Red Pitaya

Specifications

Raphael Frey
Noah Hüsser

July 25, 2017
Version 0.0.1

Contents

CHAPTER 1

Work Packages

1.1 Filter Design

Contains the work packages relating to filter design in Matlab and filter implementation in the FPGA tool chain.

1.1.1 Filter Research

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Raphael Frey
DESCRIPTION:	Research the digital filter technologies which will be needed for this project. In particular, this includes FIR and CIC filters, since Xilinx provide predefined blocks for these filters in their toolchain. Besides the general knowledge on digital filters, the Xilinx toolchain must be researched in order to understand the FIR and CIC filter blocks and allow their usage.

1.1.2 Matlab Scripts for Filter Design

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey
Description:	Matlab scripts which design various filter chains. These consist of one or several dispatcher scripts where the filter chains are specified in terms of their frequency band properties, and scripts which actually design the filters according to these specifications. Specifically, scripts for generating CIC filters and their compensation filters, as well as FIR filters are needed. For FIR filters, the resulting coefficients are to be saved in files so that they can be loaded by the Xilinx tool chain. If possible Matlab's parallelism should be exploited to reduce filter design times.

1.1.3 TCL Scripts for Filter Evaluation

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey
Description:	In order to assess the resource usage of the filter chains which are designed by Matlab, the respective filter blocks (CIC, FIR) need to be implemented in Vivado and a bitstream must be compiled. Since the number of filters generated by package 1.1.2 can be very large (dozens or even hundreds of filters), this process must be automated to be of any use. For this, TCL scripts are used. For FIR filters, the scripts load the coefficient files which have been generated by Matlab. Vivado will generate usage reports which can then be automatically post-processed.

1.1.4 Documentation

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey
Description:	Documentation for the filter design tool chain. The documentation should be sufficiently detailed so that the tool chain can be used by a person who wishes to compile a new bitstream with different filters.

1.2 Documentation

1.2.1 Disposition

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Noah Hüsser
Description:	A general outline of the thesis document.

1.2.2 Report

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey, Noah Hüsser
Description:	The actual thesis document.

1.3 Firmware

1.3.1 Linux

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	Boot an Ubuntu Linux onto the Red Pitaya.The target here is to have a ARM Linux distribution which boots fine on the RedPitaya. The official sources from the RedPitaya project should be used as they contain a lot of patches for the RedPitaya hardware. A standalone toolchain has to be developped as the compiled versions do not contain the right device tree (DT) and board support package (BSP). Compiling in the right DT and BSP is not part of this task.

1.3.2 Server Application

1.3.2.1 Design Decisions

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	It has to be determined what technology should be used to provide the server side of the data transmission. The server needs to be able to perform IOCTL calls and to serve data with the chosen protocol over the network. It has to run on the arm linux and shouldn't have too many dependencies. A C-like language will most likely serve best since it's close to the kernel.

1.3.2.2 Server Application

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	A server application has to be written that reads data from the Logger and transmits it over the network. It needs to be able to configure the logger via RPCs that are mapped to IOCTLs. It should read data from /dev/logger which is presented by the kernel module.

1.3.2.3 Documentation

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Raphael Frey, Noah Hüsser
DESCRIPTION:	Documentation for the server application build process.

1.3.3 FPGA

1.3.3.1 Minimal Working Example

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	Get a minimal working example project running on the Red Pitaya. This requires importing the cores from Pavel Denim's project as well as compiling the Linux for the Red Pitaya. Correct functionality of the Red Pitaya hardware, particularly the ADC, is to be verified via Anton Potochnik's frequency counter.

1.3.3.2 Port Zynq Logger

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	Port the Zynq Logger to the Red Pitaya. The main challenges will be the porting of its interface (for which a block `axis2datalanes` is developed), as well as understanding and properly implementing the device tree and the kernel module which is needed for the Linux OS to interface with the logger hardware.

1.3.3.3 Filter Chains

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey
Description:	Implement the filter chains as designed in Matlab on the FPGA. This requires correct functionality of the FIR and CIC filter blocks.

1.4 Validation

1.4.1 Build Process

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Raphael Frey
Description:	Build the entire project (Linux, bitstream), flash result onto the Red Pitaya and verify correct functionality.

