AN879

APPLICATION FRAMEWORK COMMAND LINE INTERFACE FOR RF4CE

This document describes the command line interface (CLI) functionality provided in the Application Framework for RF4CE-based applications.

New in This Revision

Initial release.

Contents

1 Introduction

The Silicon Labs RF4CE offering includes CLI commands that enable the user to seamlessly interact with and control the communication between RF4CE devices. These commands allow the user to easily perform simple tasks such as, but not limited to, pairing devices and viewing pairing information, sending custom messages between devices, setting RF4CE network parameters, and accessing security keys. Upon reaching the end of this document, the user will not only have a strong sense of the vast capabilities of the RF4CE CLI commands, but also will be able to use them to control RF4CE functionality.

To begin, there will be a number of walkthrough sections, each going over a different piece of basic functionality. The walkthrough scenario will consist of two basic RF4CE applications: one target and one controller. Both of the devices will include the plugins rf4ce-profile and rf4ce-zrc11. The commands will be referred to in `this font` and any output of the commands will be presented in *italics*.

2 Getting Started

When starting over with a new application, it is sometimes helpful to do a complete reset of the nodes. This involves tearing down any old networks and clearing the pairing table. To do this, issue this command on both the target and the controller:

```
plugin rf4ce-profile stop
```

It is also sometimes helpful to issue a reset command in order to clean out the RAM of the device:

```
reset
```

The combination of these first two commands is a good starting point if a situation arises where the nodes are just not communicating properly. For a sanity check, one can make sure that the pairing tables are empty. On both the controller and the target, issue the command:

```
plugin rf4ce-profile print
```

The printout should show something like *0 of D pairings* used, where *D* is the total number of pairings in the pairing table.

3 Network Formation

For the two devices to be able to communicate with each other, the controller must be paired with the target. First, though, both devices need to start their network operations. To do this, issue the following command to both nodes:

```
plugin rf4ce-profile start
```

After a couple of seconds, the target will create a network. Check the target's network parameters using this command:

```
info
```

Next, the discovery and pairing process needs to take place. There are multiple ways to use the CLI commands to pair devices. Two such methods are discussed here. First, one can pair the devices using the parameters of the network that the target has created. Let the PAN ID and node ID of the target be 0x1234 and 0xABCD, respectively. First, issue the command to the target, so that it starts its auto-discovery process:

```
plugin rf4ce-zrc11 pair recipient
```

Immediately after, issue the following command to the controller, with the network parameters of the desired target:

```
plugin rf4ce-zrc11 pair originator 0x1234 0xABCD 0xFF
```

To check if the pairing succeeded, print the pairing table on both devices as before:

```
plugin rf4ce-profile print
```

There should be a line in the printout reading *1 of D pairings used*, where *D* is the size of the pairing table.

Now, consider the case where either the target's network info is not known or there is simply the need for a controller to pair with any other target. The previous pairing routine can be repeated with wildcards for the PAN ID and node ID. Remember, this step cannot be performed right after the previous one, because the entry in the pairing table would simple be a repeat. Again, start off by instructing the target to start the auto discovery process:

```
plugin rf4ce-zrc11 pair recipient
```

Now, on the controller, issue the command:

```
plugin rf4ce-zrc11 pair originator 0xFFFF 0xFFFF 0xFF
```

Again, one can easily check if the pairing was successful by viewing the pairing table:

```
plugin rf4ce-profile print
```

Now the devices should be ready to communicate.

4 Sending Messages

Finally, the nodes are ready to communicate. In order for the commands in this section to work, the two devices must have already been paired.

To verify that a message has correctly been sent between two devices, one can use packet capture. Furthermore, for encrypted traffic, the security key may need to be known in order to make sure that the plain text sent over the air was correct. To discover what key the pairing is using, issue the following familiar command:

```
plugin rf4ce-profile print
```

At the end of the pairing entry, the 16-byte key should be displayed as an octet string. This is the key that the devices are using to encrypt their RF4CE traffic.

