RT58x Thread Quick Start Guide V1.2

Table of Contents

1. Quick Start Overview

This guide is intended to help developers quickly get started with Thread development on the Rafael RT58x platform. It provides step-by-step instructions for setting up the development environment, flashing binaries, running example applications, and verifying Thread network behavior.

1.1 Target Audience

- Embedded engineers new to Thread or RT58x
- Developers who want a quick hands-on experience with FTD, MTD, BLE integration, and basic network operations

1.2 Supported Use Cases

This guide covers the following use cases with practical examples:

- FTD / MTD Thread nodes (2.4GHz & Sub-GHz)
- BLE-enabled Thread commissioning (toggle BLE mode)
- Ping and UDP communication between Thread nodes
- (Optional) OTA upgrade and sniffer mode (covered in Chapter 5)

1.3 Hardware Requirements

- 2× RT58x development boards
- Micro USB cable ×2
- PC with USB ports
- (Optional) Android phone with Rafael_Android_BLE_demo_v1.3.2_0831.apk

1.4 Software Requirements

- IoT_EVALUATION_TOOL (for flashing firmware via ISP)
- Pre-built example binary files:
  - Thread_2P4G_FTD.bin
  - Thread_SubG_FTD.bin
  - (Optional) Toggle_SubG_BLE_FTD.bin / Toggle_SubG_BLE_MTD.bin
- Serial terminal (e.g., Tera Term, PuTTY)
- (Optional) BLE demo APK for mobile commissioning

2. Board introduction

All RT58x series development boards are supported by this guide. While the following diagrams use the RT581 as an example, the instructions apply to all RT58x devices. Note: Only the RT581 module supports dual-band operation (Sub-GHz and 2.4GHz). Other RT58x variants may support only one band.

