

Using a Labjack U3 with Python on Linux

Raul Monsalve

SESE, Arizona State University

September 28, 2013

Pre-requisites

The drivers/packages required for controlling a Labjack on Linux with Python are:

- ▶ build-essential (linux package)
- ▶ libusb-1.0-0-dev (linux package)
- ▶ Python 2.5, 2.6, or 2.7 (linux package)
- ▶ exodriver (labjack driver)
- ▶ LabJackPython (labjack driver)

For an example of installation, go to:

`http://labjack.com/support/linux-and-mac-os-x-drivers`

The rest of this document describes some of the basic Python functions available from the LabJackPython driver, and provides some examples. It is a condensed version of the descriptions contained in the file `u3.py` provided in the driver.

Initialization

In this example, the U3 is opened automatically.

```
>>> import u3
>>> d = u3.U3()
>>> .... operations .....
```

In this example, the U3 is opened and closed manually.

```
>>> import u3
>>> d = u3.U3(autoOpen = False)
>>> d.open()
>>> .... operations .....
```

General Configuration

Function `configU3` is used to load default configuration at power-up.

```
>>> import u3
>>> d = u3.U3()
>>> d.configU3() # loading default configuration
>>> .... operations ....
```

It can also change the configuration, WHICH REMAINS AFTER DISCONNECTION - RECONNECTION.

```
>>> import u3
>>> d = u3.U3()
>>> # configure FIOs 0 and 2 to analog
>>> d.configU3( FIOAnalog = 5 )
>>> .... operations ....
```

General Configuration

Function `configIO` is similar to `configU3`, used to change the configuration. However, the changes are NOT PERMANENT after reboot.

```
>>> import u3
>>> d = u3.U3()
>>> # change all FIOs to digital
>>> d.configIO( FIOAnalog = 0 )
>>> .... operations ....
```

Configuration of Flexible Input/Outputs

The keyword `FIOAnalog` in functions `configU3` and `configIO` selects which inputs/outputs are analog.

The flexible input/outputs are named FIO0 - FIO7. They are controlled with 8 bits; the one with the lowest weight controls FIO0, and the one with the highest weight controls FIO7.

The value of `FIOAnalog` can be set as a decimal. To set inputs FIO0 and FIO2 as analog, the decimal value is

$$[0\ 0\ 0\ 0\ 0\ 1\ 0\ 1] \cdot [128\ 64\ 32\ 16\ 8\ 4\ 2\ 1]^T = 5. \quad (1)$$

Read State of Digital and Analog Channels

Function to read the state of digital I/O (channel has to be set to digital beforehand). It will not change the direction of channel (will keep an input as input, or output as output).

```
>>> import u3
>>> d = u3.U3()
>>> print d.getDIOState(4) # 0 - 7 = FIO0 - FIO7
```

Function to read the voltage of analog input (channel has to be set to analog beforehand).

```
>>> import u3
>>> d = u3.U3()
>>> print d.getAIN(6) # 0 - 7 = FIO0 - FIO7
```

Read/Write State of Digital Channels

Function to read the state of digital I/O (channel has to be set to digital beforehand). It will force the direction of channel to input.

```
>>> import u3
>>> d = u3.U3()
>>> print d.getDIState(4) # 0 - 7 = FIO0 - FIO7
```

Function to set the state of digital I/O to 0 or 1 (channel has to be set to digital beforehand). It will force the direction of channel to output.

```
>>> import u3
>>> d = u3.U3()
>>> # 6 = FIO6, 1 = high, 0 = low
>>> print d.setDOState(6, state = 1)
```

Temperature of Labjack

Reads the internal temperature sensor on the U3. Returns the temperature in Kelvin.

```
>>> import u3
>>> d = u3.U3()
>>> print d.getTemperature()
```

EXAMPLE

The following program reads the voltage of analog input 7 and changes the state of digital output 0. The update time is 3 seconds.

```
#!/usr/bin/python

# Initialization
import time, u3
d = u3.U3()

# Channels 4 - 7 set as analog inputs. Channels 0 - 3 are left as digital
d.configU3( FIOAnalog = 240 )

# Channels 4 - 7 set as analog inputs. Channels 0 - 3 are left as digital
#d.configIO( FIOAnalog = 240 )

# Set digital outputs to ZERO
d.setDOState(0, state = 0)
d.setDOState(1, state = 0)
d.setDOState(2, state = 0)
d.setDOState(3, state = 0)
```

EXAMPLE

```
# Selection of FIO channels
dig_channel = 0
ana_channel = 7

# Loop reading analog inputs and switching digital outputs
flag = 0
state = 0
while flag < 1:

    if state == 0:
        state = 1
    elif state == 1:
        state = 0

    d.setDOState(dig_channel, state)
    str1 = 'digital state of FIO' + str(dig_channel) + ': ' + str(d.getDIOState(dig_channel))
    print str1

    str2 = 'analog voltage of FIO' + str(ana_channel) + ': ' + str(d.getAIN(ana_channel))
    print str2

    time.sleep(3)
```