CSE 331/503

Computer Organization

Final Project: 16-bit MIPS Single Cycle

Due Date: Jan 18 2023, 17:00

In this assignment you will design a 16-bit MIPS processor. It is same as the 32bit MIPS but the register contents are 16 bits and there are 16 registers. Therefore, each register address is 4-bit.

Opcode (6-bits)	Rs (4-bits)	Rt (4-bits)	Rd (4-bits)	Shamt (4-bits)	Func (6-bits)	0000

R-type

Opcode (6-bits)	Rs (4-bits)	Rt (4-bits)	Immediate Field (16-bits)	00

I-type

Opcode (6-bits)	Addr (10-bits)	0000 0000 0000 0000

J-type

1. Each instruction is 32-bits.

2. As you have 16 bits of register contents. You can implement a li $rt, imm instruction which can directly load a 16-bit number into $rt. Implementation of li is a MUST.

3. A word is 16-bits for that MIPS.

4. The Data Memory will hold 2^16 words of data therefore jump instructions can directly give the word address with 16-bits.

5. There will be no byte access therefore do not implement lb, sb instructions.

6. Instruction Memory holds 1024 instructions, each of which is 32-bits. Therefore a PC of 10-bits is enough. Or you can use the least significant 10 bits of PC.

7. You will verify the design using a Verilog testbench and simulation. Write a small assembler so that you can convert instructions to machine code. This is needed for simulations.

8. There will be problem sessions for the assignment. In these sessions you will learn to design a 32-bit MIPS taught in the course. Therefore, if you follow them, it will be easier for you to complete that project. If you do not follow them the project will be harder. That's your choice. There will be no attendance.

9. This project will have a higher contribution in your total grade than the previous ones. Start today than it is easy. Start at the last 3 days before submission, then it is very hard.

10. Late submission is allowed for 3 days. Each day's degradation is 25pts.

11. Your TA can introduce some new requirements for submission. Follow the announcements.

12. This project is not described here in every detail. For full details attend the PS for that assignment.

13. The instructions you MUST implement for full points are as follows. Mult instruction brings 25 BONUS points.:

add	`add $1,$2,$3`	$1=$2+$3	
subtract	`sub $1,$2,$3`	$1=$2-$3	
add immediate	`addi $1,$2,100`	$1=$2+100	"Immediate" means a constant number
load word	`lw $1,100($2)`	$1=Memory[$2+100]	Copy from memory to register
store word	`sw $1,100($2)`	Memory[$2+100]=$1	Copy from register to memory
branch on equal	`beq $1,$2,100`	if($1==$2) go to PC+4+100	Test if registers are equal
branch on not equal	`bne $1,$2,100`	if($1!=$2) go to PC+4+100	Test if registers are not equal
set on less than	`slt $1,$2,$3`	if($2<$3)$1=1; else $1=0	Test if less than. If true, set $1 to 1. Otherwise, set $1 to 0.
set on less than immediate	`slti $1,$2,100`	if($2<100)$1=1; else $1=0	Test if less than. If true, set $1 to 1. Otherwise, set $1 to 0.
jump	`j 1000`	go to address 1000	Jump to target address
jump register	`jr $1`	go to address stored in $1	For switch, procedure return

jump and link	`jal 1000`	$ra=PC+4; go to address 1000	Use when making procedure call. This saves the return address in $ra
and	`and $1,$2,$3`	$1=$2&$3	Bitwise AND
or	`or $1,$2,$3`	$1=$2|$3	Bitwise OR
and immediate	`andi $1,$2,100`	$1=$2&100	Bitwise AND with immediate value
or immediate	`ori $1,$2,100`	$1=$2|100	Bitwise OR with immediate value
shift left logical	`sll $1,$2,10`	$1=$2<<10	Shift left by constant number of bits
shift right logical	`srl $1,$2,10`	$1=$2>>10	Shift right by constant number of bits