

Textul si imaginile din acest document sunt licentiate

Attribution-NonCommercial-NoDerivs
CC BY-NC-ND



Codul sursa din acest document este licentiat

Public-Domain

Esti liber sa distribui acest document prin orice mijloace consideri (email, publicare pe website / blog, printare, sau orice alt mijloc), atat timp cat nu aduci nici un fel de modificari acestuia. Codul sursa din acest document poate fi utilizat in orice fel de scop, de natura comerciala sau nu, fara nici un fel de limitari.

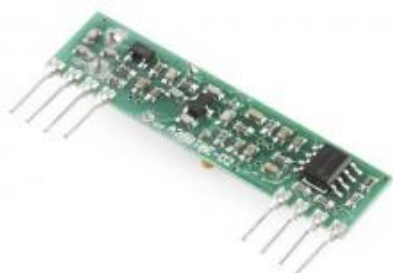
Realizarea unui sistem de tip Home Automation (Partea a V-a)

Conectarea unor dispozitive antiefracție la platforma OpenHab

Utilizând instalarea platformei OpenHab realizată în lecția anterioară vom adăuga sistemului nostru de tip Home Automation o serie de dispozitive antiefracție îmbogățind astfel funcționalitatea sistemului. Cele mai simple și ieftine dispozitive de securitate sunt furnizate de sistemele de alarmă fără fir domestice: senzor de deschidere a ușii, senzor de mișcare (PIR), senzor de vibrație pentru protecția ferestrelor, senzor de fum sau chiar de inundație. Avantajul acestor dispozitive este costul scăzut, disponibilitatea de a fi achiziționate separat de un sistem de securitate complet (ca piese de schimb sau componente de extindere a sistemelor de securitate existente) și de ce nu, avantajul unor dispozitive gata realizate și ușor de integrat într-un sistem personalizat propriu. Un material interesant despre integrarea acestui tip de dispozitive în cadrul unei sistem de alarmă bazat pe platforma Arduino este proiectul **Home Watch** din cadrul cărții ”10(zece) proiecte cu Arduino”:



