AdafruitEMC2101 Library Documentation

Release 1.0

Bryan Siepert

Jun 15, 2021

Contents

CircuitPython driver for EMC2101 brushless fan controller

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device
- Register

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-emc2101
```

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

```
sudo pip3 install adafruit-circuitpython-emc2101
```

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-emc2101
```

CHAPTER 3

Usage Example

```
import time
import board
from adafruit_emc2101 import EMC2101

i2c = board.I2C()  # uses board.SCL and board.SDA

emc = EMC2101(i2c)
print("Setting fan speed to 25%")
emc.manual_fan_speed = 25
time.sleep(2)  # longer sleep to let it spin down from 100%
print("Fan speed", emc.fan_speed)
```

CHAPTER 4

Contributing

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

CHAPTER 5

Documentation

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

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/emc2101_simpletest.py

```
# SPDX-FileCopyrightText: 2020 Bryan Siepert, written for Adafruit Industries
#
# SPDX-License-Identifier: MIT
import time
import board
from adafruit_emc2101 import EMC2101

i2c = board.I2C()  # uses board.SCL and board.SDA
emc = EMC2101(i2c)
while True:
    print("Setting fan speed to 25%")
    emc.manual_fan_speed = 25
    time.sleep(2)  # longer sleep to let it spin down from 100%
    print("Fan speed", emc.fan_speed)
    time.sleep(1)

    print("Setting fan speed to 50%")
    emc.manual_fan_speed = 50
    time.sleep(1.5)
    print("Fan speed", emc.fan_speed)
    time.sleep(1)

    print("Setting fan speed to 75%")
    emc.manual_fan_speed = 75
    time.sleep(1.5)
    print("Fan speed", emc.fan_speed)
    time.sleep(1)
```

(continues on next page)

(continued from previous page)

```

    print("Setting fan speed to 100%")
    emc.manual_fan_speed = 100
    time.sleep(1.5)
    print("Fan speed", emc.fan_speed)
    time.sleep(1)

    print("External temperature:", emc.external_temperature, "C")
    print("Internal temperature:", emc.internal_temperature, "C")

    print("")
    time.sleep(0.5)
```

6.2 LUT Usage Example

Use the temperature to fan speed Look Up Table to automatically control the fan speed. This example requires more memory than the first one because it needs to use the extended *adafruit_emc2101.emc2101_lut.EMC2101_LUT* driver to access LUT functionality.

Listing 2: examples/emc2101_lut_example.py

```
# SPDX-FileCopyrightText: 2020 Bryan Siepert, written for Adafruit Industries
#
# SPDX-License-Identifier: MIT
import time
import board
from adafruit_emc2101.emc2101_lut import EMC2101_LUT as EMC2101

i2c = board.I2C()  # uses board.SCL and board.SDA

FAN_MAX_RPM = 1700
emc = EMC2101(i2c)
emc.manual_fan_speed = 50
time.sleep(1)
emc.lut[27] = 25
emc.lut[34] = 50
emc.lut[42] = 75
emc.lut_enabled = True
emc.forced_temp_enabled = True
print("Lut:", emc.lut)
emc.forced_ext_temp = 28  # over 25, should be 25%
time.sleep(3)
print("25%% duty cycle is %f RPM:" % emc.fan_speed)


emc.forced_ext_temp = 35  # over 30, should be 50%
time.sleep(3)
print("50%% duty cycle is %f RPM:" % emc.fan_speed)

emc.forced_ext_temp = 43  # over 42, should be 75%
time.sleep(3)
print("75%% duty cycle is %f RPM:" % emc.fan_speed)
```

6.3 PWM Tuning

Adjust the EMC2101s PWM settings to fit your application. This example requires more memory than the first one because it needs to use the extended *adafruit_emc2101.emc2101_lut.EMC2101_LUT* driver to access LUT functionality.

Listing 3: examples/emc2101_set_pwm_freq.py

```
# SPDX-FileCopyrightText: 2020 Bryan Siepert, written for Adafruit Industries
#
# SPDX-License-Identifier: MIT
import time
import board
from adafruit_emc2101.emc2101_lut import EMC2101_LUT as EMC2101

i2c = board.I2C()  # uses board.SCL and board.SDA

emc = EMC2101(i2c)
emc.set_pwm_clock(use_preset=False)
# Datasheet recommends using the maximum value of 31 (0x1F)
# to provide the highest effective resolution
emc.pwm_frequency = 14

# This divides the pwm frequency down to a smaller number
# so larger divisor = lower frequency
emc.pwm_frequency_divisor = 127

while True:
    print("External temperature:", emc.external_temperature, "C")
    emc.manual_fan_speed = 50
    time.sleep(1.5)
    print("Fan speed:", emc.fan_speed, "RPM")
    time.sleep(1)
```

6.4 adafruit_emc2101

Brushless fan controller

- Author(s): Bryan Siepert

6.4.1 Implementation Notes

Hardware:

- Adafruit EMC2101 Breakout (Product ID: 4808)

Software and Dependencies:

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

class adafruit_emc2101.**ConversionRate**
Options for `accelerometer_data_rate` and `gyro_data_rate`

class adafruit_emc2101.**EMC2101**(*i2c_bus*)

Basic driver for the EMC2101 Fan Controller.

