# Acceleration & Light Sensors

## ECE 476 Advanced Embedded Systems

## Jake Glower - Lecture #24

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

# Introduction:

The Pi-Pico has four 12-bit analog inputs

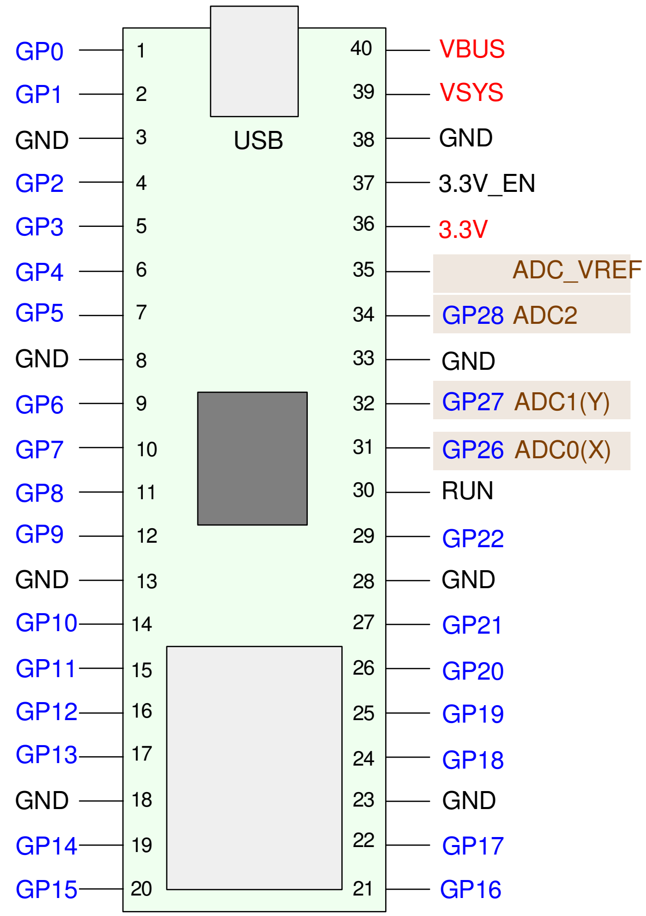
- Three are connected to the I/O pins

With these A/D inputs, you can

- Directly read sensors whose output is voltage
- Read sensors with resistance and current outputs (requires some circuitry).

This lecture looks at measuring acceleration and light, With these, we can

- Build a Magic 8-Ball,
- Measuring your Vertical Leap, and
- Build a solar tracker (electronic sunflower)



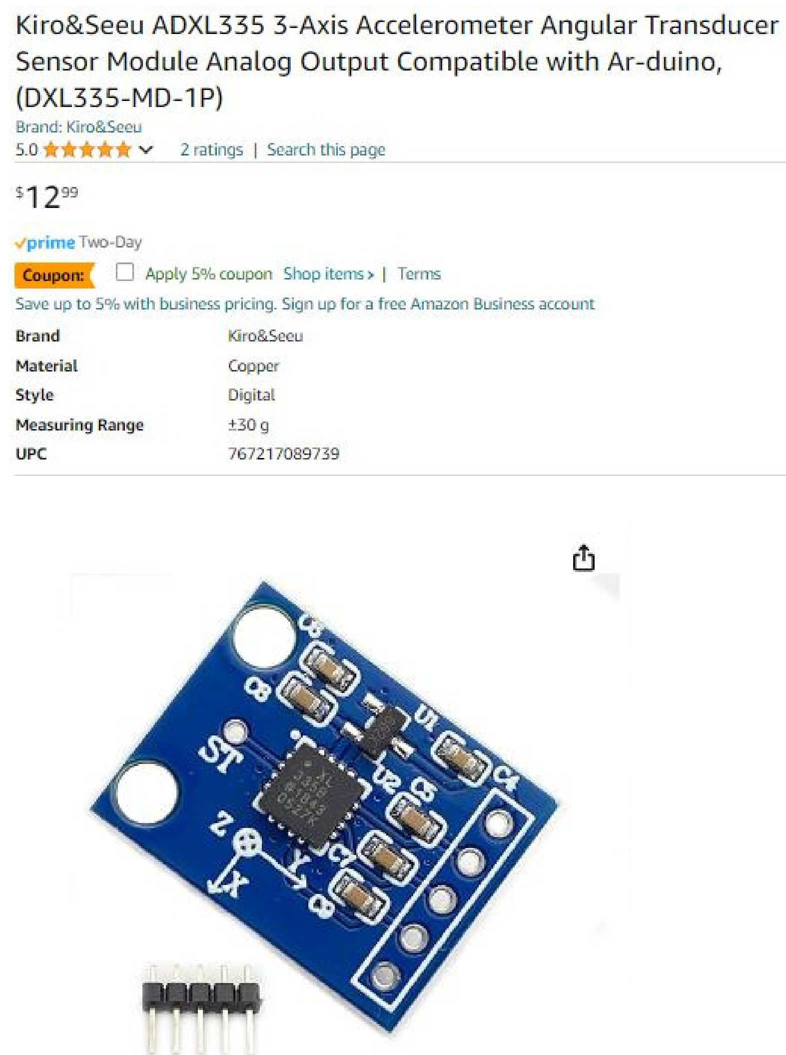
# Analog Acceleration (ADXL335)

