DesignSpark.Pmod Documentation

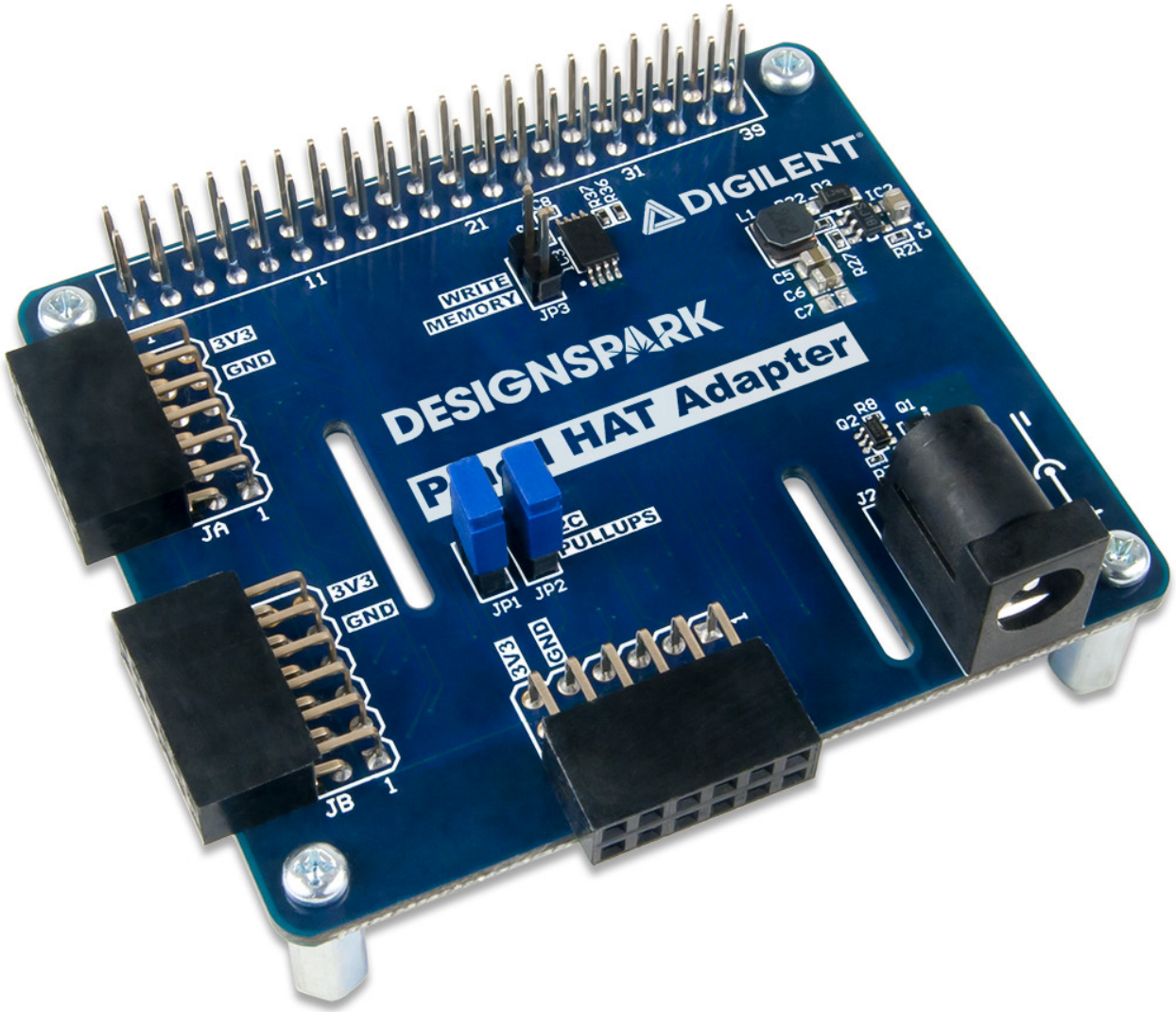
Release 0.1.0

RS Components Ltd

Feb 26, 2020

Contents

DESIGNSPARK Pmod Library



Python library to support using Pmods with a Raspberry Pi and the DesignSpark Pmod HAT.

