
sparkfun_eeeprom_py

Release 0.0.1

SparkFun Electronics

Jul 26, 2021

CONTENTS:

- 1 Contents 3**
- 2 Supported Platforms 5**
- 3 Dependencies 7**
- 4 Documentation 9**
- 5 Installation 11**
 - 5.1 PyPi Installation 11
- 6 Example Use 13**
- 7 Table of Contents 15**
 - 7.1 API Reference 15
 - 7.1.1 qwiic_eeprom 15
 - 7.2 Example One - Basic Read Write 19
 - 7.3 Example Two - Settings 21
 - 7.4 Example Three - Advanced I2C 22
 - 7.5 Example Four - User Options 24
 - 7.6 Example Five - Interface Test 27
- 8 Indices and tables 33**
- Python Module Index 35**
- Index 37**

Python module for the [SparkFun Qwiic EEPROM Breakout - 512Kbit](#)

This python package is a port of the existing [SparkFun External EEPROM 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](#).

CONTENTS

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

SUPPORTED PLATFORMS

The Qwiic EEPROM Python package currently supports the following platforms:

- [Raspberry Pi](#)

DEPENDENCIES

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

DOCUMENTATION

The SparkFun Qwiic EEPROM module documentation is hosted at [ReadTheDocs](#)

INSTALLATION

5.1 PyPi Installation

This repository is hosted on PyPi as the [sparkfun-qwiic-eeeprom](#) 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-eeeprom
```

For the current user:

```
pip install sparkfun-qwiic-eeeprom
```

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-qwiic-eeeprom-<version>.tar.gz
```


EXAMPLE USE

See the examples directory for more detailed use examples.

```
from __future__ import print_function
import qwiic_eeprom
import time
import sys

def run_example():

    print("\nSparkFun Qwiic EEPROM, Example 1\n")
    my_eeprom = qwiic_eeprom.QwiicEEPROM()

    if my_eeprom.begin() != True:
        print("\nThe Qwiic EEPROM isn't connected to the system. Please check your_
↪connection", \
            file=sys.stderr)
        return

    print("\nEEPROM ready!")

    print("\nMem size in bytes: " + str(my_eeprom.length()))

    my_value1 = 200 # 0xC8
    my_eeprom.write_byte(300, my_value1) # (location = 0x12C, data byte)
    my_read1 = my_eeprom.read_byte(300) # (location = 0x12C)
    print("\nI read: " + str(my_read1)) # Should be 0xC8 = 200

    my_value2 = -366
    my_eeprom.write_int(10, my_value2)
    my_read2 = my_eeprom.read_int(10) # (location)
    print("\nI read: " + str(my_read2))

    my_value3 = -7.35
    my_eeprom.write_float(9, my_value3)
    my_read3 = my_eeprom.read_float(9) # (location)
    # Round to 2-decimal point precision
    my_read3 = round(my_read3, 2)
    print("\nI read: " + str(my_read3))

if __name__ == '__main__':
```

(continues on next page)

(continued from previous page)

```
try:
    run_example()
except (KeyboardInterrupt, SystemExit) as exErr:
    print("\nEnding Example 1")
    sys.exit(0)
```

TABLE OF CONTENTS

7.1 API Reference

7.1.1 qwiic_eeprom

Python module for the SparkFun Qwiic EEPROM Breakout - 512Kbit.

This package is a port of the existing [SparkFun External EEPROM Arduino Library](https://github.com/sparkfun/SparkFun_External_EEPROM_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_eeprom.**QwiicEEPROM**(*address=None, i2c_driver=None*)
Qwiic EEPROM

param address The I2C address to use for the device. If not provided, the default address is used.

param i2c_driver An existing i2c driver object. If not provided a driver object is created.

return The GPIO device object.

rtype Object

begin()

Initialize the operation of the Qwiic EEPROM. Run `is_connected()`.

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

Return type bool

disable_poll_for_write_complete()

Disable polling of when a write has completed

Returns Nothing

Return type Void

enable_poll_for_write_complete()

Enable I2C polling of when a write has completed

Returns Nothing

Return type Void

erase(*to_write=0*)

Erase entire EEPROM.

