
sparkfun*qwii_cmicro_led*

Release 0.9.0

May 13, 2021

Contents:

1	Contents	3
2	Supported Platforms	5
3	Dependencies	7
3.1	Documentation	7
3.2	Installation	7
3.2.1	PyPi Installation	7
3.2.2	Local Installation	7
3.3	Example Use	8
4	Table of Contents	9
4.1	API Reference	9
4.1.1	qwiic_micro_oled	9
4.2	Bitmap Example	15
4.3	Cube Example	18
4.4	Complete Example	21
4.5	Basic Operation	27
5	Indices and tables	31
	Python Module Index	33
	Index	35

Python package for the qwiic [Micro OLED board](#)

This package is a port of the [SparkFun Micro OLED Breakout Arduino Library](#)

This package can be used in conjunction with the overall [SparkFun qwiic Python Package](#)

New to qwiic? Take a look at the entire [SparkFun qwiic ecosystem](#).

CHAPTER 1

Contents

- *Supported Platforms*
- *Dependencies*
- *Installation*
- *Documentation*
- *Example Use*

CHAPTER 2

Supported Platforms

The qwiic Python package current supports the following platforms:

- Raspberry Pi
- NVidia Jetson Nano
- Google Coral Development Board

This driver package depends on the qwiic I2C driver: [Qwiic_I2C_Py](#)

3.1 Documentation

The SparkFun qwiic Micro OLED module documentation is hosted at [ReadTheDocs](#)

3.2 Installation

3.2.1 PyPi Installation

This repository is hosted on PyPi as the [sparkfun-qwiic-micro-oled](#) package. On systems that support PyPi installation via pip, this library is installed using the following commands

For all users (note: the user must have sudo privileges):

```
sudo pip install sparkfun-qwiic-micro-oled
```

For the current user:

```
pip install sparkfun_qwiic_micro_oled
```

3.2.2 Local Installation

To install, make sure the setuptools package is installed on the system.

Direct installation at the command line:

```
python setup.py install
```

To build a package for use with pip:

```
python setup.py sdist
```

A package file is built and placed in a subdirectory called dist. This package file can be installed using pip.

```
cd dist
pip install sparkfun_micro_oled-<version>.tar.gz
```

3.3 Example Use

See the examples directory for more detailed use examples.

```
import qwiic_micro_oled
import sys

def runExample():

    # These three lines of code are all you need to initialize the
    # OLED and print the splash screen.

    # Before you can start using the OLED, call begin() to init
    # all of the pins and configure the OLED.

    print("\nSparkFun Micro OLED - Hello Example\n")
    myOLED = qwiic_micro_oled.QwiicMicroOled()

    if myOLED.isConnected() == False:
        print("The Qwiic Micro OLED device isn't connected to the system. Please_
↪check your connection", \
            file=sys.stderr)
        return

    # Before you can start using the OLED, call begin() to init all of the pins and_
↪configure the OLED.
    myOLED.begin()

    myOLED.clear(myOLED.PAGE) # Clear the display's buffer

    myOLED.print("Hello World") # Add "Hello World" to buffer

    # To actually draw anything on the display, you must call the display() function.
    myOLED.display()

runExample()
```

4.1 API Reference

4.1.1 qwiic_micro_oled

Python module for the [Qwiic Micro OLED Display](<https://www.sparkfun.com/products/14532>)

This python package is a port of the existing [SparkFun Micro OLED Arduino Library](https://github.com/sparkfun/SparkFun_Micro_OLED_Arduino_Library)

This package can be used in conjunction with the overall [SparkFun qwiic Python Package](https://github.com/sparkfun/Qwiic_Py)

New to qwiic? Take a look at the entire [SparkFun qwiic ecosystem](<https://www.sparkfun.com/qwiic>).

```
class qwiic_micro_oled.QwiicMicroOled (address=None, i2c_driver=None)
```

Parameters

- **address** – The I2C address to use for the device. If not provided, the default address is used.
- **i2c_driver** – An existing i2c driver object. If not provided a driver object is created.

Returns The Micro OLED device object.

Return type Object

```
begin ()
```

Initialize the operation of the Micro OLED module

Returns Returns true if the initialization was successful, otherwise False.

Return type bool

```
circle (x0, y0, radius, color=None, mode=None)
```

Draw a circle on the display. A color can be specified. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x0** – The X center position for the circle
- **y0** – The Y center position for the circle.
- **radius** – The radius of the circle
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixl to the screen bufffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

clear (*mode, value=0*)

Clear the display on the OLED Device.

Parameters

- **mode** – To clear GDRAM inside the LCD controller, pass in the variable mode = ALL, and to clear screen page buffer pass in the variable mode = PAGE.
- **value** – The value to clear the screen to. Default value is 0

Returns No return value

connected

Determine if a Micro OLED device is conntected to the system..

Returns True if the device is connected, otherwise False.

Return type bool

contrast (*contrast*)

Set the OLED contract value from 0 to 255. Note: Contrast level is not very obvious on the display.

Parameters **contrast** – Contrast Value between 0-255

Returns No return value

display ()

Display the current screen buffer on the Display device. Bulk move the screen buffer to the SSD1306 controller's memory so that images/graphics drawn on the screen buffer will be displayed on the OLED.

