Application Report
SLAA465C–August 2010–Revised September 2012

CC430 RF Examples

Miguel Morales and Dung Dang

ABSTRACT

This application report describes the library components and implementation of the RF code examples for the CC430F61xx and CC430F51xx sub-1GHz system-on-chip (SoC) devices. The examples include support for packet-handling modes, including variable and fixed length filtering, as well as the asynchronous and synchronous modes of communication. The document also explains the key RF configuration register settings to enable each mode.

Project collateral and source code discussed in this application report can be downloaded from http://www.ti.com/lit/zip/slac525.

Table 1. Abbreviations

Acronym	Definition
CCR	Capture/Compare Register
CRC	Checksum
FIFO	First-In First-Out
LQI	Link Quality Index
RSSI	Received Signal Strength Indicator
RX	Receive
TX	Transmit

Contents

List of Figures

List of Tables

1 CC430 Modes of Communication

There are essentially two different methods to transmit and receive packets on the CC430: packetized and non-packetized modes. The difference is basically whether the data is received or transmitted through an on-chip 64-byte FIFO or provided directly to the radio in either a synchronous or an asynchronous bit-stream. The following code examples are described in detail in this application report:

Packet-Handling Mode

- Packet size greater than FIFO size (variable and fixed-length filtering)
- Packet size less than FIFO size (variable and fixed-length filtering)

Synchronous or Asynchronous Mode

- Asynchronous communication
- Synchronous communication

2 Packet-Handling Mode

Packet-handling mode leverages the 64-byte FIFO on the CC430 and is a very useful tool for automatically handling common protocol requirements. For example, automatic preamble and sync-word insertion and interpretation, data whitening, packet length, address, and CRC filtering. The default preamble and sync-word modes of the CC430 are used in the code examples; only two different types of length filtering are shown: fixed-length and variable-length packet mode. For a detailed description of the address and CRC filtering capabilities of the radio or of the infinite packet-length filtering mode, see the *CC430 Family User's Guide* (SLAU259) [1].

A typical application packet is shown in Figure 1.

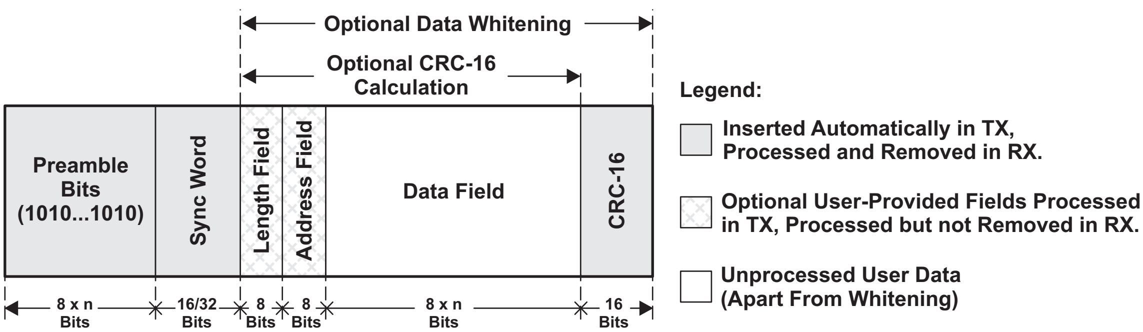

Figure 1. Typical Packet Format

The maximum possible payload size is fundamentally determined by the size of the data FIFO, 64-bytes. The payload includes the length field (optional, only in variable-length mode), the address field (optional, only when address filtering is enabled), and the data field. This means that when a mode is chosen in which the length or address fields are required for filtering the maximum application payload size is reduced.

The maximum application payload size can be further reduced if you choose to leverage an additional feature of the packet-handler: the ability to automatically append two status bytes to every full packet that is read from the RX FIFO. These two status bytes are depicted in Table 2 and Table 3. This append feature can be very useful; it reduces the overhead required to extract the RSSI, LQI, and CRC information specific to the received packet, because no additional read commands are required. The firmware must only continue to read from the RX FIFO and account for the 2-byte reduction in the maximum payload size.