Parameters `to_write` – byte to write into each spot of EEPROM

Returns Nothing

Return type void

get_I2C_buffer_size()

Return the size of the TX buffer

Returns I2C_BUFFER_LENGTH_TX

Return type int

get_memory_size()

Return the size of EEPROM

Returns memory_size_bytes

Return type int

get_page_size()

Get the page size

Returns Current page size off EEPROM in bytes

Return type int

get_page_write_time()

Get the current time required per page write

Returns Time required per page write

Return type int

is_busy(*i2c_address=255*)

Returns true if the device is not answering (currently writing).

Parameters `i2c_address` – I2C address of EEPROM. Larger EEPROMs have two addresses.

Returns True if the IC is busy, false otherwise

Return type bool

is_connected(*i2c_address=255*)

Determine if a Qwiic EEPROM device is connected to the system

Parameters `i2c_address` – I2C address of EEPROM. Larger EEPROMs have two addresses.

Returns True if the device is connected, false otherwise.

Return type bool

length()

Returns the memory size of the EEPROM

Returns memory_size_bytes

Return type int

read(*eeprom_location, num_bytes*)

Bulk read from EEPROM. Handles breaking up read amt into 32 byte chunks (can be overridden with `set_I2C_buffer_size()`) Handles a read that straddles the 512kbit barrier

Parameters

- **eeprom_location** – address of EEPROM to start reading from
- **num_bytes** – number of bytes to be read from external EEPROM

Returns a list of bytes read from EEPROM

Return type list

read_byte(*eeprom_location*)

Read exactly one byte from EEPROM at a given address location

Parameters **eeprom_location** – location in EEPROM to read byte from

Returns byte read from EEPROM

Return type byte

read_float(*eeprom_location*)

Read a 32-bit float from a given EEPROM location

Parameters **eeprom_location** – location in EEPROM to read float from

Returns float read from EEPROM

Return type float

read_int(*eeprom_location*)

Read a 32-bit signed int from a given EEPROM location

Parameters **eeprom_location** – location in EEPROM to read int from

Returns int read from EEPROM

Return type int

read_string(*eeprom_location*, *string_length*)

Read a string of given length from any address of EEPROM

Param **eeprom_location**: location in EEPROM to read string from

Parameters **string_length** – number of chars to read from EEPROM

Returns string read from EEPROM

Return type string

set_I2C_buffer_size(*buff_size*)

Set the size of the TX buffer

Parameters **buff_size** – the size of the I2C buffer

Returns nothing

Return type Void

set_memory_size(*mem_size*)

Set the size of memory in bytes

Parameters **mem_size** – memory size in bytes

Returns Nothing

Return type void

set_page_size(*page_size*)

Set the size of the page we can write at a time

Parameters **page_size** – new page size in bytes

Returns Nothing

Return type void

set_page_write_time(*write_time_ms*)

Set the number of ms required per page write

Parameters **write_time_ms** – write time in ms

Returns Nothing

Return type Void

write(*eeprom_location*, *data_list*)

Write large bulk amounts to EEPROM. Limits write to the I2C buffer size (default is 32 bytes).

Parameters

- **eeprom_location** – 2-byte EEPROM address to write to
- **data_list** – list of data bytes to be written to EEPROM sequentially, starting at the EEPROM address

Return type Void

Returns nothing

write_byte(*eeprom_location*, *byte_to_write*)

Write a single byte to given EEPROM location

Parameters

- **eeprom_location** – location in EEPROM to byte to
- **byte_to_write** – byte to write to EEPROM

Returns Nothing

Return type Void

write_float(*eeprom_location*, *float_to_write*)

Write a 32-bit float to a given EEPROM location

Parameters

- **eeprom_location** – location in EEPROM to write float to
- **float_to_write** – float to write to EEPROM

Returns Nothing

Return type Void

write_int(*eeprom_location*, *int_to_write*)

Write a signed 32-bit int to a given EEPROM location

Parameters

- **eeprom_location** – location in EEPROM to write int to
- **int_to_write** – int to write to EEPROM

Returns Nothing

Return type Void

write_string(*eeprom_location*, *string_to_write*)

