AN1403: Wi-SUN PHY Mode Switch on EFR32FG25 with RAILtest

This document is an introduction to the Wi-SUN PHY Mode Switch feature using the RAILtest example application. It describes:

- How to program EFR32FG25 devices to enable the Wi-SUN Mode Switch feature
- Known issues

Proprietary is supported on all EFR32FG devices, although Mode Switch and OFDM modulation is available only on EFR32FG25 devices. In this document only the RAILtest example application is used.

KEY POINTS

- EFR32FG25 devices support Wi-SUN mode switch available with Wi-SUN FAN 1.1
- The release has some minor limitations detailed in the Known Issues section

1 Introduction

1.1 Mode Switch Definitions

The Wi-SUN FAN 1.1 specification describes the Modulation and Data Rate (MDR) Negotiation, with impacts at both the stack layer and the PHY mode switch.

This document only addresses the PHY mode switch as defined in on Wi-SUN PHY Technical Profile Specification 2V00 section 6.1.4, which describes Wi-SUN Mode Switch based on a specific Physical layer Protocol Data Unit (PPDU) frame using Mode Switch PHY header (PHR). This mechanism will be referred to as **Wi-SUN PHY mode switch** (for the function) or **mode switch PPDU** (for the specific PPDU name) to avoid ambiguity.

Note that Wi-SUN PHY Mode Switch is not compliant with the mode switch described in IEEE std 802.15.4-2020.

1.2 EFR32FG25 Support of Mode Switch and Deliveries

EFR32FG25 supports Wi-SUN PHY mode switch. The RAILtest application lets you test the Mode Switch PPDU with a specific radio configuration using RAILtest commands described in the following sections.

Wi-SUN PHY Mode Switch allows PHY switching from a Frequency Shift Keying(FSK) PHY to another FSK PHY, or from an FSK PHY to an Orthogonal frequency-division multiplexing (OFDM) PHY. The configuration is achieved through the Radio Configurator in Simplicity Studio. An example of *radio_settings.radioconf* file for North America (NA), Japan (JP), and Europe (EU) is provided with the current document.

2 Set Up the Wi-SUN PHY Mode Switch Configuration

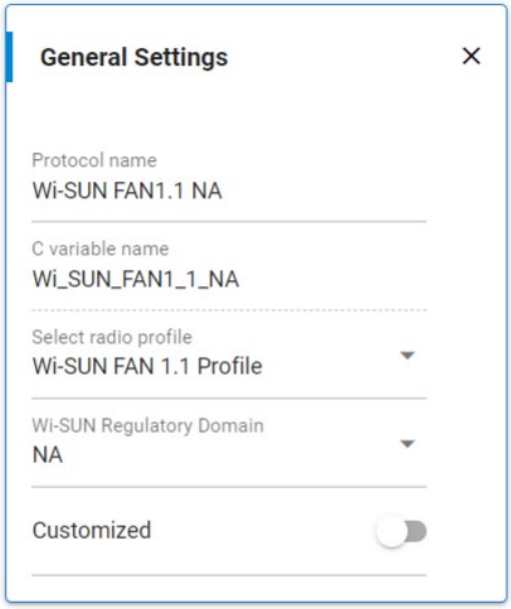
2.1 Mode Switch Configuration with the Radio Configurator

Once a new RAILtest project is created, the Radio Configurator opens automatically. You can then select the `Wi-SUN FAN 1.1 Profile`. The Radio Configurator provides a set of Wi-SUN FSK and OFDM PHYs as part of the Wi-SUN FAN 1.1 profile. The applicable Wi-SUN Regulatory Domain (for example NA, JP, EU, or BZ) according to Wi-SUN FAN 1.1 should be also selected.

To get a Mode Switch configuration between Wi-SUN PHYs, a channel-based multi-PHY configuration is required. More details on channel-based multi-PHY can be found in https://community.silabs.com/s/article/rail-tutorial-multi-phy-usecases-and-examples?language=en_US.

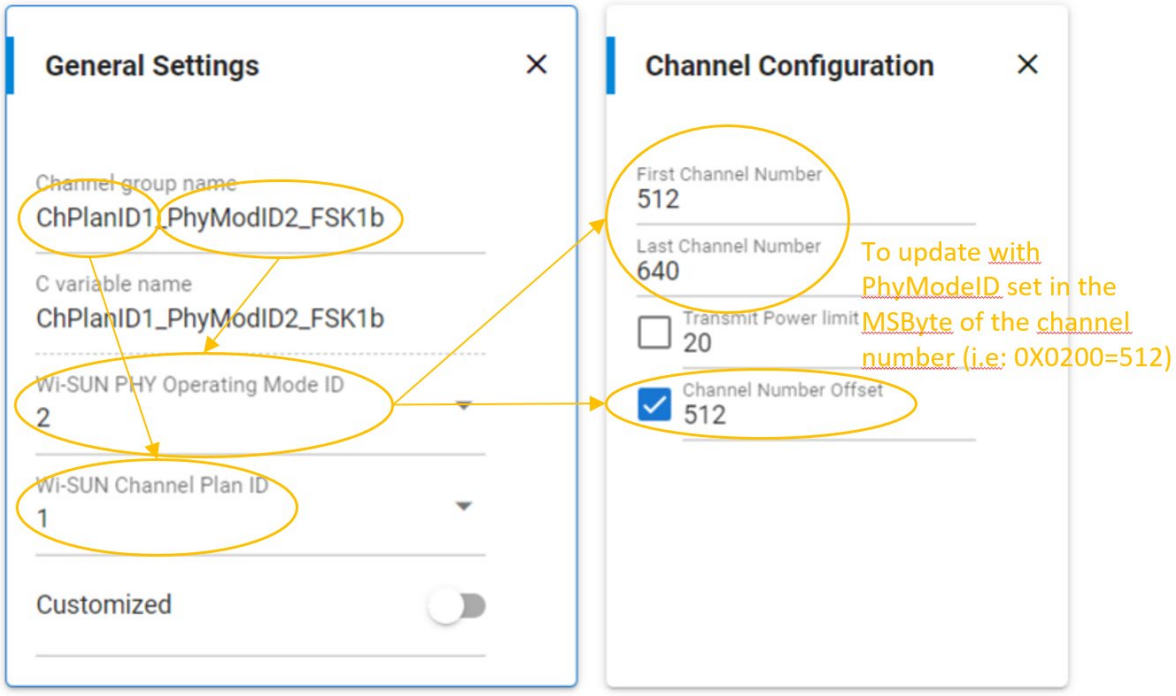
In the Radio Configurator, follow the steps below for an example for NA:

1. In the Protocol, select the **Wi-SUN FAN 1.1 Profile** and the **Wi-SUN Regulatory Domain**. In the following figure, the Wi-SUN **NA** Regulatory Domain is selected.



2. Update the default channel configuration. To get a Mode Switch configuration, you can add the PhyModeID as the MSByte of the channel number, which is on 2 bytes. This way, RAIL can associate each PhyModeID with a different channel number, which will be useful in Rx.

