Building Machine Vision Applications using OpenMV

Class 3: Working with the OpenMV I/O

June 10, 2020
Jacob Beningo

Presented by:

DesignNews

CEC CONTINUING EDUCATION CENTER

Digi-Key ELECTRONICS

Course Overview

Topics:

- Introduction to Machine Vision and OpenMV
- Writing our First OpenMV Application
- **Working with the OpenMV I/O**
- Utilizing Machine Learning to Detect Objects
- Designing a Machine Vision Application

Presented by:

Session Overview

- MicroPython
- MicroPython Libraries
- GPIO Example
- Analog Example
- UART Example
- SPI Example
- CAN Example

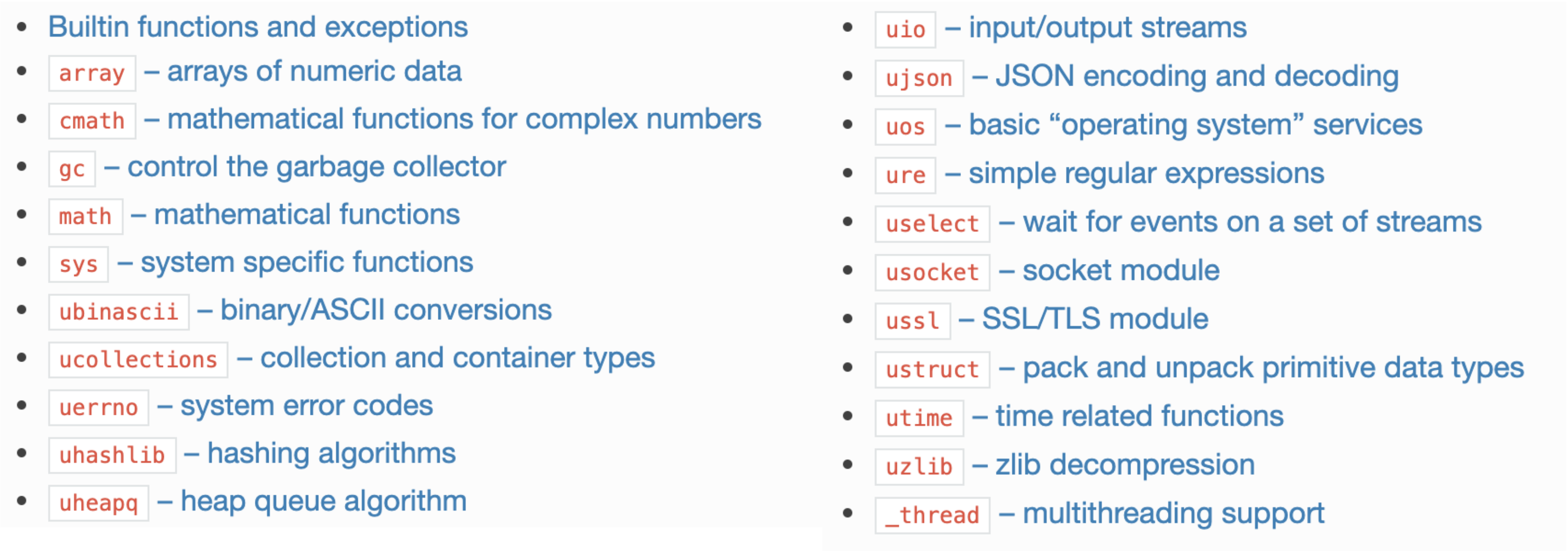
Presented by:

MicroPython

Definition: "MicroPython is a lean and efficient implementation of the Python 3 programming language that includes a small subset of the Python standard library and is optimised to run on microcontrollers and in constrained environments." (Source: micropython.org)

