Security on TI IEEE 802.15.4 Compliant RF Devices

By J.Hønsi

Keywords

- *RF*
- *ZigBee*
- *IEEE 802.15.4*
- *AES*
- *CCM**
- *CC2420*
- *CC2430*
- *CC2520*
- *CC2530*
- *MSP430*
- *IAR*

1 Introduction

This application note introduces IEEE 802.15.4 security and gives practical examples using Texas Instruments' family of IEEE 802.15.4 compliant RF chipsets; the CC2420, CC2430, CC2520 and CC2530.

The main focus will be on the CC2520, TI's second-generation IEEE 802.15.4 compliant RF transceiver.

This document will also explain how to implement CCM* mode on the CC2430, CC2530 and CC2420 with the use of auxiliary software.

The application note is accompanied by a software example which implements a secure point to point link between two IEEE 802.15.4 compliant RF nodes, using MAC Data frames.

The example demonstrates CCM* security mode on all three chipsets, using a common RF protocol; BasicRF. This protocol implements a small part of the IEEE 802.15.4 specification, including a subset of the IEEE 802.15.4 security suite.

This document is NOT a general discussion on security and AES.

) The IEEE 802.15.4-2006 specification uses the abbreviation CCM mode (Counter-Counter-Mode 'Star'). The '*' is thus not a footnote in this context.

2 Table of contents

3 Abbreviations

The following abbreviations are used in this document:

ACK	Acknowledge
AES	Advanced Encryption Standard
CCM	Counter-Counter Mode
CBC	Cipher Block Chaining
CBC-MAC	Cipher Block Chaining Message Authentication Code
CRC	Cyclic Redundancy Check
CTR	Counter Mode
DK	Development Kit
ECB	Electronic Code Book
EM	Evaluation Module
EB	Evaluation Board
FCF	Frame Control Field
FCS	Frame Check Sequence
FIFO	First In First Out
IAR EW	IAR Embedded Workbench
IDE	Interactive Development Environment
IV	Initialization Vector
PAN	Personal Area Network
MAC	Medium Access Control (layer)
MIC	Message Integrity Code
MCU	Microcontroller Unit
NONCE	Number used Once
SPI	Serial Peripheral Interface
TI	Texas Instruments

4 IEEE 802.15.4 Security

IEEE 802.15.4 is a standard which specifies the physical layer and medium access control for low-rate wireless personal area networks (LR-WPAN's). The current version of the standard is the 2006 revision on which this application note is based. IEEE 802.15.4 optionally encrypts frames using the AES-128 algorithm, offering data confidentiality (encryption) and message integrity. Sequential freshness is used to reject replayed frames.

4.1 General Overview

The MAC frame format is composed of a header, a payload and a frame check sequence (FCS). The details are shown in Figure 1.

Bytes:1	2	1	0 to 20	0/5/6/10/14	N	2
Len	FCF	SEQ	Address	Auxiliary Security Header	Frame payload	FCS
Header					Payload	

Figure 1. General Frame Format

The maximum length of an IEEE 802.15.4 frame is 127 bytes; therefore, only the 7 least significant bits are used in the length field. The sequence number is incremented for each frame. Addresses can be 16-bit, 64-bit or not included, depending on the setting in the Frame Control Field(FCF). The Frame Control Sequence is a CRC check that is appended to the frame.

Bits: 0-2	3	4	5	6	7-9	10-11	12-13	14-15
Frame Type	Security Enabled	Frame Pending	ACK Req.	Intra PAN	Reserved	Dest Addr Mode	Reserved	Src Addr Mode

Figure 2 . Frame Control Field

- Four frame types are defined in the specification:
  - *Beacon*,
  - *Data*
  - *Command*
  - *Acknowledge.*
- Bit 3 of the FCF enables security.
- Bit 6 is set if the frame is destined for nodes within the Personal Area Network (PAN).
- The three address modes that are defined in the specification are: PAN identifier/ address field not present, 16-bit address and 64-bit address.

4.2 Secure Frames

Frames are identified as being secure when the *security enable* bit of the FCF field is set and the *Security Level* field of the *security control* byte is different from 0 (see Table 1). The frame must contain an auxiliary header field if the *security enable* bit is set.

Bytes: 1	4	0/1/5/9
Security control	Frame counter	Key identifier

Figure 3. Auxiliary Security Header

Please refer to Figure 1 for an idea of the *auxiliary security headers* position in the frame.

The Frame Counter has a length of 4 bytes and is used to ensure the freshness (uniqueness) of each frame. The *Security Contro*l field specifies the *Security Level* (bit 0-2) (ref. Table 1) and the *key identifier mode (bit 3-4).* The *key identifier* sub-field is included only if the *key identifier mode* field is greater than zero.

In the software accompanying this application node, the *key identifier mode* is always 0. For more details on the auxiliary security header, please see [1], section 7.6.2.

4.3 Nonces

Nonce (number used once) is a term to describe a data entity being exchanged to ensure the authenticity of a frame. It is a number which contains information about the sender and adds a random element to protect against replay attacks. The IEEE 802.15.4 nonce is a 13-byte number which contains the 64-bit MAC address of the originator, the security control byte and a 32-bit frame counter.

Bytes: 8	4	1
Source address	Frame counter	Security level

Figure 4. CCM* Nonce

Please refer to Figure 1 for an idea of the *CCM* nonce's* position in the frame.

The **nonce** is implicitly a part of a secure frame, as the source address is contained in the general header and the frame counter/Security Level are contained in the auxiliary security header (see Figure 1). The 64-bit IEEE address is unique and thus ensures that no other node can create the same nonce. The nonce is the main component of the 16 byte initialization vector (**IV**) used for CCM* encryption and decryption. In addition to the nonce, the IV consists of a single-byte **flags field** and a two-byte **sequence counter**. The sequence counter keeps track of 16-byte blocks within a frame. The flags field contains information on whether or not to authenticate and the lengths of the sequence counter and Message Integrity Code (MIC). In CCM* the sequence counter length is fixed and the MIC length can be 0, 4, 8 or 16 (bytes).

Note: the **flags field** and the **sequence counter** are NOT transmitted over the air; they are internal to the sender and receiver, respectively.

4.4 Security Suites

IEEE 802.15.4-2006 specifies seven security suites. All of these can be implemented using the CCM* security operation. The major difference from IEEE 802.15.4-2003 is that the latter depends on both CCM and CBC-MAC security operations to achieve the same objective.

Security level	Security suite name	Description
0	None	No security
1	MIC-32	32 bit authentication
2	MIC-64	64 bit authentication
3	MIC-128	128 bit authentication
4	ENC	Encryption only
5	ENC-MIC-32	Encryption and 32 bit authentication
6	ENC-MIC-64	Encryption and 64 bit authentication
7	ENC-MIC-128	Encryption and 128 bit authentication

Table 1. Security modes in IEEE 802.15.4-2006

The software example referred to in this document uses ENC-MIC-64, Security Level 6. In IEEE-802.15.4-2006, ENC-MIC is used with MAC Command frames, ENC is used for MAC Data frames and MIC is used for Beacon frames.

Note: ACK frames are not authenticated or encrypted.

4.5 Security Modes

This section will go though the different block ciphered security modes integrated into TI's radio devices.

- ECB
- Counter Mode (CTR)
- CBC-MAC
- CCM
- CCM*

These modes combine basic AES-128 encryption with various counter mechanisms to the implement the security suites described in 6.

4.5.1 ECB Mode

ECB encryption is the basic building block of all the security suites supported by IEEE 802.15.4-2006. ECB-encryption on its own works on 128 bits (16 bytes) at a time and is as such unsuited for many applications. In practical terms this means pure AES-128 encryption on 16 byte (128 bit) data blocks.

