Matter

Devendra Tewari

April 19, 2022

Matter

Matter

Figure 1:

BY

Figure 2:

Objective

- Why Matter?
- Core concepts
- Build and test on Linux
- Build on macOS and test on ESP32
- Device commissioning / setup
- Zigbee Cluster Library Specification
- Contributing to the project

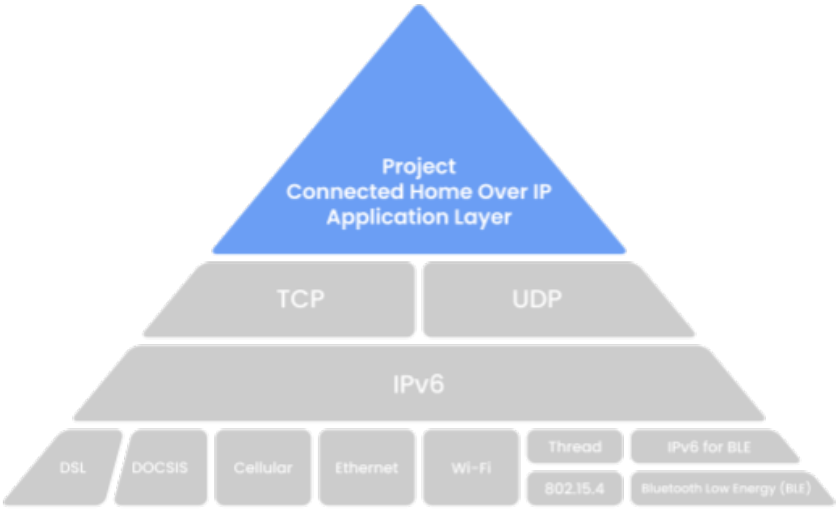
Why Matter?

- Simple, interoperable, reliable, and secure
- Promoted by industry leading device manufacturers
- Apache-2 licensed source code, free of royalties
- Devices available sometime in 2022
- Preliminary support available on Android 12 and iOS 15

Core concepts

Matter	HomeKit	Zigbee
Attribute	Characteristic	Attribute
Binding	Event subscription	Binding
Cluster	Services	Cluster
Commissioning / Rendezvous	Pairing	Association
Controller / Commissioner	Admin	Coordinator
Device or Node	Accessory	Device or Node
Endpoint	Profile	Endpoint
Fabric	Network	Network

Internet Protocols



Architecture

How to use it today?

- Start by reading the docs at https://github.com/project-chip/connectedhomeip
- Try it out with Linux on a Raspberry Pi
- Try it out with an embedded device such as ESP32

Code Repository

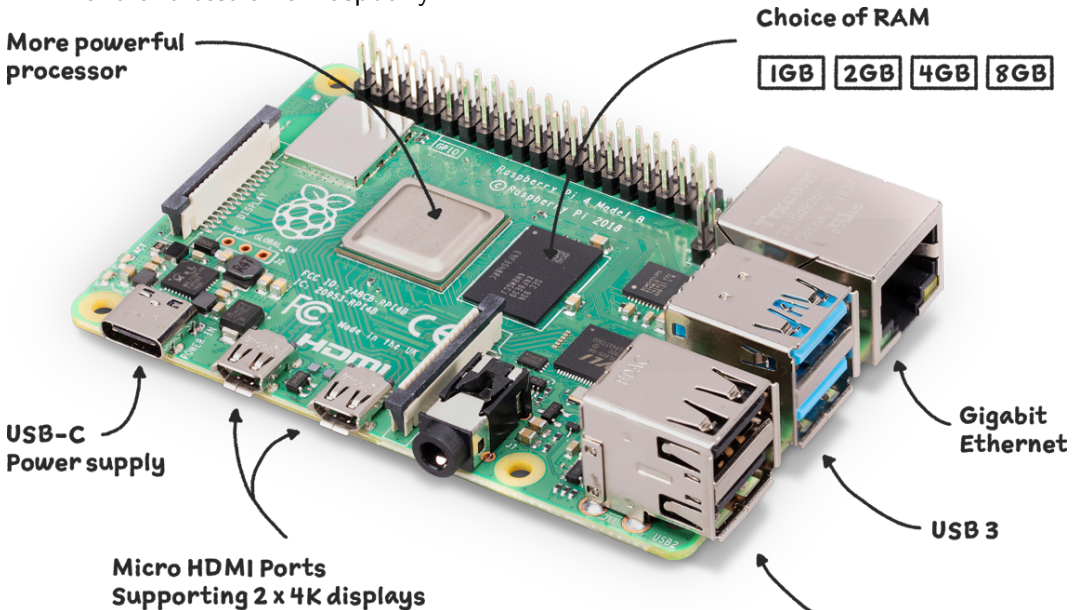
- BUILD.gn
- CONTRIBUTING.md
- build
- build_overrides
- docs
- examples
  - all-clusters-app
  - chip-tool
- scripts
- src
  - platform
    - ESP32
    - Linux
- third_party
  - zap
- zzz_generated

