Testing and evaluation of the integratability of the Senior processor

Examensarbete utfört i Datateknik
vid Tekniska högskolan vid Linköpings universitet
av
Alexander Hedin
LiTH-ISY-EX--11/4510--SE
Linköping 2011
Testing and evaluation of the integratability of the Senior processor

Examensarbete utfört i Datateknik vid Tekniska högskolan i Linköping av

Alexander Hedin

LiTH-ISY-EX--11/4510--SE

Handledare: Andreas Ehliar
ISY, Linköpings universitet

Examinator: Olle Seger
ISY, Linköpings universitet

Linköping, 10 June, 2011
The first version of the Senior processor was created as part of a thesis project in 2007. This processor was completed and used for educational purposes at Linköpings University. In 2008 several parts of the processor were optimized and the processor expanded with additional functionality as part of another thesis project. In 2009 an EU funded project called MULTI-BASE started, in which the Computer Division at the Department of Electrical Engineering participated in. For their part of the MULTI-BASE project, the Senior processor was selected to be used. After continuous revision and development, this processor was sent for manufacturing.

The assignment of this thesis project was to test and verify the different functions implemented in the Senior processor. To do this a PCB was developed for testing the Senior processor together with a Virtex-4 FPGA. Extensive testing was done on the most important functions of the Senior processor. These tests showed that the manufactured Senior processor works as designed and that it alone can perform larger calculations and use external hardware accelerators with the help of its various interfaces.
Abstract

The first version of the Senior processor was created as part of a thesis project in 2007. This processor was completed and used for educational purposes at Linköpings University. In 2008 several parts of the processor were optimized and the processor expanded with additional functionality as part of another thesis project. In 2009 an EU funded project called MULTI-BASE started, in which the Computer Division at the Department of Electrical Engineering participated in. For their part of the MULTI-BASE project, the Senior processor was selected to be used. After continuous revision and development, this processor was sent for manufacturing.

The assignment of this thesis project was to test and verify the different functions implemented in the Senior processor. To do this a PCB was developed for testing the Senior processor together with a Virtex-4 FPGA. Extensive testing was done on the most important functions of the Senior processor. These tests showed that the manufactured Senior processor works as designed and that it alone can perform larger calculations and use external hardware accelerators with the help of its various interfaces.
Den första versionen av Senior processorn skapades som en del i ett examensarbe-
te under 2007, denna processor färdigställdes och användes i utbildningssyfte på
Linköping Universitet. 2008 optimerades flera delar av processorn och utökades
med extra funktionalitet som del av ytterligare ett examensarbete. 2008 startade
ett EU finansierat projekt vid namn MULTI-BASE, som ISYs Datortekniks avdel-
nings deltar i. Till deras del av MULTI-BASE projektet valdes Senior processorn att
anvandas, efter ytterligare utveckling skickades denna processor för tillverkning.

Detta examensarbete hade i uppgift att testa och verifiera de olika funktionerna
som Senior processorn har implementerats med. För att göra detta tillverkades ett
cretskort som ska användas för att testa Senior processorn tillsammans med en
Virtex-4 FPGA. Utförliga tester gjordes på de viktigaste funktionerna hos Senior
processorn, dessa tester visade att den tillverkade Senior processorn fungerar som
planerat. Den kan på egen hand utföra större beräkningar och använda sig av
externa hårdvare acceleratorer med hjälp av sina olika gränssnitt.
I want to thank my supervisor, for helping me throughout the entire project. Without him the results achieved would not have been possible.

My examiner, for taking the time and giving me the opportunity to have my thesis defences as planned.

Professor Dake Liu, for giving me this opportunity to work with this project.

My office mates, for being helpful and making the long days enjoyable.

My fiancée, who has been very understanding and extremely patient.

And my son, for making me forget the bad days and giving me the motivation to keep on working.
# Contents

1 Introduction ........................................ 1
   1.1 MULTI-BASE ........................................ 1
   1.2 Objective .......................................... 2
   1.3 Notations .......................................... 2
   1.4 Target Audience ................................... 2
   1.5 Report outline ................................... 3
   1.6 Abbreviations ................................... 3

2 The Senior Processor ............................... 5
   2.1 Senior DSP Core .................................. 6
      2.1.1 Accumulators ................................ 6
      2.1.2 Registers ..................................... 6
      2.1.3 Instruction set ................................ 6
      2.1.4 FFT addressing ................................ 6
   2.2 Internal memories .................................. 7
   2.3 Interfaces .......................................... 7
      2.3.1 SPI ............................................. 7
      2.3.2 Peripheral interface ......................... 8
      2.3.3 Accelerator interface ....................... 10
   2.4 Interrupt controller ................................ 10
   2.5 DMA controller ................................... 10

3 Test planning ........................................ 13
   3.1 Order of tests ................................... 13
   3.2 Interfaces .......................................... 13
   3.3 Functionality ...................................... 14
   3.4 Simulations ....................................... 14

4 PCB development process ......................... 15
   4.1 CAD tools .......................................... 15
   4.2 Abstract design .................................. 15
      4.2.1 Signals of interest ........................... 16
      4.2.2 Available testing equipment ............... 17
      4.2.3 Design decisions ............................. 17
      4.2.4 Design completion ......................... 18
# Contents

4.3 Schematic .......................... 19
  4.3.1 Available data .................. 19
  4.3.2 Component symbols .............. 19
  4.3.3 Finished schematic ............. 22

4.4 Layout ............................ 26
  4.4.1 Available data .................. 26
  4.4.2 Constructing pads, cells and parts .... 27
  4.4.3 Placement ....................... 28
  4.4.4 Routing .......................... 28
  4.4.5 Plane fill ....................... 30

4.5 Manufacturing ..................... 30

5 Testing process ..................... 35
  5.1 Equipment ........................ 35
  5.2 Senior PCB tests ................. 36
    5.2.1 Unmounted PCB ............... 36
    5.2.2 Mounted PCB .................. 36
  5.3 Senior processor tests .......... 37
    5.3.1 SPI verification .............. 37
    5.3.2 Measuring power consumption .. 37
    5.3.3 Functionality tests .......... 38
  5.4 Tests with Accelerator 1 : SIMD processor .... 38
  5.5 Tests with Accelerator 2 : Reciprocals .. 39

6 Results ............................. 41
  6.1 Senior PCB review ............... 41
    6.1.1 Unmounted PCB ............... 41
    6.1.2 Mounted PCB .................. 42
  6.2 Senior processor results .......... 44
    6.2.1 SPI evaluation ............... 44
    6.2.2 Power consumption .......... 44
    6.2.3 Functionality ............... 45
  6.3 Accelerator 1 : SIMD processor .... 46
  6.4 Accelerator 2 : Reciprocals ... 47

7 Conclusion .......................... 49
  7.1 Senior PCB ....................... 49
  7.2 Senior processor .................. 49

8 Future work ........................ 51
  8.1 What is left ....................... 51
  8.2 Possible applications ............ 51

A Senior PCB component specifications 55
Chapter 1

Introduction

The Computer Engineering Division of the Department of Electrical Engineering at Linköpings University, is participating in a European Commission funded project called MULTI-BASE [19]. The Computer Engineering Division has chosen to use the Senior Processor for their part in the MULTI-BASE project.

When the MULTI-BASE project started in 2008 the Senior processor only existed in Verilog code and had only been used for teaching purposes and in simulations. For the MULTI-BASE project, a selected number of tape-outs were made of the Senior processor. Work was initiated on a PCB to test the functionality of the taped-out version of the Senior processor.

1.1 MULTI-BASE

The MULTI-BASE project is funded by the European Commission and it is a Framework Programme 7 project [18]. The MULTI-BASE project was launched in January 2008 and is running for three years. The project is a collaboration between seven partners, both from industry and academia.

The complete name for Framework Programme 7 is the Seventh Programme for Research and Technological Development and it is the EU’s main instrument for funding research in Europe. The project will last for 6 years, from 2007 until 2013.

The motivation behind the MULTI-BASE project is to strengthen Europe’s leading position in end-to-end, high speed, mobile network systems technology. The MULTI-BASE consortium has focused on three areas of research:

- multi-tasking radio
- scalable and reconfigurable multi processor technology
- algorithm/architecture co-design for maximum energy efficiency.

For the University’s role in the MULTI-BASE project, a processor was needed that was easily integratable, with the ability to communicate with and control
a wide range of hardware. No such processor was available to the University at the start of the MULTI-BASE project, therefore it was decided to use the available Senior processor [16] and develop it further to match the requirements of the desired processor.

1.2 Objective

The Senior processor has been under development during the entire MULTI-BASE project. The Verilog code for the processor, additional units and the tool chain, from simulator to assembler has been continuously evaluated and updated. The development of a PCB that would be used to test the Senior processor, was started during the middle of 2010. But due to technical problems almost all data was lost.

At the start of this thesis project, only a rough schematic of the PCB was available. The schematic contained the Senior Processor, an AVR32 and power supplies. Cells and pads were also available.

The objective of this thesis project was to verify that the taped-out version of the Senior processor was functional and that it could be used for its purpose in the MULTI-BASE project.

To complete this objective, the following tasks had to be carried out:

- Completion of the Senior PCB
- Test and verification of the Senior processors basic functions
- Test and verification of the Senior processors use of external hardware.

1.3 Notations

The results of this thesis assignment has been achieved from an iterative process. Unexpected problems in later stages of the development process, were solved by going back to earlier stages, revise the current solution and redo the subsequent stages.

This report has been structured according to the different phases of the work process, instead of the chronological order of the different tasks. Writing in chronological order would have given a better understanding of the work process, but the report would have been to excessive and hard to structure.

1.4 Target Audience

This report is intended for people with basic knowledge of DSP technology and testing and verification of hardware in general. The level of this report is on a graduate level, in the fields of electronic and computer engineering.
1.5 Report outline

Chapter 1 includes a short introduction to the MULTI-BASE project, the main objectives of this thesis assignment and the outline of the report.

Chapter 2 describes the Senior processor and explains the features that are of interest for this thesis.

Chapter 3 describes the process of testing the Senior processor. Which interfaces that are of interest, how to test the Senior processors functionality, what kind of simulations that has been done and what they have shown.

The development process of the Senior PCB, from the rough schematic to the manufactured PCB is explained in chapter 4.

Chapter 5 goes through all tests that were conducted, that both verify that the PCB is functional and that the Senior processor works as intended.

Chapter 6 shows the results obtained from the tests and compare them to the results of previous simulations.

Chapter 7 describes the conclusions drawn from the tests and the simulations.

Chapter 8 discusses what is left to test regarding the PCB and the Senior processor.

1.6 Abbreviations
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGU</td>
<td>Address Generation Unit</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CISC</td>
<td>Complex Instruction Set Processor</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital-to-Analog Converter</td>
</tr>
<tr>
<td>DMA</td>
<td>Direct Memory Access</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processor</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In, First Out</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field-Programmable Gate Array</td>
</tr>
<tr>
<td>IC</td>
<td>Integrated Circuit</td>
</tr>
<tr>
<td>I/O</td>
<td>Input/Output</td>
</tr>
<tr>
<td>JTAG</td>
<td>Joint Test Action Group</td>
</tr>
<tr>
<td>LED</td>
<td>Light Emitting Diode</td>
</tr>
<tr>
<td>MCU</td>
<td>Microcontroller</td>
</tr>
<tr>
<td>MIMO</td>
<td>Multiple Input Multiple Output</td>
</tr>
<tr>
<td>MISO</td>
<td>Master in, Slave Out</td>
</tr>
<tr>
<td>MOSI</td>
<td>Master Out, Slave In</td>
</tr>
<tr>
<td>NC</td>
<td>Not Connected</td>
</tr>
<tr>
<td>NOP</td>
<td>No Operation Performed</td>
</tr>
<tr>
<td>OPA</td>
<td>Operand A</td>
</tr>
<tr>
<td>OPB</td>
<td>Operand B</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>PLL</td>
<td>Phase-Locked Loop</td>
</tr>
<tr>
<td>PM</td>
<td>Program Memory</td>
</tr>
<tr>
<td>RISC</td>
<td>Reduced Instruction Set Computer</td>
</tr>
<tr>
<td>RTL</td>
<td>Register Transfer Language</td>
</tr>
<tr>
<td>SCLK</td>
<td>System Clock</td>
</tr>
<tr>
<td>SD/MMC</td>
<td>Secure Digital/MultiMediaCard</td>
</tr>
<tr>
<td>SIMD</td>
<td>Single Instruction, Multiple Data</td>
</tr>
<tr>
<td>SMA</td>
<td>SubMiniature version A</td>
</tr>
<tr>
<td>SPI</td>
<td>Serial Peripheral Interface</td>
</tr>
<tr>
<td>SS</td>
<td>Slave Select</td>
</tr>
<tr>
<td>TCK</td>
<td>Test Clock</td>
</tr>
<tr>
<td>TMS</td>
<td>Test Mode Select</td>
</tr>
<tr>
<td>TDI</td>
<td>Test Data In</td>
</tr>
<tr>
<td>TDO</td>
<td>Test Data Out</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver/Transmitter</td>
</tr>
<tr>
<td>UCF</td>
<td>Universal Communication File</td>
</tr>
</tbody>
</table>
Chapter 2

The Senior Processor

The Senior processor was developed at the Division of Computer Engineering at Linköpings University. The first version was developed as a thesis project in 2007. This processor was revised and used for educational purpose. In 2008 it was used in a new thesis project [16], where the objective was to optimize various parts of the DSP core. This version of the Senior processor resembles the current version used in the MULTI-BASE project.

