Μεταπτυχιακή Διατριβή

Θέμα : Σχεδίαση και υλοποίηση λογισμικού προσομοίωσης επεξεργαστή αρχιτεκτονικής MIPS με δυνατότητα υπολογισμού μοντέλων κατανάλωσης ισχύος σε C++

Επίβλεψη : Καθηγητής κος Γεώργιος Σταμούλης
Εκπόνηση : Αθανάσιος Μπαρέκας

Σύντομη περιγραφή στα Ελληνικά

Η συγκεκριμένη μεταπτυχιακή διατριβή, εκπονήθηκε στα πλαίσια του προγράμματος σπουδών του μεταπτυχιακού προγράμματος με τίτλο «Επιστήμη και Τεχνολογία Υπολογιστών, Τηλεπικοινωνιών και Δικτύων» του τμήματος Μηχανικών ηλεκτρονικών υπολογιστών τηλεπικοινωνιών και δικτύων του Πανεπιστήμιου Θεσσαλίας.

Το θέμα της συγκεκριμένης μεταπτυχιακής διατριβής πραγματεύεται την υλοποίηση ενός εργαλείου προσημείωσης αρχιτεκτονικής υπολογιστών Mips, όσον αφόρα την απόδοση του ρυθμού ολοκλήρωσης, την λειτουργία των διαφόρων υπομονάδων που συμμετέχουν στη επεξεργασία και τέλος στην κατανάλωση ισχύος ανά υπομονάδα με βάση την εισαγωγή κυματομορφών που προέρχονται από διάφορα μοντέλα κατανάλωσης.

Η αρχιτεκτονική η οποία υλοποιήθηκε στο συγκεκριμένο λογισμικό προσημείωσης είναι ένα απλό μοντέλο αρχιτεκτονικής MIPS με μερική επικάλυψη και στατική πρόβλεψη διακλαδώσεων. Οι κυματομορφές κατανάλωσης ισχύος φορτώνονται κατά τη διάρκεια της εκτέλεσης και είναι διαφορετικές ανά υπομονάδα επεξεργασίας και ανάλογες με την εντολή που εκτελείται εκείνη τη στιγμή στην συγκεκριμένη υπομονάδα.

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Class Documentation

Cpu Class Reference

`#include <cpu.h>`

Public Member Functions

- **Cpu** ()
- void **setfetchState** (int state)
- int **getfetchState** ()
- void **setdecodeState** (int state)
- int **getdecodeState** ()
- void **setexecState** (int state)
- int **getexecState** ()
- void **setmemState** (int state)
- int **getmemState** ()
- void **setwbState** (int state)
- int **getwbState** ()

Public Attributes

- **Fetch f**
- **Decode d**
- **Execute e**
- **Memory m**
- **WriteBack w**

Detailed Description

@ This Class represents a whole Mips **Cpu** unit object

Definition at line 14 of file cpu.h.

Constructor & Destructor Documentation

Cpu::Cpu ()

@ Default constructor, when called creates a new cpu object and sets all cpu phases state to -1

Definition at line 3 of file cpu.cpp.

Member Function Documentation

int Cpu::getdecodeState ()

@ Gets the number of the instruction being decoded

Returns:

int

Definition at line 29 of file cpu.cpp.

int Cpu::getexecState ()

@ Gets the number of the instruction being executed

Returns:

int

Definition at line 39 of file cpu.cpp.

int Cpu::getfetchState ()

@ Gets the number of the instruction being fetched

Returns:

int

Definition at line 19 of file cpu.cpp.

int Cpu::getmemState ()

@ Gets the number of the instruction accessing the memory

Returns:

int

Definition at line 49 of file cpu.cpp.

int Cpu::getwbState ()

@ Gets the number of the instruction writing the register file

Returns:

int

Definition at line 59 of file cpu.cpp.

void Cpu::setdecodeState (int *state*)

@ Sets the number of the instruction being decoded

Parameters:

state	

Definition at line 24 of file cpu.cpp.

void Cpu::setexecState (int *state*)

@ Sets the number of the instruction being executed

Parameters:

state	

Definition at line 34 of file cpu.cpp.

void Cpu::setfetchState (int *state*)

@ Sets the number of the instruction being fetched

Parameters:

state	

Definition at line 14 of file cpu.cpp.

void Cpu::setmemState (int *state*)

@ Sets the number of the instruction accessing the memory

Parameters:

state	

Definition at line 44 of file cpu.cpp.

void Cpu::setwbState (int *state*)

@ Sets the number of the instruction writing the register file

Parameters:

state	

Definition at line 54 of file cpu.cpp.