Write a string to a given EEPROM location

Parameters

- **eeprom_location** – location in EEPROM to write string to

- **string_to_write** – string to write to EEPROM

Returns Nothing

Return type Void

7.2 Example One - Basic Read Write

Listing 1: examples/qwiic_eeprom_ex1_basic_read_write.py

```

1  #!/usr/bin/env python
2  # -----
3  # qwiic_eeprom_ex1_basic_read_write.py
4  #
5  # This example demonstrates how to read and write various variables to memory.
6  # -----
7  #
8  # Written by Priyanka Makin @ SparkFun Electronics, July 2021
9  #
10 # This python library supports the SparkFun Electronics qwiic sensor/
11 # board ecosystem on a Raspberry Pi (and compatable) single board
12 # computers.
13 #
14 # More information on qwiic is at https://www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun by buying a board!
17 #
18 # =====
19 # Copyright (c) 2021 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining
22 # a copy of this software and associated documentation files (the
23 # "Software"), to deal in the Software without restriction, including
24 # without limitation the rights to use, copy, modify, merge, publish,
25 # distribute, sublicense, and/or sell copies of the Software, and to
26 # permit persons to whom the Software is furnished to do so, subject to
27 # the following conditions:
28 #
29 # The above copyright notice and this permission notice shall be
30 # included in all copies or substantial portions of the Software.
31 #
32 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
33 # EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
34 # MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
35 # IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
36 # CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
37 # TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
38 # SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
39 # =====
40 # Example 1
41
42 from __future__ import print_function

```

(continues on next page)

(continued from previous page)

```

43 import qwiic_eeprom
44 import time
45 import sys
46
47 def run_example():
48
49     print("\nSparkFun Qwiic EEPROM, Example 1\n")
50     my_eeprom = qwiic_eeprom.QwiicEEPROM()
51
52     if my_eeprom.begin() != True:
53         print("\nThe Qwiic EEPROM isn't connected to the system. Please check your_
↪connection", \
54             file=sys.stderr)
55         return
56
57     print("\nEEPROM ready!")
58
59     print("\nMem size in bytes: " + str(my_eeprom.length()))
60
61     my_value1 = 200 # 0xC8
62     my_eeprom.write_byte(300, my_value1) # (location = 0x12C, data byte)
63     my_read1 = my_eeprom.read_byte(300) # (location = 0x12C)
64     print("\nI read: " + str(my_read1)) # Should be 0xC8 = 200
65
66     my_value2 = -366
67     my_eeprom.write_int(10, my_value2)
68     my_read2 = my_eeprom.read_int(10) # (location)
69     print("\nI read: " + str(my_read2))
70
71     my_value3 = -7.35
72     my_eeprom.write_float(9, my_value3)
73     my_read3 = my_eeprom.read_float(9) # (location)
74     # Round to 2-decimal point precision
75     my_read3 = round(my_read3, 2)
76     print("\nI read: " + str(my_read3))
77
78 if __name__ == '__main__':
79     try:
80         run_example()
81     except (KeyboardInterrupt, SystemExit) as exErr:
82         print("\nEnding Example 1")
83         sys.exit(0)
84

```


7.3 Example Two - Settings

Listing 2: examples/qwiic_eeprom_ex2_settings.py

```

1  #!/usr/bin/env python
2  # -----
3  # qwiic_eeprom_ex2_settings.py
4  #
5  # This example demonstrates how to set the various settings for a given EEPROM.
6  # Read the datasheet! Each EEPROM will have specific values for:
7  # Overall EEPROM size in bytes (512kbit = 64000, 256kbit = 32000)
8  # Bytes per page write (64 and 128 are common)
9  # Whether write polling is supported
10 # -----
11 #
12 # Written by Priyanka Makin @ SparkFun Electronics, July 2021
13 #
14 # This python library supports the SparkFun Electronics qwiic sensor/
15 # board ecosystem on a Raspberry Pi (and compatible) single board
16 # computers.
17 #
18 # More information on qwiic is at https://www.sparkfun.com/qwiic
19 #
20 # Do you like this library? Help support SparkFun by buying a board!
21 #
22 # =====
23 # Copyright (c) 2021 SparkFun Electronics
24 #
25 # Permission is hereby granted, free of charge, to any person obtaining
26 # a copy of this software and associated documentation files (the
27 # "Software"), to deal in the Software without restriction, including
28 # without limitation the rights to use, copy, modify, merge, publish,
29 # distribute, sublicense, and/or sell copies of the Software, and to
30 # permit persons to whom the Software is furnished to do so, subject to
31 # the following conditions:
32 #
33 # The above copyright notice and this permission notice shall be
34 # included in all copies or substantial portions of the Software.
35 #
36 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
37 # EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
38 # MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
39 # IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
40 # CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
41 # TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
42 # SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
43 # =====
44 # Example 2
45
46 from __future__ import print_function
47 import qwiic_eeprom
48 import time