Table 2. Received Packet Status Byte 1 Description

Bit	Field Name	Description
7-0	RSSI	RSSI value

Table 3. Received Packet Status Byte 2 (Second Byte Appended After Data) Description

Bit	Field Name	Value	Description
7	CRC_OK		
		0	CRC error in received data
		1	CRC for received data OK (or CRC disabled)
6-0	LQI		Indicating the link quality

All packet handler examples enable the appended bytes in the RX FIFO.

2.1 Less Than FIFO Size (Less Than 64 bytes)

Packets less than 64 bytes in length fit within the FIFO. The algorithm of the packet transfer is, therefore, fairly straightforward. The CC430 core need only be notified when a packet has begun being transmitted or received, using the sync word interrupt, and then again when the packet has finished being transmit or received, using the end of packet interrupt.

In receive mode, the CC430 core can read up to 64 bytes from the FIFO into a buffer following the end-of-packet interrupt. The same is true for transmit mode. Because the status bytes are automatically appended in all packet handling examples, the maximum packet size is 62 bytes for fixed-length packet mode and 61 bytes for variable-length packet mode.

Figure 2 shows a block diagram of the algorithm used to transmit and receive a packet.

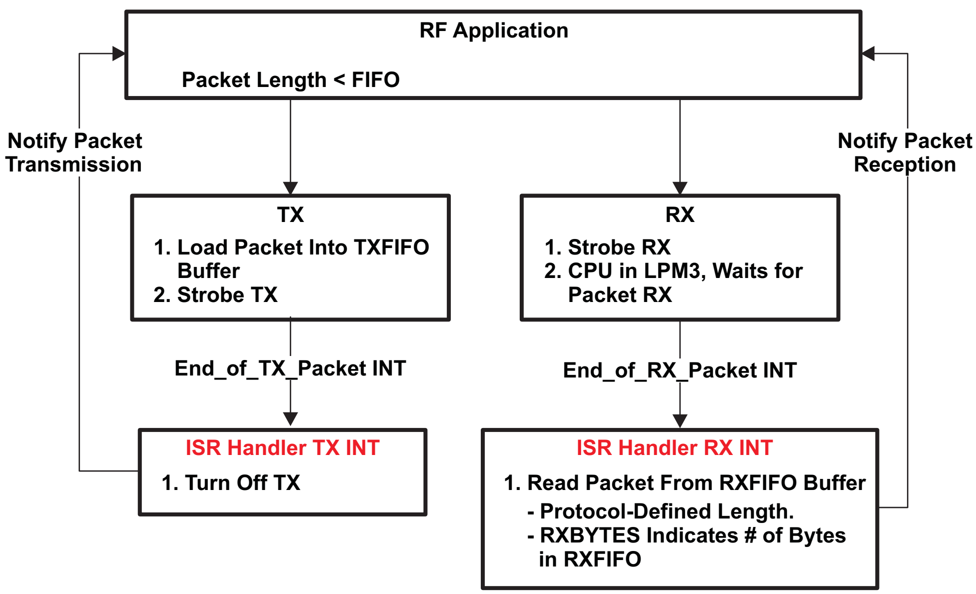

Figure 2. Program Flow for RF TX and RX With Packet Length Less Than FIFO

2.1.1 Key CC1101 Configuration Register Settings

Table 4. PKTCTRL0 – Packet Automation Control

Bit	Field Name	Value	Description
5-4	PKT_FORMAT	00b	Normal mode, use FIFOs for RX an TX
1-0	LENGTH_CONFIG	01b	Variable packet length. Packet length configured by the first byte after sync word.

PKTLEN – Packet Length

Value in PKTLEN sets the fixed packet length. All packets received that are not of this length are filtered out.