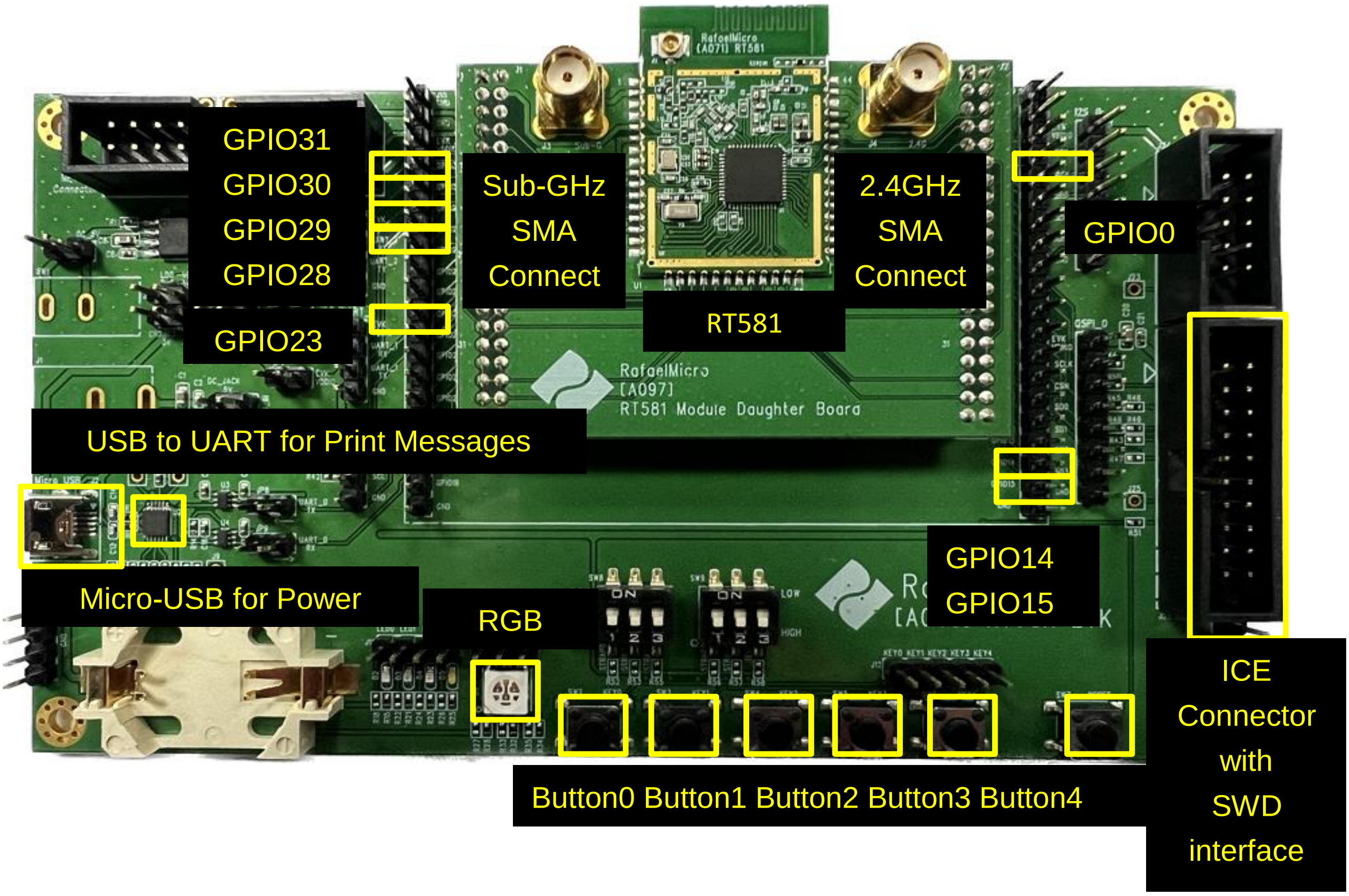


Figure 2-1. RT581 Development Board Overview

- Firmware Flashing Instructions

To flash .bin files to the RT58x board, use the IoT_EVALUATION_TOOL provided by Rafael Micro, Flashing steps are detailed in Chapter 3: IoT_EVALUATION_TOOL Tool Please refer there for step-by-step instructions.

3. Flashing Firmware with IoT_EVALUATION_TOOL

Before running any Thread examples, you must flash the correct binary image onto your RT58x development board. This chapter explains how to use the IoT_EVALUATION_TOOL to flash .bin files via USB in ISP mode.

3.1 Required Tools

Item	Description
RT58x board	Any RT58x-series module (RT581 used in this guide)
Micro-USB cable	For power and UART/ISP connection
PC	With Windows and USB port
IoT_EVALUATION_TOOL	Firmware flashing utility provided by Rafael
.bin file	Pre-built example firmware

The same flashing procedure applies to all RT58x devices.

3.2 Downloading the Tool

Get the latest IoT_EVALUATION_TOOL from Rafael Micro's SDK package or documentation site.

No installation required—just unzip and run the executable.

3.3 Flashing Procedure

Step1. Open this tool and select “ISP” from the options.

Figure 3-1. IoT_EVALUATION_TOOL front page

Step2. Select the USB COM port for downloading the development board.

Figure 3-2. IoT_EVALUATION_TOOL com port

Step3. Choose the corresponding bin file.

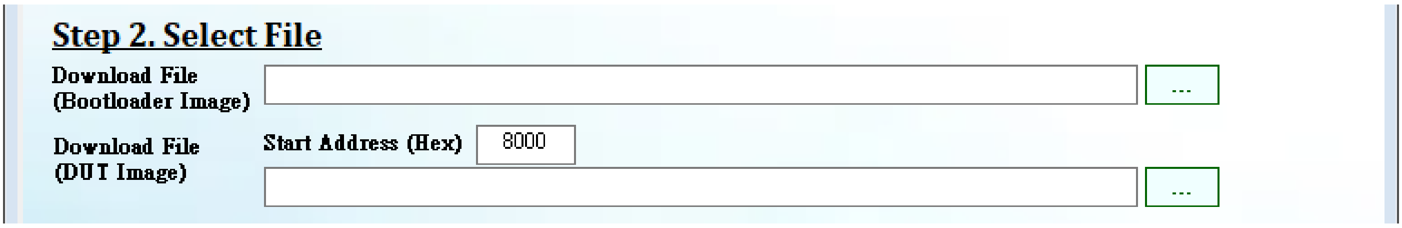


Figure 3-3. IoT_EVALUATION_TOOL select file

Step4. Execute "ISP Connect", and when prompted, press the reset button on the board.

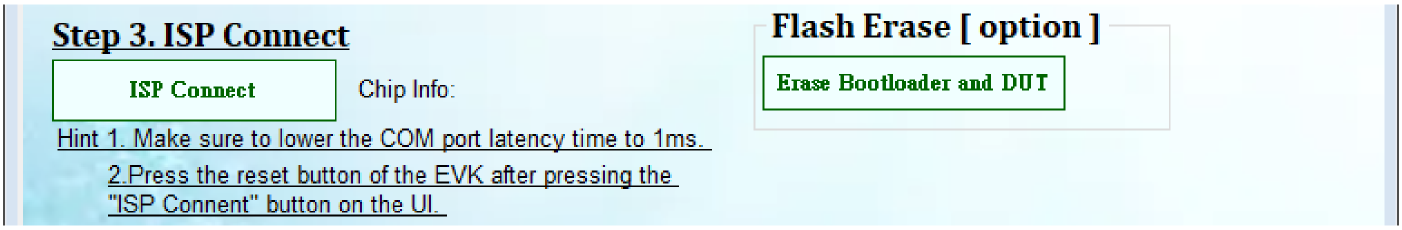


Figure 3-4. IoT_EVALUATION_TOOL ISP connect

Step5. Execute download.

