CPU Architecture

Third Laboratory

Hanan Ribo, Boris Braginsky and Prof. Hugo Guterman

6.5.18

Table of contents

List of Figures

List of Tables

1. Aim of the Assignment

- Design, synthesize and analyze a simple MIPS compatible CPU.
- Understanding in FPGA memory structure

2. Definition and prior knowledge

The aim of this laboratory is to design a simple MIPS compatible CPU. The CPU will use a PIPELINED architecture and must be capable of performing instructions from MIPS instruction set. The design will be executed on the Altera Board.
The MIPS architecture is Harvard architecture in order to increase throughput and simplify the logic.
There is need to implement floating point instructions (ADD, SUB and MUL from previous work) and floating point register file.
You must decide by yourself what MIPS instructions you should implement for test described in part 6.
For additional information regarding MIPS CPU, Architecture, ISA and instructions see MIPS technical documents [1].

3. Assignment definition

You must design a pipelined MIPS compatible CPU (at least **4 stages**).
All the possible hazards must be solved in hardware!!!
The architecture must include a MIPS ISA compatible CPU with data and program memory for hosting data and code. The block diagram of the architecture is given in Figure 1. The CPU will have a standard MIPS register file. The top level and the MIPS core must be structural. The design must be compiled and loaded to the Altera board for testing. A single clock (CLK) should be used in the design.

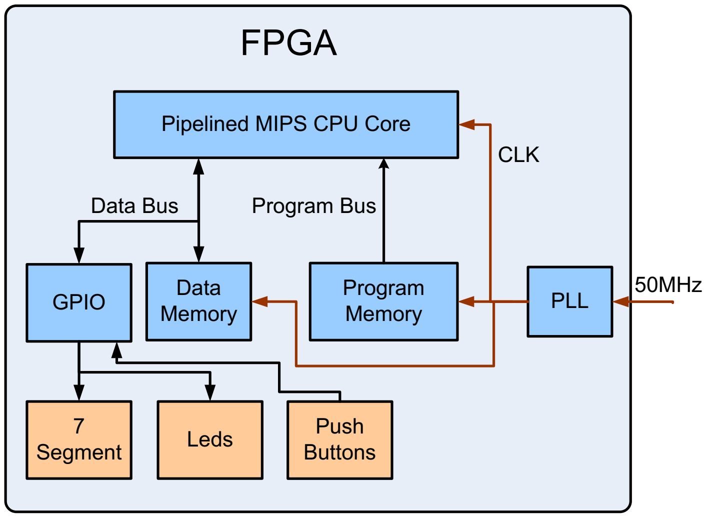

Figure 1 : System architecture

The PLL [2] is used to make a higher frequency from the 50MHz clock and is used in FPGA compilation only.
The GPIO (General Purpose I/O) is a simple buffer registers mapped to some data address (Higher than data memory) that enables the CPU to output data to LEDS and 7-Segment and to read the Push-Buttons state.
The CPU will be based on standard 32bit MIPS ISA and the Instructions will be 32 bit wide. The following table shows the MIPS instruction format. For more information see MIPS technical documents [1].

Type	**-31-**	**format (bits)**				**-0-**
R	opcode (6)	rs (5)	rt (5)	rd (5)	shamt (5)	funct (6)
I	opcode (6)	rs (5)	rt (5)	immediate (16)		
J	opcode (6)	address (26)				

Table 1 : MIPS Instruction format

The memory maximal sizes and latency will be according to Table 2

Memory	**Maximal Size**	**Write Latency**	**Read Latency**
Program Memory	1KByte	1 clk	1-2 clk
Data Memory	1KByte	1 clk	1-2 clk

Table 2 : Memory sizes and latency

NOTE: All the memories must be implemented using FPGA RAM Blocks. Use MegaWizard Plug in Quartus to generate the memories and to set their content.

Pay attention, due to the memory latencies and non-pipelined architecture a single instruction will be executed in several clocks , that means that you need to hold the program counter till the instruction execution finishes.

4. Compiler , Simulator an Memory

The MARS compiler and simulator, or any other can be used to compile and simulate the assembly code. MARS compiler can also export the memory contents into the file in format that VHDL can easily read. It can also simulate a cache performance.
The mars compiler, installation instructions and documentation are available at: http://courses.missouristate.edu/KenVollmar/MARS/

5. Add-ons to MIPS

The FPU module from Work 2 must be connected to CPU. Additionally, interrupts from buttons must be created, using memory mapped registers and interrupt vector. Also 7 segments must be used with memory mapped registers. If necessary other IO devices (switches, led) can be used with memory mapped registers too.

