

Steps for running C-program for AVR microcontroller on Linux

Step:1 Installation of required packages

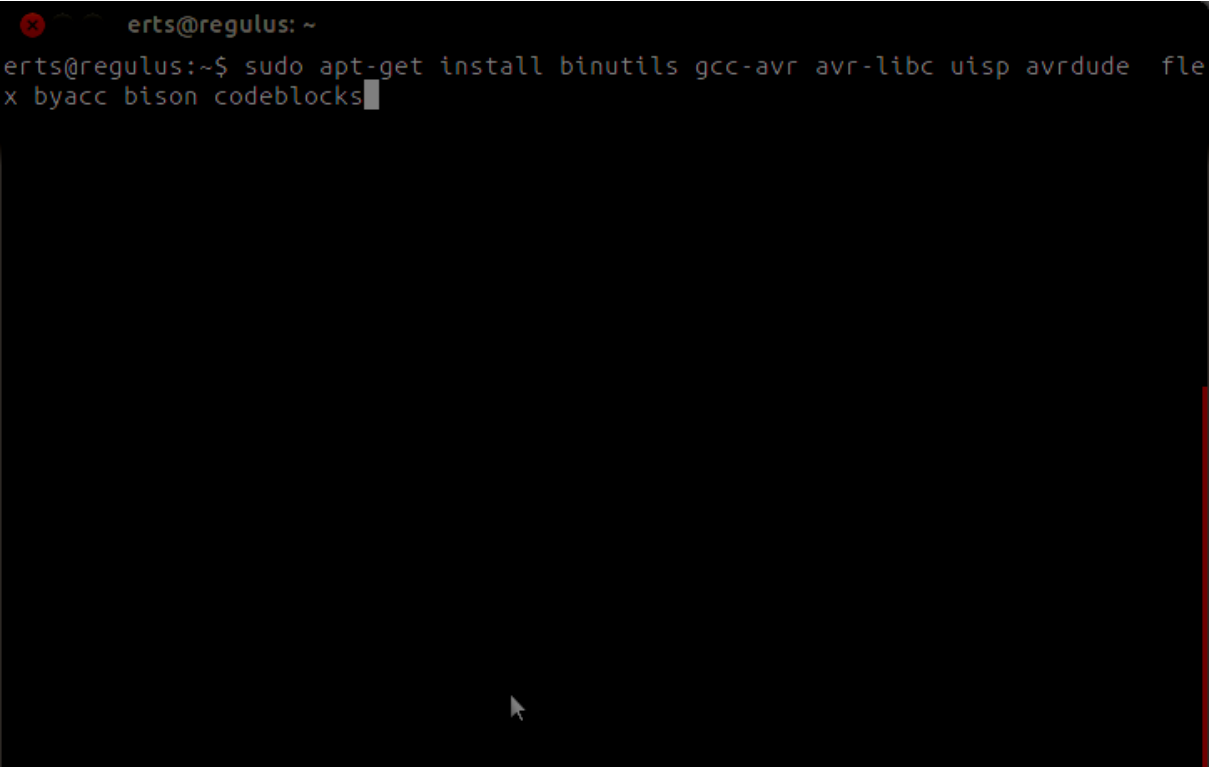
We will first install all the required packages. This includes

1. binutils - Tools like the assembler, linker.
2. gcc-avr - The GNU C cross compiler for AVR microcontrollers.
3. avr-libc - Package for the AVR C library, containing many utility functions.
4. uisp - Tool for in system programming of AVR microcontrollers.
5. avrdude - Utility to program AVR microcontroller. It supports the STK500v2 programmer
6. flex - Lexical analyser.
7. bison - Parser generator for C.
8. byacc - Another enhanced Parser generator for C. Modified version of Bison parser.
9. codeblocks - Integrated Development Environment(IDE) for writing C-program and compiling the program to generate the .hex file which can be loaded on microcontroller.

These packages will be installed through terminal. Open linux terminal.

Figure-1 below shows the command which need to be entered in terminal to install the packages.

```
sudo apt-get install binutils gcc-avr avr-libc uisp avrdude flex byacc bison codeblocks
```



The screenshot shows a terminal window with a black background and white text. The prompt is 'erts@regulus: ~'. The command entered is 'sudo apt-get install binutils gcc-avr avr-libc uisp avrdude flex byacc bison codeblocks'. The cursor is at the end of the command. The terminal window has a title bar with a red close button and the text 'erts@regulus: ~'.

Figure-1 Package installation through linux terminal

Note:1.Package installation will require internet connectivity.

2. In majority of ubuntu distribution Terminal can be launched by using shortcut key ctrl+alt+t

Step:2 Creating Project with Codeblocks

Codeblocks is an IDE which is used for writing C-program for AVR and compile it to generate the .hex file.

