Application Note:
SimpliciTI Frequency Agility

Author: Larry Friedman

Texas Instruments, Inc.
San Diego, California USA

Version	Description	Date
0.90	Prerelease draft	12/19/2007
0.91	Prerelease draft update: add sequence diagrams	12/31/2007
1.00	General release	02/01/2008
1.10	Changed document title, updated title page, modified formatting	03/24/2009

Table of Contents

List of Figures

List of Tables

1. Introduction

This document presents a high level description of the implementation of Frequency Agility (FA) feature as realized in the SimpliciTI protocol.

2. References

[1] SimpliciTI Specification

[2] SimpliciTI Developers Notes

[3] SimpliciTI Channel Table Information

3. Feature Description

The feature is intended to make it possible for a network of SimpliciTI-compliant devices to be able to migrate among different channels. The purpose of such migration is assumed to be to avoid a noisy channel in which the communications among the SimpliciTI devices is disrupted to the point where it compromises the network functionality.

The feature is not intended to accommodate a rapidly changing train of channel changes such as one might expect in a network supporting frequency hopping mode of operation. The intended use of the feature is not to spread the energy out over a spectrum as a policy, but rather to permit channel changes as an exceptional behavior to avoid contention on the channel.

3.1. Channel table

The intent is that the population of channels be kept small. The list of possible channels is kept in a channel table. The default number in the SimpliciTI distribution is 4 channels. The specific values chosen depend on the characteristics of the radio. The default values should be selected to minimize known values of contention. For example, for the 2.4 GHz radios we try to avoid choices that overlay typical WiFi channels. See [3] for details on the default channel and frequency configurations.

Both the number of channels and the default table of channels are easily modified in header files. In the current design these are set at build time for each device. To work correctly each device in the network must have been built with the same channel table.

From the application perspective channels are referenced as logical channels. The logical channel numbers run consecutively from 0 through (sizeof(Channel Table) – 1). In effect the logical channel number is the index into the channel table that contains the actual radio value to be set.

If the feature is not enabled the first entry in the table will be the channel that is used by default.

The number of entries in the channel table is controlled by the macro **`__mrfi_NUM_LOGICAL_CHANS__`** defined in the radio-specific section of the file **`.\Components\mrfi\mrfi_defs.h`**. The channel table is kept in the radio-specific file as shown below:

Radio	File containing Channel Table
802.15.4 (CC2430)	`.\Components\mrfi\radios\family4\mrfi_radio.c`
CC251x/CC2500/CC111x/CC1100	`.\Components\mrfi\radios\common\mrfi_f1f2.c`

Table 1: File location of Channel Table definition

These may be changed at user discretion.

3.2. Network topology

Support for FA requires that the SimpliciTI network be configured with an Access Point. The Access Point manages the FA feature by communicating with other devices using the Network Frequency Application Port (see [1]).

3.3. Channel Migration

Channel migration is initiated by the Access Point. The migration can be initiated by any one of a number of direct actions. It can be initiated by user intervention, for example, pressing a 'Channel Change' button.

It can also be initiated by algorithmic inference that the current channel is noisy. The Access Point can monitor the current channel for noise and decide on its own that the current channel should be changed.

Though not currently implemented, the Access Point could initiate a channel change based on a request from an End Device. This could occur, for example, if a user pressed a 'Change Channel' button on a Remote Control implemented as a SimpliciTI End Device.

4. Device behavior

Brief descriptions of device behaviors are presented n the following sections.

4.1. Access Point

4.1.1. Startup

At startup the Access Point sets the channel to the default channel. By convention this is logical channel 0.

4.1.2. Channel migration

Channel migration always originates with the Access Point object and is therefore always an active process.

When the Access Point invokes channel migration it sends a broadcast frame to the Frequency Application port. The application payload of this message contains the logical channel number to which to change. All non-sleeping devices should hear this message and change channels accordingly. A prudent implementation might send this broadcast more than once to try and ensure compliance by all devices (especially Range Extenders) that are always on.

The recovery from having missed this message will differ depending on device type. Various scenarios are discussed in the following sections.

There are two stimuli that can initiate a channel migration.

4.1.2.1.Direct User intervention

In this case the user of the device initiates the channel change. This could be done, for example, by pressing a 'Channel Change' button on the device such as is frequently done on cordless phones.