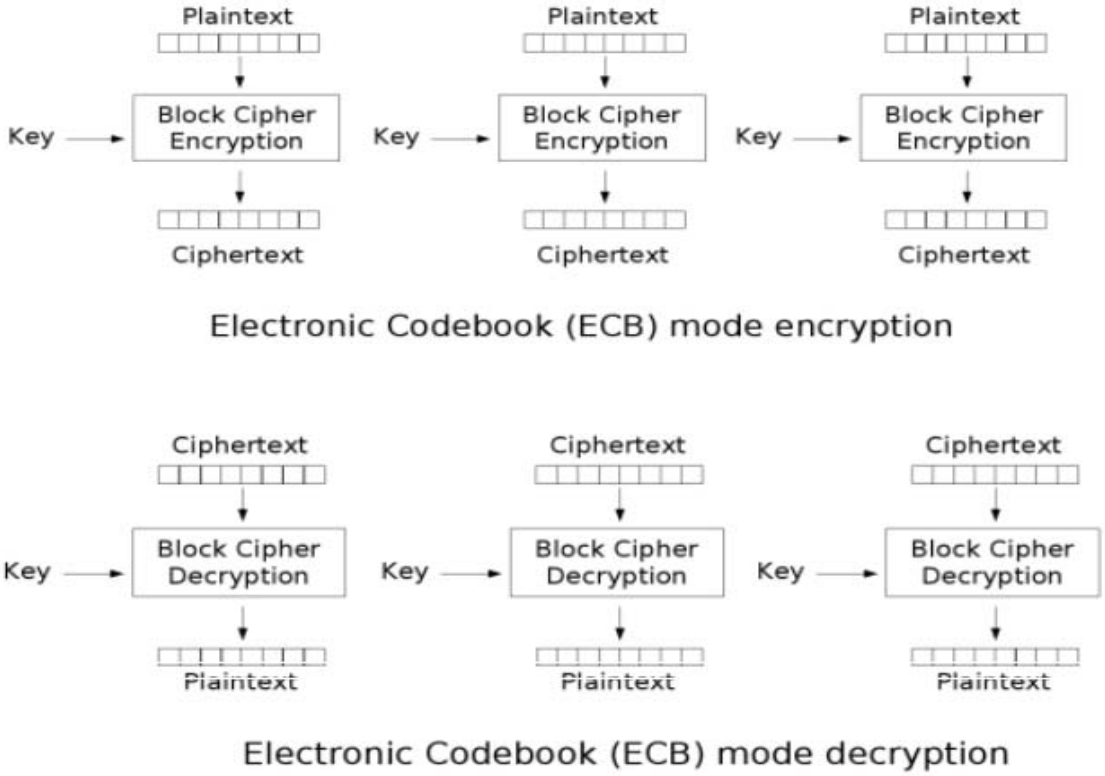

Figure 5. ECB Encryption

ECB encryption is supported on all the four radio chipsets covered by this application note. This makes it possible to use simpler encryption methods in proprietary protocols. The CC2520 even provides separate block XOR instructions to aid ECB mode operations.

The major flaw in pure ECB encryption is that identical blocks of plaintext will produce identical blocks of cipher text making it easy to recognise patterns in the data. This is easily seen in Figure 6.

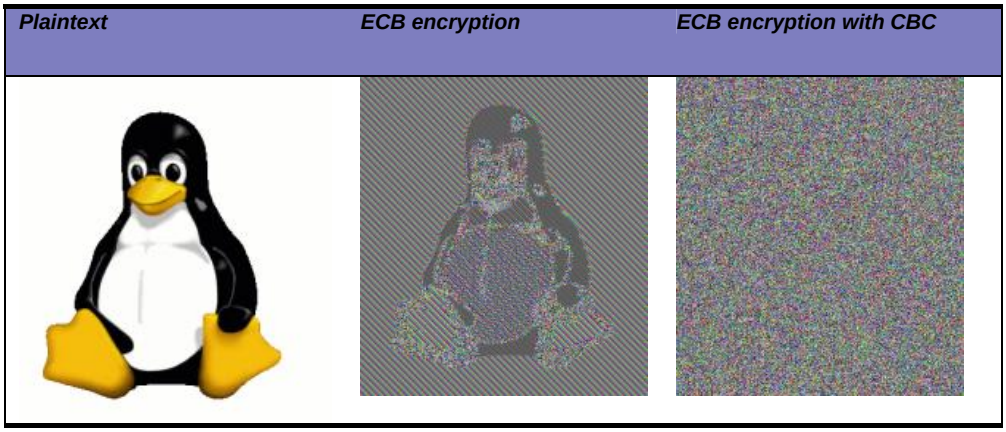

Figure 6. ECB Encryption Example from Wikipedia

This problem is solved by introducing **cipher-block chaining** (CBC) mode where each block of plaintext is XORed with the previous cipher text block before being encrypted. With CBC

mode each cipher text block is dependent on all plaintext blocks processed up to that point. To make each message unique, an initialization vector (IV) must be used in the first block. The first cipher text block is generated by encrypting the **initialization vector** and XORing the plaintext.

Decryption is simple: the plaintext is restored by XORing the cipher text with the encrypted initialization vector. In practical terms this means that the initialization vector must be exchanged across the link together with the encrypted data.

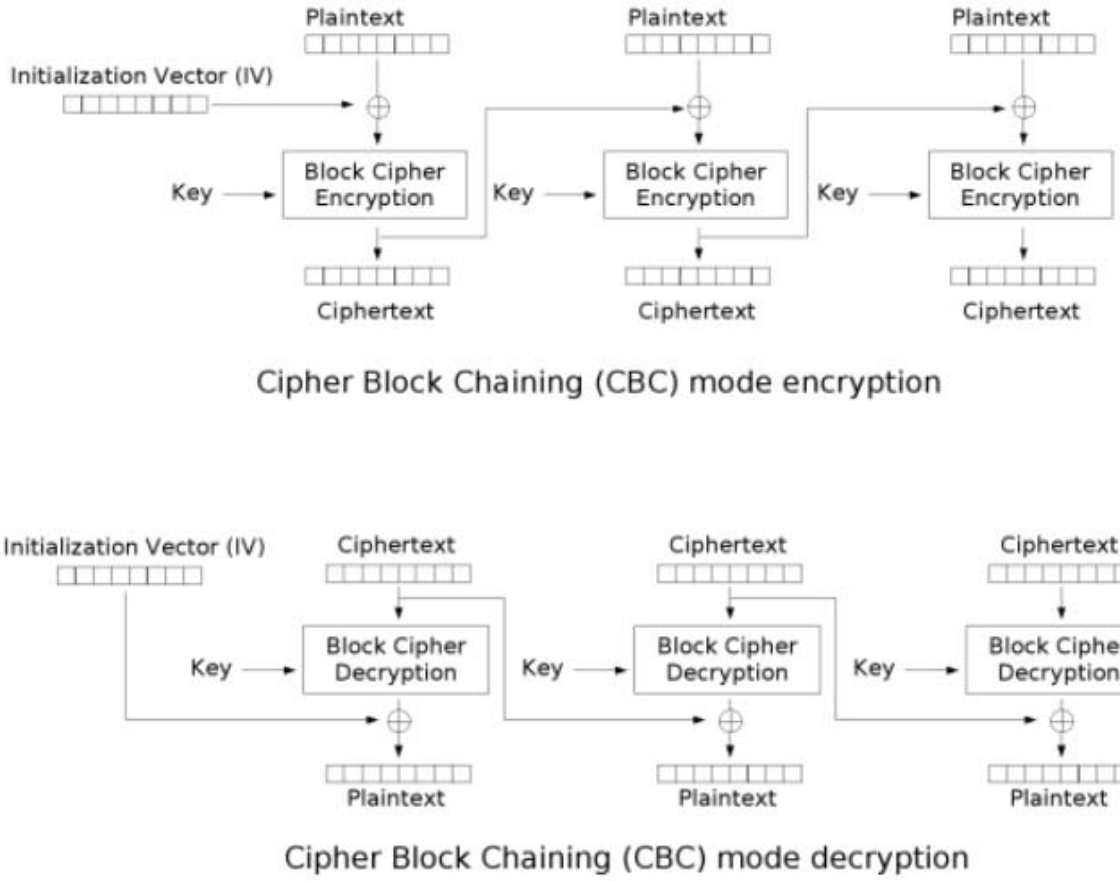

Figure 7. CBC Encryption

4.5.2 Counter Mode (CTR)

Counter Mode is an encryption only mode. In order to ensure freshness a *nonce* that is unique for each frame is introduced. The nonce is known by the sender and receiver. The *nonce* is encrypted using a key known only by the sender and receiver. The encrypted nonce is then added to the plaintext data using XOR. CTR mode works on 16 byte blocks. The last block is padded with zeros if the data to be encrypted is **not** a multiple of 16,

CTR mode on one block is thus identical to ECB encryption of IV + XOR with plaintext. Counter increments ensures sequential freshness.

4.5.3 CBC-MAC Mode

CBC-MAC does not include encryption of the frame, only authentication. The data to be authenticated is divided into blocks of 16 bytes. If the data length is not a multiple of 16 the last block is padded with zeros.

The encrypted Message Integrity Code (MIC) that is the result of the operation has a length of 128 bits (16 bytes) but can be truncated by taking the M leftmost bytes only. Valid numbers for M are 4, 8, or16, depending on whether the programmer chooses to implement the 32, 64, or 128-bit authentication.

Bytes:1	2	1	0 to 20	N	4/8/16	2
Len	FCF	SEQ	Address	Payload	MIC	FCS

Figure 8. CBC-MAC Frame

4.5.4 CCM Mode

CCM is a combination of CTR and CBC-MAC. It both encrypts and authenticates a frame. CCM will first authenticate the frame using CBC-MAC and thereafter encrypt the payload using CTR mode encryption.

