MeshNetics™

ZigBit™

ZigBit™ Evaluation Kit 1.2
User's Guide

Disclaimer

MeshNetics believes that all information is correct and accurate at the time of issue. MeshNetics reserves the right to make changes to this product without prior notice. Please visit MeshNetics website for the latest available version.

MeshNetics does not assume any responsibility for the use of the described product or convey any license under its patent rights.

MeshNetics warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with MeshNetics standard warranty. Testing and other quality control techniques are used to the extent MeshNetics deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

Trademarks

MeshNetics®, ZigBit, eZeeNet, ZigBeeNet, SensiLink, as well as MeshNetics and ZigBit logos are trademarks of MeshNetics Ltd.

All other product names, trade names, trademarks, logos or service names are the property of their respective owners.

Technical Support

Please e-mail your product-related questions to support@meshnetics.com. If you like to speak to one of our product specialists you can call the phone number listed below.

Development Support

Software customization services can be provided on terms and conditions mutually agreed by MeshNetics and end-user.

Contact Information

MeshNetics

9 Dmitrovskoye Shosse, Moscow 127434, Russia

Tel: +7 (495) 725 8125

Office hours: 8:00am – 5:00pm (Central European Time)

Fax: +7 (495) 725 8116

E-mail: support@meshnetics.com

www.meshnetics.com

Table of Contents

List of Figures

List of Tables

1. Introduction

Intended Audience and Purpose

This document is intended for engineers and software developers working with ZigBit Evaluation Kit. The kit may be used to evaluate the performance and features of the ZigBit modules and the eZeeNet software.

Safety and Precautions

The product contains electronics, which are electrically sensitive. Please take necessary precautions when using such devices. MeshNetics does its best to protect the product components from electrostatic discharge phenomena, but we encourage our users to follow common guidelines to avoid electrostatics by using proper grounding, etc.

The product follows the FCC (Part 15)/CE rules applicable to the devices radiating in the uncontrolled environment.

Any modifications of the hardware, its components or improper use of the product can cause an uncontrolled violation of the in-band or out-band radiation levels. It can result in progressing violation of emission level limits, thus causing harmful interference.

Please check your local regulations to make sure that the product's electromagnetic radiation level specified in this document complies.

Precautions

The product radiates power in the microwave band. Although the levels are considered to be low (less than 2 mW), it is reasonable to protect the operating personnel from possible harmful impact of the electromagnetic field. When the parts of the product are turned on, an operator should avoid touching the PCB antenna and the board itself. The recommended distance between an operator and antenna should be more than 20 centimeters.

The AC/DC adapters included into the product contain high voltage circuits. General precautions, like checking the power cord before use if boards are mains powered, should be taken.

The ZigBit™ Evaluation Kit contains fragile components. Please handle with care.

Related documents

[1]   ZigBit™ OEM Module. Product Datasheet. MeshNetics Doc. M-251~01

[2]   eZeeNet™ IEEE802.15.4/ZigBee Software. Product Datasheet. MeshNetics Doc. M-251~02

[3]   eZeeNet™ Software 1.6. SerialNet™ Reference Manual. AT-Command Set. MeshNetics Doc. P-EZN-452~01

[4]   eZeeNet™ Software 1.6. eZeeNet™ API. Reference Manual. MeshNetics Doc. P-EZN-452~02

[5]   MeshBean2-P p/n WDB-A1281-P1 Rev. 2.0. Schematics. MeshNetics Doc. P-MB2P-461~01

[6]   ZigBit™ Module. Application Note. ZigBit Power Consumption Testing. MeshNetics Doc. AN-481~01

[7]   ZigBee Document 053474r08, February 17, 2006

[8]   Serial asynchronous automatic dialing and control. ITU-T Recommendation V.250, 05/99

[9]   IEEE Std 802.15.4-2003 IEEE Standard for Information technology – Part 15.4 Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)

[10]   TSL2550 Ambient Light Sensor With Smbus Interface. TAOS Datasheet TAOS029E. –February 2006.
        http://www.taosinc.com/images/product/document/tsl2550-e67.pdf

[11]   LM73 2.7V, SOT-23, 11-to-14 Bit Digital Temperature Sensor with 2-Wire Interface. National Semiconductor Corporation Datasheet DS201478. July 2006.
        http://www.national.com/pf/LM/LM73.html#Datasheet

[12]   CP2102, Single-Chip USB to UART Bridge, Rev. 1.1 9/05, www.silabs.com

[13]   AVR Studio. User Guide.
        http://www.atmel.com/dyn/resources/prod_documents/doc2510.pdf.

[14]   JTAGICE mkII User Guide.
        http://www.atmel.com/dyn/resources/prod_documents/doc2562.pdf

[15]   avr-libc Reference Manual 1.4.3

[16]   WinAVR User Manual – 20060125/ By Eric B. Weddington.

[17]   Using the GNU Compiler Collection/ By Richard M. Stallman and the GCC Developer Community.

[18]   8-bit AVR Microcontroller with 64K/128K/256K Bytes in-System Programmable Flash. ATMEL Preliminary Doc. 2549J-AVR-09/06.
        http://www.atmel.com/dyn/resources/prod_documents/doc2549.pdf

Abbreviations and Acronyms

API	Application Programming Interface
BOM	Bill of Materials
Channel Mask	Channel mask is a number that defines the set of working channels.
Coordinator	Within ZigBee networks, the ZigBee coordinator is responsible for starting the network and for choosing certain key network parameters. The network may be extended through the use of ZigBee router.
DIP	Dual In-line Package
EEPROM	Electrically Erasable Programmable Read-Only Memory
End-device	In ZigBee networks, the ZigBee end-device provides sensor data sent to a router and is requesting a router periodically in duty cycle. End-device is often subjected to power management restrictions, so it may be in sleeping mode most of the time.
ESD	Electrostatic Discharge
GUI	Graphical User Interface

HAL	Hardware Abstraction Layer
IDE	Integrated Development Environment
JTAG	Digital interface for debugging of embedded devices, also known as IEEE 1149.1 standard interface
LED	Light Emitting Diode
LQI	Link Quality Indicator
MAC	Medium Access Control layer
MCU	Microcontroller Unit. In this document, it also means the processor, which is the core of ZigBit module
MIPS	Million Instructions per Second
NWK	Network layer
PAN ID	Personal Area Network Identifier. In ZigBee, it is 16-bit number which must be unique for each one of multiple networks working on the same frequency channel
PCB	Printed Circuit Board
PHY	Physical layer
RF	Radio Frequency
Router	In ZigBee networks, routers transfer data and control messages through the network using a hierarchical routing strategy. The ZigBee coordinator is also responsible for routing.
RS-232	Serial binary data interconnection interface, which is commonly used in computer serial ports
RSSI	Received Signal Strength Indicator
TOS	Open-source operating system TinyOS
USB	Universal Serial Bus
VCP	Virtual Com Port
WSN	Wireless Sensor Network
ZEK	ZigBit Evaluation Kit
ZDK	ZigBit Development Kit
ZigBee	Wireless networking standard targeted at low-power sensor applications

2. Evaluation Kit Overview

ZigBit™ Evaluation Kit (ZEK) is a simple, out-of-the-box solution designed for evaluating wireless sensor networks (WSNs). It provides evaluation boards based on the ZigBit module and eZeeNet Software.

ZigBit Evaluation Kit includes:

1. MeshBean2 board (2 pieces) with ZigBit module and PCB antenna
2. MeshBean2 board (1 piece) with ZigBit module and dual-chip antenna on ZigBit
3. AC/DC power adapter (3 pieces) with USA and European connectors
4. USB cable (3 pieces)
5. RS-232 cable (2 pieces)
6. Software & Documentation CD (1 piece).

Figure 1. The Evaluation Kit delivery set

MeshBean2 board is intended for the evaluation of the ZigBit module's performance. The ZigBit module with the embedded eZeeNet Software provides the MeshBean2 board's wireless connectivity and makes it function as a node in a ZigBee network.

The MeshBean2 board can be configured to operate as a network coordinator, a router or an end-device, by means of setting DIP-switches (see Section 2.3.2) and/or sending AT-commands. The node's role is defined by the embedded applications.

The boards are delivered with a ZigBit preprogrammed with the Hardware Test and Serial Bootloader firmware. For full list of demo applications see Section 2.4.

All the necessary technical documents on the MeshBean2 board (schematic, BOM, Gerber files, etc.) are available upon request.

2.1. Hardware General Specifications

MeshBean 2 basic parameters are presented in Table 1.

Table 1. MeshBean2 Board Specifications

Parameter	Value
RF	
Compliance	2.4 GHz IEEE 802.15.4-2003
Operating Band	2400–2483.5 MHz
TX Output Power	from -17 dBm to +3 dBm
RF Transceiver	AT86RF230-ZU
Antenna	2.4 GHz (PCB on-board antenna or on-chip antenna)
MCU	
Microcontroller	ATmega1281V
RAM	8 kBytes
Flash Memory	128 kBytes
EEPROM	4 kBytes
Performance	Up to 4 MIPS throughput at 4 MHz Clock
Power	
Power Supply	Dual AA Type Battery, automatically switched to USB or AC/DC adapter
Over-Voltage Protection	Yes
Reverse Polarity Protection	Yes
Operating Voltage Range	1.8...3.6 V
Voltage Supervisor	Yes
Miscellaneous	
Sensors	Digital: Ambient Light/ Ambient Air Temperature
LED Indicators	3 programmable color status LEDs external power supply status LED
Switches	3 DIP switches
Buttons	2 programmable buttons
Size	60 x 63 x 24 mm
Operating Temperature Range	-40°C to 85°C. Minor degradation of clock stability may occur beyond the -20°C to +70°C range.

