inst.eecs.berkeley.edu/~cs61c

UC Berkeley CS61C : Machine Structures

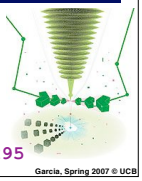
Lecture 26
CPU Design: Designing a Single-cycle CPU, pt 2

2007-03-19

Lecturer SOE Dan Garcia

www.cs.berkeley.edu/~ddgarcia

3.6 TB DVDs? Maybe! ⇒ Researchers at Harvard have found a way to use light-focusing "optical antennas" to get way past the diffraction limit and focus light smaller.

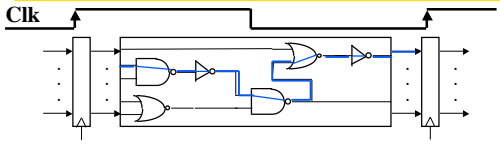


technologyreview.com/Infotech/18295

How to Design a Processor: step-by-step

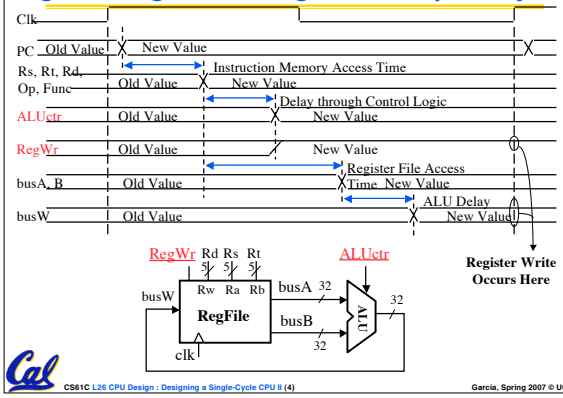
1. Analyze instruction set architecture (ISA) => datapath requirements
   - meaning of each instruction is given by the *register transfers*
   - datapath must include storage element for ISA registers
   - datapath must support each register transfer
2. Select set of datapath components and establish clocking methodology
3. Assemble datapath meeting requirements
4. Analyze implementation of each instruction to determine setting of control points that effects the register transfer.
5. Assemble the control logic

Clocking Methodology



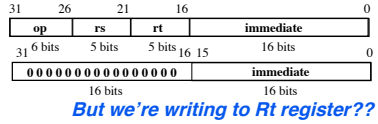
- Storage elements clocked by same edge
- Being physical devices, flip-flops (FF) and combinational logic have some delays
  - Gates: delay from input change to output change
  - Signals at FF D input must be stable before active clock edge to allow signal to travel within the FF (set-up time), and we have the usual clock-to-Q delay
- "Critical path" (longest path through logic) determines length of clock period

Register-Register Timing: One complete cycle

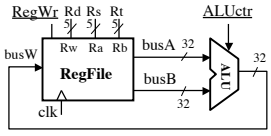


3c: Logical Operations with Immediate

- R[rt] = R[rs] op ZeroExt[imm16]]

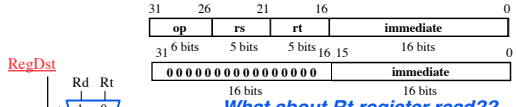


But we're writing to Rt register??



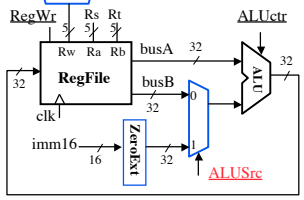
3c: Logical Operations with Immediate

- R[rt] = R[rs] op ZeroExt[imm16]]



RegDst

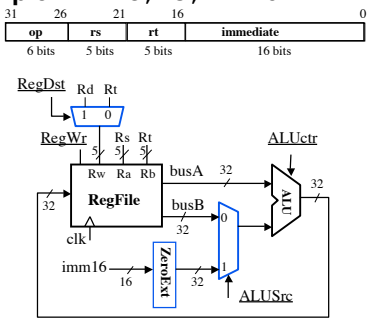
What about Rt register read??



- Already defined 32-bit MUX; Zero Ext?

