RemoTI Developer’s Guide

Document Number: SWRU198D

Table of Contents

TABLE OF CONTENTS 2
REFERENCES 4
LIST OF FIGURES 4
LIST OF TABLES 4
ACRONYMS AND DEFINITIONS 5

1. OVERVIEW 7
1.1 INTRODUCTION 7
1.2 RF4CE ARCHITECTURE 7
1.3 RF4CE CERTIFICATION 8
1.4 REMOTI 8
1.4.1 Stack architecture 8
1.4.2 Software API 9

2. TOPOLOGY AND NODE TYPES 11
2.1 LOGICAL NODE TYPES 11
2.2 TARGET 11
2.3 CONTROLLER 11
2.4 TOPOLOGY 11

3. NODE CONFIGURATION 13
3.1 ADDRESSING 13
3.1.1 IEEE address 13
3.1.2 Short address 13
3.2 STACK INITIALIZATION 13
3.2.1 Start modes 13
3.2.2 Target cold startup 13
3.3 TYPICAL NODE STARTUP SEQUENCE USING RTI INTERFACE 14

4. PAIRING 15
4.1 OVERVIEW 15
4.2 ZRC "PUSH BUTTON" PAIRING 15
4.3 USER CONSIDERATIONS 15
4.3.1 Pairing table entry 15
4.3.2 Maximum number of entries 15
4.3.3 Matching configuration 15
4.3.4 Pairing with security 16
4.3.5 Unpair 16

5. DATA TRANSMISSION 17
5.1 OVERVIEW 17
5.2 TRANSMISSION OPTIONS 17
5.2.1 Ack/NoAck 17
5.2.2 Unicast/Broadcast 17
5.2.3 MultiChannel or SingleChannel 17
5.3 COMBINATIONS OF TRANSMISSION OPTIONS 17
5.4 USER CONSIDERATIONS 18
5.4.1 API 18
5.4.2 Standby mode 18
5.5 ZID INTERRUPT PIPE 18

6. POWER SAVINGS 20
6.1 OVERVIEW 20
6.2 RADIO CONTROL 20
6.2.1 Controller 20
6.2.2 Target 20
6.3 MCU POWER CONTROL 21
6.3.1 Usage 21
6.3.2 MCU sleep with RNP 21

7. SECURITY 22
7.1 OVERVIEW 22
7.2 CONFIGURATION OPTIONS 22
7.2.1 Enabling security 22
7.2.2 Key distribution 22
7.2.3 Key 22
7.3 USER CONSIDERATIONS 22
7.3.1 Key compromise 22
7.3.2 Secure pairing range 23
7.3.3 Payload overhead 23
7.3.4 Broadcast packets 23

8. FREQUENCY AGILITY 24
8.1 OVERVIEW 24

8.2 FREQUENCY AGILITY 24
8.3 USAGE 24
9. GENERAL INFORMATION 25
9.1 DOCUMENT HISTORY 25
ADDRESS INFORMATION 25

List of Figures

FIGURE 1: RF4CE PROTOCOL STACK 7
FIGURE 2: REMOTI SOFTWARE ARCHITECTURE 9
FIGURE 3: LOGICAL NODE TYPES 11
FIGURE 4: EXAMPLE RF4CE TOPOLOGIES 12
FIGURE 5: INITIALIZATION 14
FIGURE 6: RF4CE DATA PACKET CAPTURE 17
FIGURE 7: INTERRUPT PIPE TRANSMISSION 19
FIGURE 8: STANDBY MODE RECEIVER DUTY-CYCLE 20

List of Tables

TABLE 1: TRANSMISSION OPTIONS FOR RF4CE 17
TABLE 2: DOCUMENT HISTORY 25

Acronyms and Definitions

Device	A physical object consisting of an IEEE 802.15.4 radio
Node	A device containing RF4CE functionality.
Target	A Node that implements the Target functionality as defined in the RF4CE specification (for example, a TV)
Controller	A Node that implements the Controller functionality as defined in the RF4CE specification (for example, a remote control)
CERC	Consumer Electronics for Remote Controls, the standard application profile defined by the RF4CE consortium
ZRC	New name for CERC profile
ZID	ZigBee Input Device, a profile for more advanced input devices.
GDP	Generic Device Profile, a profile to support compatibility, as well as collect common functionality, between future profiles.
SoC	System-on-chip, an IC consisting of microcontroller as well as the radio transceiver like the CC253x devices.
UART	Universal Asynchronous Receiver/Transmitter
SPI	Serial Peripheral Interface
API	Application Programming Interface
RNP	RemoTI Network Processor. An application configuration of the RemoTI stack that configures the CC253x device as a network processor
OS	Operating system (software)
PHY	Physical layer (layer 1 of a communication protocol stack)
MAC	Medium Access Control (layer 2 of a communication protocol stack)
PAN	Personal Area Network
TV	Television
STB	Set top box
IEEE	Institute of Electrical and Electronics Engineers

References

[R1] ZigBee RF4CE Specification Version 1.01 (ZigBee Alliance document 094945r00ZB)

[R2] ZigBee RF4CE CERC Profile Specification (ZigBee Alliance document 094946r00ZB)

[R3] ZigBee RF4CE ZRC Profile Specification Errata Version 1.00 (ZigBee Alliance document 095275r10ZB), **http://www.zigbee.org/Standards/ZigBeeRemoteControl/download.aspx**

[R4] ZigBee RF4CE ZID Profile Specification (ZigBee Alliance document 105557r18ZB), **http://www.zigbee.org/Standards/ZigBeeInputDevice/download.aspx**

[R5] IEEE Std. 802.15.4-2006, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs).

