CS 61C:
Great Ideas in Computer Architecture
Sequential Elements, Synchronous Digital Systems

Instructors:

Vladimir Stojanovic & Nicholas Weaver

http://inst.eecs.Berkeley.edu/~cs61c/sp16

Levels of Representation/Interpretation

High Level Language Program (e.g., C)

Compiler

Assembly Language Program (e.g., MIPS)

Assembler

Machine Language Program (MIPS)

Machine Interpretation

| Hardware Architecture Description (e.g., block diagrams) |
| :---: |

Architecture Implementation

Logic Circuit Description (Circuit Schematic Diagrams)

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```
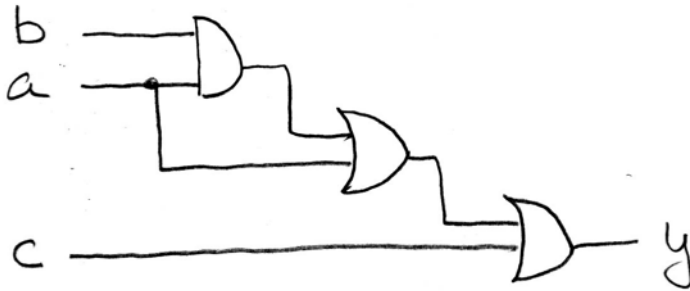
```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

Anything can be represented as a *number*, i.e., data or instructions

