Chipcon

CC1010 Code examples

Revision 1.4

Chipcon provide a range of code examples for the CC1010. They are written for execution on the Evaluation Board contained in the CC1010 Development Kit.

These examples are richly commented, and are described in detail in the file header of the main source code file of each project. The examples illustrate use of the HAL and CUL libraries, can serve as good starting points for custom applications, and serve to demonstrate features of the CC1010 and RF communication in general. Most of the examples can be compiled in the evaluation version of the Keil compiler; a few of the advanced examples can only be compiled in the full version due to code size. This is noted in the descriptions in the table below. Code size is for code compiled using full size optimisation in version 7.03 of the Keil C µVision 2 compiler.

Where libraries are used, there are different project files for the evaluation and the full version of the compiler. The files for the evaluation version have _eval appended, and the full version have _full appended to their file name. The evaluation version does not support libraries, so all HAL and CUL files are included in the project. In the full version project, library files are used.

Most of the RF examples have three versions available, set up for the 433 MHz, 868 MHz and 915 MHz bands. Switching between the various versions is done by changing targets in Keil µVision 2. The different versions are implemented by using conditional compilation in the pre-processor, so that different versions of the RF_RXTXPAIR_SETTINGS struct is used depending on the symbol defined (FREQ433, FREQ868, FREQ915).

Please note that all the RF examples except tempBroadcast require two Evaluation Boards (ie. two CC1010DK development kits) to do something useful (blink LEDs, write to the serial ports etc.) The tempBroadcast example can be run using only one kit, one of the CC1010EM modules can be used stand-alone, reporting the local temperature to the CC1010EB.

Name	Directory	Rev.	Description	Code size (bytes)
adc	adc	1.2	Demonstrates use of the A/D converter. The program reads the value of A/D channel 0, which is connected to the potmeter on the Evaluation Board. The value read is used to control the speed of a binary counter, which is output on the LEDs. Turning the pot will therefore control the speed at which the LEDs are blinking. Pressing switch 1 will halve the blinking speed.	321
clockmodes	clockmodes	1.2	Demonstrates the various clock modes. The program switches between the high-speed and 32 kHz oscillators. The red LED blinks when the 32 kHz oscillator is active, the green LED blinks when the high-speed oscillator is active.	604
des_1	des\des1	1.2	Demonstrates DES encryption and decryption using OFB and CFB modes. Some random data is encrypted with a random key. The encrypted data is then decrypted. If an error occurs, the red LED is lit. If both operations succeeded, the yellow and green LEDs are lit.	1316
des_2	des\des2	1.2	Demonstrates Triple-DES encryption and decryption using OFB and CFB modes. This example operates in exactly the same way as des_1, but uses Triple-DES instead of DES.	1316
flash	flash	1.2	Demonstrates writing to Flash using HAL functions. If successful, the yellow, blue and green LEDs will be lit. If there is an error, the red LED will be lit.	1597
halRFtest	rf\halRFtest	1.2	Demonstrates use of the HAL halRFSendPacket() and halRFReceivePacket() functions. Press any of the four buttons on the EB to transmit a short message. If a valid message is received, an acknowledge message is sent back. The blue LED is lit during transmit, the yellow LED is lit during receive, the green LED is lit if a valid message or acknowledge is received, the red LED is lit if no acknowledge is received after transmitting a message.	2074
int_ext	interrupts\ int_ext	1.2	Demonstrates the use of external interrupts. If the external interrupts are triggered by pushing buttons 2 and 3, the yellow and green LEDs are lit.	90
int_others	interrupts\ int_others	1.2	Demonstrates the use of all interrupts except the external interrupt and the timer interrupts. First, the ADC interrupt will trigger on a high value, turn the potensiometer knob to activate this. The red LED will be lit when this interrupt is triggered. Next, the DES interrupt is triggered, when this occurs the yellow LED is lit. The real-time clock interrupt will occur 10 seconds later, and will light up the green LED. Finally, a Flash page is written, and the Flash interrupt turns on the blue LED and turns off the others.	988
int_timers	interrupts\ int_timers	1.2	Demonstrates the use of timer interrupts. All of the four timers are set up to generate an interrupt, and each timer lights one of the LEDs once the interrupt occurs.	1582
ledSwitch	ledSwitch	1.2	Demonstrates use of the LEDs and switches. Pressing a switch turns off the corresponding LED.	66