Step:2.1

Start of Codeblocks looks similar to one shown in figure-2 shown below. Click on Create a new project to start.

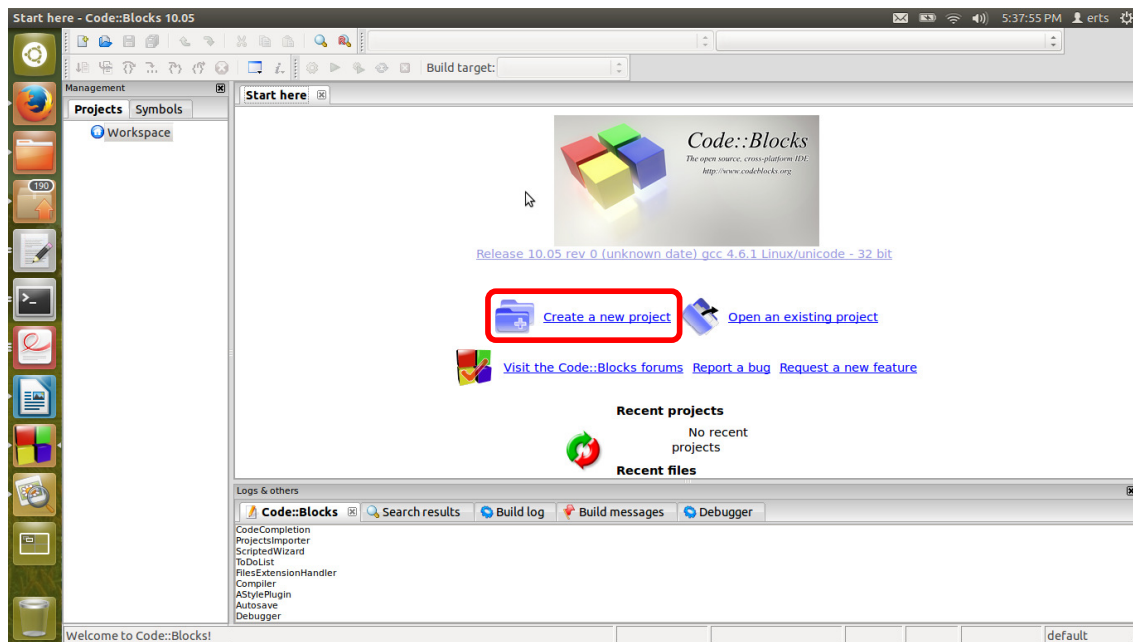


Figure-2: Codeblocks Welcome screen

Step:2.2

Select AVR Project from Project template category and click Go. Illustrated in figure-3 below.