```

(continues on next page)

(continued from previous page)

```

49 import sys
50
51 def run_example():
52
53     print("\nSparkFun Qwiic EEPROM, Example 2\n")
54     my_eeprom = qwiic_eeprom.QwiicEEPROM()
55
56     if my_eeprom.begin() != True:
57         print("\nThe Qwiic EEPROM isn't connected to the system. Please check your_
↪connection", \
58             file=sys.stderr)
59         return
60
61     print("\nEEPROM ready!")
62
63     # Set settings for this EEPROM
64     my_eeprom.set_memory_size(512000/8) # In bytes. 512kbit = 64kbyte
65     my_eeprom.set_page_size(128)      # in bytes. Has 128 byte page size.
66     my_eeprom.disable_poll_for_write_complete() # Supports I2C polling of write_
↪completion
67     my_eeprom.set_page_write_time(3)    # 3 ms max write time
68
69     print("\nMem size in bytes: " + str(my_eeprom.get_memory_size()))
70     print("\nPage size in bytes: " + str(my_eeprom.get_page_size()))
71
72     my_value = -7.35
73     my_eeprom.write_float(20, my_value) # (location, data)
74     my_read = my_eeprom.read_float(20)  # (location)
75     # Round to 2-decimal point precision
76     my_read = round(my_read, 2)
77     print("\nI read: " + str(my_read))
78
79 if __name__ == '__main__':
80     try:
81         run_example()
82     except (KeyboardInterrupt, SystemExit) as exErr:
83         print("\nEnding Example 1")
84         sys.exit(0)

```

7.4 Example Three - Advanced I2C

Listing 3: examples/qwiic_eeprom_ex3_advanced_i2c.py

```

1  #!/usr/bin/env python
2  # -----
3  # qwiic_eeprom_ex3_advanced_i2c.py
4  #
5  # This example demonstrates how to pass a custom EEPROM address.
6  # -----
7  #

```

(continues on next page)

(continued from previous page)

```

8  # Written by Priyanka Makin @ SparkFun Electronics, July 2021
9  #
10 # This python library supports the SparkFun Electronics qwiic sensor/
11 # board ecosystem on a Raspberry Pi (and compatable) single board
12 # computers.
13 #
14 # More information on qwiic is at https://www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun by buying a board!
17 #
18 # =====
19 # Copyright (c) 2021 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining
22 # a copy of this software and associated documentation files (the
23 # "Software"), to deal in the Software without restriction, including
24 # without limitation the rights to use, copy, modify, merge, publish,
25 # distribute, sublicense, and/or sell copies of the Software, and to
26 # permit persons to whom the Software is furnished to do so, subject to
27 # the following conditions:
28 #
29 # The above copyright notice and this permission notice shall be
30 # included in all copies or substantial portions of the Software.
31 #
32 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
33 # EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
34 # MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
35 # IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
36 # CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
37 # TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
38 # SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
39 #=====
40 # Example 3
41
42 from __future__ import print_function
43 import qwiic_eeprom
44 import time
45 import sys
46
47 def run_example():
48
49     EEPROM_ADDRESS = 0b1010001 # 0b1010(A2 A1 A0): A standard I2C EEPROM with the ADRO_
    ↪ bit set to VCC
50
51     print("\nSparkFun Qwiic EEPROM, Example 3\n")
52     my_eeprom = qwiic_eeprom.QwiicEEPROM(0x51)
53
54     if my_eeprom.begin() != True:
55         print("\nThe Qwiic EEPROM isn't connected to the system. Please check your_
    ↪ connection", \
56             file=sys.stderr)
57         return

```

(continues on next page)

(continued from previous page)

```

58
59     print("\nEEPROM ready!")
60
61     my_value = -7.35
62     my_eeprom.write_float(20, my_value)    # (location, data)
63     my_read = my_eeprom.read_float(20)    # (location to read)
64     # Round to 2-decimal point precision
65     my_read = round(my_read, 2)
66     print("\nI read: " + str(my_read))
67
68 if __name__ == '__main__':
69     try:
70         run_example()
71     except (KeyboardInterrupt, SystemExit) as exErr:
72         print("\nEnding Example 1")
73         sys.exit(0)

