Application Note: JN-AN-1069

802.15.4 Wireless UART with Flow Control

This Application Note describes how to create a wireless UART link between the UARTs on two JN5148 or JN5139 evaluation kit boards. The features of the application include:

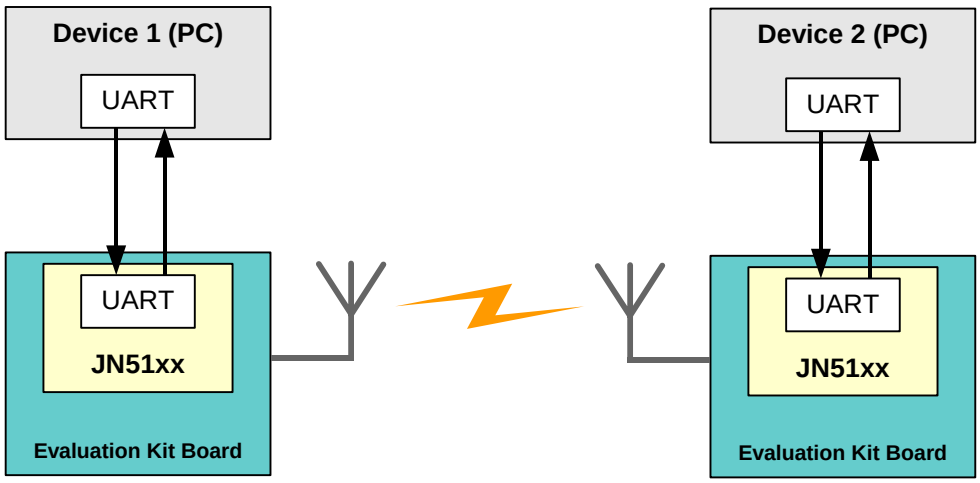
- **Replacement of serial cables with wireless connectivity**
- **Quick addition of low-cost wireless connectivity to products using a UART link for communications**
- **UART hardware flow control using the RTS/CTS lines to control the flow of data into and out of the UART**
- **Flow control using a simple protocol to control the flow of data over the radio**

This application was developed using the 802.15.4 Stack API and Integrated Peripherals API, both part of the Jennic SDK available free-of-charge from the Support area of the Jennic web site (www.jennic.com/support).

Application Overview

The application uses a pair of development kit boards to form a wireless network. Data received by a UART on one board is transmitted via the wireless radio link to the second board and output by the second board's UART, and vice versa. This allows two devices (such as PCs) to communicate via a wireless radio link.

This is illustrated in the figure below:



© Jennic 2009

Two different binaries are built as part of this application:

- The Co-ordinator initially creates the network, then runs in the same way as the End Device.
- The End Device application initially joins the network, then runs in the same way as the Co-ordinator.

It is assumed that a permanent power source is available at both ends of the wireless link, allowing the radio to always be active and ready to transmit or receive data.

Starting the Application

Connect each board to a PC using the serial cables provided with the evaluation kit. UART 0 on the boards is used by default.

- The Co-ordinator illuminates LED2 when the network is started and the End Device is allowed to join the network.
- The End Device illuminates LED2 once it has joined the network.
- Once running, LED1 flashes regularly.

Running the Application

> **!** ***Caution****: Ensure that the Flash Programmer software is disconnected from the serial port on the PC if using the same serial port for programming and running the application.*

A terminal emulator, such as HyperTerminal, can be used to send data between the boards. The serial connection is 115200 bps, 8 data bits, no parity, 1 stop bit with hardware flow control, by default.

Data entered into one board's HyperTerminal window is passed to the board's UART and then transmitted over the radio to the other board. Here, it is received and passed via the board's UART to that board's HyperTerminal window.

Using Controller Board with LCD

When the Co-ordinator or End Device binary is run on a controller board with an LCD screen, the LCD will display a looping count of the data bytes transmitted to and received from the UART (and over the radio link).