```
0000  1001  1100  0110  1010  1111  0101  1000
1010  1111  0101  1000  0000  1001  1100  0110
1100  0110  1010  1111  0101  1000  0000  1001
0101  1000  0000  1001  1100  0110  1010  1111
```

Register File
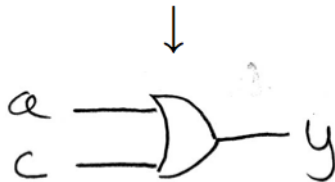
ALU

2
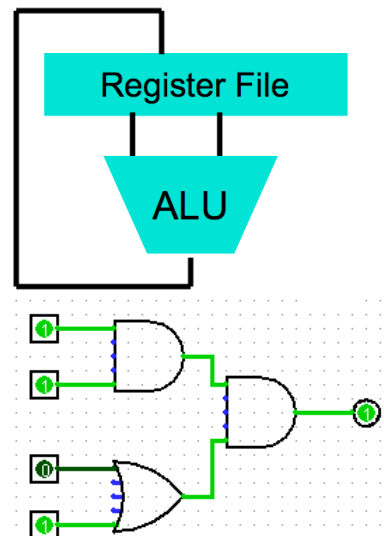
Boolean Algebra: Circuit & Algebraic Simplification

original circuit

$$y = ((ab) + a) + c$$

equation derived from original circuit

$$= ab + a + c$$

algebraic simplification

$$= a(b + 1) + c$$
$$= a(1) + c$$
$$= a + c$$

simplified circuit

Laws of Boolean Algebra

$X \overline{X} = 0$	$X + \overline{X} = 1$	Complementarity
$X \, 0 = 0$	$X + 1 = 1$	Laws of 0's and 1's
$X \, 1 = X$	$X + 0 = X$	Identities
$X \, X = X$	$X + X = X$	Idempotent Laws
$X \, Y = Y \, X$	$X + Y = Y + X$	Commutativity
$(X \, Y) \, Z = Z \, (Y \, Z)$	$(X + Y) + Z = Z + (Y + Z)$	Associativity
$X \, (Y + Z) = X \, Y + X \, Z$	$X + Y \, Z = (X + Y) \, (X + Z)$	Distribution
$X \, Y + X = X$	$(X + Y) \, X = X$	Uniting Theorem
$\overline{X} \, Y + X = X + Y$	$(\overline{X} + Y) \, X = X \, Y$	Uniting Theorem v. 2
$\overline{X \, Y} = \overline{X} + \overline{Y}$	$\overline{X + Y} = \overline{X} \, \overline{Y}$	DeMorgan's Law