This User's Guide contains general information only. Detailed specifications of the ZigBit module are available in the ZigBit datasheet [1] which contains exhaustive information on the ZigBit interfaces, voltage levels, power consumption etc.

> **NOTE:**
>
> The absolute maximum ratings for the ZigBit module are specified in the ZigBit datasheet [1]. These are the stress ratings only and functional operation of the device at these conditions is not implied. Exposure to absolute maximum rating conditions for the extended periods may affect device's reliability.

2.2. MeshBean2 Featured Components

2.2.1. ZigBit Module

ZigBit module is an ultra-compact, low-power, high sensitivity2.4GHz 802.15.4/ZigBee OEM module from MeshNetics. ZigBit module is based on Atmel's Z-Link 2.4GHz platform. It includes ATmega1281V Microcontroller and AT86RF230 RF Transceiver.

In ZEK, it is delivered installed on MeshBean2 board.

Two different versions of ZigBit modules are available [1]: a version with balanced RF port for applications where the benefits of PCB or external antenna can be utilized, and a version with a chip antenna satisfying the needs of size-sensitive applications.

2.2.2. Sensors

The board uses light sensor TSL2550T from TAOS (see [10]) and temperature sensor LM73CIMK (see [11]) from National Semiconductors. Both sensors are connected in parallel to the I^2C bus. For more information on the sensors see their datasheets on the manufacturers' corresponding websites www.taosinc.com and www.national.com.

2.2.3. USB-COM Bridge

USB to RS-232 bridge controller CP2102 from Silicon Labs installed on the board provides seamless USB interface (see [12]). As a result the USB port is visible as generic COM port with a particular number. The driver set for Windows and other operating systems can be downloaded from the manufacturer's website www.silabs.com.

2.3. MeshBean2 Board Design

The MeshBean2 board contains the ZigBit module, which is functioning as ZigBee/802.15.4 transceiver. It also includes sensors, buttons, DIP-switches, and some interfaces on the Expansion Connector.

The board includes the following interfaces:

- USB 2.0 port
- RS-232 interface
- Buffered I^2C interface with ESD protection and voltage level translation
- Light and temperature sensors
- 2 push buttons controlling the software
- 1 reset button
- 3 DIP switches
- 3 software-controlled LEDs
- Symmetrical dipole PCB antenna (for balanced RF output version of ZigBit only)
- JTAG connector for software download and debugging

- Power connector (3 V)
- 20-pin expansion connector to access specific ZigBit's interfaces
- Battery compartment for AA-size batteries
- 3 configuration jumpers
- 3 clamps for power consumption measurements.

Also, the board contains an internal voltage regulator to supply most of the components with 3.6 V. This is needed if ZigBit's MCU is to be run at 8 MHz[1].

NOTE:

Normally ZigBit module is powered directly by the batteries, USB or AC/DC adapter (via protection circuitry); however, Jumper J2 (see Table 5) can switch ZigBit to 3.6 V supply.

See Figure 2 for MeshBean2 layout and Figure 3 for the board functional diagram.
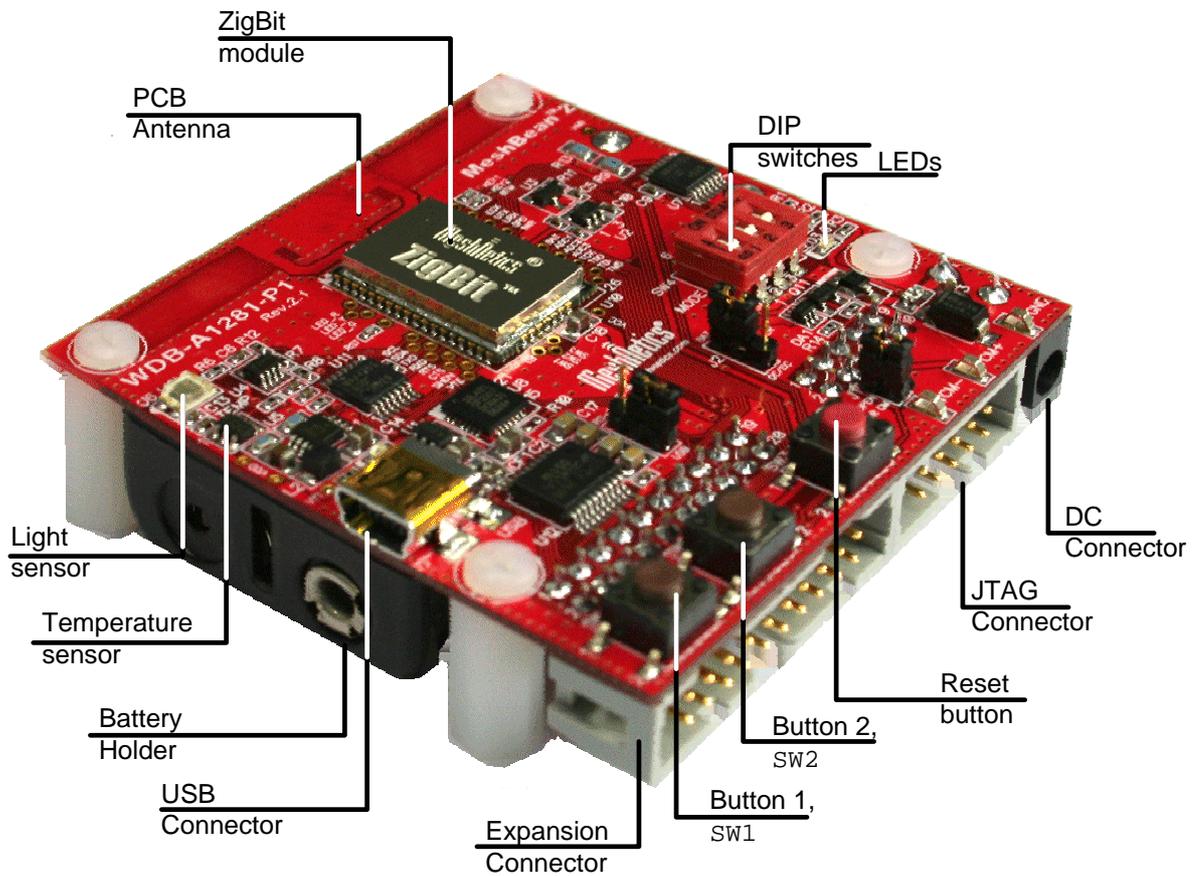
Figure 2. MeshBean2 with PCB on-board antenna

---

[1] 8MHz requires changes in the eZeeNet Software that normally runs at 4 MHz in order to extend the voltage range and decrease power consumption.
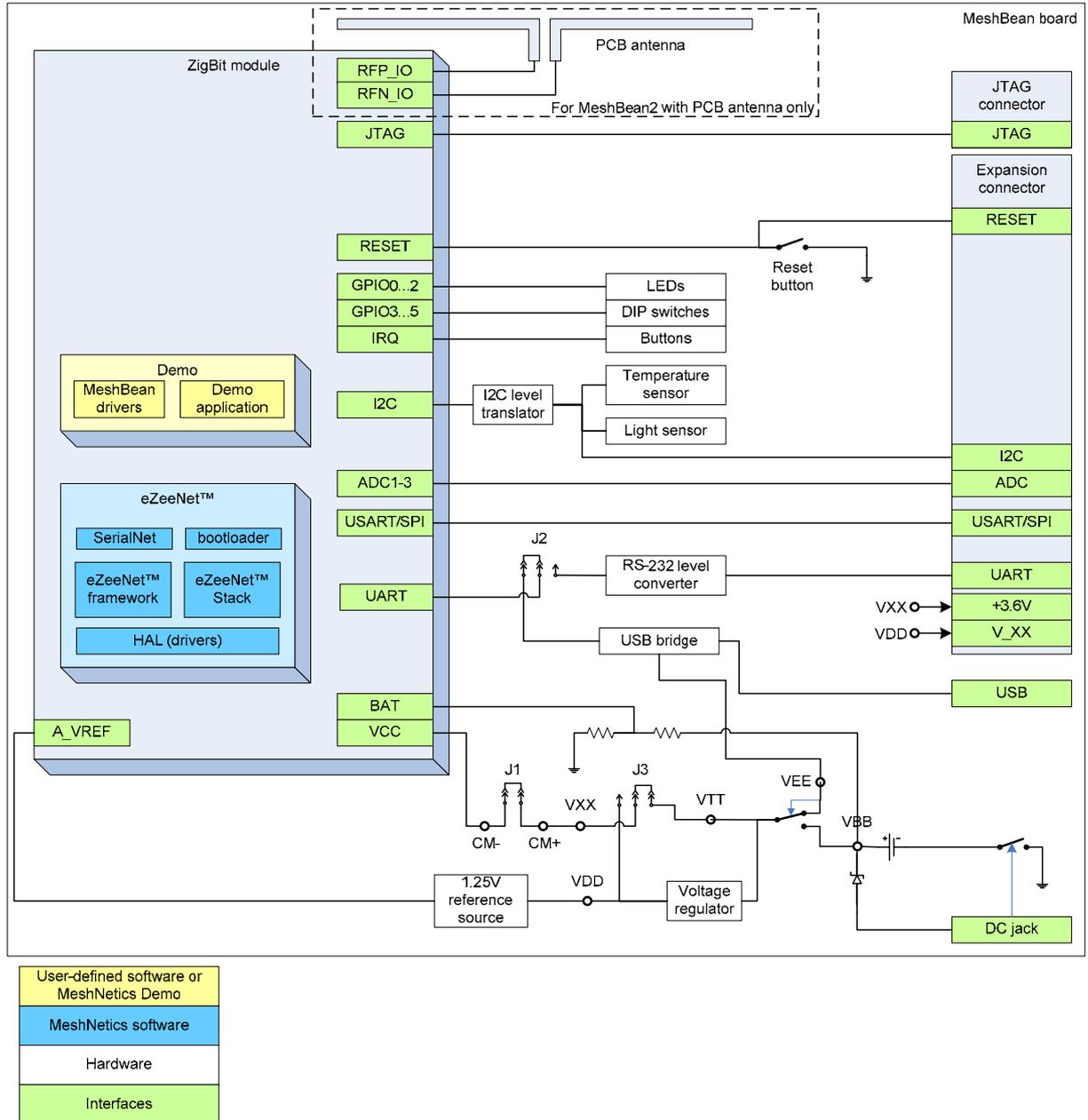
Figure 3. MeshBean2 functional diagram

2.3.1. Connectors and Jumpers

The board connector pinouts and jumper settings are presented in Table 2 through Table 7.