The Senior test chip was synthesized and manufactured to be able to run with a clock frequency of 200 MHz.

Figure 2.1. Overview of the Senior test-chip

In Figure 2.1 the content of the Senior test-chip is shown. The Senior processor consists of the Senior DSP Core, a PM and two DMs, a Host interface, an Accelerator and Peripheral interface, a DMA engine and an interrupt controller.
The parts of the Senior processor that are of interest in this thesis project will be explained in this chapter.

2.1 Senior DSP Core

The Senior processor is a 6-stage pipeline, single issue DSP processor for applications such as audio decoding, voice encoding, program flow control for video encoding and bit manipulation. It has a RISC architecture with added CISC functionality to be able to perform convolution calculations. Jump and subroutine instructions can use up to three delay slots, decreasing the number of unused clock cycles and the need of NOP instructions. The AGU has been implemented with a hardware repeat function, which reduces the number of clock cycles compared when using conditional jump instructions.

2.1.1 Accumulators

There are 4 accumulators available in the Senior core, acr0 to acr3, that are 32-bit wide and are used for double precision calculations by the ALU and MAC units. The MAC unit uses 8 guards bits, giving a 40 bit internal computation resolution for the accumulators.

2.1.2 Registers

The Senior core has 32 general register, mainly used as instruction buffers. These registers, r0 to r31, are 16-bit wide and are addressed using 5 bit binary code. There are also 32 special purpose registers, sr0 to sr31, each with their specific function. All special purpose register are 16-bit wide, except to one used for bit reversal which is 3-bit wide. They are also addressed using 5-bit binary code, but unlike the general registers, they can only be assigned values using move instructions.

2.1.3 Instruction set

The Senior processor has an instruction set of over 60 instructions. These instructions include short and long arithmetic, short logic, short shift, move-load-store, iterative, flow control and alias instructions. The iterative instruction is the repeat function mentioned in Section 2.1. The flow control instructions consist of the jump, call, return and NOP instructions. The alias instruction have no physical implementation, since these instructions are used for handling the software stack located in DM1.

2.1.4 FFT addressing

The Senior processors instructions set has been optimized for performing the common FFT algorithm and the AGU has also altered for this purpose. It contains special hardware for generating addresses during FFT calculations. Using special
purposes registers and DM0 as the locations for the FFT input samples, the dedicated hardware generates addresses for real and imaginary part of each sample for each butterfly operation.

2.2 Internal memories

The Senior processor has one 32-bit PM and two 16-bit DM, that are addressed using 32-bits. The PM has up to \(2^{16}\) words while the DMs, DM0 and DM1, has a up to \(2^{15}\) words each. The least significant bit of the DM address is used to assign which bank of the DM to address. 8-bit data words can be read or written to the DMs at every address, but 16-bit words can only be read and written on even address spaces.

2.3 Interfaces

The Senior processor shown in Figure 2.1 has three main interfaces, the Host, Accelerator and Peripheral interfaces. The Host interface for the Senior processor is an SPI, while the Accelerator and Peripheral interfaces are both parallel interfaces.

2.3.1 SPI

The SPI is used as the boot loader interface for the Senior processor, shown in Figure 2.1. It is used for four functions:

- Start and stop the Senior processor
- Write to the Senior processors program memory
- Read from the Senior processors program memory
- Send data to the DMA controller.

The commands used for these functions are shown in Table 2.1. The UPDATE command is used to start and stop the Senior processor and also send data to the DMA controller.

The SPI consists of 4 wires, specified in Table 2.2. The interface is synchronized to the Senior processors system clock. To ensure correct operation and to avoid clock domain crossings, the serial clock (SCLK) is run at a slower speed than the system clock. The Senior processor acts as slave in SPI communications, meaning that the processor waits for SPI activity and responds accordingly.

The SPI protocol sends data according to Figure 2.2. When the SS is initiated the data transaction starts. 48 bits of data are sent to the Senior processor on the MOSI wire. At the same time the previous 48 bits are returned by the Senior processor on the MISO wire.
Table 2.1. SPI commands

<table>
<thead>
<tr>
<th>Command name</th>
<th>Command format</th>
</tr>
</thead>
</table>
| UPDATE       | [47:46]: Set to 0  
               | [33:32]: Content of page register  
               | [31]: Running/Reset state of core (1 to reset, 0 to start execution).  
               | [16]: If 1, write data to DMA engine. If 0, do nothing.  
               | [15:0]: Data to be sent to the DMA engine. |
| RESERVED     | [47:46]: Set to 1 |
| WRITE_PM     | [47:46]: Set to 2  
               | [45:32]: Address in program memory  
               | [31:0]: Instruction to write |
| READ_PM      | [47:46]: Set to 3  
               | [45:32]: Address in program memory |

Table 2.2. SPI signals [5]

<table>
<thead>
<tr>
<th>Signal</th>
<th>Explanation</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>MISO</td>
<td>Serial data</td>
<td>Output from Senior</td>
</tr>
<tr>
<td>MOSI</td>
<td>Serial data</td>
<td>Input to Senior</td>
</tr>
<tr>
<td>SCLK</td>
<td>Serial clk</td>
<td>Input to Senior</td>
</tr>
<tr>
<td>SS</td>
<td>Serial select (active low)</td>
<td>Input to Senior</td>
</tr>
</tbody>
</table>

When an SPI transaction is made, the result from the previous transaction is returned. In order to read the result of one SPI transaction, a second transaction must be made to view the result from the first one. When the READ_PM command is used, the result from the previous SPI transaction is returned during the READ_PM instruction. During the next SPI instruction, the result from the READ_PM is returned.

### 2.3.2 Peripheral interface

The Peripheral interface [16], shown in Figure 2.1 as the general purpose I/O, consists of three signals: data_i[16:0], data_o[16:0] and addr_o[5:0]. data_i[16] acts as the read strobe and data_o[16] acts as the write strobe. data_i[15:0] are inputs to the Senior processor, while the other signals are outputs.

--- Example 2.1: Uses of out-instructions, part of serial test program ---

```plaintext
set r0, 0x1234
nop
out 0x11, r0
set r0, 0xabcd
```
In Example 2.1 the register r0 is set to different values and the peripheral signals are set to these values using the out instruction. The out instruction sets data_o[15:0] to the hexadecimal value of r0, addr_o[5:0] to the hexadecimal address 0x11 and data_o[16] goes high during command execution. data_o[15:0] retains the assigned value until a new value is assigned by the out instruction. In Example 2.1, data_o[15:0] is set to the hexadecimal value 0x1234 by the first out instruction and holds that value until the next out instruction is issued.

---

**Example 2.2: Uses of in-instructions, part of the FFT program**

```plaintext
;; Read the inputs and put them in dm0
;; Real part first then the imaginary, in each sample
set ar0, 0
repeat read_input, 128 ; Points*2
in r20, 0x10
nop
st0 (ar0++), r20
read_input
```

In Example 2.2 the register r20 is set to the value of the I/O port data_i[15:0] and the data_i[16] signal goes high during command execution. 0x10 is the address for the memories synthesized on the FPGA board in this example.

The Peripheral instructions can read or write one 16-bit word each clock cycle.

---

**Figure 2.2.** The SPI protocol in use
2.3.3 Accelerator interface

The Accelerator interface [16], shown in Figure 2.1 as the off-chip accelerator interface, consists of OPA[15:0], OPB[15:0], code[19:0] and strobe. All signals are outputs from the Senior processor. OPA and OPB are the operands that are used by the off-chip accelerators. Code holds information on which accelerator that has been called, how many operands to use and an accelerator code that can be used for additional information to the accelerator.

The Accelerator interface uses special instruction commands to execute. Instead of the in/out instruction used by the Peripheral interface, the Accelerator interface uses a 32-bit word to assign different values to OPA, OPB and code.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Specification</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4:0]</td>
<td>OPB</td>
<td>-</td>
</tr>
<tr>
<td>[9:5]</td>
<td>OPA</td>
<td>-</td>
</tr>
<tr>
<td>[22:10]</td>
<td>Accelerator code</td>
<td>[12:0]</td>
</tr>
<tr>
<td>[23]</td>
<td>Core wait (not used)</td>
<td>[13]</td>
</tr>
<tr>
<td>[29:26]</td>
<td>AP (Accelerator Pointer)</td>
<td>[19:16]</td>
</tr>
<tr>
<td>[31:30]</td>
<td>Type (2'b11 when accelerator)</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2.3. Accelerator instruction code

Table 2.3 shows the function of the different bits in the accelerator word.

For example the bits [4:0] indicates which register that holds the value for OPB. Using this table an accelerator instruction can be written. The accelerator word “.dw 0xcb2b6825” was created using this table. It is a accelerator type instruction, with 2 as Accelerator pointer. It uses operands A and B, with code hexadecimal value 0xAD, OPA is stored in register r1 and OPB is stored in register r5. There exist plug-in support in the simulator to create accelerator instructions with proper names. For this project the use of manually created instruction words was preferred. This gave total control over the accelerator bit pattern in the test codes.

2.4 Interrupt controller

An interrupt controller has been added to the Senior processor. This controller will not be tested in this thesis.

2.5 DMA controller

A DMA controller has been added to the Senior processor architecture. This allows for large amounts of calculation data to be transferred to, from or between the Senior processors internal memories. There was not enough time to focus on and
extensively test this controller. Minor calculations on the DMA controllers setup time could be performed because it is used by the FFT test program.
Chapter 3

Test planning

The objective of this thesis was to evaluate the different aspects of the Senior pro-
cessors integratability. Power consumption, operations at different clock frequen-
cies, functionality of the different interfaces and other functions that are related
to integratability would be tested.

This chapter describes what parts of the Senior processor that needed to be
tested, what could be of interest to test and in what order the various functions
would be tested.

3.1 Order of tests

Before any tests on integratability could be performed, it had to be verified that
the taped-out Senior processor was functional. This could not be tested with a
single test, but through a series of tests. The SPI had to be tested first, since
this is the only way to communicate with the Senior processor. If the SPI did
not work as described in Section 2.3.1, then it would be more difficult to verify
that the Senior processor works as in simulations. How this would be a problem
is explained in Section 3.2.

If the SPI functionality can be verified, we can program the Senior proces-
sor with smaller programs to test the other interfaces and the Senior processors
functionality. If the Senior processors functionality can be verified, then we can
start testing the Senior processor against external hardware and measure different
factors that determine the Senior processors integratability.

3.2 Interfaces

All of the Senior processor interfaces, except the SPI, can be tested using simple
Senior assembler programs. The testing process can be broken down into 5 steps:

1. Write a program that runs an interface operation

2. Run the program
3. Record the necessary interface variables

4. Compare with the expected data

5. Write a new program for the next operation and GOTO 2.

The Accelerator and Peripheral interface operations take only one clock cycle to execute. These interfaces do not have to be measured over time to verify correctness.

The SPI operates according to the protocol illustrated in Figure 2.2. To verify its functionality, multiple measurements have to be recorded over time.

If the previous SPI command is not returned as described in Section 2.3.1, the Senior processor could still be programmed correctly. Using the READ_PM command the content of the PM can be read and verified. If this command does not return the expected value, then it could be that the PM has not been programmed correctly or that the READ_PM command is not functional. If this is the case the only way to know if the Senior processor has been programmed correctly, is to start the Senior core and measure the interfaces to verify that the expected instructions are issued. If this proved unsuccessful, it cannot be determined where the fault lies.

3.3 Functionality

To test the functionality of the Senior processor, a test-suite is available together with the Senior processors source code. This test-suite was used to test and verify the Senior processor on an FPGA before it was manufactured. This test-suite consists of several tests for all of the Senior processors instructions.

The plan was to replace the FPGA version of the Senior processor, with the taped-out version and run the test-suite in the same manner as in previous tests. If all the tests of the test-suite pass, then the taped-out Senior processor has at least the same functionality as the FPGA version.