The initialization vector (IV) used for encryption consists of the 13-byte *nonce (see* Figure 4*),* combined with the *flags field* and the *sequence counter.*

Bytes:1	13	2
Flags	Nonce	Sequenc e counter

Figure 9. Initialization Vector

The **sequence counter** counts 16 byte blocks within a frame. The one-byte **flags** field is composed as shown in Figure 10.

Bits: 7	6	5..3	2..0
Reserved	Adata	M’	L’

Figure 10. Flags Field

Bit 7 is reserved and should be set to 0. Adata indicates whether there is additional authentication data or not. M’ is the *length of the size* of the authentication field. M’ is encoded as (M-2)/2 and valid values are even numbers from 4 to 16. L’ is the size of the length field (fixed 2) encoded as L-1.

Security level	*Security suite*	*M*	*M'*	*L*	*L'*	*Flags*
0	None	0	0	2	0	0x00
1	MIC-32	4	1	2	1	0x49
2	MIC-64	8	3	2	1	0x59
3	MIC-128	16	7	2	1	0x79
4	ENC	0	0	2	1	0x01
5	ENC-MIC-32	4	1	2	1	0x49
6	ENC-MIC-64	8	3	2	1	0x59
7	ENC-MIC-128	16	7	2	1	0x79

Table 2. Security Level / Flag Fields

4.5.5 CCM* Mode

CCM* is an extension of CCM. It is used in IEEE 802.15.4-2006, is backwards compatible with CCM and adds the option to only encrypt data.

CCM* as described in IEEE 802.15.4-2006 makes it possible to cover all the standard security suites with one operation whereas IEEE 802.15.4-2003 relied on both the CCM and CBC-MAC modes.

4.5.6 Secure Frame Example

The frames described here are from the *light-switch* example accompanying this application note. The frame *) is a MAC Data Frame which is both encrypted and authenticated, with *security level* 6 and *key mode identifier* 0.

*) Strictly speaking this frame format is invalid according to IEEE 802.15.4, but authentication is added in order to visualize how the MIC is generated.

Each frame is identical apart from the sequence number field. The example implements a toggle-switch where each frame toggles the switch *if the single-byte payload is zero*. Figure 11 shows the frames without encryption. The plot is from the TI packet sniffer[7]; the raw frame data have been included to complete the picture. Please note that the byte-order in multi-byte fields is *Little Endian **), whereas the TI Packet Sniffer displays multi-byte values *Big Endian*.

*) Little Endian means that the least significant byte is transmitted first.

Length	Frame control field					Sequence number	Dest. PAN	Dest. Address	Source Address	MAC payload
	Type	Sec	Pnd	Ack.req	PAN_compr					00
12	DATA	0	0	1	1	0x00	0x2007	0xBEEF	0x2520	
0C	61 88					00	07 20	EF BE	20 25	00

Length	Frame control field					Sequence number	Dest. PAN	Dest. Address	Source Address	MAC payload
	Type	Sec	Pnd	Ack.req	PAN_compr					00
12	DATA	0	0	1	1	0x01	0x2007	0xBEEF	0x2520	
0C	61 88					01	07 20	EF BE	20 25	00

Figure 11. Light Switch Toggle, Unsecured Frame

The software example uses fixed **security level 6** (ENC-MIC-64) and fixed **key identifier mode 0** which means that the reception software does not have to investigate the security control field further. Please refer to [1], section 7.6.2.1 for a detailed explanation.

Length	Frame control field								Sequence number	Dest. PAN	Dest. Address	Source Address	Encrypted MAC payload
	Type		Sec		Pnd	Ack.req		PAN_compr					06 00 00 00 00 B0 61
25	DATA		1		0	1		1	0x00	0x2007	0xBEEF	0x2520	A9 16 06 D3 29 53 E9

19 69 88 00 07 20 EF BE 20 25 *06 00 00 00 00* **B0**
61 A9 16 06 D3 29 53 E9

Length	Frame control field								Sequence number	Dest. PAN	Dest. Address	Source Address	Encrypted MAC payload
	Type		Sec		Pnd	Ack.req		PAN_compr					06 01 00 00 00 E3 01
25	DATA		1		0	1		1	0x01	0x2007	0xBEEF	0x2520	EA 8C 60 AB 91 26 DF

19 69 88 01 07 20 EF BE 20 25 *06 01 00 00 00* **E3**
01 EA 8C 60 AB 91 26 DF

Figure 12. Light Switch Toggle, Secured Frame

The first byte of the payload is the *Security Control,* and then follows the *Frame Counter.* The **BOLD RED** field of Figure 12 is the encrypted payload; the rest of the payload shown red is the MIC.

5 Security on TI IEEE 802.15.4 Compliant RF Devices

The IEEE 802.15.4 compliant radios from Texas Instruments are all equipped with an AES-128 security hardware engine. Protocol and application developers may use this to implement secure links in a wireless network without having to implement all the security operations in software. The RF chips interface to the protocol software on varying levels;

- The CC2520 implements CCM* directly through the 'CCM' and 'UCCM' instructions.
- The CC2420 implements CCM but requires a CCM* software add-on using CTR-mode for MIC=0. Counter increments must be done by the MCU.
- The CC2430 and CC2530 RF System-On-Chip includes a powerful AES coprocessor which is accessed over an internal serial interface; either through byte-transfer or DMA. The CC2430 and the CC2530 does not support CCM or CCM* directly. CCM security operations must be implemented in software by combining the CTR and CBC_MAC operations.

	ECB	*CTR*	*CBC-MAC*	*CCM*	*CCM**
CC2420	HW	HW	HW	HW	HW+SW
CC2430	HW	HW	HW	HW+SW	HW+SW
CC2520	HW	HW	HW	HW	HW
CC2530	HW	HW	HW	HW+SW	HW+SW

Table 3. Security Mode Implementations on TI IEEE 802.15.4 Compliant Radios

5.1 CC2520

The CC2520 has comprehensive support for security operations. In addition to the modes listed in Table 3; it has on-chip support for block XOR operations, counter incrementing and key/nonce reversal. The CC2520 provides full flexibility on chip memory usage. Keys, initialisation vectors and buffers may be located anywhere in memory.

The CC2520 has direct support for CCM* through the CCM and UCCM instructions. In addition the INC instruction makes it possible to increment counters *on-chip*, avoiding time and power-consuming SPI-transfers. CCM operations can take place anywhere in memory, the RXBUFMOV and TXBUFCP instructions moves data quickly to/from the FIFOs, leaving the CC2520 free to continue data reception and transmission while security operations take place in parallel. Please refer to [4] for a detailed description of the CC2520.

Security operation	*Instructions*	*Description*
ECB	ECB,ECBO	Electronic Code Book, encryption of 16 byte bocks
Counter mode	CTR,UCTR	Counter mode
CBC-MAC mode	CBCMAC,UCBCMAC	CBC-MAC authentication
CCM	CCM,UCCM	Counter-counter mode
CCM*	CCM,UCCM	
Counter increment	INC	On-chip counter increment
Block reversal	MEMCPR	On-chip block reversal

Table 4. CC2520 Security Operations

5.1.1 Transmission with CCM*

The following steps must be performed once when the link is initialized:

- Generate the key and TX nonce
- Load and reverse the TX nonce/sequence counter (or reverse in software before loading)
- Load and reverse the key (or reverse in software before loading)

The following steps must be performed at each frame transmission:

- Load the frame in the CC2520 RAM with the MEMWR instructions
- Perform the CCM operation on the address the frame was loaded to **+ 1** (excluding the length byte)
- Move the frame to the TX FIFO with the TXBUFCP instruction
- Start transmission
- Increment the counter in the TX nonce

5.1.2 Reception with CCM*

The following steps must be performed once when the link is initialized:

- Generate the key and RX nonce
- Load and reverse the RX nonce (or reverse in software before loading)
- Load and reverse the key (or reverse in software before loading)

The following steps must be performed at each frame reception:

- Wait for the complete frame received
- Copy the frame from the RX FIFO to a work buffer
- Copy frame counter value from received frame into RX nonce
- Copy source address to nonce
- Perform the UCCM operation on the address the frame was loaded to **+ 1** (excluding the length byte)
- Verify the authentication status in the DPUSTAT register

5.1.3 CCM Instruction

The CCM instruction has the following signature

CCM(uint8 **p**, uint8 **k**, uint8 **c**, uint8 **n**, uint16 **a**, uint16 **e**, uint8 **f**, uint8 **m**);

Arguments:

- **p** – priority (0 or 1)
- **k** – key address / 16
- **c** – number of bytes to authenticate and encrypt, typically the frame payload
- **n** – nonce address / 16
- **a** – frame buffer, the whole frame except the length field
- **e** – set to 0 for inline authentication/encryption
- **f** – number of bytes to authenticate, typically the whole frame
- **m** – Message Integrity Code length decoded as follows:
  - 0 : 0 byte MIC
  - 1 : 4 byte MIC (security suites MIC-32 and ENC-MIC-32)
  - 2 : 8 byte MIC (security suites MIC-64 and ENC-MIC-64)
  - 4 : 16 byte MIC (security suites MIC-128 and ENC-MIC-128)

The payload is encrypted inline and the MIC appended to the frame. Care must be taken to add the MIC length to the frame length before transmission starts.