Returns No return value

draw_bitmap (*bitArray*)

Draw Bitmap image on screen. To use, create int array that is 64x48 pixels (384 bytes). Then call .draw_bitmap and pass it the array.

Parameters **bitArray** – The bitmap to draw

Returns No return value

draw_char (*x, y, c, color=None, mode=None*)

Draw character c using color and draw mode at x,y. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X position on the display
- **y** – The Y position on the display
- **c** – The character to draw
- **color** – The color to draw. If not set, the default foreground color is used.

- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

flip_horizontal (*flip*)

Flip the graphics on the OLED horizontally.

Returns No return value

flip_vertical (*flip*)

Flip the graphics on the OLED vertically.

Returns No return value

font_height

The height of the current font

Returns height of the font

Rvalue integer

font_type

Return the font type number of the current font.

Returns Font type number.

Rvalue integer

font_width

The width of the current font

Returns width of the font

Rvalue integer

get_font_height ()

The height of the current font

Returns height of the font

Rvalue integer

get_font_start_char ()

Return the starting ASCII character of the current font, not all fonts start with ASCII character 0. Custom fonts can start from any ASCII character.

Returns Starting character of the current font.

Rvalue integer

get_font_total_char ()

The total number of characters in the current font.

Returns Total number of characters

Rvalue integer

get_font_type ()

Return the font type number of the current font.

Returns Font type number.

Rvalue integer

get_font_width ()

The width of the current font

Returns width of the font

Rvalue integer

get_lcd_height ()

The height of the display in pixels

Returns height of the display

Rvalue integer

get_lcd_width ()

The width of the display in pixels

Returns width of the display

Rvalue integer

get_screenbuffer ()

Return a pointer to the start of the RAM screen buffer for direct access.

Returns The internal screen buffer

Return type integer array

get_total_fonts ()

Return the total number of fonts loaded into the MicroOLED's flash memory.

Returns Total number of fonts available

Rvalue integer

height

The height of the display in pixels

Returns height of the display

Rvalue integer

invert (*inv*)

Invert the display of the display. The WHITE color of the display will turn to BLACK and the BLACK will turn to WHITE.

Parameters **inv** – If True, the screen is inverted. If False the screen is set to Normal mode.

Returns No return value

is_connected ()

Determine if a Micro OLED device is connected to the system..

Returns True if the device is connected, otherwise False.

Return type bool

line (*x0, y0, x1, y1, color=None, mode=None*)

Draw a line starting at and ending at specified coordinates, with a given color. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x0** – The X starting position for the line
- **y0** – The Y starting position for the line.
- **x1** – The X ending position for the line
- **y1** – The Y ending position for the line.

- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

line_h (*x, y, width, color=None, mode=None*)

Draw a horizontal line defined by a starting position and width. A color can be specified. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X starting position for the line
- **y** – The Y starting position for the line.
- **width** – The width (length) of the line
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

line_v (*x, y, height, color=None, mode=None*)

Draw a vertical line defined by a starting position and width. A color can be specified. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X starting position for the line
- **y** – The Y starting position for the line.
- **height** – The height (length) of the line
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

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

Draw a pixel at a given position, with a given color. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X position on the display
- **y** – The Y position on the display
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

print (*text*)

Print a line of text on the display using the current font, starting at the current position.

Parameters **text** – The line of text to write.

Returns No return value

rect (*x, y, width, height, color=None, mode=None*)

Draw a rectangle on the display. A color can be specified. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X starting position for the rectangle
- **y** – The Y starting position for the rectangle.
- **width** – The width of the rectangle
- **height** – The height of the rectangle
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

rect_fill (*x, y, width, height, color=None, mode=None*)

Draw a filled rectangle on the display. A color can be specified. Pixel copy mode is either Normal (source copy) or XOR

Parameters

- **x** – The X starting position for the rectangle
- **y** – The Y starting position for the rectangle.
- **width** – The width of the rectangle
- **height** – The height of the rectangle
- **color** – The color to draw. If not set, the default foreground color is used.
- **mode** – The mode to draw the pixel to the screen buffer. Value can be either XOR or NORM. Default is NORM

Returns No return value

scroll_left (*start, stop*)

Set row start to row stop on the OLED to scroll left. Refer to <http://learn.microview.io/intro/general-overview-of-microview.html> for explanation of the rows.

Parameters

- **start** – The starting position on the display
- **stop** – The stopping position on the display

Returns No return value

scroll_right (*start, stop*)

Set row start to row stop on the OLED to scroll right. Refer to <http://learn.microview.io/intro/general-overview-of-microview.html> for explanation of the rows.

Parameters

- **start** – The starting position on the display
- **stop** – The stopping position on the display

Returns No return value

scroll_stop ()

Stop scrolling operation.

Returns No return value

set_color (*color*)

Set the current draw's color. Only WHITE and BLACK available.

Parameters **color** – Color Value

Returns No return value

set_column_address (*colAddress*)

Set SSD1306 column address.

Parameters **colAddress** – The column address command and address

Returns No return value

set_cursor (*x, y*)

Set the current cursor position for writing text

Parameters

- **x** – The X position on the display
- **y** – The Y position on the display

Returns No return value

set_draw_mode (*mode*)

Set current draw mode with NORM or XOR.

Parameters **mode** – Draw Mode

Returns No return value

set_font_type (*font_type*)

Set the current font type number, ie changing to different fonts base on the type provided.