Boolean Algebraic Simplification Example

$$y = ab + a + c$$

Boolean Algebraic Simplification Example

$$y = ab + a + c$$

a	b	c	y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

$$= a(b + 1) + c \quad \textit{distribution, identity}$$

$$= a(1) + c \quad \textit{law of 1's}$$

$$= a + c \quad \textit{identity}$$

Signals and Waveforms: Grouping
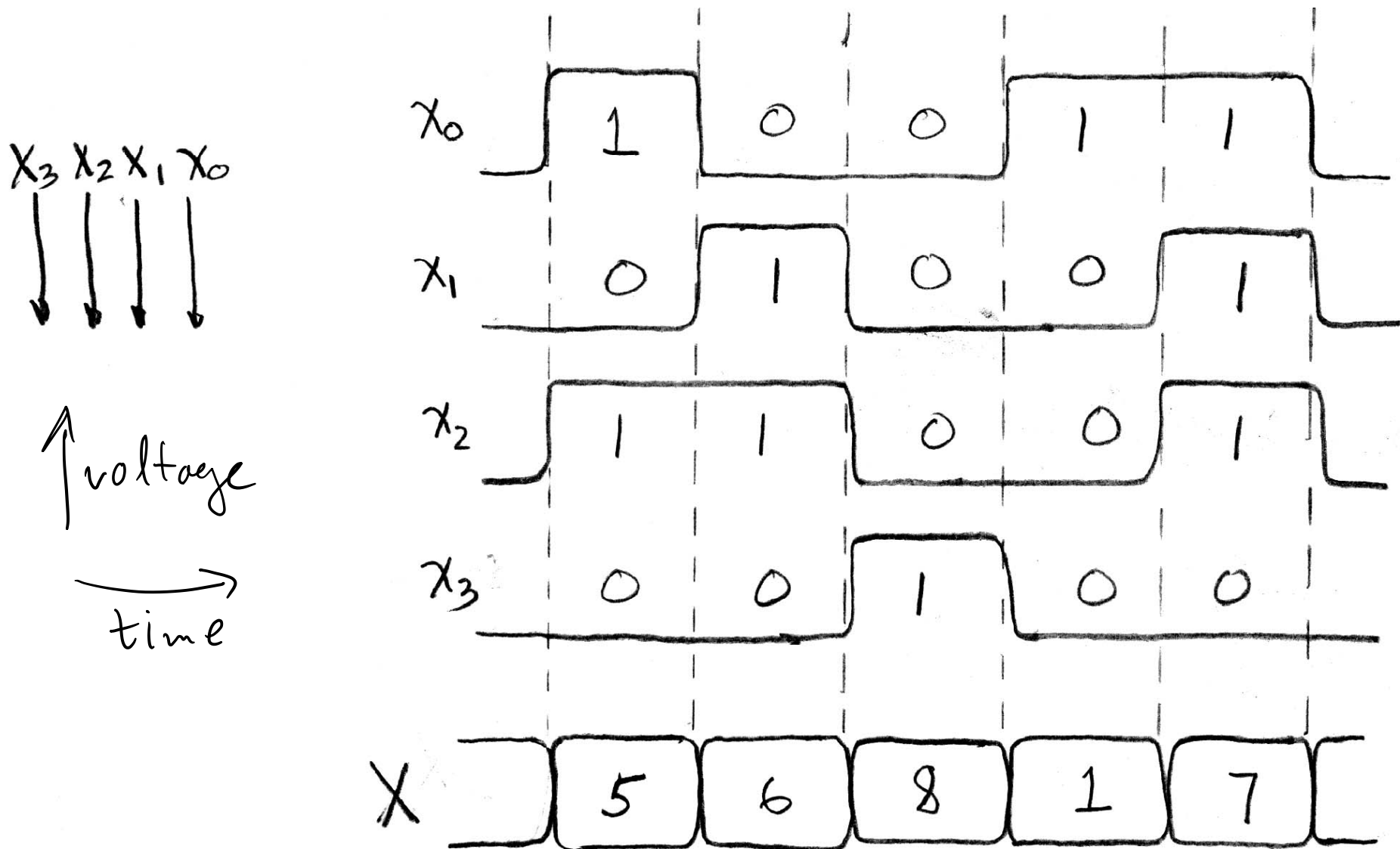
Signals and Waveforms: Circuit Delay

$$A = [a_3, a_2, a_1, a_0]$$
$$B = [b_3, b_2, b_1, b_0]$$

adder propagation delay
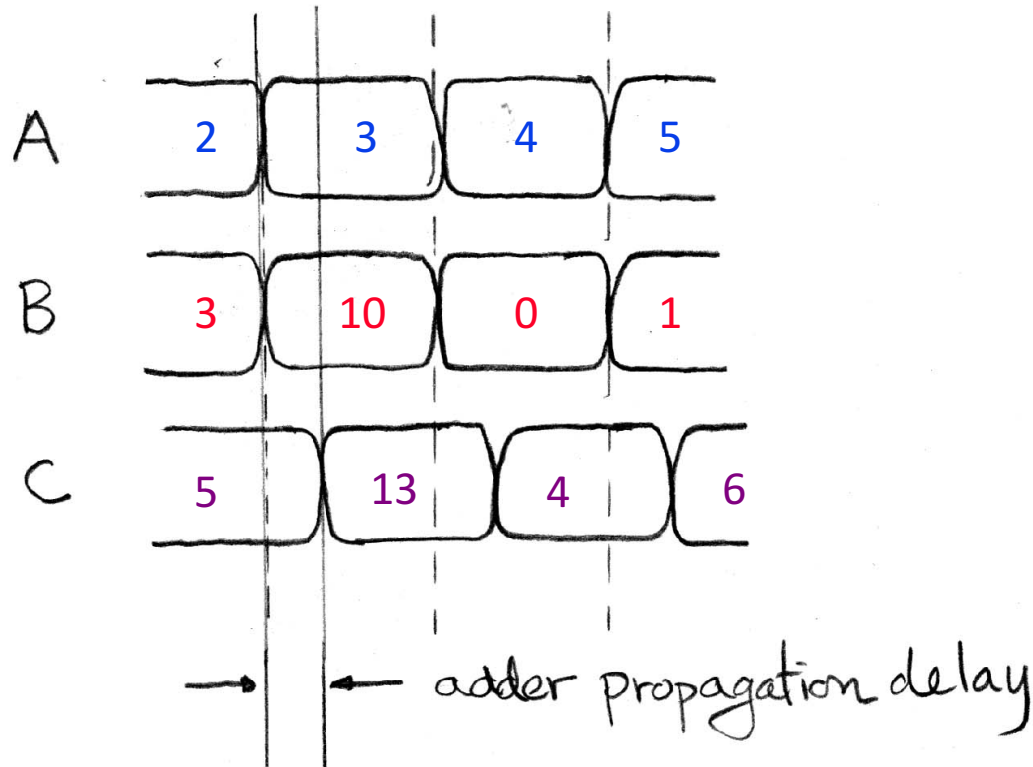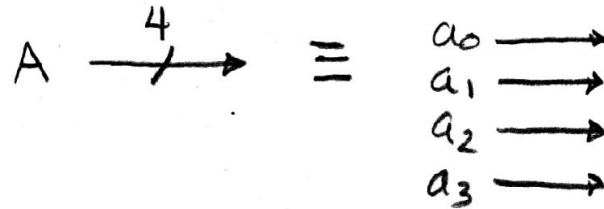
Sample Debugging Waveform
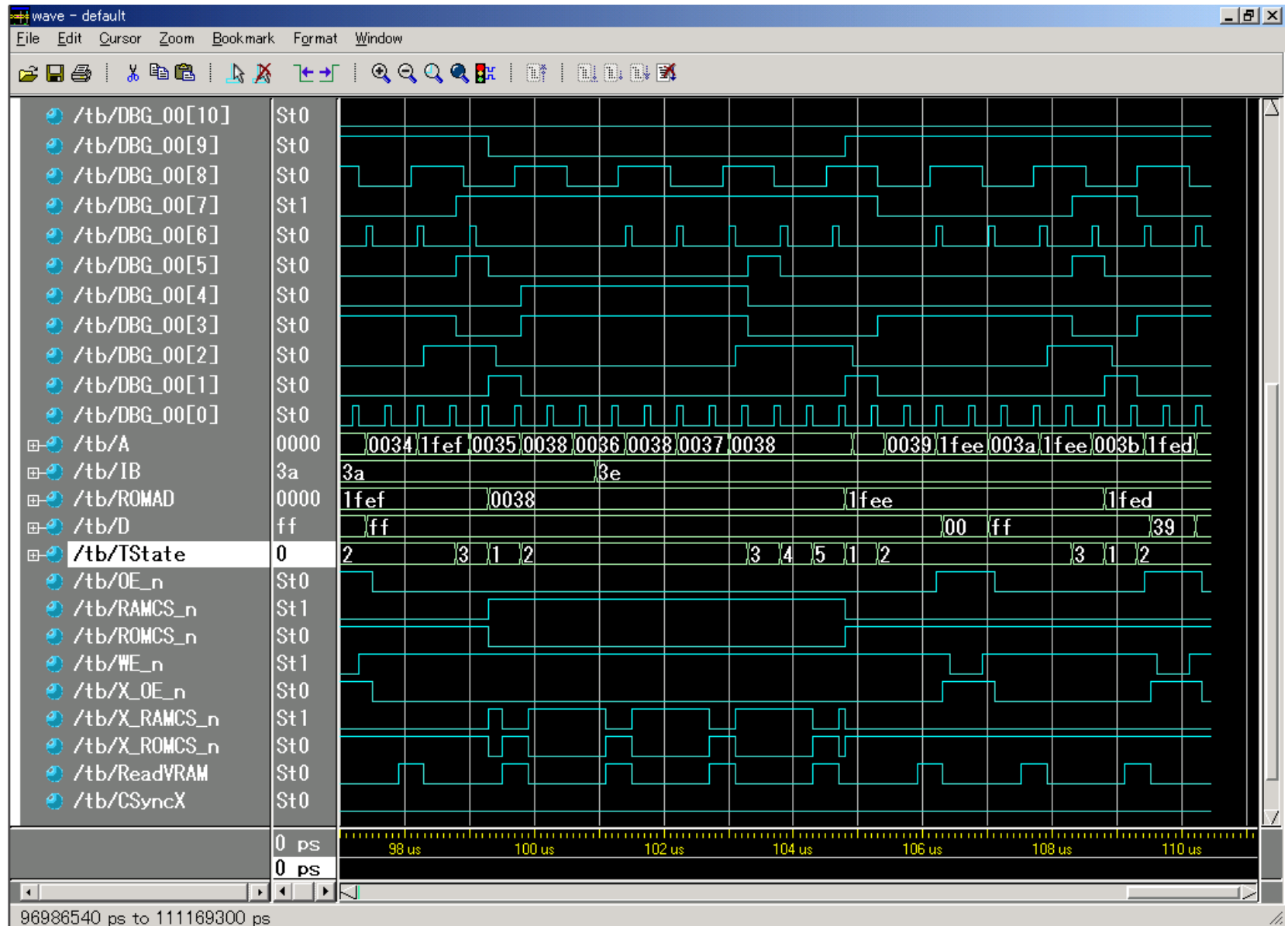
Clickers/Peer Instruction

- Simplify $Z = A+BC + \overline{A}\,(\overline{BC})$

- A: $Z = 0$

- B: $Z = \overline{A(1+ BC)}$

- C: $Z = (A + BC)$

- D: $Z = BC$

- E: $Z = 1$

Type of Circuits

- *Synchronous Digital Systems* consist of two basic types of circuits:
  - Combinational Logic (CL) circuits
    - Output is a function of the inputs only, not the history of its execution
    - E.g., circuits to add A, B (ALUs)
  - Sequential Logic (SL)
    - Circuits that "remember" or store information
