


Administrative Matters

- Homework #4
 - Due November 5th
 - Watch out for "regular printing" vs. "revised printing" issue
- Reading Assignment
 - Chapter 5
- Extra Credit Quiz #1
 - Wednesday, November 7th
 - Take about 10 minutes
 - 5 short answer questions
 - Cover chapter 5 lecture and homework 4
 - Worth extra 5 quiz points!
- Midterm #2
 - Monday, November 19th
 - Cover chapter 5, 6, and a little bit 7
 - Cover homework 4 and 5
 - 15% of your grade



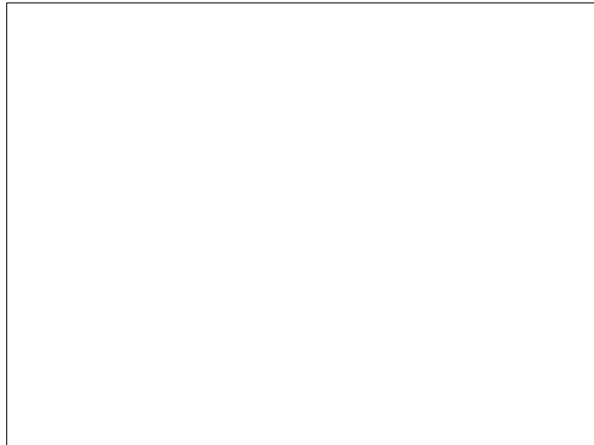
©2004 Morgan Kaufmann Publishers

Chapter Five

The Processor: Datapath and Control

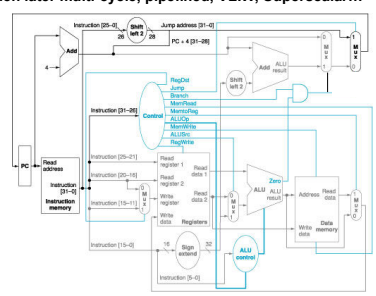
(continue)

©2004 Morgan Kaufmann Publishers



Where are we headed?

- Single-cycle implementation
 - Then later multi-cycle, pipelined, VLIW, Superscalar...

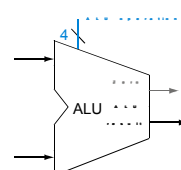


©2004 Morgan Kaufmann Publishers

Control

- ALU control input

0000	AND
0001	OR
0010	add
0110	subtract
0111	set-on-less-than
1100	NOR
- lw, sw
 - add
- R-type instruction
 - AND, OR, subtract, add, slt, nor
- beq
 - subtract

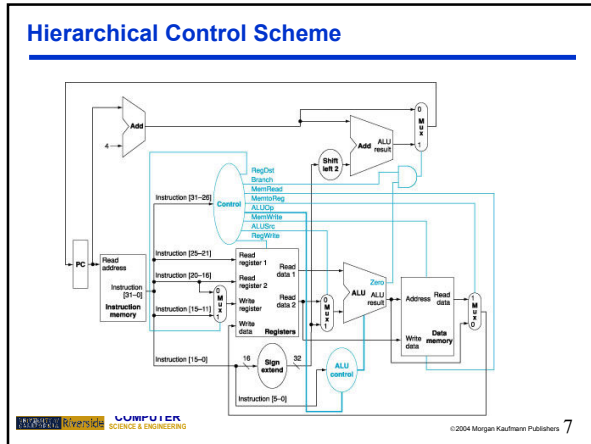


©2004 Morgan Kaufmann Publishers

Control

- Must describe hardware to compute 4-bit ALU control input
 - given instruction type
 - 0 = lw, sw
 - 01 = beq, arithmetic
 - 10 = arithmetic
 - Main control generate ALUop
 - ALUop computed from instruction type
 - Another ALU-control takes ALUop and function code
 - To generate actual 4 bit control

