

User Manual
Rev. 1.3
CC2420DBK Demonstration Board Kit

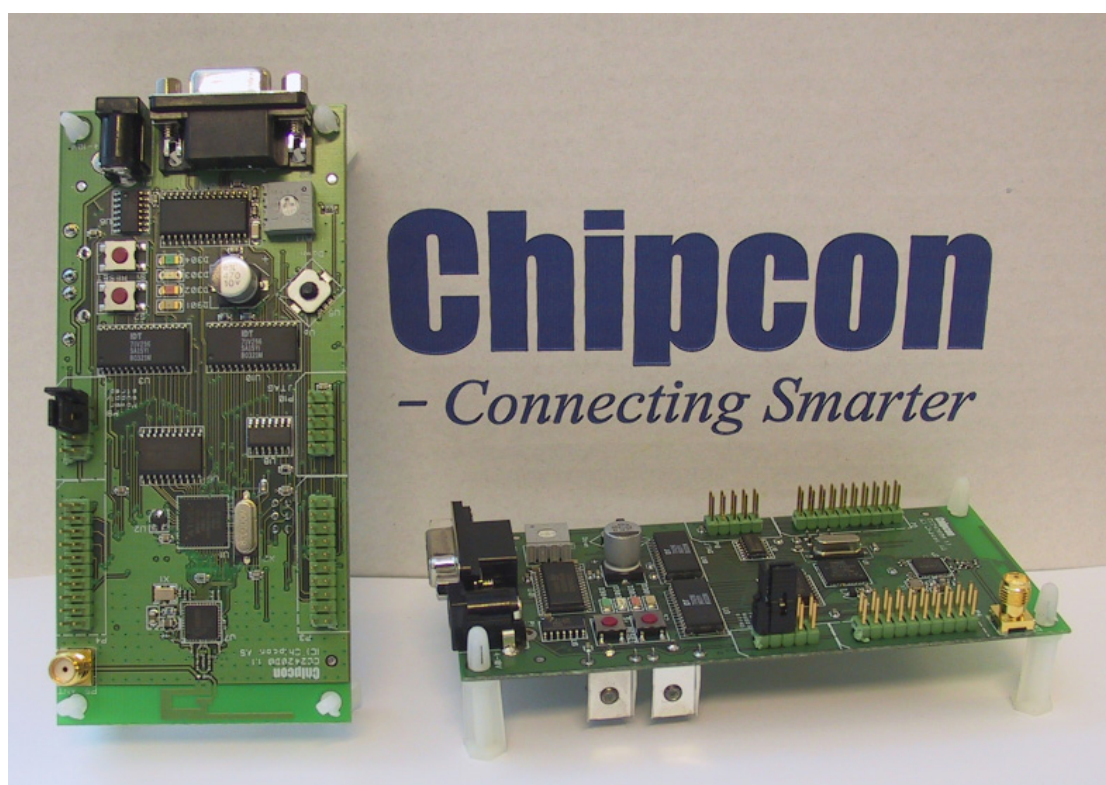


Table of contents

INTRODUCTION	3
PCB	4
ANTENNA	5
POWER SUPPLY SECTION	7
RS-232 INTERFACE	8
MICROCONTROLLER AND USER INTERFACE	8
EXTERNAL RAM	8
FLASH MEMORY REQUIREMENTS	8
CC2420DB LOW POWER MODE SUPPORT	8
JUMPER SETTINGS FOR DEMONSTRATION BOARD CONTROL	9
CC2420DB CURRENT MEASUREMENT AND CONSUMPTION	11
PCB LAYOUT	16
BILL OF MATERIALS	25
USING THE CC2420DB DEMONSTRATION BOARD	30
CC2420 SOFTWARE DOWNLOAD AND EVALUATION USING AVR STUDIO 4	30
PREPROGRAMMED 64-BIT IEEE ADDRESS	30
PREPROGRAMMED FUSES	31
PREPROGRAMMED LOCK BITS	33
JTAG ICE OR JTAG ICE MKII	33
SAVING THE IEEE ADDRESS OF THE CC2420DB	34
RESTORING THE IEEE ADDRESS OF THE CC2420DB	36
A NEW DEBUGGING PROJECT WITH CC2420DB	38
BOOTLOADER	43
INTRODUCTION TO EXAMPLES PROVIDED WITH CC2420DB	45
LIBRARIES AND EXAMPLES	45
HARDWARE DEFINITION FILES	45
HARDWARE ABSTRACTION LIBRARY (HAL)	45
BASIC RF LIBRARY	46
SOFTWARE EXAMPLE PROGRAMMED ON CC2420DB	46
DATA PACKET DESCRIPTION	47
DATA TRANSMISSION PROTOCOL	48
TROUBLESHOOTING	49
IT DOES NOT WORK	49
I CANNOT PROGRAM THE AVR USING THE SERIAL PORT	49
BOOT RESET VECTOR ENABLE	49
BOOTLOADER	49
REFERENCES	50
ADDRESS INFORMATION	51

Introduction

The CC2420 is a single-chip IEEE 802.15.4 compliant and ZigBee™ ready RF transceiver. It provides a highly integrated, flexible low-cost solution for applications using the world wide unlicensed 2.4 GHz frequency band. The CC2420DBK demonstration board kit is a complement to the development kit (DK) as the hardware is representative of an actual application, and it is well suited as a prototyping platform for application code.

The CC2420DBK Demonstration Board Kit includes two CC2420DB Demonstration Boards. These boards contain a CC2420 with necessary support components, an Atmel Atmega128L AVR microcontroller, 32 kBytes external RAM, a PCB antenna, as well as a joystick, buttons and LED's that can be used to implement a visual user application interface. The demonstration board is also furnished with connectors where all of the internal signals on the PCB are available.

This User Manual describes how to use the CC2420DBK Demonstration Board Kit. Atmel's AVR Studio and associated software is used to program and debug software. Please see the reference section of this document for links to Atmel's documentation on how to use their tools.

Your CC2420DBK Demonstration Board Kit should contain the following items:

Kit contents	
<i>Item</i>	<i>Number of articles</i>
Demonstration Board (CC2420DB)	2
Quick Start instructions	1
CC2420DB Important Notice (From revision 1.5 and newer)	1
CC2420 sample kit	1
RS-232 cable	2

Important:

Contact your local telecommunication authorities before transmitting an RF signal to ensure that there are no local restrictions on the use of the 2.400 –2483.5 ISM band. The CC2420 operates in the 2.4 GHz frequency band. Although this frequency band is usually described as “world-wide”, some countries do not allow unlicensed operation in this band.

PCB

PCB depicts an overview of the CC2420DB with description of the various components embedded on the PCB.

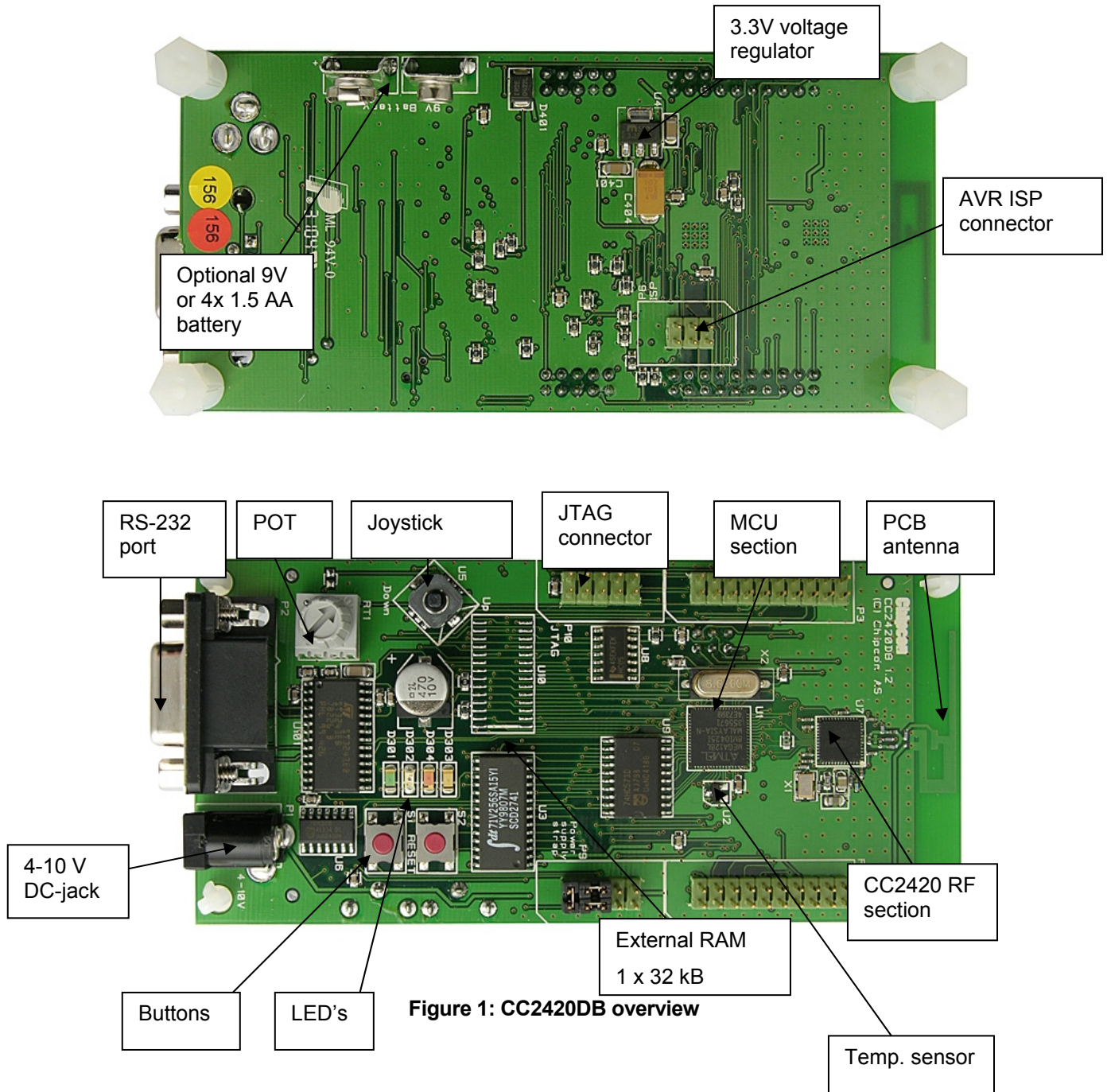


Figure 1: CC2420DB overview

RF Section

The CC2420DB RF section includes all the necessary components for correct operation. The CC2420 is connected to a 16 MHz crystal. A small 2.4 GHz PCB antenna is also included. RF test and measurement equipment can be connected to the SMA on the PCB, by swapping

capacitor placement of C63 and C62. See schematics for details. The RF layout is identical to the CC2420EM, which is part of the CC2420DK Development Kit.

Antenna

The PCB antenna is a so-called Inverted-F type. The Inverted-F antenna is a wire monopole where the top section is folded down to be parallel with the ground plane. By folding the antenna down you will reduce the height and maintain a resonant wire length. A capacitance will be introduced to the input impedance of the antenna due to the parallel section. However, as a rule of thumb design guide, the length + height (L+H) will be approximately equal to a quarter wavelength ($\lambda/4$). A reduction of the antenna height (H) will in general decrease the antenna bandwidth. Please see Figure 2 for the antenna dimensions.

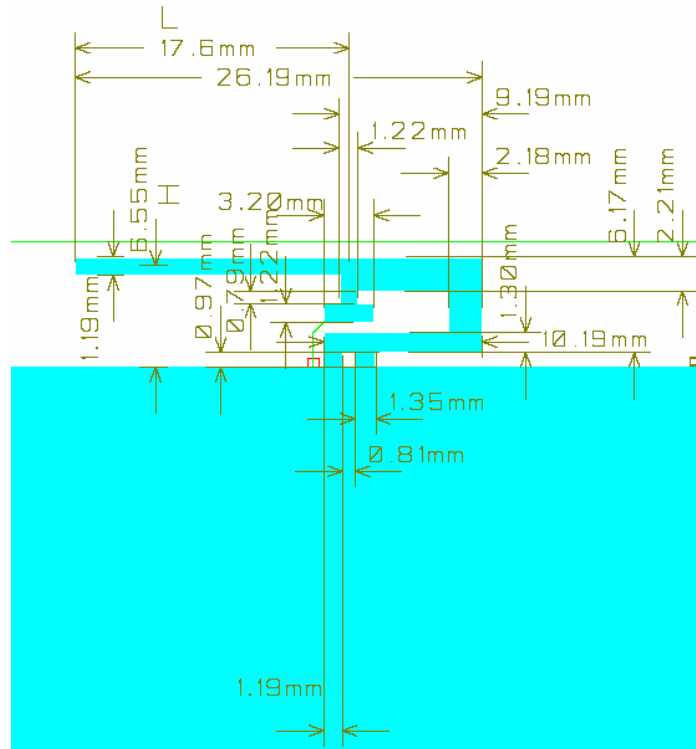


Figure 2: Antenna dimensions

The height of the antenna is defined as the distance from the ground plane to the parallel section, and the length is from the end of the antenna parallel section to the feed point.

Length: $L = (26.19 \text{ mm} - 9.19 \text{ mm} + 1.22 \text{ mm} \times \frac{1}{2}) = 17.61 \text{ mm}$

Height: $H = (6.17 \text{ mm} + 0.97 \text{ mm} - 1.19 \text{ mm} \times \frac{1}{2}) = 6.55 \text{ mm}$

Quarter wavelength: $\lambda/4 = (L + H) = 17.61 \text{ mm} + 6.55 \text{ mm} = 24.16 \text{ mm}$

The physical size of the antenna is decreased somewhat compared to the theoretical length, ($\lambda = c/4 \cdot f = 3 \times 10^8 / 4 \times 2.45 \times 10^9$) = 30.61 mm

The radiated pattern for the PCB antenna was measured with the horizontal polarization direction for the CC2420DB by measuring with vertical and horizontal mounting of the demonstration board.

Important:

In practical range testing outdoors with line-of-sight (LOS) and use of the wireless dimmer application example, the following range has been verified at Chipcon Norway with optimal SmartRF Studio settings for the respective data rates:

1. Data rate settings 250 kbps, range measured: 170 meters

Please note that these range tests were performed with a simple packet protocol, no link margin, no robust protocol, close to the sensitivity limit of CC2420, and no retransmission.

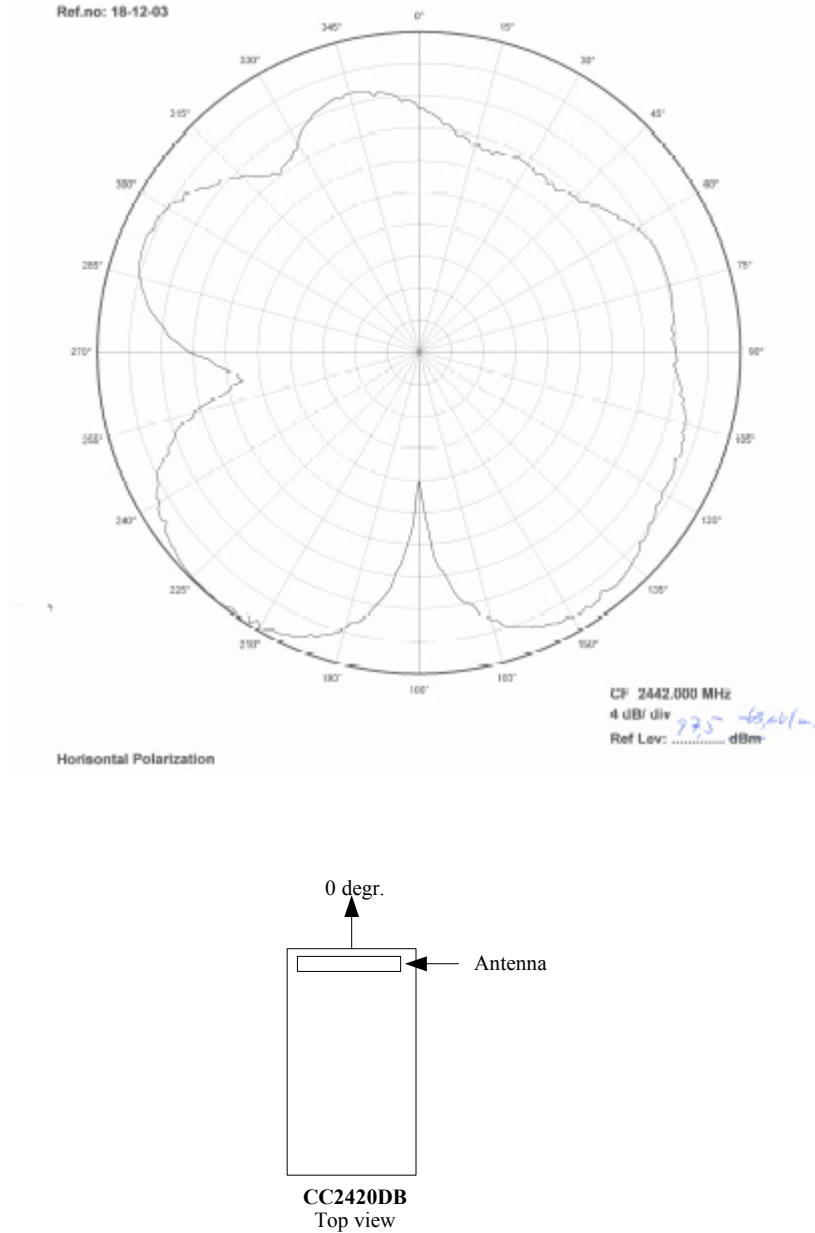


Figure 3: Radiated pattern horizontal mounting

Figure 3 depicts the antenna pattern while the CC2420DB is mounted horizontally with the antennas parallel section aligned to the 0 degree direction.