© Jennic 2009

Software Design

Apart from the creation and joining of the network, the software which runs on each board is identical.

The source files required to build the application are described below.

AN1069_WUART_FC_Coord.c

This file contains starting code and main loop for 802.15.4 Co-ordinator:

- Selects the quietest channel
- Starts the network
- Allows an End Device to associate with the Co-ordinator, then illuminates LED2
- A timer event calls the protocol transmit function to handle data for transmission, and flashes LED1
- Data received on the stack is passed to the protocol receive function for handling

AN1069_WUART_FC_EndD.c

This file contains starting code and main loop for 802.15.4 End Device:

- Scans the channels to search for the Co-ordinator
- Associates with the Co-ordinator, once found, and then illuminates LED2
- A timer event calls the protocol transmit function to handle data for transmission and flashes LED1
- Data received on the stack is passed to the protocol receive function for handling

radio.c

This files contains code to receive and transmit general 802.15.4 data packets.

serialq.c

This file contains code which maintains a pair of circular queues for use with the UART.

The receive queue is filled with data received by the UART. When the queue increases above a certain level, the UART is disabled from receiving further data, causing the hardware flow control to turn on.

The receive queue is emptied by the protocol transmitting the data over the radio. When the queue falls below a certain level, the UART is enabled to receive data again, causing the hardware flow control to turn off.

The transmit queue is filled with data received by the protocol over the radio. The protocol will not allow the other board to send data if there is insufficient space in the transmit queue.

The transmit queue is emptied as data is transmitted by the UART.

 © Jennic 2009

uart.c

This file contains code which operates the UART using an interrupt handler that is called when characters are received or are ready to be transmitted:

- Data received by the UART is added to the receive serial queue
- Data for transmission by the UART is removed from the transmit queue and transmitted by the UART

Hardware flow control using RTS and CTS is implemented in one of two ways:

- **Manual Hardware Flow Control** is used when the define **UART_AUTOFLOW** is set to **FALSE**. In this mode, the RTS line is lowered by software as the receive queue begins to fill and is raised as the queue begins to empty. An interrupt is programmed to occur when the CTS line is changed. The state of the CTS line is read and the transmission of data is allowed or denied dependent upon the state of the CTS line.
- **Automatic Hardware Flow Control** can be enabled in the UART by setting the define **UART_AUTOFLOW** to **TRUE**. In this mode, when the 16-byte receive FIFO begins to receive the 16th byte, the UART's RTS line is lowered and then raised once space is available in the FIFO again. When the UART's CTS line is low, the UART will not transmit any data from the transmit FIFO.

> **!** ***Caution:*** *The FTDI 3V Serial-to-USB cables and drivers supplied with the evaluation kits do not react quickly enough to the RTS being lowered. This results in an extra character being sent to the UART, which is then lost. When using two UARTs with automatic flow control, they should react quickly enough to prevent such data loss. Manual Hardware Flow Control is therefore recommended when using the evaluation kits and FTDI cables.*

protocol.c

This file contains code which manages the protocol to transmit and receive data over the radio.

The protocol maintains a single packet containing the data to be transmitted next:

- If the packet is empty, it is filled with the next data from the receive queue.
- If a packet is ready for transmission, the protocol is initiated.

The basic protocol for A transmitting a data packet to B is as follows:

- A sends an enquiry packet to B
- If B has space in the UART transmit queue, B responds with an ack packet.
- A sends the data packet to B
- If B has space in the UART transmit queue, it adds the data to the queue and responds with an ack packet.
- If B does not have space to receive a packet, it responds with a nack packet. A then pauses for a short while before retrying.