[R6] CC253x Users Guide, SWRU191, ***http://www.ti.com/lit/SWRU191***

[R7] RemoTI API document, SWRA268, *can be found in the installation folder of RemoTI*

[R8] OSAL API document, SWRA194, *can be found in the installation folder of RemoTI*

[R9] HAL Driver API document, SWRA193, *can be found in the installation folder of RemoTI*

[R10] RemoTI Network Processor Interface Specification, *can be found in the installation folder of RemoTI*

[R11] RemoTI Sample Applications User's Guide, SWRU201, ***http://www.ti.com/lit/SWRU201***

[R12] RemoTI SimpleApp App Note, SWRA286, ***http://www.ti.com/lit/SWRA286***

[R13] RemoTI Power Consumption App Note, SWRA263, ***http://www.ti.com/lit/SWRA263***

[R14] RemoTI Coexistence Testing App Note, SWRA285, **http://www.ti.com/lit/SWRA285**

1. Overview

1.1 Introduction

RF4CE (Radio Frequency for Consumer Electronics) is an industry consortium that was formed to create a new protocol standard for RF-based remote control applications in the consumer electronics segment. The consortium was founded by Sony, Philips, Samsung and Panasonic with Texas Instruments, Freescale and OKI Semiconductor as the contributing members.

The first version of the network specification and application profile was completed in December'08. Later, in March'09, the **RF4CE** consortium merged with the ZigBee Alliance for the purpose of maintaining the specification going forward and managing the certification of platforms and products that conform to the specification. The name of the specification is now **ZigBee RF4CE**. For more information, see http://www.zigbee.org/rf4ce

1.2 RF4CE architecture

The **RF4CE** protocol stack architecture is illustrated in Figure 1. It is based on the IEEE 802.15.4-2006 standard [R4].

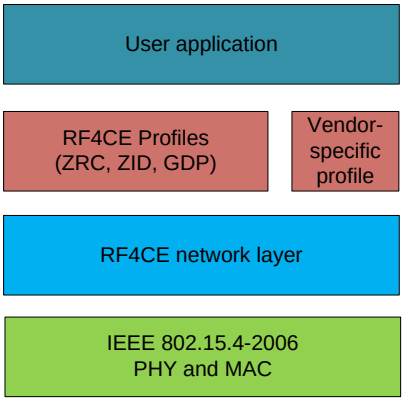

Figure 1: RF4CE protocol stack

The PHY layer is a 250kbps DSSS (direct sequence spread spectrum) radio operating in the 2.4-GHz band. The 2.4-GHz band is an unlicensed ISM band and is available worldwide. This makes it an attractive choice for consumer electronics applications as they are usually targeted for global markets. The spread spectrum modulation scheme provides a robust wireless link that is resistant to fading and interference.

There are 16 channels available at the 2.4-GHz band for 802.15.4 operation. These are numbered 11 through 26 and the center frequency of each channel is $Fc = 2405 + 5\,(k - 11)$ MHz, for $k = 11, 12, \ldots, 26$. The **RF4CE** protocol has selected 3 of those channels (channels 15, 20 and 25) for operation. These channels were selected to minimize interference from WiFi networks.

The MAC layer provides a scheme for transmission and acknowledgement of packets upto 128 bytes in length. It uses a CSMA-CA (carrier sense multiple access with collision avoidance) mechanism for channel access and packet transmission. This allows multiple devices to coexist in the same vicinity and share the same radio spectrum.

On top of that is the network layer that contains the core **RF4CE** functionality. These include

- Node discovery mechanism
- Pairing features to allow simple, foolproof way of connecting appropriate nodes together

- Power savings features to maximize battery life on remote controls and minimize standby power consumption on appliances
- Frequency agility mechanism to dynamically detect and avoid interference
- Security features
- Reliable packet transfer with multiple options and ability to carry standard and vendor-specific profile commands

The application framework contains a standard application profile called **ZRC** that defines most of the commands used in simple remote control applications and is also extensible with vendor-specific commands. It is also possible for users to define their own private profile with fully proprietary commands.

1.3 RF4CE certification

The ZigBee Alliance has put in place a test and certification program for **RF4CE** products. There are two kinds of certifications available – Platform certification and Product certification.

The platform certification is for suppliers of **RF4CE** hardware and software. It includes testing of the network layer and below for compliance to the specification as well as interoperability with other implementations. It is intended so that developers who use or evaluate **RF4CE** platforms from various suppliers can be assured of the technical features supported by the platform and over-air interoperability with platforms from other suppliers.

The product certification is an optional step for manufacturers of end products incorporating **RF4CE** functionality. The purpose of this certification is to ensure that end products made by different vendors will interoperate with each other. It is necessary to start with a certified platform and use a standard application profile in order to achieve this certification. More details on the certification program can be ZigBee website.

1.4 RemoTI

RemoTI is a complete hardware and software development kit from Texas Instruments for developing RF4CE-compliant applications. It is based on the CC253x devices, true SoC (System-On-Chip) solutions for 2.4-GHz 802.15.4 applications [R5]. The CC253x devices combine an 802.15.4 RF transceiver, microcontroller, upto 256-KB of in-system programmable flash memory, up to 8-KB of RAM and a full range of peripherals.

The RemoTI software package includes an **RF4CE**-compliant protocol stack, the **ZRC** profile, sample application code and PC tools.

The RemoTI hardware and software package is undergoing final tests as an *RF4CE Certified Platform*. It is also a Golden Unit for compliance testing and certification, which means that other implementations of the **RF4CE** stack have to be tested against it before being issued compliance certification.

1.4.1 Stack architecture

The RemoTI software architecture is illustrated in Figure 2. It consists of the following software components.

1.4.1.1 OSAL

This is a simple operation system environment for the SoC. It includes features for task management, message passing, queuing, memory management, timers etc. This component is included as source code. The API and additional details are available in the *OSAL API* document [R7].

