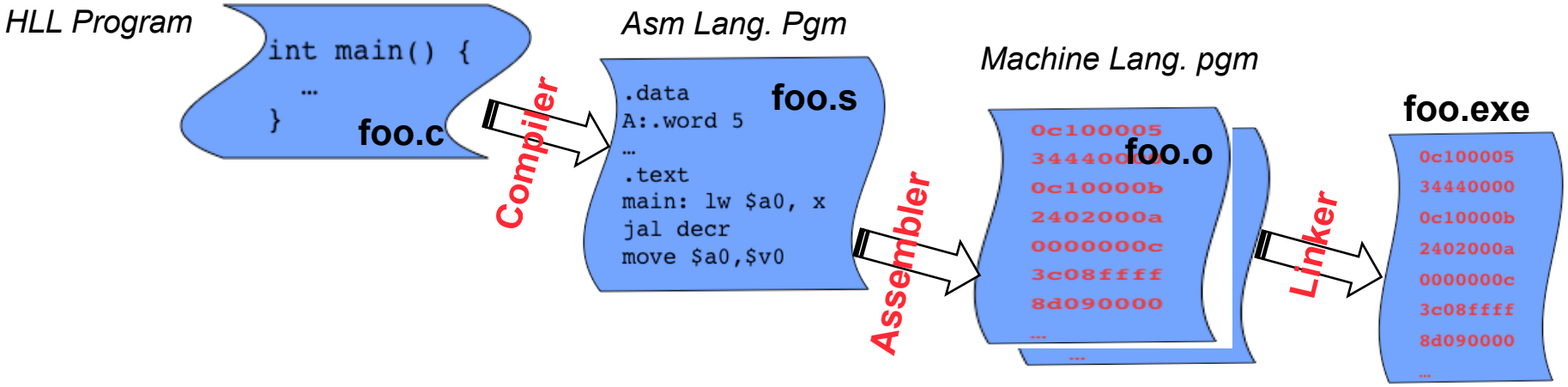
CS61CL Machine Structures

Lec 8 – State and Register Transfers

David Culler
Electrical Engineering and Computer Sciences
University of California, Berkeley

CS61CL Road Map

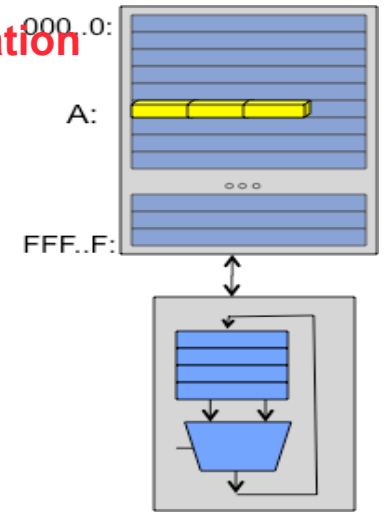


Software

Hardware

Instruction Set Architecture

Machine Organization



Instr. Set Proc. | **I/O system**

Datapath & Control

Digital Design

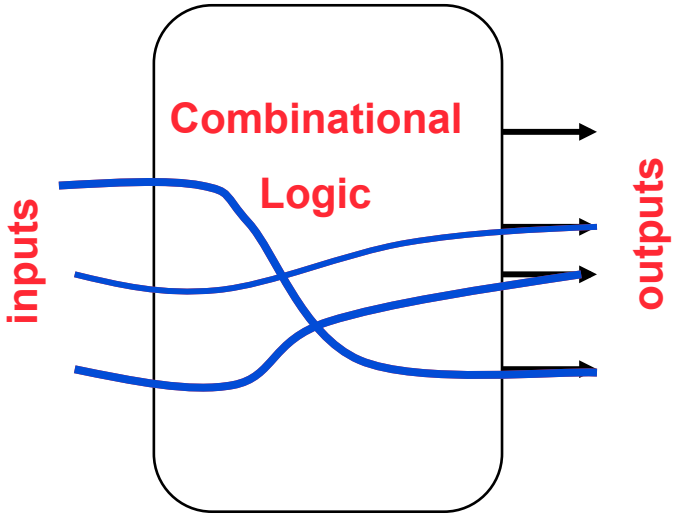
Circuit Design

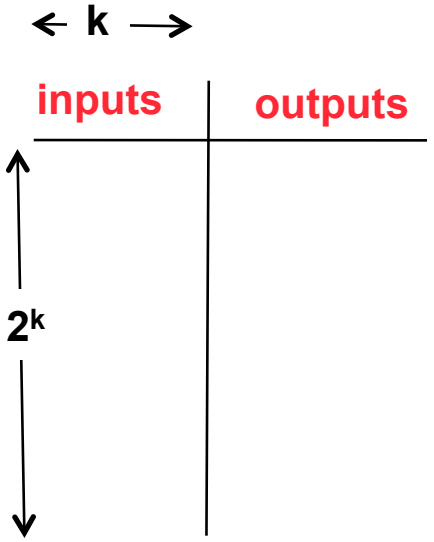
Layout & fab

Semiconductor Materials

Review: Combinational Logic

- Any boolean function can be expressed as an acyclic connection of gates
- Often specified by a truth table

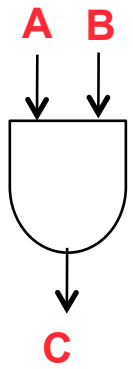


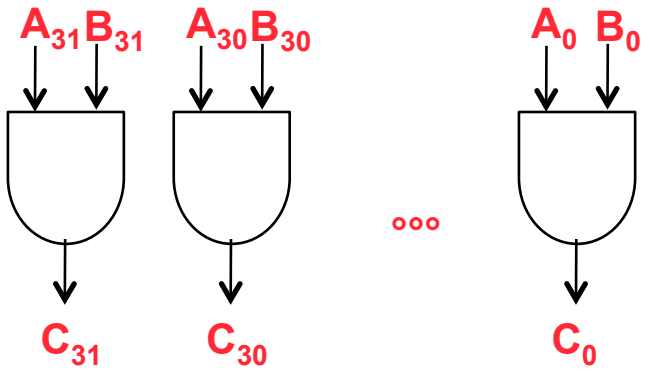


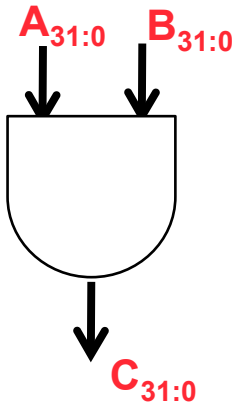
- Outputs are purely a function of the inputs
  - no history, no state