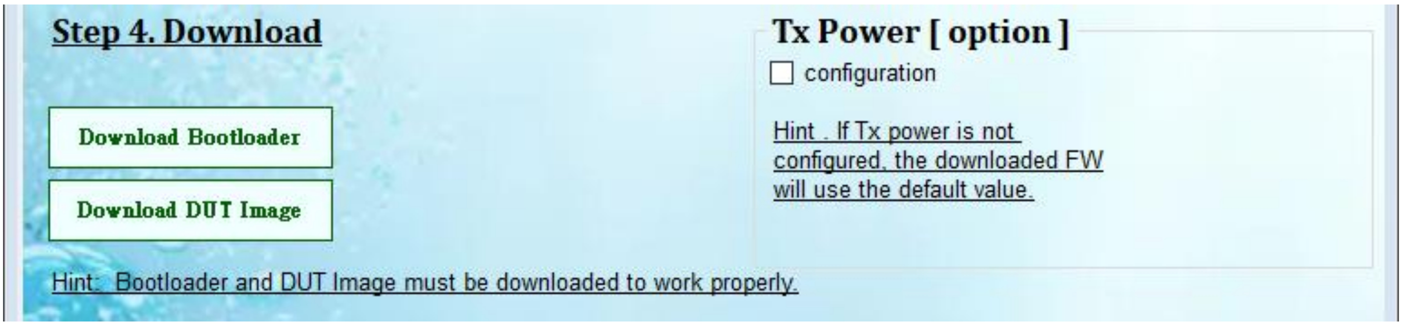


Figure 3-5. IoT_EVALUATION_TOOL download button

4. Example Applications and Communication Test

This chapter introduces several example use cases on the RT58x platform, including FTD and MTD nodes operating on 2.4GHz and Sub-GHz, BLE-based commissioning, and network communication tests using PING and UDP commands.

4.1 2.4GHz Thread Network (FTD + MTD)

4.1.1 Requirements

- 2× RT58x boards
- Firmware:
  - Device 1: Thread_2P4G_FTD.bin
  - Device 2: Thread_2P4G_FTD.bin / Thread_2P4G_MTD.bin

4.1.2 Steps

- Flash firmware to both devices.
- Open serial terminals.
- Power on the Device 1:
  - It automatically becomes Leader.
  - Confirm log with:

```
Change to detached
Change to Leader
fd00:db8:0:0:0:ff:fe00:fc00          Leader ALOC
fd00:db8:0:0:0:ff:fe00:1c00          RLOC
fd00:db8:0:0:5038:3233:3202:6cfc     ML-EID
fe80:0:0:0:5238:3233:3202:6cfc       LLA
```

Note: In the same network key, there will be only one leader.

- Power on the Device 2:
  - It attaches as a Child.
  - Confirm log with:

```
Change to detached
Change to Child
fd00:db8:0:0:0:ff:fe00:1c01          RLOC
fd00:db8:0:0:5038:3233:3202:99fc     ML-EID
fe80:0:0:0:5238:3233:3202:99fc       LLA
```

  - If you use FTD.bin for a while and find that there is no other router, you can upgrade to Router.

- Confirm log with:

```
Change to router
fd00:db8:0:0:0:ff:fe00:4800            RLOC
fd00:db8:0:0:5038:3233:3202:99fc       ML-EID
fe80:0:0:0:5238:3233:3202:99fc         LLA
```

4.2 Sub-GHz Thread Network (FTD + MTD)

If you do not enable CFG_USE_CENTRAK_CONFIG, the steps are the same as step 4.1, please skip it directly.

4.2.1 Requirements

- 2× RT58x boards
- Firmware:
  - Device 1: Thread_SubG_FTD.bin
  - Device 2: Thread_SubG_FTD.bin / Thread_SubG_MTD.bin

4.2.2 GPIO Setup (MTD skip)

- Device 1: Connect GPIO23 to GND → becomes Leader
- Device 2: Leave GPIO23 floating → becomes Router or Child

4.2.3 Steps

- Flash firmware to both devices.
- Open serial terminals.
- Power on the Device 1:
  - It automatically becomes Leader.
  - Confirm log with:

```
Change to detached
Change to Leader
fd00:db8:0:0:0:ff:fe00:fc00            Leader ALOC
fd00:db8:0:0:0:ff:fe00:1c00            RLOC
fd00:db8:0:0:5038:3233:3202:6cfc       ML-EID
fe80:0:0:0:5238:3233:3202:6cfc         LLA
```

Note: In the same network key, there will be only one leader.