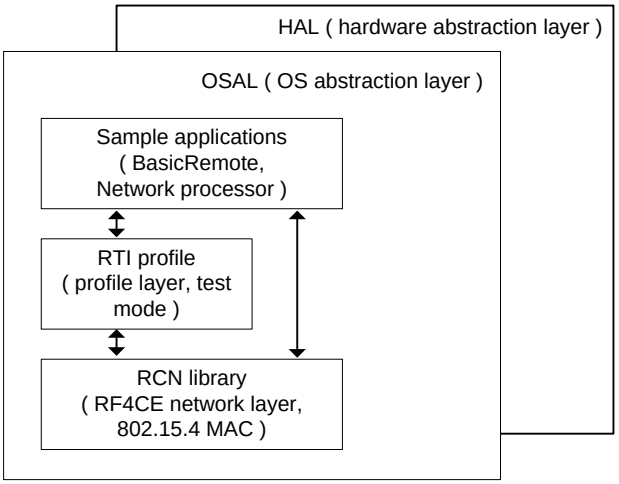

Figure 2: RemoTI software architecture

1.4.1.2 Hardware abstraction layer

This component provides an abstract interface to the hardware available on chip and on the board. It includes firmware for the UART and SPI communication interfaces, Keypad on the remote control, LED's and IR generation. This code is included in source to allow the user to modify to suit the hardware available on their product. The API and additional details are available in the *HAL Driver API* document [R8].

1.4.1.3 RCN library

This is the core RF4CE stack and includes the RF4CE network layer, the IEEE 802.15.4 MAC layer and the radio firmware. This component in included as an object code library. More details including the API are available in the *RemoTI API* document [R6].

1.4.1.4 RTI profile

This is an implementation of the RF4CE profiles; ZRC, GDP and ZID. It includes the pairing mechanism defined in ZRC profile. There are also additional test mode features available. This component in included as source code. More details including the API are available in the *RemoTI API* document [R6].

1.4.1.5 Sample applications

This is sample code that demonstrates how to easily build different applications with the RemoTI stack. The BasicRemote application implements the basic features of a remote control. The NetworkProcessor application implements a target node that communicates with a host processor over UART, SPI or I^2C interface.

Extended features include the over-air-download feature and the serial bootloader feature. The over-air-download feature is available with the BasicRemote application and can be used to upgrade the firmware on the remote control from the Target node. The serial bootloader feature is available with the NetworkProcessor application and can be used to upgrade the firmware on the Target node from the host processor.

More details on the sample applications are available in the *RemoTI Sample Applications User's Guide* document [R10].

1.4.2 Software API

The RemoTI software API is described in detail in [R6]. Note that the API is exposed at both the RCN and RTI layers. The application developer can choose to develop to either of them. The RTI layer includes the **ZRC** application profile as well as the test mode features. So it is easier and quicker for the user to develop a complete solution.

The RCN layer exposes the full functionality of the **RF4CE** network layer specification and thus provides more features and flexibility for the application developer. However, it is best if the

developer has access to the network layer specification [R1] to effectively use all these features. The user would have to implement mechanisms for features like pairing etc. that are included with RTI.

2. Topology and node types

2.1 Logical node types

There are two logical nodes types in **RF4CE** – **Target** and **Controller**. The logical node type determines the role and functionality of the node in the **RF4CE** communication paradigm. Figure 3 illustrates the mapping of the end product functionality to the logical node types.

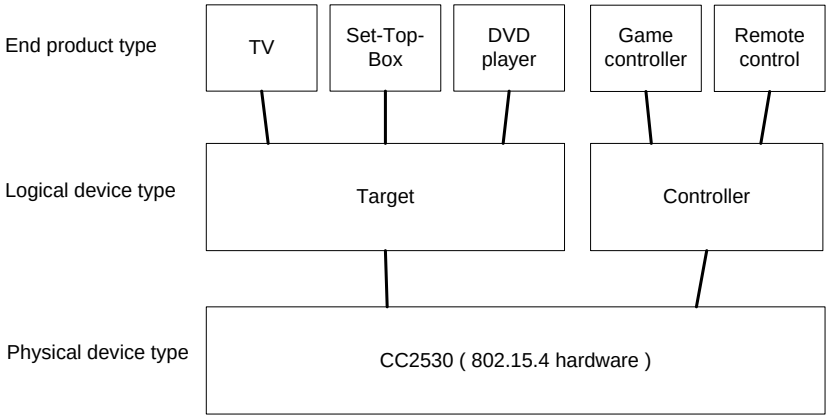

Figure 3: Logical node types

2.2 Target

The **Target** node type is intended to be used in end products like TV, STB etc.

This node type is capable of establishing its own network and can allow other nodes to establish pairing links with it.

A **Target** node implements Frequency Agility function and can change its operating channel as necessary based on local interference conditions.

Since this node is usually the recipient of the application data packets, its radio is placed in the "receive" state when the node is idle (except when it is placed in a special *"standby"* mode where the radio receiver is duty-cycled to conserve power consumption). So, it is typically used in main-powered nodes (like TV, STB etc.).

2.3 Controller

The **Controller** node type is intended for use in applications like remote controls etc.

A **Controller** node will typically establish pairing links with one or more **Target** nodes in its neighborhood and send application data packets.

This node is usually the initiator of transmissions, so its radio may be turned off when in idle state. So this node type is suitable for battery operated nodes.

2.4 Topology

RF4CE is a fully peer-to-peer system with direct communications between nodes. Figure 4 show some possible example topologies. Note that the nodes shown in the topologies should not be considered to be part of a single network. Instead, it is more accurate to consider each node as a free-standing entity with the ability to communicate with other nodes that are within its radio range.