```

7.5 Example Four - User Options

Listing 4: examples/qwiic_eeprom_ex4_user_options.py

```

1  #!/usr/bin/env python
2  # -----
3  # qwiic_eeprom_ex4_user_options.py
4  #
5  # This example demonstrates how to record various user settings easily
6  # to EEPROM.
7  # -----
8  #
9  # Written by Priyanka Makin @ SparkFun Electronics, July 2021
10 #
11 # This python library supports the SparkFun Electronics qwiic sensor/
12 # board ecosystem on a Raspberry Pi (and compatable) single board
13 # computers.
14 #
15 # More information on qwiic is at https://www.sparkfun.com/qwiic
16 #
17 # Do you like this library? Help support SparkFun by buying a board!
18 #
19 # =====
20 # Copyright (c) 2021 SparkFun Electronics
21 #
22 # Permission is hereby granted, free of charge, to any person obtaining
23 # a copy of this software and associated documentation files (the
24 # "Software"), to deal in the Software without restriction, including
25 # without limitation the rights to use, copy, modify, merge, publish,
26 # distribute, sublicense, and/or sell copies of the Software, and to
27 # permit persons to whom the Software is furnished to do so, subject to
28 # the following conditions:
29 #

```

(continues on next page)

(continued from previous page)

```

30 # The above copyright notice and this permission notice shall be
31 # included in all copies or substantial portions of the Software.
32 #
33 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
34 # EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
35 # MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
36 # IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
37 # CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
38 # TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
39 # SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
40 #=====
41 # Example 4
42
43 from __future__ import print_function
44 import qwiic_eeprom
45 import time
46 import sys
47 # TODO: remove this?
48 from collections import namedtuple
49 from dataclasses import *
50
51 LOCATION_SETTINGS = 0
52 SETTINGS_SIZE = 10 # 2 ints + 2 bools = 10 bytes total
53
54 # Load the current settings from EEPROM into the settings struct
55 def load_user_settings(my_mem, settings):
56     # Uncomment these lines to forcibly erase the EEPROM and see how the defaults are set
57     # print("\nErasing EEPROM, be patient please")
58     # my_mem.erase()
59
60     # Check to see if EEPROM is blank. If the first four spots are zeros then we can
61     ↪ assume the EEPROM is blank.
62     num_bytes = 4
63     if (my_mem.read_int(LOCATION_SETTINGS) == 0): # (EEPROM address to read, thing
64     ↪ to read to)
65         # At power on, settings are set to defaults within the tuple
66         # So go record the tuple as it currently exists so that defaults are set
67         record_user_settings(my_mem, settings)
68
69         print("\nDefault settings applied")
70     else:
71
72         settings.baud_rate == my_mem.read_int(0)
73         settings.log_date == my_mem.read_byte(4)
74         settings.enable_IMU == my_mem.read_byte(5)
75         cal_val_temp = my_mem.read_float(6)
76         # Round to 2-decimal point precision
77         settings.cal_value == round(cal_val_temp, 2)
78
79 # Record the current settings into EEPROM
80 def record_user_settings(my_mem, settings):
81     # Use our individual write functions to make writing this class easier

```

(continues on next page)

(continued from previous page)

```

80 my_mem.write_int(0, settings.baud_rate)
81 time.sleep(0.1)
82 my_mem.write_byte(4, settings.log_date)
83 time.sleep(0.1)
84 my_mem.write_byte(5, settings.enable_IMU)
85 time.sleep(0.1)
86 my_mem.write_float(6, settings.cal_value)
87 time.sleep(0.1)
88
89 def run_example():
90
91     print("\nSparkFun Qwiic EEPROM, Example 4\n")
92     my_eeprom = qwiic_eeprom.QwiicEEPROM()
93
94     if my_eeprom.begin() != True:
95         print("\nThe Qwiic EEPROM isn't connected to the syste. Please check your
↪connection", \
96             file=sys.stderr)
97         return
98
99     print("\nEEPROM ready!")
100
101     @dataclass
102     class Settings:
103         baud_rate: int
104         log_date: bool
105         enable_IMU: bool
106         cal_value: float
107
108     default_settings = Settings(115200, False, True, -5.17)
109
110     load_user_settings(my_eeprom, default_settings)
111
112     print("\nBaud rate: " + str(default_settings.baud_rate))
113
114     log_str = ""
115     if default_settings.log_date == True:
116         log_str = "True"
117     else:
118         log_str = "False"
119     print("\nlog_date: " + log_str)
120
121     print("\ncal_value: " + str(default_settings.cal_value))
122
123     # Now we can change something
124     print("\nEnter a new baud rate (1200 to 115200): ")
125     new_baud = input()
126     if int(new_baud) < 1200 or int(new_baud) > 115200:
127         print("\nError: baud rate out of range!")
128     else:
129         default_settings = replace(default_settings, baud_rate = int(new_baud))
130