Supported Platforms

- ESP32
- FreeRTOS
- Linux
- Mbed OS
- nRF Connect
- NXP
- Tizen
- Zephyr

Linux Device Development

- Build and test on a Raspberry Pi 4



Install build toolchain on Linux

```
sudo apt-get install git gcc g++ python pkg-config \
  libssl-dev libdbus-1-dev libglib2.0-dev \
  ninja-build python3-venv python3-dev unzip
```

Build and run all-clusters-app on Linux

```
git clone --recurse-submodules \
  https://github.com/project-chip/connectedhomeip
cd connectedhomeip
unalias python
source ./scripts/bootstrap.sh
source ./scripts/activate.sh
cd examples/all-clusters-app/linux
gn gen out/debug
ninja -C out/debug
# Delete network
./out/debug/chip-all-clusters-app --wifi
```

ESP32 Device Development

- Build on macOS and test on M5STACK Core 2

Install ESP-IDF

```
git clone https://github.com/espressif/esp-idf.git
cd esp-idf
git checkout v4.3
git submodule update --init
./install.sh
source ./export.sh
```

Build and run all-clusters-app on ESP32

```
cd connectedhomeip
unalias python
source ./scripts/bootstrap.sh
source ./scripts/activate.sh
cd examples/all-clusters-app/esp32
idf.py build
idf.py -p /dev/cu.usbserial-022D45D6 erase_flash \
  flash monitor
```

chip-tool

- Command line tool to commission and interact with devices

Install dependencies to build chip-tool on macOS

```
brew install openssl pkg-config
cd /usr/local/lib/pkgconfig
ln -s ../../Cellar/openssl@1.1/1.1.1n/lib/pkgconfig/* .
```

Build and run chip-tool on macOS

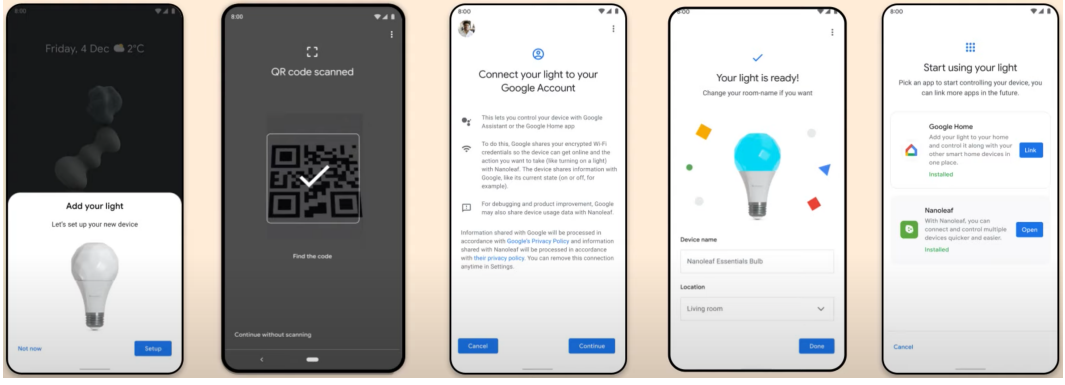
```
cd connectedhomeip
unalias python
source ./scripts/bootstrap.sh
source ./scripts/activate.sh
cd examples/chip-tool
gn gen out/debug
ninja -C out/debug
./out/debug/chip-tool onoff toggle 1 1
```