3.4 Simulations

As previously stated, the Senior processor has been tested and verified using simulations before manufacturing. The simulator used for these tests, was used to test and verify larger programs before they were run on the actual Senior processor. Harmful code could then be detected before it was run on the real Senior processor, while the expected test results could be obtained for comparison.
Chapter 4

PCB development process

To be able to test the Senior processor, the planned PCB had to be developed and manufactured. All the stages in the PCB development process will be explained in this chapter, from the abstract design to the finished delivered PCB.

All components used for the Senior PCB were found on the web shop Farnell [17], except for the voltage level converter [4] that was available in the Computer Engineering Divisions science lab.

When choosing the PCB manufacturer, it was preferred that the manufacturer also could solder all components. If such a manufacturer could not be found, the PCB had to be designed so that the components could be soldered by hand.

4.1 CAD tools

The tools used to create the Senior PCB were made by Mentor Graphics. Mentor Graphics has assembled several PCB design tools for different assignments and created a central library system the programs share. In this manner a large complicated schematic can be created and later be used as a blueprint to follow in a PCB layout program.

Design View was used to create the schematic and Expedition PCB [7] was used to create the layout based on that schematic. Schematic symbols, layout cells and padstacks were created using corresponding editors and coupled together as parts using PartsDB Editor.

4.2 Abstract design

A basic design for the Senior PCB was available at the start of this thesis project, as shown in Figure 4.1. The PCB had the Senior test-chip, an SD/MMC socket, an AVR acting as MCU, a PLL and voltage regulators mounted. The signals that were of importance had also been added to the design.

This high level schematic was a good start but more detail was needed. This schematic only allowed the Senior processor to be tested in a specific manner. The
Figure 4.1. Available high-level schematic for the Senior test-board and test setup

system and its signals had to be analyzed in order to alter the Senior PCB so that new testing methods could be possible.

4.2.1 Signals of interest

The Senior test-chip has 144 pins, that are specified in Table 4.1.

Table 4.1. Senior test-chip signals

<table>
<thead>
<tr>
<th>Functionality</th>
<th># pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerator interface</td>
<td>53</td>
</tr>
<tr>
<td>Peripheral interface</td>
<td>40</td>
</tr>
<tr>
<td>Core VDD/GND</td>
<td>24</td>
</tr>
<tr>
<td>I/O VDD/GND</td>
<td>16</td>
</tr>
<tr>
<td>SPI</td>
<td>4</td>
</tr>
<tr>
<td>IRQ</td>
<td>3</td>
</tr>
<tr>
<td>clock/reset</td>
<td>2</td>
</tr>
<tr>
<td>NC</td>
<td>2</td>
</tr>
<tr>
<td>Pins used</td>
<td>142</td>
</tr>
<tr>
<td>Pins used for communication</td>
<td>100</td>
</tr>
</tbody>
</table>

Out of the 144 pins, 22 are not of interest, the GND pins and the NC pins. All other pins are of interest to be either measured and/or recorded. Generally the outputs from the Senior processor are of more interest than the inputs, since these are generated off-chip.

The VDD pins are of interest for measuring the power consumption, the core VDD more than the I/O VDD.

The clock pin is important to be able to measure. Depending on the clock
4.2 Abstract design

generator used, high frequency clock signals might be skewed which could result in timing issues. If these can be detected a large amount of time spent on debugging can be avoided.

The reset signal is an input to the Senior processor. If we can generate it properly, then there is no reason to measure it.

The Accelerator and Peripheral interfaces are important to be able to measure, as these interfaces will communicate with external hardware. If we are not able to verify what information the Senior processor is sending on these interfaces, we cannot determine if it is the external hardware or the Senior processor that is not operating properly. To be able to determine if there is an external hardware problem, the corresponding inputs to the Senior processor are necessary to measure.

As stated in Section 2.3.1, the SPI is used as the boot loader interface. Therefore the SPI is more important to measure than the other interfaces.

The interrupt signals can be regarded in the same manner as the reset signal. These signals are inputs to the Senior processor and generated by external hardware. If the signals are generated properly, there is no reason to measure them.

4.2.2 Available testing equipment

As shown in Figure 4.1, the Senior PCB is connected to an FPGA host board. The host board in mind for this project was the Xilinx Virtex-4 LX Evaluation Board. This board is fitted with two 140 pins AvBus sockets. Of these 280 pins, 183 pins can be used for I/O, easily allowing all of the Senior processor’s 100 I/O pins to be connected to the FPGA.

To measure the power consumption of the Senior processor, a KEITHLEY 2701 Ethernet Multimeter was available in the research lab at the Division of Computer Engineering. It has a resolution of 0.1 µV on a 100 mV range and a 10 nA resolution on a 20 mA range. It has the option of measuring one of 20 different voltage sources and two current sources. This allows several values to be measured without the need of extra multimeters or moving the measurement probes. With this setup one would only have to connect measurement cables once, then select which source to measure on the multimeter.

For observing signal behavior, a LeCroy waveminer LT342 was used. The LT342 is a digital oscilloscope with a sample frequency of 500 MHz. This oscilloscope works well for measuring signals with a frequency of 50 MHz or lower.

Also available was the Tektronix TLA721 Logic Analyzer, a high performance logic analyzer that has a state/timing speed from 100 MHz up to 2 GHz. This logic analyzer is ideal for measuring different signals when running the Senior processor at higher clock frequencies.

4.2.3 Design decisions

The main goal for the Senior PCB was that it would be tested together with an FPGA host board. Later it was decided to be the Xilinx Virtex-4 LX.
Evaluation Board. The FPGA board would work as the external accelerator, with different accelerators synthesized on it. But because of the short time frame for completing the Senior PCB and testing it, it was decided that two versions of the Senior PCB would be created.

The first version, called Senior light, would only contain the necessary components to test and verify the basic functionality of the Senior processor. This includes the Senior processor chip and pins for SPI, clock, reset, and power. This would also be a way to learn how to use the CAD tools available, since these programs had not been used before by the author. Senior light would be tested using a computer and a Xilinx Parallel Cable 4 with a flying wire adaptor.

The Xilinx Parallel Cable 4 has an internal voltage converter, so that it can be used to program Xilinx FPGA boards operating at different voltage levels. This cable can run in “cable 3 compatibility“ mode, which makes it easy to use with a program running on a Linux system. Any system can be used, but the Linux system gives easy access to the different serial and parallel ports on the computer. Using these features, the Parallel Cable could be connected to the SPI pins of the Senior processor. With a C implementation of the SPI protocol, communication could in theory be achieved between a computer and the Senior processor.

The second version was called Senior big and would contain all the components that were left out from Senior light. When the Senior light version was finished, it was decided to merge the two versions. The new PCB would hold all of Senior big’s components and have the option of only testing the Senior processor.

This decision spawned a new idea for the Senior PCB, that it should be able to run in different modes. Two modes were conceived, FPGA mode and Stand-alone mode. In FPGA mode, the Senior PCB would be run together with the Virtex-4 FPGA. The FPGA would act as an external hardware accelerator and test-bench for the Senior processor. In Stand-alone mode, the Senior PCB would be able to run without the FPGA board. One way to demonstrate that the board was operating on its own would be to connect an SD card with an audio file on, which would be decoded and played by the Senior processor.

4.2.4 Design completion
The high level schematic shown in Figure 4.1 holds the major components that should be included in the PCB design. To be able to run the Senior PCB in different modes and to create the PCB schematic, the high level schematic had to be supplemented with additional components.

To run the Senior PCB in Stand-alone mode, jumpers needed to be added to VDD, GND, SPI, clock and reset wires. To be able to test the Stand-alone mode using the SD socket as described in the end of Section 4.2.3, a DAC and jumpers were connected to data_o[3:0] wires, which double as the audio channel.

To communicate with the Virtex-4 FPGA, AvBus connections were added to the schematic.

The finished high level schematic for the Senior PCB can be seen in Figure 4.2. Missing in this schematic is the PLL that could be seen in Figure 4.1. Why it is missing will be explained in Section 4.3.2
With the high level design of the Senior PCB finished, work could start on the hardware schematic which was used to create the Senior PCB layout.

### 4.3.1 Available data

Access was granted to the PCB library used by the Division of Computer Engineering. Several of the components needed were already implemented in this library, these were also used as references when creating the components missing. The schematic shown in Figure 4.3 was available at the start of this project. The only components available were the Senior processor, AVR, SD card socket and voltage regulators. The resistors connected to the voltage regulators set the voltage level on the wires.

### 4.3.2 Component symbols

The components that are needed for the Senior PCB can be seen in the high level schematic in Figure 4.2. To finish the Senior PCB schematic, we had to decide which “off the shelf” components on the general IC market to use on the finished Senior PCB. For more specified information regarding the Senior PCB components, read Appendix A.

#### Available components

The Senior processor had already been added to the central library.
Figure 4.3. Initial Senior PCB Schematic
The SD socket in the schematic could not be found on the market \[17\], therefore an appropriate available socket \[10\] was chosen and added to the central library according to its data sheet.

The AVR in Figure 4.3 is a 32-bit AVR Micro-controller \[2\] with a 512 KB flash memory. This AVR also comes with a flash memory size of 256, 128 or 64 KB. The packaging of these AVRs is the same, so choosing which AVR32 to use could be based on the flash memory size. The main purpose of the AVR is to be used for boot loading the Senior processor.

The PLL that is seen in Figure 4.3 is missing from Figure 4.2. The reason for this is that the AVR32 can generate a clock signal output, with a frequency up to 240 MHz using its internal PLL \[2\]. The PLL component was replaced with the PLL of the AVR, this is also one of the main reasons for using the AVR32.

The voltage regulators were of type LM117, which is a linear voltage regulator with an adjustable voltage of 1.2 V up to 37 V. This particular regulator would stand for over a third of the total manufacturing cost of the Senior PCB, therefore it was replaced with a cheaper alternative \[15\].

The resistors were initially of package model 1206, which is a quite large component compared to other resistors available on the market \[17\]. With the recommendation from an employee at the Division of Computer Engineering, the resistor package was changed to 0603. The 0603 package is only a quarter of the size of the 1206 package and is of a better quality.

### Added components

Jumpers were added to the VDD, SPI, data_o/DAC and a pair of control wires. This feature is from the Senior light version, making it possible to feed external signals to the Senior processor instead of signals from the AVR32 or FPGA.

To be able to provide the Senior PCB with power, a 5 V power socket \[11\] was added. The socket chosen has a secondary power source. If no contact was connected to the socket, it would instead draw its power for the FPGA board. Both AvBus connections can provide a 5 V current, which was used as secondary power.

If the power socket was used with an external power supply, there was the risk of connecting the power cable with the wrong polarity. The solution to protect the components on the Senior PCB from this problem, was to insert a diode \[13\] after the power socket, connecting VDD and GND in the wrong direction. If the power was applied with the wrong polarity, the diode will short circuit the PCB. This would spare the components on the PCB, but the power supply would be at risk. Generally the voltage generators that are used in labs have built in protection against current overloads.

One issue that had to be solved regarding the components chosen, was the voltage levels of the interfaces. A 3.3 V component can read and interpret a 2.5 V component, but a 2.5 V component could be fried if it receives signals from a 3.3 V component. Therefore a voltage level converter was needed for safe 3.3 V to 2.5 V communications. The Senior processors interfaces have a voltage level of 2.5 V. The Virtex-4 FPGA board \[3\] has the option of selecting the output voltage.
level on some of its I/O pins, to either 3.x, 2.5 or 1.2 V. The non selectable I/O
pins have a voltage level of 3.x V and the AVR32 has only 3.3 V I/O interfaces. A
voltage level converter was therefore used for the signals from the AVR32 to the
Senior processor and the signals that the AVR32 had in common with the Virtex-4
FPGA.

Because of the limited amount of power on the Senior PCB available for the
audio, a 3.5 mm audio socket \[14\] was connected to the output of the DAC. A
pair of small earphones or loudspeakers with additional power supply could be
connected so the generated signal could be heard.

Not shown in Figure \[4.2\] are three LEDs with resistors that are connected to
the main power supply, the 3.3 V wire and the AVR32. These LEDs work as
indicators that power has been properly connected.

Capacitors were added adjacent to the voltage regulators, AVR32 and the
Senior processor. This would maintain a stable power supply to these components.
The size of the capacitors needed by the AVR32 was unknown. Capacitors of
package module 0603 were added as a precautionary measure, since this module
is available in a wide range of sizes.

A SMA contact was added, providing the option of testing the Senior processor
using an external clock signal, not generated by the FPGA board or AVR32. A
reset button \[12\] and jumpers were added to the reset signal, allowing the Senior
processor to receive a reset signal from the FPGA board or the reset button.