4.1.2.2. Algorithmic intervention

It is possible for the Access Point to monitor the current channel for noise. A policy can be implemented based on RSSI measurements that a channel change is required. This allows the Access Point to decide on its own that channel migration should be activated.

4.2. End Device

End Device behavior depends to some extent on the End Device capabilities. These are addressed in the following sections.

4.2.1. Startup

All End Devices regardless of configuration start up the same way. If FA is active (implying that an Access Point is part of the network) the End Device begins by scanning the channel population. As part of the Join discipline, for each channel the End Device sends a Frequency Application Ping frame and waits for a reply. The frame is sent to the broadcast address. Only an Access Point will reply to this frame.

A scan results in a list (typically of length 1) of the channels on which the End Device receives a reply to the Frequency Application Ping frame. The End Device then tries the usual Join discipline on each of these channels. If more than one channel is in the list, which implies that there is more than one active network within range, the Join Token should be the object governing the network to which the End Device has access.

4.2.2. Channel migration

The mechanism of channel migration differs depending on the characteristics of the End Device configuration.

4.2.2.1. Always on

In the case of an always-on device (regardless of the remainder of the device configuration) the network Frequency Application can receive the Access Point broadcast message announcing the channel change. In this case the device switches to the announced logical channel.

It does not matter if there are no user applications expecting messages. An always-on device is assumed to be mains powered. It is then permissible to leave the radio in receive mode when idle. In this case the Frequency Application will receive the broadcast frame from the Access Point.

4.2.2.2. Polling

Polling devices that sleep between polls will miss the Access Point broadcast Frequency Application message. Broadcasts are not stored for later forwarding. Polling devices must discover that the channel has changed.

The means used for this discovery is based on the polling discipline. When a polling device polls it always expects a reply. If no frames are waiting it will receive a frame spoofed by the Access Point with no application payload. If a frame is waiting that frame will be forwarded.

If no reply is received it is possible that the channel has changed. After a configurable number of poll reply failures the device will scan for the current channel and then repeat the polling procedure.

4.2.2.3.Sleeping Transmit-only

Two kinds of sleeping transmit-only devices are considered. The more general case is the one in which applications on a device only transmit but the device is equipped with a transceiver so it can receive messages. The second case is one in which the device has a transmit-only radio without the capability of receiving messages.

4.2.2.3.1.Transmit-only with transceiver

In the case in which no application receives frames but the device has a transceiver there are three possibilities. First, the application could act as if it had no receive capability and behave as in Section 4.2.2.3.2 below. In this case the channel change is never detected. Second, the application could do a channel scan each time before sending a frame. Third, the application could be written to expect a reply that would serve as an acknowledgment.

The only way to detect a channel migration having missed the Access Point announcement is to expect to receive a frame and then not receive it. This is the basis of the channel scan mechanism but there is no reason to waste the resources needed to scan when channel migration is a low frequency event. Better to build in an application level acknowledgment where one isn't necessary in support of the application messaging discipline. Then scanning takes place only when failing to get the acknowledge message.

4.2.2.3.2.Transmit-only with transmitter only

The only option in this case is for all transmissions from such a device to be done on each and every channel for each frame sent. Preferably this cycle should be repeated at least once for insurance.

4.2.3. Missed message mitigation

Always-on devices are expected to receive the broadcast announcement from the Access Point that the channel is to be changed. If a device misses this message then unless some application periodically expects to receive a frame the changed channel will never be detected.

Always-on devices should be built with protection against this scenario. A backup to missing the Access Point announcement is to employ one of the strategies discussed in Section 4.2.2.3.1.

4.3. Range Extender

4.3.1. Startup

The Range Extender startup sequence is the same as that of an always-powered End Device. See Section 4.2.1.

4.3.2. Channel migration

Because the Range Extender is always on it can be expected to receive the Access Point announcement of a channel change. If this message is missed other strategies are necessary depending on whether there is a peer application running on the Range Extender device.

The main purpose of the Range Extender is to replay frames. The Range Extender need not and typically will not have any peer applications running. If it did it could use the strategies used by the always-on End Device (see Section 4.2.3).

Since Range Extenders typically will not have peer applications running they should periodically do a channel scan to detect a channel migration. The period of this scan is left unspecified.

4.4. Linking

Unlike joining, the current implementation of linking makes no provision for FA. That is, the link procedure will not discover if the device is on the incorrect channel at link time. The listener will listen on an incorrect channel indefinitely. If the linking device is on the incorrect channel the linking will continue to fail because it will not get a response.