Parameters **type** – The type to set the font to.

Returns No return value

set_page_address (*pageAddress*)

Set SSD1306 page address.

Parameters **pageAddress** – The page address command and address

Returns No return value

width

The width of the display in pixels

Returns width of the display

Rvalue integer

write (*c*)

Write a character on the display using the current font, at the current position.

Parameters **c** – Character to write. A value of '\n' starts a new line.

Returns 1 on success

4.2 Bitmap Example

Listing 1: examples/qwiic_micro_oled_bitmap.py

```

1  #!/usr/bin/env python
2  #-----
3  # qwiic_micro_oled_hello.py
4  #
5  # Simple Example for the Qwiic MicroOLED Device
6  #-----
7  #
8  # Written by SparkFun Electronics, May 2019
9  #
10 # This python library supports the SparkFun Electroncis qwiic
11 # qwiic sensor/board ecosystem on a Raspberry Pi (and compatable) single
12 # board computers.
13 #
14 # More information on qwiic is at https:# www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun. Buy a board!
17 #
18 #=====
19 # Copyright (c) 2019 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining a copy
22 # of this software and associated documentation files (the "Software"), to deal
23 # in the Software without restriction, including without limitation the rights
24 # to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
25 # copies of the Software, and to permit persons to whom the Software is
26 # furnished to do so, subject to the following conditions:
27 #
28 # The above copyright notice and this permission notice shall be included in all
29 # copies or substantial portions of the Software.
30 #
31 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
32 # IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
33 # FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
34 # AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
35 # LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
36 # OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
37 # SOFTWARE.
38 #=====
39 # Example - simple command to display a bitmap on the OLED.
40 #
41
42 from __future__ import print_function
43 import qwiic_micro_oled
44 import sys
45
46
47 bender = [ \
48     0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x7F, 0xBF, 0xDF, 0x5F, 0x5F, \
49     ↪0x5F, 0x5F, \
50     0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, \
51     ↪0x5F, 0x5F, \
52     0x5F, 0xDF, 0xBF, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
53     ↪0xFF, 0xFF, \

```

(continues on next page)

(continued from previous page)

```

52  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x07, 0xF9, 0xFE, 0x07, 0x01, 0x00, 0x00, 0xF8, \
↳0xFE, 0xFF, \
53  0xFF, 0xFF, 0x1F, 0x1F, 0x1F, 0xFF, 0xFF, 0xFE, 0xFC, 0xF8, 0xF0, 0xE0, 0x00, 0x00, \
↳0x00, 0x00, \
54  0xE0, 0xF0, 0xF8, 0xFC, 0xFE, 0xFF, 0xFF, 0x1F, 0x1F, 0x1F, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFE, 0xF8, \
55  0x00, 0x00, 0x01, 0x07, 0xFE, 0xF9, 0x07, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xFF, \
56  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xF9, 0xE7, 0xDC, 0xB0, 0xA0, 0x40, 0x41, \
↳0x47, 0x4F, \
57  0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x4F, 0x47, 0x43, 0x40, \
↳0x40, 0x40, \
58  0x43, 0x47, 0x4F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x5F, 0x4F, 0x47, \
↳0x43, 0x40, \
59  0x40, 0xA0, 0xB0, 0xDE, 0xE7, 0xF9, 0xFE, 0x1F, 0x0F, 0x07, 0x73, 0x79, 0xFF, 0xFF, \
↳0xFF, 0xFF, \
60  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0x7F, \
61  0xBF, 0x5F, 0xEF, 0x0F, 0xEF, 0xEF, 0xDF, 0xDF, 0x1F, 0xDF, 0xDF, 0xDF, 0xDF, 0x1F, \
↳0xDF, 0xDF, \
62  0xDF, 0xDF, 0xDF, 0x1F, 0xDF, 0xDF, 0xDF, 0xEF, 0x0F, 0xEF, 0xDF, 0xBF, 0x7F, 0xFF, \
↳0xFF, 0xFF, \
63  0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0xFF, 0xFF, 0xFF, 0xBE, 0x9C, 0xC0, 0xE0, 0xF0, 0xF9, \
↳0xFF, 0xFF, \
64  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xC0, \
65  0xB7, 0x6F, 0xEE, 0x00, 0xDE, 0xDE, 0xDE, 0xDD, 0x00, 0xDD, 0xDD, 0xDD, 0xDD, 0x00, \
↳0xDD, 0xDD, \
66  0xDD, 0xC5, 0xC1, 0x00, 0xC9, 0xC5, 0xC1, 0x01, 0xC8, 0xC4, 0x42, 0x80, 0xC0, 0xE8, \
↳0xE4, 0xE2, \
67  0xE0, 0xE0, 0xEF, 0xEF, 0xE6, 0xF0, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xFF, \
68  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xFF, \
69  0xFF, 0xFF, 0xFE, 0xFE, 0xFD, 0xFD, 0xFD, 0xFB, 0xF8, 0xFB, 0xFB, 0xFB, 0xFB, 0xF8, \
↳0xFB, 0xFB, \
70  0xFB, 0xFB, 0xFB, 0xF8, 0xFB, 0xFD, 0xFD, 0xFC, 0xFE, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xFF, \
71  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, \
↳0xFF, 0xFF
72 ]