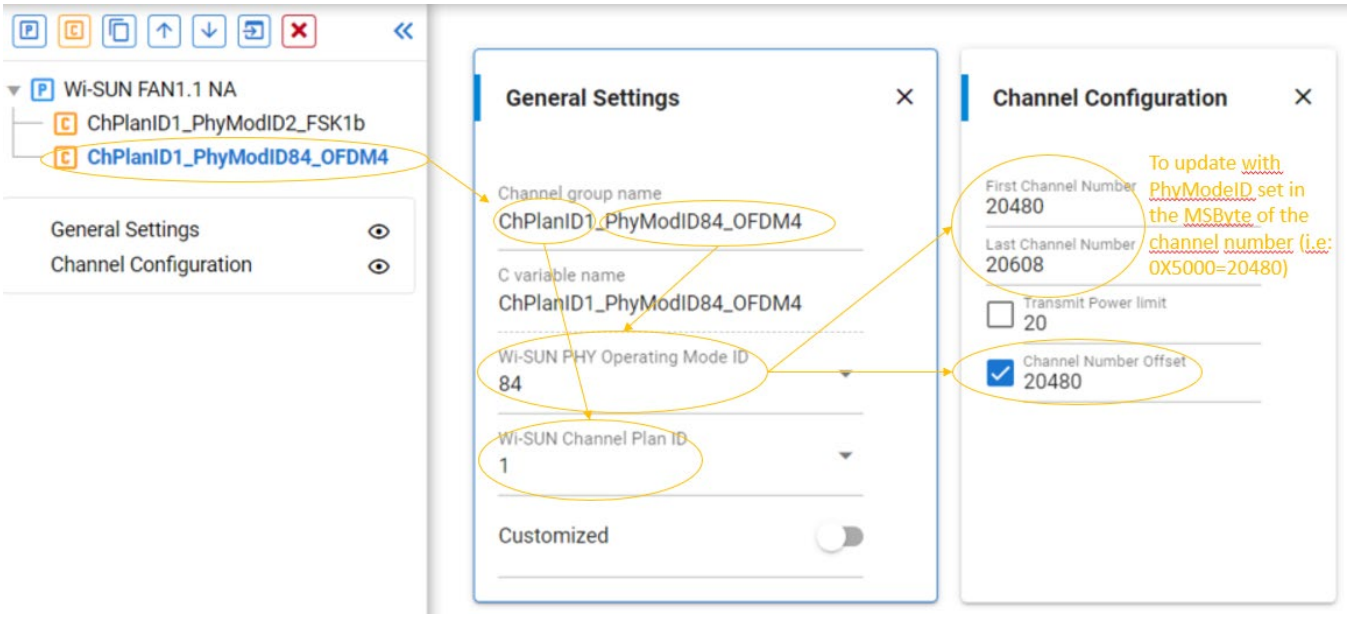
   For example, a PhyModeID = 2 (2FSK mode 1b 50 kbps mi=1) provides a channel number of 0x0200=512, as shown.



Note: The Wi-SUN Channel Plan ID list proposed depends on the Wi-SUN Regulatory Domain and on the selected Wi-SUN PHY Mode ID. The Wi-SUN Channel Plan ID selected supports automatically getting the base frequency and the channel spacing according to the Wi-SUN FAN 1.1 specification.

3. Add additional channel configurations for other PhyModeID using **Create new channel**:

4. Enter a **Channel group name** and the parameters as shown in the following example.



5. Add other channels and configure them as previously described for Mode Switch configuration. Use the following tables to compute the base channel number.

FSK PHY mode References	FEC enabled	FSK PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)
#1a	no	1	0x0100 = 256
#1b	no	2	0X0200 = 512
#2a	no	3	0x0300 = 768
#2b	no	4	0x0400 = 1024
#3	no	5	0x0500 = 1280
#4a	no	6	0x0600 = 1536
#4b	no	7	0x0700 = 1792
#5	no	8	0x0800 = 2048

FSK PHY mode References	FEC enabled	FSK PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)
#1a	yes	17	0x1100 = 4352
#1b	yes	18	0X1200 = 4608
#2a	yes	19	0x1300 = 4864
#2b	yes	20	0x1400 = 5120
#3	yes	21	0x1500 = 5376
#4a	yes	22	0x1600 = 5632
#4b	yes	23	0x1700 = 5888
#5	yes	24	0x1800 = 6144

Notes:

- **By default, the Mode Switch PPDU is always from an FSK to an FSK or an OFDM PhyModeID.** Therefore, after a switch to an OFDM PhyModeID, the application should use the `setchannel` RAILtest CLI command on both the Tx and Rx side to go back on an FSK PhyModeID. Alternatively, the RAILtest command `ModeSwitchLife 1` could be used (see section 3.1 The trig-ModeSwitchTx RAILtest CLI Command).

- Changing OFDM MCS within the same option does not require **Mode Switch PPDU but only to set the MCS using `set802154phr`.**
- The OFDM PHY in the *.radioconf* file for Mode Switch only requires selecting the lowest MCS of the OFDM option available in the selected Wi-SUN Regulatory Domain as described below for the NA Regulatory Domain. Nevertheless, the MSByte of the channel number could use MCS0 PhyModeID.

OFDM options	PhyModeID for MC0	Channel number (MSByte = PhyModeID and first channel=0)	PhyModeID with the lowest MCS for NA
Option 1	0x20 = 32	0x2000 = 8192	MCS2 0x22 = 34
Option 2	0x30 = 48	0x3000 = 12288	MCS3 0x33 = 51
Option 3	0x40 = 64	0x4000 = 16384	MCS4 0x44 = 68
Option 4	0x50 = 80	0x5000 = 20480	MCS4 0x54 = 84

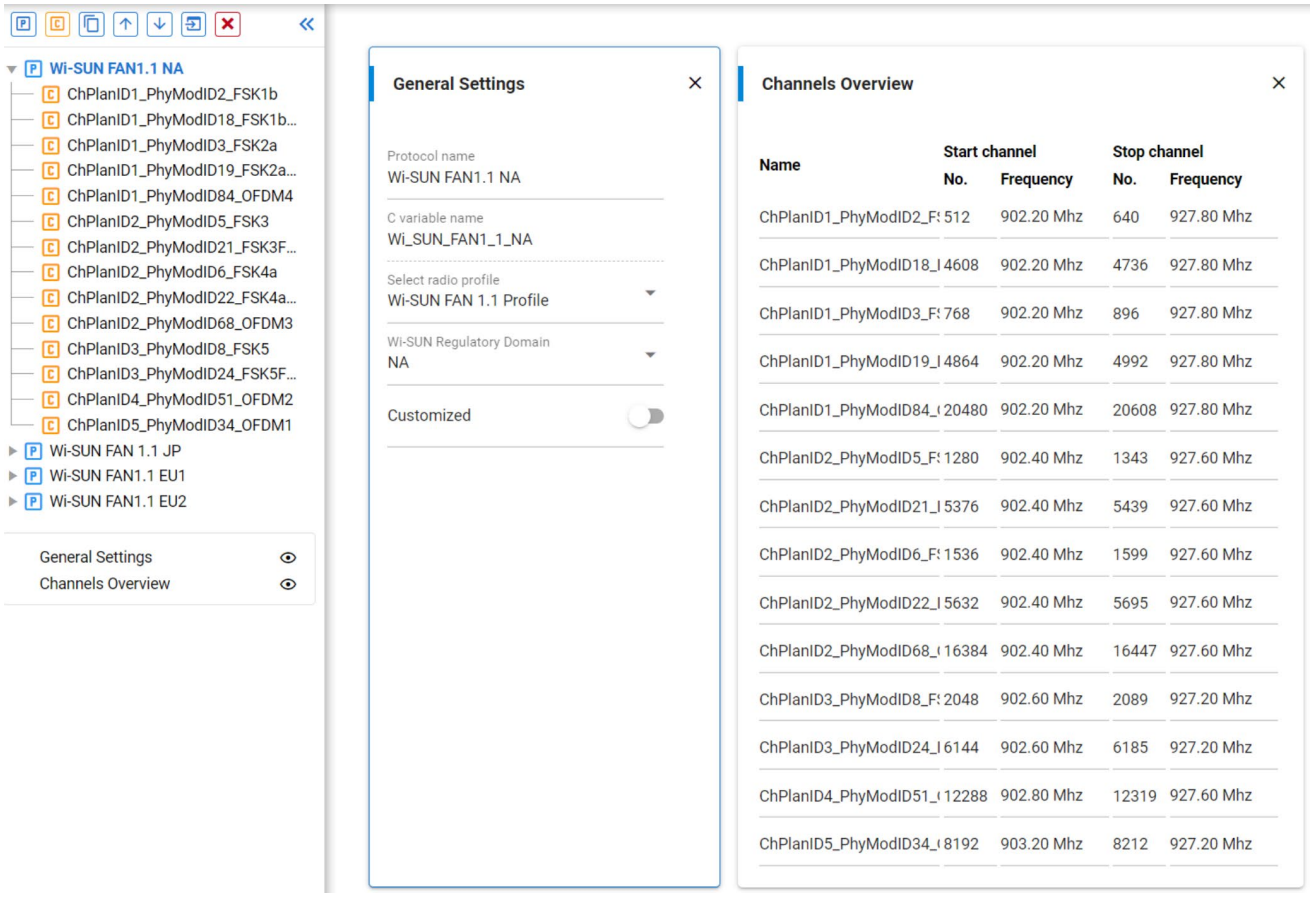
The `set802154phr` selects the MCS to be used and the PHYmodeID parameter in the `trigModeSwitchTx` selects the OFDM option, as described in section 3.4 Script Example.

2.2 Mode Switch *radio_settings.radioconf* Example for NA, JP, EU

NA, JP, and EU Mode Switch *radio_settings.radioconf* file is provided as an example with this document. This section provides details on the radio configuration settings.