This implies that the linking procedure should occur in close temporal proximity to another service or exchange which can be guaranteed to discover the correct channel. For example, linking immediately after joining will likely not fail due to channel migration.[1] All the SimpliciTI sample applications work this way.

If linking procedure is required at some later time and the device is possibly on the incorrect channel a scan can be done using the **SMPL_Ioctl()** interface presenting an opportunity for link success based on temporal proximity as above.

[1] There is the small possibility of a channel migration between the join and the link. But even in this case if the receiver is on the AP broadcast channel migration message will likely be processed.

5. Scenarios

The following drawings represent the general device behaviors described above in a pictorial format. Not every possible permutation is shown.

5.1. Startup behavior

Startup is the same for all non-Access Point devices.

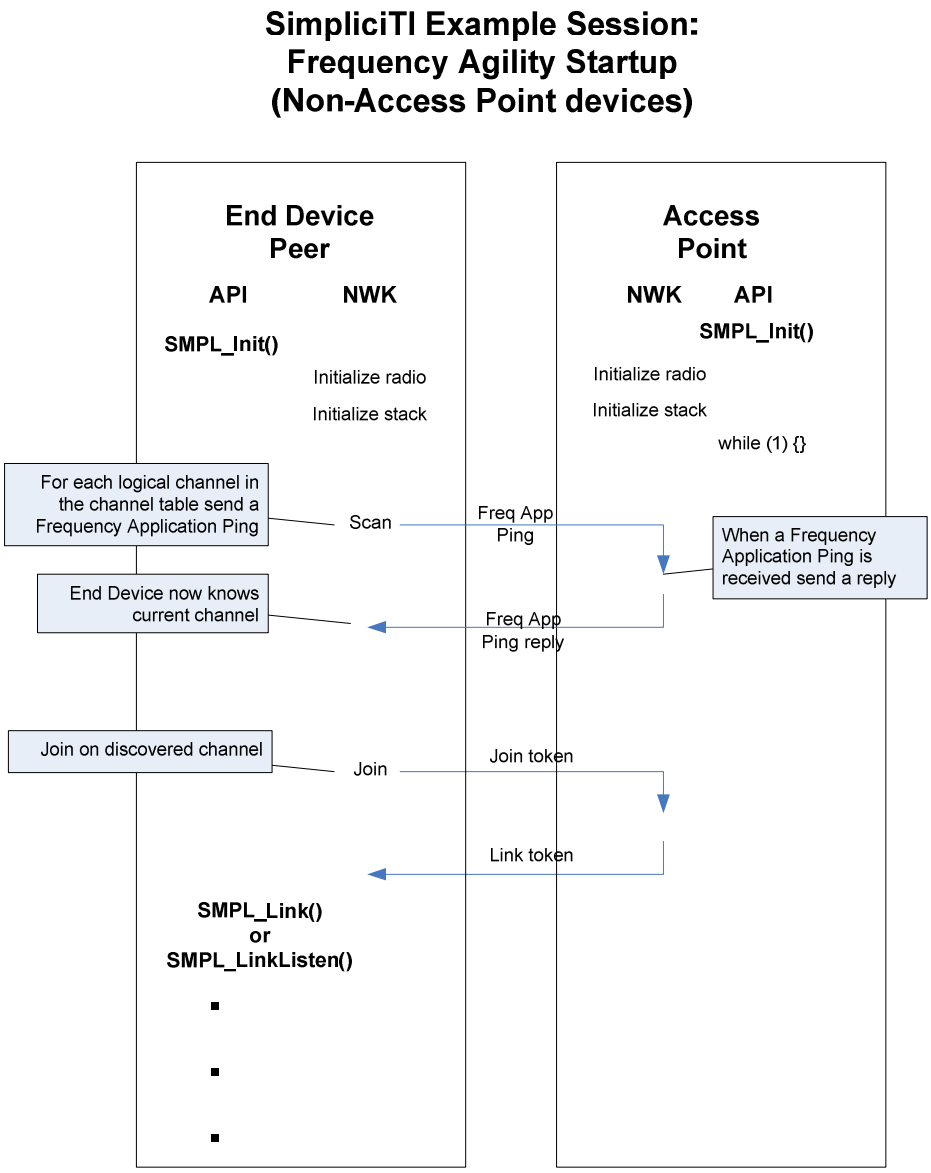

Figure 1: Frequency Agility Startup sequence

5.2. Initiation of channel migration

The following diagram indicates how the Access Point initiates a channel change.

