AdafruitRGB$_{D}isplayLibraryDocumentation$

Release 1.0

Michale McWethy

Oct 09, 2019

Contents

1 Dependencies **3**

2 Usage Example **5**

3 Contributing **7**

4 Building locally **9**
4.1 Sphinx documentation 9

5 Table of Contents **11**
5.1 Simple test 11
5.2 `adafruit_rgb_display.rgb` 13
5.3 `adafruit_rgb_display.hx8353` 14
5.4 `adafruit_rgb_display.ili9341` 14
5.5 `adafruit_rgb_display.s6d02a1` 15
5.6 `adafruit_rgb_display.ssd1331` 15
5.7 `adafruit_rgb_display.ssd1351` 16
5.8 `adafruit_rgb_display.st7735` 16

6 Indices and tables **17**

Python Module Index **19**

Index **21**

Port of display drivers from https://github.com/adafruit/micropython-adafruit-rgb-display to Adafruit CircuitPython for use on Adafruit's SAMD21-based and other CircuitPython boards.

Note: This driver currently won't work on micropython.org firmware, instead you want the micropython-adafruit-rgb-display driver linked above!

This CircuitPython driver currently supports displays that use the following display-driver chips: HX8353, HX8357, ILI9341, S6D02A1, ST7789, SSD1331, SSD1351, and ST7735.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device

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

CHAPTER 2

Usage Example

```
import busio
import digitalio
from board import SCK, MOSI, MISO, D2, D3

from adafruit_rgb_display import color565
import adafruit_rgb_display.ili9341 as ili9341


# Configuration for CS and DC pins:
CS_PIN = D2
DC_PIN = D3

# Setup SPI bus using hardware SPI:
spi = busio.SPI(clock=SCK, MOSI=MOSI, MISO=MISO)

# Create the ILI9341 display:
display = ili9341.ILI9341(spi, cs=digitalio.DigitalInOut(CS_PIN),
                          dc=digitalio.DigitalInOut(DC_PIN))

# Main loop:
while True:
    # Clear the display
    display.fill(0)
    # Draw a red pixel in the center.
    display.pixel(120, 160, color565(255, 0, 0))
    # Pause 2 seconds.
    time.sleep(2)
    # Clear the screen blue.
    display.fill(color565(0, 0, 255))
    # Pause 2 seconds.
    time.sleep(2)
```

CHAPTER 3

Contributing

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

CHAPTER 4

Building locally

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-rgb_display --
↪library_location .
```

4.1 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/rgb_display_simpletest.py

```
# Quick test of TFT FeatherWing (ST7789) with Feather M0 or M4
# This will work even on a device running displayio
# Will fill the TFT black and put a red pixel in the center, wait 2 seconds,
# then fill the screen blue (with no pixel), wait 2 seconds, and repeat.
import time
import random
import digitalio
import board
import displayio

from adafruit_rgb_display.rgb import color565
import adafruit_rgb_display.st7789 as st7789

displayio.release_displays()

# Configuratoin for CS and DC pins (these are FeatherWing defaults on M0/M4):
cs_pin = digitalio.DigitalInOut(board.D5)
dc_pin = digitalio.DigitalInOut(board.D6)
reset_pin = digitalio.DigitalInOut(board.D9)

# Config for display baudrate (default max is 24mhz):
BAUDRATE = 24000000

# Setup SPI bus using hardware SPI:
spi = board.SPI()

# Create the ST7789 display:
```

(continues on next page)

(continued from previous page)

```
display = st7789.ST7789(spi, cs=cs_pin, dc=dc_pin, rst=reset_pin, baudrate=BAUDRATE)

# Main loop:
while True:
    # Fill the screen red, green, blue, then black:
    for color in ((255, 0, 0), (0, 255, 0), (0, 0, 255)):
        display.fill(color565(color))
    # Clear the display
    display.fill(0)
    # Draw a red pixel in the center.
    display.pixel(display.width//2, display.height//2, color565(255, 0, 0))
    # Pause 2 seconds.
    time.sleep(2)
    # Clear the screen a random color
    display.fill(color565(random.randint(0, 255),
                          random.randint(0, 255),
                          random.randint(0, 255)))
    # Pause 2 seconds.
    time.sleep(2)
```

Listing 2: examples/rgb_display_ili9341test.py

