AdafruitTrellisM4 Library Documentation

Release 1.0

Scott Shawcroft

Oct 20, 2019

Contents

This high level library provides objects that represent Trellis M4 hardware.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the Adafruit library and driver bundle.

1.1 Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install adafruit-circuitpython-trellism4
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-trellism4
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-trellism4
```

CHAPTER 2

Usage Example

This example prints out the coordinates of a button each time it is pressed and released:

```
import time
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

current_press = set()
while True:
    pressed = set(trellis.pressed_keys)
    for press in pressed - current_press:
        print("Pressed:", press)
    for release in current_press - pressed:
        print("Released:", release)
    time.sleep(0.08)
    current_press = pressed
```

CHAPTER 3

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

4.1 Zip release files

To build this library locally you'll need to install the circuitpython-build-tools package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-trellism4 --
↪library_location .
```

4.2 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the index.html in your browser to view them. It will also (due to -W) error out on any warning like Travis will. This is a good way to locally verify it will pass.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/trellism4_simpletest.py

```
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

while True:
    pressed = trellis.pressed_keys
    if pressed:
        print("Pressed:", pressed)
```

5.2 adafruit_trellism4

CircuitPython library for the Trellis M4 Express.

- Author(s): Scott Shawcroft, Kattni Rembor

5.2.1 Implementation Notes

Hardware:

Add link to Trellis M4 Express when product is released.

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class adafruit_trellism4.**TrellisM4Express**(*rotation=0*)

Represents a single Trellis M4 Express. Do not use more than one at a time.

Parameters **rotation** – Allows for rotating the Trellis M4 Express in 90 degree increments to different positions and utilising the grid from that position. Supports `0`, `90`, `180`, and `270`. `0` degrees is when the USB facing away from you. Default is 0.

```
import time
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

current_press = set()
while True:
    pressed = set(trellis.pressed_keys)
    for press in pressed - current_press:
        print("Pressed:", press)
    for release in current_press - pressed:
        print("Released:", release)
    time.sleep(0.08)
    current_press = pressed
```

pixels = None

Sequence like object representing the 32 NeoPixels on the Trellis M4 Express, Provides a two dimensional representation of the NeoPixel grid.

This example lights up the first pixel green:

```
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

trellis.pixels[0, 0] = (0, 255, 0)
```

Options for `pixels`:

`pixels.fill`: Colors all the pixels a given color. Provide an (R, G, B) color tuple (such as (255, 0, 0) for red), or a hex color value (such as 0xff0000 for red).

This example colors all pixels red:

```
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

trellis.pixels.fill((255, 0, 0))
```

`pixels.width` and `pixels.height`: The width and height of the grid. When `rotation` is 0, `width` is 8 and `height` is 4.

This example colors all pixels blue:

```
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

for x in range(trellis.pixels.width):
    for y in range(trellis.pixels.height):
        trellis.pixels[x, y] = (0, 0, 255)
```

`pixels.brightness`: The overall brightness of the pixel. Must be a number between 0 and 1, where the number represents a percentage between 0 and 100, i.e. `0.3` is 30%.

This example sets the brightness to `0.3` and turns all the LEDs red:

```
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

trellis.pixels.brightness = 0.3

trellis.pixels.fill((255, 0, 0))
```

pressed_keys

A list of tuples of currently pressed button coordinates.

```
import time
import adafruit_trellism4

trellis = adafruit_trellism4.TrellisM4Express()

current_press = set()
while True:
    pressed = set(trellis.pressed_keys)
    for press in pressed - current_press:
        print("Pressed:", press)
    for release in current_press - pressed:
        print("Released:", release)
    time.sleep(0.08)
    current_press = pressed
```

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index