1.5 Frontend

1.5.1 Techology Research

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Noah Hüsser
Description:	Research the possibilities to design and implement a nice frontend for the RedPitaya recording system. The RedPitaya Project has a built-in webinterface which allows the display of signals and rudimentary calculations. Furthermore Prof. Gut has implemented a spectrum analyzer in Java which interfaces with the RedPitaya. Both approaches are feasible and should be evaluated properly. Furthermore it has to be evaluated how the data should be transferred to the frontend. This descision correlates a lot with the selection of a programming language since e.g. web technologies do not support UDP. Finally, existing libraries have to examined very carefully. There is a lot of libraries around data transmission and GUI design, but not all of them are feasible.

1.5.2 Implementation of a basic Plotter

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	Techology Research
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	At first a simple plotter will be implemented. It should not do more than plot a given vector of data points on a canvas. The plotting should be done performantly at 60 frames per second not being dependant on the size of the data vector. Of course more data will slow down the plotting, but it should perform so well that the size of vectors up to 64 kilosamples do plot nicely. For this task it is recommended to use existing libraries which make use of the GPU such as OpenGL.

1.5.3 Implementation of a basic Receiver

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	Techology Research, TODO: server
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	A simple receiver which receives recorded samples over the network should be created. For this task a simple emmitter which emmits test samples without actually recording anything should be implemented. In further tasks it can be used conveniently without the need of a running RedPitaya board. The receiver should simply receive samples as packets with lengths of powers of two over the network. Again, a good, performant and publicly accepted library should be used to reduce workload and ensure reliability.

1.5.4 Combining the Plotter and the Receiver

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	Implementation of a basic Plotter, Implementation of a basic Receiver
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	The plotter and the receiver should be combined to display vecors received over the network. This should happen fluently without any delays.

1.5.5 Setting a Trigger on the Logger Core

Start Date:	
End Date:	
Hours:	
Ancestors:	Combining the Plotter and the Receiver
Descendants:	
Assignee:	Noah Hüsser
Description:	The application should have a possibility to set a trigger type for any channel with the possibility to do this using a GUI.

1.5.6 Implement Proper Scaling

Start Date:	
End Date:	
Hours:	
Ancestors:	Setting a Trigger on the Logger Core
Descendants:	
Assignee:	Noah Hüsser
Description:	The GUI should not only plot a qualitatively correct signal but also show it's quantities. A grid should be programmed into the GUI which can be used to read metrics of a signal. A panel on the GUI should display the current grid size. It should be possible to scale signals using the mouse or the GUI. Grids should adjust with the current scale.

1.5.7 Implement Trigger Modes

Start Date:	
End Date:	
Hours:	
Ancestors:	Combining the Plotter and the Receiver
Descendants:	
Assignee:	Noah Hüsser
Description:	It should be possible to use the scope application in *normal*, *auto*, and *single* mode. Those modes should behave as follows:

normal	the server sends a packet when a trigger was fired and repeats this procedure
auto	the server sends a packet when a trigger was fired or a timeout has been hit and re
single	the server sends a packet after a first trigger and does not repeat this procedure

1.5.8 Implement an FFT

Start Date:	
End Date:	
Hours:	
Ancestors:	None
Descendants:	
Assignee:	Noah Hüsser
Description:	An FFT should be calculated and displayed live.

1.5.9 Implement Power Measuremenets

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	Calculate the power in the signal. It has to be possible to calculate the power between two markers to limit the calulation on a certain part of the signal.

1.5.10 Implement SNR Calculation

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	The SNR of the signal between two markers has to be calculated automatically. As a reference use the Java applicaion of Prof. Gut.

1.5.11 Implement THD Calculation

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	The THD of the signal spectrum has to be calculated automatically. As a reference use the Java application of Prof. Gut.

1.5.12 Implement Dual Channel Possibilities

START DATE:	
END DATE:	
HOURS:	
ANCESTORS:	None
DESCENDANTS:	
ASSIGNEE:	Noah Hüsser
DESCRIPTION:	The entire transmission and plotting of all the signals should be implemented for dual channel capabilities. The FPGA code supports up to eight channels whilst only supplying two to the server since the RedPitaya board only supports two physical channels.