



Combinatorial Logic Design

Multiplexers and ALUs

CS 64: Computer Organization and Design Logic
Lecture #14

Ziad Matni
Dept. of Computer Science, UCSB

Administrative

- Remaining on the calendar... *This supersedes anything on the syllabus*

DATE	TOPIC	ASSIGNMENTS
Thu. 3/1	Simplifying Digital Logic Functions	Lab 6 (due Fri. 3/2)
Tue. 3/6	Combinatorial Logic	
Thu. 3/8	Sequential Logic	Lab 7 (due Fri. 3/9)
Tue. 3/13	Finite State Machines	
Thu. 3/15	Ethics	Labs 8 and 9 (due Fri. 3/16)

Lecture Outline

- Combinatorial Logic
- Selection using Multiplexers
- Basic ALU Design

Exercise 2

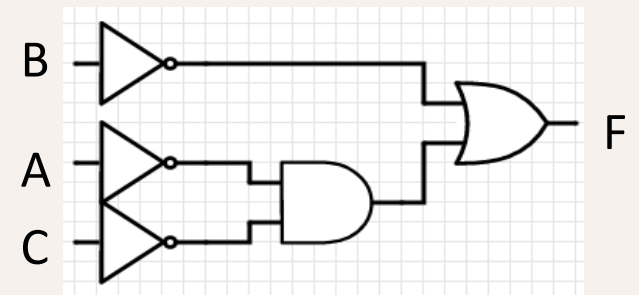
Class Ex.

- Given the following truth table, draw the resulting logic circuit

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

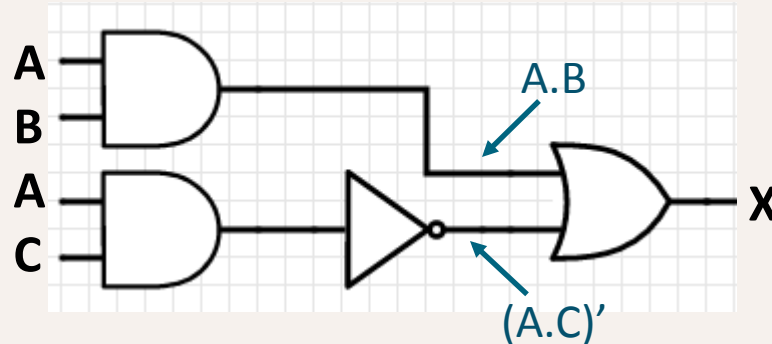
		AB			
		00	01	11	10
C	0	1	1		1
	1	1			1

$$F(A,B,C) = B' + A'.C'$$



Exercise 3

- Given the following schematic of a circuit, (a) write the function and (b) fill out the truth table:



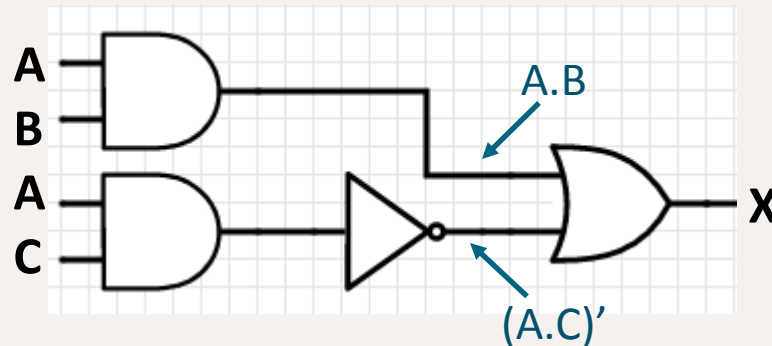
$$X = A.B + (A.C)'$$

(note that also means: $X = A.B + A' + C'$)

A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Exercise 3

- Given the following schematic of a circuit, (a) write the function and (b) fill out the truth table:



$$X = A.B + (A.C)'$$

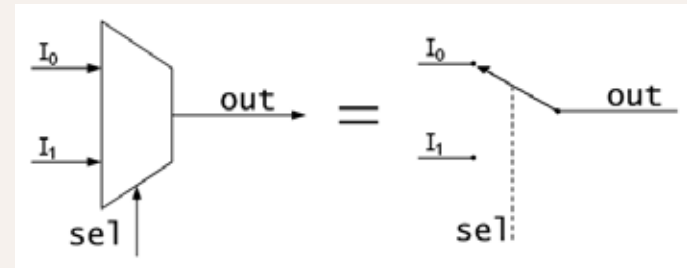
(note that also means: $X = A.B + A' + C'$)