Schematic symbols were created for all components not already available in the
central library.

4.3.3 Finished schematic

The finished schematic for the Senior PCB can be seen in Figures \[4.4\] and \[4.5\].
This was the result from several attempts to connect all wires in a manner that
would result in an easily developed PCB, with as short connections as possible
with low signal and plane crossing.

When working with a program like Design View, it is easy to connect all the
signals between each component in a system. As seen in Figure \[4.4\], very few con-
nections are drawn between components using lines. Instead internal connectors
are used, which makes the schematic easy to survey.

All components in the schematic are only representations of the physical com-
ponents. As seen on the Senior processor in the middle of Figure \[4.4\], all interfaces
are neatly grouped and easy to survey. This is preferable when you want to be sure
that all wires are properly connected. The drawback with this is that if the phys-
ical reality is unknown or uncertain, the layout corresponding to the schematic
could be difficult to implement. An example of that is the AvBus connections in
Figure \[4.5\].

In figure \[4.6\] the top of the AvBus 2 symbol is shown with its connections. As
it can be seen, the signals are not connected in consecutive order. This is due to
the pin-list of the Senior processor, where the signals are not in consecutive order.
Figure 4.4. Finished Senior PCB Schematic, page 1
Figure 4.5. Finished Senior PCB Schematic, page 2
For parallel signals it is important that all the wires are of equal length, to remove any difference in transmission delay. When transmission frequency increases, small delay differences among signals can become potential problems.

Because of the inexperience in using the available CAD tools and the time available, it was decided instead to keep the wire lengths as short as possible. Minimizing the wire lengths would not solve the difference in transmission delays, in fact the shorter the wires are the larger the relative difference between transmission delays becomes. But negative impacts such as wire impedance and signal distortion becomes smaller with shorter wires. [9]

To be able to use the Tektronix TLA721 Logic Analyzer described in Section 4.2.2, probe connections were added. A schematic symbol of the probe can be seen in Figure 4.7. These were placed between the Senior processor and the AvBus connections, to minimize the potential disturbance these connections would cause. The physical size of these probes, limited the number of probes to four which in turn limited the number signals that could be measured. Random signals from different interfaces were connected to the probes, to partially measure how the different parts of the different interfaces would behave. See Figure 4.5 for probe connections.
4.4 Layout

When the schematic was finished for the Senior PCB, it was used as a blueprint to create the PCB layout.

When the Senior PCB was designed it was decided to construct a 4-layer PCB. The manufacturing cost depends on how many layer there are in the design and there are fewer manufacturers who can manufacture PCBs with more than four layers. If only two layers had been used, the Senior PCB would have to be of a larger size to be able to route all the wires. Using more than four layers for the PCB layout would have resulted in a smaller PCB but the manufacturing cost would be much higher. The number of layers for the PCB is not of importance when constructing the PCB schematic.

In this section, the task of constructing parts is covered. Parts in the central library system consist of a schematic symbol and a layout cell with pads assigned. A complete schematic can be created using only symbols and a complete layout only using cells but there would not be any connection between the two. Parts are used to link the schematic and layout, so changes in one would be updated in the other. Parts are usually constructed before they are used in the schematic, but it was more suitable to process that task in this section. The statements in section 4.4.2 are based on experience and observations made on previous work by other developers.

4.4.1 Available data

Almost all components that were going to be used on the Senior PCB, had already been implemented together with the initial schematic or available in the Computer Engineering Divisions PCB library.
The SD/MMC, reset button, voltage regulators and 3.5 mm audio socket had symbols already available, but corresponding cells with pads had to be created. The Senior processor had a cell connected to it, but the size of the pads and the distances between them were not correct so they needed to be altered.

The components that were missing from the available libraries were the DAC[10] and the large pin-lists used for JTAG/UART and clock signals.

### 4.4.2 Constructing pads, cells and parts

When creating the pads and cells, data sheets containing pin-list/layout and package specifications are needed. Proper pads or cells cannot be created without them, if the physical components are not available.

Some data sheets come with a recommended pad description. Such a description can include how the component should be mounted on the PCB, what kind of pads to use and what measurements the pads should have.

If a pad description is not available, it is a matter of calculating a reasonable value for the pads dimensions and distances between them. For through mounted components such as the SMA contact, it is enough if the through holes are about 0.48 mm larger than the component leads. If for example a lead has a diameter of 1 mm, the through hole should have 1.48 mm diameter. With surface mounted components, it is easy to say the larger the pad size the better. The room for error when soldering increases when the size of the pads decreases.

The Senior processor has 144 pins, 36 pins on each side of the chip. These pins have a very small contact area and are closely spaced together. If the pads are too small and the chip is not aligned properly, a large amount of pins will have a bad or no connection to their pads. If the pads are large but do not have enough space between them, they could be soldered together. There is a trade-off between the size of the pads and the distances between them, in this project the distance between two pads was usually one pad of the same size.

One thing that is not specified in many data sheets is which unit that is used for measurements. Many data sheets measure in millimeters and others measure in inches or mils. A mil is a thousandth of an inch. Depending on the measurement values, one can determine which metric that is used.

A warning must be made, regarding reading data sheets for pads and cell construction. Some data sheets use one metric for measurements and include the other as reference. The reference values should absolutely not be used as values when constructing the cells and pads. These values are only references and have been rounded. The increased size of a through hole is a good example. The recommended value in millimeter is 0.48, but the recommended value in inches is 0.019. When converting between inches and millimeter, 0.019 inches becomes 0.4826 mm and 0.48 mm becomes 0.01889764 inches rounded to eighth decimals. With simpler and larger components, using the reference values could work without major impact. But for smaller and more complex components such as the Senior processor, using the reference values could make the PCB and component incompatible.
4.4.3 Placement

With no requirements on minimizing the size of the Senior PCB, it was given the measurements 100x160 mm. This size would allow all components to fit and to be grouped according to their use. How the components were finally placed can be seen in Figure 4.8. The measurements used for the PCB is based on the size of PCBs previously developed by the Computer Engineering Division.

When placing the components on the Senior PCB, the main focus was minimizing the distance between the Senior processor and the AvBus connections. To be able to attach the Senior PCB to the Virtex-4 FPGA board, the AvBus connections were placed under the PCB at one end, seen to the bottom of Figure 4.8. This would allow the Senior PCB to be connected to and not cover too much of the FPGA board. The Senior processor was first placed in the middle, parallel with the AvBus contacts. Later it was turned at a 45 degree angle, the reason for this will be explained in Section 4.4.4.

The voltage level converter and jumper headers for the SPI, DAC, clock, reset and control signals were placed close to the Senior processor. This was done to minimize transmission delay and signal distortion due to reflections.

The other components were grouped based on their functionality. The power related components, power socket, diode and voltage regulators were grouped on one side of the PCB, seen to the left of Figure 4.8. The audio related components, SD/MMC socket, 3.5 mm audio socket and DAC were placed on the opposite side. The AVR32 was placed between all components, with UART and JTAG headers placed on the opposite side of the AvBus connections.

Since a large amount of capacitors were connected to both the Senior processor and the AVR32, half of the capacitors were placed under the PCB, opposite of the capacitors on the top. Capacitors on both sides could share the same VDD and GND vias.

4.4.4 Routing

The finished routing took several attempts to complete, since each time the routing was finished, better solutions were discovered and changes to connections were made. This is the explanation for the unordered signals on the AvBus, mentioned in Section 4.3.3. The connections between the Senior processor and the AvBus contacts was the biggest challenge with the routing.

It was decided early in the development process, that if the components had to be soldered by hand, the most important signals of the Senior processor would be connected to one of the AvBus contacts and the others to the second AvBus. The Accelerator interface, SPI, clock, reset, interrupt and control signals were connected to AvBus 1 and the Peripheral interface to AvBus 2. If there would be problems soldering or short time to deadline, then only AvBus 1 would be needed to be soldered. There are two reasons for this.

First off, the corresponding connections on the Virtex-4 FPGA Board are located very close to each other, 600 mil (15,24 mm) according to the Virtex-4 schematic. With such large components placed close together, the second contact would be difficult to solder.
Figure 4.8. Finished component placement on Senior PCB
Secondly, since the AvBus contacts have a large amount of pins, where every fourth pin is either a GND or VDD pin, it is very important to solder these components properly and avoid soldering pads together.

On the first attempt to route the wires, all signals were placed in consecutive order and grouped according to interface in the schematic. The layout result from this was that every signal, between the Senior processor and both AvBus contacts were crossing each other. Almost all wires crossed each others path, independently of how the chip was placed.

On the second attempt, the wires stayed grouped according to interface but were rearranged. This attempt gave a better result, with fewer wires crossing each others path. This attempt showed clearly that all wires connected to AvBus 1 were allocated to one side of the Senior processor and all wires on the other side were connected to AvBus 2.

For the third attempt, the signals were connected to the AvBus contacts in the same order that they were located on the Senior processor. This attempt proved successful, with no wires crossing each others path. With the chip placed in parallel with the AvBus contacts, the different distances between the pins was large, resulting in a wide difference in wire lengths. By turning the Senior processor 45 degrees, the differences was minimized and with that the wire lengths.

The other wires on the Senior PCB, were very straightforward to route. The wires were routed as short as possible, with minimum need to switch between different planes in order to get past other wires. The VDD wires were made several times larger than ordinary signal wires, so that power distribution would be even across the PCB. The final result can be seen in Figure 4.9.

4.4.5 Plane fill
To ensure a good power supply from the Virtex-4 FPGA board to all components, larger plane shapes were made for power distribution on both sides of the Senior PCB. Smaller ground planes were also created for the AvBus contacts, DAC and capacitors surrounding the Senior processor. Layer 2 was chosen to be the main ground plane, so that ground wires could be excluded.

4.5 Manufacturing
The Senior PCB layout was completed. All signals were properly connected and no wires were missing. The CAD tools could not find any design rule violation, so a Gerber file was generated and sent for manufacturing.

The manufacturing process started before the final version of the Senior PCB was finished, with first finding and contacting a PCB manufacturer. The manufacturer chosen could manufacture the Senior PCB and solder the necessary components.

Provided the preliminary specifications for the Senior PCB, a bid on the manufacturing cost was received. From the bid received, an order was placed for five Senior PCBs. Because of the size of the PCB, a minimum of six PCBs were
Figure 4.9. Finished wire routing on Senior PCB
manufactured. The sixth extra PCB would not have any components mounted on.

Since contact had been established with the manufacturer before the final Senior PCB layout had been finished, the manufacturer received several versions of the Senior PCB to estimate the manufacturing cost. In this dialog with the manufacturer, confusion arose over time regarding which files that belonged to the latest version. This resulted in that the manufacturer had ordered the wrong components, based on an old component list. Fortunately it was the latest version of the PCB layout that had been sent for manufacturing. Due to the confusion and misunderstanding between the parties, in issues regarding deliveries of components and finished product, the estimated delivery time of six work days ended up being two and a half weeks.

The delivered results can be seen in Figures 4.10 and 4.11.

**Figure 4.10.** Delivered Senior PCB, top side
Figure 4.11. Delivered Senior PCB, bottom side
Chapter 5

Testing process

The Senior PCB was finished, the manufactured test-boards had been delivered and the process of testing and verifying the Senior processor could begin.

The proper order would be to first test the delivered test-board to verify that it had been manufactured without flaws, that the components had been soldered properly, that no short circuits had been introduced and that the voltage regulators were supplying their intended voltage levels. With the verified test-board, the Senior processor could be tested in the order stated in Section 3.1. But due to the short time frame, the Senior processor was tested first.

Either way the Senior processor had to be tested by itself, before being used together with the other IC on the Senior test-board. Without testing the Senior processor by itself and if a fault occurred, the source of the fault would be difficult to determine.

The finished Senior test-boards were delayed in delivery, but the extra empty Senior PCB was delivered a week earlier. Because of the design choice to add jumper headers to power supply, SPI, clock and reset signals, tests on the Senior processor could still be conducted.

It is to be noted that the initial tests on the Senior processor could still have been conducted without the addition of jumpers. By soldering wires either to the leads of the Senior processor or preferably to the pads for the AvBus or AVR32 and connecting those wires to the Xilinx Parallel Cable 4. But this approach would not be ideal and signal disturbances could be introduced.

5.1 Equipment

The equipment that was used for performing the different tests on the PCB and the Senior processor are covered in this section. The following pieces of equipment have been covered in Section 4.2.2:

- Xilinx Virtex-4 LX Evaluation Board
- KEITHLEY 2701 Ethernet Multimeter
- LeCroy waverunner LT342
- Tektronix TLA721 Logic Analyzer.