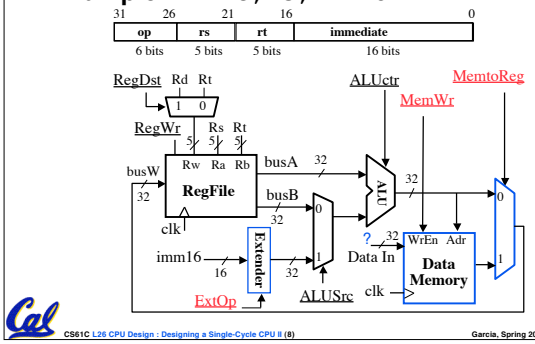
3d: Load Operations

- R[rt] = Mem[R[rs] + SignExt[imm16]]
  Example: `lw rt,rs,imm16`



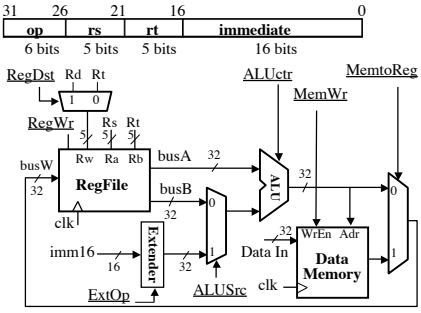
3d: Load Operations

- R[rt] = Mem[R[rs] + SignExt[imm16]]
  Example: `lw rt,rs,imm16`



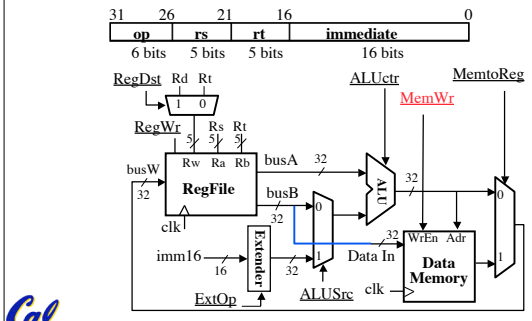
3e: Store Operations

- Mem[R[rs] + SignExt[imm16]] = R[rt]
  Ex.: `sw rt, rs, imm16`

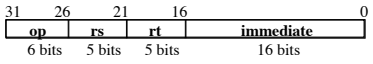


3e: Store Operations

- Mem[R[rs] + SignExt[imm16]] = R[rt]
  Ex.: `sw rt, rs, imm16`



3f: The Branch Instruction



`beq rs, rt, imm16`

- mem[PC] Fetch the instruction from memory
- Equal = R[rs] == R[rt] Calculate branch condition
- if (Equal) Calculate the next instruction's address
  - PC = PC + 4 + (SignExt(imm16) x 4)

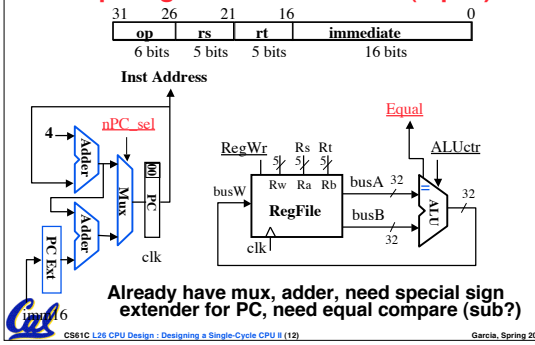
else

- PC = PC + 4

Datapath for Branch Operations

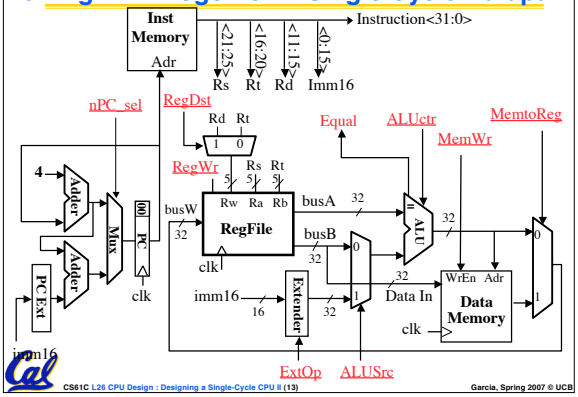
- beq rs, rt, imm16

Datapath generates condition (equal)

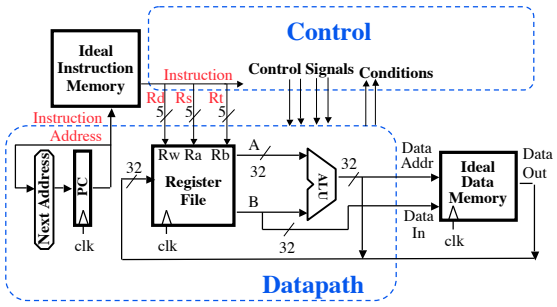


Already have mux, adder, need special sign extender for PC, need equal compare (sub?)

Putting it All Together:A Single Cycle Datapath



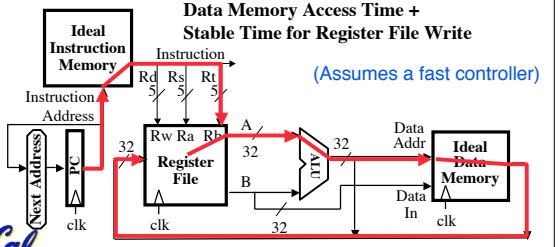
An Abstract View of the Implementation



An Abstract View of the Critical Path

Critical Path (Load Instruction) =
Delay clock through PC (FFs) +
Instruction Memory's Access Time +
Register File's Access Time, +
ALU to Perform a 32-bit Add +
Data Memory Access Time +
Stable Time for Register File Write

(Assumes a fast controller)



Administrivia

- Friday will be a webcast archived lec
  - …no need to attend class, but you can if you want to (I'll be here)

Peer Instruction

A. Truth table for mux with 4-bits of signals has 2^4 rows

B. We could cascade N 1-bit shifters to make 1 N-bit shifter for sll, srl

C. If 1-bit adder delay is T, the N-bit adder delay would also be T

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Peer Instruction Answer

A. Truth table for mux with 4-bits of signals controls 16 inputs, for a total of 20 inputs, so truth table is 2^{20} rows…FALSE

B. We could cascade N 1-bit shifters to make 1 N-bit shifter for sll, srl … TRUE

C. What about the cascading carry? FALSE

A. Truth table for mux with 4-bits of signals has 2^4 rows

B. We could cascade N 1-bit shifters to make 1 N-bit shifter for sll, srl

C. If 1-bit adder delay is T, the N-bit adder delay would also be T

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Summary: A Single Cycle Datapath

- We have everything except control signals