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Multiplexer

(Mux for short)

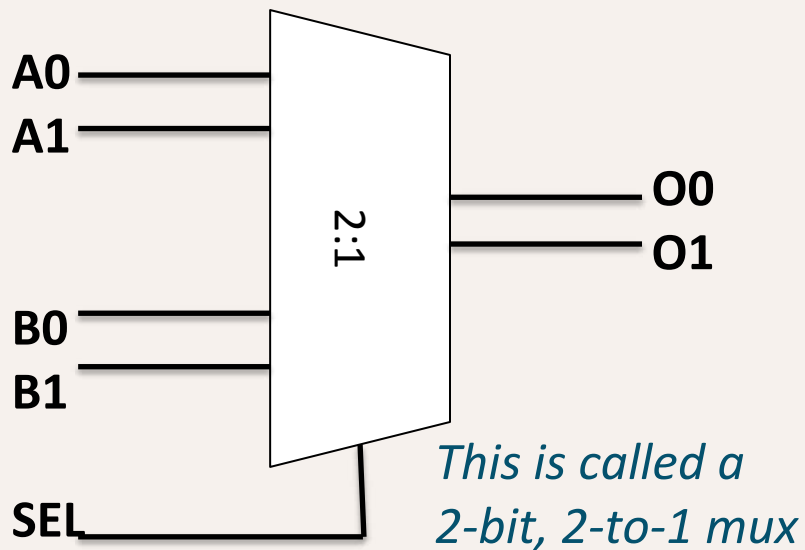
- Typically has 3 *groups of* inputs and 1 output
 - IN: 2 data , 1 select
 - OUT: 1 data



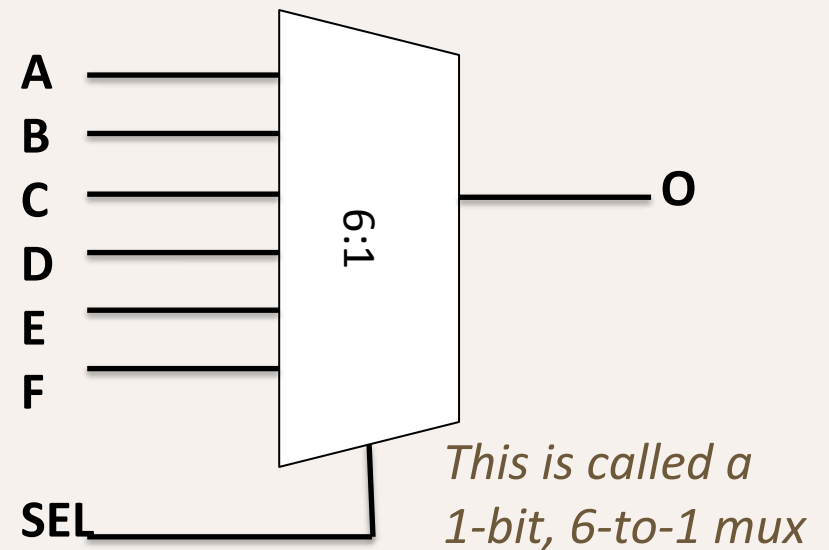
- 1 of the input data lines gets selected to become the output, based on the 3rd (select) input
 - If “Sel” = 0, then I_0 gets to be the output
 - If “Sel” = 1, then I_1 gets to be the output
- The opposite of a Mux is called a **Demultiplexer** (or **Demux**)

