Introduction to CC430 – RF1A

Peter Spevak
Introduction to CC430
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
MCUs and RF opportunities are endless

Worldwide market for wireless technology in manufacturing will grow at a 32% CAGR over the next five years, and is projected to exceed $1 billion in 2010

- ARC Advisory Group
TI Low-Power RF at a glance…

**Alarm and Security**

- **CC1110**
  - Sub 1 GHz SoC
  - 32KB Flash
  - 0.3 µA sleep current

- **CC1101 / CC430**
  - Sub 1 GHz Transceiver
  - + MSP430 MCU,
  - 500 Kbps
  - -112dBm sensitivity

**Remote Controls**

- **CC2530**
  - RF4CE
  - IEEE 802.15.4 compliant
  - System on Chip
  - RemoTI SW

- **CC2511**
  - 2.4 GHz Radio
  - 8051 MCU,
  - 32 KB Flash, USB 2.0
  - Proprietary solution

**Metering**

- **CC2530**
  - ZigBee
  - System on Chip
  - IEEE 802.15.4 compliant
  - CC259x Range Extenders

- **CC1020**
  - Narrowband
  - 12.5 KHz channel spacing
  - -118dBm sensitivity

**Wireless Audio**

- **CC2505S**
  - PurePath™ Wireless
  - Coming Soon
  - CD Quality
  - Wireless Audio

- **CC2591**
  - 2.4 GHz Range Extender
  - +22dBm output power

**Home Automation & Lighting**

- **CC2431**
  - Location Tracking
  - System on Chip
  - Solutions

- **CC2480**
  - Network Processor
  - fully certified ZigBee 2006
  - Software Stack

**Sport & Gaming**

- **CC2410**
  - Bluetooth Low Energy
  - Coming Soon
  - Single-mode BTLE SoC

- **CC2540**
  - 2.4 GHz Transceiver
  - +MSP430 MCU

**Sub 1 GHz Transceiver**

- CC1101 / CC430
  - Sub 1 GHz Transceiver
  - + MSP430 MCU,
  - 500 Kbps
  - -112dBm sensitivity

**Narrowband**

- CC1020
  - Narrowband
  - 12.5 KHz channel spacing
  - -118dBm sensitivity

**Network Processor**

- CC2480
  - Network Processor
  - fully certified ZigBee 2006
  - Software Stack

**Location Tracking**

- CC2431
  - Location Tracking
  - System on Chip
  - Solutions

**Home Automation & Lighting**

- **CC2410**
  - Bluetooth Low Energy
  - Coming Soon
  - Single-mode BTLE SoC

- **CC2540**
  - 2.4 GHz Transceiver
  - +MSP430 MCU

**ZigBee**

- CC2530
  - ZigBee
  - System on Chip
  - IEEE 802.15.4 compliant
  - CC259x Range Extenders
MSP430 + CCxxxxxx Radio Solutions

The MSP430 product family supports a variety of wireless protocols

<table>
<thead>
<tr>
<th>Communication Method</th>
<th>Point-to-Point</th>
<th>Star Network</th>
<th>Mesh Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard-Based Network</td>
<td>IEEE 802.15.4 CC2420</td>
<td>IEEE 802.15.4 CC2420 with TIMAC</td>
<td>IEEE 802.15.4 CC2420 with TIMAC and Z-STACK</td>
</tr>
<tr>
<td></td>
<td>MSP430F22xx/ MSP430F161x/ MSP430FG461x*</td>
<td>MSP430F22xx/ MSP430F161x/ MSP430FG461x*</td>
<td>MSP430F22xx/ MSP430F161x/ MSP430FG461x*</td>
</tr>
<tr>
<td>Proprietary Network</td>
<td>Sub 1-GHz CC1100/CC1150, CC1020/CC1070, CC1000/CC1050 Any MSP430 Device</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.4 GHz CC2500/CC2550 Any MSP430 Device</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* These are recommended devices. Any MSP430 can be used.

SimpliciTI, TI-MAC (802.15.4), Z-Stack (ZigBee & ZigBee Pro)
### TI’s Three RF solutions

<table>
<thead>
<tr>
<th>Application</th>
<th>RF Systems-on-Chip</th>
<th>Application MCU RF Radio</th>
<th>Application MCU+ RF Protocol Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>RF SoC (CC430, CC2430, CC2510, CC1110)</td>
<td>MSP430 (F54xx, F24xx, F26xx)</td>
<td>MSP430</td>
</tr>
<tr>
<td>Wireless Protocols</td>
<td></td>
<td></td>
<td>Protocol processor (CC2480, CC430)</td>
</tr>
<tr>
<td>Radio</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sub 1-GHz Wireless Applications

Demand for more intelligence & cutting the cable
- Property and asset IDs
- Networked sensors
- Heat cost allocator
- Automatic meter reading
- Alarm and security

Looking for power in new places
- Solar
- Thermal
- Motion and vibration

Intersection of higher performance, lower cost and low power
What is the CC430?

• It is a subfamily of the MSP430 with 7 derivatives
• CPU core is based on the MSP430F5xx technology
• Radio is based on the CC1101 Sub 1GHz transceiver
• “CC” points to TI LPW SRD products “430” to MSP430 Microcontrollers.
• Supports 300-348MHz, 387- 464MHz and 779-928MHz
• This includes in EU the so called 868MHz band (852-870MHz) the 433MHz band and the US 315MHz and 915MHz (902-928MHz) band.
The CC430 – Integrated System-On-Chip

CC430

Low Power RF IC
Radio frequency

MSP430
Application and protocol processor

Low-power RF SoC

Low Power < 1 GHz RF Transceiver
- High sensitivity
- Low current consumption
- Excellent blocking performance
- Flexible data rate & modulation format
- Backwards compatible

MSP430 5xx MCU
- Ultra-low power
- High analog performance
- High level of integration
- Ease of development
- Sensor interface
CC430 application spaces

- **Consumer & Home area networking**
  - Watch / heart monitor combination for monitoring miles and calories
  - Enough processing for wireless networking and batteries that last many years

- **Remote monitoring**
  - Low power sensor networks for distributed status reporting of integrity, temperature, humidity,
  - Harvest energy from motion, vibration and heat

- **Asset Tracking**
  - Encryption enables secure wireless reporting information
  - Location, tamper detection and temperature monitoring
TI Low-Power RF – EU Sub 1GHz

Alarm and Security

- **CC1110**
  - Sub 1 GHz SoC
  - 32KB Flash
  - 0.3 uA sleep current

- **CC1101 / CC430**
  - Sub 1 GHz Transceiver
  - + MSP430 MCU,
    - 500 Kbps
    - -112dBm sensitivity

Sub Metering - AMR

- **CC1110**
  - Sub 1 GHz SoC
  - 32KB Flash
  - 0.3 uA sleep current

- **CC1020**
  - Narrowband
  - 12.5 KHz channel spacing
  - -118dBm sensitivity

Smart Metering

- **CC1110 / CC430**
  - Sub 1 GHz Transceiver
  - + MSP430 MCU,
    - 500 Kbps
    - -112dBm sensitivity

Home Automation & Lighting

- **CC1101 / CC430**
  - Sub 1 GHz Transceiver
  - + MSP430 MCU,
    - 500 Kbps
    - -112dBm sensitivity

- **CC1110**
  - Sub 1 GHz SoC
  - 32KB Flash
  - 0.3 uA sleep current
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
Global frequency bands

- USA: 315/433/915 MHz
- Europe: 433/868 MHz
- Asia: 315/433/470/780 MHz
- Africa: 433 MHz
- Japan: 315/426/950 MHz
- 2.4 GHz
Short-Range Wireless Comparison

Different Value Drivers for Different Applications

Proprietary Low Power Radio
- Gaming
- PC Peripherals
- Audio
- Meter Reading
- Building Mgmt.
- Automotive

< 1GHz Standard
- Building Automation
- Residential Control
- Industrial
  --- Apps. in both freq. bands ---
- Tracking
- Sensors
- Home Automation / Security
- Meter Reading

ZigBee/802.15.4

Wi-Fi/802.11
- PC Networking
- Home Networking
- Video Distribution

UWB
- Wireless USB
- Video/audio links

Range
1000m
100m
10m
1m

Data Rate (bps)
1k
10k
100k
1M
10M
Basic Building Blocks of an RF System

- **RF-IC**
  - Transmitter / Receiver
  - Transceiver
- **Microcontroller**
  - For protocol and application processing
  - System-on-Chip (SoC): typically transceiver with integrated uC
- **Crystal**
  - Reference frequency for the LO and the carrier frequency
- **Balun**
  - Balanced to unbalanced
  - Converts a differential signal to a single-ended signal or vice versa
  - Impedance Matching
- **Filter**
  - Used if needed to pass regulatory requirements / improve selectivity
- **Antenna**
Important Factors for Radio Range

• Antenna (gain, sensitivity to body effects etc.)
• Sensitivity
• Channel Selectivity
• Output power
• Radio pollution (selectivity, blocking, IP3)
• Environment (Line of sight, obstructions, reflections, multi-path fading)
Antennas, commonly used

- **PCB antennas**
  - Little extra cost (PCB)
  - Size demanding at low frequencies
  - Good performance possible
  - Complicated to make good designs

- **Whip antennas**
  - Expensive (unless piece of wire)
  - Good performance
  - Hard to fit in many applications

- **Chip antennas**
  - Expensive
  - OK performance
  - Small size
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
CC430F613x Block Diagram
Shared Reference

- Excellent PSRR and temperature coefficient
- Selectable internal references
  - 1.5, 2.0, 2.5V
- Backwards compatible with existing reference (ADC12)
- Sampled reference mode provides tradeoff power consumption <> accuracy
  - Can be used for Comp_B or LCD_B reference generation
  - Not recommended for use with ADCs / DACs
Comparator_B

- Inverting and non-inverting terminal input multiplexer
- High-speed, normal, and ultra-low power modes

- Internal output to Timer A capture
- Selectable RC filter for comparator output
Comparator_B – Reference Voltage Generator