    - aka "State Elements"
    - E.g., memories and registers (Registers)

Uses for State Elements

- Place to store values for later re-use:
  - Register files (like $1-$31 in MIPS)
  - Memory (caches and main memory)
- *Help control flow of information between combinational logic blocks*
  - State elements hold up the movement of information at input to combinational logic blocks to allow for orderly passage

Accumulator Example

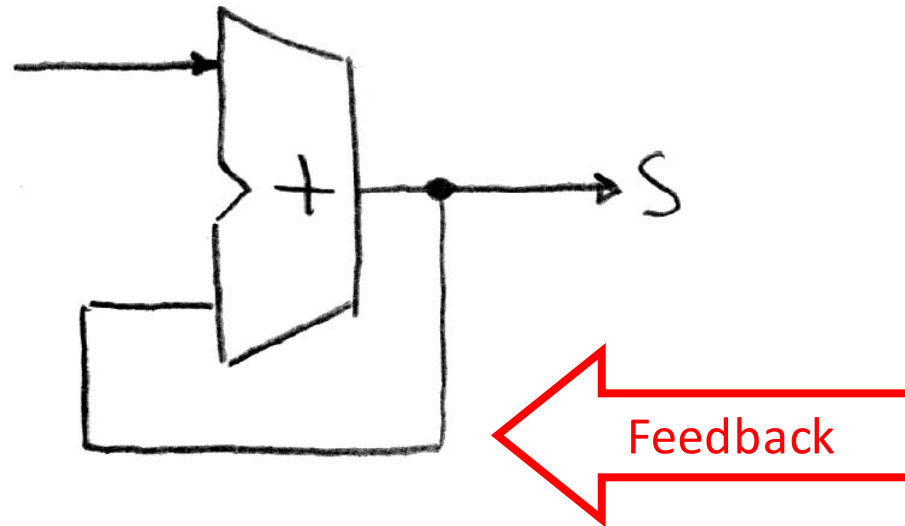Why do we need to control the flow of information?
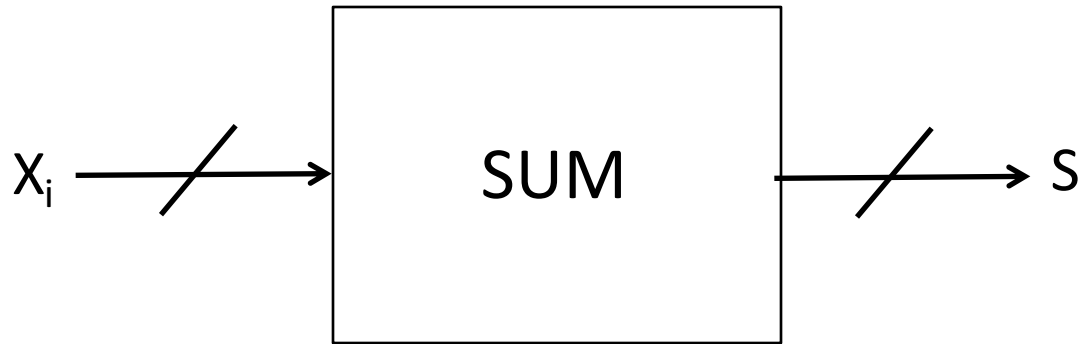
Want:
```
S=0;
for (i=0;i<n;i++)
    S = S + Xᵢ
```

$$S=0;$$
$$\text{for } (i=0; i<n; i++)$$
$$S = S + X_i$$

Assume:
- Each X value is applied in succession, one per cycle
- After n cycles the sum is present on S
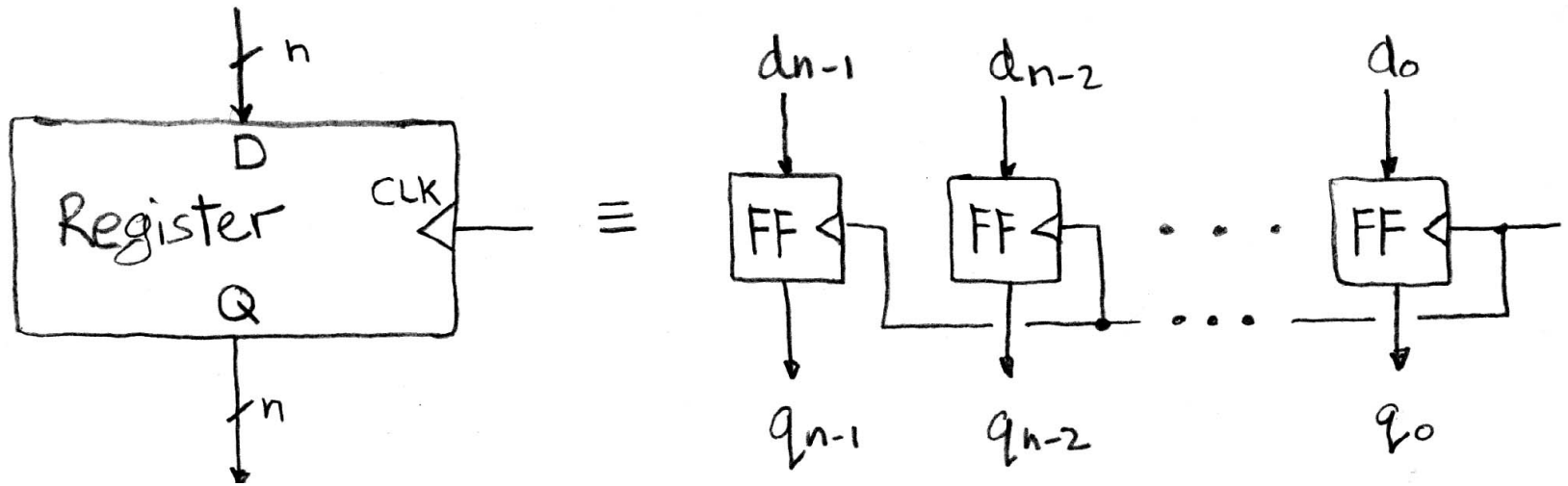
First Try: Does this work?

No!

Reason #1: How to control the next iteration of the 'for' loop?
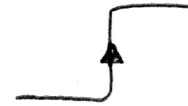
Reason #2: How do we say: 'S=0'?

Register Internals

- n instances of a "Flip-Flop"

- Flip-flop name because the output flips and flops between 0 and 1