Figure 13. CC2520 CCM Instruction

5.1.4 UCCM Instruction

The UCCM instruction has the following signature

UCCM(uint8 **p**, uint8 **k**, uint8 **c**, uint8 **n**, uint16 **a**, uint16 **e**, uint8 **f**, uint8 **m**);

Arguments:

- **p** – priority (0 or 1)
- **k** – key address / 16
- **c** – number of bytes to authenticate and decrypt, typically the frame payload
- **n** – nonce address / 16
- **a** – frame buffer, the whole frame except the length field
- **e** – set to 0 for inline authentication/encryption
- **f** – number of bytes to authenticate, typically the whole frame
- **m** – Message Integrity Code length decoded as follows:
  - 0 : 0 byte MIC
  - 1 : 4 byte MIC (security suites MIC-32 and ENC-MIC-32)
  - 2 : 8 byte MIC (security suites MIC-64 and ENC-MIC-64)
  - 4 : 16 byte MIC (security suites MIC-128 and ENC-MIC-128)

The MIC code at the end of the frame buffer will be compared to the MIC generated by the authentication of the incoming frame. The result is available in the DPUSTAT registers, bits AUTHSH and AUTSHL for high and low priority instructions respectively.

Figure 14. CC2520 UCCM Instruction

5.1.5 INC Instruction

The INC instruction is tailor-made for incrementing Little-Endian counters on-chip. It can increment integers of 8, 16 and 32 bits.

INC(uint8 **p**, uint8 **k**, uint8 **c**, uint8 **n**, uint16 **a**, uint16 **e**, uint8 **f**, uint8 **m**);

Arguments:

- **p** – priority (0 or 1)
- **c** - counter size (0=8 bits, 1=16 bits, 2= 32 bits)
- **a** - address of least significant byte

Figure 15. INC Instruction

5.2 CC2430 and CC2530

The CC2430 and CC2530 are both a System-On-Chip with an integrated AES co-processor. The security engine is accessed via a fast internal serial interface using either DMA or byte transfer. It requires blocks to be loaded and read out between each operation. CCM is not directly supported, i.e. it has to be implemented in software.

The AES co-processor has the following features:

- ECB, CBC, CFB, OFB, CTR, and CBC- MAC modes
- 128-bits key and IV/NONCE
- DMA transfer trigger capability

The AES co-processor can be accessed either via DMA or by direct byte operations. Design note DN108 [10] gives details on how to use the AES coprocessor both with and without DMA. The software example accompanying this document uses direct byte operations.

The following register makes up the interface to the AES co-processor:

- ENCCS – encryption control and status register
- ENCDI – encryption input data
- ENCDO – encryption output data

Please refer to the corresponding datasheets [3] and [5] for a detailed description of the CC2430 and the CC2530.

NB! The software accompanying this application node implements CCM and CCM* security for the CC2430 and the CC2530

5.3 CC2420

The CC2420 implements the same functionality as the CC2520 (apart from CCM*), but has fixed locations for keys, initialisation vectors and buffers. The programming model is thus

more restrictive as it requires more swapping of data between the MCU and the radio (increased SPI traffic) to implement corresponding functionality.

Security operations are prepared by loading the data into RAM and writing configuration registers. The SAES, STXENC and SRXDEC are used to execute the security operations. The status of the encryption can be monitored by the ENC_BUSY bit of the SPI status byte.

Security operations are performed in the FIFOs on a frame-by-frame basis. Therefore it is important to ensure that no data arrives in the RXFIFO when inline security operations take place. This protection is provided by setting a bit in the ENCCS.

Please refer to [2] for a detailed description of the CC2420.

6 Secure Link Software Example

The software example used in this application note is deliberately chosen to be as simple as possible. It implements a toggle-switch where the switch node sends a 1-byte payload (always 0x00) to the light node.

The application and RF protocol layer is common to all supported MCU and RF platforms.

6.1 BasicRF Protocol Description

The BasicRF protocol is a subset of IEEE 802.15.4, using MAC Data Frames and ACK Frames only. The protocol implements point-to-point links and there is no beaconing or dynamic network formation involved. The protocol employs a fixed PAN ID and short address. The *Security Level* is hard-coded to 6, which gives ENC-MIC-64 according to the IEEE 802.15.4-2006 standard. BasicRF is used in several proprietary link examples for the CC2520DK [6]. It is also used to in AN033 [9]; which describes a CC2420 point-2-point link application which is quite similar to the light switch example used here.

6.2 Software architecture

The software example runs on all RF devices covered by this document and extends the CC2520DK Light Switch example with security features. The firmware is the same for both the *switch* and *light* application, the role is chosen by operating buttons and joysticks on the target board.

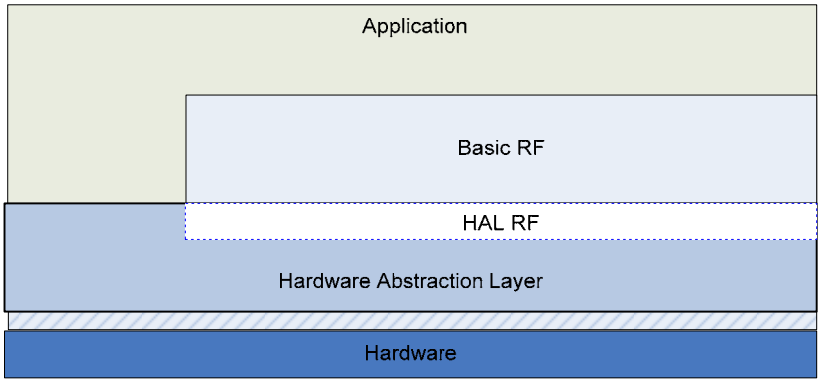

Figure 16 - Software Architecture

6.3 Target Platforms

The example software executes on multiple MCU target platform/radio chipset combinations.

Platform	*CC2520EM*	*CC2430EM*	*CC2530EM*	*CC2420EM*
SMARTRF05/CCMSP2618	YES	-	-	YES
SMARTRF05EB	-	YES	YES	-
SMARTRF04EB	-	YES	YES	-
MSPEXPFG4618	-	-	-	YES

Table 5 - Target Platform/Radio Combinations

6.4 Software Installation

The source code is delivered in a single archive, swra209.zip. When unpacked, the files are placed in the sub-folder 'Light_Switch' as shown in Figure 17.

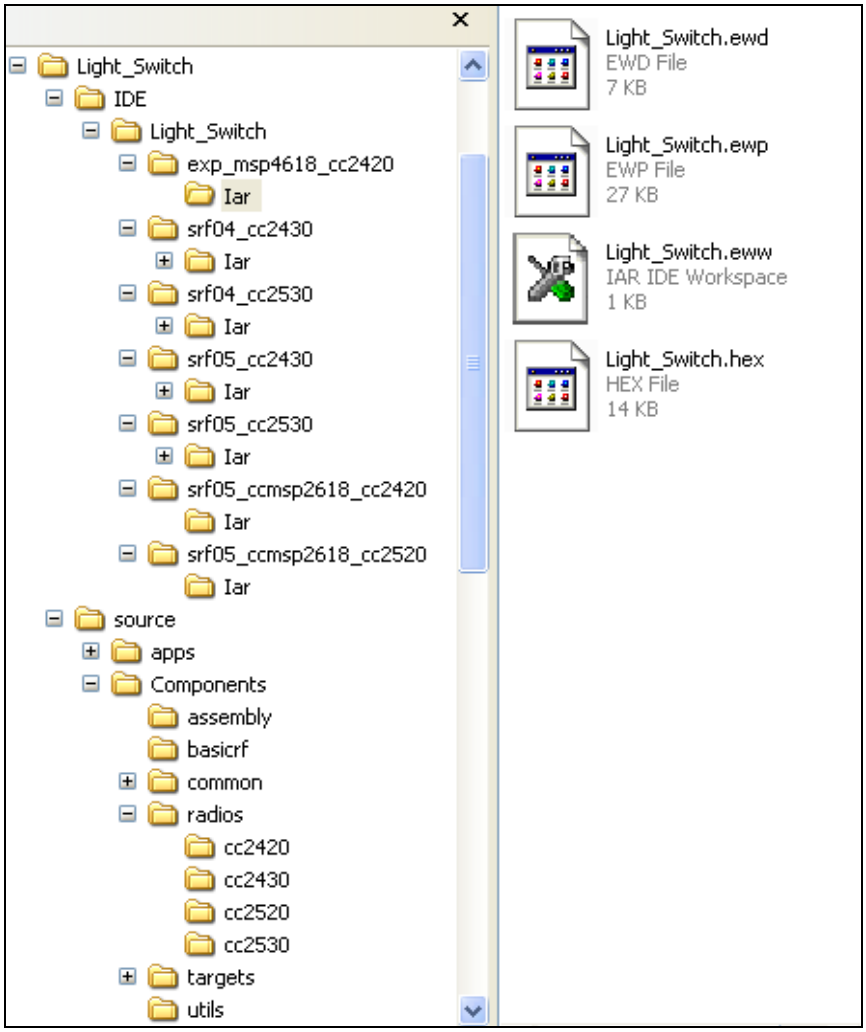

Figure 17. Code Overview

Project files are stored in the IDE sub-folder, which has one sub-folder for each target. Under each target folder there is a folder 'Iar' which contains the workspace file, project file, debugger file and the executable (HEX-file). The source code is best browsed using IAR.