```
# Quick test of TFT FeatherWing (ILI9341) with Feather M0 or M4
# Will fill the TFT black and put a red pixel in the center, wait 2 seconds,
# then fill the screen blue (with no pixel), wait 2 seconds, and repeat.
import time
import random
import busio
import digitalio
import board

from adafruit_rgb_display.rgb import color565
import adafruit_rgb_display.ili9341 as ili9341


# Configuratoin for CS and DC pins (these are FeatherWing defaults on M0/M4):
cs_pin = digitalio.DigitalInOut(board.D9)
dc_pin = digitalio.DigitalInOut(board.D10)

# Config for display baudrate (default max is 24mhz):
BAUDRATE = 24000000

# Setup SPI bus using hardware SPI:
spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)

# Create the ILI9341 display:
display = ili9341.ILI9341(spi, cs=cs_pin, dc=dc_pin, baudrate=BAUDRATE)

# Main loop:
while True:
    # Fill the screen red, green, blue, then black:
    for color in ((255, 0, 0), (0, 255, 0), (0, 0, 255)):
        display.fill(color565(color))
    # Clear the display
    display.fill(0)
    # Draw a red pixel in the center.
```

(continues on next page)

(continued from previous page)

```
    display.pixel(display.width//2, display.height//2, color565(255, 0, 0))
    # Pause 2 seconds.
    time.sleep(2)
    # Clear the screen a random color
    display.fill(color565(random.randint(0, 255),
                          random.randint(0, 255),
                          random.randint(0, 255)))
    # Pause 2 seconds.
    time.sleep(2)
```

5.2 adafruit_rgb_display.rgb

Base class for all RGB Display devices

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.rgb.**Display**(*width*, *height*)

Base class for all RGB display devices :param width: number of pixels wide :param height: number of pixels high

fill(*color=0*)

Fill the whole display with the specified color.

fill_rectangle(*x*, *y*, *width*, *height*, *color*)

Draw a rectangle at specified position with specified width and height, and fill it with the specified color.

hline(*x*, *y*, *width*, *color*)

Draw a horizontal line.

image(*img*, *rotation=0*)

Set buffer to value of Python Imaging Library image. The image should be in 1 bit mode and a size equal to the display size.

init()

Run the initialization commands.

pixel(*x*, *y*, *color=None*)

Read or write a pixel at a given position.

vline(*x*, *y*, *height*, *color*)

Draw a vertical line.

class adafruit_rgb_display.rgb.**DisplaySPI**(*spi*, *dc*, *cs*, *rst=None*, *width=1*, *height=1*, *baudrate=12000000*, *polarity=0*, *phase=0*, *, *x_offset=0*, *y_offset=0*)

Base class for SPI type devices

read(*command=None*, *count=0*)

SPI read from device with optional command

reset()

Reset the device

write(*command=None*, *data=None*)

SPI write to the device: commands and data

class adafruit_rgb_display.rgb.**DummyPin**

Can be used in place of a `DigitalInOut()` when you don't want to skip it.

deinit()
Dummy DigitalInOut deinit

direction
Dummy direction DigitalInOut property

pull
Dummy pull DigitalInOut property

switch_to_input(**args*, ***kwargs*)
Dummy switch_to_input method

switch_to_output(**args*, ***kwargs*)
Dummy switch_to_output method

value
Dummy value DigitalInOut property

adafruit_rgb_display.rgb.**color565**(*r*, *g=0*, *b=0*)
Convert red, green and blue values (0-255) into a 16-bit 565 encoding. As a convenience this is also available in the parent adafruit_rgb_display package namespace.

5.3 adafruit_rgb_display.hx8353

A simple driver for the HX8353-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.hx8353.**HX8353**(*spi*, *dc*, *cs*, *rst=None*, *width=128*, *height=128*)
A simple driver for the HX8353-based displays.