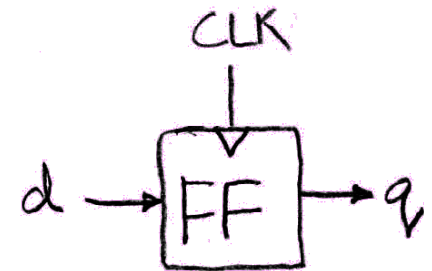- D is "data input", Q is "data output"
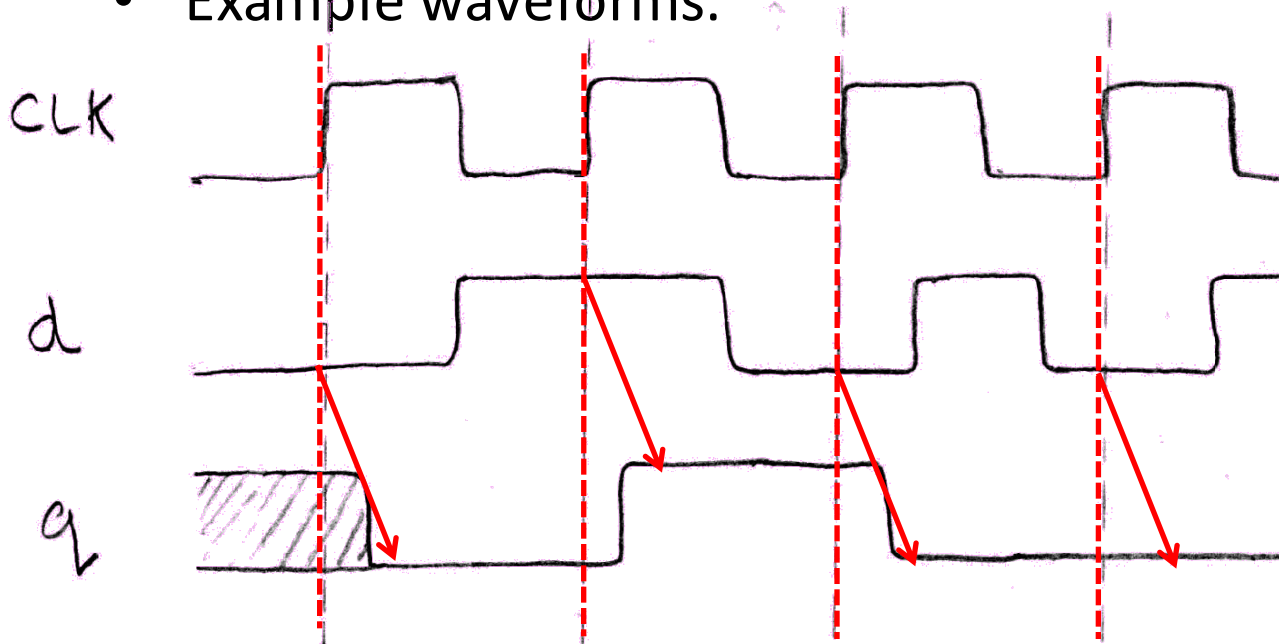
- Also called "D-type Flip-Flop"

Flip-Flop Operation

- Edge-triggered d-type flip-flop
  - This one is "positive edge-triggered"

- "On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored."
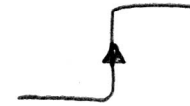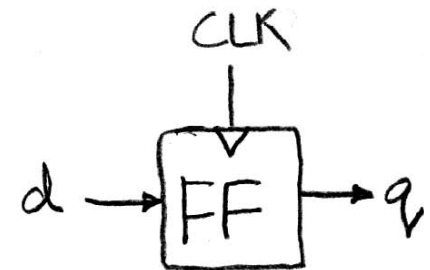
- Example waveforms:

Flip-Flop Timing

- Edge-triggered d-type flip-flop
  - This one is "positive edge-triggered"

- "On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored."
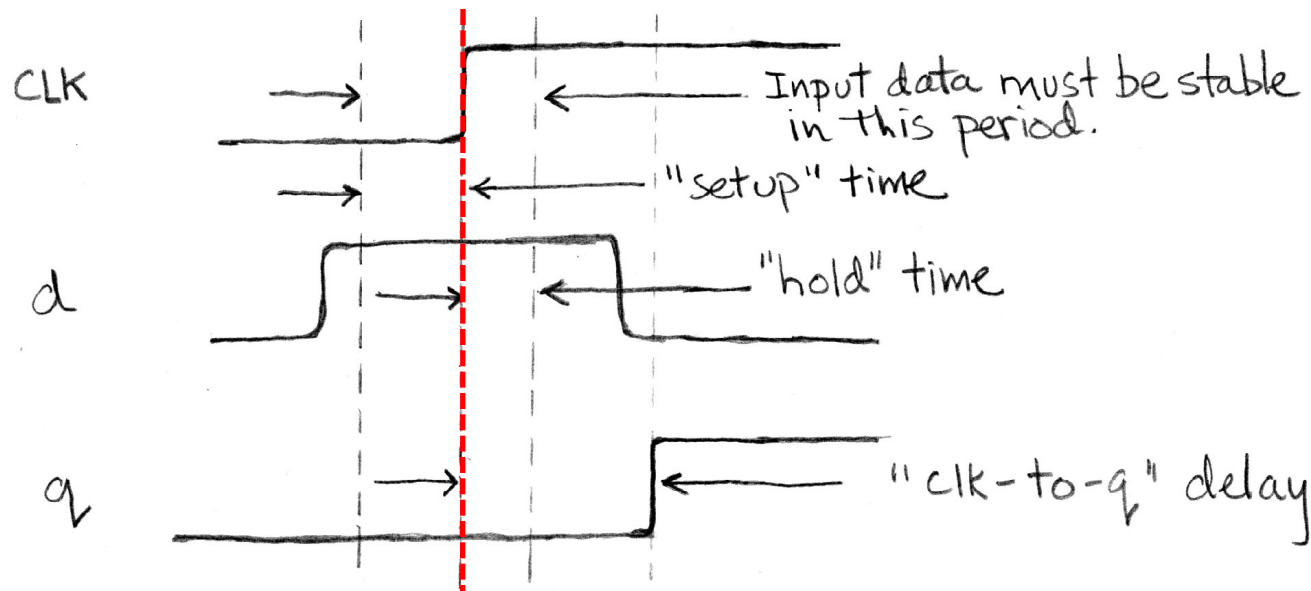
- Example waveforms (more detail):

CLK

d

q

Input data must be stable in this period.

"setup" time

"hold" time

"clk-to-q" delay

CLK

d → FF → q

Camera Analogy Timing Terms

- Want to take a portrait – timing right before and after taking picture

- *Set up time* – don't move since about to take picture (open camera shutter)

- *Hold time* – need to hold still after shutter opens until camera shutter closes

- *Time click to data* – time from open shutter until can see image on output (viewscreen)

Hardware Timing Terms

- Setup Time: when the input must be stable *before* the edge of the CLK