Mux Configurations

Muxes can have I/O that are multiple bits

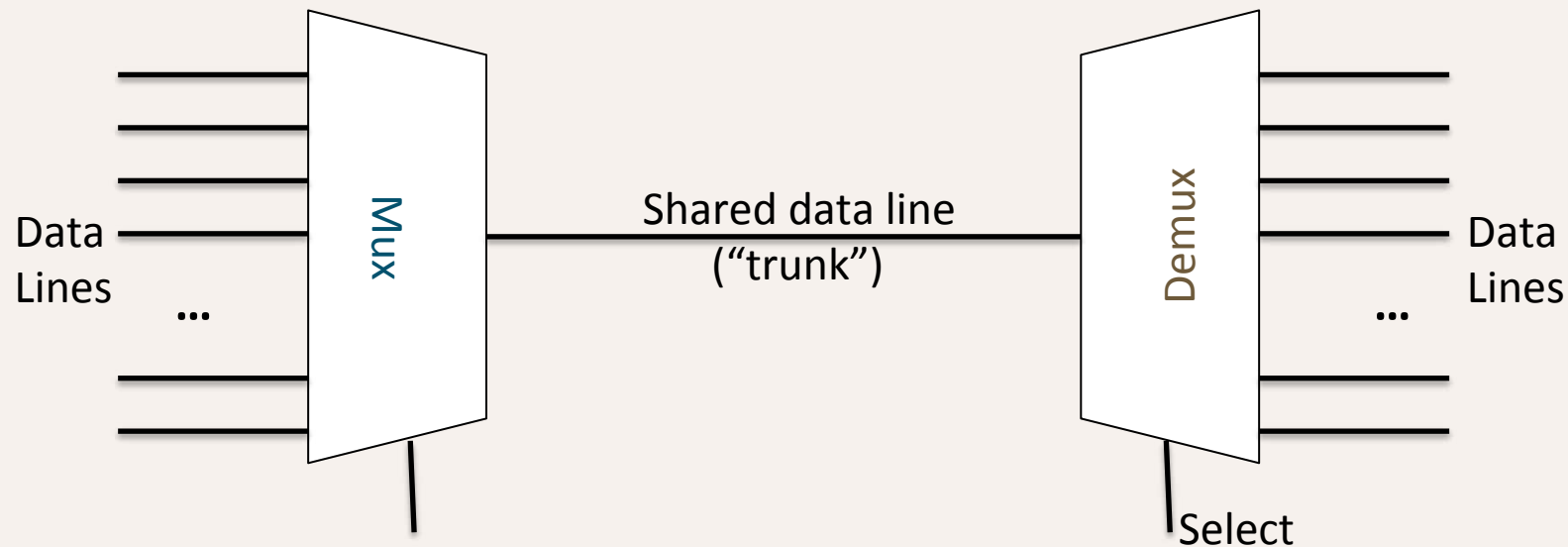


Or they can have more than two data inputs



The Use of Multiplexers

- Makes it possible for several signals (variables) to share one resource
 - Very commonly used in data communication lines



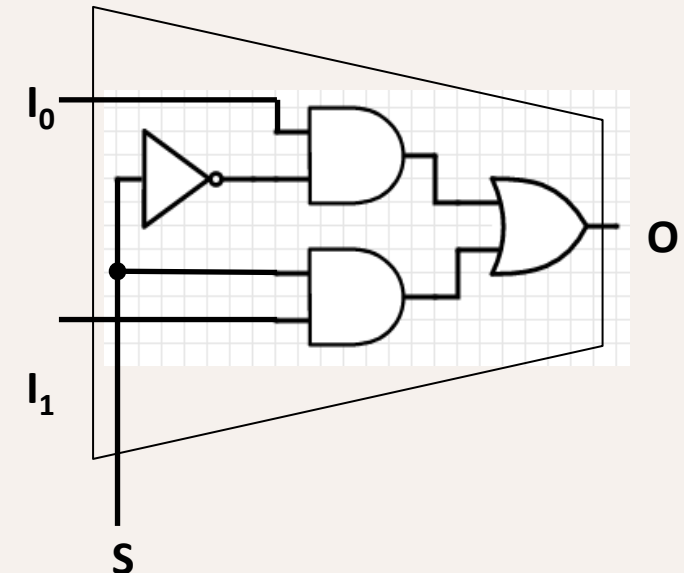
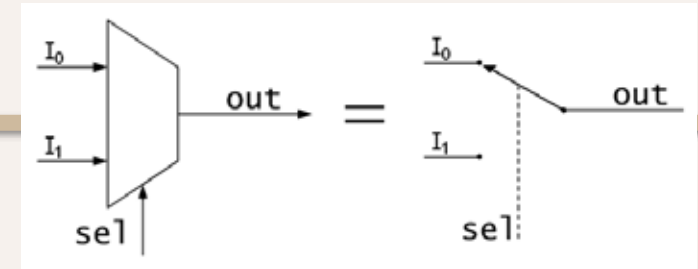
Mux Truth Table and Logic Circuit