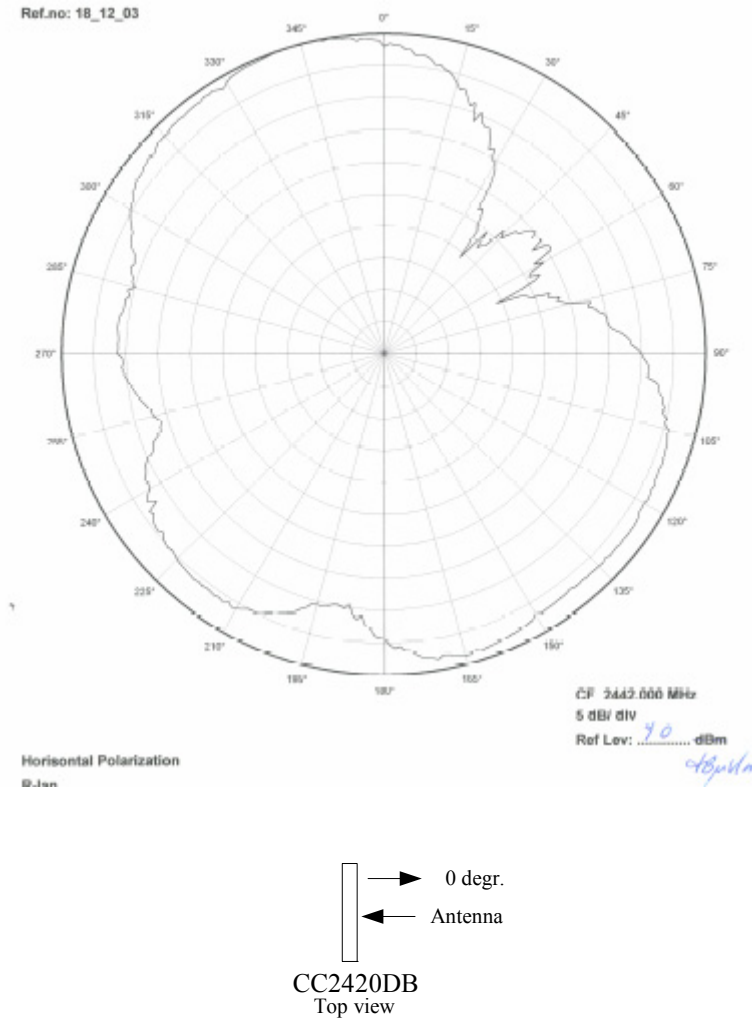


Figure 4: Radiated antenna pattern vertical mounting

Figure 4 depicts the antenna pattern while the CC2420DB is mounted vertically with the antennas parallel section aligned to the 0 degree direction.

The peak antenna gain is -5 dBi, the corresponding peak field strength is 90dBuV/m.

Power supply section

The power supply section contains two voltage regulators: a 3.3 V regulator for use by the microcontroller and the I/O pins of the CC2420. The internal regulator of CC2420 is used to generate the 1.8 voltage supply applied for powering of the CC2420 core.

A diode prevents permanent damage if wrong polarity is applied to the board. There are two power connectors; a 2.5mm DC jack-type connector allows you to connect an unregulated battery eliminator easily (the positive supply is on the center pin), and a connector for a 9V battery on the bottoms side of the PCB. It is also possible to use 4 AA or AAA alkaline cells to power the CC2420DB if a suitable battery pack is used.

RS-232 interface

A serial port is included on the CC2420DB. This port is used when software is programmed into the AVR MCU using the bootloader, and is also used by several of the example programs. The port includes support for RTS/CTS-type hardware flow control (handshaking).

Microcontroller and user interface

The microcontroller used is an AVR Atmega128L from Atmel. This controller has 128 KB of Flash program memory, 4 KB of SRAM data memory and 4 KB of non-volatile EEPROM data memory. The controller is interfaced to the CC2420 via its built-in SPI interface as well as some general I/O pins.

The MCU is also connected to four LED's, a joystick and an extra button for user interface purposes. The different examples use these peripherals differently. An analog temperature sensor, a potentiometer and 64K of external RAM are also included.

An ISP and a JTAG ICE connector are provided for programming the AVR without using the serial port. In this case an Atmel AVR JTAG or ISP programmer should be connected to their respective connectors.

All of the I/O pins are connected to footprints for 2 x 10 pin-row connectors. These connectors are compatible with Agilent logic analyzer probes, and can be used either for testing or for prototyping. For instance, it is possible to add a daughter board with additional circuitry using these connectors.

External RAM

The lower 4 Kbytes of the external RAM is overlaid by the Atmega128L internal register and RAM. CC2420DB includes this memory because it can be used for debugging purposes, to buffer and store data if desired. The 32 kB RAM size is not a requirement nor necessary for a low cost a Full Functional Device (FFD) or Reduced Functional Device (RFD). Please refer to the Atmega128L data sheet for more details.

Flash Memory Requirements

The flash requirement for a FFD device with the Chipcon MAC will be approximate 20 kBytes and 10 - 16 kBytes for a RFD. Additional flash size must be available for the application.

Important:

The memory requirement will be dependant on compiler, optimization level, hardware platform and feature used in the stack.

CC2420DB Low Power Mode support

To apply a low power mode with CC2420DB for application development, it will be necessary to apply an additional clock source together with the 8 MHz crystal. The Atmega128L provides the Timer/Counter oscillator pins (TOSC1 and TOSC2), and these pins are available on the CC2420DB connector P3. The oscillator is optimized for use with a 32.768 kHz crystal. This clock source can be connected the following ways.

1. A crystal can be connected directly between the pins.
2. An external clock source can be applied to the TOSC1. Atmel does not recommend this method.

The 32.768 kHz crystal will clock the timer 0 and then the overflow or compare match timer interrupt is applied to wake up Atmega128 from power save mode.

Name	Connector P3	Atmega128
TOSC1 (PG4)	Pin 9	Pin 19
TOSC2 (PG3)	Pin 7	Pin 18

Table 1, Low power mode signal

To demonstrate an application using the power down mode:

1. Apply 4x 1.5 AA batteries with a battery clip to the battery connector.
2. The library macros will be made available on how to set the ATmega128L into the various power mode that can be applied to minimize the current draw on CC2420DB.
3. Apply the joystick switch as the external interrupt to wake up the ATmega128L to idle mode and run an application.

CC2420DB temperature range

The CC2420DB can support a temperature range of -30°C to $+85^{\circ}\text{C}$ limited by the crystal tolerance and the CC2420 temperature range.

Jumper settings for demonstration board control

The CC2420DB demonstration board has three default jumper settings for proper operation. Figure 5 depicts the placement of the three jumpers J1, J2 and J3. They are mounted on connector P9. Please see CC2420DB schematics page 1 for more details.

Jumper	Purpose	Comments
J1	To apply power of 3.3 volt to the circuitry on the PCB board	An option can be to remove the jumper and connect an ampere meter in series to measure current consumption
J2	To apply power of 3.3 volt to the potentiometer and temperature sensor on the PCB board	Can be removed to disconnect the devices if not used, to decrease current consumption
J3	To always have the RS-232 driver enabled, hence the driver is forced on	This jumper can be disconnected and instead controlled by software if desired

Table 2 Jumpers explanation

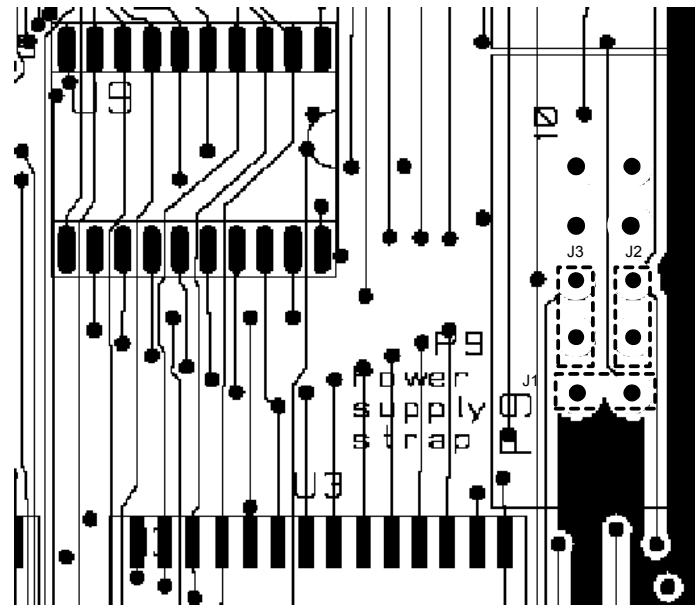


Figure 5 Jumpers

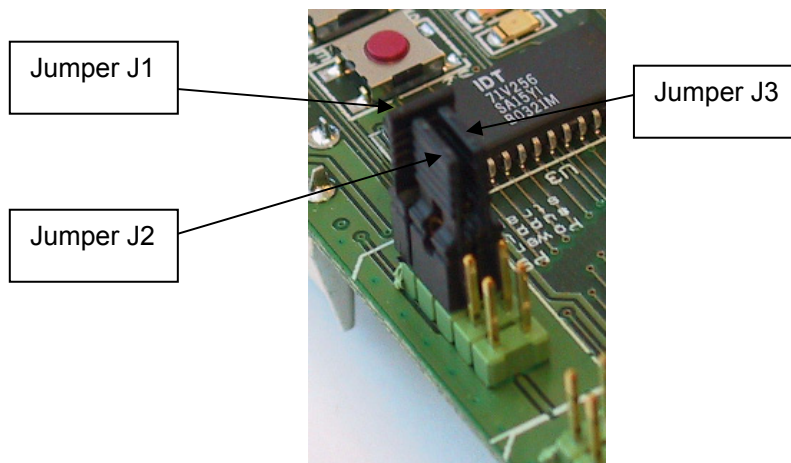


Figure 6 Jumpers on CC2420DB

To enable software control of the RS-232 driver U101, the jumper J3 can be moved.

Step-By-Step procedure:

1. Move RS-232 jumper J3 from P9 to P3 between pin 3 and 4
2. It will share the same port pin PE4 as the yellow LED, so when RS-232 is on the LED will be set.

CC2420DB Current Measurement and Consumption

It is fully possible to measure the actual current consumption with CC2420DB. To measure the current jumper J1 must be removed and an ampere meter connected in series. The current measurement is performed for the current total draw of the CC2420DB.

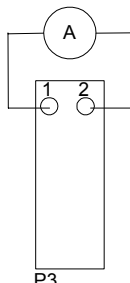


Figure 7: Connector P3 with ampere meter attached

To help minimize the current consumption of the CC2420DB for a specific IEEE 802.15.4 devices the default jumper J3 can be moved to enable software control of the RS-232 driver.

The external RAM will always draw 2mA in any power down mode since its power supply cannot be disabled. Hence, if the external RAM (U3) is removed the current consumption will decrease with 2mA.

The jumper J2 is controlling the power for the potentiometer (RT1) and temperature sensor (U2), which will also contribute to the total current consumption in any power down mode.

AVR I/O pins			
AVR Pin Number	AVR pin name	Pin usage	I/O connector
1	PEN	N.C.	N/A
2	PE0 RXD0/(PDI)	Joystick up / ISP MOSI	P6 pin 4, P3 pin 6
3	PE1 (TXD0/PDO)	Joystick right / ISP MISO	P6 pin 1, P3 pin 8
4	PE2 (XCK0/AIN0)	Joystick centre push button	P3 pin 5
5	PE3 (OC3A/AIN1)	Red LED	P3 pin 11
6	PE4 (OC3B/INT4)	Yellow LED	P3 pin 4
7	PE5 (OC3C/INT5)	Switch S2	P3 pin 10
8	PE6 (T3/INT6)	Interrupt from joystick	P3 pin 12
9	PE7 (IC3/INT7)	General purpose I/O	P3 pin 14
10	PB0 (SS)	SPI Interface, CSn, CC2420	P4 pin 13
11	PB1 (SCK)	SPI Interface, SCLK, CC2420	P6 pin 3, P4 pin 4
12	PB2 (MOSI)	SPI Interface, SI, CC2420	P4 pin 8
13	PB3 (MISO)	SPI Interface, SO, CC2420	P4 pin 6
14	PB4 (OC0)	Orange LED	P3 pin 18
15	PB5 (OC1A)	VREG_EN, Enable volt regulator on CC2420	P4 pin 10
16	PB6 (OC1B)	RESETn, CC2420	P4 pin 16
17	PB7 (OC2/OC1C)	Green LED	P3 pin 16
18	PG3/TOSC2	External 32 kHz crystal / General purpose I/O	P3 pin 7
19	PG4/TOSC1	External 32 kHz crystal / General purpose I/O	P3 pin 9
20	RESET	Reset: Switch S2, ISP, JTAG	P4 pin 3, P6 pin 5, P10 pin 6
21	VCC	Supply power	N/A
22	GND	Ground	N/A
23	XTAL2	Crystal Oscillator	N/A
24	XTAL1	Crystal Oscillator	N/A
25	PD0 (SCL/INT0)	FIFOP interrupt from CC2420	P4 pin 14
26	PD1 (SDA/INT1)	FIFO interrupt from CC2420	P4 pin 12
27	PD2 (RXD1/INT2)	UART 1, RS232, RXD1	P3 pin 15
28	PD3 (TXD1/INT3)	UART 1, RS232, TXD1	P3 pin 19
29	PD4 (IC1)	Polling SFD on CC2420	P4 pin 19
30	PD5 (XCK1)	UART 1, RS232, RTS	P3 pin 13
31	PD6 (T1)	Polling CCA on CC2420	P4 pin 18

AVR I/O pins			
AVR Pin Number	AVR pin name	Pin usage	I/O connector
32	PD7 (T2)	UART 1, RS232, CTS	P3 pin 17
33	PG0 (WR)	WR_N, Write RAM	N/A
34	PG1 (RD)	RD_N, Read RAM	N/A
35	PC0 (A8)	RAM address	N/A
36	PC1 (A9)	RAM address	N/A
37	PC2 (A10)	RAM address	N/A
38	PC3 (A11)	RAM address	N/A
39	PC4 (A12)	RAM address	N/A
40	PC5 (A13)	RAM address	N/A
41	PC6 (A14)	RAM address	N/A
42	PC7 (A15)	Select RAM, (upper / lower 32K)	N/A
43	PG2 (ALE)	Latch enable	N/A
44	PA7 (AD7)	RAM address + Data	N/A
45	PA6 (AD6)	RAM address + Data	N/A
46	PA5 (AD5)	RAM address + Data	N/A
47	PA4 (AD4)	RAM address + Data	N/A
48	PA3 (AD3)	RAM address + Data	N/A
49	PA2 (AD2)	RAM address + Data	N/A
50	PA1 (AD1)	RAM address + Data	N/A
51	PA0 (AD0)	RAM address + Data	N/A
52	VCC	Supply power	N/A
53	GND	Ground	N/A
54	PF7 (ADC7/TDI)	JTAG interface	P10 pin 9
55	PF6 (ADC6/TDO)	JTAG interface	P10 pin 3
56	PF5 (ADC5/TMS)	JTAG interface	P10 pin 5
57	PF4 (ADC4/TCK)	JTAG interface	P10 pin 1
58	PF3 (ADC3)	Temperature sensor (ADC)	P4 pin 9
59	PF2 (ADC2)	Joystick down	P4 pin 7
60	PF1 (ADC1)	Joystick left	P4 pin 5
61	PF0 (ADC0)	Potentiometer (ADC)	P4 pin 11
62	AREF	Reference voltage for ADC	P4 pin 2
63	GND	Ground	N/A
64	AVCC	ADC voltage ref	N/A

