

Volume

1

A-WIT TECHNOLOGIES INC.

... a passion for execution ...

CS470000 RF Transceiver Reference Guide Manual

Version 1.1

A-WIT TECHNOLOGIES INC.

CS470000 RF Transceiver Reference Manual

© A-WIT Technologies Inc.
Phone (800) 985-AWIT • Fax (800) 985-2948

Table of Contents

Registering Your C Stamp or C Stamp	
Related Product	1
Introduction to the CS470000 RF	
Transceiver	1
Technical Specifications	1
General Information	2
FCC and IC Notices	3
Notices	4
Getting Support	5
Installing the Microchip MPLAB and C	
Compiler Software	5
Installing the A-WIT C Stamp Quick	
Programmer	5
Installing the USB Software	7
Setting Up the C Stamp Software	
Templates	7
Documentation	7
Creating your RF Transceiver Programs	8
Downloading and Running Your Program	12
RADIOCMD_CS470000	14
RADIOIN_CS470000	17
RADIOOUT_CS470000	18
Terms and Conditions	20

Introduction to the CS470000 RF Transceiver

The CS470000 RF module provides a very easy and low-cost solution for sending and receiving data between C Stamps or between a C stamp and a PC from 1000 ft. (depending on conditions). The module is compatible with the C Stamp microcomputer's supplies and signal levels, and has a power-down feature to conserve energy, as required in battery based projects. Controlling the module is made easy with A-WIT's supplied software commands that use the serial capability of the C Stamp. This simple interface is all that is required to control the RF module.

Registering Your C Stamp or C Stamp Related Product

At A-WIT Technologies we respect your privacy; however, we do ask you to register your C Stamp or C Stamp related product, so you can receive free of charge product updates. The registration procedure is simple. Just send an e-mail to tech_support@a-wit.com with the word "REGISTRATION x" in the subject line, where "x" is the product number that you purchased. If you purchased more than one product, send an e-mail for each different product.

Introduction to the CS470000 RF Transceiver

The CS470000 RF Transceiver Module is a drop-in wireless solution that can add RF connectivity to any data system. It transfers a standard asynchronous serial data stream.

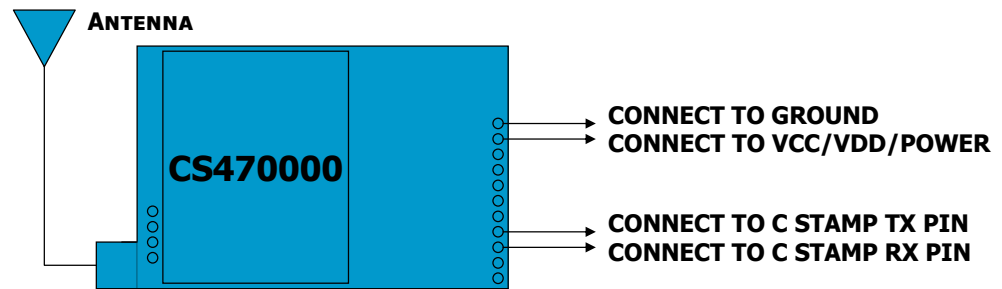
Technical Specifications

- PCB Size: 1.6" x 2.8"
- Power: 2.85 - 5.50 VDC

- Current: ~10 mA normal operation
- Transmit Current: 55 mA
- Receive Current: 35 mA
- Power-Down Current: < 20 μ A
- Data Rate: 9,600 baud
- Frequency: ISM 902 - 928 MHz
- Transmission up to 1000 ft. (300 m), based on environment conditions
- Compact and breadboard-friendly
- Compatible with C Stamp microcomputer

General Information

The following figure describes the minimal pin-out connectivity of the CS470000 RF Transceiver Module. The TX and RX connections to the C Stamp can be to any I/O pins, except the built-in asynchronous serial transmitter (Pin 24) and receiver (Pin 25), and the ATN pin (Pin 26).



In the figure above, in the right hand side connector, Pin 1 is the lower pin, and Pin 11 is the pin at the top of the connector. The C Stamp microcontroller can shut down and wake-up the RF module by controlling the Sleep Pin (Pin 2) of the RF module. To put the RF module in sleep mode, assert the Sleep Pin HIGH, and to wake-up the RF module, de-assert the Sleep Pin LOW. If an RF transmission or reception is in progress, the module will complete that communication before it goes into sleep mode.

Pin 3 (DO) is the data output from the RF Module to the host microcontroller.

Pin 4 (DI) is the data input to the RF Module from the host microcontroller.