```

```

73
74 def runExample():
75
76     # These three lines of code are all you need to initialize the
77     # OLED and print the splash screen.
78
79     # Before you can start using the OLED, call begin() to init
80     # all of the pins and configure the OLED.
81
82
83     print("\nSparkFun Micro OLED Bitmap Example\n")
84     myOLED = qwiic_micro_oled.QwiicMicroOled()
85
86     if not myOLED.connected:
87         print("The Qwiic Micro OLED device isn't connected to the system. Please check_
↳your connection", \

```

(continues on next page)

(continued from previous page)

```

88     file=sys.stderr)
89     return
90
91 myOLED.begin()
92 # clear(ALL) will clear out the OLED's graphic memory.
93 # clear(PAGE) will clear the Arduino's display buffer.
94 myOLED.clear(myOLED.PAGE) # Clear the display's memory (gets rid of artifacts)
95
96 myOLED.draw_bitmap(bender)
97 # To actually draw anything on the display, you must call the
98 # display() function.
99 myOLED.display()
100
101 if __name__ == '__main__':
102     try:
103         runExample()
104     except (KeyboardInterrupt, SystemExit) as exErr:
105         print("\nEnding OLED bitmap Example")
106         sys.exit(0)

```

4.3 Cube Example

Listing 2: examples/qwiic_micro_oled_cube.py

```

1  #!/usr/bin/env python
2  #-----
3  # qwiic_micro_oled_cube.py
4  #
5  # Simple Example for the Qwiic MicroOLED Device
6  #-----
7  #
8  # Written by SparkFun Electronics, May 2019
9  #
10 # This python library supports the SparkFun Electronics qwiic
11 # qwiic sensor/board ecosystem on a Raspberry Pi (and compatible) single
12 # board computers.
13 #
14 # More information on qwiic is at https://www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun. Buy a board!
17 #=====
18 # Copyright (c) 2019 SparkFun Electronics
19 #
20 # Permission is hereby granted, free of charge, to any person obtaining a copy
21 # of this software and associated documentation files (the "Software"), to deal
22 # in the Software without restriction, including without limitation the rights
23 # to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
24 # copies of the Software, and to permit persons to whom the Software is
25 # furnished to do so, subject to the following conditions:
26 #
27 # The above copyright notice and this permission notice shall be included in all
28 # copies or substantial portions of the Software.
29 #
30 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

```

(continues on next page)

(continued from previous page)

```

31 # IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
32 # FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
33 # AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
34 # LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
35 # OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
36 # SOFTWARE.
37 #=====
38 # Example - simple command to draw a cube the OLED.
39 #
40
41 from __future__ import print_function, division
42 import qwiic_micro_oled
43 import sys
44 import time
45 import math
46
47 d = 3
48 px = [-d, d, d, -d, -d, d, d, -d ]
49 py = [-d, -d, d, d, -d, -d, d, d ]
50 pz = [-d, -d, -d, -d, d, d, d, d ]
51
52 p2x = [0,0,0,0,0,0,0,0]
53 p2y = [0,0,0,0,0,0,0,0]
54
55 r = [0,0,0]
56
57 SHAPE_SIZE=600
58 def drawCube(oled):
59
60     global p2x, p2y, r
61
62
63     r[0]=r[0] + math.pi/180.0 # Add a degree
64     r[1]=r[1] + math.pi/180.0 # Add a degree
65     r[2]=r[2] + math.pi/180.0 # Add a degree
66     if r[0] >= 360.0*math.pi/180.0:
67         r[0] = 0
68     if r[1] >= 360.0*math.pi/180.0:
69         r[1] = 0
70     if r[2] >= 360.0*math.pi/180.0:
71         r[2] = 0
72
73     scrWidth = oled.get_lcd_width()
74     scrHeight = oled.get_lcd_height()
75
76     for i in range(8):
77
78         px2 = px[i]
79         py2 = math.cos(r[0])*py[i] - math.sin(r[0])*pz[i]
80         pz2 = math.sin(r[0])*py[i] + math.cos(r[0])*pz[i]
81
82         px3 = math.cos(r[1])*px2 + math.sin(r[1])*pz2
83         py3 = py2
84         pz3 = -math.sin(r[1])*px2 + math.cos(r[1])*pz2
85
86         ax = math.cos(r[2])*px3 - math.sin(r[2])*py3
87         ay = math.sin(r[2])*px3 + math.cos(r[2])*py3

```

(continues on next page)