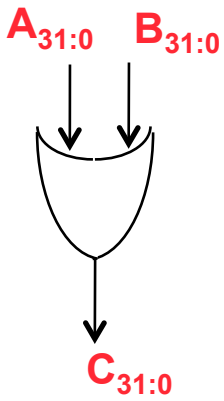
Examples: Logical Operations

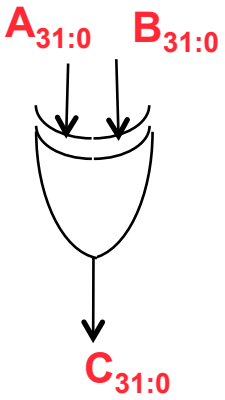
C = A & B

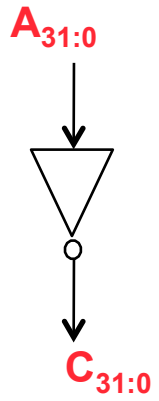








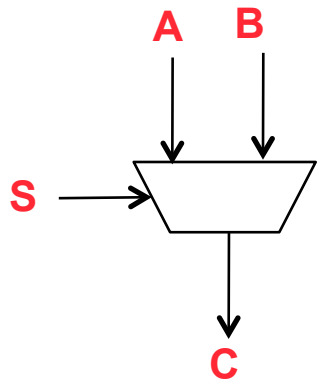


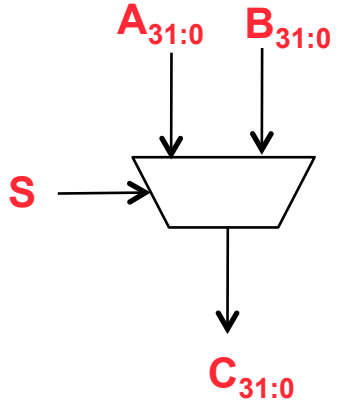


Example: Multiplexor

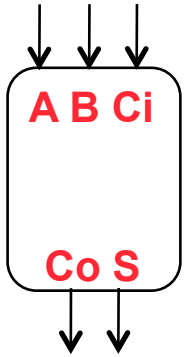
C = S ? A : B

C = (S & A) | (~S & B)