Pin 6 (RESETn) is an input to the RF Module that is almost always HIGH and only LOW when the radio is reset. Since the RF module has an onboard reset monitor, this pin can be left disconnected. Pin 6 utilizes a 10 K Ω Pull-Up resistor already installed in the module. However; this pin can be used to reset the RF Module under the control of the host microcontroller.

Pin 7 (RXLED) of the RF Module is an output that is normally driven LOW, but it is driven HIGH by the radio to indicate RF data reception. This pin can be tied through a resistor to an LED to get visual indication that RF data is being received.

Pin 8 (TXLED) is an output from the RF Module that is normally driven HIGH and can be tied through a resistor to an LED to indicate the that the module has power. It also pulses on/off when data is transmitted over-the-air.

Pin 10 (POWER) is the power supply pin.

Pin 11 (GROUND) is the ground pin.

NOTE: If after you send the first packet to the radio modem via a RADIOOUT and then listen with a RADIOIN for a reply from a remote radio, you will get an RFIN_BUFULL return code but the buffer either contains an echo of the transmitted packet, a partial packet, or nulls. The results vary. The only thing consistent is that it always happens after the first RADIOOUT. All remaining transmissions are correct. So that first transmission reception pair should be discarded. The radio is behaving like a Hayes modem which is in command mode and not yet switched to data mode. This happens when the modem settings are as it is shipped from the factory.

FCC and IC Notices

The CS470000 RF Module complies with Part 15 of the FCC rules and regulations.

The CS470000 RF Module complies with IC (Industry Canada) Certification.

Getting Started

This chapter is a quick start guide to using the CS470000 RF Transceiver Module with the C Stamp. This assumes you have two C Stamps with appropriate connection kits or development boards with the RESET and START circuitries, at least 1 LED connected to each C Stamp, the RF Transceiver Modules properly connected to the C Stamps, and an optional utility input button circuit connected to the first C Stamp. You will also need a programming cable, power supply, PC running Windows® 2000/XP/Media, with a quantity of RAM recommended for the OS, sufficient free hard disk drive space for the software installations, CD-ROM drive, Internet access (recommended only), and available port compatible with your programming cable.

Notices

CSTAMP™ and CSTAMP™ Related Hardware Products, Software Products and Documentation are developed and distributed by A-WIT Technologies, Inc. All rights reserved by A-WIT Technologies, Inc. A-WIT SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL A-WIT BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

MPLAB C-18 and MPLAB C-18 Users Guide is reproduced and distributed by A-WIT Technologies, Inc. under license from Microchip Technology Inc. All rights reserved by Microchip Technology Inc. MICROCHIP SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO

EVENT SHALL MICROCHIP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

Getting Support

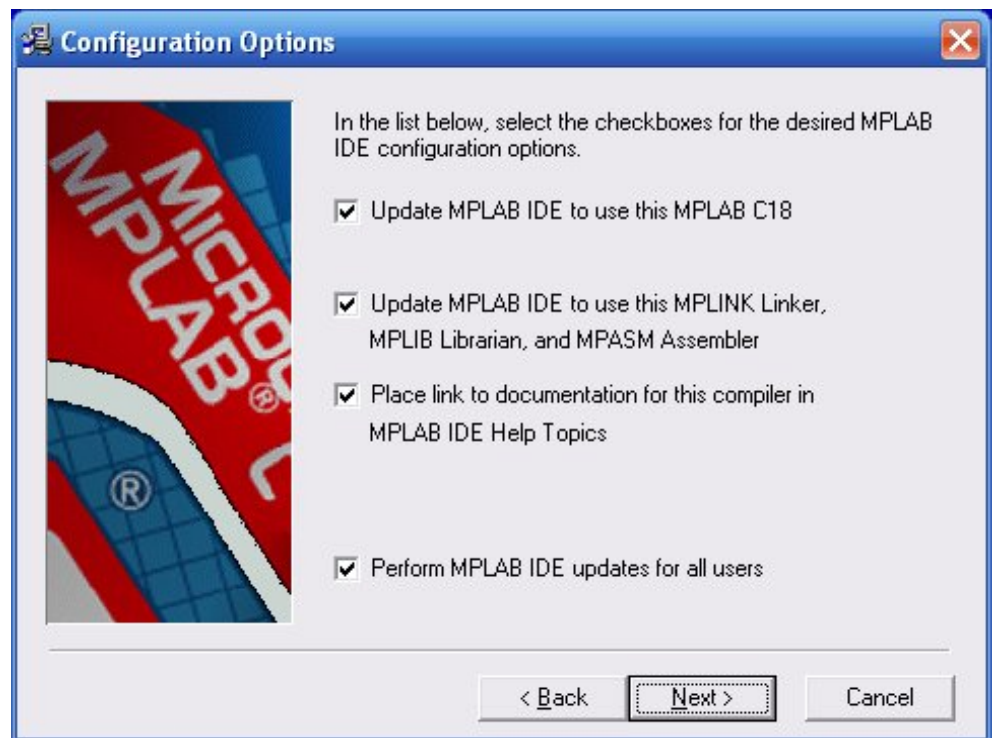
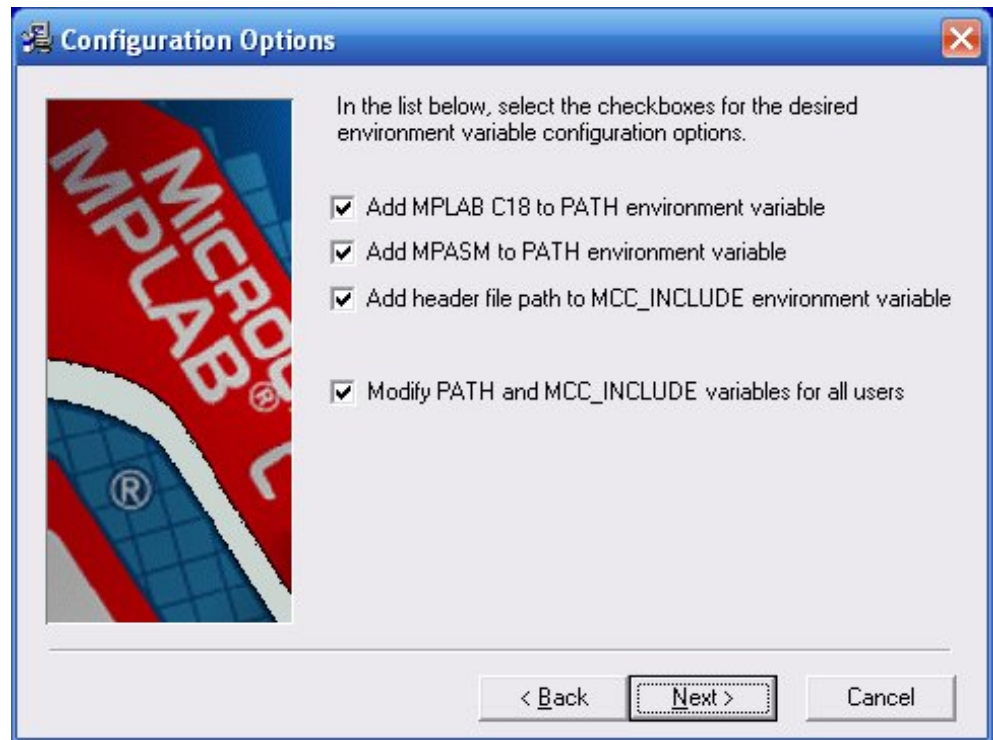
If possible, please check the C Stamp website www.c-stamp.com under SUPPORT for any updates to documentation, changes, or notices that may have become available since your Installation CD was produced. If you continue to have any issues for which a solution is not found in the aforementioned website, please e-mail tech_support@a-wit.com for help.

Installing the Microchip MPLAB and C Compiler Software

The first step is to install the Microchip MPLAB software that you will use to develop your programs.

Insert your A-WIT provided Installation CD in your CD drive. Go to the MPLAB directory in the CD and double click on the “MPLAB vX.XX Install” file in that directory. Follow the installation steps, prompts, and directions provided by the installer software, accepting all the default options.

After the MPLAB installation is complete, switch to the C18 directory in the CD, and double click on the file in that directory. Follow the installation steps, prompts, and directions provided by the installer software, accepting all the default options. The only exceptions to accepting all the default options is that on the 5th and 6th windows of the installation process for the C18 Compiler, you have to select everything as shown in the figures below. This will ensure that MPLAB is configured to use the C18 Compiler.



Installing the A-WIT C Stamp Quick Programmer

To install the A-WIT C Stamp Quick Programmer, switch to the CSTAMPQP directory in the CD using Windows Explorer, and double click on the file in that directory. Follow the installation steps, prompts, and directions provided by the installer software, accepting all the default options.

Installing the USB Software

If you purchased a product with a USB download cable, make sure that the A-WIT provided CD is in the CD drive of your PC and insert the USB cable in the USB port of your PC. Windows auto detects the new USB device. If Windows prompts you to install drivers for the USB cable device, follow the installation steps, prompts, and directions provided by the installer software, accepting all the default options.