- Power on the Device 2:
  - It attaches as a Child.

- Confirm log with:

```
Change to detached
Change to Child
fd00:db8:0:0:0:ff:fe00:1c01            RLOC
fd00:db8:0:0:5038:3233:3202:99fc       ML-EID
fe80:0:0:0:5238:3233:3202:99fc         LLA
```

- If you use FTD.bin for a while and find that there is no other router, you can upgrade to Router.
- Confirm log with:

```
Change to router
fd00:db8:0:0:0:ff:fe00:4800            RLOC
fd00:db8:0:0:5038:3233:3202:99fc       ML-EID
fe80:0:0:0:5238:3233:3202:99fc         LLA
```

- Optional: Use the `nwk` command to view Network Info form leader.

```
nwk
index role parent rloc extaddr rssi
==============================================
[1] child 0400 0401 50383233320299fc -5
==============================================
total num 1
Done
```

4.3 Special circumstances.

If you see 2 devices becoming leaders, please confirm the following:

- Ensure the network key is the same.

```
>networkkey
fe83448a6729feababfe29678a4483fe
Done
```

- If in the Sub-GHz band, verify if the data rate is the same.

```
=============DataRate FSK_300K==========
```

This log is displayed during boot-up.

- Use a scan to confirm if you can receive data from each other.

```
scan
| PAN  | MAC Address      | Ch | dBm | LQI |
+------+------------------+----+-----+-----+
| 8f28 | cafe00000000001d |  3 | -58 | 107 |
| 8f28 | cafe000000000035 |  3 | -43 | 145 |
| 8f28 | cafe00000000001b |  3 | -47 | 135 |
| 8f28 | cafe00000000002a |  3 | -37 | 160 |
| 8f28 | cafe00000000001a |  3 | -61 |  99 |
| 8f28 | cafe00000000002d |  3 | -48 | 132 |
| 8f28 | cafe000000000019 |  3 | -52 | 122 |
| 8f28 | cafe00000000001f |  3 | -45 | 140 |
| 8f28 | cafe00000000000d |  3 | -54 | 117 |
| 8f28 | cafe000000000021 |  3 | -47 | 135 |
| 8f28 | cafe000000000008 |  3 | -53 | 119 |
| 8f28 | cafe000000000014 |  3 | -49 | 130 |
| 8f28 | cafe000000000023 |  3 | -48 | 132 |
| 8f28 | cafe00000000003c |  3 | -51 | 124 |
| 8f28 | cafe00000000001e |  3 | -44 | 142 |
Done
```

Figure 4-1. Scan log

4.4 Sub-GHz Thread Network (FTD + MTD Toggle)

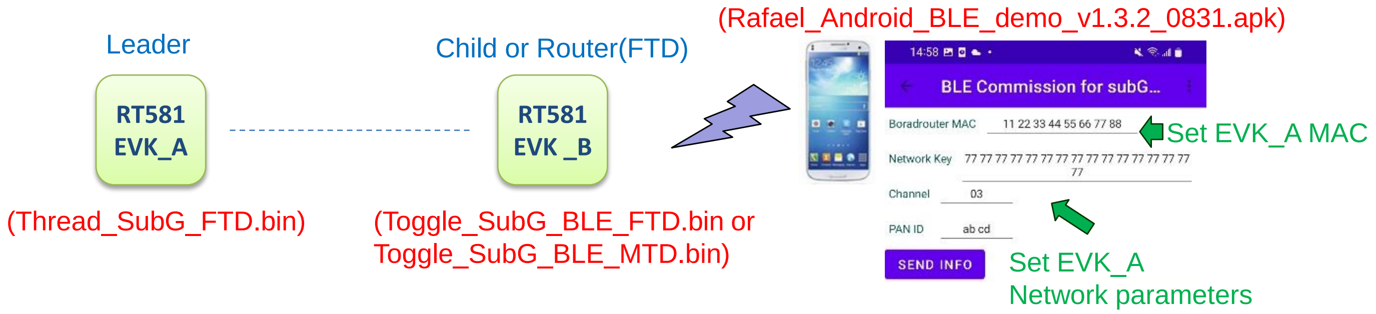


Figure 4-2. Networking Architecture

4.4.1 Requirements

- 2× RT58x boards
- Firmware:
  - Device 1: Toggle_SubG_FTD.bin
  - Device 2: Toggle_SubG_BLE_FTD.bin / Toggle_SubG_BLE_MTD.bin
- Android Phone with Rafael_Android_BLE_demo_v1.3.2_0831.apk

4.4.2 Steps

- Flash firmware to both devices.