Features include:

- Simple interfaces for supported Pmods
- Checking that Pmod and port capabilities match

- Checking for port usage conflicts
- Usage examples

CHAPTER 1

Supported Pmods

The following Pmods are currently supported:

- PmodAD1 12-bit ADC
- PmodHB3 2A H-bridge driver
- PmodISNS20 20A Current Sensor
- PmodMIC3 MEMS Microphone Module
- PmodOLEDrgb 96x64 RGB OLED Display [1]
- PmodTC1 K Type Thermocouple Module with Wire

[1] *Builds on the excellent luma.oled and luma.core libraries from Richard Hull and contributors.*

CHAPTER 2

Installation

DesignSpark.Pmod can be installed from PyPi using pip. See the documentation for details.

CHAPTER 3

Documentation

Installation and API documentation, along with examples, can be found at:

http://designspark-pmod.readthedocs.io

For Pmod HAT documentation, including the reference manual and schematic, see:

https://reference.digilentinc.com/reference/add-ons/pmod-hat/start

CHAPTER 4

Table of Contents

4.1 Installation

This guide assumes that you are running Raspbian Stretch.

First enable SPI:

```
pi@raspberrypi:~$ sudo raspi-config
```

Selecting:

- Option 5 - Interfacing
- P4 - SPI
- Enable → YES

Then exit raspi-config.

Next update the package lists:

```
pi@raspberrypi:~$ sudo apt-get update
```

Then install the Raspbian dependencies:

```
pi@raspberrypi:~$ sudo apt-get install python-pip python-dev libfreetype6-dev libjpeg-
↪dev build-essential
```

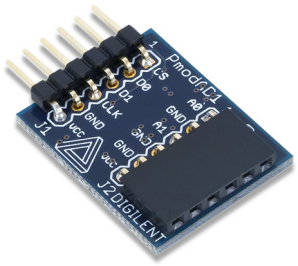
Finally, install DesignSpark.Pmod and dependencies from PyPi:

```
pi@raspberrypi:~$ sudo pip install designspark.pmod
```

4.2 Pmod Information

Details of currently supported Pmods can be found below.

4.2.1 AD1



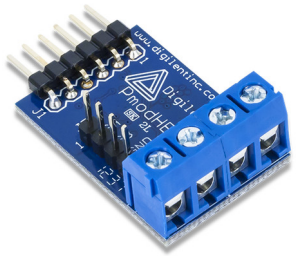
The Digilent Pmod AD1 (Revision G) is a two channel 12-bit analog-to-digital converter that features Analog Devices' AD7476A. With a sampling rate of up to 1 million samples per second, this Pmod™ is capable of excelling in even the most demanding audio applications.

Features:

- Two channel 12-bit analog-to-digital converter
- Simultaneous A/D conversion at up to one MSa per channel
- Two 2-pole Sallen-Key anti-alias filters
- Small PCB size for flexible designs 0.95 in × 0.8 in (2.4 cm × 2.0 cm)
- 6-pin Pmod port with GPIO interface

Note: Only a single channel (A1) is supported at present due to the way that SPI is configured.

4.2.2 HB3



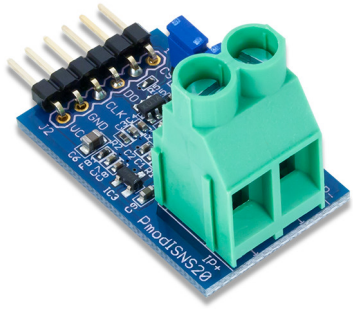
The Pmod HB3 utilizes a full H-Bridge circuit to allow users to drive DC motors from the system board. Two external pins are provided on the Pmod for sensor feedback on the DC motor, if desired.

Features:

- 2A H-bridge circuit
- Drive a DC motor with operation voltage up to 12V
- Screw terminal blocks for connection to the motor
- Separate header for external motor feedback

- 6-pin Pmod port with GPIO interface

4.2.3 ISNS20

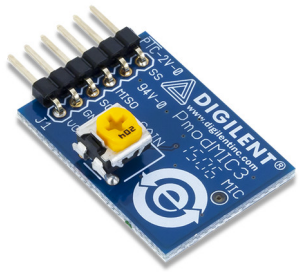


The Digilent Pmod ISNS20 (Revision A) is a small current sense module with a digital SPI interface. The board combines an Allegro ACS722 Hall Effect current sensor with a 12-bit analog-to-digital converter from Texas Instruments. The Pmod ISNS20 is quick, accurate, and easy to use for a variety of applications.