IMPORTANT NOTE:

All manipulations of connectors or jumpers should be done when the board is not powered!

Table 2. Expansion connector pinout

Pin	Name	I/O	Description
1	UART_RTS	Input	Request to Send Pin. Active Low

Pin	Name	I/O	Description
2	UART_TXD	Input	Transmit Data pin (meaning that the host device will transmit the data to this line).
3	UART_CTS	Output	Clear To Send signal from the module. Active low
4	UART_RXD	Output	Receive Data pin (meaning that the host device will receive the data from this line).
5	GND		Digital/analog ground
6	GND		Digital/analog ground
7	I2C_CLK	Input	I^{2}C clock. It is connected to the I2C_CLK pin of the module via low-voltage level translators. For details, refer to ZigBit datasheet [1].
8	I2C_DATA	Bidirectional	I^{2}C data. It is connected to the I2C_DATA pin of the module via low-voltage level translators. For details, refer to ZigBit datasheet [1].
9	+3.6V	Output	Output of internal voltage regulator. Normally, the voltage is 3.6 V.
10	V_XX	Output	ZigBit supply voltage.
11	RESET	Input	Reset pin. Active low. This pin is connected in parallel to the RESET button on the board.
12	USART_TXD	Output	This is Transmit Data pin for USART0 interface of the ZigBit module. It is connected directly to the USART0_TXD pin of the module. For details, refer to ZigBit datasheet [1].
13	USART_RXD	Input	This is Receive Data pin for USART0 interface of the ZigBit module. It is connected directly to the USART0_RXD pin of the module. For details, refer to ZigBit datasheet [1].
14	USART_CLK	Input	This is Clock Data pin for USART0 interface of the ZigBit module. It is connected directly to the USART0_EXTCLK pin of the module. For details, refer to ZigBit datasheet [1].
15	GND		digital/analog ground
16	ADC_INPUT1	Input	ADC input. This pin is connected directly to the ADC_INPUT_1 pin of the module. For details, refer to ZigBit datasheet [1].
17	ADC_INPUT2	Input	ADC input. This pin is connected directly to the ADC_INPUT_2 pin of the module. For details, refer to ZigBit datasheet [1].
18	ADC_INPUT3	Input	ADC input. This pin is connected directly to the ADC_INPUT_3 pin of the module. For details, refer to ZigBit datasheet [1].
19	GND		Digital/analog ground
20	GND		Digital/analog ground

GENERAL NOTES:

Pins 12, 13, 14, 16, 17, 18 are not buffered and driven by the MCU pins directly. Thus this interface should be used with precautions at the low supply voltages to avoid damaging the module.

Pins 7 and 8 are connected via voltage level translators with ESD protection. Thus these pins can be used easily to connect extra I^2C sensors without extra logic.

Voltage on the `V_XX` pin does not depend on the state of jumper `J1` and ammeter connection between clamps `CM+`, `CM-`.

Table 3. JTAG connector pinout

Pin	Name	Description
1	JTAG_TCK	Scan clock
2	JTAG_GND	Digital ground
3	JTAG_TDO	Test data output
4	JTAG_VCC	Controller supply voltage
5	JTAG_TMS	Test mode select
6	JTAG_RST	Reset controller; active low
7	N_Cont	Not connected
8	N_Cont	Not connected
9	JTAG_TDI	Test data input
10	JTAG_GND	Digital ground

NOTE:

JTAG header pinout is compatible with ATmega JTAGICE mkII in-circuit emulator connector.

Table 4. `J1` jumper settings: current measurement

Jumper position	Description
`J1` is connected	This position is used for normal operation.
`J1` is disconnected	In this position, the ZigBit module is not powered while remaining parts of the board are powered. This position is used to measure the current consumption of the ZigBit module (see Section 3.7).

Table 5. `J2` jumper settings: ZigBit power source

Jumper position	Description
J2 connects pins POWER and BAT	ZigBit is powered by primary source (battery, USB or AC/DC adapter)
J2 connects POWER and USB	ZigBit is powered by 3.6 V internal voltage regulator

Table 6. `J3` jumper settings: RS-232/USB selection

Jumper position	Description
`J3` connects central pin and the `RS-232` pin	The board will use RS-232 port (available on the expansion connector) for connection to the host
`J3` connects central pin and the `USB` pin	The board will use USB for connection to the host

> **IMPORTANT NOTES:**
>
> Any other position of jumpers `J2 and J3` or their omission may cause permanent damage of the hardware.
>
> Powering the board without `J1` jumper and ammeter connection between clamps `CM+` and `CM-` may cause a permanent damage of the hardware.

Table 7. RS-232 cable pinout

Signal	Expansion connector	RS-232 connector
RXD	`4: UART_RXD`	2
TXD	`2: UART_TXD`	3
CTS	`3: UART_CTS`	8
RTS	`1: UART_RTS`	7
GND	`5, 6, 15, 19, 20: GND`	5

2.3.2. Buttons, Switches and LEDs

The board includes 2 buttons, 3 DIP switches, one Reset button that generates hardware reset signal, 3 software-defined LEDs (green, yellow and red) and one blue LED indicating powering the board from the USB. Any on-board button, DIP-switch and LED can be controlled with the application running on the ZigBit.

For instance, the status of any DIP-switch will be ignored when running SerialNet (see Section 5). DIP switches can be tested when running the Hardware Test application (see Section 3.6).

2.4. eZeeNet Software

eZeeNet Software from MeshNetics is a robust IEEE802.15.4/ZigBee software that runs on ZigBit modules and organizes a self-healing, self-organizing mesh network. It is specifically tailored for easy-to-use networking in sensing, control, monitoring and data acquisition applications. t provides easy to use networking, with a routing mechanism that optimizes network traffic and reduces power consumption.

eZeeNet Software offers a user-friendly API for network and smart power management, including data exchange, network formation/node join, PAN ID management, channel selection, TX power control etc. The eZeeNet provides a wide range of software interfaces for standard peripherals of supported hardware modules. The eZeeNet comes with the Framework layer which eases application development and simplifies integration.

Another configuration of eZeeNet Software, SerialNet, enables the user to develop customized WSN applications without programming the modules directly or writing any embedded software (see Section 5).

The structure of eZeeNet Software is presented in Figure 4. It is detailed in datasheet [2].
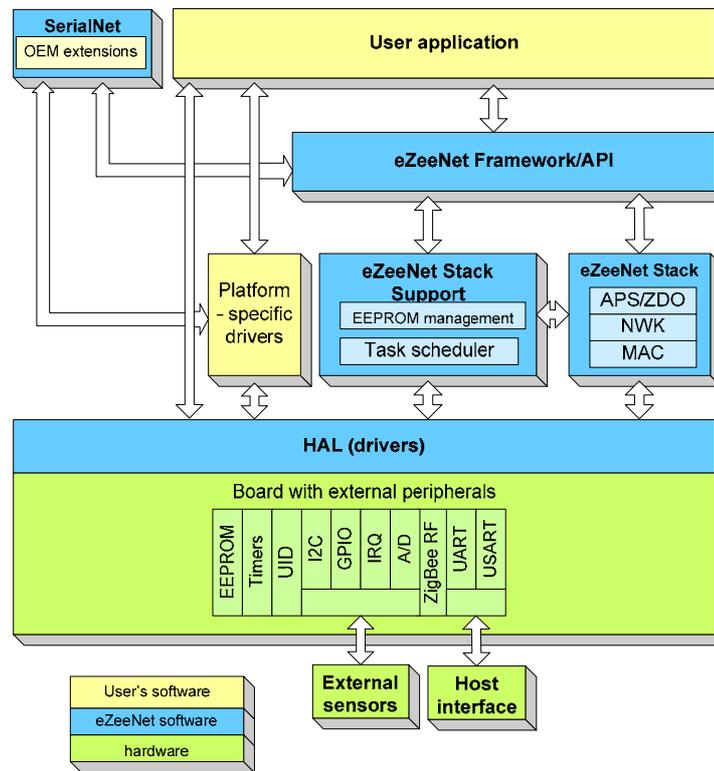

Figure 4. eZeeNet Block Diagram

ZigBit Evaluation Kit contains the following demo applications delivered in binary form:

- Hardware Test
- WSN Demo
- SerialNet application.