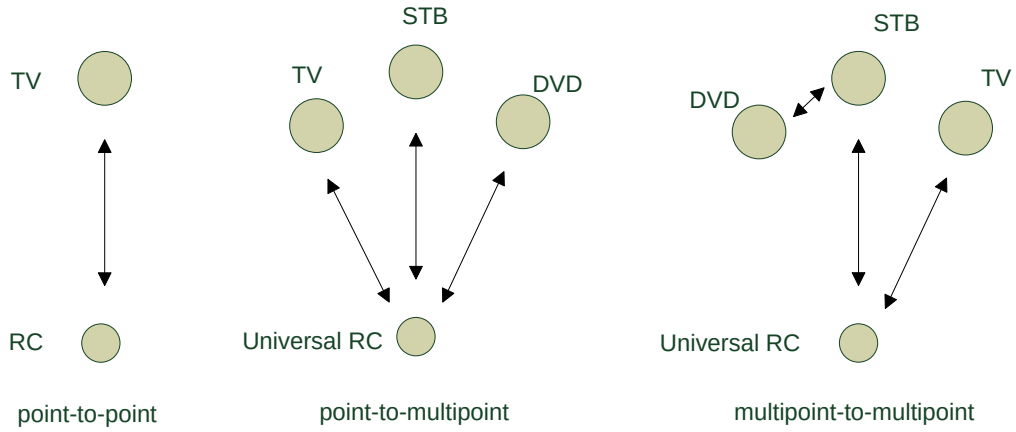

Figure 4: Example RF4CE topologies

3. Node configuration

3.1 Addressing

3.1.1 IEEE address

All 802.15.4 devices must have a 64-bit IEEE address (also called as extended address or 64-bit address). These addresses are globally unique and are assigned and maintained by the IEEE organization and are fixed for the life of the device. More information on these addresses is available at the following website http://standards.ieee.org/regauth/oui/index.shtml

Each CC253x device comes pre-programmed with its own unique 64-bit IEEE address. This is located within the flash information page of the chip. So the user does not have to procure or program the IEEE address.

In some situations, the user may wish to use their own IEEE address in their end products. The RemoTI stack contains provisions for allowing this by reserving a fixed flash location, the 8 bytes starting at offset 0x7E8 on the last flash page, for this purpose (note that this location is just before the flash lock bits, see [R5]). The user can program their own IEEE address at this location in little-endian order. If the user does not wish to use this feature, this location must be filled with the default *all F's* value.

Upon startup, the RemoTI stack will first check the reserved location above for a valid IEEE address. If it finds any value other *all F's*, it will use that as the device IEEE address. If that location is filled with *all F's*, then the stack will use the IEEE address programmed in the CC253x information page.

3.1.2 Short address

When operating in an 802.15.4 network, a device may use a shorter 16-bit address to reduce overhead in packet transmissions. This is called the short address. This address is only unique among devices that are on that network.

In **RF4CE**, the **Target** nodes automatically assign themselves a short address when first initialized. This is in the `ShortAddress` state attribute. The **Controller** nodes are assigned a different short address by each of the **Target** nodes they are paired with. This is stored in the `nwkAddress` field of the pairing entry. The **Controller** uses the appropriate short address when communicating with each **Target**. This is automatically taken care of by the RemoTI stack.

3.2 Stack initialization

3.2.1 Start modes

The **RF4CE** network layer may be initialized in either a "warm" start or a "cold" start mode.

A "warm" startup is when a node was previously operational in an **RF4CE** network and is restarting operation with the same setup. This is usually the case when the node undergoes a reset either accidentally or due to battery change etc. In this case, it is expected that the node will operate exactly as before by restoring pairing tables and network information from non-volatile memory.

A "cold" startup is when a node is starting operation for the first time. In some cases, the node may have been previously started but the application does not want to restore the previous state and instead wants to perform a fresh start. In this case also, a "cold" start may be used. In this case, any saved network layer information will be ignored.

When using the RTI interface, the start mode can be chosen via the `StartupOption` configuration parameter.

3.2.2 Target cold startup

When a **Target** is first initialized, it first does a scan of the channels (channel numbers 15, 20 and 25) for energy levels as well as existing 802.15.4 systems. It selects the channel with the lowest energy level as the operational channel (`CurrentChannel` state attribute).

It then selects a random 16-bit value as the network identifier (PAN ID in 802.15.4 MAC terminology) and also allocates itself a random 16-bit number as the short address. It then starts an 802.15.4 PAN. The **Target** will also allocate short addresses to any node that pairs with it.

3.3 Typical node startup sequence using RTI interface

The application has to perform the following task sequence to start as an **RF4CE** node, see Figure 5. See the SimpleApp application note [R11] for example software to configure and initialize a node.

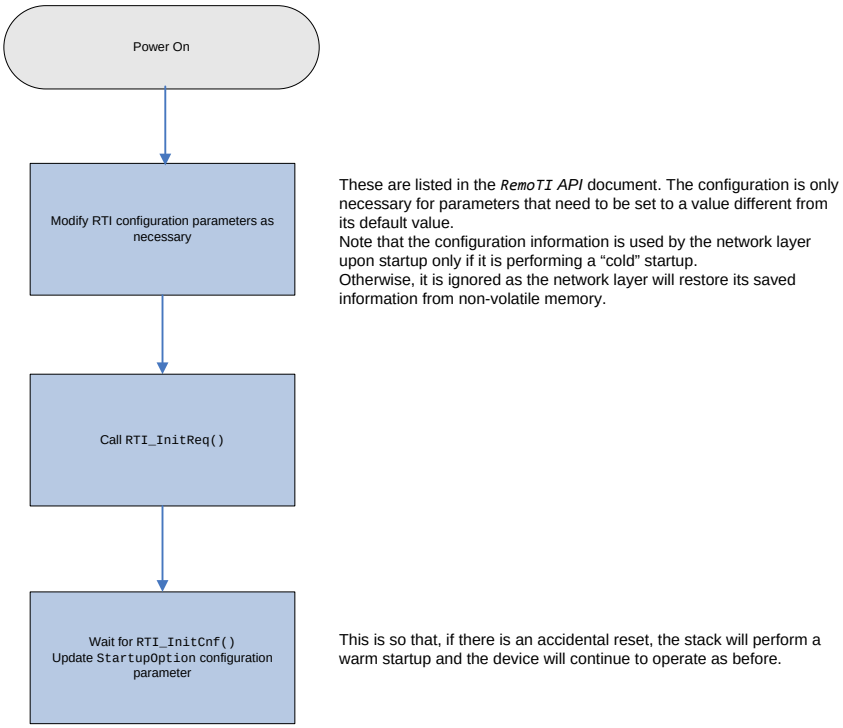

Figure 5: Initialization

4. Pairing

4.1 Overview

A pairing is a logical link between two nodes at the application layer.

