Objectives. Learn how to enhance the datapath and control for a multicycle implementation of a simple ALU.

Put all your answers on these two sheets. The assignment is due at the beginning of class on February 26.

1. Modify the datapath and control for the multicycle implementation to add the jal (jump and link) instruction. Remember that the jal instruction is like a j (jump) instruction, but it also places the address of the instruction following the jal instruction in register $31$ as a return address. Examine the figures below and on the following page showing the datapath and the finite state machine (FSM) control for the multicycle implementation, respectively. Make the appropriate modifications to both the datapath and the FSM control figures to support the jal instruction. (100 points)
Instruction fetch

Instruction decode/ register fetch

Start

MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00

ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00

ALUSrcA = 0
ALUSrcB = 10
ALUOp = 10

ALUSrcA = 1
ALUSrcB = 00
ALUOp = 10

PCWrite
PCSource = 10

ALUSrcA = 0
ALUSrcB = 1
ALUOp = 00

RegDst = 1
RegWrite
MemtoReg = 0

Write-back step

MemRead
IorD = 1

MemWrite
IorD = 1

RegDst = 0
RegWrite
MemtoReg = 1

Memory address computation

(Op = 'LW') or (Op = 'SW')

Execution

Branch completion

Jump completion

Memory access

Memory access

R-type completion