The Hardware Test application (see Section 3.6) is a simple application which tests correct operation of major MeshBean2 board components.

The WSN Demo application is a ZEK feature program demonstrating the WSN performance, as presented in details in Section 4.

The SerialNet is intended to control the WSN nodes via serial AT-commands (Section 5). eZeeNet Software lets AT-commands be interpreted locally or forwarded for execution on remote nodes.

The following sample applications are delivered with source code:

- WSN Demo
- Low Power
- Ping-Pong
- Peer-To-Peer
- Blink minimal sample application.

3. Getting Started

3.1. Overview

This section provides step-by-step instructions on how to handle the boards, to make initial testing and reprogram the firmware, to run sample applications and start developing custom applications using eZeeNet API.

3.2. System Requirements

Before employing the Kit, a user should become aware of the minimum system requirements (see Table 8).

Table 8. System requirements

Parameter	Value	Note
PC		
CPU	Intel Pentium III or higher, 800 MHz	
RAM	128 Mbytes	
Hard disk free space	50 MBytes	
JTAG emulator	JTAGICE mkII emulator with cable	Needed to download firmware into the MeshBean2 board through JTAG (see Appendix B). It can be acquired from distributors of this product.
Software		
Operating system	Windows2000/XP	
USB driver	CP210x USB to UART Bridge VCP Driver	Needed to connect MeshBean2 to PC via USB port (see Section 3.4).
IDE	AVR Studio 4.12 + Service Pack + WinAVR	Needed to download firmware image through JTAG (see Appendix B).
Serial Bootloader utility		Needed to download firmware image without using JTAG (see Appendix B).
WSN Monitor installer package		Needed to install the WSN Monitor application (see Section 4.5).
Java machine	Java Runtime Environment JRE 5.0 Update 8 (version 1.5.0)	Needed to run the WSN Monitor application (see Section 4.5).

3.3. PC Software Installation

USB driver fitting different Windows versions can be downloaded from the manufacturer's site:

http://www.silabs.com/tgwWebApp/public/web_content/products/Microcontrollers/USB/en/mcu_vcp.htm.

Install the VCP driver kit from Silicon Laboratories and connect the MeshBean2 board to the USB port. Windows should detect the new hardware, and driver installation wizard will appear. Follow the instructions provided. When the process is completed, make sure that the driver is installed successfully and the new COM port is present in the device list. Invoke the Device Manager window shown in Figure 5, by commanding: (`Start/Settings/Control Panel/System/Hardware/Device Manager`. To resolve possible problems see Section 3.4.
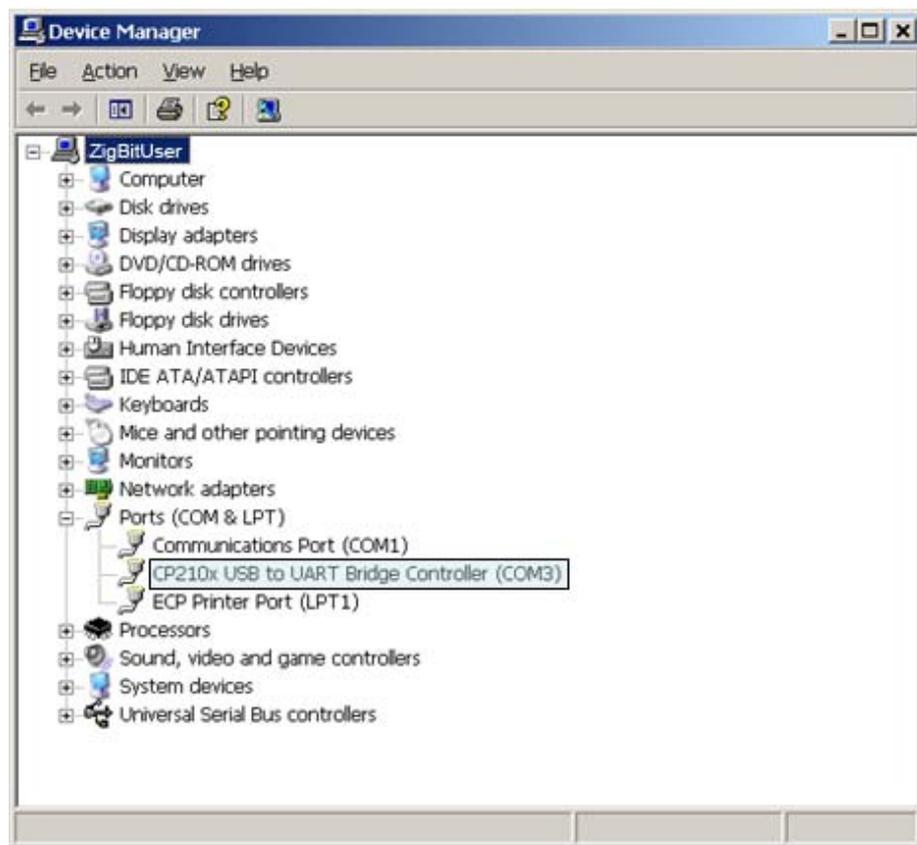

Figure 5. COM port drivers in the Windows Device Manager window

Running WSN Monitor requires Java machine, version not older than 1.5.0 (build 1.5.0_08-b03). Download Java Runtime Environment (JRE) 5.0 Update 8 from http://java.sun.com/javase/downloads/index.jsp and follow the upcoming installation instructions.

After you get Java machine and USB driver on your PC, run the WSN Monitor installer package from the ZEK Distribution CD (see Appendix A) and follow the instructions. There may be other Java instances occasionally installed on your computer. To avoid confusion, edit `start.bat` file in the directory containing the WSN Monitor to provide full path to the Java 1.5.0 executable filespecify its file name extension (`.exe`) explicitly.

> **NOTE:**
>
> Unlike WSN Monitor, the Serial Bootloader utility (see Section 6) does not require any special installation. Just copy the program from the ZEK distribution CD to any directory on your PC.

Current version of the AVR Studio [13] with Service Pack may be freely downloaded from the Atmel's website (http://www.atmel.com). Simply launch the downloaded installer programs and follow set up instructions.

The WinAVR suite of development tools can be downloaded from http://sourceforge.net/projects/winavr. To install WinAVR just follow the setup instructions.

3.4. Connecting the Board to PC

The board can be connected to the PC via the RS-232 port, usuing serial cable, or via USB, using USB cable. The connection mode is controlled by setting of jumper J3 (see Table 6).

> **IMPORTANT NOTE:**
>
> Both ports share the same physical port on the board and cannot be used simultaneously.

You can use serial port to connect the MeshBean2 to both PC and any other unit supporting RS-232 connectivity. RS-232 cable pinout is presented in Table 7.

USB connection is preferable when portable PCs are used. Batteries are not required when the board is powered via USB. Yet another advantage of using the USB connection is possibility to connect several boards to a single PC.

Different applications running on the board may require different settings of the COM port (the particular settings for the specific cases will be described later in this document).

Using the RS-232 port does not require specific drivers for Windows or other operating systems. On the contrary, USB connection requires installation of the driver for a specific USB-COM virtual port (see details in Section 3.3).

> **IMPORTANT NOTES:**
>
> When USB connection is used, the COM-port number could be changed by the Windows operating system if the board is reconnected. Use Windows Control Panel to check the actual port number.
>
> Under some circumstances, the boards can conflict with other USB devices already installed. In such cases, the Windows Device Manager may show a problem occurred during the plug-and-play procedure or it may not detect the USB-UART bridge controller at all. Possible solution is to change the USB ID for the board using special utility available from the USB controller manufacturer. See Section 7 for details.

3.5. Powering the Boards

The boards can be powered by two AA-size batteries, via the USB host port, if connected that way, or via AC/DC adaptor. The nominal voltage is 3 V. Using AC/DC adaptor automatically disables AA batteries. Using USB port disables AC/DC adaptor source.

In order to make accurate measurements of sensor parameters, the battery powering is recommended. USB powering is not stable enough, which can affect transmitting power level and in other RF parameters.

IMPORTANT NOTES:

Although the board includes voltage protection circuitry, using AC/DC adaptors other than those included in the Kit may cause permanent damage of the board.

It is strongly recommended to check the power supply voltage before programming the boards by Serial Bootloader or by JTAG. Power drops, happened during the programming process, can result an inoperable state of the ZigBit or its permanent damage.

Using the discharged batteries (when the voltage is below the specified limit) can also cause damage of flash memory or EEPROM. If that would happen, programming might fail, if using the Serial Bootloader the only option then would be then to use the JTAG emulator (see 0).

Using nickel-cadmium rechargeable batteries is allowed but with some precautions. Their cell potential is nominally 1.2 V. Although a pair gives 2.4 V thus fitting the operating voltage range (see Section 2.1) still it is lower than 3 V which a pair of the most popular alkaline cells give. Hence, nickel-cadmium rechargeable batteries may not be a proper alternative of the alkaline cells for all applications.

3.6. Testing the Board

Begin evaluation of the network features with testing hardware. Initially, all boards are preprogrammed with the Hardware Test Application that allows checking the interfaces installed on the board, excluding RF port. To accomplish the tests, connect the board to the PC and run standard Hyper Terminal utility which is a part of Windows 2000/XP: `Start/All Programs/ Accessories/ Communications/ Hyper Terminal`. COM-port parameters should be set to the values given in Table 9.