- Hold Time: when the input must be stable *after* the edge of the CLK

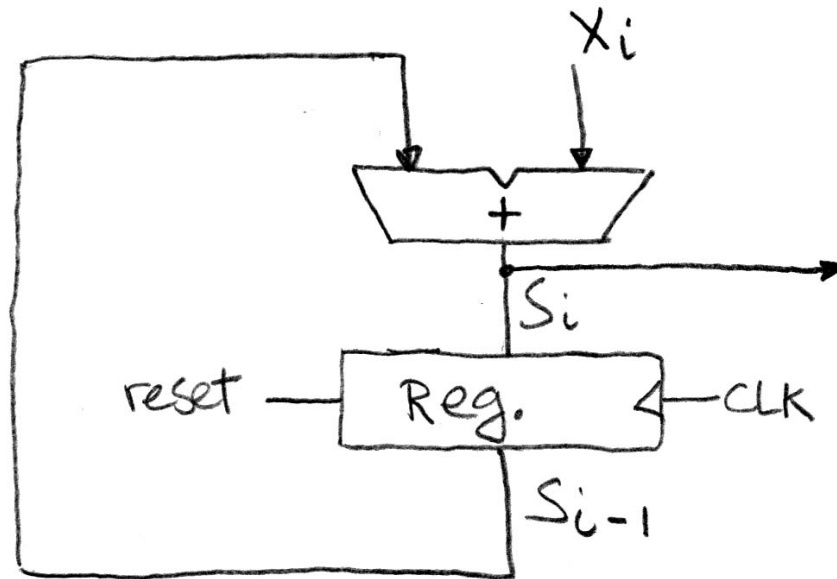- "CLK-to-Q" Delay: how long it takes the output to change, measured from the edge of the CLK
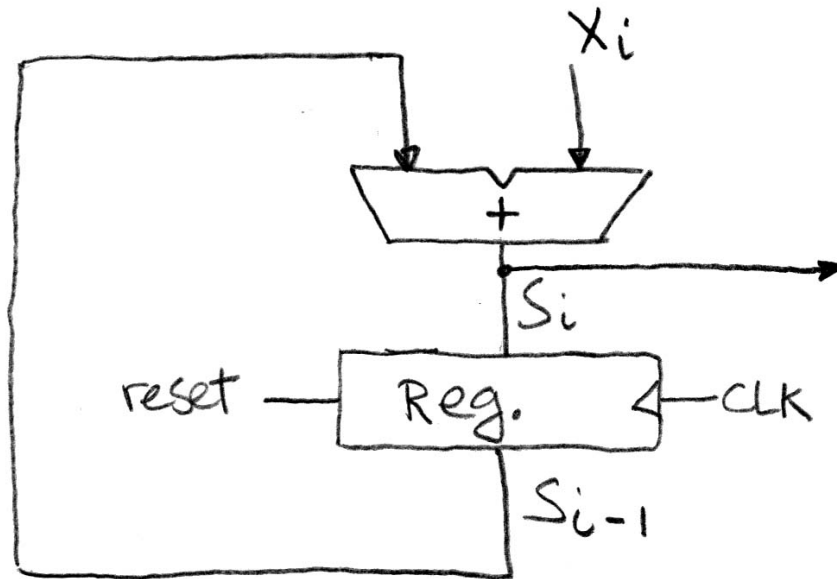
- Reset input to register is used to force it to all zeros (takes priority over D input).

- S_{i-1} holds the result of the ith-1 iteration.

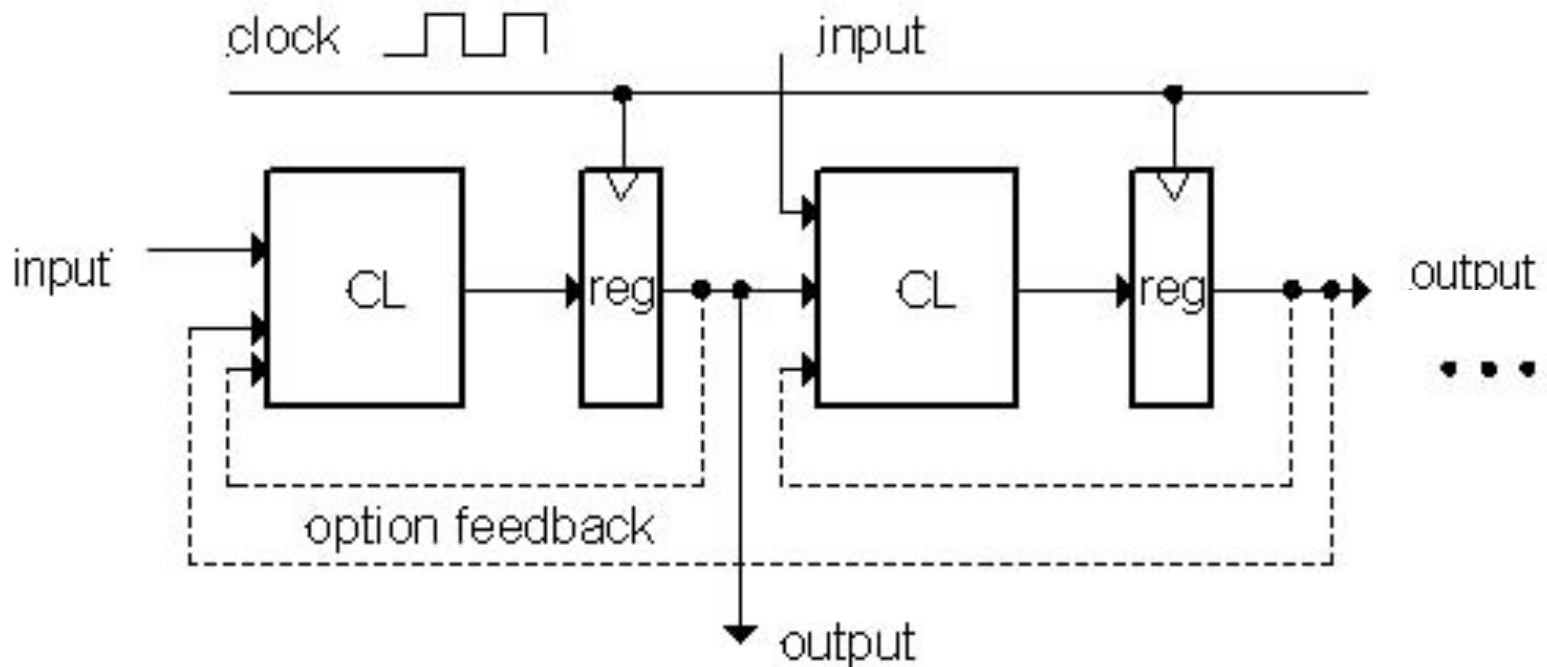- Analyze circuit timing starting at the output of the register.
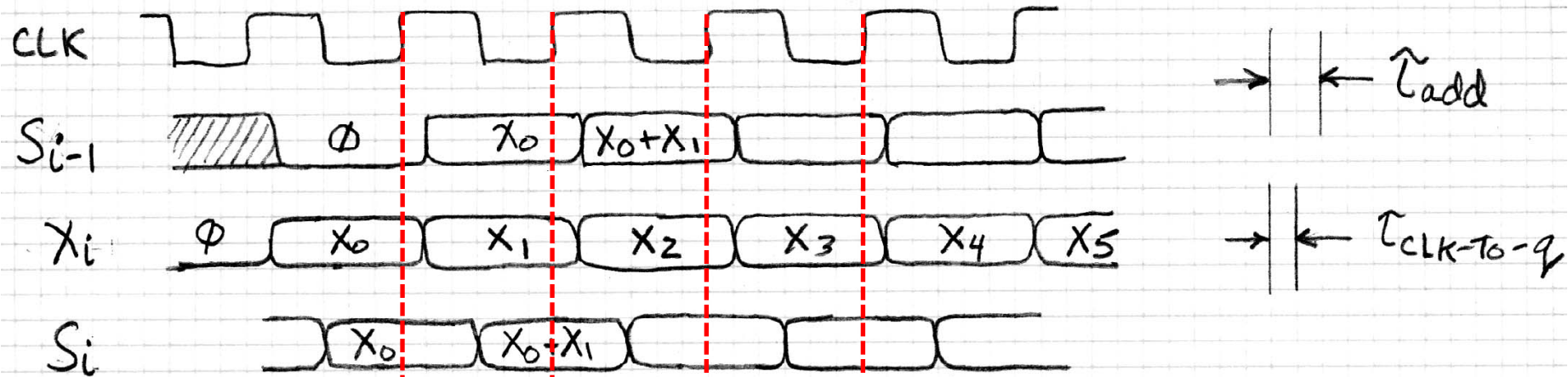
- reset signal shown.

- Also, in practice X might not arrive to the adder at the same time as S_{i-1}

- S_i temporarily is wrong, but register always captures correct value.

- In good circuits, instability never happens around rising edge of clk.
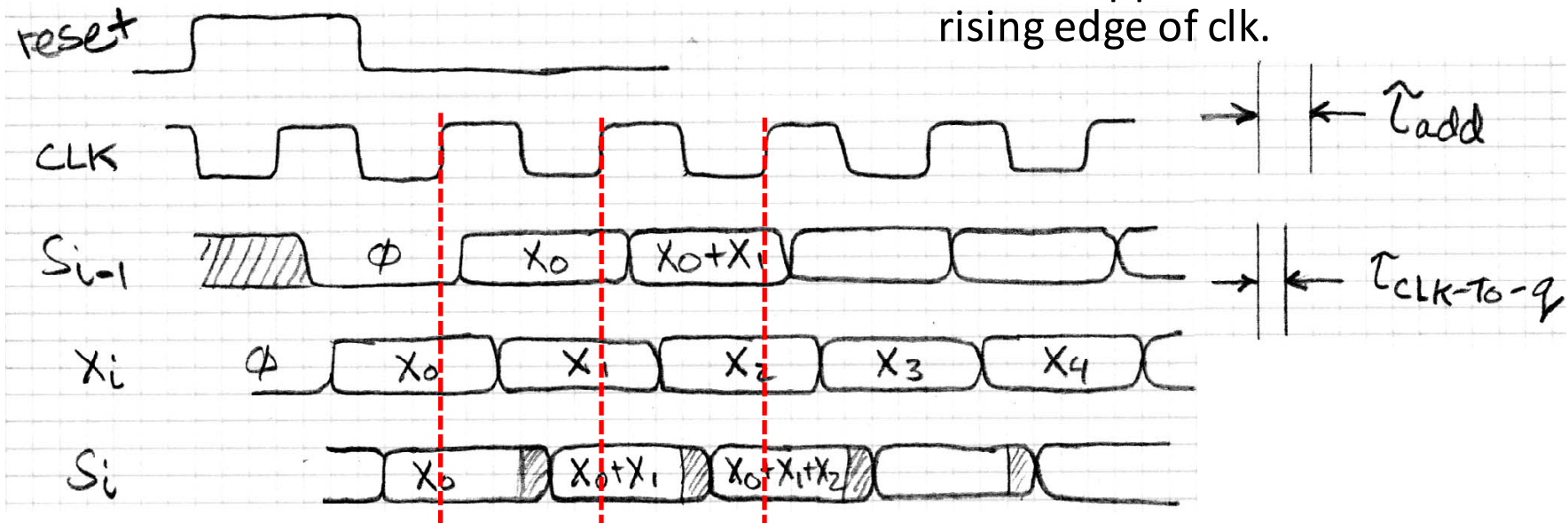
Model for Synchronous Systems

- Collection of Combinational Logic blocks separated by registers
- Feedback is optional
- Clock signal(s) connects only to clock input of registers
- Clock (CLK): steady square wave that synchronizes the system
- Register: several bits of state that samples on rising edge of CLK (positive edge-triggered) or falling edge (negative edge-triggered)