6.5 Software Build

The code is prepared for three MCU architectures: CC2430 (8051), CC2530 (8051) and MSP430FGx618. IAR is the tool used to build executables. The MSP430 target code is compiled using IAR Embedded Workbench, compiler version 4.10A. The CC2430 and CC2530 target code is compiled using IAR Embedded Workbench, compiler version 7.51A.

NB! The code is too large to be used with IAR Kick-Start versions.

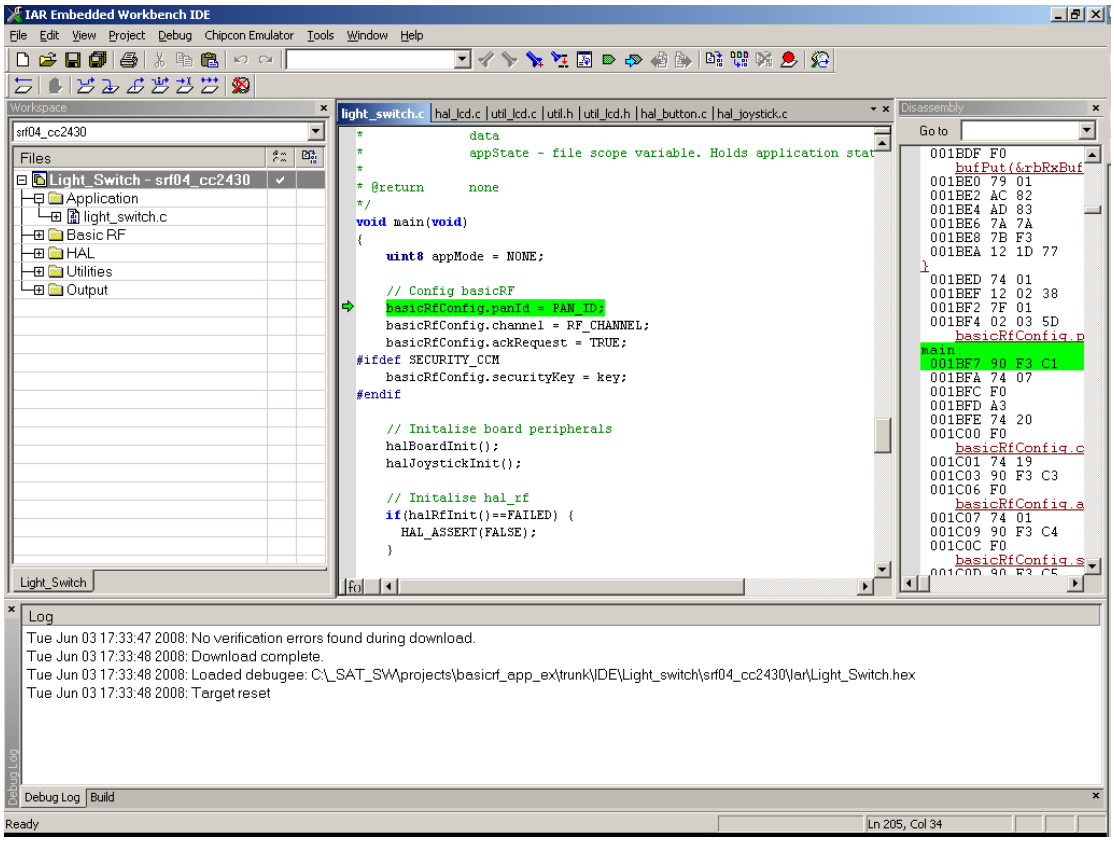

Figure 18. IAR EW after downloading to Target

Project->*Rebuild All* builds the complete image; F7 builds the project (changes only). Then type CTRL-D to download the image to the target. A screen similar to the one shown in Figure 18 should appear. Before this you may get a pop-up about a 'Stack Warning' which can be ignored by clicking on the 'OK' button.

The IAR projects do not by default come with debug information. Unless you need to debug the program you may carry on by pressing the reset button on the target system. You may alternatively start executing by hitting F5 (or '**Debug>Go**' in IAR). If source level debugging is required, please set the Linker>Output option to generate debug information for C-Spy. See Figure 19.

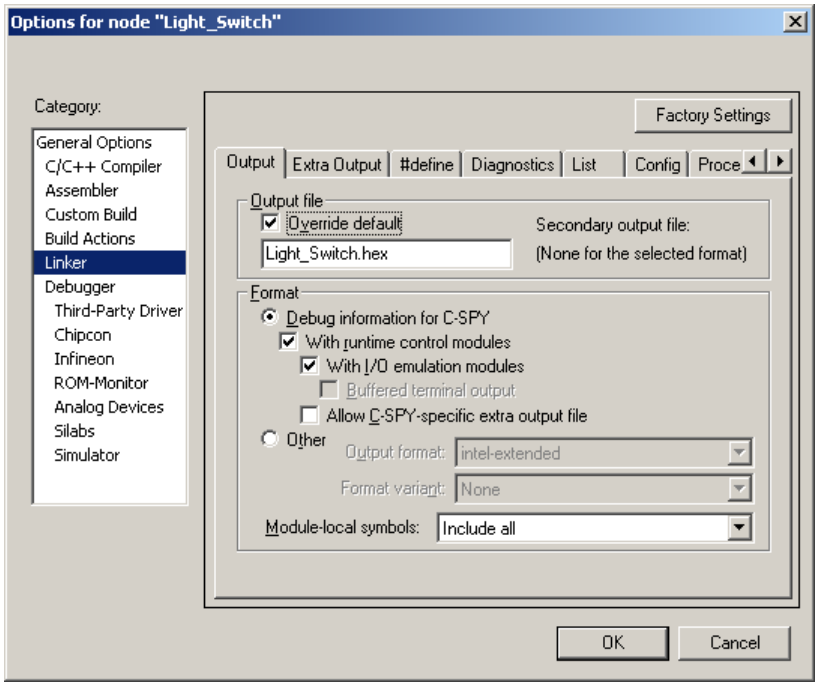

Figure 19. IAR linker with C-Spy output

6.6 Software Execution

Running the example requires 2 of the RF nodes listed in Table 5 programmed with the supplied software.

The example implements a wireless light switch application. One of the nodes is configured as a **light controller**, while the other node is configured as a **light switch**.

Run the example as follows:

- Reset boards.
- Press S1 button to enter the application menu.
- Choose operation. The menu is navigated by pressing the joystick right or left. Select device mode ‘Switch’ on one of the nodes, and ‘Light’ on the other node.
- Press S1 button to confirm the choice.

The light switch application example is now ready. LED D2 on the ‘Light’ node can now be toggled by *pressing* the joystick on the ‘Switch’ node.

The data sent out from the switch device can be observed with a TI Packet Sniffer. The example uses IEEE 802.15.4 channel 25 (2475 MHz).

Note! The **Experimenters Board** (EXP4618) does not have a joystick or an advanced LCD. The item ‘Switch’ will appear as ‘L S’ and ‘Light’ as ‘Light’. The S2 button is used to navigate through the menu. The function of the S1 button is identical to the other platforms. On power-up the display will only show the text ‘CC2420’. When the device mode has been confirmed the symbols **Tx** and **Rx** respectively will appear.

Note! On the SmartRF05EB board, the jumper on P1 pins 17-19 must be removed.

7 CC2420 SmartRF Studio Example

This example shows how to use SmartRF Studio [8] to perform a simple encryption/decryption in CTR, CBCMAC and CCM mode. The examples assume that a sender node has been prepared to send the frames described in this section. This may be done with SmartRF Studio on any CC2420, CC2430 or CC2530 card.