Table 3: AVR I/O pins

P3 pinout			
Pin number	Signal name in schematic	Pin usage	AVR pin
1	N.C.	N/A	N/A
2		Unregulated supply voltage (Insert R101 to connect)	
3	FORCE_ON	RS-232 on/off	05 PE3
4	PE4	Yellow LED	06 PE4
5	PE2	Joystick centre push button	04 PE2
6	PE0	ISP MOSI / Joystick up	02 PE0
7	PG3	External 32 kHz crystal / General purpose I/O	18 PG3
8	PE1	ISP MISO / Joystick right	03 PE1
9	PG4	External 32 kHz crystal / General purpose I/O	19 PG4
10	PE5	Push button S2	07 PE5
11	PE3	Red LED	N/A
12	PE6	Joystick interrupt	08 PE6
13	RTS	RS-232 handshaking	30 PD5
14	PE7	External interrupt / General purpose I/O	09 PE7
15	RXD1	RS-232 data from PC	27 PD2
16	PB7	Green LED	17 PB7
17	CTS	RS-232 handshaking	32 PD7
18	PB4	Orange LED	14 PB4
19	TXD1	RS-232 data to PC	28 PD3
20	GND	Ground	N/A

Table 4: P3 pinout

P4 pinout				
Pin number	Signal name in schematic	Pin usage	CC2420 pin	AVR pin
1	N.C.	N/A	N/A	N/A
2	AREF	Voltage reference for A/D Converter.	N/A	62
3	RESET	Reset	N/A	20
4	SCLK	CC2420 SPI clock, AVR ISP	32 (SCLK)	11
5	PF1	Joystick left	N/A	60
6	SO	CC2420 SPI out	34 (SO)	13
7	PF2	Joystick	N/A	59
8	SI	CC2420 SPI in	33 (SI)	12
9	PF3	Temperature sensor	N/A	58
10	VREG_EN	CC2420 voltage regulator enable	41 (VREG_EN)	15
11	PF0	Potentiometer	N/A	61
12	FIFO	CC2420 FIFO	30 (FIFO)	26
13	CSn	CC2420 SPI chip select	31 (CSn)	10
14	FIFOP	CC2420 FIFOP	29 (FIFOP)	25
15	3.3V	3.3V Regulated supply	N/A	N/A
16	RESETn	CC2420 Reset	21 (RESETn)	16
17	N.C.	N/A	N/A	N/A
18	CCA	CC2420 Clear Channel Assessment	28 (CCA)	31
19	SFD	CC2420 Start of Frame Delimiter	27 (SFD)	29
20	GND	Ground	N/A	N/A

Table 5: P4 pinout

PCB layout

RF circuits operating at high frequencies are normally sensitive to the physical layout of the PCB. Chipcon has carefully optimized the layout of the CC2420DB Demonstration Board and we therefore recommend that our customer copy, at least the RF parts and the decoupling around the CC2420 when making their own PCB designs.

The PCB is of a 4-layer type in order to provide a well-defined ground plane as well as adequate routing space. The laminate used is standard FR-4 board material. The PCB is 1.0mm thick, with layer 1 on the topside, layers 2 and 3 are internal layers and layer 4 is on the bottom side. Layers 1 and 4 are used for routing, while layer 2 is a ground plane and layer 3 is used for power routing. All areas in the RF section that are not utilized for routing are filled with copper connected to ground to provide RF shielding. The ground planes on all layers are stitched together with closely spaced vias. Please see figure 8 for a board stack up for the CC2420DB PCB.

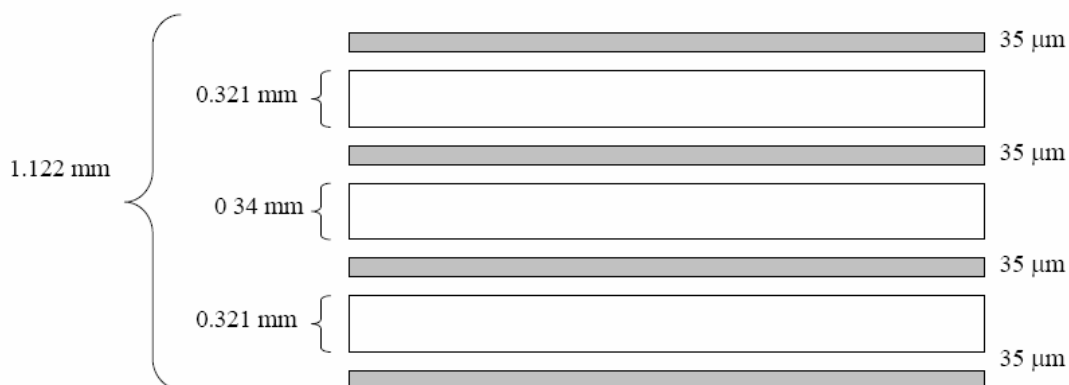
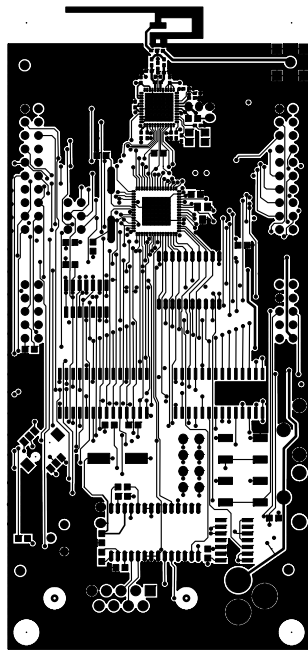
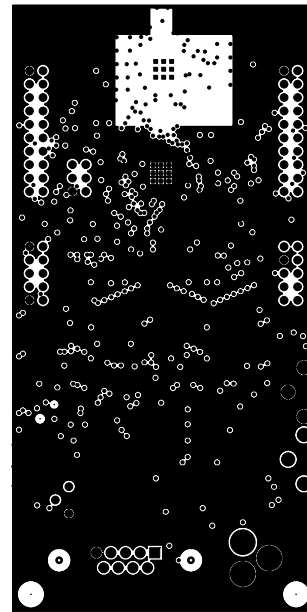


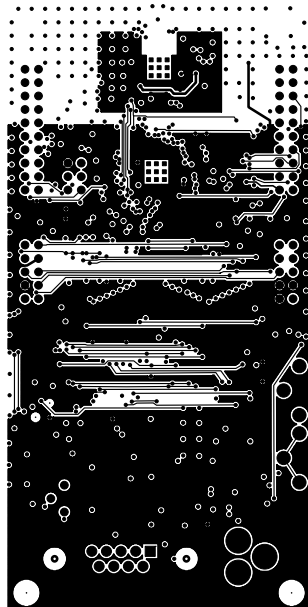
Figure 8: CC2420DB Layer Build Up



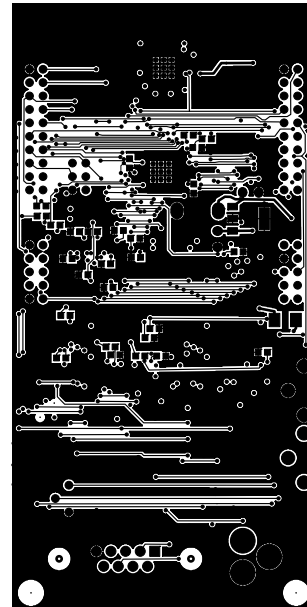
Layer 1



Layer 2

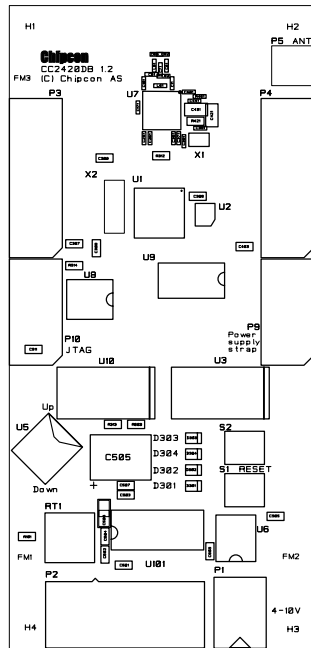


Layer 3

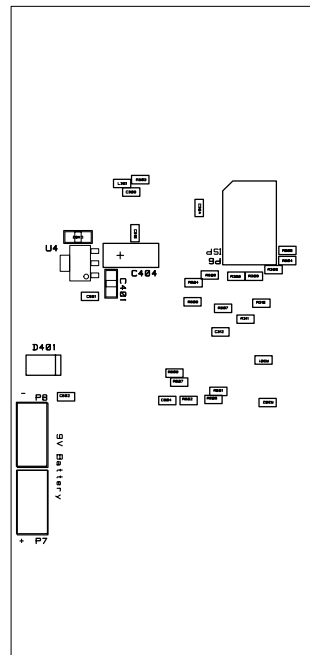


Layer 4

Figure 9: CC2420DB PCB layout



Top Assembly



Bottom Assembly

Figure 10: CC2420DB PCB layout silkscreen

Schematic

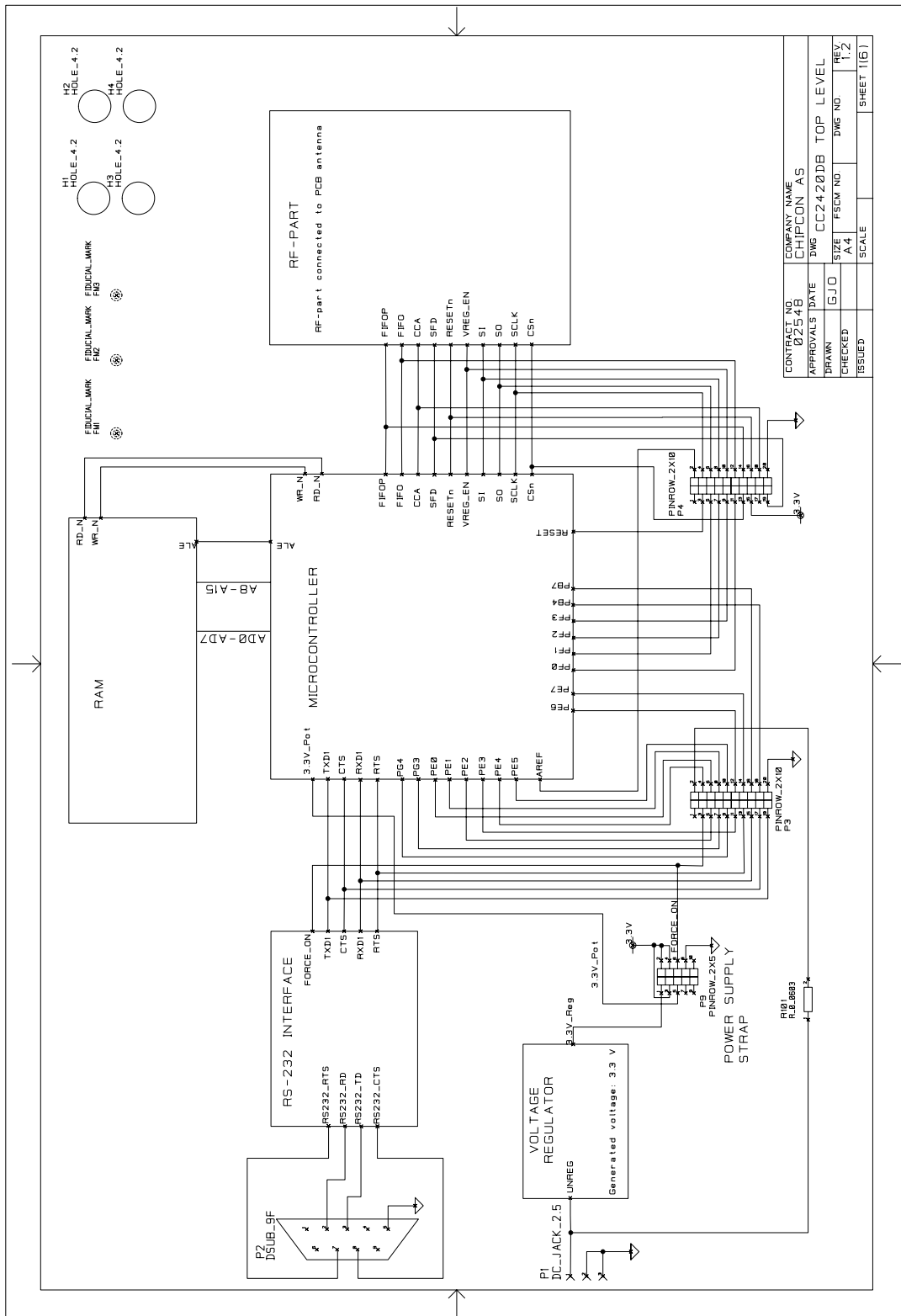


Figure 11: CC2420DB schematic page 1

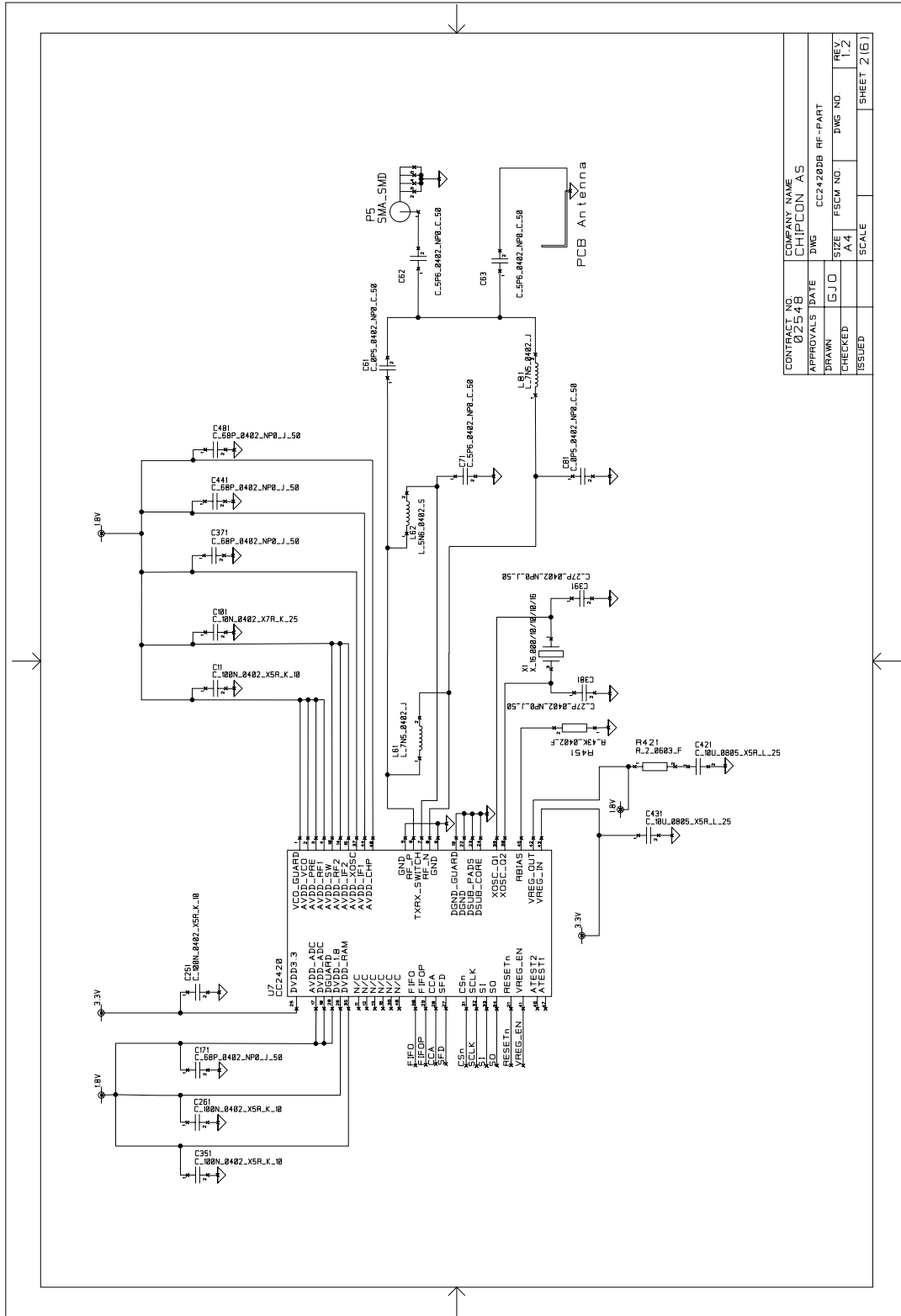


Figure 12: CC2420DB schematic page 2

CONTRACT NO.	02548	COMPANY NAME	CHIPCON AS
APPROVALS	DATE	DWG	CC2420DB RF-PART
DRAWN	GJO	SIZE	A4
CHECKED		FSCM NO.	
ISSUED		SCALE	
			SHEET 2 (6)