2.2 Greater Than FIFO Size (up to 255 Bytes)

Network packets greater than 64 bytes in length do not fit within the FIFO. In this case, the transmit and receive algorithms cannot depend solely on the end-of-packet interrupt to occur because the RX and TX FIFOs would most certainly overflow and underflow, respectively, while waiting. Rather, the firmware must periodically poll for a FIFO status byte and read from the RX FIFO or write to the TX FIFO in bursts until the whole packet has been received or transmitted.

The code examples that implement packet transmissions of a greater size than the on-chip FIFOs initialize a timer to trigger interrupts every 1.2 ms in transmit mode and every 2.6 ms in receive mode. During the timer interrupt service routine, a function is called that requests a status byte from the RF1A interface. The status byte indicates the information shown in Figure 3.

7	6	5	4	3	2	1	0
RF_$\overline{\text{RDY}}$	**RF_STATEx**			**FIFO_BYTES_AVAILx**			

RF_$\overline{\text{RDY}}$	Bit 7	Radio core Ready		
		0	Radio core is ready. Crystal oscillator has stabilized.	
		1	Radio core is not ready. Crystal oscillator not stable.	
RF_STATEx	Bits 6-4	State of the radio core main state machine.		
		000	IDLE	Idle state. Also reported for some transitional states
		001	RX	Receive mode
		010	TX	Transmit mode
		011	FSTXON	Fast TX ready
		100	CALIBRATE	Frequency synthesizer calibration is running
		101	SETTLING	PLL is settling.
		110	RXFIFO_OV	RX FIFO overflow
		111	TXFIFO_OV	TX FIFO overflow
FIFO_BYTES_AVAILx	Bits 3-0	Number of bytes available in the RX FIFO or TX FIFO. Depending on the MSB of the instruction, these bits indicate either the number of bytes available for read from the RX FIFO (MSB = 1) or the number of bytes that can be written to the TX FIFO (MSB = 0). When FIFO_BYTES_AVAILx = 1111. then 15 or more bytes are available or free.		

Figure 3. Status Byte Contents

The two pieces of information used in the code example are extracted from the RF_STATE and FIFO_BYTES_AVAILx bits. The RF_STATE bits let the firmware know if the FIFO transaction has failed, meaning an under or overflow condition has occurred or the radio finds itself unexpectedly in IDLE mode. The FIFO_BYTES_AVAILx bits tell the firmware how many bytes are left in the TX or RX FIFO, depending on the most significant bit of the instruction that was previously passed to the RF1A interface.

Knowing the number of bytes that are available in the RX or TX FIFO allows the firmware to send and receive a packet up to 255 bytes in length in up to 16-byte increments. Figure 4 shows the block diagram of the send and receive algorithms.