2.2.1 NA Wi-SUN Regulatory d=Domain

For example, this Radio Configurator snapshot shows the Mode Switch Multi-PHY-based channel configuration for NA with Channel Plan ID 1, 2, 3, 4, and 5.



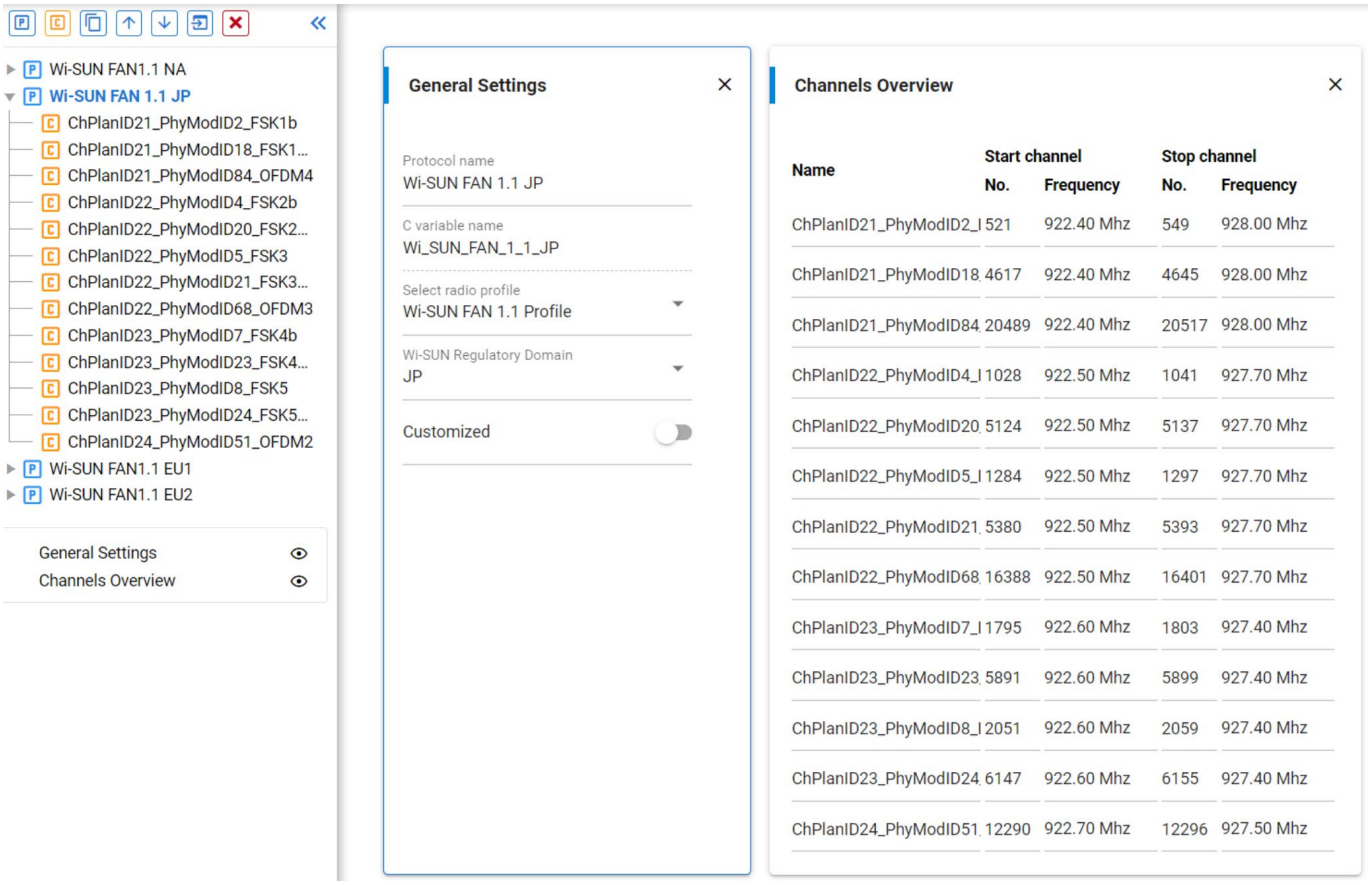
The Mode Switch configuration for the NA Regulatory Domain on `configindex 0` is summarized in the following table.

ChanPlanID	Base frequency (Hz)	Channel spacing (Hz)	PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)	Last channel number	PHY modes References
1	902200000	200000	2	0x0200 = 512	640	#1b
1	902200000	200000	18	0x1200 = 4608	4736	#1b with FEC
1	902200000	200000	3	0x0300 = 768	896	#2a
1	902200000	200000	19	0x1300 = 4864	4992	#2a with FEC
1	902200000	200000	80	0x5000 = 20480	20608	OFDM Option 4 MCS set with set802154phr
2	902400000	400000	5	0x0500 = 1280	1343	#3
2	902400000	400000	21	0x1500 = 5376	5439	#3 with FEC
2	902400000	400000	6	0x0600 = 1536	1599	#4a
2	902400000	400000	22	0x1600 = 5632	5695	#4a with FEC
2	902400000	400000	64	0x4000 = 16384	16447	OFDM Option 3 MCS set with set802154phr
3	902600000	600000	8	0x0800 = 2048	2089	#5
3	902600000	600000	24	0x1800 = 6144	6185	#5 with FEC
4	902800000	800000	48	0X3000 = 12288	12319	OFDM Option 2 MCS set with set802154phr
5	903200000	1200000	32	0x2000 = 8192	8212	OFDM Option 1 MCS set with set802154phr

Mode switch can be enabled across all PHYs specified in the **PhyModeID** column.

2.2.2 JP Wi-SUN Regulatory Domain

This Radio Configurator snapshot shows the Mode Switch Multi-PHY-based channel configuration for JP with Channel Plan ID 21, 22, 23, and 24.



The Mode Switch configuration for the JP Regulatory Domain on `configindex 1` is summarized in the following table.

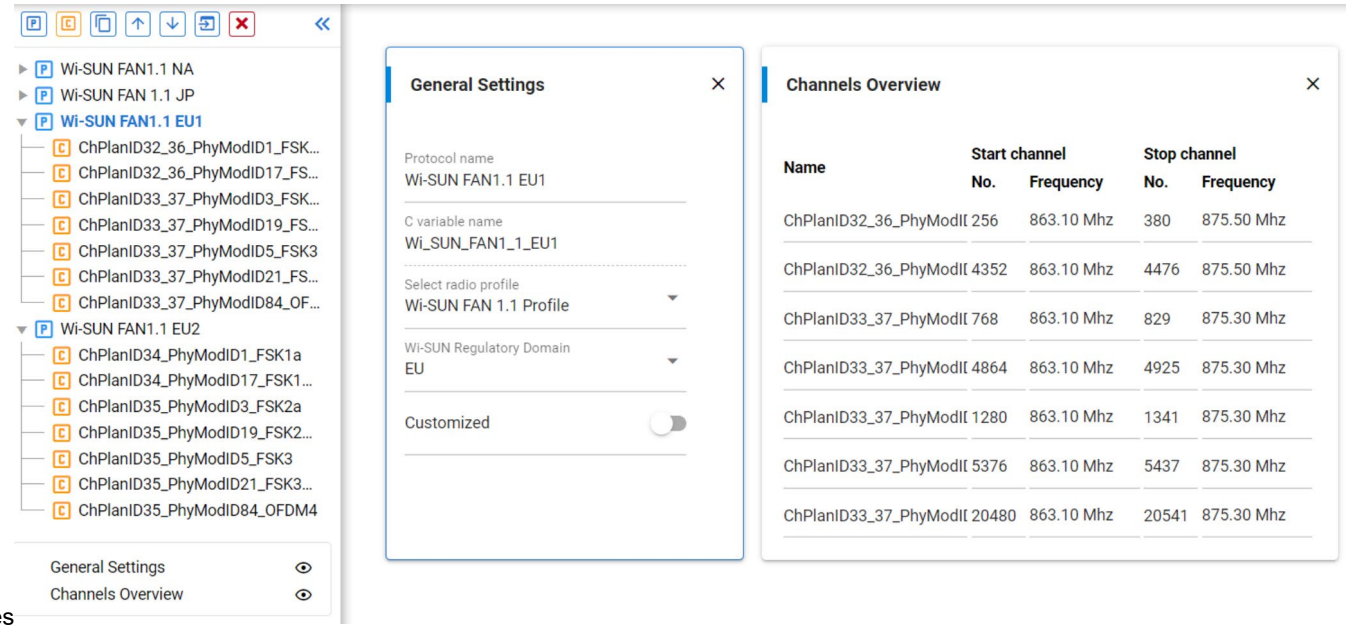
ChanPlanID	Base frequency (Hz)	Channel spacing (Hz)	PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)	First Channel number (not masked)	Last Channel number	PHY modes References
21	920600000	200000	2	0x0200 = 512	521	549	#1b
21	920600000	200000	18	0x1200 = 4608	4617	4645	#1b with FEC
21	920600000	200000	80	0x5000 = 20480	20489	20517	OFDM Option 4 MCS set with set802154phr
22	920900000	400000	4	0x0400 = 1024	1028	1041	#2b
22	920900000	400000	20	0x1400 = 5120	5124	5137	#2b with FEC
22	920900000	400000	5	0x0500 = 1280	1284	1297	#3
22	920900000	400000	21	0x1500 = 5376	5380	5393	#3 with FEC
22	920900000	400000	64	0x4000 = 16384	16388	16401	OFDM Option 3 MCS set with set802154phr
23	920800000	600000	7	0x0700 = 1792	1795	1803	#4b
23	920800000	600000	23	0x1700 = 5888	5891	5899	#4b with FEC
23	920800000	600000	8	0x0800 = 2048	2051	2059	#5
23	920800000	600000	24	0x1800 = 6144	6147	6155	#5 with FEC
24	921100000	800000	48	0X3000 = 12288	12290	12296	OFDM Option 2 MCS set with set802154phr