After the USB adapter has been installed, open a Windows Explorer window from the Accessories sub-menu in the Start menu, and right click on My Computer. Proceed to select Properties, and then select the Hardware tab. Click on the Device Manger button, and expand the Ports (COM & LPT) branch. Make a note of the COM port that has been assigned to the USB-to-Serial adapter. This is the port that should be selected in the C Stamp programmer software.

Setting Up the C Stamp Software Templates

To set up the C Stamp Software Templates, switch to the CSTAMP_Template directory in the CD using Windows Explorer, and double click on the file in that directory. Follow the installation steps, prompts, and directions provided by the installer software, accepting all the default options.

Documentation

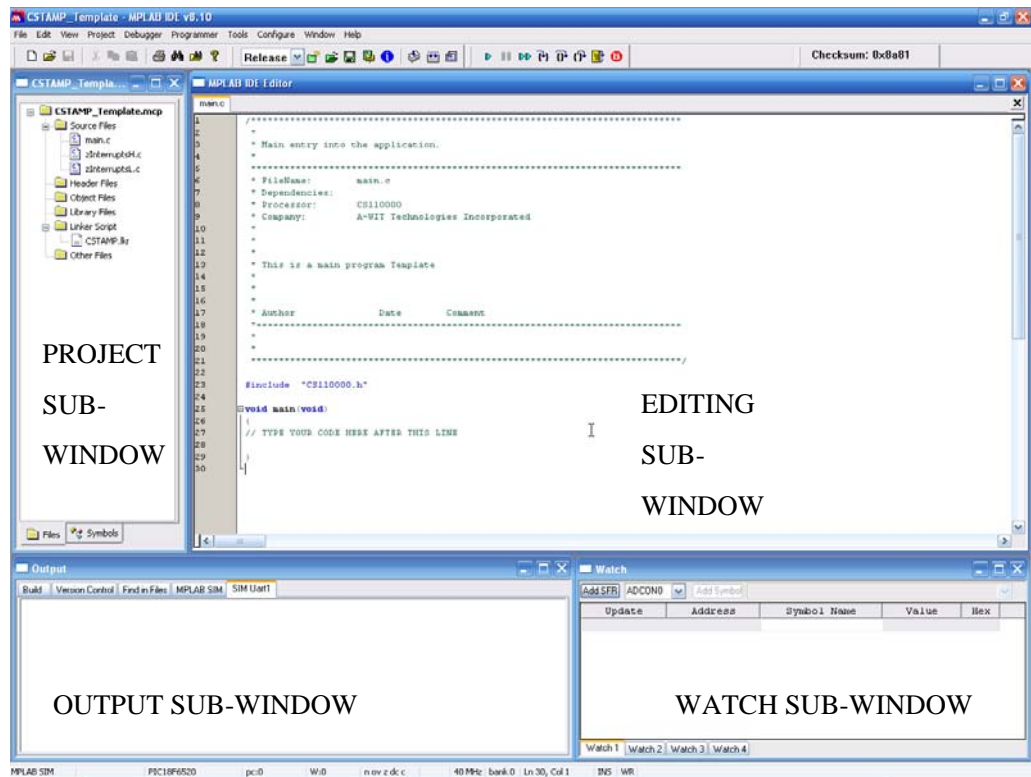
Copy the DOCS directory from the C Stamp Installation CD to your C:\A-WIT directory. This directory contains all the C Stamp related documentation in PDF format.

Creating your RF Transceiver Programs

For each program follow the following instructions. Create a directory where you want to have all the files for your program; for example, RFA_APP. We recommend making this directory under your C:\A-WIT directory, so you can have all your CSTAMP related files in one place.

Copy the all files in your C Stamp Software Templates directory C:\A-WIT\CSTAMP_Template to the directory you just made.

Open the Microchip MPLAB IDE application. As shown the following figure, the IDE has several sub-windows. Depending on the resolution of your screen, your sub-windows may have a different layout. However; you can move and resize these into the position that you want to fit your screen, and your layout for that particular project will get saved upon answering yes to the prompt of saving the workspace when you exit the software development environment.



Go to the “File” menu to “Open Workspace...”. Then navigate to your program directory, and open CSTAMP_Template.mcw.

Right click on CSTAMP_Template.mcp in the “Project” sub-window, and “Save as...” the name of your program project after you have navigated to your program directory. For example, your program project could be named “RFA_APP”. Now when you open the Microchip MPLAB IDE (Integrated Development Environment), and go to your program directory to open the workspace for your program, you will see a .mws file with the name of your program preceding it. This is the file that you should open any time you want to work on your program.