It is important the keys and nonces are loaded *litlle endian* before security operations are executed.

```
09 31 31 31 31 31 31 31 31 00 00 00 01 05 00 01 (Big Endian)
01 00 05 01 00 00 00 31 31 31 31 31 31 31 31 09 (Little Endian)
```

Figure 20. Nonce

```
2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C (Big Endian)
3C 4F CF 09 88 15 F7 AB A6 D2 AE 28 16 15 7E 2B (Little Endian)
```

Figure 21. Key

```
21 49 88 00 22 00 02 00 01 00 05 01 00 00 00 30 30 31 32 33 34 35 36 37 38 39 3A 3B
3C 3D 3E 3F
```

Figure 22. Unsecured Frame

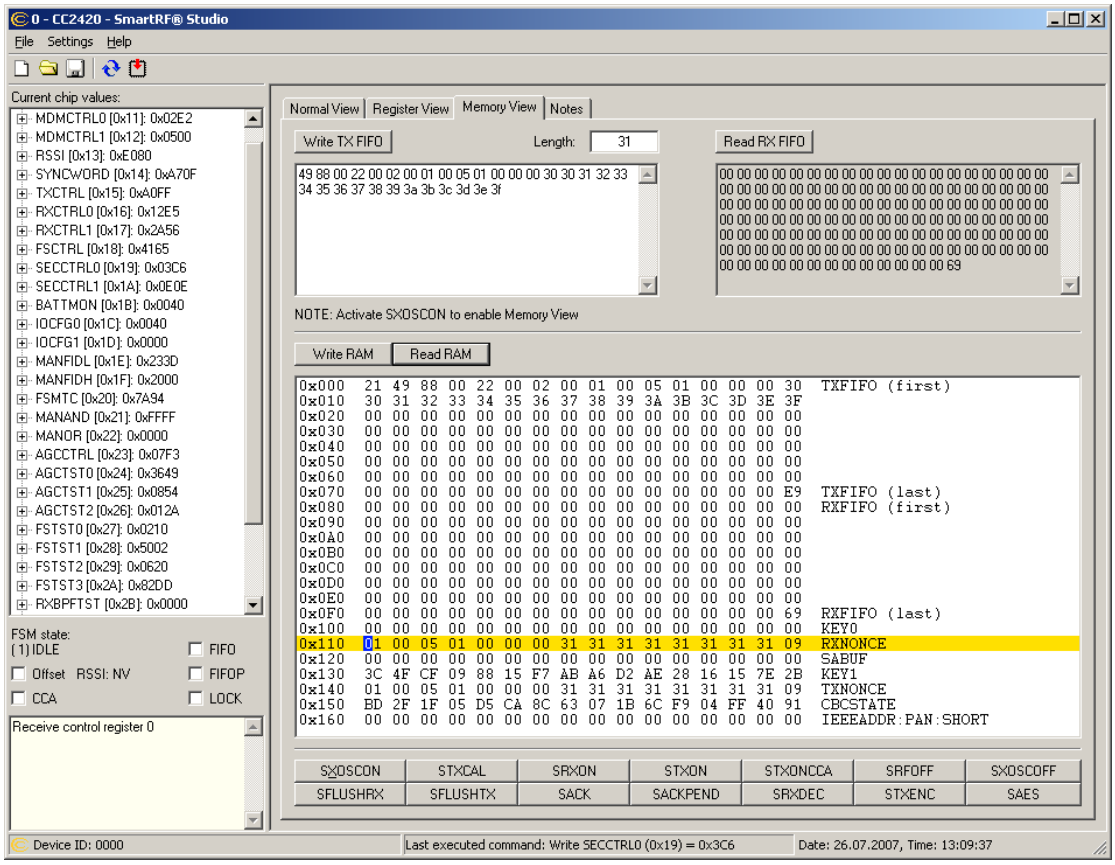

Figure 23. CC2420 RAM after loading of Key and Nonce

7.1 CTR mode

```
21 49 88 00 22 00 02 00 01 00 05 01 00 00 00 94 76 E0 16 74 A9 30 78 59 81 DA BA 7C A9 75
65 BE
```

Figure 24. Secured Frame.

The secured frame must be generated by a sender, using SmartRF Studio or proprietary firmware.

Run the example as follows:

- Set the security control registers as follows:
  SECCTRL0: 0x03C6 (CTR mode, key 1)
  SECCTRL1: 0x0E0E (14 bytes encrypted)
- Load the key into KEY into RAM at address 0x130 (KEY1).
- Load the RX nonce into RAM at address 0x100 (RXNONCE)
- Strobe RXON
- Send the frame from the other radio (assuming the frame has already been prepared with SmartRF Studio)
- Strobe SRXDEC

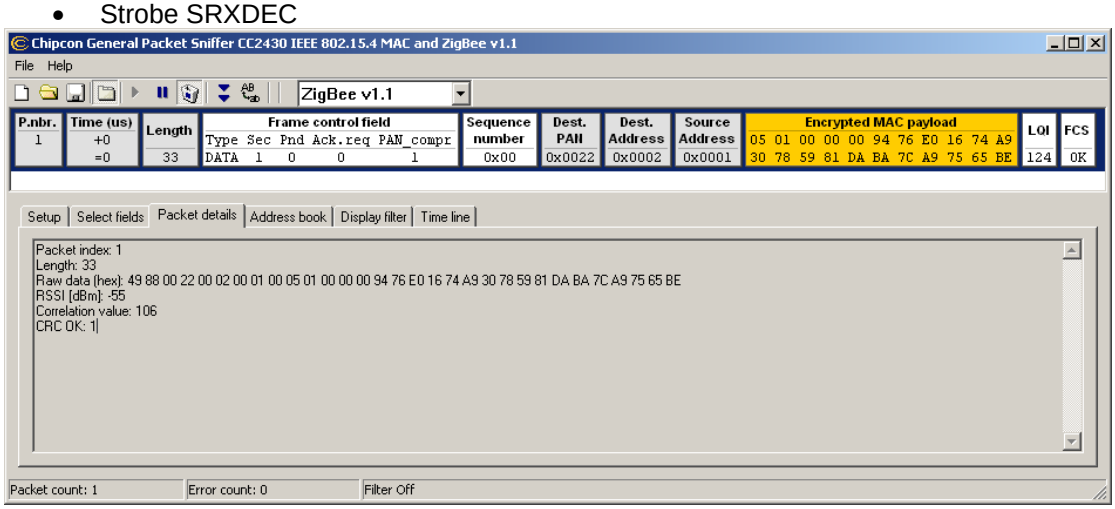

Figure 25. Secured frame captured over the air

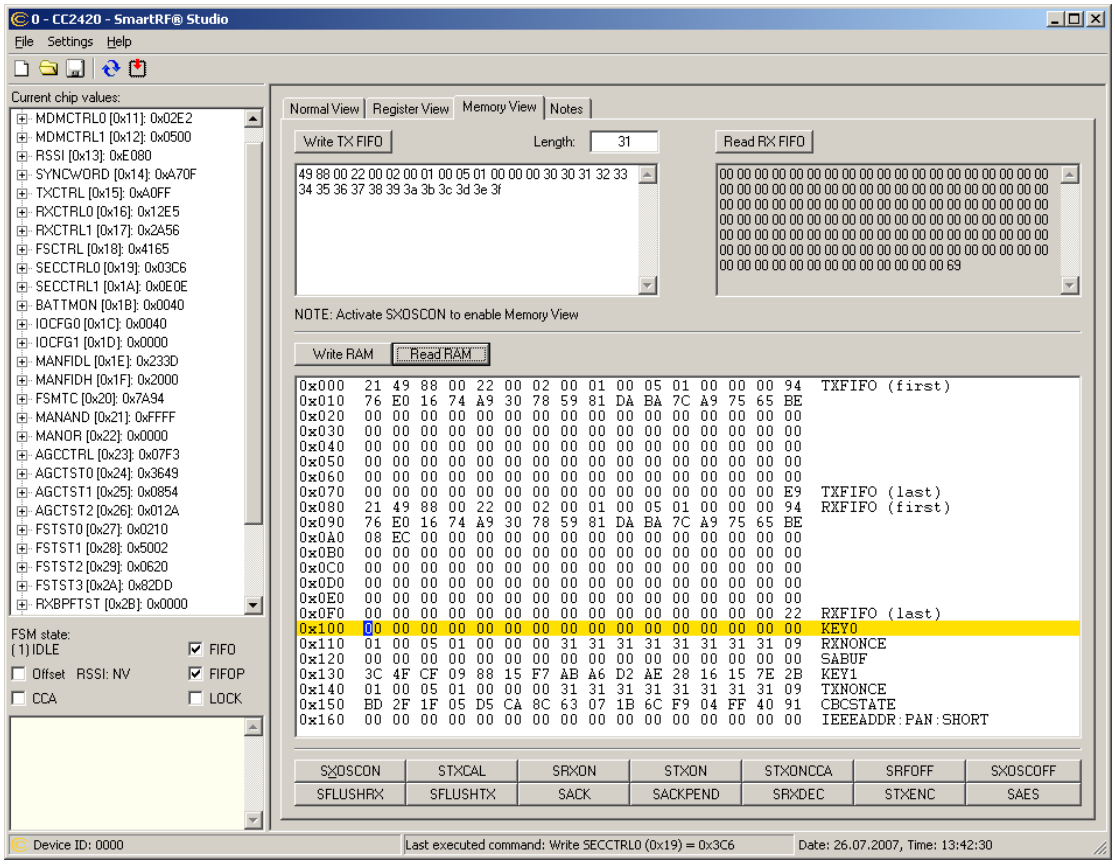

Figure 26. Secured Frame received in RXFIFO

The raw frame has now been received and is stored in the CC2420 RXFIFO. The only remaining operation is to execute the 'SRXDEC' command.