Features:

- High accuracy current sensor
- Measure current with 120Hz/20kHz/80kHz jumper selections
- ±20A DC or AC input
- Accurate to within ±2%
- 12-bit ADC
- 6-pin Pmod port with SPI interface
- Follows Digilent Pmod Interface Specification Type 2

4.2.4 MIC3



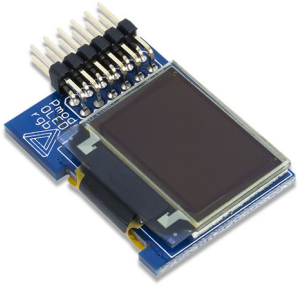
The Digilent Pmod MIC3 (Revision A) is small microphone module with a digital interface. With a Knowles Acoustics SPA2410LR5H-B MEMs microphone and Texas Instrument's ADCS7476 12-bit Analog-to-Digital Converter, you can capture your audio inputs with ease.

Features:

- MEMS Microphone module with digital interface
- Transform audio inputs with 12-bit A/D converter

- Adjust incoming volume with on-board potentiometer
- Up to 1 MSPS of data
- 6-pin Pmod port with SPI interface
- Follows Digilent Pmod Interface Specification Type 2

4.2.5 OLEDrgb

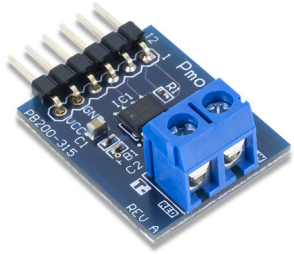


The Digilent Pmod OLEDrgb (Revision B) is an organic RGB LED module with a 96×64 pixel display capable of 16-bit color resolution.

Features:

- 96×64 pixel RGB OLED screen
- 0.8“ x 0.5” graphical display
- 16-bit color resolution
- Two low-power display shutdown modes
- 12-pin Pmod connector with SPI interface

4.2.6 TC1



The Digilent Pmod TC1 (Revision A) is a cold-junction thermocouple-to-digital converter module designed for a classic K-Type thermocouple wire. With Maxim Integrated’s MAX31855, this module reports the measured temperature in 14-bits with 0.25°C resolution.

Features:

- K-type thermocouple-to-digital converter
- Wide temperature range of -73°C to 482°C with provided wire

- ±2°C accuracy from -200°C to 700°C
- 14-bit with 0.25°C resolution
- Cold-junction temperature compensation
- 6-pin Pmod port with SPI interface
- Follows Digilent Pmod Interface Specification Type 2

4.3 Basic Examples

4.3.1 AD1

12-bit analog-to-digital converter.

Print volts out to the terminal

Read ADC channel A1, print the voltage measured out to the terminal, sleep for 0.8s and repeat.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
Read ADC channel A1 and print volts out.
"""

from DesignSpark.Pmod.HAT import createPmod
import time

if __name__ == '__main__':
    adc = createPmod('PmodAD1','JBA')
    time.sleep(0.1)

    try:
        while True:
            volts = adc.readA1Volts()
            print(volts)
            #val = adc.readA1()
            #print(val)
            time.sleep(0.8)
    except KeyboardInterrupt:
        pass
    finally:
        adc.cleanup()
```

Requirements

- PmodAD1 module connected to port JBA
- A voltage source connected to ADC channel A1

4.3.2 HB3

2A H-bridge circuit for DC motor drive up to 12V.

Spin motor

This example:

1. Spins the motor forwards for 20 seconds
2. Commands the motor to stop and pauses for 2 seconds
3. Spins the motor in reverse for 20 seconds
4. Commands the motor to stop and pauses for 2 seconds
5. Ramps up the speed across 100 steps in the forward direction
6. Ramps down the speed across 100 steps in the forward direction
7. Ramps up speed across 100 steps in the reverse direction
8. Ramps down the speed across 100 steps in the reverse direction
9. Loops back to (1)

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
Spin motor forwards then backwards.
Ramp speed up and then down in forward direction.
Ramp speed up and then down in reverse direction.
"""

from DesignSpark.Pmod.HAT import createPmod
import time

if __name__ == '__main__':

    motor = createPmod('PmodHB3','JAA')

    try:
        while True:

            print('fwd')
            motor.forward(20)
            time.sleep(2)
            motor.stop()
            time.sleep(2)
            print('rev')
            motor.reverse(20)
            time.sleep(1)
            motor.stop()
            time.sleep(2)


            print ('ramp up fwd')
```