Double click on the main.c source and type the following code fragment where it is indicated. You can omit the comments for brevity, as they are written here to offer clarifications of what the code does. Do pay attention; however, to the indentation of the code blocks between curly brackets for loops, if statements, etc. Although indenting the code is not a requirement for the compiler to parse your code (i.e. any blank spaces are ignored by the compiler), it does help tremendously to make your code much more readable, and consequently, it makes finding any errors easier. Keywords and function names in the code fragment below are bolded.

After you START the C Stamp in user mode as explained in the “Downloading and Running Your Program” section (this will not be the RESET/BOOT/DOWNLOAD mode), the program will run. Each program assumes that you have the RF Transceiver Module connected to Pin 1 for transmission into the RF Module, and Pin 2 for reception from the RF Module, the LED in each setup connected to Pin 46, and the utility button in the first setup connected to Pin 37. The first program monitors the button, and if the button is pushed, it will send a signal (a byte with a value of 1) to the second setup. The second setup will monitor transmission from the first setup, and if a byte with the value of 1 is received, it will light the LED and it will send a byte with a value of 1 back to the first setup signaling that the “turn on the LED” command was received. After this action is taken, the program will end execution. The first program will also monitor transmission from the second to receive the acknowledgement that the “turn on the LED” command was received. When this handshake is received, it will turn on its own LED, and end execution. The programs stay in that mode indefinitely until you restart each of them by pushing and releasing the RESET button while holding the START button and then letting go of the latter.

FIRST PROGRAM SEGMENT

```
// Declare some necessary variables
BIT button_pushed; // For button
BYTE data[] = {1};

// Power off LED connected to pin 46
STPIND(46, LOW);
```

```

// Wait for button connected to pin 37 to be pushed
button_pushed = FALSE;
while(!button_pushed){
    button_pushed = BUTTON(37, LOW, HIGH, 5);
}

// Send byte of 1
RADIOOUT_CS470000 (data, 1, 1);

// Reset the data
data[0] = 0;

// Get handshake
RADIOIN_CS470000 (0, data, 1, 2);

// Turn on LED if communication was successful
if(data[0]) STPIND(46, HIGH);

STOP();

```

SECOND PROGRAM SEGMENT

```

// Declare some necessary variables
BYTE data[] = {0};

// Power off LED connected to pin 46
STPIND(46, LOW);

// Get data
RADIOIN_CS470000 (0, data, 1, 2);

// If data is 1, turn on LED and send acknowledgement
if(data[0]){
    STPIND(46, HIGH);
}
// Send byte of 1
RADIOOUT_CS470000 (data, 1, 1);
}
else{
// send back a 0
RADIOOUT_CS470000 (data, 1, 1);
}

STOP();

```

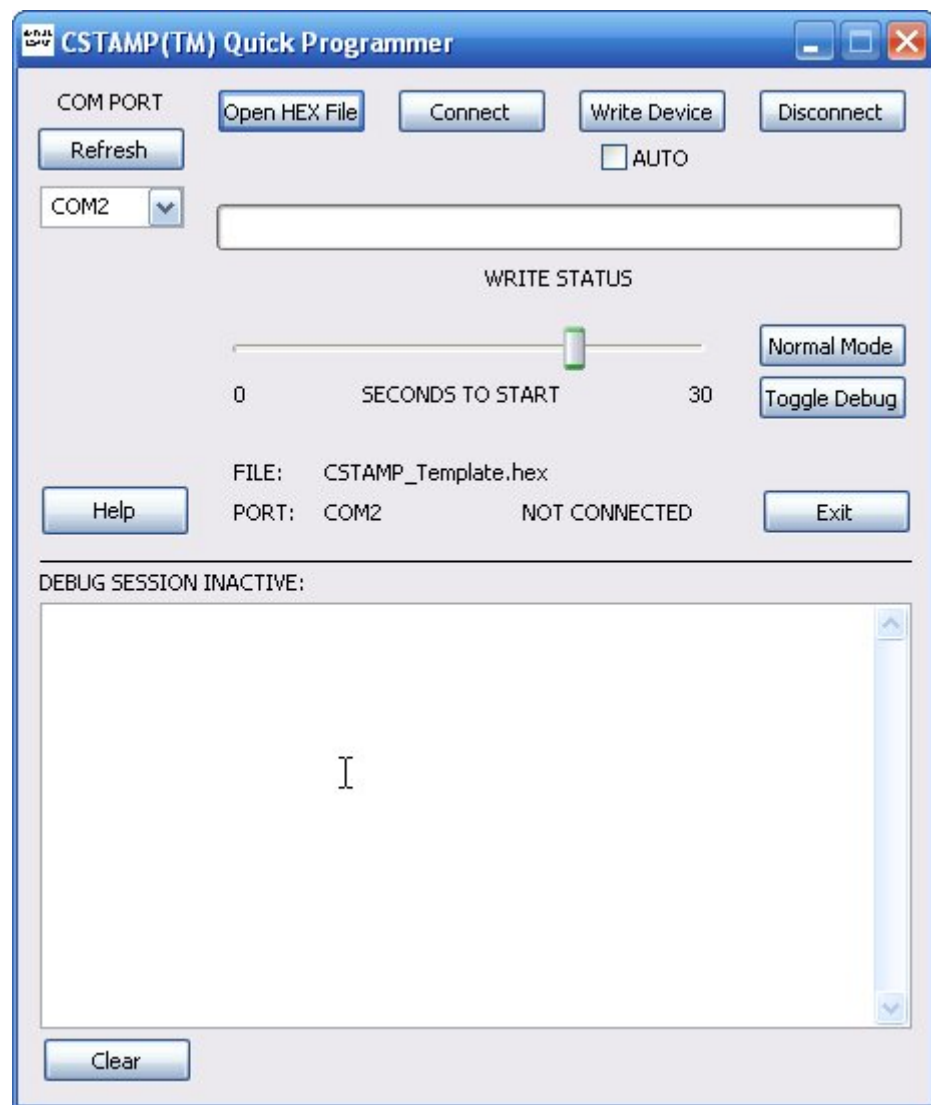
Save your program from the “File” menu or by clicking on the appropriate icon in the tool bar. Then “Build All” from the “Project” menu or from the tool bar.

