https://www.instructables.com/Super-Simple-Raspberry-Pi-433MHz-Home-Automation/

from datetime import datetime

import matplotlib.pyplot as pyplot

import RPi.GPIO as GPIO

The *RPi.GPIO* and *datetime*libraries were included with my Raspbian distribution, but I had to install the *matplotlib* library as follows:

sudo apt-get install python-matplotlib

<https://www.instructables.com/RF-433-MHZ-Raspberry-Pi/>

|  |  |  |
| --- | --- | --- |
|  |  |  |

**RECEIVE**

#!/usr/bin/env python3

import argparse

import signal

import sys

import time

import logging

from rpi\_rf import RFDevice

rfdevice = None

# pylint: disable=unused-argument

def exithandler(signal, frame):

rfdevice.cleanup()

sys.exit(0)

**logging.basicConfig(level=logging.INFO, datefmt='%Y-%m-%d %H:%M:%S',**

**format='%(asctime)-15s - [%(levelname)s] %(module)s: %(message)s', )**

parser = argparse.ArgumentParser(description='**Receive dec code via 433/315MHz GPIO dev**')

parser.add\_argument('-g', dest='gpio', type=int, default=27, help="**GPIO pin (Def: 27)**")

args = parser.parse\_args()

signal.signal(signal.SIGINT, exithandler)

rfdevice = RFDevice(args.gpio)

rfdevice.enable\_rx()

timestamp = None

logging.info("**Listening for codes on GPIO** " + str(args.gpio))

while True:

if rfdevice.rx\_code\_timestamp != timestamp:

timestamp = rfdevice.rx\_code\_timestamp

**logging.info(str(rfdevice.rx\_code) +**

**" [pulselength " + str(rfdevice.rx\_pulselength) +**

**", protocol " + str(rfdevice.rx\_proto) + "]")**

time.sleep(0.01)

rfdevice.cleanup()

**SEND**

#!/usr/bin/env python3

import argparse

import logging

from rpi\_rf import RFDevice

**logging.basicConfig(level=logging.INFO, datefmt='%Y-%m-%d %H:%M:%S',**

**format='%(asctime)-15s - [%(levelname)s] %(module)s: %(message)s',)**

parser = argparse.ArgumentParser(description=**'Sends decimal code via 433/315MHz GPIO**')

parser.add\_argument('code', metavar='CODE', type=int, help="**Decimal code to send**")

parser.add\_argument('-g', dest='gpio', type=int, default=17, help="**GPIO pin (Def: 17)**")

parser.add\_argument('-p', dest='pulselength', type=int, default=None,

help="**Pulselength (Default: 350)**")

parser.add\_argument('-t', dest='protocol', type=int, default=None, help="**Prot (Defa: 1)**")

args = parser.parse\_args()

rfdevice = RFDevice(args.gpio)

rfdevice.enable\_tx()

if args.protocol:

protocol = args.protocol

else:

protocol = "**default**"

if args.pulselength:

pulselength = args.pulselength

else:

pulselength = "**default**"

**logging.info(str(args.code) +**

**" [protocol: " + str(protocol) +**

**", pulselength: " + str(pulselength) + "]")**

rfdevice.tx\_code(args.code, args.protocol, args.pulselength)

rfdevice.cleanup()