The ADXL335 is an accelerometer with

- 3-5V operation,
- Operable over a full +/- 30g range
- With three analog outputs:
  - x'', y'', z''
  - 0G reads 0.7V
  - Sensitivity = 0.15V / g

Scaling for m/s2:

- offset = 0.7V
- gain = 200 / 65535



## Python Code

The nice thing about analog sensors is they're easy to figure out

- Give it power and ground
- You can then read the outputs

The analog outputs can be read

- With a volt meter,
- With an oscilloscope, or
- With the A/D inputs

Add a scaling factor and offset to make the units m/s2

```
from machine import ADC
from time import sleep_ms

a2d0 = ADC(0)
a2d1 = ADC(1)
a2d2 = ADC(2)

def Read_ADXL335():
    k = 200 / 65535
    x1 = a2d0.read_u16()
    y1 = a2d1.read_u16()
    z1 = a2d2.read_u16()

    Ax = k * (x1 - 13843)
    Ay = k * (y1 - 13779)
    Az = k * (z1 - 13843) - 3.6
    return([Ax, Ay, Az])

while(1):
    [Ax, Ay, Az] = Read_ADXL335()
    print(Ax, Ay, Az)
    sleep_ms(200)
```

Shell

```
       X          Y          Z
     0.146      0.000      9.556
    -0.098      0.000      9.605
     0.000     -0.146      9.458
```

# GY521 Digital Accelerometer

https://peppe8o.com/using-gyroscope-and-accelerometer-with-mpu6050-raspberry-pi-pico-and-micropython/

The GY-521 accelerometer is a digital accelerometer, capable of

- 3V to 5V operation
  - it will work with 3.3V
- +/- 2g up to +/- 16g accelerations
- +/- 250 to +/- 2000 degrees per second rotation
- Both I2C and SPI communications.

GY-521 MPU-6050 MPU6050 3-Axis Accelerometer Gyroscope Sensor Module 6-axis Accelerometer Gyroscope Sensor Module IIC I2C 3-5V for Arduino 5pcs

Visit the Teyleten Robot Store

4.0 15 ratings | Search this page

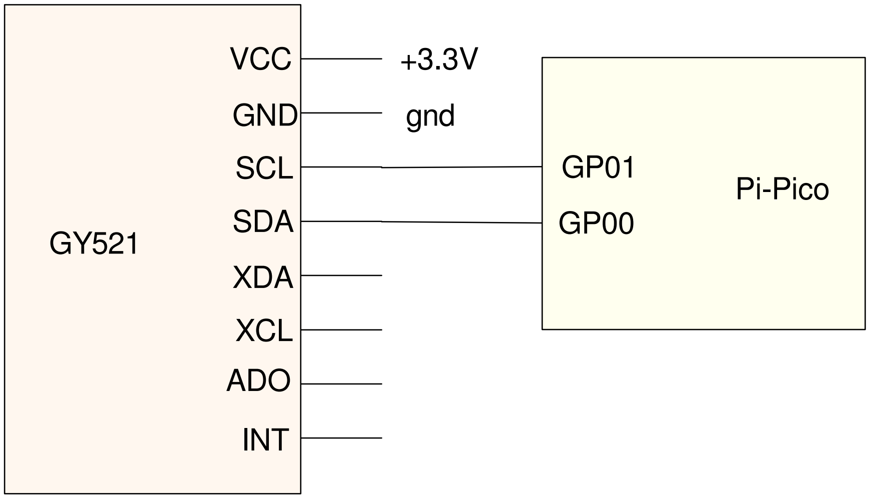
$11.88 ($2.38 / Item)

# GY521 Hardware

- https://peppe8o.com
- Very good tutorial on using the GY-521 with a Pi-Pico
- I2C interface

In terms of wiring for I2C,

- Vcc = 3.3V
- GND = 0V
- SCL = GP01
- SDA = GP00



# GY521 Software

https://peppe8o.com/using-gyroscope-and-accelerometer-with-mpu6050-raspberry-pi-pico-and-micropython/

Go to the Peppe80.com

- Download the driver libraries are loaded onto your Pi-Pico
  - imu.py
  - vector3d.py

