DAT096 Lab Manual

Bhavishya Goel

January 21, 2012

1 Introduction

This lab manual aims to introduce you to the hardware and software framework that will be used for the DAT096 project course. At the end of this lab you should be able to do the following:

- Synthesize VHDL code for FPGA
- Load the synthesized bit file to FPGA
- Cross-compile the mpg123 source code for the Leon3 processor
- Load a sample MP3 file and a compiled binary into on-board DRAM
- Run code on hardware and collect profiling data

This lab does not affect your grade in the course and there is no deliverable expected from this lab. This lab exists solely to get you started on the project. To get the most out of this lab, you should have read the project specifications document before coming to the lab.

2 VHDL Code Synthesis

We will use the Leon3 soft processor core to run our MP3 decoder as mentioned in the specification document. Leon3 core's VHDL code is available as open source from Aeroflex Gaisler as part of GRLIB [2] from the Aeroflex Gaisler's website. GRLIB is a set of IP cores provided by Gaisler for system on chip (SOC) development using the Leon3 processor. During this project, we will work within the framework of GRLIB to easily interface our own VHDL modules with the Leon3 processor. You may want to give a quick glance to the GRLIB product brief [2] to get an idea of how different things get together when working with the Leon3 processor.

Start with extracting the design template given by Tron System AB to your home directory from the tar file located at `http://www.cse.chalmers.se/research/group/Fusion/MP3.bz2`. You will do all your development in this directory. Go to leon3-digilent-atlys folder. It contains the top vhdl module *leon3mp.vhd* that instantiates the Leon3 processor and other peripherals. For this project, you will need to understand how to enable/disable and instantiate peripherals from this file, but not for this lab. The *Makefile* in this folder has a variable called *GRLIB* that sets the path to the complete GRLIB framework.

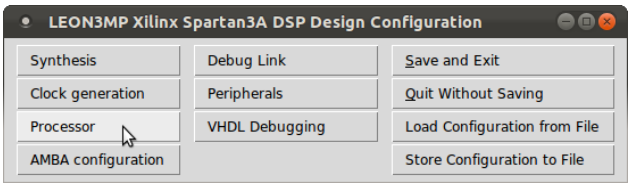

Figure 1: Main Configuration Window

This framework includes the vhdl code for the Leon3 processor and its supporting peripherals. For the most part of this project, you will not need to be concerned about the Leon3's VHDL code or the code of any other peripherals in GRLIB.

Before we can start synthesizing the design, we will need to assign a unique MAC address to our FPGA board, since we will use the Ethernet to load and debug the software. Each project group will have a unique MAC address and IP number, which you will use during the entire duration of the project. Your group will be given your MAC address and IP from the lab instructor in the first lab. Alternately, you can also find it in the file *MAC_IP* at *leon3-digilent-atlys/groupXX* where *XX* is your group number.

In the folder *leon3-digilent-atlys*, enter command

```
$ make xconfig
```

This will open the GRLIB configuration window as shown in figure 2. Click on *Debug Link* button to open the debug configuration window as in figure 2. Enter your IP and MAC address in their respective fields. Press *Main Menu* button and then *Save and Exit* on the main window to save your configuration options in *config.vhd* file.

You can start with design synthesis. You will need to start with setting the environment variables for Xilinx ISE[1].

```
$ source /chalmers/sw/sup/xil/13.2/ISE_DS/settings64.sh
```

Once you have the ise tools in your path, you can use the makefile to synthesize the VHDL code as below.

```
$ make ise
```

Depending upon the speed of your computer and your routing complexity, this may take between 1-2 hours to complete. So let it run and open another terminal to start other tasks. For this lab, you can use the bit file provided in the template in the folder *groupXX* where *XX* is your group number.

3 Loading bit file in FPGA

Because of software limitations, we cannot program FPGA board from linux machines in the lab. There are four Windows machines designated in the lab for this purpose. Each Windows machine is attached to one

[1]There is some problem with the group permissions for xilinx directory. If that problem is not resolved by the time your lab starts, you can skip this step for now and use the existing bit file.

