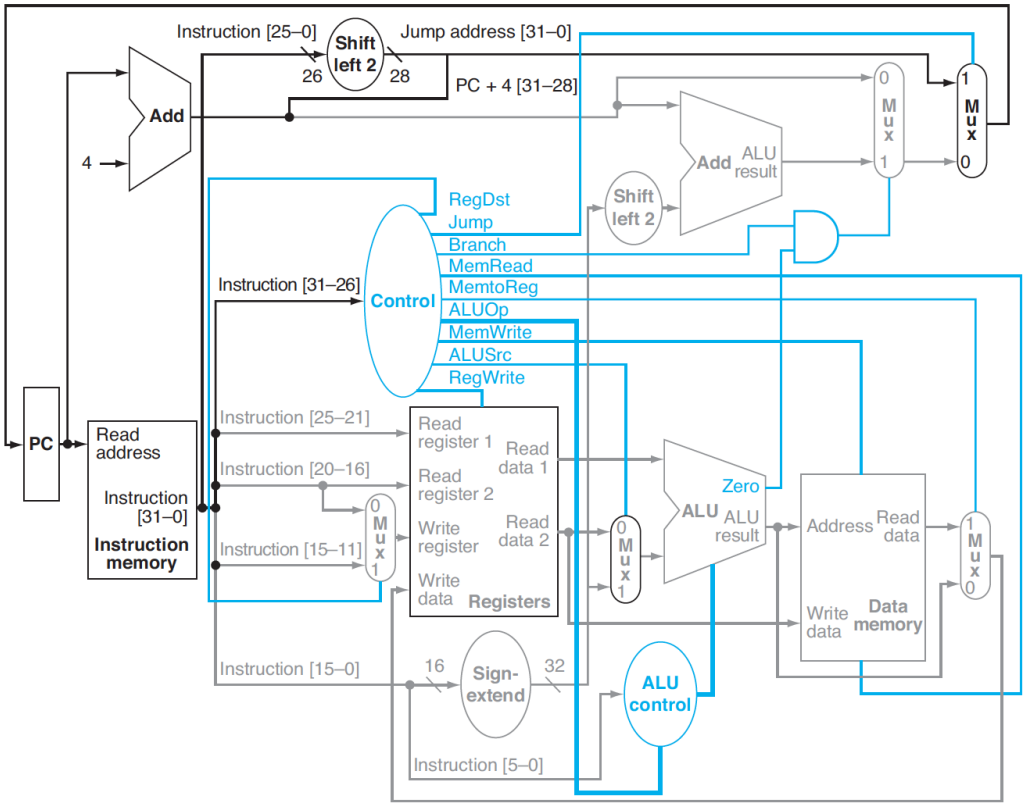
CS154 Computer Architecture
Winter Term 2019

Lab Assignment 03

Consider the following data path and the set of 9 instructions of the single-cycle processor:

- The memory-reference instructions `lw` (load word) and `sw` (store word)
- The arithmetic-logical instructions `add`, `sub`, `AND`, `OR`, and `slt`
- The instructions `beq` (branch equal) and `j` (jump)

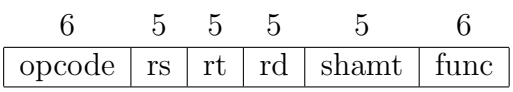
In the following pages you will be asked to add new instructions to the processor and make appropriate changes on the data path in order to support the existing 9 instructions and the new ones. You need to clearly specify new functionality on the existing units (such as ALU) and/or new components you may need (MUXes, control lines, gates, etc), and draw them on the data path figure.



Completely specify the logic values of the existing control signals (RegDst, Branch, MemRead, etc) and of the new ones you may need to introduce for the new instruction. Consider each new instruction independently from the other new instructions. Make copies of this figure as needed. See the 1-page large figure at the end.