- Selectable reference voltages
- Shared reference can provide 1.2 V, 1.5 V, 2.0 V, 2.5 V
- Automatic voltage hysteresis generator
• CAPD shut down capability to avoid cross currents with analog voltage levels at I/Os
LCD_B

- Blinking of individual segments at a programmable frequency
- Programmable frame rate
- Regulated charge pump
- Software-driven contrast control
- Integrated drivers to decouple LCD load from the bias generation
LCD_B Bias Generation

- Bias voltages V2 to V5 can be generated using programmable internal or external $V_{LCD}$
- Integrated drivers provide stable node voltage and good contrast to large LCD segments
- When using an external resistor ladder:
  - $V_{LCD}$ may be sourced from the internal charge pump
  - R33 may serve as a switched LCD output
AES Encryption Accelerator

• Main features
  – En- and decryption according to AES FIPS PUB 197
  – Off-line key generation for decryption
  – Byte and word-access to key, input and output data
  – AES ready interrupt flag
  – Built-in DMA support
  – The interpretation of word data can be configured as little-endian (AESENDIAN = 0) or as big-endian with (AESENDIAN = 1)
ADC12_A Enhanced Features

- VREF settling time
  - 50us vs. 17ms
- Tighter temp coefficient on internal reference
  - ±50ppm vs. ±100ppm
- Lower power modes
  - Selectable speed vs power
  - References automatically shut down to conserve power
- Higher clock dividers for faster system clocks
- ~6 x lower current than ADC12
  - 220uA for ADC active
  - 100uA for 2.5V VREF active
USCI Enhanced Features

- Interrupts re-designed
  - Separate vectors for USCI_A & USCI_B – no more sharing or bit-testing
  - Interrupt vector generator register
  - Simplifies USCI interrupt operations
  - Reduces code size
Port mapping

- Each output signal is mapped to several output pins (per device datasheet)
- Each port Px.y (P1 – P3) pin has its own PxMAPy register
- Mapping is runtime re-configurable
  - Single configuration per PUC Reset ... OR ...
  - PMARECFG bit allows runtime re-configurations
- Port mapping configuration is password protected
- Write access is locked when...
  - Invalid password is written while access is granted
  - Timeout counter reaches 32
- All mappable pins also support PM_Analog functionality
  - Disables port logic to prevent parasitics
# CC430 Synchronous Mode

## Timer0_A5 connections

<table>
<thead>
<tr>
<th>DEVICE INPUT SIGNAL</th>
<th>MODULE INPUT NAME</th>
<th>MODULE BLOCK</th>
<th>MODULE OUTPUT SIGNAL</th>
<th>DEVICE OUTPUT SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM_TA0CLK</td>
<td>TACLK</td>
<td>Timer</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>ACLK (internal)</td>
<td>ACLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMCLK (internal)</td>
<td>SMCLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFCLK/192(1)</td>
<td>INCLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA0CCR0A</td>
<td>CCI0A</td>
<td></td>
<td>TA0</td>
<td>PM_TA0CCR0A</td>
</tr>
<tr>
<td>DVss</td>
<td>CCI0B</td>
<td>CCR0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVss</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVcc</td>
<td>Vcc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA0CCR1A</td>
<td>CCI1A</td>
<td></td>
<td>TA1</td>
<td>PM_TA0CCR1A</td>
</tr>
<tr>
<td>CBOUT (internal)</td>
<td>CCI1B</td>
<td>CCR1</td>
<td></td>
<td>ADC12 (internal)(2)</td>
</tr>
<tr>
<td>DVss</td>
<td>GND</td>
<td></td>
<td></td>
<td>ADC12SHSx = (1)</td>
</tr>
<tr>
<td>DVcc</td>
<td>Vcc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA0CCR2A</td>
<td>CCI2A</td>
<td></td>
<td>TA2</td>
<td>PM_TA0CCR2A</td>
</tr>
<tr>
<td>ACLK (internal)</td>
<td>CCI2B</td>
<td>CCR2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVss</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVcc</td>
<td>Vcc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA0CCR3A</td>
<td>CCI3A</td>
<td></td>
<td>TA3</td>
<td>PM_TA0CCR3A</td>
</tr>
<tr>
<td>GDO1 from Radio (internal)</td>
<td>CCI3B</td>
<td>CCR3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVss</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVcc</td>
<td>Vcc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA0CCR4A</td>
<td>CCI4A</td>
<td></td>
<td>TA4</td>
<td>PM_TA0CCR4A</td>
</tr>
<tr>
<td>GDO2 from Radio (internal)</td>
<td>CCI4B</td>
<td>CCR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVss</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVcc</td>
<td>Vcc</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Clock from RF1A
Signals from RF1A
RF data
RF data clk
other signals
## CC430 Asynchronous Mode

<table>
<thead>
<tr>
<th>Device Input Signal</th>
<th>Module Input Name</th>
<th>Module Block</th>
<th>Module Output Signal</th>
<th>Device Output Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM_TA1CLK</td>
<td>TACLK</td>
<td>Timer</td>
<td>NA</td>
<td>PZ</td>
</tr>
<tr>
<td>ACLK (internal)</td>
<td>ACLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMCLK (internal)</td>
<td>SMCLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFCLK/192(1)</td>
<td>INCLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA1CCR0A</td>
<td>CCI0A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RF Async. Output</strong> (internal)</td>
<td><strong>CCI0B</strong></td>
<td><strong>CCR0</strong></td>
<td><strong>TA0</strong></td>
<td><strong>RF Async. Input</strong> (internal)</td>
</tr>
<tr>
<td>DVSS</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVC</td>
<td>VCC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA1CCR1A</td>
<td>CCI1A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBOUT (internal)</td>
<td>CCI1B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVSS</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVC</td>
<td>VCC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PM_TA1CCR2A</td>
<td>CCI2A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACLK (internal)</td>
<td>CCI2B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVSS</td>
<td>GND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DVC</td>
<td>VCC</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Clock from RF1A  
RF data from RF1A  
RF data to RF1A
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
CC430 Derivatives
CC430 Derivatives – CC430F613x

CC430F613x Functional Block Diagram

- Unified Clock System
- ACLK
- SMCLK
- MCLK
- XIN (32kHz)
- XOUT

- P1.x/P2.x
- P3.x/P4.x
- P5.x
- RF_XIN
- RF_XOUT
- 2x8
- 2x8
- 1x8
- (26MHz)

- DMA Controller
  - 3 Channel
- CPUXV2 incl. 16 Registers
- EEM (S: 3+1)
- JTAG Interface
- Spy-Bi-Wire
- Flash
  - 32kB
  - 16kB
- RAM
  - 4kB
  - 2kB
- CRC16
- SYS Watchdog
- Port Mapping Controller
- MPY32
- I/O Ports
  - P1/P2
  - 2x8 I/Os
  - 1x16 I/Os
- I/O Ports
  - P3/P4
  - 2x8 I/Os
  - 1x16 I/Os
- I/O Ports
  - P5
  - 1x8 I/Os

- Packet Handler
  - Digital RSSI
  - Carrier Sense
  - PQL / LQI
  - CCA
- Sub-1GHz Radio
  - (CC1101)
- CPU Interface
- MODEM
- Frequency Synthesizer
- RF/ANALOG TX & RX
- Power Mgmt
- LDO
- SVM
- SSV
- Brownout
- TA0
  - 5 CC Registers
- TA1
  - 3 CC Registers
- RTC_A
- USCI_A0
  - (UART, I2C, SPI)
- USCI_B0
  - (SPI, I2C)
- LCD_B
  - 96 Segments
  - 1, 2, 3, 4 Mux
- AES 128
  - Security Env/Decryption

- Texas Instruments
CC430 Derivatives – CC430F612x

CC430F612x Functional Block Diagram

- DMA Controller: 3 Channel
- CPUXV2: incl. 16 Registers
- EEM (S: 3+1)
- JTAG Interface
- Spy-Bi-Wire
- Power Mgmt: LDO SVM/SVS Brownout
- TA0: 5 CC Registers
- TA1: 3 CC Registers
- RTC_A
- USCI_A0 (UART, IrDA, SPI)
- USCI_B0 (SPI, I2C)
- LCD_B: 96 Segments, 1,2,3,4 Mux
- AES128: Security En-/Decryption
- RF_PHY
- RF_XIN, RF_XOUT (26MHz)
- Packet Handler
- Digital RSSI
- Carrier Sense
- PQL / LQI
- CCA
- SUB-1GHz Radio (CC1101)
- CPU Interface
- MODEM
- Frequency Synthesizer
- RF/ANALOG TX & RX
- RF_P, RF_N
CC430 Derivatives – CC430F513x
CC430 derivatives

**With LCD module**
- **CC430F6137** (LCD, ADC, Comparator B, 64-Pin)
  - 32KB+512B FLASH
  - 4KB RAM
- **CC430F6135** (LCD, ADC, Comparator B, 64-Pin)
  - 16KB+512B FLASH
  - 2KB RAM
- **CC430F6127** (LCD, Comparator B, 64-Pin)
  - 32KB+512B FLASH
  - 4KB RAM
- **CC430F6125** (LCD, Comparator B, 64-Pin)
  - 16KB+512B FLASH
  - 2KB RAM

**Without LCD module**
- **CC430F5137** (ADC, Comparator B, 48-Pin)
  - 32KB+512B FLASH
  - 4KB RAM
- **CC430F5135** (ADC, Comparator B, 48-Pin)
  - 16KB+512B FLASH
  - 2KB RAM
- **CC430F5133** (ADC, Comparator B, 48-Pin)
  - 8KB+512B FLASH
  - 2KB RAM

With LCD module
Without LCD module
## CC430 - Packages

<table>
<thead>
<tr>
<th>$T_A$</th>
<th>PACKAGED DEVICES$^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PLASTIC 64-PIN QFN (RGC)</td>
</tr>
<tr>
<td>$-40^\circ C$ to $85^\circ C$</td>
<td>CC430F6137IRGC</td>
</tr>
<tr>
<td></td>
<td>CC430F6135IRGC</td>
</tr>
<tr>
<td></td>
<td>CC430F6127IRGC</td>
</tr>
<tr>
<td></td>
<td>CC430F6126IRGC</td>
</tr>
<tr>
<td></td>
<td>CC430F6125IRGC</td>
</tr>
</tbody>
</table>

(1) For the most current package and ordering information, see the Package Option Addendum at the end of this document, or see the TI web site at [www.ti.com](http://www.ti.com).