It is mandatory to create a pairing link between two nodes before they can exchange application level data packets. Typically, a pairing is created between a **Target** and a **Controller** node. However, it is also possible to also create pairing links between two **Target** nodes.

When a pairing link is created, the application is given a pairing link identifier. The application can then send packets to the destination by referring to the paring link identifier. Note that this identifier has significance only within the node.

The **RemoTI** network layer will store the pairing table entries in non-volatile memory. So when a node is power-cycled, it will be able to restore its previous pairing links along with other network information (assuming it goes through a "warm" startup).

4.2 ZRC "push button" pairing

The **ZRC** application profile specifies the "push-button" mechanism for establishing pairing links. In this scheme, the user has to "push a button" on each of the two nodes in order to establish pairing.

To use this pairing mechanism, the application on the **Target** should issue the `RTI_AllowPairReq()` function call while the application on the **Controller** (or **Target** in case of **Target**-to-**Target** pairing) should issue the `RTI_PairReq()` function. The functions may be invoked on the nodes in any order. However, there is timeout of 30 seconds specified in **ZRC** profile to ensure the procedure terminates in case of any error. Refer to [R11] for example user application software to perform the pairing.

Note that the term "push button" is used figuratively to represent a stimulus to the application to invoke the necessary functions. This stimulus can be via a user action (like a button push) or via some other means. The user application is responsible for detecting the stimulus and invoking the appropriate functions. For example, a **Target** node application may check upon powerup if it is already paired, and if not, automatically invokes the function.

4.3 User considerations

4.3.1 Pairing table entry

The detailed structure of the pairing table entry should be found from the interface definition files in the software package (see `rcn_attribs.h` file). Each pairing entry requires 43 bytes of data memory and non-volatile memory space.

The pairing table entries can be accessed via the `RTI_ReadItem()` and `RTI_WriteItem()` functions. First, the index of the pairing entry to be accessed should be written to the `CurrentPTEntryIndex` item. Then the `CurrentPTEntry` item can be accessed.

4.3.2 Maximum number of entries

The maximum number of pairing links that can be supported by a node is defined by the `MaxPairingTableEntries` constant parameter. This is set to 10 in this software release. Note that it is possible to have more pairing table entires. However that will require different software libraries that can support the extra entries. Please contact lpwsupport@ti.com for this.

4.3.3 Matching configuration

In order to successfully create a pairing link, it is important that the two nodes have complementary applications. This requires that the `SupportedTargetTypes` configuration parameter on the **Controller** and the `DeviceTypeList` configuration parameter on the **Target** must have at least one common device type.

4.3.4 Pairing with security

If security is enabled for the nodes involved in a pairing, the pairing process will have shorter range due to the key distribution protocol (see 7.3.2).

4.3.5 Unpair

In some cases, it may be necessary to unpair an existing link or to delete existing pairing entries.

To remove all pairing entries in a node, a reset followed by a cold startup will be sufficient. To remove a single pairing entry, the entry should be read, modified by setting the `pairingRef` member field to an invalid value of `0xFF` and then written back. Note that these methods of removing the pairing table entries will not remove the entry from the corresponding nodes.

For a clean unpair where the nodes at both ends of the pairing link can remove the entry, both RTI implementation of **ZRC** profile and RCN interface provides an unpair function. Note that the standard **ZRC** profile does not provide an unpairing mechanism.

5. Data transmission

5.1 Overview

All communications in **RF4CE** occur via direct, point-to-point transmissions. There is no relaying or routing capability.

Figure 6 shows a sniffer capture of a typical RF4CE packet with a standard profile command. Note that that header overhead added by the network layer varies depending on whether or not security is used etc. But in all cases, the application will have atleast 90 bytes of payload in each packet.

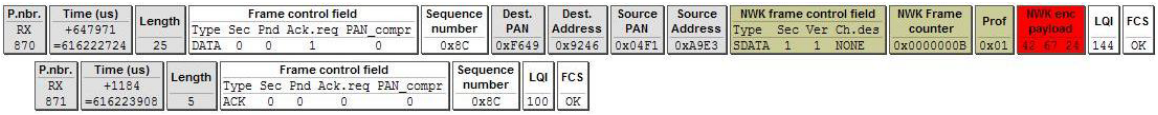

Figure 6: RF4CE data packet capture

The **RF4CE** protocol contains several transmission options that are available to the application when requesting a packet transmission to another paired node. These are described in the following section.

5.2 Transmission options

5.2.1 Ack/NoAck

This option determines whether the source node should request an acknowledgement (802.15.4 MAC-level acknowledgement) from the destination node upon receipt of the original packet.

5.2.2 Unicast/Broadcast

This option determines if the packet is transmitted to a specific destination node (unicast) or if it is broadcast to all nodes that are in radio range of the source node.

5.2.3 MultiChannel or SingleChannel

This option determines if the packet is transmitted on a single channel or on all three **RF4CE** operational channels.

5.3 Combinations of transmission options

The combination of these options is summarized in Table 1.

Table 1: Transmission Options for RF4CE

Options	Usage	*Application*
Unicast/Ack/SC	The packet is transmitted on a single channel (taken from the pairing entry for the destination node). The packet will be re-transmitted on this channel for upto `nwkMaxFirstAttemptFrameRetries` times if acknowledgement is not received from the destination.	*This option can be used to achieve high data throughput if the destination channel is fixed (using application-specific means). The limited retransmissions provide some level of reliability.*
Unicast/Ack/MC	The operation is similar to the Unicast/Ack/SC with the exception that, if no acknowledgement is received, the node will continue to re-transmit the packet on all three channels sequentially	*This is the most reliable option and should be used by default.*

	(e.g., C1-C2-C3-C1-C2…) for upto 1 second (nwkcMaxDutyCycle time) or until an ack is received.	
Unicast/NoAck/SC	The packet is transmitted once on a single channel (taken from the pairing entry for the destination node).	*This option can be used to achieve even higher data throughputs than the Unicast/Ack/SC mode. The reliability is lower due to lack of retransmissions. The destination channel should be known to the sender for this option to be useful.*
Unicast/NoAck/MC	The packet is transmitted once on each of the three **RF4CE** channels.	*This is intended for application-specific uses.*
Broadcast/Ack/MC	This is an invalid combination of options.	
Broadcast/Ack/ SC	This is an invalid combination of options.	
Broadcast/NoAck/ SC	The packet is broadcast once on a single channel (set to nwkBaseChannel).	*This is used to reach all nodes within radio range on a specific channel.*
Broadcast/NoAck/ MC	*The packet is broadcast once on each of the three **RF4CE** channels (starting with nwkBaseChannel).*	*This is used to reach all nodes within radio range.*

5.4 User considerations

5.4.1 API

To transmit a data packet, use the RTI_SendDataReq() function. The transmission options for each packet transmission are selected via the txOptions parameter in this function. The application must wait for the completion of the packet transmission (i.e., until the RTI_SendDataCnf() callback is issued by the stack) before issuing another request.