- Open serial terminals.
- Power on the Device 1:
  - It automatically becomes Leader.
  - Confirm log with:

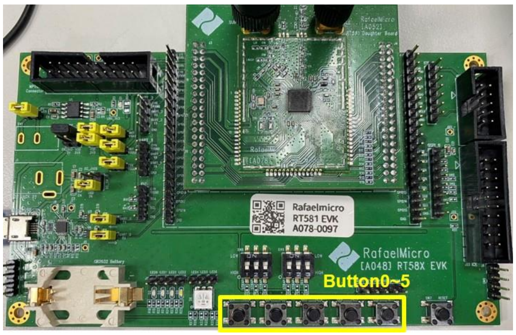
```
Change to detached
Change to Leader
fd00:db8:0:0:0:ff:fe00:fc00          ← Leader ALOC
fd00:db8:0:0:0:ff:fe00:1c00          ← RLOC
fd00:db8:0:0:5038:3233:3202:6cfc     ← ML-EID
fe80:0:0:0:5238:3233:3202:6cfc       ← LLA
```

  Note: In the same network key, there will be only one leader.

  - Check network info:

```
>extaddr
503832333202bafd
>networkkey
fe83448a6729feababfe29678a4483fe
>channel
3
Done
>panid
0xabcd
Done
```

- Power on the Device 2:
  - Press Button0 on Device 1 to start BLE advertising, enable BLE Commissioning.



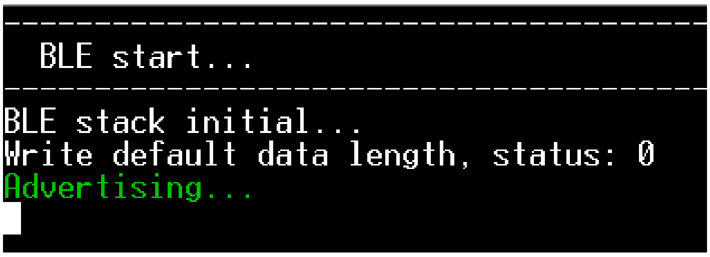


Figure 4-3. Button Operational

- Use Rafael_Android_BLE_demo APP to commission setting.

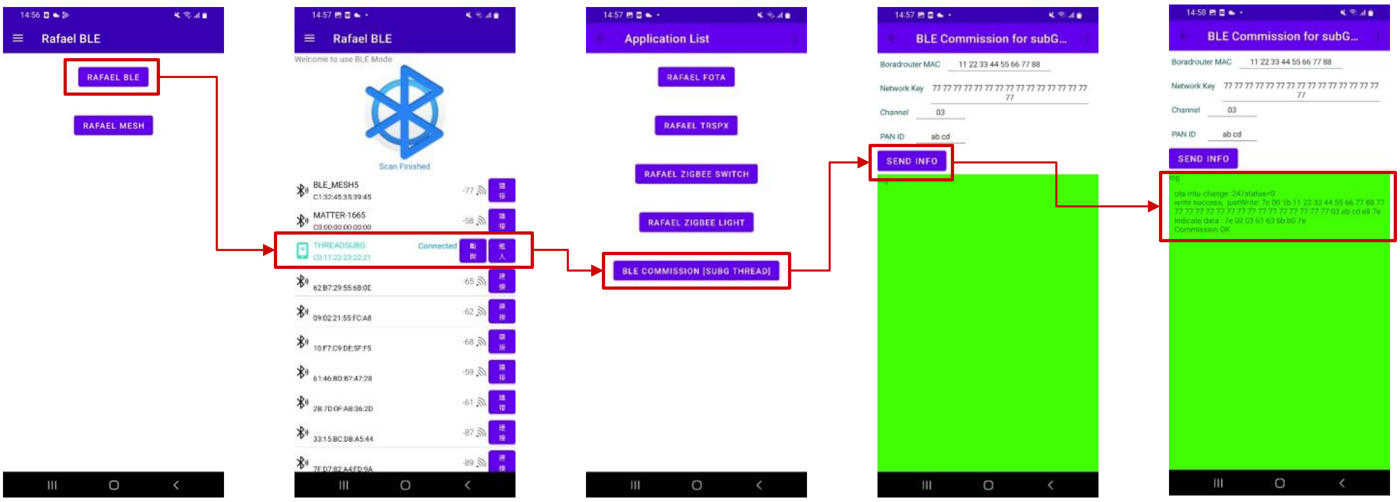
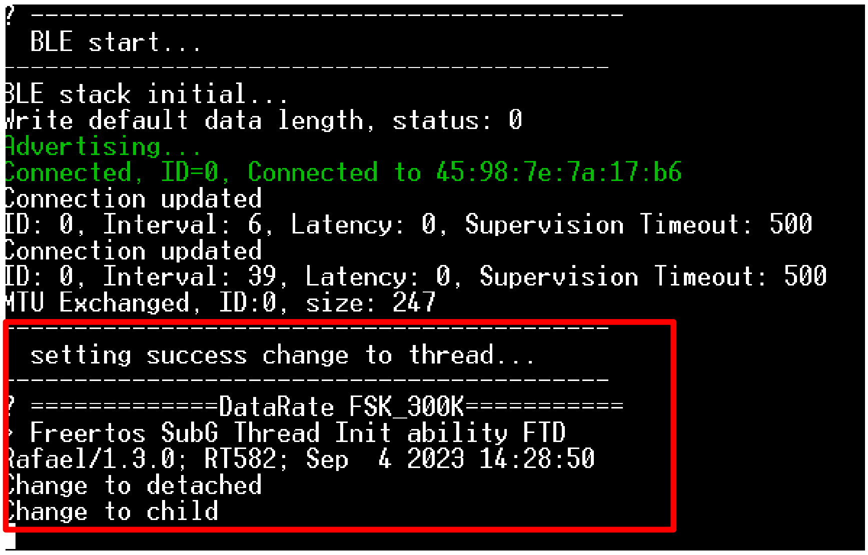