```
>>> import busio
>>> import digitalio
>>> import board
>>> from adafruit_rgb_display import color565
>>> import adafruit_rgb_display.hx8353 as hx8353
>>> spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
>>> display = hx8353.HX8383(spi, cs=digitalio.DigitalInOut(board.GPIO0),
...     dc=digitalio.DigitalInOut(board.GPIO15))
>>> display.fill(0x7521)
>>> display.pixel(64, 64, 0)
```

5.4 adafruit_rgb_display.ili9341

A simple driver for the ILI9341/ILI9340-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.ili9341.**ILI9341**(*spi*, *dc*, *cs*, *rst=None*, *width=240*, *height=320*, *baudrate=16000000*, *polarity=0*, *phase=0*)
A simple driver for the ILI9341/ILI9340-based displays.

```
>>> import busio
>>> import digitalio
>>> import board
>>> from adafruit_rgb_display import color565
```

(continues on next page)

(continued from previous page)

```
>>> import adafruit_rgb_display.ili9341 as ili9341
>>> spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
>>> display = ili9341.ILI9341(spi, cs=digitalio.DigitalInOut(board.GPIO0),
...     dc=digitalio.DigitalInOut(board.GPIO15))
>>> display.fill(color565(0xff, 0x11, 0x22))
>>> display.pixel(120, 160, 0)
```

scroll (*dy=None*)
Scroll the display by delta y

5.5 adafruit_rgb_display.s6d02a1

A simple driver for the S6D02A1-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.s6d02a1.**S6D02A1** (*spi, dc, cs, rst=None, width=128, height=160*)
A simple driver for the S6D02A1-based displays.

```
>>> import busio
>>> import digitalio
>>> import board
>>> from adafruit_rgb_display import color565
>>> import adafruit_rgb_display.s6d02a1 as s6d02a1
>>> spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
>>> display = s6d02a1.S6D02A1(spi, cs=digitalio.DigitalInOut(board.GPIO0),
...     dc=digitalio.DigitalInOut(board.GPIO15), rst=digitalio.DigitalInOut(board.
↪GPIO16))
>>> display.fill(0x7521)
>>> display.pixel(64, 64, 0)
```

5.6 adafruit_rgb_display.ssd1331

A simple driver for the SSD1331-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.ssd1331.**SSD1331** (*spi, dc, cs, rst=None, width=96, height=64*)
A simple driver for the SSD1331-based displays.

```
import busio
import digitalio
import board
from adafruit_rgb_display import color565
import adafruit_rgb_display.ssd1331 as ssd1331
spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
display = ssd1331.SSD1331(spi, cs=digitalio.DigitalInOut(board.GPIO0),
                          dc=digitalio.DigitalInOut(board.GPIO15),
                          rst=digitalio.DigitalInOut(board.GPIO16))

display.fill(0x7521)
display.pixel(32, 32, 0)
```

write(*command=None*, *data=None*)
write procedure specific to SSD1331

5.7 adafruit_rgb_display.ssd1351

A simple driver for the SSD1351-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.ssd1351.**SSD1351**(*spi*, *dc*, *cs*, *rst=None*, *width=128*, *height=128*)
A simple driver for the SSD1351-based displays.

```
>>> import busio
>>> import digitalio
>>> import board
>>> from adafruit_rgb_display import color565
>>> import adafruit_rgb_display.ssd1351 as ssd1351
>>> spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
>>> display = ssd1351.SSD1351(spi, cs=digitalio.DigitalInOut(board.GPIO0),
...     dc=digitalio.DigitalInOut(board.GPIO15), rst=digitalio.DigitalInOut(board.
↪GPIO16))
>>> display.fill(0x7521)
>>> display.pixel(32, 32, 0)
```

5.8 adafruit_rgb_display.st7735

A simple driver for the ST7735-based displays.

- Author(s): Radomir Dopieralski, Michael McWethy

class adafruit_rgb_display.st7735.**ST7735**(*spi*, *dc*, *cs*, *rst=None*, *width=128*, *height=128*)
A simple driver for the ST7735-based displays.

