Study text

Arduino WSN

Lukáš Němec

Contents

1 Introduction

Purpose of this text is to provide general introduction to Arduino and applications which can be made using Arduino platform. Focus of second and third part will be at wireless communication and making wireless sensor networks with Arduino based nodes. All used source codes can be found at Edu-Hoc project home, `https://github.com/crocs-muni/Edu-hoc`; at official Arduino website, `www.arduino.cc`; or at JeeLib library website, `http://jeelabs.net/projects/jeelib/wiki`.

2 Arduino - general view

Arduino is an open-source platform and also phenomenon of last few years. It offers hardware itself and also software for programming ATMega micro controllers. Thre are not only official Arduino boards, but also large number of clones and derivatives. These are motivated either by cheaper price or by added functionality compared to official ones. First category typically includes Chinese clones like Funduino or others, while second category consists usually of specialized boards like JeeLink and JeeNode USB, which we are going to use.

From all containing Arduino platform we will use only development environment, because our focus will be on ad-hoc networks, we will use specialized boards with build-in radio module, as was mentioned earlier in the text. Nevertheless we will start with absolute basics of working with Arduino.

2.1 Arduino IDE

Arduino IDE is very simple environment for development programs, from practical point of view, it is just a little bit tweaked text editor with added support for Arduino. It supports all major OS, so you can run it on Linux, Windows or OS X. It can run without installation, but this option does not support communication with boards over serial port. So this option is useful for programing, but if you want to sent program to board itself or communicate with it, then it is highly recommended to install Arduino IDE on your computer.

You can benefit from this environment also if you decide to use your favorite text editor for actual coding, then Arduino IDE could be used just for compiling the code or because of other tools it offers.

Installation Installation is not difficult at all. Many Linux distributions will ease the process, because Arduino IDE package is usually present in the system repositories. Only thing you should care about is the version of such package, ideally it should be 1.5 or higher.

In case of Windows OS, recommended approach is to download installation files directly from official webpage `http://arduino.cc/en/Main/Software` where you will find

executable files, which will either install Arduino IDE on your machine, or just run Arduino IDE itself (in case you do not have access to administrator account on machine you are working).

If for whatever reason you need to install Arduino IDE some other way (e.g. directly from source code), then all the information you should need is again accessible on official webpage `http://arduino.cc/en/Main/Software`.

Tools

Verification One of the most important tool which environment offers is verifier which sole purpose is to check for syntax errors of your code. This option is unfortunately not real time, so it is more than recommended to check your syntax once a while.

Compilation Second tool is compiler, which also takes care of upload to the board. Fist you are required to select your version of Arduino board, in some cases board which is the most similar to one you have (especially important if you have clones or various derivates). Next you should check port setting and set the port which is the board connected to. In case of only one board connected, IDE usually selects the right port autonomously, but it is recommended to check it. It is also possible to connect multiple board at once and then you need to select the right one.

In case of issues with detection of board it is recommended to restart whole IDE and try again. Also it is good practice to connect the board before you start the IDE.

Serial monitor Serial monitor is last of the important tools which are useful to know. Although its functionality can be substituted bu any program, which allows communication over serial port (e.g. PySerial), on the other hand Serial Monitor is the most accessible tool you have.

After start you are required to set up communication frequency (same value as in source code) and then you can read information send from the board or send instructions to the board.

If you can see characters, which make no sense, it is more than likely, that it is an issue with different setup of frequency on board and Serial Monitor.

2.2 Arduino Code basics

Arduino can be seen as special case of C language. Almost everything is the same, only thing changed is `main` function, which is divided into two parts `setup` and `loop`. Therefore you are able to write cycles same as in C language, same for conditions, functions and everything else you might ever need. In addition to this, Arduino adds it's own functions and large amount of libraries, which are usually targeted to accessory hardware. You can also find many other libraries which make writing the code in C less painful.

`setup` As name suggests, this function is mainly for setup. It is run after the board is powered on and only once. It takes care of initialization of everything you might need. Thus it is ideal for setup of serial port communication, pin setup and initializatio of libraries.

`loop` Function which contains the main part of program, and this function can be seen as an infinite cycle. It is called after `setup` function, so we can count on all being initialized. This function will loop until the board is disconnected from power or restarted. It is good practice to write code in this function with respect to infinite loop and also if you need regular checking of some event (button pressed) it is recommended to either have code so fast, that checking once a cycle is short enough time frame or run check several times in one run of cycle.[1]

Serial One of the most important libraries you will ever encounter. It manages serial port on the side of board. First you have to initialize it in `setup` function, together with frequency setting. Then you can start using it, even immediately after initialization in `setup` function. All functions from this library has to be called as `Serial.read()` not just `read()`.

`print` Basic function for sending data, debugging values and text over serial port. It is possible to print value itself of format as it is more convenient (e.g. hexadecimal values). Also it is possible to use `println`, which in addition sends new line character. When using this function you have to remember, that this function is asynchronous, therefore it sends a return value before it starts to send characters over serial port.

You can also use `write` (for writing binary data) or `flush` (waits for completion of writing).

`read` Reads data from serial port and returns read value. It is convenient to use together with `available` function, that returns number of available bytes on serial port. You can use this function for sending commands to the board or initial setup, which cannot be hard coded.

3 WSN node, JeeLib basics

3.1 JeeNode HW

General specs

Radio

[1] some boards have dedicated pins for interrupts, which can step into `loop` function and provide immediate event handling. You can connect special function to the interrupt pin, which takes care of interrupt service

Pins

3.2 JeeLib basics

How to add to IDE

Header format

Send unicast msg

Receive msg

2 motes network

4 WSN network

4.1 Simple apps

Alive

Sniffer

4.2 Network

General options

Fixed routing