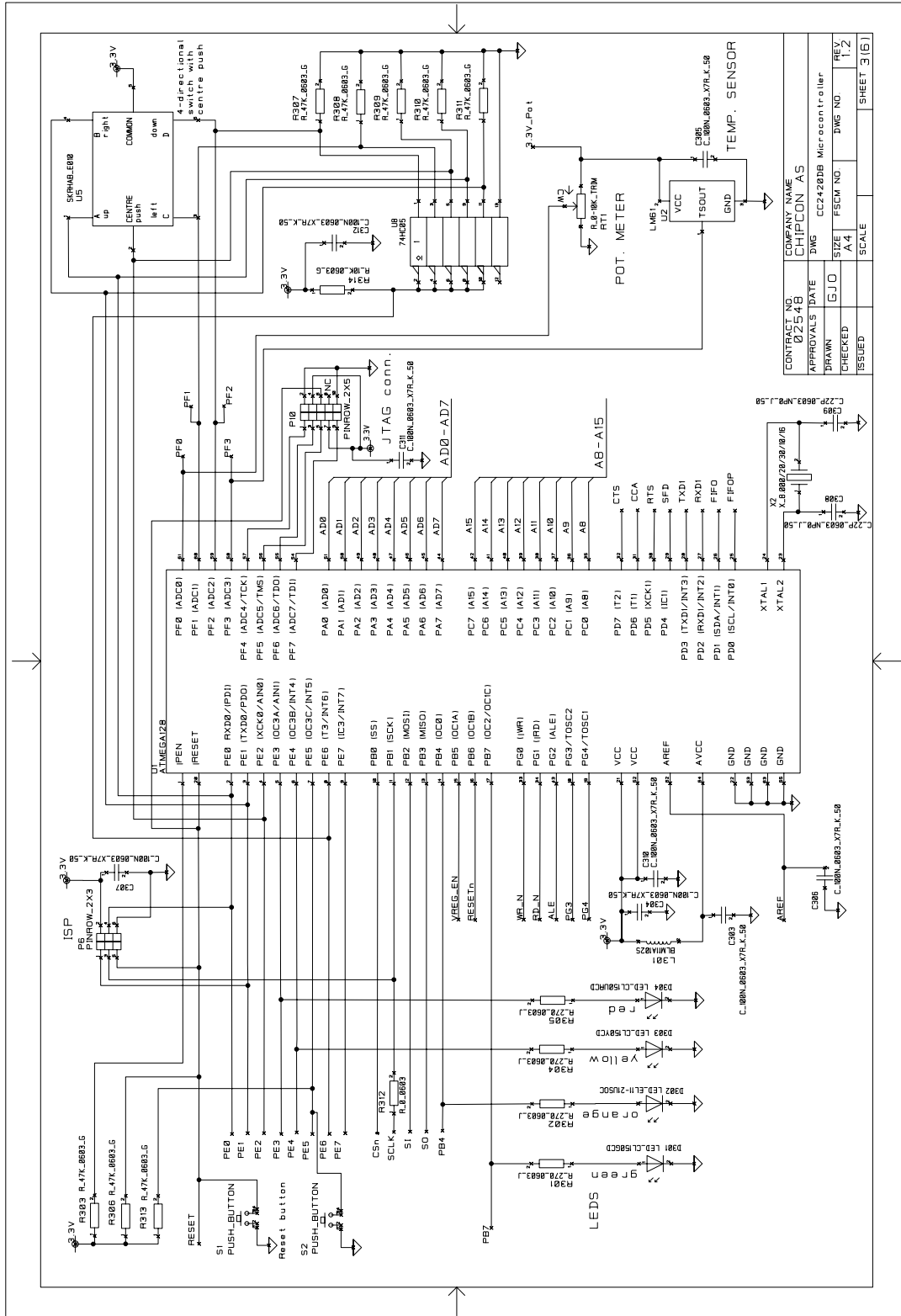


Figure 13: CC2420DB schematic page 3

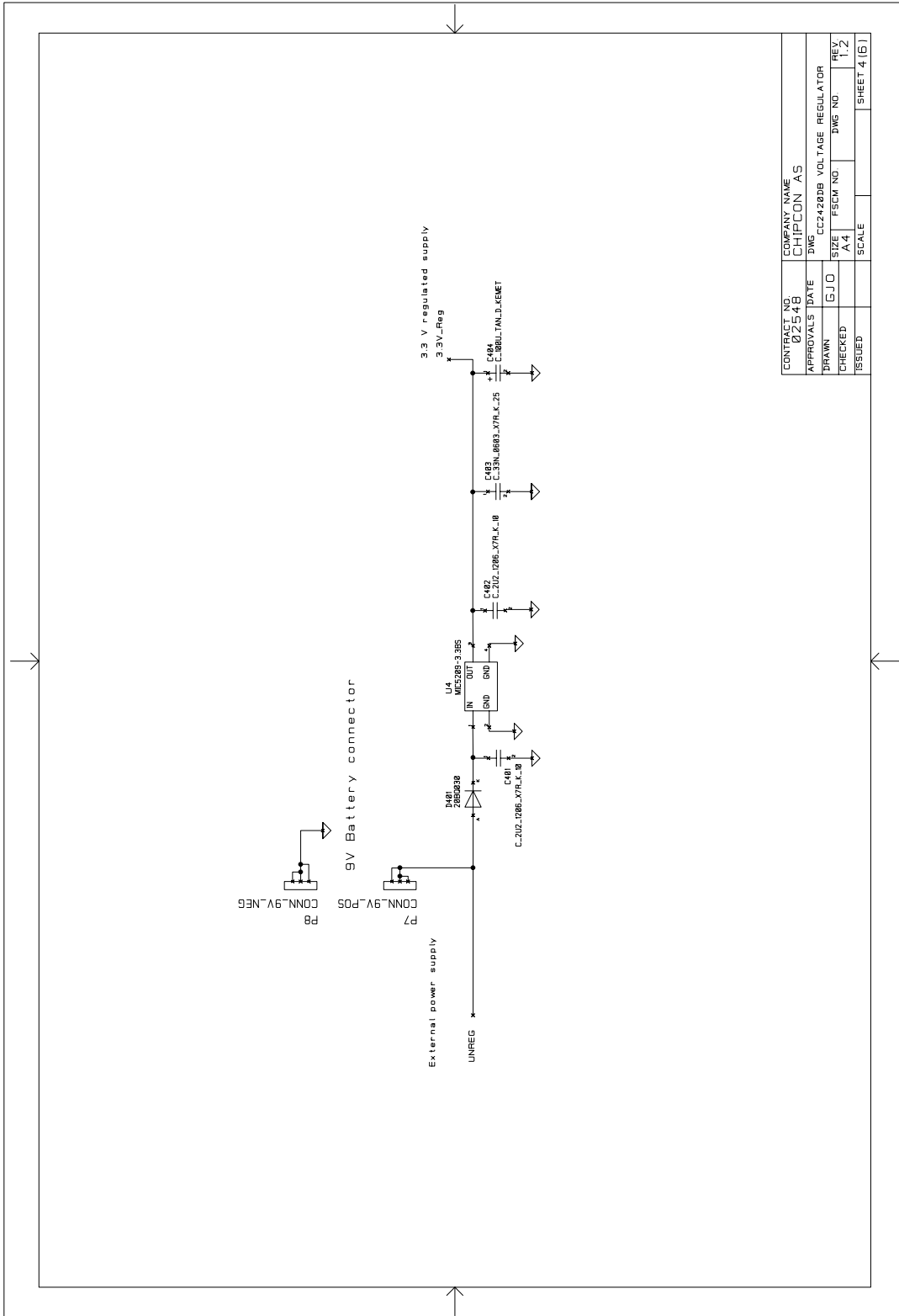
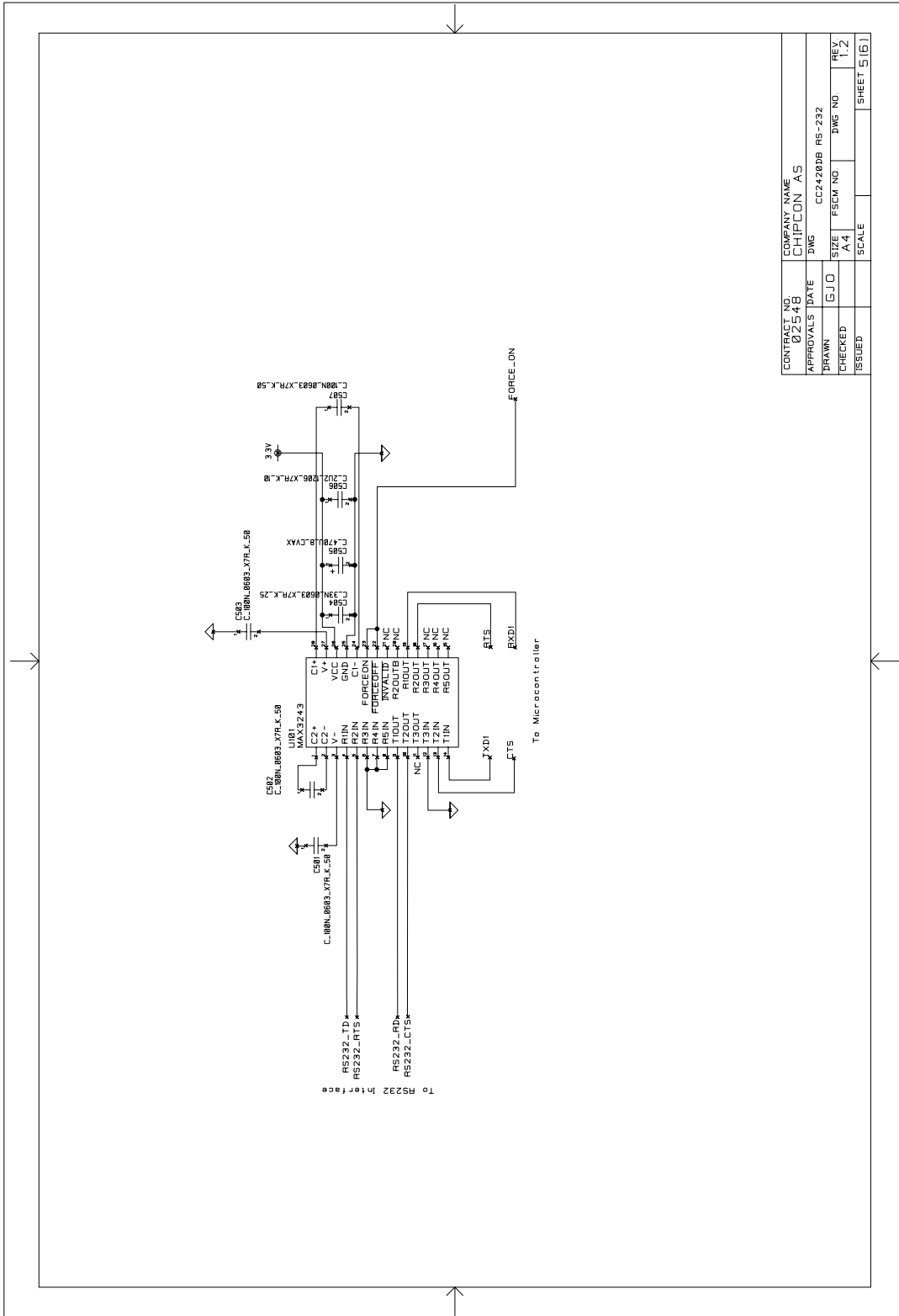
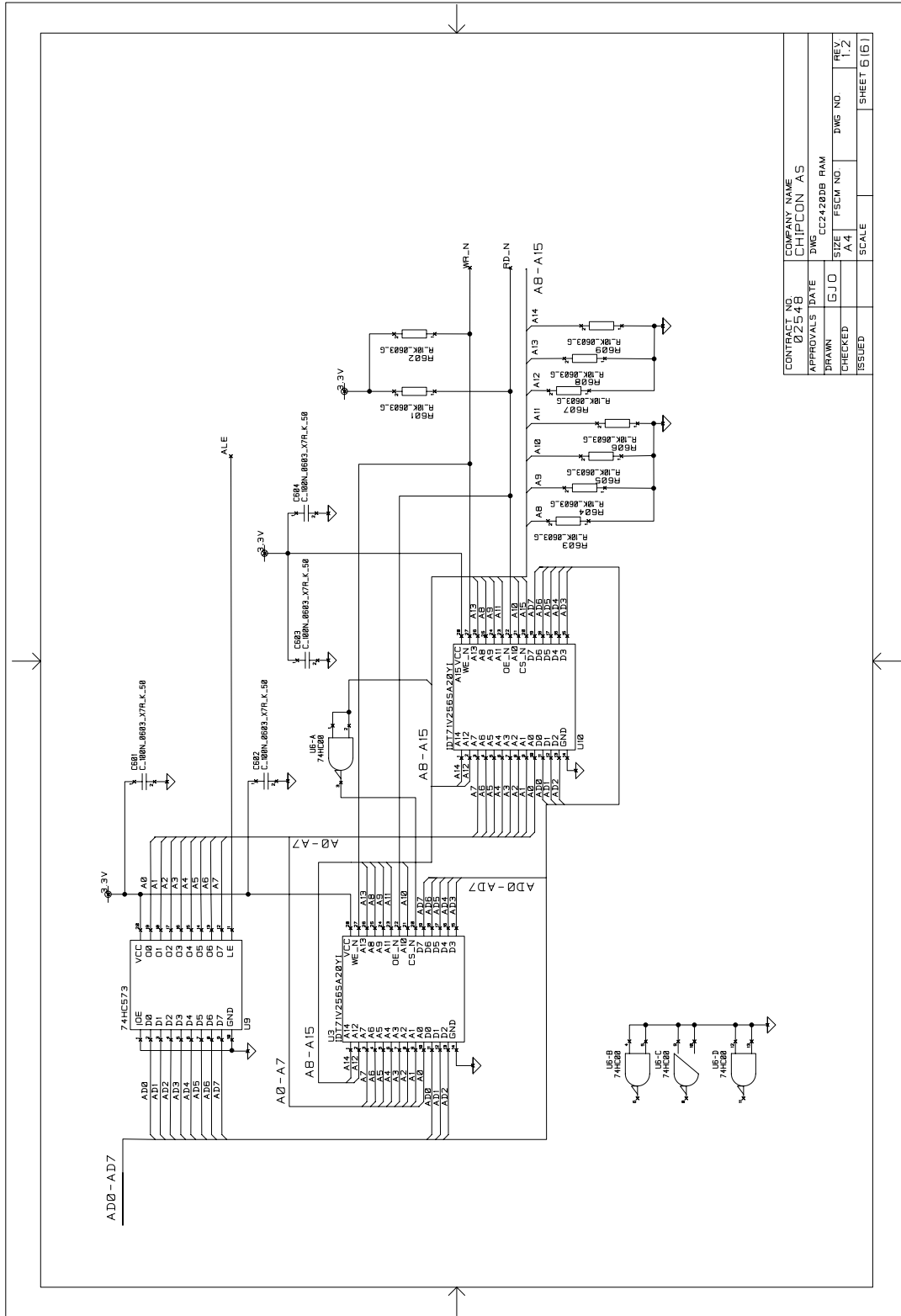


Figure 14: CC2420DB schematic page 4



CONTRACT NO.	02548	COMPANY NAME	CHIPCON AS
APPROVALS	DATE	DWG	CC2420DB RS-232
DRAWN	GJO	SIZE	A4
CHECKED		FSCW NO.	DWG NO.
ISSUED		SCALE	SHEET 5 (6)

Figure 15: CC2420DB schematic page 5



CONTRACT NO.	02548	COMPANY NAME	CHIPCON AS
APPROVALS	DATE	DWG	CC2420DB RAM
DRAWN	GJO	SIZE	A4
CHECKED		FSCW NO.	
ISSUED		SCALE	
		DWG NO.	
		REV.	1.2
		SHEET	6 (6)

Figure 16: CC2420DB schematic page 6

Bill of Materials

Bill of materials, CC2420DB Top-level section			
Reference	Description	Value	Part
P1	DC jack, 2.5mm centre pin		DC_JACK_2.5
P2	D-Sub, 9 pin, female		DSUB_9F
P3	Pin row, 2x10		PINROW_2X10
P4	Pin row, 2x10		PINROW_2X10
P9	Pin row, 2x5		PINROW_2X5
R101	Resistor 0603	0Ω	R_0_0603 Not mounted