Mode switch can be enabled across all PHYs specified in the **PhyModeID** column.

Note: In Japan, a channel mask applies and forbids the use of the lower channels. The **First Channel number (not masked)** column indicates the first valid channel that can be used.

2.2.3 EU Wi-SUN Regulatory Domain Case 1

This Radio Configurator snapshot shows the Mode Switch Multi-PHY-based channel configuration for EU1. In this example, EU 1 includes both 863-870 MHz and 863-876 MHz bands, resulting in Channel Plan IDs 32, 33, 36, and 37, as shown.



yes

Note the ChanPlanID32 and 36 have the same configuration except that ChanPlanID36 uses the maximum number of channels. The same is true for ChanPlanID33 and 37.

The mode switch configuration for the EU Regulatory Domain on `configindex 2` is summarized in the following table.

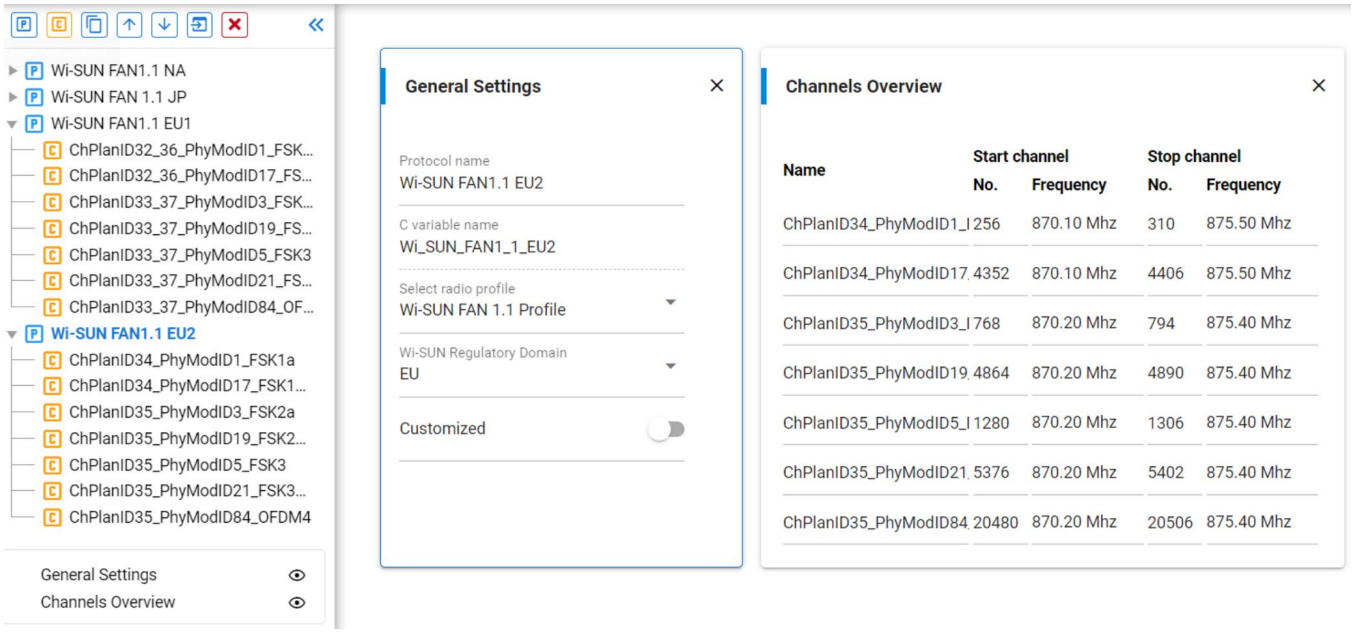
ChanPlanID	Base frequency (Hz)	Channel spacing (Hz)	PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)	First Channel number	Last Channel number	PHY modes References
32	863100000	100000	1	0x0100 = 256	256	324	#1a
32	863100000	100000	17	0x1100 = 4352	4352	4420	#1a with FEC
33	863100000	200000	3	0x0300 = 768	768	802	#2a
33	863100000	200000	19	0x1300 = 4864	4864	4898	#2a with FEC
33	863100000	200000	5	0x0500 = 1280	1280	1314	#3
33	863100000	200000	21	0x1500 = 5376	5376	5410	#3 with FEC
33	863100000	200000	80	0x5000 = 20480	20480	20514	OFDM Option 4 MCS set with set802154phr
36	863100000	100000	1	0x0100 = 256	256	380	#1a
36	863100000	100000	17	0x1100 = 4352	4352	4476	#1a with FEC
37	863100000	200000	3	0x0300 = 768	768	829	#2a
37	863100000	200000	19	0x1300 = 4864	4864	4925	#2a with FEC
37	863100000	200000	5	0x0500 = 1280	1280	1341	#3
37	863100000	200000	21	0x1500 = 5376	5376	5437	#3 with FEC
37	863100000	200000	80	0x5000 = 20480	20480	20541	OFDM Option 4 MCS set with set802154phr

Mode switch can be enabled across all PHYs specified in the **PhyModeID** column.

Note: In EU, a channel mask applies and forbids the use of some available channels in the middle and at the end of the RF band. The channel masking in the middle of the RF band is taken into account for Mode Switch thanks to a RAIL callback provided by the stack (see section 4 RAIL_lib API Usage for Mode Switch PPDU).

2.2.4 EU Wi-SUN Regulatory Domain Case 2

This Radio Configurator snapshot shows the Mode Switch Multi-PHY-based channel configuration for EU2 (870-876 MHz) with Channel Plan ID 34 and 35.



The Mode Switch configuration for the EU Regulatory Domain on `configindex 3` is summarized in the following table.

ChanPlanID	Base frequency (Hz)	Channel spacing (Hz)	PhyModeID	Channel number (MSByte = PhyModeID and first channel=0)	First Channel number	Last Channel number	PHY modes References
34	870100000	100000	1	0x0100 = 256	256	310	#1a
34	870100000	100000	17	0x1100 = 4352	4352	4406	#1a with FEC
35	870200000	200000	3	0x0300 = 768	768	794	#2a
35	870200000	200000	19	0x1300 = 4864	4864	4890	#2a with FEC
35	870200000	200000	5	0x0500 = 1280	1280	1306	#3
35	870200000	200000	21	0x1500 = 5376	5376	5402	#3 with FEC
35	870200000	200000	80	0x5000 = 20480	20480	20506	OFDM Option 4 MCS set with set802154phr

Mode switch can be enabled across all PHYs specified in the **PhyModeID** column.

Note: In EU, a channel mask applies and forbids the use of some available channels in the middle and at the end of the RF band. The channel masking in the middle of the RF band is taken into account for Mode Switch thanks to a RAIL callback provided by the stack (see section 4 RAIL_lib API Usage for Mode Switch PPDU).