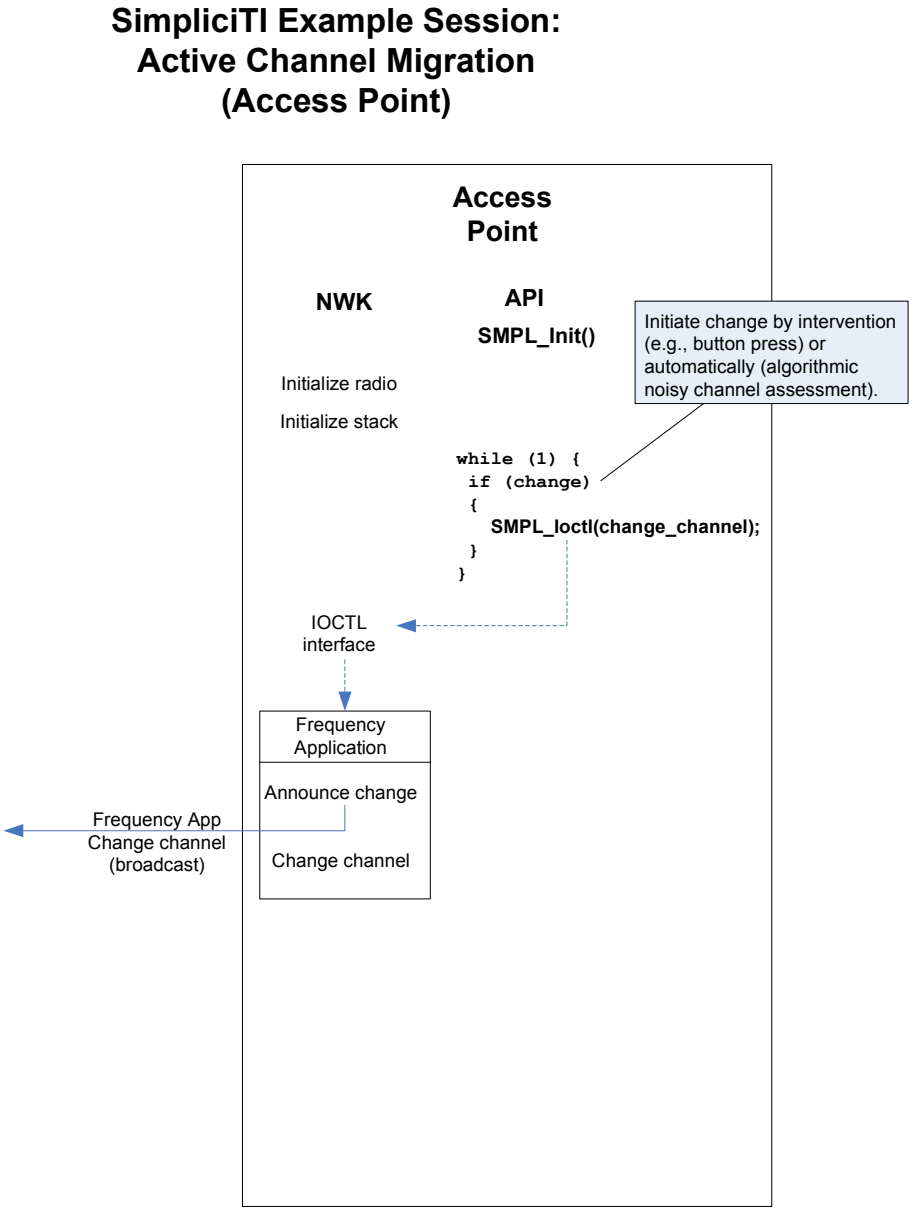

Figure 2: Access Point channel migration initiation

An example of an auto-migration algorithm is implemented in the 'Access Pont as data hub' example in the SimpliciTI distribution. See Release Notes for versions 1.0.4 and later for details.

5.3. Channel migration: polling device

This sequence represents behavior when a polling device faces a channel migration. The polling device does not hear the broadcast channel change message from the AP. The polling device receives no reply to the initial poll and scans to recover.