You're then ready to read the acceleration and rotational rates

- see following code

The execution time is

- 1.96ms to read the acceleration in the XYZ direction
- 1.96ms to read the gyro (rotational velocities) about the XYZ axis

```
# Blog: https://peppe8o.com
# Date: Aug 25th, 2021
# Version: 1.0

from imu import MPU6050
import time
from machine import Pin, I2C

i2c = I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)
imu = MPU6050(i2c)

while True:
    ax=round(imu.accel.x,2)
    ay=round(imu.accel.y,2)
    az=round(imu.accel.z,2)
    gx=round(imu.gyro.x)
    gy=round(imu.gyro.y)
    gz=round(imu.gyro.z)
    tem=round(imu.temperature,2)
    print(ax,ay,az,gx,gy,gz,tem,end="\r")
    time.sleep(0.2)
```

shell

```
x"      y"    z"  gx  gy  gz  temp
0.19  0.0  0.98  -2  0  -2  28.41
```

## Magic 8-Ball

Now that we can measure acceleration, let's build a Magic 8-Ball

To operate the Magic 8-Ball,

- Shake the Magic 8-Ball
- Ask it a question, then
- Turn it over to see what the answer is.

# Code

- Full code is on Bison Academy
- Lecture #24

The following code uses the analog accelerometer to measure the acceleration in the z-direction.

- Start with the z-axis pointing down
- Ask a question
- Shake the accelerometer up and down briskly three times, then
- Turn it over to get your answer.

```
Fortune = ['Signs point to yes']

def Read_ADXL335():
    k = 200 / 65535
    z1 = a2d2.read_u16()
    Az = k * (z1 - 13843) - 3.6
    return(Az)

print('Shake for a reading')

while(1):
    print('--------')
    # Shake three times
    for i in range(0,3):
        #print('Shake #',i)
        while(Az < 20):
            Az = Read_ADXL335()
        while(Az > 5):
            Az = Read_ADXL335()
    #print('Your Fortune')
    N = randrange(20)
    print(Fortune[N])
    sleep(1)
```

Results:

- Shows off better in the video

## Measuring your Vertical Leap

You can also measure how high you can jump:

- When at rest, you should have an acceleration of 9.8 m/s2 due to gravity
- When in free-fall, the acceleration you feel should drop to zero m/s2

By measuring how long you are experiencing zero g's, you should be able to

- Measure the duration of  your jump, and then
- Calculate how high you jumped as

$$d = \frac{1}{2}at^2$$

where t is the time from your apogee to the ground, or

$$d = \frac{1}{2}a\left(\frac{t}{2}\right)^2$$

where t is the total time of your jump.

- Total time in free-fall (zero g's)

## Free-Fall Test:

Start with Heart Beat code

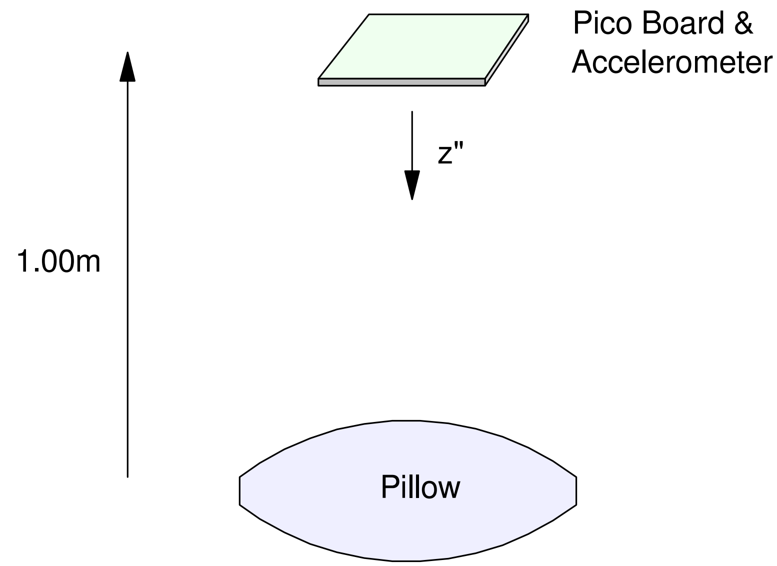
- Replace sensor with the ADXL335
  - a digital one would work too

Check the code works

- Drop the board 1.00m
- Sample every 5ms for 2 seconds
- Record acceleration as it falls