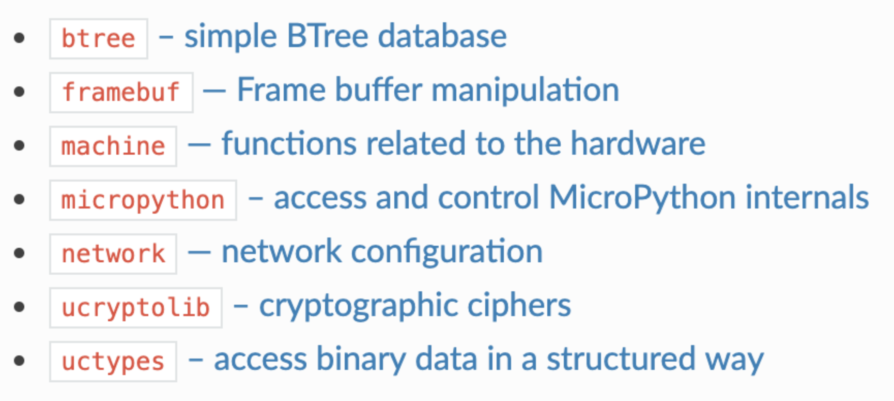
MicroPython Library Overview

- http://docs.micropython.org/en/latest/library/index.html

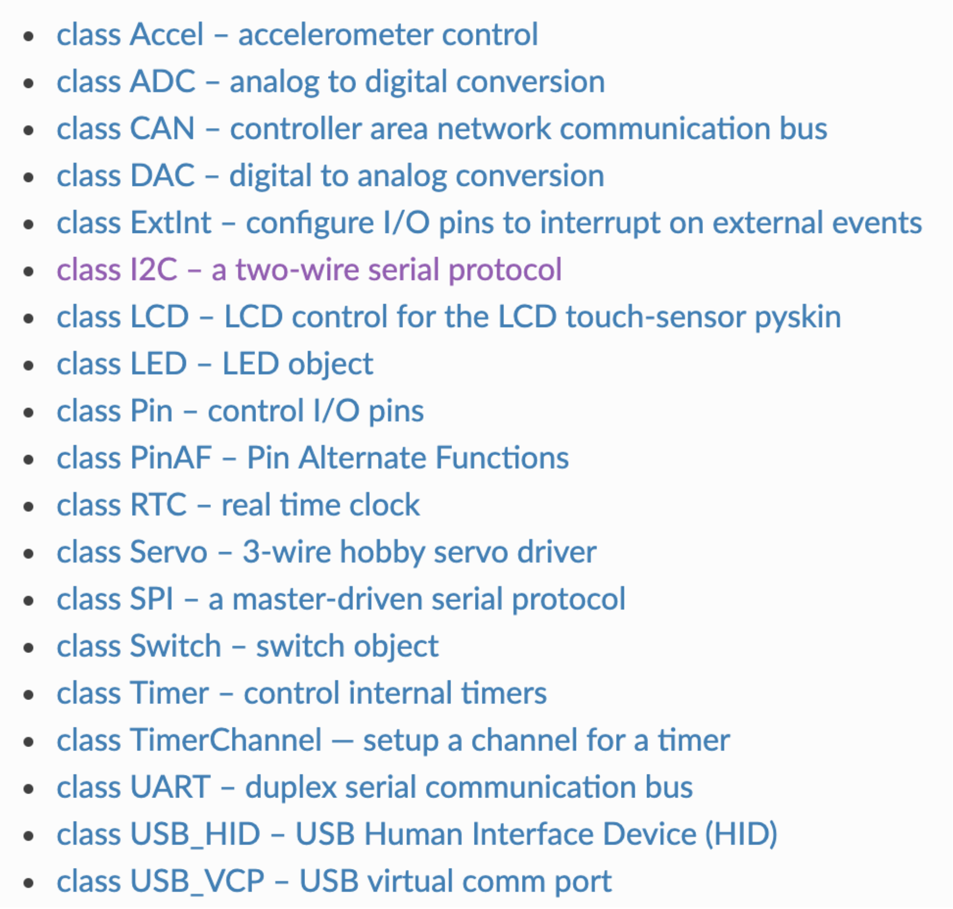


- Builtin functions and exceptions
- `array` – arrays of numeric data
- `cmath` – mathematical functions for complex numbers
- `gc` – control the garbage collector
- `math` – mathematical functions
- `sys` – system specific functions
- `ubinascii` – binary/ASCII conversions
- `ucollections` – collection and container types
- `uerrno` – system error codes
- `uhashlib` – hashing algorithms
- `uheapq` – heap queue algorithm
- `uio` – input/output streams
- `ujson` – JSON encoding and decoding
- `uos` – basic “operating system” services
- `ure` – simple regular expressions
- `uselect` – wait for events on a set of streams
- `usocket` – socket module
- `ussl` – SSL/TLS module
- `ustruct` – pack and unpack primitive data types
- `utime` – time related functions
- `uzlib` – zlib decompression
- `_thread` – multithreading support

