
Adafruit HT16K33 Library Documentation

Release 1.0

Radomir Dopieralski

Jan 14, 2020

Contents

1	Dependencies	3
2	Installing from PyPI	5
3	Usage Example	7
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Simple test	13
6.2	adafruit_ht16k33.ht16k33	16
6.3	Matrix Displays	16
6.4	Segment Displays	16
7	Indices and tables	19
	Python Module Index	21
	Index	23

This is a library for using the I²C-based LED matrices with the HT16K33 chip. It supports both 16x8 and 8x8 matrices, as well as 7- and 14-segment displays.

- **Notes**

1. This library is intended for Adafruit CircuitPython's API. For a library compatible with MicroPython machine API see this [library](#).
2. This library does not work with the Trellis 4x4 LED+Keypad board. For that product use: [CircuitPython Trellis Library](#)

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-ht16k33
```

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

```
sudo pip3 install adafruit-circuitpython-ht16k33
```

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-ht16k33
```


CHAPTER 3

Usage Example

```
# Import all board pins and bus interface.
import board
import busio

# Import the HT16K33 LED matrix module.
from adafruit_ht16k33 import matrix

# Create the I2C interface.
i2c = busio.I2C(board.SCL, board.SDA)

# Create the matrix class.
# This creates a 16x8 matrix:
matrix = matrix.Matrix16x8(i2c)
# Or this creates a 8x8 matrix:
#matrix = matrix.Matrix8x8(i2c)
# Or this creates a 8x8 bicolor matrix:
#matrix = matrix.Matrix8x8x2
# Finally you can optionally specify a custom I2C address of the HT16k33 like:
#matrix = matrix.Matrix16x8(i2c, address=0x70)

# Clear the matrix.
matrix.fill(0)

# Set a pixel in the origin 0,0 position.
matrix[0, 0] = 1
# Set a pixel in the middle 8, 4 position.
matrix[8, 4] = 1
# Set a pixel in the opposite 15, 7 position.
matrix[15, 7] = 1
matrix.show()

# Change the brightness
matrix.brightness = 8
```

(continues on next page)

(continued from previous page)

```
# Set the blink rate  
matrix.blink_rate = 2
```

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](#).

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/ht16k33_matrix_simpletest.py

```
1  # Basic example of clearing and drawing a pixel on a LED matrix display.
2  # This example and library is meant to work with Adafruit CircuitPython API.
3  # Author: Tony DiCola
4  # License: Public Domain
5
6  # Import all board pins.
7  import board
8  import busio
9
10 # Import the HT16K33 LED matrix module.
11 from adafruit_ht16k33 import matrix
12
13
14 # Create the I2C interface.
15 i2c = busio.I2C(board.SCL, board.SDA)
16
17 # Create the matrix class.
18 # This creates a 16x8 matrix:
19 matrix = matrix.Matrix16x8(i2c)
20 # Or this creates a 8x8 matrix:
21 #matrix = matrix.Matrix8x8(i2c)
22 # Or this creates a 8x8 bicolor matrix:
23 #matrix = matrix.Matrix8x8x2(i2c)
24 # Finally you can optionally specify a custom I2C address of the HT16k33 like:
25 #matrix = matrix.Matrix16x8(i2c, address=0x70)
26
27 # Clear the matrix.
```

(continues on next page)

(continued from previous page)

```

28 matrix.fill(0)
29
30 # Set a pixel in the origin 0,0 position.
31 matrix[0, 0] = 1
32 # Set a pixel in the middle 8, 4 position.
33 matrix[8, 4] = 1
34 # Set a pixel in the opposite 15, 7 position.
35 matrix[15, 7] = 1