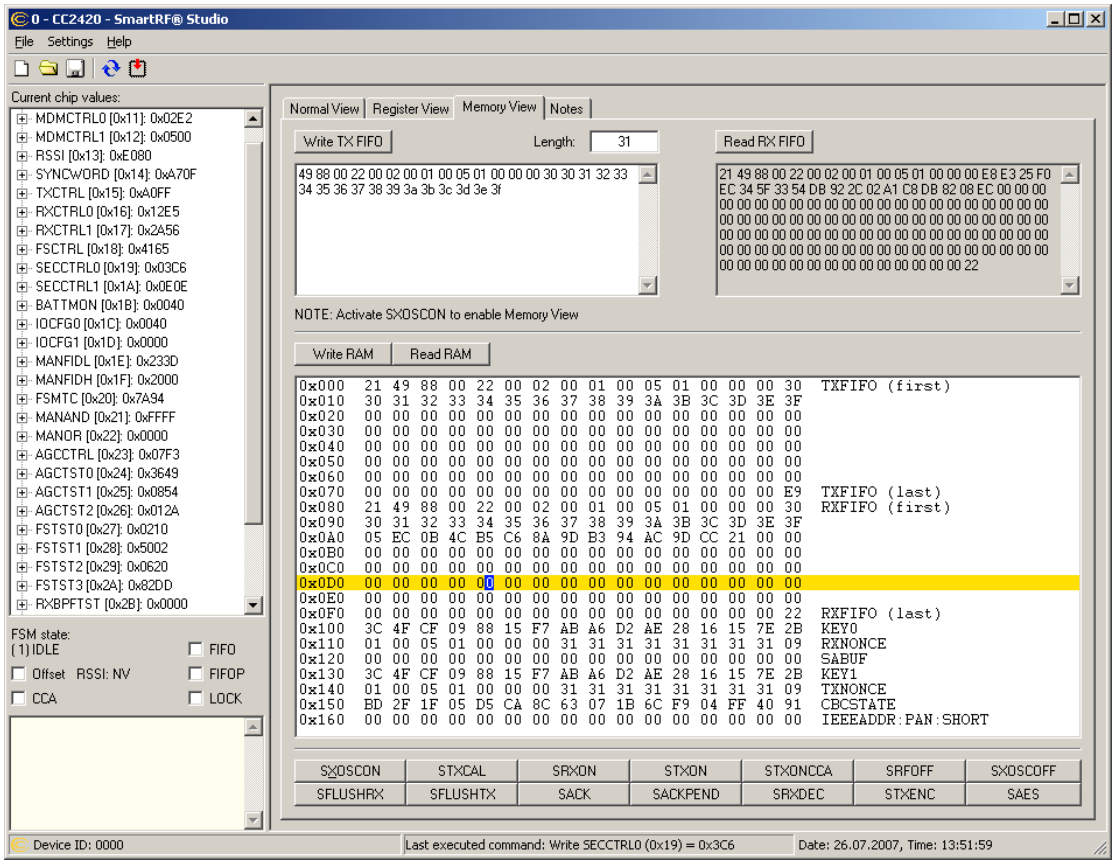

Figure 27. Frame decrypted with SRXDEC

The frame has now been decrypted and the payload can be observed in 0x90 to 0x9F.

7.2 CBC-MAC mode

25 49 88 00 22 00 02 00 01 00 05 01 00 00 00 30 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F C7 43 23 EA

Figure 28. Secured Frame

The secured frame must be generated by a sender, using SmartRF Studio or proprietary firmware.

Run the example as follows:

- Set the security control registers as follows:
  SECCTRL0: 0x03C5 (CBC-MAC mode, key 1)
  SECCTRL1: 0x0000 (encrypt 14 bytes)
- Load the key into KEY into RAM at address 0x130 (KEY1).
- Load the RX nonce into RAM at address 0x100 (RXNONCE)
- Strobe RXON
- Send the frame from the other radio (assuming the frame has already been prepared with SmartRF Studio)
- Strobe SRXDEC

- The last byte of the MIC has been replaced with '00', the authentication has been successful