MicroPython Libraries

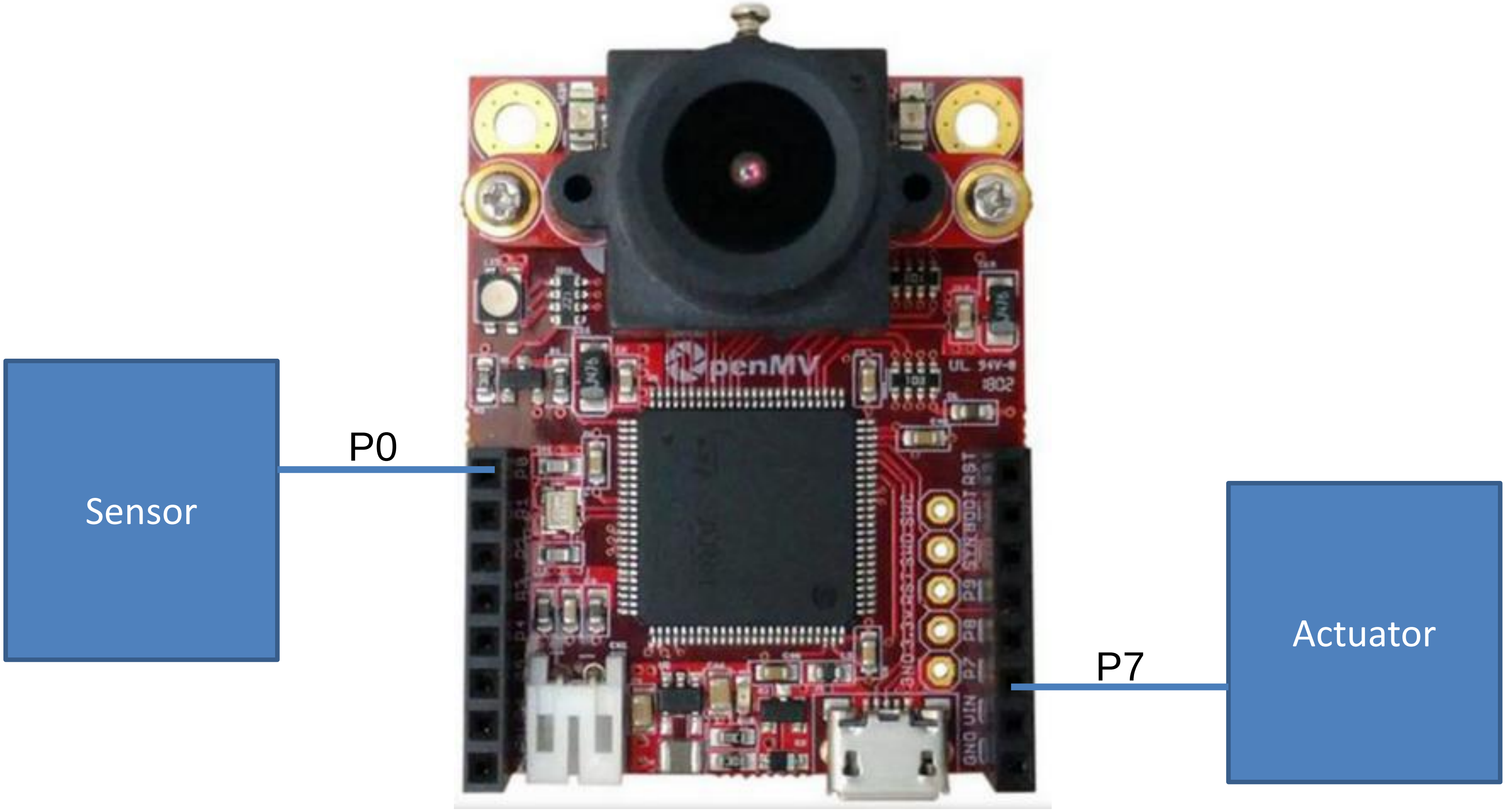


- `btree` – simple BTree database
- `framebuf` — Frame buffer manipulation
- `machine` — functions related to the hardware
- `micropython` – access and control MicroPython internals
- `network` — network configuration
- `ucryptolib` – cryptographic ciphers
- `uctypes` – access binary data in a structured way