There are a handful of simple commands to control communication between devices, and two will be discussed here. The simplest form of communication is to send raw bytes in the network payload. Say that the desired payload is this sequence of bytes: 0x12 0x34 0x56 0x78. To send this information, issue the following command on either the target or the controller:

```
plugin rf4ce-profile send 0 0xFF {12 34 56 78}
```

That command tells the device to send a network payload of 0x12 0x34 0x56 0x78 to pairing index 0 with a profile ID of 0xFF (the wildcard profile ID).

Another simple form of communication is through the rf4ce-zrc11 plugin. This type of transmission can be visualized as a user pressing the "select" button on a TV remote (the controller) and the TV (the target) receiving that message. From the controller, issue the command:

```
plugin rf4ce-zrc11 press 0 0x00 {00} 1
```

This means, send a one-time command of 0x00 (the "select" control code) with a payload of 0x00 to the RF4CE device at pairing index 0.

5 Command Index

Here is a comprehensive index for all of the RF4CE CLI commands. They are grouped by plugin. The format of the table rows is as follows.

RF4CE plugin name	CLI command	First argument type and name	First argument description	Description of the CLI command

		Second argument type and name	Second argument description	
		Third argument type and name	Third argument description	

Here is an example table row.

rf4ce-my-functionality	action	INT8U channel	The channel on which to take the action	Performs the action for my-functionality on channel and with IEEE address.
		IEEE_ADDRESS address	The IEEE address of the node with which to attempt the action.	

One can execute this command by typing, for example:

```
plugin rf4ce-my-functionality action 15 {01 02 03 04 05 06 07 08}
```

Note: When there are arguments of type BOOLEAN, the capitalized words TRUE and FALSE may appear in the description of the argument. Here, the user is still expected to enter a nonnegative integer between 0 and 255 (inclusive). Any non-zero value will evaluate to TRUE.

Plugin Name	Command	Arguments		Description
rf4ce-profile	start	(None)		Start an RF4CE network
rf4ce-profile	stop	(None)		Stop the network operations of an RF4CE device
rf4ce-profile	pair	INT8U channel	The logical channel of the device with which to pair	Initiate the pairing process
		INT16U panId	The PAN identifier of the device with which to pair	

		IEEE_ADDRESS eui64	The IEEE address of the device with which to pair	
		INT8U keyExchangeTransferCount	The number of transfers the target should use to exchange the link key with the pairing originator	
rf4ce-profile	send	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send an RF4CE profile message
		INT8U profileId	The profile ID to be included in the RF4CE network header of the outgoing RF4CE network DATA frame	
		OCTET_STRING message	The message to be sent	
rf4ce-profile	vendor	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send a vendor-specific RF4CE profile message

		INT8U profileId	The profile ID to be included in the RF4CE network header of the outgoing RF4CE network DATA frame	
		INT16U vendorId	The vendor ID to be included in the RF4CE network header of the outgoing RF4CE network DATA frame	
		OCTET_STRING message	The message to be sent	
rf4ce-profile	unpair	INT8U pairingIndex	The index of the entry in the pairing table with which to unpair	Initiate the unpairing process
rf4ce-profile	power	INT32U dutyCycle	The duty cycle of a device in milliseconds	Set the power saving parameters
		INT32U activePeriod	The active period of a device in milliseconds	
rf4ce-profile	agility	INT8U rssiWindowSize	The number of the most recent RSSI reads that are taken into consideration to decide whether a channel switch is	Set the frequency agility parameters

