AdafruitRA8875 Library Documentation

Release 1.0

Melissa LeBlanc-Williams

Jan 14, 2020

Contents

This is a full featured CircuitPython Library for the RA8875 that included all of the hardware accelerated drawing functions as the original Arduino library. A lot of the functionality has been streamlined with a focus on ease of use that is still flexible enough to make full use of the hardware. For instace, Graphics and Text mode switching is now automatic and handled in the background.

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

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-ra8875
```

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

```
sudo pip3 install adafruit-circuitpython-ra8875
```

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-ra8875
```

CHAPTER 3

Usage Example

See examples/ra8875_simpletest.py and examples/ra8875_bmptest.py for examples of the module's usage. When running the bmptest, be sure to upload the blinka.bmp image to the root folder as well.

CHAPTER 4

Contributing

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

CHAPTER 5

Documentation

For information on building library documentation, please check out this guide.

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/ra8875_simpletest.py

```
# Quick test of RA8875 with Feather M4
import time
import busio
import digitalio
import board

import adafruit_ra8875.ra8875 as ra8875
from adafruit_ra8875.ra8875 import color565

BLACK = color565(0, 0, 0)
RED = color565(255, 0, 0)
BLUE = color565(0, 255, 0)
GREEN = color565(0, 0, 255)
YELLOW = color565(255, 255, 0)
CYAN = color565(0, 255, 255)
MAGENTA = color565(255, 0, 255)
WHITE = color565(255, 255, 255)

# Configuration for CS and RST pins:
cs_pin = digitalio.DigitalInOut(board.D9)
rst_pin = digitalio.DigitalInOut(board.D10)
int_pin = digitalio.DigitalInOut(board.D11)

# Config for display baudrate (default max is 6mhz):
BAUDRATE = 6000000

# Setup SPI bus using hardware SPI:
```

(continues on next page)

(continued from previous page)

```
spi = busio.SPI(clock=board.SCK, MOSI=board.MOSI, MISO=board.MISO)

# Create and setup the RA8875 display:
display = ra8875.RA8875(spi, cs=cs_pin, rst=rst_pin, baudrate=BAUDRATE)
display.init()

display.fill(RED)
time.sleep(0.500)
display.fill(YELLOW)
time.sleep(0.500)
display.fill(BLUE)
time.sleep(0.500)
display.fill(CYAN)
time.sleep(0.500)
display.fill(MAGENTA)
time.sleep(0.500)
display.fill(BLACK)
display.circle(100, 100, 50, BLACK)
display.fill_circle(100, 100, 49, BLUE)

display.fill_rect(10, 10, 400, 200, GREEN)
display.rect(10, 10, 400, 200, BLUE)
display.fill_round_rect(200, 10, 200, 100, 10, RED)
display.round_rect(200, 10, 200, 100, 10, BLUE)
display.pixel(10, 10, BLACK)
display.pixel(11, 11, BLACK)
display.line(10, 10, 200, 100, RED)
display.fill_triangle(200, 15, 250, 100, 150, 125, YELLOW)
display.triangle(200, 15, 250, 100, 150, 125, BLACK)
display.fill_ellipse(300, 100, 100, 40, BLUE)
display.ellipse(300, 100, 100, 40, RED)
display.curve(50, 100, 80, 40, 2, BLACK)
display.fill_curve(50, 100, 78, 38, 2, WHITE)

display.txt_set_cursor(display.width // 2 - 200, display.height // 2 - 20)
display.txt_trans(WHITE)
display.txt_size(2)
testvar = 99
display.txt_write("Player Score: " + str(testvar))

display.touch_init(int_pin)
display.touch_enable(True)

x_scale = 1024 / display.width
y_scale = 1024 / display.height

# Main loop:
while True:
    if display.touched():
        coords = display.touch_read()
        display.fill_circle(int(coords[0]/x_scale), int(coords[1]/y_scale), 4,␣
↪MAGENTA)
        display.txt_color(WHITE, BLACK)
        display.txt_set_cursor(display.width // 2 - 220, display.height // 2 - 20)
        display.txt_size(2)
        display.txt_write("Position (" + str(int(coords[0]/x_scale)) + ", " +
                          str(int(coords[1]/y_scale)) + ")")
```