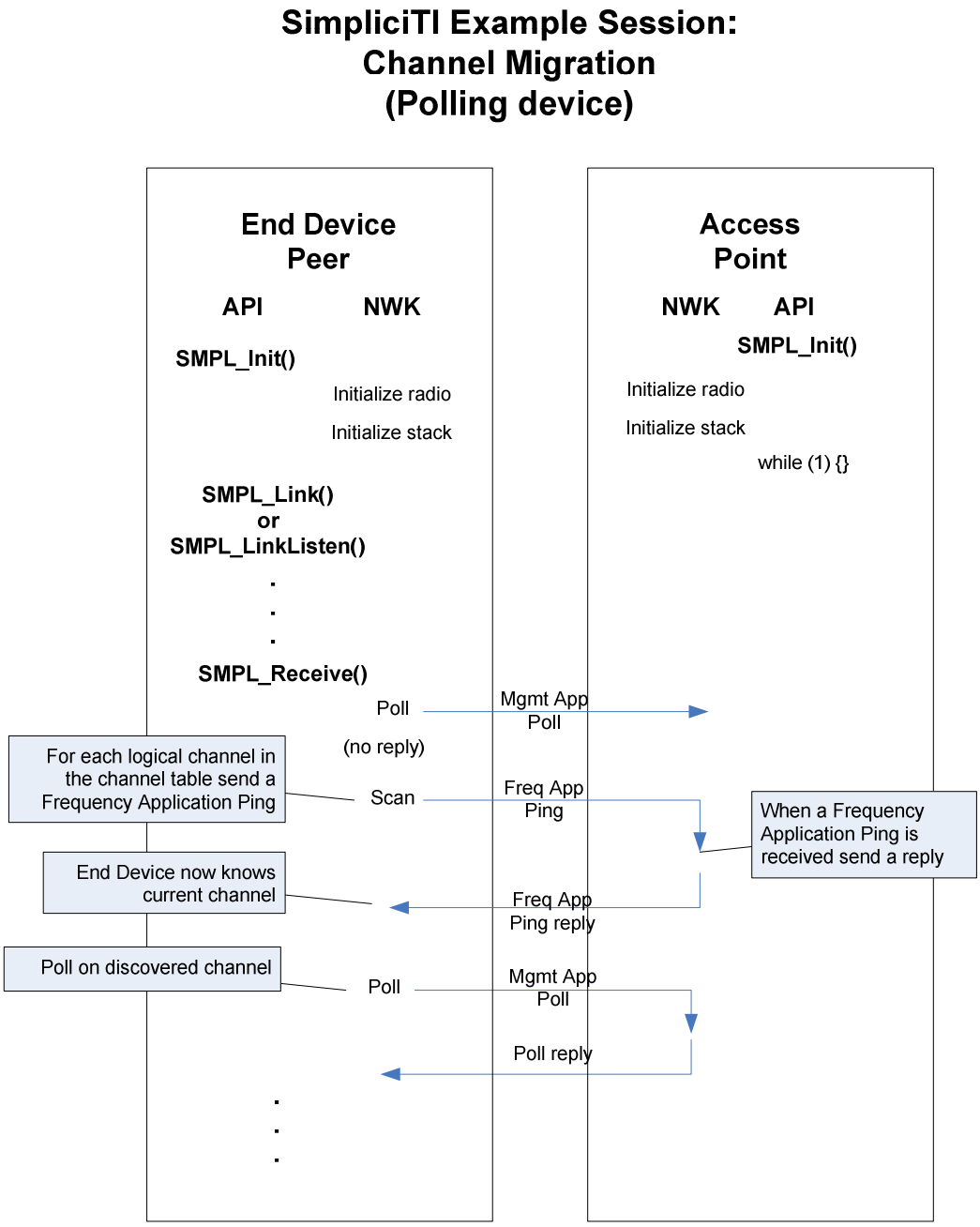

Figure 3: Channel migration with polling device.

5.4. Channel migration: TX-only sleeping device

This sequence shows the device described in Section 4.2.2.3.1. This is an End Device object that functionally does not need to listen but can do so by nature of its hardware support.

For example, a temperature sensor that awakens and sends a temperature measurement periodically does not need to listen. However, hardware permitting, this kind of device must have the capability to listen in order to detect a channel change. The 'listen' aspect must be implemented at the SimpliciTI application level as does the channel migration logic itself. It is realized as a **SMPL_Receive()** call either polled or stimulated by a callback.