©2004 Morgan Kaufmann Publishers



Hierarchical Control

- Main-control: opcode -> ALUop
- ALU-control: ALUOp&Funct -> ALU control input

Instruction opcode	ALUop	Instruction operation	Funct field	Desired ALU action	ALU control input
Lw	00	Load word	xxxxxx	Add	0010
Sw	00	Store word	xxxxxx	Add	0010
Beq	01	Branch equal	xxxxxx	Subtract	0110
R-type	10	Add	100000	Add	0010
R-type	10	Subtract	100010	Subtract	0110
R-type	10	AND	100100	And	0000
R-type	10	OR	100101	Or	0001
R-type	10	Set on less than	101010	Set on less than	0111

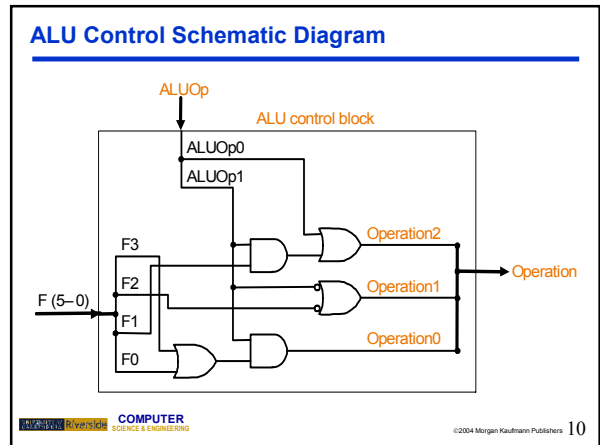
©2004 Morgan Kaufmann Publishers 8

ALU-Control

- Describe it using a truth table (can turn into gates):
- 11 is not used
 - Can be treated as don't care
- 00 (lw,sw) is always add (0010)
- 01 (beq) is always sub (0110)
 - Optimized to x1
- 10 (arithmetic) will depend on the last 4 bits of funct

ALUOp	ALUOp0	F5	F4	F3	F2	F1	F0	Operation
0	0	X	X	X	X	X	X	0010
X	1	X	X	X	X	X	X	0110
1	X	X	X	0	0	0	0	0010
1	X	X	X	0	0	1	0	0110
1	X	X	X	0	1	0	0	0000
1	X	X	X	0	1	0	1	0001
1	X	X	X	1	0	1	0	0111