Bill of materials, CC2420DB RF Section			
Reference	Description	Value	Part
C101	Capacitor 0402	10nF, 10%	C_10N_0402_X7R_K_25
C11	Capacitor 0402	100nF, 10%	C_100N_0402_X5R_K_10
C171	Capacitor 0402	68pF, 5%	C_68P_0402_NP0_J_50
C251	Capacitor 0402	100nF, 10%	C_100N_0402_X5R_K_10
C261	Capacitor 0402	100nF, 10%	C_100N_0402_X5R_K_10
C351	Capacitor 0402	100nF, 10%	C_100N_0402_X5R_K_10
C371	Capacitor 0402	68pF, 5%	C_68P_0402_NP0_J_50
C381	Capacitor 0402	27pF, 5%	C_27P_0402_NP0_J_50
C391	Capacitor 0402	27pF, 5%	C_27P_0402_NP0_J_50
C421	Capacitor 0805	10uF, 15%	C_10U_0805_X5R_L_25
C431	Capacitor 0805	10uF, 15%	C_10U_0805_X5R_L_25
C441	Capacitor 0402	68pF, 5%	C_68P_0402_NP0_J_50
C481	Capacitor 0402	68pF, 5%	C_68P_0402_NP0_J_50
C61	Capacitor 0402	0.5pF ± 0.25pF	C_0P5_0402_NP0_C_50
C62	Capacitor 0402	5.6pF ± 0.25pF	C_5P6_0402_NP0_C_50 Not mounted
C63	Capacitor 0402	5.6pF ± 0.25pF	C_5P6_0402_NP0_C_50
C71	Capacitor 0402	5.6pF ± 0.25pF	C_5P6_0402_NP0_C_50
C81	Capacitor 0402	0.5pF ± 0.25pF	C_0P5_0402_NP0_C_50
L61	Inductor 0402	7.5nH, 5%	L_7N5_0402_J
L62	Inductor 0402	5.6nH, +/-0.3nH	L_5N6_0402_S
L81	Inductor 0402	7.5nH, 5%	L_7N5_0402_J

Bill of materials, CC2420DB RF Section			
Reference	Description	Value	Part
P5	Surface-mount SMA, straight		SMA_SMD
R421	Resistor 0603	2Ω	R_2_0603_F
R451	Resistor 0402	43kΩ, 1%	R_43K_0402_F
U7	Single-chip transceiver		CC2420
X1	Crystal, ceramic SMD 4x2.5mm		X_16.000/10/15/30/16

Note: The crystal X1 mounted on the CC2420DB board is a 16.000 MHz crystal, with ±10 ppm initial tolerance, ± 15 ppm drift over temperature and a temperature range of -30° C to +85° C. The crystal is designed for 16 pF load capacitance. In an actual application, the tolerance, drift and temperature range of the crystal must be considered with application requirements in mind. Please consult the data sheet more information. It is possible to choose a larger crystal package to save cost. The crystal should have an ESR of 60 Ω or less.

Bill of materials, CC2420DB MCU section			
Reference	Description	Value	Part
C303	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C304	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C305	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C306	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C307	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C308	Capacitor 0603	22pF, 5%	C_22P_0603_NP0_J_50
C309	Capacitor 0603	22pF, 5%	C_22P_0603_NP0_J_50
C310	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C311	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C312	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
D301	LED, green, SMD		LED_CL150GCD
D302	LED, orange, SMD		LED_EL11-21USOC
D303	LED, yellow, SMD		LED_CL150YCD
D304	LED, red, SMD		LED_CL150URCD
L301	EMI filter bead		BLM11A102S
P10	Pin row, 2x5		PINROW_2X5
P6	Pin row, 2x3		PINROW_2X3
R301	Resistor 0603	270Ω, 5%	R_270_0603_J
R302	Resistor 0603	270Ω, 5%	R_270_0603_J
R303	Resistor 0603	47kΩ, 2%	R_47K_0603_G

Bill of materials, CC2420DB MCU section			
Reference	Description	Value	Part
R304	Resistor 0603	270 Ω , 5%	R_270_0603_J
R305	Resistor 0603	270 Ω , 5%	R_270_0603_J
R306	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R307	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R308	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R309	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R310	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R311	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R312	Resistor 0603	0 Ω	R_0_0603
R313	Resistor 0603	47k Ω , 2%	R_47K_0603_G
R314	Resistor 0603		R_10K_0603_G
RT1	Trimming pot, 10K, knob		R_0-10K_TRIM
S1	Push button, SMD		PUSH_BUTTON
S2	Push button, SMD		PUSH_BUTTON
U1	Atmel microcontroller, MLF64		ATMEGA128
U2	Temperature sensor, SOT23		LM61
U5	4-directional switch (Joystick)		SKRHAB_E010
U8	Hex inverter, oc		74HC05
X2	Crystal, HC-49-SMD		X_8.000/20/30/10/16

Bill of materials, CC2420DB Voltage regulator section			
<i>Reference</i>	<i>Description</i>	<i>Value</i>	<i>Part</i>
C401	Capacitor 1206	2.2uF, 10%	C_2U2_1206_X7R_K_10
C402	Capacitor 1206	2.2uF, 10%	C_2U2_1206_X7R_K_10
C403	Capacitor 0603	33nF, 10%	C_33N_0603_X7R_K_25
C404	Capacitor, tantalum, low-ESR	100uF± 20	C_100U_TAN_D_KEMET
D401	Schottkey diode, 2A		20BQ030
P7	9V battery connector (positive)		CONN_9V_POS
P8	9V battery connector (negative)		CONN_9V_NEG
U4	3.3 V low drop regulator 500 m		MIC5209-3.3BS

Bill of materials, CC2420DB RS-232 section			
<i>Reference</i>	<i>Description</i>	<i>Value</i>	<i>Part</i>
C501	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C502	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C503	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C504	Capacitor 0603	33nF, 10%	C_33N_0603_X7R_K_25
C505	Capacitor Low Impedance		C_470U_8_CVAX
C506	Capacitor 1206	2.2uF, 10%	C_2U2_1206_X7R_K_10
C507	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
U101	RS-232 Transceiver, 3-5V		MAX3243

Bill of materials, CC2420DB RAM			
<i>Reference</i>	<i>Description</i>	<i>Value</i>	<i>Part</i>
C601	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C602	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C603	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
C604	Capacitor 0603	100nF, 10%	C_100N_0603_X7R_K_50
R601	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R602	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R603	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R604	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R605	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R606	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R607	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R608	Resistor 0603	10k Ω , 2%	R_10K_0603_G
R609	Resistor 0603	10k Ω , 2%	R_10K_0603_G
U10	32kB SRAM		IDT71V256SA20YI
U3	32kB SRAM		IDT71V256SA20YI
U6	Quad NAND gate		74HC00
U9	Octal latch with 3-State output		74HC573

Table 6: Bill of materials

Using the CC2420DB Demonstration Board

The CC2420DB Demonstration Board is designed for quick prototyping, and also for quick practical evaluation such as range testing.

Software can be programmed into the AVR microcontroller either using an external programmer such as the AVR ISP programmer, AVR JTAG ICE or using the serial port to communicate with the bootloader that is programmed into the MCU when the CC2420DB is shipped from the factory. If the bootloader is overwritten when using an external programmer, the AVR must be re-programmed with the bootloader before it is possible to program the AVR via the serial port again. Note that due to the 8 MHz crystal used with the Atmel controller, some higher Baud rates cannot be supported for communication between computer and CC2420DB using the serial port. There is a table in the Atmel ATmega 128 data sheet that list the Baud rate error for different crystal frequencies and Baud rates.

CC2420 Software Download and Evaluation Using AVR Studio 4

Use a serial cable to connect the serial interface of the CC2420DB to the serial port of a PC running AVR Studio. Our SmartRF[®] Studio cannot communicate with the CC2420DB, but can be used to find register values for the CC2420.

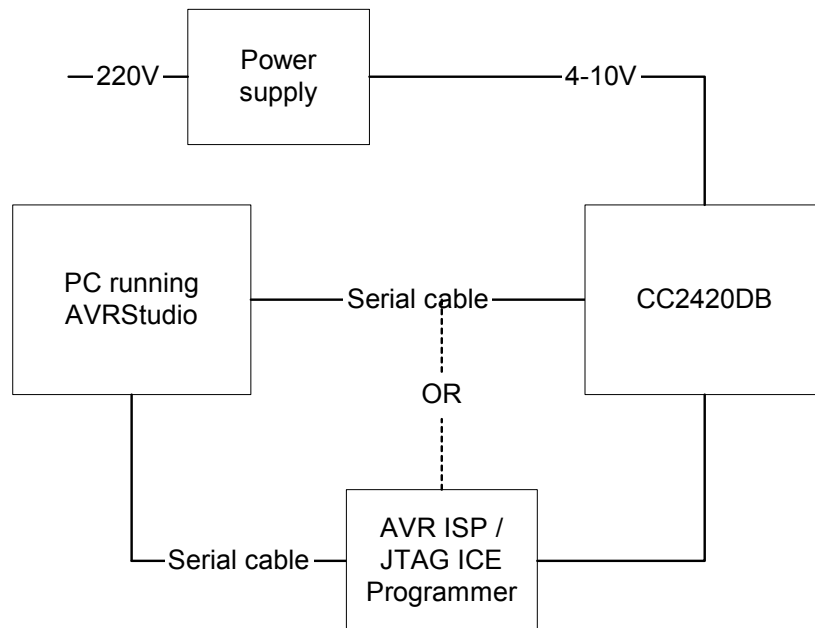


Figure 17: Software download using AVR Studio 4

Preprogrammed 64-bit IEEE Address

Each CC2420DB in a Demonstration Board Kit revision 1.6 or newer is preprogrammed with a unique 64-bit IEEE address. These addresses are stored in Little-Endian format, in the lowest 8 bytes of the EEPROM of the Atmega128L microcontroller. This places the least significant byte (LSB) at location 0x0000 and the most significant byte (MSB) at location 0x0007. The IEEE address is also displayed on a sticker affixed to each CC2420DB.

Preprogrammed Fuses

The configuration of the processor fuses can be performed through the SPI interface with the AVR ISP programmer or the JTAG ICE programmer and AVR Studio 4. The JTAG ICE fuse window is depicted in figure 18 and 19. The CC2420DB is configured with the following fuses enabled:

1. On Chip Debug Enabled
2. JTAG Interface Enabled
3. Serial program downloading (SPI) enabled
4. Preserve EEPROM memory through the Chip Erase Cycle
5. Boot Flash section size = 512 Boot start address
6. Boot vector enabled (default address = \$0000)
7. Brown-out detection level at VCC 2.7 V
8. Brown-out detection enabled
9. Ext.Crystal/Resonator High Freq.; Start Up time: 16CK + 64 ms [CKSEL=1111 SUT=11]

Note:

When using the JTAG ICE mkII or the AVR ISP programmer the boot vector fuse must be disabled.

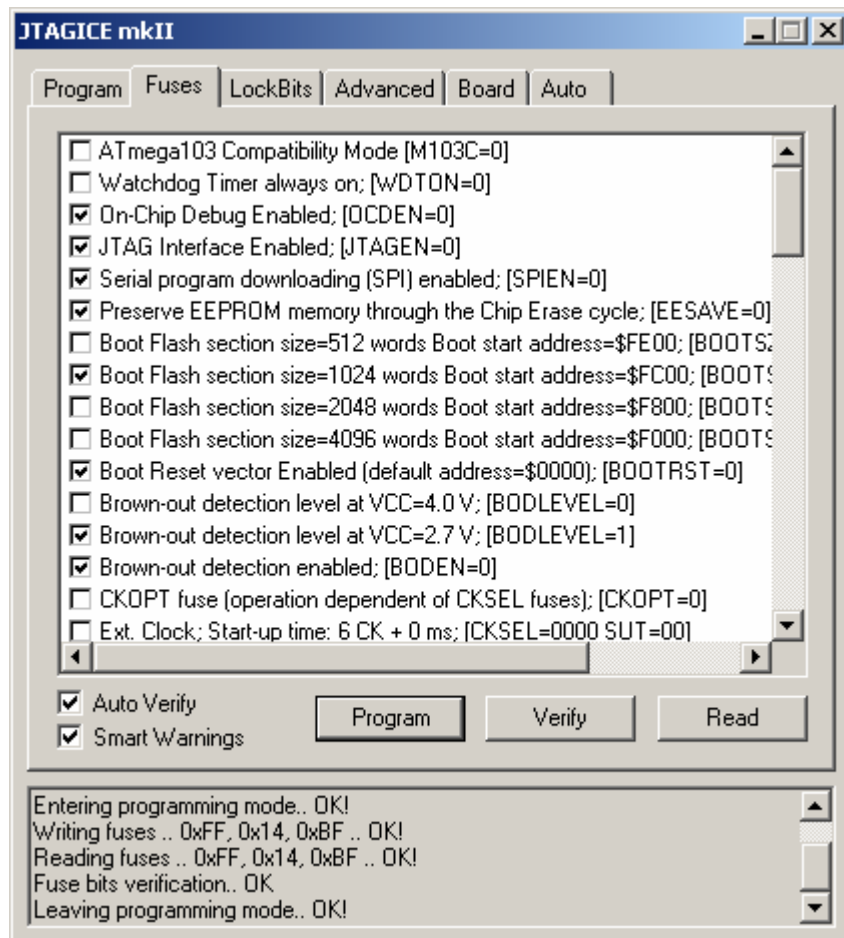


Figure 18: AVR Studio Fuses Window CC2420DB

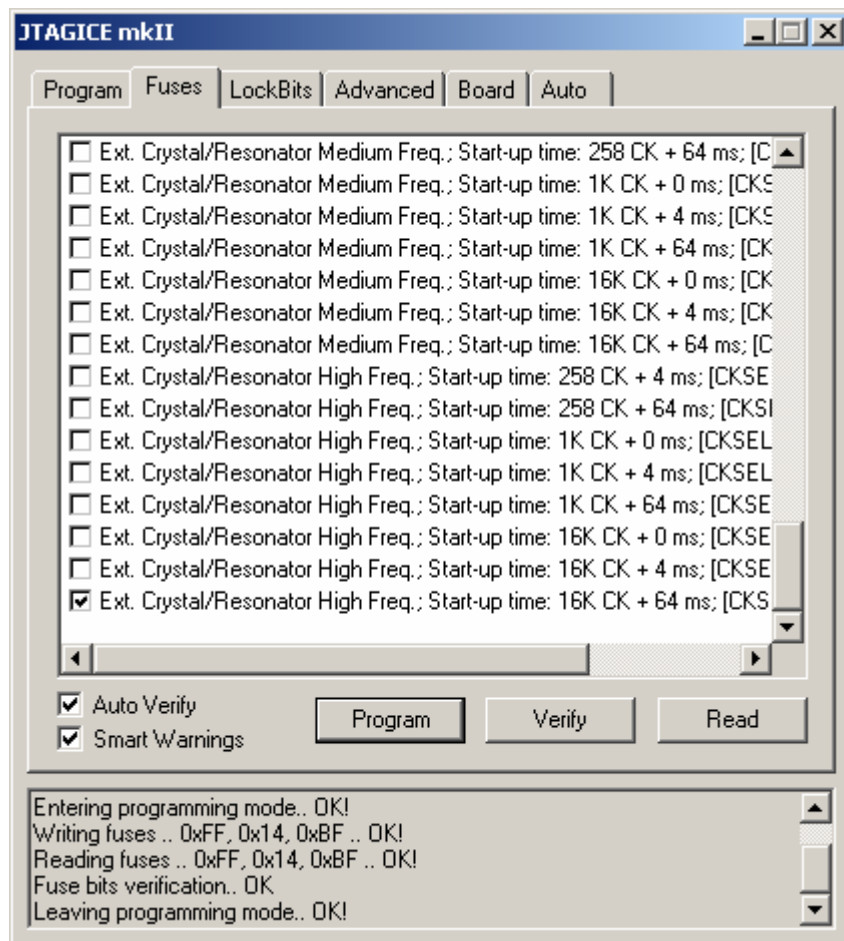


Figure 19: AVR Studio Fuses Window CC2420DB

Preprogrammed Lock Bits

The following lock bit is preprogrammed to protect the bootloader from accidental overwrites.

1. Boot Loader Protection Mode 2: SPM prohibited in Boot Loader Section

Note:

The only way to erase this bit is to use JTAG ICE mkII or AVR ISP and run an erase cycle of the flash using AVR Studio. See figure 20.