(2) Package drawings, thermal data, and symbolization are available at [www.ti.com/packaging](http://www.ti.com/packaging).
CC430 – Packages 64-pin RGC
CC430 – Packages 48-pin RGZ
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
MSP430 CPUXv2

- C-compiler friendly
- CPU registers expanded to 20-bits
- Address instructions
  - Direct 20-bit CPU register access
  - Atomic instructions
- Instruction compatible with all MSP430’s
- Extension word allows all instructions
  - Direct access to 1MB address space
  - Bit, byte, word and address-word data
  - Repeat instruction function
CC430 Memory Map

- Page-free 16-bit addressing
- User-definable interface to the Boot Strap Loader (BSL)
- 4 User Info Segments
  - INFO_A can be locked
- Factory data brought out to its own location
  - Factory calibration values
    - ADC offset, temperature, DCO
  - Unique product ID
- Vector table starts at 0xFF80 for compatibility with previous MSP430’s
- RF1A registers from 0x0F00 – 0x0F3F
CC430 Operating Modes

- LPM0 – LPM4 are the same as previous MSP430 generations
- BOR resets device, executes internal boot code, registers default
- SVS/M generate POR/interrupts in event of low voltage condition
CC430 Unified Clock System (UCS)

- Six independent clock sources
  - Low Freq
    - LFXT1
    - VLO
    - REFO
  - High Freq
    - RF XT2 (26 MHz)
    - DCO – optional FLL
    - MODOSC
- FLL reference selectable from three, divisible low-freq sources
  - LFXT1
  - REFO
  - XT2
- ACLK / SMCLK / MCLK can be driven from any source
- MODOSC provided to modules
  - Flash controller & ADC12
- Clocks on demand
Radio Core Operations

1. Change RF Core State

2. Access Configuration Registers
   (to configure RF Core settings)

3. Access TX/RXFIFO
   (to access transmit / receive data)
CC430 Radio
CC430 Radio – TX Path

Diagram showing the TX Path with components such as LNA, ADC, Demodulator, Frequency Synthesizer, Modulator, Packet Handler, RXFIFO, TXFIFO, BIAS, and XOSC.
CC430 Radio – TX Path – Power Amplifier

Main Parameters

• Up to +10dBm output power to 50Ohm
• Built in attenuation steps down to -30dBm
• Built-in LDO keeps the output power stable across the operating range of the transceiver 2.2V – 3.6V
• Output is balanced >> requires a balun for a single ended antenna
CC430 Radio – TX Path – PA Table

- Used for PA power ramp up and ramp down
- Used for OOK and ASK modulation
- No huge power load changes from 0 to max. causing VCO sweeps over the close channels
- Cleaner RF environment in RF networks with a lot of TDMA transmission.
- Good for meeting the ETSI requirements on power ramping

Figure 22-11. PA_POWER and PATAABLE

Figure 22-12. Shaping of ASK Signal
CC430 Radio– TX Path - Synthesizer

Purpose of the Synthesizer

• Generate in cooperation with the PLL and VCO the required output frequency.

Main parameters

• Reference crystal frequency range 26MHz – 27MHz
• Programmed Frequency resolution 397Hz – 412Hz

Main features

• Very high frequency resolution, can be used to correct frequency tolerance of the reference crystal.
• 1ppm at 868MHz are 868Hz >> resolution better than 0.5ppm
• Enables compensation of the frequency drift due to temperature
Purpose of the Modulator

• Generate frequency modulation schemes, 2-FSK, 2-GFSK and MSK

Main parameters

• Supports programmable data rates in the range 1.2kBaud – 500kBaud
• Defines deviation according to

Table 22-13. Symbol Encoding for 2-FSK/2-GFSK Modulation

<table>
<thead>
<tr>
<th>FORMAT</th>
<th>SYMBOL</th>
<th>CODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-FSK/2-GFSK</td>
<td>0</td>
<td>– Deviation</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>+ Deviation</td>
</tr>
</tbody>
</table>

\[ f_{dev} = \frac{f_{xosc}}{2^{17}} \times (8 + \text{DEVIAITON}_M) \times 2^{\text{DEVIAITON}_E} \]

The default values give ±47.607 kHz deviation, assuming 26-MHz crystal frequency.

Main features

• Supports 2-FSK, 2-GFSK and MSK modulation formats.
• For amplitude modulation the PA-Table is used.
• Supports automatic Manchester encoding, when enabled.
CC430 Radio MIB – RX Path
CC430 Radio – RX Path

LNA

- Amplify the RF signal with the lowest possible noise figure
- Highest possible saturation point

Image rejection Mixer

- Converts the high frequency signal down to IF frequency.
- Suppresses the image resulting from mixing the carrier frequency with the VCO frequency for down conversion.
- This serves the same purpose as SAW filters.
# CC430 Radio – RX Path

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>868 MHz, 1.2 kBaud data rate, sensitivity optimized, MDMCFG2.DEM_DCFILT_OFF=0</td>
<td>-111</td>
<td>dBm</td>
<td>Sensitivity can be traded for current consumption by setting MDMCFG2.DEM_DCFILT_OFF=1. The typical current consumption is then reduced from 18.0 mA to 15.7 mA at sensitivity limit. The sensitivity is typically reduced to -109 dBm</td>
</tr>
<tr>
<td>Saturation</td>
<td>-14</td>
<td>dBm</td>
<td>FIFO_THR_CLOSE_IN_RX=0. See more in DN010 [11]</td>
</tr>
<tr>
<td>Adjacent channel rejection</td>
<td>37</td>
<td>dB</td>
<td>Desired channel 3 dB above the sensitivity limit. 100 kHz channel spacing</td>
</tr>
<tr>
<td>Alternate channel rejection</td>
<td>37</td>
<td>dB</td>
<td>Desired channel 3 dB above the sensitivity limit. 100 kHz channel spacing</td>
</tr>
<tr>
<td>Image channel rejection, 868MHz</td>
<td>31</td>
<td>dB</td>
<td>IF frequency 152 kHz&lt;br&gt;Desired channel 3 dB above the sensitivity limit</td>
</tr>
</tbody>
</table>

Copied from CC1101, characterization of CC430 not finished yet.
## CC430 Radio – RX Path

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver sensitivity</td>
<td>-103</td>
<td>dBm</td>
<td></td>
</tr>
<tr>
<td>Saturation</td>
<td>-16</td>
<td>dBm</td>
<td>FIFOTHR_CLOSE_IN_RX=0. See more in DN010 [11]</td>
</tr>
<tr>
<td>Adjacent channel rejection</td>
<td>20</td>
<td>dB</td>
<td>Desired channel 3 dB above the sensitivity limit. 200 kHz channel spacing</td>
</tr>
<tr>
<td>Alternate channel rejection</td>
<td>30</td>
<td>dB</td>
<td>Desired channel 3 dB above the sensitivity limit. 200 kHz channel spacing</td>
</tr>
<tr>
<td>Image channel rejection, 868MHz</td>
<td>23</td>
<td>dB</td>
<td>IF frequency 152 kHz. Desired channel 3 dB above the sensitivity limit</td>
</tr>
</tbody>
</table>

Copied from CC1101, characterization of CC430 not finished yet.
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
CC430 Architecture Overview

DMA Controller
3 Channel

25MHz CPUXV2 incl. 16 Registers

EEM (S: 3+1)

JTAG Interface

Spy-Bi-Wire

RAM 2kB 1kB

Power Mgmt
LDO SVM/SVS Brownout

ADC12

CRC16

SYS Watchdog

Flash 32kB 16kB

Comp_B

ADC

REF Voltage Reference

I/O Ports P1/P2 2x8 I/Os

PA 1x16 I/Os

I/O Ports P3/P4 2x8 I/Os

PB 1x16 I/Os

I/O Ports P5 1x8 I/Os

Port Mapping Controller

Flash

RAM

CRC16

SYS Watchdog

MPY32

Timers

Timer0_A5

Timer1_A3

Timer2_A7

Timer3_A9

RTC_A

USCI_A0 (UART, IrDA, SPI)

USCI_B0 (SPI, I2C)

LCD_B 96 Segments 1.2.3.4 Mux

AES128 Security En-/Decryption

Packet Handler
Digital RSSI
Carrier Sense
POI / LQI
CCA

Sub-1GHz Radio
(CC1101)

CPU Interface

MODEM

Frequency Synthesizer

RF/ANALOG TX & RX

RF_P RF_N
Interface between MSP430 and CC1101

Test Mode “look a like mode”

Standard/Application Mode

RADIO CONTROL

SPI

FEC / INTERLEAVER

PACKET HANDLER

RXFIFO

TXFIFO

INTERFACE TO MCU

MSP430 Interface

RF_P
RF_N

RC OSC

BIAS

XOSC

RBIAS

RF_XIN

RF_XOUT

LNA

ADC

FREQ SYNTH

MODULATOR

DEMODULATOR

ADC

0

90

TXFIFO

RXFIFO

RC OSC

sclk

cen

si

so

MAB

MDB

ifclk

rw

byte

ms

Standard/Application Mode

Texas Instruments
RF1A Radio Interface
RF1A Interface

MSP430

RF IF Configuration Registers
RF1A Logical Channel Registers
RF1A Direct FIFO
RF1A Core Interrupt Vector

Radio Core

Instruction Set
Command Strobes
Read/Write Instructions
RX FIFO
TX FIFO

Event-driven Interrupts
Radio Core Registers

* Direct FIFO not implemented in PG0.6 *
MSP430 Radio Interface Logic

Instruction based radio controller
- Instructions are written byte- or word-wise into INSTR register
- Appropriate number of data (instruction parameters) are written into DIN register
- Writing INSTR register initiates status update at STAT register
- Writing DIN register initiates output data at DOUT register