Figure 4-4. Rafael_Android_BLE APP

- Check Toggle_SubG_BLE device change to Thread child.



```
? ----------------------------------------
  BLE start...
----------------------------------------
BLE stack initial...
Write default data length, status: 0
Advertising...
Connected, ID=0, Connected to 45:98:7e:7a:17:b6
Connection updated
ID: 0, Interval: 6, Latency: 0, Supervision Timeout: 500
Connection updated
ID: 0, Interval: 39, Latency: 0, Supervision Timeout: 500
MTU Exchanged, ID:0, size: 247
----------------------------------------
  setting success change to thread...
----------------------------------------
? ============DataRate FSK_300K==========
> Freertos SubG Thread Init ability FTD
Rafael/1.3.0; RT582; Sep  4 2023 14:28:50
Change to detached
Change to child
```

Figure 4-5. Networking Done log

4.5 Communication Test: Ping

4.5.1 Purpose

- Verify IPv6 communication between two devices.

4.5.2 Steps

- Use ipaddr to get ML-EID or RLOC:
- Leader log with:

```
>Ipaddr
fd00:db8:0:0:0:ff:fe00:fc00            Leader ALOC
fd00:db8:0:0:0:ff:fe00:1c00            RLOC
fd00:db8:0:0:5038:3233:3202:6cfc       ML-EID
fe80:0:0:0:5238:3233:3202:6cfc         LLA
Done
```

- Other Device log with:

```
>Ipaddr
fd00:db8:0:0:0:ff:fe00:4800            RLOC
fd00:db8:0:0:5038:3233:3202:99fc       ML-EID
fe80:0:0:0:5238:3233:3202:99fc         LLA
Done
```

- Example:

```
>ping fd00:db8:0:0:5038:3233:3202:99fc
> 16 bytes from fd00:db8:0:0:5038:3233:3202:99fc: icmp_seq=5 hlim=64
time=0ms
1 packets transmitted, 1 packets received. Packet loss = 0.0%. Round-trip
min/avg/max = 0/0.0/0 ms.
Done
```

4.6 Communication Test: UDP

4.6.1 Setup

- On both devices:

```
> udp open
Done
> udp bind :: 1234
Done
```

- Send UDP message:

```
> udp send <target_ip> 1234 hello
Done
```

- Example:

```
> udp send fd00:db8:0:0:5038:3233:3202:99fc 1234 hello_99fc
Done
```

- Receiver will show:

```
>10 bytes from fd00:db8:0:0:5038:3233:3202:6cfc 1234 hello_99fc
```

Appendix A. Thread Role introduction

This section will provide a brief introduction to the role of Thread.

