Assignment No. 7

Computer Organisation - CS220

Problem Statement: We want to create a new processor CSE-BUBBLE that has the instruction set architecture as shown in Tab.1 below. We assume that the processor word size and instruction size are 32 bits and we use the VEDA memory as implemented in Assignment 4. The VEDA memory has two parts: a) Instruction Memory, b) Data Memory. You have to make sure that these two sections do not overlap. We also assume that RISC-BUBBLE has total 32 registers of which certain registers follow the same roles as in MIPS-32 ISA. For example, Program Counter (PC) register holds the address of the next instruction.

The target is to create the op-code formats for the given ISA, shown in Table 1, decide upon the data path elements (such as addition, subtraction, shift, jump) and then develop different verilog modules for the same along with the finite state machine for the control path.

Finally we shall build a single-cycle instruction execution unit for CSE-BUBBLE.

Table 1: Instruction Set for CSE-BUBBLE

Class	Instruction	Meaning
Arithmetic	add r0, r1, r2	r0=r1+r2
	sub r0, r1, r2	r0=r1-r2
	addu r0, r1, r2	r0=r1+r2 (unsigned addition, not 2's complement)
	subu r0,r1,r2	r0=r1-r2 (unsigned addition, not 2's complement)
	addi r0,r1,1000	r0=r1+1000
	addiu r0,r1, 1000	r0=r1+1000 (unsigned additon, not 2's complement)
Logical	and r0,r1,r2	r0= r1 & r2
	or r0,r1,r2	r0=r1 | r2
	andi r0,r1, 1000	r0= r1 & 1000
	ori r0,r1, 1000	r0= r1 | 1000
	sll r0, r1, 10	r0=r1<<10 (shift left logical)
	srl r0, r1, 10	r0=r1>>10 (shift right logical)
Data transfer	lw r0,10(r1)	r0=Memory[r1+10] (load word)
	sw r0,10(r1)	Memory[r1+10]=r0 (store word)
Conditional Branch	beq r0,r1,10	if(r0==r1) go to PC+4+10 (branch on equal)
	bne r0,r1,10	if(r0!=r1) go to PC+4+10 (branch on not equal)
	bgt r0,r1,10	if(r0>r1) go to PC+4+10 (branch if greater than)
	bgte r0,r1, 10	if(r0>=r1) go to PC+4+10 (branch if greter than or equal)
	ble r0,r1, 10	if(r0<r1) go to PC+4+10 (branch if less than)
	bleq r0,r1, 10	if(r0<=r1) go to PC+4+10 (branch if less than or equal)
Unconditional Branch	j 10	jump to address 10
	jr r0	jump to address stored in register r0
	jal 10	ra=PC+4 and jump to address 10
Comparison	slt r0,r1,r2	if(r1<r2) r0=1 else r0=0
	slti 1,2,100	if(r1<100) r0=1 else r0=0

Processor Development Strategy:

Now to do the project, we shall follow the below-mentioned steps and every step should be documented in the project report.

1. **[PDS1]** Decide the registers and their usage protocol.
2. **[PDS2]** Decide upon the size for instruction and data memory in VEDA.

3. **[PDS3]** Design the instruction layout for R-, I- and J-type instructions and their respective encoding methodologies.
4. **[PDS4]** Now design and implement an instruction fetch phase where the instruction next to be executed will be stored in the instruction register.
5. **[PDS5]** Next design and implement a module for instruction decode to identify which data path element to execute given the opcode of the instruction.
6. **[PDS6]** Design and implement the Arithmetic Logic Unit (ALU) in top-down approach to develop different modules for different types of instructions. There might be some hardware parts in the ALU that can be shared by different modules if required. This will reduce the footprint of the hardware.
7. **[PDS7]** Design and implement the branching operation along the ALU implementation.
8. **[PDS8]** Design the finite state machine for the control signals to execute the processor. Please ensure that every instruction should be executed in single clock cycle. Finally write the test benches to simulate the CSE-BUBBLE.
9. **[PDS9]** Develop the MIPS code for *Bubble Sort*. Then convert it into machine code following the ISA given above.
10. **[PDS10]** Store the machine code in the instruction memory and execute CSE-BUBBLE. Store the output of the bubble sort in the data memory.

Project Progress Plan:

- **Week of 20th March and 22rd March:** PDS1, PDS2, PDS3, PDS4, PDS5.
- **Week of 27th March and 29rd March:** PDS6, PDS7
- **Week of 3rd April and 5th April:** PDS8
- **Week of 10th Arpil and 12th April:** PDS9 and PDS10.

Submission Instruction:

- Create a zip file that will contain the verilog modules, test benches and the documentation or report in PDF format for the assignment. The name of the zip file will be $Assignment7\_RollNo1\_RollNo2.zip$.
- In every verilog source files and the testbench you should clearly mention your name and roll no.
- No hard-coding is allowed.
- It is recommended that you add comments in the code to increase readability.
- The TAs will be checking the progress of the project every week.
- If any group does not upload their codes, they will not get any marks even if they show the output in the lab.
- Do not copy from other group. You will get zero.
- The deadline to upload the code in HelloIITK is: 13th April, 2023.