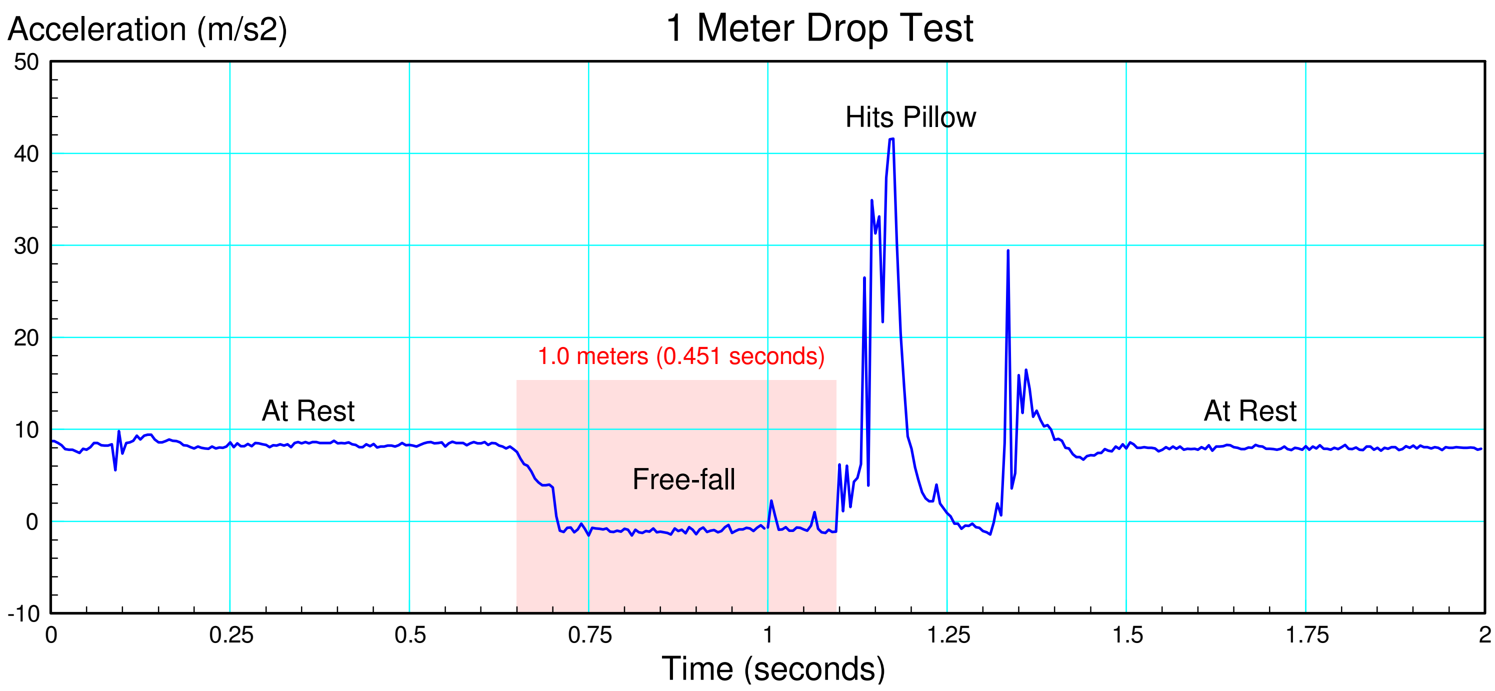
You should see

- Acceleration = 0.00 while falling
- Time of fall corresponds to 1.00m



# Free-Fall Results:

- Acceleration = 0 (ish) while in free fall
- Time with Az < 2.00 is 0.400 seconds
- Corresponds to a distance of 0.784m
  - sort of works



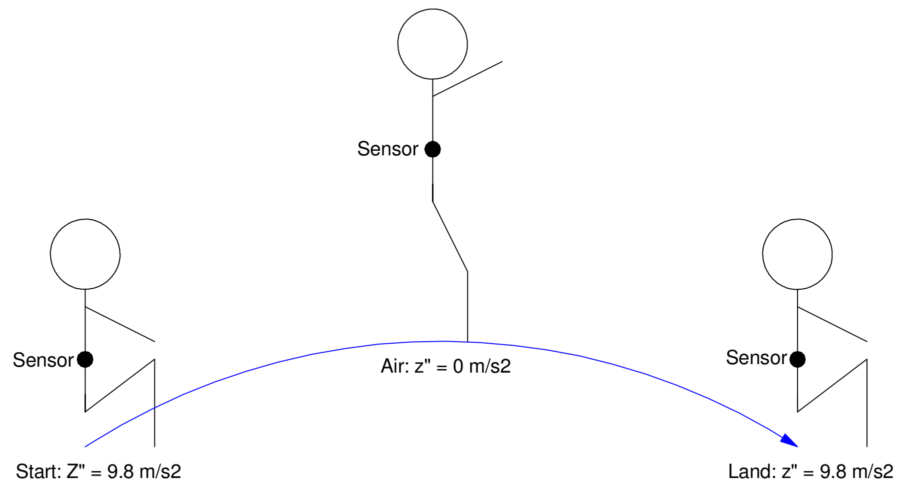
## Jump Test:

Repeat with the sensor on my waist

- Crouch down, getting ready
- Press GP15
  - Wait for a beep
- Jump as high as I can
  - about 14cm
  - hey, I'm getting old

