................................................................. 9
Figure 7 Session dialog .........................................................................................10
Figure 8 Select session file ....................................................................................10
Figure 9 Image converter tab ................................................................................11
Figure 10 PNG/JPEG validator tab .........................................................................12
Figure 11 Video converter tab ...............................................................................13
Figure 12 Audio converter tab ..............................................................................14
Figure 13 Generate font tab ..................................................................................15
Figure 14 Generate Font: Legacy Format .................................................................16
Figure 15 Generate Font: Extended Format ............................................................17
Figure 16 Generate Animation ..............................................................................18
Figure 17 Generate Flash .....................................................................................20
Figure 18 Detect Flash ..........................................................................................21
Figure 21 Program flash.......................................................................................22
Figure 22 Read flash.............................................................................................22
Figure 23 Enable "Install Blob" .............................................................................23
Figure 24 Flash Sample Code. ..............................................................................23
Figure 25 Asset Compressor ................................................................................24
Figure 26 Compile Custom Touch Firmware .........................................................25
Figure 27 Bin2C Converter ..................................................................................26
Figure 28 Sample code: load ASTC image ..............................................................27
Figure 29 Sample code: call Load_image() .............................................................27
Figure 30 Image converter example on EAB .........................................................28
Figure 31 Image converter example: Output folder ...............................................29
Figure 32 Image converter example: LCD screen .................................................29
Figure 33 Video converter example on EAB ..........................................................30
Figure 34 Video converter example: Output folder ................................................30
Figure 35 Video converter example: sample code ................................................31
Figure 36 Video converter example: LCD screen ....................................................31
Figure 37 Audio converter example on EAB ..........................................................32
Figure 38 Audio converter example: Output folder ...............................................32
Figure 39 Audio converter example: Sample code ...............................................33
Figure 40 Generate font example on EAB ...............................................................34
Figure 41 Generate font example: Output folder ....................................................34
Figure 42 Generate font example: sample code .....................................................35
Figure 43 Generate font example: LCD screen .......................................................35
Figure 44 Generate animation example on EAB .....................................................36
Figure 45 Generate animation example: Output folder ..........................................36
Figure 46 Generate animation example: Sample code ..........................................37
Figure 47 Generate animation example: Generate flash .......................................37
Figure 48 Generate animation example: Map file....................................................37
Figure 49 Generate animation example: Program flash ..................................................38
Figure 50 Generate animation example: LCD screen ..................................................38
# Appendix B - Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.5</td>
<td>User Guide for EVE Asset Builder 2.0.0 release</td>
<td>12-08-2020</td>
</tr>
<tr>
<td>Version 1.6</td>
<td>Updated user guide for EVE Asset Building 2.1.0</td>
<td>17-12-2020</td>
</tr>
</tbody>
</table>