In the following sequence both peers as well as the Access Point are shown. One peer (End Device) is always awake and can hear the channel migration announcement from the Access Point. The other peer (End Device) is the sleeping device the only needs to send periodic messages to the peer. The peers implement an acknowledgment discipline at the application level which provides a mechanism for the sleeping device to detect the channel migration.

The following sequence diagram begins after all devices have started up and the peers have joined and linked successfully. These sequences have appeared elsewhere (see [2]).

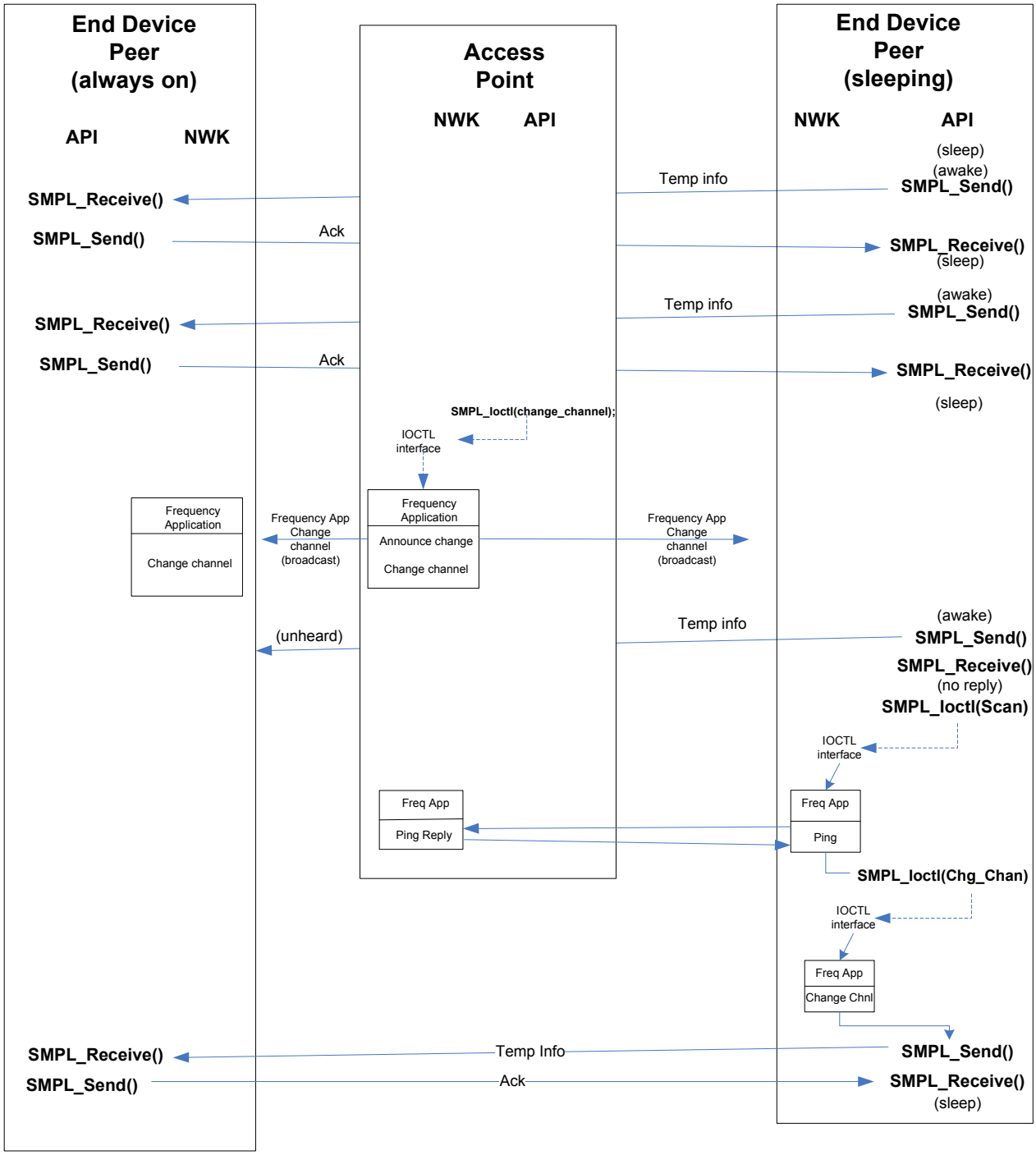

Figure 4: Channel migration with application-implemented acknowledge on sleeping device