1. Add the "Nor" instruction to the single cycle processor.
   The opcode: 0_{hex}; The function: 27_{hex}

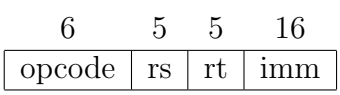
`nor rd, rs, rt`



R[rd] ← (R[rs] | R[rt])′
PC ← PC + 4

2. Add the "Add Immediate" instruction to the single cycle processor.
   The opcode: 8_{hex}

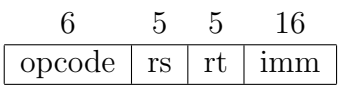
`addi rt, rs, imm`



R[rt] ← R[rs] + SignExtImm
PC ← PC + 4

3. Add the "Set Less Than Immediate" instruction to the single cycle processor.
   The opcode: A_{hex}

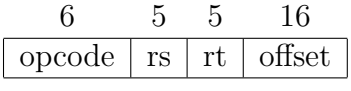
`slti rt, rs, imm`



if R[rs] < SignExtImm
    R[rt] ← 1
else
    R[rt] ← 0
PC ← PC + 4

4. Add the "Branch On Not Equal" instruction to the single cycle processor.
   The opcode: 5_{hex}

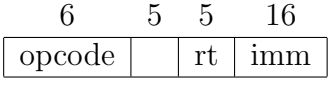
`bne rt, rs, label`



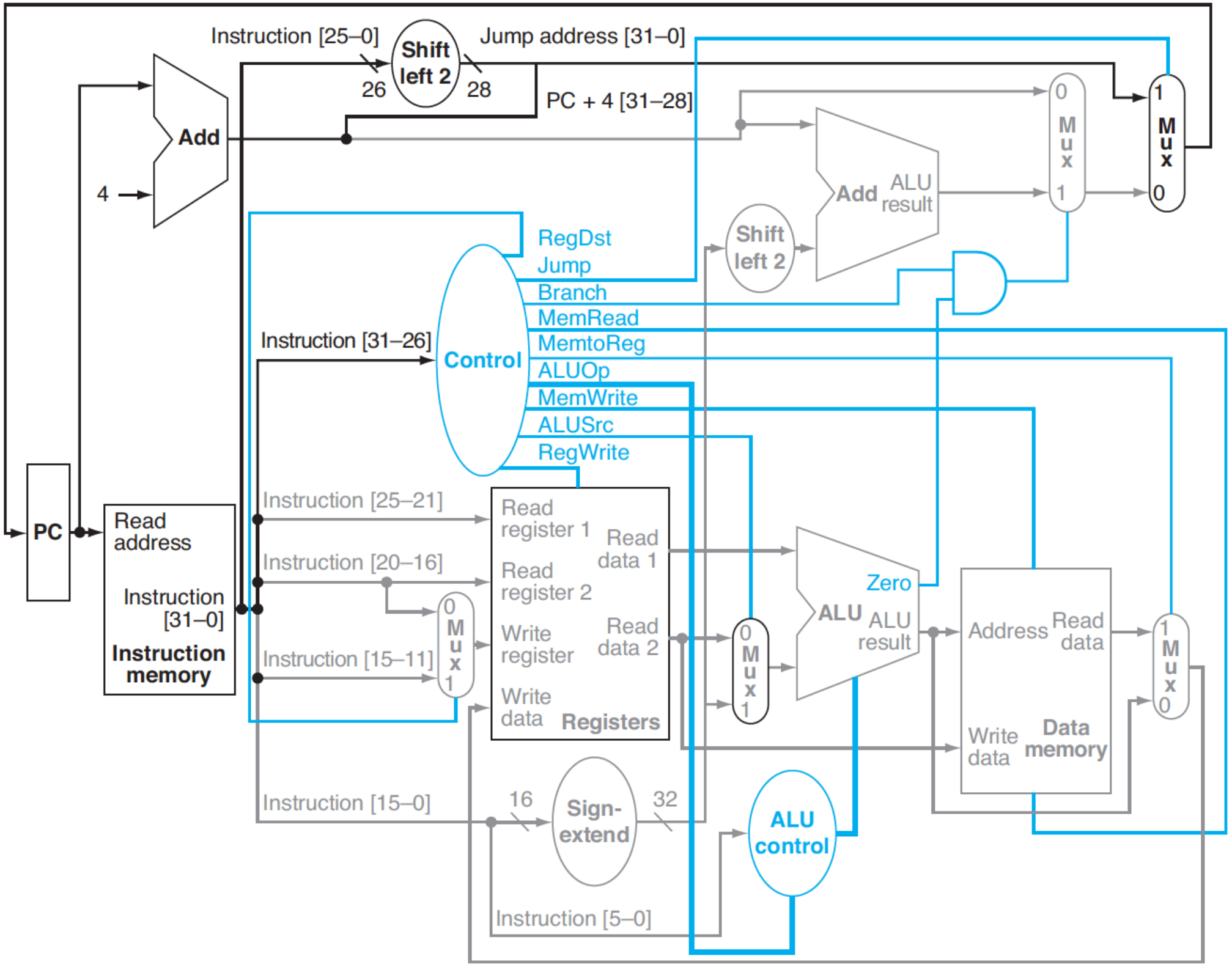
if R[rs] ≠ R[rt]
    PC ← PC + 4 + SignExtOffset
else
    PC ← PC + 4

5. Add the "Load Upper Immediate" instruction to the single cycle processor.
   The opcode: F_{hex}

`lui rt, imm`



R[rt] ← Imm16 || 0000_{hex}
PC ← PC + 4


Instruction [25–0]
Shift left 2
Jump address [31–0]
26
28
PC + 4 [31–28]
Add
4
0
M
u
x
1
1
M
u
x
0
Add
ALU result
Shift left 2
RegDst
Jump
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Instruction [31–26]
Control
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
16
Sign-extend
32
ALU
Zero
ALU result
ALU control
Address
Read data
Write data
Data memory