When the first tests were conducted on the Senior processor, a Simpson 420 Function Generator was used to generate the system clock. This is a simple function generator, capable of amplitude and DC offset adjustments, which is necessary to create a proper waveform. It is capable of generating a square wave formed signal, with a theoretical frequency range from zero up to 1.1 MHz. In reality the signal was far from ideal at higher frequencies, but this signal was good enough to test the Senior processor. The Virtex-4 FPGA was later used instead for this assignment.

The Powerbox PB3100 65W Triple Output Power Supply was used in these tests to generate the 1.2 V used by the Senior core and the 2.5 V used for the I/O signals.

A CHY 21C Multimeter was used to conduct smaller preliminary measurements on voltage and current levels. This was also used to confirm that the PCB wires had been connected properly and that there were no short circuits in the PCB.

Before the Virtex-4 FPGA was programmed to act as test bench or interface to the Senior processor, the Xilinx Parallel Cable 4 was used in the manner discussed in Section 4.2.3 to communicate with the Senior processor. This approach was also used when measuring the Senior processors consumption levels at higher frequencies.

5.2 Senior PCB tests

The different preparations and tests done with the Senior PCB are covered in this section. The results and alterations made because of these results will be disclosed in Section 6.1.

5.2.1 Unmounted PCB

The extra manufactured Senior PCB arrived a week earlier than the mounted PCBs. Not much could be done to test the empty Senior PCB in its current state and no new defects were visible.

To conduct the initial tests on the Senior processor, the processor and headers necessary where soldered to the empty PCB. Wires were soldered to the voltage level converter pads, connecting the Senior processor to the signal headers, shown in Figure 5.1. The CHY 21C Multimeter was used to verify that the soldering had been done properly, by measuring on the leads of the Senior processor and the signal headers.

5.2.2 Mounted PCB

The mounted PCBs arrived a week later, they were screened for defects and components not mounted by the manufacturer were soldered by hand. The Senior
5.3 Senior processor tests

5.3.1 SPI verification

As planned in Section 3.1, the SPI was tested first. The complemented extra PCB from Section 5.2.1 was used, together with the needed equipment to conduct these tests. The test setup for the Senior PCB is shown in Figure 5.2. The details of this test and how it was conducted can be read in the included test report, Appendix B.

5.3.2 Measuring power consumption

The power consumption depending on the clock frequency was measured using the FFT algorithm mentioned in Chapter 2 as the benchmark program.

The tested and proven FFT algorithm, along with corresponding input data was available in the provided source code. Corresponding output data was attained using simulations. For this test the finished Senior test-board was used, together with the Virtex-4 FPGA board and other needed equipment. The details of this test and how it was conducted can be read in the included test report, Appendix E.
5.3.3 Functionality tests

To test the Senior processors functionality, the test-suite mentioned in Section 3.3 was used. The available FPGA program contained all components needed including the FPGA version of the Senior processor.

To be able to use this FPGA program together with the Senior test-board, the FPGA version of the Senior processor was removed from the code. It was replaced with the necessary in and out ports in the main Verilog file and the UCF file was updated with the pins that were going to be used.

To conduct the test, the Senior test-board, the Xilinx platform Cable USB and Serial cable where connected to the Virtex-4 FPGA board and the code alterations had been completed. This test setup is shown in Figure 5.3.

5.4 Tests with Accelerator 1 : SIMD processor

The SIMD processor was the first accelerator to be tested together with the Senior processor. The main function of the SIMD processor is to perform MIMO operations, which is an important function in base-band stations.

The SIMD processors MIMO operations, were tested in a similar manner as when testing the Senior processors functionality. The available FPGA program contains the SIMD processor. With the changes done in Section 5.3.3 the SIMD processor could easily be tested using the available C programs. The details of this test and how it was conducted can be read in the included test report, Appendix F.
5.5 Tests with Accelerator 2 : Reciprocals

One part of the MIMO accelerator test that is of interest is the setup time for the DMA transactions. This was measured by introducing special accelerator commands into the Senior program, that were captured by a module on the FPGA that calculated the setup times.

The details of this test and how it was conducted can be read in the included test report, Appendix G.

Figure 5.3. Test setup for Senior functional tests

5.5 Tests with Accelerator 2 : Reciprocals

The second accelerator that was tested together with the Senior processor was a 16-bit reciprocal accelerator. No available source code could be found for this kind of accelerator, so one was designed from scratch. This accelerator was proven to work by itself and simulated using ModelSim, confirming it was working as designed.

This reciprocal accelerator, seen in Figure 5.4, consists of 17 reciprocal stages connected one after another, creating a pipe-line architecture. It has a 64 words FIFO memory, where all calculated values are stored until they are retrieved. When the memory is full, no new values can be stored until the old values are retrieved. When a value has been calculated, an interrupt signal goes high so that the Senior processor will be notified that a computation has been completed. This interrupt signal is optional, meaning that the Senior processor does not need to use this signal to be able to use the accelerator.
Figure 5.4. Schematic for the reciprocal accelerator
Chapter 6

Results

In this chapter the manufactured PCB and the results from the different tests will be discussed. More detailed results can be found in the test appendices.

6.1 Senior PCB review

It was immediately noticed that the silkscreen, with signal markings and additional warning texts, was missing. Signal markings in the silkscreen were covering wires on the top level. The manufacturer chose to exclude the silkscreen, instead of correcting the faults because of the short time to dead-line. The absence of the silkscreen did not affect the functionality of the PCB.

During the first tests on the Senior processor using the unmounted PCB, the Tektronix Logic Analyzer was used to observe parts of the peripheral interface. When the Tektronix probe was connected, the measurement equipment did not register any activity on the Senior processor. It was found that the Tektronix probe short circuited the Senior PCB, the explanation was found in a schematic of a different system that used these probes.

The schematic showed that every third pad on the probe was connected to GND, this was not the case with the probes on the Senior PCB. When the probes were added to the Senior PCB design, there were no specifications available on how the probe should be connected. The probes available in the PCB central library gave no detail on how they should be connected either.

Because of this the Tektronix Logic Analyzer cannot be used together with the Senior PCB. The probes where originally an extra feature added in the later stages of the PCB development, so the loss of this measurement option would not be a problem when conducting the other tests.

6.1.1 Unmounted PCB

During the soldering process of the necessary components for the initial Senior tests, described in Section 5.2.1, it was found that the Senior processor and the pads on the Senior PCB did not match. The pads were to far apart, when the
chip was placed perfectly in the middle of the pads, to corner pins would barely touch the corner pads.

When the available cells were reviewed in the development process, it was found that the Senior processor cell had the wrong measurements. The pads were too small and placed too close to the center. The pads were replaced and the proper measurements were inserted, but the reference values were used instead of the real values from the data sheet [6]. The warning in Section [4.4.2] was derived from this mistake.

Figure 6.1. Close-up on leads of Senior processor, slight difference in angles can be seen between the different leads

When a new cell is created using the Cell editor, it can be chosen what kind of metrics to use during cell editing. Millimeters or mil can be chosen, where mil is the default metric. It could not be found if the metrics used could be changed after the cell had been created. The metrics for the available Senior cell was mil and so was the reference values. These values were not far from the correct measurement, but with 36 pins in a row, the difference added up and the last pad did not align with the last pin.

This was solved by gently bending the leads on the Senior processor, as seen in Figure [6.1] so that all pads and leads were aligned.

When soldering the wire connections replacing the voltage level converter, two SPI wires were soldered to the wrong pads, making the Senior processor appear dead when the test was conducted. Those wires were re-soldered and all signals were properly connected.

While removing one of these wires, the actual pad on the PCB was torn off, destroying the PCB wire connection. This broken connection can be seen in Figure [6.2] Utmost care must be taken when wires are to be re-soldered, otherwise the PCB could be permanently damaged.

6.1.2 Mounted PCB

Some faults were discovered with the mounted Senior PCB. The power socket was not the intended type, it was only a quarter of the size it should be, see Figure [6.3]
Figure 6.2. Close-up on the voltage level converters solder pads, with the broken pad encircled

for comparison. This was the author’s fault, the wrong component was chosen to be ordered from Farnell.

Figure 6.3. Comparison between the intended power socket (to the right) and the power socket used (to the left)

The 3.5 mm audio socket mounted on the PCB, is not same socket as was initially chosen. The situation was known in advance from the manufacturer. The audio socket that had been delivered to them had five pins instead of the intended three. It was shown to be the socket manufacturer’s fault, since the product ID corresponded with the one that was originally ordered. The component could still be used, the extra pins could be removed and the component would still fit the PCB.

One major fault was found with the usage of the AVR32, after reading the documentation [2] in more detail. An external crystal is needed to generate the 240 MHz clock signal mentioned in Section 1.3.2 and that crystal had been left out of the design. After checking through the design thoroughly, it was found that
the pins chosen for the UART connection can also be used to connect the missing crystal. The headers added for the UART can be used for mounting and soldering the crystal, see Figure 6.4. The AVR32 can generate a slow clock on its own, but if higher frequencies are required the UART has to be replaced with the required crystal.

Figure 6.4. UART through holes, which can be used to connect oscillator crystal

6.2 Senior processor results

6.2.1 SPI evaluation

The SPI test was positive, since the SPI communication was fully functional and all available commands could be used. It was noticed that the MISO signal, returns the result from the previous command one clock cycle earlier than stated in Section 2.3.1. After researching the RTL code it was found that the noticed behavior was correct and the documentation was incomplete. The data being sent one cycle early is not a problem that affects the functionality. These results can be read in the test rapport in Appendix B.

6.2.2 Power consumption

Since the Senior processor had no estimated values on its power consumption, all that can be said about the measurement values are that they are consistent. There is a clear relationship, between the power consumption during calculations and when the Senior processor is idle, independantly of the clock frequency used. The consumption increases evenly when the clock frequency increased and no unexpected values were observed.

The test was mainly conducted using one Senior test-board, but a second test-board was used to test higher frequencies and it showed the same pattern as the first test-board.

This test also confirmed the maximum operational clock frequency of the Senior processor. Previous tests showed that the Senior processor could operate at 300
6.2 Senior processor results

MHz at the intended 1.2 V. But after these tests, the new recommended maximum frequency is 250 MHz, higher frequencies could be harmful for the Senior processor.

These results can be read in the test rapport in Appendix [E].

6.2.3 Functionality

The simulated Senior processor had passed the test-suite using a clock frequency of 25 MHz, the real processor should at least be able to pass the test-suite using the same frequency to be considered successfully manufactured.

**run_fpga_design.c and FPGA program**

The available “run_fpga_design.c” program was used to communicate with the FPGA board and the Senior processor. This program can read Senior hex-files and programs the Senior processor accordingly. A similar version was written to talk to the Senior processor using the parallel port.

These were the first tests where the Senior test-board was connected together with the Virtex-4 FPGA board and the first time the available test programs and codes were used.

At first the communication between the Senior processor and the FPGA would not work. The test program verified that the proper commands were sent to the senior processor but the processor did not execute. The test program compares the data sent to the processor with the returned values. According to the program the Senior processor was programmed as intended. After adding additional test code, it was found that the processor was receiving the data sent, but the data was incorrect. The C command “strtol” was used to read the hex files that contained the Senior programs and this functions saturated Senior hex codes larger than 0x7ffffff for all 32-bit systems. All instructions with hex codes larger than 0x7fffffff, such as jumps and accelerator instructions were broken. The problem was solved by replacing “strtol” with “strtoul”.

This fault could be the reason why a previous thesis project had trouble getting the FPGA version of the Senior processor to operate as in simulations. When working on a 64-bit system, the “strtol” function will read all 32-bit values as positive 64-bit values and therefore this function would not be a problem.

The results from the first test were slightly different from the simulation values. The problem was thought to be a delay issues in the FPGA. A 2-stage pipeline was introduced in the FPGA, assuring that the inputs would be synchronous. This resolved the delay issue, but the test could now not perform correct calculations.

After several tests, it was found that residual data in the data memory interfered with the calculations. After a clear_data_memory function was inserted at the start of the C program, the test could be correctly performed.

**Test-suite**

After the communication and execution problems had been sorted out in the run_fpga_design.c and FPGA programs, the actual functionality test could be conducted.
Since the Senior FPGA test had been performed in 25 MHz, that frequency was chosen as the start frequency.

At 25 MHz all tests passed except the “rotate.asm” program. “rotate.asm” consists of several rotate operations, repeated after each other and in different orders. All individual rotate calls passed the test.

By commenting parts of the rotate test, the entire test passed. Several test showed that it did not matter what part of the code was commented, as long as some code was commented the test passed.

The reason for this lies in the data memories in the FPGA. There are two 1024 words data memories, that are written to using the out instruction. The rotate test generates 2224 words of output.