©2004 Morgan Kaufmann Publishers 9

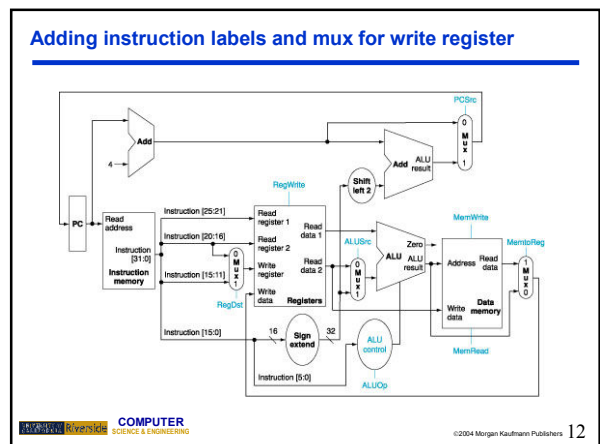


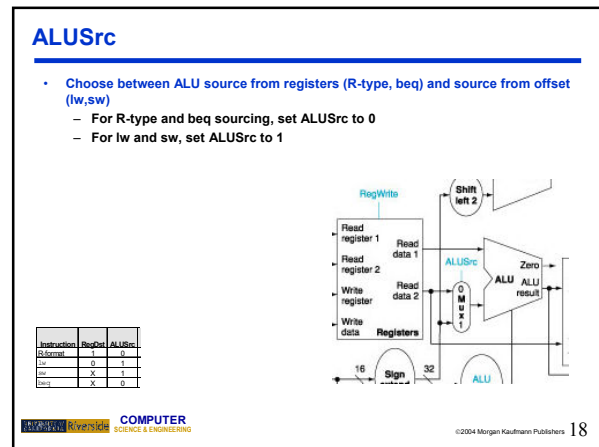
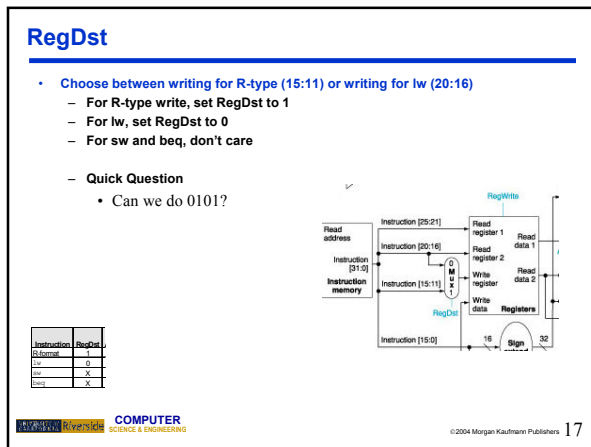
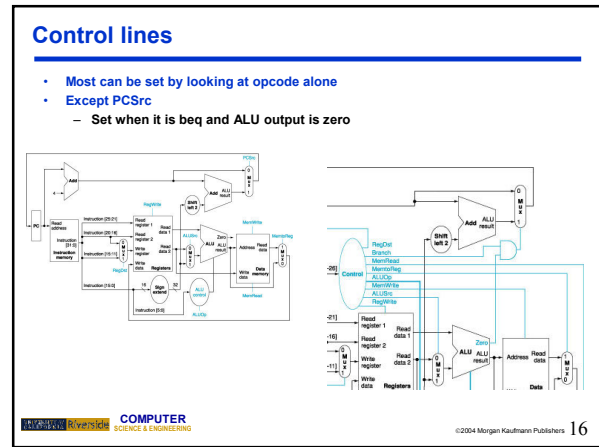
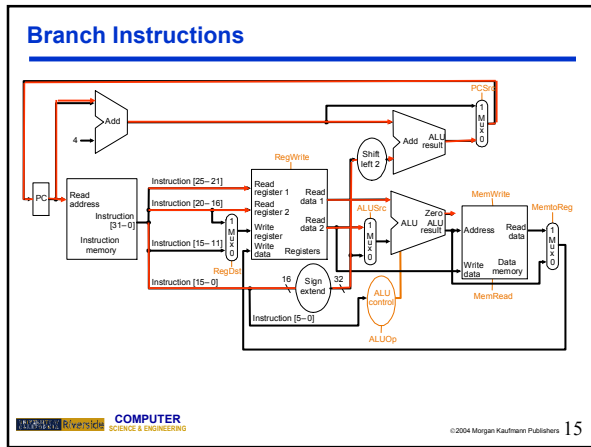
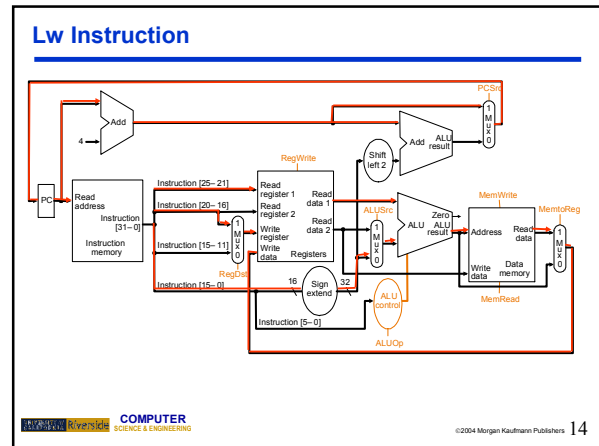
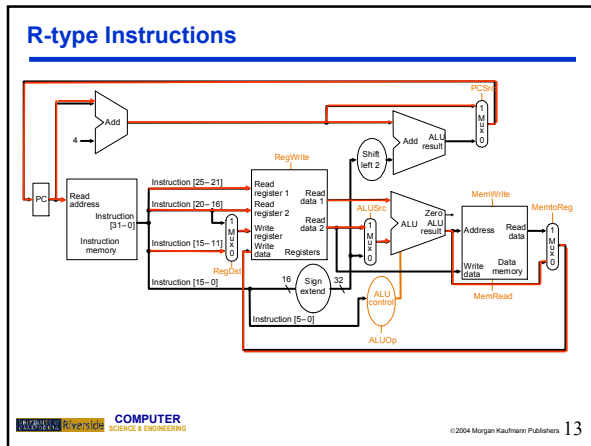
Main Control