(continued from previous page)

```

88     az = pz3-150
89
90     p2x[i] = scrWidth/2+ax*SHAPE_SIZE/az
91     p2y[i] = scrHeight/2+ay*SHAPE_SIZE/az
92
93     oled.clear(oled.PAGE)
94
95     for i in range(3):
96
97         oled.line(p2x[i],p2y[i],p2x[i+1],p2y[i+1])
98         oled.line(p2x[i+4],p2y[i+4],p2x[i+5],p2y[i+5])
99         oled.line(p2x[i],p2y[i],p2x[i+4],p2y[i+4])
100
101     oled.line(p2x[3],p2y[3],p2x[0],p2y[0])
102     oled.line(p2x[7],p2y[7],p2x[4],p2y[4])
103     oled.line(p2x[3],p2y[3],p2x[7],p2y[7])
104     oled.display()
105
106 def runExample():
107
108     # These three lines of code are all you need to initialize the
109     # OLED and print the splash screen.
110
111     # Before you can start using the OLED, call begin() to init
112     # all of the pins and configure the OLED.
113
114     print("\nSparkFun Micro OLED Cube Example\n")
115     myOLED = qwiic_micro_oled.QwiicMicroOled()
116
117     if not myOLED.connected:
118         print("The Qwiic Micro OLED device isn't connected to the system. Please_
↳check your connection", \
119               file=sys.stderr)
120         return
121
122     myOLED.begin()
123     # clear(ALL) will clear out the OLED's graphic memory.
124     # clear(PAGE) will clear the Arduino's display buffer.
125     myOLED.clear(myOLED.ALL) # Clear the display's memory (gets rid of artifacts)
126     # To actually draw anything on the display, you must call the
127     # display() function.
128     myOLED.display()
129
130
131     while True:
132
133         drawCube(myOLED)
134         time.sleep(.01)
135
136
137
138 if __name__ == '__main__':
139     try:
140         runExample()
141     except (KeyboardInterrupt, SystemExit) as exErr:
142         print("\nEnding OLED Cube Example")
143         sys.exit(0)

```


4.4 Complete Example

Listing 3: examples/qwiic_micro_oled_demo.py

```

1  #!/usr/bin/env python
2  #-----
3  # qwiic_micro_oled_demo.py
4  #
5  # Simple Example for the Qwiic MicroOLED Device
6  #-----
7  #
8  # Written by SparkFun Electronics, May 2019
9  #
10 # This python library supports the SparkFun Electronics qwiic
11 # qwiic sensor/board ecosystem on a Raspberry Pi (and compatible) single
12 # board computers.
13 #
14 # More information on qwiic is at https://www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun. Buy a board!
17 #
18 #=====
19 # Copyright (c) 2019 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining a copy
22 # of this software and associated documentation files (the "Software"), to deal
23 # in the Software without restriction, including without limitation the rights
24 # to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
25 # copies of the Software, and to permit persons to whom the Software is
26 # furnished to do so, subject to the following conditions:
27 #
28 # The above copyright notice and this permission notice shall be included in all
29 # copies or substantial portions of the Software.
30 #
31 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
32 # IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
33 # FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
34 # AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
35 # LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
36 # OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
37 # SOFTWARE.
38 #=====
39 # Example - simple command to setup the OLED.
40 #
41
42 from __future__ import print_function, division
43 import qwiic_micro_oled
44 import time
45 import sys
46 import math
47 from random import randint
48
49 #-----
50 def pixelExample(myOLED):
51
52     print("Pixels!")
53

```

(continues on next page)

(continued from previous page)

```

54     lWidth = myOLED.get_lcd_width()
55     lHeight = myOLED.get_lcd_height()
56     for i in range(128):
57         myOLED.pixel(randint(0, lWidth), randint(0, lHeight))
58         myOLED.display()
59
60     myOLED.clear(myOLED.PAGE)
61     #-----
62     def lineExample(myOLED):
63
64         middleX = myOLED.get_lcd_width() // 2
65         middleY = myOLED.get_lcd_height() // 2
66
67         lineWidth = min(middleX, middleY)
68
69         print("Lines!")
70
71         for i in range(2):
72
73             for deg in range(0, 360, 15):
74
75                 xEnd = lineWidth * math.cos(deg * math.pi / 180.0)
76                 yEnd = lineWidth * math.sin(deg * math.pi / 180.0)
77
78                 myOLED.line(middleX, middleY, middleX + xEnd, middleY + yEnd)
79                 myOLED.display()
80                 time.sleep(.05)
81
82             for deg in range(0, 360, 15):
83
84                 xEnd = lineWidth * math.cos(deg * math.pi / 180.0)
85                 yEnd = lineWidth * math.sin(deg * math.pi / 180.0)
86
87                 myOLED.line(middleX, middleY, middleX + xEnd, middleY + yEnd, myOLED.
88 ↪BLACK, myOLED.NORM)
89                 myOLED.display()
90                 time.sleep(.05)
91     #-----
92     def shapeExample(myOLED):
93
94         print("Shapes!")
95
96         # Silly pong demo. It takes a lot of work to fake pong...
97         paddleW = 3 # Paddle width
98         paddleH = 15 # Paddle height
99
100        lWidth = myOLED.get_lcd_width()
101        lHeight = myOLED.get_lcd_height()
102
103        # Paddle 0 (left) position coordinates
104        paddle0_Y = (lHeight // 2) - (paddleH // 2)
105        paddle0_X = 2
106
107        # Paddle 1 (right) position coordinates
108        paddle1_Y = (lHeight // 2) - (paddleH // 2)
109        paddle1_X = lWidth - 3 - paddleW

```

(continues on next page)