Record the acceleration for 2 sec

- Sampling rate = 5.00ms



# Result: 14cm Jump

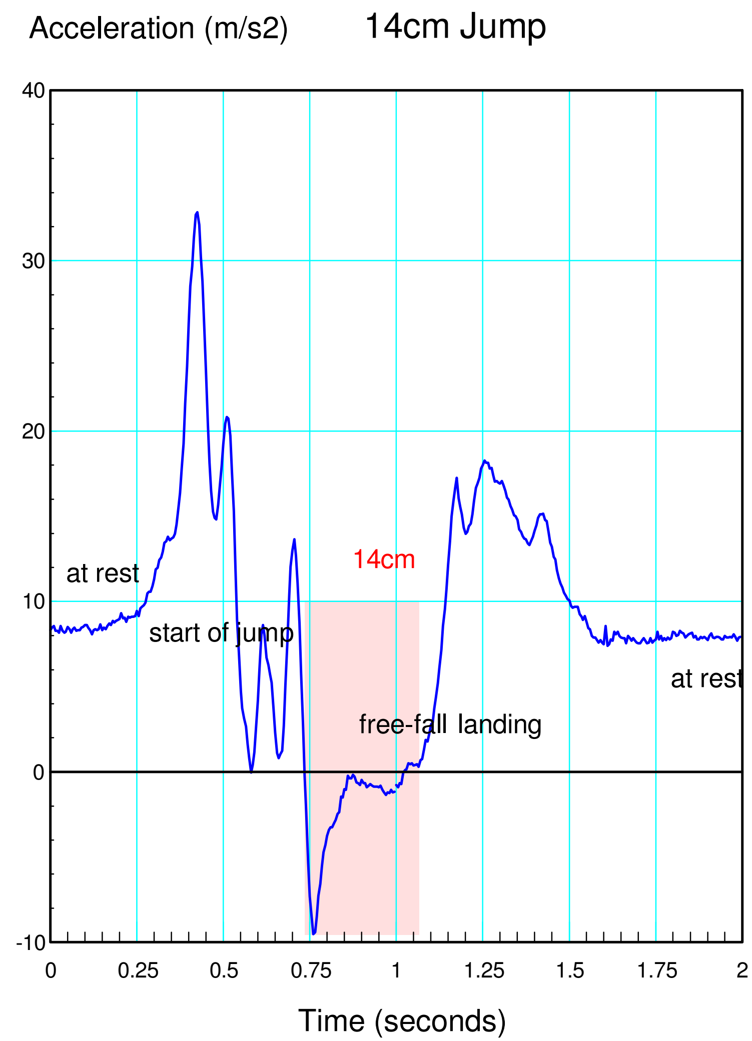
- Estimate from observation

From the data:

- "air time": Az < 1.00 m/s2
- Start of jump: 0.735 seconds
- End of jump: 1.080 second
- Duration = 345ms
- Height = 14.58 cm

which seems about right.

With a little more coding, the time of flight can be measured and displayed automatically.



# New Topic: Measuring Light

CdS Light Sensors

- Adafruit PDV-P8001
- Photo from Adafruit 161 Light Sensor

Recall...

- A thermistor is a sensor
  - Resistance changes with temperature

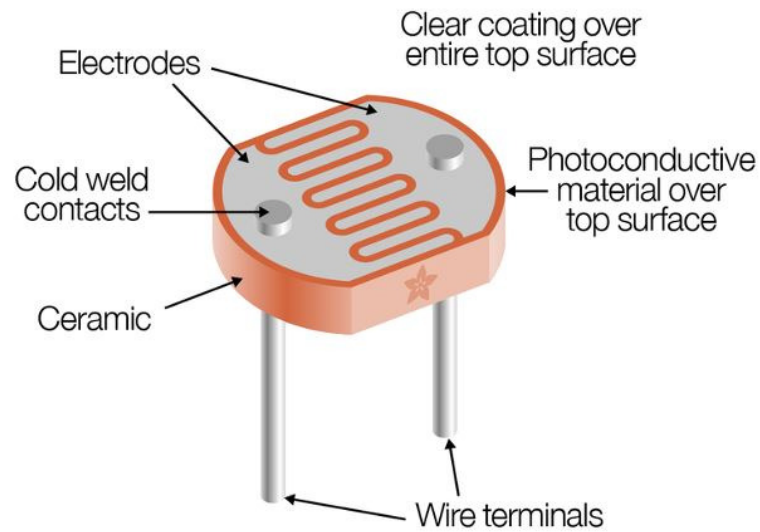
Replace the thermistor

- A sensor where R changes with light

and you have a light sensor

Cadmium Sulfide (CdS) is a chemical whose resistance changes with light level.

- This makes CdS a light sensor



# R vs. Lux Relationship

In general:

$$R = a \cdot lux^{-b}$$

Find {a, b} by

- Take two measurements
  - Two equations and two unknowns
- Look at the data sheets.