The simulations are run with larger simulated memories, than the memories implemented on the FPGA. The previous data will be written over in the FPGA memories if too many out instructions are issued and that is why the full rotate test failed.

With a modified rotate, only producing the permitted amount of out instructions, the entire test-suite passes.

These tests also showed that with frequencies higher than 100 MHz, the communication between the Senior test-board and the Virtex-4 FPGA board experienced timing issues. The reason for this issue is not completely known. It could be a fault with the routing in the Senior PCB, or the internal architecture of the Senior processor. But the reason is probably due to how the Verilog code has been realized in the FPGA. With the current setup, the maximum operational frequency is 100 MHz.

### 6.3 Accelerator 1 : SIMD processor

The results from the MIMO tests using the FPGA version of the Senior processor, were comparable to the results from using the Senior test-board.

Due to the known timing issues in the Senior to FPGA connections, if the clock signal was inverted from the FPGA board to the Senior processor, the Senior processor can receive the proper output from the SIMD.

In this test, the Senior processor first calls the SIMD processor on the FPGA board. The SIMD processor calculates the results for the MIMO operations, returns the results to the Senior processor which send the results back to the FPGA board. The SIMD processor is probably working correctly, but due to this back and forth communication the results can be altered due to the delays in the system.

These results can be read in detail in the test rapport in Appendix F.

<table>
<thead>
<tr>
<th>DMA task</th>
<th>clock cycles</th>
<th>lines of code (Senior)</th>
<th>data sent/retrieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2081</td>
<td>77</td>
<td>128</td>
</tr>
<tr>
<td>2</td>
<td>286</td>
<td>32</td>
<td>118</td>
</tr>
<tr>
<td>3</td>
<td>184</td>
<td>34</td>
<td>128</td>
</tr>
</tbody>
</table>

Table 6.1. DMA measurements
The results from the DMA tests are shown in Table 6.1. Task 1 (DMA1) sends the calculation data to the SIMD processor, task 2 (DMA2) sends the MIMO program to the SIMD and task 3 (DMA3) retrieves the results to the Senior processor. Ten lines of code in each task consist of two wait loops, one for the DMA channel to finish configuration and the other for sending/receiving data. Additional tests have shown that it is during the transmission loops that the large amount of clock cycles are measured. Why the amount of clock cycles is so high compared to the amount of data needed to be sent is unknown to the author.

--- Example 6.1: Sending data using out instructions ---

```
set ar0, 0
repeat loop, 5
    set r0, (ar0++)
    set r1, (ar0++)
    out 0x14, r0
    out 0x14, r1
loop
```

In theory, the time it takes to send or receive data using DMA is the setup time plus the transmission time. After the DMA has been properly set up, one 16-bit word of data should be able to be sent or received every clock cycle. Using regular out instructions, it would take a minimum of 2 clock cycles to send a 16-bit word of data. Example 6.1 shows how such a program could look like. Using this as a reference, it can be illustrated how much time can be saved by using DMA transactions. This illustration can be shown in Figure 6.5. According to this graph, the DMA tasks are useful with larger amounts of data. With smaller transactions, it is preferable to use I/O-instructions.

### 6.4 Accelerator 2: Reciprocals

As stated in Section 5.5, the reciprocal accelerator has been simulated and proven to work as intended. When the accelerator was tested together with the Senior processor, it could receive a value and calculate the corresponding reciprocal which could be read by the Senior processor. Further tests showed that when several values were sent after another, only the reciprocal for the last value could be read. But if the values were sent with an interval of 13 to 16 clock cycle distance, two values could be read by the Senior processor.

The conclusion is that the reciprocal accelerator can perform correct calculations by request from the Senior processor, when it is provided with only one value, where the results are read before it receives a new one. The pipeline stages together with the FIFO register does not work as initially intended. This fault
Figure 6.5. Graph, number of clock cycles depending on number of data word. The value in the parenthesis is the setup time needed.

could be from when the accelerator is synthesized for the FPGA, but it is probably because of how the design of the accelerator has been implemented. What is known is that the fault does not lie with the Senior processor, since it has been proven to work in earlier tests.
Chapter 7

Conclusion

7.1 Senior PCB

The Senior PCB created in this project was able to do the tasks it was designed for. The Senior processor could be mounted on the board and tested together with the Virtex-4 FPGA board. The additional headers allowed for different testing options which could be useful if something would be wrong with the PCB.

Of course there are some parts of the PCB that could have been done differently. The placement of the components could have been closer, which would allow for a smaller PCB size. Using different layers when routing the signals, could have allowed for a tighter routing and shorter wire length. Not leaving out components such as the crystal mention in Section 6.1 would also improve the overall quality of the Senior PCB.

But the results are still satisfying when taking in consideration that this is the first PCB ever created by the author, with no prior experience using the available CAD tools.

This thesis project has demonstrated that extra added functionality, alternative solutions and back up plans could be of great use if something should go missing of be forgotten. The lessons learned in this project, will be of great use in the next.

7.2 Senior processor

This thesis project has shown that the manufactured Senior processor works as it is intended.

- Basic measurements on the power consumption levels has been performed
- The functionality of the Senior processor has been proved
- Tests with external hardware has been conducted and the tests show that Senior can communicate and use such hardware
• The Senior processor has been proved to operate at 250 MHz, 50 MHz higher than what it has been synthesized for.

Tests have also shown that the processor can operate at 300 MHz, but this is most likely harmful for the processor.

The integratability of the Senior processor cannot be rated with a numerical value, since there are no graded scales to compare with the results from the test conducted. But what can be said is that the tests do not show any unexpected or unwanted results and that the manufactured Senior processor works as intended.
Chapter 8

Future work

There was not enough time to explore and test all the features of the Senior test-board, some parts could have be tested further and others have not been tested at all.

8.1 What is left

Not all components on the Senior test-board have been tested during this thesis project. The major components that have been left out are the AVR32 and DAC. These component are part of the Stand-alone mode described in Section 4.2.3. Since testing the Senior processor together with the FPGA host board went so well, there was no need to get the Stand-alone mode operational in order to verify the Senior processor.

If more time would have been available, different hardware accelerators would have been of interest to test and test the Senior processor together with several accelerators at the same time.

But these issues could have already been looked at, since the Senior test-board is already being used by others for testing the Senior processor. One of the Senior test-boards has traveled to China and is currently being tested by the creator of the unfinished test-board this project is based on.

8.2 Possible applications

In its current state, the Senior test-board would be testable together with proper hardware accelerators. But the accelerators would have to be connected to the same kind of AvBus connections that are mounted on the Virtex-4 FPGA board to be able to communicate with the Senior processor.

Since the Senior processor has been proven to work, it may be of interest to include it in future PCBs with other hardware. To use the Senior processor in larger systems and have it performing larger tasks.
Bibliography


[13] ON SEMICONDUCTOR. 1500 watt mosorb zener transient voltage suppressors - 1n5908g.
[14] PRO SIGNAL. 3.5 mm audio socket - mj-352w-0.  


[16] Markus Svensson and Thomas Österholm. Optimization and verification of a  
   integrated dsp, 5 December 2008. ISRN LiTH-ISY-EX–08/4215-SE.


   project.eu/.

Appendix A

Senior PCB component specifications

The following components were originally decided and ordered to be mounted on the Senior PCB.

The part numbers are the ones taken from Farnell as reference for the manufacturer.

The layout labels corresponds to PCB shown in figure 4.8.
<table>
<thead>
<tr>
<th>Component</th>
<th>Layout label</th>
<th>Spec.</th>
<th>Part nr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVR32</td>
<td>U5</td>
<td>256 KB RAM</td>
<td>1715453</td>
</tr>
<tr>
<td>Lm317h</td>
<td>U17,U18,U19</td>
<td></td>
<td>1261208</td>
</tr>
<tr>
<td>DAC</td>
<td>U6</td>
<td></td>
<td>1824917</td>
</tr>
<tr>
<td>LED</td>
<td>U8</td>
<td>Yellow</td>
<td>1226419</td>
</tr>
<tr>
<td></td>
<td>U9</td>
<td>Green</td>
<td>1226397</td>
</tr>
<tr>
<td></td>
<td>U7</td>
<td>Red</td>
<td>1226390</td>
</tr>
<tr>
<td>Resistance</td>
<td>R9</td>
<td>66.5</td>
<td>1170670</td>
</tr>
<tr>
<td>0603</td>
<td>R8</td>
<td>150</td>
<td>1469760</td>
</tr>
<tr>
<td></td>
<td>R1,R2,R3,R4</td>
<td>243</td>
<td>1570734</td>
</tr>
<tr>
<td></td>
<td>R5</td>
<td>402</td>
<td>1180248</td>
</tr>
<tr>
<td></td>
<td>R6,R7</td>
<td>470</td>
<td>9332146</td>
</tr>
<tr>
<td>Capacitors</td>
<td>C3, C2</td>
<td>2.2n</td>
<td>1457726</td>
</tr>
<tr>
<td>0603</td>
<td>C50-C69</td>
<td>10n</td>
<td>1759022</td>
</tr>
<tr>
<td></td>
<td>C9,C12,C16-C35</td>
<td>100n</td>
<td>1833840</td>
</tr>
<tr>
<td></td>
<td>C1,C11,C13,C10,C42,C44,C46,C48</td>
<td>2.2u</td>
<td>1845734</td>
</tr>
<tr>
<td></td>
<td>C43,C45,C47,C49</td>
<td>22u</td>
<td>1838742</td>
</tr>
<tr>
<td>SD</td>
<td>U13</td>
<td></td>
<td>1558174</td>
</tr>
<tr>
<td>Power Socket</td>
<td>J4</td>
<td></td>
<td>1368647</td>
</tr>
<tr>
<td>Diode</td>
<td>D1</td>
<td></td>
<td>1458994</td>
</tr>
<tr>
<td>Voltage level</td>
<td>IC1</td>
<td>adg3247bru</td>
<td></td>
</tr>
<tr>
<td>Converter</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMA</td>
<td>J8</td>
<td></td>
<td>1248990</td>
</tr>
<tr>
<td>Audio socket</td>
<td>U16</td>
<td>3.5 mm</td>
<td>1267400</td>
</tr>
<tr>
<td>AvBus</td>
<td>J6,J7</td>
<td></td>
<td>1098361</td>
</tr>
<tr>
<td>Reset button</td>
<td>SW1</td>
<td></td>
<td>143637</td>
</tr>
<tr>
<td>Headers</td>
<td>P1,P2,P3,P4,P5,P6,P7,P30,P31</td>
<td>1x2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P21,P22,P23,P24,P32</td>
<td>1x3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>U20</td>
<td>2x3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>U10,U11</td>
<td>2x5</td>
<td></td>
</tr>
</tbody>
</table>
Appendix B

Testing of Senior processor
First evaluation

This test was to verify that the Senior processor was operational. This was done by first verify that the SPI was functional and then programming the Senior processor with smaller programs.

B.1 Test setup

The Senior processor and headers for SPI, power supply, clock and test signals were mounted on the board.

To communicate with the Senior processor that has a 2.5 I/O voltage level, a Xilinx parallel cable 4 was used. Its internal voltage converter allowed it to communicate with a PC and together with a “flying wire adaptor” with the SPI pins, where the connections were as followed in Table B.1.

\[
\begin{array}{c|c}
\text{Xilinx parallel 4 cable} & \text{Senior SPI} \\
\hline
\text{TCK} & \text{SCLK\_I} \\
\text{TMS} & \text{SS\_BAR\_I} \\
\text{TDI} & \text{MOSI\_I} \\
\text{TDO} & \text{MISO\_O} \\
\text{VREF} & 2.5 \text{ V} \\
\text{GND} & \text{GND} \\
\text{INIT} & \_ \\
\end{array}
\]

The clock frequency was generated with a Simpson 420 Function Generator, which could only generate low frequency signals up to 1 MHz at best.
For higher frequencies the Virtex-4 board was used, together with an “AVNET Audio/Visual card” to get additional output pins to connect to the test board. The FPGA was programmed with a clock-divider, that forwarded the FPGA clock signal to the test board. This program limited the clock frequency to 50 MHz, but allowed the frequency to be changed during testing using the switches mounted on the FPGA board.

### B.2 Test cases

The SPI communication was tested with an initial program, that wrote instructions to the PM and then read back the content of the PM to confirm that it had been programmed correctly. The SPI is implemented so that the results from the previous command, is returned during the next transmission. With this functionality the SPI was confirmed twice, first with the return result while programming the PM and then by reading the PM.

The Senior processor was first tested with a simple toggle program, toggling the data_o[16] pin every third clock cycle. This was to confirm that the Peripheral interface responded when using out instructions. This program was also used when the Senior processors power consumption at different clock frequencies was evaluated.

