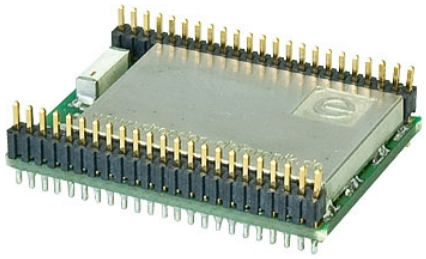
AVR Eclipse Tutorial

For the SES lab it is recommend to use the Eclipse IDE and install the plugins needed to program the microcontroller and use an SVN repository. This setup and how to use it are described in this document. Students who prefer another work environment may use it but should not expect any support for setting up the toolchain, the build system and so on.



Installation Prerequisites

- A PC with Ubuntu (or some other Linux), Windows or Mac OS X for which you own administration rights

Eclipse

(You can skip this section, if you have already installed Eclipse with CDT plugin.)

From the Eclipse homepage (http://www.eclipse.org/downloads/) download *Eclipse IDE for C/C++ Developers* for your operating system. Note that Eclipse requires an installed Java JRE. Afterwards you can extract (unzip) the downloaded file into a folder of your choice. Eclipse can be started by calling the file *eclipse*, which can be found in the main directory of the program.

AVR Toolchain

For working with AVR microcontrollers the corresponding toolchain has to be installed, which contains the GCC compiler, libraries, software for flashing the device, etc.

- Windows users download and install the Atmel avr-gcc toolchain and avrdude individually.
  - First, go to http://www.atmel.com/tools/atmelavrtoolchainforwindows.aspx and download the Atmel AVR 8-Bit Toolchain Windows and execute the installer (you will have to register with your E-Mail to get a copy)
  - Avrdude can be downloaded as executable via http://download.savannah.gnu.org/releases/avrdude/. Select avrdude-6.1-mingw32.zip and copy or move the executable avrdude.exe to your toolchain's bin directory
  - Make sure, that your toolchain's bin directory is listed in your PATH variable
  - Furthermore, MinGW is required if it is not yet installed on your computer. Open http://sourceforge.net/projects/mingw/files/. Download and execute the file *mingw-get-setup.exe*. Follow the installation, use default settings, don't use an installation path with whitespaces! In the MinGW GUI select *msys-base* and *mingw32-base* for installation. It is necessary to add the folder of MinGW's bin directory to the *Path* system variable of Windows. Go to *System Properties->Advanced->Environment Variables...*. Add the bin folders of MinGW and MSYS to the PATH system variable.
- Ubuntu users have to install the packages *avrdude*, *gcc-avr*, and *avr-libc*;
- Other Linux Distributions: If your Linux distribution does not have any avr-gcc packages in its package sources, you can get a precompiled version at

http://www.atmel.com/tools/atmelavrtoolchainforlinux.aspx; you also need to get avrdude and build it from source (http://download.savannah.gnu.org/releases/avrdude/)

- For Mac OS X, please download *CrossPack* http://www.obdev.at/products/crosspack/index.html and install the program.

Programmer & Virtual COM Port Drivers (Windows only!)

In the SES lab we are using the *Olimex ISP programmer* for flashing the device. The required USB drivers can be downloaded from https://www.olimex.com/Products/. Navigate to *AVR* and then to *AVR-ISP500* to find the newest driver.

Later in the course you will need to create a data connection between the SES-board and the PC via a special USB-FTDI cable. However, if you connect the USB-FTDI cable for the first time, the corresponding driver might be missing. In this case simply obtain the newest version of the FTDI drivers from http://www.ftdichip.com/Drivers/VCP.htm.

Programmer & Virtual COM Port Access Rights (Linux only)

In order to get the right to write to the hardware, you need to put yourself into the tty and dialout user groups by executing the following commands.

```
sudo usermod -a -G tty yourUserName
sudo usermod -a -G dialout yourUserName
```

Log off and log on again for the changes to take effect!

AVR Eclipse Plugin

Eclipse is highly extendable. A dedicated plugin exists for AVR programming, which eases the usage of the AVR toolchain.

Please start Eclipse. Note that each time you start Eclipse, you will be prompted to select a workspace directory. A workspace contains different projects. Select a directory, in which you want to store all source code files for the SES lab.

