What the Smart Home IoT Router Should be like

修订历史

版本	日期	原因	作者
V1.00	2016/06/13	Recipe	Jiankai.li

Contents

1. DIY a Versatile Home Router

1.1 Brief

This is a smart home IoT Router with a thermal printer as well as Grove Sensors and Grove actuators, you can write something in the web page of your local network, the printer will print it out. You can also control the relay and read the Grove Sensor values remotely.

1.2 About this project

A few months ago, I shared my project about Router named Smart router with WiFi Connection Visualization. http://www.instructables.com/id/Make-a-Colorful-Smart-Router/ which can tell you how many devices connected to your router and how fast the net speed, many people thought it was so cool to make a router according to their own situations.

This time, I made a router again for my grandpa who had Alzheimer's disease. Recently, every morning when he got up always forget what should he do that day. So, I made a router connecting a thermal printer, every night before he go to bed, he can write something that should to be done next day in the web page and the printer will print it out. Then every morning that he should to do is to look what the printer has already print out. Other people can also write something that they want to tell grandpa in the web page as well. Take me for example, I always write I love him every day before he gets up.

In addition, the router also connected Grove Sensors and Grove Relay. I can control the relay and upload the sensor values to MediaTek Cloud Sandbox.

I hope that there will be more people pay attention to this open source project. Let's do something special for special people like my grandpa who had Alzheimer's disease.

Well, now let's get started the journey of making.

You can find this project on INSTRUCTABLES website.
https://www.instructables.com/id/ReRouter-Make-an-Extensible-IoT-Router/

1.3 What do we need

- LinkIt Smart 7688 Duo

http://www.seeedstudio.com/item_detail.html?p_id=2574

- Grove – Led Strip

http://www.seeedstudio.com/item_detail.html?p_id=1665

- Grove - Temperature&Humidity Sensor (HDC1000)

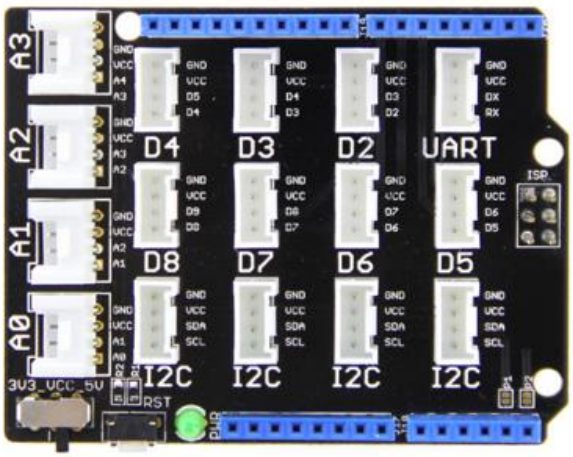
http://www.seeedstudio.com/item_detail.html?p_id=2535