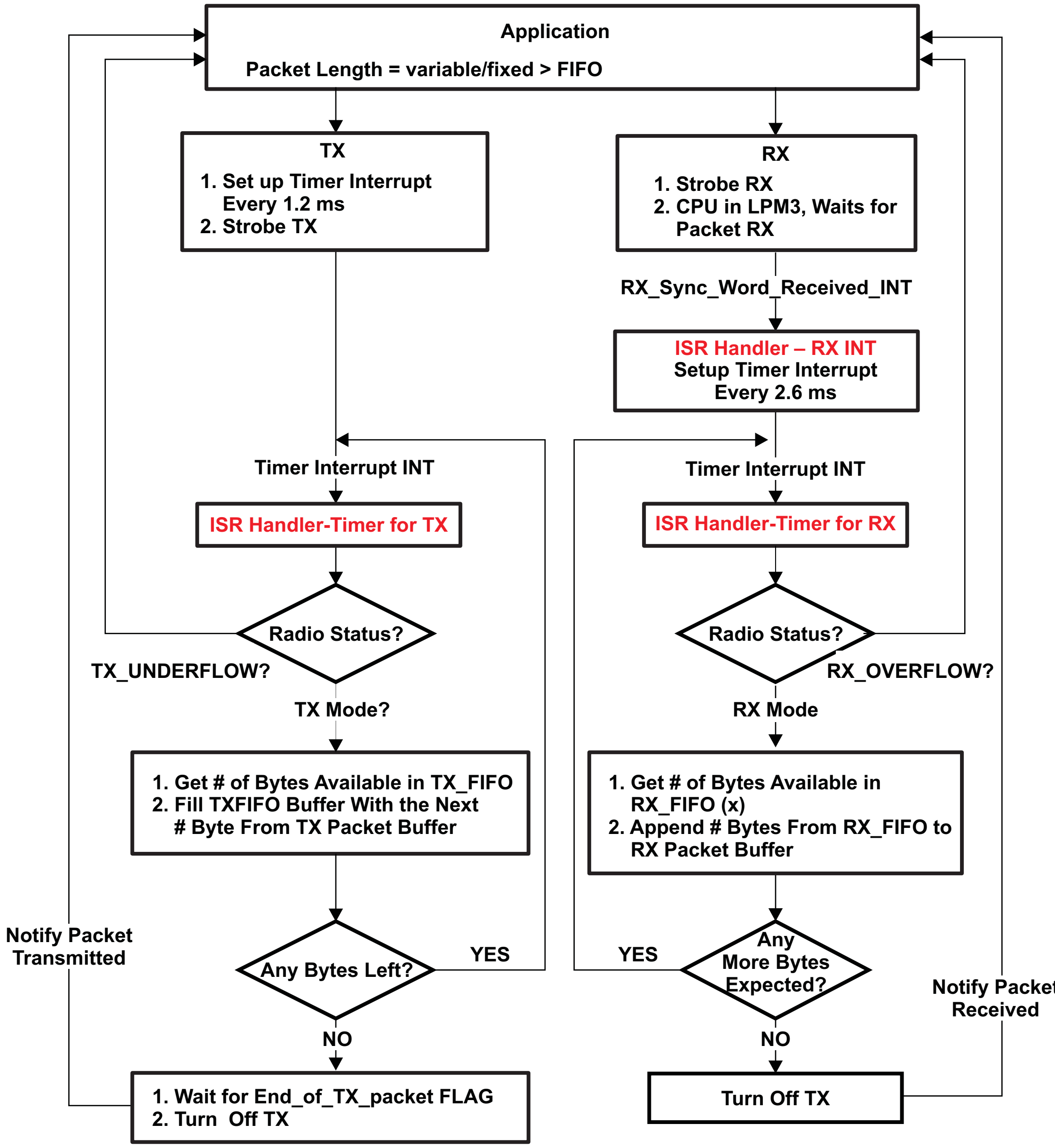

Figure 4. Program Flow for RF TX and RX With Packet Length Greater Than FIFO

2.2.1 Key CC1101 Configuration Register Settings

Table 5. PKTCTRL0 – Packet Automation Control

Bit	Field Name	Value	Description
5-4	PKT_FORMAT	00b	Normal mode, use FIFOs for RX an TX
1-0	LENGTH_CONFIG	00b	Fixed packet length mode. Length configured in PKTLEN register

PKTLEN – Packet Length

Value in PKTLEN sets the fixed packet length. All packets received that are not of this length are filtered out.

3 Synchronous and Asynchronous Modes

The synchronous and asynchronous modes of communication are available if you prefer to handle a raw bit stream instead of using the 64-byte FIFOs. These modes are also useful for making RF measurements; for example, sensitivity testing. Because a radio in transmit mode is essentially modulating a sampled bit stream, the input signal must run at less than or equal to half the raw data rate of the radio transmissions to avoid losing information. In the case of the examples, the raw data rate is 38.4 kbps. The signal that is transmitted and received is a 50% duty cycle square wave of 19.2 kHz.

There are two ways that this bit stream can be provided to the radio:

- A timer output that is internally connected to the radio
- An external pin connection that is port mapped to the GDOx signals

For those that have used the CC1101 in synchronous or asynchronous mode, the external GDOx transmission and reception is exactly the same as a stand-alone CC1101.

The asynchronous and synchronous code examples each implement one of the two methods. The asynchronous example shows how to use the internal connections from the timer output to the radio. The synchronous mode code example shows how to use an external GDOx pin.