It is possible that both A and B send an enquiry packet at the same time. In such a situation, the following rules apply:

- The device that last sent a data packet yields to the other device. If B were the last device to send data to A then B would yield to A by sending an ack packet. A would then be able to send data.
- If A is able to send data in the previous scenario, A will pause before sending another enquiry packet, in the expectation that B is ready to send data.
- The enquiry packets indicate if the device is going to yield. In the case where both devices are going to take the same action, the End Device will yield to the Co-ordinator.

display.c

This file contains general purpose routines to maintain a single page on the LCD screen.

Each line can have a text string and two hexadecimal values displayed. Functions are provided to set each of these items independently and a draw function is provided to draw the page.

wuartdisp.c

This file contains code to maintain an LCD screen (using **Display.c**), showing running counters of the data transmitted and received by the UART and radio.

wuart.h

This is the common header containing global configuration data.

 © Jennic 2009

Compatibility

The software provided with this Application Note has been tested with the following Jennic kits and SDK versions:

Product Type	Part Number	Version	Supported Chips	Supported Protocols
Evaluation Kit	JN5139-EK000	-	JN5139	802.15.4, JenNet
	JN5139-EK010	-	JN5139	802.15.4, JenNet, ZigBee
	JN5148-EK010	-	JN5148	802.15.4, ZigBee PRO
SDK Libraries	JN-SW-4030	v1.5	JN5139, JN5121	802.15.4, JenNet, ZigBee
	JN-SW-4040	v1.1	JN5148	802.15.4, ZigBee PRO
SDK Toolchain	JN-SW-4031	v1.1	JN5139, JN5121	-
	JN-SW-4041	v1.0	JN5148	-

Building and Downloading the Application

The software provided with this Application Note can be built for the Jennic JN5148 or JN5139 wireless controller. However, the available build methods differ between these chip types:

- JN5148 applications can be built using the Eclipse IDE or makefiles
- JN5139 applications can be built using the Code::Blocks IDE or makefiles

Note that different makefiles are provided for JN5148 and JN5139.

In order to build the supplied software, the application's folder must be placed in the **Application** folder of the Jennic SDK installation:

- For JN5148: **<JENNIC_SDK_ROOT>\Application**
- For JN5139: **<JENNIC_SDK_ROOT>\cygwin\jennic\SDK\Application**

where **<JENNIC_SDK_ROOT>** is the path into which the Jennic SDK was installed (by default, this is **C:\Jennic**). The **Application** directory is automatically created when you install the Jennic SDK.

Build the application as described in the appropriate section below, depending on whether you intend to use Eclipse, Code::Blocks or makefiles.

Using Eclipse (JN5148 Only)

To build the application and load it into JN5148 boards, follow the instructions below:

1. Ensure that the project directory is located in

<JENNIC_SDK_ROOT>\Application

where **<JENNIC_SDK_ROOT>** is the path into which the SDK was installed.

2. Start the Eclipse platform and import the relevant project files (**.project** and **.cproject**) as follows:

a) In Eclipse, follow the menu path **File>Import** to display the **Import** dialogue box.

b) In the dialogue box, expand **General**, select **Existing Projects into Workspace** and click **Next**.

© Jennic 2009

c) Enable **Select root directory**, browse to the Jennic **Application** directory and click **OK**.

d) In the **Projects** box, select the project to be imported and click **Finish**.

3. Build an application. To do this, ensure that the project is highlighted in the left panel of Eclipse and use the drop-down list associated with the hammer icon in the Eclipse toolbar to select the relevant build configuration – once selected, the application will automatically build. Repeat this for each application in the project.

   The binary files will be created in the relevant **Build** directories, the resulting filenames indicating the chip type (**JN5148**) for which they were built.

4. Load the resulting binary files (Release or Software Debug) into the boards. You can do this using the Jennic JN51xx Flash Programmer, which can be launched from within Eclipse or used directly (and is described in the *JN51xx Flash Programmer User Guide (JN-UG-3007)*).