<https://books.google.ro/books?id=aMB8BgAAQBAJ>

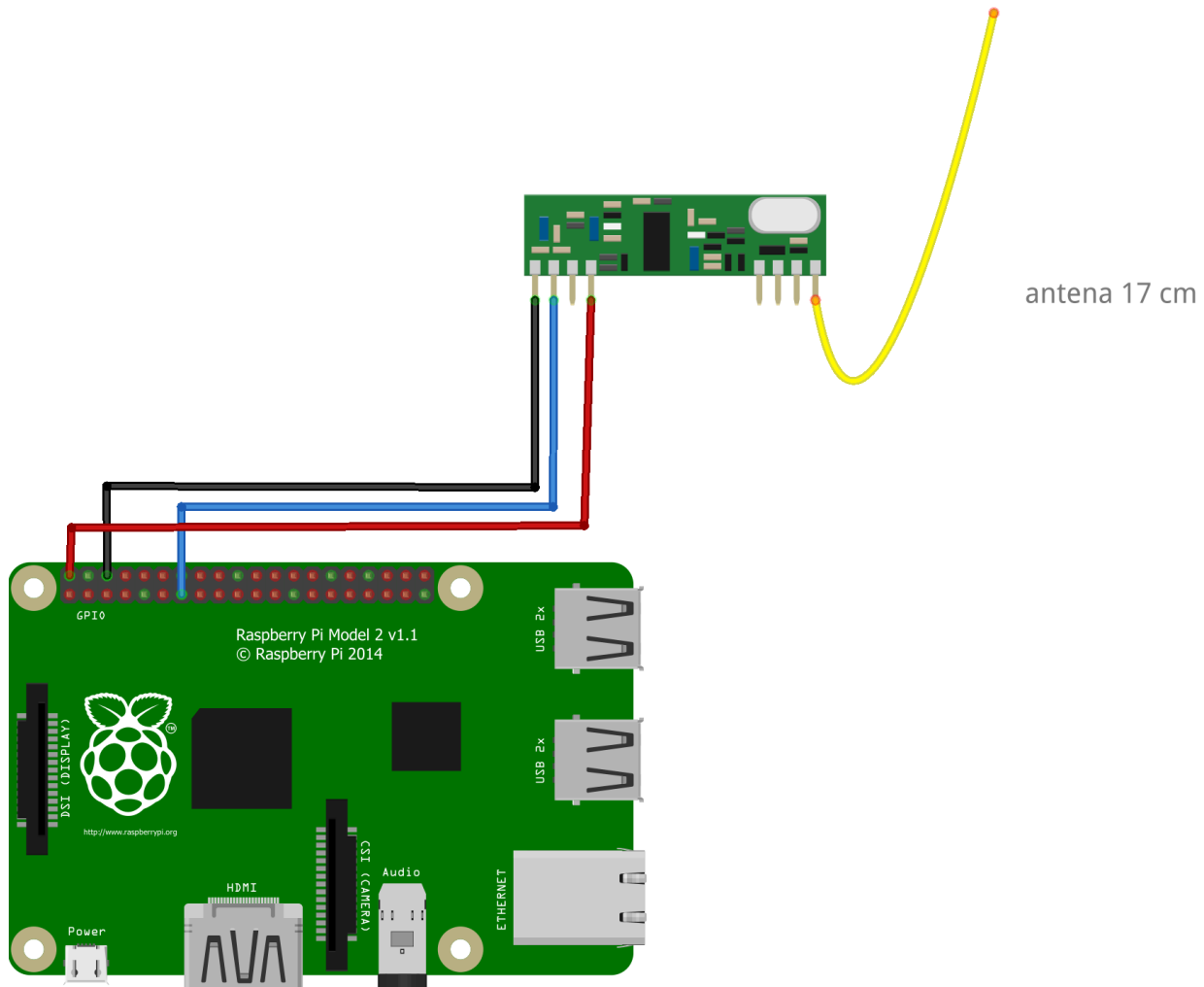


Dispozitivele de securitate de acest tip funcționează pe baza comunicației radio în bandă ISM de 433MHz modulație ASK și pot fi ”interceptate” utilizând un receptor radio foarte ieftin și ușor de utilizat atât în combinație cu platforma Arduino cât și cu placa de dezvoltare Raspberry Pi:

https://www.robofun.ro/wireless/wireless-433/receptor_radio

Conectarea receptorului radio la placa de dezvoltare Raspberry Pi necesită realizarea a trei conexiuni:

- Pinul de GND al modului radio la pinul 6 (GND) al plăcii Raspberry Pi;
- Pinul Data al modului radio la pinul 11 al plăcii Raspberry Pi;
- Pinul Vcc al modului radio la pinul 2 (5V) al plăcii Raspberry Pi.



Pentru mai multe detalii legate de conectarea modului radio la placa Raspberry Pi se pot consulta și următoarele resurse on-line:

pi-switch: Send and receive 433MHZ signals with a Raspberry Pi

<http://lexruee.ch/2015/07/31/pi-switch:-Send-and-receive-433MHZ-signals-with-a-Raspberry-Pi.html>

