

10/19/26

COMP 311: *Computer Organization!*

Lecture 16: Building the ALU



The University
of North Carolina
at Chapel Hill

<https://tinyurl.com/comp311-fa26>

Logistics Update



- WA3 due tomorrow
 - Will post solutions Wednesday morning
 - Quiz 4 Wednesday
 - Lab 2 due on 10/22 (Thursday)
-

Lab 3
ALU
😊

Quiz Topics



- **Boolean algebra and De Morgan's laws:** simplify expressions and move negations onto individual variables.
- **Gate minimization:** transform circuits using NAND/NOR gates and inverters; compare transistor counts.
- **Karnaugh maps:** find simplified SOP expressions, including wraparound groups and don't-care values. (X)
- **Multiplexer circuit design:** implement Boolean functions with muxes and build larger muxes from smaller ones.
- **Priority circuits:** determine which request wins and derive truth tables and output equations.
- **Half and full adders:** determine sum and carry outputs from inputs.
- **Ripple-carry addition and subtraction:** trace carries, use two's complement for subtraction, and identify signed overflow.
- **Basic ALU operation:** trace data and control signals through Boolean, arithmetic, and shift units, as covered Monday.

Plan for Class Today



- 3:35-4:00: K-Map Review
- 4:00-4:15: Finish Adder/Subtractors
- 4:15-4:20: Break
- 4:20-4:50: Building the ALU

Guiding Question 😊

How do we implement
our math instructions
with hardware?



K-Map Review

Don't Cares in K-Maps



AB \ CD	CD				
		00	01	11	10
00	0	0	0	0	0
01	0	1	X	0	
11	0	1	X	0	
10	0	0	0	0	0

4
choices

Q1: Find the Minimized SOP Equation



		<u>CD</u>			
<u>AB</u>		00	01	11	10
	00	1	0	X	1
	01	0	1	X	1
	11	1	1	X	X
	10	1	1	X	X

Handwritten annotations on the Karnaugh map:

- Blue circles around cells (00,00), (00,10), (01,00), and (10,00).
- Red circle around cells (01,01), (11,01), (11,11), and (10,11).
- Green circle around cells (00,00), (00,10), (01,01), (01,11), (11,01), (11,11), (10,11), and (10,10).
- Purple circle around cells (11,11), (10,11), (11,10), and (10,10).
- Red and blue markings on 'X' cells with lines indicating groupings.

Q2: Find the Minimized SOP Equation



AB \ CD	CD			
	00	01	11	10
00	1	1	X = 1	1
01	1	0	X = 0	1
11	1	1	X = 1	X = 1
10	1	0	X = 0	X = 1

Q1: Find the Minimized SOP Equation



AB \ CD	CD			
	00	01	11	10
00	1	0	X	1
01	0	1	X	1
11	1	1	X	X
10	1	1	X	X

$$Y = A + C + \bar{B}\bar{D} + BD$$

Q2: Find the Minimized SOP Equation



AB \ CD	CD			
	00	01	11	10
00	1	1	X	1
01	1	0	X	1
11	1	1	X	X
10	1	0	X	X

$$Y = \bar{A}\bar{B} + AB + \bar{D}$$

Why would I not want to circle this blue group?



AB \ CD	CD			
	00	01	11	10
00	1	1	X	1
01	1	0	X	1
11	1	1	X	X
10	1	0	X	X

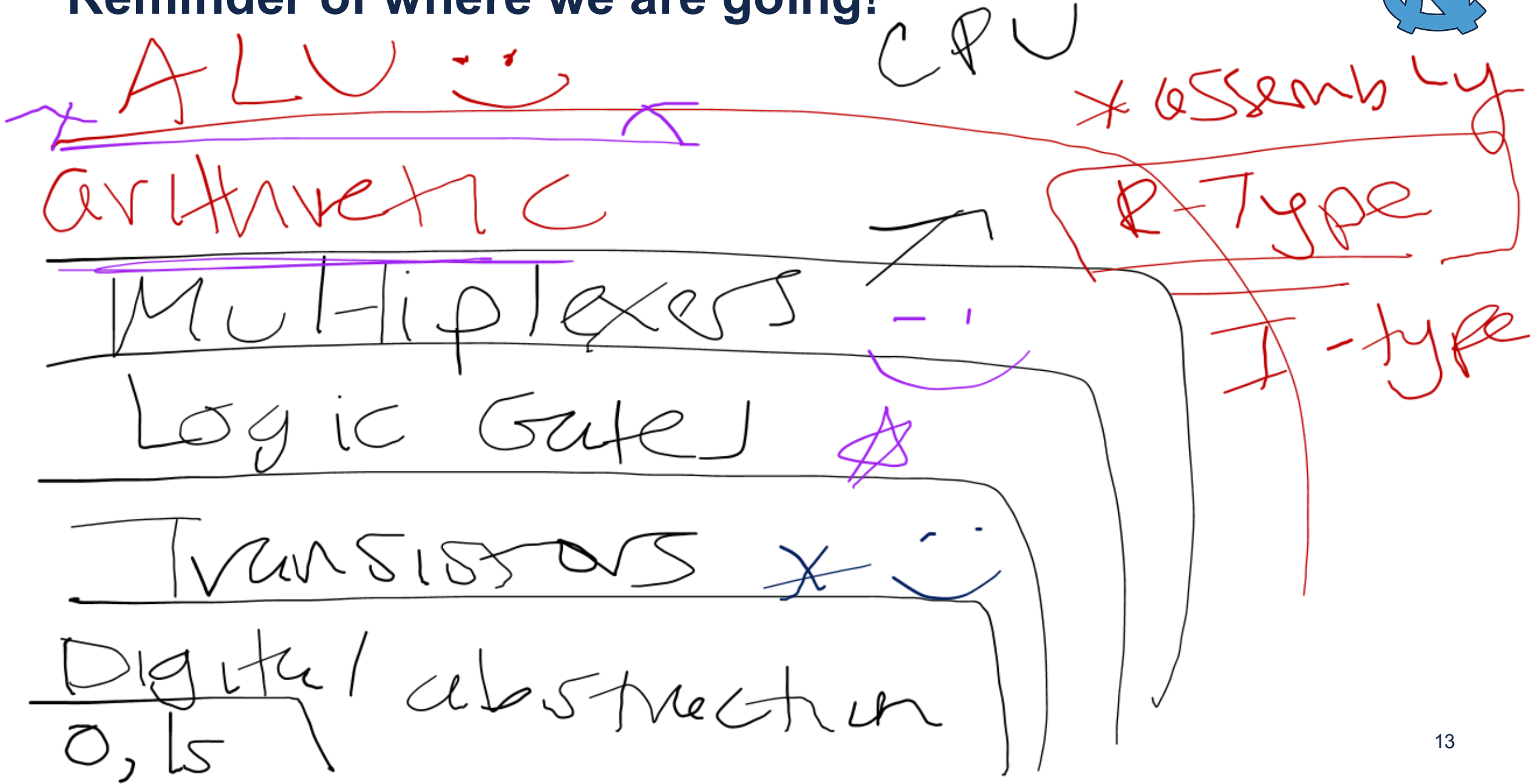
Why would I not want to circle this blue group?



AB \ CD	CD			
	00	01	11	10
00	1	1	X	1
01	1	0	X	1
11	1	1	X	X
10	1	0	X	X

We want our implementation to use the fewest gates possible. Adding this additional group would increase the number of gates needed.

Reminder of where we are going!



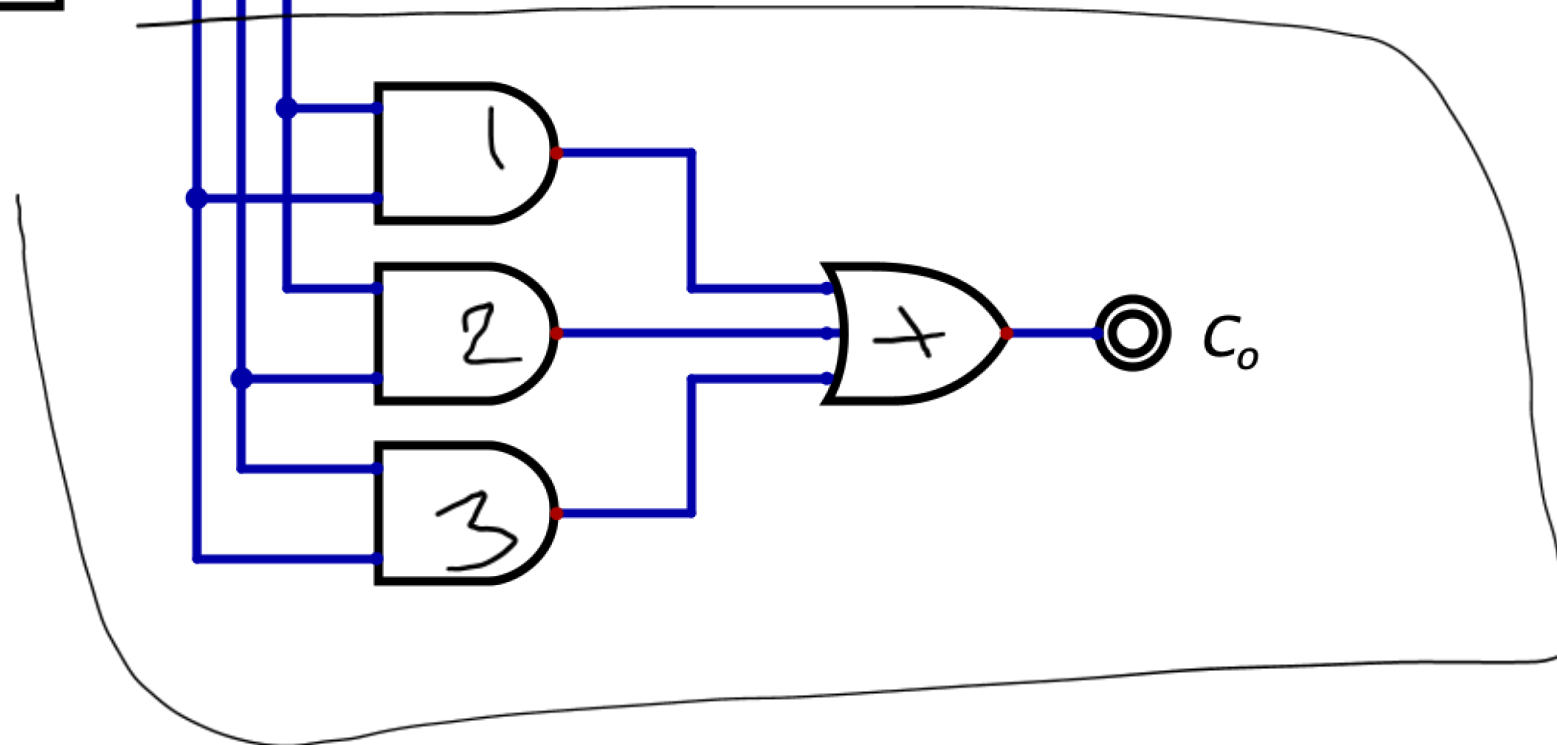
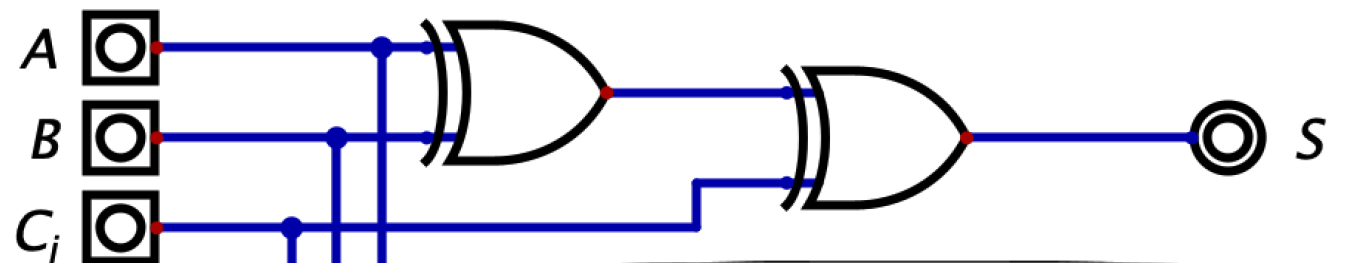


Adder Circuits!