Using Code::Blocks (JN5139 Only)

To build each application and load it into a JN5139 board, follow the instructions below:

1. Ensure that the project directory is located in

   <JENNIC_SDK_ROOT>\cygwin\jennic\SDK\Application

   where **<JENNIC_SDK_ROOT>** is the path into which the Jennic SDK was installed.

2. Open the appropriate Code::Blocks project file (**.cbp** files in the **CodeBlocksProject** directory) and build.

   The project filenames include the chip type (**JN5139**) for which the binary is to be built.

   The binary file will be created in the **5139_Build** directory, the resulting filename matching that of the project file used to create it.

3. Load the resulting binary file into the board. You can do this using the Jennic JN51xx Flash Programmer, which can be launched from within Code::Blocks or used directly (and is described in the *JN51xx Flash Programmer User Guide (JN-UG-3007)*).

> **!** ***Caution:*** *If problems occur when using your current Code::Blocks version, download the latest version along with the latest version of this Application Note from the Support area of the Jennic web site (www.jennic.com/support).*

 © Jennic 2009

Using Makefiles

The application for each node type (Co-ordinator, End Device) has its own **Build** directory, which contains the makefiles for the application.

Different makefiles are provided for JN5148 and JN5139 – a JN5148 makefile is simply called **Makefile** while a JN5139 makefile is called **Makefile_JN5139.mk**.

To build an application and load it into a JN51xx board, follow the instructions below:

1. Ensure that the project directory is located in

   <JENNIC_SDK_ROOT>\Application for JN5148

   <JENNIC_SDK_ROOT>\cygwin\jennic\SDK\Application for JN5139

   where **<JENNIC_SDK_ROOT>** is the path into which the Jennic SDK was installed.

2. Navigate to the **Build** directory for the application to be built and follow the instructions below for your chip type:

   For JN5148:

   At the command prompt, enter:

   ```
   make clean all
   ```

   Note that for the JN5148, you can alternatively enter the above command from the top level of the project directory, which will build the binaries for both applications.

   For JN5139:

   At the command prompt, enter:

   ```
   make -f Makefile_JN5139.mk clean all
   ```

   In both the above cases, the binary file will be created in the relevant **Build** directory, the resulting filename indicating the chip type (**JN5148** or **JN5139**) for which the application was built.

3. Load the resulting binary file into the board. To do this, use the Jennic JN51xx Flash Programmer, described in the *JN51xx Flash Programmer User Guide (JN-UG-3007)*.

Revision History

Version	Notes
1.0	First release
2.0	Support for JN5148 device added and support for JN5121 device removed

© Jennic 2009

Important Notice

Jennic reserves the right to make corrections, modifications, enhancements, improvements and other changes to its products and services at any time, and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders, and should verify that such information is current and complete. All products are sold subject to Jennic's terms and conditions of sale, supplied at the time of order acknowledgment. Information relating to device applications, and the like, is intended as suggestion only and may be superseded by updates. It is the customer's responsibility to ensure that their application meets their own specifications. Jennic makes no representation and gives no warranty relating to advice, support or customer product design.

Jennic assumes no responsibility or liability for the use of any of its products, conveys no license or title under any patent, copyright or mask work rights to these products, and makes no representations or warranties that these products are free from patent, copyright or mask work infringement, unless otherwise specified.

Jennic products are not intended for use in life support systems/appliances or any systems where product malfunction can reasonably be expected to result in personal injury, death, severe property damage or environmental damage. Jennic customers using or selling Jennic products for use in such applications do so at their own risk and agree to fully indemnify Jennic for any damages resulting from such use.

All trademarks are the property of their respective owners.

Jennic Ltd
Furnival Street
Sheffield
S1 4QT
United Kingdom

Tel: +44 (0)114 281 2655
Fax: +44 (0)114 281 2951
E-mail: info@jennic.com

For the contact details of your local Jennic office or distributor, refer to the Jennic web site:

www.Jennic.com
TECHNOLOGY FOR A CHANGING WORLD