			required or not	
		INT8U channelChangeReads	The number of RSSI reads above the RSSI threshold that will trigger a channel switch	
		INT8S rssiThreshold	The RSSI threshold above which a channel will be considered congested	
		INT16U readInterval	The interval length in seconds between two consecutive RSSI reads	
		INT8U readDuration	The exponent of the number of scan periods of the RSSI read process	
rf4ce-profile	lqi	INT8U threshold	The LQI threshold below which discovery requests will be rejected	Set the discovery LQI threshold parameter
rf4ce-profile	print	(None)		Print the pairing table
rf4ce-gdp	push-button	BOOLEAN setPending	0 to clear the flag or any other value to set it	Set or clear the "push-button stimulus received" flag
rf4ce-gdp	initiate-key-exchange	INT8U pairingIndex	The index of the entry in the pairing table with which to initiate key exchange	Initiate key exchange
rf4ce-gdp	set-validation	BOOLEAN success	0 if validation has failed or any other value if has succeeded	Accept or reject the validation
rf4ce-gdp	push-attribute	INT8U pairingIndex	The index of the entry in the pairing table to which to send the	Send a Push Attributes command

			command	
		INT8U profileId	The profile id	
		INT16U vendorId	The vendor id	
		INT8U attributeId	The attribute id	
		INT16U entryId	The entry id	
		OCTET_STRING value	The value	
rf4ce-gdp	identify	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send an Identify command
		INT16U vendorId	The vendor id	
		INT8U flags	The identify flags	
		INT16U timeS	The identify time in seconds	
rf4ce-gdp-identification-client	user-interaction	(None)		Indicate that user interaction has occurred
rf4ce-gdp-identification-server	identify	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send an Identify command to an identification client. If the local device is a polling server and the identification client is also a polling client, the plugin will wait for the next poll before sending the command
		INT8U flags	The identify flags	
		INT16U timeS	The identify time in seconds	
rf4ce-zrc11	pair originator	INT16U panId	The PAN ID of the destination device for the discovery	Initiate the push-button pairing operation as an originator
		INT16U nodeId	The network address of the destination device for the discovery	

		INT8U searchDevType	The device type to discover	
rf4ce-zrc11	pair recipient	(None)		Initiate the push-button pairing operation as a recipient
rf4ce-zrc11	press	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send User Control Pressed command *
		INT8U rcCommandCode	The HDMI CEC operand [UI Command]	
		OCTET_STRING rcCommandPayload	The additional operands, if any, required by the HDMI CEC command	
		BOOLEAN atomic	0 if the command should repeat or anything else if the command only be sent once	
rf4ce-zrc11	release	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send User Control Released command *
		INT8U rcCommandCode	The HDMI CEC operand [UI Command]	
rf4ce-zrc11	discovery	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send Command Discovery Request command
rf4ce-zrc20	bind	INT8U searchDevType	The device type to discover	Initiate the binding procedure
rf4ce-zrc20	proxy-bind	INT16U panId	The pan id of the recipient	Initiate the proxy binding procedure
		IEEE_ADDRESS ieee	The IEEE address of the recipient	

rf4ce-zrc20	start	INT8U pairingIndex	The index of the entry in the pairing table to which to send the action	Start an action *
		INT8U actionBank	The action bank	
		INT8U actionCode	The action code	
		INT8U actionModifier	The action modifier bits	
		INT16U actionVendorId	The action vendor id	
		OCTET_STRING actionData	The action data	
		BOOLEAN atomic	0 for an atomic action or 1 for a repeating action	
rf4ce-zrc20	stop	INT8U pairingIndex	The index of the entry in the pairing table to which the action was sent	Stop an action *
		INT8U actionBank	The action bank	
		INT8U actionCode	The action code	
		INT8U actionModifier	The action modifier bits	
		INT16U actionVendorId	The action vendor id	
rf4ce-zrc20-action-mapping-client	clearentries	(None)		Delete all action mapping entries
rf4ce-zrc20-action-mapping-client	removeentry	INT8U pairingIndex	Pairing index	Delete action mapping entry
		INT8U actionDeviceType	Action device type	
		INT8U actionBank	Action bank	
		INT8U actionCode	Action code	
rf4ce-zrc20-action-mapping-client	getentry	INT8U pairingIndex	The index of the entry in the pairing table to which to send the action	Get an action mapping entry
		INT8U actionDeviceType	The action device type	