Maximum Clock Frequency

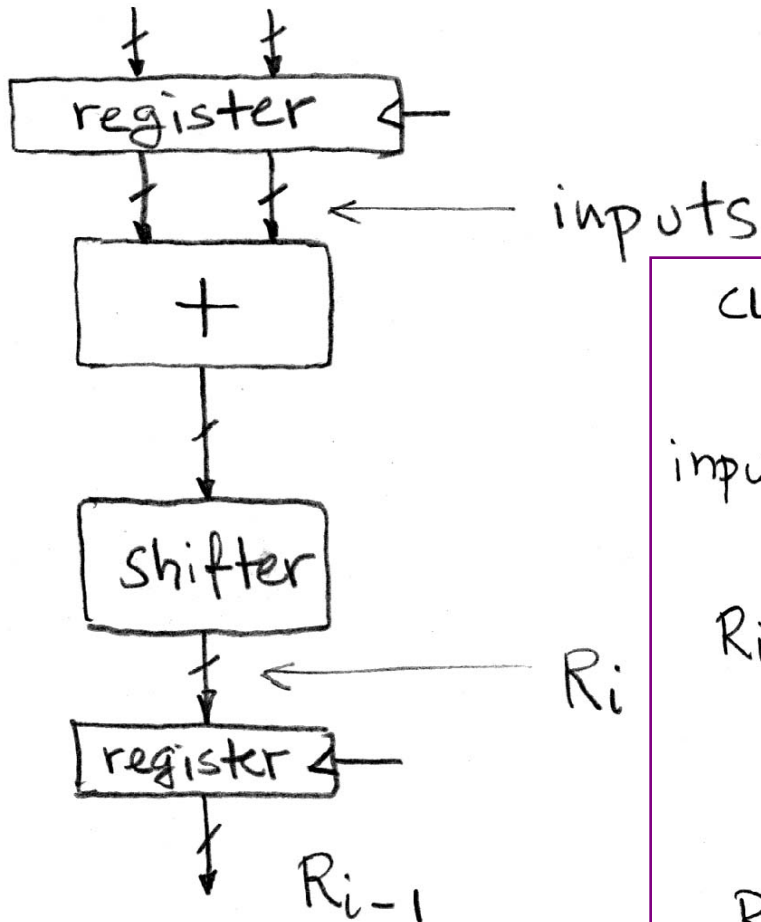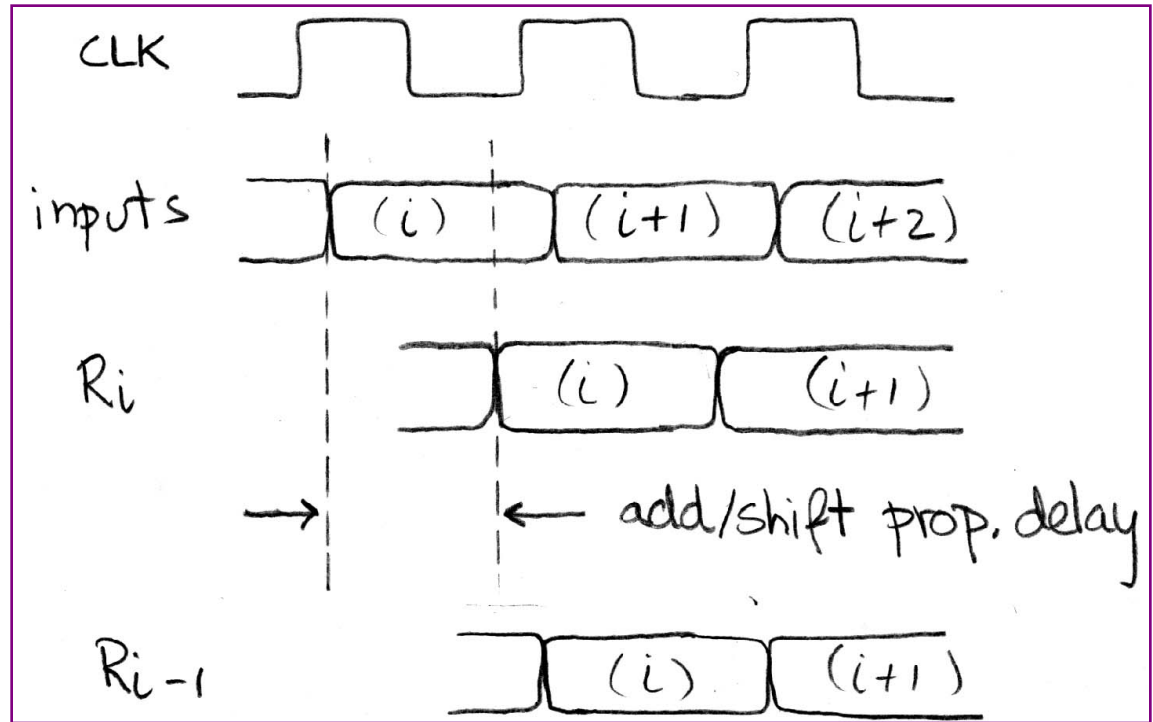- What is the maximum frequency of this circuit?
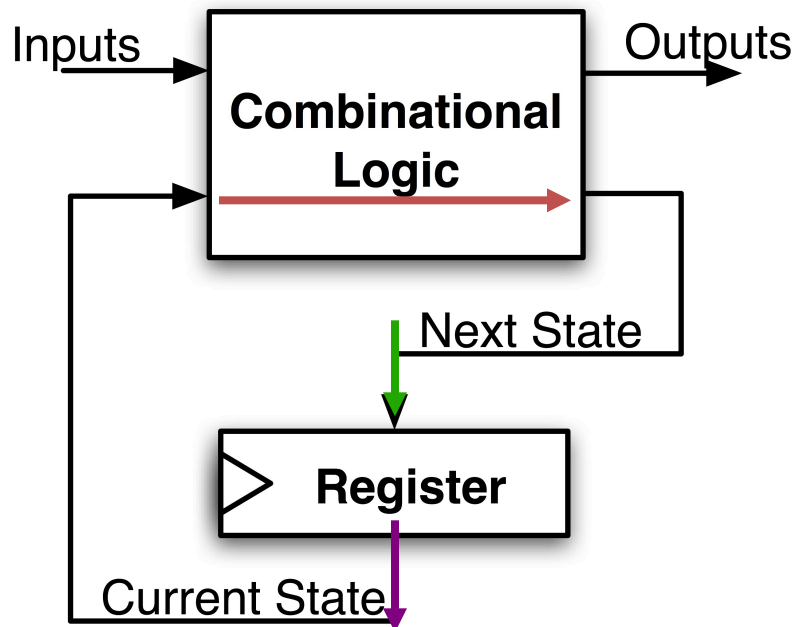
Hint:
Frequency = 1/Period

Period = Max Delay = CLK-to-Q Delay + CL Delay + Setup Time
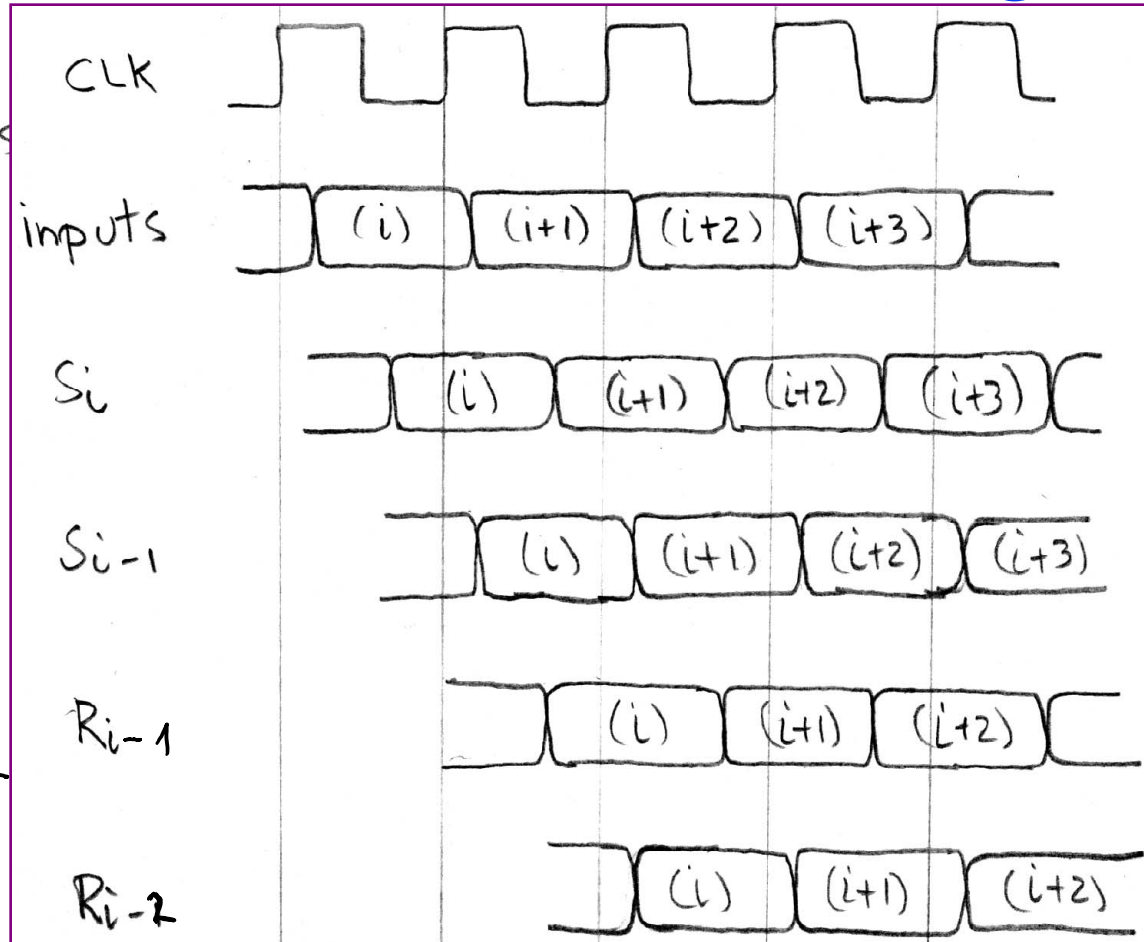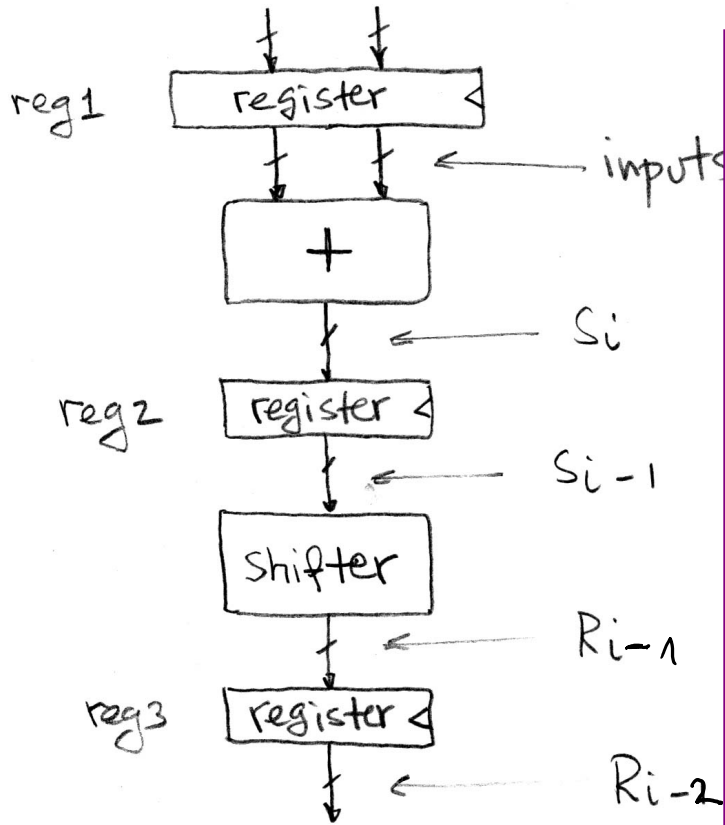
Critical Paths

Timing…

Note: delay of 1 clock cycle from input to output.
Clock period limited by propagation delay of adder/shifter.

Pipelining to improve performance

Timing…

- Insertion of register allows higher clock frequency
- More outputs per second (higher bandwidth)
- But each individual result takes longer (greater latency)

Recap of Timing Terms

- Clock (CLK) - steady square wave that synchronizes system

- Setup Time - when the input must be stable <u>before</u> the rising edge of the CLK

- Hold Time - when the input must be stable <u>after</u> the rising edge of the CLK

- "CLK-to-Q" Delay - how long it takes the output to change, measured from the rising edge of the CLK

- Flip-flop - one bit of state that samples every rising edge of the CLK (positive edge-triggered)

