matter 环境搭建

一、matter 简介

Matter 协议利用现有 IP 技术（包括 Wi-Fi 和 Thread）为智能家居构建统一的无线连接生态系统。基于 IP 的网络为制造商提供简化的开发，同时提高消费者的设备兼容性。

连接标准联盟 (CSA) 秉承互联体验应该变得更加可靠、安全和有效协同工作这一理念，推出了 Matter（原称 IP 互联家庭项目）。Matter 作为统一的应用层，使用统一的语言，用于使在不同 IP 协议下运行的设备在不同的平台上进行通信，并可协调物联网的开发路径。

Matter 使设备制造商更容易制造与智能家居和语音服务（如 Amazon Alexa、Apple® Siri®、Google Assistant™ 等）兼容的设备。Matter 协议的第一个规范发布将在 Wi-Fi 和 Thread 网络层运行，并将使用蓝牙低能耗进行调试。

二、开发环境搭建

1. 工具

Linux machine（ubuntu-20.04.2.0-desktop-amd64）、Raspberry Pi 4B(Ubuntu20.04.0.3 LTS)、BL602.

其中 Linux machine 是做代码编辑编译使用的，树莓派需要刷 Ubuntu20.04.0.3 LTS。也可以用装 Ubuntu20.04.0.3 LTS 的 pc 代替树莓派。

2. 网络环境

必须用翻墙网，以解决 git clone 和 activate.sh 时可能出现的网络问题，导致 文件不完整。 或者配置公司的 DNS 和网关： # 查看当前的 Gateway 信息 $ route -n # 把当前的网关删除，并配置新网关（使用 VPN）

$ sudo route del default gw 0.0.0.0; sudo route add default gw 10.1.0.217 && route -n # 修改 DNS server 为 8.8.8.8

$ echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf > /dev/null

note：如果拉取的文件不完整，后续会出现各种编译问题。

3. 控制端环境搭建

1>Building and installing

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get install git gcc g++ python pkg-config libssl-dev libdbus-1-dev libglib2.0-dev libavahi-client-dev ninjabuild python3-venv python3-dev python3-pip unzip libgirepository1.0-dev libcairo2-dev bluez avahi-daemon

$ sudo apt-get install pi-bluetooth (如果是非树莓派，则执行 sudo apt-get install bluetooth)

$ reboot

2>Clone the Project CHIP repository

$ git clone https://github.com/project-chip/connectedhomeip.git

branch : master commit id : 031edd0ece116f8b685744f21d3a1c036ffb2247

$ cd connectedhomeip

$ git checkout 031edd0ece116f8b685744f21d3a1c036ffb2247

$ git submodule update --init （该步骤耗时较长）

$ source scripts/activate.sh （该步骤耗时较长）

If this script says the environment is out of date, it can be updated by running:

$ source scripts/bootstrap.sh (usually do not running this step)

Build and Run the Python CHIP Device Controller

$ ./scripts/build_python.sh -m platform

$ source ./out/python_env/bin/activate　　$ chip-device-ctrl

至此，控制端的环境已经搭建好了。

4.602 侧

1>安装依赖

$ sudo apt-get install git gcc g++ python pkg-config libssl-dev libdbus-1-dev libglib2.0-dev libavahi-client-dev ninja-build　python3-venv python3-dev python3-pip unzip libgirepository1.0-dev　libcairo2-dev avahi-daemon