To install the Eclipse AVR Plugin, select *Help→Install New Software...*. Click on *Add* and insert the site http://avr-eclipse.sourceforge.net/updatesite/ and the name *AVR* into the form. After pressing OK you have to wait a few seconds until Eclipse loads the information concerning the plugin. Put a check mark on *AVR Eclipse Plugin* and push the button *Next*. Now follow the instructions (ignore warnings) until the plugin is installed. At the end Eclipse should be restarted.

SVN integration with Eclipse

To use SVN from within Eclipse, you first have to install the subversive plugin and choose an SVN connector. From the *Help→Install New Software...*, choose the default eclipse update site (e.g., Luna), select *Collaboration→Suberversive SVN Team Provider* and install it. When the installation process is finished, restart Eclipse and on restart, choose a SVN Connector (e.g., SVNKit) from the appearing list.

Working with projects

Creating a project

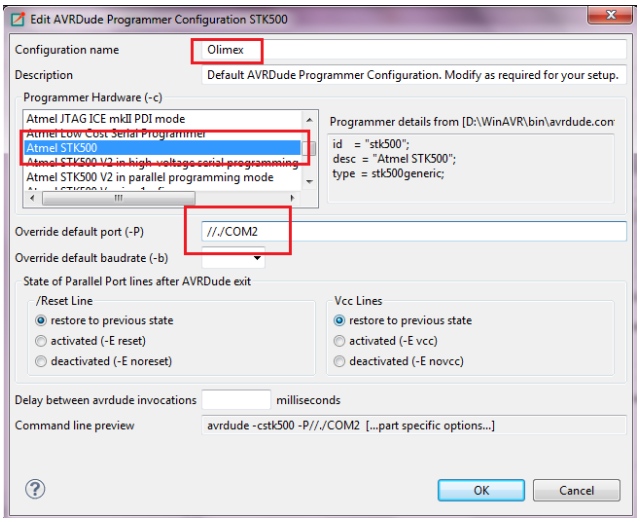
After starting Eclipse you can select *File→New→Project...*. For the item *C/C++* mark *C Project*. After pressing *Next* you have to enter the name of the project (e.g. *Blinker*). Use as project type *AVR Cross Target Application*, *EmptyProject*. Then click on *Finish*. You will be asked for the MCU type (ATmega128RFA1) and the CPU frequency (16 MHz). Now, the empty project is visible in the left window of Eclipse. Note that depending on the project type the appearance (perspective) of Eclipse changes. In the example, the C-Perspective is activated.

Configuration of the project

You have to adjust some project settings. For this purpose you should click with the right mouse button on the project folder in the left window and select *Properties*. First select *AVR→AVR-DUDE*. To be able to flash an image to the microcontroller device, *AVRDude* has to be configured properly. Currently no configuration for the programmer exist, so push the button *New*.

Step 1 AVR

Please insert *Olimex* as configuration name and select from the list *STK500*. Finally, you have to insert into the field *Override Default Port* the correct USB port identifier on which the programmer is plugged in (e.g. */dev/ttyACM0* for Ubuntu/Linux, *//./COM2* for Windows or /dev/cu.usbmodem411 for MAC OS X). The proper filled form is shown in the right figure. After pressing *OK* you can select the newly created programmer configuration.



Now, select *AVR→Target Hardware* and set *ATMega128RFA1* as MCU type (if the board is connected you can also call *Load from MCU*). The clock frequency should be set to 16 MHz.

step 2 build setting

Select *C/C++ Build→Settings* and mark the check field *Generate Hex file for Flash memory*. Press *OK* to quit the configuration windows.

Only for Windows: Select *C/C++ Build* and unmark the check field *Use default build command* and type in *mingw32-make*.

step 3 make

Finally, click with the right mouse button on the project folder in the left window and select *Index→Rebuild*.

Putting the project under SVN control / Importing an existing project

The graded exercises as well as the ses driver library which you are going to create in course of this semester have to be submitted using our SVN server. For general information about version control with SVN, check the web. Using Eclipse with Subversive, to put a project under version control, right click its name and select *Team→Share Project...*. First time you are sharing a project, you have to create a new repository location:

https://svn.ti5.tu-harburg.de/courses/ses/2016/teamX (where X is you team number)

Select/Create the correct repository location and click next. In the next wizard, choose Simple Mode and enter the desired or required name for the folder (e.g., we require you ses driver library to reside in folder ses) and click finish. Select the files you want to commit (please **do not check in binary files or libraries, but just the source and header files and the project files**) and click commit.