Full Adder

$$S = A \oplus B \oplus C_i$$

$$C_o = AC_i + AB + BC_i$$

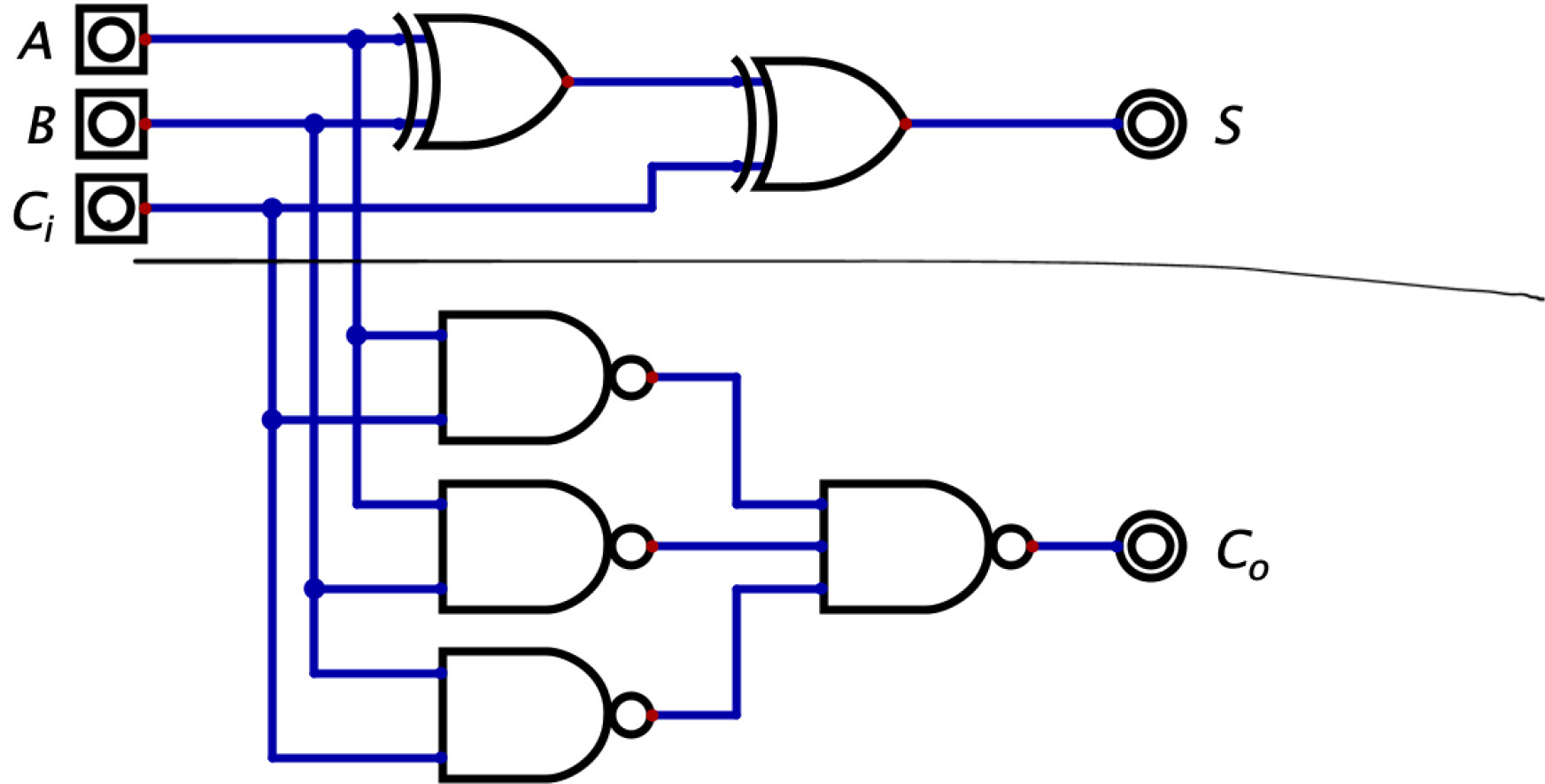


Full Adder: transistor minimization



$$S = A \oplus B \oplus C_i$$

$$C_o = AC_i + AB + BC_i$$

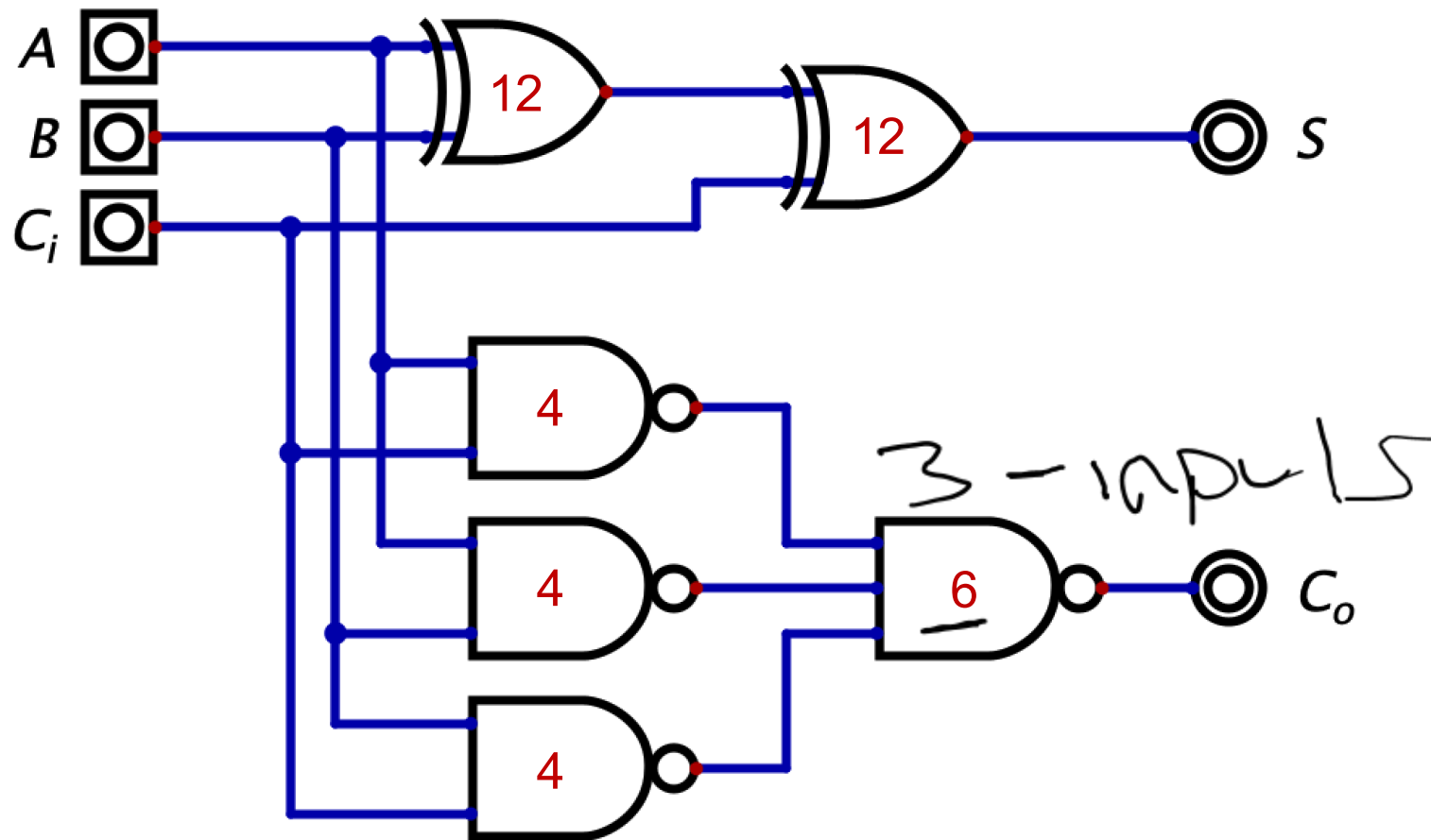




Full Adder: transistor minimization

$$S = A \oplus B \oplus C_i$$

$$C_o = AC_i + AB + BC_i$$



Transistor count = 42

Full Adder: Alternate implementation



A	B	C _i	C _o	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	0	1	0
1	1	1	1	1

$$S = A \oplus B \oplus C$$

$$C_o =$$

Full Adder: Alternate Implementation



A	B	C _i	C _o	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	0	1	0
1	1	1	1	1

$$S = A \oplus B \oplus C$$

1 2 3 4

$$C_o = \bar{A}BC_i + A\bar{B}C_i + AB\bar{C}_i + ABC_i$$

$$C_o = \bar{A}BC_i + A\bar{B}C_i + AB(\bar{C}_i + C_i)$$

$$C_o = \bar{A}BC_i + A\bar{B}C_i + AB$$

$$C_o = C_i(\bar{A}B + A\bar{B}) + AB$$

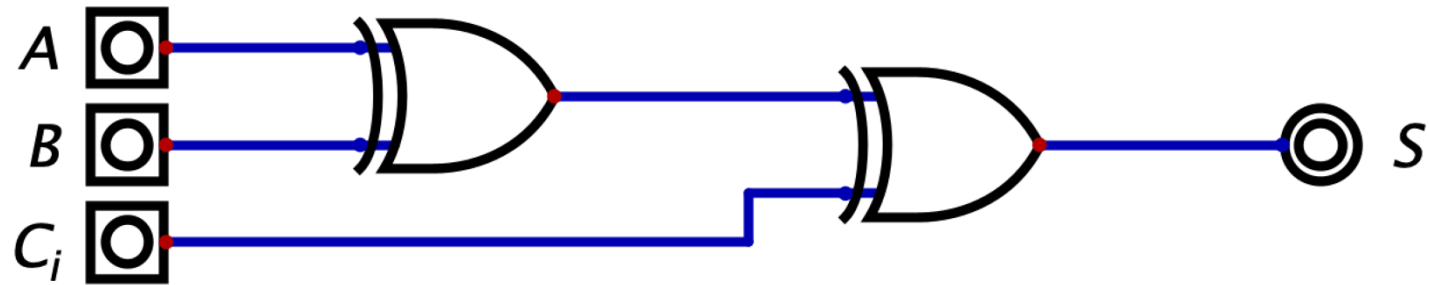
$$C_o = C_i(A \oplus B) + AB$$

Full Adder: Alternate Implementation



$$S = A \oplus B \oplus C_i$$

$$C_o = C_i(A \oplus B) + AB$$



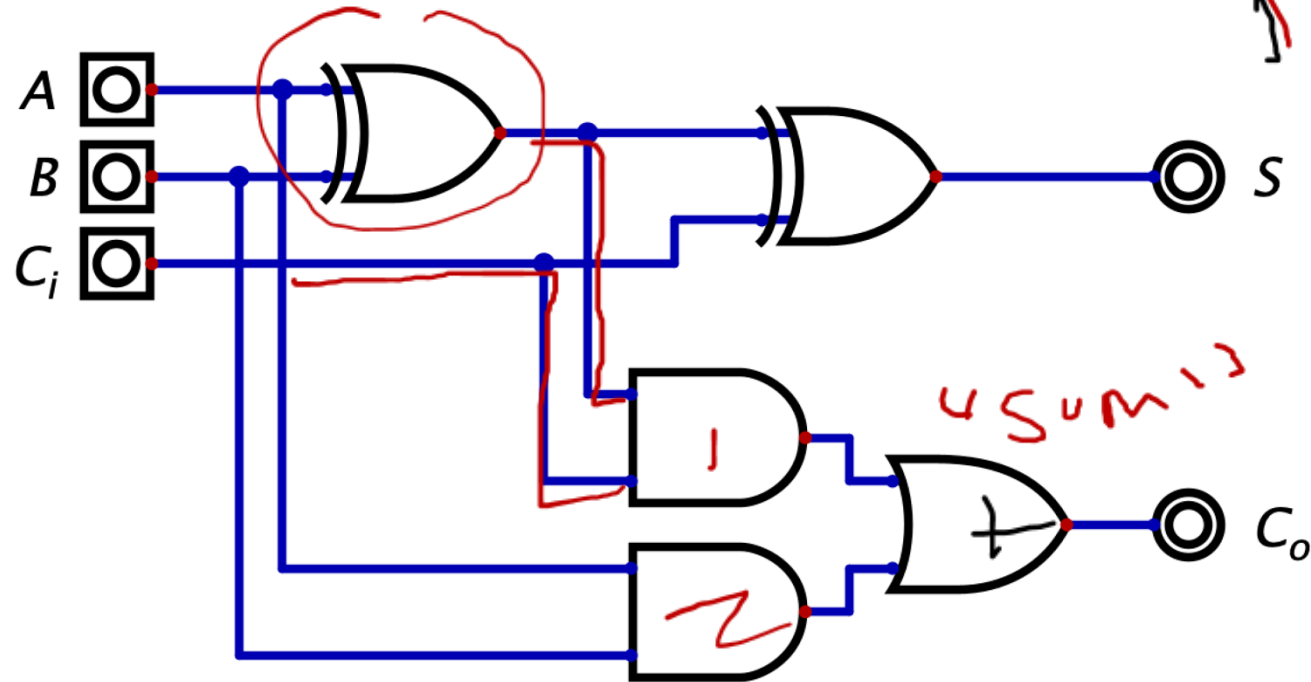
 C_o

Full Adder: Alternate Implementation



$$S = A \oplus B \oplus C_i$$

$$C_o = C_i \underbrace{(A \oplus B)}_{\uparrow} + \underbrace{AB}_Z$$

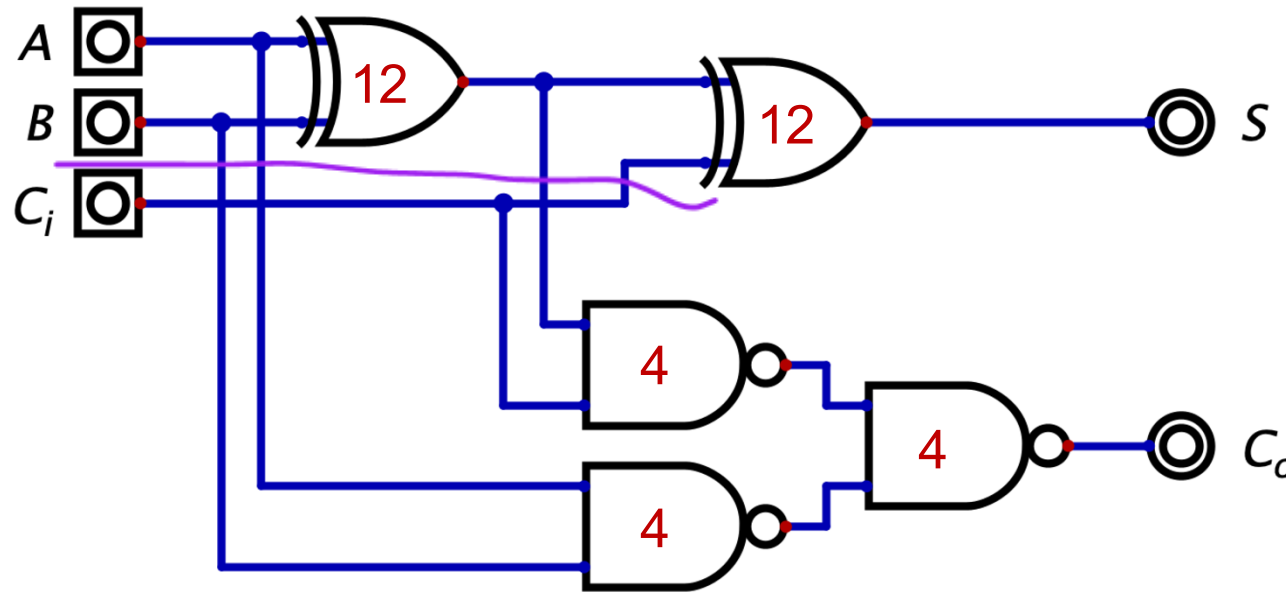


Full Adder: Reducing Transistor Count



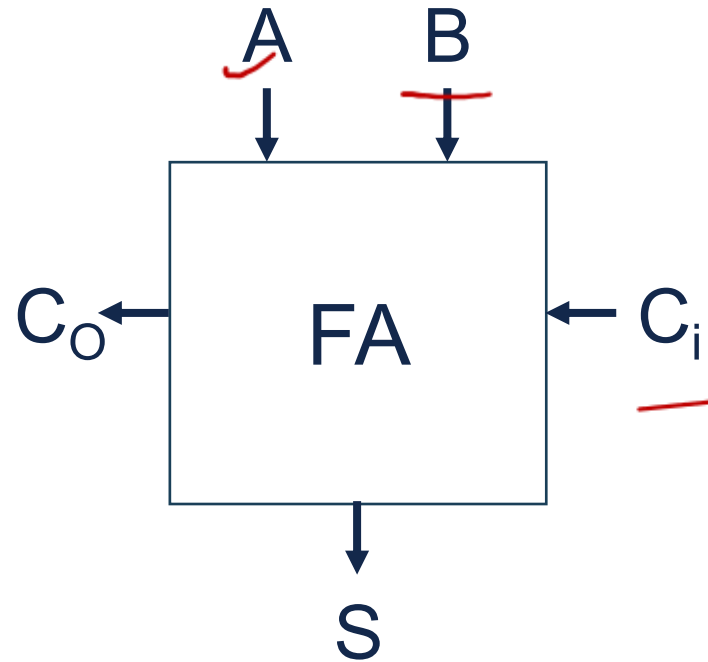
$$S = A \oplus B \oplus C_i$$

$$C_o = C_i(A \oplus B) + AB$$



Transistor Count: 36

Full Adder Abstraction



4-bit Adder



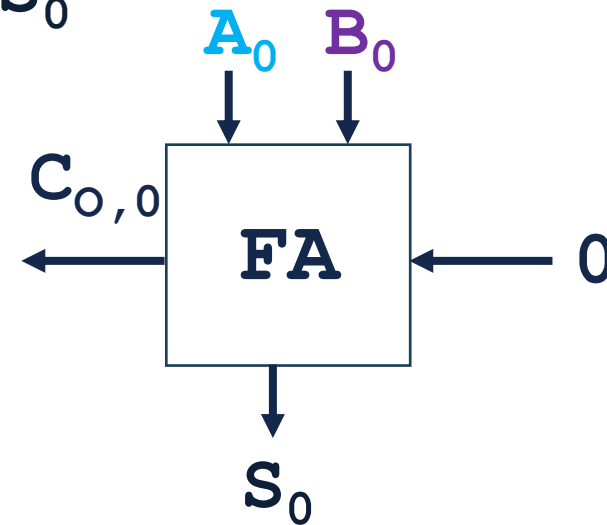
$$\begin{array}{cccc} C_{0,3} & C_{0,2} & C_{0,1} & C_{0,0} \\ & A_3 & A_2 & A_1 & A_0 \\ + & B_3 & B_2 & B_1 & B_0 \\ \hline & S_3 & S_2 & S_1 & S_0 \end{array}$$

4-bit Adder



- How would you use 4 full-adders to add together A and B?

$$\begin{array}{cccc} C_{o,3} & C_{o,2} & C_{o,1} & C_{o,0} \\ + & A_3 & A_2 & A_1 & A_0 \\ & B_3 & B_2 & B_1 & B_0 \\ \hline S_3 & S_2 & S_1 & S_0 \end{array}$$

