**Red pitaya**

**Red Pitaya** is an [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware)  made to alternate for many expensive laboratory measurement and control instruments.

Red Pitaya offers open-source-software measurement and control tools that consists of easy-to-use visual programming software and free of charge, ready-to-use open-source, web-based test and measurement instruments running on powerful, credit card-sized boards. With a single click, the board can transform into a web-based oscilloscope, spectrum analyzer, signal generator, LCR meter, Bode analyzer, or one of many other applications. Red Pitaya can be controlled by using Matlab, LabVIEW, Python & Scilab.

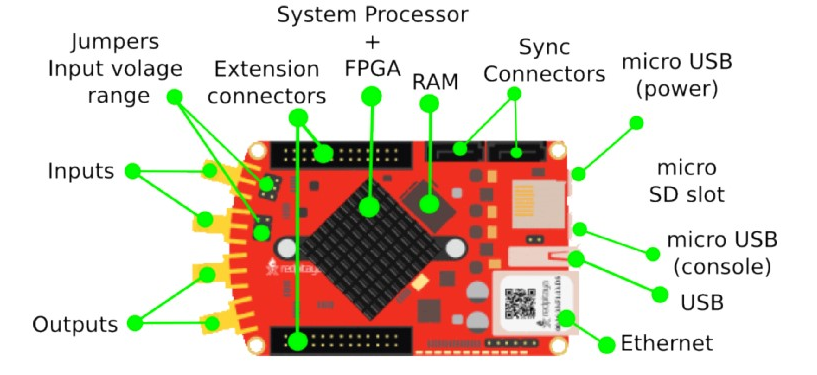
**Key Features**

* High performance hardware
* Replaces most essential instruments like Oscilloscope, Spectrum analyzer, Signal generator, LCR meter
* LAN or wireless access from any WEB browser via tablet or a PC regardless of the OS (MAC, Linux, Windows, Android, iOS) LabVIEW and MATLAB® interface.
* Possibility to make your own application and share it with others
* Open source software

**Software**  
Red Pitaya is based on GNU/Linux operating system and can be customized at different programming levels. Available software interfaces include: HDL, C/C++, scripting languages, MATLAB and HTML based web interfaces.

Red Pitaya software is open source and can be downloaded from GitHub. All development tools are free.

<https://shop.trenz-electronic.de/en/Products/Red-Pitaya/>



<https://redpitaya.readthedocs.io/en/latest/developerGuide/125-10/vs.html>

**Technologies Used**

**Programming Language**

1. Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991. This scripting language is amazing to use for machine learning purpose as it has many machine learning library / api support.

To write code in python we choose python version 3.7.3.

To make our solution running in python version 3.7.3 we have made our raspberry pi default python version from 2.7 to 3.7.3.

**CSV** file to store data getting from Red pitaya

**Paramiko** install for ssh connection. As red pitaya starts with ssh connection from red pitaya and we can do it either manually connecting ssh connection or can make it automated in programming. Do make it automated we installed paramiko library for establishing ssh connection between red pitaya and resberrypie

Matplotlib install to plot the signal coming from red pitaya. While doing the research with the signal or data we needed to plot to visualize the process.

Pygame install to play sound.

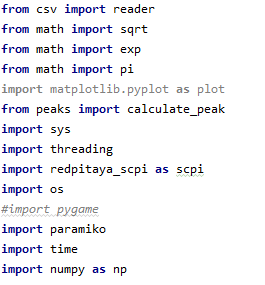
Pip install pygame

**NumPy** is the fundamental package for scientific computing with **Python. For example we did fft for the signal and also got the peaks of the fft. These are the functionality were achieved by numpy library.**

**Data Extraction: Data extraction** is the act or process of retrieving **data** out of **data** sources for further **data** processing or **data** storage. In our solution we continuously acquire 16k samples of signal on fast analog inputs from red pitaya and store it in csv file format and later we do the feature extraction to use the features for our machine learning part.

**Feature Extraction**: In [machine learning](https://en.wikipedia.org/wiki/Machine_learning) feature extraction starts from an initial set of measured data and builds derived values ([features](https://en.wikipedia.org/wiki/Feature_(machine_learning))) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction involves reducing the number of resources required to describe a large set of data. In our solution after we acquire data from red pitaya we extract feature from these data. For example, features can be peaks of the signal high peak or low peak, shape of the signal, length etc.

This writing we will use while write about object detection python script



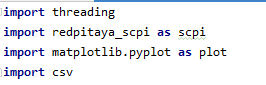
In our programming we have to use different library to get advantage of some complex mechanism.

1. Csv, library to read data and write from csv files
2. Math for mathematical squat root, exponential, pi and other operations in machine learning.
3. Matplotlib for plot the signal for more visualization.
4. Peaks in our own written python script
5. Threading for running asynchronous process continuously. As we need to acquire signal from redpitaya in every n seconds.
6. Redpitaya\_scpi to run all the scpi commands into redpitaya
7. Pygame to play sound on the background
8. Finally, numpy for doing more complex stochastic operation for example Fouriar Transform of a signal, mean, variance.

**Step 1**

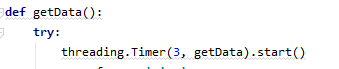
**Data** acquire from red pitaya: For supervise learning we first need to collect data from red pitaya to raspberry pie. Our main target is to recognize, human wall and car. To collect data from red pitaya which will feed in machine learning part we have to run our data\_get\_common\_code.py file. This file will continuously receive signal data from red pitaya and store into a csv format file for future use. Let’s see how the data acquisition works.