- class Accel – accelerometer control
- class ADC – analog to digital conversion
- class CAN – controller area network communication bus
- class DAC – digital to analog conversion
- class ExtInt – configure I/O pins to interrupt on external events
- class I2C – a two-wire serial protocol
- class LCD – LCD control for the LCD touch-sensor pyskin
- class LED – LED object
- class Pin – control I/O pins
- class PinAF – Pin Alternate Functions
- class RTC – real time clock
- class Servo – 3-wire hobby servo driver
- class SPI – a master-driven serial protocol
- class Switch – switch object
- class Timer – control internal timers
- class TimerChannel — setup a channel for a timer
- class UART – duplex serial communication bus
- class USB_HID – USB Human Interface Device (HID)
- class USB_VCP – USB virtual comm port

GPIO Example



Sensor

P0

P7

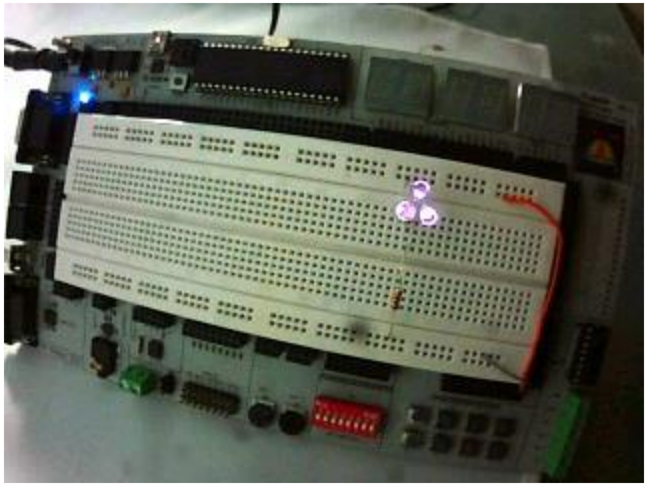
Actuator

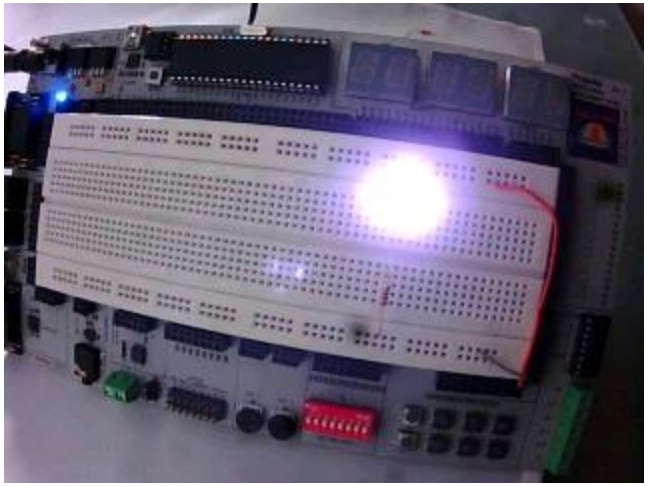
Presented by:

GPIO Example

```
import sensor, image, time
import pyb

# OpenMV Camera Init (Not shown to save space)

P0 = pyb.Pin("P0", pyb.Pin.IN)
P7 = pyb.Pin("P7", pyb.Pin.OUT_OD)
P7.value(0)

while(True):
    if P0.value() is False:
        img = sensor.snapshot()
        P7.value(1)
    else:
        P7.value(0)
```

