CSCE 2301/230: Digital Design I

Spring 2015

Project 2 – Option 2: mini MIPS CPU

Design Report

Seba Youssef ID: 900121233

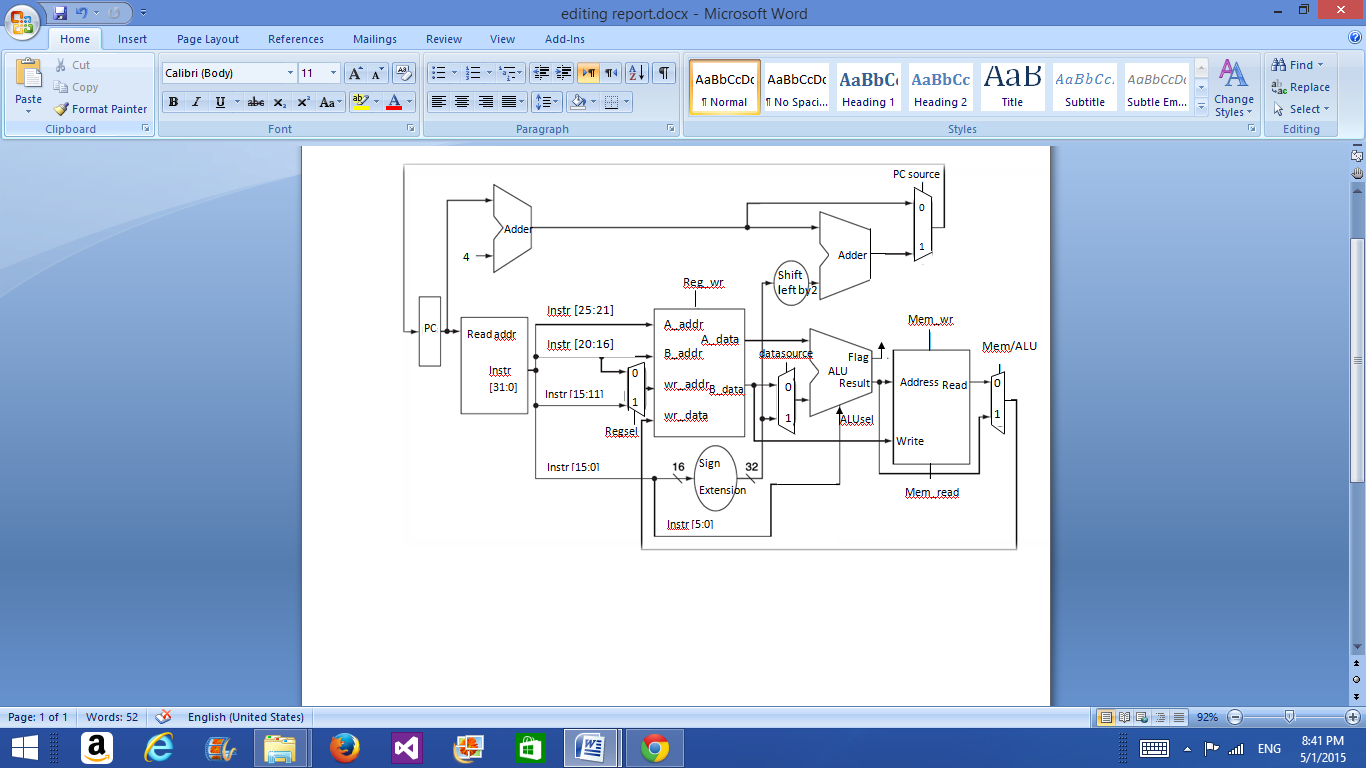
Nesma Badr ID: 900130606

**The following diagram illustrates the building blocks required to implement a single cycle 32-bit MIPS CPU. The top-level module involves the data path (ALU, register file, buses) and control unit.**

**The instructions supported by the CPU are:**

* **R-format instructions: Add, Sub, Or, And, Jr, Slt**
* **I-format instructions: Addi, Ori, Andi, Beq, Bne, Lw, Sw**
* **J-format instructions: J, Jal**

**Data path:**



1[Grab your reader’s attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]

Memtoreg

6

**J-Format Instructions + JR: (Data path & control unit):**

**Jump:**

* The signal is generated by the control unit when opcode = 000010.
* The program counter receives the signal and changes the PC to the J offset read, which is then passed to the memory to fetch the required instruction.

**JAL:**

* The signal is generated by the control unit when opcode = 000011.
* The program counter receives the signal and changes the PC to the J offset read, which is then passed to the memory to fetch the required instruction.
* When JAL is activated, the register file sets register 31 ($ra) to the current PC prior to update.

**JR:**

* The signal is generated by the control unit when opcode = 001000
* The program counter receives the value in register $ra (register 31) and sets the PC to this value. (PC = $ra + 1) The PC is then passed to the memory to fetch the next instruction.

**Control Unit:**

Mem\_wr

Mem\_read MemtoReg

Jump

Regsel

Reg\_wr

ALUsel

Datasource

Branch

Branch

Control Unit

Jal

Jr

Instr [31:26]

**Control signals:**

**The control unit generates the control signals based on the op-code received from the instruction word (Instr[31:26]). The control signals then determine the instruction to be implemented.**

- **"Regsel"** determines which register receives the value, it is the "rd" (bit#11- bit#15) in case of R-format instructions (op-code = 000000) and "rt" (bit#16- bit#20) in case of I-format instructions, except beq/bne.

-"**Reg\_wr**" is 0 when there's a J instruction, jr , beq, or bne and 1 otherwise.

-"**ALUsel**" is the "function" (bit#0-bit#5) field in the R format instructions, but in the I and J format instructions it is determined by the opcode (bit#26- bit #31) field.

-"**Datasource**" is 0 in the R-format instructions (with opcode 000000) while it is 1 in the I format instructions (excluding bne, beq). .

-"**Mem\_wr**" is 1 if it is a sw instruction, and 0 otherwise.

-"**Mem\_read"** is 1 if it is a lw instruction.

-"**MemtoReg**" is 1 in case of R-format, sw, lw

**Modules:**

**ALU:**

Arithmetic operations are performed in all formats except for J-format & JR instructions. It receives the opcode/function and selects the corresponding operation.

**Control Unit:**

Receives opcodes and function and produces signals based on the instruction format.

**ReadInstruction:**

Reads the 32 bit instruction word from the memory and splits it into the corresponding fields (rs,rt,rd,imm,joffset).

**Counter: (PC)**

Increments 1 to the current address in case of all formats except for J, Jr, JAL, beq, bne.

**Regfile:**

Initializes the registers with values. Register 1 is assigned ALUresult, to be displayed on the board. Register 31 stores the address of Jal in case of jal/jr instructions.

**SignExtention:**

Extends the 16 bit immediate to 32 bits.

**Counter2:**

Counter used by the display, clock divider & decoder.

**BinaryDecoder:**

It selects which digit to be displayed.

**7-segment Display:**

It receives the ALU result, and displays the least significant 16 bits as hexadecimal.

**Memory1 (ROM):**

A 32-bit ROM is used to store the instruction words

**Memory2 (Simple Dual port RAM):**

A 32 bit RAM is used to support the load and store word instructions. It receives the signal generated by the control unit and correspondingly reads or writes to memory.