- Grove – Relay

http://bz.seeedstudio.com/depot/Grove-Relay-p-769.html

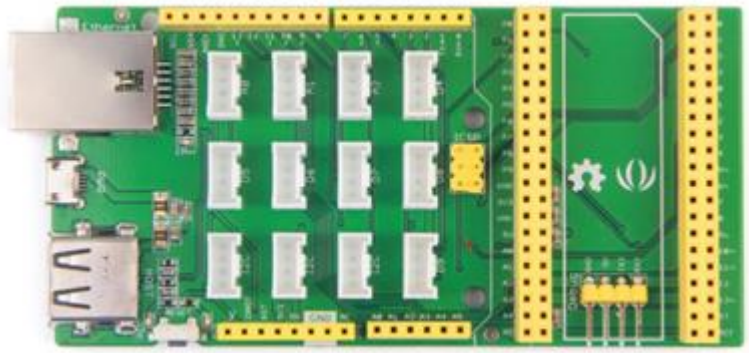
- Base Shield V2

http://www.seeedstudio.com/item_detail.html?p_id=1378



- Arduino Breakout for LinkIt Smart 7688 Duo

http://www.seeedstudio.com/item_detail.html?p_id=2576

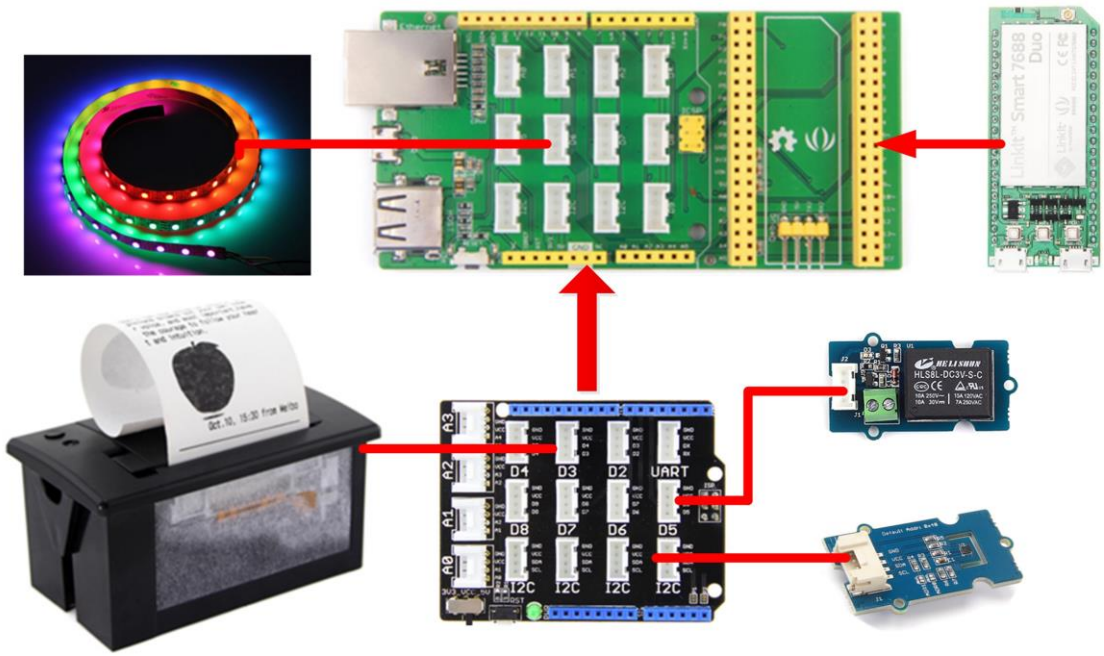


- Embedded Thermal Printer

http://www.seeedstudio.com/item_detail.html?p_id=1621

1.4 Wiring

Now, we should connect the Grove Modules to the Arduino Breakout for LinkIt Smart 7688 Duo just as below picture shows.



Modules	Port
Grove – Led Strip	D6 port on Arduino Breakout
Grove – Relay	D5 port on Base Shield
Grove - Temperature&Humidity Sensor (HDC1000)	I2C Port on Arduino Breakout
Embedded Thermal Printer	D3 port on Arduino Breakout

Modules	Port
Grove – Led Strip	D6 port on Arduino Breakout
Grove – Relay	D5 port on Base Shield
Grove - Temperature&Humidity Sensor (HDC1000)	I2C Port on Arduino Breakout
Embedded Thermal Printer	D3 port on Arduino Breakout

1.5 Software Work

The software work of the router consists of 5 parts, compatible with the previous router, control the relay from Mediatek Cloud Sandbox(MCS), upload the Grove Sensors to MCS, a website to write something for printer to print out, the code on ATmega32U4.

1.5.1 Setup the Linkit Smart 7688 Duo

I assume you have been familiar with Linkit Smart 7688 Duo, if not, please click here to get started. http://www.seeedstudio.com/wiki/LinkIt_Smart_7688_Duo, then go to the MT7688 terminal by SSH or serial.

Modify the configure file to change MT7688 into router mode.

```
vi /etc/config/network
```

Change the configure interface ‘lan’ and ‘wan’ as below shows.

```
config interface 'lan'
    option proto 'static'
    option netmask '255.255.255.0'
    option ipaddr '192.168.100.1'
```

```
•
config interface 'wan'
   option ifname 'eth0'
   option proto 'dhcp'
```

```
root@mylinkit:/# vi /etc/config/network

config interface 'loopback'
        option ifname 'lo'
        option proto 'static'
        option ipaddr '127.0.0.1'
        option netmask '255.0.0.0'

config globals 'globals'
        option ula_prefix 'fd90:bald:c0a4::/48'

config interface 'lan'
        option proto 'static'
        option netmask '255.255.255.0'
        option ipaddr '192.168.100.1'

config switch
        option name 'switch0'
        option reset '1'
        option enable_vlan '0'

config interface 'wan'
        option ifname 'eth0'
        option proto 'dhcp'
```

We use the yunbridge for communicating between MT7688 and ATmega32U4, so, enable the yunbridge.

```
> uci set yunbridge.config.disabled='0'
> uci commit
```

Requests is the only Non-GMO HTTP library for Python, safe for human consumption. Requests allows you to send organic, grass-fed HTTP/1.1 requests, without the need for manual labor, I use it to communicate with MCS.

I use flask web framework for this Web APP, Flask is a lightweight Python web framework based on Werkzeug, Jinja 2 and good intentions. You can click http://flask.pocoo.org/ to learn more information.

Next, install Python dependencies.

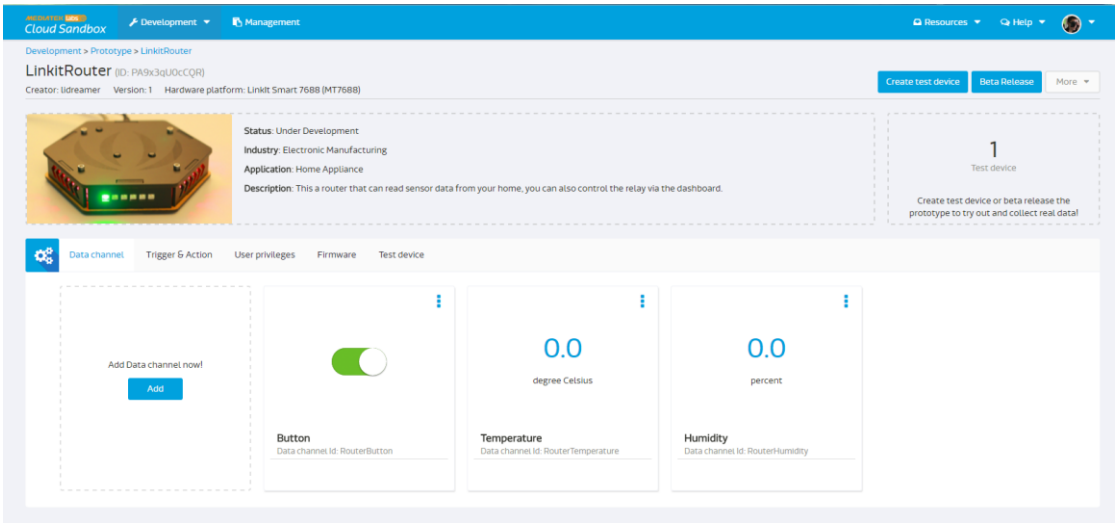
```
pip install requests flask Flask-WTF wtforms
```

1.5.2 Configure the MediaTek Cloud Sandbox

MediaTek Cloud Sandbox (MCS) is a cloud based data service platform for Internet of Things devices. https://mcs.mediatek.com/ If you don't have an account please click https://mcs.mediatek.com/oauth/en/signup to sign up.