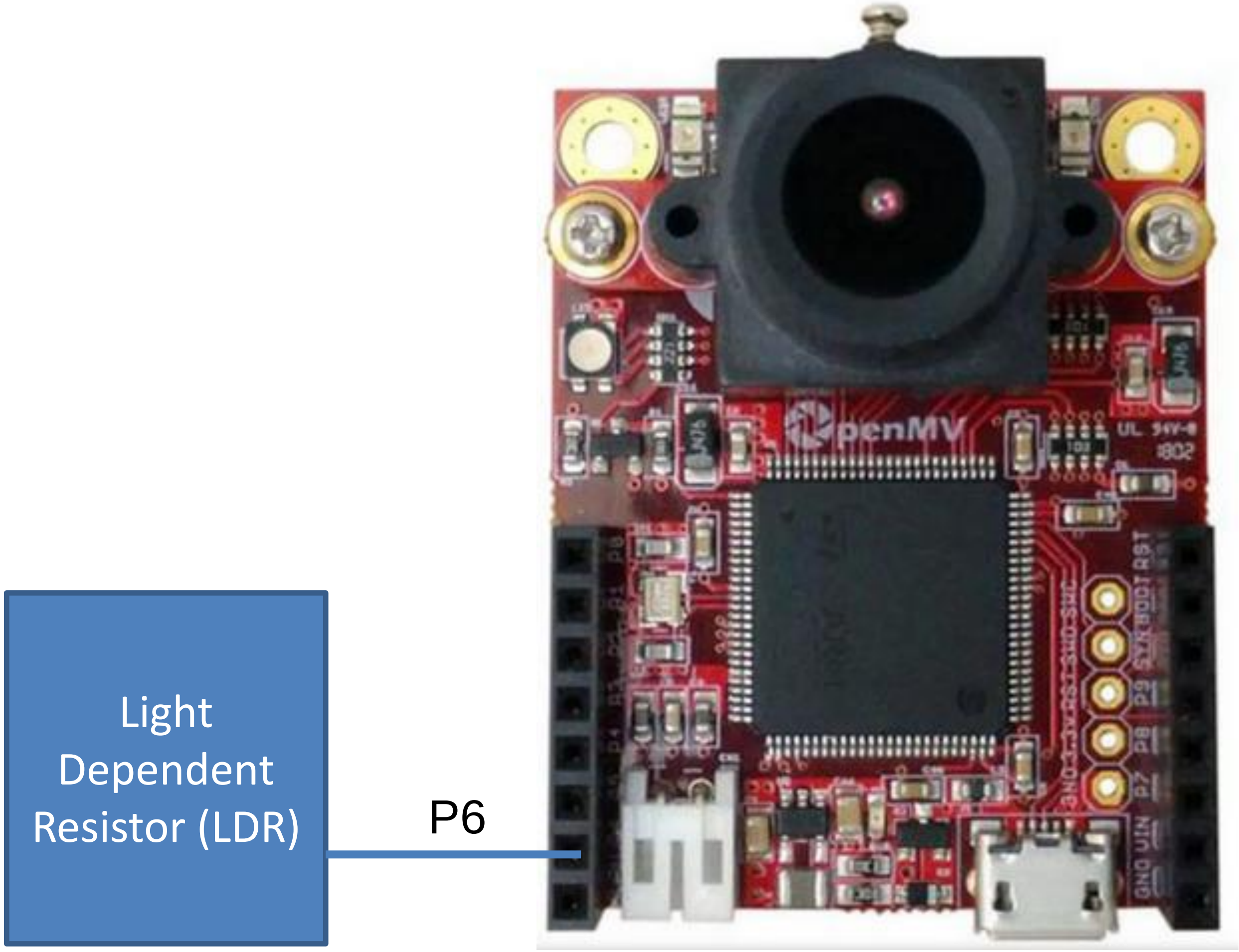
Analog Example





Source: openmv.io

Analog Example



Light Dependent Resistor (LDR)