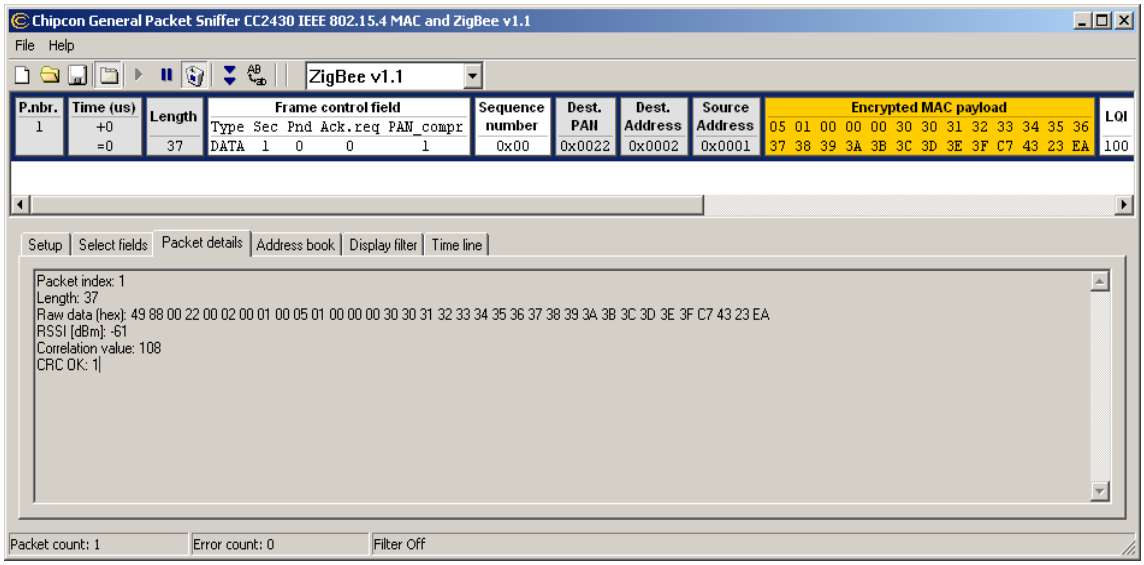

Figure 29. Secured Frame captured over the Air

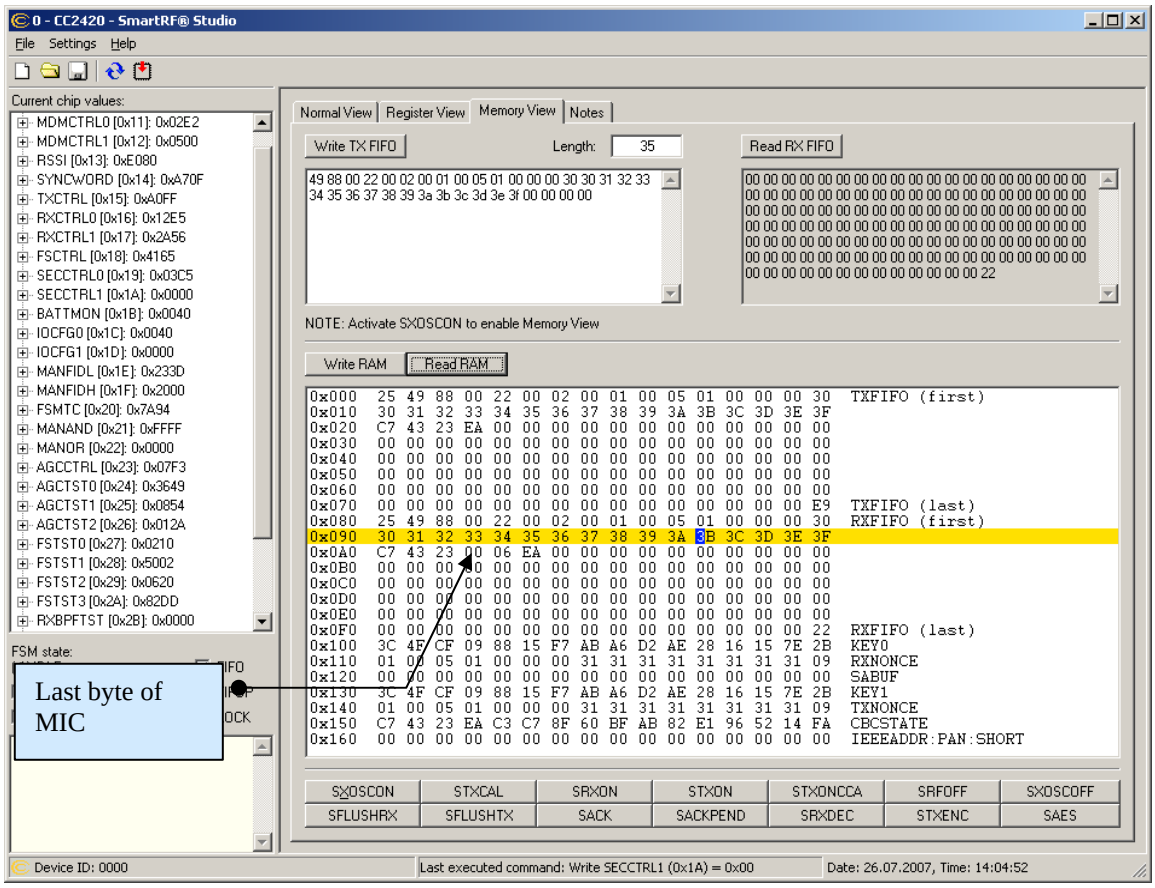

Figure 30. CC2420 RAM after SRXDEC

7.3 CCM mode

```
25 49 88 00 22 00 02 00 01 00 05 01 00 00 00 94 76 E0 16 74 A9 30 78 59 81 DA BA 7C A9 75
65 BE 26 25 B4 A3
```

Figure 31. Secured Frame

The secured frame must be generated by a sender, using SmartRF Studio or proprietary firmware.

Run the example as follows:

- Set the security control registers as follows:
  SECCTRL0: 0x03C7 (CCM mode, key 1)
  SECCTRL1: 0x0E0E (encrypt 14 bytes)
- Load the key into KEY into RAM at address 0x130 (KEY1).
- Load the RX nonce into RAM at address 0x100 (RXNONCE)
- Strobe RXON
- Send the frame from the other radio (assuming the frame has already been prepared with SmartRF Studio)
- Strobe SRXDEC
- The last byte of the MIC has been replaced with '00', the authentication has been successful

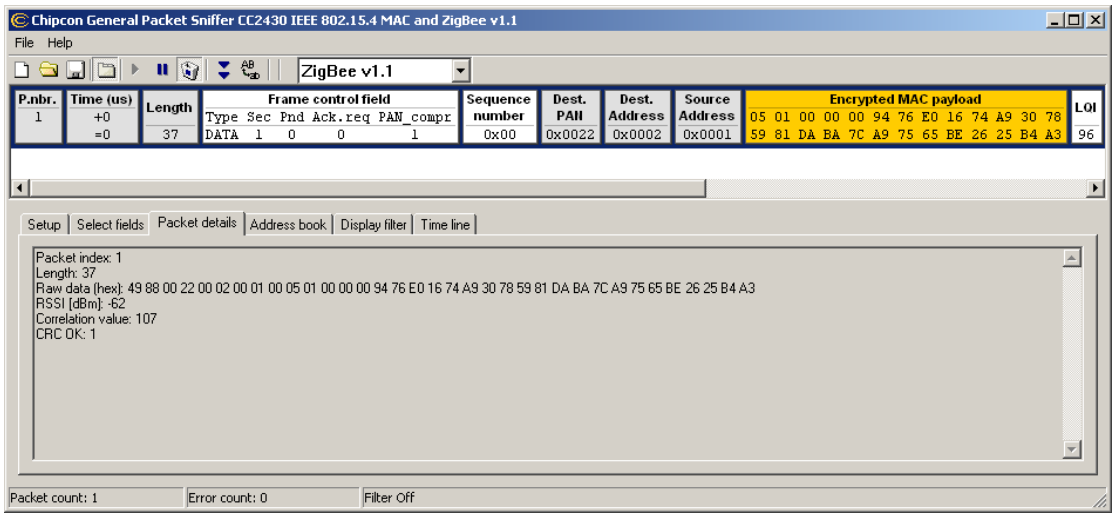

Figure 32. CCM Frame over the Air

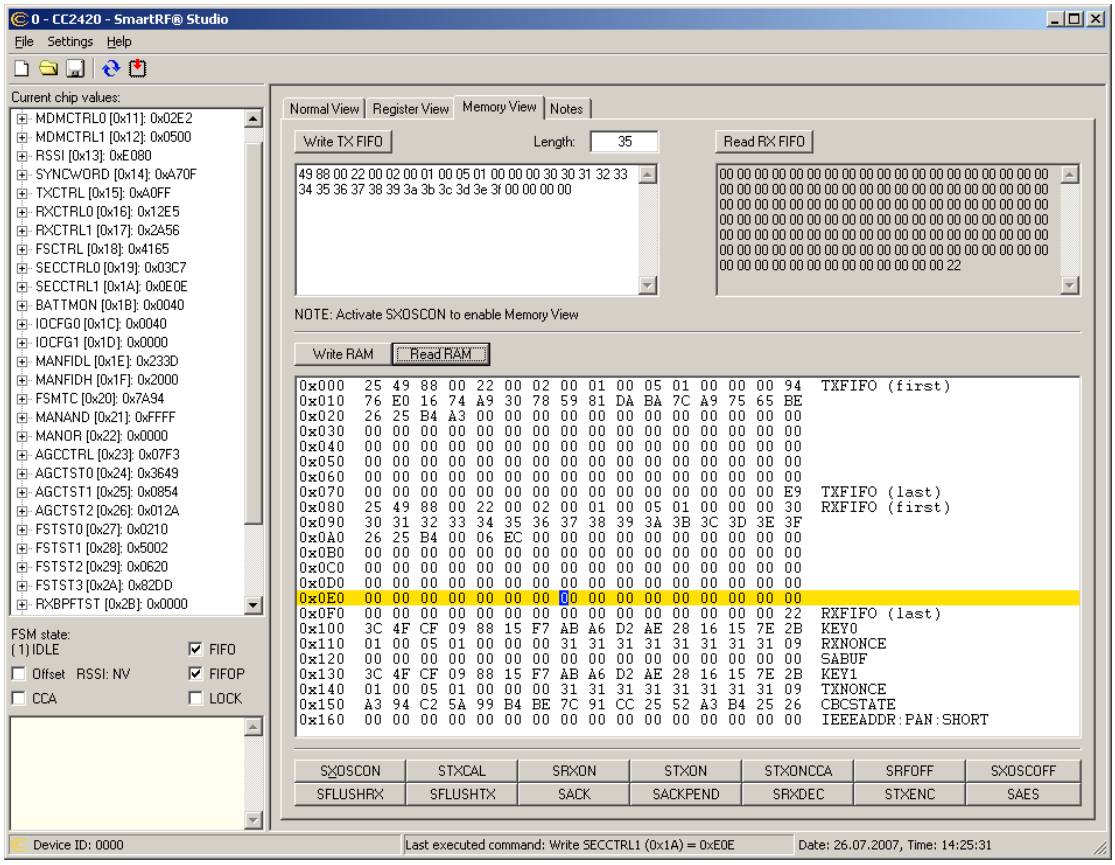

Figure 33. CC2420 RAM after SRXDEC

8 References

[1] IEEE 802.15.4 2006 Specification

[2] CC2420 Data Sheet (cc2420.pdf)

[3] CC2430 Data Sheet (cc2430.pdf)

[4] CC2520 Data Sheet (cc2520.pdf)

[5] CC2530 Data Sheet(cc2530.pdf)

[6] CC2520 SW Examples User's Guide (swru137)

[7] Texas Instruments Packet Sniffer (swrc045)

[8] Smart RF Studio (swrc046)

[9] AN033 Application Note (swra059a)

[10] DN108 Design Note (swra172a)

[11] Wikipedia article on AES

[12] Wikipedia article on ECB

[13] Wikipedia article on CCM

9 General Information

9.1 Document History

Revision	Date	Description/Changes
SWRA209A	2008.07.09	Updated to include CC2530.
SWRA209	2008.06.04	Initial release.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

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

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI 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 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. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

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