Member Data Documentation

Decode Cpu::d

Decode instruction phase cpu member object

Definition at line 25 of file cpu.h.

Execute Cpu::e

Executing instruction phase cpu member object

Definition at line 26 of file cpu.h.

Fetch Cpu::f

Fetch instruction phase cpu member object

Definition at line 24 of file cpu.h.

Memory Cpu::m

Accessing memory phase cpu member object

Definition at line 27 of file cpu.h.

WriteBack Cpu::w

Writing registers phase cpu member object

Definition at line 28 of file cpu.h.

The documentation for this class was generated from the following files:

- **cpu.h**
- **cpu.cpp**

Decode Class Reference

`#include <decode.h>`

Public Member Functions

- **Decode** ()
- int **PerformDecode** (**FileLoader** l)

Detailed Description

@ This Class represents the decoding phase of a Mips CPU unit

Definition at line 10 of file decode.h.

Constructor & Destructor Documentation

Decode::Decode ()

@ Default constructor, when called creates a new decode phase object

Definition at line 12 of file decode.cpp.

Member Function Documentation

int Decode::PerformDecode (FileLoader *l*)

@ Function that takes the next instruction and decodes it

Parameters:

l	a **FileLoader** object used to access the instructions list

Returns:

int returns 1 if the the current instruction is stalling else returns 0

Definition at line 17 of file decode.cpp.

The documentation for this class was generated from the following files:

- **decode.h**
- **decode.cpp**

EnergyModel Class Reference

`#include <energymodel.h>`

Public Member Functions

- **EnergyModel** ()
- **EnergyModel** (int **Model**)

Static Public Attributes

- static int **Model**
- static int **Energy**

Detailed Description

Definition at line 4 of file energymodel.h.

Constructor & Destructor Documentation

EnergyModel::EnergyModel ()

EnergyModel::EnergyModel (int *Model*)

Member Data Documentation

int EnergyModel::Energy `[static]`

Definition at line 11 of file energymodel.h.

int EnergyModel::Model `[static]`

Definition at line 10 of file energymodel.h.

The documentation for this class was generated from the following file:

- **energymodel.h**

Execute Class Reference

`#include <execute.h>`

Public Member Functions

- **Execute** ()
- int **PerformExecute** (**FileLoader** l)

Detailed Description

@ This Class represents the executing phase of a Mips CPU unit

Definition at line 10 of file execute.h.

Constructor & Destructor Documentation

Execute::Execute ()

@ Default constructor, when called creates a new execute phase object

Definition at line 10 of file execute.cpp.

Member Function Documentation

int Execute::PerformExecute (FileLoader *l*)

@ Function that takes the next instruction and executes it

Parameters:

l	a **FileLoader** object used to access the instructions list

Returns:

int returns the instruction number executed

Definition at line 15 of file execute.cpp.

The documentation for this class was generated from the following files:

- **execute.h**
- **execute.cpp**

Fetch Class Reference

`#include <fetch.h>`

Public Member Functions

- **Fetch** ()
- void **setFetchPc** (int num)
- int **getFetchPc** ()
- void **PerformFetching** (int &p, **FileLoader** l)
- int **FetchAdder** (int pc, int inc)

Detailed Description

@ This Class represents the fetching phase of a Mips CPU unit

Definition at line 13 of file fetch.h.

Constructor & Destructor Documentation

Fetch::Fetch ()

@ Default constructor, when called creates a new fetch phase object

Definition at line 9 of file fetch.cpp.