Table 9. COM-port settings for hardware testing

Option	Value
Data Rate	38400 bps
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None

When the testing software runs normally, all the board LEDs are blinking. Reports are generated each 5 sec (see Figure 6) including the status of buttons, DIP switches and sensors. To check hardware, you can perform simple manipulations with the board: press the buttons, move the DIP-switches, screen the light sensor with your palm, touch the temperature sensor with your finger and so on. You should see the changes in parameters reported by Hyper Terminal window (see Figure 6).

NOTE:

During the operation, if you reconnect the board to USB or power the board off, the operating system may arbitrarily change the port number for this particular USB connection. Hyper Terminal does not recognize such changes. If this happened, you have to reconnect Hyper Terminal to the proper port. Simply select `File/New Connection` menu item and repeat the same procedure that you were doing at the beginning.

Hardware Test Application is delivered in image files (see Appendix A) and can be downloaded to a board anytime.

Figure 6. Hardware test report

3.7. Measuring Power Consumption

The board allows the ZigBit module's power consumption measurement. To perform the measurements, simply connect ammeter to the clamps CM+ and CM- and remove jumper J1. Make sure that the board is powered by batteries only. However, such measurements would not be absolutely correct, because power is consumed by the interfaces and the peripherals connected to the ZigBit. To make the measurements correctly, all interfaces excluding RF ports should be disconnected from ZigBit module. Refer to the Application note [6] for details.

3.8. Antenna Precautions

Each type of antenna: PCB antenna and dual-chip antenna – ,were matched and tuned, taking into account all the components adjacent to the antenna, including the ZigBit module shield, battery compartment and plastic legs. Any object placed in close proximity to the antenna can affect its performance. Do not put the module into enclosure. Do not mount the board on metal surface. Do not use metal screws longer than 5 mm to fasten the board on legs. These factors may affect the performance.

Mount plastic legs from the bottom side only (where the battery compartment is located). Use plastic screws to fasten the legs. Do not use the legs made of different plastic material. Omitting the plastic legs may significantly affect the antenna performance.

The pattern of antenna is wide. In far field zone, electromagnetic radiation appears stronger in horizontal plane in the direction normal to the dipoles. But the pattern is more complex at the distances of several centimeters. This fact should be taken into account in hardware testing, in WSN evaluation and application development. An approximate field pattern is given in the ZigBit datasheet [1].

4. WSN Demo Application

4.1. Overview

The networking performance of ZigBit platform is demonstrated with the WSN Demo application which is based on the eZeeNet Software API. This application comprises embedded firmware, supporting functions for coordinator, router and end device, and the GUI part – the WSN Monitor which is run on PC.

Thanks to the WSN Demo application embedded, the MeshBean2 boards are organized into a set of nodes constituting a wireless network. In duty circle, end devices and routers update the on-board sensor readings and send them in packets to coordinator. That data is displayed on WSN Monitor panes as temperature, light and battery level measurements.

End device is mostly sleeping, consuming very low power, and it wakes up shortly for activities each 10 seconds. During the sleep period, you can force end device for waking up by pressing the SW1 button.

Router sends data each 1 second. The coordinator transmits the received packets, along with its own sensor data, to the GUI application (WSN Monitor), using UART.

The network size is not limited by WSN Demo. In real time, WSN Monitor visualizes the network topology in a tree form. It also displays the node parameters like addresses, node sensor information and node link quality data.

Measured in dBm, RSSI indicates link's current condition. With the resolution not better than 3 dBm, this is not a very accurate measurement. LQI is a certain numeric value defined within the 0...255 range to measure the link quality. Larger values mean better link, while values close to zero indicate poor condition.

Using WSN Monitor controls you can change the network channel mask, node timeouts and you can reset any node remotely.

NOTES:

The WSN Demo is extended to support both ZigBit and other platform devices [2]. Heterogeneous configurations can be implemented. For instance, a set of AVR RZ502/STK500/STK501 modules, RCBs from Atmel and MeshNetics' Meshbean2 boards can operate as a single WSN. The connected PC with WSN Monitor will visualize the network information.

You can run the WSN Demo without connecting to PC. The LEDs of a board indicate the board current state and activities.

In regard to the WSN Demo, using the boards is described in Section 4.3. GUI is described in Section 4.5. Operation instructions are given in Section 4.6.

The application source code is delivered with the ZigBit Development Kit.

4.2. Programming the Boards

All the boards can be programmed in two ways: either you can use Serial Bootloader utility (see Section 4.2.1) or you can do it under AVR Studio, using JTAG emulator. For instance, JTAGICE mkII from Atmel [14][2] (see Section 4.2.2) is employed.

IMPORTANT NOTE:
Be careful in making choice for programming the node. Each one of MeshBean2 boards come with the bootstrap downloaded into the MCU, which is needed to run Serial Bootloader. If JTAG has been used, it will make impossible using Serial Bootloader, unless bootstrap is reloaded to the board.

To get interconnected in WSN network each node should be identified with a unique MAC address. Once MAC address is not defined by hardware, the node addressing ID should be programmed. Programming a MeshBean2 board with MAC address can be performed in two alternative ways.

MAC address can be downloaded to a board by means of Serial Bootloader running with the key specified within command line (see Section 6).

Otherwise, MAC address can be programmed using SerialNet AT-commands, as described in [3].

4.2.1. Using Serial Bootloader

To program a board using Serial Bootloader do the following steps:

1. Connect the board to the PC via USB or RS-232 port depending on the setting of jumper J3 (see Table 6).
2. Run Serial Bootloader specifying the image file, COM port and the optional keys in command line (see Section 6).
3. Press reset button on the board.
4. Release reset button on the board. Serial Bootloader waits for approximately 30 seconds for the button to be released. If this does not happen, the booting process will stop.

NOTE:
If a node has been configured as end-device and it is currently controlled by the running application, the node should be powered off before reprogramming.

Make sure that J3 setting on the board corresponds to the actual RS-232/USB connection type.

Serial Bootloader indicates the operation progress. Once loading is finished successfully, the board will restart automatically. If loading fails, Serial Bootloader will indicate the reason. In the rare cases, booting process can fail due to the communication errors between the board and the PC. If this happened, try to repeat booting or try to use normal RS-232 port instead of USB. If booting fails, the previous program written to the board can be corrupted, but the board can be reprogrammed again.

---

[2] Another JTAG programmer may be also used but it should be compatible with the Atmel 1281 MCU.

4.2.2. Using JTAG

In programming the board using JTAG, link JTAG emulator together with the MeshBean on-board JTAG connector (see Figure 2). Start downloading process following the instructions from [13] and [14].

Check ON the following options in Fuses Tab before downloading the images through JTAG:

```
Brown-out detection disabled; [BODLEVEL=111]


JTAG Interface Enabled; [JTAGEN=0]


Serial program downloading (SPI) enabled; [SPIEN=0]


Boot Flash section size=1024 words Boot start
address=$FE00;[BOOTSZ=10]


Divide clock by 8 internally; [CKDIV8=0]


Int. RC Osc.; Start-up time: 6 CK + 65 ms; [CKSEL=0010
SUT=01]
```

Uncheck the rest of options. Make sure the following hex value string appears at the bottom of Fuses Tab: 0xFF, 0x9D, 0x62.

Additionally check ON the following option if the nodes will be programmed with Serial Bootloder:

```
Boot Reset vector Enabled (default address=$0000);
[BOOTRST=0]
```

Make sure the following hex value string appears at the bottom of Fuses Tab: 0xFF, 0x9C, 0x62.

By default, each board (MCU) comes programmed with the fuse bit latter option above enabled.

4.3. Using the Boards

At node startup, current channel mask is regularly read from EEPROM. If channel mask was previously loaded to EEPROM using Serial Bootloader, then no special actions described below is needed before starting WSN Demo. Nevertheless, if you need to load channel mask to EEPROM from flash (from an image file) then initializing the node for the first time must be performed as follows.

Press and hold the SW1 onboard button first (see Figure 2). Turn power on the board with holding the button pressed for at least 1 second. The LED1 will get flashing 3 times. Next, LED1, LED2 and LED3 will start blinking for 2 sec to indicate the acceptance of channel mask in EEPROM.

NOTE:

With the above operation completed, the channel mask preloaded to EEPROM will be lost.

Starting the WSN Demo, do the following:

1. Configure one single node as a coordinator, and the others as routers/end-devices (see Table 10)

2. Connect the coordinator node to the PC using USB port on the coordinator board

3. Power on the coordinator node

4. Run WSN Montitor (see Section 4.6.1)

5. Power on and reset the rest of the nodes.

NOTE:

When running WSN Demo, channel mask can be changed anytime later with sending the command from the WSN Monitor (see Section 4.6.4). Issued recently from WSN Monitor and received on a node, channel mask is permanently stored in its EEPROM after getting power on and off. To restore the default channel mask in EEPROM repeat a node reinitializing procedure described above in this section.

Table 10. DIP-switch configurations used in WSN Demo

DIP-switches			Description
1	2	3	
ON	ON	X	Board is configured to be a coordinator.
ON	OFF	X	Board is configured to be a router.
OFF	OFF	X	Board is configured to be an end-device.

Coordinator organizes the wireless network automatically. Upon starting, any node informs the network on its role. At that moment LED1, LED2 and LED3 are flashing once on router, they are flashing twice on end device and they are flashing three times on coordinator.

After joining the network, a node starts sending data to the coordinator which is indicated by LEDs.

WSN activity is observed in two ways:

- controlling the on-board LEDs (see LED indication described in Table 11)
- controlling the network information through the WSN Monitor installed on PC.