3 RAILtest Script to Use Mode Switch

Before using the Mode Switch PPDU, reading *Getting Started with Wi-SUN PHY Configuration* is recommended. This is available on the Wi-SUN site in https://docs.silabs.com/.

3.1 The trigModeSwitchTx RAILtest CLI Command

The GSDK provides a new RAILtest CLI command, `trigModeSwitchTx`, to manage the Mode switch PPDU in Tx. After some RAIL initialization, this command sends the Mode Switch PPDU with the specified PHR to inform the Rx nodes to switch to the new PhyModeID.

```
> help trigModeSwitchTx

trigModeSwitchTx    Transmit a Mode Switch packet then transmit packets on the new PHY with current
                    TX options. Depending on modeSwitchLife configuration, after all iterations are
                    done, it either stays on the new PHY or returns to the base PHY.
                    [uint8] new PhyModeId, i.e. ID of the PHYMode we are switching to
                    [uint8] number of packets transmitted on the new PHY
                    [uint32opt] number of times the sequence mode switch packet followed by data
                    packets is repeated, default is 1 if argument is absent
                    [uint32opt] delayMilliseconds before switching back to base PHY after all packets
                    have been transmitted on the new PHY, default is 0 if argument is absent.
                    Multitimer is used for delay greater than 0.
> help modeSwitchLife
modeSwitchLife      Return to the base PHY after all data packets transmitted for TX or after
                    MODE_SWITCH_START event and multitimer expiration after first data packet
                    reception on new phy for RX.
                    [uint8] 0=stay on new PHY (normal Wi-SUN FAN behaviour)
                            1=return to base PHY (special test mode)
```

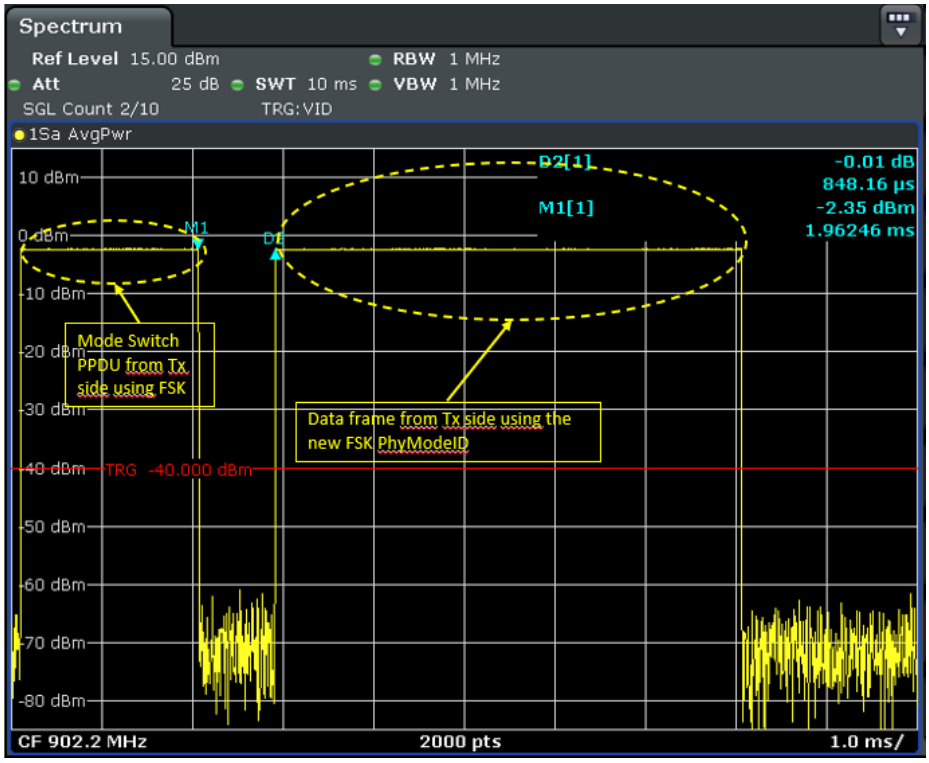
With `trigModeSwitchTx`, the CCA checking after the Mode switch PPDU is not implemented. To add it, call `setlbtmode csma` and `setlbtparams 0 0 1 -80 0 160 0` before calling `trigModeSwitchTx`. After the Mode Switch PPDU, the data frame must be transmitted within a maximum delay of 1.5 ms, as per the specification.

Note: the two last optional fields in `trigModeSwitchTx` and the `modeSwitchLife` command are only useful for test purposes.

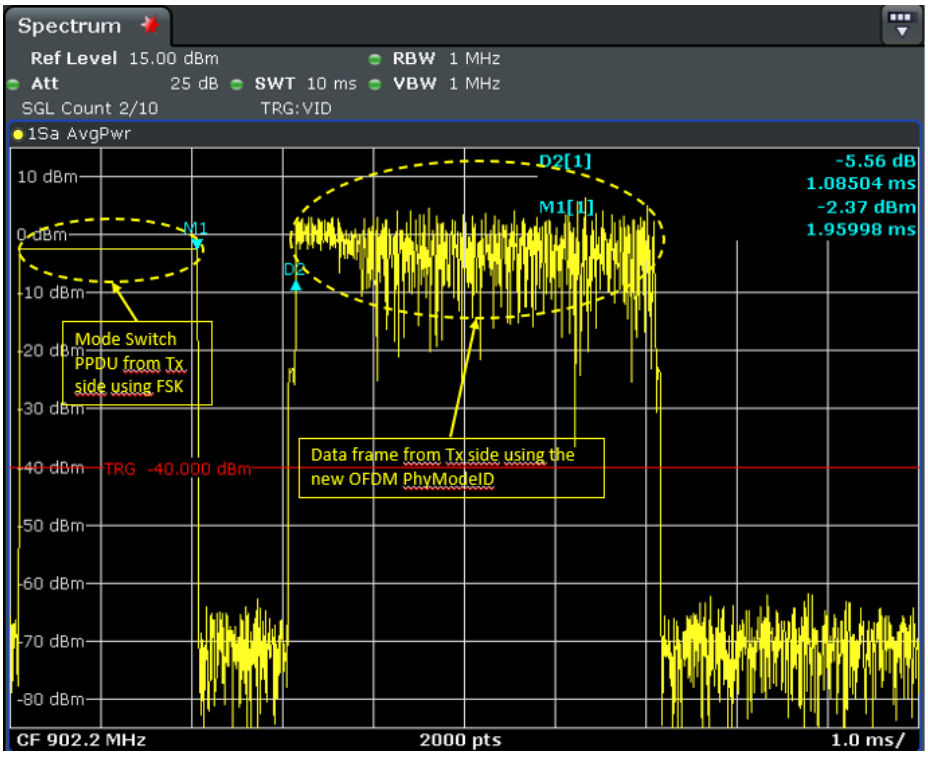
Note: The Mode Switch PPDU is transmitted in broadcast manner. Therefore, there is no MAC address filtering, and all receivers are impacted.

3.2 Mode Switch PPDU Snapshot in Zero Span Spectrum

The following shows a Mode Switch example from FSK to another FSK PhyModeID with CSMA-CA enabled, and illustrates the timing of frames during PhyModeID switch:



The following illustrates a Mode Switch from FSK to OFDM PhyModeID with CSMA-CA enabled.



Note: After the Mode Switch PPDU, the data frame might use another carrier frequency. The resolution bandwidth (RBW) of the spectrum analyzer should be set according to this new frequency, so that zero span can detect the data frame transmitted.

3.3 FSK and OFDM Channel-Based Multi-PHY: IR Calibration Initialization

For FSK or OFDM PHY (during initialization), after a `setchannel` on the corresponding PHY, an `rx 1` or `tx 1` command is required to perform the calibration. When using multiple PHYs including an OFDM PhyModeID, an IR calibration for OFDM must be performed. Silicon Labs strongly recommends loading one OFDM PHY (during initialization) using the RAILtest CLI commands `setchannel` on the corresponding OFDM channel and `rx 1` to perform the calibration. Once image rejection calibration has been performed with a Wi-SUN OFDM PHY, you do not need to do it with a Wi-SUN FSK PHY.

3.4 Script Example

This section presents a complete example showing how to configure and send the Mode Switch with the EFR32FG25 radio using the RAILtest application. It assumes the EFR32FG25 is flashed with an image with a Wi-SUN Mode Switch configuration.