P6

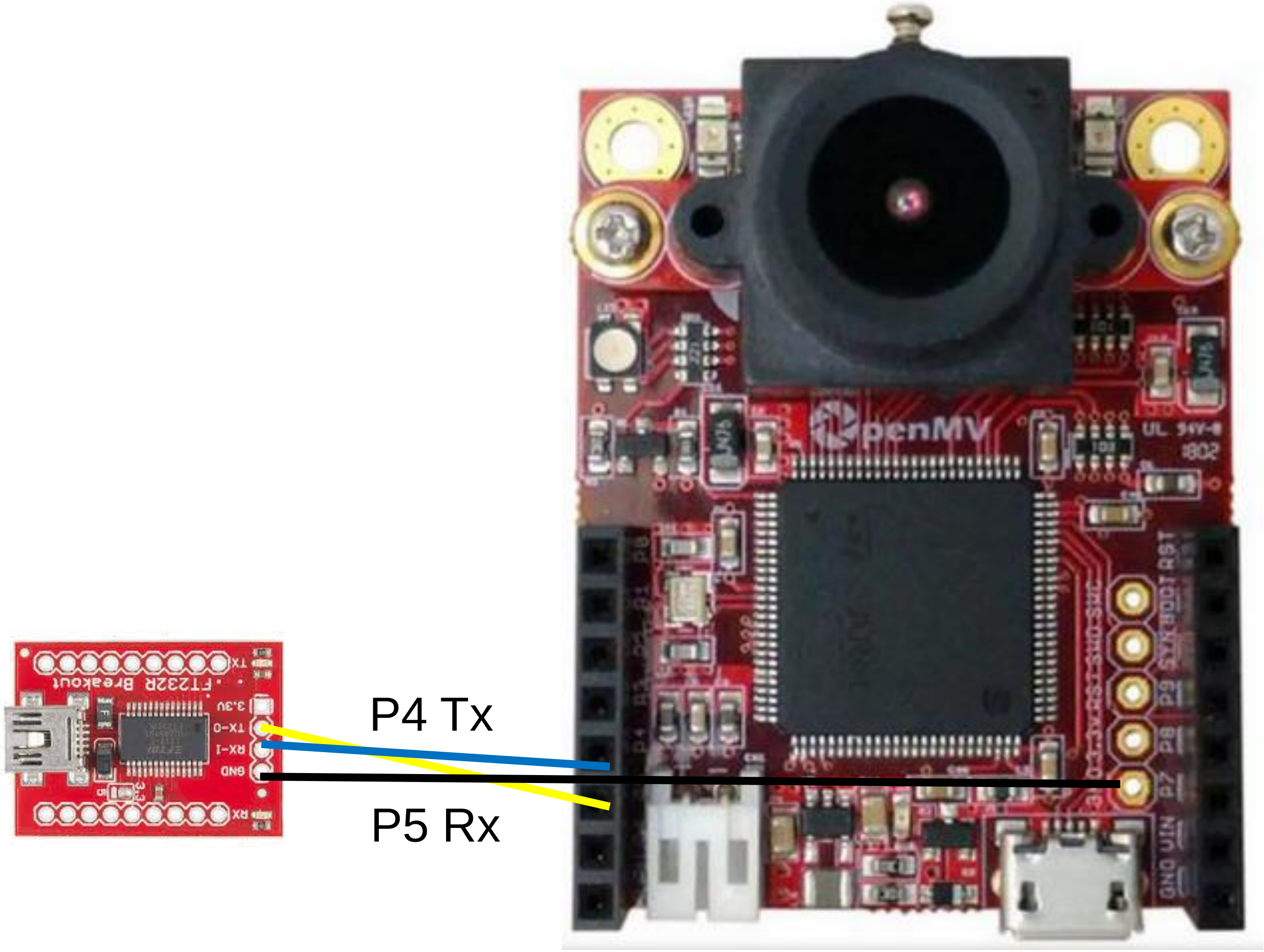
Analog Example

```
import sensor, image, time
import pyb

# OpenMV Camera Init (Not shown to save space)

LDR = pyb.ADC(pyb.Pin('P6'))
irLED = pyb.LED(4)

while(True):
    if LDR.read() > 3000:
        irLED.on()
    else:
        irLED.off()

    img = sensor.snapshot()
```

May want to change for testing!

Adjust based on testing!