Note that a pairing must have been previously established between the two nodes before the applications can exchange data.

5.4.2 Standby mode

When a **Target** node is placed in **Standby** mode, its radio receiver is being duty-cycled (see 6.2.2 for more details). Consequently, it can only reliably receive packets that are transmitted with the Unicast/Ack/MC transmission option. Hence, the other transmission options should only be used if the sending node is aware that the destination is in active mode.

5.5 ZID Interrupt Pipe

ZID defines its own mode of transmission which is an abstraction of a combination of the txOptions given in 5.2. When a packet is sent in the interrupt pipe, the ZID co-layer in RTI will first send it as a Unicast/Ack/MC message. Then within a window of aplIntPipeUnsafeTxWindowTime the consequtive messages are sent Unicast/NoAck/SC. After the expiration of aplIntPipeUnsafeTxWindowTime the next message is sent

Unicast/Ack/MC before `aplIntPipeUnsafeTxWindowTime` is started again. Please see example in Figure 7.

Figure 7 is extracted from the ZID profile specification .

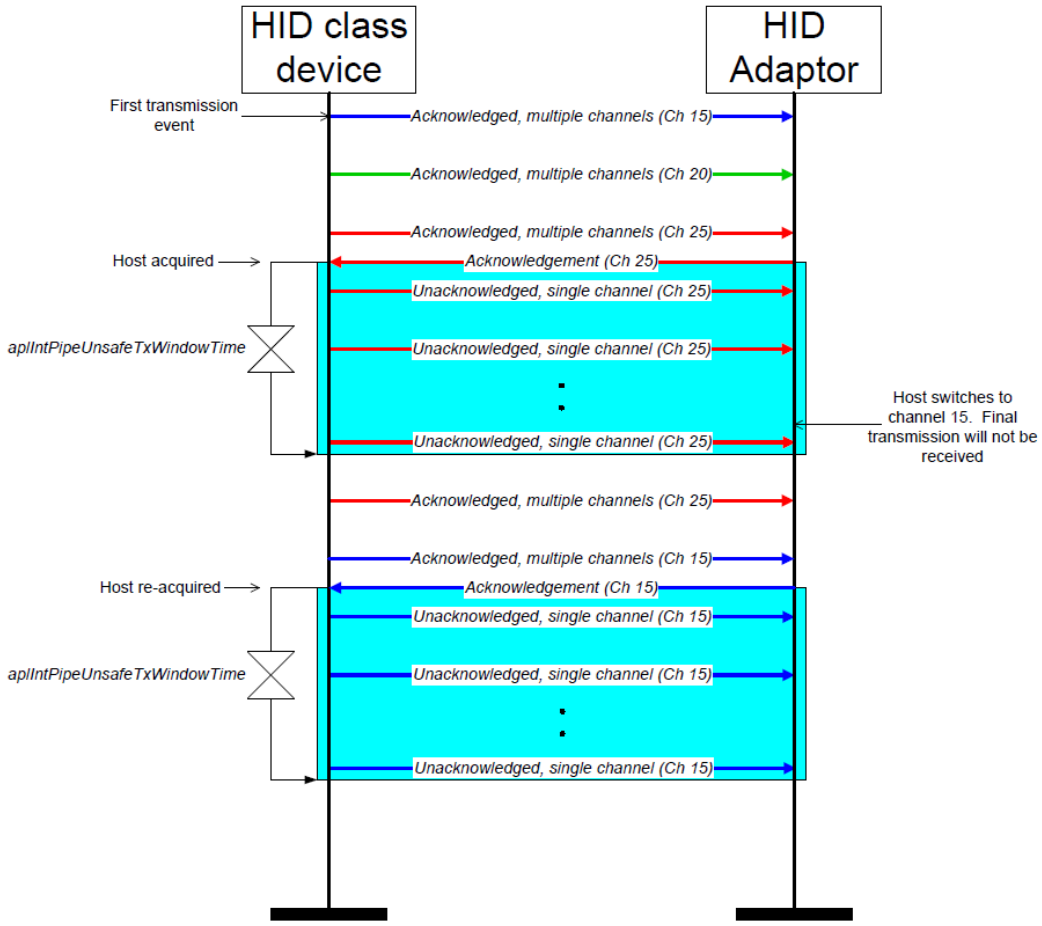

Figure 7: Interrupt pipe transmission

6. Power savings

6.1 Overview

RemoTI has automatic power savings feature for both **Target** and **Controller** nodes. The power savings include control of the radio as well as the MCU.

For detailed measurements of the actual power consumption in various configurations as well as oscilloscope captures, please refer to [R12].

6.2 Radio control

6.2.1 Controller

A **Controller** node is normally the initiator of a packet transmission. So it has the radio turned off by default. When a packet transmission is initiated, the stack will automatically turn on the radio for the duration of the transmission as well as the reception of any acknowledgement, if necessary.

If the application wishes to keep the radio receiver turned on during other times, it has to explicitly control it via the `RTI_RxEnableReq()` function. This is usually used when the application transmits a packet and is expecting a response from the destination node (for example, in two-way remote applications).