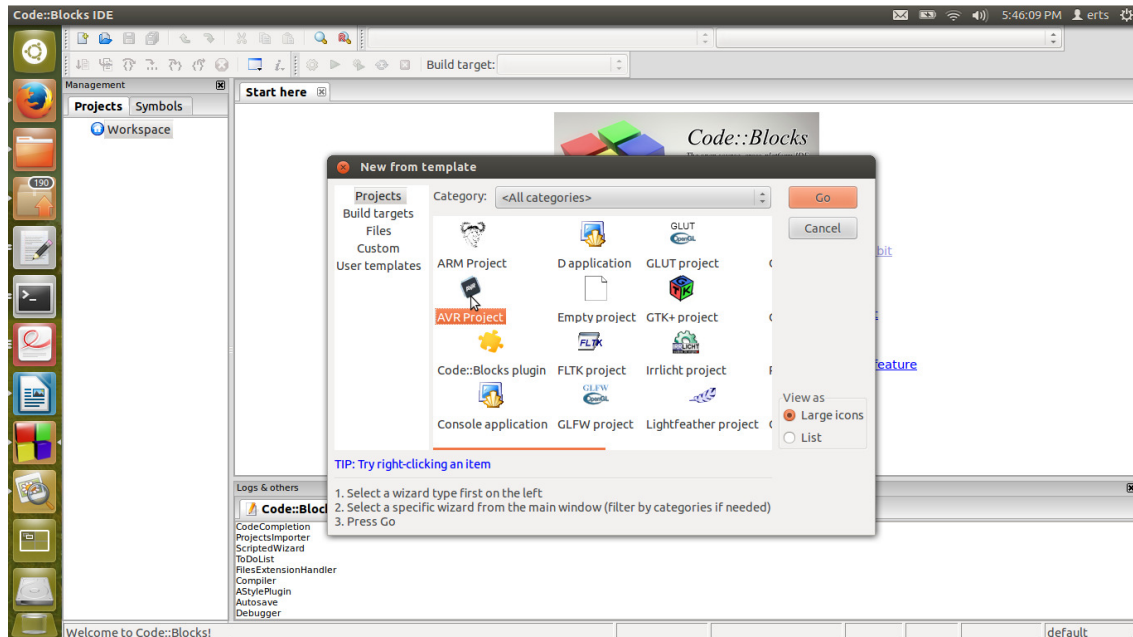


Figure-3: Selecting AVR project from project template

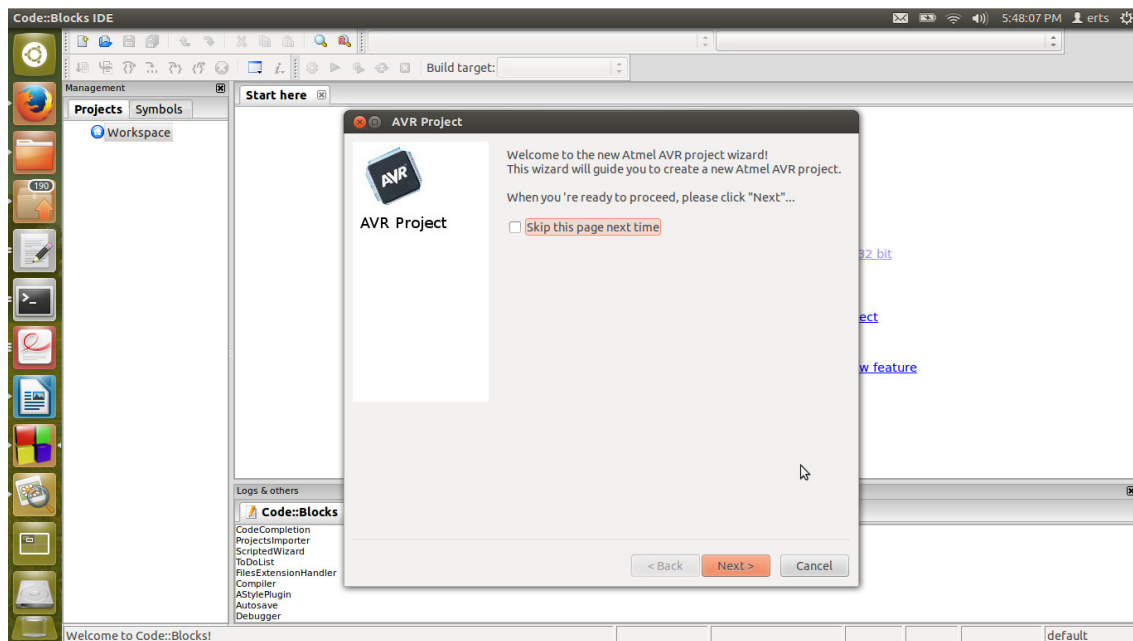


Figure-4 Welcome Screen

Step:2.3

Enter Project title and project folder name, as illustrated in figure-5. As shown in figure-6, leave other settings as default. Make sure that compiler selected is **GNU AVR GCC compiler**. Click on next button.