MCS has already provided some useful tutorials, you can create the prototype and test device in MCS step by step following https://mcs.mediatek.com/resources/latest/tutorial/7688_tutorial

Then in my project I create a new prototype for Linkit Smart 7688 named LinkitRouter, and add three data channels for this prototype they're Button Temperature and Humidity just as picture below shows.

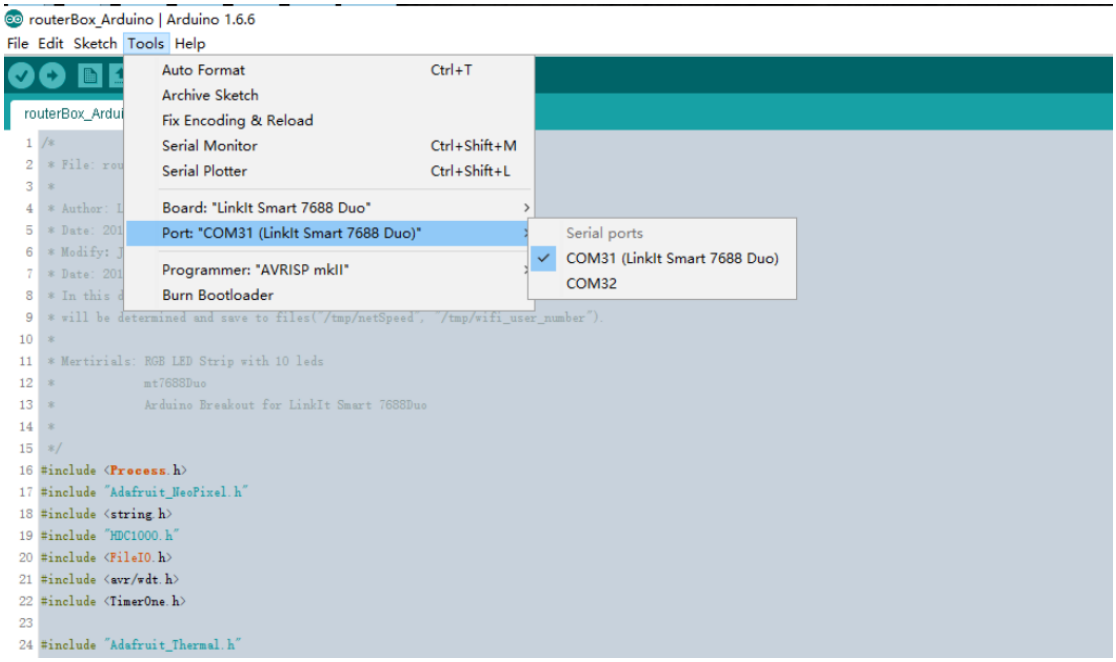


Now, you'll get the Device ID, Device Key, and 3 Data Channel ID, you can find the information in the test device detail page, and we will need them later.

Name	Value	Remark
deviceId	Dsre1qRQ	Unique Identifier for this Test Device, copy your own deviceId in the device detail page.
deviceKey	DFbtsNWg4AuLZ30v	Unique API Key for this Test Device, copy your own deviceKey in the device detail page.
dataChannelId	RouterButton	Data Channel ID for relay control
dataChannelId	RouterTemperature	Data Channel ID for reading temperature values
dataChannelId	RouterHumidity	Data Channel ID for reading humidity values

1.5.3 Upload the Arduino Code

1. Download demo code at https://github.com/Lee-Kevin/20.IoTRouter
2. Click "Download zip" button on right side of webpage to download all codes.
3. Decompress the downloaded zip files to"C:\Users\Administrator\Documents\Arduino\" and remove "-master" in decompressed file name.
4. Launch Arduino IDE.
5. Click Sketch>Add file to add routerBox_Arduino.ino file from"C:\Users\Administrator\Documents\Arduino\20.IoTRouter\RouterArduinoCode\routerBox_Arduino.ino"
6. Click Tools -> Board and select "Linkit Smart 7688 Duo" and chose the right port, as shown in the picture below.



7. Press CTRL +U to upload codes to your board. Wait a while, there will be saying “Done uploading”.
8. Congratulations, you have already completed the whole of the work.