Auto-Read feature simplifies the read accesses
- Saves dummy-writes
- Speeds up processing
- Simplifies usage

* Direct FIFO not implemented in PG0.6 *
RF1A Logical Channels

Figure 22-3. Logical Channels Between Radio Interface and Radio Core
RF1A interface format

- Byte-access registers can be used stand-alone OR grouped to form word-access registers
- Radio core > Big-Endian (LSB last) ; MSP430 > Little-Endian (MSB last)
  - By default, conversion from MSP430 to radio core is automatic ( → RF1AIFCTL0 )
Care must be taken that the exact number of bytes required by the radio are written.
Care must be taken that the exact number of bytes provided by the radio are read.
E.g. If only one byte is required for a write or provided to be read, do NOT use 16-bit access.
Code should be written such that error flags never occur.
RF1A Radio Core Instruction

8 bits Instruction

<table>
<thead>
<tr>
<th>R/W</th>
<th>B</th>
<th>A6 - A0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>6</td>
<td>5  4  3 2 1 0</td>
</tr>
<tr>
<td>R/W</td>
<td>Bit 7</td>
<td>Read or Write Access to Radio Core</td>
</tr>
<tr>
<td>B</td>
<td>Bit 6</td>
<td>Burst Access to Radio Core</td>
</tr>
<tr>
<td>B</td>
<td>Bits 5-0</td>
<td>Radio Configuration Registers Address Which address is between 0x00 and 0x29</td>
</tr>
</tbody>
</table>

8 bits Status Byte

<table>
<thead>
<tr>
<th>RF_RDYn</th>
<th>RF_STATEEx</th>
<th>FIFO_BYTES_AVAILx</th>
</tr>
</thead>
<tbody>
<tr>
<td>7  6  5 4 3 2 1 0</td>
<td>RF_RDYn</td>
<td>Bit 7</td>
</tr>
<tr>
<td>RF_STATEEx</td>
<td>Bit 6-4</td>
<td>State of the radio core state machine</td>
</tr>
<tr>
<td>FIFO_BYTES_AVAILx</td>
<td>Bits 3-0</td>
<td>Number of Bytes available in the RX FIFO or TX FIFO</td>
</tr>
</tbody>
</table>

Texas Instruments
RF1A Radio Core Command Strobes

Table 22-6, CC1101-Based Radio Core Instruction Set – Command Strobes

<table>
<thead>
<tr>
<th>Instruction Mnemonic</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFSTXON</td>
<td>[d011 0061]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: enable and calibrate frequency synthesizer (if MCGM0_FS_AUTOCAL = 1). If in RX (with CCA), go to a wait state where only the synthesizer is running (for quick RX/TX turnaround). Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SXOFF</td>
<td>[d011 0010]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: radio core to enter SLEEP state. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SCAL</td>
<td>[d011 0011]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: calibrate frequency synthesizer and turn it off. SCAL can be strobed from IDLE mode without setting manual calibration mode (MGSMD_FS_AUTOCAL = 0). Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>$RX</td>
<td>[d011 0100]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: enable RX. Perform calibration first if coming from IDLE and MCGM0_FS_AUTOCAL = 1. If in RX state and CCA is enabled, only go to TX if channel is clear. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>STX</td>
<td>[d011 0101]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: enable TX, if in IDLE state and perform calibration first if MCGM0_FS_AUTOCAL = 1. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SIDLE</td>
<td>[d011 0110]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: exit RX/TX, turn off frequency synthesizer, and exit WOR mode, if applicable. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SWOR</td>
<td>[d011 1200]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: start automatic RX polling sequence (Wake-on-Radio, WOR) as described in. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SWRST</td>
<td>[d011 1201]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: radio core to enter SLEEP state. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SWFD</td>
<td>[d011 1210]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: flush the RX FIFO buffer. Only issue SWFD in IDLE or RX_OVERFLOW states. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SFRX</td>
<td>[d011 1211]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: flush the TX FIFO buffer. Only issue SFRX in IDLE or TX_UNDERFLOW states. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SFTX</td>
<td>[d011 1100]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: reset WOR timer to Event1 value. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
<tr>
<td>SWORST</td>
<td>[d011 1101]</td>
<td>e&lt;s555 5555&gt;</td>
<td>Command strobe: reset WOR timer to Event1 value. Returns status byte with bytes available in TX FIFO when x=0 and with bytes available in RX FIFO when x=1.</td>
</tr>
</tbody>
</table>

*Bug in current rev*:
The RF1A core considers all strobe commands are treated as read-access, which will then only return RF1A RX FIFO status.

**Workaround:** Write to an unused register (such as 0x27/0x28) to read the RF1A TX FIFO status afterwards.

- **Single-Byte Instruction**
- **Initiate Internal Sequence**
- **13 Command Strobes**
  - **MSB = 0** -> Status Byte returns number bytes available in **TX FIFO**
  - **MSB = 1** -> Status Byte returns number of bytes available in **RX FIFO**
- **Burst bit (B) is always 0**
- **Executed immediately**
RF1A Radio Core Command Strobes

Reset RF1A Core
RF1AINSTRB = RF_SRES;
status = RF1ASTATB;

Go to IDLE State
RF1AINSTRB = RF_SIDLE;
status = RF1ASTATB;

Turn on RX Mode
RF1AINSTRB = RF_SRVX;
status = RF1ASTATB;

Turn off RX Mode
RF1AINSTRB = RF_SIDLE;
status = RF1ASTATB;
RF1AINSTRB = RF_SFRX;
status = RF1ASTATB;

Turn on TX Mode
(Transmit data in TX FIFO)
RF1AINSTRB = RF_STX;
status = RF1ASTATB;

NOTE:
Entry into a state != IDLE
or SLEEP requires
PMMVCORE >= 0x02.

All RF1A instructions (e.g., RF_STX) are defined in cc430x631x.h header file.
## RF1A Radio Core Access Instructions

**All in Interface Registers!**
- No Timing constraint – automatic synchronization
- Trigger with interrupt flags

### Table 22-7. CC1101-Based Radio Core Instruction Set

<table>
<thead>
<tr>
<th>Instruction Mnemonic</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Description</th>
</tr>
</thead>
</table>
| SNGLREGRD            | i[10aa aaaa] d[--- ---] | c[ssss ssss] d[dddd dddddd] | Read register at address [a] w i ≤ 0x2E  
  Returns status [s] and content register [d]. Status contains bytes available RX FIFO. |
| SNGLREGWR            | i[00aa aaaa] d[0000 dddddd] | a[aaa aaaa] d[ssss ssss] | Write data [a] into register at address [a] with a ≤ 0x2E.  
  Returns status [s]. Status contains bytes available in TX FIFO. |
| REGRD                | i[11aa aaaa] d[--- ---] | s[ssss ssss] d[dddd dddddd] | Read registers starting at address [a] with a ≤ 0x2E.  
  Returns status [s]. Status contains bytes available in TX FIFO. |
| REGWR                | i[01aa aaaa] d[ddddd dddddd] | s[ssss ssss] d[ssss ssss] | Write data [d] into registers starting at address [a] with a ≤ 0x2E.  
  Returns status [s]. Status contains bytes available in TX FIFO. |
  Returns status [s] and content of radio core status register [d] with 0x30 ≤ a ≤ 0x3D. Status contains bytes available in RX FIFO. |
| SNGLPATABRD          | i[1011 1110] d[--- ---] | s[ssss ssss] d[dddd dddddd] | Read single byte from table with power amplifier settings.  
  Returns status [s] and one byte of PA table [d]. Status contains bytes available in RX FIFO. |
  Returns status [s]. Status contains bytes available in TX FIFO. |
| PATABRD              | i[1111 1110] d[--- ---] | s[ssss ssss] d[dddd dddddd] | Read table with power amplifier settings.  
  Returns status [s] and content of table [d]. Status contains bytes available in RX FIFO. |
  Returns status [s]. Status contains bytes available in RX FIFO. |
| SNGLRXRD             | i[1011 1111] d[--- ---] | s[ssss ssss] d[dddd dddddd] | Read single byte from receive FIFO.  
  Returns status [s] and received byte [d]. Status contains bytes available in RX FIFO. |
  Returns status [s]. Status contains bytes available in RX FIFO. |
| RXFIFORD             | i[1111 1111] d[--- ---] | c[ssss ssss] d[dddd dddddd] | Read data from receive FIFO.  
  Returns status [s] and received bytes [d]. Status contains bytes available in RX FIFO. |
  Returns status [s]. Status contains bytes available in TX FIFO. |

### Inputs and Outputs

- **[x1x2aa aaaa]**  
  - **[ssss ssss]**  
  - **[dddd dddddd]**

- **x1** > R/W access
- **x2** > Burst access
- **a** > Instruction address bit
- **s** > Status bit
- **d** > Data bit
- **-** > Don’t care bit
RF1A Radio Core Access Instructions

\[ RF1AINSTR = \text{Instruction} + \text{Register Address} \]

Read Status Register

\[
\begin{align*}
RF1AINSTRB &= RF\_STATREGRD + RSSI; \\
RF1ADINB &= 0x00; \quad \text{// dummy write to initiate RF1ADOUTB} \\
\text{channel} &= RF1ADOUTB; \\
\end{align*}
\]

Read Single Register

\[
\begin{align*}
RF1AINSTRB &= RF\_SNGLEREGRD + CHANNR; \\
RF1ADINB &= 0x00; \quad \text{// dummy write to initiate RF1ADOUTB} \\
\text{channel} &= RF1ADOUTB; \\
\end{align*}
\]

Alternatively, RF1AINSTRB and RF1ADINB can be combined

\[
RF1AINSTRW = ((RF\_SNGLEREGRD + CHANNR) << 8) + 0x00; \\
\text{// 0x00 is still a “dummy write” to initiate RF1ADOUTB} \\
\text{channel} &= RF1ADOUTB; \\
\]

Write Single Register

\[
\begin{align*}
RF1AINSTRB &= RF\_SNGLREGWR + CHANNR ; \\
RF1ADINB &= \text{channel}; \\
\end{align*}
\]

