AN-EJUMP

Setting Up a Raspberry Pi for Use with eJump ITE-8951 Based Boards

TABLE OF CONTENTS

1. Introduction

This application note is intended for engineers targeting at implementing a software interface on a raspberry pi for use with any of the eJump T-CON boards. It is accompanied by the following supporting files, which you can download from our website:

- C++ Demo Code for Raspberry Pi (ejump_rp_usb)
- C++ Demo Code for Raspberry Pi (ejump_rp_vcom)
- C++ Demo Code for Raspberry Pi (ejump_rp_waveform)
- C++ API Guideline by eJump
- Information about e-paper waveform modes

This application note currently supports the following eJump T-CON boards:

- EJ8951-1W, for use with e-paper display ED133UT2
- EJ8951-2W, for use with e-paper displays ED420TT3 and ED420TT1
- EJ8951-4W, for use with e-paper displays EC312TT2, ED312TT2, and ED312TT3
- EJ8951EL-1W, for use with e-paper displays EL133US1, EL097TR1, ED133UT2, ES133UT2, and ED113TC2
- EJ8951EL-1W-50, for use with e-paper displays ED113TC2 and ED113TC1
- EJ1000, for use with e-paper displays ED280TT1 and ED253TT1

The software interface for the eJump T-CON boards can be implemented using either C++ or Python. It can be used in conjunction with any image processing library. However, the demo software provided by Beck Elektronik uses OpenCV for the C++ example and Pillow for the python example.

2. Setting Up the Raspberry Pi

2.1. What Equipment You Need

- A computer with an integrated or external micro-SD card slot.
- A spare monitor that you can connect to the Raspberry Pi's HDMI video output (micro-HDMI for Raspberry Pi 4).
- A spare keyboard and mouse.
- A Raspberry Pi, at best at least version 4.
- A micro-SD card with a speed of at least 10 MB/s (class 10 or higher) and a capacity of at least 16 GB. Values between 16 GB and 64 GB should work fine.
- A micro-USB power supply for the Raspberry Pi. You can use any smartphone power supply for this, as long as it is capable of driving a current of min. two amps, with three amps being preferred.
- A USB to micro-USB cable for connecting the Raspberry Pi with the TCON board.
- Any of the supported eJump TCON boards (as listed on page 4) with API 2.0 for driving the e-paper.
- An e-paper display compatible with the chosen eJump TCON board (refer to page 4 for a list of compatible e-paper displays for each supported TCON board).

2.2. Setting Up the Operating System

- Plug the micro-SD card into your computer.
- Download and install the official “Raspberry Pi Imager” tool from the Raspberry Pi website (https://www.raspberrypi.org/software/).
- Start the “Raspberry Pi Imager” tool.
- Click on “CHOOSE OS” and select the first entry “Raspberry Pi OS (32-bit)”.
- Click on “CHOOSE STORAGE” and select the micro-SD card. Be sure to select the correct micro-SD card, as all data on the selected card will be permanently and irretrievably deleted!
- Click on “WRITE” and wait until a popup message is shown, stating that flash procedure is finished and that you can now remove the micro-SD card from the computer.
- Insert the micro-SD card into the Raspberry Pi.
- Connect the Raspberry Pi to the monitor, keyboard, and mouse.
- Connect the Raspberry Pi with the power supply. Soon, the Raspberry Pi will start-up and show some content on the screen.
- Follow the setup instructions given to you on the Raspberry Pi (default user account, language, and WLAN settings) and let it update.
- Connect the eJump TCON board to the e-paper.
- Connect the Raspberry Pi to the eJump TCON board.
- Now your Raspberry Pi is ready to be setup for the eJump TCON board.

2.3. Preparing the Raspberry Pi

- Open the console. There are several ways to do so:
  - Via shortcut CTRL + ALT + T
  - Via "Raspberry Menu" → "Accessories" → "LXTerminal"
  - Via the fourth button in the top left (the black rectangle with the < in it).
- Click anywhere in the console to make it active. Type in the following command and press ENTER afterwards to update all the local package lists:

```
sudo apt update
```

It may take a while for the command to be completed (a green text with your username will reappear, followed by ~ $).