		INT8U actionBank	The action bank	
		INT8U actionCode	The action code	
rf4ce-zrc20-action-mapping-client	setentry0	INT8U pairingIndex	The pairing index	Set an action mapping entry 0
		INT8U actionDeviceType	The action device type	
		INT8U actionBank	The action bacon	
		INT8U actionCode	The action code	
		INT8U mappingFlags	The mapping flags	
rf4ce-zrc20-action-mapping-client	setentry1	INT8U rfConfig	The RF config	Set an action mapping entry 1
		INT8U rf4ceTxOptions	The RF4CE TX options	
		OCTET_STRING actionData	The action data	
rf4ce-zrc20-action-mapping-client	setentry2	INT8U irConfig	The IR config	Set an action mapping entry 2
		INT8U irVendorId	The IR vendor ID	
		OCTET_STRING irCode	The IR code	
rf4ce-zrc20-action-mapping-client	setentry	(None)		Set an action mapping entry
rf4ce-zrc20-action-mapping-client	printentry	(None)		Print the action mapping entry stored temporarily
rf4ce-zrc20-action-mapping-client	getcount	(None)		Get mappable action count
rf4ce-zrc20-action-mapping-client	getmappableaction	INT8U entryIndex	The entry index	Get mappable action at index
rf4ce-zrc20-action-mapping-server	clearmappableactions	INT8U pairingIndex	The pairing index	Clear mappable actions from the action mapping server corresponding to a pairing index
rf4ce-zrc20-action-mapping-server	setmappableaction	INT8U pairingIndex	The pairing index	Set mappable action from the action mapping server
		INT8U entryIndex	The entry index	
		INT16U actionDeviceType	The action device type	

		INT8U actionBank	The action bank	
		INT8U actionCode	The action code	
rf4ce-zrc20-action-mapping-server	getmappableaction	INT8U pairingIndex	The pairing index	Get mappable action from the action mapping server
		INT16U entryIndex	The entry index	
rf4ce-mso	bind	(None)		Start the MSO binding procedure
rf4ce-mso	validate	(None)		Validate a controller
rf4ce-mso	terminate	(None)		Terminate the validation procedure
rf4ce-mso	abort	BOOLEAN full	TRUE for a full abort or FALSE otherwise	Abort the validation procedure
rf4ce-mso	press	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send User Control Pressed command *
		INT8U rcCommandCode	The HDMI CEC operand [UI Command]	
		OCTET_STRING rcCommandPayload	The additional operands, if any, required by the HDMI CEC command	
		BOOLEAN atomic	TRUE if the user control is atomic or FALSE if it should repeat	
rf4ce-mso	release	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send User Control Released command *
		INT8U rcCommandCode	The HDMI CEC operand [UI Command]	
rf4ce-mso	set	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send Set Attribute Request command

		INT8U attributeId	The attribute id to store	
		INT8U index	The index of the element to store	
		OCTET_STRING value	The value of the attribute element to store	
rf4ce-mso	get	INT8U pairingIndex	The index of the entry in the pairing table to which to send the command	Send Get Attribute Request command
		INT8U attributeId	The attribute id to retrieve	
		INT8U index	The index of the element to retrieve	
		INT8U valueLen	The length of the attribute element to retrieve	

*__Note__: The commands `plugin rf4ce-zrc11 press` and `plugin rf4ce-zrc11 release` are the same as `plugin rf4ce-mso press` and `plugin rf4ce-mso release`, respectively. Furthermore, these pairs of commands are similar to `plugin rf4ce-zrc20 start` and `plugin rf4ce-zrc20 stop`. A user can utilize this fact in multiple ways. For example, if there is a situation where an RF4CE MSO application needs the functionality to send User Control commands, the user does not have to use the extra flash to include a ZRC plugin for the sole purpose of being able to send those commands.

CONTACT INFORMATION

Silicon Laboratories Inc.

400 West Cesar Chavez
Austin, TX 78701
Tel: 1+(512) 416-8500
Fax: 1+(512) 416-9669
Toll Free: 1+(877) 444-3032

Please visit the Silicon Labs Technical Support web page for ZigBee products:
www.silabs.com/zigbee-support and register to submit a technical support request