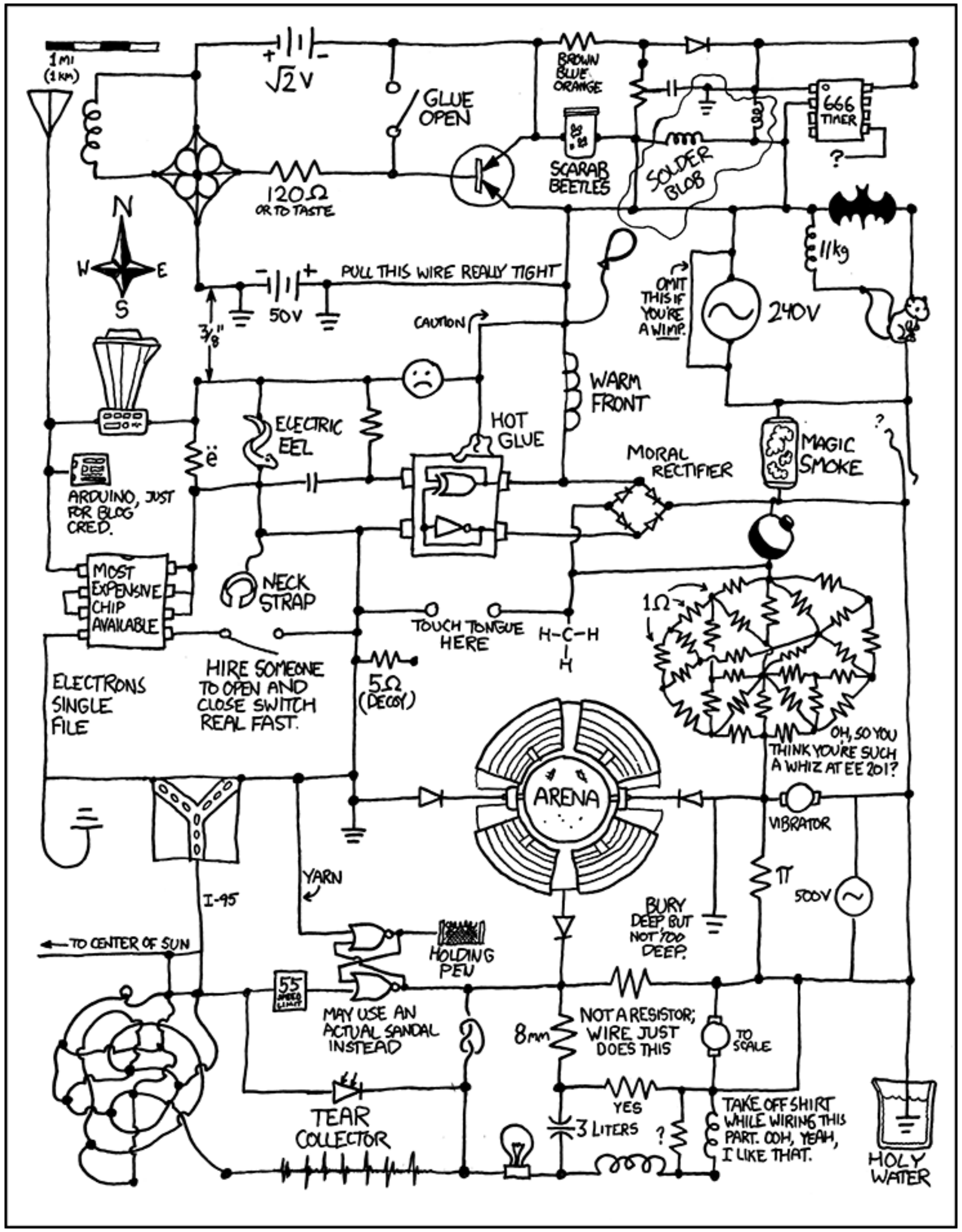
CS 261
Fall 2019

Mike Lam, Professor


1 MI (1 KM)
√2 V
GLUE OPEN
BROWN BLUE ORANGE
666 TIMER
?
SCARAB BEETLES
SOLDER BLOB
120Ω OR TO TASTE
N
W
E
S
50V
3/8"
PULL THIS WIRE REALLY TIGHT
CAUTION
OMIT THIS IF YOU'RE A WIMP.
240V
1/kg
ë
ELECTRIC EEL
HOT GLUE
WARM FRONT
MORAL RECTIFIER
MAGIC SMOKE
ARDUINO, JUST FOR BLOG CRED.
MOST EXPENSIVE CHIP AVAILABLE
NECK STRAP
TOUCH TONGUE HERE
H-C-H
H
1Ω
ELECTRONS SINGLE FILE
HIRE SOMEONE TO OPEN AND CLOSE SWITCH REAL FAST.
5Ω (DECOY)
OH, SO YOU THINK YOU'RE SUCH A WHIZ AT EE 201?
ARENA
VIBRATOR
500V
I-95
YARN
BURY DEEP, BUT NOT TOO DEEP.
TO CENTER OF SUN
HOLDING PEN
55 SPEED LIMIT
MAY USE AN ACTUAL SANDAL INSTEAD
8mm
NOT A RESISTOR; WIRE JUST DOES THIS
TO SCALE
YES
3 LITERS
?
TAKE OFF SHIRT WHILE WIRING THIS PART. OOH, YEAH, I LIKE THAT.
TEAR COLLECTOR
HOLY WATER