3.1 Asynchronous Mode

In asynchronous mode, the signal being transmitted or received is asynchronous to the RF clock. This mode allows full phase, data rate, and protocol flexibility. However, it is also demands the highest CPU loading, because the timing and synchronization of the bit streams must be managed carefully by the CC430 core to ensure good communication. High CPU loading typically leads to higher current consumption. For higher data rates, it is also dependent on a high-accuracy MCU clock.

Because the asynchronous communication example uses the internal timer connection, it is important to know where that connection exists. For more information, see the device-specific data sheet, which shows that the Timer_A1 Capture Compare Register 0 is the interface to the radio (see Short-Form Description > Peripherals > TA1 in that document).

CC430F613x, CC430F612x, CC430F513x MSP430 SoC With RF Core (SLAS554) [2]

CC430F614x, CC430F514x, CC430F512x SoC With RF Core (SLAS555) [3]

Table 6. TA1 Signal Connections

Device Input Signal	Module Input Name	Module Block	Module Output Signal	Device Output Signal
PM_TA1CLK	TACLK	Timer	N/A	
ACLK (internal)	ACLK			
SMCLK (internal)	SMCLK			
RFCLK/192	INCLK			
PM_TA1CCR0A	CCI0A	CCR0	TA0	PM_TA1CCR0A
RF Asynchronous Output (internal)	CCI0B			RF Asynchronous Input (internal)
DVSS	GND			
DVCC	V_{CC}			
PM_TA1CCR1A	CCI1A	CCR1	TA1	PM_TA1CCR1A
CBOUT (internal)	CCI1B			
DVSS	GND			
DVCC	V_{CC}			
PM_TA1CCR2A	CCI2A	CCR2	TA2	PM_TA1CCR2A
ACLK (internal)	CCI2B			
DVSS	GND			
DVCC	V_{CC}			

The TA1CCR0 capture/compare register should be used to generate or capture the outgoing and incoming signals. The code flow for the transmitting node, depicted in Figure 5, shows how the Timer_A1 CCR0 and CCR1 outputs are initialized to toggle at frequency of 38.4 kbps – resulting in a 19.2-kHz square wave that is both provided internally to the radio as well as externally for inspection on P2.2. Figure 5 shows the block diagram of the transmitting node firmware.

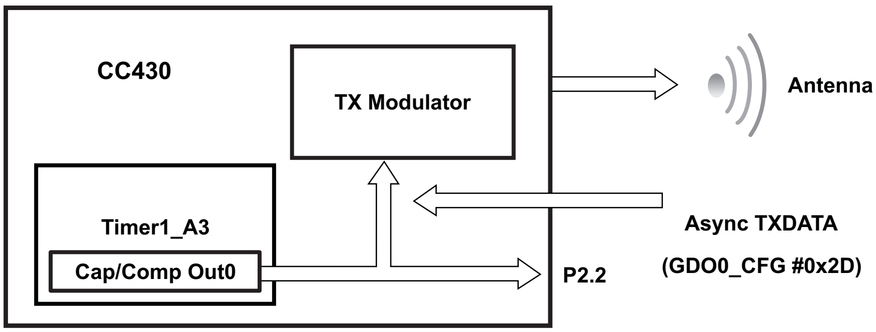

Figure 5. Asynchronous Mode TX With Timer_A CCRx Internal Connections

The receiving node enters receive mode with the GDOx signal mapped to P2.6 and the TA1CCR0 capture interrupt enabled so that the incoming signal can be proven on either the timer interface or with an oscilloscope. Figure 6 shows the block diagram of the receiving node firmware.