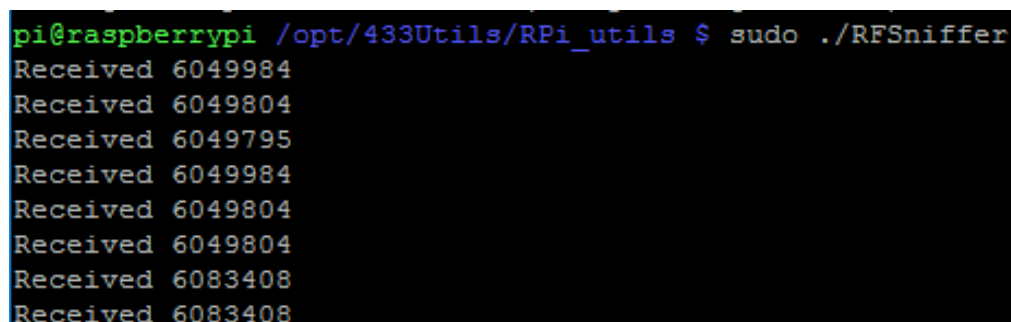
Decode 433 MHz signals w/ Raspberry Pi & 433 MHz Receiver

<http://www.princetronics.com/how-to-read-433-mhz-codes-w-raspberry-pi-433-mhz-receiver/>

Pentru implementarea comunicației între modulul radio și placa de dezvoltare Raspberry Pi vom instala suita de utilitare *433Utils* bazată pe biblioteca *rc-switch*. Pentru acest lucru vom rula următoarele comenzi (se presupune că avem instalat utilitarul *git*, dacă nu vom rula și instrucțiunea `sudo apt-get install git-core`):

```
cd /opt
sudo git clone --recursive git://github.com/ninjablocks/433Utils.git
cd 433Utils/RPi_utils/
sudo make
sudo ./RFSniffer
```

Rularea utilitarului *RFSniffer* ne va permite să vizualizăm codurile transmise de dispozitivele antiefracție cu care dorim să lucrăm:



```
pi@raspberrypi /opt/433Utils/RPi_utils $ sudo ./RFSniffer
Received 6049984
Received 6049804
Received 6049795
Received 6049984
Received 6049804
Received 6049804
Received 6083408
Received 6083408
```

Pentru a putea face legătura între utilitarul *RFSniffer* (între codurile radio transmise de dispozitivele de securitate) și platforma OpenHab trebuie să transformăm utilitarul într-un serviciu al sistemului de operare – să-l facem să pornească odată cu sistemul și să ruleze în fundal în mod continuu.

Primul pas este modificarea codului sursă al utilitarului (*RFSniffer.cpp*) după cum urmează:

linia 20:

```
printf("Received %i\n", mySwitch.getReceivedValue() );
```

se va modifica în

```
printf("%i\n", mySwitch.getReceivedValue() );
```

iar înainte de linia 28 (instrucțiunea `mySwitch.resetAvailable();`) vom insera instrucțiunea:

```
fflush(stdout);
```

La final vom recompila utilitarul cu ajutorul comenzii *make* prezentă în succesiunea de comenzi anterioară apoi vom crea un director în care vom muta utilitarul RFSniffer și vom crea scripturile de legătură cu platforma OpenHab.

```
cd /opt
sudo mkdir 433
sudo cp /opt/433Utils/RPi_utils/RFSniffer /opt/433
sudo chmod -R 755 433
```

Pentru a porni în mod automat vom crea în directorul */etc/init.d/* fișierul *RFSniffer* cu următorul conținut:

```
#!/bin/sh
# /etc/init.d/RFSniffer

### BEGIN INIT INFO
# Provides:          RFSniffer
# Required-Start:    $all
# Required-Stop:     $all
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: RFSniffer log
# Description:       RFSniffer log
### END INIT INFO

case "$1" in
    start)
        echo "Starting RFSniffer"
        /opt/433/RFSniffer > /opt/433/sniff.txt &
        ;;
    stop)
        echo "Stopping RFSniffer"
        killall RFSniffer
        ;;
    *)
        echo "Usage: /etc/init.d/RFSniffer {start|stop}"
        exit 1
        ;;
esac
```

și vom integra acest script în sistemul de inițializare a sistemului de operare:

```
sudo chmod 755 /etc/init.d/RFSniffer
sudo update-rc.d RFSniffer defaults
```

Acum, la repornirea sistemului, utilitarul *RFSniffer* va rula în fundal în mod automat și va trimite toate codurile radio primite în fișierul */opt/433/sniff.txt* .