(continued from previous page)

```

110 ball_rad = 2 #Ball radius
111 # // Ball position coordinates
112 ball_X = paddle0_X + paddleW + ball_rad
113 ball_Y = randint(1 + ball_rad, lHeight - ball_rad) #paddle0_Y + ball_rad
114 ballVelocityX = 1 # Ball left/right velocity
115 ballVelocityY = 1 # Ball up/down velocity
116 paddle0Velocity = -1 # Paddle 0 velocity
117 paddle1Velocity = 1 # Paddle 1 velocity
118
119
120 while (ball_X - ball_rad > 1) and (ball_X + ball_rad < lWidth - 2):
121
122     # // Increment ball's position
123     ball_X += ballVelocityX
124     ball_Y += ballVelocityY
125     # // Check if the ball is colliding with the left paddle
126     if ball_X - ball_rad < paddle0_X + paddleW:
127
128         # // Check if ball is within paddle's height
129         if (ball_Y > paddle0_Y) and (ball_Y < paddle0_Y + paddleH):
130
131             ball_X +=1 # Move ball over one to the right
132             ballVelocityX = -ballVelocityX # Change velocity
133
134     # Check if the ball hit the right paddle
135     if ball_X + ball_rad > paddle1_X:
136
137         # Check if ball is within paddle's height
138         if (ball_Y > paddle1_Y) and (ball_Y < paddle1_Y + paddleH):
139
140             ball_X -= 1 # Move ball over one to the left
141             ballVelocityX = -ballVelocityX # change velocity
142
143     # // Check if the ball hit the top or bottom
144     if (ball_Y <= ball_rad) or (ball_Y >= (lHeight - ball_rad - 1)):
145
146         # Change up/down velocity direction
147         ballVelocityY = -ballVelocityY
148
149     # // Move the paddles up and down
150     paddle0_Y += paddle0Velocity
151     paddle1_Y += paddle1Velocity
152
153     # // Change paddle 0's direction if it hit top/bottom
154     if (paddle0_Y <= 1) or (paddle0_Y > lHeight - 2 - paddleH):
155
156         paddle0Velocity = -paddle0Velocity
157
158     # // Change paddle 1's direction if it hit top/bottom
159     if (paddle1_Y <= 1) or (paddle1_Y > lHeight - 2 - paddleH):
160
161         paddle1Velocity = -paddle1Velocity
162
163     # Draw the Pong Field
164     myOLED.clear(myOLED.PAGE) # Clear the page
165
166     # Draw an outline of the screen:

```

(continues on next page)

(continued from previous page)

```

167     myOLED.rect(0, 0, lWidth - 1, lHeight)
168
169     # Draw the center line
170     myOLED.rect_fill(lWidth//2 - 1, 0, 2, lHeight)
171
172     # Draw the Paddles:
173     myOLED.rect_fill(paddle0_X, paddle0_Y, paddleW, paddleH)
174     myOLED.rect_fill(paddle1_X, paddle1_Y, paddleW, paddleH)
175
176     # # Draw the ball:
177     myOLED.circle(ball_X, ball_Y, ball_rad)
178
179     # Actually draw everything on the screen:
180     myOLED.display()
181     time.sleep(.01) # Delay for visibility
182
183     time.sleep(.2)
184
185     #-----
186     def textExamples(myOLED):
187
188         print("Text!")
189
190         # Demonstrate font 0. 5x8 font
191         myOLED.clear(myOLED.PAGE) # Clear the screen
192         myOLED.set_font_type(0) # Set font to type 0
193         myOLED.set_cursor(0, 0) # Set cursor to top-left
194         # There are 255 possible characters in the font 0 type.
195         # Lets run through all of them and print them out!
196         for i in range(256):
197
198             # You can write byte values and they'll be mapped to
199             # their ASCII equivalent character.
200             myOLED.write(i) # Write a byte out as a character
201             myOLED.display() # Draw on the screen
202             # time.sleep(.05)
203
204             # We can only display 60 font 0 characters at a time.
205             # Every 60 characters, pause for a moment. Then clear
206             # the page and start over.
207             if (i%60 == 0) and (i != 0):
208
209                 time.sleep(.1)
210                 myOLED.clear(myOLED.PAGE) # Clear the page
211                 myOLED.set_cursor(0, 0) # Set cursor to top-left
212
213         time.sleep(.5) # Wait 500ms before next example
214
215         # Demonstrate font 1. 8x16. Let's use the print function
216         # to display every character defined in this font.
217         myOLED.set_font_type(1) # Set font to type 1
218         myOLED.clear(myOLED.PAGE) # Clear the page
219         myOLED.set_cursor(0, 0) # Set cursor to top-left
220         # Print can be used to print a string to the screen:
221         myOLED.print(" !\"#$%&'()*+,-./01234")
222         myOLED.display() # Refresh the display
223         time.sleep(1)

```

(continues on next page)