```

(continues on next page)

(continued from previous page)

```

131     print("\nThis is the new baud rate: ")
132     print(default_settings.baud_rate)
133
134     record_user_settings(my_eeprom, default_settings)
135
136 if __name__ == '__main__':
137     try:
138         run_example()
139     except (KeyboardInterrupt, SystemExit) as exErr:
140         print("\nEnding Example 1")
141         sys.exit(0)
142

```

7.6 Example Five - Interface Test

Listing 5: examples/qwiic_eeprom_ex5_interface_test.py

```

1  #!/usr/bin/env python
2  # -----
3  # qwiic_eeprom_ex5_interface_test.py
4  #
5  # This example demonstrates how to read and write various variables to memory
6  # -----
7  #
8  # Written by Priyanka Makin @ SparkFun Electronics, July 2021
9  #
10 # This python library supports the SparkFun Electronics qwiic sensor/
11 # board ecosystem on a Raspberry Pi (and compatable) single board
12 # computers.
13 #
14 # More information on qwiic is at https://www.sparkfun.com/qwiic
15 #
16 # Do you like this library? Help support SparkFun by buying a board!
17 #
18 # =====
19 # Copyright (c) 2021 SparkFun Electronics
20 #
21 # Permission is hereby granted, free of charge, to any person obtaining
22 # a copy of this software and associated documentation files (the
23 # "Software"), to deal in the Software without restriction, including
24 # without limitation the rights to use, copy, modify, merge, publish,
25 # distribute, sublicense, and/or sell copies of the Software, and to
26 # permit persons to whom the Software is furnished to do so, subject to
27 # the following conditions:
28 #
29 # The above copyright notice and this permission notice shall be
30 # included in all copies or substantial portions of the Software.
31 #
32 # THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
33 # EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF

```

(continues on next page)

(continued from previous page)

```

34 # MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
35 # IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
36 # CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
37 # TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
38 # SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
39 #=====
40 # Example 5
41
42 from __future__ import print_function
43 import qwiic_eeprom
44 import time
45 import sys
46 import random
47
48 def run_example():
49
50     print("\nSparkFun Qwiic EEPROM, Example 5\n")
51     my_eeprom = qwiic_eeprom.QwiicEEPROM()
52
53     if my_eeprom.begin() != True:
54         print("\nThe Qwiic EEPROM isn't connected to the system. Please check your_
↪connection", \
55             file=sys.stderr)
56         return
57
58     print("\nEEPROM ready!")
59
60     my_eeprom.set_memory_size(51200 / 8)    # Qwiic EEPROM is 24512C (512kbit)
61     # my_eeprom.set_page_size(128)
62     # my_eeprom.disable_poll_for_write_complete()
63
64     all_tests_passed = True
65
66     print("\nMem size in bytes: " + str(my_eeprom.length()))
67
68     # Erase test
69     # -----
70     print("\nErasing EEPROM, please be patient!")
71     start_time = time.time()
72     my_eeprom.erase()
73     end_time = time.time()
74     print("\nTime to erase all EEPROM: " + str(end_time - start_time) + "s")
75
76     # Byte sequential test
77     # -----
78     print("\n")
79     print("\n8 bit tests")
80     my_value1 = 200
81     my_value2 = 23
82     random_location = random.randrange(0, my_eeprom.length())
83
84     start_time = time.time()

```

(continues on next page)