Sequential Circuits

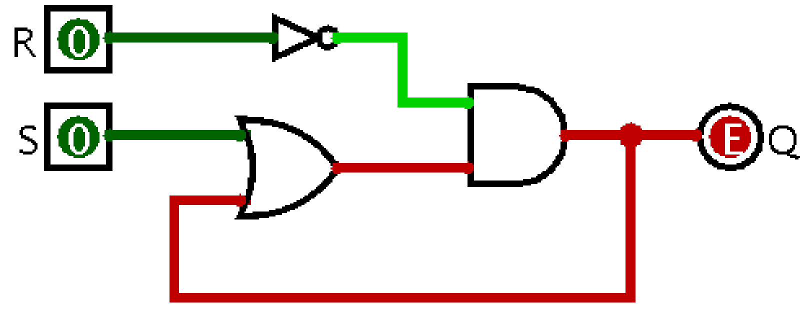
Circuits

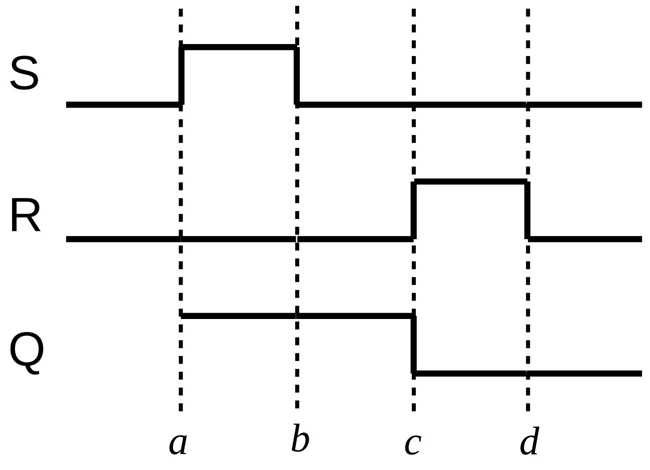
- Circuits are formed by linking gates (or other circuits) together
  - Inputs and outputs
    - Link output of one gate to input of another
    - Some circuits have multiple inputs and/or outputs
  - Combinational circuits: outputs are a boolean function of inputs
    - Not time-dependent
    - Used for computation
  - Sequential circuits: output is dependent on previous outputs
    - Time-dependent
    - Used for memory

Circuit memory

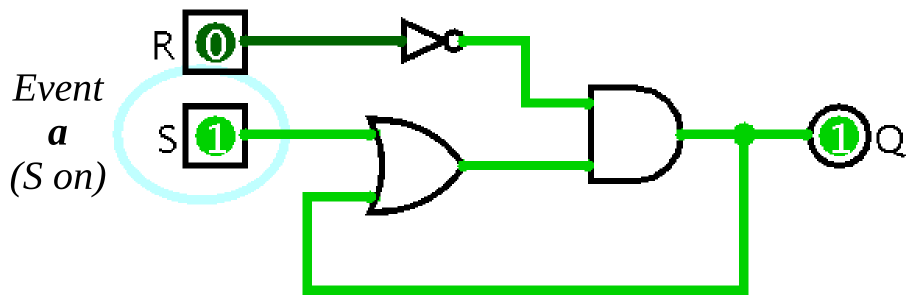
- Question: How do we make a circuit “remember” something?
  - Answer: Create a feedback loop!
  - Creates a “storage” circuit, often called a latch
  - Truth table must include previous state
  - Alternatively, draw a timing diagram
    - Shows how input/output signals change with respect to time
    - Given input signals in diagram, we can determine output signals

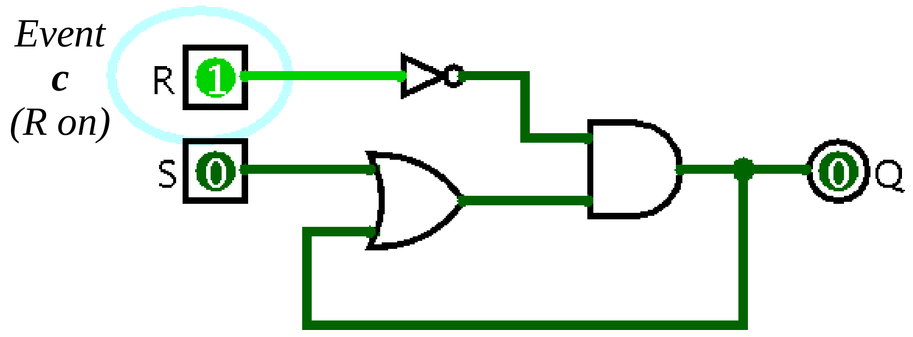
SR AND-OR latch

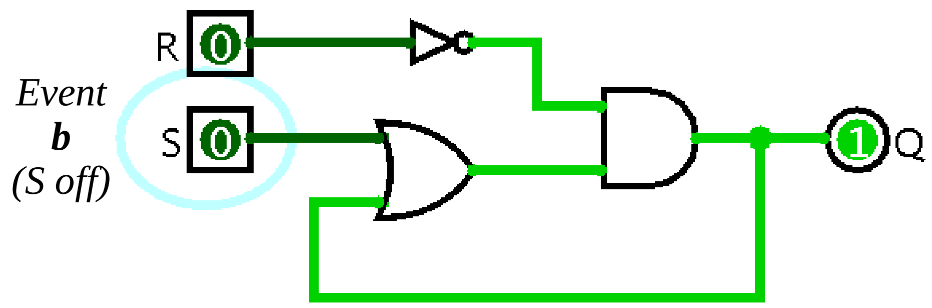


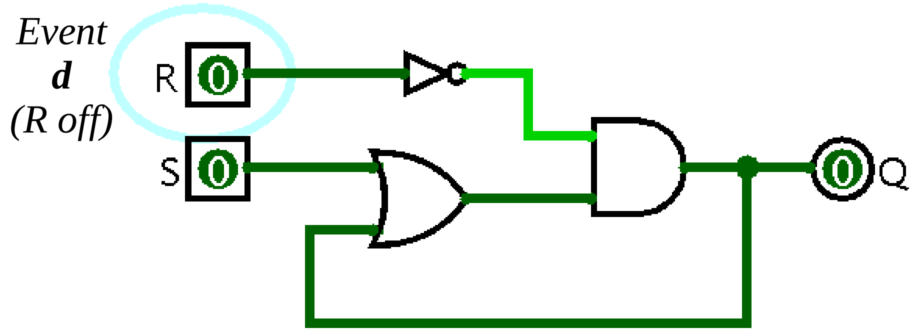


S = "set"   R = "reset"