Device types

Full Thread Device

A Full Thread Device (FTD) always has its radio on, subscribes to the all-routers multicast address, and maintains IPv6 address mappings. There are three types of FTDs:

- Router
- Router Eligible End Device (REED) — can be promoted to a Router
- Full End Device (FED) — cannot be promoted to a Router

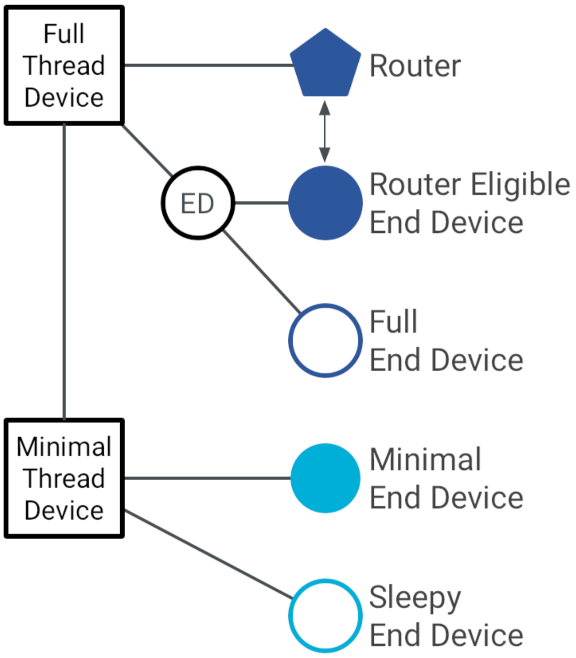
An FTD can operate as a Router (Parent) or an End Device (Child).

Minimal Thread Device

A Minimal Thread Device does not subscribe to the all-routers multicast address and forwards all messages to its Parent. There are two types of MTDs:

- Minimal End Device (MED) — transceiver always on, does not need to poll for messages from its parent
- Sleepy End Device (SED) — normally disabled, wakes on occasion to poll for messages from its parent

An MTD can only operate as an End Device (Child).



Roles

Border Router

A Border Router is a device that can forward information between a Thread network and a non-Thread network (for example, Wi-Fi). It also configures a Thread network for external connectivity.

Any device may serve as a Border Router.

Note: This SDK does not provide.

Leader

The Thread Leader is a Router that is responsible for managing the set of Routers in a Thread network. It is dynamically self-elected for fault tolerance, and aggregates and distributes network-wide configuration information.

Note: There is always a single Leader in each Thread network partition.

Router

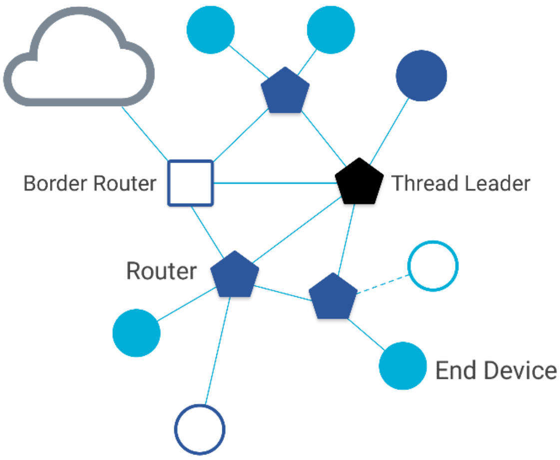
A Router is a node that:

- forwards packets for network devices
- provides secure commissioning services for devices trying to join the network
- keeps its transceiver enabled at all times

End Device

An End Device (ED) is a node that:

- communicates primarily with a single Router
- does not forward packets for other network devices
- can disable its transceiver to reduce power



Appendix B. Thread IPv6 Addressing

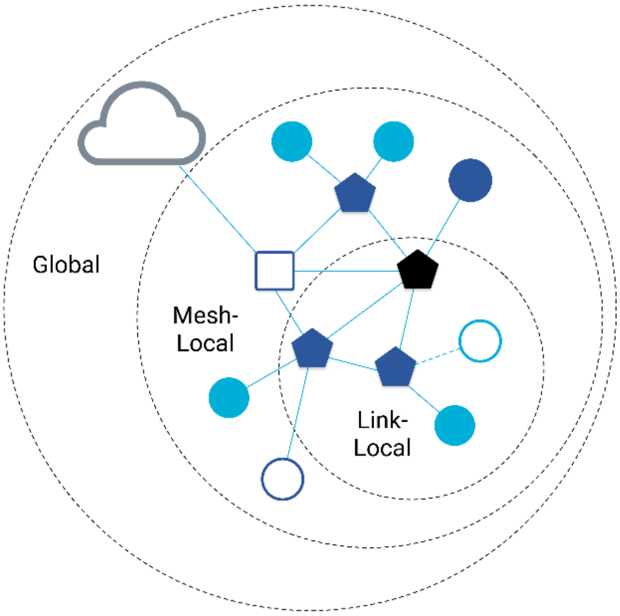
This section will provide a brief introduction to the IPv6 address of Thread.