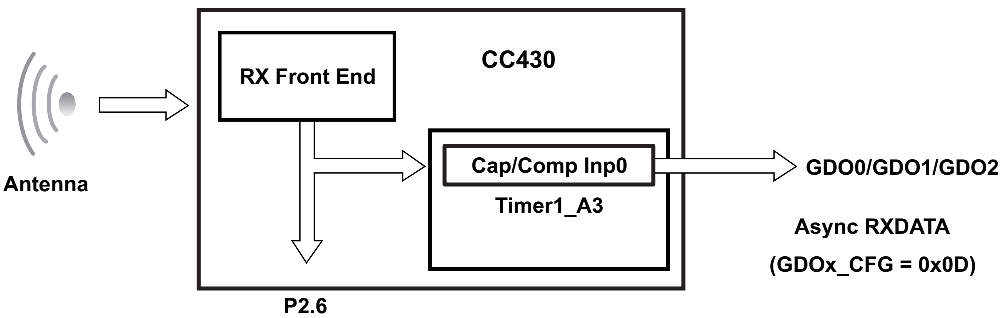

Figure 6. Asynchronous Mode RX With Timer_A CCRx Internal Connections

3.1.1 Key CC1101 Configuration Register Settings

Table 7. PKTCTRL0 – Packet Automation Control

Bit	Field Name	Value	Description
5-4	PKT_FORMAT	11b	Asynchronous serial mode

Table 8. IOCFG0 – GDOx Signal Selection (x = 0, 1, or 2)

Bit	Field Name	Value	Description
5-0	GDO0_CFG	Transmit - 0x2E (46)	3-state
		Receive - 0x0D (13)	Serial Data Output. Used for asynchronous serial mode

Transmitting Node – The importance of this register setting, which maps different radio functionality to the GDO0 pin, is that GDO0_CFG NOT be set to 0x2D, which would select GDO0 as the transmission signal. Instead, the firmware intends to provide the signal through the internal connection of the timer output. The high-impedance setting is just an easy place-holder for the GDOx functionality that does not affect the system.

Receiving Node – The GDO0 signal reflects the serial data stream output from the radio, which is then port mapped to a port pin for verification of the reception by an oscilloscope.

3.2 Synchronous Mode

In synchronous mode, the transmit or receive signal is synchronous to an RF clock that is provided on GDO2. Synchronous mode provides great flexibility when it comes to the protocol and packet contents, with significantly less CPU load. This is due to the fact that the CC430 core can react to the edges of the data clock signal instead of being 100% responsible for the timing of the bit stream. Usually, the firmware can operate at a slower MCLK; however, the data rate and phase flexibility of the bit stream is significantly reduced, because the radio must be able to derive the required data rate from the RF reference clock, and there can be no additional delays introduced to the continuous RF clock to change phase.

In the synchronous code example, GDO0 is port mapped to P2.6. The GDO0_CFGx bits in the IOCFG0 configuration register are set such that the signal provided to the GDO0 signal is the bit stream to be transmitted. The TA1 timer still generates a 19.2-kHz waveform, but to P2.4, which must be jumpered to P2.6 to serve as the input waveform. Figure 7 shows the block diagram for the transmitting node.

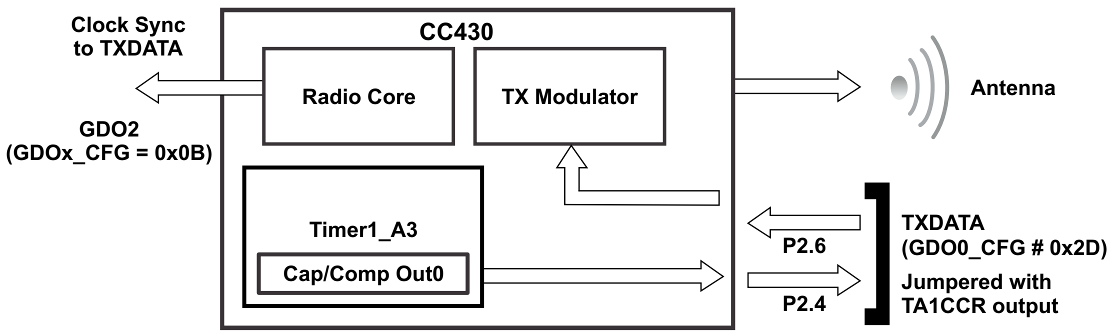

Figure 7. Synchronous Mode TX With Timer_A CCRx Connections

The receive node maps the GDO0 and GDO2 signals to P2.6 and P2.7, respectively, in order to be inspected and verified using an oscilloscope. Figure 8 shows the block diagram for the receiving node.