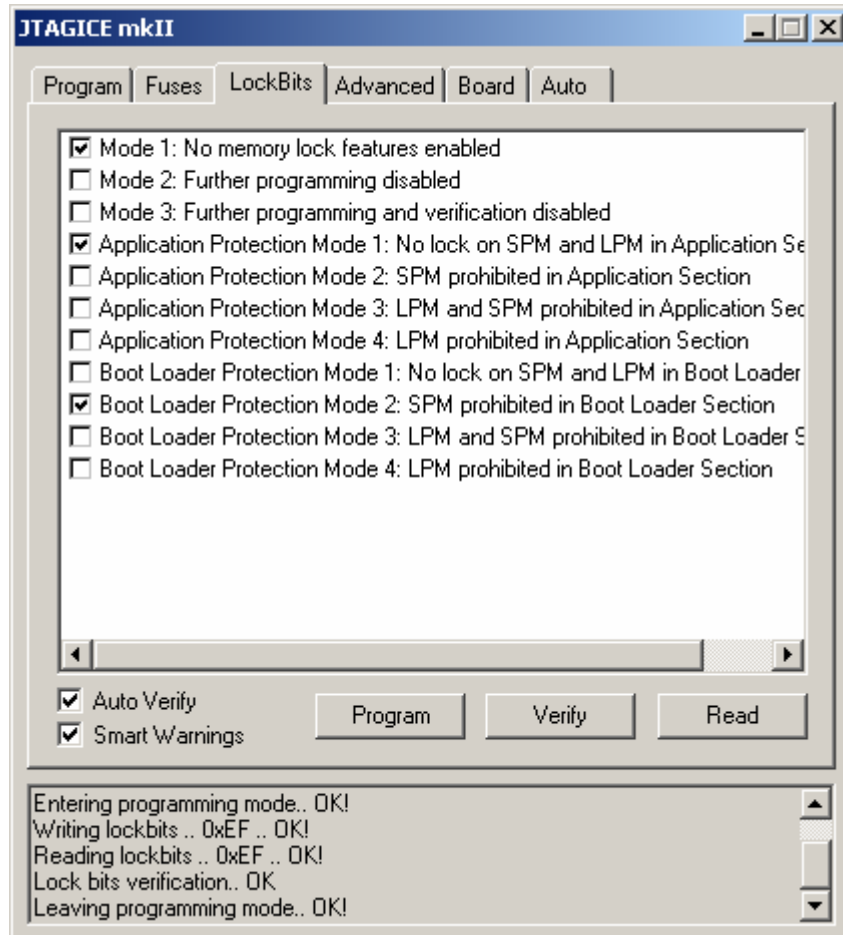


Figure 20: AVR Studio Lock Bits Window

JTAG ICE or JTAG ICE mkII

The CC2420DB provides a connector for the JTAG ICE programmer to communicate with AVR Studio. This interface can be used to debug or download source code using Coff files or hex files. The JTAG connector is P10. The Coff files are built using the make file option, make extcoff. See the make file enclosed with our software example for details.

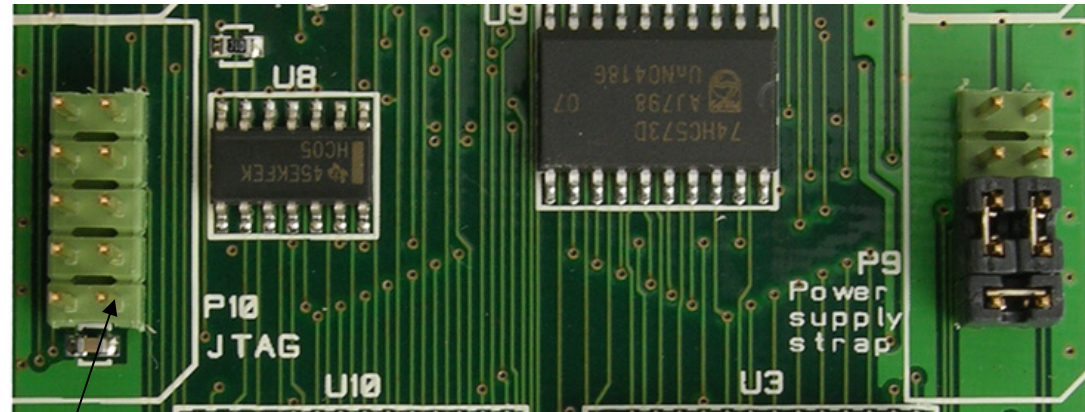
Step-by-Step procedure:

1. Connect JTAG ICE to connector P10 on CC2420DB. Please refer to figure 20 for location of the connector
2. Connect power to CC2420DB
3. Disable the *Boot Reset Vector Enabled* fuse

4. Start AVR Studio and establish a new project only one time. See section on how to start a new debugging project
5. Load the select Coff file, built using the make file option extcoff
6. Start debugging

Note:

When using the JTAG ICE programmer the boot vector fuse must be disabled.



Pin 1

Figure 21: JTAG ICE connector P10 on CC2420DB

Saving the IEEE Address of the CC2420DB

Before any debugging is performed on the CC2420DB, the content of the Atmega128L EEPROM memory must be read and saved to a disk file on the development computer. This will allow the IEEE address to be restored if the EEPROM gets erased or corrupted. The AVR Studio program in conjunction with the JTAG ICE or AVR ISP is applied to upload the EEPROM data from the Atmega128L microcontroller and recorded to the disk file specified by the user.

When a project is first created in AVR Studio, the EEPROM will be erased when the .cof file is downloaded to the CC2420DB. Due to this, it is important to save the EEPROM content to a disk file before creating the new project. The following procedure can be applied to save EEPROM data initially or at any desired time:

- With power removed from the CC2420DB, connect the JTAG ICE, JTAG ICE mkII or AVR ISP to their respective connectors on the CC2420DB. Make sure the JTAG cable is oriented correctly. Please refer to figure 21 to see where pin 1 is located.

- Open AVR Studio and click on the AVR button. When AVR Studio detects the programming device connected to the CC2420DB the programming settings window will appear. Please refer to figure 22.

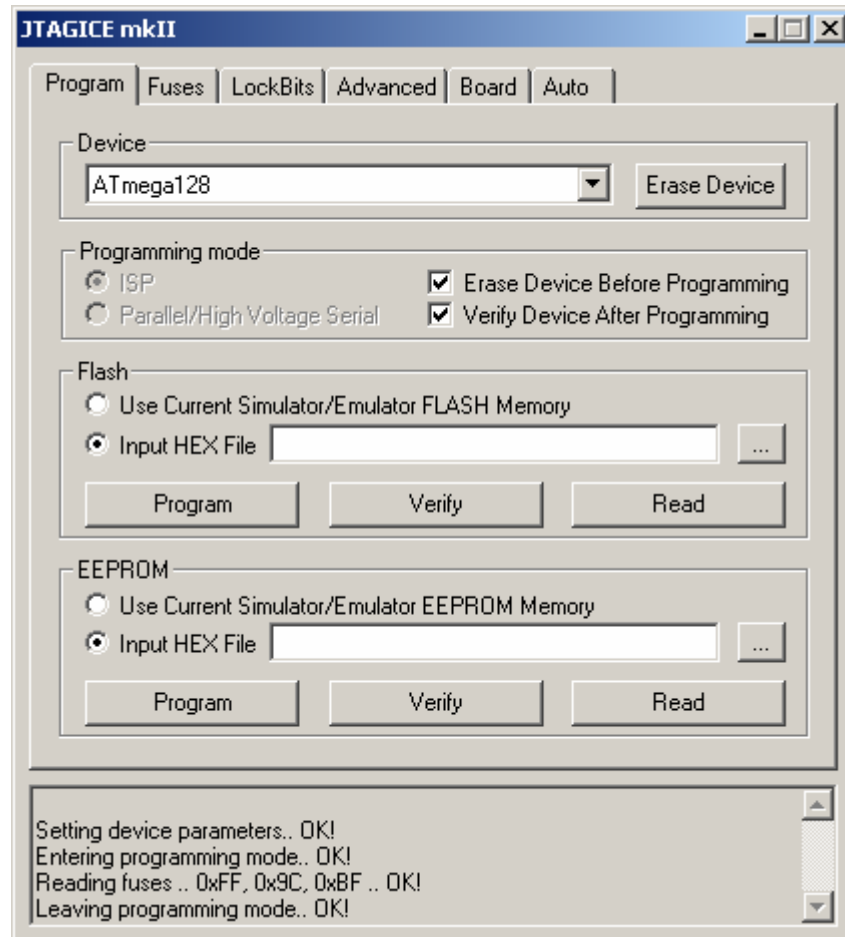


Figure 22: JTAG ICE mkII Program Window

- Click on the Read button in the EEPROM area to display the Save As dialog window:

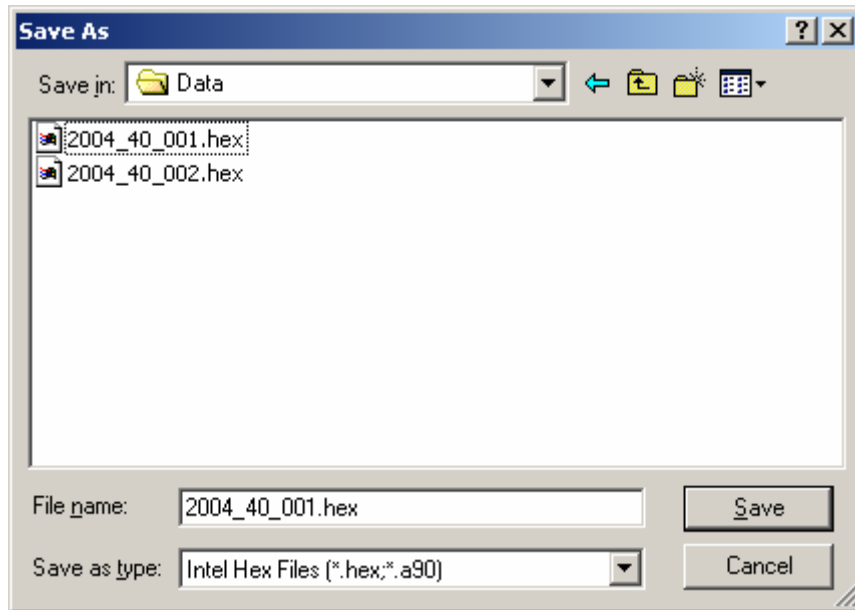


Figure 23: EEPROM Save As Dialog Window

- Browse to the appropriate folder to save the EEPROM data file. For example the data can be stored at the following location: **C:\Chipcon\Projects\CC2420DB\Data**
- Type a new filename and click on the Save button to start the upload and file saving process. The example above shows filenames that were created using the serial numbers for each board in a kit.
- If you are going to save the EEPROM data for another CC2420DB board repeat this procedure

Restoring the IEEE Address of the CC2420DB

The Atmega128L EEPROM memory is used to store the Chipcon assigned unique 64-bit address. Occasionally, the content of the EEPROM content must be restored from a disk file. The AVR Studio in conjunction with JTAG ICE, JTAG ICE mkII or AVR ISP will be used to download the EEPROM data from a disk file specified by the user to the Atmega128L EEPROM memory.

When a project is first created in AVR Studio, the EEPROM will be erased when the .cof file is downloaded to the CC2420DB. For this reason, the EEPROM memory content must be reloaded from the previously saved disk file after creating the new AVR Studio project. The following procedure can be used to restore the initially EEPROM data.

- With power removed from the CC2420DB, connect the JTAG ICE, JTAG ICE mkII or AVR ISP to the their respective connectors on the CC2420DB. Make sure the JTAG cable is oriented correctly. Please refer to figure 21 to see where pin 1 is located.
- Open AVR Studio and click on the AVR button. When AVR Studio detects the programming device connected to the CC2420DB the programming settings window will appear. Select the **Fuses** tab: Please refer to figure 24.
- Disable the *Preserve EEPROM memory through the Chip Erase Cycle* fuse and click on the **Program** button. Examine the fuses dialog window for proper fuse programming. Please refer to figure 24.

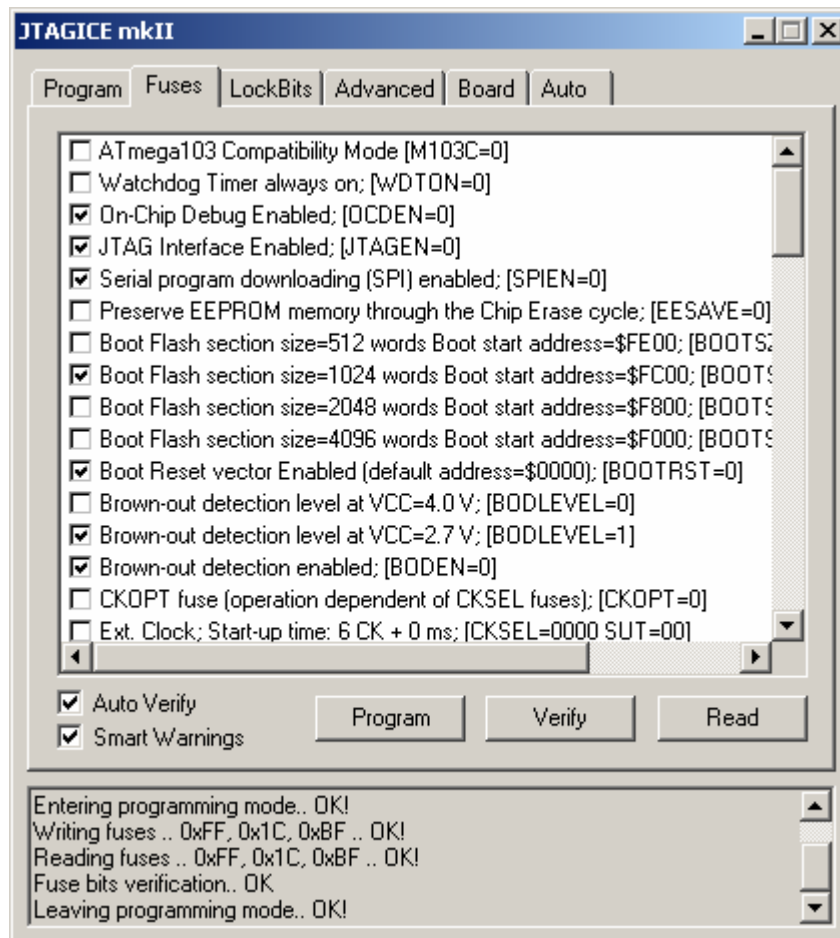


Figure 24: JTAG ICE mkII Fuses Dialog Window

- Return to the **Program** tab and click on the **Browse** button in the EEPROM area to display the **Open** dialog window to browse to the appropriate folder to find the needed EEPROM data file.
- Select an EEPROM data file and click the **Open** button to return to the JTAG ICE, JTAG ICE mkII or AVR ISP dialog window.
- Back at the respective programming device dialog window, click the **Program** button in the EEPROM area to start the download process.
- Return to the **Fuses** tab, enable the *Preserve EEPROM memory through the Chip Erase Cycle* fuse and click **Program** button. Examine the fuses dialog window for proper fuse programming

A New Debugging Project With CC2420DB

When AVR Studio opens a debugging file (.cof file generated by the GCC tools), it looks for a previous project file (.aps) with the same filename. If none is found (new project) or if the user answers no when prompted with “A project already exists using this object file, would you load this?”, the **Select device platform and device** settings dialog window will appear. This indicates that a new project file (.aps) is being created and initialized. Make the settings, as shown in figure 25, and click the **Finish** button to continue

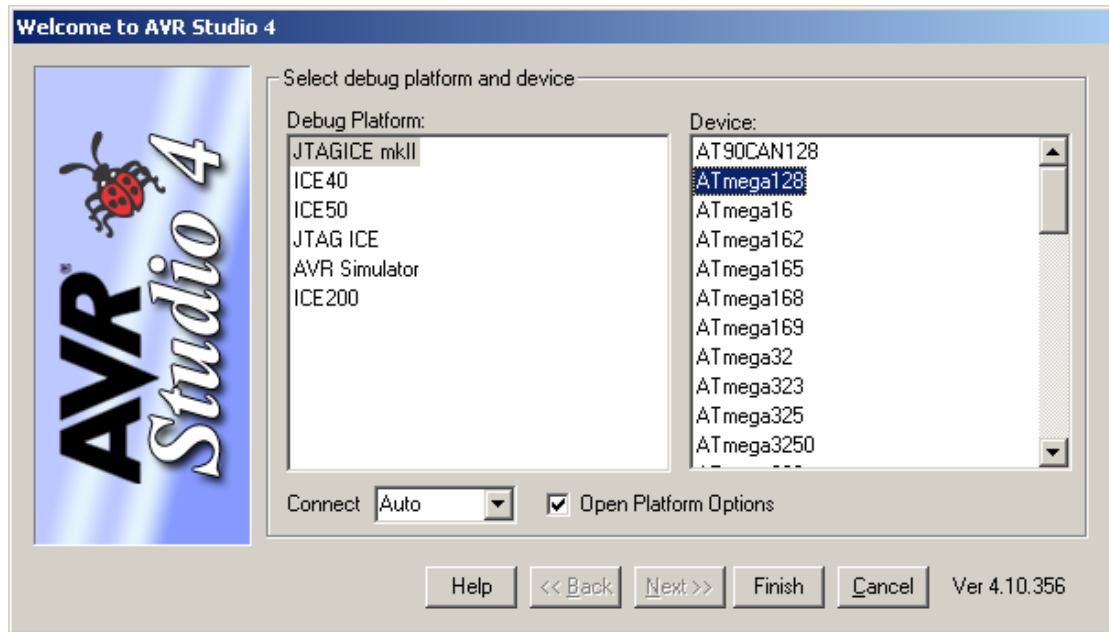


Figure 25: JTAG ICE mkII Device/Platform Settings

Now, select the **Connection** and check for the following settings. If different than shown below, make the required changes. Since this is a new project, settings for the **Debug** tab can't be specified- the microcontroller's Flash and EEPROM will be erased before the program is downloaded to Flash. Click on the **OK** button to continue. Please refer to figure 26 for reference.