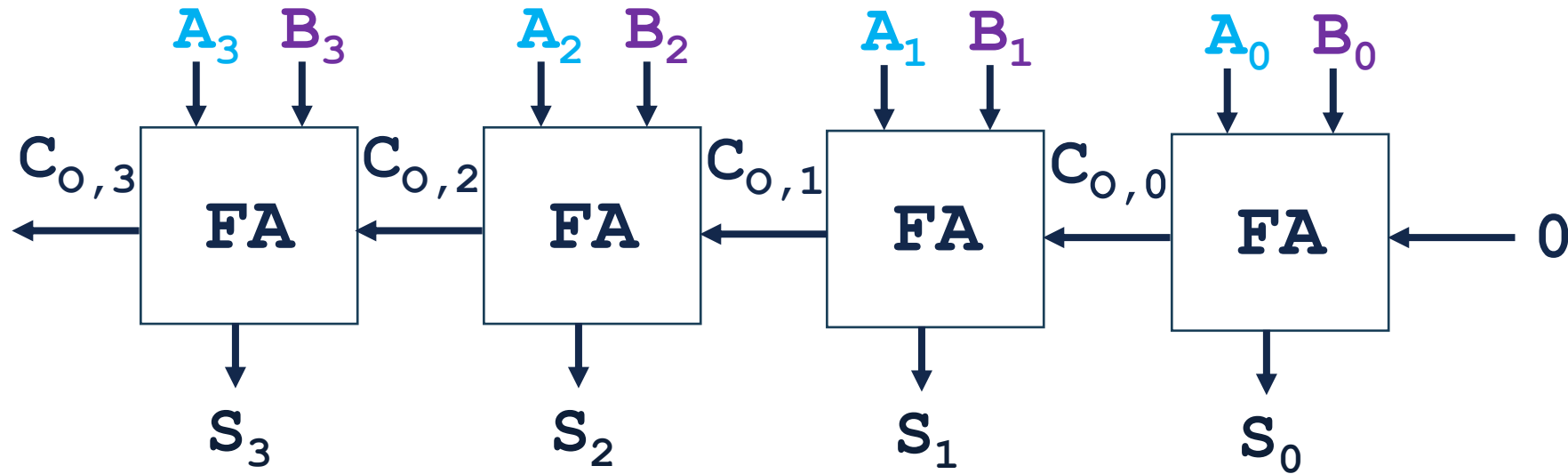


4-bit Adder

(ripple carry adder!)



$$\begin{array}{r} C_{0,3} \quad C_{0,2} \quad C_{0,1} \quad C_{0,0} \\ + \quad A_3 \quad A_2 \quad A_1 \quad A_0 \\ \quad B_3 \quad B_2 \quad B_1 \quad B_0 \\ \hline S_3 \quad S_2 \quad S_1 \quad S_0 \end{array}$$





Adder-Subtractor Circuit

Recall: Two's Complement



Ex: Represent -15 in two's complement with 7 bits

- To represent a negative number, take the positive representation of the number, flip the bits, and add 1

Unsigned: 0b 000 1111

Flip bits: 0b 111 0000

Add one: 0b 111 0001

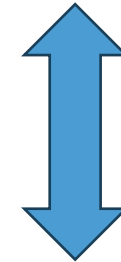
Recall Two's Complement Subtraction



$$A - B == A + (-B)$$

$$\begin{array}{r} -10 \\ - \quad 4 \\ \hline -14 \end{array}$$

$$\begin{array}{r} 10110_2 \\ - \quad 00100_2 \\ \hline \end{array}$$



Equivalent
Operations

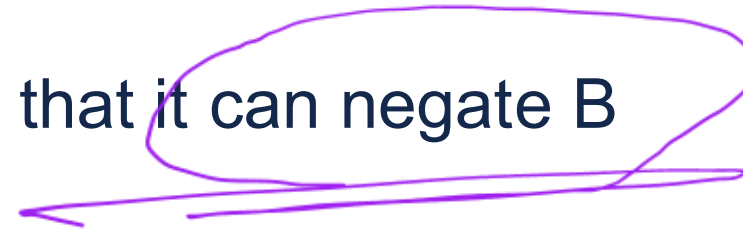
$$\begin{array}{r} -10 \\ + \quad -4 \\ \hline -14 \end{array}$$

$$\begin{array}{r} 10110_2 \\ + \quad 11100_2 \\ \hline 10010_2 \end{array}$$

Subtraction



- Instead of building a separate circuit for subtraction, we will modify the ripple carry adder such that it can perform addition and subtraction
- When we want to perform $A - B$
 - We will actually perform $A + (-B)$
 - We need to modify our circuit such that it can negate B



Subtraction



- Let's say that we want to perform the operation $A - B$ where $A = 5$ and $B = 1$.
- We will modify the circuit such that it can flip each bit of B and add 1.

$$\begin{array}{r} 0101_2 \\ - 0001_2 \\ \hline \end{array}$$

$$\begin{array}{r} 0101_2 \\ + 1110_2 \\ + 1_2 \\ \hline 0100_2 \end{array}$$

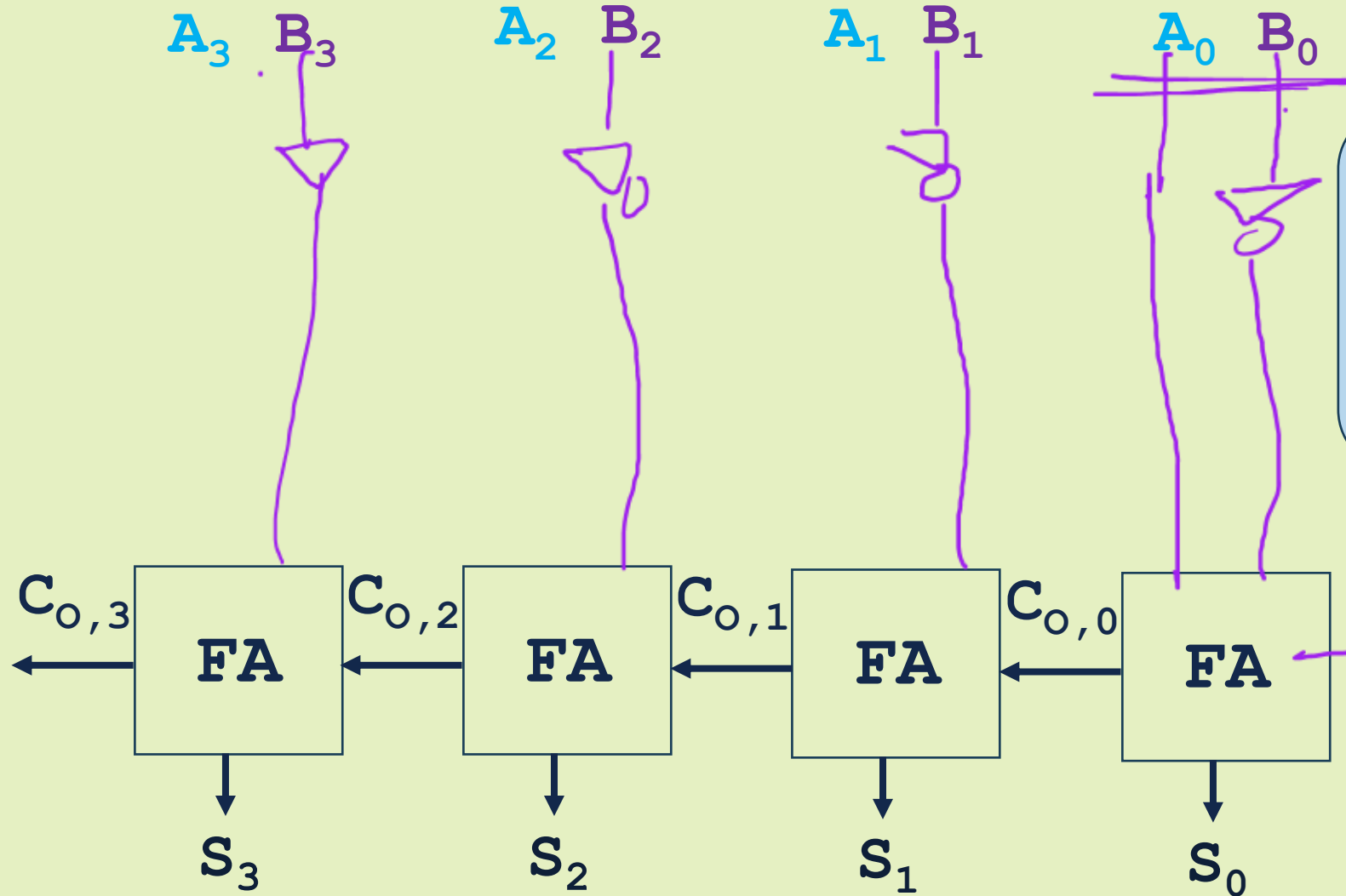
Keep A the same

Flip the bits of B

Add 1

Subtractor

$$A = B$$



How would you wire up this circuit to have it correctly perform subtraction?

Subtractor



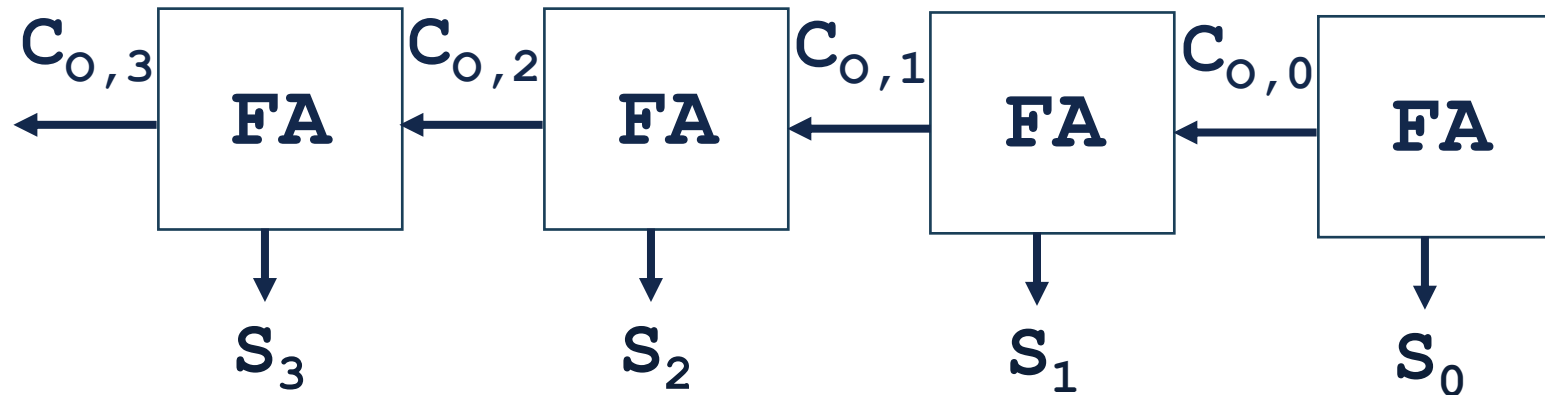
A_3 B_3

A_2 B_2

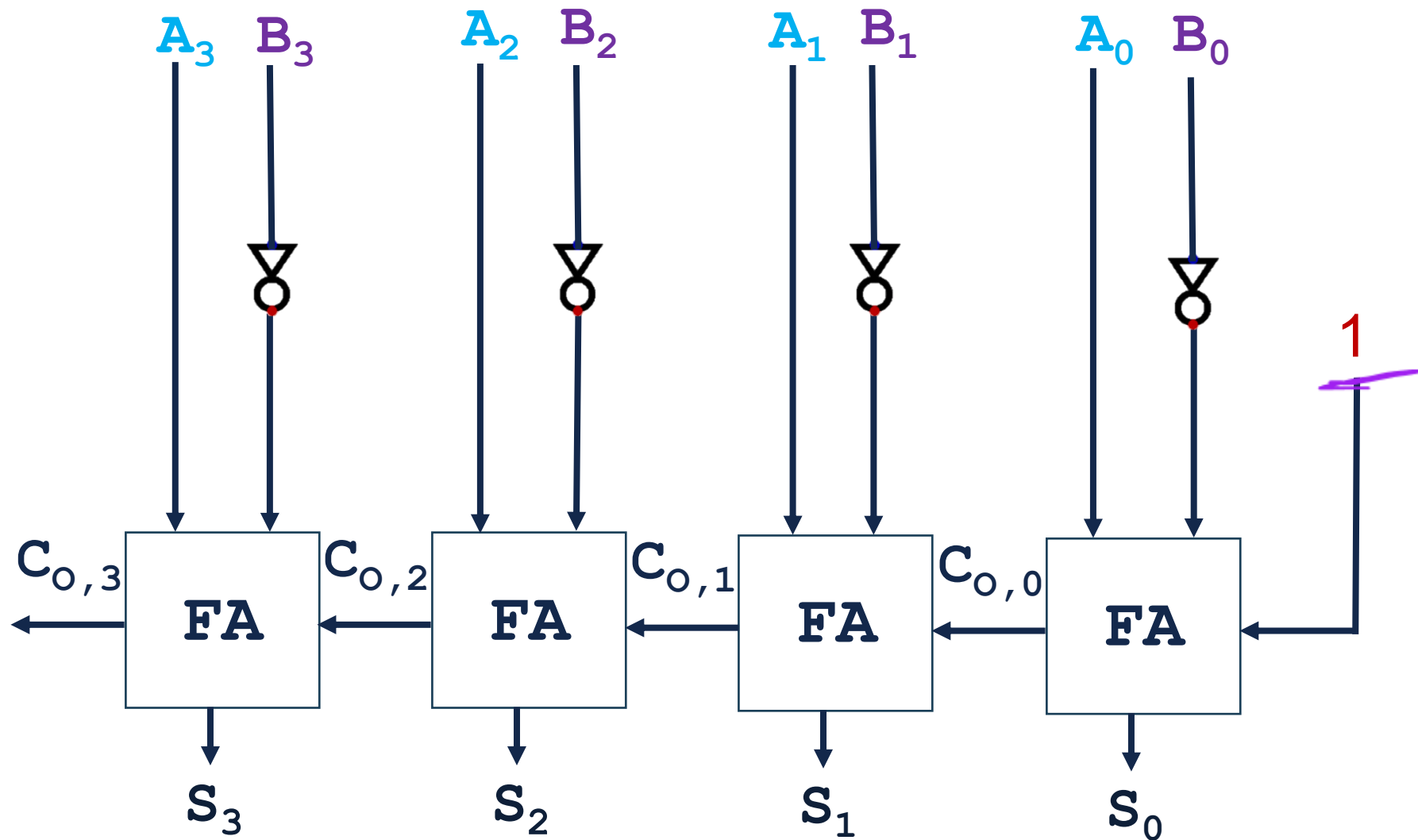
A_1 B_1

A_0 B_0

How would you wire up this circuit to have it correctly perform subtraction?



