Adafruit HT16K33 Library Documentation

Release 1.0

Radomir Dopieralski

Feb 06, 2019

Contents

This is a library for using the I^2C-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

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 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-ht16k33 --
↪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/matrix.py

```
# Basic example of clearing and drawing a pixel on a LED matrix display.
# This example and library is meant to work with Adafruit CircuitPython API.
# Author: Tony DiCola
# License: Public Domain

# Import all board pins.
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(i2c)
# Finally you can optionally specify a custom I2C address of the HT16k33 like:
#matrix = matrix.Matrix16x8(i2c, address=0x70)

# Clear the matrix.
```

(continues on next page)

(continued from previous page)

```
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
```

Listing 2: examples/segments.py

```
# Basic example of setting digits on a LED segment display.
# This example and library is meant to work with Adafruit CircuitPython API.
# Author: Tony DiCola
# License: Public Domain

import time

# Import all board pins.
import board
import busio

# Import the HT16K33 LED segment module.
from adafruit_ht16k33 import segments


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

# Create the LED segment class.
# This creates a 7 segment 4 character display:
display = segments.Seg7x4(i2c)
# Or this creates a 14 segment alphanumeric 4 character display:
#display = segments.Seg14x4(i2c)
# Finally you can optionally specify a custom I2C address of the HT16k33 like:
#display = segments.Seg7x4(i2c, address=0x70)

# Clear the display.
display.fill(0)

# Can just print a number
display.print(42)
time.sleep(2)

# Or, can set indivdual digits / characters
# Set the first character to '1':
display[0] = '1'
# Set the second character to '2':
display[1] = '2'
# Set the third character to 'A':
display[2] = 'A'
# Set the forth character to 'B':
display[3] = 'B'
```

5.2 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 addess 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.

5.3 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.

5.4 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.

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.

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.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index