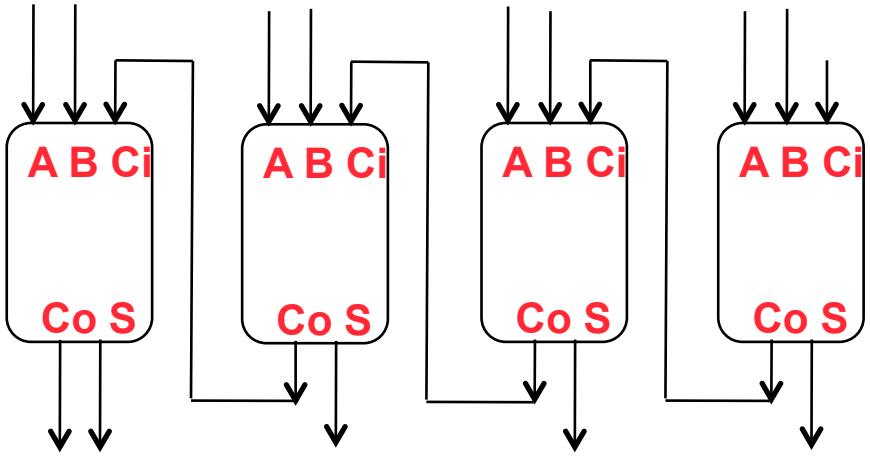


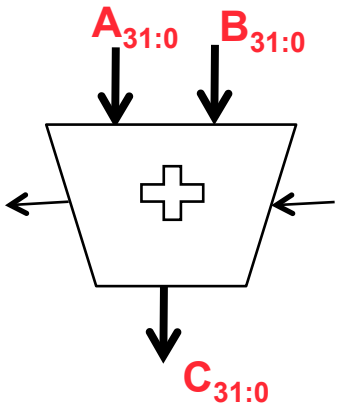


Example: Adder

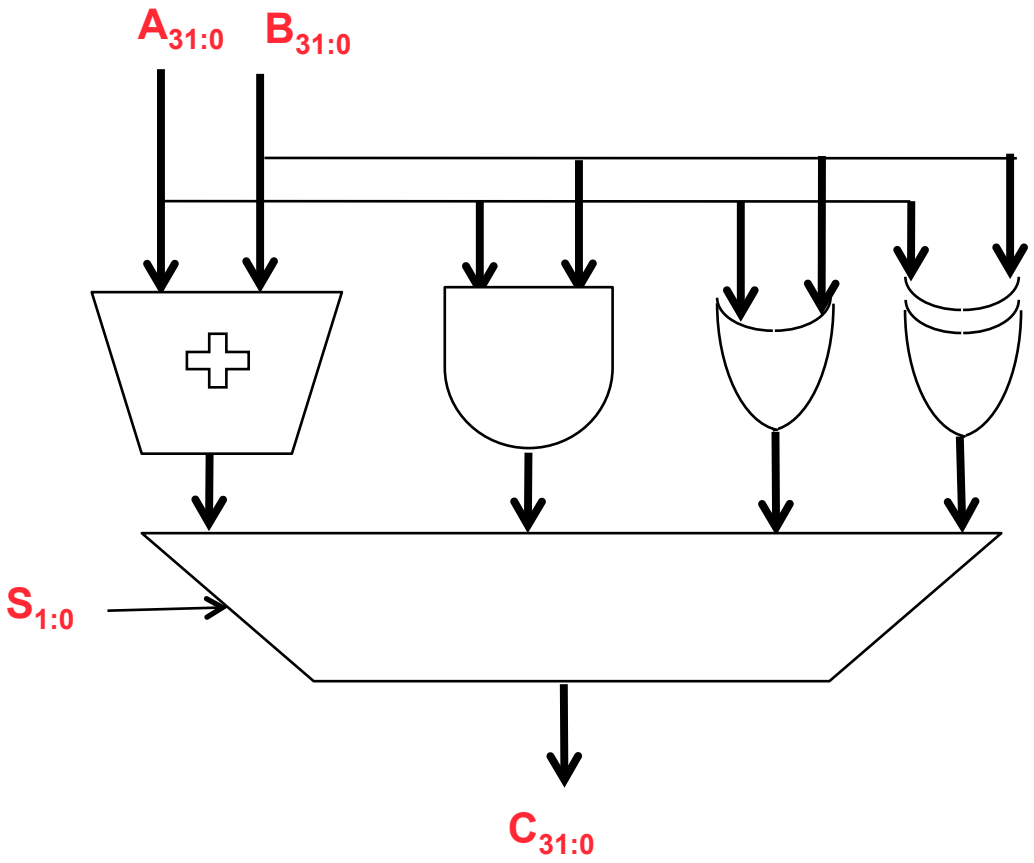


Ci	A	B	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	1	1	1

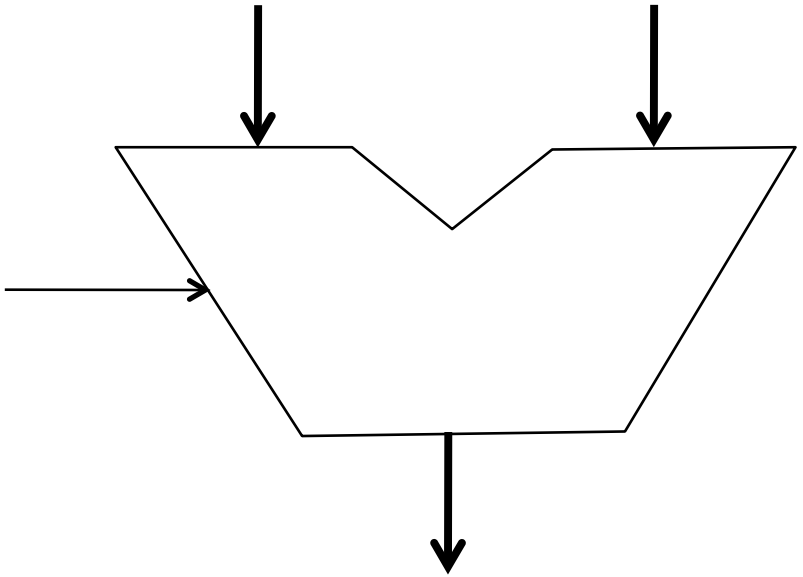




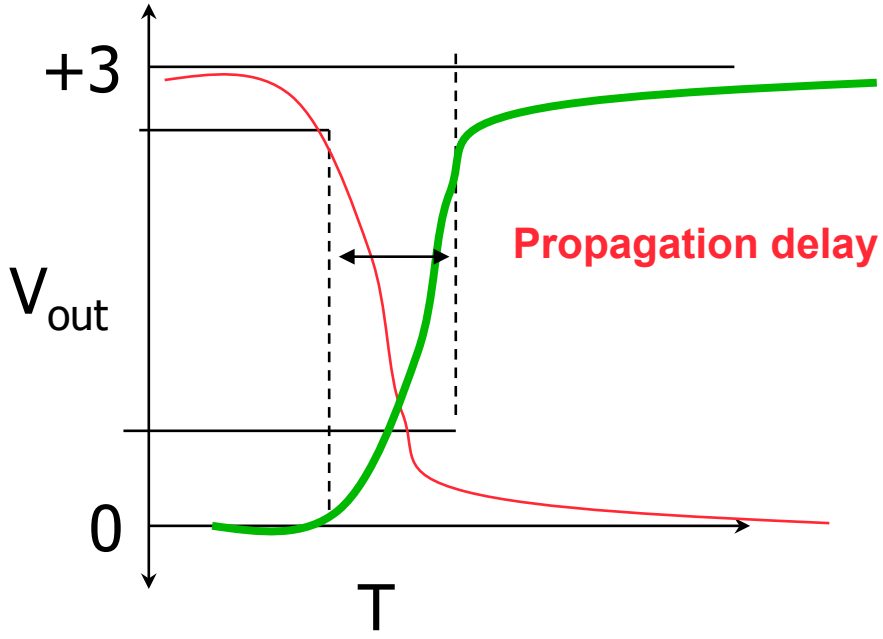
Example: Arithmetic Logic Unit



ALU

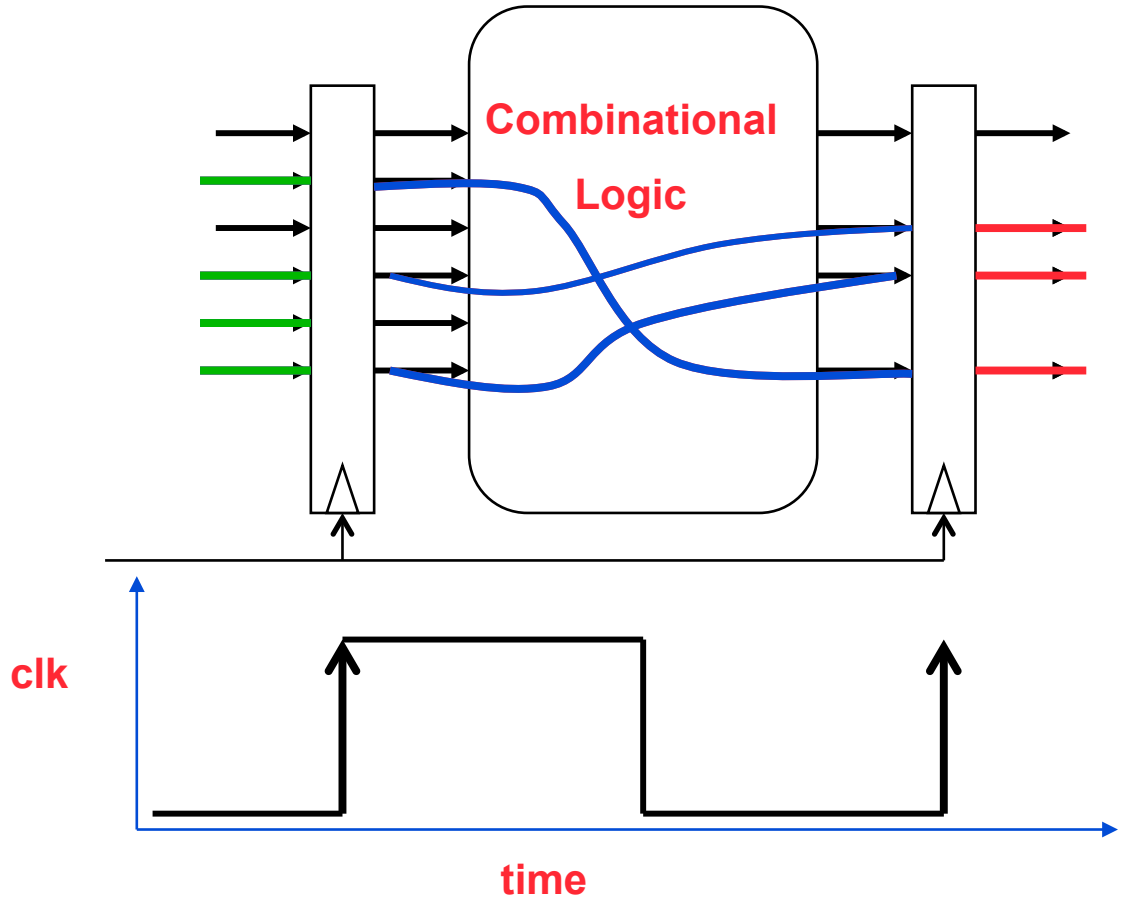


Element of Time



- **Logical change is not instantaneous**
- **Broader digital design methodology has to make it appears as such**
