PROGRAMMING PROJECT 2

Due: 29/05/2020 23:00

In this project, you are required to extend the MIPS single-cycle implementation by implementing additional instructions. You will use ModelSim simulator [1] to develop and test your code. You are required to implement your assigned **6 instructions** (selected from following 28 instructions). *Note that your set of* ***6 instructions*** *will be emailed to you or your group member.*

R-format (11): balrn, balrz, sll, srl, sllv, srlv, jmadd, jmor, jmsub, jalr, jr

I-format (13): xori , andi, ori, bneal, bgez, bgezal, bgtz, blez, bltzal, bltz, jrs, jrsal, jpc

J-format (4): baln, balz, balv, jal

You must design the revised single-cycle datapath and revised control units which make a processor that executes your instructions as well as the instructions implemented already in the design. After designing the new enhanced processor, you will implement it in Verilog HDL.

Our instructions

2. andi	I-type opcode=12	andi $rt, $rs, Label	Put the logical AND of register $rs and the zero-extended immediate into register $rt.
4. bneal	I-type opcode=45	bneal $rs, $rt, Target	if R[rs] != R[rt], branches to PC-relative address (formed as beq & bne do), link address is stored in register 31
14. jmor	R-type funct=37	jmor $rs,$rt	jumps to address found in memory [$rs\|$rt], link address is stored in $31
16. jalr	R-type func=9	jalr $rs, $rd	Unconditionally jump to the address found in register $rs, link address is stored in $rd.
18. balv	J-type opcode=33	balv Target	if Status [V] = 1, branches to pseudo-direct address (formed as jal does), link address is stored in register 31
28. srlv	R-type func=6	srlv $rd, $rt, $rs	shift register $rt to right by the value in register $rs, and store the result in register $rd.

Status Register

Some of the conditional branches test the Z and N bits in the Status register. So the MIPS datapath will need to have a Status register, with the following 3 bits: Z (if the ALU result is zero) , N (if the ALU result is negative) or V (if the ALU result causes overflow). The Status register will be loaded with the ALU results each clock cycle.

instruction	syntax	jump address	link address
jmor	jmor $rs,$rt	PC ← M[$rs\|$rt]	---
jalr	jalr $rs, $rd	PC ← $rs	R[rd] ← PC + 4

SRLV -- ***Shift right logical variable***

Description:	Shifts a register value right by the amount specified in $s and places the value in the destination register. Zeroes are shifted in.
Operation:	$d = $t >> $s; advance_pc (4);
Syntax:	srlv $d, $t, $s
Encoding:	`0000 00ss sstt tttt dddd d000 0000 0110`

balz	balz Target	if Status [Z] = 1	Pseudo-direct address PC ← PC[31:28]||[25:0]||00	R[31] ← PC + 4

balv olursa yalnızca status[V]=1 --> V=overflow olursa branch ve link

bneal-->registerlar eşit değilse +link yap