6.2.2 Target

A **Target** node is normally the recipient of packets. So the radio receiver is turned on by default. However, to conserve power during long periods of inactivity, it is possible to put the node in **Standby** mode by invoking the `RTI_StandbyReq()` function.

In this mode, the radio receiver is automatically duty-cycled to reduce power consumption. The stack will keep the receiver turned on for a brief time window periodically. If a packet is detected during that time, the receiver will be kept turned on until the whole packet is received and passed up to the application layer. The node will continue to remain in **Standby** mode. The application has to explicitly bring the node out of **Standby** mode, if it chooses to, based on the contents of the received packet. The duty cycling of the **Standby** mode is shown in Figure 8.

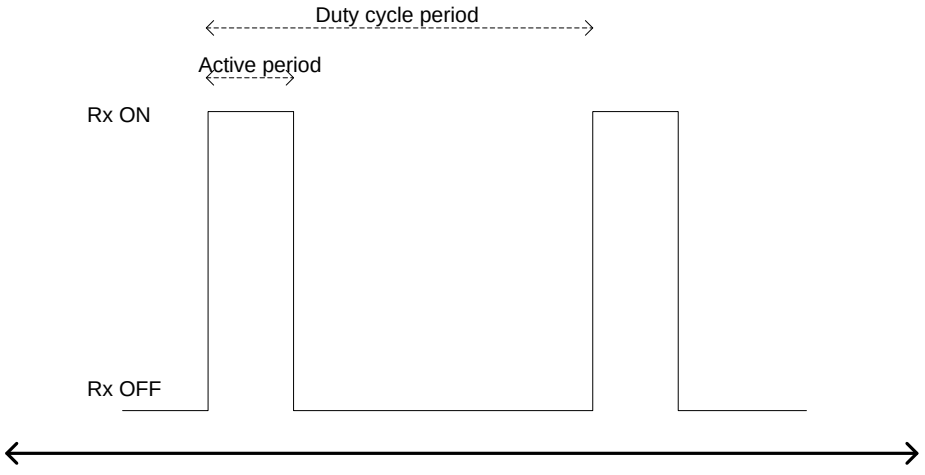

Figure 8: StandBy mode receiver duty-cycle

The **StandBy** mode parameters can be configured through the `DefaultStandbyActivePeriod` and the `DefaultStandbyDutyCycle` configuration parameters. Note that these are only valid for a Target node type. The **RF4CE** specification

requires that the active period is atleast 16 milliseconds and that the duty cycle period is atmost 1 second.

When the **`Target`** is in this mode, any node that wishes to send data to it has to use the default transmission options (Unicast/Ack/MC). This will cause the packet to be retransmitted continuously on all channels and one of the transmissions will overlap with the receiver active period within the duty cycle period.

If **`Target`** power savings is not important, the `DefaultStandbyDutyCycle` parameter value can be reduced. This will improve the responsiveness (reduced packet latency) of the **`Target`** node when it is in **`StandBy`** mode.

6.3 MCU power control

6.3.1 Usage

The RemoTI stack OS has ability to put the CC253x processor in low power mode automatically based on software activity. The MCU sleep mode is controlled by the application via the `RTI_EnableSleepReq()` and `RTI_DisableSleepReq()`functions.

If MCU sleep is enabled, the processor will automatically be put into a low power mode if there are no software events to process. If there is an event(s) scheduled for a future time, the processor will be put in a low power mode with the sleep timer running for the necessary time and then reactivated at the appropriate time. If no future events are scheduled, the processor will be put into a low power mode without the sleep timer operating. In this case, it can be reactivated through an external interrupt (like a button push, host processor control signal etc.).

For details on the CC253x devices' processor and its low power modes and sleep timer, refer to [R5].

6.3.2 MCU sleep with RNP

When using the RNP (RemoTI Network Processor), the MCU power management mechanisms for the host processor and the RNP have to be designed carefully so that communication between the two processors is not lost.

The details of the RNP interface with the host processor are available in [R9].

6.3.2.1 SPI interface

When using the RNP configured with SPI transport, the MCU sleep mode can be permanently enabled. This is because the SPI transport interface is designed to automatically wakeup the RNP (via the MRDY signal) when the host wishes to communicate with it. It also provides for a wakeup signal (SRDY) from the RNP to the host so that the host processor can also enter low power mode and still receive asynchronous communications from the RNP.

6.3.2.2 UART interface

If using the RNP configured with UART transport, enabling the MCU sleep mode will disable the UART receiver on the RNP.

If the host processor wants to communicate with an RNP that has MCU sleep enabled, it should first transmit a NULL character (0x00) and then wait to receive a NULL character in return. This exchange will bring the RNP processor out of any low power mode and also disable the MCU sleep mode. After this, regular UART communications can take place with the RNP. The host processor must re-enable the MCU sleep mode if it wishes the RNP processor to enter low power mode at a later point.

Note that the RNP assumes that the host processor is always ready to receive to UART communications. Some host processors (like the MSP430 processor family) can automatically come out of low power mode when detecting activity on its UART receiver.

6.3.2.3 I^2C interface

When using the RNP configured with I^2C transport, the MCU sleep mode can be permanently enabled. This is because the I^2C transport interface is designed to automatically wakeup the RNP (via the MRDY signal) when the host wishes to communicate with it. It also provides for a wakeup signal (SRDY) from the RNP to the host so that the host processor can also enter low power mode and still receive asynchronous communications from the RNP.

7. Security

7.1 Overview

The **RF4CE** protocol stack provides built in security feature. Since RF transmission is omni-directional unlike IR and can have a larger transmission range, it is possible for a person to exercise RF control from a longer range. This is usually a useful feature; however it can also be misused. To prevent such a scenario, it is recommended that all RF remote control applications should use security.

RF4CE supports a network layer security mechanism that is used if the applications on both ends of the pairing link choose to use security. If security is desired, a security key is established automatically during the pairing process.

The security mechanism in **RF4CE** is based on the AES-CCM mode of operation. This is a well-known security scheme that is also used in many other wireless protocols.