6.2 Bitmap test

Listing 2: examples/ra8875_bmptest.py

```
# Quick bitmap test of RA8875 with Feather M4
import busio
import digitalio
import board

import adafruit_ra8875.ra8875 as ra8875
from adafruit_ra8875.ra8875 import color565
try:
    import struct
except ImportError:
    import ustruct as struct

WHITE = color565(255, 255, 255)

# Configuration for CS and RST pins:
cs_pin = digitalio.DigitalInOut(board.D9)
rst_pin = digitalio.DigitalInOut(board.D10)

# Config for display baudrate (default max is 6mhz):
BAUDRATE = 8000000

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

# Create and setup the RA8875 display:
display = ra8875.RA8875(spi, cs=cs_pin, rst=rst_pin, baudrate=BAUDRATE)
display.init()
display.fill(WHITE)

def convert_555_to_565(rgb):
    return (rgb & 0x7FE0) << 1 | 0x20 | rgb & 0x001F

class BMP(object):
    def __init__(self, filename):
        self.filename = filename
        self.colors = None
        self.data = 0
        self.data_size = 0
        self.bpp = 0
        self.width = 0
        self.height = 0
        self.read_header()

    def read_header(self):
        if self.colors:
            return
        with open(self.filename, 'rb') as f:
            f.seek(10)
            self.data = int.from_bytes(f.read(4), 'little')
            f.seek(18)
            self.width = int.from_bytes(f.read(4), 'little')
            self.height = int.from_bytes(f.read(4), 'little')
            f.seek(28)
```

(continues on next page)

(continued from previous page)

```
            self.bpp = int.from_bytes(f.read(2), 'little')
            f.seek(34)
            self.data_size = int.from_bytes(f.read(4), 'little')
            f.seek(46)
            self.colors = int.from_bytes(f.read(4), 'little')

    def draw(self, disp, x=0, y=0):
        print("{:d}x{:d} image".format(self.width, self.height))
        print("{:d}-bit encoding detected".format(self.bpp))
        line = 0
        line_size = self.width * (self.bpp//8)
        if line_size % 4 != 0:
            line_size += (4 - line_size % 4)
        current_line_data = b''
        with open(self.filename, 'rb') as f:
            f.seek(self.data)
            disp.set_window(x, y, self.width, self.height)
            for line in range(self.height):
                current_line_data = b''
                line_data = f.read(line_size)
                for i in range(0, line_size, self.bpp//8):
                    if (line_size-i) < self.bpp//8:
                        break
                    if self.bpp == 16:
                        color = convert_555_to_565(line_data[i] | line_data[i+1] << 8)
                    if self.bpp == 24 or self.bpp == 32:
                        color = color565(line_data[i+2], line_data[i+1], line_data[i])
                    current_line_data = current_line_data + struct.pack(">H", color)
                disp.setxy(x, self.height - line + y)
                disp.push_pixels(current_line_data)
            disp.set_window(0, 0, disp.width, disp.height)

bitmap = BMP("/ra8875_blinka.bmp")
x_position = (display.width // 2) - (bitmap.width // 2)
y_position = (display.height // 2) - (bitmap.height // 2)
bitmap.draw(display, x_position, y_position)
```

6.3 adafruit_ra8875.ra8875

A Driver Library for the RA8875

- Author(s): Melissa LeBlanc-Williams

6.3.1 Implementation Notes

Hardware:

- RA8875 Driver Board for 40-pin TFT Touch Displays - 800x480: https://www.adafruit.com/product/1590

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

class adafruit_ra8875.ra8875.**RA8875**(*spi*, *cs*, *rst=None*, *width=800*, *height=480*, *baudrate=6000000*, *polarity=0*, *phase=0*)