Member Function Documentation

int Fetch::FetchAdder (int *pc*, int *inc*) `[inline]`

@ Function that sets the program counter to the next instruction

Parameters:

pc	the current program counter
inc	the increment

Returns:

int return the new instruction number that the program counter is set

Definition at line 53 of file fetch.h.

int Fetch::getFetchPc ()

@ Function that gets the instruction number being fetched

Returns:

int the instruction number being fetched

Definition at line 34 of file fetch.cpp.

void Fetch::PerformFetching (int & *p*, FileLoader *l*)

@ Function that simulates the fetching phase of a MIPS CPU

Parameters:

p	the instruction number being fetched
l	a **FileLoader** object used to access the instructions list

Definition at line 14 of file fetch.cpp.

void Fetch::setFetchPc (int *num*)

@ Function that sets the instruction number being fetched

Parameters:

num	the instruction number being fetched

Definition at line 29 of file fetch.cpp.

The documentation for this class was generated from the following files:

- **fetch.h**
- **fetch.cpp**

FileLoader Class Reference

`#include <fileloader.h>`

Public Member Functions

- **FileLoader** ()

Public Attributes

- std::vector< string > **Instructions**
- std::vector< **Instruction** > **FormatedInstructions**
- std::vector< int > **RegisterFile**
- std::vector< int > **Memory**

Friends

- istream & **operator>>** (istream &in, **FileLoader** &loader)
- ostream & **operator<<** (ostream &out, const **FileLoader** &shower)

Detailed Description

@ Class that parses the MIPS assembly code and stores the needed information into vectors also reads the register file and the memory

Definition at line 15 of file fileloader.h.

Constructor & Destructor Documentation

FileLoader::FileLoader ()

@ The default constructor that stores the filenames of the source code , of the register file and of the memory

Definition at line 11 of file fileloader.cpp.

Friends And Related Function Documentation

ostream& operator<< (ostream & *out*, const FileLoader & *shower*) `[friend]`

@ overloaded operator <<

Parameters:

out	output stream
shower	a **FileLoader** Object

Returns:

ostream &operator overloads the << operator

Definition at line 42 of file fileloader.cpp.

istream& operator>> (istream & *in*, FileLoader & *loader*) `[friend]`

@ overloaded operator >> for getting the simulators input

Parameters:

in	input stream
loader	a **FileLoader** Object

Returns:

istream &operator >> overloads the >> operator

Definition at line 21 of file fileloader.cpp.

Member Data Documentation

std::vector<Instruction> FileLoader::FormatedInstructions

stores the assembly code instructions after being transformed to the apropriate format

Definition at line 26 of file fileloader.h.

std::vector<string> FileLoader::Instructions

stores the assembly code instructions

Definition at line 25 of file fileloader.h.

std::vector<int> FileLoader::Memory

stores the memory values

Definition at line 28 of file fileloader.h.

std::vector<int> FileLoader::RegisterFile

stores the register file values

Definition at line 27 of file fileloader.h.

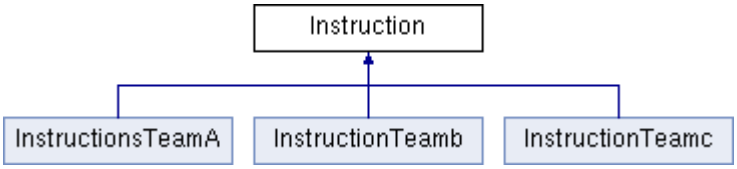
The documentation for this class was generated from the following files:

- **fileloader.h**
- **fileloader.cpp**

Instruction Class Reference

`#include <instruction.h>`

Inheritance diagram for Instruction:



Public Member Functions

- **Instruction** ()
- **Instruction** (string opcode, int r1, int r2, int rg, int imm, int L)
- string **getOpCode** ()
- void **setOpcode** (string s)
- void **setR1** (int num)
- int **getR1** ()
- void **setR2** (int num)
- int **getR2** ()
- void **setRg** (int num)
- int **getRg** ()
- void **setImm** (int num)
- int **getImm** ()
- void **setL** (int num)
- int **getL** ()

Detailed Description

@ Class that analyzes the instruction fields

Definition at line 12 of file instruction.h.

Constructor & Destructor Documentation

Instruction::Instruction ()

@ The default constructor that initializes an instruction as a "nop" instruction

Definition at line 6 of file instruction.cpp.

Instruction::Instruction (string *opcode*, int *r1*, int *r2*, int *rg*, int *imm*, int *L*)

@ A constructor that initializes an instruction as a non "nop" instruction

Parameters:

opcode	the instructions opcode
r1	the instructions r1 field
r2	the instructions r2 field
rg	the instructions rg field
imm	the instructions immediate operator field
L	the instructions jump adresse field

Definition at line 16 of file instruction.cpp.