- Register - several bits of state that samples on rising edge of CLK or on LOAD (positive edge-triggered)

Clickers/Peer Instruction
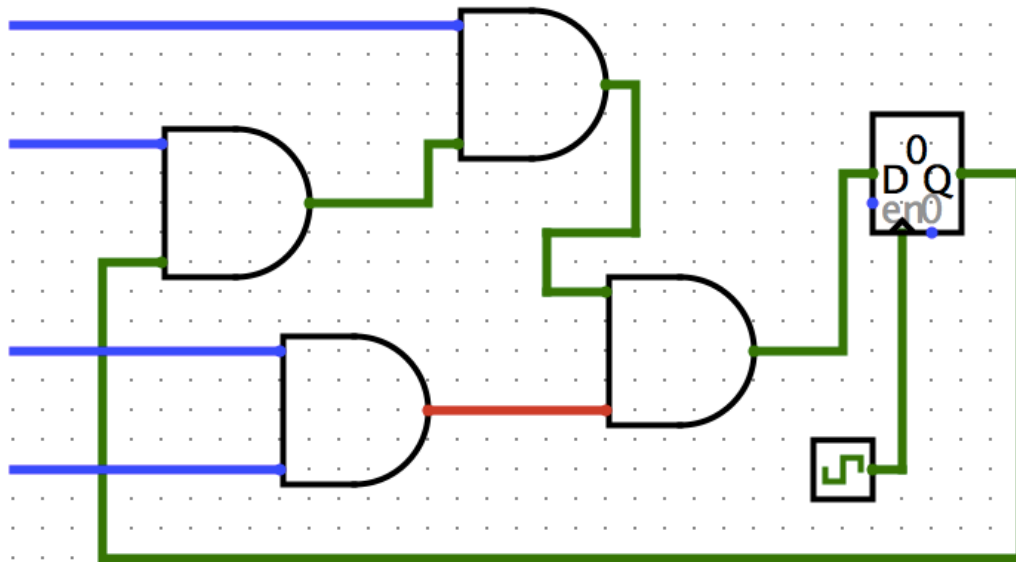
Clock->Q 1ns
Setup 1ns
Hold 1ns
AND delay 1ns

What is maximum clock frequency? (assume all unconnected inputs come from some register)

- A: 5 GHz
- B: 200 MHz
- C: 500 MHz
- D: 1/7 GHz
- E: 1/6 GHz

Administrivia

- No lecture on Wed – study for midterm
- Midterm 1 is on Thu 2/25
  - Time 6-8pm
  - Location
    - 2050 VLSB(aa-lz)
    - 1 Pimentel(ma-zz)
  - Covers up to and including 02/17 lecture (CALL 2)
  - 1 handwritten, double sided, 8.5"x11" cheat sheet
  - We'll give you MIPS green sheet
- Emails sent-out to students requiring special accommodation for the exam – please respond

Study Advice

1. Review slides, book, worksheets, etc. and add to your cheatsheet as you do so
   a. This step is not the end
2. Take a mock exam in the allotted time, using only your cheatsheet
3. Go over solutions, look at *why* the answers are what they are (especially for questions you answered incorrectly)
4. Update cheatsheet as necessary
5. if (!perfect) goto 2;