- What kind of instructions do we have
 - Opcode is always 31:26
 - Registers to read (rs, rt) are always 25:21 and 20:16
 - Based register is always 25:21
 - 16 bit off-set is always at 15:0
 - Destination register is either 15:11 or 20:16
 - Need muxing

R-Type	rs	rt	rd	shamt	funct
0	rs	rt	rd	shamt	funct
31:26	25:21	20:16	15:11	10:6	5:0
lw, sw	rs	rt	addr		
35 or 43	25:21	20:16	15:0		
beq					
4	rs	rt	addr		
31:26	25:21	20:16	15:0		

©2004 Morgan Kaufmann Publishers 11





MemtoReg

- Choose between regfile source from ALU (R-type) or data memory (lw)
 - For R-type, set MemtoReg to 0
 - For lw, set MemtoReg to 1
 - For sw and beq, don't care

Instruction	RegDst	ALUSrc	MemtoReg	RegWrite
R-format	1	0	0	1
lw	0	1	1	1
sw	X	1	X	0
beq	X	0	0	X

©2004 Morgan Kaufmann Publishers 19

RegWrite

- Write to Registers when it is R-type or lw, don't write to register when it is sw or beq
 - For R-type and lw, RegWrite=1
 - For sw and beq, RegWrite=0
- Quick Question
 - Can we do 1101?

Instruction	RegDst	ALUSrc	MemtoReg	Reg Write
R-format	1	0	0	1
lw	0	1	1	1
sw	X	1	X	0
beq	X	0	0	0

©2004 Morgan Kaufmann Publishers 20

MemRead

- Read from memory ONLY when it is a lw
 - For lw, set MemRead=1
 - Otherwise, MemRead=0

Instruction	RegDst	ALUSrc	MemtoReg	Reg Write	Mem Read
R-format	1	0	0	1	0
lw	0	1	1	1	1
sw	X	1	X	0	0
beq	X	0	0	X	0

©2004 Morgan Kaufmann Publishers 21

MemWrite

- Write to memory ONLY if we have sw
 - For sw, set MemWrite=1
 - Otherwise, MemWrite=0
- Quick Question
 - Can we have Mem Read to be 1101?

Instruction	RegDst	ALUSrc	MemtoReg	Reg Write	Mem Read	Mem Write
R-format	1	0	0	1	0	0
lw	0	1	1	1	1	0
sw	X	1	X	0	0	1
beq	X	0	0	0	0	0

©2004 Morgan Kaufmann Publishers 22

Branch

- Needed for PCsrc calculation
 - For beq, set Branch=1
 - Otherwise, Branch=0

Instruction	RegDst	ALUSrc	MemtoReg	Reg Write	Mem Read	Mem Write	Branch	ALUSrc1	ALUSrc0
R-format	1	0	0	1	0	0	0	0	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	0	0	0	0	1	0	1

©2004 Morgan Kaufmann Publishers 23

Instruction	RegDst	ALUSrc	MemtoReg	Reg Write	Mem Read	Mem Write	Branch	ALUSrc1	ALUSrc0
R-format	1	0	0	1	0	0	0	0	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	0	0	0	0	1	0	1

©2004 Morgan Kaufmann Publishers 24

Single Cycle – Steps of each instruction

Inst. Type	Functional Units Used				
	Instruction fetch	Register read	ALU	Register write	
R-type	Instruction fetch	Register read	ALU	Register write	
Load	Instruction fetch	Register read	ALU	Memory access	Register write
Store	Instruction fetch	Register read	ALU	Memory access	
Branch	Instruction fetch	Register read	ALU		
Jump	Instruction fetch				

Our Simple Control Structure

- All of the logic is combinational
- We wait for everything to settle down, and the right thing to be done
 - ALU might not produce “right answer” right away
 - we use write signals along with clock to determine when to write
- Cycle time determined by length of the longest path



We are ignoring some details like setup and hold times