1.5.4 Download the code

Now, we have already install the requirements, download the code from github.

cd ~ & git clone https://github.com/Lee-Kevin/20.IoTRouter

```
root@mylinkit:~# cd ~ & git clone https://github.com/Lee-Kevin/20.IoTRouter
Cloning into '20.IoTRouter'...
warning: templates not found /usr/share/git-core/templates
remote: Counting objects: 113, done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 113 (delta 5), reused 107 (delta 3), pack-reused 0
Receiving objects: 100% (113/113), 419.29 KiB | 91.00 KiB/s, done.
Resolving deltas: 100% (5/5), done.
Checking connectivity... done.
Checking out files: 100% (78/78), done.
[1]+  Done                    cd ~
root@mylinkit:~# ls
20.IoTRouter    wifiPrinter_js
```

Navigate to 20.IoTRouter and you can find there’re 3 folders, RouterArduinoCode, RouterScript and Drawings. RouterArduinoCode is the code that should be run on Arduino, RouterScript is the Python code should to be run on Linkit Smart 7688. Now we mv the RouterScript to /root/.

cd 20.IoTRouter/ & mv RouterScript/ /root/

```
root@mylinkit:~# cd 20.IoTRouter/
root@mylinkit:~/20.IoTRouter# ls
RouterArduinoCode  RouterScript
root@mylinkit:~/20.IoTRouter# mv RouterScript/ /root/
root@mylinkit:~/20.IoTRouter# ls
RouterArduinoCode
```

Navigate to /root/RouterScript/ and edit the UpdateSensor.py and relay.py using vi editor

cd ~ & cd /root/RouterScript/

vi UpdateSensor.py

vi relay.py

Change the deviceID, deviceKey and dateChannelID to your device. As shown in the following figure.

```
import requests
import time
import logging
logging.basicConfig(level='INFO')
class UpdateSensor:
    url = "https://api.mediatek.com/mcs/v2/devices/DBPY02ol/datapoints.csv"

    payload = ""
    headers = {
        'devicekey': "Gi7LGQpVbqzVYItK",
        'cache-control': "no-cache",
        'content-type': "text/csv",
        'connection': "close",
        # 'postman-token': "258188c6-e125-ff88-1af1-b6e46e03898d"
        }
    def __init__(self):
        pass
    def update(self,RouterTemp,RouterHumi):
        self.payload = "RouterTemperature,," + RouterTemp
        response = requests.request("POST", self.url, data=self.payload, headers=self.headers)
        self.payload = "RouterHumidity,," + RouterHumi
        response = requests.request("POST", self.url, data=self.payload, headers=self.headers)
if __name__ == '__main__':
    sensor = UpdateSensor()
    while True:
        try:
            f = open("/tmp/Sensor/temp")
            temp = f.read()
            logging.info("The temp :%s" % str(temp))
            f.close()
            f = open("/tmp/Sensor/humi")
            humi = f.read()
- UpdateSensor.py 1/39 2%
```

```
import requests
import socket
import threading
import logging
import mraa

# change this to the values from MCS web console
DEVICE_INFO = {
    'device_id' : 'DBPY02ol',
    'device_key' : 'Gi7LGQpVbqzVYItK'
}

# change 'INFO' to 'WARNING' to filter info messages
logging.basicConfig(level='INFO')

heartBeatTask = None

def establishCommandChannel():
    # Query command server's IP & port
    connectionAPI = 'https://api.mediatek.com/mcs/v2/devices/%(device_id)s/connections.csv'
    r = requests.get(connectionAPI % DEVICE_INFO,
                headers = {'deviceKey' : DEVICE_INFO['device_key'],
                           'Content-Type' : 'text/csv'})
    logging.info("Command Channel IP,port=" + r.text)
    (ip, port) = r.text.split(',')

    # Connect to command server
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((ip, int(port)))
    s.settimeout(None)

    # Heartbeat for command server to keep the channel alive
- relay.py 1/67 1%
```

1.5.5 Setup the the startup script

Navigate to /root/RouterScript/

```
cd ~ & cd /root/RouterScript/
```

```
root@mylinkit:~/20.IoTRouter# cd ~ & cd /root/RouterScript/
root@mylinkit:~/RouterScript# ls
NetStatus.py      printer           requirements.txt  runserver.py
UpdateSensor.py   relay.py          router
```

Then setting the startup script.

```
chmod +x router & mv router /etc/init.d/
/etc/init.d/router enable
/etc/init.d/router start &
```

And then reboot the router.

```
reboot
```

When the router is reboot done, you have done all the software work.

1.6 Hardware Work

In this part, we need to make an box as the picture below shows, this work is by my workmate Nosk, and he is an intelligent industry designer, you can download the drawings by click here.
https://github.com/Lee-Kevin/20.IoTRouter/tree/master/Drawings
Then we need to cut the the board using laser cutting. I guess you don't have a laser cutting at home, you can find some in the hacker space near from you easily. If there's no hacker space nearby, you can try the Laser Cutting Service supply by Seeed.