Alternatively, RF1AINSTRB and RF1ADINB can be combined

\[
RF1AINSTRW = ((RF\_SNGLREGWR + CHANNR) << 8) ) + \text{channel};
\]
RF1A Radio Core Burst Access Instructions

Read Multiple Registers (Read Burst)
Radio core auto-increments the address to the next register

\[
\begin{align*}
RF1AINSTRB &= RF\_REGRD + IOCFG2; \quad \text{// address of the 1st register} \\
RF1ADINB &= 0x00; \quad \text{// dummy write to initiate RF1ADOUTB} \\
iocfg2 &= RF1ADOUTB; \quad \text{// read register IOCFG2} \\
RF1ADINB &= 0x00; \quad \text{// dummy write} \\
iocfg1 &= RF1ADOUTB; \quad \text{// read register IOCFG1 (next address)} \\
RF1ADINB &= 0x00; \quad \text{// dummy write} \\
iocfg0 &= RF1ADOUTB; \quad \text{// read register IOCFG0 (next address)} \\
&\ldots
\end{align*}
\]

Efficient Code using **For Loop**

\[
\begin{align*}
RF1AINSTRB &= RF\_REGRD + \text{STARTING\_REG\_ADDRESS;} \\
&\text{for (i=0; i< length; i++)} \quad \text{// read length bytes} \\
&\{ \\
&\quad RF1ADINB = 0x00; \quad \text{// dummy write} \\
&\quad \text{buffer[i] = RF1ADOUTB;} \quad \text{// write DOUTB to buffer} \\
&\}
\end{align*}
\]

All RF1A register definitions (e.g., IOCFG2) are defined in cc430x631x.h header file
Write Multiple Registers (Write Burst)
Radio core auto-increments the address to the next register

```c
RF1AINSTRB = RF_REGWR + MDMCFG4;
RF1ADINB = mdmcfg4; // write to modem configuration reg. 4
RF1ADINB = mdmcfg3; // write to modem configuration reg. 3
RF1ADINB = mdmcfg2; // write to modem configuration reg. 2
RF1ADINB = mdmcfg1; // write to modem configuration reg. 1
RF1ADINB = mdmcfg0; // write to modem configuration reg. 0
...
```

Using **For Loop**

```c
RF1AINSTRB = RF_REGWR + STARTING_REG_ADDRESS;
for (i=0; i< length; i++) // write length bytes to RF1A Registers
    RF1ADINB = buffer[i]; // write to configuration register
```
RF1A Radio Core Auto-Read

- 1-byte auto-read adds one byte to the output FIFO after reading it
- There are also word-access auto-read registers (e.g. - RF1ADOUTW2B)
  * 2-byte auto-reads add two bytes to the output FIFO after reading it
RF1A Radio Core Auto-Read Code

Read Status Register

\[
\text{RF1AINSTR1B} = \text{RF STATREGRD} + \text{RSSI};
\]
\[
\text{channel} = \text{RF1ADOUTB};
\]

Read Single Register

\[
\text{RF1AINSTR1B} = \text{RF_SNGLEREGRD} + \text{CHANNR};
\]
\[
\text{channel} = \text{RF1ADOUTB};
\]

Read Multiple Register

Burst mode

\[
\text{RF1AINSTR1B} = \text{RF_REGRD} + \text{MDMCFG4};
\]
\[
\text{mdmcfg4} = \text{RF1ADOUT1B}; \quad \text{// reads 1 byte MDMCFG4 & initiates 1 auto-read byte}
\]
\[
\text{mdmcfg3} = \text{RF1ADOUT2B}; \quad \text{// reads 1 byte MDMCFG3 & initiates 2 auto-read bytes}
\]
\[
\text{mdmcfg2} = \text{RF1ADOUT1W}; \quad \text{// reads 2 bytes MDMCFG2-1 & initiates 1 auto-read byte}
\]
\[
\text{mdmcfg0} = \text{RF1ADOUTB}; \quad \text{// reads 1 byte MDMCFG0 & no more auto-read}
\]

Read Multiple Register

Burst mode Using For Loop

\[
\text{RF1AINSTR1B} = \text{RF_REGRD} + \text{START_REG_ADDRESS};
\]
\[
\text{for (i=0; i<length-1; i++)} \quad \text{// read length-1 bytes using auto-read}
\]
\[
\text{buffer}[i] = \text{RF1ADOUT1B}; \quad \text{// reads 1 byte & initiates 1 auto-read byte}
\]
\[
\text{buffer[length-1]} = \text{RF1ADOUTB}; \quad \text{// reads 1 byte with no further auto-read}
\]

RF1AINSTR\textsuperscript{2W} = Instruction + Register Address

1 or 2 : number of bytes prepared for auto-read (next read)
B or W : number of bytes currently read

Texas Instruments
Radio Core Status Registers

<table>
<thead>
<tr>
<th>RF_RDYn</th>
<th>RF_STATEx</th>
<th>FIFO_BYTES_AVAILx</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF_RDYn</td>
<td>Bit 7</td>
<td>Radio core ready</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Radio core is ready. Crystal oscillator has stabilized.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Radio core is not ready. Crystal oscillator not stable.</td>
</tr>
<tr>
<td>RF_STATEx</td>
<td>Bits 6-4</td>
<td>State of the radio core main state machine.</td>
</tr>
<tr>
<td></td>
<td>000</td>
<td>IDLE Idle state. Also reported for some transitional states.</td>
</tr>
<tr>
<td></td>
<td>001</td>
<td>RX Receive mode</td>
</tr>
<tr>
<td></td>
<td>010</td>
<td>TX Transmit mode</td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>FSTXON Fast TX ready</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>CALIBRATE Frequency synthesizer calibration is running.</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td>SETTLING PLL is settling.</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>RX_OVERFLOW RX FIFO overflow</td>
</tr>
<tr>
<td></td>
<td>111</td>
<td>TX_UNDERFLOW TX FIFO underflow</td>
</tr>
</tbody>
</table>

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.

- 12 Status Registers
- Accessible via Status Read Commands
TXFIFO & RXFIFO Examples

<table>
<thead>
<tr>
<th>SNGLRXRD</th>
<th>i:[1011 1111]</th>
<th>s:[ssss ss]</th>
<th>d:[---- ----]</th>
<th>Read single byte from receive FIFO. Returns status [s] and received byte [d]. Status contains bytes available in RX FIFO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNGLTXWR</td>
<td>i:[0011 1111]</td>
<td>s:[ssss ss]</td>
<td>d:dddd dddd</td>
<td>Write single byte [d] into transmit FIFO. Returns status [s]. Status contains bytes available in TX FIFO.</td>
</tr>
<tr>
<td>RXFIFORD</td>
<td>i:[1111 1111]</td>
<td>s:[ssss ss]</td>
<td>d:dddd dddd</td>
<td>Read data from receive FIFO. Returns status [s] and received bytes [d]. Status contains bytes available in RX FIFO.</td>
</tr>
<tr>
<td>TXFIFOWR</td>
<td>i:[0111 1111]</td>
<td>s:[ssss ss]</td>
<td>d:dddd dddd</td>
<td>Write data bytes [d] into transmit FIFO. Returns status [s]. Status contains bytes available in TX FIFO.</td>
</tr>
</tbody>
</table>

Read RXFIFO

\[
\begin{align*}
  \text{RF1AINSTR1B} &= \text{RF\_SNGLREGRD} | \text{RXBYTES}; \\
  \text{length} &= \text{RF1ADOUTB}; \\
  \text{RF1AINSTR1B} &= \text{RF\_REGRD} | \text{RXFIFORD}; \quad \text{//RF\_REGRD optional (already included)} \\
  \text{for} \ (i=0; \ i<\text{length}-1; \ i++) & \quad \text{// read length-1 bytes using auto-read} \\
    \text{buffer}[i] &= \text{RF1ADOUT1B}; \quad \text{// read 1 byte & initiates 1 auto-read byte} \\
  \text{buffer}[\text{length}-1] &= \text{RF1ADOUTB}; \quad \text{// read 1 byte with no further auto-read}
\end{align*}
\]

Write TXFIFO

\[
\begin{align*}
  \text{RF1AINSTRB} &= \text{RF\_REGWR} | \text{TXFIFOWR}; \quad \text{//RF\_REGWR optional (already included)} \\
  \text{for} \ (i=0; \ i<\text{length}; \ i++) & \quad \text{// write length bytes to RF1A Registers} \\
    \text{RF1ADINB} &= \text{buffer}[i]; \quad \text{// write byte to TXFIFO}
\end{align*}
\]
CC430 RF Core

Inside the RF Radio Core
Radio Core State Chart

- IDLE: Idle state
- RX: Receive mode
- TX: Transmit mode
- FSTXON: Fast TX Ready
- Calibrate: Freq. Synth. is calibrating
- Settling: PLL is settling
- RXFIFO_OV: RX FIFO Overflow
- TXFIFO_OV: TX FIFO Overflow