```

Listing 2: examples/ht16k33_segments_simpletest.py

```

1  # Basic example of setting digits on a LED segment display.
2  # This example and library is meant to work with Adafruit CircuitPython API.
3  # Author: Tony DiCola
4  # License: Public Domain
5
6  import time
7
8  # Import all board pins.
9  import board
10 import busio
11
12 # Import the HT16K33 LED segment module.
13 from adafruit_ht16k33 import segments
14
15 # Create the I2C interface.
16 i2c = busio.I2C(board.SCL, board.SDA)
17
18 # Create the LED segment class.
19 # This creates a 7 segment 4 character display:
20 display = segments.Seg7x4(i2c)
21 # Or this creates a 14 segment alphanumeric 4 character display:
22 #display = segments.Seg14x4(i2c)
23 # Finally you can optionally specify a custom I2C address of the HT16k33 like:
24 #display = segments.Seg7x4(i2c, address=0x70)
25
26 # Clear the display.
27 display.fill(0)
28
29 # Can just print a number
30 display.print(42)
31 time.sleep(2)
32
33 # Or, can set individual digits / characters
34 # Set the first character to '1':
35 display[0] = '1'
36 # Set the second character to '2':
37 display[1] = '2'
38 # Set the third character to 'A':
39 display[2] = 'A'
40 # Set the forth character to 'B':
41 display[3] = 'B'
42 time.sleep(2)
43
44 # Or, can even set the segments to make up characters
45 if isinstance(display, segments.Seg7x4):

```

(continues on next page)

(continued from previous page)

```

46     # 7-segment raw digits
47     display.set_digit_raw(0, 0xFF)
48     display.set_digit_raw(1, 0b11111111)
49     display.set_digit_raw(2, 0x79)
50     display.set_digit_raw(3, 0b01111001)
51 else:
52     # 14-segment raw digits
53     display.set_digit_raw(0, 0x3F2D)
54     display.set_digit_raw(1, 0b0011111100101101)
55     display.set_digit_raw(2, (0b00111111, 0b00101101))
56     display.set_digit_raw(3, [0b00111111, 0b00101101])

```

Listing 3: examples/ht16k33_bicolor24_simpletest.py

```

1  # Basic example of using the Bi-color 24 segment bargraph display.
2  # This example and library is meant to work with Adafruit CircuitPython API.
3  # Author: Carter Nelson
4  # License: Public Domain
5
6  import time
7
8  # Import board related modules
9  import board
10 import busio
11
12 # Import the Bicolor24 driver from the HT16K33 module
13 from adafruit_ht16k33.bargraph import Bicolor24
14
15 # Create the I2C interface
16 i2c = busio.I2C(board.SCL, board.SDA)
17
18 # Create the LED bargraph class.
19 bc24 = Bicolor24(i2c)
20
21 # Set individual segments of bargraph
22 bc24[0] = bc24.LED_RED
23 bc24[1] = bc24.LED_GREEN
24 bc24[2] = bc24.LED_YELLOW
25
26 time.sleep(2)
27
28 # Turn them all off
29 bc24.fill(bc24.LED_OFF)
30
31 # Turn them on in a loop
32 for i in range(24):
33     bc24[i] = bc24.LED_RED
34     time.sleep(0.1)
35     bc24[i] = bc24.LED_OFF
36
37 time.sleep(1)
38
39 # Fill the entire bargraph
40 bc24.fill(bc24.LED_GREEN)

```

6.2 adafruit_ht16k33.ht16k33

- Authors: Radomir Dopieralski & Tony DiCola for Adafruit Industries

class adafruit_ht16k33.ht16k33.**HT16K33** (*i2c*, *address=112*, *auto_write=True*)

The base class for all displays. Contains common methods.

Parameters

- **address** (*int*) – The I2C address of the HT16K33.
- **auto_write** (*bool*) – True if the display should immediately change when set. If False, *show* must be called explicitly.

auto_write

Auto write updates to the display.

blink_rate

The blink rate. Range 0-3.

brightness

The brightness. Range 0-15.

fill (*color*)

Fill the whole display with the given color.

show ()

Refresh the display and show the changes.

6.3 Matrix Displays

class adafruit_ht16k33.matrix.**Matrix16x8** (*i2c*, *address=112*, *auto_write=True*)

A double matrix or the matrix wing.

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

Get or set the color of a given pixel.

class adafruit_ht16k33.matrix.**Matrix8x8** (*i2c*, *address=112*, *auto_write=True*)

A single matrix.