Parameters **i2c_bus** (*I2C*) – The I2C bus the EMC is connected to.

Quickstart: Importing and using the device

Here is an example of using the *EMC2101* class. First you will need to import the libraries to use the sensor

```
import board
from adafruit_emc2101.emc2101_lut import EMC2101_LUT as EMC2101
```

Once this is done you can define your `board.I2C` object and define your sensor object

```
i2c = board.I2C()  # uses board.SCL and board.SDA
emc = EMC2101(i2c)
```

Now you have access to the *manual_fan_speed* attribute to setup the desired fanspeed

```
emc.manual_fan_speed = 25
```

If you need control over PWM frequency and the controller's built in temperature/speed look-up table (LUT), you will need *emc2101_lut.EMC2101_LUT* which extends this class to add those features, at the cost of increased memory usage.

conversion_rate
The rate at which temperature measurements are taken. Must be a *ConversionRate*

dac_output_enabled
When set, the fan control signal is output as a DC voltage instead of a PWM signal

external_temperature
The temperature measured using the external diode

fan_speed
The current speed in Revolutions per Minute (RPM)

forced_ext_temp
The value that the external temperature will be forced to read when *forced_temp_enabled* is set. This can be used to test the behavior of the LUT without real temperature changes

forced_temp_enabled
When True, the external temperature measurement will always be read as the value in *forced_ext_temp*

initialize()
Reset the controller to an initial default configuration

internal_temperature
The temperature as measured by the EMC2101's internal 8-bit temperature sensor

invert_fan_output
When set to True, the magnitude of the fan output signal is inverted, making 0 the maximum value and 100 the minimum value

lut_enabled
Enable or disable the internal look up table used to map a given temperature to a fan speed.

When the LUT is disabled (the default), fan speed can be changed with *manual_fan_speed*. To actually set this to True and modify the LUT, you need to use the extended version of this driver, *emc2101_lut.EMC2101_LUT*

manual_fan_speed
The fan speed used while the LUT is being updated and is unavailable. The speed is given as the fan's PWM duty cycle represented as a float percentage. The value roughly approximates the percentage of the fan's maximum speed

spinup_drive
The drive strength of the fan on spinup in % max RPM

spinup_time
The amount of time the fan will spin at the current set drive strength. Must be a *SpinupTime*

tach_limit
The maximum /minimum speed expected for the fan

class adafruit_emc2101.**SpinupDrive**
Options for spinup_drive

class adafruit_emc2101.**SpinupTime**
Options for spinup_time

6.5 adafruit_emc2101.emc2101_lut

Brushless fan controller: extended functionality

- Author(s): Bryan Siepert, Ryan Pavlik

6.5.1 Implementation Notes

Hardware:

- Adafruit EMC2101 Breakout (Product ID: 4808)

Software and Dependencies:

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

The class defined here may be used instead of *adafruit_emc2101.EMC2101*, if your device has enough RAM to support it. This class adds LUT control and PWM frequency control to the base feature set.

class adafruit_emc2101.emc2101_lut.**EMC2101_LUT**(*i2c_bus*)
Driver for the EMC2101 Fan Controller, with PWM frequency and LUT control.

See *adafruit_emc2101.EMC2101* for the base/common functionality.

Parameters **i2c_bus** (*I2C*) – The I2C bus the EMC is connected to.

initialize()
Reset the controller to an initial default configuration

lut
The dict-like representation of the LUT, actually of type *FanSpeedLUT*

lut_enabled
Enable or disable the internal look up table used to map a given temperature to a fan speed. When the LUT is disabled fan speed can be changed with *manual_fan_speed*

lut_temperature_hysteresis
The amount of hysteresis in Degrees Celsius of hysteresis applied to temperature readings used for the LUT. As the temperature drops, the controller will switch to a lower LUT entry when the measured value is below the lower entry's threshold, minus the hysteresis value

pwm_frequency
Selects the base clock frequency used for the fan PWM output

pwm_frequency_divisor
The Divisor applied to the PWM frequency to set the final frequency

set_pwm_clock(*use_preset=False*, *use_slow=False*)

Select the PWM clock source, choosing between two preset clocks or by configuring the clock using *pwm_frequency* and *pwm_frequency_divisor*.

Parameters

- **use_preset** (*bool*) – True: Select between two preset clock sources False: The PWM clock is set by *pwm_frequency* and *pwm_frequency_divisor*
- **use_slow** (*bool*) – True: Use the 1.4kHz clock False: Use the 360kHz clock.

Returns None

Raises

- **AttributeError** – if use_preset is not a `bool`
- **AttributeError** – if use_slow is not a `bool`

class adafruit_emc2101.emc2101_lut.**FanSpeedLUT**(*fan_obj*)
A class used to provide a dict-like interface to the EMC2101's Temperature to Fan speed Look Up Table.

Keys are integer temperatures, values are fan duty cycles between 0 and 100. A max of 8 values may be stored.

To remove a single stored point in the LUT, assign it as `None`.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index

T