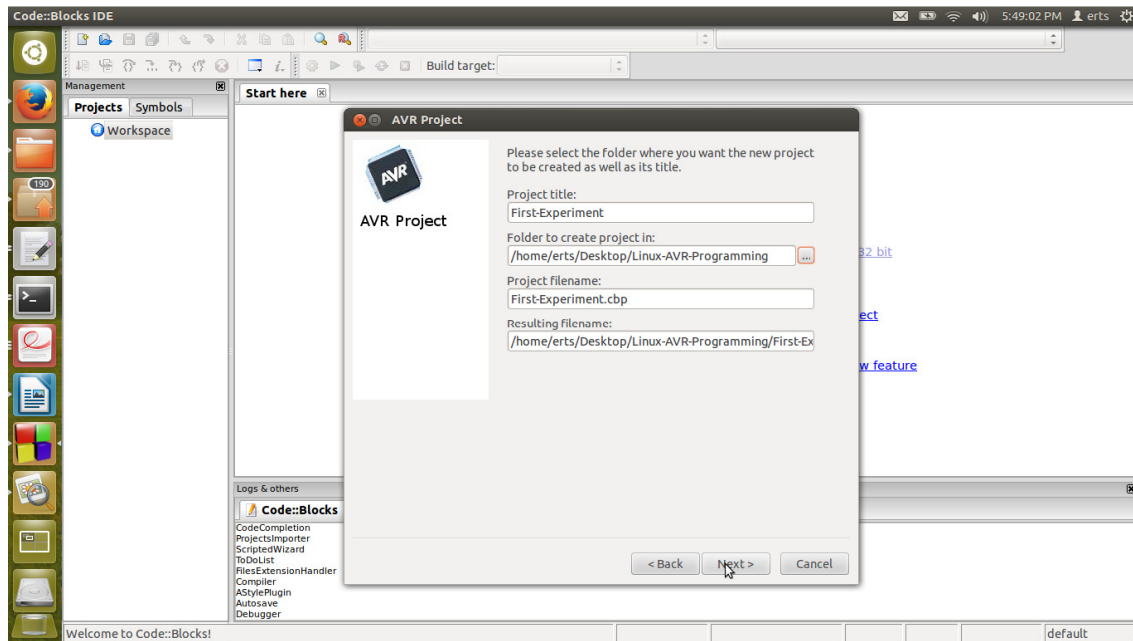


Figure-5: Project name and folder selection window

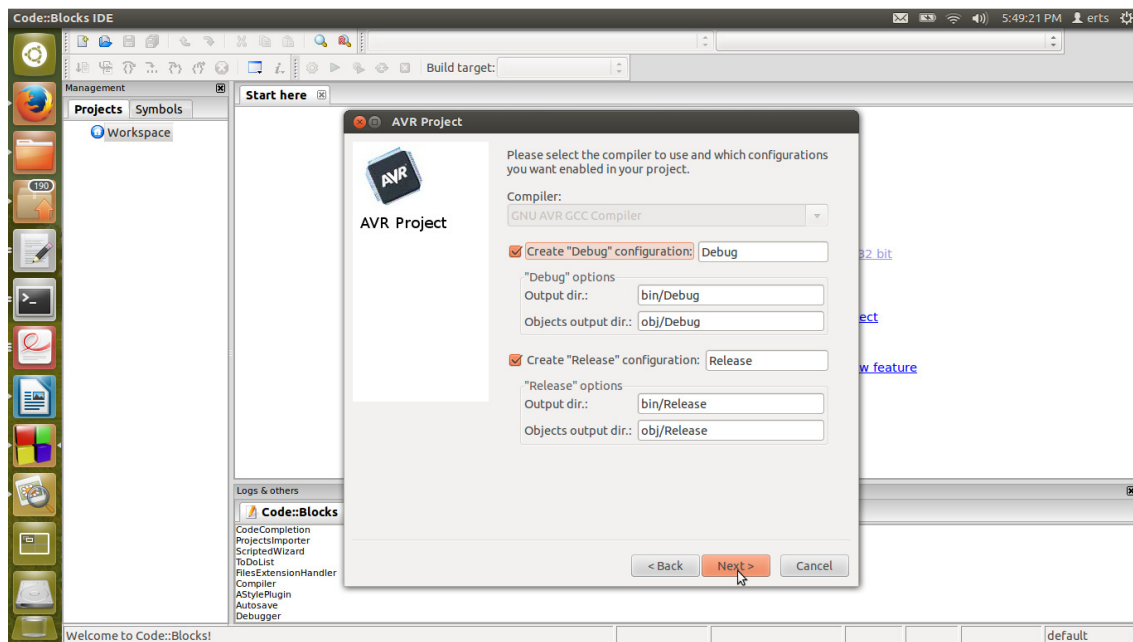


Figure-6: Compiler selection window

Step2.4

In next window select the processor as atmega2560. **Select the processor from drop-down list only.**

Select the crystal frequency (F_CPU) as 14745600UL. Check Create hex files button and leave other setting as default, as illustrated in figure-7 below. Click on Finish button.