6. Test

As a test bench you need to write code which first of all have initialize necessary interrupts from buttons/switches. Secondary sort floating point vector (pre-stored in data memory, vector size is 8) and show the sorting vector on 7 segments display with delay of 1s.

7. Requirements

You have to do the following tasks:

- Modelsim Simulation with maximal coverage
- Analyze the critical path, explain where is it in the VHDL file and design and Find the maximal operating clock
- Load the design into the FPGA and verify the simulation results

The following must be presented in the report.

1. Top level block review diagram of your design.
2. For each block in the top level design:
    - RTL Viewer results
    - Logic usage for each block (Combinational and Flip-Flops)
    - Graphical description (a square with ports going in and out).
    - Port Table (direction, size, functionality).
    - Short description.
3. Maximum(Critical) path of your design – explain where it is in the code and how it is possible to optimize it if you would have more time. What is the maximum clock frequency?
4. Minimum path analysis.
5. Comparison between FPGA output and MATLAB simulation of the process.
6. Style - Contents with page numbers, Images and tables will be numbered. The caption of an images and tables below the images or tables.
7. Elaborated analysis and wave forms:
    - Test bench description –

- Regular cases that were checked.
- Extreme cases that were checked
- Forbidden cases that were not checked.
- Maximal Frequency and critical paths from Timing Analyzer
- Proof of work using Signal Tap
- **<u>One</u>** basic waveform to explain the system timing.

8. CPI and maximal frequency analysis.
9. Conclusions and future work (how can the system be improved if you'd had more time)

Design requirements:

1. The design must be well commented.
2. The system must work from only one clock.
3. Resets in the system must be synchronous except the main system asynchronous reset.
4. Do not use latches.

A zip file form of id1_id2.zip where id1 and id2 are the identification number of the submitters, and id1 < id2 will be submitted to HighLearn site. The file will contain:

- The full *.doc or *.pdf report file, with operation manual of the watch.
- Modelsim simulations
- **Quartus project including the VHDL files and the project itself**
- The SignalTap files used in project debug
- Quartus Project SOF file
- The ZIP must not contain temporary files ("db" and "db_incremental" and "work" directories).
-
- The ZIP file directory structure must be arranged according to Table 3.
- After organizing the ZIP file, verify compilation and that no files are missing.

Directory	**Contains**	**Comments**
VHDL	Project VHDL files	Only VHDL files , excluding test bench
TB	VHDL files that are used for test bench	
SIM	Modelsim project “.mpf” and waveform “.do” files	Do not place files that are not relevant for compilation or is a result of compilation
DOC	Project documentation	Readme.txt and PDF report file

Quartus	Quartus Project files	Do not place files that are not relevant for compilation or is a result of compilation
CODE	The assembly source code	

Table 3 : Directory Structure

8. Grading policy

Weight	Task	Description
30%	Requirements	The entire list of requirements in the assignment paper
20%	Documentation	The "clear" way in which you presented the requirements and the analysis
20%	Design	The overall system design, code styling, comments, logic usage and code/files organization.
20%	Analysis and Test	The correct analysis of the system and test bench coverage
10%	Conclusions	Asserted conclusions on the work you've done

Table 4 : Grading

Under the above policies you'll be also evaluated using common sense:

- Your files will be compiled and checked, the system must work.
- Your design and architecture must be intelligent, effective and well organized.
- The design must be well commented

Submission date 27/5/2018 to the borisbr@post.bgu.ac.il
Short presentation in the LAB on 31/5/2018 between 16-18.

For a late submission the penalty is 2^{days} .
For early submission the reward is 2*days (up to 3 days).

9. References

[1]. MIPS32® Architecture for Programmers Volume I to III (from HighLearn under Final Project)
[2]. ALTPLL User Guide: http://www.altera.com/literature/ug/ug_altpll.pdf
[3]. Altera RAM user guide: http://www.altera.com/literature/ug/ug_ram_rom.pdf
[4]. Altera Megafunction User Guide: www.altera.com/literature/ug/ug_intro_to_megafunctions.pdf
[5]. Bin2Hex utility
32bit - http://www.keil.com/download/docs/113.asp
64bit - http://www.ht-lab.com/freeutils/bin2hex/bin2hex.html