Graphics Library Class for the Display. Contains all the hardware accelerated geometry Functions. For full display functionality, use this class. Valid display sizes are currently 800x480 and 480x272.

circle(*x_center*, *y_center*, *radius*, *color*)

Draw a circle (HW Accelerated)

Parameters

- **x_center** (*int*) – The X coordinate of the center of the circle
- **y_center** (*int*) – The Y coordinate of the center of the circle
- **radius** (*int*) – The radius of the circle
- **color** (*int*) – The color of the circle

curve(*x_center*, *y_center*, *h_axis*, *v_axis*, *curve_part*, *color*)

Draw a Curve (HW Accelerated) This is basically a quarter of an ellipse.

Parameters

- **x_center** (*int*) – The X coordinate of the focal point of the curve
- **y_center** (*int*) – The Y coordinate of the focal point of the curve
- **h_axis** (*int*) – The length of the horizontal axis of the full ellipse
- **v_axis** (*int*) – The length of the vertical axis of the full ellipse
- **curve_part** (*byte*) – A number between 0-3 specifying the quarter section
- **color** (*int*) – The color of the curve

ellipse(*x_center*, *y_center*, *h_axis*, *v_axis*, *color*)

Draw an ellipse (HW Accelerated)

Parameters

- **x_center** (*int*) – The X coordinate of the center of the ellipse
- **y_center** (*int*) – The Y coordinate of the center of the ellipse
- **h_axis** (*int*) – The length of the horizontal axis
- **v_axis** (*int*) – The length of the vertical axis
- **color** (*int*) – The color of the ellipse

fill(*color*)

Fill the Entire Screen (HW Accelerated)

Parameters **color** (*int*) – The color to Fill the screen

fill_circle(*x_center*, *y_center*, *radius*, *color*)

Draw a filled circle (HW Accelerated)

Parameters

- **x_center** (*int*) – The X coordinate of the center of the circle
- **y_center** (*int*) – The Y coordinate of the center of the circle
- **radius** (*int*) – The radius of the circle
- **color** (*int*) – The color of the circle

fill_curve(*x_center*, *y_center*, *h_axis*, *v_axis*, *curve_part*, *color*)
Draw a Filled Curve (HW Accelerated) This is basically a quarter of an ellipse.

Parameters

- **x_center** (*int*) – The X coordinate of the focal point of the curve
- **y_center** (*int*) – The Y coordinate of the focal point of the curve
- **h_axis** (*int*) – The length of the horizontal axis of the full ellipse
- **v_axis** (*int*) – The length of the vertical axis of the full ellipse
- **curve_part** (*byte*) – A number between 0-3 specifying the quarter section
- **color** (*int*) – The color of the curve

fill_ellipse(*x_center*, *y_center*, *h_axis*, *v_axis*, *color*)
Draw a Filled Ellipse (HW Accelerated)

Parameters

- **x_center** (*int*) – The X coordinate of the center of the ellipse
- **y_center** (*int*) – The Y coordinate of the center of the ellipse
- **h_axis** (*int*) – The length of the horizontal axis
- **v_axis** (*int*) – The length of the vertical axis
- **color** (*int*) – The color of the ellipse

fill_rect(*x*, *y*, *width*, *height*, *color*)
Draw a filled rectangle (HW Accelerated)

Parameters

- **x** (*int*) – The X coordinate of the left side of the rectangle
- **y** (*int*) – The Y coordinate of the top side of the rectangle
- **width** (*int*) – The width of the rectangle
- **height** (*int*) – The height of the rectangle
- **color** (*int*) – The color of the rectangle

fill_round_rect(*x*, *y*, *width*, *height*, *radius*, *color*)
Draw a filled rounded rectangle

Parameters

- **x** (*int*) – The X coordinate of the left side of the rectangle
- **y** (*int*) – The Y coordinate of the top side of the rectangle
- **width** (*int*) – The width of the rectangle
- **height** (*int*) – The height of the rectangle
- **radius** (*int*) – The radius of the corners
- **color** (*int*) – The color of the rectangle

fill_triangle(*x1*, *y1*, *x2*, *y2*, *x3*, *y3*, *color*)
Draw a Filled Triangle (HW Accelerated)