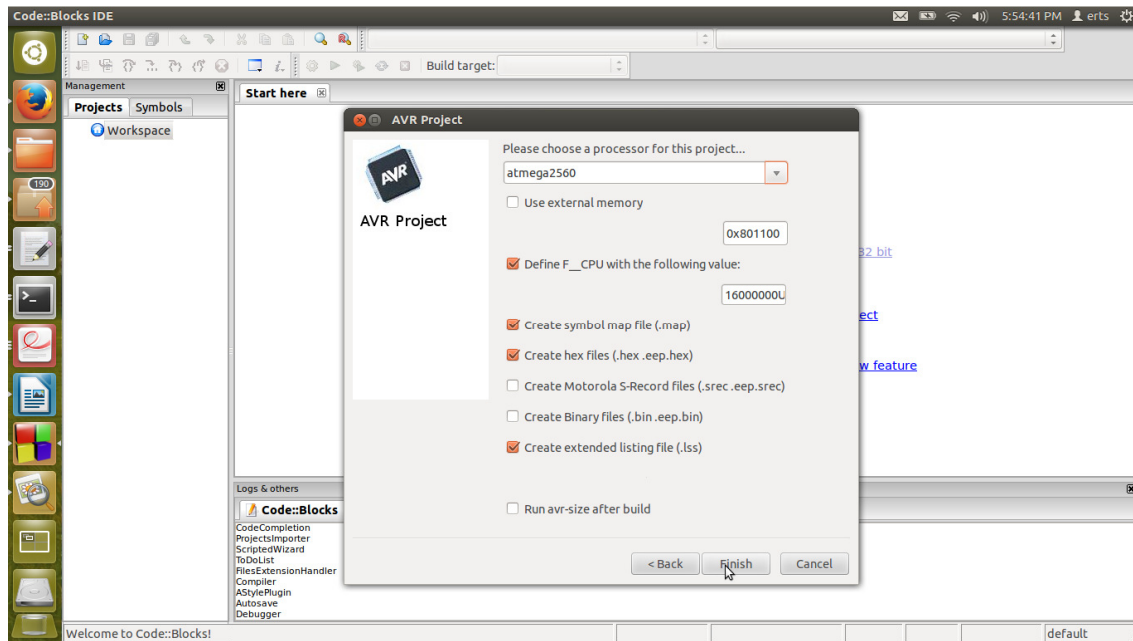


Figure-7: Compiler selection window

Step2.5 Text editor will open as shown in figure-8 below. We can start writing the C-program.

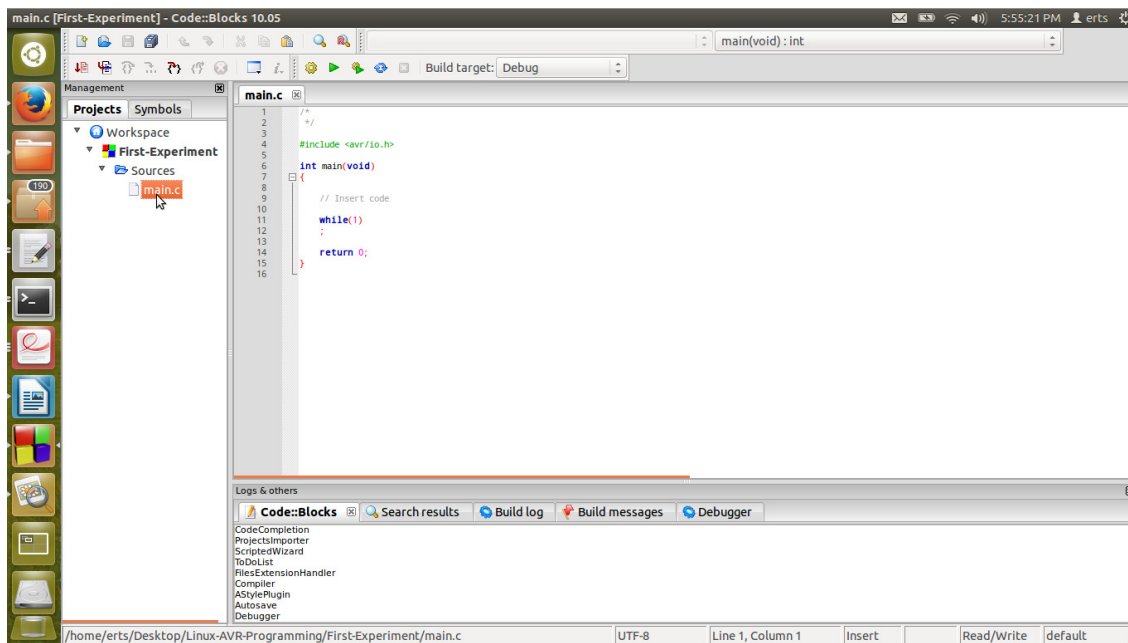


Figure-8: CodeBlocks Text editor for writing C program

Code can be compiled by clicking on build button as shown in figure-9 below.