PDV-P8001 CdS from Adafruit

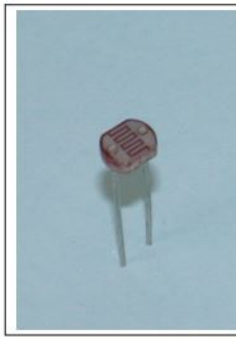
- At 10 Lux, 3k < R < 11k
- Sensitivity (b term)

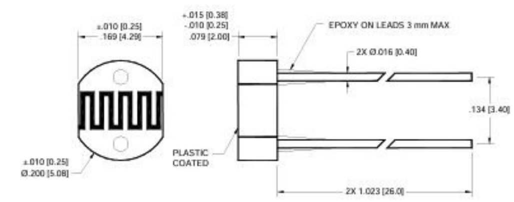
$$\frac{\ln(R_{100}) - \ln(R_{10})}{\ln(E_{100}) - \ln(E_{10})} = 0.6$$

Advanced Photonix, Inc.

***CdS Photoconductive Photocells***

**PDV-P8001**






**FEATURES**

- Visible light response
- Sintered construction
- Low cost

**DESCRIPTION**

The **PDV-P8001** are (CdS), Photoconductive photocells designed to sense light from 400 to 700 nm. These light dependent resistors are available in a wide range of resistance values. They're packaged in a two leaded plastic-coated ceramic header.

**APPLICATIONS**

- Camera exposure
- Shutter controls
- Night light Controls

**ABSOLUTE MAXIMUM RATING** (TA)= 23°C UNLESS OTHERWISE NOTED

| SYMBOL | PARAMETER | MIN | MAX | UNITS |
|---|---|---|---|---|
| $V_{pk}$ | Applied Voltage | | 150 | V |
| $P_{d\ \Delta p o/\Delta t}$ | Continuous Power Dissipation | | 100 | mW/°C |
| $T_O$ | Operating and Storage Temperature | -30 | +75 | °C |
| $T_S$ | Soldering Temperature* | | +260 | °C |

* 0.200 inch from base for 3 seconds with heat sink.

**ELECTRO-OPTICAL CHARACTERISTICS RATING** (TA)= 23°C UNLESS OTHERWISE NOTED

| SYMBOL | CHARACTERISTIC | TEST CONDITIONS | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| $R_D$ | Dark Resistance | After 10 sec. @ 10 Lux @ 2856 °K | 0.2 | | | MΩ |
| $R_I$ | Illuminated Resistance | 10 Lux @ 2856 °K | 3 | | 11 | KΩ |
| S | Sensitivity | LOG(R100)-LOG(R10)** / LOG(E100)-LOG(E10)*** | | 0.6 | | Ω/Lux |
| λrange | Spectral Application Range | Flooded | 400 | | 700 | nm |
| λpeak | Spectral Application Range | Flooded | | 520 | | nm |
| $t_r$ | Rise Time | 10 Lux @ 2856 °K | | 55 | | ms |
| $T_f$ | Fall Time | After 10 Lux @ 2856 °K | | 20 | | ms |

**R100, R10: cell resistances at 100 Lux and 10 Lux at 2856 °K respectively .
***E100, E10: luminances at 100 Lux and 10 Lux 2856 °K respectively.