The following steps are required:

- For all PA selected in the channel-based multi-PHY, the initialization should be done (only required once at initialization with FSK PA configuration before the OFDM PA configuration).
  - Set up the PA configuration.
  - Set the recommended PA level (check the maximum level depending on the radio board, 16 dBm for EFR32FG25).
- If IR calibration is enabled including an OFDM PhyModeID, then call `setchannel OFDM_base_channel` for IR calibration (see previous section).
- Mode Switch RAIL initialization must be done in Tx and Rx:
  - `Enable802154 rx 110 192 672 0`: Enable RAIL 802.15.4 management (needed to get Mode Switch enabled in RAIL). Note the timing parameters provided are for ACK management. This is not used in mode switch PPDU. Before calling this command, radio should be in idle mode.
  - `setRxOptions 4`: Enable DUALSYNC in Tx and Rx in case of FSK FEC usage in the channel-based multi-PHY.
  - `set802154g 7`: Enable Mode Switch and dynamic FEC, which are required for Mode Switch. Before calling this command, a FSK PHY should be selected.
  - `setPromiscuousMode 1`: Avoid MAC filtering on packet received.
  - `setTxdelay 10`: Set 10 ms between data frames.
  - `setTxLength 63`: Set the data frame length. This could be another frame length value. Remember to call `set802154Phr` after a new `setTxLength`. Note that FSK frame length corresponds to setTxLength value -2 and that OFDM frame length corresponds to setTxLength value -4.
  - `Resetcounters`: Reset the status counters.
- `setEventConfig 0x0 0x0 24576 24576` and `printEvents 0x0 24576`: In Rx, Mode Switch events are enabled and printed. The RAIL_EVENT_IEEE802154_MODESWITCH_START event occurs on Mode Switch PPDU reception. The RAIL_EVENT_IEEE802154_MODESWITCH_END event occurs in the following cases:
  - A timeout when no data packet is received after the mode switch packet (or if the packet is received too late)
  - A CRC error
  - A bad MAC address on the data packet received after the mode switch packet. With `setPromiscuousMode 1`, there is no MAC address filtering.
- `Set802154phr`: Uses different parameters between FSK and OFDM frame, as described in QSG181: Silicon Labs Wi-SUN Quick-Start Guide. To switch on OFDM PHY, `Set802154phr` should be set with the selected MCS. The new PhyModeID set in `trig-ModeSwitchTx` selects only the OFDM option.
- `trigModeSwitchTx`: set the new PhyModeID and the number of data frames to transmit.

These steps result in the following script:

```
# PA configuration/init (to do only one time to switch with correct PA from FSK to OFDM)
# and IR calibration using OFDM option and channel common to all regions (20489)
# and selection of FSK PHY channel (512) before calling set802154g
> rx 0
> setchannel 512
> rx 1
> rx 0
> setpowerconfig RAIL_TX_POWER_MODE_SUBGIG_POWERSETTING_TABLE 3600 10
> setpower 140
> setchannel 512
> rx 1
```

```
> rx 0
> setchannel 20489
> rx 1
> rx 0
> setpowerconfig RAIL_TX_POWER_MODE_OFDM_PA_POWERSETTING_TABLE 3600 10
> setpower 140
> setchannel 20489
> rx 1
> setchannel 512
> rx 1

# Mode Switch RAIL initialization for Mode Switch
> rx 0
> enable802154 rx 110 192 672 0
> setRxOptions 4
> set802154g 7
> rx 1
> setPromiscuousMode 1
> setTxDelay 10
> setTxLength 63
> resetCounters

# Only on Rx side: enable Mode Switch events:
# RAIL_EVENT_IEEE802154_MODESWITCH_START and RAIL_EVENT_IEEE802154_MODESWITCH_END
> setEventConfig 0x0 0x0 24576 24576
> printEvents 0x0 24576


# Only on Tx side to switch to FSK PHY: FSK PHR + trigModeSwitchTx
# with one packet sent with FSK PhyModId=3
> Set802154phr 1 0 1
> trigModeSwitchTx 3 1

# Only in Tx side to switch to OFDM PHY: one packet sent with OFDM PHR MCS4 and OFDM PhyModeID 80
> Set802154phr 2 4 0
> trigModeSwitchTx 80 1

# After Mode Switch step we should come back to FSK base channel in Tx and Rx side with:
> setchannel 512
```

Note: More details on RAILtest commands can be found in UG409: RAILtest User's Guide.

3.5 A Complete FSK-to-FSK Mode Switch Example in NA region

```
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1516549207}}
>> setchannel 512
  {{(setchannel)}{channel:512}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1516862048}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1517030918}}
> setpowerconfig RAIL_TX_POWER_MODE_SUBGIG_POWERSETTING_TABLE 3600 10
  {{(setpowerconfig)}{suc-
cess:true}{mode:RAIL_TX_POWER_MODE_SUBGIG_POWERSETTING_TABLE}{modeIndex:0}{voltage:3600}{rampTime:7
}}
> setpower 140
  {{(setpower)}{powerLevel:70}{power:140}}
> setchannel 512
  {{(setchannel)}{channel:512}}> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1517474858}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1519142526}}
> enable802154 rx 110 192 672 0
```

```
  {{(enable802154)}{802.15.4:Enabled}{rxDefaultState:Rx}{txDefaultState:Rx}{idleTiming:110}
  {turnaroundTime:192}{ackTimeout:672}{defaultFramePending:False}}
> setRxOptions 4
  {{(setRxOptions)}{storeCrc:False}{ignoreCrcErrors:False}{enableDualSync:True}{trackAborted:False}
  {removeAppendedInfo:False}{rxAntenna:Any}{frameDet:On}}
> set802154g 7
  {{(set802154g)}{15.4G_GB868:True}{15.4G_DynamicFEC:True}{Wi-SUN_ModeSwitch:True}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1519819463}}
> setPromiscuousMode 1
  {{(setPromiscuousMode)}{PromiscuousMode:Enabled}}
> setTxDelay 10
  {{(setTxDelay)}{txDelay:10}}
> setTxLength 63
  {{(setTxLength)}{TxLength:63}{TxLength Written:63}}
> resetCounters
  {{(resetCounters)}{UserTxCount:0}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
{AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:0}{RxCrcErrDrop:0}{SyncDetect:0}
{NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}
{ackTxFpAddrFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}{Channel:512}
{AppMode:None}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}{RxOverflow:0}
{AddrFilt:0}{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:0}{TxChannelBusy:0}{TxClear:0}
{TxCca:0}{TxRetry:0}{UserTxStarted:0}{PaProtect:0}{SubPhy0:0}{SubPhy1:0}{SubPhy2:0}{SubPhy3:0}
{rxRawSourceBytes:0x000000000}}
```

Note that after a `setchannel` it is recommended to do "`rx 1`" to get the PHY loaded.

The Mode Switch Rx events configuration:

```
> setEventConfig 0x0 0x0 24576 24576
  {{(setEventConfig)}{Mask:0x600000000000}{Values:0x600000000000}}
> printEvents 0x0 24576
  {{(printEvents)}{enablePrintEvents:0x600000000000}}
```

On the Tx side, the `set802154phr` and `trigModeSwitchTx` FSK usage to switch from mode 1b to mode 2a:

```
> Set802154phr 1 0 1
{{(Set802154phr)}{PhrSize:2}{PHR:0x8210}}
{{(Set802154phr)}{len:63}{payload: 0x10 0x82 0xb8 0x01 0x33 0x44 0x55 0x0f 0x77 0x88 0x99 0xaa 0xbb
0xcc 0xdd 0xee 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20
0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34
0x35 0x36 0x37 0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e}}
> trigModeSwitchTx 3 1
{{(txModeSwitch)}{PacketTx:Enabled}{None:Disabled}{Time:1522467866}}
{{(modeSwitchChangeChannel)}{channel:768}}
{{(appMode)}{None:Enabled}{PacketTx:Disabled}{Time:1522482020}}
{{(txEnd)}{txStatus:Complete}{transmitted:1}{lastTxTime:1522481935}{timePos:6}{lastTx-
Start:1522475758}{ccaSuccess:2}{failed:0}{lastTxStatus:0x000000000}{txRemain:0}{isAck:False}}
> status
{{(status)}{UserTxCount:2}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
{AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:0}{RxCrcErrDrop:0}{SyncDetect:0}
{NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}
{ackTxFpAddrFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}{Channel:768}
{AppMode:None}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}{RxOverflow:0}{AddrFilt:0}
{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:0}{TxChannelBusy:0}{TxClear:2}{TxCca:2}
{TxRetry:0}{UserTxStarted:2}{PaProtect:0}{SubPhy0:0}{SubPhy1:0}{SubPhy2:0}{SubPhy3:0}
{rxRawSourceBytes:0x000000000}}
```

