Datapath and control:
Single-cycle implementation, part 2

CS207, Fall 2004
October 29, 2004

MIPS datapath implementation


ALU control unit: output values
(control lines to ALU)

- 0000: AND
- 0001: OR
- 0010: add
  - addition, lw, sw
- 0110: sub
  - subtraction, branching
- 0111: slt
- (1110: NOR)

ALU control unit: input values

- Function field of instruction (6 bits)
- ALUOp: generated by main control unit
  - 00: lw, sw
  - 01: sub
  - 10: use function field
- Note: Multiple control levels simplifies the control unit design (increases speed!)
Main control unit: input signals

- 6-bit opcode from instruction
- R-format: 000000
  - asserted outputs: RegDst, RegWrite, ALUOp1
- lw: 100011
  - asserted outputs: ALUSrc, MemtoReg, RegWrite, MemRead
- sw: 101011
  - asserted outputs: ALUSrc, MemWrite
  - don't cares: RegDst, MemtoReg
- beq: 000100
  - asserted outputs: Branch, ALUOp0
  - don't cares: RegDst, MemtoReg

Main control unit: output signals

- RegDst
  - 0 = Write register is in rt field (R-format instruction)
  - 1 = Write register is in rd field (lw instruction)
- ALUSrc
  - 0 = Second operand is a register
  - 1 = Second operand is “immediate”
- MemtoReg
  - 0 = Data to write to register is from ALU
  - 1 = Data to write to register is from memory

Datapath: R-type instructions

Datapath: {lw, sw}

Datapath: \textit{beq}

Jump instruction

- Target address
  - upper 4 bits of (PC+4)
  - 26-bit immediate field of instruction
  - 00 (b/c immediate field is a word offset)
- requires additional multiplexor and additional control signal (set when opcode = 2)

Datapath: \textit{jump}

Note: The output of the AND gate should go to the first MUX in the top right corner, not the second.

Problems with single-cycle implementation

- All instructions have CPI of 1
  - not practical!
  - longest instruction dictates clock cycle time
- Cannot make common case fast b/c we are constrained by worst-case delays
- Each element can be used only once per clock
  - duplicate elements needed