Subtractor



Adder/subtractor

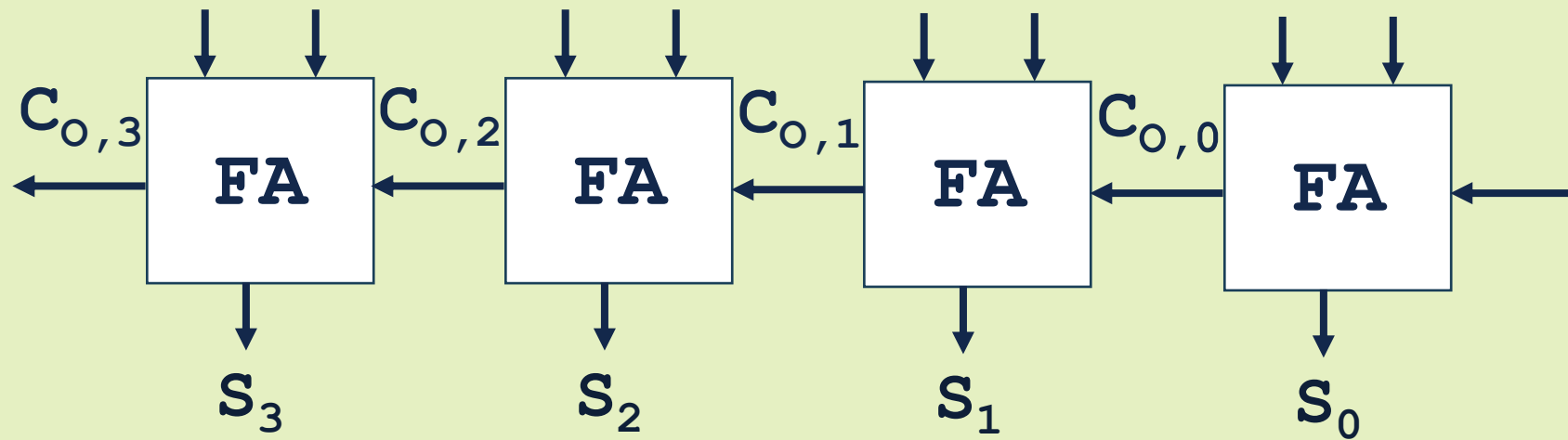


A_3 B_3

A_2 B_2

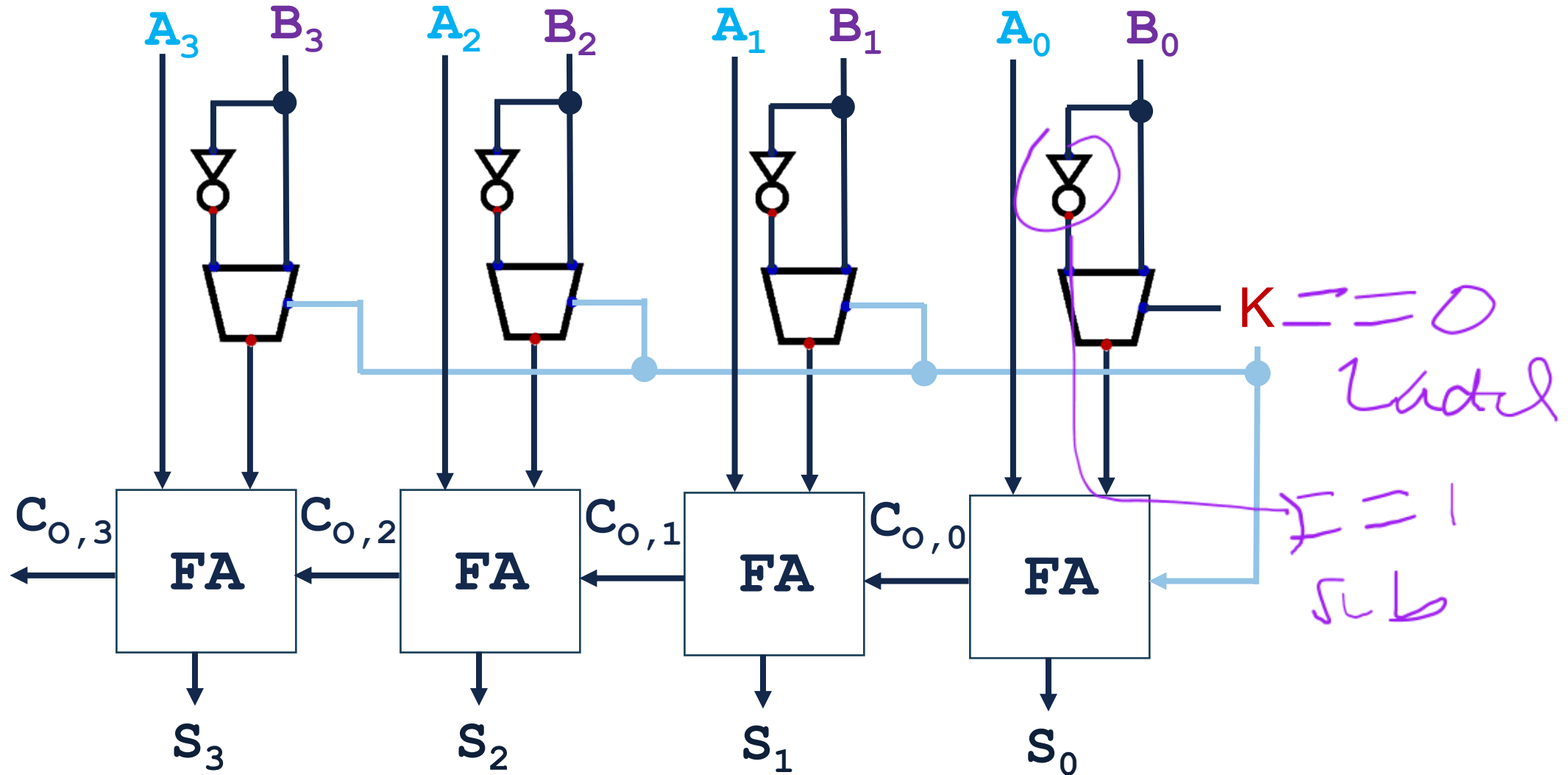
A_1 B_1

A_0 B_0



K If $K == 0$, $A + B$
If $K == 1$, $A - B$

Adder/Subtractor Implementation 1



XOR



$$N \oplus 1 = \bar{N}$$

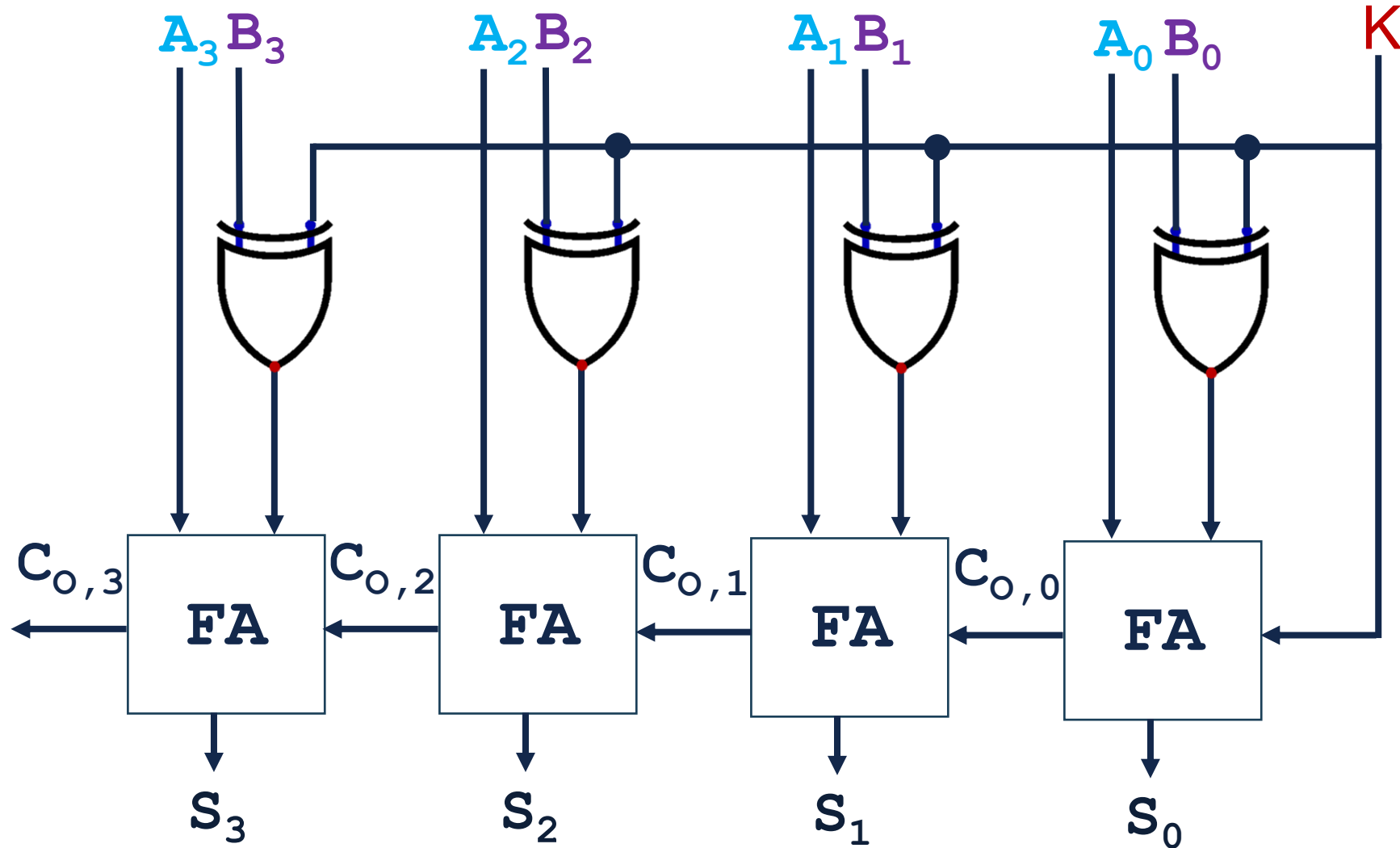
$$N \oplus 0 = N$$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