1-bit Mux

I_0	I_1	S	O
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

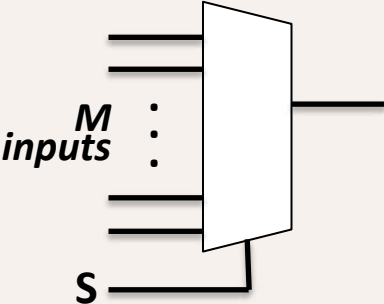
S	$I_0 \ I_1$			
	00	01	11	10
0			1	1
1		1	1	

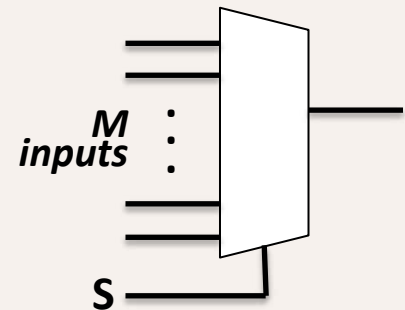
$$O = S \cdot I_1 + S' \cdot I_0$$



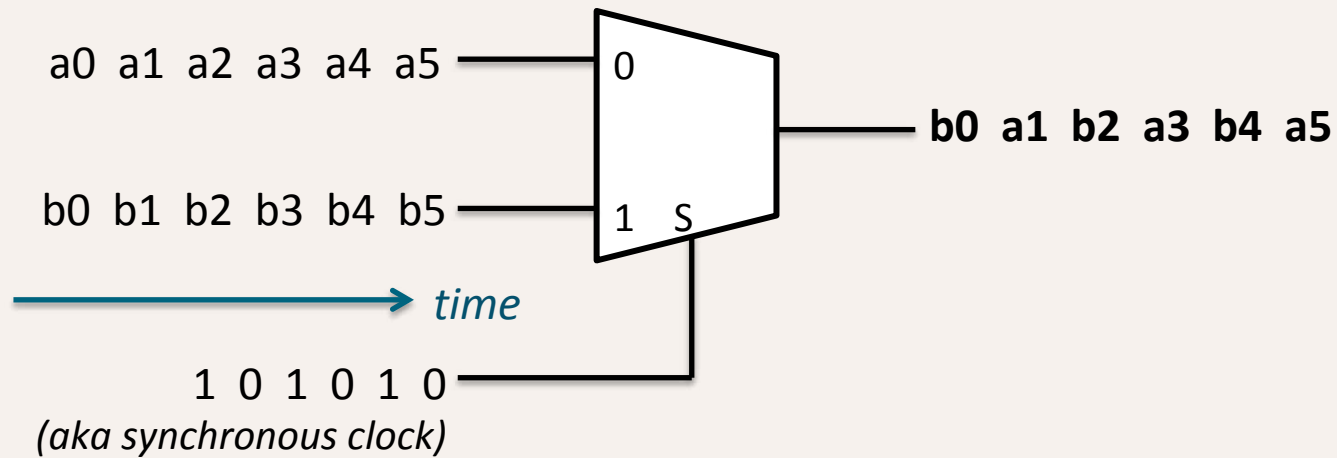
• = lines are physically connected

Beyond 1-bit Muxes

- General mux form: **N-bit, M-to-1**
 - Where:
N = how wide the data bus is (in bits, min. 1)
M = how many inputs to the mux (min. 2)
 - The “select” input (S) has to be able to select
1 out of M inputs
 - So, if $M = 2$, S should be at least 1 bit ($S = 0$ for one line, $S = 1$ for the other)
 - But if $M = 3$, S should be at least **2 bits** (why?)
 - If $M = 4$, S should be at least ???
 - At least 2 bits
 - If $M = 5$, S will have to be ???
 - At least 3 bits
- 
- The diagram shows a standard multiplexer symbol. On the left side, there are four horizontal lines representing inputs. To their left, the text "inputs" is written, followed by a vertical ellipsis and the letter "M". On the bottom left, a single line represents the select input, labeled "S". A trapezoidal box represents the multiplexer itself, with its wider base on the left where the inputs and select line enter, and a narrower base on the right. A single output line extends from the right side of the box.



What Does This Circuit Do?