Unlike Windows, where you download programs from their respective websites, Linux distributions provide a unified way of downloading most programs, the so-called package manager. This program – `apt` – keeps track of a huge list of available programs and where to find them. The command shown above updates this list so that newly available programs or newly available versions of already known programs can be installed via `apt` (e.g. Microsoft Teams via `sudo apt install teams`).

The command "`sudo`" executes the following command with extended permissions (like an admin account on Windows). This is necessary for installing new packages or accessing external hardware and should always be used with caution.

- Open the Chromium browser (second button in the top left, i.e. a blue globe).
- Download "*C++ Demo Code for Raspberry Pi (ejump_rp_usb)*" from our website. After being downloaded there will be a zipped folder "ejump_rp_usb.zip" inside your "Downloads" folder. Just leave it there for now.
- Open the file explorer (third button in the top left, i.e. a yellowish folder).
- Switch back to the terminal. There, type in the following command and press ENTER afterwards to unpack the folder:

```
unzip Downloads/ejump_rp_usb.zip
```

If everything succeeds, there should be a new folder "ejump_rp_usb" in the same location where "Downloads" and "Documents" reside, as shown in Figure 1.

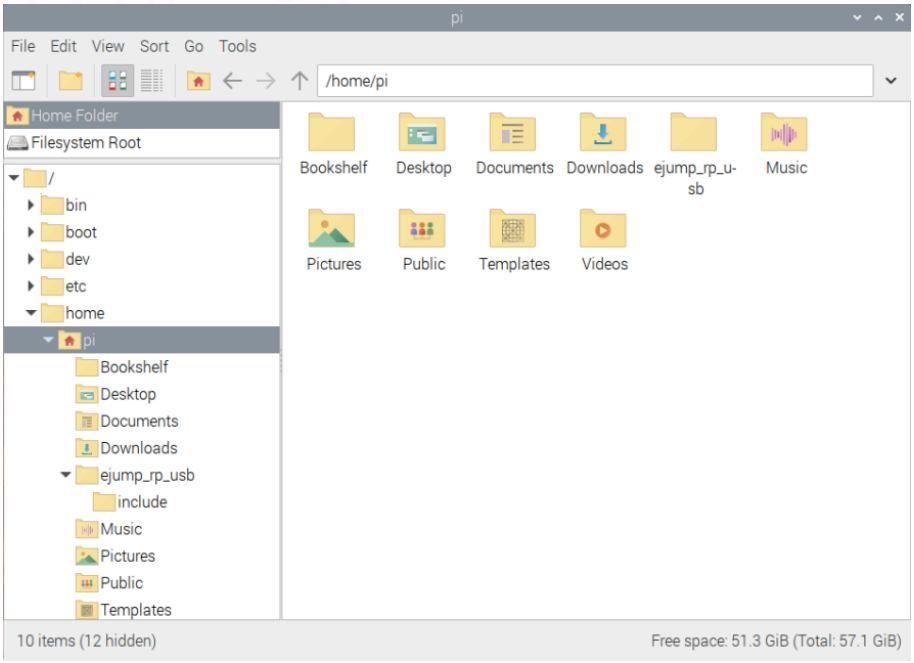

Figure 1. Structure of Home Folder After Unzipping.

- Download the current version of the eJump library “libtcon-dev_usb.so” from our website (it’s the same library for all eJump T-CON boards) and place it inside the newly created “ejump_rp_usb” folder. The content of the folder “ejump_rp_usb” should now look like this:

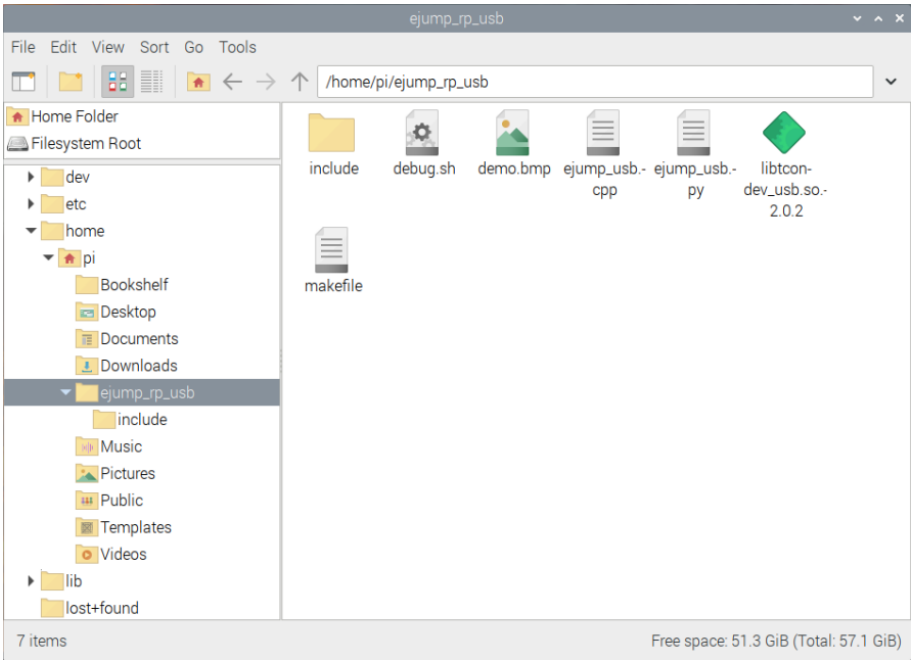

Figure 2. Structure of Example Program.

- Navigate back to the terminal. Type in the following command to switch to the previously extracted “ejump_rp_usb” folder:

```
cd ejump_rp_usb
```

With the command “cd” you can change the active directory of the terminal. The command takes a relative path to the target directory. When opening a new terminal, it will always start in the home folder of the user (the one with the “Downloads” and “Documents” folder within), as indicated by the shorthand path name “~”. Hence, by typing in “`cd ejump_rp_usb`” we switch to the folder “ejump_rp_usb” within that home folder. To show a list of all files and folders within the currently active directory, just type “`ls`”.

- After that, execute the following command to move the eJump library to a place where Linux can find it (while removing the version number from the name):

```
sudo mv libtcon-dev_usb.so.* /usr/lib/libtcon-dev_usb.so
```

Unlike Windows, Linux has quite a well-defined folder structure, i.e. specific types of files reside in special places. Executables will be in `/bin` and `/usr/bin`, config files in `/etc`, libraries (i.e. runtime dependencies of executables) in `/lib` and `/usr/lib`, etc. For more information about the Linux folder structure, see this Wikipedia article.

- Finally, the library libusb must be installed. Type in the following command:

```
sudo apt install libusb-1.0-0-dev
```

When the console asks for permission during installation, just press ENTER.

The program “libusb” is not an actual program like “Microsoft Word” or “Firefox” with a graphical user interface, but instead provides a generic interface for USB communication with external hardware, so that other programs do not have to reinvent the wheel. It is open source and used by the eJump library to communicate with the T-CON board.

3. Running the C++ Example

3.1. Installing All the Prerequisites

- The C++ example needs the image library OpenCV. Type in the following command to install it on the Raspberry Pi:

```
sudo apt install libopencv-dev
```

When the console asks for permission during installation, just press ENTER.

The open-source library “OpenCV” is like “libusb” also no ordinary program with a user interface, but instead provides algorithms and methods for manipulating arbitrary image data. Ranging from simple tasks like scaling or rotating images, it can perform more advanced tasks like pattern recognition or machine learning. Initially developed as an open-source tool by Intel, it is used by the example program to load and process the images.

- Now that OpenCV is installed, ask Linux to rescan the system for new libraries. That way, it can use these when building new applications:

```
sudo ldconfig
```

3.2. Displaying Images on the E-Paper Display

- Now that everything is ready, execute below command to compile the program:

```
make
```

If everything succeeds, there should be two new files in the “ejump_rp_usb” folder and the output of the console should look like the image in Figure 3.