If we XOR B_i with K

- When $K = 0$, $B_i \oplus K = B_i$
- When $K = 1$, $B_i \oplus K = \bar{B}_i$

Adder/Subtractor Implementation 2

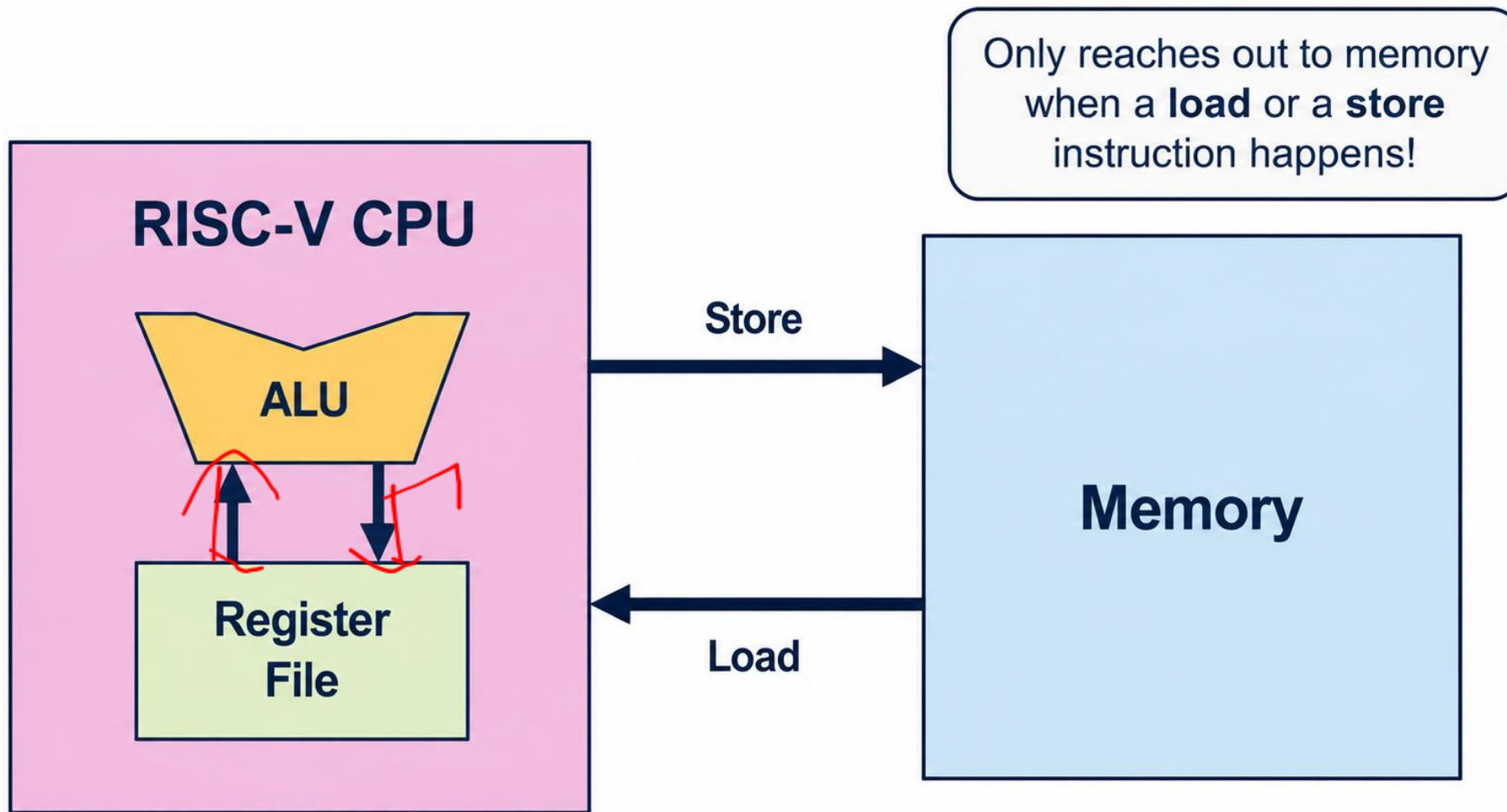




Arithmetic Logic Unit 😊



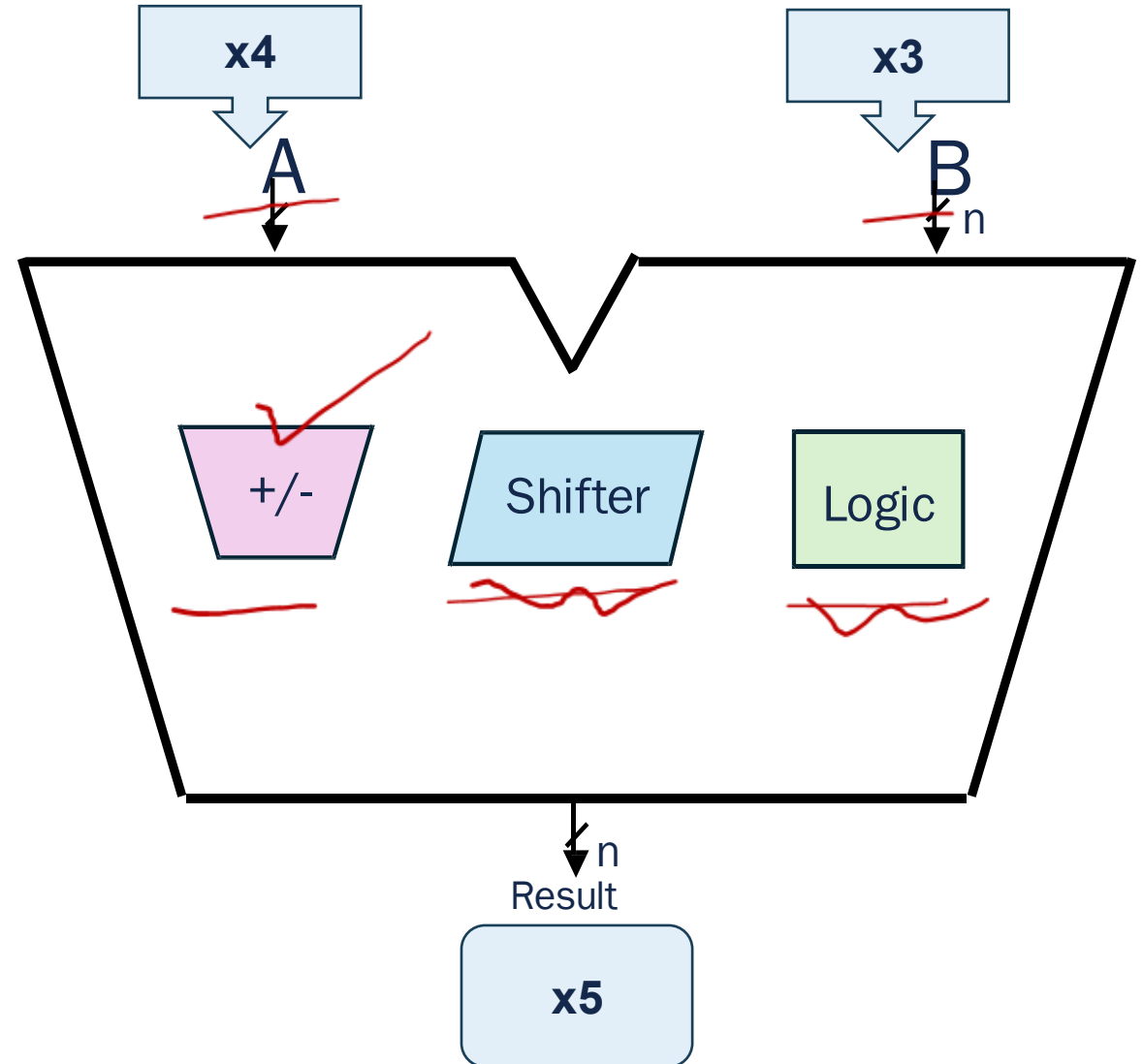
Does this slide look familiar?? 😊

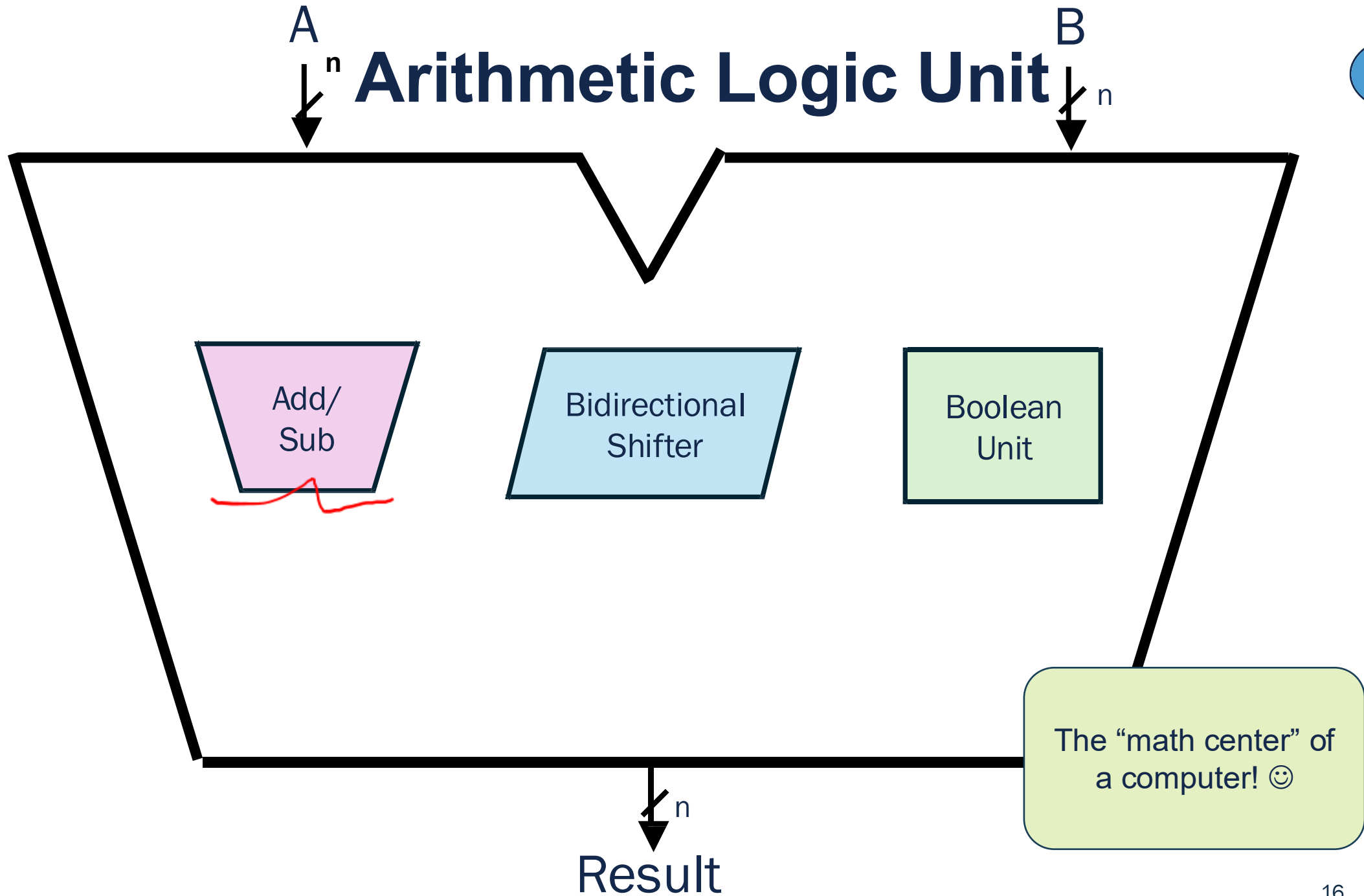


R-type Data Instructions Use the ALU

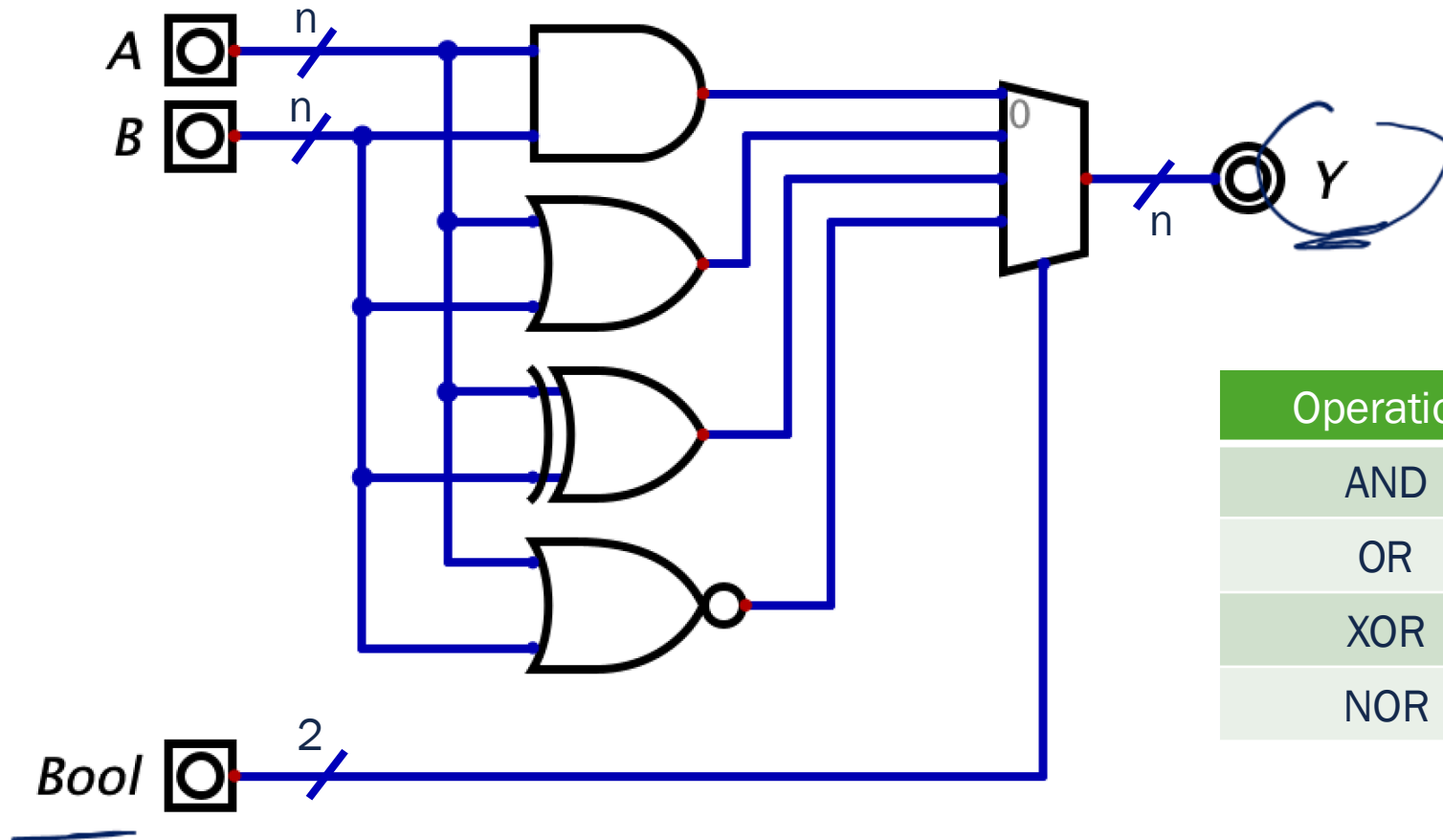


- add x5, x4, x3



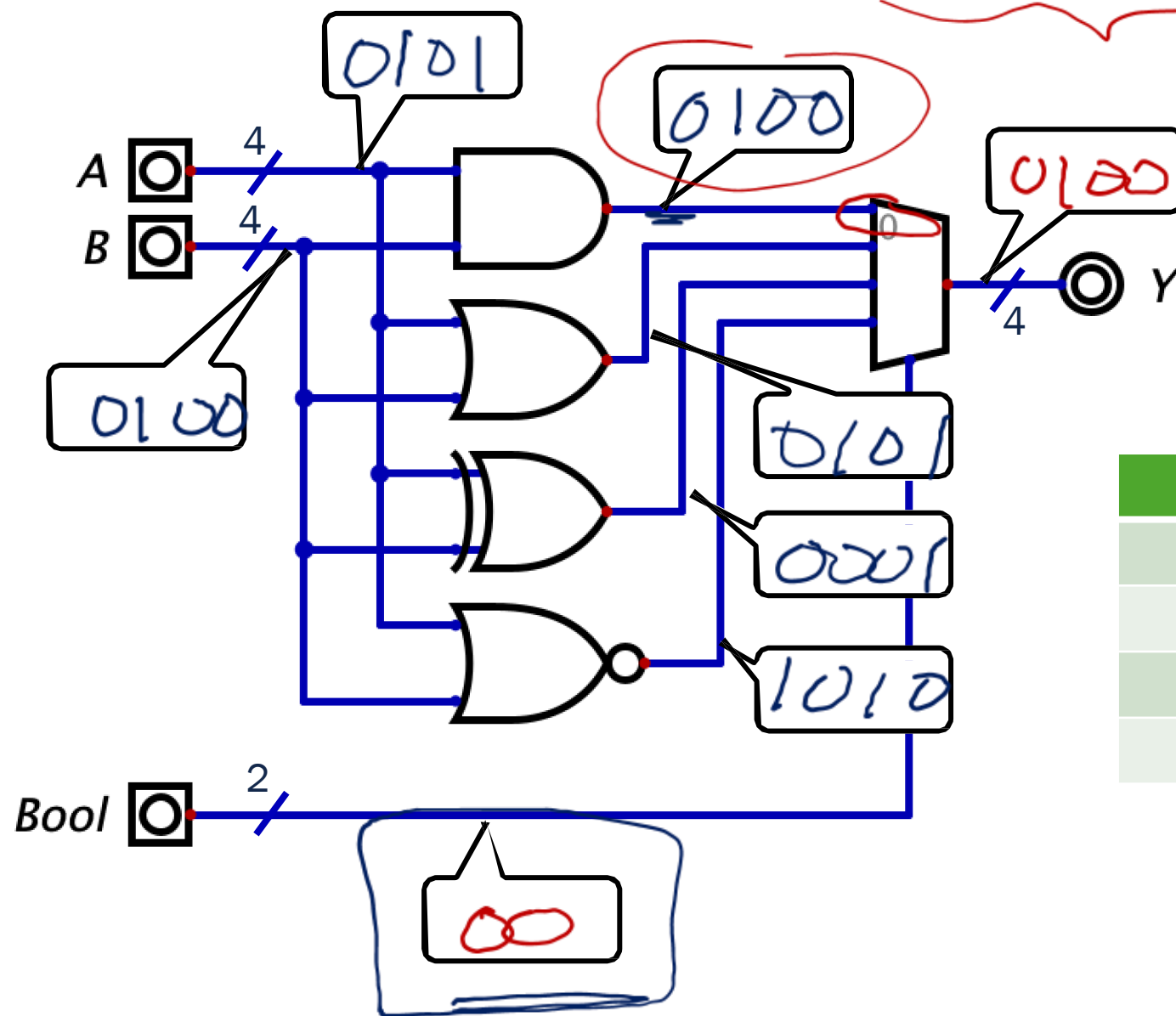


An Example Boolean Unit



Operation	Expression	Bool
AND	$A \times B$	0b00
OR	$A + B$	0b01
XOR	$A + B$	0b10
NOR	$A + B$	0b11

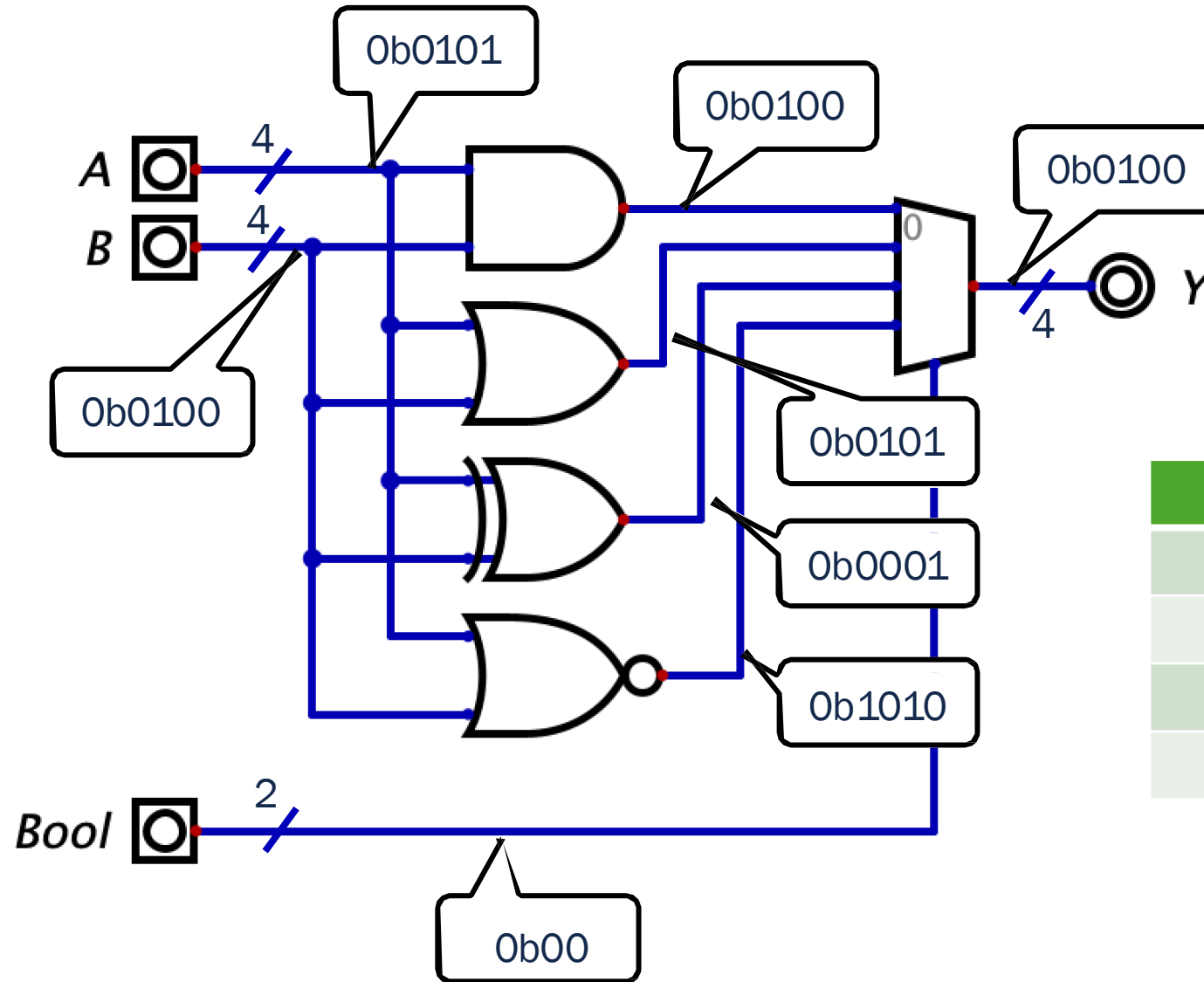
Let's say $n = 4$. I want to perform $Y = A \text{ AND } B$ where A is 5 and B is 4.



