ZigBee Topology Project

Bo Chen

November 12, 2015

Abstract

This project tries to get the ZigBee network topology and show the network map to the user. This can help the user know more about the network traffic.

Key Words: ZigBee, Python3

1 Getting Started

To start to use this project, you need to install a few new tools with your computer.

- Python3
- Graphviz(with python module installed)
- PySerial(python uart module)
- Image viewer

For beginners, you can use the pip to install graphviz and PySerial into python libs. If you are user of Ubuntu, you can just use apt-get tool to install graphviz. If you use windows, you need to configure the graphviz into system path environment manually.

2 Burn Z-Stack Image

After you have install tools, you can burn the image into ZigBee device. This section is out of the scope. You can find more about this section online.

One thing you need to take care: burn more than one device. The more device you burn, the more complex the topology is.

3 Run

Download this project and save them into you local disk. Before you can run the program, you should use the uart-to-usb cable to connect the ZigBee network coordinator with your computer.

1. close all ZigBee device
2. run the python program downloaded before
3. start the other devices in order
4. look up the generated topology image

When the program started, every time a new ZigBee device join in current network, the program would automatically update the topology image.

4 How This Works?

Every time a device join in current network, it will set the short address which is also called network address and it's parent address to the coordinator. After receiving the package from newly joined device, the coordinator deliver received package to computer using connected cable. The program analize the package and update topology image.

4.1 Package Format

We just talk about the playload here. Because the short address is 16 bits, so here we will show the four byte length package.

high 16 bit		low 16 bit	
SA_L8	SA_H8	PSA_L8	PSA_H8

SA_L8 : low 8 bits of self short address
SA_H8 : high 8 bits of self short address
PSA_L8 : low 8 bits of parent short address
PSA_H8 : high 8 bits of parent short address

4.2 Recover Address

This program use the method below to recover source address:

```
def read_run(ser_io):
    "read the data from the uart"
    while 1:
        relation = ser_io.read(4);
        self_address = hex(relation[1]) + hex(relation[0])[2:]
        parent_address = hex(relation[3]) + hex(relation[2])[2:]
        relation_queue.put((self_address, parent_address))
        time.sleep(.2)
```

5 The End

Welcome to give me some advice when you encounter a problem with this tiny interesting program. You can email at me: 294101042@qq.com.