- If you got some error messages, restart your system, open a console, and repeat the two commands “`cd ejump_rp_usb`” and “`make`”. If the error persists, execute the following command:

```
chmod 755 debug.sh && sudo ./debug.sh
```

In the “ejump_rp_usb” folder, there should be a zip file called “error_log.zip”. Please contact your sales representative at BECK Elektronik with your problem and attach this zip file for fast support. **Note:** The script may install the “zip” package if not already installed.

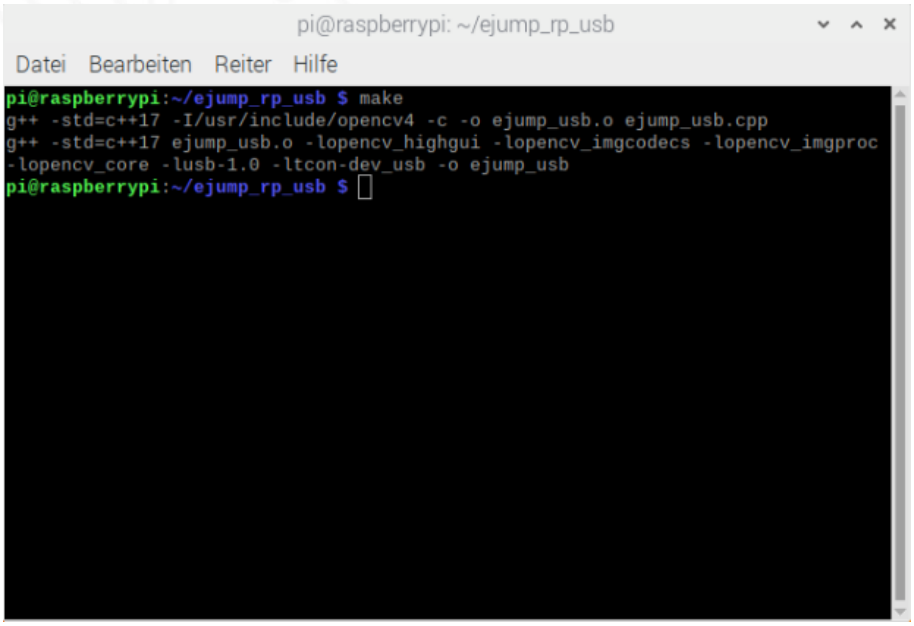

Figure 3. Command Line Output and Program Folder Structure After Running Make.

- In case everything went well, execute the following command and press ENTER afterwards to display images on the e-paper display:

```
sudo ./ejump_usb demo.bmp
```

  As depicted in Figure 4, there should be some output in the console informing you about the attached T-CON board. These values should match your e-paper display, otherwise you may have flashed the wrong firmware onto the eJump T-CON board. Next, the program will display the image “demo.bmp”.
- To show your own images, place them inside the “ejump_rp_usb” folder and run above command again with the name of your image instead of “demo.bmp”. To see a list of file formats currently supported by OpenCV, refer to this webpage.
- If you want to restart the program after reboot, open a fresh terminal and execute the following command to jump back into the example program folder:

```
cd ejump_rp_usb
```

- Then, execute the following command to start the program again:

```
sudo ./ejump_usb demo.bmp
```