Commissioning

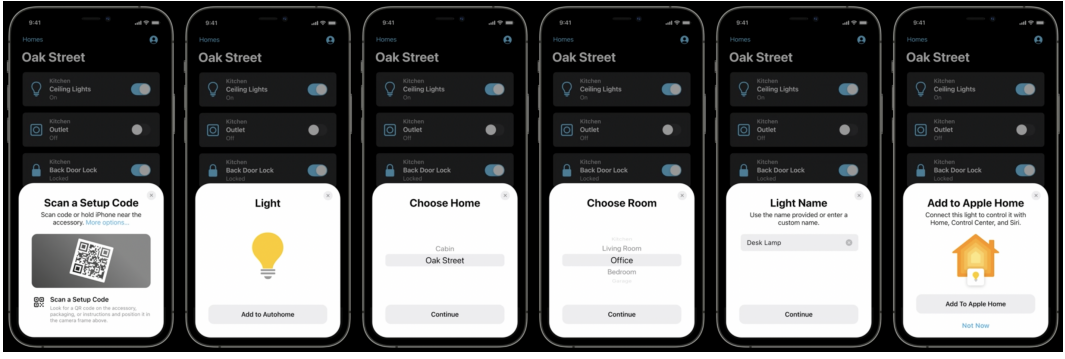
- Configures device into a Matter fabric
- Pair device with multiple controllers / admins
- Commissioning over BLE/Wi-Fi using `chip-tool`

```
./out/debug/chip-tool pairing ble-wifi \
  1 ssid "password" \
  20202021 3840
```

Device Setup Flow on Android 12



Device Commissioning on iOS 15



Pair with another controller / admin

- Open commissioning window on device

```
./out/debug/chip-tool pairing \
  open-commissioning-window \
  1 1 400 2000 3840
```

- Use `onnetwork` pairing to discover devices and pair with first device found

```
./out/debug/chip-tool pairing onnetwork 0 20202021
```

Read attributes using chip-tool

```
./out/debug/chip-tool onoff read on-off 1 1
./out/debug/chip-tool pressuremeasurement read measured-value 1 1
./out/debug/chip-tool relativehumiditymeasurement read measured-value 1 1
./out/debug/chip-tool temperaturemeasurement read measured-value 1 1
CHIP: [DMG]                 }
CHIP: [DMG]
CHIP: [DMG]                     Data = -32768,
CHIP: [DMG]                 DataVersion = 0x0,
CHIP: [DMG]             },
```

Write attributes using chip-tool

```
./out/debug/chip-tool onoff write on-time 5 1 1
./out/debug/chip-tool onoff read on-time 1 1
CHIP: [DMG]                 }
CHIP: [DMG]
CHIP: [DMG]                     Data = 5,
CHIP: [DMG]                 DataVersion = 0x0,
CHIP: [DMG]             },
```

Send commands using chip-tool

```
./out/debug/chip-tool onoff toggle 1 1
./out/debug/chip-tool onoff read on-off 1 1
CHIP: [DMG]                 }
CHIP: [DMG]
CHIP: [DMG]                     Data = true,
CHIP: [DMG]                 DataVersion = 0x0,
CHIP: [DMG]             },
```

View device configuration using ZAP Tool

- Endpoints are defined (along with the clusters and attributes they contain) in a `.zap` file which then generates code and static structures to define the endpoints
- Run Zigbee Cluster Configurator

```
brew install nvm
nvm use stable
cd connectedhomeip
cd third-party/zap/repo
npm i
npm run zap
```

- Open `examples/all-clusters-app/all-clusters-common/all-clusters-app.zap`
- Data definition specified in Zigbee Cluster Library Specification

Contributing to Matter

- Read CONTRIBUTING.md
- Submit bugs and features to https://github.com/project-chip/connectedhomeip/issues
- Change code
- Run automated test suite on host using act e.g. on macOS

```
brew install act
act -j test_suites_linux
```

- Run tests on device using chip-tool

```
./out/debug/chip-tool tests TestCluster --node-id 1
```

- Submit pull request via GitHub for maintainers to review and merge