(continued from previous page)

```

224 myOLED.clear(myOLED.PAGE)
225 myOLED.set_cursor(0, 0)
226 myOLED.print("56789:<=>?@ABCDEFGH")
227 myOLED.display()
228 time.sleep(1)
229
230
231 myOLED.clear(myOLED.PAGE)
232 myOLED.set_cursor(0, 0)
233 myOLED.print("JKLMNOPQRSTUVWXYZ[\\]^")
234 myOLED.display()
235 time.sleep(1)
236
237 myOLED.clear(myOLED.PAGE)
238 myOLED.set_cursor(0, 0)
239 myOLED.print("_`abcdefghijklmnopqrs")
240 myOLED.display()
241 time.sleep(1)
242
243 myOLED.clear(myOLED.PAGE)
244 myOLED.set_cursor(0, 0)
245 myOLED.print("tuvwxyz{|}~")
246 myOLED.display()
247 time.sleep(1)
248
249 # Demonstrate font 2. 10x16. Only numbers and '.' are defined.
250 # This font looks like 7-segment displays.
251 # Lets use this big-ish font to display readings from the
252 # analog pins.
253 for i in range(25):
254
255     myOLED.clear(myOLED.PAGE)           # Clear the display
256     myOLED.set_cursor(0, 0)             # Set cursor to top-left
257     myOLED.set_font_type(0)             # Smallest font
258     myOLED.print("A0: ")                # Print "A0"
259     myOLED.set_font_type(2)             # 7-segment font
260     myOLED.print("%.3d" % randint(0,255))
261
262     myOLED.set_cursor(0, 16)            # Set cursor to top-middle-left
263     myOLED.set_font_type(0)             # Repeat
264     myOLED.print("A1: ")
265     myOLED.set_font_type(2)
266
267     myOLED.print("%.3d" % randint(0,255))
268     myOLED.set_cursor(0, 32)
269     myOLED.set_font_type(0)
270     myOLED.print("A2: ")
271     myOLED.set_font_type(2)
272     myOLED.print("%.3d" % randint(0,255))
273
274     myOLED.display()
275     time.sleep(.1)
276
277 # Demonstrate font 3. 12x48. Stopwatch demo.
278 myOLED.set_font_type(3) # Use the biggest font
279 ms = 0
280 s = 0

```

(continues on next page)

(continued from previous page)

```

281
282 while s <= 5:
283
284     myOLED.clear(myOLED.PAGE)      # Clear the display
285     myOLED.set_cursor(0, 0) # Set cursor to top-left
286     if s < 10:
287         myOLED.print("00")      # Print "00" if s is 1 digit
288     elif s < 100:
289         myOLED.print("0")      # Print "0" if s is 2 digits
290
291     myOLED.print(s)              # Print s's value
292     myOLED.print(":")           # Print ":"
293     myOLED.print(ms)            # Print ms value
294     myOLED.display()            # Draw on the screen
295     ms +=1                      # Increment ms
296     if ms >= 10 : #If ms is >= 10
297         ms = 0                  # Set ms back to 0
298         s +=1                   # and increment s
299
300     # Demonstrate font 4. 31x48. Let's use the print function
301     # to display some characters defined in this font.
302     myOLED.set_font_type(4)     # Set font to type 4
303     myOLED.clear(myOLED.PAGE)   #Clear the page
304     myOLED.set_cursor(0, 0) #Set cursor to top-left
305
306     # Print can be used to print a string to the screen:
307     myOLED.print("OL")
308     myOLED.display()            # Refresh the display
309     time.sleep(1)
310
311     myOLED.clear(myOLED.PAGE)
312     myOLED.set_cursor(0, 0)
313     myOLED.print("ED")
314     myOLED.display()
315     time.sleep(1)
316
317     myOLED.set_font_type(1)
318     myOLED.clear(myOLED.PAGE)
319     myOLED.set_cursor(0, 0)
320     myOLED.print("DONE!")
321     myOLED.display()
322     time.sleep(1)
323
324
325 #-----
326
327 def runExample():
328
329     # These three lines of code are all you need to initialize the
330     # OLED and print the splash screen.
331
332     # Before you can start using the OLED, call begin() to init
333     # all of the pins and configure the OLED.
334
335
336     print("\nSparkFun Micro OLED Everything Example\n")
337     myOLED = qwiic_micro_oled.QwiicMicroOled()

```

(continues on next page)

(continued from previous page)

```

338
339     if not myOLED.connected:
340         print("The Qwiic Micro OLED device isn't connected to the system. Please_
↪check your connection", \
341             file=sys.stderr)
342         return
343
344     myOLED.begin()
345     # clear(ALL) will clear out the OLED's graphic memory.
346     # clear(PAGE) will clear the Arduino's display buffer.
347     myOLED.clear(myOLED.ALL) # Clear the display's memory (gets rid of artifacts)
348     # To actually draw anything on the display, you must call the
349     # display() function.
350     myOLED.display()
351     time.sleep(1)
352
353     myOLED.clear(myOLED.PAGE)
354
355     print("-"*30)
356     pixelExample(myOLED)
357     print("-"*30)
358     lineExample(myOLED)
359     print("-"*30)
360     shapeExample(myOLED)
361     print("-"*30)
362     textExamples(myOLED)
363     print("-"*30)
364     print("DONE")
365
366     #-----
367
368 if __name__ == '__main__':
369     try:
370         runExample()
371     except (KeyboardInterrupt, SystemExit) as exErr:
372         print("\nEnding OLED Everything Example")
373         sys.exit(0)

```

4.5 Basic Operation

Listing 4: examples/qwiic_micro_oled_hello.py

```

1  #!/usr/bin/env python
2  #-----
3  # qwiic_micro_oled_hello.py
4  #
5  # Simple Example for the Qwiic MicroOLED Device
6  #-----
7  #
8  # Written by SparkFun Electronics, May 2021
9  #
10 # This python library supports the SparkFun Electroncis qwiic
11 # qwiic sensor/board ecosystem on a Raspberry Pi (and compatable) single
12 # board computers.