Instruction

Operation	Expression	Bool
AND	$A \times B$	0b00
OR	$A + B$	0b01
XOR	$A \oplus B$	0b10
NOR	$A + B$	0b11

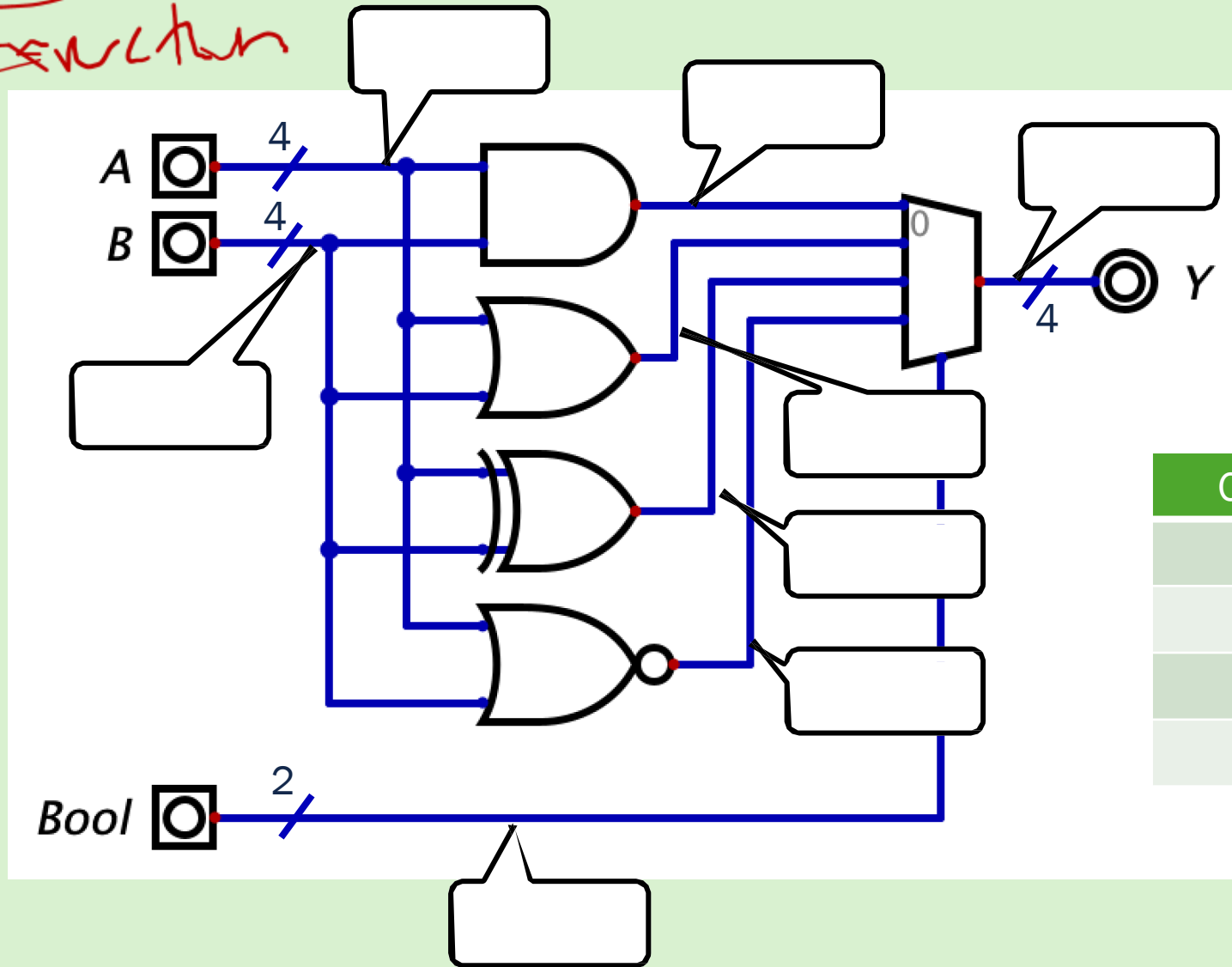
Let's say $n = 4$. I want to perform $Y = A \text{ AND } B$ where A is 5 and B is 4.



Operation	Expression	Bool
AND	$A \times B$	0b00
OR	$A + B$	0b01
XOR	$A \oplus B$	0b10
NOR	$A + B$	0b11

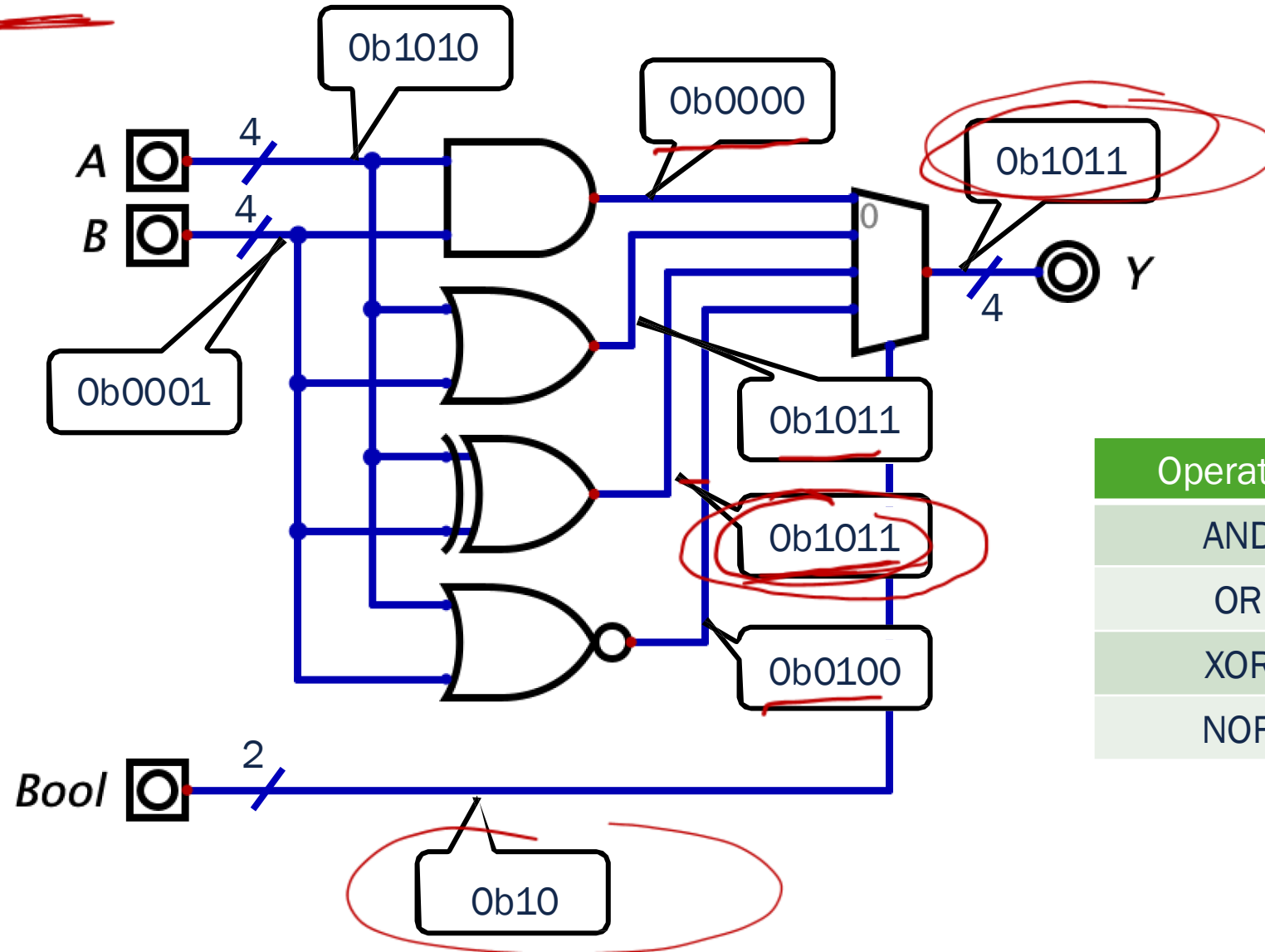
What are the values on each signal when we want to perform $Y = A$
~~XOR~~ B where A is 0b1010 and B is 0b0001?

425vctm



Operation	Expression	Bool
AND	$A \times B$	0b00
OR	$A + B$	0b01
XOR	$A \oplus B$	0b10
NOR	$A + B$	0b11

What are the values on each signal when we want to perform $Y = A \text{ XOR } B$ where A is $0b1010$ and B is $0b0001$?

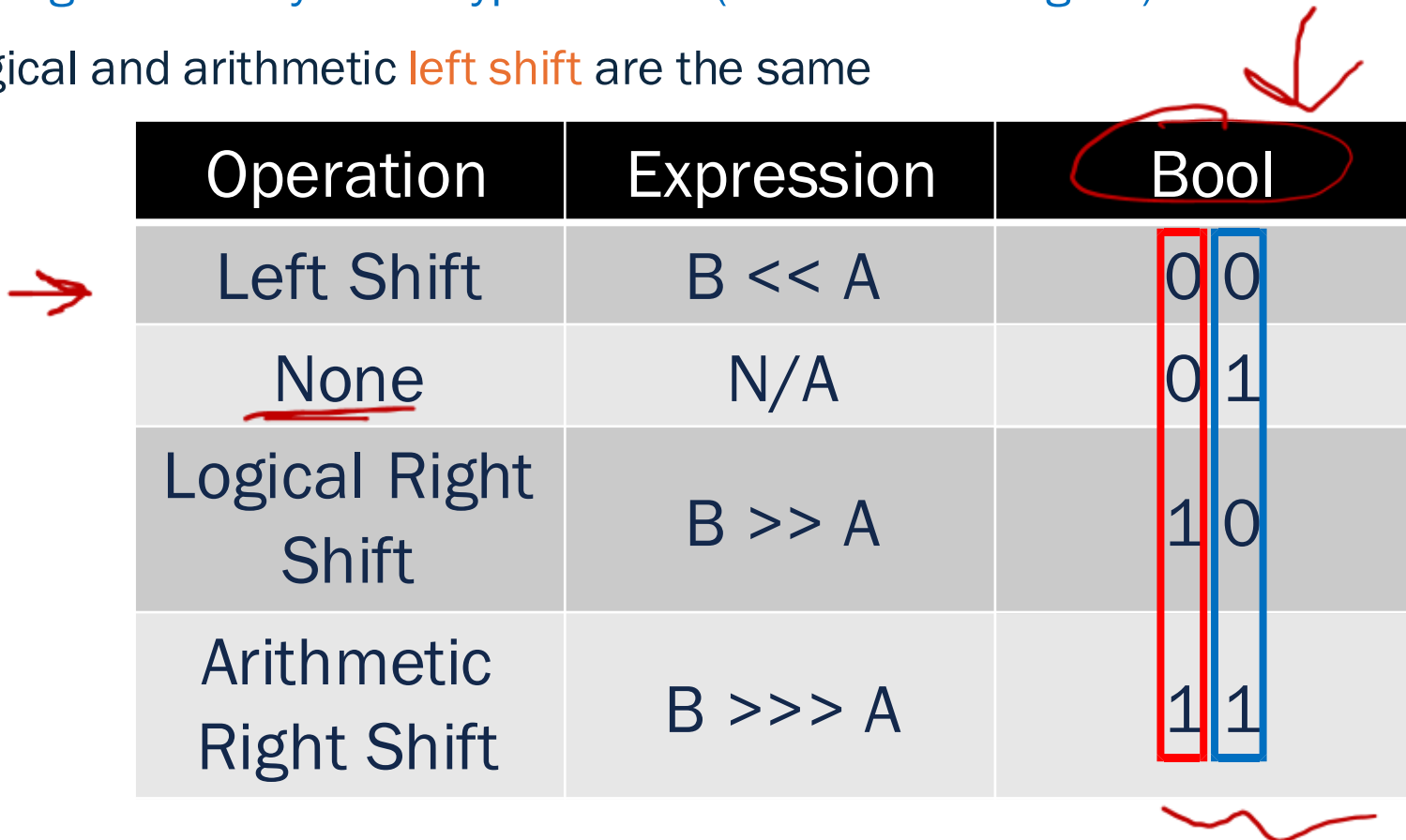


Operation	Expression	Bool
AND	$A \times B$	0b00
OR	$A + B$	0b01
XOR	$A \oplus B$	0b10
NOR	$\overline{A + B}$	0b11



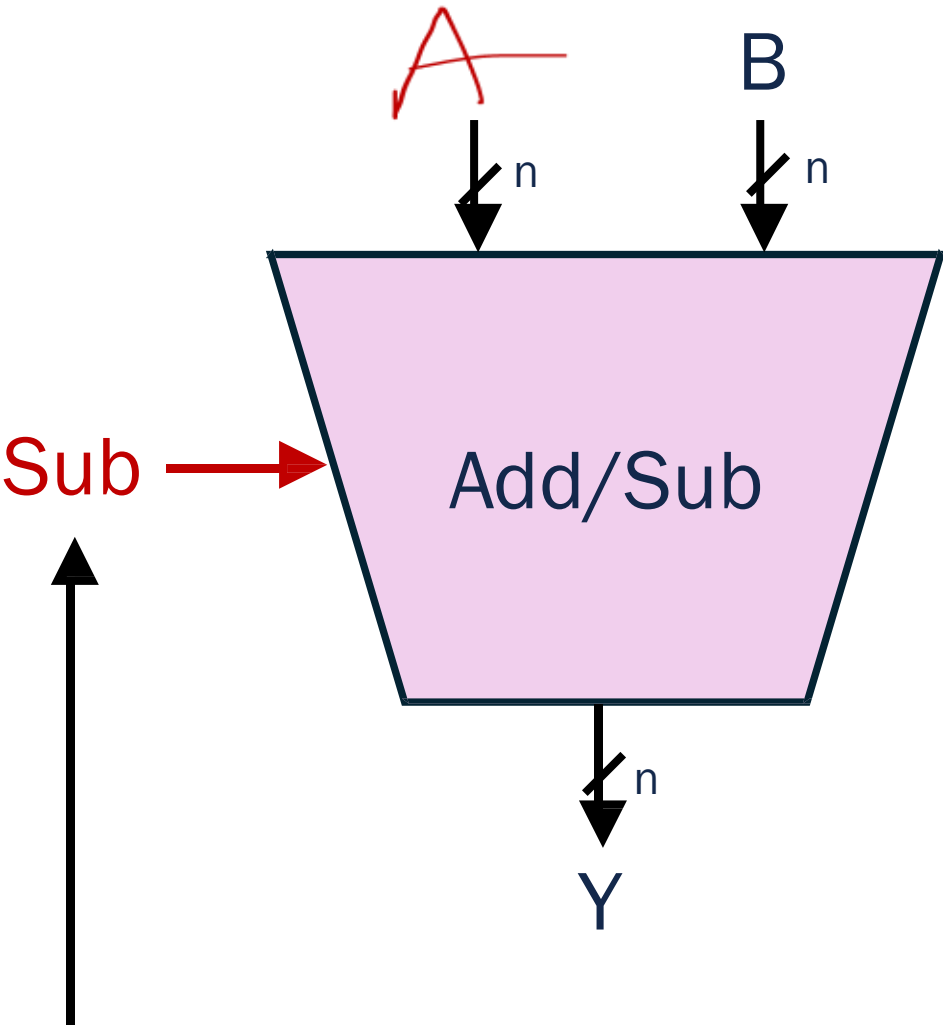
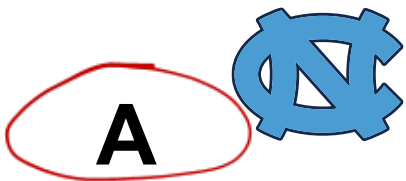
Bidirectional Shifter