  - **Clocking, delay estimation, glitch avoidance**

What makes Digital Systems tick?

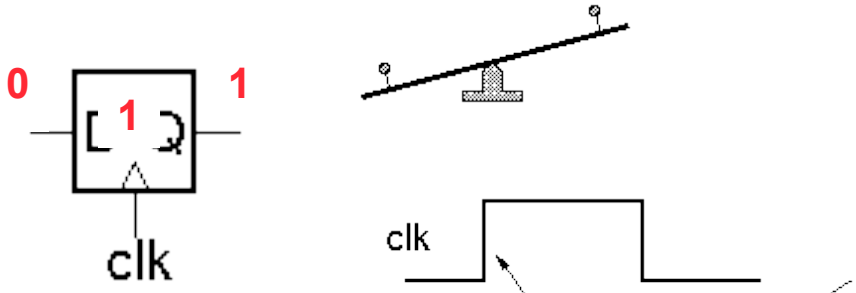


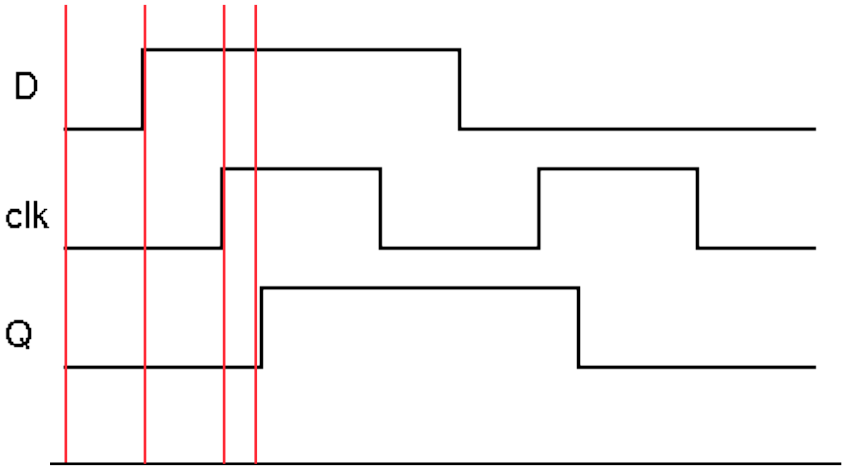
Administrative Issues

- HW 6 due tonight
- Project 2 dues Monday 10/26
  - bimodal check-off
  - testing tools available tomorrow
  - they are really picky
- Project 1 grading almost done
  - Friday
- HW 7 – discuss

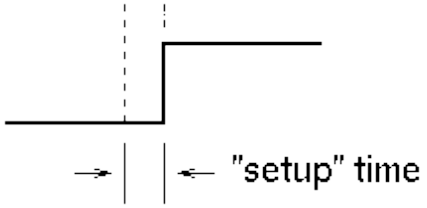
- Midterm 2 on 11/9 as in original schedule
  - 11/11 is holiday