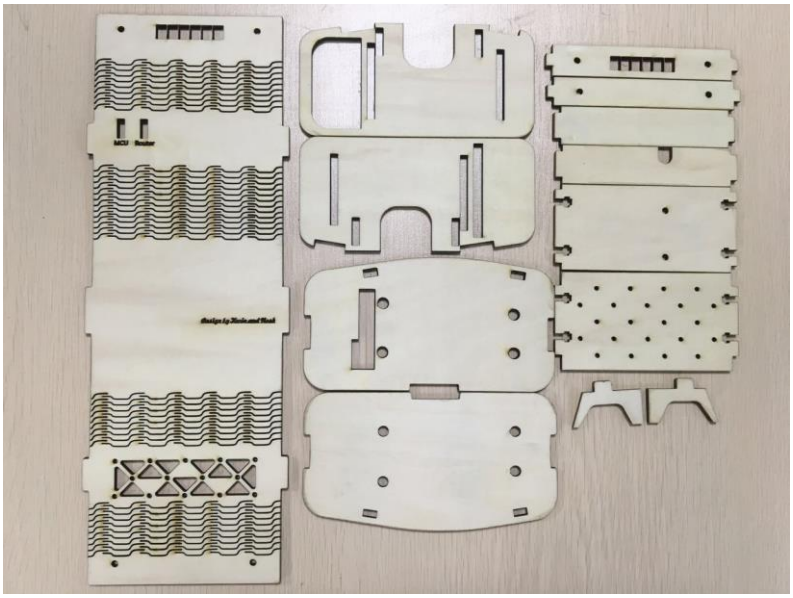
1.6.1 Prepare the things

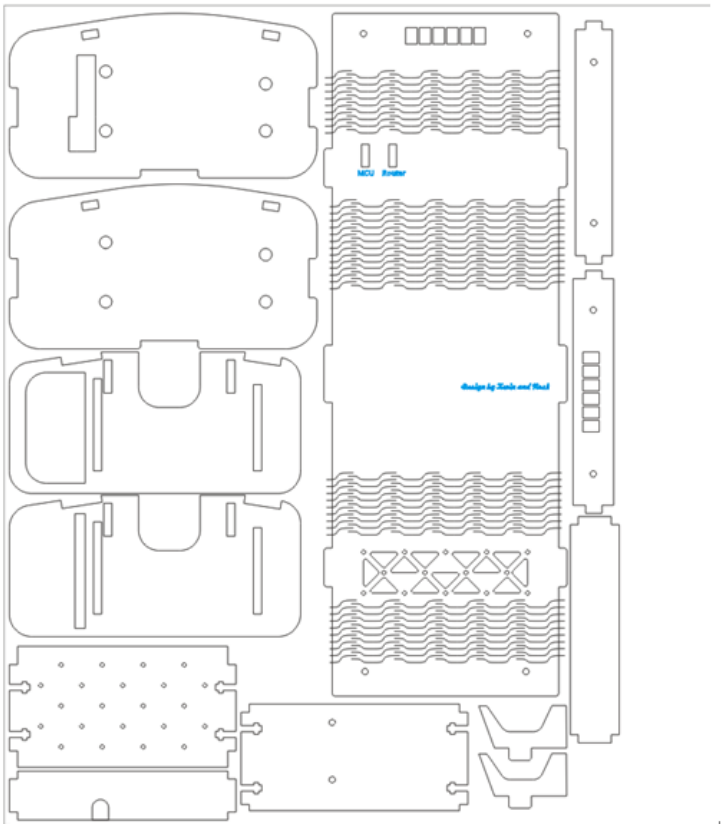
- 4mm wood
- M3*10 Screws – 8 PCS
- M3*8 Scews – 2 PCS
- M3*5+6 Studs – 2 PCS
- M3 Screw nuts – 10 PCS
- R30100 Nylon rivets – 4 PCS
- R2075 Nylon rivets – 6 PCS

Tools:

- 502 glue
- Scotch tape
- Screw driver
- Laser Cutter

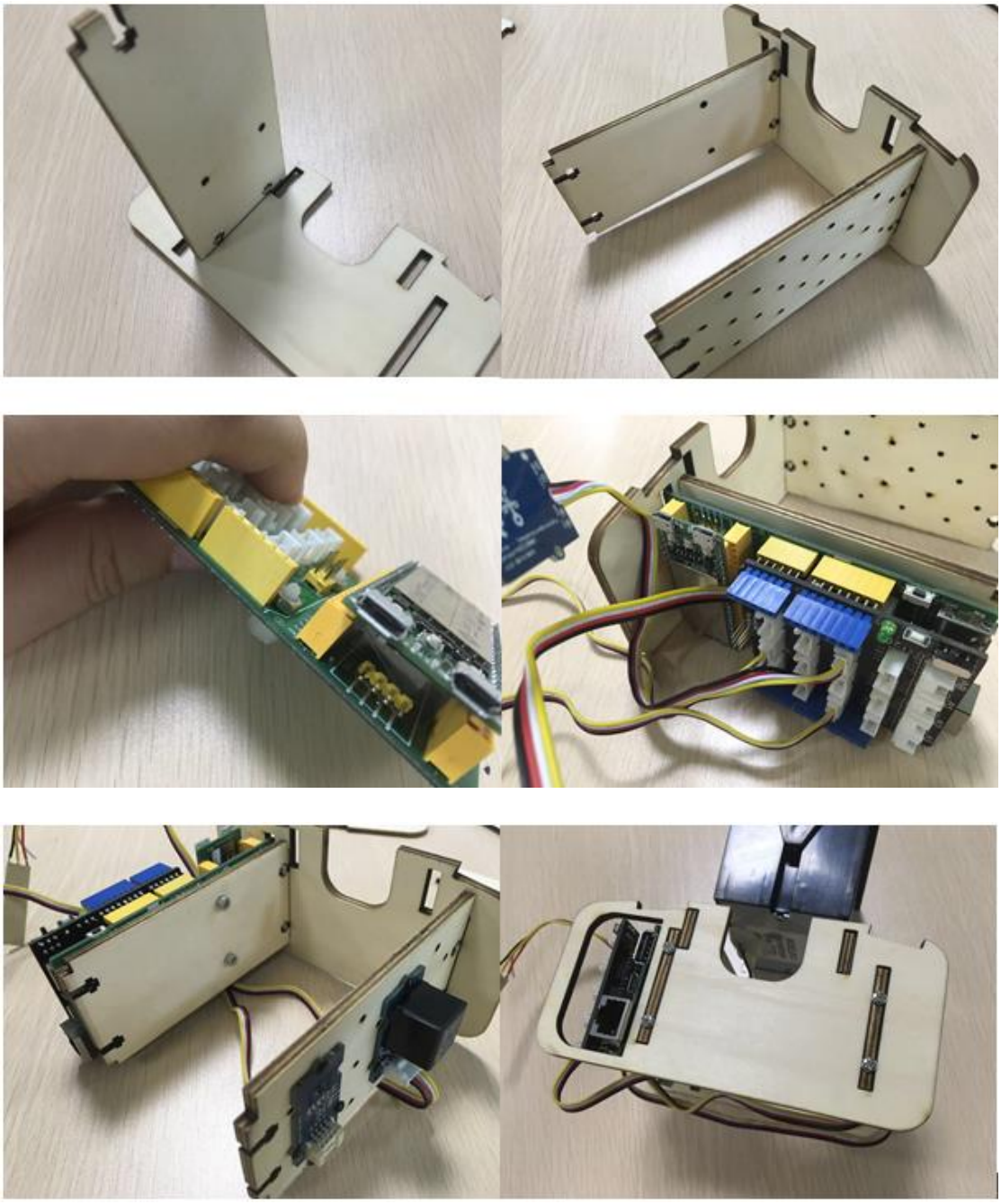
1.6.2 Laser cut the wood

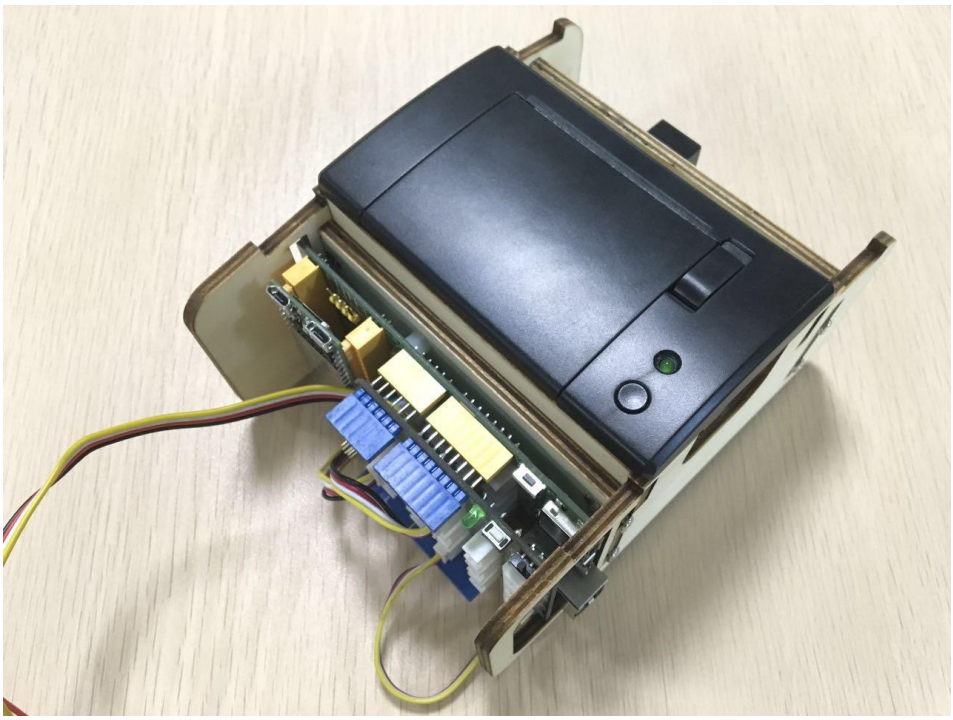




Here's we use 4mm wood~

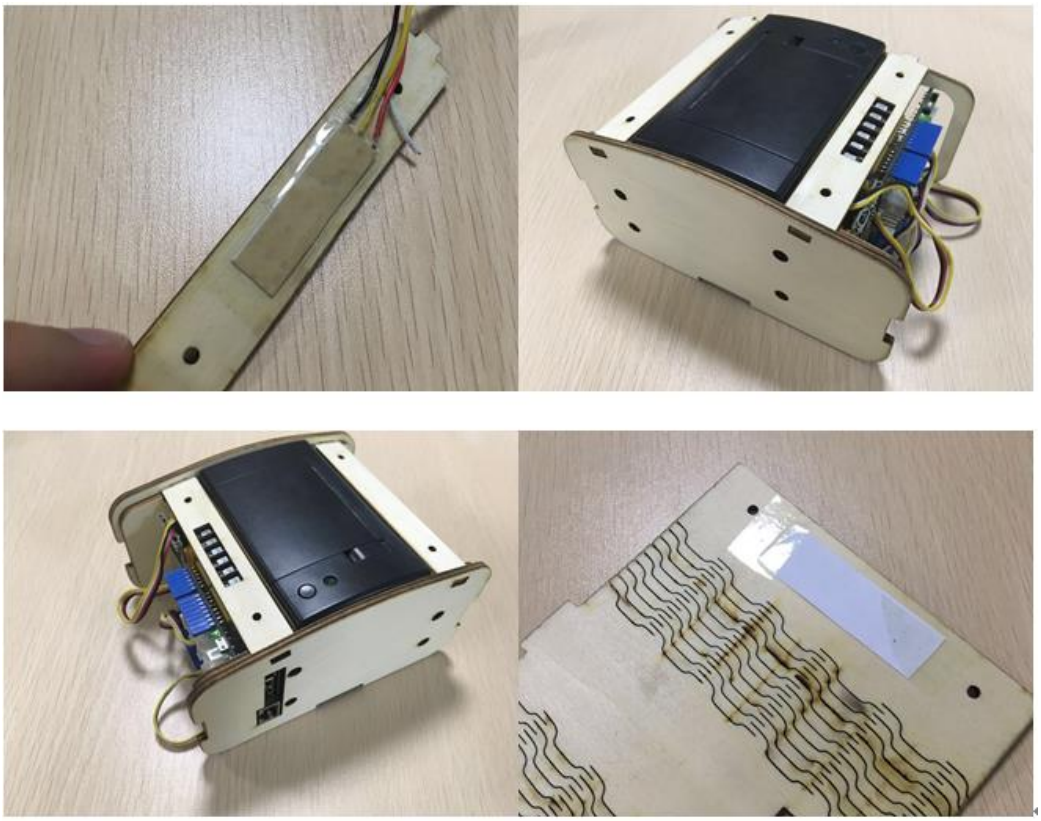
1.6.3 Hardware Assembly