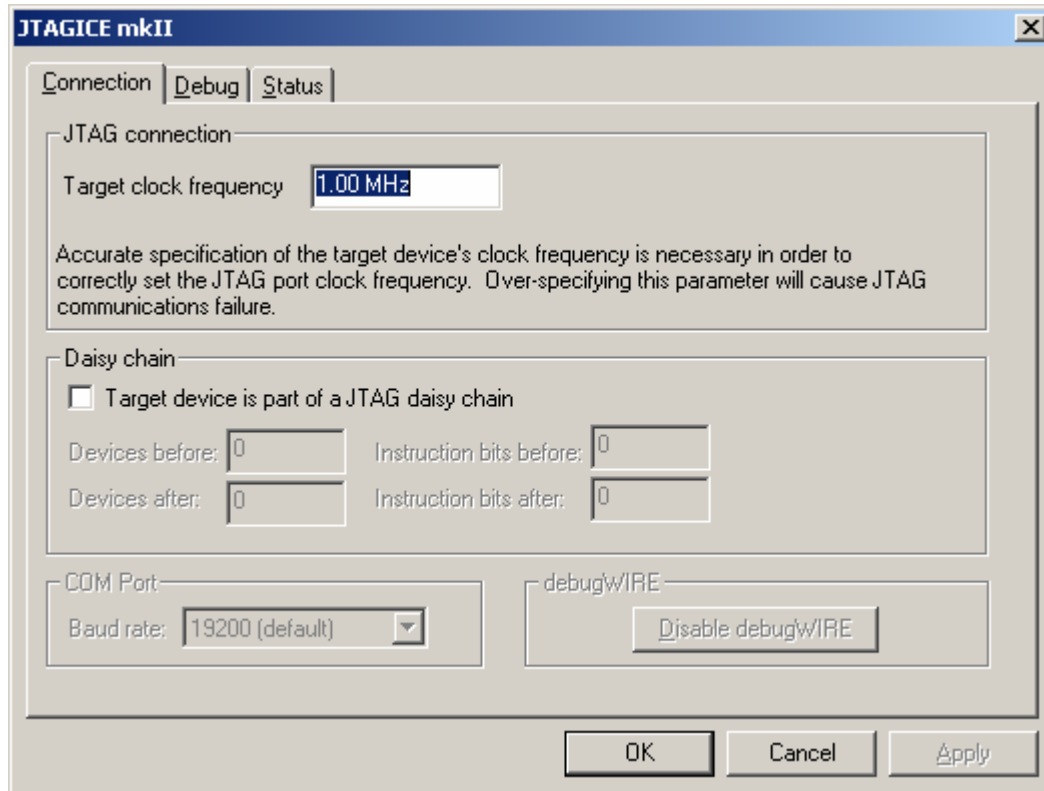


Figure 26: JTAG ICE mkII Connection Settings

At this point AVR Studio downloads the program into the CC2420DB Atmega128L Flash memory, graphically showing progress at the bottom of the screen. If successful, the program counter is reset and AVR Studio enters debugging mode. Completion or error status is shown in the message box at the bottom of the window.

The debugging dialog window will appear as depicted in figure 27. Now, settings for subsequent downloading of project files can be specified and saved in the .aps file. Select the **Debug** tab on the menu of the AVR Studio window click **JTAGICE mkII Options** item on the menu: [Debug] -> [JTAGICE mkII Options]

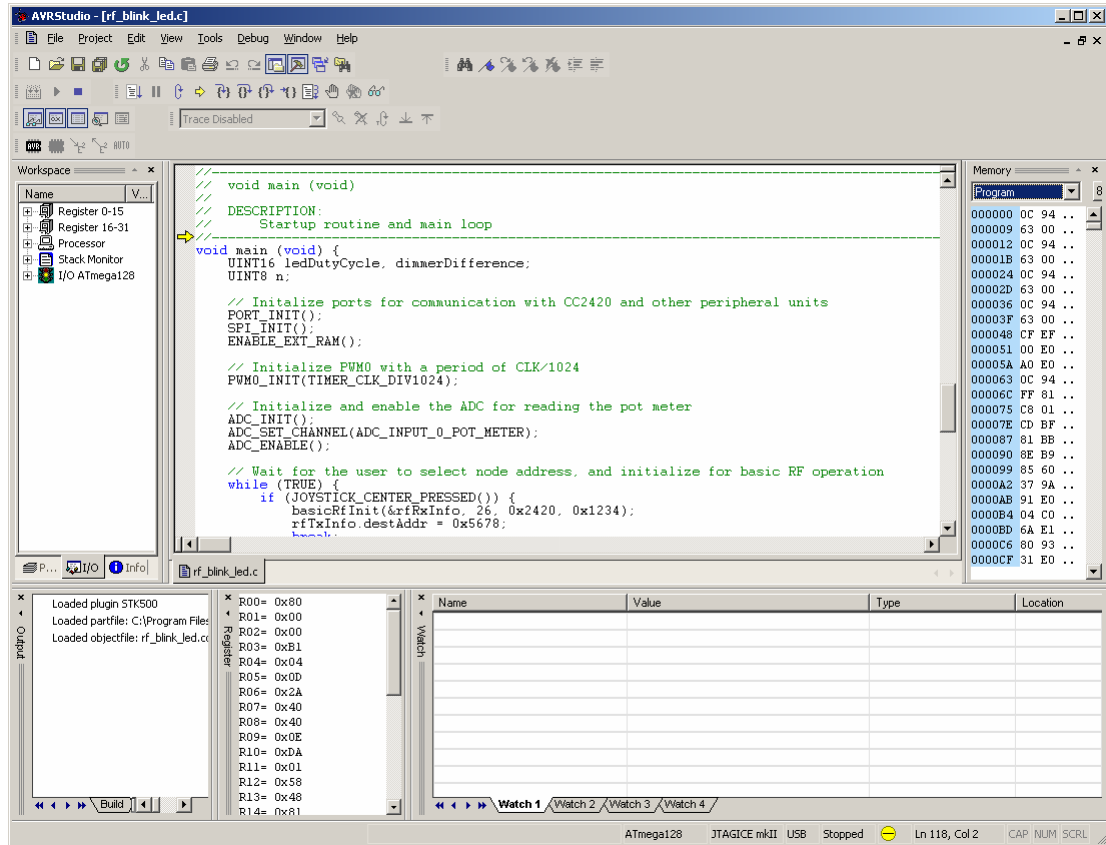


Figure 27: AVR Debugging Dialog Window

Now, select the **Debug** tab and check for the following settings. If different than shown in figure 28, make the required changes. The most important setting here is the “*Preserve EEPROM contents when reprogramming device*”. Click on the **Apply** buttons to update the project file (.aps) and then click the **OK** button to continue.

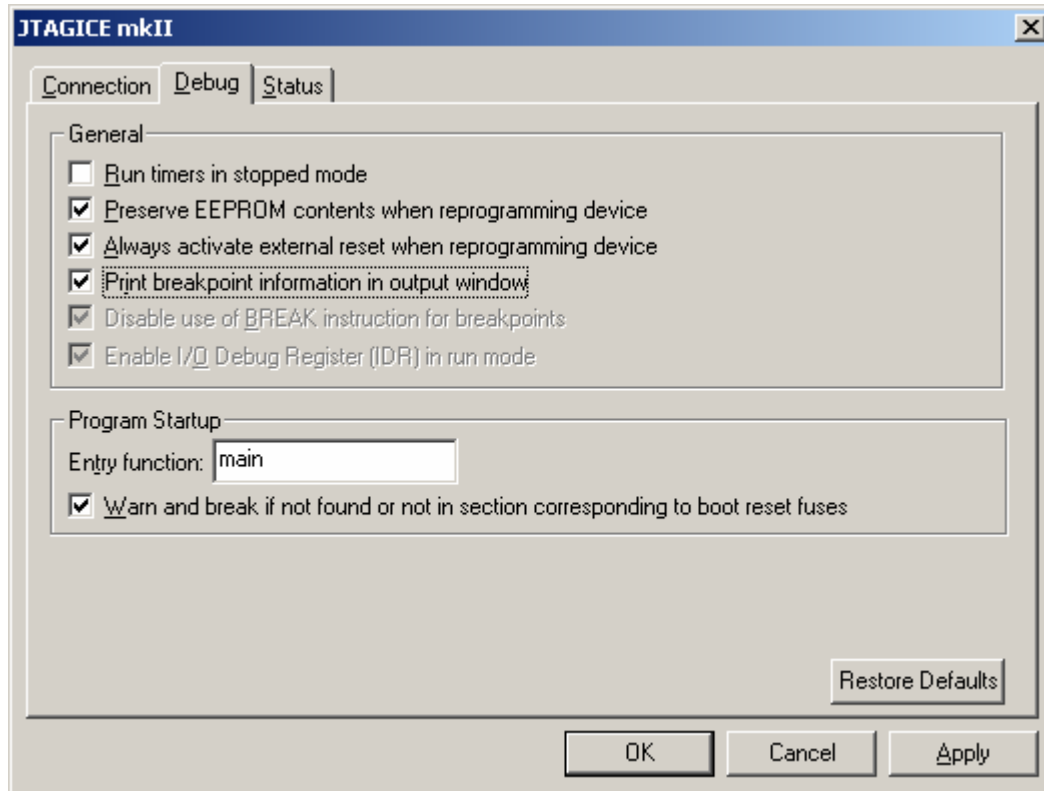


Figure 28: JTAG ICE mkII Debug Settings Window

Since the IEEE address was erased when the program was initially downloaded when establishing a new project. The address must be restored before debugging or normal operation begun. Save project and exit debugging. Follow procedure in the section “*Restoring the IEEE Address of the CC2420DB*”. The next time the same debugging project is reloaded with AVR Studio the project settings are configured properly.

In System Programming

Using the AVRISP in-circuit programmer or similar devices, the microcontroller can be reprogrammed. Both the program code memory (flash) and the EEPROM can be reprogrammed.

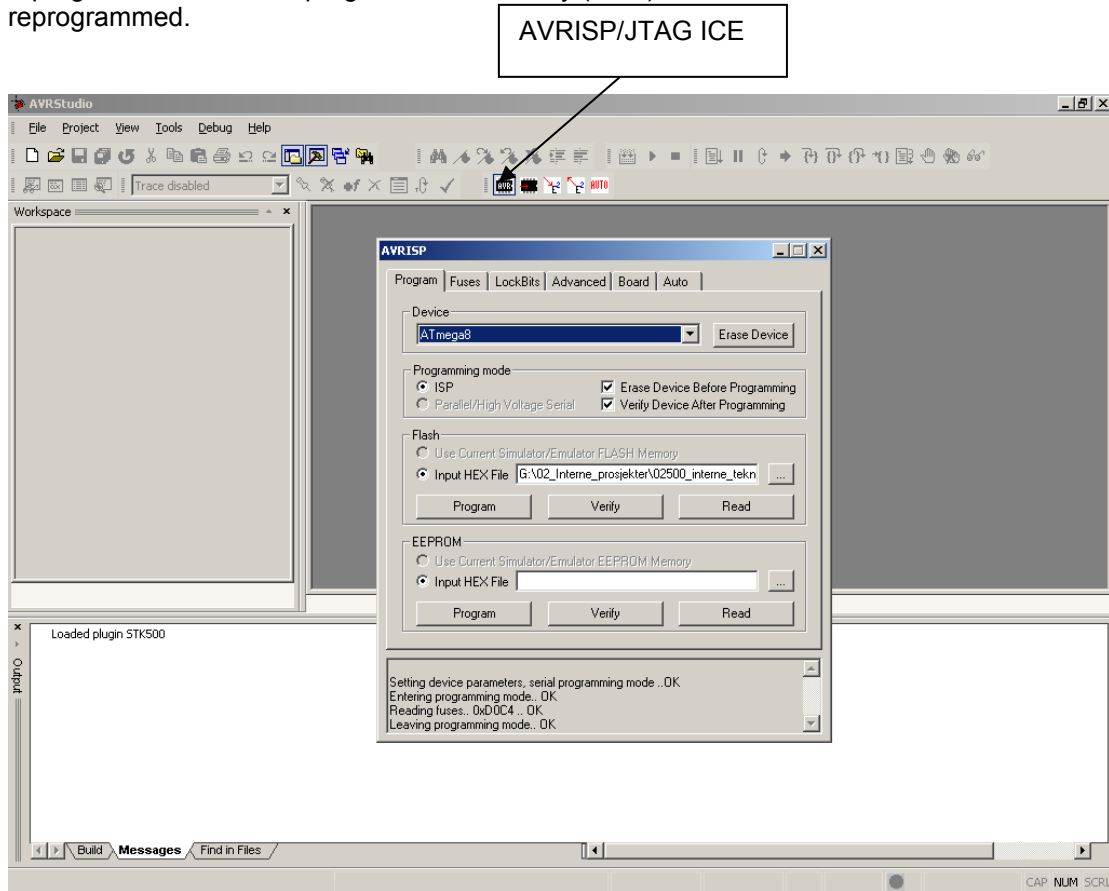


Figure 29: AVRISP Program Window used to program flash or EEPROM

Note:

When using the AVR ISP programmer the boot vector fuse must be disabled.

The device used for programming must be connected to the CC2420DB programming socket. The programming interface uses 4 signal lines. These four lines can be accessed at the 6-pin connector, P6. The pin out and description is listed in the table below.

Signal/pin	AVR programming socket, pin
MISO	1
SCK	3
MOSI	4
RESET	5
VCC	2
GND	6

Table 7: ISP header connector

Bootloader

The CC2420DB is shipped with a bootloader. This loader communicates with AVR Studio. After reset the bootloader must have a way to determine to start the program mode or to run the application software residing in the application code section of the flash. To start the bootloader the push button S2 on CC2420DB is used to enter program mode. This button is held low after reset while starting the AVR Prog from the Tools menu in AVR Studio until the bootloader program window is loaded. Please see Figure 30 and Figure 31.