IPv6 Scopes

There are three scopes in a Thread network for unicast addressing:

- Link-Local — all interfaces reachable by a single radio transmission
- Mesh-Local — all interfaces reachable within the same Thread network
- Global — all interfaces reachable from outside a Thread network

The first two scopes correspond to prefixes designated by a Thread network. Link-Local have prefixes of fe80::/16, while Mesh-Local have prefixes of fd00::/8.



Unicast address

Common unicast types are detailed below.

Link-Local Address (LLA)	
An EID that identifies a Thread interface reachable by a single radio transmission.	
Example	fe80::54db:881c:3845:57f4
IID	Based on 802.15.4 Extended Address
Scope	Link-Local
Details	◦ Used to discover neighbors, configure links, and exchange routing information ◦ Not a routable address ◦ Always has a prefix of fe80::/16

Mesh-Local EID (ML-EID)	
An EID that identifies a Thread interface, independent of network topology. Used to reach a Thread interface within the same Thread partition. Also called a Unique Local Address (ULA).	
Example	fde5:8dba:82e1:1:416:993c:8399:35ab
IID	Random, chosen after commissioning is complete
Scope	Mesh-Local
Details	◦ Does not change as the topology changes ◦ Should be used by applications ◦ Always has a prefix fd00::/8

Routing Locator (RLOC)	
Identifies a Thread interface, based on its location in the network topology.	
Example	fde5:8dba:82e1:1::ff:fe00:1001
IID	0000:00ff:fe00:RLOC16
Scope	Mesh-Local
Details	◦ Generated once a device attaches to a network ◦ For delivering IPv6 datagrams within a Thread network ◦ Changes as the topology changes ◦ Generally not used by applications

Anycast Locator (ALOC)	
Identifies a Thread interface via RLOC lookup, when the RLOC of a destination is not known.	
Example	fde5:8dba:82e1:1::ff:fe00:fc01
IID	0000:00ff:fe00:fcXX
Scope	Mesh-Local
Details	◦ fcXX = ALOC destination, which looks up the appropriate RLOC ◦ Generally not used by applications

Global Unicast Address (GUA)	
An EID that identifies a Thread interface on a global scope, beyond a Thread network. (This SDK does not provide)	
Example	2000::54db:881c:3845:57f4
IID	◦ SLAAC — Randomly assigned by the device itself ◦ DHCP — Assigned by a DHCPv6 server ◦ Manual — Assigned by the application layer
Scope	Global
Details	◦ SLAAC — Randomly assigned by the device itself ◦ DHCP — Assigned by a DHCPv6 server ◦ Manual — Assigned by the application layer

Multicast address

Multicast is used to communicate information to multiple devices at once. In a Thread network, specific addresses are reserved for multicast use with different groups of devices, depending on the scope.

IPv6 Address	Scope	Delivered to
ff02::1	Link-Local	All FTDs and MEDs
ff02::2	Link-Local	All FTDs
ff03::1	Mesh-Local	All FTDs and MEDs
ff03::2	Mesh-Local	All FTDs

Note: That Sleepy End Devices (SEDs) are not included as a recipient in the multicast table above.

Anycast address

Anycast is used to route traffic to a Thread interface when the RLOC of a destination is not known. An Anycast Locator (ALOC) identifies the location of multiple interfaces within a Thread partition. The last 16 bits of an ALOC, called the ALOC16, is in the format of 0xfcXX, which represents the type of ALOC.

For example, an ALOC16 between 0xfc01 and 0xfc0f is reserved for DHCPv6 Agents. If the specific DHCPv6 Agent RLOC is unknown (perhaps because the network topology has changed), a message can be sent to a DHCPv6 Agent ALOC to obtain the RLOC.

Thread defines the following ALOC16 values:

ALOC16	Type
0xfc00	Leader
0xfc01 – 0xfc0f	DHCPv6 Agent
0xfc10 – 0xfc2f	Service
0xfc30 – 0xfc37	Commissioner
0xfc40 – 0xfc4e	Neighbor Discovery Agent
0xfc38 – 0xfc3f 0xfc4f – 0xfcff	Reserved

Revision History

Revision	Description	Owner	Date
V1.0	Initial version	Jiemin	2023/10/16
V1.1	Add and modify Chapter 6.	Jiemin	2024/07/29
V1.2	Chapter organization	Jiemin	2025/06/02