Table 11. LED indication implied in WSN Demo

Node State	LED state		
	LED1 (Red)	LED2(Yellow)	LED3(Green)
Network searching	OFF	OFF	blinking
Having joined to network			ON
Message receiving		flashing	ON
Message transmitting	flashing		ON
Changing channel mask	blinking	blinking	blinking
Sleeping (for end device only)	OFF	OFF	OFF

	LED state
Role indication (at startup)	All LEDs are flashing once on router, twice on end device and three times on coordinator

Idle (invalid DIP switch configuration)		ON	
MAC address missed	ON		

If you turn on the power on the coordinator, it will switch to the active state, even if no child nodes are present. This is normal, it means that the coordinator is ready and child nodes can join the network with coordinator's PAN ID.

By default, coordinator uses predefined PAN ID valued as `D162`, which is recognized by all routers. Changing PAN ID is not allowed.

NOTES:

The predefined value of D162 originates from eZeeNet version V. 1.6; it is supposed to change with next versions of eZeeNet.

If coordinator is not present or it is not turned on, the routers will stay in the network search mode. In this mode, routers scan the selected frequency channels continuously to search for a network with the selected PAN ID.

In rare cases, if radio channel is busy on the selected frequency channels the coordinator node will stay in the network searching mode. If this happened, you should change the channel in changing channel mask, using WSN Monitor.

4.4. Sensors Data and Battery Level Indication

Each board measures temperature, light and its own battery level; they send the data values to the coordinator and, further to the PC. The WSN Monitor application displays the values next to a node icon along with presenting them in graphs (see Section 4.5).

The temperature sensor measures ambient temperature. The sensor's actual resolution is actually better than 1 degree, but the sensor data is depicted in the WSN Monitor charts with 1 degree resolution.

Light sensor measures ambient illumination in Lux.

Typical accuracy of the battery voltage indicator is about 0.1 V, which is enough for most applications and self-monitoring tasks.

NOTES:

In case of the board is powered via USB port, the battery level can be shown improperly. Typically, it shows 0.6 V due to the power protection circuitry. However, if batteries are installed into the battery compartment, when the board is connected to the USB the battery level indication will be correct.

In case of the board is powered via USB port, the heating voltage regulator, which is located next to the temperature sensor, can distort the sensor readings. Use battery-powered boards for more accurate measurements.

4.5. WSN Monitor

WSN Monitor is a PC-based GUI application for WSN Demo that serves to display WSN topology and other information about WSN network. See WSN Monitor screen in Figure 7. It contains the Network Topology Pane, Sensor Data Graph Pane, Node Data Table Pane and Toolbars.

In real time, Network Topology Pane displays the network tree. That helps to control the network formation and evolution while the nodes join, send data or leave. The Network Topology pane updates automatically while the network nodes are discovered and joined through coordinator. The nodes which are denoted with their names are presented in icons with the node data tips. The nodes are organized into a tree once interconnected via parent/child links tipped with RSSI and LQI values.

Node Data Pane displays node sensor data (see Section 4.4). It is presented in graphs and in table. There are other parameters for each node which are presented in table. Node Data Pane includes a pull-down Sensor Selection menu to display a particular sensor data.

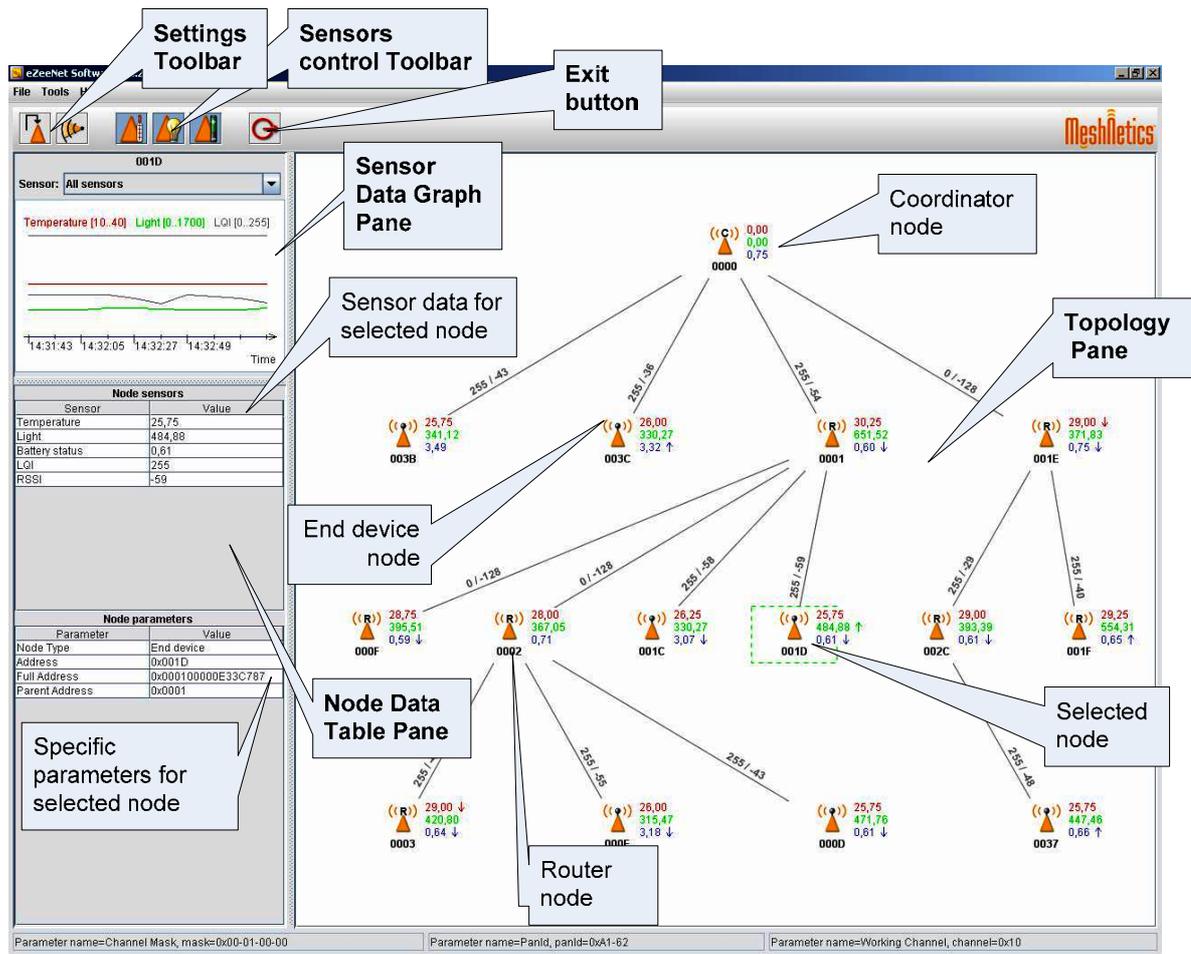Use text menu upside and toolbars controlling visualization.

Figure 7. WSN Monitor GUI

Node names are contained in the node titling file. By default, it is located at "C:\Program Files\Meshnetics\WSN Monitor\resources\configuration\NodeNames.txt". Nevertheless, the root directory of the program can be set up during installation.

This file has the following format. It is headed by the string "`-- NodeNames --`", which is followed by the list in lines each containing 64-bit MAC address and the node name. For example, see Figure 8. If "NodeNames.txt" file is not found or its format is not recognized, the WSN Monitor will designate its own names.
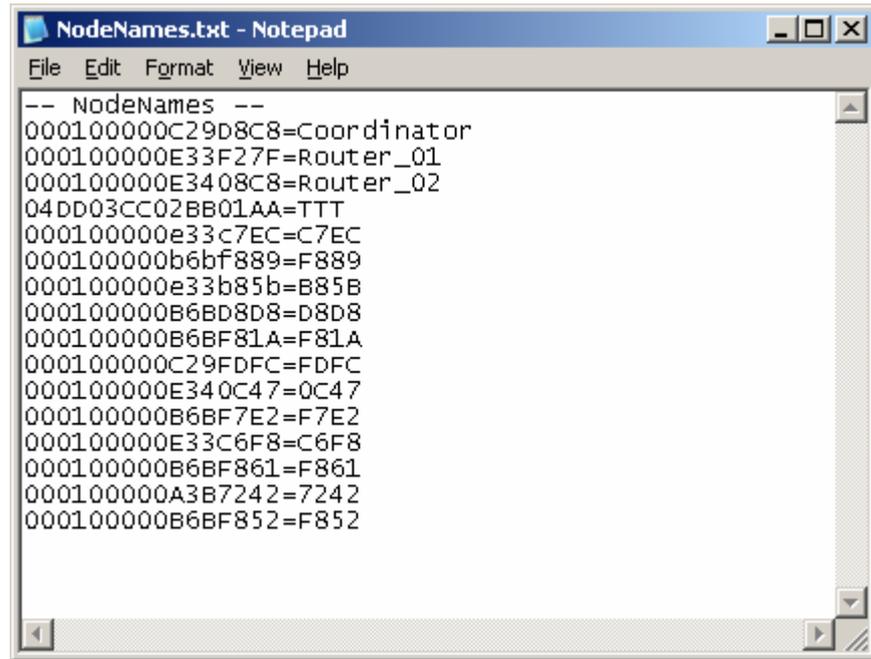
Figure 8. Example of file containing the node titles

4.6. Operating the WSN Demo

4.6.1. Starting WSN Demo on MeshBean2 nodes

First, connect the coordinator node to the USB or RS-232 port, according to the `J3` jumper setting (see Table 6). Then run the WSN Monitor application. At start up, WSN monitor will attempt using the default COM port to connect to the coordinator. The WSN Monitor screen pops up but the coordinator node icon will not yet appear on the Topology Pane yet (see Figure 7). You have to set a proper COM port via `Tools/Settings` menu (see Figure 9). Restart the program if the icon does not appear.

