

Control Unit (single cycle implementation)

- Control unit sends control signals to data path and memory depending
 - on the opcode (and function field)
 - results in the ALU (for example for Zero test)
- These signals control
 - muxes; read/write enable for registers and memory etc.
- Some “control” comes directly from instruction
 - register names
- Some actions are performed at every instruction so no need for control (in this single cycle implementation)
 - incrementing PC by 4; reading instr. memory for fetching next inst.

4/20/2004

CSE378 Control unit
Single cycle impl.

1

Building the control unit

- Decompose the problem into
 - Data path control (register transfers)
 - ALU control
- Setting of control lines by control unit totally specified in the ISA
 - for ALU by opcode + function bits if R-R format
 - for register names by instruction (including opcode)
 - for reading/writing memory and writing register by opcode
 - muxes by opcode
 - PC by opcode + result of ALU

4/20/2004

CSE378 Control unit
Single cycle impl.

2

Example

- Limit ourselves to:
 - R-R instructions: add, sub, and, or, slt –
 - OPCODE = 0 but different function bits
 - Load-store: lw, sw
 - Branch: beq
- ALU control
 - Need to add for: add, lw, sw
 - Need to sub for: sub, beq
 - Need to and for :and
 - Need to or for :or
 - Need to set less than for : slt

4/20/2004

CSE378 Control unit
Single cycle impl.

3

ALU Control

- ALU control: combination of opcode and function bits
- Decoding of opcodes yields 3 possibilities hence 2 bits
 - ALUOp1 and ALUOp2
- ALU control:
 - Input 2 ALUOp bits and 6 function bits
 - Output one of 5 possible ALU functions
 - Of course lots of don't care for this *very* limited implementation

4/20/2004

CSE378 Control unit
Single cycle impl.

4

Implementation of Overall Control Unit

- Input: opcode (and function bits for R-R instructions)
- Output: setting of control lines
- Can be done by logic equations
- If not too many, like in RISC machines
 - Use of PAL's (cf. CSE 370).
 - In RISC machines the control is “hardwired”
- If too large (too many states etc.)
 - Use of microprogramming (a microprogram is a hardwired program that interprets the ISA)
- Or use a combination of both techniques (Pentium)

4/20/2004

CSE378 Control unit
Single cycle impl.

5

Where are control signals needed (cf. Figure 5.17)

- Register file
 - RegWrite (Register write signal for R-type, Load)
 - RegDst (Register destination signal: rd for R-type, rt for Load)
- ALU
 - ALUSrc (What kind of second operand: register or immediate)
 - ALUOp (What kind of function: ALU control for R-type)
- Data memory
 - MemRead (Load) or MemWrite (Store)
 - MemtoReg (Result register written from ALU or memory)
- Branch control
 - PCSrc (PC modification if branch is taken)

4/20/2004

CSE378 Control unit
Single cycle impl.

6

How are the control signals asserted (cf. Fig 5.19)

- Decoding of the opcode by control unit yields
 - Control of the 3 muxes (RegDst, ALUSrc, MemtoReg): 3 control lines
 - Signals for RegWrite, Memread, Memwrite: 3 control lines
 - Signals to activate ALU control (e.g., restrict ourselves to 2)
 - Signal for branch (1 control line)
 - decoding of opcode ANDed with ALU zero result
- Input Opcode: 6 bits
- Output 9 control lines (see Figure 5.27)