On the Rx side, the expected event and the Rx frame are returned:

```
{{(event)}{timestamp:2942451808}{eventName:RAIL_EVENT_IEEE802154_MODESWITCH_START}}
{{(rxPacket)}{len:63}{timeUs:2942453545}{timePos:5}{crc:Pass}{filterMask:0x0}{rssi:-8}{lqi:255}
{phy:0}{isAck:False}{syncWordId:0}{antenna:0}{channelHopIdx:254}{channel:768}{ed154:255}
{lqi154:255}{payload: 0x10 0x82 0xb8 0x01 0x33 0x44 0x55 0x0f 0x77 0x88 0x99 0xaa 0xbb 0xcc 0xdd 0xee
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20 0x21 0x22 0x23
```

```
0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e}}
> status
{{(status)}{UserTxCount:0}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
{AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:1}{RxCrcErrDrop:0}{SyncDetect:2}
{NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}{ackTxFpAd-
drFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}{Channel:768}{AppMode:N
one}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}{RxOverflow:0}{Ad-
drFilt:0}{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:1}{TxChannelBusy:0}{TxClear:0}
{TxCca:0}{TxRetry:0}{UserTxStarted:1}{PaProtect:0}{SubPhy0:9}{SubPhy1:0}{SubPhy2:0}{SubPhy3:0}
{rxRawSourceBytes:0x000000000}}
```

Notes:

1. In Rx status, there are two SyncDetect frames corresponding to Mode Switch PPDU plus the data frame using the new PhyModeID. There is only one Rxcount frame corresponding to the data frame received. The channel is also changed to 768, which is expected.
2. After a Mode Switch PPDU, RAIL does not come back automatically to the FSK base PHY. In the example above, `setchannel 512` must be used to return to the base FSK, before a potential switch to another PHY.

3.6 A Complete FSK-to-OFDM Mode Switch Example in NA Region

This example shows the initialization of the two PAs and the Mode Switch in RAIL with returned console messages:

```
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1516549207}}
>> setchannel 512
  {{(setchannel)}{channel:512}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1516862048}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1517030918}}
>  setpowerconfig  RAIL_TX_POWER_MODE_SUBGIG_POWERSETTING_TABLE  3600  10    {{(setpowerconfig)}{suc-
cess:true}{mode:RAIL_TX_POWER_MODE_SUBGIG_POWERSETTING_TABLE}{modeIndex:0}{voltage:3600}{rampTime:7
}}
> setpower 140
  {{(setpower)}{powerLevel:70}{power:140}}
>> setchannel 512
  {{(setchannel)}{channel:512}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1517474858}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1517611032}}
> setchannel 20489
  {{(setchannel)}{channel:20489}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1517885057}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1518019524}}
>  setpowerconfig  RAIL_TX_POWER_MODE_OFDM_PA_POWERSETTING_TABLE  3600  10    {{(setpowerconfig)}{suc-
cess:true}{mode:RAIL_TX_POWER_MODE_OFDM_PA_POWERSETTING_TABLE}{modeIndex:2}{voltage:3600}{rampTime:
7}}
> setpower 140
  {{(setpower)}{powerLevel:118}{power:135}}
> setchannel 20489
  {{(setchannel)}{channel:20489}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1518471563}}
> setchannel 512
  {{(setchannel)}{channel:512}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1518776089}}
> rx 0
  {{(rx)}{Rx:Disabled}{Idle:Enabled}{Time:1519142526}}
> enable802154 rx 110 192 672 0
  {{(enable802154)}{802.15.4:Enabled}{rxDefaultState:Rx}{txDefaultState:Rx}{idleTiming:110}
```

```
  {turnaroundTime:192}{ackTimeout:672}{defaultFramePending:False}}
> setRxOptions 4
  {{(setRxOptions)}{storeCrc:False}{ignoreCrcErrors:False}{enableDualSync:True}{trackAborted:False}
{removeAppendedInfo:False}{rxAntenna:Any}{frameDet:On}}
> set802154g 7
  {{(set802154g)}{15.4G_GB868:True}{15.4G_DynamicFEC:True}{Wi-SUN_ModeSwitch:True}}
> rx 1
  {{(rx)}{Rx:Enabled}{Idle:Disabled}{Time:1519819463}}
> setPromiscuousMode 1
  {{(setPromiscuousMode)}{PromiscuousMode:Enabled}}
> setTxDelay 10
  {{(setTxDelay)}{txDelay:10}}
> setTxLength 63
  {{(setTxLength)}{TxLength:63}{TxLength Written:63}}
> resetCounters
  {{(resetCounters)}{UserTxCount:0}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
{AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:0}{RxCrcErrDrop:0}{SyncDetect:0}
{NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}
{ackTxFpAddrFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}{Channel:512}
{AppMode:None}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}{RxOverflow:0}
{AddrFilt:0}{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:0}{TxChannelBusy:0}{TxClear:0}
{TxCca:0}{TxRetry:0}{UserTxStarted:0}{PaProtect:0}{SubPhy0:0}{SubPhy1:0}{SubPhy2:0}{SubPhy3:0}
{rxRawSourceBytes:0x000000000}}
```

Note that after a `setchannel` it is recommended to do "`rx 1`" to get the PHY loaded.

The Mode Switch Rx events configuration:

```
> setEventConfig 0x0 0x0 24576 24576
  {{(setEventConfig)}{Mask:0x600000000000}{Values:0x600000000000}}
> printEvents 0x0 24576
  {{(printEvents)}{enablePrintEvents:0x600000000000}}
```

On the Tx side, the `set802154phr` OFDM and `trigModeSwitchTx` from FSK to OFDM to switch from mode #1b to OFDM option 4 MCS4:

```
> Set802154phr 2 4 0
  {{(Set802154phr)}{PhrSize:4}{PHR:0x1b80400}}
  {{(Set802154phr)}{len:63}{payload: 0x00 0x04 0xb8 0x01 0x33 0x44 0x55 0x0f 0x77 0x88 0x99 0xaa 0xbb
0xcc 0xdd 0xee 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20
0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34
0x35 0x36 0x37 0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e}}
> trigModeSwitchTx 80 1
  {{(txModeSwitch)}{PacketTx:Enabled}{None:Disabled}{Time:3925975352}}
  {{(modeSwitchChangeChannel)}{channel:20480}}
  {{(appMode)}{None:Enabled}{PacketTx:Disabled}{Time:3925988859}}
  {{(txEnd)}{txStatus:Complete}{transmitted:1}{lastTxTime:3925988787}{timePos:6}
  {lastTxStart:3925984019}{ccaSuccess:2}{failed:0}{lastTxStatus:0x000000000}{txRemain:0}
  {isAck:False}}
> status
  {{(status)}{UserTxCount:2}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
  {AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:0}{RxCrcErrDrop:0}{SyncDetect:0}
  {NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}
  {ackTxFpAddrFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}
  {Channel:20480}{AppMode:None}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}
  {RxOverflow:0}{AddrFilt:0}{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:0}{TxChannelBusy:0}
  {TxClear:2}{TxCca:2}{TxRetry:0}{UserTxStarted:2}{PaProtect:0}{SubPhy0:0}{SubPhy1:0}{SubPhy2:0}
  {SubPhy3:0}{rxRawSourceBytes:0x000000000}}
```

On the Rx side, the expected event and the Rx frame are returned {phy:4} for MCS4:

```
{{(event)}{timestamp:4285050697}{eventName:RAIL_EVENT_IEEE802154_MODESWITCH_START}}
{{(rxPacket)}{len:59}{timeUs:4285055046}{timePos:5}{crc:Pass}{filterMask:0x0}{rssi:-
6}{lqi:203}{phy:4}{isAck:False}{syncWordId:0}{antenna:0}{channelHopIdx:254}{channel:20480}
{ed154:255}{lqi154:255}{payload: 0x03 0x04 0xb8 0x01 0x33 0x44 0x55 0x0f 0x77 0x88 0x99 0xaa 0xbb
0xcc 0xdd 0xee 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20
```