2>编译

$ cd connectedhomeip/examples/lighting-app/bouffalolab/bl602

$./go

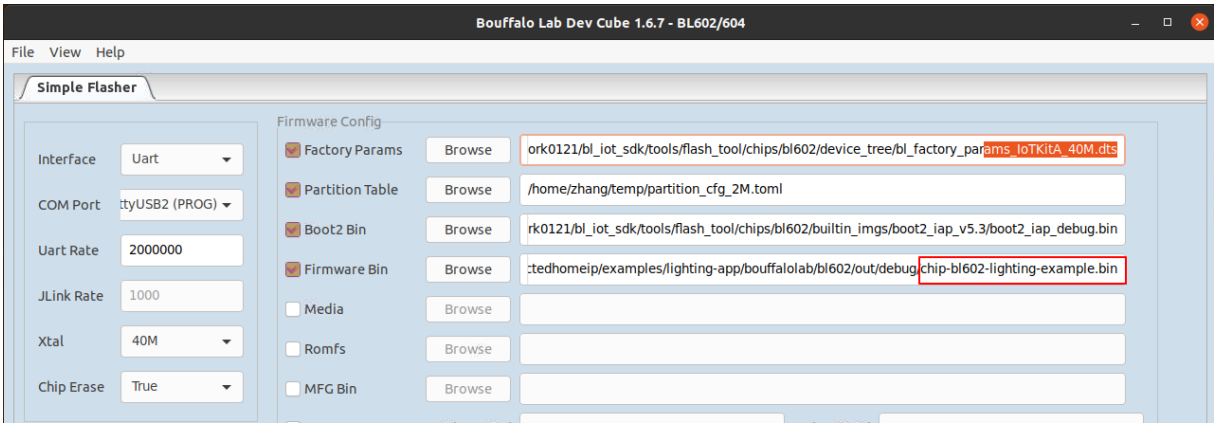
初次编译，会自动更新依赖和环境，耗时较长。（务必保证可以访问外网）

出现下图，则编译完成

```
[702/708] ld ./chip-bl602-lighting-example.out
Memory region         Used Size  Region Size  %age Used
           rom:          0 GB        44 KB      0.00%
         flash:     1106914 B         4 MB     26.39%
        ram_tcm:       97600 B       192 KB     49.64%
       ram_wifi:       55024 B        56 KB     95.95%
[708/708] stamp obj/default.stamp
→ bl602 git:(matter_031edd0e) ✗
```

$ cd matter_bl/connectedhomeip/examples/lock-app/bl602/out/debug/

其中 chip-bl602-lock-example.bin 是我们编出的 bin，使用我们 602 的烧写工具，其他配置和 602 烧写的配置一样，将 Firmware bin 选择该 bin。



重启板子

三、controller 控制 602 操作

1. 建立网路连接

$ cd connectedhomeip

$ chip-device-ctrl

$ chip-device-ctrl >

$chip-device-ctrl > set-pairing-wifi-credential TESTSSID P455W4RD

$chip-device-ctrl > connect -ble 3904 20202021 135246

3840 :discrinator

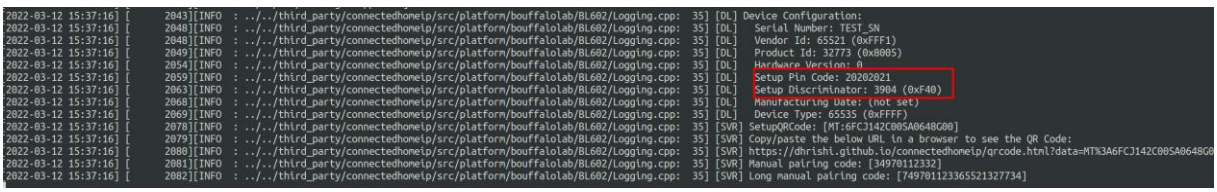
TESTSSID：ssid

P455W4RD:psk

135246:nodeid（可自定义）

20202021：pincode

这些参数可以在设备起来后的 log 中获取：



其中 pincode 和 discrinator 是代码和配置文件中指定的，其他的我们在测试过程中可以自己指定。

2. 控制

chip-device-ctrl > zcl OnOff On 135246 1 0

chip-device-ctrl > zcl OnOff Off 135246 1 0

chip-device-ctrl > zcl LevelControl MoveToLevel 135246 1 1 level=10 transitionTime=0 optionMask=0 optionOverride=0 #改变 level 的值，调灯光亮度

chip-device-ctrl > zcl ColorControl MoveToHue 135246 1 1 hue=100 direction=0 transitionTime=0 optionsMask=0 optionsOverride=0

chip-device-ctrl > zcl ColorControl MoveToSaturation 135246 1 1 saturation=200 transitionTime=0 optionsMask=0 optionsOverride=0 #调颜色