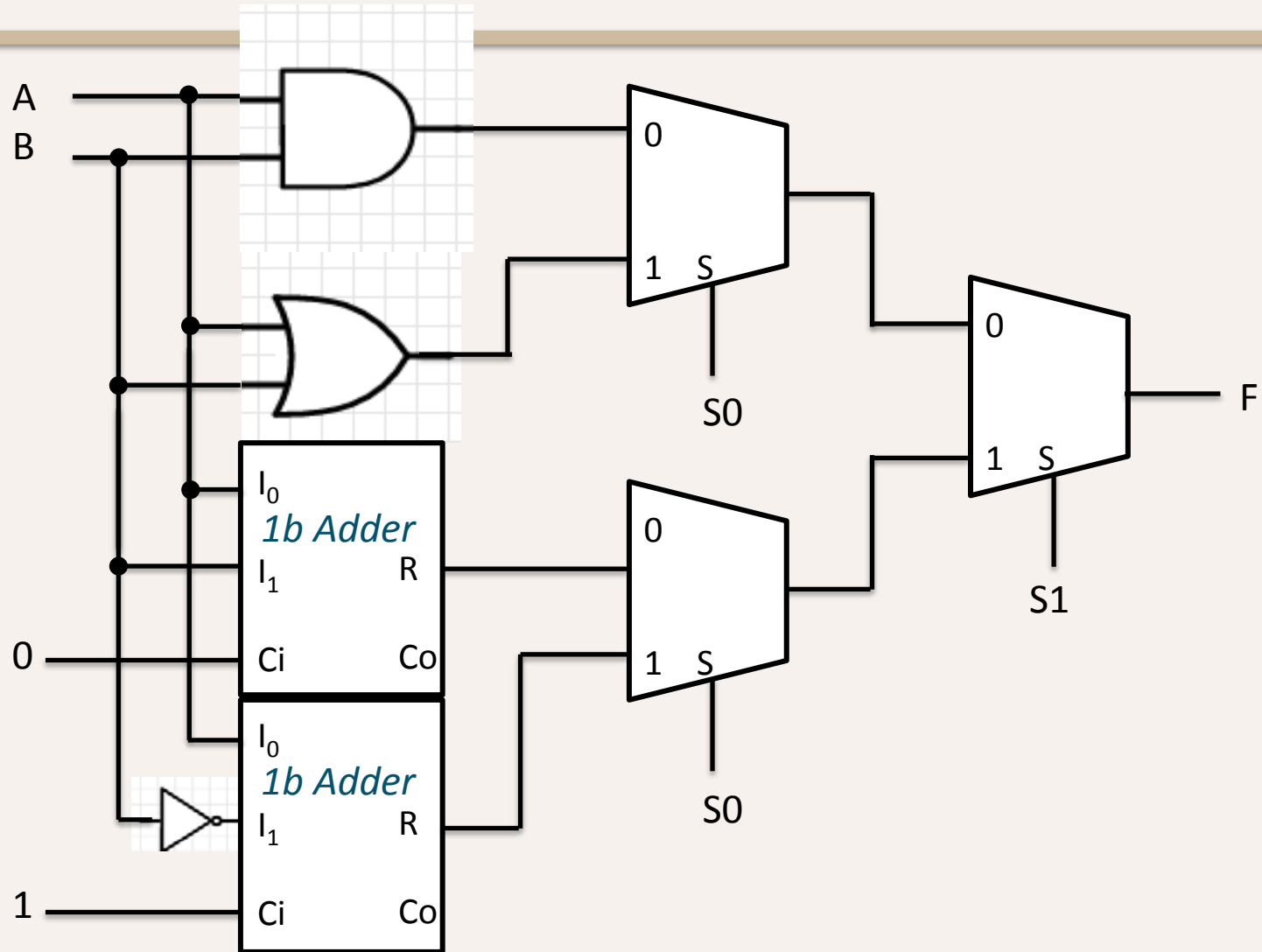


a0	a1	a2	a3	a4	a5
b0	b1	b2	b3	b4	b5
1	0	1	0	1	0
b0	a1	b2	a3	b4	a5

→ time

What Does This Circuit Do?