(continues on next page)

(continued from previous page)

```
            for i in range(100):
                motor.forward(i)
                time.sleep(.1)

            print ('ramp down fwd')
            for i in range(100):
                motor.forward(100-i)
                time.sleep(.1)

            motor.stop()
            time.sleep(2)

            print ('ramp up rev')
            for i in range(100):
                motor.reverse(i)
                time.sleep(.1)

            print ('ramp down rev')
            for i in range(100):
                motor.reverse(100-i)
                time.sleep(.1)

    except KeyboardInterrupt:
        pass

    finally:
        motor.cleanup()
```

Requirements

- PmodHB3 module connected to port JAA
- DC motor
- Motor power supply

4.3.3 ISNS20

±20A DC or AC input, high accuracy current sensor.

Print milliamps out to the terminal

Read the current sense module, print the milliamps out to the terminal, sleep for 0.8s and repeat.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
```

(continues on next page)

(continued from previous page)

```
Read current and print out milliamps.
"""

from DesignSpark.Pmod.HAT import createPmod
import time

if __name__ == '__main__':

    isens = createPmod('PmodISNS20','JBA')
    time.sleep(0.1)

    try:
        while True:
            mA = isens.readMilliAmps()
            print(mA)
            time.sleep(0.8)
    except KeyboardInterrupt:
        pass
    finally:
        isens.cleanup()
```

Requirements

- PmodISNS20 module connected to port JBA
- Suitable current source, e.g. a power supply and load

4.3.4 MIC3

Knowles Acoustics SPA2410LR5H-B MEMs microphone and Texas Instrument's ADCS7476 12-bit Analog-to-Digital Converter.

Display mic level

Print a continous sound level reading out to the terminal.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
Print a continous sound level reading out.
"""

from DesignSpark.Pmod.HAT import createPmod
import time

s = ':'
lut = [s]
for i in range(128):
    s+=':'
```

(continues on next page)

(continued from previous page)

```
    lut.append(s)

if __name__ == '__main__':

    mic = createPmod('PmodMIC3','JBA')
    time.sleep(0.1)

    try:
        while True:
            int = mic.readIntegerValue()
            #print(int)
            print(lut[int>>5])
            snd = mic.readPhysicalValue()
            #print(snd)

            #time.sleep(0.01)
    except KeyboardInterrupt:
        pass
    finally:
        mic.cleanup()
```

Requirements

- PmodMIC3 module connected to port JBA

4.3.5 OLEDrgb

Organic RGB LED module with a 96×64 pixel display capable of 16-bit color resolution.

Display text in a bounding box

Display the text "Hello, World!" in a bounding box.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
Display Hello, World! in bounding box.
"""

from DesignSpark.Pmod.HAT import createPmod
from luma.core.render import canvas
from luma.oled.device import ssd1331

if __name__ == '__main__':
    try:
        oled = createPmod('PmodOLEDrgb','JA')
        device = oled.getDevice()
```

(continues on next page)

(continued from previous page)

```
        with canvas(device) as draw:
            draw.rectangle(device.bounding_box, outline="white", fill="black")
            draw.text((16,20), "Hello, World!", fill="white")

        while True:
            pass
    except KeyboardInterrupt:
        pass
    finally:
        oled.cleanup()
```

Luma.Core & Luma.OLED API: `luma.core.render.canvas`, `luma.oled.device.ssd1331`.

Requirements

- PmodOLEDrgb connected to port JA

4.3.6 PmodTC1

A cold-junction thermocouple-to-digital converter module designed for a classic K-Type thermocouple wire. Features a temperature range of -73°C to 482°C with provided wire.

Print celsius out to the terminal

Print the celsius reading out to the terminal, sleep for 0.8s and repeat.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Copyright (c) 2017 RS Components Ltd
# SPDX-License-Identifier: MIT License

"""
Print the celsius reading out.
"""

from DesignSpark.Pmod.HAT import createPmod
import time

if __name__ == '__main__':

    therm = createPmod('PmodTC1','JBA')
    time.sleep(0.1)

    try:
        while True:
            cel = therm.readCelcius()
            print(cel)
            #intn = therm.readInternal()
            #print(intn)
            time.sleep(0.8)
    except KeyboardInterrupt:
        pass
    finally:
```

