Universitatea Tehnică “Gheorghe Asachi”, Iași
Facultatea de Automatică și Calculatoare

Raport intermediar despre desfășurarea proiectului de licență

Student: Ungurean Marcel Dumitru

Grupa: 1404A

Specializare: Calculatoare

Profesor îndrumător: Sef lucr.dr.ing. Andrei Stan

1. Titlul lucrării de licență

Implementarea unui processor RISC dupa arhitectura setului de instructiuni RISC-V.

2. Ce își propune proiectul

Proiectul isi propune implementarea pe un FPGA (Field-programmable gate array) a unui procesor cu arhitectura RISC (Reduced instruction set computer) dupa arhitectura setului de instructiuni RISC-V dezvoltata in cadrul Departamentului EECS al Universității Berkeley din California.

RISC-V (“risk-five”) este o noua arhitectura a unui set de instructiuni desemnata original sa ajute cercetarea in cadrul arhitecturii calculatoarelor si in educatie iar acum se apropie de a fi o architectura standard “deschisa” pentru implementari industrial, etc.

3. Descrierea proiectului

Prima etapa consta in definirea unui sistem de calcul RISC pentru procesorul care urmeaza a fi implementat in conformitate cu specificatiile date de arhitectura setului de instructiuni RISC-V. In prima faza se doreste realizarea implementarii considerand doar setul de instructiuni pentru operarea cu numere intregi de lungime fixa pe 32 biti. Ulterior se va considera si adaugarea de extensii pentru operarea cu numere intregi de lungime fixa pe 64 biti, operarea cu numere in virgula mobila cu simpla precizie sau dubla precizie.

Sistemul va avea 31 de registrii de uz general care vor putea stoca valori intregi (X1-X31) plus registrul X0 care va contine valoarea 0 aceasta fiind o valoare constanta. Pe langa acest set de registrii mai exista un registru aditional, un numerator , care va contine adresa instructiunii curente.

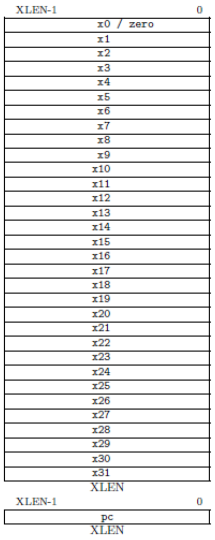

Figure 2.1: RISC-V user-level base integer register state.

Formatul pentru instructiunile de baza va fi cel reprezentat in figura urmatoare:

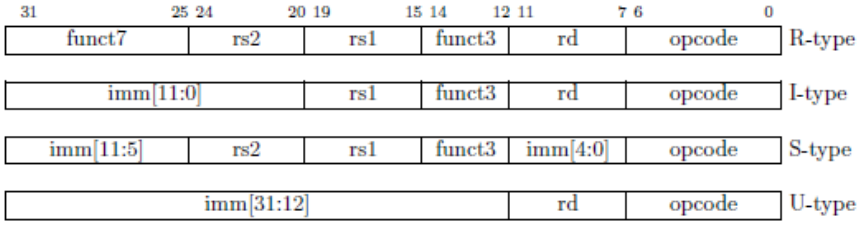

Figure 2.2: RISC-V base instruction formats.

Urmatoarea etapa presupune implementarea si simularea procesorului intr-un limbaj de descriere hardware cum ar fi Verilog si ModelSim. Dar fiindca partea de verificarea a unui astfel de model complex poate fi mare consumatoare de timp folosind un limbaj precum SystemVerilog pentru realizarea acestui lucru am decis folosirea limbajului de programare CHISEL (Constructing Hardware in a Scala Embedded Language).

Chisel este un limbaj de descriere hardware, dezvoltat de aceeasi universitate Berkeley din California, care suporta descrierea de design-uri hardware avansate folosind generatoare de cod parametrizabile si limbaje hardware.

Am ales Chisel in principal pentru motivul ca poate genera pe modului/componentei descris/e , atat codul Verilog care sa poata fi sintetizabil cat si un simulator software C++ de viteza relativ mare. Astfel pe parcursul dezvoltarii proiectului voi putea realiza in acelasi timp si verificarea functionala a modulelor descrise.

Ultima etapa presupune sintetizarea codului Verilog generat incarcarea acestuia pe un FPGA bazat pe technologia Xilinx si testarea acestuia prin rularea unor programe de complexitati diferite pentru a analiza diverse aspecte de performanta a procesorului implementat.

4. Stadiul curent al proiectului

Pana acum :…

Documentare … (TO DO)

Instalarea tool-urilor necesare pentru dezvoltarea proiectului.

Realizarea unor module de test in vederea validarii generarii de cod Verilog + testarea functionala a acestuia in C++.

Stadiul curent: Realizarea arhitecturii sistemului de calcul .

5. Bibiliografie

Websites:

http://riscv.org/specifications/

https://chisel.eecs.berkeley.edu/latest/chisel-tutorial.pdf

http://www.scala-sbt.org/documentation.html

Books:

Computer Organization and Design The Hardware Software Interface 3rd Edition 2004 – David A. Patterson, John L. Hennesy