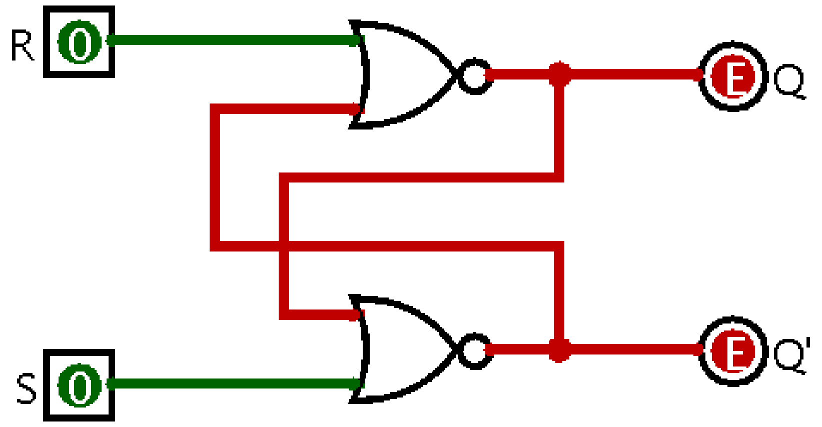






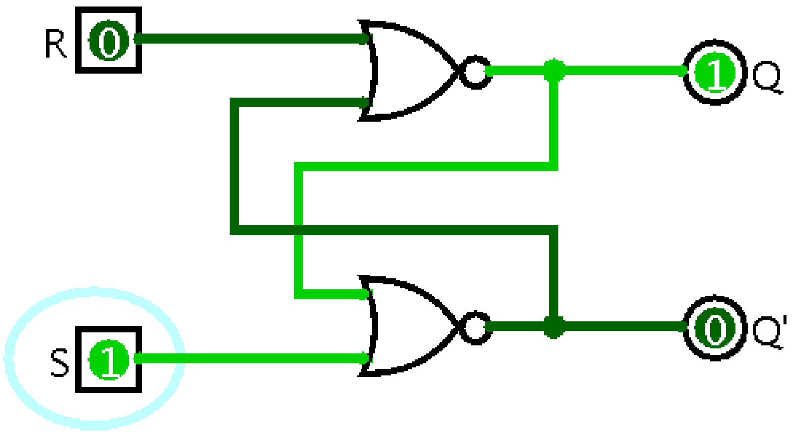


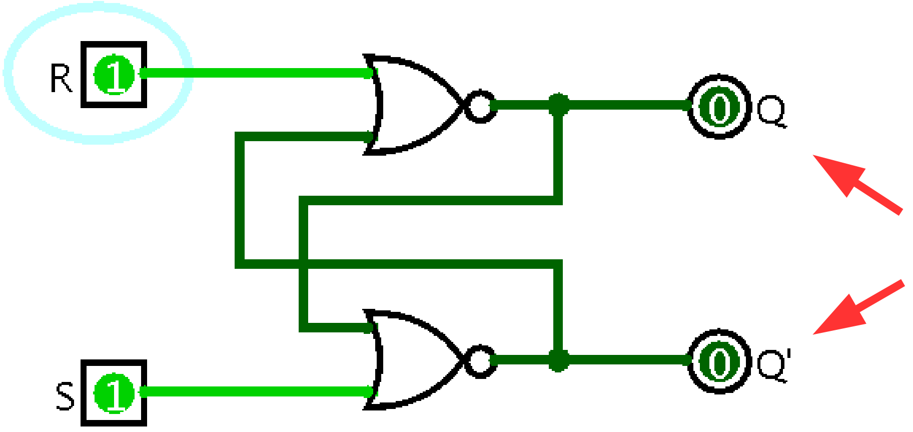
SR NOR latch



Works similarly to AND-OR, but requires one fewer gate (and it is a universal gate!)

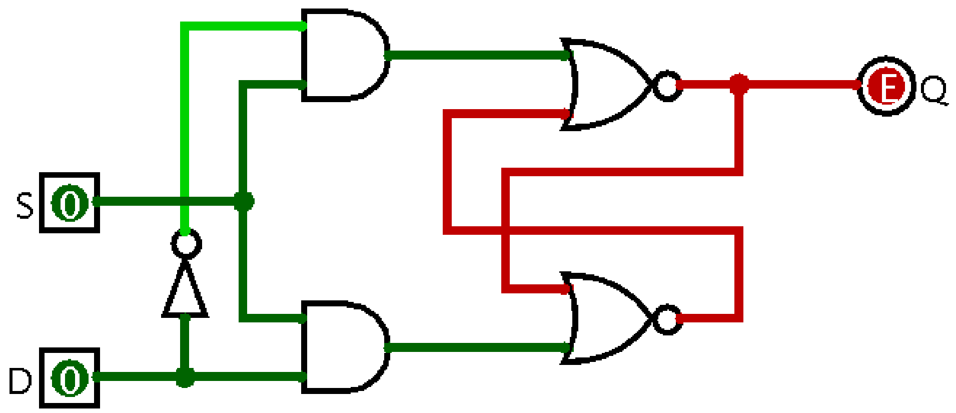
Question: What happens if we turn both R and S on at the same time?