If the code was typed correctly, you will have a file in your program directory with the name of your program project and a .HEX extension. An example is RFA_APP.HEX. This is the file that you will download to the C Stamp, as explained up ahead.

If you get an error message or an indication that your program did not build successfully in the “Output” sub-window of the IDE, you probably have one or more syntax error(s). Double click on the line of the “Output” sub-window that mentions the error, and the program line that most likely contains the error will be indicated in the sub-window where you were editing your program. Correct as necessary, and “Build All” again until you get a successful .HEX file output.

Downloading and Running Your Program

Power up your KIT, and connect the KIT to the PC with the provided cable. Upon power up, the C Stamp will be in RESET/BOOT/DOWNLOAD mode. To go back to this mode at any time, just push and let go of the RESET button. Then open the A-WIT C Stamp Quick Programmer application shown in the next figure.



The first step is to choose the serial port that you are using from the drop-down menu. Then click on “Refresh”, so that the program registers your selection. Your selection should show in the status area of the program next to “PORT:”. Then click on “Open HEX File” and load/select the HEX file that you had previously created during the

development of your program. The status area should indicate that the file has been loaded successfully. This is what will be downloaded to the C Stamp. Then click on “Connect”, and the PC will be connected to the KIT, and the status area should indicate so. To download the HEX file to the C Stamp, just click on “Write Device”, and you should see the progress bar after a few seconds, as the HEX file is downloaded. At this point, you can click on “Disconnect” to disconnect the PC from the KIT, disconnect the serial cable from both the PC and the KIT, and start your program manually at the KIT. To do this just push and let go of the RESET button while pushing the START button. Then you can let go of the START button. Alternatively, you can click on “Normal Mode” to start your program from the PC. This will also disconnect the program/PC from the KIT. Then you can disconnect the serial cable from the PC and the KIT. You can also instruct the CSTAMP™ Quick Programmer to wait several seconds before starting your program from the PC and disconnecting by adjusting the “SECONDS TO START” slide. This feature is useful in case you want to keep the PC connected with the serial cable, but need time to manually set up something in a circuit that you have built. If this is not the case it can just be left at the default of “0”, and your program will start from the PC right away. After you click “Normal Mode” and your program is started, the CSTAMP™ Quick Programmer will not be communicating with the C Stamp any longer, so if you want to reconnect, you must click on “Connect” again.

Accessory Specific Functions and Commands Reference

This chapter describes the functions and commands that are specific to the software support of different types of accessories that are available from A-WIT Technologies to complement the function and projects developed with the C Stamp. The user should consult the manual for a specific accessory for full information on connectivity and usage.

RADIOCMD_CS470000

```
BYTE RADIOCMD_CS470000(NIBBLE command,  
                        WORD Wparameter, BYTE RXpin,  
                        BYTE TXpin);
```

The **RADIOCMD** function sends a command to the CS470000 RF Module according to the table below. If the function is successful, it returns a non zero number depending on the command; otherwise, it returns zero. This could mean that there was an error in the arguments of the function or some other problem.

command is a variable/constant/expression (1 – 3) indicating the RADIO command to send.