A Bit of state: D-type edge-triggered flip-flop



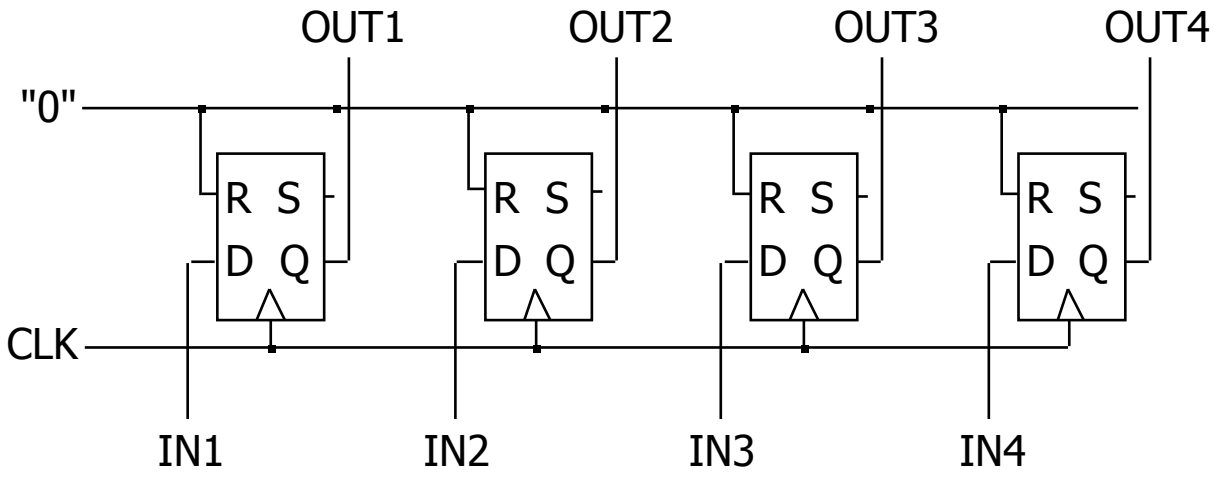


- **The edge of the clock is used to *sample* the "D" input & send it to "Q" (positive edge triggering).**
  - **At all other times the output Q is independent of the input D (just stores previously sampled value).**
  - **The input must be stable for a short time before the clock edge.**

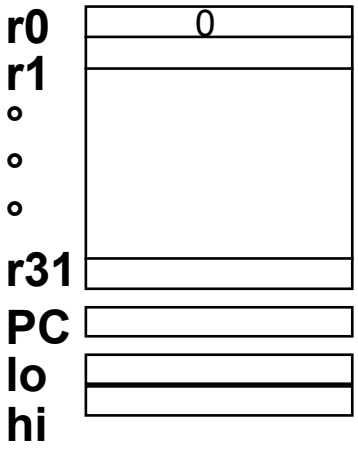


Registers

- **Collections of flip-flops with similar controls and logic**
  - **Stored values somehow related (e.g., form binary value)**
  - **Share clock, reset, and set lines**
  - **Similar logic at each stage**



What “registers” do we need?



Programmable storage

2^32 x bytes

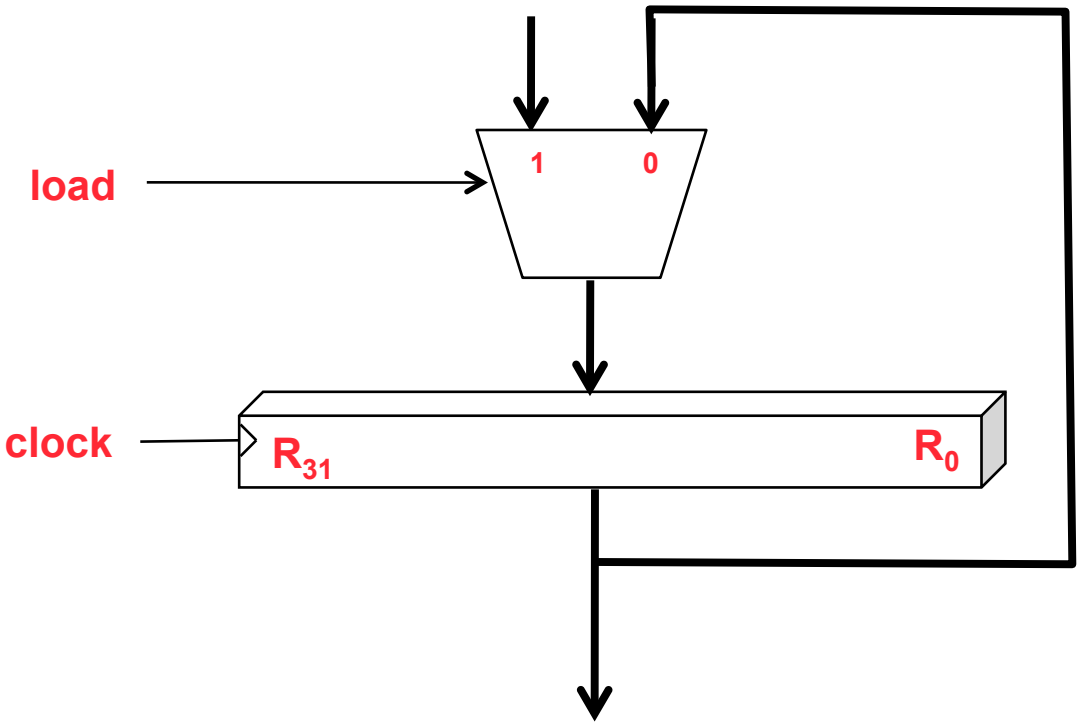
31 x 32-bit GPRs (R0=0)

32 x 32-bit FP regs (paired DP)