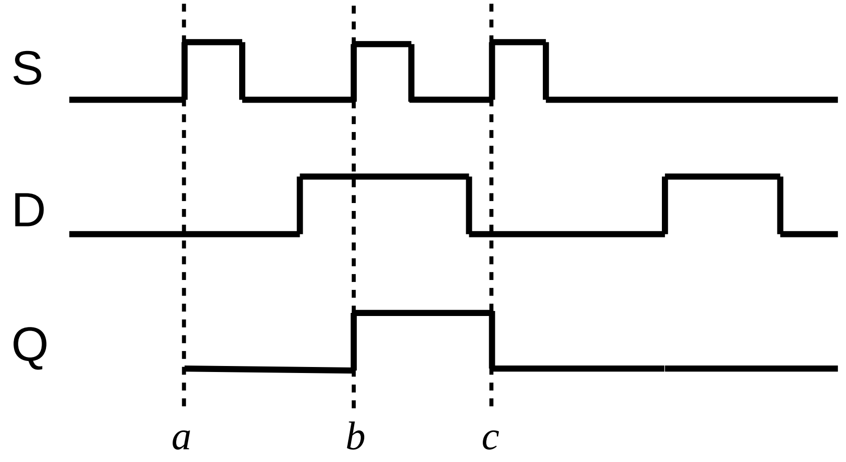




Disallow S=1, R=1 because Q’ ≠ !Q

D latch



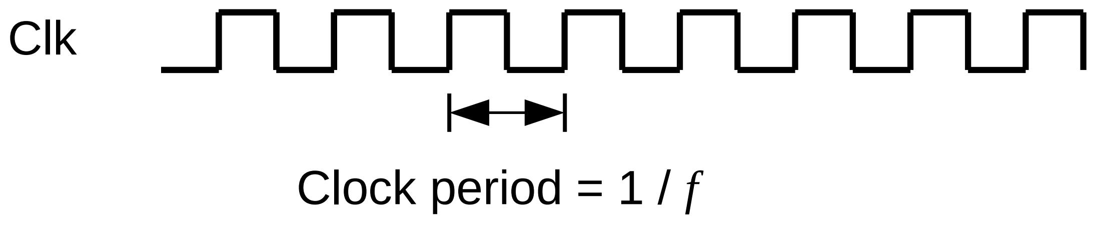


From “Code” book: S = “Save that bit!”

- As long as S is on, Q reflects the value of D.
- When S turns off, Q is "frozen" and retains its previous value.
- D can change while S is off with no change in Q

Clocks

- Provide oscillating signal
- Often used as “set” signal for latches
- Keeps computation and memory in sync
- Clocked latches are called flip-flops
- The clock period is the inverse of the frequency (measured in *hertz*)
- The length of a clock period determines the minimum time an instruction takes to execute

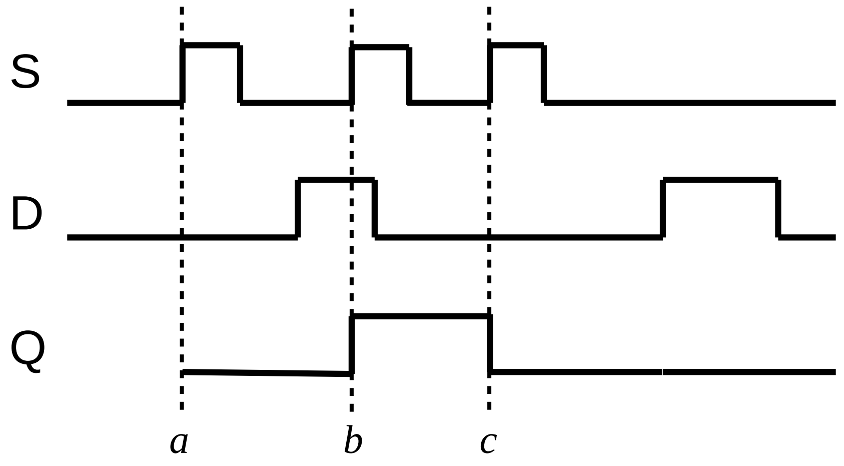


Flip-flop types

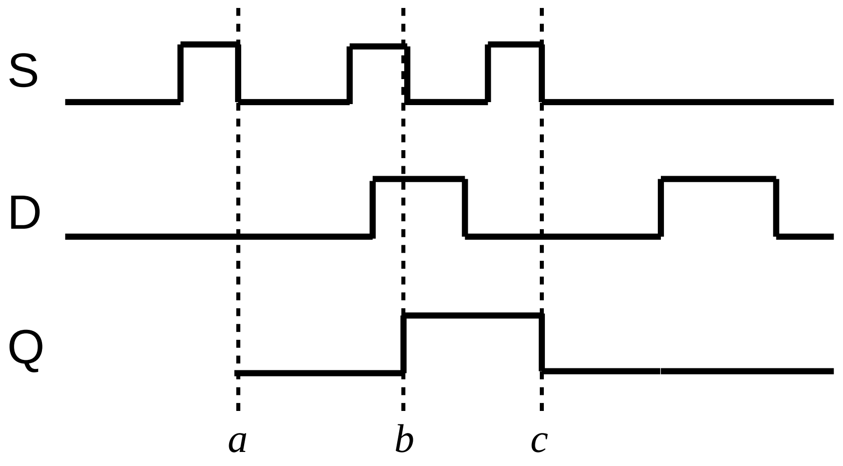
- SR: “set-reset”
- D: “data” bit + clock
- T: “toggle”
- JK: like SR + T (toggle when S=1, R=1)
  - J is S, K is R
- Any of these can be used to build the others
- Also can be built from basic logic gates in multiple ways

Signal changes

- The original D latch reflects D input on Q as long as “set” is on
- Edge-triggered flip-flops change Q on rising edge of “set” signal
- Master-slave flip-flops change Q on falling edge of “set” signal