UART Example



P4 Tx

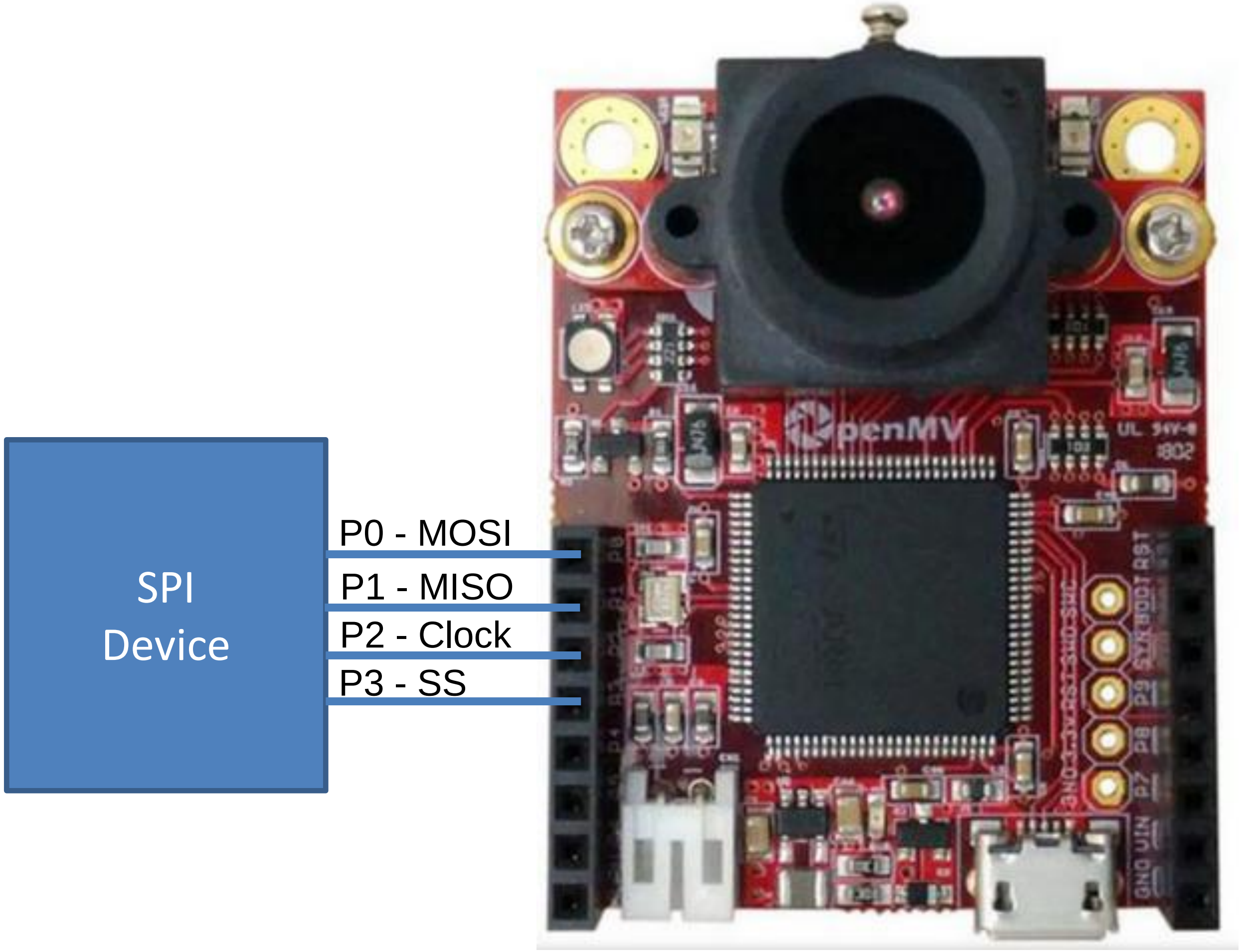
P5 Rx

UART Example

```
import sensor, image, time
import pyb

# OpenMV Camera Init (Not shown to save space)

uart3 = pyb.UART(3, 115200, timeout_char = 1000)

uart3.write("Hello World!\n\r")

while(True):
    while uart3.any() is not 0:
        uart3.write(uart3.read(1))

    img = sensor.snapshot()
```

read(5)
read()
readline()
readinto(buffer)