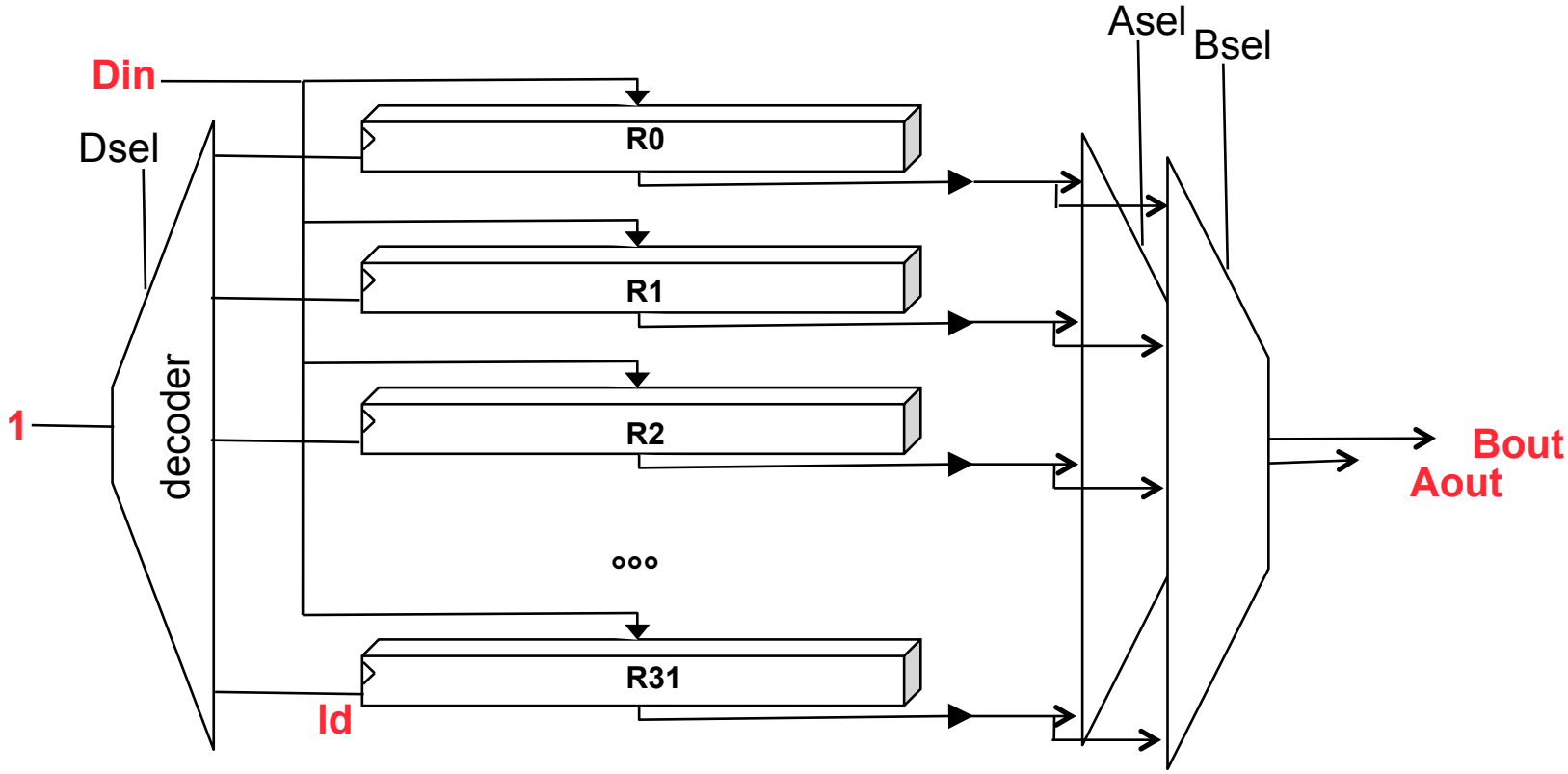
HI, LO, PC

- **“read” vs use the output**
- **“write” on the clock edge => Load**
- **Load Control**