4.6.2. Setting up node timeouts

The `Tools/Settings` menu contains several parameters. The timeouts are used to setup the visualization for coordinator, routers and end devices considering them having disappeared from the network due to link drop, power down, or reset. A node timeout means the awaiting time during which the WSN Monitor is expecting to receive data packet from that node, updating the Topology tree. To get it smooth for changes in topology picture, setting timeouts to 3 sec is recommended for coordinator and router and that to 30 sec for end device. Those timeouts cover 3 periods between packet sending.
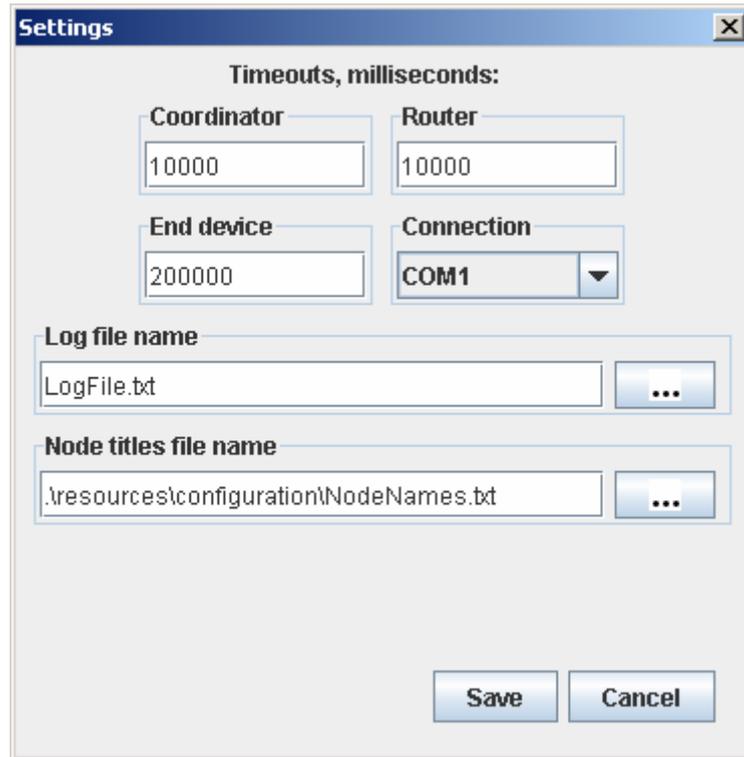
Figure 9. WSN Monitor Tools/Settings menu

4.6.3. Node Reset

A node can be reset by means of the WSN Monitor using the `Tools/Send Command` menu (see Figure 10). Node can be either identified by its MAC address or be selected from the list of the nodes which are currently present in the Topology Pane.
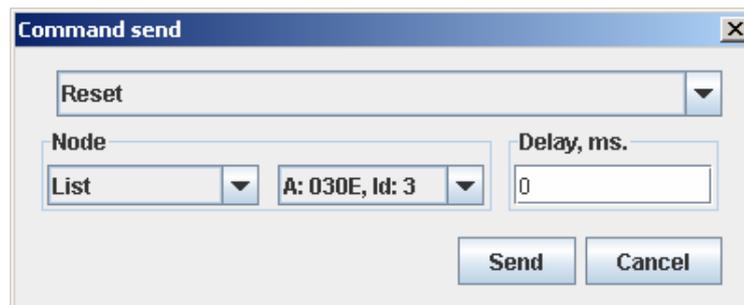

Figure 10. Resetting the node

4.6.4. Changing Frequency Channels

The network operation is supported on 16 upper channels in 2.4 GHz band, with the numbers of `11(0x0B)` through `26(0x1A)`. Use `Tools/Send Command` dialog box to set channel mask. By default, current channel mask is displayed there (see Figure 10). Enter mask directly in `hex` format or click "..." button.

> **NOTE:**
>
> Channel mask is a bitfield which defines the channels available. The 5 most significant bits ($b_{27}, \ldots, b_{31}$) of channel mask should be set to 0. The rest 27 least significant bits ($b_0, b_1, \ldots b_{26}$) indicate availability status for each of the 27 valid channels (1=available, 0=unavailable).
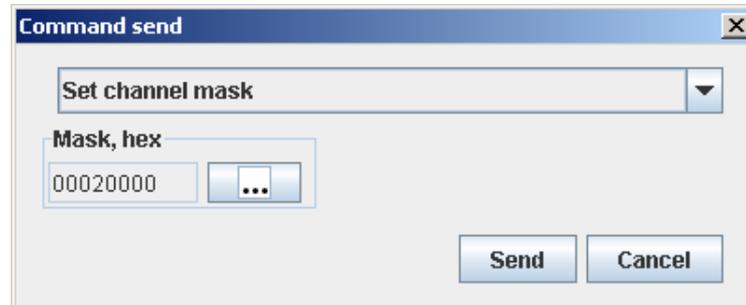

Figure 11. Setting channel mask dialog box

Otherwise you can open another dialog box by clicking the "…" button. Use checkboxes to select the channels thus setting some of them ON (see Figure 12).
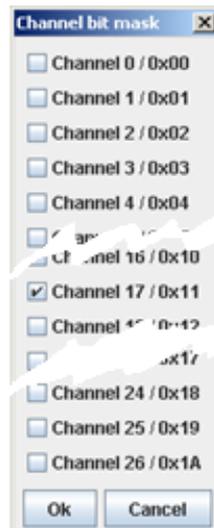

Figure 12. Setting the channel mask using checkboxes

When changing channel mask, coordinator sends the command to all of the nodes and waits for 1 minute more after having received the last packet using old channel mask. Next, coordinator forms the network on the new channel.

When channel mask command is being accepted by router or end device the node stops sending packets for 1 minute and starts blinking the LED1, LED2 and LED3. Next, it leaves the network and proceeds joining using new channel mask.

When router is rejoining, the network indication LED, LED3, is blinking. Upon joining, LED3 is ON.

When end device is rejoining, the network indication LED, LED3, is blinking. Upon joining, LED3 turns ON. LED1 flashes shortly to indicate sending a packet, LED1 flashes shortly to indicate having received acknowledgement. Next, all LEDs turn OFF when end device is falling to sleep.

When channel mask is being changed, the topology screen may not display real topology tree. After changing channel mask, the network topology tree is actualized.

4.6.5. Visualization of the Sensor Data

To help monitoring the network structure and the status of a particular node in real time the network topology is visualized on the Topology Pane (see Figure 7). Visualization helps to control the network formation and evolution while the nodes join, send data or leave. The Network Topology pane updates automatically during the network nodes have been discovered and joined through coordinator.

Having the topology diagram and using the GUI controls, user can select any node to monitor the node activity and see the node data in three different forms:

- Text table
- Chart
- Sensor data values on the Topology Diagram . These have arrows next to them indicating relative increase or decrease.

Topology Pane displaces temperature and light sensor readings as well as battery level for each selected node (marked by dashed line). Also, these data values are shown on the Sensor Data Graph Pane. You can easily check how they change over time.

The Sensor Data Graph Pane includes a pull-down Sensor Selection menu for sensor data to display. Use the button on the Sensor Control Toolbar to display the desired types of sensor data.

5. SerialNet

SerialNet is a configuration of eZeeNet Software which allows control over the most of the ZigBit/eZeeNet functionality through any communication interface using standardized Hayes-like AT-command set.

The commands come from RS-232/USB interface in simple text form, and the language principles are described in ITU-T V.250 recommendation (see [8]).

NOTES:

Strictly, the SerialNet is an application developed "on top" of eZeeNet API.

Before running SerialNet application make sure that the corresponding image file is downloaded to each of the node's board correspondingly (see Appendix A).

See the set of supported AT-commands, syntax and detailed description of each one in the Reference Manual [3]. Chapter "Examples" of that document shows how to use the commands for doing the following:

- to control LED and DIP switches
- to create a network (set the node roles and addresses)
- to transmit data between the nodes
- to manage PAN ID and frequency channels
- to forward commands for remote execution
- to control end-device power consumption.

Thanks to flexibility of AT-commands, you can create your own scenarios of communications that reflect your particular applications and needs. The examples will provide you with a good toolset for your network evaluation.

A variety of terminal programs provide capability to write AT-commands scripts and to analyze the responses from a board. This extends your capability of remote control over MeshBean2 boards. For example, you can simulate the sensor readings that should be sent periodically to the coordinator node, or implement other scenarios.

Before running the application, check the connection between the board and the PC and set in the terminal software (see Table 12) the following parameters of the COM-port.

Table 12. COM-port settings for SerialNet Demo

Option	Value
Data Rate	38400 bps
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	none

To run SerialNet application, download the corresponding image file (see Table 15) into the boards. Follow step-by-step instructions from the `Examples` Section of the document [3]. Use AT-command samples to control your network conditions and to manage data transmission.

NOTE:

The `+IFC` command and the `+IPR` command change the rate and flow control parameters of RS-232/USB port. If any of these commands are used, the COM-port settings on the terminal program running on the PC should be changed appropriately.