Name	Directory	Rev.	Description	Code size (bytes)
powermodes	powermodes	1.2	Demonstrates the various power saving modes. Initially, the CC1010 is in the high-speed active mode. When switch 1 is pressed, the CC1010 enters idle mode. Timer 0 is started, and the green LED is lit when the timer interrupt is triggered. Pressing switch 2 will turn on the yellow LED and cause the CC1010 to enter power-down mode. Power-down mode can be exited by pressing the reset button, the program will then start from the beginning again.	1001
pwm	pwm	1.2	Demonstrates using the timers for PWM (pulse-width modulation). Timer 2 is configured for PWM operation. Pulse width is adjusted by turning the potmeter on the Evaluation Board Connect an oscilloscope to P3.4 to observe the PWM waveform.	948
rng	rng	1.2	Demonstrates the random number generator built into the CC1010. The program generates true random bytes and uses the four least significant bits to blink the LEDs.	563
rx_tx_switch	rf\rx_tx_switch	1.2	Simple application that lets the user do RF measurements even when the CC1010EM module is not plugged into the Evaluation Board.	753
spi	spi	1.2	Demonstrates SPI master functionality. The CC1010EB must be connected to an SPI EEPROM.	1876
chat	rf\chat	1.3	Demonstrates the use of the SPP (simple packet protocol) library. Link up to 16 CC1010EB boards together in a wireless chat network. The network consists of one server and up to 15 clients. Users can type broadcast messages, which are transmitted to all other members of the network. Also included is the possibility of sending encrypted, private messages to only one of the other participants. Please note that due to a bug in previous versions, version 5.0 or later of the HyperTerminal program is required. *Note: This program is too big to be compiled with the evaluation version of the Keil compiler.*	15472
rf232	rf\rf232	1.3	Demonstrates the use of the SPP (simple packet protocol) library. Two CC1010EB boards can be used as a wireless null-modem “cable” using this software. SPP supports error checking and automatic retransmission of failed packets. *Note: This program is too big to be compiled with the evaluation version of the Keil compiler.*	6252
sppTest	rf\sppTest	1.2	Demonstrates the use of the SPP (simple packet protocol) library. The user can push the buttons on the Evaluation Board to send different types of SPP messages. If the green LED is lit, it indicates a successful transmission/reception, if the yellow LED is lit, it indicates a timeout, and if the red LED is received, it indicates an invalid packet or an invalid acknowledge. *Note: This program is too big to be compiled with the evaluation version of the Keil compiler.*	4728

Name	Directory	Rev.	Description	Code size (bytes)
tempBroadcast	rf\tempBroadcast	1.3	Demonstrates the use of the SPP (simple packet protocol) library and the temperature sensor mounted on the CC1010EM module. Up to 16 modules can form a simple wireless network, exchanging temperature information. Multiple access functionality is performed by randomised time-division multiplexing. By plugging one of the CC1010EM modules into an Evaluation Board, the module can communicate with a PC via a serial port. For this to work, the terminal program on the PC should be configured to emulate a VT100 terminal. This example can be run with one EM module operating stand-alone, reporting the temperature to another EM plugged into the EB. *Note: This program is too big to be compiled with the evaluation version of the Keil compiler.*	12277
timer01	timer\01	1.2	Demonstrates use of Timer 0. An interrupt triggered by Timer 0 blinks the yellow and red LEDs every 60 ms.	957
timer23	timer\23	1.2	Demonstrates use of Timer 3. An interrupt triggered by Timer 3 blinks the yellow and red LEDs. Pressing switch 1 increases the timeout, pressing switch 2 decreases it.	773
uart1	uart\uart1	1.3	Demonstrates use of UART 0. A serial cable should be connected between UART 0 and a COM port on the PC. HyperTerminal should be run on the PC with a baud rate of 57600 bauds. The program will then echo all characters typed on the PC.	136
uart2	uart\uart2	1.3	Demonstrates the use of UART 0. A serial cable should be connected between UART 0 and a COM port on the PC. HyperTerminal should be run on the PC with a baud rate of 57600 bauds. The program will display a menu in the terminal program and allow the user to make various choices. The program can also record ASCII data into XRAM and play it back.	3641

Revision history:

1.0	2002-08-30	Initial release
1.1	2002-09-17	Added Flash example, made 433, 868, 915 MHz versions of RF examples. First public release.
1.2	2002-11-20	Added rf232 and chat examples. tempBroadcast modified so that it can run on stand-alone EM modules. All examples use the new HAL start-up macros.
1.3	2003-05-13	Corrections: 1. Corrected macro typo (TX interrupt check) in UART0 interrupt service routine. 2. Corrected all RF applications for compatibility with new version of CUL: Avoid using sppSetupRF to recalibrate because it reset the custom callbacks. Removed timer initialization from sppSetupRF (including the clkFreq argument) sppStartTimer(...) and SPP_INIT_TIMEOUTS() must now be called manually at startup 3. Modified RF232: Disabled timer 3 interrupt in UART0_ISR (Caused the program to hang on TX_BUFFER_LIMIT) Removed SPP timer function calls after recalibration (sppSetupRF) 4. tempBroadcast corrected so that a name change will take effect immediately.
1.3	2003-06-27	Synchronised CUL library with latest HAL library update + adjusted/tuned LOCK monitor in halRFSetRXTXOff.