Multi-PAN Release Notes

EmberZNET

Exported on 06/08/2020

Table of Contents

Version	Date	Notes
1.0	5-June-2020	Initial revision

This document describes functionality, usage, and limitations regarding the multi-PAN feature.

1 Background

The multi-PAN feature builds upon the existing multi-network feature. The multi-network feature is described in AN724: Designing for Multiple Networks on a Single Zigbee Chip and can be found in the documentation folder in the SDK. It can also be found on the documents section of www.silabs.com.[1] It is highly recommended to read that document first before reading about the multi-PAN feature.

The multi-PAN feature is used to create a host/NCP application that can support up to two coordinator networks. The two networks use the same radio to send and receive packets on their own distinctive PAN IDs.

1 http://www.silabs.com.

2 Distinction From Multi-Network

Whereas the multi-network feature limits the number of always-on networks to one, the multi-PAN feature allows for two always-on networks, both of which must be coordinators. The multi-PAN feature also enforces a single radio channel and transmission power value for both networks, given that both networks must operate simultaneously.

3 Getting Started

3.1 Plugins

This release demonstrates the multi-PAN feature with a single plugin for the Network Co-Processor framework.

- Multi-PAN Library - this plugin is required in NCP applications to support the multi-PAN feature.

3.2 Zigbee Stack: Zigbee Device Types

When building a host application, the Zigbee Stack tab is the page where a user configures the local device type. When using the classical multi-network feature, the user could create, at most, one receiver-on-when-idle network (coordinator or router). Host applications are now able to configure two coordinator networks in this configuration window. It is imperative that the matching NCP image be compiled with the multi-PAN library plugin enabled, else the feature will not work during runtime.

3.3 Applications

This release offers several sample applications to demonstrate the multi-PAN feature.

- MpZ3TcCustomTcHost - A host application meant to demonstrate the multi-PAN feature. This is a Zigbee 3.0 coordinator on the first network and a coordinator with no security on the second network. It is meant to connect to an NCP running the Multi-PAN NCP UART HW application.
- Multi-PAN NCP UART HW - An NCP application meant to demonstrate the multi-PAN feature. This is an application to be loaded onto a Silicon Labs EFR32MG12, EFR32MG13, EFR32MG14, or EFR32MG series 2 part. This application has the multi-PAN library enabled, and it is meant to work with the MpZ3TcCustomTcHost host application.

For the second network index on the multi-PAN coordinator, which has no security, joining devices will need to also have no security to join the network successfully. This can be configured on the Zigbee Stack tab, Zigbee PRO Network Configuration, Security Type selection. Note that choosing No Security also sets the default stack profile to 0. A sample application is not provided for this scenario.

4 APIs

There are no new APIs created for the multi-PAN feature. The multi-PAN feature relies on the already existing multi-network APIs to get and set network context before issuing API calls or operating in a callback.

The paradigm for multi-PAN (and multi-network) is that a user sets the current network to the desired index, issues any desired APIs, and then restores the network context to the one used by the application framework. Conversely, in callbacks issued by the framework, the user can set the current network index to the one returned by the callback code, perform any desired operations, and then restore the network index to the one used by the application framework. Both of these concepts are discussed in AN724.

5 Examples

5.1 API Example

The snippet below demonstrates the the multi-PAN feature using API calls. The code forms a centralized Z3.0 network on the first index and forms a security-less network on the second index. Devices may be joined on each network.

```
// Set the context to the first network
emberSetNetwork(0);
emberAfPluginNetworkCreatorStart(true); // centralized network

// Wait for EMBER_NETWORK_UP, then open network for joining
emberAfPluginNetworkCreatorSecurityOpenNetwork();

// Join devices as desired
// Trust center link keys may be updated for joining devices

// Disable local and remote 802.15.4 permitJoin flags, so that devices
// joining on the second network don't accidentally join this network
emberAfPluginNetworkCreatorSecurityCloseNetwork();

// Move to the second network
emberSetNetwork(1);
emberFormNetwork(params);   // params.channels must match the single channel from the first network!

// Wait for EMBER_NETWORK_UP, then open network for joining
emberPermitJoining(60);     // pJoin for 60 seconds

// Join devices as desired
```

5.2 CLI Example

The same example is achieved below using CLI.

```
# Set the context to the first network
network set 0
plugin network-creator start 1

# Wait for EMBER_NETWORK_UP, then open network for joining
plugin network-creator-security open-network

# Join devices as desired
# Trust center link keys may be updated for joining devices

# Disable local and remote 802.15.4 permitJoin flags, so that devices
# joining on the second network don't accidentally join this network
plugin network-creator-security close-network

# Move to the second network
network set 1
network form 15 3 0xBEEF
# NOTE: channel argument must match the channel from the first network!

# Wait for EMBER_NETWORK_UP, then open network for joining
network pjoin 60

# Join devices as desired
```

6 Known Issues and Limitations

6.1 Supported Compilers

Various runtime issues have been reported when compiling with the GCC ARM compiler. It is recommended to use the IAR ARM compiler for this alpha release. In Simplicity Studio 5, select File > New > Silicon Labs Project Wizard, and under Toolchain select Simplicity IDE / IAR ARM before moving to the next dialog.

6.2 Maximum Supported Networks

Two networks are currently supported by the multi-PAN feature. Both networks must be coordinators.

6.3 Supported Platforms

The multi-PAN feature is limited to EFR32MG12, EFR32MG13, EFR32MG14, and EFR32MG2 platforms.

6.4 Channel Selection and Channel Changing

Since the two coordinator networks operate at the same time, the two networks must use the same radio channel and transmission power.

Because the two networks must operate on the same radio channel, avoid any network- or user-induced channel changes. The stack does not currently reject channel changes. The stack does not also validate the two networks' transmission power values.

6.5 Memory Considerations

ember-configuration.c is a file that allocates tables based on either user-selected or default sizes. The child table, for example, is allocated based on EMBER_CHILD_TABLE_SIZE. It is important to understand that this value is the size **per network**, and not total in the system. This means that both networks 1 and 2 have an EMBER_CHILD_TABLE_SIZE-entry child table, for a total of 2*EMBER_CHILD_TABLE_SIZE entries. There are several tables in ember-configuration.c that allocate tables per network in this manner. A user should be aware of this detail when sizing various tables.