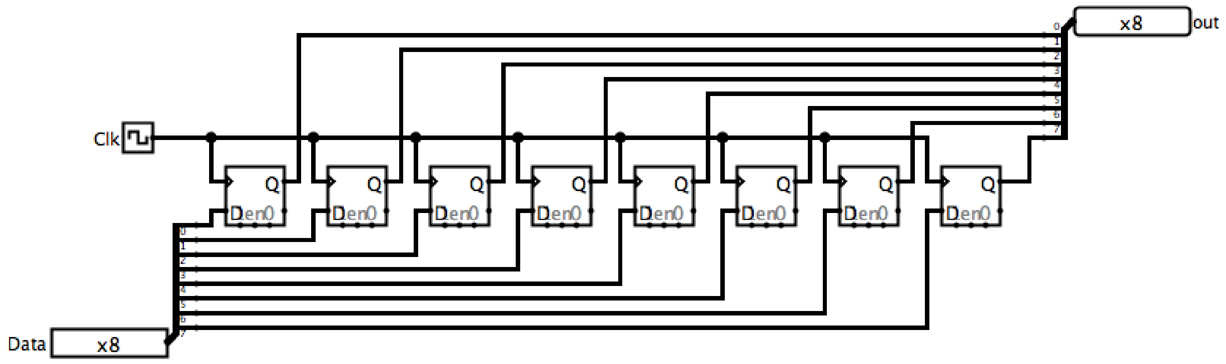
Edge-triggered D latch

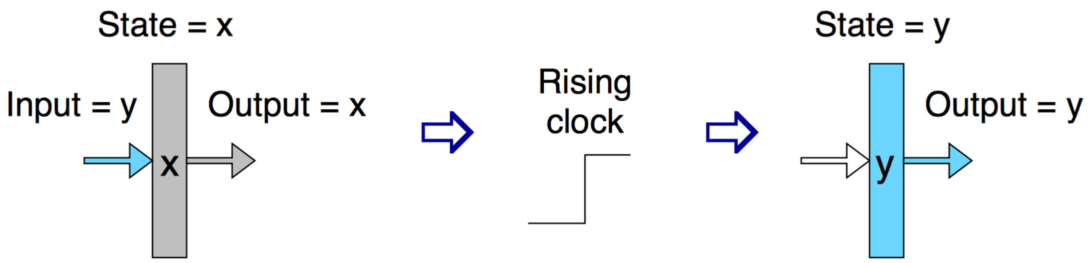


Master-slave D latch

Registers

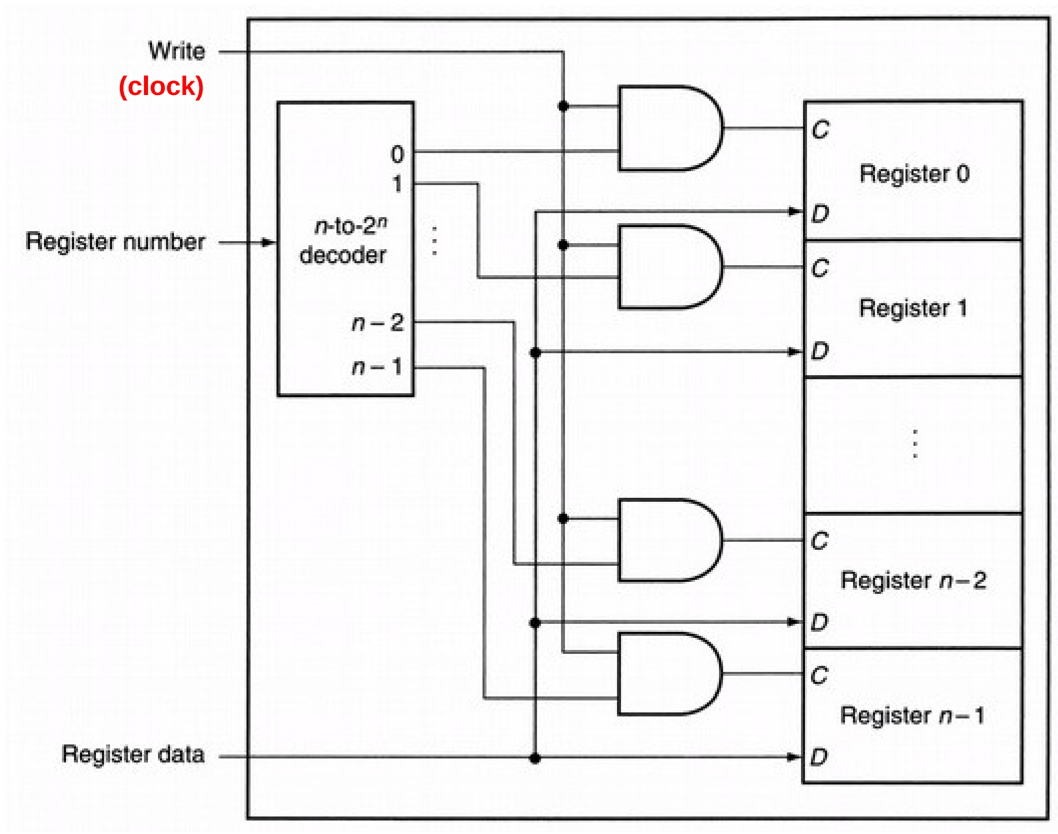
- Registers: arrays of flip-flops with a single set/clock input
- Connected by buses (groups of wires) to other components
- Edge triggering allows computation to stabilize before results are saved
- Caveat: difference between hardware registers and program registers
  - Former are physical, latter are logical (and stored in a register file)