Wparameter is a word variable/constant/expression that is used by different commands. This parameter is not used by all the commands. If the parameter is not used by a command, any value can be passed to the function (e.g. **ZERO**), and it will be ignored. However; something always must be passed in all the arguments. If the usage of the parameter is not specified in the table below, then it is not used and ignored by the function.

RXpin is a variable/constant/expression that specifies the I/O pin to be used as a receiver. This pin will be set to input mode.

TXpin is a variable/constant/expression that specifies the I/O pin to be used as a transmitter. This pin will be set to output mode.

command Symbol	Value	Command	Description
RF_DB	1	Receive Signal Strength	Returns the signal strength (in decibels) of the last received packet. Range: 37 - 106
RF_DT	2	Destination Address	Sets the address that identifies the module's address and the destination of the RF packet. The address is specified by wparameter , and it can be any value from 0x0 to 0xFFFF in hexadecimal, or from 0 to 65535 in decimal. The factory default for the RF Module address is 0x0 in hex. Only radio modems having matching addresses can communicate with each other. If successful, this command returns 200.
RF_MK	3	Address Mask	Sets the address mask to configure local and global address space. The address mask is specified by wparameter , and it can be any value from 0x0 to 0xFFFF in hexadecimal, or from 0 to 65535 in decimal. The factory default for the RF Module address mask is 0xFFFF in hex. If successful, this command returns 200.

Destination Addresses and Masks provide the means to set up global or local addresses for establishing module groups, subnets, etc. The Destination Address network layer provides for more granular isolation of radio modems. The Destination Addresses and Masks can be used to:

- Set up point-to-point and point-to-multipoint network configurations
- Provide greater flexibility in establishing module groups, subnets, etc.

Each radio modem in a network can be configured with a 16-bit Destination Address to establish selective communications within a network. This address is set to one of 65535 values using the **RF_DT** (Destination Address) Command. The default Destination Address is 0.

All radio modems with the same Destination Address can transmit and receive data among themselves. Radio modems having different Destination Addresses still detect and listen to the data (in order to maintain network synchronization); however, the data is discarded rather than passing on through the DO pin.

CS470000 Radio Modems are packet based. This means that all data shifted into one module is packetized and sent out the antenna port. Because these RF modules use a peer-to-peer architecture, all modules will receive the packet and decide whether to pass it to the host or to throw it away. Each transmitted packet contains information about the transmitting module.

Any module that receives a packet will check the address values and decide what to do with the packet. The options are as follows:

- Receive the packet as a global packet
- Receive the packet as a local packet
- Discard the packet

The mask parameter can be used to allow a base module to receive data from a range of addresses. It may also be used to configure "subnets" of modules that communicate in a group together.

The RF Module uses the bit-wise "AND" operation to qualify the Destination Addresses and Address Masks. This operation is performed bit by bit on each of the 16 bits in the Destination Address and Address Mask parameters. The table below shows the Bit-Wise AND Truth Table.

<i>Bit-Wise AND Operation</i>		
Operand 1	AND Operand 2	= Result
0	0	0
0	1	0
1	0	0
1	1	1

The Address Mask can be used as an additional method of facilitating communications between modules. The Address Mask can be set to one of 65535 possible values using **RF_MK** (Address Mask) Command. The default value of the **RF_MK** Parameter is 0xFFFF.

All transmitted data packets contain the Destination Address of the transmitting module. When a transmitted packet is received by a module, the Destination Address of the transmitter (contained in the packet) is logically “ANDed” (bitwise) with the Address Mask of the Receiver. If the resulting value matches the Destination Address of the Receiver, or if it matches the Receiver Address Mask, the packet is accepted. Otherwise, the packet is discarded.

Note: When performing this comparison, any “0” values in the Receiver Address Mask are treated as irrelevant and are ignored.

For example, suppose that two RF modules are setup as follows. RF Module 1 has a Destination Address of 0x1 and an Address Mask of 0xFFFF, and RF Module 2 has a Destination Address of 0x1 and an Address Mask of 0x000F. In this case RF Module 1 will always transmit to RF module 2, but it will receive from any module with any Destination Address. RF Module 2; however, will also transmit to RF Module 1, but it will only receive from modules with Destination Addresses from 0 through 15 (0x0 through 0xF in hex). This range, of course, includes RF Module 1.