6. Serial Bootloader

Serial Bootloader allows downloading an image file into the board using USB/RS-232 connection. Serial Bootloader is controlled by the command-line options. Usage:

```
bootloader –p port_number –f file_name [-b baud_rate] [-h]
[-s bootstrap_size] [-M MAC address] [-C Channel mask]
[P PANID]
```

Table 13. Bootloader Options

Option	Description	Default value
-p port	COM port	
-f file_name	Name of Motorola SREC file	
-b baud_rate	Baud rate in bits per second (1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200)	38400
-h	Hardware flow control, if used	None
-s size	The size of bootstrap code in words (512, 1024, 2048, 4096)	1024
-M MAC	MAC address in HEX format to be assigned to the node	
-C channel mask	Channel mask in HEX format to be assigned to the network	
-P PANID	PANID in HEX format to be assigned to the network	

Example for using Serial Bootloader:

```
bootloader –f wsndemo.srec –p COM5 –M 1 –C 100000 –P 5320
```

The above command demonstrates loading the WSN Demo image into the node connected to PC through COM5, with assigning the following parameters:

MAC address = `0x1`
Channel mask = `0x100000`
PANID = `0x5320`.

Serial Bootloader can be used apart of downloading any image. The following command

```
Bootloader –p COM5 –M 2 –C 100000 –P 5320
```
is intended to assign to a node the parameters below:

MAC address = `0x2`
Channel mask = `0x100000`
PANID = `0x5320`.

Baud rate, flow control modes, the bootstrap code size are set for a node to the default values (see Table 13) if the corresponding options are omitted in command line.

Serial Bootloader recognizes image files in Motorola hexadecimal format, also known as SREC format. Such file names have the `.srec` extension. Motorola SREC files for Serial Bootloader contain both flash memory and EEPROM images. A user's application

developed with AVR Studio can be converted into the SREC format using the `AVR-objcopy` utility so it becomes downloadable via serial booting process.

IMPORTANT NOTES:

Serial Bootloader is designed so that if any of `-M`, `-C` or `-P` options is specified in command line, it overrides the corresponding parameter which was recently stored in EEPROM. Furthermore, the relevant value(s) which was set inside the downloaded image file will be ignored.

Without the `-f` option explicitly used, i.e. without any image file download, applying any other option in command line can be used to change the node EEPROM settings with no interference to the application code downloaded before.

There are some minor limitations to the software downloaded via serial booting process. Serial Bootloader cannot rewrite the upper 2 kBytes of memory starting from the address `0xFC00`, because this part of memory is used by special bootstrap code.

The bootstrap code residing in the MCU is an own part of the eZeeNet software supporting communication with the Serial Bootloader. Normally, it should not get disabled. However, due to the nature of booting protocol, bootstrap code delays the software reset process for approximately 500 msec. If it is not acceptable, the serial bootloading feature can be disabled by reprogramming the corresponding fuse bits [18], using JTAG emulator (see Section 4.2.2).

7. Troubleshooting

In case of any operational problem with your system please check the power first, and make sure that all of your equipment is properly connected.

Check if your PC conforms to the minimum system requirements (see Section 3.2). Check if the PC interfaces (COM, USB) are present and drivers are installed.

Check on LED indication of a node if it is not responding or behaving unusually. Make sure the DIP switches are set according to the application running on the board.

You can retest the particular node as described in Section 3.6, if needed.

You may be required to reset the node.

The following list represents some typical problems that you may encounter while working with the Evaluation Kit and possible solutions.

Table 14. Typical problems and solutions

Problem	Solution
The board does not indicate its activity with LEDs.	Make sure that either WSN Demo image or Hardware Test image is loaded. For SerialNet, the LED status is controlled by AT-commands.
In effort to connect several boards to the same PC their detection fails due to ID recognition conflict.	Detect ID for any single connected board using the `USBView.exe` utility from Silicon Laboratories which can be downloaded from http://www.silabs.com/tgwWebApp/public/web_content/products/Microcontrollers/USB/en/USBXpress.htm. You can use the `CP210xSetIDs.exe` utility from Silicon Laboratories which is included in AN144SW. It is described at http://www.silabs.com/public/documents/tpub_doc/anote/Microcontrollers/Interface/en/an144.pdf and can be downloaded from http://www.silabs.com/public/documents/software_doc/othersoftware/Microcontrollers/Interface/en/an144sw.zip.
WSN Monitor fails to start.	Locate the executable file for Java installed in your system. Next, locate the directory containing the WSN Monitor application installed. Edit the `start.bat` command file there. Specify full path to the located Java executable file in the default command line and its file name extension. Then save the command file edited. With these instructions performed, WSN Monitor should start correctly. See sample command line below: `C:\jre1.5\ bin\java.exe -classpath .;lib\junit.jar;lib\log4j-1.3.jar;lib\jgraph.jar;lib\comm.jar -Xmx200m com.meshnetics.controller.MainClass`
No node is shown on the Topology Pane in the WSN Monitor	Check if the WSN Monitor uses proper COM port and if not, change it and restart the program.
WSN Monitor shows NO DATA in the Sensor Data Graph Pane.	No node is selected. Select the required node by mouse-clicking on it.

Problem	Solution
Node titles displayed on the Topology Pane do not show node destinations.	The displayed titles do not necessarily relate to the node functions but they can be redefined by user anytime. These names are stored in the node title file (see Section 4.5) along with MAC addresses mapped to the nodes.
At WSN Monitor startup, all node's LEDs are blinking, or none of them is flashing.	The WSN Demo application was not downloaded into the node. Load this application to the node.
Neither Serial Bootloader nor another application work with a node, except for the Hardware Test.	Make sure that J3 setting on the board corresponds to the actual RS-232/USB connection type. Make sure the microcontroller flash memory was not erased before, and the bootstrap was not lost there after having the node programmed through JTAG.

Appendices

Appendix A.   Distribution CD File Structure

Table 15. The CD contents

Directory/File	Description
Readme.html	Installation instructions and quick start guide
./Documentation	Documentation
./Information	Datasheets and application notes
./Bootloader/Bootloader.exe	Serial Bootloader executable file
./Test/HardwareTest.srec ./Test/HardwareTest.hex	Hardware Test image files
./Demo/WSN/WSNDemo.srec ./Demo/WSN/WSNDemo.hex	WSN Demo image files
./Demo/WSN/WSNMonitorSetup.exe	WSN Monitor Installer
./Demo/SerialNet/SerialNet.srec ./Demo/SerialNet/SerialNet.hex	SerialNet image files

Appendix B.   Using JTAG emulator

Programming with JTAG gives more flexibility in managing the loading process, but requires special hardware. For Windows environment we recommend using the AVR Studio 4.12 + Service Pack. AVaRICE 2.40 may be used for Linux. In both cases, the recommended JTAG emulator is JTAGICE mkII from Atmel. Other programming devices can be utilized as well, but make sure that the particular model supports programming an Atmega1281 MCU before use.

Motorola HEX files for Serial Bootloader contain both flash memory and EEPROM images. AVR Studio does not support downloads of such combined files into the MCU. Instead, it requires separate images for flash memory and EEPROM, and it recognizes files in the Intel HEX format only. This is the reason why we also deliver 2 separate image files in the Intel HEX format that are downloadable by AVR Studio or other JTAG tools. Both files have the same names but different extensions. EEPROM image has `.eep` extension while flash image has `.hex` extension. To download the firmware, follow the instructions from the device manufacturer's manuals [13], [14], [15]. Sample pop-up window is shown on Figure 13.
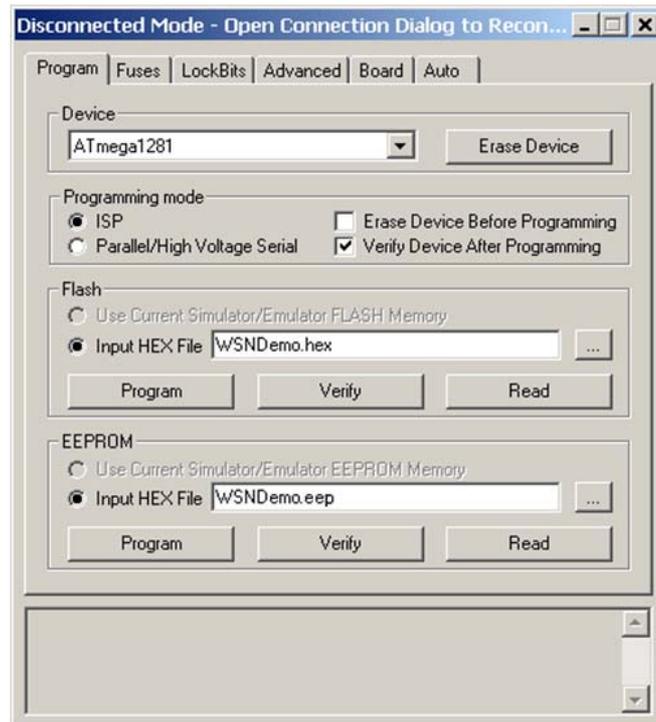
Figure 13. AVR Studio dialog box for JTAG firmware downloading

The well-known command line utility `avrdude`, which is a part of WinAVR environment (http://sourceforge.net/projects/winavr) can be used for download as well. This utility recognizes both Intel HEX and Motorola HEX formats.

IMPORTANT NOTES:

To avoid accidental corruption of the bootstrap code that supports serial booting and provides normal operation of the eZeeNet Software, it is recommended to avoid erasing device and changing any fuse bit or any lock bit.

For JTAG programming, the Boot Reset vector should be disabled. To enable serial booting this fuse bit should be enabled. Do not change the other fuse bits.