- The left bit tells you the direction of the shift (left or right)
- The right bit tells you the type of shift (arithmetic or logical)
- Logical and arithmetic **left shift** are the same



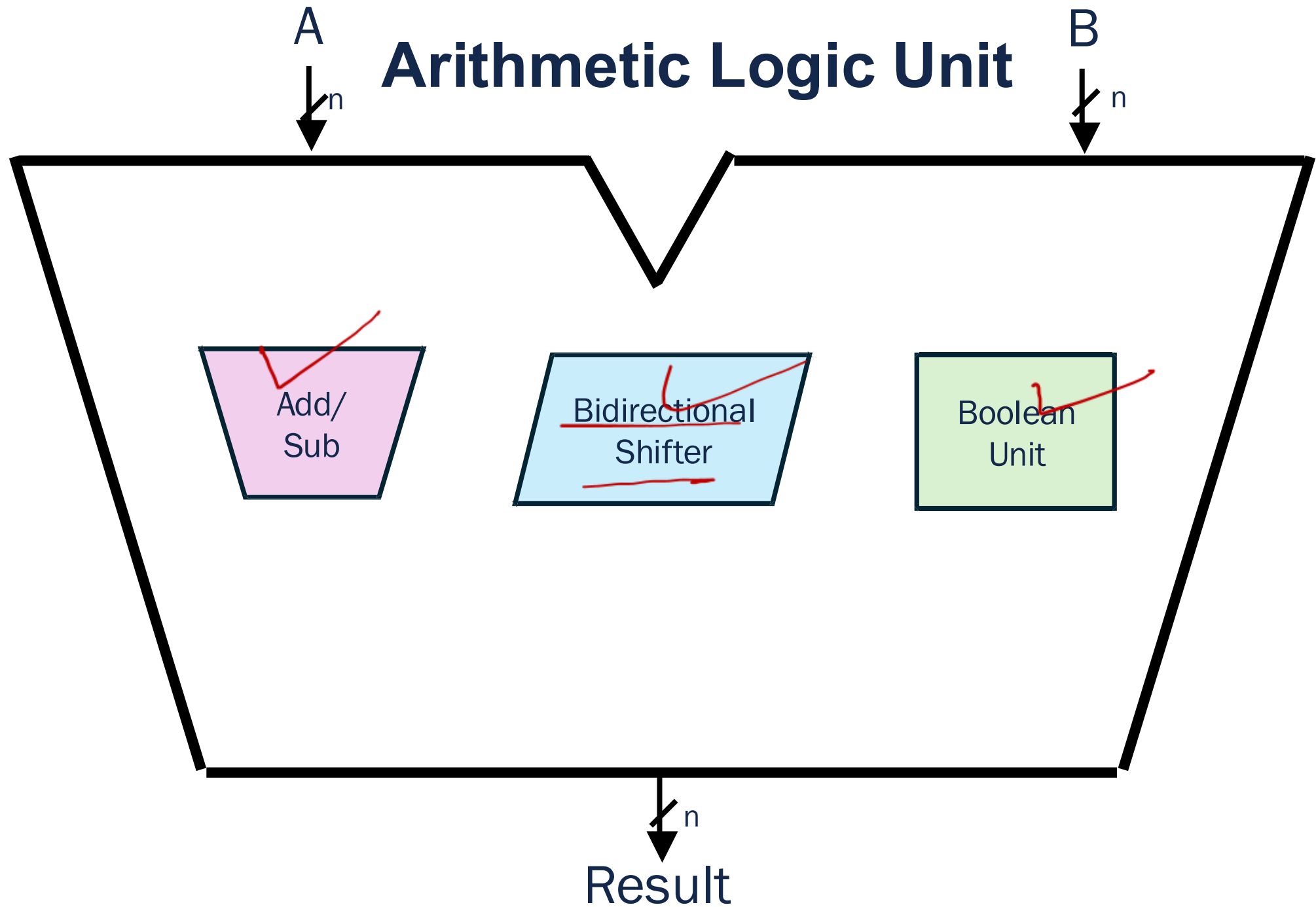
Operation	Expression	Bool
Left Shift	$B \ll A$	0 0
<u>None</u>	N/A	0 1
Logical Right Shift	$B \gg A$	1 0
Arithmetic Right Shift	$B \ggg A$	1 1