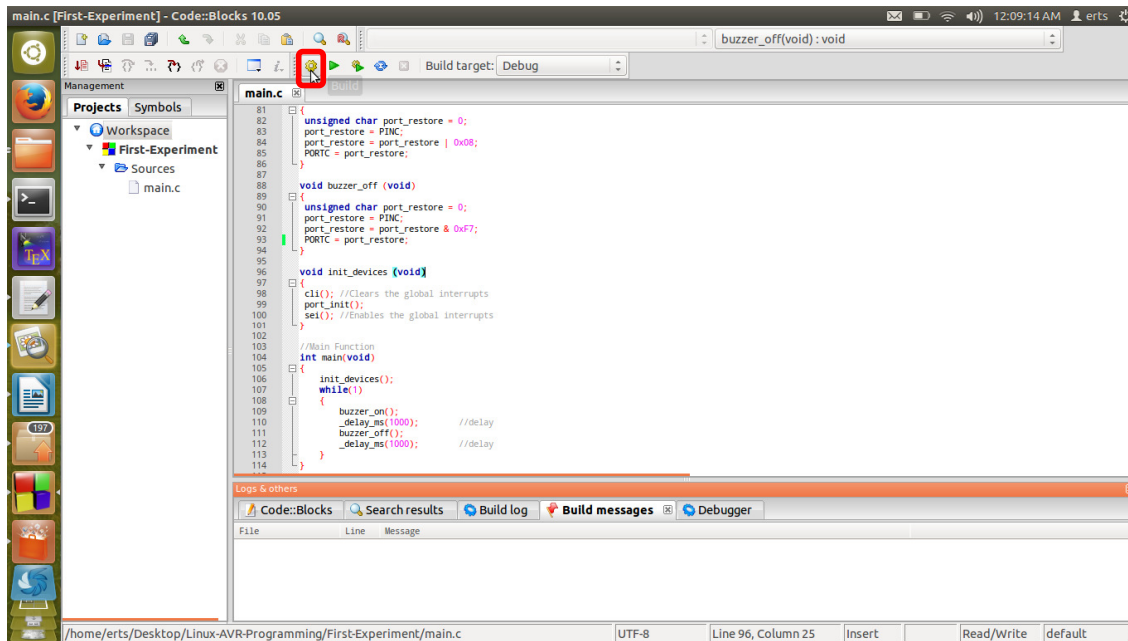


Figure-9 Compiling/Building the program

As shown in figure-10, “Build messages” window at the bottom of IDE shows various messages generated after compilation. Errors if any, will also be shown in this window.

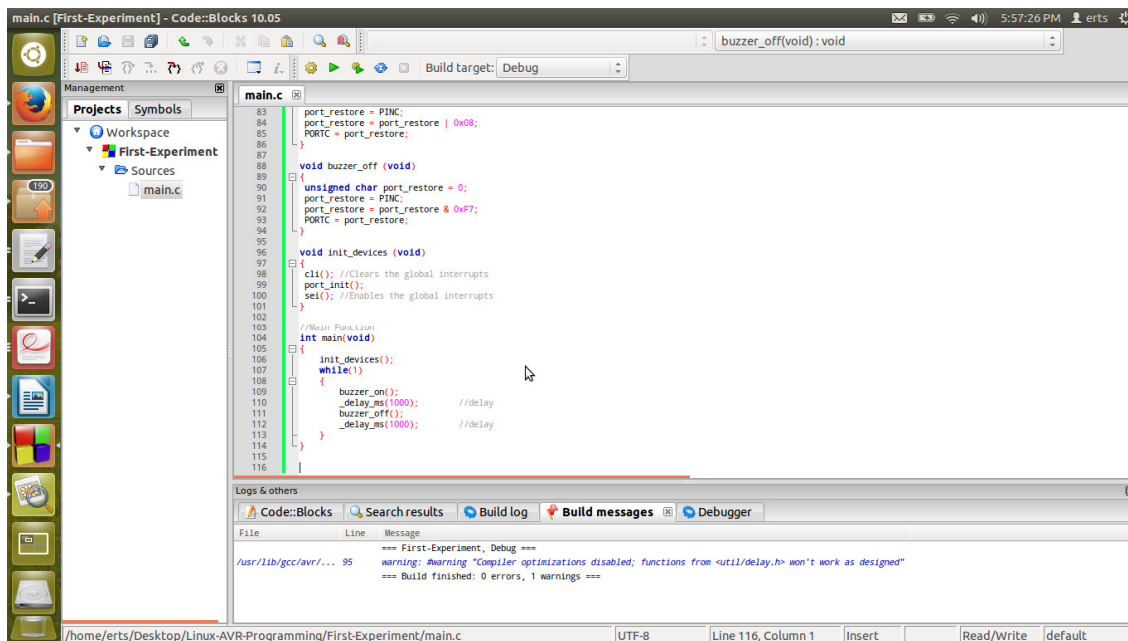


Figure-10 Build message window

Step2.6

After successful compilation of program, .hex file will be generated. This .hex file can be easily located inside the bin folder which is present in the project folder.

Next step is to load the generated .hex file in microcontroller memory.

Step:3 Loading hex file on microcontroller memory

Step3.1

Download and save the script file ("avrdude_script.sh") supplied with document on your desktop or any desired location. This script file has bunch of commands which will help us load the .hex file on microcontroller. We have to execute this script file with a .hex file as one argument. Below mentioned steps will describe the process in detail.