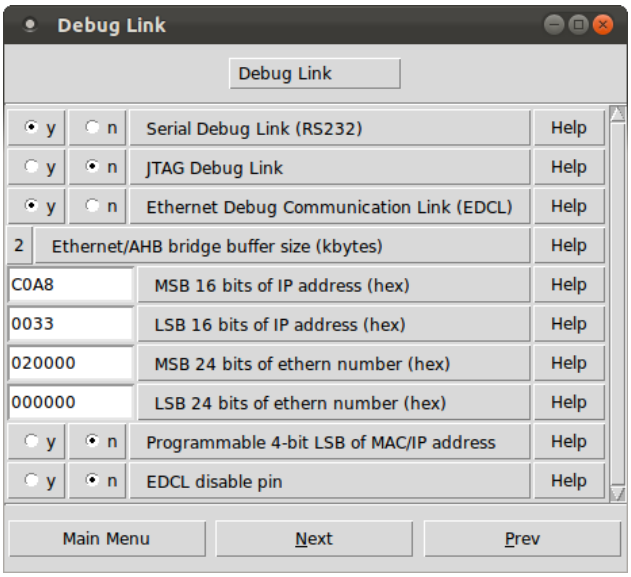

Figure 2: Debug Link Configuration Window

FPGA board. Use the FPGA board which has the sticker with your group's number. You will use the same board for the entirety of this course.

If your linux filesystem is not already mounted, map it from *\\sol.ita.chalmers.se\CID*. If you don't know how, ask the lab instructor. Open *Start Menu→Programs→Digilent→Adept* on the Windows machine. Choose the bit file belonging to your group and click on *Program*. Once the FPGA is successfully programmed, you can return to your linux machine.

4 Cross-compilation of MPG123 Code

The folder *mpg123_source_code* contains the mpg123 source code ported to the Leon3 processor by Tron engineers. Enter the folder and type following commands to cross-compile the source code for the Leon3 processor[2].

```
$ export PATH=/opt/rtems-4.10/bin:$PATH
$ make mp3
```

PATH was updated to add SPARC RTEMS compiler binaries to the system path. The make command creates the application binary *mpg123* to be loaded into system memory of the design.

Before we can load and run application on FPGA, we need a sample MP3 file to test. This can be found in the *samples* directory located in the top folder. To load this data in the memory of the FPGA board, we

[2]For the first lab, skip this step and use the precompiled binary *Precompiled/mpg123-mp3rawnoidct* as SPARC RTEMS compiler is not installed in the lab machines yet. You can also download rtems binaries [3]to your personal laptop and try to compile mpg123

will need to convert it into Motorola SREC format. We can use the program *bin2srec* to do this conversion. Enter following commands in your terminal from *MP3* folder:

```
$ bin/bin2srec -o 41000000 -a 4 -l 16 samples/test.mp3 > mp3.srec
```

This will convert the *test.mp3* file to SREC format, with the MP3 data stored at the fixed address *0x41000000*. The ported mpg123 code reads MP3 encoded data from this fixed address and stores decoded data at *0x41800000*

5 Profiling

To communicate with your FPGA board, we will use the debug tool called *grmon* [1] provided by Aeroflex Gaisler. Enter following commands in the terminal (skip the first command if grmon is already in path).

```
$ PATH=/chalmers/sw/unsup64/grlib-gpl-1.1.0-b4113/bin:$PATH
$ grmon -eth -u -ip <your board's IP address>
```

If everything went right, you should now be connected to your board. At the GRLIB prompt, you can type

```
GRLIB> info sys
```

to take a look at your hardware configuration.

Load the MP3 file and your binary into the memory using following commands:

```
GRLIB> load mp3.srec
GRLIB> load mpg123_source_code/mpg123
```

Now turn on profiling, run the code and look at profiling results.

```
GRLIB> profile 1
GRLIB> run
GRLIB> prof
```

Look at most time consuming functions and look up these functions in the software. Start thinking about which of these functions you would like to accelerate.

Type *help* at the GRLIB prompt to see what other functions are supported by GRLIB. For example, you can examine the memory address, write to memory address, dump a certain address range to file and so on.

References

[1] Aeroflex Gaisler AB. GRMON User Manual. http://www.gaisler.com/doc/grmon.pdf.

[2] Aeroflex Gaisler AB. Leon3/GRLIB Product Brief. http://www.gaisler.com/doc/Leon3%20Grlib%20folder.pdf.

[3] Gaisler Research AB. Download Cross Compiler System. http://www.gaisler.com/cms/index.php?option=com_content&task=view&id=161&Itemid=109.