If larger programs would be run, the results from the calculations could not be confirmed with the present test setup since the data pins could not be measured. With this in mind a program was designed to read the value of register r0 and send the value of each bit on the data_o strobe signal data_o[16]. Since registers r0 to r31 do not reset if the Senior processor is reset, the Senior processor can be reprogrammed to check the values of each bit in these registers. Depending on the value of the desired bit, data_o strobe would be triggered so it can be read using the TDO pin on the Xilinx cable. The MISO signal of the SPI is only of interest when reading a PM address or checking the return signal from the previous SPI call.

### B.3 Results

The first test program could toggle the data_o strobe signal as intended. Using this program as a benchmark, the current of the 2.5 V I/O voltage was measured at different clock frequencies. The results from these measurements and the test program can be found in Appendix C.

Reading r0 using the “read register” program proved successful. The read r0 program can be found in Appendix D.

### B.4 Conclusions

Since the mounted test-boards could not be delivered on time, not all functionality of the Senior processor could be tested. But thanks to good design choices the
Senior processor could still be tested with minimum hardware, the test conducted proved the following:

- Writing data to the data_o pins seems possible, but only the strobe signal has been verified. This also verifies that the Senior processor can be programmed and run I/O instructions.

- Until the finished board is delivered, results from larger programs can be verified for correctness using the “read register” function.

- None of the tests show any signs of the Senior processor being faulty or functionality missing. But how the Senior processor works with external hardware is unknown.
Appendix C

First test program

C.1 Program

This is the first Senior program used to verify that the Senior processor could be programmed and could use the Peripheral interface.

```
nop
nop
nop
set r0, 0xffff
set r1, 0x0000
loop
  nop
  nop
  pm_write(7, 0x98400140); jump ds3, loop
out 0x11, r0
out 0x11, r1
nop
```

C.2 Measurement results

As Figure C.1 shows, when the clock-rate increases the power consumption level of the Senior processor increases. The values have an exponential characteristic. As the frequency increases, the power consumption could be estimated to increase in a faster rate in comparison. But since the tests where conducted with a much lower clock frequency, than what is intended for the Senior processor and with only a few values used for comparison, this conclusion is not valid. The equipment used in these tests for measuring the power consumption levels, could present different values for the same test from time to time. The values presented in Table C.1 should be seen as a median of the different measurements taken in this test.
Table C.1. Measurement results. SPI frequency: 40 KHz (test 1-3), 100Hz (test 4-6)

<table>
<thead>
<tr>
<th>Test</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLK period (us)</td>
<td>2.95</td>
<td>1.45</td>
<td>0.9656</td>
<td>1005</td>
<td>99.7</td>
<td>9.995</td>
</tr>
<tr>
<td>I/O current (clk off) (mA)</td>
<td>-</td>
<td>2.32</td>
<td>2.19</td>
<td>2.21</td>
<td>2.13</td>
<td>2.13</td>
</tr>
<tr>
<td>I/O current (idle) (mA)</td>
<td>2.15</td>
<td>2.13</td>
<td>2.13</td>
<td>2.13</td>
<td>2.18</td>
<td>2.14</td>
</tr>
<tr>
<td>I/O current (running) (mA)</td>
<td>2.62</td>
<td>2.785</td>
<td>2.915</td>
<td>2.38</td>
<td>2.41</td>
<td>2.5</td>
</tr>
<tr>
<td>I/O current (core stop) (mA)</td>
<td>2.13</td>
<td>2.17</td>
<td>2.19</td>
<td>2.14</td>
<td>2.14</td>
<td>2.14</td>
</tr>
</tbody>
</table>

Figure C.1. Power consumption results at different clock frequencies
Appendix D

Read register program

This is the read register program that was used to verify the correctness of the Senior processor, during power consumption measurements.

/* Calls the read_r0_bit(i) for each bit in the register and returns the result */
int read_r0() {
    int results = 0;
    int i;
    for (i = 15; i >= 0; i = i - 1) {
        results = (results << 1) | read_r0_bit(i);
    }
    return results;
}

/* Writes a program to PM that checks the value of bit I in r0. If a 1 is found the strobe signal is toggled. */
int read_r0_bit(int bit)
{
    int r0_pin = 0x02080000|(1<<bit);
    int i;
    char ret;
    int value;

    stop_core();
    pm_write(0,0x81000000); //nop
    pm_write(1,0x81000000); //nop
    pm_write(2,r0_pin); // set r4, r0_pin
    pm_write(3,0x81000000); //nop
    pm_write(4,0x50884000); //and r4, r0
    pm_write(5,0x804002c3); //jump.ne one
pm_write(6,0x81000000); //nop
pm_write(7,0x81000000); //nop
pm_write(8,0x81000000); //nop
pm_write(9,0x804001c0); //jump zero
pm_write(10,0x81000000); //nop
pm_write(11,0x04c00011); //out 0x11, r0
pm_write(12,0x04c00011); //out 0x11, r0
pm_write(13,0x04c00011); //out 0x11, r0
pm_write(14,0x984002c0); //jump ds3, one
pm_write(15,0x04c00011); //out 0x11, r0
pm_write(16,0x04c00011); //out 0x11, r0
pm_write(17,0x04c00011); //out 0x11, r0

i = 0;
value = 0;

start_core();

while (i < 100) //Read the value on the 'flying wire' TDO
{
    read_status(&ret);
    //If a one is found, end the loop and stop core
    if (ret&miso_o)
    {
        i = 100;
        value = 1;
    }
    else
    {
        i = i + 1;
    }
}
stop_core();
return value;
Appendix E

FFT test report

This test was for measuring the power consumption of the Senior processor at different clock frequencies. The FFT algorithm was used as the benchmark program and the results were checked so that all measurements were on correct calculations.

E.1 Test setup

A temporary test computer was used during these measurements, for development and verification of the tests the office computer was used.

Xilinx Parallel Cable 4 with a “flying wire adapter” was connected to the jumper headers on the Senior test-board, in a similar manner as previous tests.

Table E.1. Flying wire adaptor to Senior test-board connections

<table>
<thead>
<tr>
<th>Xilinx parallel 4 cable</th>
<th>Senior SPI interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCK</td>
<td>SCLK</td>
</tr>
<tr>
<td>TMS</td>
<td>SS</td>
</tr>
<tr>
<td>TDI</td>
<td>MOSI</td>
</tr>
<tr>
<td>TDO</td>
<td>data_o[16]</td>
</tr>
<tr>
<td>VREF</td>
<td>2.5 V</td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>INIT</td>
<td>-</td>
</tr>
</tbody>
</table>

The VREF was connected to one of the 2.5 V headers using an extra header, with a wire connecting the two headers.

The Senior test-board was mounted on the FPGA board as intended using the AvBus contacts.

Xilinx platform Cable USB was used to program the FPGA.
For measuring the power consumption, the KEITHLEY 2701 Ethernet Multimeter was used. Probe wires were connected between the two CORE_VDD headers and the multimeter channels.

### E.2 FPGA program

The program used on the FPGA was named high_speed_clk_gen and it contains the following components:

- A DCM_adv module was used for generating the clock signal. The clock output is CLK_FX and it has three values that need to be changed for different frequencies. CLKFX_MULTIPLY and CLKFX_DIVIDE are used to multiply and divide the incoming clock signals, which will become the new CLK_FX. DFS_FREQUENCY_MODE should be assigned to “LOW” for output frequencies between 32 MHz and 210 MHz, over 210 MHz and up to the max of 300 MHz the value assigned should be “HIGH”.

- A small module was used to manipulate the LEDs on the FPGA board. The module consist of a 32 bit counter that increments with each clock cycle. The 8 highest bits are outputs to the LEDs and the rate at which the LEDs flicker indicates the speed of the clock.

- A module named OFDDRRSE, is used to send a stable clock signal from the FPGA to the Senior processor. It uses the FPGAs internal clock signal to generate a new clock signal that it forwards to the Senior processor. This enables a more stable clock signal to Senior, than if only the FPGA clock signal would be forwarded.

IMPACT was used to program the FPGA.

### E.3 Senior program

The following sections explain different part of the FFT test program. Each section is named according to the function under discussion.

#### E.3.1 FFT_TEST

A C program that resembles the run_fpga_design was used for these tests. The main function used to handle the test cases is called fft_test. The hex files containing the test data and Senior programs are read by load_hex. The data memory is first filled with twiddle factors using a download_dm function. Input data for the FFT calculations are transferred next using the store_values function and are stored on address 0x5000. Expected results are transferred after that and also by store_value and are stored on address 0x4200. Last function to run is the download_pm which downloads the FFT program to the program memory.
The start_core function starts the FFT calculations. The calculations are either stopped by the stop_core function in the middle of a calculation after a predetermined time has passed or the program waits for the stop signal generated by the Senior program.

When the calculations have been halted one way or the other, the function read_fft_results is run.

E.3.2 Stop signal

The stop signal consists of a loop that has been added at the end of the Senior program. Below follows an example of how the loop could be constructed.

```
end_loop
  out 0x11, r0
  out 0x11, r0
  out 0x11, r0
  jump ds3 end_loop
  out 0x11, r0
  out 0x11, r0
  out 0x11, r0
```

When the out function is called, the data_o strobe (data_o[16]) goes high and this is what is used as the stop indicator. Since no other out instructions are used in the program, this will work as a stop signal for this test setup.

The C program that handles the test puts itself into an infinite loop while waiting for the stop signal to go high. When the stop signal goes high the loop is stopped. With this approach no calculations are needed to determine how long time a certain program will take to run.

E.3.3 download_dm

Works in the same manner as in run_fpga_design, the stop signal is added at the end.

E.3.4 store_values

Takes a file that contains one 16-bit hex value per line, writes a program to Senior that will load to memory at a given address with these values. It works as download_dm, but it takes a file name, number of words, step length and the address where to save the data. This allows the functions to be used by different program to store different kinds of data.

E.3.5 download_pm

Works in the same manner as in run_fpga_design, the stop signal has been added in the end.
E.3.6 read_fft_results

Takes the number of values that wishes to be read as input. Writes a program to Senior that loads a dm value into r0 and reads this value with read_r0. When reading the fault accumulation values, if the data is not equal to zero the values are printed in the terminal. To indicate that the program has not for some reason stopped and how many values that are left to test, the numbers of values left to test is printed in the terminal every eight value.

E.4 Control of results

When an FFT calculation has been completed, the following code is executed.

```
 nop
 set f11 ,0
 move ar0 , outputAdr ; address of calculated results
 set step0 ,2
 set r0 , 0x01FE ; offset expected
 set r1 , 0x03FE ; offset xor
 set r2 , 0x05FE ; offset fault
 repeat filt , 128
 ld0 r16 , ( ar0++) ; result
 ld0 r17 , ( ar0 , r0) ; expected result
 ld0 r18 , ( ar0 , r1) ; xor
 ld0 r19 , ( ar0 , r2) ; calculation fault collector
 xor r20 , r16 , r17 ; xor results with expected
 xor r21 , r16 , r18 ; xor previous xor
 or r19 , r20 ; collect fault
 st0 ( ar0 , r1) , r21 ; store xor
 st0 ( ar0 , r2) , r19 ; store fault
 filt
```

This program compares the calculated results on address 0x40xx with expected results on address 0x42xx by using a XOR function. The results from the XOR functions is included to the fault collection value on address 0x46xx using an OR function. It is these results that are printed on the terminal by the read_fft_results function if they are not equal to zero. If no values are printed then the calculations have been successful.

E.4.1 Test implementation

The FPGA was programmed with the high_speed_clk_gen, when the programming was finished the multimeter indicated activity on the Senior processor.

The FFT calculations were run during a 30 second period, the value of the current was recorded at the start and end of calculations, during the print phase and after the test had finished. The value of the current was also recorded right after the FPGA had been reprogrammed with the new clock frequency.
The first measurements conducted used a frequency of 300 MHz, these values did not seem reliable. At the first run the current was at 35 mA during calculations, but with each run the current increased with 1-2 mA. Eventually the power consumption increased drastically to 48 mA and the calculations showed to be incorrect. After this incident this certain test-board could only perform a few FFT calculations at 300 MHz before the calculations failed.

The frequency was reduced to 200 MHz, where several more calculations could be done with correct result. For each measurement the frequency was decreased by 25 MHz all the way down to 50 MHz.

After this measurement cycle the frequency was increased to 225 MHz, where accurate calculations could be performed. At 250 MHz, correct calculations could also be performed. At 275 MHz, no accurate calculations were performed.

A second test-board was used and the tests were run at 200 MHz, the results were accurate and the measured values matched with the first test-board. Increased the frequency to 275 MHz and accurate calculations could still be made. But at 300 MHz, no correct calculations could be made.