În cadrul lecției de față vom integra cu platforma OpenHab doi senzori de ușă. Un astfel de senzor, bazat pe un red-switch – un întrerupător de proximitate, emite un anumit cod radio în poziția deschis semnalizând astfel deschidere ușii pe care a fost montat. Vom utiliza doi senzori pentru a prinde atât poziția de deschis cât și cea de închis – ușa închisă va declanșa un senzor iar ușa deschisă va declanșa celălalt senzor. Cu alte cuvinte vom avea două coduri radio pe care le vom interpreta în platforma OpenHab.

Pentru a face acest lucru vom crea fișierul *sniff* în directorul */opt/433/* cu următorul conținut:

```
aux=$(tail -n 1 /opt/433/sniff.txt);
if [ $aux == "6049804" ]; then
    echo "OPEN";
elif [ $aux == "6049984" ]; then
    echo "CLOSE";
fi
```

Acest fișier script are rolul de a transforma codurile numerice (6049804 – codul transmis de senzorul ce indică ușă deschisă, 6049984 – senzorul ce indică ușă închisă) în cuvintele cheie OPEN și CLOSE. Acest script va fi apelat de fișierele de configurare ale platformei OpenHab. Astfel în fișierul */etc/openhab/configurations/items/casamea.items* vom adăuga următoarea linie:

```
String    USA          "Ușa intrare [%s]"
          {exec="<[/bin/bash@@-c@@/opt/433/sniff:3000:REGEX((.*?))]"}
```

iar în fișierul */etc/openhab/configurations/sitemaps/casamea.sitemap* :

```
Frame label="Securitate" {
    Text item=USA icon="door-closed"
}
```

Pentru ca apelul scriptului extern să funcționeze este necesară instalarea componentei suplimentare *binding-exec* specifică platformei OpenHab:

```
sudo apt-get install openhab-addon-binding-exec
```

La accesarea interfeței OpenHab noul element (USA) se va regăsi sub forma următoare:



După același principiu se pot integra și alte tipuri de senzori specifici sistemelor de alarmă domestice ce funcționează în bandă radio ISM.

Bineînțeles, posibilitățile funcționale ale platformei OpenHab nu se limitează la cazurile expuse în suita de lecții de față dar sperăm că prezentarea acestor exemple să trezească interesul pentru această platformă extrem de puternică. Câteva exemple de alte proiecte ce utilizează platforma OpenHab:

Building a Home Automation System with OpenHAB to Control LEDs Wirelessly
<http://makezine.com/projects/building-a-home-automation-system-with-openhab-to-control-leds-wirelessly/>

\$20 Wireless Arduino Home Automation w/ OpenHAB
<https://hackaday.io/project/1720-20-wireless-arduino-home-automation-w-openhab>

Home automation using RF mesh network and arduino
<https://hackaday.io/project/5691-home-automation-using-rf-mesh-network-and-arduino>

Arduino openHAB Garage Door Control
<https://hackaday.io/project/4027-arduino-openhab-garage-door-control>

Hacking the Wink Hub to be more user-friendly for use with OpenHab and other home automation stacks
<https://hackaday.io/project/4368-winkhub-integration>

MQTT WiFi controlled mains halogen dimmer
<https://hackaday.io/project/9093-mqtt-wifi-controlled-mains-halogen-dimmer>

My openHAB Raspberry Pi Arduino XBee Led Zeppelin
<https://hackaday.io/project/8487-my-openhab-raspberry-pi-arduino-xbee-led-zeppelin>

A story of home automation with openHAB, Z-Wave, and MQTT
<http://jpmens.net/2014/01/14/a-story-of-home-automation/>

5 Open Source Home Automation Projects We Love
<https://www.fastcompany.com/3038442/elasticity/5-open-source-home-automation-projects-we-love>