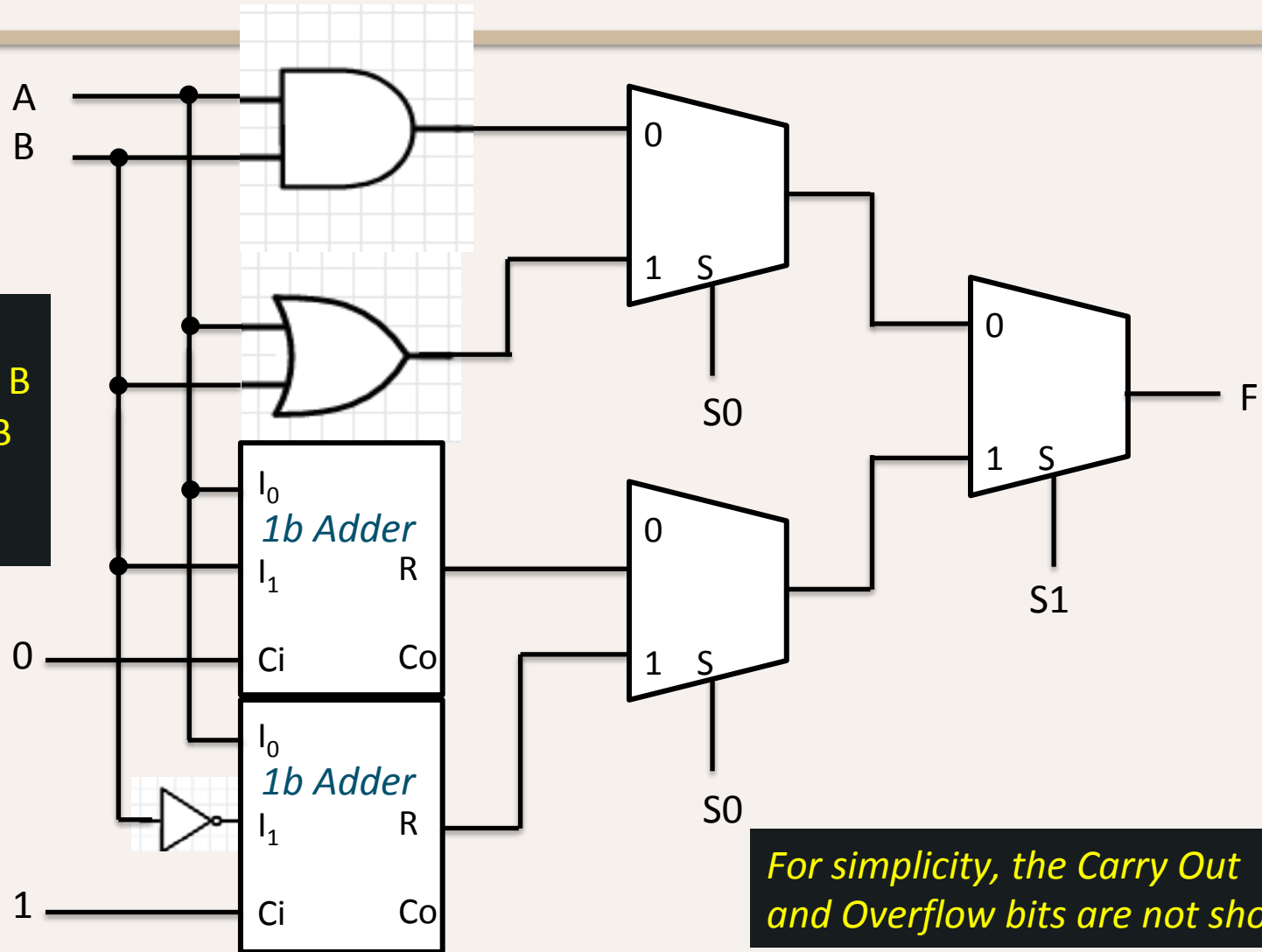
Class Ex.



What Does This Circuit Do?

Class Ex.

S1	S0	F
0	0	$A \& B$
0	1	$A \mid B$
1	0	$A + B$
1	1	$A - B$



For simplicity, the Carry Out and Overflow bits are not shown

logic.ly

File Edit View Tools Simulate Help

Input Controls

Toggle Switch

Push Button

Clock

High Constant

Low Constant

Output Controls

Light Bulb

Digit

Logic Gates

Buffer

NOT Gate

AND Gate

NAND Gate

OR Gate

NOR Gate

XOR Gate

XNOR Gate

Simulation of Combinatorial Logic

- Go to:
<https://logic.ly/demo/>

IN-CLASS DEMONSTRATION

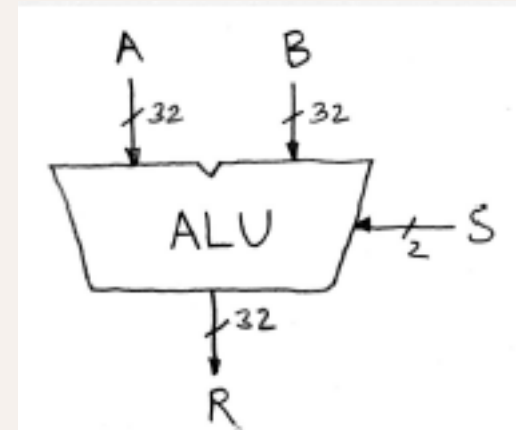
(Needed for Lab #7)