The security mechanism provides the following features

- Privacy – The application payload is encrypted to prevent eavesdroppers from reading the contents.
- Message integrity – The application payload is protected with a message integrity check to prevent modification of the contents.
- Replay protection – The packets are stamped with a frame counter to prevent the same packet from being recorded and retransmitted at a later time.

The C253x devices have hardware AES and CCM engines that are used to apply security as well as hardware random number generator that is used to generate random keys.

7.2 Configuration options

7.2.1 Enabling security

To setup a security key during the pairing process, use the security capable sub-field of `NodeCapabilities` configuration parameter. Security is setup for a pairing link if both nodes have this option enabled.

Once security is setup, it can be enabled or disabled on a packet by packet basis. When sending a data packet, security is chosen via the `txoption` parameter in the `RTI_SendDataReq()` function. If this option is chosen but no security key is setup, an error will result.

It is recommended that security is always enabled on a node and for each packet transmission.

7.2.2 Key distribution

The security key distribution occurs within the pairing procedure. The **`Target`** node generates the key during the pairing process and distributes it to the **`Controller`** node in a number of packets transmitted using low transmit power.

7.2.3 Key

The actual key that will be used for security is automatically generated by the RemoTI network layer. The CC253x hardware random number generator is used to generate the security material.

7.3 User considerations

7.3.1 Key compromise

Note that the RF4CE security scheme has a *"window of vulnerability"* during the key distribution process. If a malicious user is sniffing the channels at the exact time and location when the pairing link is established, the key can be compromised. This will render all future security useless.

It is assumed that the chances of the above happening are quite small and acceptable. If that is not the case for a given application, a more robust, non-standard key distribution scheme should be used.

7.3.2 Secure pairing range

If security is used, the key distribution packets are transmitted using low transmit power (-22dbm on the CC253x devices). This means that the pairing process will only succeed if the two nodes are closer to each other than usual. The transmit power for the key distribution packets can be modified by the application developer through the `rcnSecKeyTxPwr` parameter in the `rcn_config.c` file.

7.3.3 Payload overhead

If security is enabled, each packet will have an additional security header of about 4 bytes. This will reduce total payload available to the application by the same amount.

7.3.4 Broadcast packets

Security is only available for unicast packets as the keys are setup in a pair-wise basis. Broadcast packets are transmitted without any security. So users should be careful to use broadcast packets only when necessary.

Note that broadcast data frame can only be receive by device paired with the originator device.

Alternately, an application may add application level encryption and decryption for specific broadcast packets using non-standard encryption. A security key to be used within a broadcast group may be exchanged using secure links already setup between the nodes. This key can be used to secure the application payload for broadcast packets.

8. Frequency Agility

8.1 Overview

The 2.4-GHz unlicensed band has a lot of interference sources like WiFi networks, microwave ovens, cordless phone and Bluetooth devices. The **RF4CE** network layer has the frequency agility feature which attempts to ensure that the system is always operating on the best available channel.

For details on the actual performance of the RemoTI system in the presence of interference, refer to the *RemoTI Coexistence Testing application note* [R13]. Note that the performance depends both on the radio performance as well as the protocol stack.

8.2 Frequency Agility

The **Target** nodes are responsible for continuously measuring the local channel conditions and migrate to a new channel if there is 24nterference on the current channel. The actual algorithm for determining when to migrate to a new channel is out-of-scope of the specification and is left upto to each implementation.

The **Controller** node keeps track of the channel for each of the **Target** nodes it is paired with (this information is stored in the pairing table entry). When the application transmits a data packet, the network layer will automatically ensure that the channel is changed appropriately. However, it is possible that the **Target** has moved to a new channel and the information with the **Controller** is no longer accurate. In this case, if the *MultiChannel* txoption (see 5.2.3) is used, the **Controller** will try to transmit the packet on all channels (if the initial transmission was a failure). If it successfully transmits the packet on a different channel, it will update the information accordingly.

8.3 Usage

The Frequency Agility feature is built into the RemoTI network layer and is always operating in **Target** nodes. The basic operation is as follows (the parameters referred to below are in the `rcn_config.c` file)

- The network layer continuously measures the energy level on the current channel every 2 milliseconds.
- Among the last `rcnFAMinNumSamples` (default 32) samples, the algorithm will verify if atleast `rcnFANoisySamplesTh` (default 16) have energy level above `rcnFALQITh` level (default -72dBm)
- If the above condition is met, the channel is determined to be "busy" and the node will move to the next channel. Otherwise, the node will continue to collect new energy level samples and perform the above condition check.

The above algorithm ensures the **Target** node will find the best available channel. However, there is a danger that all channels are "busy" according to the above alogirhtm and the **Target** node will be continuously moving between them. To reduce this problem, there is a second level algorithm that operates as follows

- If the node has migrated through all 3 channels without staying on any single channel for more than `rcnFAShortDurationTh` duration `(default 60 seconds)`, then the node will temporarily suspend the basic algorithm operation for `rcnFASuspendDur` duration `(default 60 seconds)`. During this time, the node will migrate to the channel that had the best conditions (minimum number of noisy samples) in the most recent round of measurements.

9. General Information

9.1 Document History

Table 2: Document History

Revision	Date	Description/Changes
SWRU198A 1.0	2009-04-30	Initial release
SWRU198B 1.1	2009-06-24	Updated unpair section
SWRU198C 1.3	2011-09-30	Updated the SW architecture to support ZID and GDP, as well as I2C interface for RNP. Added information about the ZID interrupt pipe transmission alternative.
SWRU198D 1.3.1	2012-11-05	Update to 1.3.1 release.

Address Information

Texas Instruments Norway AS
Gaustadalléen 21
N-0349 Oslo
NORWAY
Phone: +47 22 95 85 44
Fax: +47 22 95 85 46
Web site: http://www.ti.com/lpw

Texas Instruments Incorporated
Low-Power RF Software Development
9276 Scranton Road, Suite 450
San Diego CA 92121
United States of America
Phone: +1 858 638 4294
Fax: +1 858 638 4202
Web site: http://www.ti.com/lpw

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265