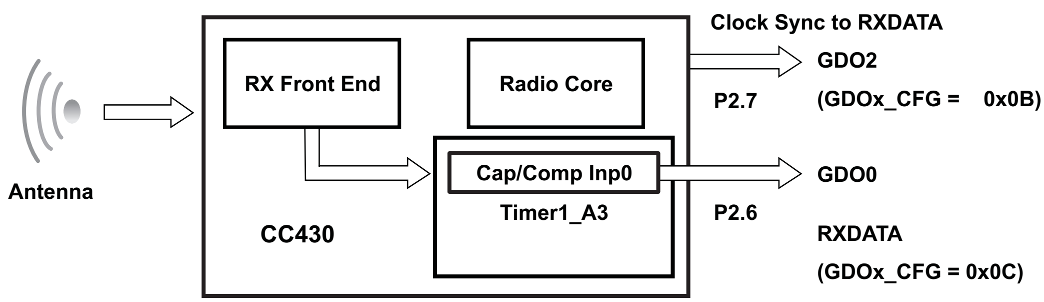

Figure 8. Synchronous Mode RX With Timer_A CCRx Connections

Table 9. PKTCTRL0 – Packet Automation Control

Bit	Field Name	Value	Description
5-4	PKT_FORMAT	01	Synchronous serial mode. Used for backwards compatibility.

Table 10. IOCFG0 – GDOx Signal Selection (x = 0, 1, or 2)

Bit	Field Name	Value	Description
5-0	GDO0_CFG	Transmit - 0x2D (46)	Serial input data is taken from GDO0
		Receive - 0x0C (12)	Serial Synchronous Data Output. Used for synchronous serial mode.

Transmitting Node – This register setting maps serial input data to be accepted from the GDO0 signal, which can then be mapped to a port pin in order for the bit stream to be provided externally.

Receiving Node – This register setting maps the serial synchronous data output to the GDO0 signal, which can then be mapped to a port pin in order for the received bit stream to be inspected externally.

Table 11. IOCFG2 – GDOx Signal Selection (x = 0, 1, or 2)

Bit	Field Name	Value	Description
5-0	GDO2_CFG	Transmit - 0x0B (12) Receive - 0x0B (12)	Serial clock. Synchronous to the data in synchronous serial mode.

Transmitting and Receiving Nodes – This register setting provides the RF bit clock to the GDO2 signal, which can then be port mapped to a port pin to be viewed externally.

4 Code Organization

The RF examples projects were developed for the EM430F6137RF900 and EM430F6147RF900 target boards in both Code Composer Studio™ IDE and IAR Embedded Workbench™ IDE. Each project demonstrates TX and RX communication in one of the modes discussed in this application report. The projects shown in Table 12 are included in the associated code package (SLAC525) for both IDEs in their respective workspaces (RF_Examples_IAR and RF_Examples_CCS):

Table 12. Example Projects

1	Fixed_LT_FIFO	Packet with fixed length and less than FIFO size
2	Fixed_GT_FIFO	Packet with fixed length and greater than FIFO size
3	Fixed_LT_FIFO	Packet with variable length and less than FIFO size
4	Fixed_LT_FIFO	Packet with variable length and greater than FIFO size
5	Asynchronous_comm	Asynchronous mode
6	Synchronous_comm	Synchronous mode

Each project has frequency build configurations for both 868 MHz (ETSI) and 915 MHz (FCC) frequency bands.

5 References

1. *CC430 Family User's Guide* (SLAU259)
2. *CC430F613x, CC430F612x, CC430F513x MSP430 SoC With RF Core Data Sheet* (SLAS554)
3. *CC430F614x, CC430F514x, CC430F512x SoC With RF Core Data Sheet* (SLAS555)