



RDM-T8FZ
RF Transceiver Module for 868 MHz
Application Notes
Rev 1.0



Reindeer Systems Pvt Ltd

Excellence Through Innovation

No. 67, 53rd Street,
9th Avenue, Ashok Nagar,
Chennai – 600083
India.



Document Revision History

Revision No.	Date	Description/Changes
V1.0	15/9/2009	Initial Release



Table of Contents

1.Introduction	4
2.Specifications	4
3.Application Firmware Explanation.....	5
3.1 Void Main (void).....	5
3.2 Interrupt Service Routine (ISR).....	6
4.Header Files	8
5. Using the sample code with other Microcontrollers	9
6. Contact Us.....	10
6.1 Technical Support.....	10
6.2 Sales Support	10



1. Introduction

This application note describes the usage of RDM-T8FZ module as a transceiver for transmitting and receiving data. The main objective is to explain how wireless transmission of data can be achieved from one device to another using a microcontroller's UART.

2. Specifications

The sample code is developed based on the R8C series of microcontrollers from Renesas. The specifications are mentioned below.

Platform : HEW (High Performance Embedded Workshop)
Language : Embedded C
Controller : Renesas R8C tiny series
Daughter board : RDM-T8FZ
Mother board : RDSRF-232
Frequency : 868 MHz
Interface with RDM-T8FZ : SPI

Application : Data communication between RF transceivers.
Serial configuration : Baud rate: 9600bps, Parity: none, Data Bits: 8, Stop Bits: 1

Below mentioned is the list of files used in this application

- 1) RDM_T8FZ_Main.c – This contains the main program.
- 2) RDM_T8FZ_Function.h – This file contains all the function definitions.
- 3) RDM_T8FZ_IOdefine.h – This file contains all definition for all the I/O ports.
- 4) RDM_T8FZ_Prototype.h – This file contains all the variable declarations.
- 5) RDM_T8FZ_Register.h – This file contains all the register declaration for the RF Section.



3. Application Firmware Explanation

3.1 Void Main (void)

The Program enters into the main () function after power-up. All the functions used for configuring the microcontroller and the RF module have been explained below.

Function: **Oscillator_configuration ();**

This function is used to configure the various register of the microcontroller like register control - 0 used to configure the XIN clock, register control-1 used to configure the low speed on-chip oscillator, select and enable the high speed on-chip oscillator. In the sample application firmware the internal high speed on-chip oscillator is used as the main system clock. The operating frequency of the clock is 20MHz.

Function: **Peripheral_Initialization ();**

- a. **sfr_init ();** This function is used to configure the input, output ports and also for enabling the corresponding pull-ups for the I/O's.
- b. **serial_init ();** This function is used to configure the UART- 0 of the microcontroller. All the serial settings like serial baud rate, parity etc are configured here.
- c. **timerx_init ();** This function is used for configuring the internal timer of the microcontroller. The timer is configured such that it gives an interrupt every 1 millisecond. This timer is used for creating any delays or any other timing needed.

Function: **CC1101_Initialization ();**

This function used to initialize the RDM-T8FZ module and its registers.

- a. **resetcc1101 ();** This function is used to reset the RDM-T8FZ module. The module is reset using the reset strobe.
- b. **configurecc1101 ();** Configuration of CC1101 is done by programming the 8-bit registers. The optimum configuration data can be found by using the SmartRF® Studio software. Complete descriptions of the registers are given in the datasheet. The Module has to be configured as mentioned in the sample code for transmission and reception of data.
- c. **BurstWrite2CC1101 (CC1101PATABLE, &power [0], 1);** This function is used to configure the output power level in dbm of the RF module.



Function: issuestrobe (SRX);

This function is used to enable the receive mode of the module. As soon as this strobe is issued the module will go into receive mode and can receive data.

Function: int_init ();

This function is used to initialize the external interrupt – 0 of the microcontroller. This interrupt is used to by the module to indicate the microcontroller that a valid data has been received in the RXFIFO.

3.2 Interrupt Service Routine (ISR)

3.2.1 Void uart0 (void)

This uart0 interrupt service routine is used to receive the data from hyper terminal. When data is received from PC it is stored in the 'tx_buffer'. When the complete packet has been received the 'txisr()' function will be enabled so that the packet can be transferred to the RF module.