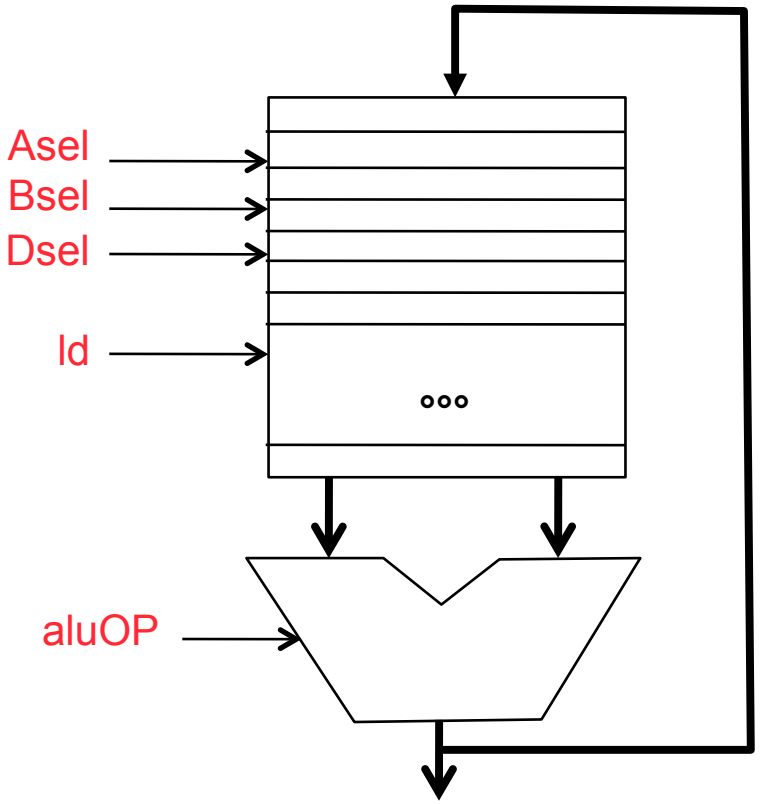
Register with Load Control



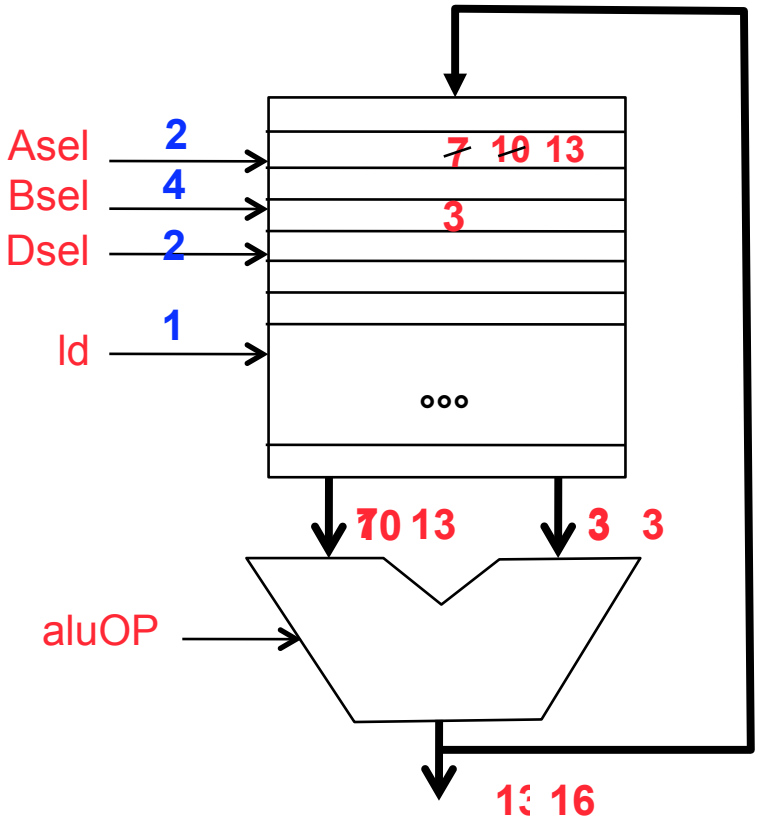
Register File



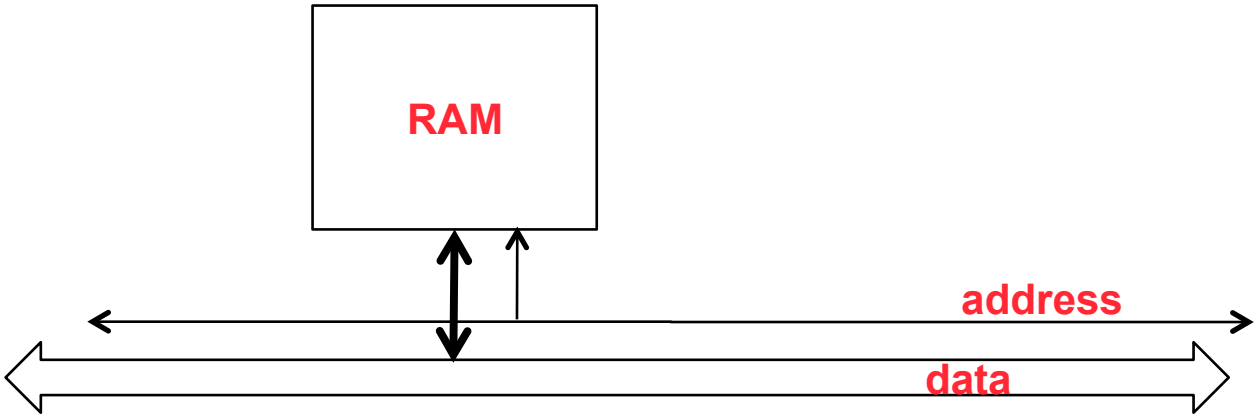
Towards a Data Path



Exercise a Data Path

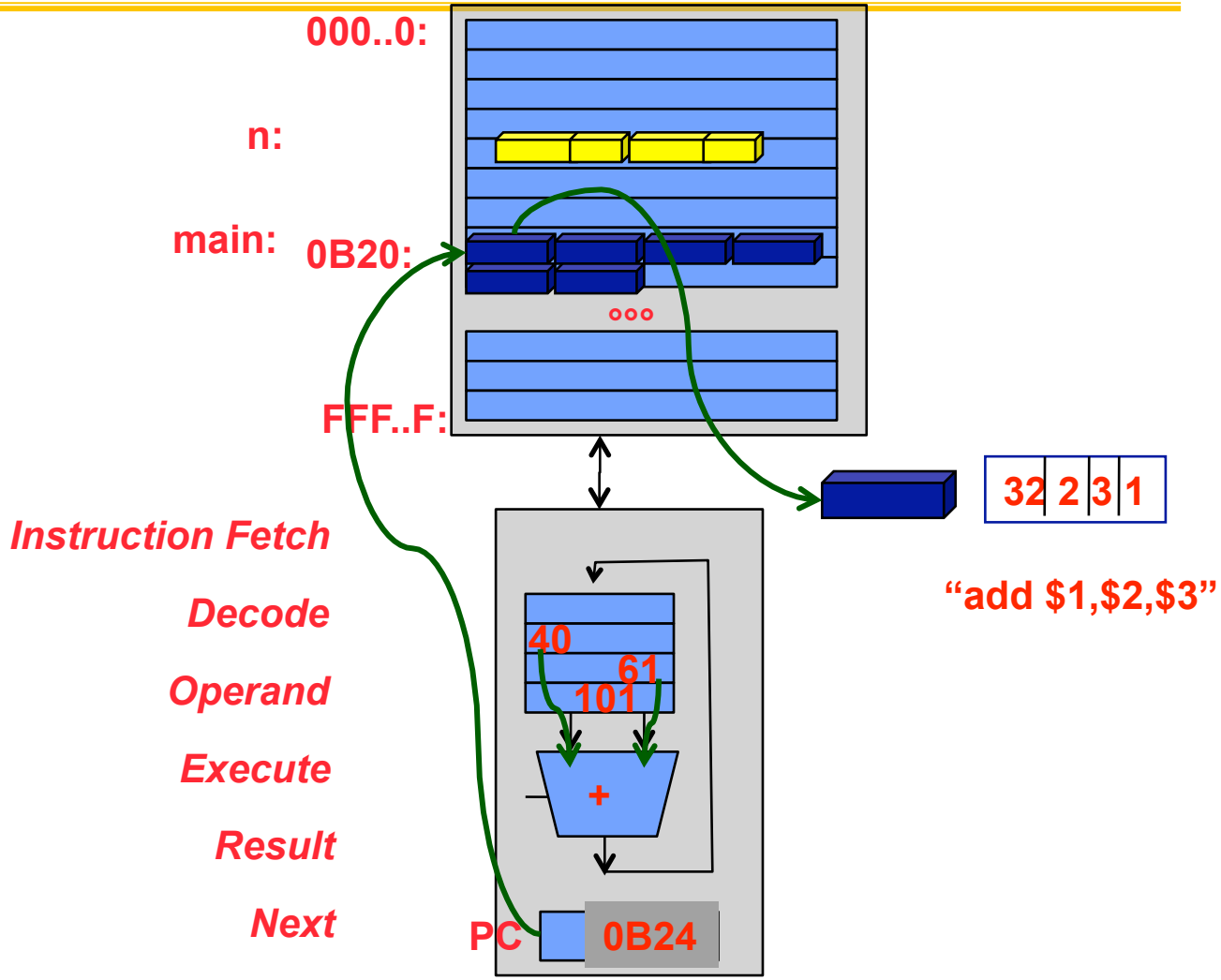


What about RAM - Randomly Accessed Memory?