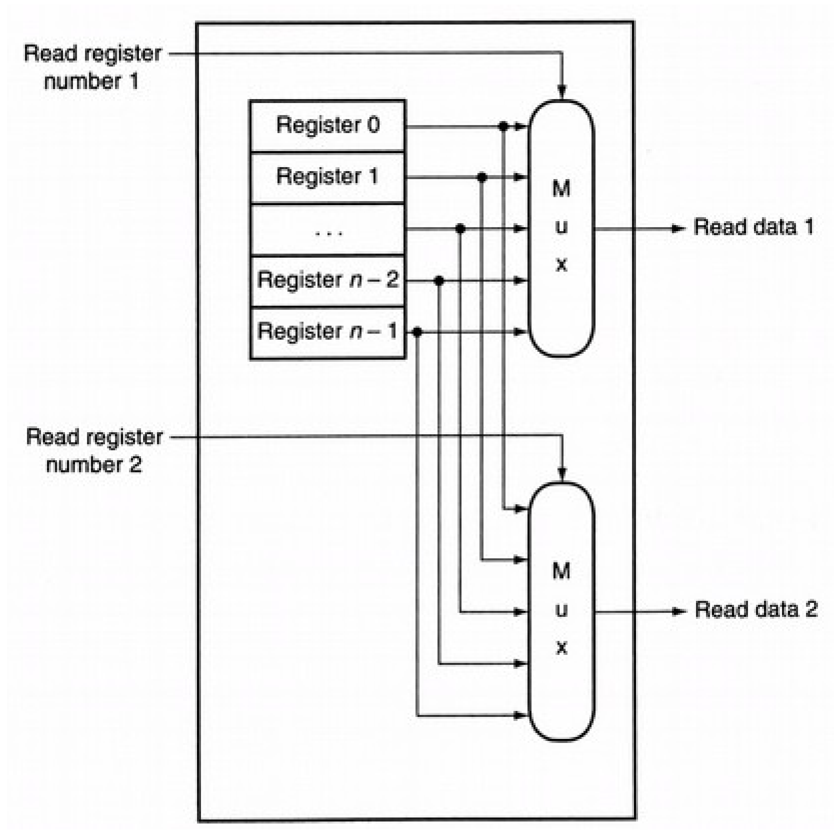




Register files

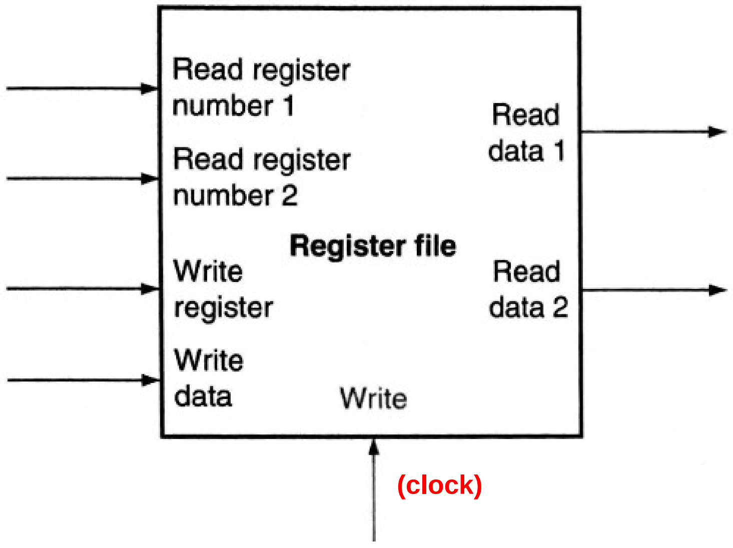
- Register files: multiple registers w/ read/write ports
  - Use multiplexors and decoders to differentiate



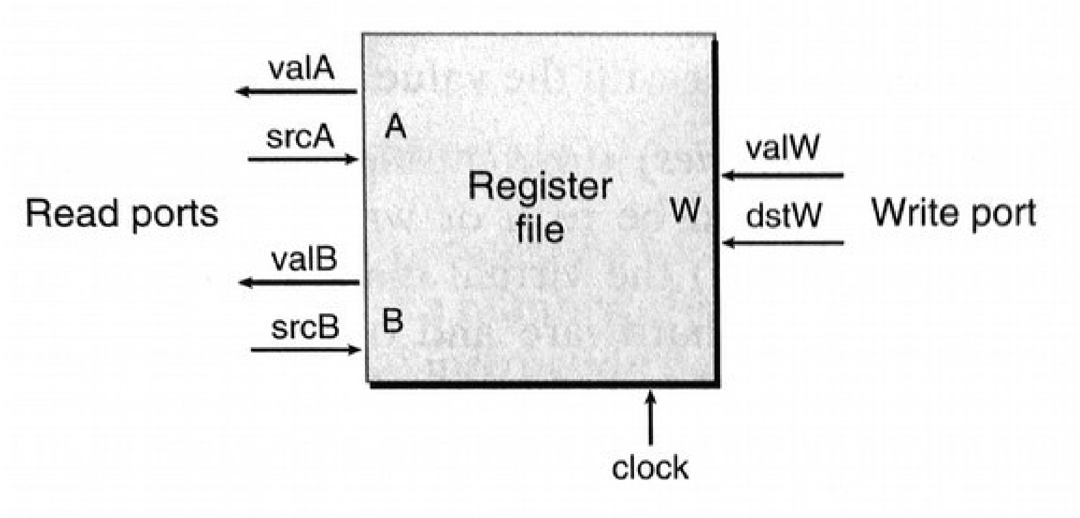


Register files

- Register files: multiple registers w/ read/write ports
  - Use multiplexors and decoders to differentiate



