CS 61C: Great Ideas in Computer Architecture (Machine Structures)
Control Implementation

Instructors:
Randy H. Katz
David A. Patterson
http://inst.eecs.Berkeley.edu/~cs61c/fa10

Agenda
• Datapath Control
• Administrivia
• Technology Break
• Controller Implementation

Register-Register Timing:
One Complete Cycle

Datapath's Control Signals
• ExtOp: “zero”, “sign”
• ALUsrc: 0 ⇒ regB; 1 ⇒ immed
• ALUctr: “ADD”, “SUB”, “OR”
• MemWr: 1 ⇒ write memory
• MemtoReg: 0 ⇒ ALU; 1 ⇒ Mem
• RegDst: 0 ⇒ “rt”; 1 ⇒ “rd”
• RegWr: 1 ⇒ write register

Single Cycle Datapath during Store
• Data Memory [R[rs] + SignExt[imm16]] = R[rt]

Agenda
• Datapath Control
• Administrivia
• Technology Break
• Controller Implementation
Single Cycle Datapath during Store

- Data Memory \( R[rs] \text{ + SignExt}(imm16) = R[rt] \)

egin{itemize}
  
  
  - \text{Instr fetch unit}

  
  
  - \text{MemoryWrite=0}

\end{itemize}

Single Cycle Datapath during Branch

- If \( (R[rs] = R[rt]) = 0 \), then Zero = 1; else Zero = 0

egin{itemize}
  
  
  - \text{Instr fetch unit}

  
  
  - \text{MemoryWrite=0}

\end{itemize}

Instruction Fetch Unit at the End of Branch

- If \( (Zero = 1) \) then \( \text{PC} = \text{PC} + 4 + \text{SignExt}(imm16) \times 4 \); else \( \text{PC} = \text{PC} + 4 \)

Single Cycle Datapath during Jump

- New PC = \( \{ \text{PC}[31..28], \text{target address, 00} \} \)

Single Cycle Datapath during Jump

- New PC = \( \{ \text{PC}[31..28], \text{target address, 00} \} \)
How do we modify this to account for jumps?

**Agenda**

- Datapath Control
- Administrivia
- Technology Break
- Controller Implementation

**Query**
- Can Zero still get asserted?
- Does nPC_sel need to be 0?
- If not, what?

**Given Datapath: RTL → Control**
Summary of the Control Signals (1/2)

- Add = rtype
- Sub = rtype
- Ori = ~op
- Lw = op
- Ori = ~op
- Rtype = ~op

Where:
- ALUctr[1] = or
- ALUctr[0] = sub + beq (assume ALUctr is 00 ADD, 01 SUB, 10 OR)

Jump = jump
nPCsel = beq
MemWrite = sw
RegWrite = add + sub + ori + lw
ALUSrc = ori + lw + sw
RegDst = add + sub

Boolean Expressions for Controller

RegSrc = add + sub
ALUsrc = ori + lw + sw
MemWrite = lw
nPCsel = beq
Jump = jump
ExitIp = lw + sw
ALUctr[0] = or

Where:
- rtype = ~op, ~op, ~op, ~op, ~op, ~op
- ori = ~op, ~op, ~op, ~op, ~op, ~op
- lw = ~op, ~op, ~op, ~op, ~op, ~op
- beq = ~op, ~op, ~op, ~op, ~op, ~op
- Jump = ~op, ~op, ~op, ~op, ~op, ~op
- add = rtype + func, +func, +func, +func, +func, +func
- sub = rtype + func, +func, +func, +func, +func, +func

Controller Implementation

How do we implement this in gates?

Summary: Single-cycle Processor

- Five steps to design a processor:
  1. Analyze instruction set & datapath requirements
  2. Select set of datapath components & establish clock methodology
  3. Assemble datapath meeting the requirements
  4. Analyze implementation of each instruction to determine setting of control points that affects the register transfer.
  5. Assemble the control logic
     - Formulate logic equations
     - Design circuits