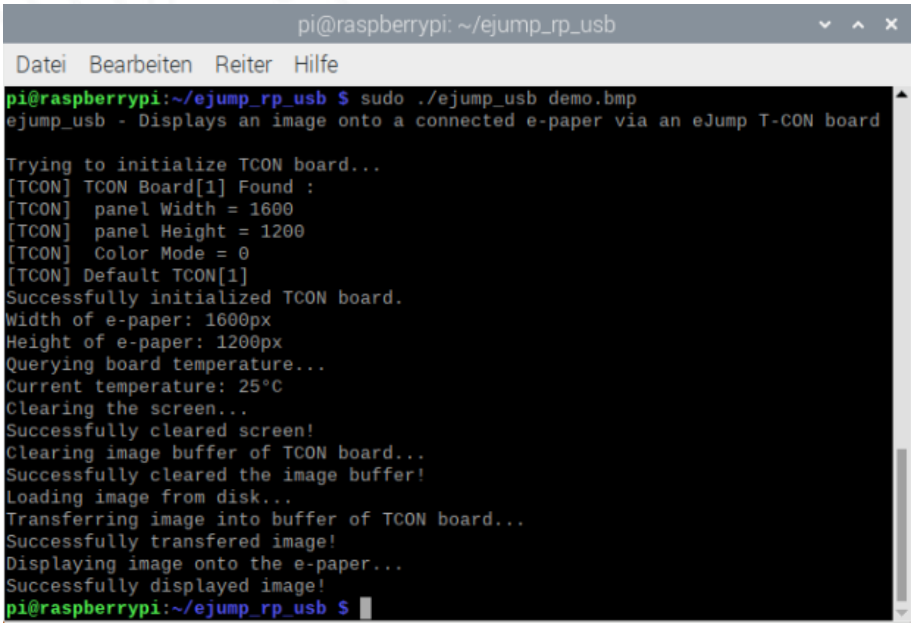

Figure 4. Command Line Output After Displaying an Image.

3.3. Updating VCOM

- To update VCOM, download the program “ejump_rp_vcom” from our website and execute the same steps as with displaying images up until “make”, i.e. open a new terminal, unzip the program folder, move to the program folder and execute make.
- Then, run the following command to update VCOM (the number is the absolute, i.e. non-negative value of your target VCOM in mV, e.g. 2100 for -2.1 V):

```
sudo ./ejump_vcom 2100
```

3.4. Updating the Waveform

- To update the waveform, download the program “ejump_rp_waveform” from our website and execute the same steps as with displaying images up until “make”, i.e. open a new terminal, unzip the program folder, navigate to the program folder and execute make.
- Then, run the following command to update the waveform (the waveform file must be in the “ejump_rp_waveform” folder):

```
sudo ./ejump_waveform waveform.wbf
```

4. Running the Python Example

4.1. Installing All the Prerequisites

- The python example needs the image library Pillow. To install it, the python package manager itself needs to be updated first. Type in the following command to do so:

```
python3 -m pip install --upgrade pip
```

- Then, execute the following command to install Pillow:

```
python3 -m pip install --upgrade Pillow
```

4.2. Displaying Images on the E-Paper Display

- Now that everything is ready, execute the following command to display images onto the e-paper display:

```
sudo python3 ejump_usb.py demo.bmp
```

  As depicted in Figure 4, there should be some output in the console informing you about the attached T-CON board. These values should match your e-paper display, otherwise you may have flashed the wrong firmware onto the eJump T-CON board. Next, the program will display the image “demo.bmp”.
- To show your own images, place them inside the “ejump_rp_usb” folder and run above command again with the name of your image instead of “demo.bmp”. To see a list of file formats currently supported by Pillow, refer to this webpage.
- If you want to restart the program after reboot, open a fresh terminal and execute the following command to jump back into the example program folder:

```
cd ejump_rp_usb
```

- Then, execute the following command to start the program again:

```
sudo python3 ejump_usb.py demo.bmp
```

4.3. Updating VCOM

- To update VCOM, download the program “ejump_rp_vcom” from our website and execute the same steps as with displaying images, i.e. open a new terminal, unzip the program folder and move to the program folder.
- Then, run the following command to update VCOM (the number is the absolute, i.e. non-negative value of your target VCOM in mV, e.g. 2100 for -2.1 V):

```
sudo python3 ejump_vcom.py 2100
```

4.4. Updating the Waveform

- To update the waveform, download the program “ejump_rp_waveform” from our website and execute the same steps as with displaying images, i.e. open a new terminal, unzip the program folder and move to the program folder.
- Then, run the following command to update the waveform (the waveform file must be in the same folder as the executable):

```
sudo python3 ejump_waveform.py waveform.wbf
```