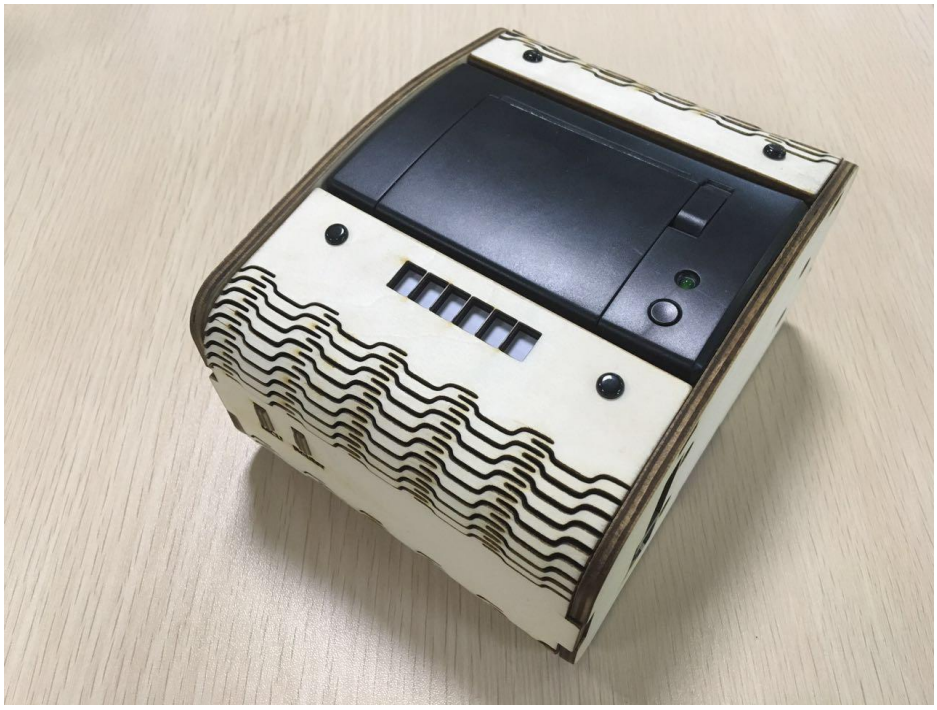
- Fixed the frame by M3*10 screw and nut. (Image1,2)
- Mounting stud on PCB, then fixed it on wood. (Image 3,4)
- Fixed sensor and relay on wood by R2075 nylon rivet. (Image 5)
- Fixed the printer. (Image 6,7)