(continued from previous page)

```

85 my_eeprom.write_byte(random_location, my_value1)    # (location, data)
86 end_time = time.time()
87 my_eeprom.write_byte(random_location + 1, my_value2)
88 print("\nTime to record byte: " + str(end_time - start_time) + " s")
89
90 start_time = time.time()
91 my_eeprom.write_byte(random_location, my_value1)    # (location, data)
92 end_time = time.time()
93 print("\nTime to write identical byte to same location (should be ~0s): " + str(end_
↪time - start_time) + " s")
94
95 start_time = time.time()
96 response1 = my_eeprom.read_byte(random_location)
97 end_time = time.time()
98 print("\nTime to read byte: " + str(end_time - start_time) + " s")
99
100 response2 = my_eeprom.read_byte(random_location + 1)
101 print("\nLocation " + str(random_location) + " should be " + str(my_value1) + ": " + ↵
↪str(response1))
102 print("\nLocation " + str(random_location + 1) + " should be " + str(my_value2) + ":
↪" + str(response2))
103
104 if my_value1 != response1:
105     all_tests_passed = False
106 if my_value2 != response2:
107     all_tests_passed = False
108
109 # 32 bit test
110 # -----
111 print("\n")
112 print("\n32 bit tests")
113
114 my_value3 = -245000
115 my_value4 = 400123
116 random_location = random.randrange(0, my_eeprom.length())
117
118 start_time = time.time()
119 my_eeprom.write_int(random_location, my_value3)
120 end_time = time.time()
121 print("\nTime to record int32: " + str(end_time - start_time) + " s")
122 my_eeprom.write_int(random_location + 4, my_value4)
123
124 start_time = time.time()
125 response3 = my_eeprom.read_int(random_location)
126 end_time = time.time()
127 print("\nTime to read 32 bits: " + str(end_time - start_time) + " s")
128
129 response4 = my_eeprom.read_int(random_location + 4)
130 print("\nLocation " + str(random_location) + " should be " + str(my_value3) + ": " + ↵
↪str(response3))
131 print("\nLocation " + str(random_location + 4) + " should be " + str(my_value4) + ":
↪" + str(response4))

```

(continues on next page)

(continued from previous page)

```

132
133     if my_value3 != response3:
134         all_tests_passed = False
135     if my_value4 != response4:
136         all_tests_passed = False
137
138     # 32 bit sequential test
139     # -----
140     my_value5 = -341002
141     my_value6 = 241544
142     random_location = random.randrange(0, my_eeprom.length())
143
144     my_eeprom.write_int(random_location, my_value5)
145     my_eeprom.write_int(random_location + 4, my_value6)
146
147     start_time = time.time()
148     response5 = my_eeprom.read_int(random_location)
149     end_time = time.time()
150     print("\nTime to read 32 bits: " + str(end_time - start_time) + " s")
151
152     response6 = my_eeprom.read_int(random_location + 4)
153     print("\nLocation " + str(random_location) + " should be " + str(my_value5) + ": " +
↪str(response5))
154     print("\nLocation " + str(random_location + 4) + " should be " + str(my_value6) + ":
↪" + str(response6))
155
156     if my_value5 != response5:
157         all_tests_passed = False
158     if my_value6 != response6:
159         all_tests_passed = False
160
161     # Float sequential test
162     # -----
163     my_value7 = -7.35
164     my_value8 = 5.22
165     random_location = random.randrange(0, my_eeprom.length())
166
167     my_eeprom.write_float(random_location, my_value7)
168     my_eeprom.write_float(random_location + 4, my_value8)
169
170     start_time = time.time()
171     response7 = my_eeprom.read_float(random_location)
172     end_time = time.time()
173     print("\nTime to read float: " + str(end_time - start_time) + " s")
174
175     response8 = my_eeprom.read_float(random_location + 4)
176
177     # Round floats read to 2-decimal point precision
178     response7 = round(response7, 2)
179     response8 = round(response8, 2)
180
181     print("\nLocation " + str(random_location) + " should be " + str(my_value7) + ": " +
↪str(response7))

```

(continues on next page)