4.5. Using the Reference Guide with Python

- The reference guide in section 5 of this document is for C++. However, if being imported to Python by means of “ctypes”, all the API functions will be accessible from within Python with their C++ syntax as members of the object returned by ctypes (e.g. in the example below as members of `tcon_lib`):

```
tcon_lib = ctypes.cdll.LoadLibrary("/usr/lib/libtcon-dev_usb.so")
```

- Occurrences of integer types like **`short`**, **`uint8_t`**, and **`uint32_t`** will be replaced by native Python integers.
- Pointers to **`uint8_t`** will be replaced by Python arrays of type **`ctypes.c_uint8`**
- Strings (**`char*`**) may be passed directly as native Python strings
- Return codes like `LIBTCON_SUCCESS` are currently not accessible in Python and must be replaced by their respective values, e.g. 0 for `LIBTCON_SUCCESS`. Refer to section 5.6 for a description of return codes and their corresponding values.

5. API Reference

5.1. Initializing The T-CON Board

5.1.1. short tcon_init()

Signature

```
short tcon_init(
      );
```

Description
Initializes the T-CON boards.

Returns
If initialization was successful, returns the number of T-CON boards available through USB. Otherwise (return value is negative), returns one of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NOT_FOUND`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_TIMEOUT`
- `LIBTCON_ERROR_PIPE`

5.2. Query Board Information

5.2.1. short tcon_get_panel_width()

Signature

```
short tcon_get_panel_width(
          short board_id
      );
```

Description

Gets the width of the panel.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

Returns either the width of the panel or any of the following error codes:

- `LIBTCON_ERROR_NO_DEVICE`

5.2.2. short tcon_get_panel_height()

Signature

```
short tcon_get_panel_height(
          short board_id
      );
```

Description

Gets the height of the panel.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

Returns either the height of the panel or any of the following error codes:

- `LIBTCON_ERROR_NO_DEVICE`

5.2.3. short tcon_get_temp()

Signature

```
short tcon_get_temp(
          short board_id
      );
```

Description

Gets the temperature of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

Returns either the temperature of the T-CON board or any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_TIMEOUT`
- `LIBTCON_ERROR_PIPE`

5.2.4. short tcon_get_vcom()

Signature

```
short tcon_get_vcom(
          short board_id
      );
```

Description

Gets the absolute (i.e. non-negative) value of VCOM in mV of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

Returns either the absolute (i.e. non-negative) value of VCOM in mV or any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_TIMEOUT`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

T-CON boards EJ8951-2W and EJ8951-4W do not support this function. On attempt of using this function on them, they will instead return `LIBTCON_ERROR_NOT_SUPPORTED`.

5.2.5. uint32_t tcon_get_fw_version()

Signature

```
uint32_t tcon_get_fw_version(
            short board_id
      );
```

Description

Gets the firmware version of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

Returns either the firmware version or any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`

5.3. Transmitting Images

5.3.1. short tcon_ld_img()

Signature

```
short tcon_ld_img(
            short board_id,
            uint8_t *img_data,
            short width,
            short height,
            short startx,
            short starty
      );
```

Description

Transmits image data to the buffer of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`img_data`	8 bpp image data (0x00 black, 0xF0 white, lower nibble is always zero).
`width`	Width of the image.
`height`	Height of the image.
`startx`	Starting X position on the screen.
`starty`	Starting Y position on the screen.

Returns

If image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function only supports monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the following two conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.3.2. short tcon_ld_img_rgb()

Signature

```
short tcon_ld_img_rgb(
          short board_id,
          uint8_t *img_data_r,
          uint8_t *img_data_g,
          uint8_t *img_data_b,
          short width,
          short height,
          short startx,
          short starty
      );
```

Description

Transmits image data to the buffer of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`img_data_r`	8 bpp red image data.
`img_data_g`	8 bpp green image data.
`img_data_b`	8 bpp blue image data.
`width`	Width of the image.
`height`	Height of the image.
`startx`	Starting X position on the screen.
`starty`	Starting Y position on the screen.