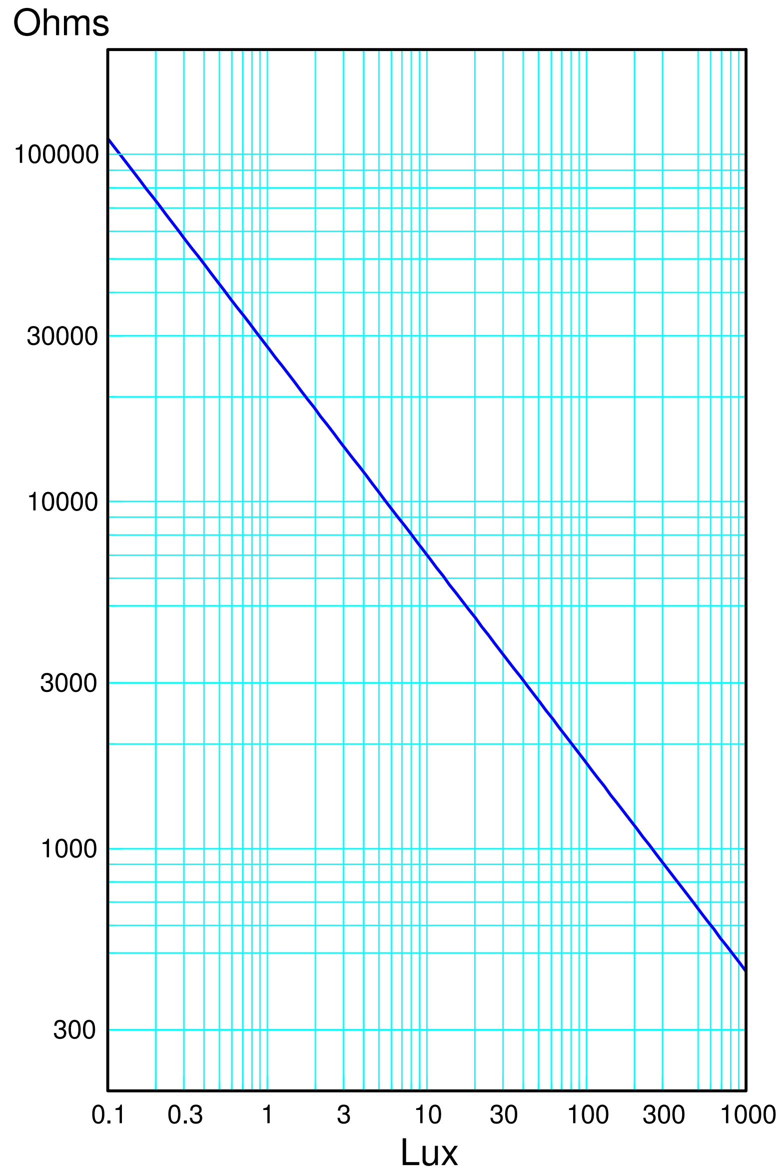
This allows you to solve for {a, b}

$$7000\Omega = a \cdot (10\ Lux)^{-0.6}$$

$$a = 27.873\Omega$$

meaning

$$R \approx 27,873 \cdot (lux)^{-0.6}$$



# Add a voltage divider

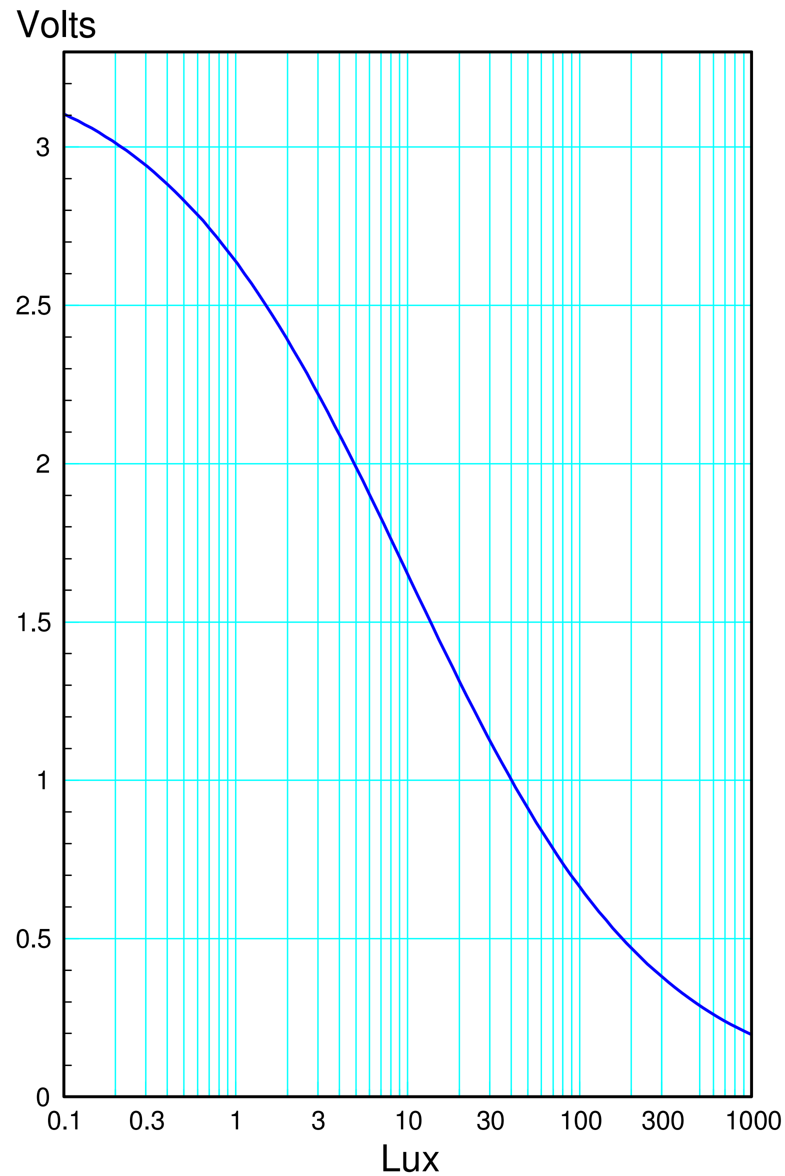
- Assume R = 7k (mid-band resistance)

```
>> Lux = logspace(-1,3,100)';
>> R = 27873 * Lux .^ -0.6;
>> V = R ./ (R + 7000) * 3.3
>> semilogx(Lux, V)
>> xlabel('Lux')
>> ylabel('Ohms')
```

By measuring voltage, you can calculate the light level

Note:

- This setup works best for 1..100 Lux
- Larger R tunes the sensor for lower light levels
- Smaller R tunes the sensor for higher light levels



# PhotoVoltaic (PV) Light Sensors

Another way to measure light is to use a photovoltaic solar cell.