```
>>> import busio
>>> import digitalio
>>> import board
>>> from adafruit_rgb_display import color565
>>> import adafruit_rgb_display.st7735 as st7735
>>> spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)
>>> display = st7735.ST7735(spi, cs=digitalio.DigitalInOut(board.GPIO0),
...     dc=digitalio.DigitalInOut(board.GPIO15), rst=digitalio.DigitalInOut(board.
↪GPIO16))
>>> display.fill(0x7521)
>>> display.pixel(64, 64, 0)
```

class adafruit_rgb_display.st7735.**ST7735R**(*spi*, *dc*, *cs*, *rst=None*, *width=128*, *height=160*)
A simple driver for the ST7735R-based displays.

init()
Run the initialization commands.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

adafruit_rgb_display.hx8353, 14
adafruit_rgb_display.ili9341, 14
adafruit_rgb_display.rgb, 13
adafruit_rgb_display.s6d02a1, 15
adafruit_rgb_display.ssd1331, 15
adafruit_rgb_display.ssd1351, 16
adafruit_rgb_display.st7735, 16

Index

A

adafruit_rgb_display.hx8353 (*module*), 14
adafruit_rgb_display.ili9341 (*module*), 14
adafruit_rgb_display.rgb (*module*), 13
adafruit_rgb_display.s6d02a1 (*module*), 15
adafruit_rgb_display.ssd1331 (*module*), 15
adafruit_rgb_display.ssd1351 (*module*), 16
adafruit_rgb_display.st7735 (*module*), 16

C

color565() (*in module adafruit_rgb_display.rgb*), 14

D

deinit() (*adafruit_rgb_display.rgb.DummyPin method*), 13
direction (*adafruit_rgb_display.rgb.DummyPin attribute*), 14
Display (*class in adafruit_rgb_display.rgb*), 13
DisplaySPI (*class in adafruit_rgb_display.rgb*), 13
DummyPin (*class in adafruit_rgb_display.rgb*), 13

F

fill() (*adafruit_rgb_display.rgb.Display method*), 13
fill_rectangle() (*adafruit_rgb_display.rgb.Display method*), 13

H

hline() (*adafruit_rgb_display.rgb.Display method*), 13
HX8353 (*class in adafruit_rgb_display.hx8353*), 14

I

ILI9341 (*class in adafruit_rgb_display.ili9341*), 14
image() (*adafruit_rgb_display.rgb.Display method*), 13
init() (*adafruit_rgb_display.rgb.Display method*), 13
init() (*adafruit_rgb_display.st7735.ST7735R method*), 16

P

pixel() (*adafruit_rgb_display.rgb.Display method*), 13
pull (*adafruit_rgb_display.rgb.DummyPin attribute*), 14

R

read() (*adafruit_rgb_display.rgb.DisplaySPI method*), 13
reset() (*adafruit_rgb_display.rgb.DisplaySPI method*), 13

S

S6D02A1 (*class in adafruit_rgb_display.s6d02a1*), 15
scroll() (*adafruit_rgb_display.ili9341.ILI9341 method*), 15
SSD1331 (*class in adafruit_rgb_display.ssd1331*), 15
SSD1351 (*class in adafruit_rgb_display.ssd1351*), 16
ST7735 (*class in adafruit_rgb_display.st7735*), 16
ST7735R (*class in adafruit_rgb_display.st7735*), 16
switch_to_input() (*adafruit_rgb_display.rgb.DummyPin method*), 14
switch_to_output() (*adafruit_rgb_display.rgb.DummyPin method*), 14

V

value (*adafruit_rgb_display.rgb.DummyPin attribute*), 14
vline() (*adafruit_rgb_display.rgb.Display method*), 13

W

write() (*adafruit_rgb_display.rgb.DisplaySPI method*), 13
write() (*adafruit_rgb_display.ssd1331.SSD1331 method*), 15