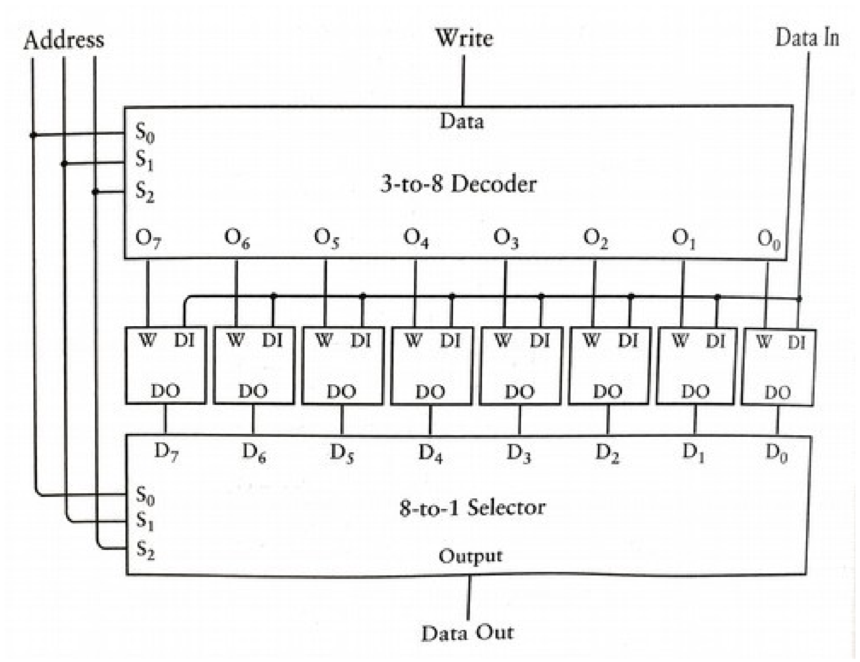
Canvas PDF version



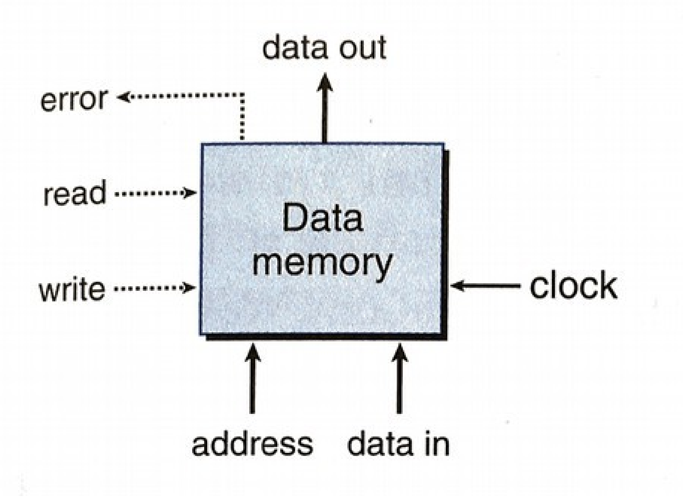
CS:APP version

Memory

- Memory: multiple flip-flops w/ address input
  - Random access memory (RAM) - can access any address at any time
  - Use decoder (translates n-bit number to 2^n "set" signals) to write data
  - Use selector (multiplexor) to read data


Address
Write
Data In
Data
3-to-8 Decoder
S_0 S_1 S_2
O_7 O_6 O_5 O_4 O_3 O_2 O_1 O_0
W DI DO
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
8-to-1 Selector
Output
Data Out

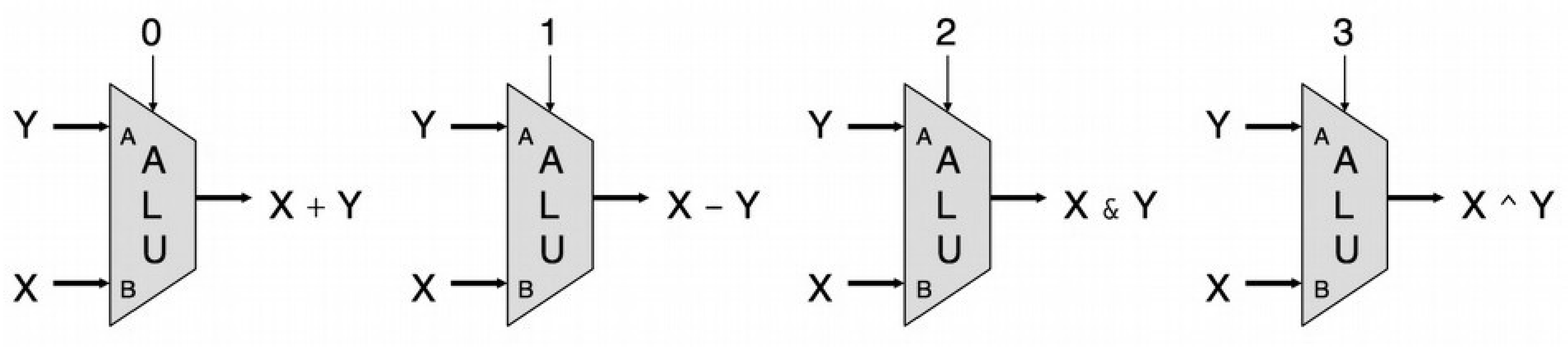
Single 8-element RAM array (3-bit addresses)