Member Function Documentation

int Instruction::getImm ()

@ gets the Instructions immediate operator

Returns:

int

Definition at line 71 of file instruction.cpp.

int Instruction::getL ()

@ gets the Instructions jumping adresse

Returns:

int

Definition at line 81 of file instruction.cpp.

string Instruction::getOpCode ()

@ gets the Instructions operation code

Returns:

string

Definition at line 26 of file instruction.cpp.

int Instruction::getR1 ()

@ gets the Instructions r1 field

Returns:

int

Definition at line 41 of file instruction.cpp.

int Instruction::getR2 ()

@ gets the Instructions r2 field

Returns:

int

Definition at line 51 of file instruction.cpp.

int Instruction::getRg ()

@ sets the Instructions rg field

Returns:

int

Definition at line 61 of file instruction.cpp.

void Instruction::setImm (int *num*)

@ sets the Instructions immediate operator

Parameters:

num	

Definition at line 66 of file instruction.cpp.

void Instruction::setL (int *num*)

@ sets the Instructions jumping adresse

Parameters:

num	

Definition at line 76 of file instruction.cpp.

void Instruction::setOpcode (string *s*)

@ sets the Instructions operation code

Parameters:

s	

Definition at line 31 of file instruction.cpp.

void Instruction::setR1 (int *num*)

@ sets the Instructions r1 field

Parameters:

num	

Definition at line 36 of file instruction.cpp.

void Instruction::setR2 (int *num*)

@ sets the Instructions r2 field

Parameters:

num	

Definition at line 46 of file instruction.cpp.

void Instruction::setRg (int *num*)

@ sets the Instructions rg field

Parameters:

num	

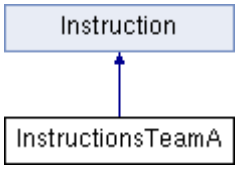
Definition at line 56 of file instruction.cpp.

The documentation for this class was generated from the following files:

- **instruction.h**
- **instruction.cpp**

InstructionsTeamA Class Reference

`#include <instructionsteama.h>`
Inheritance diagram for InstructionsTeamA:



Public Member Functions

- **InstructionsTeamA** ()
- **InstructionsTeamA** (int r1, int r2, int imm, string opcode)
- void **setR1** (int num)
- int **getR1** ()
- void **setR2** (int num)
- int **getR2** ()
- void **setImm** (int num)
- int **getImm** ()

Detailed Description

Definition at line 6 of file instructionsteama.h.

Constructor & Destructor Documentation

InstructionsTeamA::InstructionsTeamA ()

Definition at line 3 of file instructionsteama.cpp.

InstructionsTeamA::InstructionsTeamA (int *r1*, int *r2*, int *imm*, string *opcode*)

Definition at line 12 of file instructionsteama.cpp.

Member Function Documentation

int InstructionsTeamA::getImm ()

Definition at line 44 of file instructionsteama.cpp.

int InstructionsTeamA::getR1 ()

Definition at line 24 of file instructionsteama.cpp.

int InstructionsTeamA::getR2 ()

Definition at line 34 of file instructionsteama.cpp.

void InstructionsTeamA::setImm (int *num*)

Definition at line 39 of file instructionsteama.cpp.

void InstructionsTeamA::setR1 (int *num*)

Definition at line 19 of file instructionsteama.cpp.

void InstructionsTeamA::setR2 (int *num*)

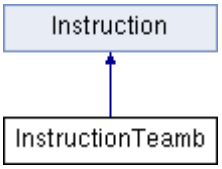
Definition at line 29 of file instructionsteama.cpp.

The documentation for this class was generated from the following files:

- **instructionsteama.h**
- **instructionsteama.cpp**

InstructionTeamb Class Reference

`#include <instructionteamb.h>`
Inheritance diagram for InstructionTeamb:



Public Member Functions

- **InstructionTeamb** ()
- **InstructionTeamb** (int rg, int r1, int imm, string opcode)
- void **setRg** (int num)
- int **getRg** ()
- void **setR1** (int num)
- int **getR1** ()
- void **setImm** (int num)
- int **getImm** ()

Detailed Description

Definition at line 6 of file instructionteamb.h.