Step3.2

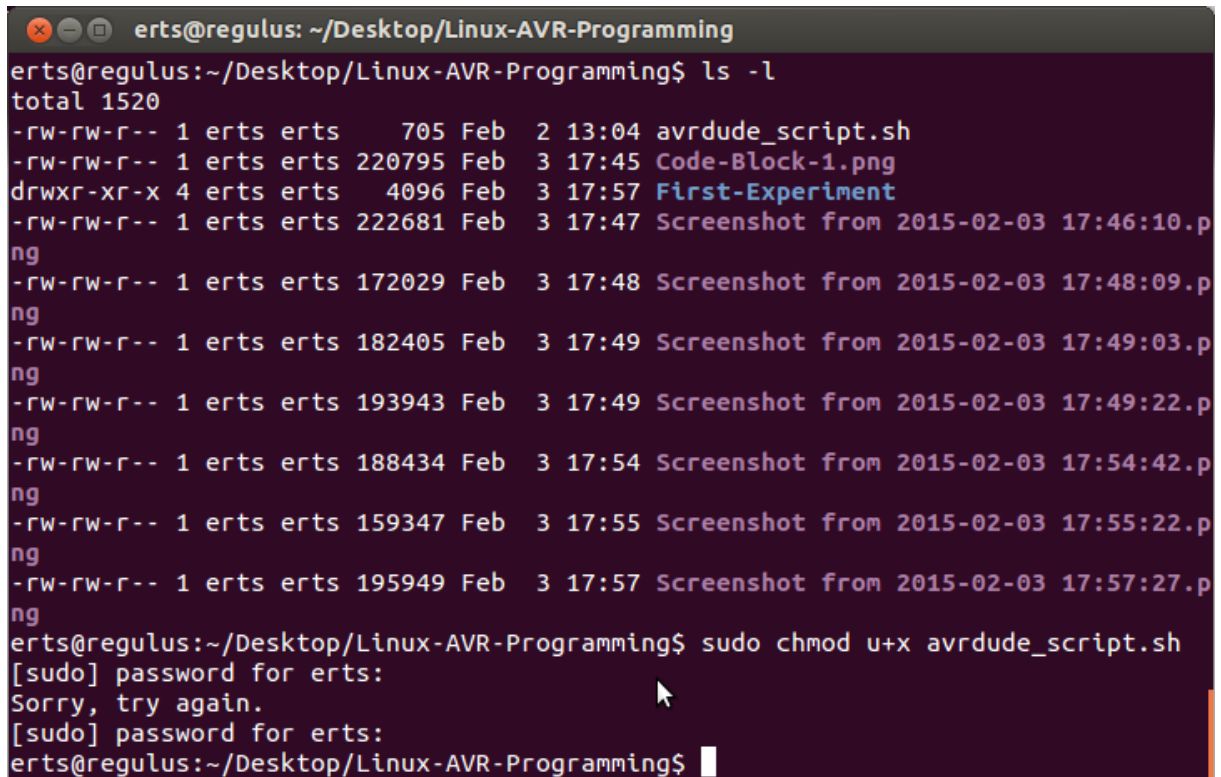
In linux we need to set file privileges and permissions for user. A user can read a file, write to file or execute a file. Script file has to have execute privilege.

Open linux terminal and cd to the folder where script file is saved. This script file can be executed with root privilege.

Use linux command `ls -l` to list the folder content in long list format. This format will display permission for root and other users.

As shown in figure-11, avrdude has only read and write privilege for root user. We have to first set a execute privilege to script file. This is done by following linux command

```
Sudo chmod u+x avrdude_script.sh
```



```
erts@regulus: ~/Desktop/Linux-AVR-Programming
erts@regulus:~/Desktop/Linux-AVR-Programming$ ls -l
total 1520
-rw-rw-r-- 1 erts erts 705 Feb 2 13:04 avrdude_script.sh
-rw-rw-r-- 1 erts erts 220795 Feb 3 17:45 Code-Block-1.png
drwxr-xr-x 4 erts erts 4096 Feb 3 17:57 First-Experiment
-rw-rw-r-- 1 erts erts 222681 Feb 3 17:47 Screenshot from 2015-02-03 17:46:10.p
ng
-rw-rw-r-- 1 erts erts 172029 Feb 3 17:48 Screenshot from 2015-02-03 17:48:09.p
ng
-rw-rw-r-- 1 erts erts 182405 Feb 3 17:49 Screenshot from 2015-02-03 17:49:03.p
ng
-rw-rw-r-- 1 erts erts 193943 Feb 3 17:49 Screenshot from 2015-02-03 17:49:22.p
ng
-rw-rw-r-- 1 erts erts 188434 Feb 3 17:54 Screenshot from 2015-02-03 17:54:42.p
ng
-rw-rw-r-- 1 erts erts 159347 Feb 3 17:55 Screenshot from 2015-02-03 17:55:22.p
ng
-rw-rw-r-- 1 erts erts 195949 Feb 3 17:57 Screenshot from 2015-02-03 17:57:27.p
ng
erts@regulus:~/Desktop/Linux-AVR-Programming$ sudo chmod u+x avrdude_script.sh
[sudo] password for erts:
Sorry, try again.
[sudo] password for erts:
erts@regulus:~/Desktop/Linux-AVR-Programming$
```

Figure-11 Setting the file permission

File permission has to be set only once on a machine.