3.2.2 Void txisr (void)

In this ISR the transfer of data from the microcontroller to the RF module takes place. Before the start of data transfer the 'issuestrobe (STX)' function is used to enable the transmission mode for the RF module.

Function: issuestrobe (STX);

This strobe is used to enable the RF transmission mode of the RF module. After this strobe has been issued any data written into the TXFIFO will be transmitter over the air.

Function: transmitter ();

The SPI interface is used to write data to the TX FIFO. When writing to the TX FIFO it is the responsibility of the MCU to avoid TX FIFO overflow. A TX FIFO overflow will result in an error in the TX FIFO content.



This function has two for loops running within it . The first for loop used to load the buffered data (which will be transmitted) from tx_buffer to CC1101TXFIFO using BurstWrite2CC1101 function. The second one is used to clear the tx_buffer after the data transmission to be completed.

3.2.3 Void int0 (void)

This interrupt function is used to call the receive function whenever the RF module indicate that a valid data has been received.

Function: receive ();

When receive mode is enabled, the RF module will remain in receive mode till an 'SIDLE' or 'STX' strobe is issued. The reception of data is indicated by the GDO0 pin of the RF module. This pin is configurable and can be used for other purpose also (refer to datasheet). When a valid sync byte has been received this pin goes into a low state. And after the complete data packet has been received this the status of this pin will change to high. The completion of the packet is decided based on the packet length configured. After the pin status has changed to high data can be read out from the RXFIFO. The SPI interface is used to read from the RX FIFO. When reading the RX FIFO the MCU must avoid reading the RX FIFO past its empty value since a RX FIFO underflow will result in an error in the data read out of the RX FIFO. Once the complete packet is transferred into the microcontroller the data is then sent out to the PC through the UART.

Function: BurstreadCC1101();

This function is used to transfer the data from the RXFIFO into the microcontroller for data processing.



4.Header Files

The program has 4 header files

1. RDM_T8FZ_IOdefine.h
2. RDM_T8FZ_Function.h
3. RDM_T8FZ_Prototypes.h
4. RDM_T8FZ_Register.h

RDM_T8FZ_IOdefine.h:

This file contains the macro definition of communication pins. Pin like sclk, so, si, csn, gdo2, gdo0 etc,

RDM_T8FZ_Function.h:

This file contains various function declarations of the program.

RDM_T8FZ_Prototypes.h:

This file consists of the global variable declaration type unsigned char, unsigned int and unsigned long.

RDM_T8FZ_Register.h:

This file contains the CC1101 registers and its addresses.



5. Using the sample code with other Microcontrollers

The sample code is developed for Renesas R8C/27 microcontroller. For interfacing the RF module with a different microcontroller, use only the below mentioned functions.

- a. `Resetcc1101 ();`
- b. `Configurecc1101 ();`
- c. `BurstWrite2CC1101(unsigned char,unsigned char*,unsigned char);`
- d. `issuestrobe (SRX);`
- e. `transmit_mode ();`
- f. `receive_mode ();`
- g. `issuestrobe (STX);`
- h. `transmitter ();`
- i. `receive ();`
- j. `BurstreadCC1101();`
- k. `unsigned char spiwrite(void);`
- l. `void spiwrite(unsigned char);`
- m. `Write2CC1101(unsigned char,unsigned char);`

Also include the header file "RDM_T8FZ_Register.h".



6. Contact Us

6.1 Technical Support

Reindeer Systems Pvt Ltd has built a solid technical support infrastructure so that you can get answers to your questions when you need them.

Our technical support engineers are available Mon-Fri between 9:30 am and 6:00 pm Indian standard time. The best way to reach a technical support engineer is to send an email to support@reindeersystems.com. E-mail support requests are given priority because we can handle them more efficiently than phone support requests.

6.2 Sales Support

Our sales department can be reached via e-mail at sales@reindeersystems.com or by phone at 91-44-45022335/337.

Our sales department is available Mon-Fri between 9:30 am and 6:00 pm.



Reindeer Systems Pvt Ltd

Excellence Through Innovation

No. 67, 53rd Street,
9th Avenue, Ashok Nagar,
Chennai – 600083
India.

Phone: 91-44-45022335, 91-44-45022337

Fax: 91-44-45022336

Website: www.reindeersystems.com