Constructor & Destructor Documentation

InstructionTeamb::InstructionTeamb ()

Definition at line 3 of file instructionteamb.cpp.

InstructionTeamb::InstructionTeamb (int *rg*, int *r1*, int *imm*, string *opcode*)

Definition at line 12 of file instructionteamb.cpp.

Member Function Documentation

int InstructionTeamb::getImm ()

Definition at line 44 of file instructionteamb.cpp.

int InstructionTeamb::getR1 ()

Definition at line 34 of file instructionteamb.cpp.

int InstructionTeamb::getRg ()

Definition at line 24 of file instructionteamb.cpp.

void InstructionTeamb::setImm (int *num*)

Definition at line 39 of file instructionteamb.cpp.

void InstructionTeamb::setR1 (int *num*)

Definition at line 29 of file instructionteamb.cpp.

void InstructionTeamb::setRg (int *num*)

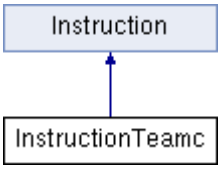
Definition at line 19 of file instructionteamb.cpp.

The documentation for this class was generated from the following files:

- **instructionteamb.h**
- **instructionteamb.cpp**

InstructionTeamc Class Reference

`#include <instructionteamc.h>`
Inheritance diagram for InstructionTeamc:



Public Member Functions

- **InstructionTeamc** ()
- **InstructionTeamc** (int rg, int r1, int r2, string opcode)
- void **setRg** (int num)
- int **getRg** ()
- void **setR1** (int num)
- int **getR1** ()
- void **setR2** (int num)
- int **getR2** ()

Detailed Description

Definition at line 6 of file instructionteamc.h.

Constructor & Destructor Documentation

InstructionTeamc::InstructionTeamc ()

Definition at line 3 of file instructionteamc.cpp.

InstructionTeamc::InstructionTeamc (int *rg*, int *r1*, int *r2*, string *opcode*)

Definition at line 12 of file instructionteamc.cpp.

Member Function Documentation

int InstructionTeamc::getR1 ()

Definition at line 34 of file instructionteamc.cpp.

int InstructionTeamc::getR2 ()

Definition at line 44 of file instructionteamc.cpp.

int InstructionTeamc::getRg ()

Definition at line 24 of file instructionteamc.cpp.

void InstructionTeamc::setR1 (int *num*)

Definition at line 29 of file instructionteamc.cpp.

void InstructionTeamc::setR2 (int *num*)

Definition at line 39 of file instructionteamc.cpp.

void InstructionTeamc::setRg (int *num*)

Definition at line 19 of file instructionteamc.cpp.

The documentation for this class was generated from the following files:

- **instructionteamc.h**
- **instructionteamc.cpp**

Memory Class Reference

`#include <memory.h>`

Public Member Functions

- **Memory** ()
- void **PerformMemory** (**FileLoader** l, int pc)

Detailed Description

@ This Class represents the memory access phase of a Mips CPU unit

Definition at line 10 of file memory.h.

Constructor & Destructor Documentation

Memory::Memory ()

@ The default constructor

Definition at line 12 of file memory.cpp.

Member Function Documentation

void Memory::PerformMemory (FileLoader *l*, int *pc*)

@ Function that performs a memory access phase for an instruction

Parameters:

l	
pc	the instruction number being accessing the memory

Definition at line 16 of file memory.cpp.

The documentation for this class was generated from the following files:

- **memory.h**
- **memory.cpp**

WriteBack Class Reference

`#include <writeback.h>`

Public Member Functions

- **WriteBack** ()
- void **PerformWriteRegisters** (**FileLoader** l, int pc)

Detailed Description

Definition at line 10 of file writeback.h.

Constructor & Destructor Documentation

WriteBack::WriteBack ()

@ A Class that simulates a Mips CPU write register files phase

Definition at line 12 of file writeback.cpp.

Member Function Documentation

void WriteBack::PerformWriteRegisters (FileLoader *l*, int *pc*)

@ Function that simulates the register file phase

Parameters:

l	
pc	

Definition at line 16 of file writeback.cpp.

The documentation for this class was generated from the following files:

- **writeback.h**
- **writeback.cp**