Step3.3

In terminal use following command to load the hex file.

```
./avrdude_script.sh -f <filename.hex>
```

Note: Filename will be the .hex file which needs to be loaded on the microcontroller. You will have to give the complete path of file. Easier way of doing this is to keep the terminal and directory window beside each other and drag the .hex file from directory and drop it into the terminal after `./avrdude_script.sh -f`

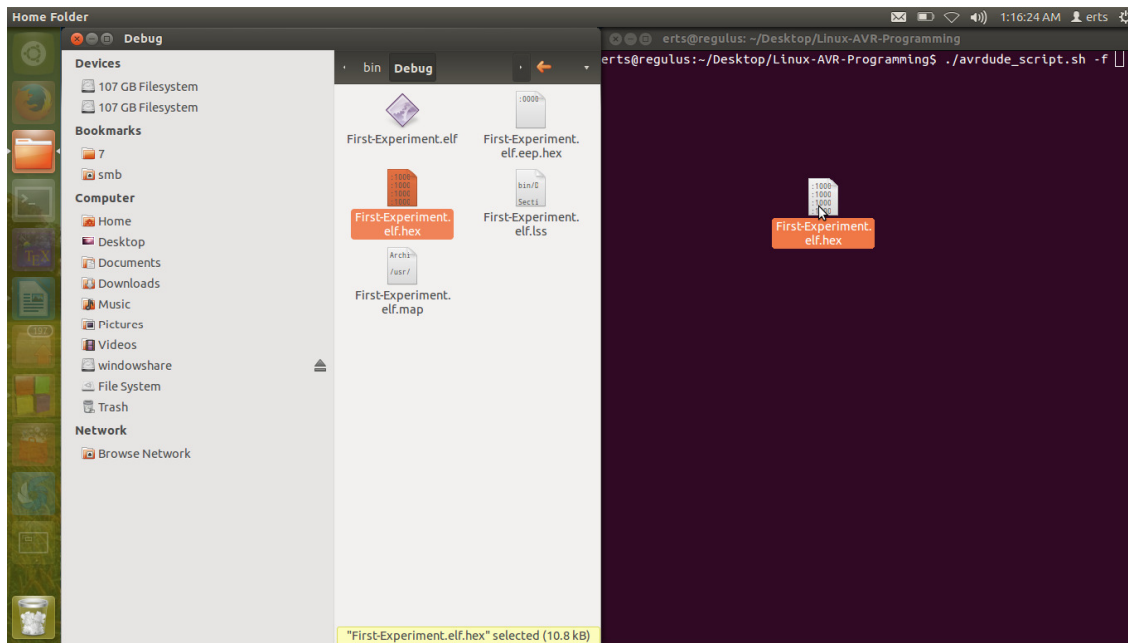


Figure-12 Drag and Drop file in terminal

Program will be loaded on microcontroller and terminal will show the status as shown in figure-13 below.


```
erts@regulus: ~/Desktop/Linux-AVR-Programming
erts@regulus: ~/Desktop/Linux-AVR-Program... x erts@regulus: ~/Desktop/Linux-AVR-Program... x
riment/bin/Debug/First-Experiment.elf.hex"
avrdude: writing flash (3824 bytes):

Writing | ##### | 100% 2.88s

avrdude: 3824 bytes of flash written
avrdude: verifying flash memory against /home/erts/Desktop/Linux-AVR-Programming
/First-Experiment/bin/Debug/First-Experiment.elf.hex:
avrdude: load data flash data from input file /home/erts/Desktop/Linux-AVR-Progr
amming/First-Experiment/bin/Debug/First-Experiment.elf.hex:
avrdude: input file /home/erts/Desktop/Linux-AVR-Programming/First-Experiment/bi
n/Debug/First-Experiment.elf.hex contains 3824 bytes
avrdude: reading on-chip flash data:

Reading | ##### | 100% 2.78s

avrdude: verifying ...
avrdude: 3824 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done. Thank you.

erts@regulus:~/Desktop/Linux-AVR-Programming$
```

References:

1. File Permission in linux. <https://help.ubuntu.com/community/FilePermissions>