RADIOIN_CS470000

```
NIBBLE RADIOIN_CS470000(WORD timeout, BYTE buffer[],  
                           BYTE Lbuffer, BYTE RXpin);
```

The **RADIOIN** function receives asynchronous serial data from the CS470000 RF Module via any I/O pin, except the built-in asynchronous serial transmitter (Pin 24) and receiver (Pin 25), and the ATN pin (Pin 26).

timeout is a variable/constant/expression (0 – 65535) that tells **RADIOIN** how long to wait in mS for incoming data. If data does not arrive in time or if the buffer gets full before the timeout condition occurs, the function will return with the appropriate return code. The value of 0 is special; it indicates that the function will wait until it receives enough data to fill the buffer before it returns. The usage of timeouts applies to each data byte being received.

buffer is the name of the array of **BYTES** that the function will use as storage to return the received data to the calling function. This can be an array of length equals to 1 byte, and the maximum length of the buffer is 255 bytes.

Lbuffer is the maximum number of bytes that can be placed in **buffer**. If fewer bytes are received than what can be placed in the buffer, the buffer gets filled from the low to high direction of its index address space (i.e. from 0 to **Lbuffer-1**).

RXpin is a variable/constant/expression that specifies the I/O pin to be used as a receiver. This pin will be set to input mode.

On exit, the function returns one of the following exit codes.

<i>return code</i>	<i>value</i>	<i>Meaning</i>
RFIN_ARGERR	0	There was at least one error in the arguments of the function. Function did not execute.
RFIN_BUFULL	1	Function returned with the buffer full of data.
RFIN_TIMEOUT	2	The function exited after a timeout condition occurred. The highest index of buffer (Lbuffer-1) contains the index of the last byte that was received successfully before the timeout occurred. If this location contains the value 0xFF, this indicates that no data was received.
RFIN_OVRERR	3	An overrun error occurred. The highest index of buffer (Lbuffer-1) contains the index of the last byte that was received successfully before the error occurred. If this location contains the value 0xFF, this indicates that no data was received.

RADIOOUT_CS470000

```
BIT RADIOOUT_CS470000(BYTE buffer[], BYTE Lbuffer,
                       BYTE TXpin);
```

The **RADIOOUT** function transmits asynchronous serial data to the CS470000 RF Module via any I/O pin, except the built-in asynchronous serial transmitter (Pin 24) and receiver (Pin 25), and the ATN pin (Pin 26).

buffer is the name of the array of **BYTES** that the function will send. This can be an array of length equals to 1 byte, and the maximum length of the buffer is 255 bytes.

Lbuffer is the maximum number of storage bytes to be sent in the buffer. The buffer gets processed for transmission from the low to high direction of its index address space (i.e. from 0 to **Lbuffer**-1).

TXpin is a variable/constant/expression that specifies the I/O pin to be used as a transmitter. This pin will be set to output mode.

On exit, the function returns one of the following exit codes.

<i>return code</i>	<i>value</i>	<i>Meaning</i>
RFOUT_ARGERR	0	There was at least one error in the arguments of the function. Function did not execute.
RFOUT_BUFEMP	1	Function returned after transmitting all data in the buffer.

Terms and Conditions

Quality Assurance

A-WIT has stringent quality control procedures in place to insure the best quality products.

90-Day Limited Warranty

A-WIT Technologies, Inc warrants its products against defects in materials and workmanship for a period of 90 days. If you discover a defect, A-WIT Technologies, Inc. will, at its option, repair, replace, or refund the purchase price. After 90 days, products can still be sent in for repair or replacement, but there will be a \$10.00USD minimum inspection/labor/repair fee (not including return shipping and handling charges).

14-Day Money-Back Guarantee

If, within 14 days of having received your product, you find that it does not suit your needs, you may return it for a refund. A-WIT will refund the purchase price of the product in the form of a check, excluding shipping/handling costs, once the product is received. This refund does not apply if the product has been altered or damaged. If you decide to return the products after the 14-day evaluation period, a 20% restocking fee will be charged against a credit.

Disclaimer

Warranty does not apply if the product has been altered, modified, or damaged. A-WIT makes no other warranty of any kind, expressed or implied, including any warranty of merchantability, fitness of the product for any particular purpose even if that purpose is known to A-WIT, or any warranty relating to patents, trademarks, copyrights or other intellectual property. A-WIT shall not be liable for any injury, loss, damage, or loss of profits resulting from the handling or use of the product shipped.

How to Return a Product

When returning, you must first e-mail sales@a-wit.com for a Return Merchandise Authorization number. No packages will be accepted without the RMA number clearly marked on the outside of the package. After inspecting and testing, we will return your product, or its replacement using the same shipping method used to ship the product to A-WIT within 30 days. In your package, please include a daytime telephone number and a brief explanation of the problem.

Please contact our Sales Department at sales@a-wit.com if you have any questions regarding our warranty policy or if you are requesting an RMA number.