In our script we have imported all the necessary library will use while the script run. Figure 1 shoes all the necessary library



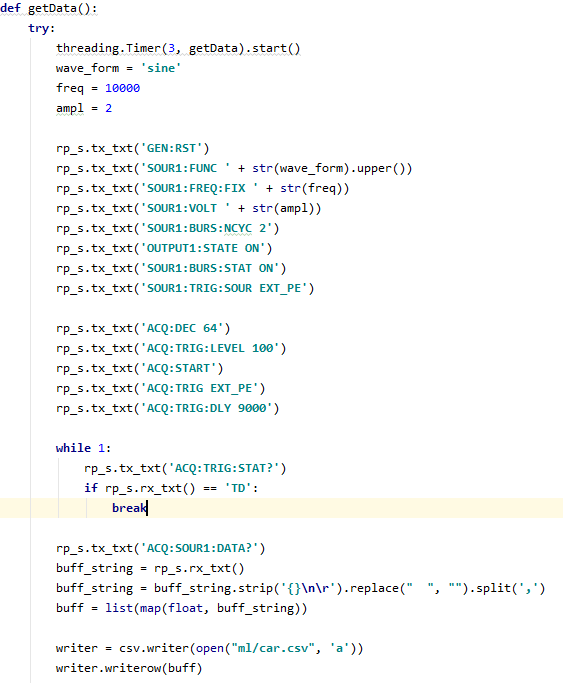
Now we will initialize scpi Object “rp\_s = scpi.scpi(**'192.168.128.1'**)” for sending scpi command to redpitaya. **192.168.128.1** is our redpitaya ip address.

The Standard Commands for Programmable Instruments (SCPI) defines a standard for syntax and commands to use in controlling programmable test and measurement devices, such as [automatic test equipment](https://en.wikipedia.org/wiki/Automatic_test_equipment) and [electronic test equipment](https://en.wikipedia.org/wiki/Electronic_test_equipment)



In this script out getData() function is call in every 3 second to get or acquire the signals from redpitaya. We used thread timer so that the method automatically in very

given seconds.

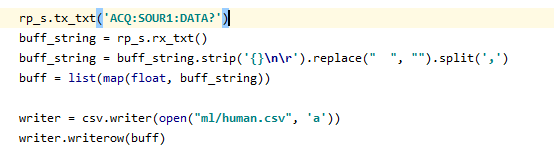


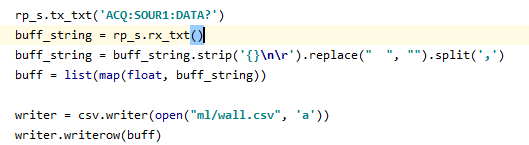
In this function we have used different scpi commands to send to the redpitaya. Let’s see what are the commands we run sequentially to acquire data.

|  |  |
| --- | --- |
| Commands | What is does |
| **GEN:RST** | Reset generator to default settings. |
| **SOUR<n>:FUNC <func> / SOUR1:FUNC** | Set waveform of fast analog outputs. |
| **SOUR1:FREQ:FIX** | Set frequency of fast analog outputs. |
| **SOUR1:VOLT** | Set amplitude voltage of fast analog outputs.Amplitude + offset value must be less than maximum output range ± 1V |
| **SOUR1:BURS:NCYC 2** | Set N number of periods in one burst. |
| **OUTPUT1:STATE ON** | Enable fast analog outputs. |
| **SOUR1:BURS:STAT ON** | Enable burst (pulse) mode. Red Pitaya will generate R number of N periods of signal and then stop. Time between bursts is P. |
| **SOUR1:TRIG:SOUR EXT\_PE** | Set trigger source for selected signal. |
| **ACQ:DEC 64** | Set decimation factor |
| **ACQ:TRIG:LEVEL 100** | Set trigger level |
| **ACQ:START** | Starts acquisition |
| **ACQ:TRIG EXT\_PE** | Disable triggering, trigger immediately or set trigger source & edge. |
| **ACQ:TRIG:DLY 9000** | Set trigger delay in samples |
| **ACQ:TRIG:STAT?** | Get trigger status |
| **ACQ:SOUR1:DATA?** | Read full buf |

Running these scpi command we continuously get buffered string data and store these in car.csv

We kept run the process until we received at least 100 data for car and then we continued the process to collect data for human and wall also. Code will be same accept for human data file will be human.csv and for wall file will be wall.csv





**Testing / Plug and Play**

<https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/>

To test our project just power resberri pie and redpitaya and wait for 40 seconds. Waiting for 40 second is important because resberry pie files, folder, services, wifi service all important services should be ready. We have made our script wait to resberry pie get ready. Point the redpitaya to a wall or human or a car. A monitor can be connected with resberri pie but it is not mandatory. We have 2 kind of output one is sound and other is printed name of the object. If we have any monitor connected with resberry pie just open the terminal that’s it, otherwise just connect a headphone or a speaker with resberry pie to listen which object is detecting.

Our object detection python code is completely automated and enough stable to predict Human, Wall and Car. To make the object detection code automated we added the object\_detection.py path in “/home/pi/.bashrc”

In bashrc last line we added sudo python /home/pi/object\_detection.py

Now whenever the raspberry pie will boot or the terminal open the object\_detection python script will run. As we continuously acquire data from red pitaya we must have a stable **ssh** connection between raspberry pie and red pitaya. Instead of manually doing it paramiko library helps us to make an automated stable ssh connection



We initially try to make ssh connection using ssh\_connection() function but if it fails because it is still not connected with redpitaya wifi, it will wait and continue to try in every 5 seconds. Finally, if it will success to make ssh connection it will never try again.