```

(continues on next page)

(continued from previous page)

```

13 #
14 # More information on qwiic is at https:# www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun. Buy a board!
17 #
18 #=====
19 # Copyright (c) 2021 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining a copy
22 # of this software and associated documentation files (the "Software"), to deal
23 # in the Software without restriction, including without limitation the rights
24 # to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
25 # copies of the Software, and to permit persons to whom the Software is
26 # furnished to do so, subject to the following conditions:
27 #
28 # The above copyright notice and this permission notice shall be included in all
29 # copies or substantial portions of the Software.
30 #
31 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
32 # IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
33 # FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
34 # AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
35 # LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
36 # OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
37 # SOFTWARE.
38 #=====
39 # Example - simple command to setup the OLED.
40 #
41
42 from __future__ import print_function
43 import qwiic_micro_oled
44 import sys
45 import time
46
47
48 def runExample():
49
50     # These three lines of code are all you need to initialize the
51     # OLED and print the splash screen.
52
53     # Before you can start using the OLED, call begin() to init
54     # all of the pins and configure the OLED.
55
56
57     print("\nSparkFun Micro OLED Hello Example\n")
58     myOLED = qwiic_micro_oled.QwiicMicroOled()
59
60     if not myOLED.connected:
61         print("The Qwiic Micro OLED device isn't connected to the system. Please_
↪check your connection", \
62             file=sys.stderr)
63         return
64
65     myOLED.begin()
66     # clear(ALL) will clear out the OLED's graphic memory.
67     # clear(PAGE) will clear the Arduino's display buffer.
68     myOLED.clear(myOLED.ALL) # Clear the display's memory (gets rid of artifacts)

```

(continues on next page)

(continued from previous page)

```
69     # To actually draw anything on the display, you must call the
70     # display() function.
71     myOLED.display()
72
73     time.sleep(2)
74
75     myOLED.clear(myOLED.PAGE) # Clear the display's buffer
76
77     myOLED.print("Hello World") # Add "Hello World" to buffer
78
79     # To actually draw anything on the display, you must call the display() function.
80     ↪ myOLED.display()
81
82
83
84 if __name__ == '__main__':
85     try:
86         runExample()
87     except (KeyboardInterrupt, SystemExit) as exErr:
88         print("\nEnding OLED Hello Example")
89         sys.exit(0)
```


CHAPTER 5

Indices and tables

- `genindex`
- `modindex`
- `search`

q

qwiic_micro_oled, 9

B

`begin()` (*qwiic_micro_oled.QwiicMicroOled* method), 9

C

`circle()` (*qwiic_micro_oled.QwiicMicroOled* method), 9

`clear()` (*qwiic_micro_oled.QwiicMicroOled* method), 10

`connected` (*qwiic_micro_oled.QwiicMicroOled* attribute), 10

`contrast()` (*qwiic_micro_oled.QwiicMicroOled* method), 10

D

`display()` (*qwiic_micro_oled.QwiicMicroOled* method), 10

`draw_bitmap()` (*qwiic_micro_oled.QwiicMicroOled* method), 10

`draw_char()` (*qwiic_micro_oled.QwiicMicroOled* method), 10

F

`flip_horizontal()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`flip_vertical()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`font_height` (*qwiic_micro_oled.QwiicMicroOled* attribute), 11

`font_type` (*qwiic_micro_oled.QwiicMicroOled* attribute), 11

`font_width` (*qwiic_micro_oled.QwiicMicroOled* attribute), 11

G

`get_font_height()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`get_font_start_char()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`get_font_total_char()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`get_font_type()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`get_font_width()` (*qwiic_micro_oled.QwiicMicroOled* method), 11

`get_lcd_height()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

`get_lcd_width()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

`get_screenbuffer()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

`get_total_fonts()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

H

`height` (*qwiic_micro_oled.QwiicMicroOled* attribute), 12

I

`invert()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

`is_connected()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

L

`line()` (*qwiic_micro_oled.QwiicMicroOled* method), 12

`line_h()` (*qwiic_micro_oled.QwiicMicroOled* method), 13

`line_v()` (*qwiic_micro_oled.QwiicMicroOled* method), 13

P

pixel() (*qwiic_micro_oled.QwiicMicroOled* method),
13

print() (*qwiic_micro_oled.QwiicMicroOled* method),
13

Q

qwiic_micro_oled (*module*), 9

QwiicMicroOled (*class in qwiic_micro_oled*), 9

R

rect() (*qwiic_micro_oled.QwiicMicroOled* method),
13

rect_fill() (*qwiic_micro_oled.QwiicMicroOled*
method), 14

S

scroll_left() (*qwiic_micro_oled.QwiicMicroOled*
method), 14

scroll_right() (*qwiic_micro_oled.QwiicMicroOled*
method), 14

scroll_stop() (*qwiic_micro_oled.QwiicMicroOled*
method), 14

set_color() (*qwiic_micro_oled.QwiicMicroOled*
method), 15

set_column_address()
(*qwiic_micro_oled.QwiicMicroOled* method),
15

set_cursor() (*qwiic_micro_oled.QwiicMicroOled*
method), 15

set_draw_mode() (*qwiic_micro_oled.QwiicMicroOled*
method), 15

set_font_type() (*qwiic_micro_oled.QwiicMicroOled*
method), 15

set_page_address()
(*qwiic_micro_oled.QwiicMicroOled* method),
15

W

width (*qwiic_micro_oled.QwiicMicroOled* attribute),
15

write() (*qwiic_micro_oled.QwiicMicroOled* method),
15