Step-by-Step procedure:

1. Connect the CC2420DB serial port to your PC COM port
2. Connect power to CC2420DB
3. Start AVR Studio
4. Hold down the S2 push button.
5. Push and release the S1 (reset button) while still holding the S2 push button
6. In AVR Studio select Tools→AVR Prog... from the menu (See figure 22)
7. Release the S2 push button
8. Browse to find your application program *.hex file
9. Click on the Flash Program button to program the flash
10. Disconnect serial port
11. Reset CC2420DB
12. Start your application program

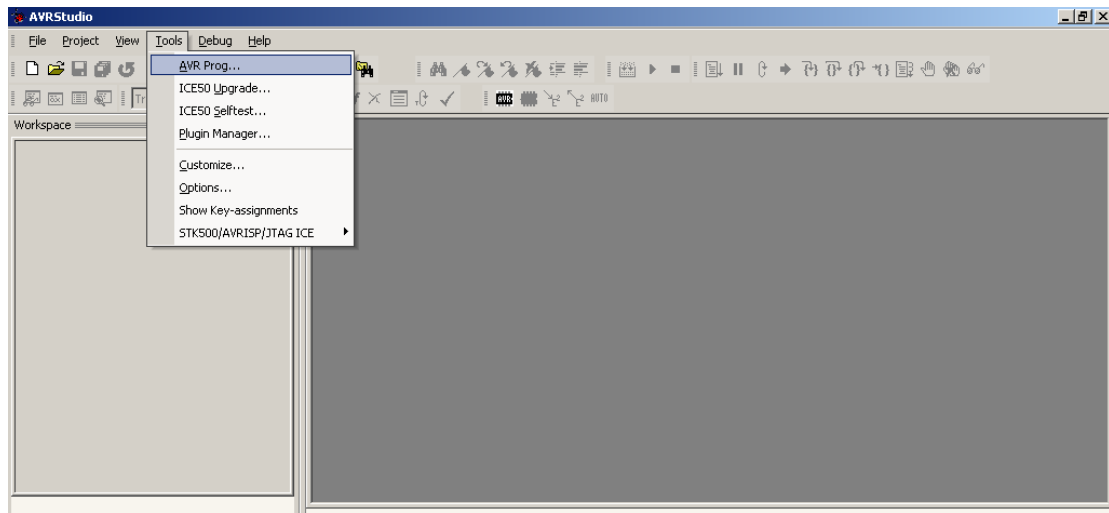


Figure 30: Starting AVR Prog in AVR Studio

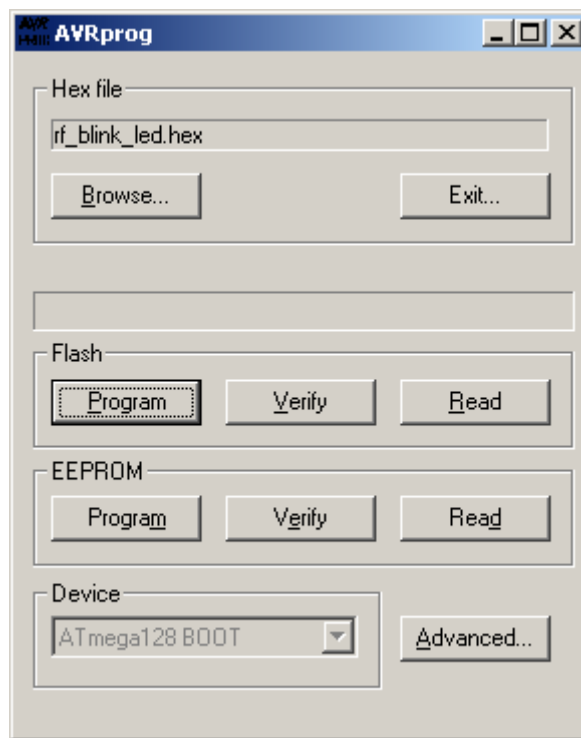


Figure 31: AVRprog bootloader Window

Important:

If the AVR ISP or the JTAG ICE programmer is used to program application examples, the bootloader will be deleted if no protection mode is applied to this section of the microcontroller flash. The bootloader is modified source code from ATMEL AVR application note AVR109, Self programming and compiled with IAR Embedded Workbench.

Introduction to Examples Provided with CC2420DB

To develop software for the CC2420DB three tools are needed.

1. A text editor, e.g. UltraEdit-32 with support for syntax highlighting and running a compiler
2. A compiler/assembler/linker, e.g. WinAVR. The WinAVR is a suite of open source software development tools for the Atmel AVR series of RISC microprocessors for the Windows platform. It includes the GCC compiler for C. This tool provides a framework for most of the features supported by the Atmega128L microcontroller used with the CC2420DB Demonstration Board.
3. Software platform provided by the processor manufacturer

Libraries and Examples

The CC2420 libraries and examples include a variety of source files to ease and support the program development. Besides the standard C libraries, the source/support files are divided into 4 main groups: Hardware Definition Files, Hardware Abstraction Library, Basic RF library and finally application example.

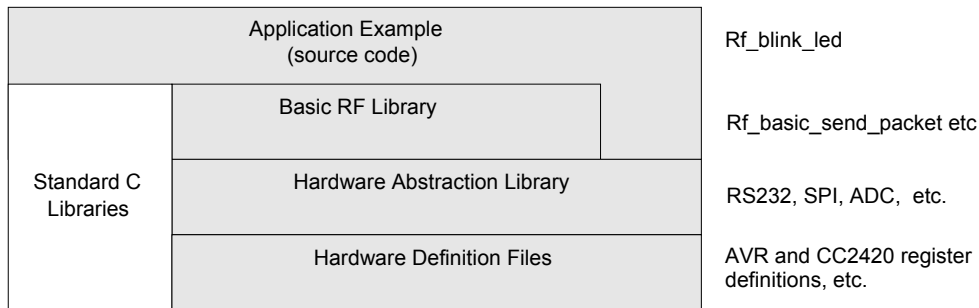


Figure 32 Library stack

Hardware Definition Files

The hardware definition files (include files) define the hardware registers in the MCU and the CC2420. They also include useful macros for the CC2420DB, and all definitions generally support the C language.

Hardware Abstraction Library (HAL)

To support quick and easy program development Chipcon provides a library of macros and functions that simplify hardware access on the CC2420. These are located in the Hardware Abstraction Library (HAL) and implement a hardware abstraction interface for the user program. As a result the user program can access the microcontroller peripherals, etc. via function/macro calls, without specific knowledge about the hardware details.

Basic RF Library

The "Basic RF" library contains simple functions for packet transmission and reception with the Chipcon CC2420 radio chip. The intention of this library is mainly to demonstrate how the CC2420 is operated, and not to provide a complete and full functional packet protocol. The protocol uses 802.15.4 MAC compliant data and acknowledgment packets, however it contains only a small subset of the 802.15.4 standard:

- Association, scanning, beacons is not implemented
- No defined coordinator/device roles (peer-to-peer, all nodes are equal)
- Waits for the channel to become ready, but does not check CCA twice (802.15.4 CSMA-CA)
- Does not retransmit packets
- Can not communicate with other networks (using a different PAN identifier)
- Short addresses only

Software Example Programmed on CC2420DB

The CC2420DB is preprogrammed with an RF link application example called "rf_blink_led", and the source code is available for download from our web site.

The program demonstrates the use of the CC2420DB libraries, including the basic RF library. The packet protocol being used is a small subset of the IEEE 802.15.4 standard. It uses an 802.15.4 MAC compatible frame format, but does not implement any other MAC functions/mechanisms (e.g. CSMA-CA). The "Basic RF" library can thus not be used to communicate with compliant 802.15.4 networks.

A pair of CC2420DBs running this program will establish a point-to-point RF link on channel 26, using the following node addresses:

- PAN ID: 0x2420 (both nodes)
- Short address:
 - 0x1234 if the joystick button is pressed down at startup
 - 0x5678 if the joystick is moved in any direction at startup

Please note that there is no so-called (PAN) coordinator.

Data packets containing a 5-byte payload will be transmitted when the pot meter is turned, or joystick center button is held down. The first byte of the payload contains the pot meter value, which is used to control the PWM duty cycle on the receiving node. The other bytes are random (never initialised). The program uses the following LED indicators:

Red: Transmission failed (acknowledgment not received)

Yellow: Transmission OK (acknowledgment received)

Orange: Remote controlled dimmer

Green: Packet received

Data packet description

The frame formats used are data and acknowledges frames according to the IEEE 802.15.4 standard. The Frame Control Field (FCF) is fixed, address information field is short addresses only and the payload data is inserted by the application.

The preamble, Start of Frame Delimiter, the FCS is generated and inserted in the package by CC2420. The data frame and acknowledge frame is depicted in the Figure 33 and Figure 34.

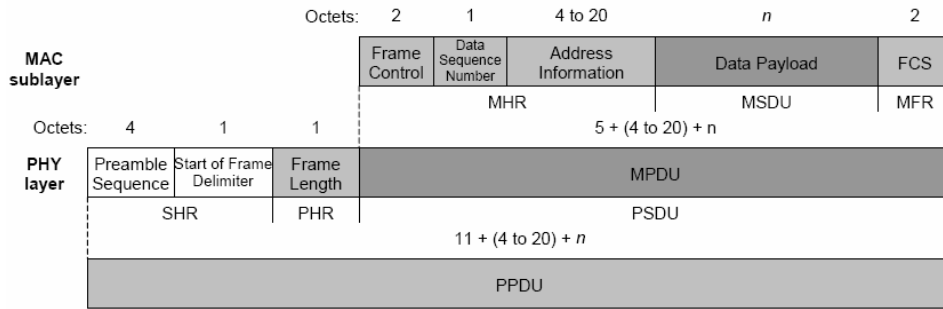


Figure 33 IEEE 802.15.4 data frame

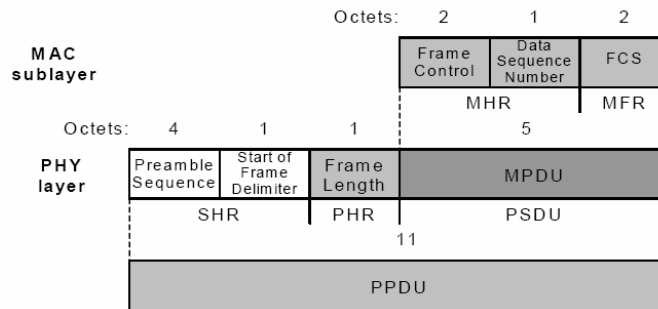


Figure 34 IEEE 802.15.4 acknowledge frame

Data transmission protocol

The data transmission protocol is very simple, only using data and acknowledgement frames.

Step-by-Step procedure:

1. Joystick is pressed on device 2
2. CC2420 device 2 transmits the data frame
3. CC2420 device 1 receives the data frame
4. Microcontroller (uC) device 1 gets the FIFOP interrupt from CC2420, RXFIFO is read, frame is checked for FCS, and the green LED is blinked
5. CC2420 device 1 waits 12 symbol periods before automatically transmitting the acknowledgement reply
6. CC2420 device 2 receives acknowledge frame
7. Microcontroller (uC) device 2 gets the FIFOP interrupt from the CC2420, RXFIFO is read, and frame is verified for FCS
8. The yellow LED will blink without further reply indicating the acknowledgement. If transmission failed (no acknowledgement) the red LED will toggle.

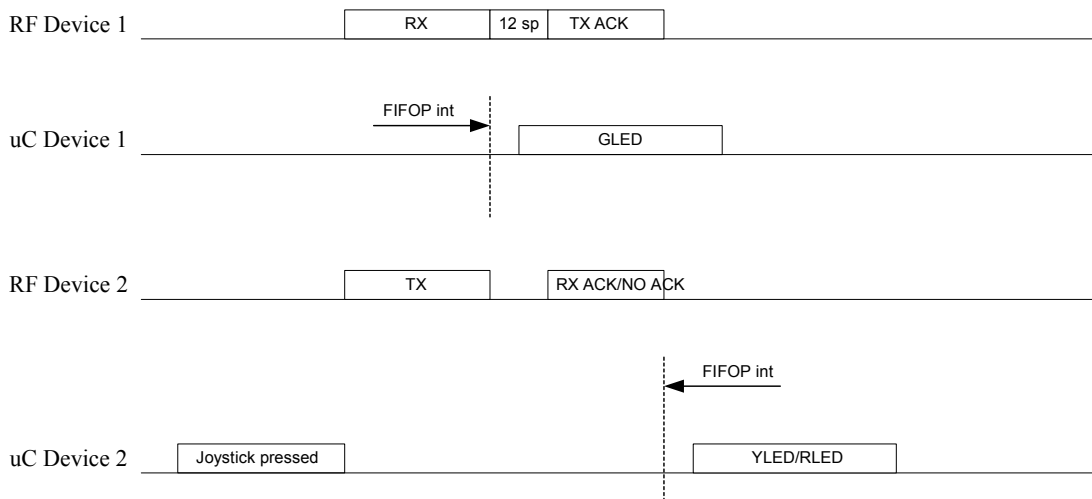


Figure 35 Data Transmission Protocol

Troubleshooting

It does not work

- Make sure that the power supply is connected to the correct pins on the power connector.
- Is the supply voltage correctly polarized? If not, the protection diode will prevent any current from flowing. + and – are indicated on the PCB. On the DC jack, the tip is + and the ring is –.

I cannot program the AVR using the serial port

- If you have programmed the AVR using an external programmer, you must use an external programmer to program the AVR with the bootloader before you can use the serial port to program the AVR again.

Boot reset vector enable

- If you are using the JTAG ICE, JTAG ICE mkII or AVR ISP with AVR studio, you must use disable the boot reset vector enable.

Bootloader

- If you get the following message in AVR Studio when trying the bootloader:
 - No supported board found AVR Prog 1.37
- Make sure that the bootloader step-by-step procedure is followed. If the procedure is followed the bootloader can be corrupted.
- Follow this procedure to reprogram the bootloader:
 - Download bootloader for CC2420 from our web site
 - Connect AVR ISP to connector P6 and a PC with running AVR Studio. See chapter about bootloader for details.
 - Apply power
 - Verify correct fuses settings according to CC2420DBK User Manual
 - Program the flash with the downloaded bootloader a90 (hex) file.
 - Please remember to set the lock bit, Mode 2 for the Bootloader Section.
 - Start the bootloader according to the User Manual
 - Program the flash with the “rf_blink_led” example using bootloader AVR Prog and the rf_blink_led.hex file.
 - Disconnect RS-232 cable and AVR ISP
 - Run the rf_blink_led code example using the CC2420DBK and verify correct performance.
- If you get the following message in AVR Studio when trying the bootloader:
 - Programming - Verifying error address: 0x0000 Expect: 0x940C Receive: 0x0000
- Follow the procedure above to reprogram the bootloader with the latest version from the Chipcon web side and check Atmega128L fuse settings.
http://www.chipcon.com/index.cfm?kat_id=2&subkat_id=12&dok_id=115

References

CC2420 Datasheet:

http://www.chipcon.com/index.cfm?kat_id=2&subkat_id=12&dok_id=115

Atmel ATmega 128L data sheet:

http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf

Atmel Application note: AVR109 Self-programming (Bootloader CC2420DBK based on this application note)

http://www.atmel.com/dyn/general/tech_doc.asp?doc_id=8053

AVR Studio:

http://www.atmel.com/dyn/products/tools.asp?family_id=607

AVR GCC:

<http://www.avrfreaks.net/AVRGCC/index.php>

WinAVR:

<http://sourceforge.net/projects/winavr/>

IEEE 802.15.4 standard

<http://standards.ieee.org/getieee802/802.15.html>

Document History

Revision	Date	Description/Changes
1.0	31.03.2004	Initial release.
1.1	06.05.2004	Updated schematics and BOM according to CC2420DB revision 1.2. References to figures corrected. Added antenna gain figure and information of Baud rate deviation on RS-232 interface to PC. Included information about the protection bit that prevents accidental overwriting of the bootloader section of flash.
1.2	26.07.2004	Updated LED text layout and BOM with new crystal specifications according CC2420DB revision 1.4. Added information about external RAM, ZigBee memory requirements, low power mode and CC2420DB temperature range. Updated microcontroller fuses and bootloader step-by-step procedure.
1.3	03.11.2004	Updated memory requirements. Added procedure on how to start a new debugging project with CC2420DB using JTAG ICE mkII. Added details about the 64-bit IEEE address preprogrammed on the each CC2420DB. Updated information about the preprogrammed fuse settings. Added information about current measurements and RS-232 driver software control. Updated Troubleshooting for bootloader.

Address Information

Web site: <http://www.chipcon.com>
E-mail: wireless@chipcon.com
Technical Support Email: support@chipcon.com
Technical Support Hotline: +47 22 95 85 45

Headquarters:

Chipcon AS
Gaustadalléen 21
NO-0349 Oslo
NORWAY
Tel: +47 22 95 85 44
Fax: +47 22 95 85 46
E-mail: wireless@chipcon.com

US Offices:

Chipcon Inc., Western US Sales Office
19925 Stevens Creek Blvd.
Cupertino, CA 95014-2358
USA
Tel: +1 408 973 7845
Fax: +1 408 973 7257
Email: USsales@chipcon.com

Chipcon Inc., Eastern US Sales Office
35 Pinehurst Avenue
Nashua, New Hampshire, 03062
USA
Tel: +1 603 888 1326
Fax: +1 603 888 4239
Email: eastUSsales@chipcon.com

Sales Office Germany:

Chipcon AS
Riedberghof 3
D-74379 Ingersheim
GERMANY
Tel: +49 7142 9156815
Fax: +49 7142 9156818
Email: Germanysales@chipcon.com

Sales Office Asia:

Chipcon AS
Unit 503, 5/F
Silvercord Tower 2, 30 Canton Road
Tsimshatsui, Hong Kong
Tel: +852 3519 6226
Fax: +852 3519 6520
Email: Asiasales@chipcon.com

Sales Office Korea & South-East Asia:

Chipcon AS
37F, Asem Tower
159-1 Samsung-dong, Kangnam-ku
Seoul, 135-798 Korea
Tel: +82 2 6001 3888
Fax: +82 2 6001 3711
Email: KAsiasales@chipcon.com

Sales Office Japan:

Chipcon AS
#403, Bureau Shinagawa
4-1-6, Konan, Minato-Ku,
Tokyo, Zip 108-0075
Japan
Tel: +81 3 5783 1082
Fax: +81 3 5783 1083
Email: Japansales@chipcon.com