Abstraction of multiple RAM arrays

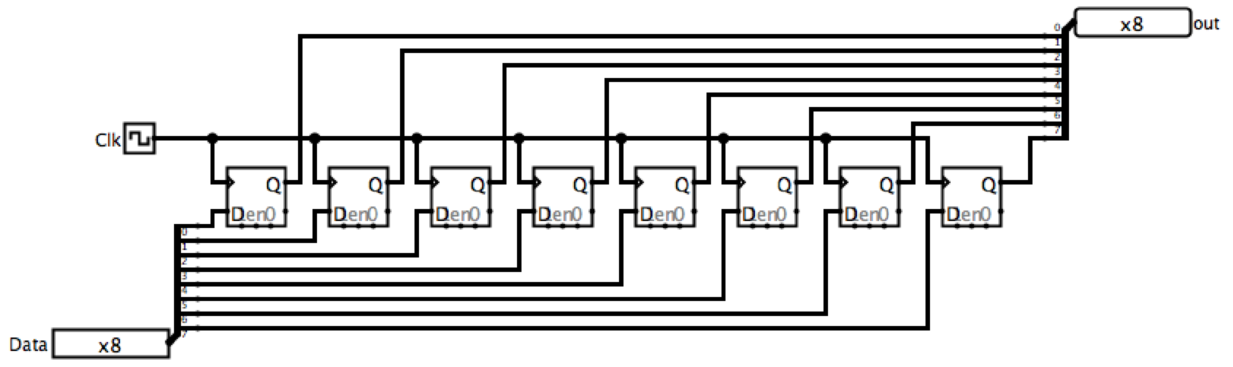
ALUs and memory

- Combine adders and multiplexors to make arithmetic/logic units
- Combine flip-flops to make register files and main memory



Basic Arithmetic Logic Unit (ALU)

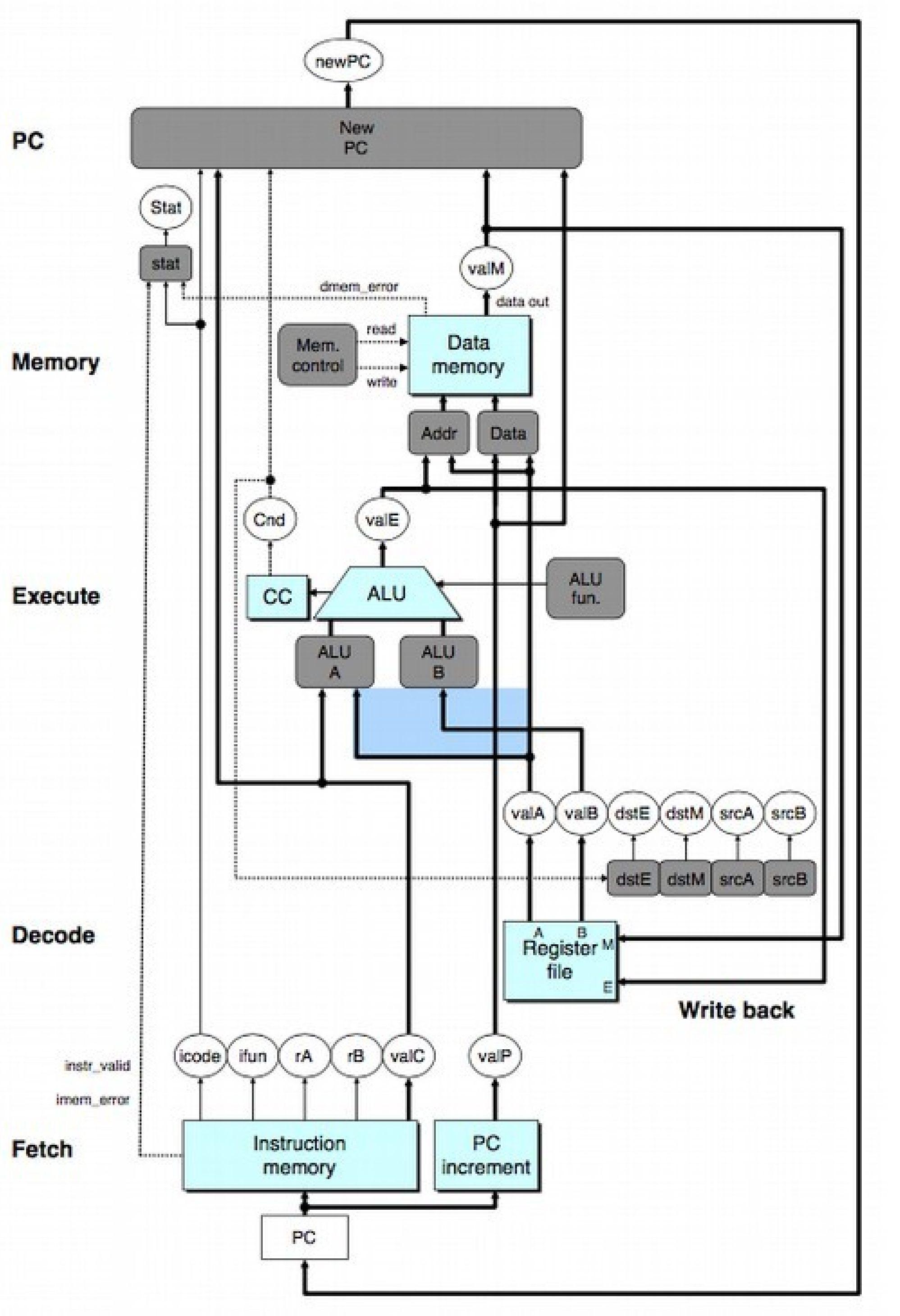
8-bit Register



CPUs

- Combine ALU with registers and memory to make CPUs

(on Thursday!)



newPC
PC
New PC
Stat
stat
dmem_error
valM
data out
Memory
read
Mem. control
Data memory
write
Addr
Data
Cnd
valE
Execute
CC
ALU
ALU fun.
ALU A
ALU B
valA
valB
dstE
dstM
srcA
srcB
dstE
dstM
srcA
srcB
Decode
A
B
M
Register file
E
Write back
icode
ifun
rA
rB
valC
valP
instr_valid
imem_error
Fetch
Instruction memory
PC increment
PC