Returns

If image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function does not support monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.3.3. short tcon_fill_img()

Signature

```
short tcon_fill_img(
          short board_id,
          uint8_t gl,
          short width,
          short height,
          short startx,
          short starty
      );
```

Description

Fills the given area of the T-CON buffer with a specific gray level.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`gl`	8 bpp gray level (0x00 black, 0xF0 white, lower nibble is always zero).
`width`	Width of the image.
`height`	Height of the image.
`startx`	Starting X position on the screen.
`starty`	Starting Y position on the screen.

Returns

If gray image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function only supports monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the following two conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.3.4. short tcon_fill_img_rgb()

Signature

```
short tcon_fill_img(
          short board_id,
          uint8_t gl_r,
          uint8_t gl_g,
          uint8_t gl_b,
          short width,
          short height,
          short startx,
          short starty
      );
```

Description

Fills the given area of the T-CON buffer with the given color.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`gl_r`	8 bpp gray level.
`gl_g`	8 bpp green level.
`gl_b`	8 bpp blue level.
`width`	Width of the image.
`height`	Height of the image.
`startx`	Starting X position on the screen.
`starty`	Starting Y position on the screen.

Returns

If gray image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function does not support monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.3.5. short tcon_fill_img_all()

Signature

```
short tcon_fill_img_all(
          short board_id,
          uint8_t gl
      );
```

Description
Fills the entire buffer of the T-CON board with the given gray level.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`gl`	8 bpp gray level (0x00 black, 0xF0 white, lower nibble is always zero).

Returns
If gray image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note
This function only supports monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the following two conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.3.6. short tcon_fill_img_all_rgb()

Signature

```
short tcon_fill_img_all_rgb(
          short board_id,
          uint8_t gl_r,
          uint8_t gl_g,
          uint8_t gl_b
      );
```

Description

Fills the entire buffer of the T-CON board with the given color.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`gl_r`	8 bpp red level.
`gl_g`	8 bpp green level.
`gl_b`	8 bpp blue level.

Returns

If color image was transferred successfully, function returns `LIBTCON_SUCCESS`. Otherwise, one of the following error codes is returned:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function does not support monochrome panels. For all other panels, return value will be `LIBTCON_ERROR_NOT_SUPPORTED`. Also note that the conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.4. Displaying Images

5.4.1. short tcon_dpy_img()

Signature

```
short tcon_dpy_img(
            short board_id,
            short wf_mode,
            short short width,
            short height,
            short startx,
            short starty
      );
```

Description

Displays the specific are of the T-CON image buffer on the e-paper.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`wf_mode`	E-paper waveform mode to use for displaying.
`width`	Width of the image.
`height`	Height of the image.
`startx`	Starting X position on the screen.
`board_id`	Starting Y position on the screen.

Returns

If image was displayed successfully, this function returns `LIBTCON_SUCCESS`. Otherwise, the return value will be any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`

Note

Note that the conditions $0 \leq$ `width`, `startx` $\leq$ panel width and $0 \leq$ `height`, `starty` $\leq$ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.4.2. short tcon_dpy_img_all()

Signature

```
short tcon_dpy_img_all(
          short board_id,
          short wf_mode
      );
```

Description

Displays the entire T-CON image buffer on the e-paper.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`wf_mode`	E-paper waveform mode to use for displaying.

Returns

If image was displayed successfully, this function returns `LIBTCON_SUCCESS`. Otherwise, the return value will be any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`

Note

Note that the conditions 0 ≤ `width`, `startx` ≤ panel width and 0 ≤ `height`, `starty` ≤ panel height must be satisfied, otherwise function will return `LIBTCON_ERROR_INVALID_PARAM`.

5.4.3. short tcon_clr_scr()

Signature

```
short tcon_clr_scr(
          short board_id
      );
```

Description

Clears the entire screen of the e-paper display.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

If screen was cleared successfully, this function returns `LIBTCON_SUCCESS`. Otherwise, this function returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_PIPE`

5.5. Controlling the T-Con Board

5.5.1. short tcon_set_vcom()

Signature

```
short tcon_set_vcom(
          short board_id,
          short vcom_value
      );
```

Description

Sets VCOM to the given value.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`vcom_value`	Absolute (i.e. non-negative) value of VCOM in mV.