* Section 3.9 in RF1A UG Chapter
# Radio Core Configuration Registers

## Table 22-19. Configuration Registers

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>REGISTER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>IOCFG32</td>
<td>GDO2 output configuration</td>
</tr>
<tr>
<td>0x01</td>
<td>IOCFG1</td>
<td>GDO1 output configuration</td>
</tr>
<tr>
<td>0x02</td>
<td>IOCFG0</td>
<td>GDO0 output configuration</td>
</tr>
<tr>
<td>0x03</td>
<td>FIFOTHR</td>
<td>RX FIFO and TX FIFO thresholds</td>
</tr>
<tr>
<td>0x04</td>
<td>SYNC1</td>
<td>Sync word, high byte</td>
</tr>
<tr>
<td>0x05</td>
<td>SYNC0</td>
<td>Sync word, low byte</td>
</tr>
<tr>
<td>0x06</td>
<td>PKTLEN</td>
<td>Packet length</td>
</tr>
<tr>
<td>0x07</td>
<td>PKTCTRL1</td>
<td>Packet automation control</td>
</tr>
<tr>
<td>0x08</td>
<td>PKTCTRL0</td>
<td>Packet automation control</td>
</tr>
<tr>
<td>0x09</td>
<td>ADDR</td>
<td>Device address</td>
</tr>
<tr>
<td>0x0A</td>
<td>CHANNR</td>
<td>Channel number</td>
</tr>
<tr>
<td>0x0B</td>
<td>FSCTRL1</td>
<td>Frequency synthesizer control</td>
</tr>
<tr>
<td>0x0C</td>
<td>FSCTRL0</td>
<td>Frequency synthesizer control</td>
</tr>
<tr>
<td>0x0D</td>
<td>FREQ2</td>
<td>Frequency control word, high byte</td>
</tr>
<tr>
<td>0x0E</td>
<td>FREQ1</td>
<td>Frequency control word, middle byte</td>
</tr>
<tr>
<td>0x0F</td>
<td>FREQ0</td>
<td>Frequency control word, low byte</td>
</tr>
<tr>
<td>0x10</td>
<td>MDMCF04</td>
<td>Modern configuration</td>
</tr>
<tr>
<td>0x11</td>
<td>MDMCFG3</td>
<td>Modern configuration</td>
</tr>
<tr>
<td>0x12</td>
<td>MDMCFG2</td>
<td>Modern configuration</td>
</tr>
<tr>
<td>0x13</td>
<td>MDMCFG1</td>
<td>Modern configuration</td>
</tr>
<tr>
<td>0x14</td>
<td>MDMCFG0</td>
<td>Modern configuration</td>
</tr>
<tr>
<td>0x15</td>
<td>DEVIATN</td>
<td>Modern deviation setting</td>
</tr>
<tr>
<td>0x16</td>
<td>MSCSM2</td>
<td>Main radio control state machine configuration</td>
</tr>
<tr>
<td>0x17</td>
<td>MSCSM1</td>
<td>Main radio control state machine configuration</td>
</tr>
<tr>
<td>0x18</td>
<td>MSCSM0</td>
<td>Main radio control state machine configuration</td>
</tr>
<tr>
<td>0x19</td>
<td>FOCCFG</td>
<td>Frequency offset compensation configuration</td>
</tr>
<tr>
<td>0x1A</td>
<td>BSCFG0</td>
<td>Bit synchronization configuration</td>
</tr>
<tr>
<td>0x1B</td>
<td>AGCCTRL2</td>
<td>AGC control</td>
</tr>
<tr>
<td>0x1C</td>
<td>AGCCTRL1</td>
<td>AGC control</td>
</tr>
<tr>
<td>0x1D</td>
<td>AGCCTRL0</td>
<td>AGC control</td>
</tr>
<tr>
<td>0x1E</td>
<td>WOREVT1</td>
<td>High byte Event 0 timeout</td>
</tr>
<tr>
<td>0x1F</td>
<td>WOREVT0</td>
<td>Low byte Event 0 timeout</td>
</tr>
<tr>
<td>0x20</td>
<td>WORCTRL</td>
<td>Wake-on-radio control</td>
</tr>
<tr>
<td>0x21</td>
<td>FREN0</td>
<td>Front-end RX configuration</td>
</tr>
<tr>
<td>0x22</td>
<td>FREN1</td>
<td>Front-end RX configuration</td>
</tr>
<tr>
<td>0x23</td>
<td>FSCAL3</td>
<td>Frequency synthesizer calibration</td>
</tr>
<tr>
<td>0x24</td>
<td>FSCAL2</td>
<td>Frequency synthesizer calibration</td>
</tr>
<tr>
<td>0x25</td>
<td>FSCAL1</td>
<td>Frequency synthesizer calibration</td>
</tr>
<tr>
<td>0x26</td>
<td>FSCAL0</td>
<td>Frequency synthesizer calibration</td>
</tr>
<tr>
<td>0x27</td>
<td></td>
<td>Reserved - read as '0'</td>
</tr>
<tr>
<td>0x28</td>
<td></td>
<td>Reserved - read as '0'</td>
</tr>
<tr>
<td>0x29</td>
<td>F3TEST</td>
<td>Frequency synthesizer calibration control</td>
</tr>
<tr>
<td>0x2A</td>
<td>PTTEST</td>
<td>Production test</td>
</tr>
<tr>
<td>0x2B</td>
<td>A0CTEST</td>
<td>AGC test</td>
</tr>
<tr>
<td>0x2C</td>
<td>TEST2</td>
<td>Various test settings</td>
</tr>
<tr>
<td>0x2D</td>
<td>TEST1</td>
<td>Various test settings</td>
</tr>
<tr>
<td>0x2E</td>
<td>TEST0</td>
<td>Various test settings</td>
</tr>
</tbody>
</table>

Address from 0x00 to 0x29
Data Rate Programming

• The TX & RX data rate is programmable from 1.2 to 500 kBaud using the following registers:
  – MDMCFG3.DRATE_M
  – MDMCFG4.DRATE_E

\[ R_{\text{DATA}} = \frac{(256 + \text{DRATE}_M) \times 2^{\text{DRATE}_E}}{2^{28}} \times f_{\text{XOSC}} \]

• Given a data rate, the following equations can be used to find suitable values for DRATE_M & DRATE_E:

\[ \text{DRATE}_E = \left\lfloor \log_2 \left( \frac{R_{\text{DATA}} \times 2^{20}}{f_{\text{XOSC}}} \right) \right\rfloor \]
\[ \text{DRATE}_M = \frac{R_{\text{DATA}} \times 2^{28}}{f_{\text{XOSC}} \times 2^{\text{DRATE}_E} \times 256} \]

NOTE: If DRATE_M is rounds to 256, increment DRATE_E & use DRATE_M = 0
Frequency Programming

• Desired channel spacing is programmed with:
  – MDMCFG0.CHANSPC_M (mantissa)
  – MDMCFG0.CHANSPC_E (exponent)
  – Maximum channel spacing for 26-MHz xtal is 405 kHz

\[ \Delta f_{\text{CHANNEL}} = \frac{f_{\text{OSC}}}{2^{16}} \times (256 + \text{CHANSPC}_M) \times 2^{\text{CHANSPC}_E} \]

• Base frequency – center of the lowest freq. chann.
  – FREQ2, FREQ1, FREQ0 registers

• Desired 8-bit channel number, CHANNR.CHAN, is multiplied by the channel offset to attain carrier frequency:

\[ f_{\text{carrier}} = \frac{f_{\text{OSC}}}{2^{16}} \times (\text{FREQ} + \text{CHAN} \times ((256 + \text{CHANSPC}_M) \times 2^{\text{CHANSPC}_E-2})) \]

• IF frequency
  – SMARTRF Studio calculated appropriate IF freq
  – Programmed into FSCTRL1.FREQ_IF
Receiver Channel Filter Bandwidth

- Receiver channel filter is programmable to match different channel width requirements
  - Best performance when: signal BW < 80% * channel BW
- Programmable using
  - MDMCFG4.CHANBW_E
  - MDMCFG4.CHANBW_M
- UG Example:
  - With the channel filter bandwidth set to 500 kHz, the signal should stay within 80% of 500 kHz, which is 400 kHz. Assuming 915-MHz frequency and ±20-ppm frequency uncertainty for both the transmitting device and the receiving device, the total frequency uncertainty is ±40 ppm of 915 MHz, which is ±37 kHz. If the whole transmitted signal bandwidth is to be received within 400 kHz, the transmitted signal bandwidth should be maximum 400 kHz – (2 x 37 kHz), which is 326 kHz.
Packet Handling Support

- Built-in support for packet handling protocol:
  - Programmable # of preamble bytes
  - 8, 16, or 32 sync word bits (programmable quality thresholds)
- Flexible packet length (Variable, fixed, infinite)
- Optional address byte and CRC-filtering
- Data whitening for improved sensitivity
### Preamble & Sync Bits

- Preamble and sync word qualifier mode is programmed using `MDMCFG2.SYNC_MODE`:

<table>
<thead>
<tr>
<th><code>MDMCFG2.SYNC_MODE</code></th>
<th>Sync Word Qualifier Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>No preamble/sync</td>
</tr>
<tr>
<td>001</td>
<td>15/16 sync word bits detected</td>
</tr>
<tr>
<td>010</td>
<td>16/16 sync word bits detected</td>
</tr>
<tr>
<td>011</td>
<td>30/32 sync word bits detected</td>
</tr>
<tr>
<td>100</td>
<td>No preamble/sync, carrier sense above threshold</td>
</tr>
<tr>
<td>101</td>
<td>15/16 + carrier sense above threshold</td>
</tr>
<tr>
<td>110</td>
<td>16/16 + carrier sense above threshold</td>
</tr>
<tr>
<td>111</td>
<td>30/32 + carrier sense above threshold</td>
</tr>
</tbody>
</table>

**Recommendation:**

If (Data Rate != 500 kbps)

{ 4-byte preamble + 4-byte sync word }

Else

{ 8-byte preamble + 4-byte sync word }

- The preamble provides bit-synchronization
  - Programmable length \( \rightarrow \) `MDMCFG1.NUM_PREAMBLE`
  - Preamble Quality Threshold \( \rightarrow \) `PKTCTRL1.PQT \times 4`

- The sync word provides byte-synchronization
  - Programmable value \( \rightarrow \) `SYNC1 & SYNC0`
  - Set `SYNC1` to preamble sequence to emulate a 1-byte sync word
  - Use `SYNC_MODE = \{3,7\}` to repeat sync word (4-byte sync word)
Packet Filtering Methods – Length

- Fixed packet length
  - \texttt{PKTCTRL0.LENGTH_CONFIG} = 0
  - Desired packet length set in \texttt{PKTLEN}

- Variable packet length
  - \texttt{PKTCTRL0.LENGTH_CONFIG} = 1
  - The length of incoming packets is configured by the first byte after the sync word
  - Maximum packet length set in \texttt{PKTLEN}

- Infinite packet length
  - \texttt{PKTCTRL0.LENGTH_CONFIG} = 2
  - Transmission and reception is continuous until manually turned off
  - See \texttt{CC430 User Guide} for suggested algorithm