- Convert sunlight to electricity
- Power (V*I) proportional to solar intensity (Watts / m2)

Solar cells are actually pretty efficient:

- 0.2% - 2% typical plants
- 1% - 2% typical crops
- 8% sugar cane
- 11% max possible with photosynthesis
- 13% - 23% commercial PV
- 39% best research grade PV

# Output of PV Cell in sunlight

- Nominal Sunlight = 1370 W/m2 (NASA)

PV cell can measure solar intensity

- Add a 47 Ohm resisistor as a load
- Measure voltage at different times of day
- Compute the energy produced
- Scale by cell's efficiency and area to get solar intensity

| Condition July 14, 2024 | Voltage | Power 47 Ohm resistor | Energy Density |
|---|---|---|---|
| Dawn (8am) | 2.08V | 92.05mW | 48% |
| Noon | 3.00V | 191mW | 100% |
| Evening: 6pm | 2.758V | 161.8mW | 84.7% |
| Dusk: 9pm | 0.15V | 0.47mW | 0.2% |

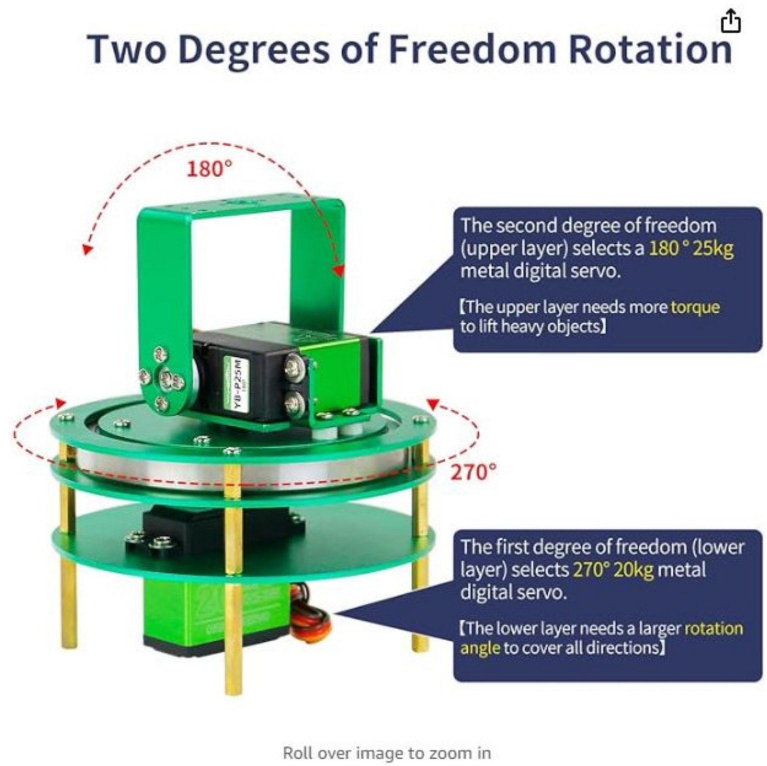
# Fun with Light Sensors

*Yahboom 2-dof kit from Amazon*

With a light sensor and a pan-tilt servo motor, you can build a

- A solar tracker
  - Point solar cells directly at the sun
  - Improves energy output
- An electronic sunflower
  - Point directly at the sun
  - Follow the sun throughout the day

*left as a homework or term project assignment*



## Summary

Acceleration sensors detect motion

- Both analog and digital are available
- Allow you to detect orientation, shaking, etc

Light sensors detect light levels

- Both resistive and voltage outputs available
- Allow you to detect room conditions
  - Light, Dark
- Allow you to track an object
  - Follow the sun

Both are pretty easy to interface to a Pi-Pico

# References


Pi-Pico and MicroPython

- https://github.com/geeekpi/pico_breakboard_kit
- https://micropython.org/download/RPI_PICO/
- https://learn.pimoroni.com/article/getting-started-with-pico
- https://www.w3schools.com/python/default.asp
- https://docs.micropython.org/en/latest/pyboard/tutorial/index.html
- https://docs.micropython.org/en/latest/library/index.html
- https://www.fredscave.com/02-about.html

Pi-Pico Breadboard Kit

- https://wiki.52pi.com/index.php?title=EP-0172

Other

- https://docs.sunfounder.com/projects/sensorkit-v2-pi/en/latest/
- https://electrocredible.com/raspberry-pi-pico-external-interrupts-button-micropython/
- https://peppe8o.com/adding-external-modules-to-micropython-with-raspberry-pi-pico/
- https://randomnerdtutorials.com/projects-raspberry-pi-pico/
- https://randomnerdtutorials.com/projects-esp32-esp8266-micropython/