Returns

If VCOM value was set successfully, return value will be `LIBTCON_SUCCESS`. Otherwise, this function returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_TIMEOUT`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

Note

T-CON boards EJ8951-2W and EJ8951-4W do not support this function. On attempt of using this function on them, they will instead return `LIBTCON_ERROR_NOT_SUPPORTED`.

5.5.2. short tcon_set_id()

Signature

```
short tcon_set_id(
          short board_id,
          short id
      );
```

Description
Sets the ID of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`id`	ID value to be set.

Returns
If ID was set successfully, returns `LIBTCON_SUCCESS`. Otherwise, this functions returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_TIMEOUT`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note
This function is only supported for T-CON boards EJ8951-2W and EJ8951-4W. For all other displays, this function will return `LIBTCON_ERROR_NOT_SUPPORTED`.

5.5.3. short tcon_update_fw()

signature

```
short tcon_update_fw(
          short board_id,
          char *filename
      );
```

Description
Updates the firmware of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`filename`	Path to the firmware file.

Returns
If firmware was updated successfully, returns `LIBTCON_SUCCESS`. Otherwise, returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_INVALID_PARAM`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`

5.5.4. short tcon_update_wf()

Signature

```
short tcon_update_wf(
          short board_id,
          char *filename
      );
```

Description
Updates the waveform of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.
`filename`	Path to the waveform file.

Returns
If firmware was updated successfully, returns `LIBTCON_SUCCESS`. Otherwise, returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`

5.5.5. short tcon_soft_reset()

Signature

```
short tcon_soft_reset(
          short board_id
      );
```

Description

Initiates a soft-reset of the T-CON board.

Parameters

`board_id`	ID of target T-CON board. If -1 is passed, first available T-CON board will be used.

Returns

If soft-reset was performed successfully, returns `LIBTCON_SUCCESS`. Otherwise, returns any of the following error codes:

- `LIBTCON_ERROR_IO`
- `LIBTCON_ERROR_ACCESS`
- `LIBTCON_ERROR_NO_DEVICE`
- `LIBTCON_ERROR_BUSY`
- `LIBTCON_ERROR_OVERFLOW`
- `LIBTCON_ERROR_PIPE`
- `LIBTCON_ERROR_NOT_SUPPORTED`

Note

This function is not supported by T-CON board EJ1000. It will return `LIBTCON_ERROR_NOT_SUPPORTED` when trying to use it with this board.

5.6. Description of error codes

The following list gives an overview over all the existing error codes and their meaning:

`LIBTCON_SUCCESS`	`0`	Function execution was successful
`LIBTCON_ERROR_IO`	`-1`	Input / Output error occurred
`LIBTCON_ERROR_INVALID_PARAM`	`-2`	Function call contained invalid parameter values. Refer to the respective function's documentation for more detailed information about possible value constraints for the individual parameters.
`LIBTCON_ERROR_ACCESS`	`-3`	Access denied (insufficient permission). Try to run the program with administrator privileges to solve this issue.
`LIBTCON_ERROR_NO_DEVICE`	`-4`	No such device found.
`LIBTCON_ERROR_NOT_FOUND`	`-5`	Entity not found.
`LIBTCON_ERROR_BUSY`	`-6`	Target device is currently busy. Try again later.
`LIBTCON_ERROR_TIMEOUT`	`-7`	Operation timed out.
`LIBTCON_ERROR_OVERFLOW`	`-8`	Overflow occurred.
`LIBTCON_ERROR_PIPE`	`-9`	Pipe error occurred.
`LIBTCON_ERROR_INTERRUPTED`	`-10`	System call interrupted.
`LIBTCON_ERROR_NO_MEM`	`-11`	Insufficient memory.
`LIBTCON_ERROR_NOT_SUPPORTED`	`-12`	Operation is not supported or unimplemented on the target T-CON board. Refer to the respective function's documentation for more detailed information about supported boards.
`LIBTCON_ERROR_OTHER`	`-99`	Other error.