Parameters

- **x1** (*int*) – The X coordinate of the first point of the triangle

- **y1** (*int*) – The Y coordinate of the first point of the triangle
- **x2** (*int*) – The X coordinate of the second point of the triangle
- **y2** (*int*) – The Y coordinate of the second point of the triangle
- **x3** (*int*) – The X coordinate of the third point of the triangle
- **y3** (*int*) – The Y coordinate of the third point of the triangle
- **color** (*int*) – The color of the triangle

hline(*x*, *y*, *width*, *color*)
Draw a Horizontal Line (HW Accelerated)

Parameters

- **x** (*int*) – The X coordinate of the beginning point of the line
- **y** (*int*) – The Y coordinate of the beginning point of the line
- **width** (*int*) – The width of the line
- **color** (*int*) – The color of the line

line(*x1*, *y1*, *x2*, *y2*, *color*)
Draw a Line (HW Accelerated)

Parameters

- **x1** (*int*) – The X coordinate of the beginning point of the line
- **y1** (*int*) – The Y coordinate of the beginning point of the line
- **x2** (*int*) – The X coordinate of the end point of the line
- **y2** (*int*) – The Y coordinate of the end point of the line
- **color** (*int*) – The color of the line

rect(*x*, *y*, *width*, *height*, *color*)
Draw a rectangle (HW Accelerated)

Parameters

- **x** (*int*) – The X coordinate of the left side of the rectangle
- **y** (*int*) – The Y coordinate of the top side of the rectangle
- **width** (*int*) – The width of the rectangle
- **height** (*int*) – The height of the rectangle
- **color** (*int*) – The color of the rectangle

round_rect(*x*, *y*, *width*, *height*, *radius*, *color*)
Draw a rounded rectangle

Parameters

- **x** (*int*) – The X coordinate of the left side of the rectangle
- **y** (*int*) – The Y coordinate of the top side of the rectangle
- **width** (*int*) – The width of the rectangle
- **height** (*int*) – The height of the rectangle
- **radius** (*int*) – The radius of the corners

- **color** (*int*) – The color of the rectangle

triangle(*x1*, *y1*, *x2*, *y2*, *x3*, *y3*, *color*)
Draw a Triangle (HW Accelerated)

Parameters

- **x1** (*int*) – The X coordinate of the first point of the triangle
- **y1** (*int*) – The Y coordinate of the first point of the triangle
- **x2** (*int*) – The X coordinate of the second point of the triangle
- **y2** (*int*) – The Y coordinate of the second point of the triangle
- **x3** (*int*) – The X coordinate of the third point of the triangle
- **y3** (*int*) – The Y coordinate of the third point of the triangle
- **color** (*int*) – The color of the triangle

vline(*x*, *y*, *height*, *color*)
Draw a Vertical Line (HW Accelerated)

Parameters

- **x** (*int*) – The X coordinate of the beginning point of the line
- **y** (*int*) – The Y coordinate of the beginning point of the line
- **height** (*int*) – The height of the line
- **color** (*int*) – The color of the line

class adafruit_ra8875.ra8875.**RA8875Display**(*spi*, *cs*, *rst=None*, *width=800*, *height=480*, *baudrate=6000000*, *polarity=0*, *phase=0*)
Drawing Class for the Display. Contains all the basic drawing functionality as well as the text functions. Valid display sizes are currently 800x480 and 480x272.

Parameters

- **spi** (*SPI*) – The spi peripheral to use
- **cs** (*DigitalInOut*) – The chip-select pin to use (sometimes labeled "SS")
- **rst** (*DigitalInOut*) – (optional) The reset pin if it exists (default=None)
- **width** (*int*) – (optional) The width of the display in pixels (default=800)
- **height** (*int*) – (optional) The height of the display in pixels (default=480)
- **baudrate** (*int*) – (optional) The spi speed (default=6000000)
- **phase** (*int*) – (optional) The spi phase (default=0)
- **polarity** (*int*) – (optional) The spi polarity (default=0)

pixel(*x*, *y*, *color*)
Draw a pixel at the X and Y coordinates of the specified color

Parameters

- **x** (*int*) – The X coordinate to set the cursor
- **y** (*int*) – The Y coordinate to set the cursor
- **color** (*int*) – The color of the pixel

push_pixels(*pixel_data*)
Push a stream of pixel data to the screen.