SPI Example



SPI Device

P0 - MOSI

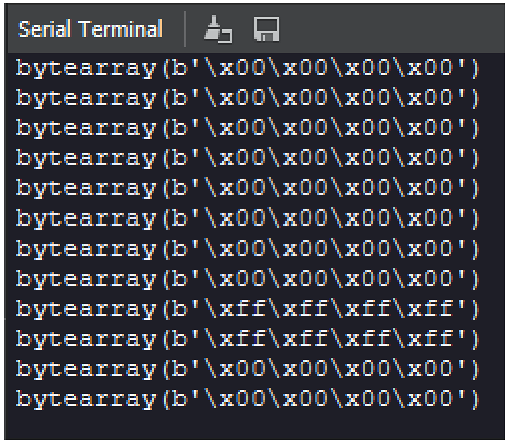
P1 - MISO

P2 - Clock

P3 - SS

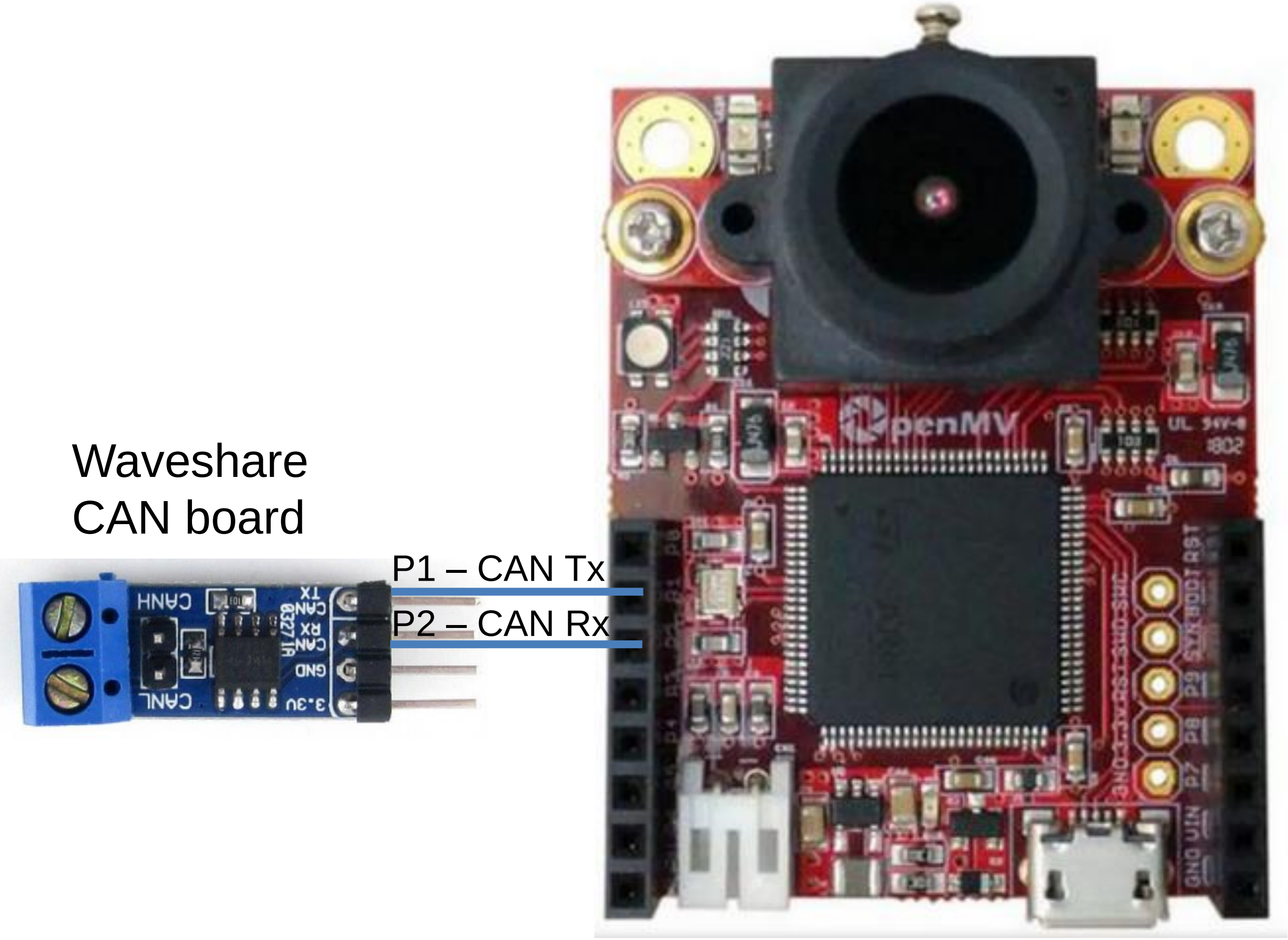
SPI Example

```
import sensor, image, time
from pyb import SPI

# OpenMV Camera Init (Not shown to save space)

spi = SPI(2, SPI.MASTER, baudrate=1000000, polarity=1, phase=0, crc=0x7)
buf = bytearray(4)

while(True):
    spi.send_recv(b'ABCD', buf)
    print(buf)

    img = sensor.snapshot()
```



```
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\xff\xff\xff\xff')
bytearray(b'\xff\xff\xff\xff')
bytearray(b'\x00\x00\x00\x00')
bytearray(b'\x00\x00\x00\x00')
```

CAN Example



Waveshare CAN board

P1 – CAN Tx

P2 – CAN Rx

Presented by:

CAN Example

```
import sensor, image, time
from pyb import CAN

# OpenMV Camera Init (Not shown to save space)

can2 = CAN(2, CAN.NORMAL)
can2.init(CAN.NORMAL, extframe=True, prescaler=4,sjw=1,bs1=5,bs2=6,auto_restart=True)

buffer = bytearray(8)
lst = [0,0,0,memoryview(buffer)]

while(True):
    can2.send('message!', 200)

    if can2.any(0) is not False:
        can2.recv(0, lst)
```

Getting MicroPython Support

- Tutorials
  - https://docs.micropython.org/en/latest/pyboard/tutorial/index.html
- Library reference
  - https://docs.micropython.org/en/latest/library/index.html
- Forum
  - https://forum.micropython.org/
- Kernel Repository
- https://github.com/micropython/micropython

Additional Resources

- Beningo.com
  - Blog, White Papers, Courses
  - Embedded Bytes Newsletter
    - http://bit.ly/1BAHYXm
- OpenMV.io

From www.beningo.com under

- Blog > CEC – Building Machine Vision Applications using OpenMV