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

Get or set the color of a given pixel.

class adafruit_ht16k33.matrix.**Matrix8x8x2** (*i2c*, *address=112*, *auto_write=True*)

A bi-color matrix.

fill (*color*)

Fill the whole display with the given color.

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

Get or set the color of a given pixel.

6.4 Segment Displays

class adafruit_ht16k33.segments.**BigSeg7x4** (*i2c*, *address=112*, *auto_write=True*)

Numeric 7-segment display. It has the same methods as the alphanumeric display, but only supports displaying a limited set of characters.

ampm
The AM/PM indicator.

bottom_left_dot
The bottom-left dot indicator.

top_left_dot
The top-left dot indicator.

class adafruit_ht16k33.segments.**Colon** (*disp, num_of_colons=1*)
Helper class for controlling the colons. Not intended for direct use.

class adafruit_ht16k33.segments.**Seg14x4** (*i2c, address=112, auto_write=True*)
Alpha-numeric, 14-segment display.

print (*value*)
Print the value to the display.

scroll (*count=1*)
Scroll the display by specified number of places.

set_digit_raw (*index, bitmask*)
Set digit at position to raw bitmask value. Position should be a value of 0 to 3 with 0 being the left most character on the display.

bitmask should be 2 bytes such as: 0xFFFF If can be passed as an integer, list, or tuple

class adafruit_ht16k33.segments.**Seg7x4** (*i2c, address=112, auto_write=True*)
Numeric 7-segment display. It has the same methods as the alphanumeric display, but only supports displaying a limited set of characters.

scroll (*count=1*)
Scroll the display by specified number of places.

set_digit_raw (*index, bitmask*)
Set digit at position to raw bitmask value. Position should be a value of 0 to 3 with 0 being the left most digit on the display.

CHAPTER 7

Indices and tables

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

a

`adafruit_ht16k33.ht16k33`, [15](#)
`adafruit_ht16k33.matrix`, [16](#)
`adafruit_ht16k33.segments`, [16](#)

A

adafruit_ht16k33.ht16k33 (module), 15
adafruit_ht16k33.matrix (module), 16
adafruit_ht16k33.segments (module), 16
ampm (adafruit_ht16k33.segments.BigSeg7x4 attribute), 16
auto_write (adafruit_ht16k33.ht16k33.HT16K33 attribute), 16

B

BigSeg7x4 (class in adafruit_ht16k33.segments), 16
blink_rate (adafruit_ht16k33.ht16k33.HT16K33 attribute), 16
bottom_left_dot (adafruit_ht16k33.segments.BigSeg7x4 attribute), 17
brightness (adafruit_ht16k33.ht16k33.HT16K33 attribute), 16

C

Colon (class in adafruit_ht16k33.segments), 17

F

fill() (adafruit_ht16k33.ht16k33.HT16K33 method), 16
fill() (adafruit_ht16k33.matrix.Matrix8x8x2 method), 16

H

HT16K33 (class in adafruit_ht16k33.ht16k33), 16

M

Matrix16x8 (class in adafruit_ht16k33.matrix), 16
Matrix8x8 (class in adafruit_ht16k33.matrix), 16
Matrix8x8x2 (class in adafruit_ht16k33.matrix), 16

P

pixel() (adafruit_ht16k33.matrix.Matrix16x8 method), 16

pixel() (adafruit_ht16k33.matrix.Matrix8x8 method), 16

pixel() (adafruit_ht16k33.matrix.Matrix8x8x2 method), 16

print() (adafruit_ht16k33.segments.Seg14x4 method), 17

S

scroll() (adafruit_ht16k33.segments.Seg14x4 method), 17

scroll() (adafruit_ht16k33.segments.Seg7x4 method), 17

Seg14x4 (class in adafruit_ht16k33.segments), 17

Seg7x4 (class in adafruit_ht16k33.segments), 17

set_digit_raw() (adafruit_ht16k33.segments.Seg14x4 method), 17

set_digit_raw() (adafruit_ht16k33.segments.Seg7x4 method), 17

show() (adafruit_ht16k33.ht16k33.HT16K33 method), 16

T

top_left_dot (adafruit_ht16k33.segments.BigSeg7x4 attribute), 17