mitutoyo Library Documentation

Release 1.0

Adrian "vifino" Pistol

Feb 24, 2020

Contents

CircuitPython implementation of the Mitutoyo Digimatic SPC interface.

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython

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

Note: This library is not available on PyPI yet. Install documentation is included as a standard element. Stay tuned for PyPI availability!

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install circuitpython-mitutoyo
```

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

```
sudo pip3 install circuitpython-mitutoyo
```

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 circuitpython-mitutoyo
```

CHAPTER 3

Usage Example

```
import board
import mitutoyo

instrument = mitutoyo.Digimatic(req=board.D0, clock=board.D1, data=board.D2)

reading = instrument.read()
print(reading)  # human formatted
print("Reading in %s: %f" %(reading.unit, reading.value))
```

CHAPTER 4

Contributing

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

CHAPTER 5

Table of Contents

5.1 Mitutoyo Caliper Readout

This example prints a connected Caliper's value on the push of the ready button on the instrument.

Listing 1: examples/mitutoyo_caliper_readout.py

```
# This is a quick example how to read values from Mitutoyo Calipers.
# Assuming a Serpente board, with `req`, `clock`, `data` and `ready` connected to
# D0, D1, D2 and D3, respectively.

import time
import board
import digitalio
import mitutoyo

pin_ready = digitalio.DigitalInOut(board.D3)
pin_ready.direction = digitalio.Direction.INPUT
pin_ready.pull = digitalio.Pull.UP

print("Hello! Press the read button on the Calipers to print the value!")

meter = mitutoyo.Digimatic(req=board.D0, clock=board.D1, data=board.D2)

while True:
    # wait until ready goes low
    while pin_ready.value:
        time.sleep(0.1)

    reading = meter.read()
    if reading:
        print(reading)

    # wait until ready goes up again to just get a single reading per press
```

(continues on next page)

(continued from previous page)

```
    while not pin_ready.value:
        time.sleep(0.1)
```

5.2 Mitutoyo Caliper USB CAD entry

This example types out the reading of a connected Caliper's value on the push of the ready button on the instrument.

Listing 2: examples/mitutoyo_usb_cad_entry.py

```
# Direct CAD entry for Mitutoyo calipers via USB HID emulation.
# Assuming a Serpente board, with `req`, `clock`, `data` and `ready` connected to
# D0, D1, D2 and D3, respectively.

import time
import board
import digitalio
import mitutoyo

import usb_hid
from adafruit_hid.keyboard import Keyboard
from adafruit_hid.keyboard_layout_us import KeyboardLayoutUS

# I/O
pin_ready = digitalio.DigitalInOut(board.D3)
pin_ready.direction = digitalio.Direction.INPUT
pin_ready.pull = digitalio.Pull.UP

meter = mitutoyo.Digimatic(req=board.D0, clock=board.D1, data=board.D2)

# USB HID keyboard emulation.
time.sleep(1)
kbd = Keyboard(usb_hid.devices)
layout = KeyboardLayoutUS(kbd)

print("Ready.")

while True:
    # wait until ready goes low
    while pin_ready.value:
        time.sleep(0.1)

    reading = meter.read()
    if reading:
        print("Typing %s." % reading)
        layout.write(str(reading) + "\n")

    # wait until ready goes up again to just get a single reading per press
    while not pin_ready.value:
        time.sleep(0.1)
```

5.3 mitutoyo: A library for the Mitutoyo Digimatic (SPC) protocol.

This library is an implementation of the Mitutoyo Digimatic protocol used to read data from gauges and scales.

It was written as a first project with CircuitPython. Data used to implement this were Mitutoyo datasheets.

- Author(s): Adrian "vifino" Pistol

5.3.1 Implementation Notes

Hardware:

- You need the 'data' and 'clock' pins configured as inputs with pullup. They are pin 2 and 3 on a Digimatic 10-pin cable.
- Connect the 'req' pin to a NPN with a 10kΩ resistor and the open collector output to '!req'. On a Digimatic 10-pin cable, '!req' is pin 5.
- Optionally, you can connect 'ready' as an input with a pullup to know when to read. On a Digimatic 10-pin cable, 'ready' is pin 4.

Software:

- CircuitPython 5.0 tested, older versions should work. MicroPython should also work, thanks to Adafruit Blinka.

class mitutoyo.**Digimatic**(***args*)

Mitutoyo Digimatic SPC implementation for CircuitPython. Provide either 'req' or 'nreq'. 'req' takes precedence.

Parameters

- **data** (*Pin*) – data pin
- **clock** (*Pin*) – clock pin
- **req** (*Pin*) – non-inverted data request pin, alternative to 'nreq'
- **nreq** (*Pin*) – inverted data request pin, alternative to 'req'

class Reading(*value*, *unit*)

A Reading from a Mitutoyo Digimatic instrument.

unit = None

The unit the reading's value is in. (str)

value = None

The value returned by the instrument. (float)

read()

Attempt to read a value from the connected instrument.

Returns A reading or none if data is unparsable

Return type *mitutoyo.Digimatic.Reading*

read_cm()

Attempt to read from a connected instrument, but always return the value in centimeters.

Returns centimeters

Return type float

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

Index