### E.5 Error sources

Potential sources for measurement disturbances in these tests would lie in the connections between the Senior test-board and the measurement equipment. The probes that were available were older and a little worn, which could distort the measurements.

### E.6 Results

<table>
<thead>
<tr>
<th>Table E.2. Measurement results test-board 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock frequency (MHz)</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>300 (FAIL)</td>
</tr>
<tr>
<td>275 (FAIL)</td>
</tr>
<tr>
<td>250</td>
</tr>
<tr>
<td>225</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>175</td>
</tr>
<tr>
<td>150</td>
</tr>
<tr>
<td>125</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>75</td>
</tr>
<tr>
<td>50</td>
</tr>
</tbody>
</table>

Table E.2 contains the measurement results from the first test-board. * indicates that the last decimals of the value was not completely stable at the time of
measurement. Since calculations at 275 MHz and 300 MHz could not be performed correctly these results cannot be seen as valid values.

### Table E.3. Measurement results test-board 2

<table>
<thead>
<tr>
<th>Clock frequency (MHz)</th>
<th>IDLE (mA)</th>
<th>MIN (mA)</th>
<th>MAX (mA)</th>
<th>READ (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 (FAIL)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>275</td>
<td>29,31*</td>
<td>45,289</td>
<td>45,313</td>
<td>29,53*</td>
</tr>
<tr>
<td>200</td>
<td>21,64*</td>
<td>33,13</td>
<td>33,134</td>
<td>-</td>
</tr>
</tbody>
</table>

Results from tests with the second test-board can be seen in Table E.3. Since calculations at 300 MHz could not be performed correctly these results cannot be seen as valid values.

## E.7 Summary

### Table E.4. Measuring results relative to each other

<table>
<thead>
<tr>
<th>Clock frequencies (MHz)</th>
<th>Δ idle (mA)</th>
<th>Δ work (mA)</th>
<th>Δ read (mA)</th>
<th>Clock frequency (MHz)</th>
<th>work idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 to 275</td>
<td>2,54</td>
<td>-13,57</td>
<td>2,35</td>
<td>300</td>
<td>1,1</td>
</tr>
<tr>
<td>275 to 250</td>
<td>2,62</td>
<td>7,51</td>
<td>2,67</td>
<td>275</td>
<td>1,66</td>
</tr>
<tr>
<td>250 to 225</td>
<td>2,6</td>
<td>4,04</td>
<td>2,61</td>
<td>250</td>
<td>1,54</td>
</tr>
<tr>
<td>225 to 200</td>
<td>2,62</td>
<td>4,09</td>
<td>2,64</td>
<td>225</td>
<td>1,54</td>
</tr>
<tr>
<td>200 to 175</td>
<td>2,63</td>
<td>4,02</td>
<td>2,65</td>
<td>200</td>
<td>1,54</td>
</tr>
<tr>
<td>175 to 150</td>
<td>2,64</td>
<td>4,06</td>
<td>2,66</td>
<td>175</td>
<td>1,54</td>
</tr>
<tr>
<td>150 to 125</td>
<td>2,66</td>
<td>4,06</td>
<td>2,68</td>
<td>150</td>
<td>1,54</td>
</tr>
<tr>
<td>125 to 100</td>
<td>2,67</td>
<td>4,07</td>
<td>2,69</td>
<td>125</td>
<td>1,54</td>
</tr>
<tr>
<td>100 to 75</td>
<td>2,69</td>
<td>4,17</td>
<td>2,71</td>
<td>100</td>
<td>1,55</td>
</tr>
<tr>
<td>75 to 50</td>
<td>2,69</td>
<td>4,22</td>
<td>2,72</td>
<td>75</td>
<td>1,54</td>
</tr>
</tbody>
</table>

In Table E.4 it is presented how different measurement values relate to each other. Δ columns shows the increase in current when the frequency is increased with 25 MHz and work/idle shows the relation between current levels during calculations and idle. All measurement values show the same relationship, the relation between work and idle is approximately 54% for all frequencies except for 300 MHz and 275 MHz which both could not perform correct calculations. This shows that the power consumption is linear to the clock frequency.

Since test-board 1 could not perform correct calculations over 250 MHz, test-board 2 was tested at 275 MHz and could perform correct calculations. To see if the measurement results on test-board 2 were reasonable, similar calculations
Table E.5. Measuring results relative to test-board 1 and 2

<table>
<thead>
<tr>
<th>Clock frequencies (MHz)</th>
<th>∆ idle (mA)</th>
<th>∆ work (mA)</th>
<th>∆ read (mA)</th>
<th>Clock frequency (MHz)</th>
<th>work/ idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test-board 2</td>
<td></td>
<td></td>
<td></td>
<td>Test-board 2</td>
<td></td>
</tr>
<tr>
<td>275 to 200</td>
<td>7.67</td>
<td>12.17</td>
<td>7.89</td>
<td>275</td>
<td>1.55</td>
</tr>
<tr>
<td>Test-board 1</td>
<td></td>
<td></td>
<td></td>
<td>Test-board 1</td>
<td></td>
</tr>
<tr>
<td>250 to 175</td>
<td>7.84</td>
<td>12.14</td>
<td>7.9</td>
<td>250</td>
<td>1.54</td>
</tr>
<tr>
<td>225 to 150</td>
<td>7.98</td>
<td>12.16</td>
<td>7.95</td>
<td>225</td>
<td>1.54</td>
</tr>
</tbody>
</table>

presented in Table E.4 are presented in Table E.5 between the results from test-board 1 and 2. The results show that the test-board 2 follows the same pattern as test-board 1.

E.8 Conclusion

The Senior processor can perform correct calculations up to 250 MHz, higher frequencies should be avoided.
Appendix F

MIMO accelerator test report

This test was to verify that the taped-out Senior processor can communicate with and use the SIMD processor to do MIMO calculations as it has previously done in simulations.

F.1 Test equipment

The following equipment was used for this test:

- Xilinx Virtex-4 LX Evaluation Board with xc4vlx60 FPGA
- RS-232 cable from FPGA to USB adapter
- Xilinx platform Cable USB for FPGA programming.

For this test two identical folders were created, containing all available code for the original test setup. The code in one of these folders was altered to be able to perform tests with the Senior test-board and the other was used for performing control tests. The Senior test-board was mounted on the FPGA for the tests with the external Senior processor and it was removed during control tests with the FPGA version.

F.2 Control test

The SIMD program available was synthesized and programmed on the FPGA board. The needed UCF file was missing from the available files, so a new one was created with the signals that were specified in the main Verilog file.

The original code was written and tested with the system running at 25 MHz. The run_fpga_design.c program was checked and the previously known faults that would not allow the program to run properly were corrected.
F.3 Test external Senior

The same test program used in the control test was used to test the external Senior processor.

F.4 Handling of incorrect tests

The following actions where taken after incorrect results had been achieved, however it has not been recorded in what order the different actions were taken. After all these actions had been taken, correct test results could be achieved.

- The C program was checked and it was noticed that fault correction was used in the control C program but not in the present C program. Fault correction was introduced into the present C program.
- A small fault in the code was noticed in fpga_top, the fault was corrected.
- The reset signal to the external Senior processor was not initiated properly, the issue was resolved.
- A gap in the 2-stage pipeline was found, causing the two first values to not be recorded. The pipeline was rewritten so that the gap was removed.
- The reset jumper on the Senior test-board was mounted wrong, the Senior processor was receiving its reset signals from the reset button, the jumper was moved.
- The clock inputs to the clock buffer, OFDDRRSE buffer, were inverted. Not inverted signals resulted in that the 7 highest results bits were recorded one cycle early.

F.5 Results

Table F.1 contains a sample of the test results that best corresponded with the control results. The total result consist of 128 words, but the first 20 presented in Table F.1 shows the last fault that deviated from a correct result.

Test 1 consisted of a modified fpga_top, the 2-stage pipe was used on the data_o signal from the Senior processor to the FPGA. An OFDDRRSE buffer was used to deliver proper clock signal from the FPGA to the Senior processor. Necessary changes to the Verilog and UCF files were made to be able to connect to the Senior test-board.

Test case 1 shows almost the same results as the control results, but the difference between all the values is that the seven most significant bits are recorded by the FPGA one clock cycle early than expected. This was probably due to the clock signal and the way it propagates through the system. First it goes through the modules in the FPGA, then out to the Senior processor and then used to handle the data transmission back to the FPGA. If the timing of the strobe signal is bad
Then it could be caught by the FPGA on the wrong flank, which would result in the FPGA reading only a part of the intended value and the rest on the next clock flank.

To counter this, the clock signals to the clock buffer on the FPGA were inverted. The new code was synthesized, FPGA reprogrammed and the test was run again. The results from this test, test 2, were correct and corresponded with the control results. Test 3 was a control test to see if the previous fault could be achieved. The results from test 3 are the same as from test 1.

A RTL simulation was conducted separately at the same time as these tests were conducted. The results from the RTL simulation were the same as the control results.

### F.6 Conclusion

The Senior test-board can be used together with an accelerator in the FPGA. But because of how the FPGA code is synthesized or how the Senior processor has been synthesized, the clock signal to the Senior processor has to be inverted to be able to achieve correct results using this accelerator.
Appendix G

SIMD - DMA test report

G.1 Equipment
The following equipment was used for this test:

- Xilinx Virtex-4 LX Evaluation Board
- Xilinx platform Cable USB
- The Senior test-board

G.2 Test implementation
To be able to calculate the different time intervals and events the mimo_tester.v module was created. This module consisted of several counters, that counted the number of cycles between different intervals and recorded the number of signals triggered.

```verilog
module mimo_tester(
    input wire clk,
    input wire reset,
    input wire acc_strobe,
    input wire irq_simd,
    input wire [19:0] code,
    input wire [7:0] SWITCH,
    output wire [7:0] led_o
);
```

Presented above is the module mimo_tester port list. The signals acc_strobe, irq_simd and code were use to measure and determine the lengths of the different intervals.

The accelerator commands “.dw 0xc0000000” and “.dw 0xe0000000” were inserted in different parts of the code to measure the intervals of interest. The “.dw 0xc0000000” command was use together with the irq_simd signal to measure the
time between the SIMD call and SIMD response, the SIMD response being the irq_simd signal.

SWITCH and led_0 were used to read the recorded test data. Depending on the different values on the SWITCH signals, different parts the recorded data will be presented on the LEDs of the FPGA board.

The code used to calculate different intervals without the interference from the irq_simd signal, was written so that an extra clock cycle was recorded. This extra cycle has been considered in the final results.

The code for the Senior processor can be found with the source code used for this project.

### G.3 Test results

It takes 4459 clock cycles for the main program to receive the calculated data from the SIMD processor after execution.

DMA task 1 has been identified to be the task that sends all matrix information to the SIMD processor, this task takes 2081 clock cycles. If you take into account the instructions to setup the mux, the total number of cycles is 2084.

DMA task 2 is the task that transfers the program data from Senior to SIMD. This task takes 286 cycles or 289 if the mux instructions are included.

Initializing, execution of the SIMD unit and data retrieving takes 2079 cycles, 2082 counting the mux instructions.

From the actual SIMD execution command until it has completed its calculations takes 1883 cycles. In this part of the program the SIMD unit sends interrupt signals that are handled in the code. Counting from the SIMD execution command up to the last interrupt sent, 1856 cycles has passed.

```assembly
; ;; Enable interrupt flag and start execution
set r5, 0x3
out IOR_CONTROL, r5
set r1, 1000
loop
    nop
decl r1
    nop
jump.ne loop
set r5, 0x4
out IOR_CONTROL, r5
```

Here is the Senior interrupt routine for handling the interrupts send by the SIMD unit.

```assembly
SIMD_INTERRUPT_ROUTINE
    set r5, 0x4
    out IOR_CONTROL, r5        ; Clear the interrupt
    out IOR_DMA_PTR, r0        ; Set DMA pointer to zero
    out IOR_CONTROL, r0        ; Set PC to zero
```
set r1, 1
reti

The code above is where the call to the SIMD unit is and the loop used to wait for the SIMD unit to finish.

Counted from when the program loads the matrix data to the SIMD and until Senior receives all the data results it takes 4455 cycles, mux instructions for DMA1 included.

As the program is currently written, first the data needs to be loaded, then the SIMD program and then the calculations can be executed. But if the SIMD would be able just to receive matrix data and directly calculate the results, running the DMA1 task and then the SIMD call and data retrieval, this execution order would take $2084 + 2082 = 4166$ cycles, with mux instructions included.

DMA task 3 is used for retrieving the calculation results, that execution time is included in SIMD init, execution and data retrieval. DMA3 takes 184 cycles to run, 187 with mux instructions included.