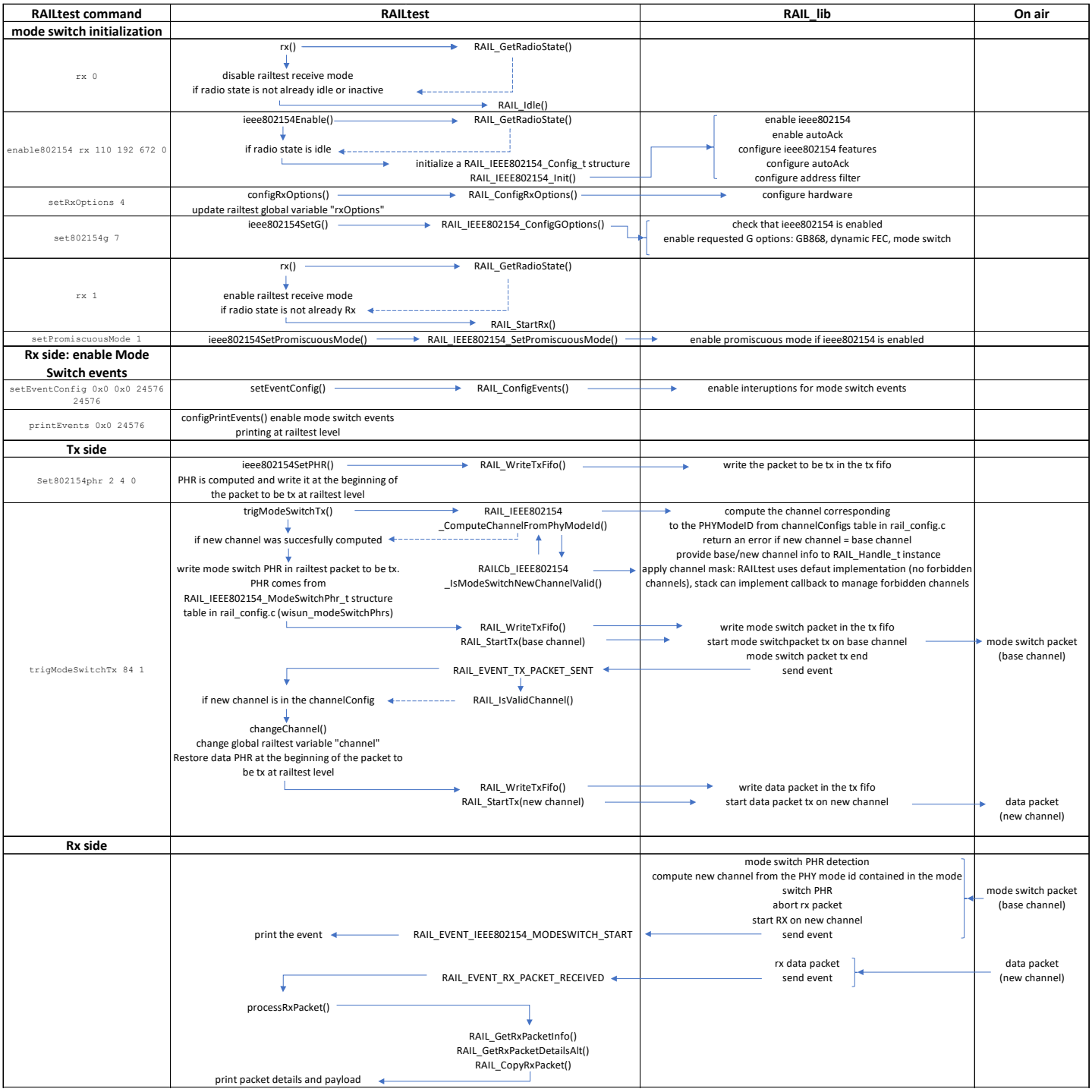
```
0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34
0x35 0x36 0x37 0x38 0x39 0x3a}}
status
{{(status)}{UserTxCount:0}{AckTxCount:0}{UserTxAborted:0}{AckTxAborted:0}{UserTxBlocked:0}
{AckTxBlocked:0}{UserTxUnderflow:0}{AckTxUnderflow:0}{RxCount:1}{RxCrcErrDrop:0}{SyncDetect:2}
{NoRxBuffer:0}{TxRemainErrs:0}{RfSensed:0}{ackTimeout:0}{ackTxFpSet:0}{ackTxFpFail:0}
{ackTxFpAddrFail:0}{RfState:Rx}{RAIL_state_active:0}{RAIL_state_rx:1}{RAIL_state_tx:0}
{Channel:20480}{AppMode:None}{TimingLost:0}{TimingDetect:0}{FrameErrors:0}{RxFifoFull:0}
{RxOverflow:0}{AddrFilt:0}{Aborted:0}{RxBeams:0}{DataRequests:0}{Calibrations:0}{TxChannelBusy:0}
{TxClear:0}{TxCca:0}{TxRetry:0}{UserTxStarted:0}{PaProtect:0}{SubPhy0:0}{SubPhy1:0}{SubPhy2:0}
{SubPhy3:0}{rxRawSourceBytes:0x000000000}}
```

Notes:

1. In Rx status, there are two SyncDetect frames corresponding to Mode Switch PPDU plus the data frame using the new PhyModeID. There is only one Rxcount frame corresponding to the data frame received. The channel is also changed to 20480, which is expected with {phy:4} for MCS4.
2. After a Mode Switch PPDU, RAIL does not come back automatically to the FSK base PHY. In the example above, `setchannel 512` must be used to return to the base FSK, before a potential switch to another PHY.

4 RAIL_lib API Usage for Mode Switch PPDU

The following figure provides details about how the RAIL_lib API is used by RAILtest to send/receive a mode switch PPDU followed by a data packet on the new channel. For each RAILtest command taken from the example script in section 3.4 Script Example, RAILtest behavior and especially the calls to the RAIL_lib API are explained. It also clarifies the most significant actions performed by RAIL_lib for the mode switch.



Notes:

- RAIL checks the first bit of the PHR to know if it is Mode Switch PPDU. If Mode Switch is not enabled through `ieee802154Enable()` function, then the Mode Switch PPDU is deleted and not processed.

- The channel mask defined in Wi-SUN FAN 1.1 is managed at the stack level through the callback used by `RAIL RAILCb_IEEE802154_isModeSwitchNewChannelValid()` as shown above.

RAIL_lib structures used for mode switch:

- RAIL_IEEE802154_Config_t: This configuration structure is used to initialize IEEE802.15.4 features, which is required to use mode switch. It is defined in the *rail_ieee802154.h* file and an example of initialization by RAILtest can be found in *rail-test/app_ci/154_rx_ci.c* file in the `ieee802154Enable()` function.
- RAIL_IEEE802154_ModeSwitchPhr_t: This structure contains a PHYModeID and the corresponding mode switch PHR. It is defined in the *rail_ieee802154.h* file.
  The *rail_config.c* file contains a RAIL_IEEE802154_ModeSwitchPhr_t table (wisun_modeSwitchPhrs) which is used to format a mode switch packet by writing the mode switch PHR corresponding to the new channel PHYModeId. An example of mode switch packet formatting can be found in *railtest/app_ci/154_rx_ci.c* file in the `trigModeSwitchTx()` function. The mode switch PHR is described in on Wi-SUN PHY Technical Profile Specification 2V00.

When a Mode switch configuration is saved in *radio_settings.radioconf*, then the generated *rail_config.c* should contain a constant table `RAIL_IEEE802154_ModeSwitchPhr_t wisun_modeSwitchPhrs` with the selected PhyModeID, as shown in the following example.

```
const RAIL_IEEE802154_ModeSwitchPhr_t wisun_modeSwitchPhrs[4] = {
  {
    .phyModeId = 2U,
    .phr = 31233U,
  },
  {
    .phyModeId = 18U,
    .phr = 43585U,
  },
  {
    .phyModeId = 3U,
    .phr = 40449U,
  },
  {
    .phyModeId = 19U,
    .phr = 20033U,
  },
};
```

5 Known Issues

This section lists only the known issues related to Mode Switch. Other known issues can be found in *Getting Started with Wi-SUN PHY Configuration.* This is available on the Wi-SUN site in https://docs.silabs.com/.

- WSTK's LCD and LEDs status are not correctly updated in Rx after `trigModeSwitchTx`.
- When CSMA-CA is enabled, the Mode Switch from PhyModeID 18 to OFDM is not successful and the event *RAIL_EVENT_IEEE802154_MODESWITCH_END* is raised.

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com