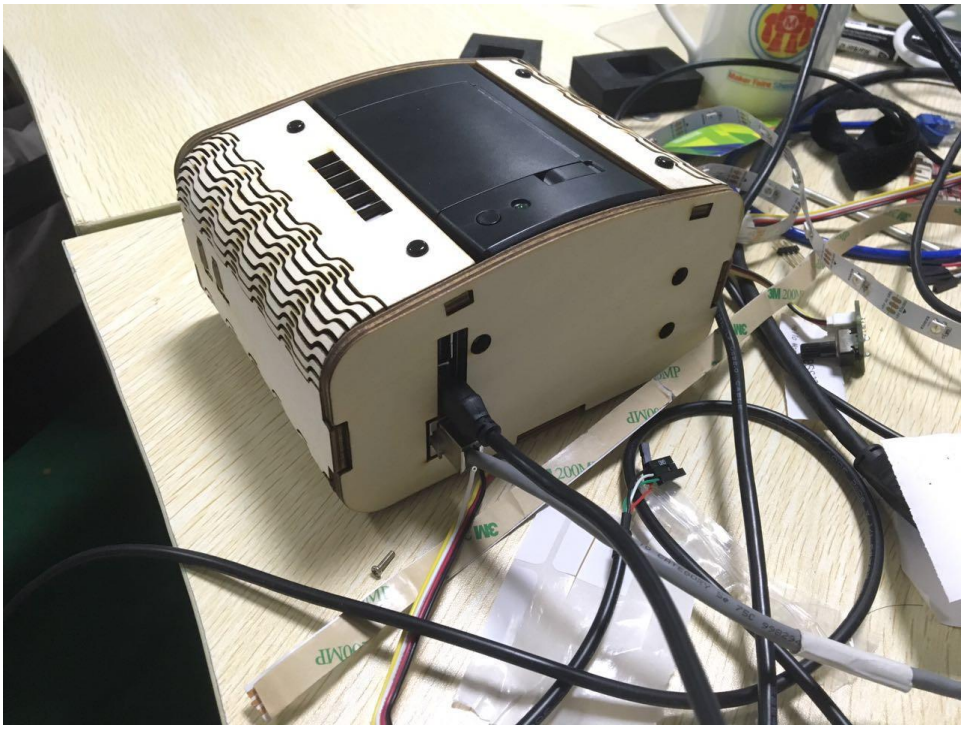
1.6.4 Shell Assembly



- Fixed the RGB LED by scotch tape. (Image 1)
- Fixed both side wood. (Image 2, 3)
- Lampshade made of paper, fixed it by scotch tape. (Image 4, 5)
- Fixed the outside shell by nylon rivet, don’t forgot to install button. (Image 6, 7, 8)

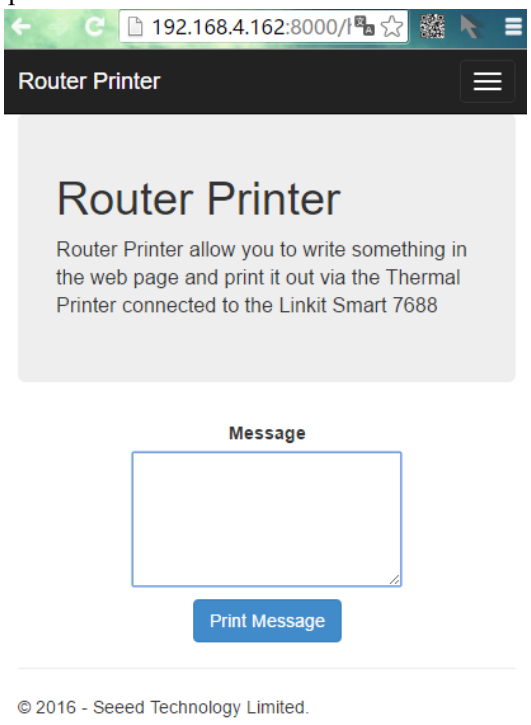
1.6.5 The finished look



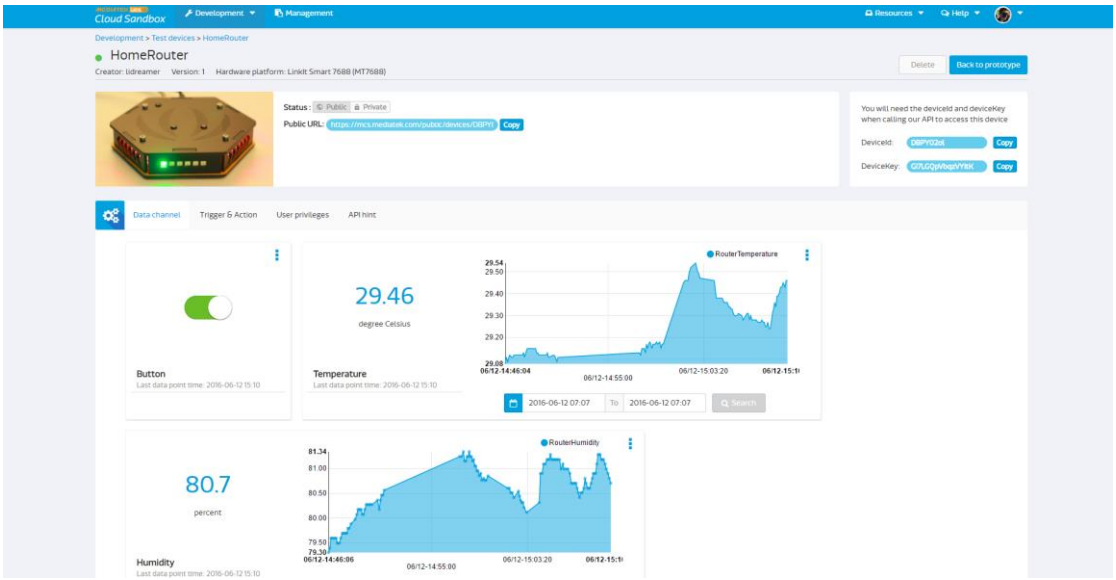


1.7 The Result

Now, you can connect to the Linkit Smart 7688 Wifi AP, and input mylinkit.local:8000 or 192.168.100.1:8000 or your 7688 local IP :8000 in your web browser, and you can see the Router Printer web page just as below shows. Input something in the textbox and click Print Message and you'll find the Printer can print it out.



The go to the MCS website, and check your our test device, you can click the button to control the relay and view the sensor values just as picture below shows.



1.8 Make. Invent. Do.

This project is made as an Open Source Project. It's a starting point. Let your creativity go wild with the mechanical, electrical and software design. Make the demo your own. Decorate it. Improve the work. No matter what, write a recipe about it.

To share and progress together.