If receive packet length > \texttt{PKTLEN}, Reset RX mode
Packet Filtering Methods - Address & CRC

- **Address Filtering**
  - Checks incoming address byte against contents of ADDR register
  - If ADDR check fails, packet is discarded & RX mode is restarted
  - The PKTCTRL1.ADR_CHK register sets the address filter mode:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Address Check Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (00)</td>
<td>No address check</td>
</tr>
<tr>
<td>1 (01)</td>
<td>Address check, no broadcast</td>
</tr>
<tr>
<td>2 (10)</td>
<td>Address check and 0 (0x00) broadcast</td>
</tr>
<tr>
<td>3 (11)</td>
<td>Address check and 0 (0x00) and 255 (0xFF) broadcast</td>
</tr>
</tbody>
</table>

- **CRC Filtering**
  - CRC-16 is executed on the entire payload following the sync word
  - Enabled by setting PKTCTRL1.CRC_AUTOFLUSH = 1
  - If CRC check fails, entire RX_FIFO is flushed, MCSCM1.RXOFF_MODE is entered (IDLE, FSTXON, TX, RX)
Modulation Formats – FSK variants

- **2-FSK** – A change in frequency identifies a zero or a one in the bit stream [1]. Deviation set in DEVIATN.DEVIATN_M / DEVIATN_E

- **GFSK** – Gaussian filter shapes input pulses previous to being modulated in order to reduce out-of-band noise

- **MSK** – Difference between higher and lower frequency is identical to half the bit rate


Modulation Formats – ASK Variants

• **ASK** –
  Amplitude Shift Keying –
  A change in amplitude identifies a zero or one in the bit stream

• **OOK** –
  On-off Keying –
  ‘Extreme’ ASK

• Modulation programmed in `MDMCFG2.MOD_FORMAT`

<table>
<thead>
<tr>
<th>Setting</th>
<th>Modulation Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (000)</td>
<td>2-FSK</td>
</tr>
<tr>
<td>1 (001)</td>
<td>GFSK</td>
</tr>
<tr>
<td>2 (010)</td>
<td>Reserved</td>
</tr>
<tr>
<td>3 (011)</td>
<td>ASK / OOK</td>
</tr>
<tr>
<td>4 (100)</td>
<td>Reserved</td>
</tr>
<tr>
<td>5 (101)</td>
<td>Reserved</td>
</tr>
<tr>
<td>6 (111)</td>
<td>Reserved</td>
</tr>
<tr>
<td>7 (111)</td>
<td>MSK</td>
</tr>
</tbody>
</table>
Output Power Programming

• Two levels of programmability
  – PATABLE can hold 8 x output power settings
  – 3-bit FREND0.PA_POWER value select PATABLE entry for use

• Provides:
  – Flexible PA power ramp up and ramp down at start & end of transmission, respectively
  – ASK modulation shaping

• When OOK modulation is used:
  – Logic 0 → PATABLE[0]
  – Logic 1 → PATABLE[1]

• All contents in PATABLE except PATABLE[0] erased when radio enters SLEEP mode.

• Output power directly & significantly affects current consumption!
Shaping and PA Ramping

- Up to 8 x PATABLE settings can be used for ASK modulation

- Counter rate = 8 x symbol rate
- Counter saturates at FRENDO.PA_POWER and 0
RSSI – Received Signal Strength Indicator

• Can be read continuously from RSSI status register until sync word detected
  – RSSI update rate is dependent on $BW_{\text{channel}}$ & AGCCTRL0.FILTER_LENGTH
  – When sync word is detected, RSSI value remains locked in until radio re-enters RX state

• RSSI value
  – Provided in 2’s complement
  – To attain absolute power level, must convert the value to decimal & subtract a typical offset value
    • Offset value is different for each baud rate & input power applied
CS – Carrier Sense Support

• Asserted on one of two conditions:
  – RSSI above a programmable absolute threshold (hysteresis included for deassertion)
    • Selectable with CARRIER_SENSE_ABS_THR
  – RSSI increase a programmable number of dB from one RSSI sample to the next
    • Useful for detecting signals with time-varying noise floor
      – Threshold selectable with AGCCTRL1.CARRIER_SENSE_REL_THR

• Used in TX-if-CCA and optional fast-RX termination algorithms
CCA, LQI & Packet Handling Support

- **CCA** – Clear Channel Assessment
  - Assertion criteria programmed in MCSM1.CCA_MODE

<table>
<thead>
<tr>
<th>Setting</th>
<th>Clear Channel Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (00)</td>
<td>Always</td>
</tr>
<tr>
<td>1 (01)</td>
<td>If RSSI below threshold</td>
</tr>
<tr>
<td>2 (10)</td>
<td>Unless currently receiving a packet</td>
</tr>
<tr>
<td>3 (11)</td>
<td>If RSSI below threshold unless currently receiving a packet</td>
</tr>
</tbody>
</table>

- **LQI** – Link Quality Indicator
  - Relative measure of link quality based on error between ideal signal & received signal
    - The higher the better
    - Affected by and not available for all modulation formats

- **RSSI, CRC, LQI** available as the last two bytes in RX_FIFO when PKTCTRL1.APPEND_STATUS = 1

  **RX pkt status byte 1:**

<table>
<thead>
<tr>
<th>BIT</th>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>7:0</td>
<td>RSSI</td>
<td>RSSI value</td>
</tr>
</tbody>
</table>

  **RX pkt status byte 2:**

<table>
<thead>
<tr>
<th>BIT</th>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>CRC_OK</td>
<td>1: CRC for received data OK (or CRC disabled) 0: CRC error in received data</td>
</tr>
<tr>
<td>6:0</td>
<td>LQI</td>
<td>Indicating the link quality</td>
</tr>
</tbody>
</table>
RF1A Interrupts

1. Radio Interface Interrupts
2. Radio Core Interrupts
3. Interrupt Handling + Example
RF1A Interrupts

RF1A Logical Channel Registers

RF1A Direct FIFO Registers *

CPU

Radio Core

Instruction Set

Command Strobes

Read/Write Instructions

Direct FIFO

TX FIFO

Conf. & Stat Registers

RX FIFO

Radio Core Interrupts

Interface Events

Radio IF Interrupt Vector

RF1A Core Interrupt Vector

RF1A/Radio Interface Interrupts

RF1A Core Interrupts

* Direct FIFO not implemented in PG0.6 *

Texas Instruments
Radio Interface Interrupts

**RF1AIFCTL1**: controls all Radio Interface Interrupt Enable (IEs) and Flags (IFG)

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFDOUTIE</td>
<td>RFSTATIE</td>
<td>RFDINIE</td>
<td>RFINSTRIE</td>
<td>Reserved</td>
<td>RFERRIE</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>r0</td>
<td>rw-0</td>
<td>r0</td>
<td>r0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFDOUTFIFG</td>
<td>RFSTATIFG</td>
<td>RFDINIFG</td>
<td>RFINSTRIFG</td>
<td>Reserved</td>
<td>RFERRIFG</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-1</td>
<td>r0</td>
<td>r-0</td>
<td>r0</td>
<td>r0</td>
</tr>
</tbody>
</table>

**Table 22-4. Radio Interface Interrupt Flags**

<table>
<thead>
<tr>
<th>Interrupt Flag</th>
<th>Interrupt Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFINSTRIEFG</td>
<td>The radio core is ready to accept the next instruction; i.e., the previous instruction was completely processed and all required data was provided.</td>
</tr>
<tr>
<td>RFDINIFG</td>
<td>The radio core is ready to accept additional data.</td>
</tr>
<tr>
<td>RFSTATIFG</td>
<td>The radio core updated the status accessible via the RF1ASTAT registers. If the instruction was provided as a word via RF1AINSTRW, the flag is set after the first data byte is also available.</td>
</tr>
<tr>
<td>RFDOUTIFG</td>
<td>Data was provided by the radio core and can be read via the RF1ADOUT registers. If the corresponding parameters were provided as 16-bit data, the flag is set only after 16-bit data is available. With the auto-read feature, the flag is set after the selected amount of data is available. With each read access of the RF1ADOUT registers, the flag is cleared; if there is still data available after the read access, the RFDOUTIFG is set again.</td>
</tr>
<tr>
<td>RFERRIFG</td>
<td>An error occurred interfacing to the radio core. The error condition can be encoded using the error flags. The error interrupt flag is set as long as one of the error flags (OPERR, OUTERR, OPOVERR, or LVERR) is set. It is cleared automatically when all error flags are cleared.</td>
</tr>
</tbody>
</table>

**RF1AIFIV**: Radio Interface Interrupt Vector Register
Radio Interface Interrupts - RFERR

RFERRIFG = 1

- OPERR = 1: Not enough or too many operands were provided for an instruction
- OUTERR = 1: Not enough data available for the executed read access
- OPOVERR = 1: Attempt to overwrite operands in RF1ADIN registers that are still being processed by the radio core. Written data is ignored
- LVERR = 1: Attempt to activate the radio core with a core voltage level PMMCCOREVx < 2

RF1AERR: contains the IFGs

RF1AERRV = error vector generator register to decode the error condition

<table>
<thead>
<tr>
<th>RF1AIFERRV Contents</th>
<th>Error Source</th>
<th>Error Flag</th>
<th>Error Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>00h</td>
<td>No error</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>02h</td>
<td>Low core voltage error</td>
<td>LVERR</td>
<td>Highest</td>
</tr>
<tr>
<td>04h</td>
<td>Operand error</td>
<td>OPERR</td>
<td></td>
</tr>
<tr>
<td>06h</td>
<td>Output data not available error</td>
<td>OUTERR</td>
<td></td>
</tr>
<tr>
<td>08h</td>
<td>Operand overwrite error</td>
<td>OPOVERR</td>
<td>Lowest</td>
</tr>
</tbody>
</table>
Radio Core Interrupts