(continues on next page)

(continued from previous page)

```
        therm.cleanup()
```

Requirements

- PmodTC1 module connected to port JBA

4.4 API

4.4.1 DesignSpark.Pmod.AD1

Interface for PmodAD1 module (AD7476A).

Note: Only a single channel (A1) is supported at present due to the way that SPI is configured.

class DesignSpark.Pmod.AD1.**PmodAD1**(*DSPMod6*)

cleanup()

readA1()

readA1Volts()

4.4.2 DesignSpark.Pmod.Error

exception DesignSpark.Pmod.Error.**Error**
Bases: exceptions.Exception

Base class for exceptions in this library.

exception DesignSpark.Pmod.Error.**incorrectModuleName**
Bases: *DesignSpark.Pmod.Error.Error*

Exception raised when the module name given does not exist in the module map.

exception DesignSpark.Pmod.Error.**incorrectPortName**
Bases: *DesignSpark.Pmod.Error.Error*

Exception raised when the port name given does not exist in the port map.

exception DesignSpark.Pmod.Error.**portCapabilityConflict**
Bases: *DesignSpark.Pmod.Error.Error*

Exception raised when the port is already using shared GPIO pins.

exception DesignSpark.Pmod.Error.**portCapabilitySupport**
Bases: *DesignSpark.Pmod.Error.Error*

Exception raised when a port does not support the module type.

exception DesignSpark.Pmod.Error.**portInUse**
Bases: *DesignSpark.Pmod.Error.Error*

Exception raised when the port is already assigned.

4.4.3 DesignSpark.Pmod.HAT

Manages Pmod HAT port resources, enforcing correct usage and avoiding conflicts.

class DesignSpark.Pmod.HAT.**DSPMod12**(*_portName*)

inUse()

setUseModule(*moduleName*)

class DesignSpark.Pmod.HAT.**DSPMod6**(*_portName*)

inUse()

setUseModule(*moduleName*)

DesignSpark.Pmod.HAT.**createPmod**(*moduleName*, *portName*)

4.4.4 DesignSpark.Pmod.HB3

Interface for PmodHB3 module.

class DesignSpark.Pmod.HB3.**PmodHB3**(*DSPMod6*)

changeFrequency(*freq*)

cleanup()

forward(*duty*)

reverse(*duty*)

stop()

4.4.5 DesignSpark.Pmod.ISNS20

Interface for PmodISNS20 module (ADC7476 + Allegro ACS722).

class DesignSpark.Pmod.ISNS20.**PmodISNS20**(*DSPMod6*)

cleanup()

readAmps()

readMilliAmps()

4.4.6 DesignSpark.Pmod.MIC3

Interface for PmodMIC3 (ADCS7476 + Knowles Acoustics SPA2410LR5H-B).

class DesignSpark.Pmod.MIC3.**PmodMIC3**(*DSPMod6*)

MIC3_NO_BITS = 12

cleanup()

dReference = 3.3

readIntegerValue()

readPhysicalValue()

4.4.7 DesignSpark.Pmod.OLEDrgb

Interface for PmodOLEDrgb module (ssd1331).

Note: Depends on luma.oled and luma.core.

class DesignSpark.Pmod.OLEDrgb.**PmodOLEDrgb**(*DSPMod12*)

cleanup()

getDevice()

powerOff()

powerOn()

4.4.8 DesignSpark.Pmod.TC1

Interface for PmodTC1 module (MAX31855).

class DesignSpark.Pmod.TC1.**PmodTC1**(*DSPMod6*)

cleanup()

readCelcius()

readError()

readFarenheit()

readInternal()

CHAPTER 5

Change log

Version	Description	Date
0.2.0	• Tidied up names, added documentation and examples	28/11/17
0.1.0	• Initial version	26/11/17

CHAPTER 6

The MIT License (MIT)

Python Module Index

d

Index

S