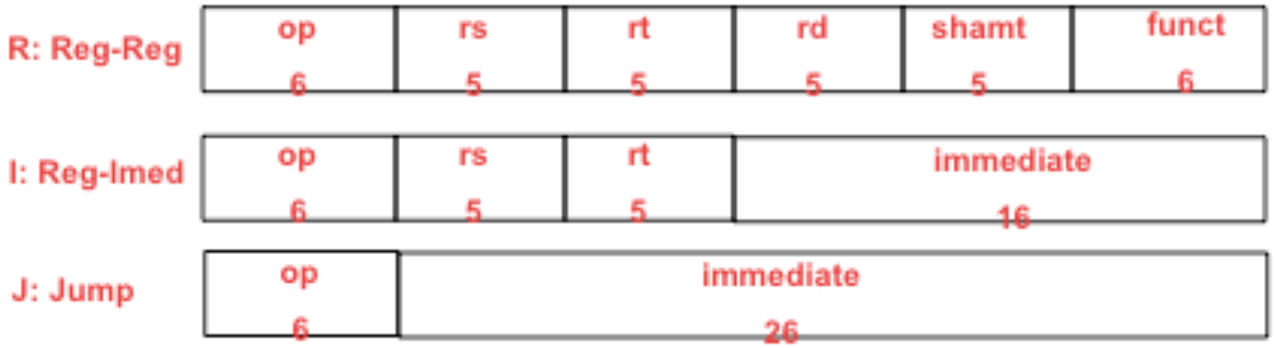
- Like a HUGE register file
  - dense, slower, low-cost storage cell (6T)
  - fewer ports
  - wider address lines
  - accessed over a “bus”
- Bus: means of composition in hardware system
  - logically related collection of wires
  - interfacing one or more sources to one or more destinations

Recall: Instruction Cycle



Register Transfers

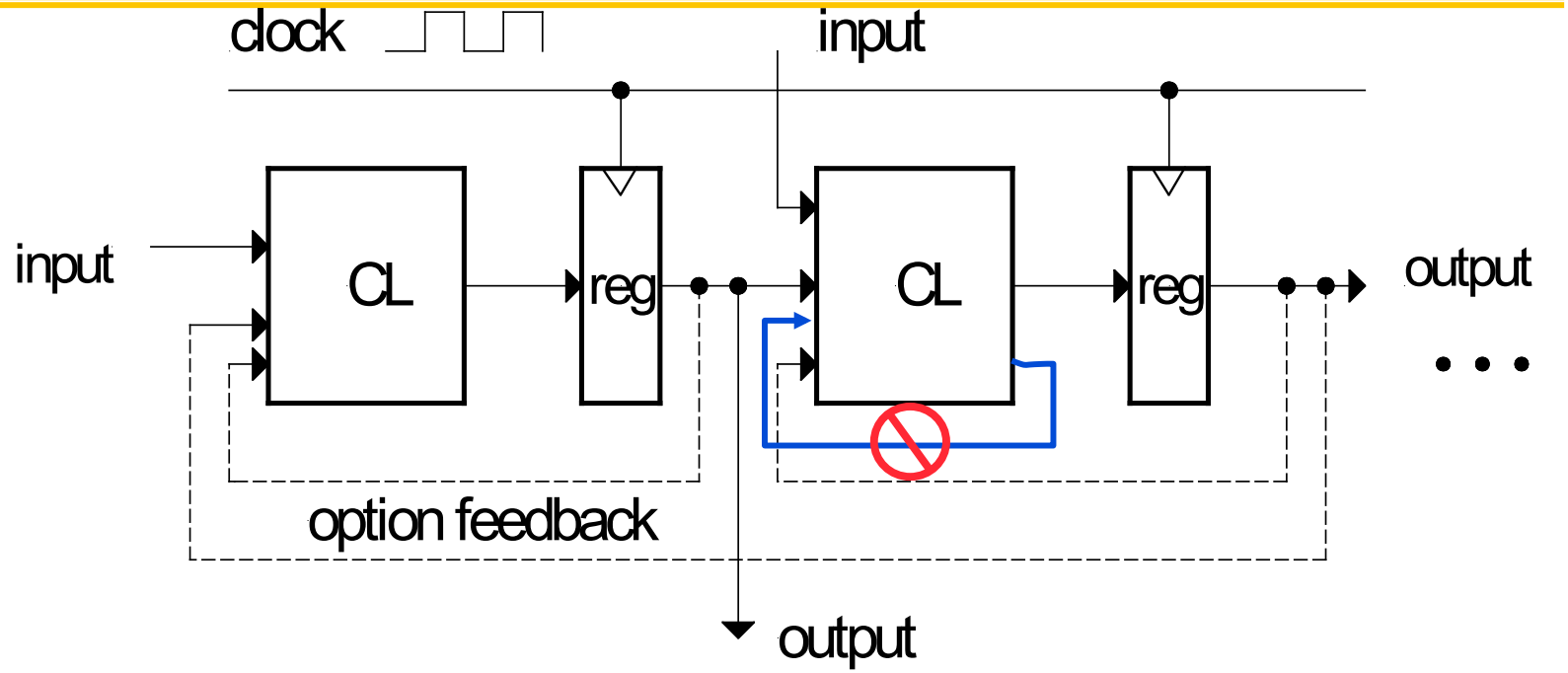
MIPS Instruction Format



- **Reg-Reg instructions (op == 0)**
  - add, sub, and, or, nor, xor, slt — R[rd] := R[rs] *funct* R[rt]; pc:=pc+4
  - sll. srl, sra — R[rd] := R[rt] *shft* shamt
- **Reg-Immed (op != 0)**
  - addi, andi, ori, xori, lui, — R[rt] := R[rs] op Im16
  - addiu, slti, sltiu
  - lw, lh, lhu, lb, lbu — R[rt] := Mem[R[rs] + signEx(Im16)]*
  - sw, sh, sb — Mem[R[rs] + signEx(Im16)] := R[rt]

10/21/09 cs61cl f09 lec 5 12

Synchronous Circuit Design



- **Combinational Logic Blocks (CL)**
  - Acyclic
  - no internal state (no feedback)
  - output only a function of inputs
- **Registers (reg)**
  - collections of flip-flops
- **clock**
  - distributed to all flip-flops
- **ALL CYCLES GO THROUGH A REG!**