Matni, CS64, Wi18

16

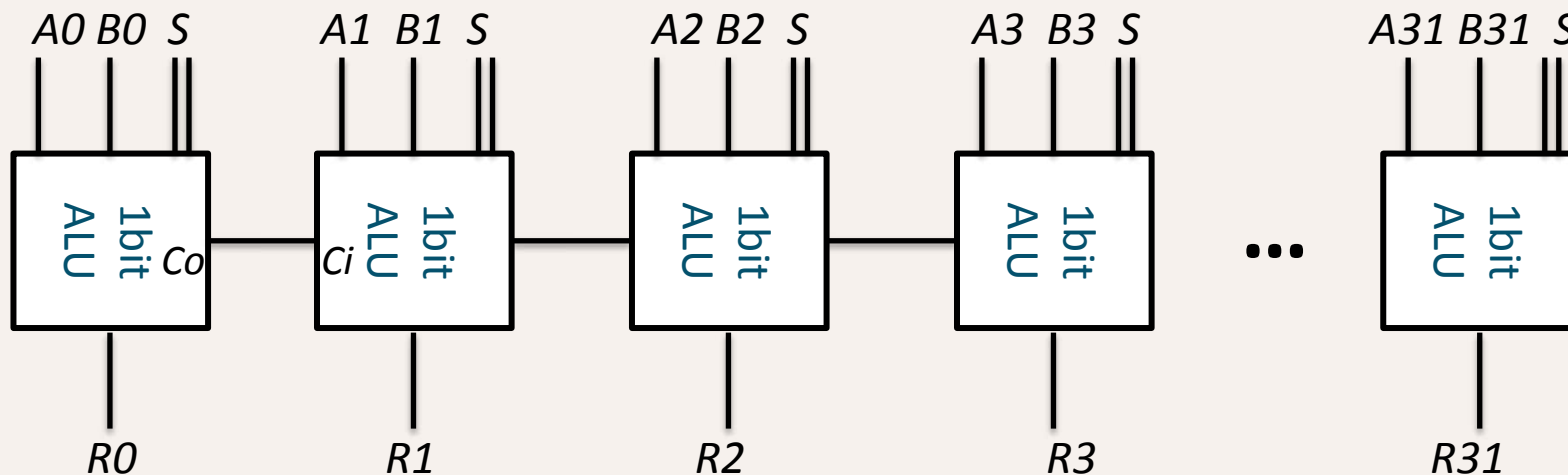
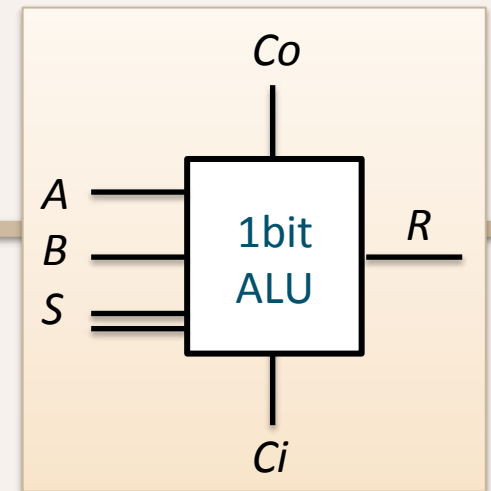
Arithmetic-Logic Unit (ALU)

- Recall: the ALU does all the computations necessary in a CPU
- The previous circuit was a simplified ALU:
 - When $S = 00$, $R = A + B$
 - When $S = 01$, $R = A - B$
 - When $S = 10$, $R = A \text{ AND } B$
 - When $S = 11$, $R = A \text{ OR } B$