Team members now can easily import the shared project using *File→Import...*, select *Projects from SVN*, create/choose the repository location, click *Next*, specify the folder where the project is located and then *Finish*. In the next wizard, check the project name and settings and again click *Finish*.

Keeping your repository clean

As mentioned, you should only commit source and header files, not the build object, executable and library files. Unfortunately, the Subversive Eclipse plugin is a bit aggressive, i.e., on committing it always tries to add all files and folders which are not yet under version control to the repository. To prevent this (and to avoid the need of always checking what to commit and what not) you can put files or folders on the SVN ignore list. Mark the files and folder you want to ignore, right-click and select *Team→Add to svn:ignore* and then select *Resources by name*. This way, Subversive won't try to commit those files/folders to the SVN in the future.

Writing code

Now you can start writing code. A new source code file is created by clicking with the right mouse button on the project name. Afterwards select the entry *New→Source File*. Insert *main.c* for the source file name and press *Finish*. A small example program is shown below, which you can adopt.

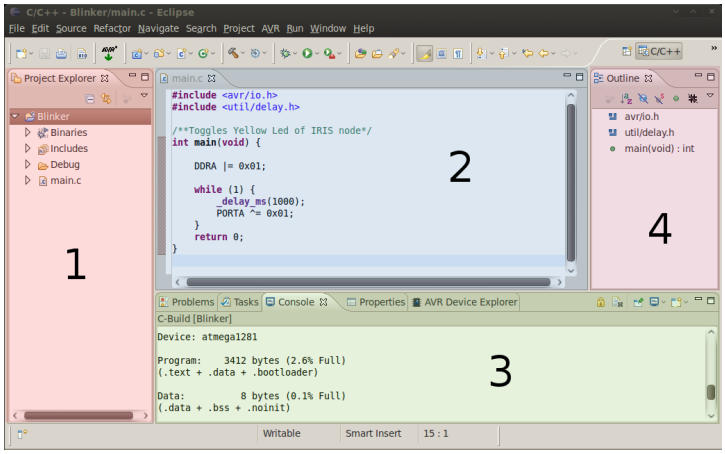
```
#include <avr/io.h>
#include <util/delay.h>

/**Toggles the red LED of the SES-board*/
int main(void) {
    DDRG |= 0x02;

    while (1) {
        _delay_ms(1000);
        PORTG ^= 0x02;
    }
    return 0;
}
```

Compiling and Flashing

By saving the file (Ctrl + s) and afterwards pressing (Ctrl + b) the source code is compiled. Via the menu item *AVR→Upload Project to Target Device* or the corresponding button (AVR lettering with green arrow), you can flash the device. Be aware of marking the project name in the left window by clicking with the left mouse button on it, otherwise flashing will fail.

The C-Perspective

The individual windows of Eclipse are referred as views. All visible views are part of a perspective. In the following the C-Perspective is introduced. For the example above, the figure on the right shows one possible appearance of Eclipse.



Area 1 Provides an overview of the projects existent in the current workspace. You can navigate between the projects and manage files.

Area 2 In this area, the editor is found. You can easily switch between open files by using the tab bar on the top. Errors or warnings are displayed directly in the source code.

Area 3 The view *Problems* lists the current errors in the source code. The *Console* shows helpful information about the compilation, e.g. required memory (FLASH, RAM) of the project.

Area 4 The *Outline* displays information about the currently opened file. This view provides a good overview of the structure (functions and variables) of the file.

Important shortcuts

Ctrl+Shift+l	lists all available shortcuts
Ctrl+s	saves current file
Ctrl+b	compiles all opened projects
Ctrl+Space	opens the content assistant for current input
Ctrl+left mouse button on identifier (e.g. function name)	goes to the definition of the identifier
Ctrl+z	undoes the last operation
Ctrl+y	redoes the last undo operation
Ctrl+f	searches for strings in the current file
Ctrl+Alt+u	flashes the microcontroller