(continued from previous page)

```

182     print("\nLocation " + str(random_location + 4) + " should be " + str(my_value8) + ":
↪ " + str(response8))
183
184     if my_value7 != response7:
185         all_tests_passed = False
186     if my_value8 != response8:
187         all_tests_passed = False
188
189     # 64 bits sequential test
190     # -----
191     print("\n")
192     print("\n64 bit tests")
193
194     # Made up list of 64-bits
195     my_value9 = [0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8]
196     random_location = random.randrange(0, my_eeprom.length())
197
198     start_time = time.time()
199     my_eeprom.write(random_location, my_value9)
200     end_time = time.time()
201     print("\nTime to record 64 bits: " + str(end_time - start_time) + " s")
202
203     start_time = time.time()
204     response9 = my_eeprom.read(random_location, len(my_value9))
205     end_time = time.time()
206     print("\nTime to read 64 bits: " + str(end_time - start_time) + " s")
207
208     print("\nLocation " + str(random_location) + " should be " + str(my_value9) + ": " + ↪
↪ str(response9))
209
210     if my_value9 != response9:
211         all_tests_passed = False
212
213     # Buffer write test
214     # -----
215     print("\n")
216     print("\nBuffer write test")
217
218     my_chars = "Lorem ipsum dolor sit amet, has in verte rem accusamus. Nulla viderer ↪
↪ inciderint eum at."
219     random_location = random.randrange(0, my_eeprom.length() - len(my_chars))
220
221     est_time = len(my_chars) / my_eeprom.get_page_size() * my_eeprom.get_page_write_
↪ time()
222     print("\nCalculated time to record array of " + str(len(my_chars)) + " characters: ~
↪ " + str(est_time) + " ms")
223
224     start_time = time.time()
225     my_eeprom.write_string(random_location, my_chars)
226     end_time = time.time()
227     print("\nTime to record string: " + str(end_time - start_time) + " s")
228

```

(continues on next page)

(continued from previous page)

```
229 start_time = time.time()
230 read_chars = my_eeprom.read_string(random_location, len(my_chars))
231 end_time = time.time()
232 print("\nTime to read string: " + str(end_time - start_time) + " s")
233
234 print("\nLocation " + str(random_location) + " string should read:\n" + my_chars)
235 print("\n" + read_chars)
236
237 if my_chars != read_chars:
238     print("\nString compare failed")
239     all_tests_passed = False
240
241 # # -----
242 if all_tests_passed == True:
243     print("\nAll tests PASSED!")
244 else:
245     print("\nOne or more tests failed. See output")
246
247 if __name__ == '__main__':
248     try:
249         run_example()
250     except (KeyboardInterrupt, SystemExit) as exErr:
251         print("\nEnding Example 1")
252         sys.exit(0)
```

INDICES AND TABLES

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

PYTHON MODULE INDEX

q

`qwiic_eeprom`, 15

INDEX

B

`begin()` (*qwiic_eeprom.QwiicEEPROM* method), 15

D

`disable_poll_for_write_complete()`
(*qwiic_eeprom.QwiicEEPROM* method), 15

E

`enable_poll_for_write_complete()`
(*qwiic_eeprom.QwiicEEPROM* method), 15

`erase()` (*qwiic_eeprom.QwiicEEPROM* method), 15

G

`get_I2C_buffer_size()`
(*qwiic_eeprom.QwiicEEPROM* method), 16

`get_memory_size()` (*qwiic_eeprom.QwiicEEPROM* method), 16

`get_page_size()` (*qwiic_eeprom.QwiicEEPROM* method), 16

`get_page_write_time()`
(*qwiic_eeprom.QwiicEEPROM* method), 16

I

`is_busy()` (*qwiic_eeprom.QwiicEEPROM* method), 16

`is_connected()` (*qwiic_eeprom.QwiicEEPROM* method), 16

L

`length()` (*qwiic_eeprom.QwiicEEPROM* method), 16

M

module
 qwiic_eeprom, 15

Q

qwiic_eeprom
 module, 15

QwiicEEPROM (class in *qwiic_eeprom*), 15

R

`read()` (*qwiic_eeprom.QwiicEEPROM* method), 16

`read_byte()` (*qwiic_eeprom.QwiicEEPROM* method), 17

`read_float()` (*qwiic_eeprom.QwiicEEPROM* method), 17

`read_int()` (*qwiic_eeprom.QwiicEEPROM* method), 17

`read_string()` (*qwiic_eeprom.QwiicEEPROM* method), 17

S

`set_I2C_buffer_size()`
(*qwiic_eeprom.QwiicEEPROM* method), 17

`set_memory_size()` (*qwiic_eeprom.QwiicEEPROM* method), 17

`set_page_size()` (*qwiic_eeprom.QwiicEEPROM* method), 17

`set_page_write_time()`
(*qwiic_eeprom.QwiicEEPROM* method), 17

W

`write()` (*qwiic_eeprom.QwiicEEPROM* method), 18

`write_byte()` (*qwiic_eeprom.QwiicEEPROM* method), 18

`write_float()` (*qwiic_eeprom.QwiicEEPROM* method), 18

`write_int()` (*qwiic_eeprom.QwiicEEPROM* method), 18

`write_string()` (*qwiic_eeprom.QwiicEEPROM* method), 18