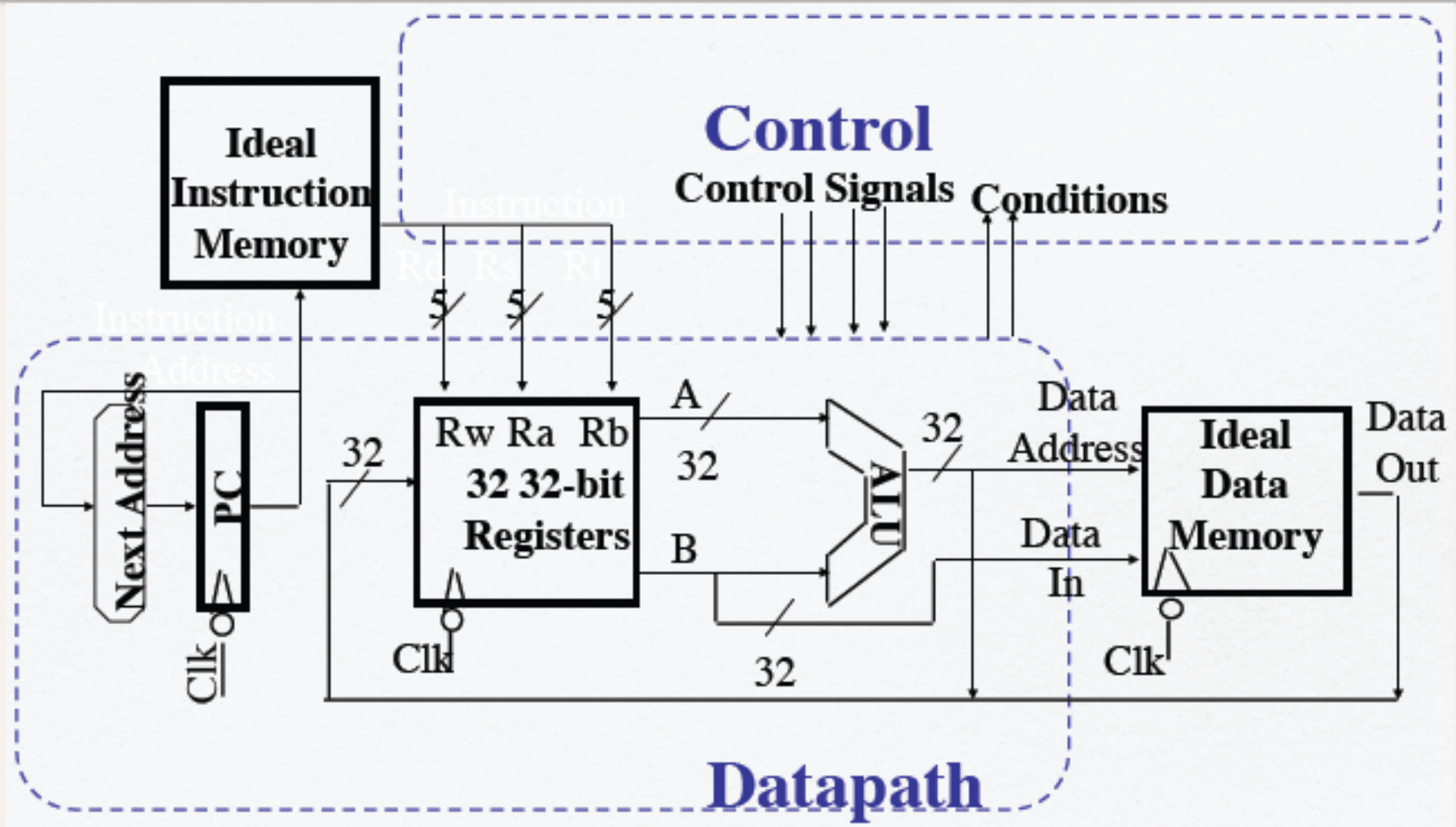


Simplified ALU

- We can string 1-bit ALUs together to make bigger-bit ALUs (e.g. 32b ALU)



Abstract Schematic of the MIPS CPU



Combinatorial vs. Sequential Logic

- The CPU schematic shows
both combinatorial and sequential logic blocks
- **Combinatorial Logic**
 - Combining multiple logic blocks
 - The output is a function **only** of the present inputs
 - There is no memory of past “states”
- **Sequential Logic**
 - Combining multiple logic blocks
 - The output is a function of **both** the present inputs and ***past*** inputs
 - There exists a memory of past “states”

Your To-Dos

- Lab 7 will be due on Friday, 3/9
 - We will take attendance for this lab on Thursday
- Lab 8 will be due on Friday, 3/16
 - I'll issue it this week, but you have more time for it
 - **Will contain info from next Monday's lecture**
- Reminder: there's a Lab 9 as well!
 - Online “quiz” on the Ethics lesson, due Fri. 3/16

</LECTURE>