Parameters **pixel_data** (*bytearray*) – Raw pixel data to push

set_bgcolor(*color*)
Set the text background color

Parameters **color** (*int*) – The color behind the text

set_color(*color*)
Set the foreground color for graphics/text

Parameters **color** (*int*) – The of the text or graphics

set_window(*x*, *y*, *width*, *height*)
Set an Active Drawing Window, which can be used in conjuntion with push_pixels for faster drawing

Parameters

- **x** (*int*) – The X coordinate of the left side of the window
- **y** (*int*) – The Y coordinate of the top side of the window
- **width** (*int*) – The width of the window
- **height** (*int*) – The height of the window

setxy(*x*, *y*)
Set the X and Y coordinates of the Graphic Cursor

Parameters

- **x** (*int*) – The X coordinate to set the cursor
- **y** (*int*) – The Y coordinate to set the cursor

txt_color(*fgcolor*, *bgcolor*)
Set the text foreground and background colors

Parameters

- **fgcolor** (*int*) – Foreground Color - The color of the text
- **bgcolor** (*int*) – Background Color - The color behind the text

txt_set_cursor(*x*, *y*)
Set the X and Y coordinates of the Text Cursor

Parameters

- **x** (*int*) – The X coordinate to set the cursor
- **y** (*int*) – The Y coordinate to set the cursor

txt_size(*scale*)
Set the Text Size (0-3)

Parameters **scale** (*byte*) – The the size to scale the Text to

txt_trans(*color*)
Set the text foreground color with a transparent background

Parameters **color** (*int*) – The color of the text

txt_write(*string*)
Write text at the current cursor location using current settings

Parameters **string** (*str*) – The text string to write

class adafruit_ra8875.ra8875.**RA8875_Device**(*spi*, *cs*, *rst=None*, *width=800*, *height=480*, *baudrate=6000000*, *polarity=0*, *phase=0*)

Base Class for the Display. Contains all the low level stuff. As well as the touch functions. Valid display sizes are currently 800x480 and 480x272.

brightness(*level*)

Configure the backlight brightness (0-255)

Parameters **level** (*byte*) – The PWM Duty Cycle

init(*start_on=True*)

Send the Init Commands for the selected Display Size

Parameters **start_on** (*bool*) – (optional) If the display should start in an On State (default=True)

pllinit()

Init the Controller PLL

reset()

Perform a hard reset

sleep(*sleep*)

Turn the display off with and set or remove the sleep state

Parameters **sleep** (*bool*) – Should we enable sleep mode

soft_reset()

Perform a soft reset

touch_enable(*touch_on*)

Enable touch functionality

Parameters **touch_on** (*bool*) – Enable/Disable the Touch Functionality

touch_init(*tpin=None*, *enable=True*)

Initialize the Touchscreen

Parameters

- **tpin** (*DigitalInOut*) – (Optional) The Touch Screen Interrupt Pin (default=None)
- **enable** (*bool*) – Enable the Touch Functionality as well

touch_read()

Read the X and Y Coordinates of the current Touch Position

Returns The coordinate of the detected touch

Return type tuple[int, int]

touched()

Check if the Screen is currently being touched. If a touch interrupt was specified, this is checked first.

Returns Is screen is currently being touched

Return type bool

turn_on(*display_on*)

Turn the display on or off

Parameters **start_on** (*bool*) – If the display should turn on or off

adafruit_ra8875.ra8875.**color565**(*r*, *g=0*, *b=0*)

Convert red, green and blue values (0-255) into a 16-bit 565 encoding.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index

V