- 15 interrupts including 3 programmable using IOCFGx
  - GDO0
  - GDO1
  - GDO2

- GDOx signals can be routed to HW output pins

- 4 associated bits for each GDOx interrupt signal:
  - **RFINx** query the actual status of a signal
  - **RFIESx** trigger an interrupt on the positive or negative edge
    - RFIESx = 0 > positive edge triggered
    - RFIESx = 1 > negative edge triggered
  - **RFIEx** enables the interrupt
  - **RFIFGx** is set when the interrupt occurs
# Radio Core Interrupts

## Table 22-5. CC1101 Radio Core Interrupt Mapping

<table>
<thead>
<tr>
<th>Interrupt Flag</th>
<th>Interrupt Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFIFG0</td>
<td>Based on GDO0 signal - programmable using IOCFG0 (0x02) register of radio core.</td>
</tr>
<tr>
<td>RFIFG1</td>
<td>Based on GDO1 signal - programmable using IOCFG1 (0x01) register of radio core.</td>
</tr>
<tr>
<td>RFIFG2</td>
<td>Based on GDO2 signal - programmable using IOCFG2 (0x00) register of radio core.</td>
</tr>
<tr>
<td>RFIFG3</td>
<td>Positive edge: RX FIFO filled or above the RX FIFO threshold. Negative edge: RX FIFO drained below RX FIFO threshold. (Equal to GDOx_CFG=0)</td>
</tr>
<tr>
<td>RFIFG4</td>
<td>Positive edge: RX FIFO filled or above the RX FIFO threshold or end of packet is reached. Negative edge: RX FIFO empty. (Equal to GDOx_CONFIG=1)</td>
</tr>
<tr>
<td>RFIFG5</td>
<td>Positive edge: TX FIFO filled or above the TX FIFO threshold. Negative edge: TX FIFO below TX FIFO threshold. (Equal to GDOx_CFG=2)</td>
</tr>
<tr>
<td>RFIFG6</td>
<td>Positive edge: TX FIFO full. Negative edge: TX FIFO below TX FIFO threshold. (Equal to GDOx_CFG=3)</td>
</tr>
<tr>
<td>RFIFG7</td>
<td>Positive edge: RX FIFO overflowed. Negative edge: RX FIFO flushed. (Equal to GDOx_CFG=4)</td>
</tr>
<tr>
<td>RFIFG8</td>
<td>Positive edge: TX FIFO underflowed. Negative edge: TX FIFO flushed. (Equal to GDOx_CFG=5)</td>
</tr>
<tr>
<td>RFIFG9</td>
<td>Positive edge: Sync word sent or received. Negative edge: End of packet or in RX when optional address check fails or RX FIFO overflows or in TX when TX FIFO underflows. (Equal to GDOx_CFG=6)</td>
</tr>
<tr>
<td>RFIFG10</td>
<td>Positive edge: Packet received with CRC OK. Negative edge: First byte read from RX FIFO. (Equal to GDOx_CFG=7)</td>
</tr>
<tr>
<td>RFIFG11</td>
<td>Positive edge: Preamble quality reached (PQI) is above programmed PQT value. Negative edge: (LPW) (Equal to GDOx_CFG=8)</td>
</tr>
<tr>
<td>RFIFG12</td>
<td>Positive edge: Clear channel assessment when RSSI level is below threshold (dependent on the current CCA_MODE setting). Negative edge: RSSI level is above threshold. (Equal to GDOx_CFG=9)</td>
</tr>
<tr>
<td>RFIFG13</td>
<td>Positive edge: Carrier sense. RSSI level is above threshold. Negative edge: RSSI level is below threshold. (Equal to GDOx_CFG=14)</td>
</tr>
<tr>
<td>RFIFG14</td>
<td>Positive edge: WOR event 0 Negative edge: WOR event 0 + 1 ACLK. (Equal to GDOx_CFG=36)</td>
</tr>
<tr>
<td>RFIFG15</td>
<td>Positive edge: WOR event 1 Negative edge: RF oscillator stable or next WOR event0 triggered. (Equal to GDOx_CFG=37)</td>
</tr>
</tbody>
</table>
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
CC1101-Based Radio Core Changes
CC1101 Digital Features not supported

- Forward Error Correction (FEC)
- Interleaving
- Wake-On-Radio (WOR)
- Direct TX & RX FIFO access
Changes

• Instructions can be executed in SLEEP mode
• ACLK is clock source for WOR
• SXOFF causes the radio to transition to SLEEP instead of XOFF state
• SFTX and SFRX can be issued in SLEEP state in addition to IDLE, TXFIFO_UNDERFLOW and RXFIFO_OVERFLOW
• In Register 0x18 MCSMO, bits 2 and 3 (used as PO_TIMEOUT in CC1101) are reserved
• In Register 0x01 IOCFG1 bit 7, GDO_DS is reserved
• Registers 0x27 RCCTRL1 an 0x28 RCCTRL0 are reserved
• Support of synch. and asynch. operation via Timer_A
• The register 0x30 PARTNUM – CHIP ID reads as 0 (0x00)
• The register 0x31 VERSION – CHIP ID reads as 6 (0x06)
• The following signals are added to the GDOx multiplexers: 30 (0x1E): RSSI_VALID
• 31 (0x1F): RX_TIMEOUT
• The RC_PD bit in register 0x20 WORCTRL is used as a ACLK_PD bit
• The PA_PD and LNA_PD signals behave as intended: PA_PD is low only in TX States, LNA_PD is low only in RX States
Introduction to CC430

1. Brief introduction to CC430
2. RF System Introduction
3. Block Diagram of the CC430F6137 & Peripherals
4. Derivatives of the CC430F61xx & CC430F51xx Family
5. RF1A Core & 6xx Core
6. RF1A Radio Interface
7. RF1A Changes vs. CC1101
8. Making RF Easy: Tools, Collateral and Support
Get to market fast

Software & Tools

Easy to use, low cost hardware and software tools get customers up and running fast:

RF support: RF reference designs, SmartRF Studio software, RF packet sniffer, design notes

Support

Extensive community of third party and academia technology solution providers

Comprehensive collateral, extensive application notes, code examples and libraries

Global customer support network – TI has the most feet on the ground

World wide training options ranging from online to hands on deep dive technical training
EM430F6137RF900
Chronos | Advanced Features at Your Disposal

- 3D Accelerometer
- Pressure & Altitude Sensor
- Temperature Sensor
- Voltage & Battery Sensor
- CR2032 Battery
- <1GHz RF
- CC430F6137 MCU
- 96 segment LCD
- Buzzer
- 2-Wire JTAG Access
- eZ430 Programmer
- RF Access Point
- Chronos Disassembly Tool
Application Notes – Antennae Design

- DN018 - Range Measurements in an Open Field Environment
- ISM-Band and Short Range Device Antennas (Rev. A)
- ISM-Band and Short Range Device Regulatory Compliance Overview
- DN002 -- Practical Sensitivity Testing
- AN058 – Antenna Selection Guide
- RSSI interpretation and timing
- SRD Antennas
- LC Filter with Improved High-Frequency Attenuation
- AN039 - Using CC1100/CC1150 in European 433/868 MHz bands
- AN50 -- Using the CC1101 in the European 868MHz SRD band
- Planned
  - Modifying the CC430 EVM to operate at 433 MHz
CC430 Software Tools

SMARTRF Studio
• Automatically configure your RF settings
• Test the RF front-end
  – Continuous RX / TX
  – Packet RX / TX
• No MSP430 code required!

SmartRF Packet Sniffer
• Analyze RF packets in real-time
• Greatly simplifies debugging
• Requires the SmartRF04 or SmartRF05 HW platform
• Supports SimpliciTI
RF Protocol Stacks

Some RF protocols have physical layer requirements
Some RF protocols are HW platform independent (SimpliciTI)
Timing/Tasks scheduling → Timer/Extra HW Resources from ‘430 Core
Topology, Range consideration, Regulation
CC430 Software Stack Solutions

Building & Home Automation
- WMBUS 868 MHz

Smart Metering

Lighting Control
- 6LoWPAN 868/915 MHz

Asset Tracking
- Dash7 433 MHz

Personal Health & Fitness
- 6LoWPAN 868/915 MHz
- BlueRobin™ 433/868/915 MHz

- Mesh / star network
- Point-to-point communication
6LoWPAN

- IPv6 over low-power wireless area networks
- Defined by IETF standards
- Highly efficient use of code and memory
- Direct end-to-end Internet integration
  - Multiple topology options

- Low-power RF + IPv6 = The Wireless Embedded Internet

- Benefits of 6LoWPAN include:
  - Open, long-lived, reliable standards
  - Easy learning curve
  - Transparent Internet integration
  - Network maintainability
  - Global scalability
  - End-to-end data flows

- 6LoWPAN enables a broad range of applications
  - Facility, building and home automation
  - Advanced metering infrastructure
  - Lighting Control

Source: 6LoWPAN: The Wireless Embedded Internet, Shelby & Bormann
WMBUS

• Use case: meter <> meter communication
  (Gas, Water Meter, Electricity Meter)

• Enables a simple star network topology that fits very well to the applications’ requirements.

• Sub 1 GHz band (868 MHz)
  • Quality and low power consumption of the transmission critical
    (Gas, Water meter, Battery Life)
Dash 7 Technology

• Ultra-Low Power, Low Bandwidth space
  • Range is scalable, 10 - 2000 m (1.2 mi)
  • 433 MHz
  • 28 kbps
• Based on the ISO 18000-7 standard
• Mandated by U.S. Department of Defense, allied militaries
• Extendable to multi-hop, sensors, security
• E.g. - Asset tracking, Tire pressure monitoring

<Add info regarding specialized HW platform>
SimpliciTI Networking Protocol

• **Key Features**
  – Low Power
  – Low Cost
  – Low Data Rate
  – Easy to learn

• **Device Configurations**
  – Access Point (AP)
  – Repeater (RE)
  – Sleeping End Device (SD)
  – End Device (D)

• **Topologies**
  – AP Star
  – AP Star w/ Repeaters
  – Peer2Peer
Support

- MSP430 Wiki
- Community: E2E Forum
  - http://community.ti.com/
  - CC430 Training Videos
- support@ti.com or your local TI contact
- www.ti.com/msp430
- www.ti.com/cc430
- www.ti.com/simpliciTI