Add/Sub

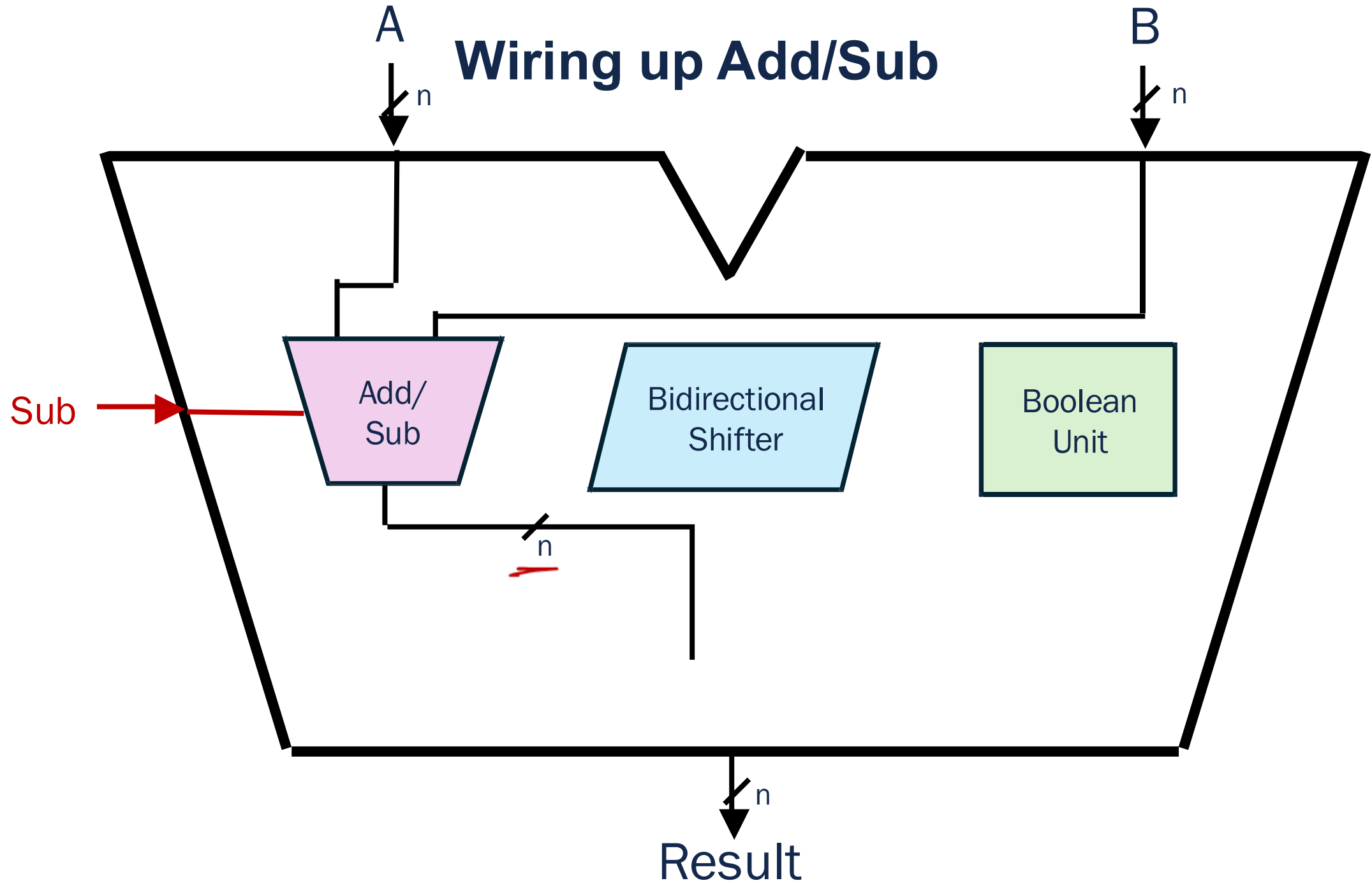


Operation	Expression	Sub
Add	$A + B$	0
Sub	$A - B$	1

Control signal: Tells the circuit which operation to perform

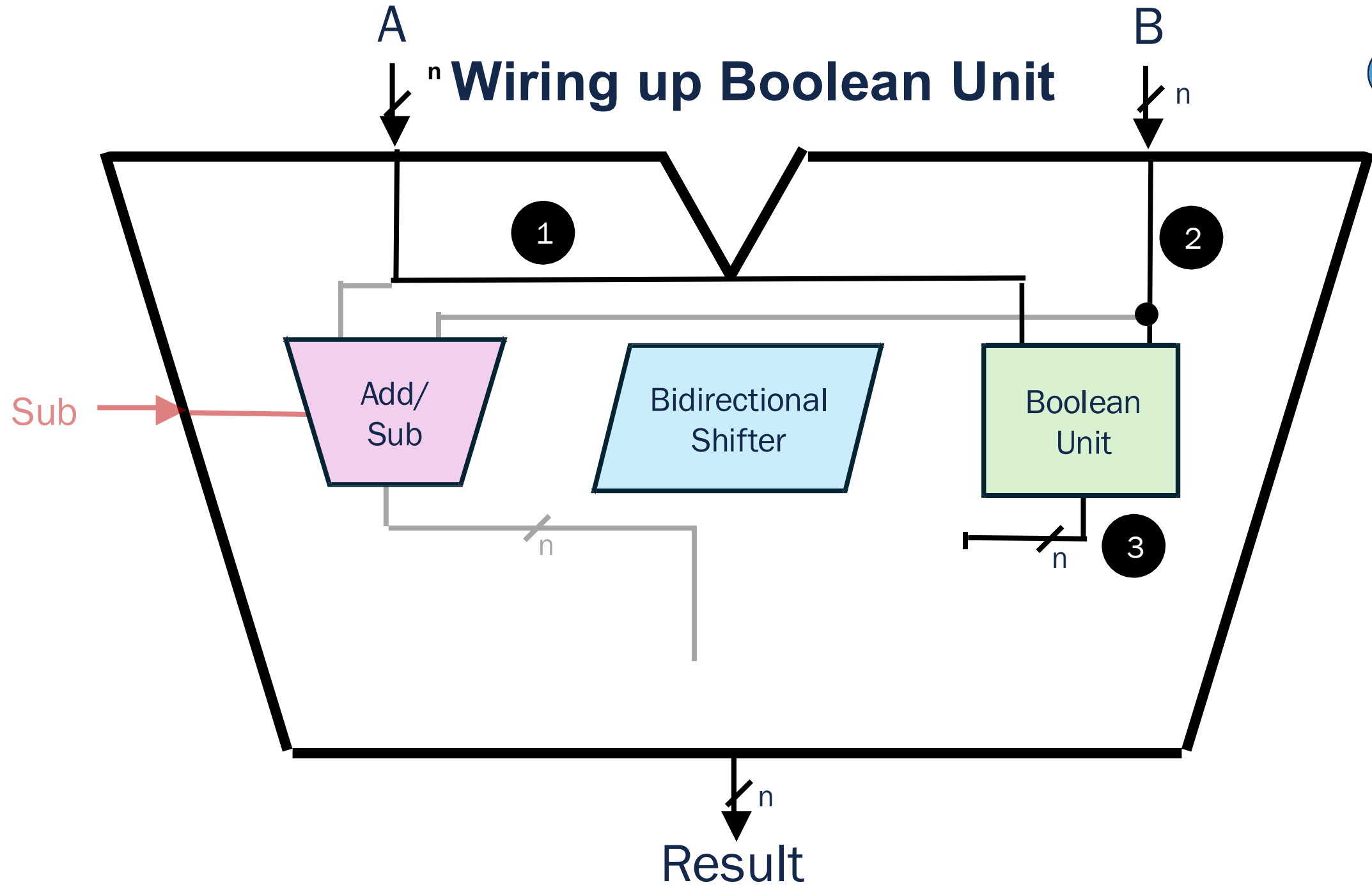


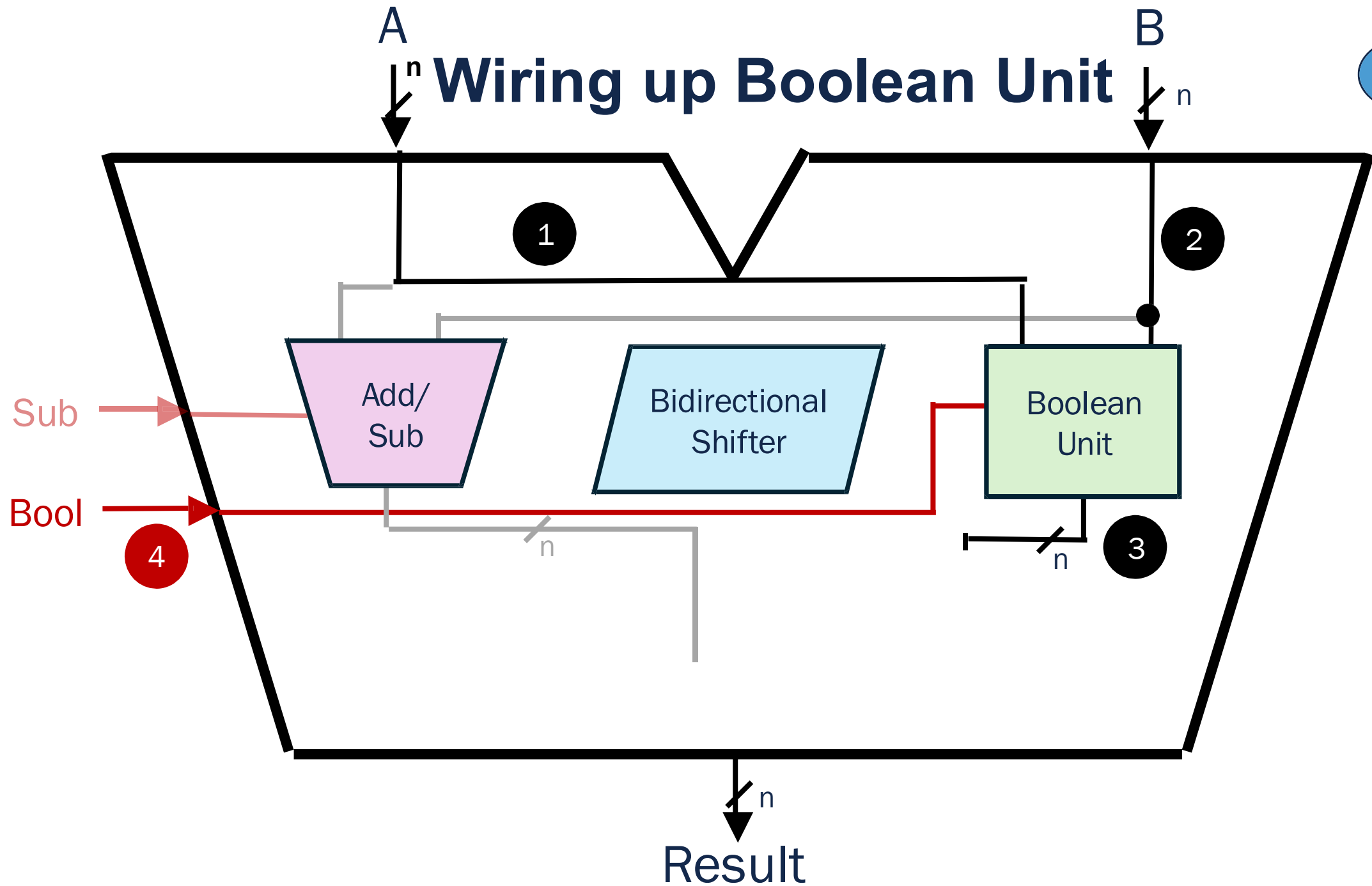
Wiring up Add/Sub



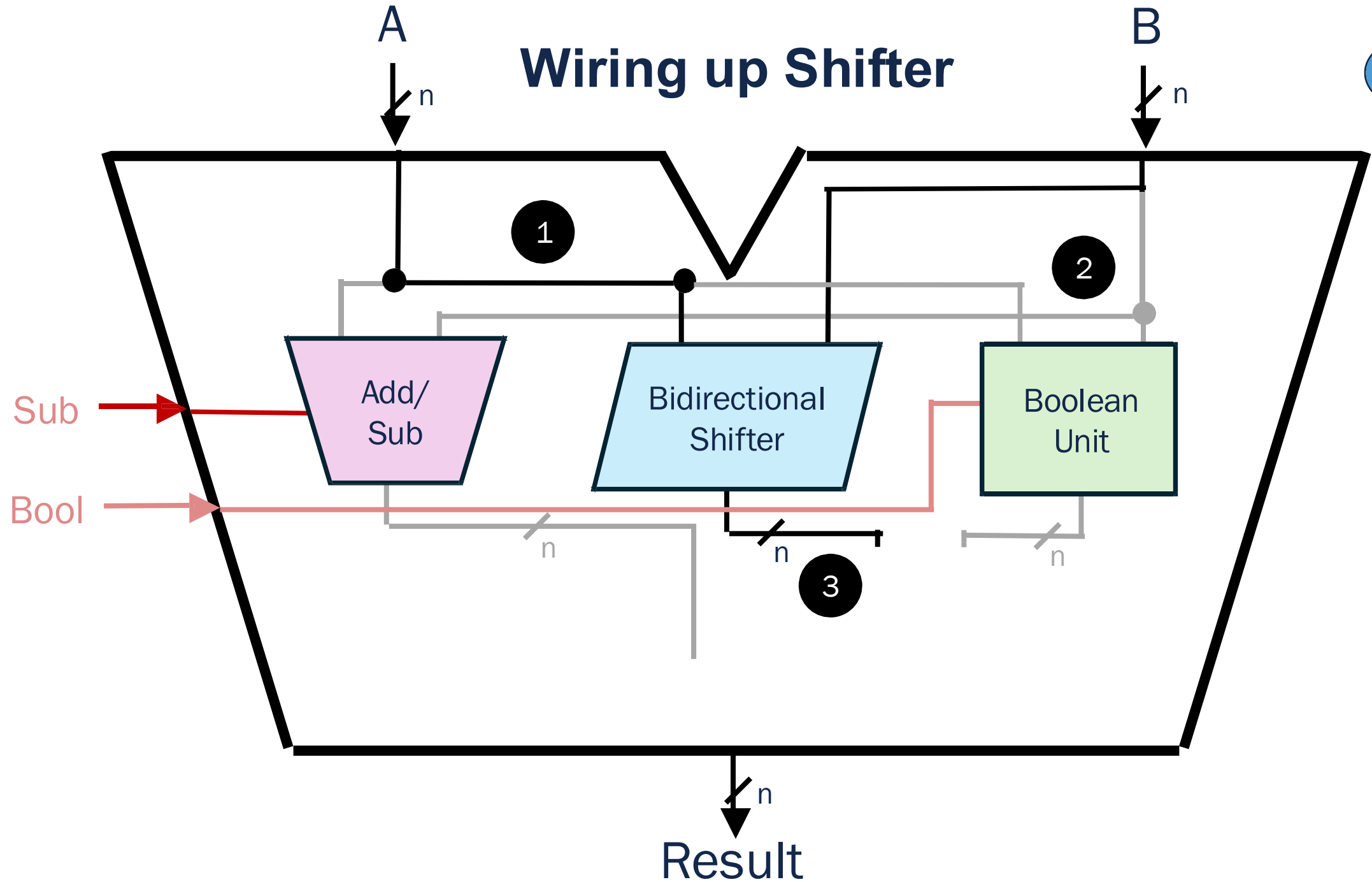


Wiring up Boolean Unit

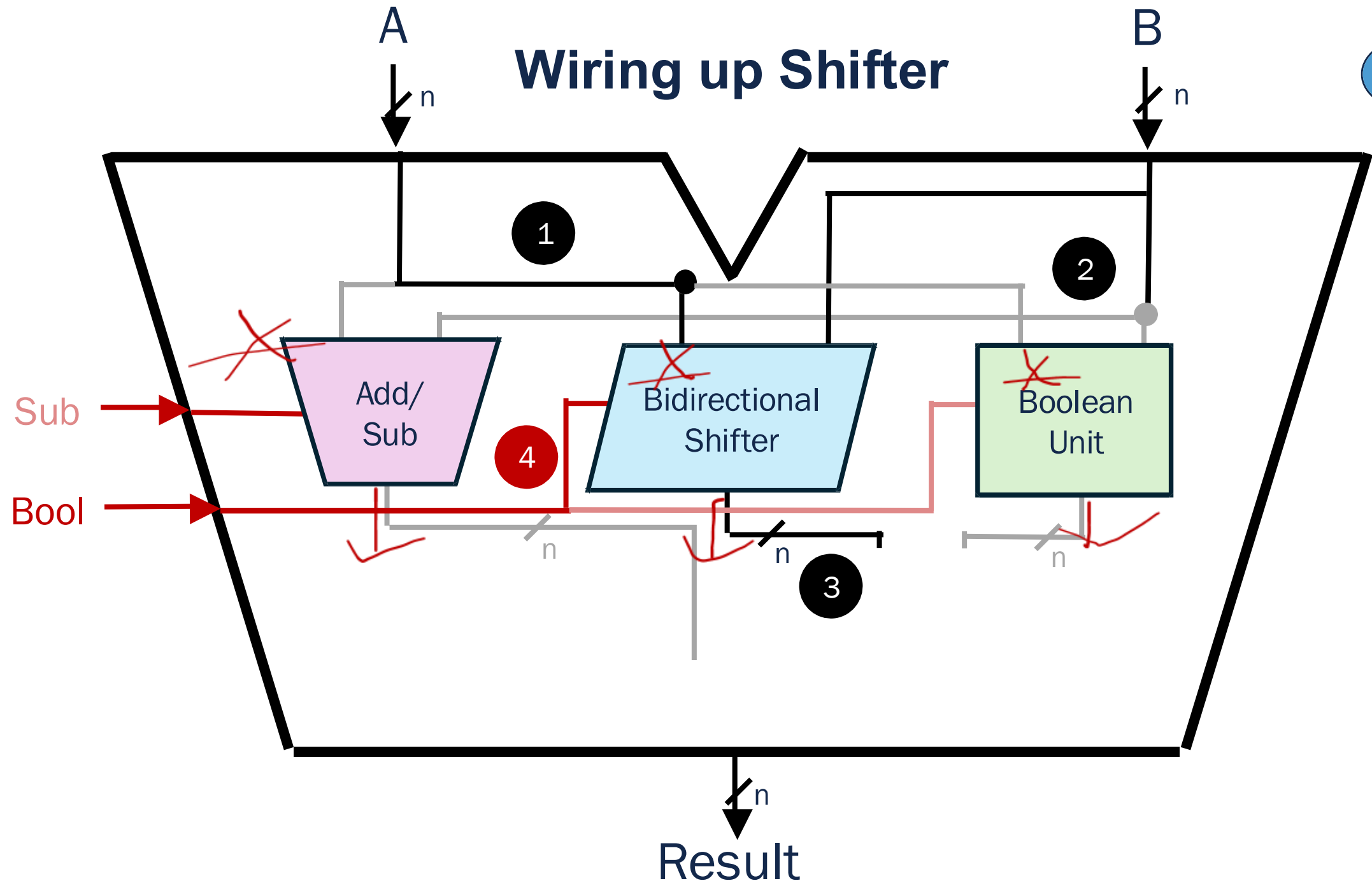




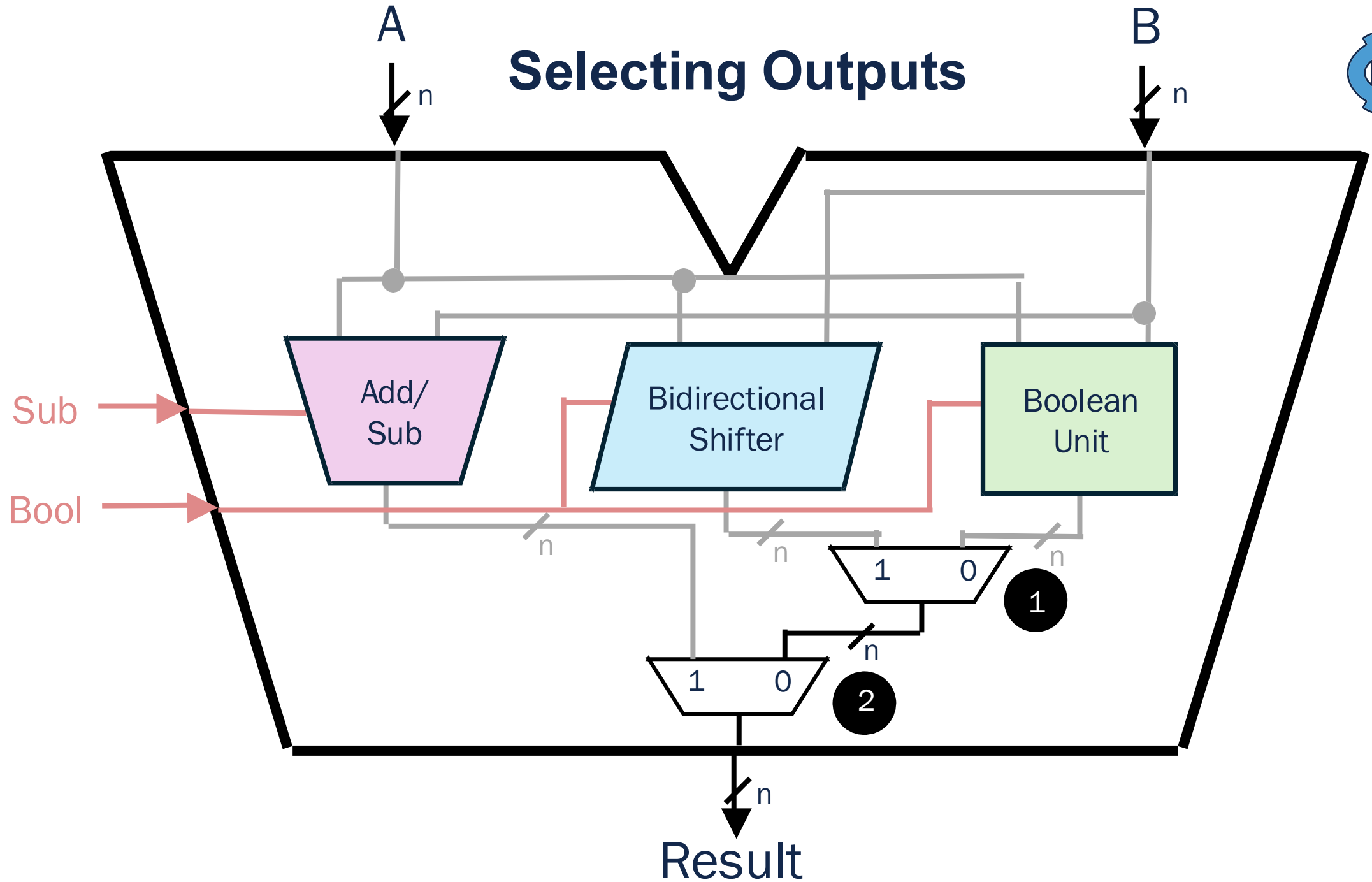
Wiring up Shifter



Wiring up Shifter

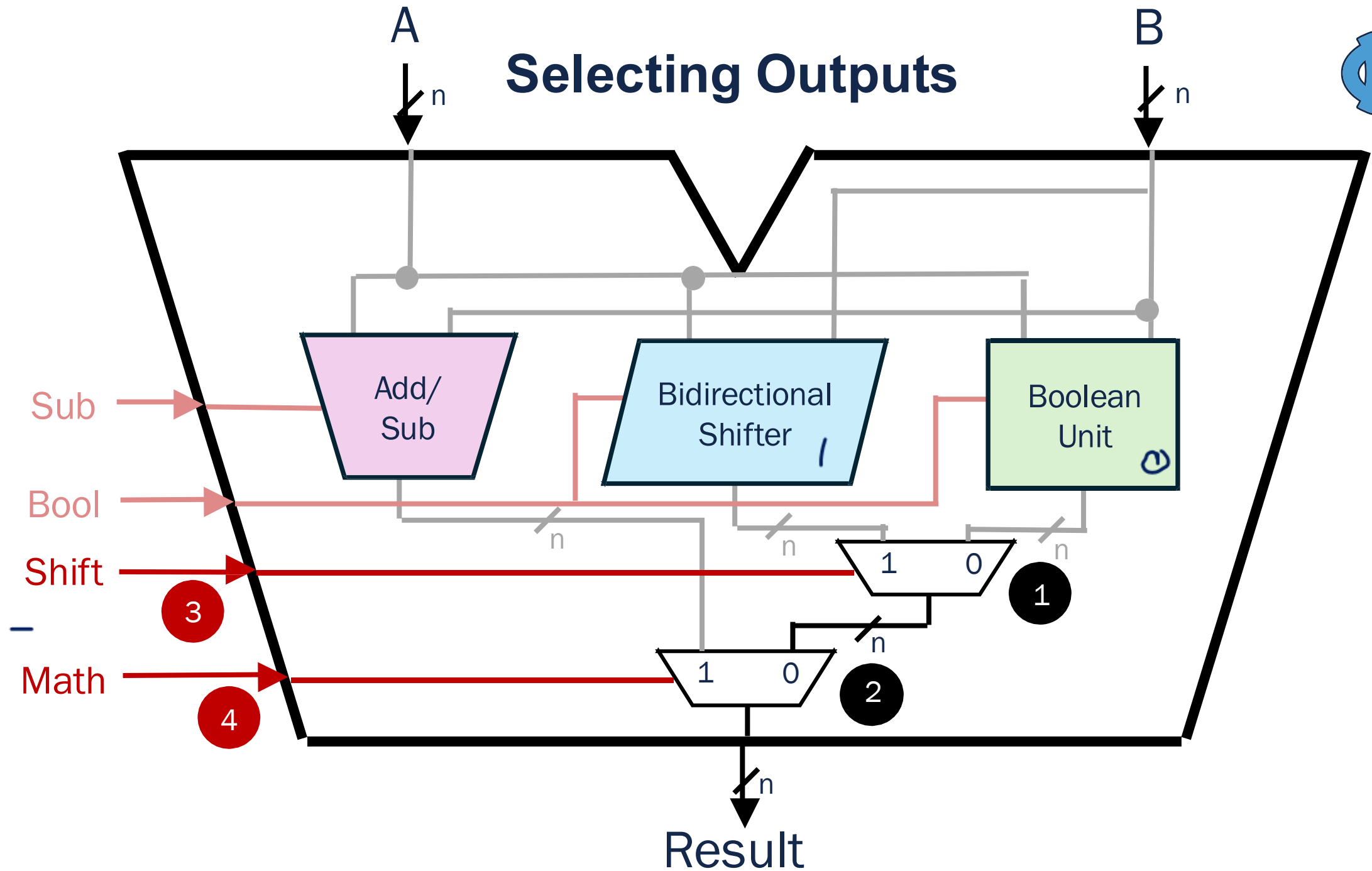


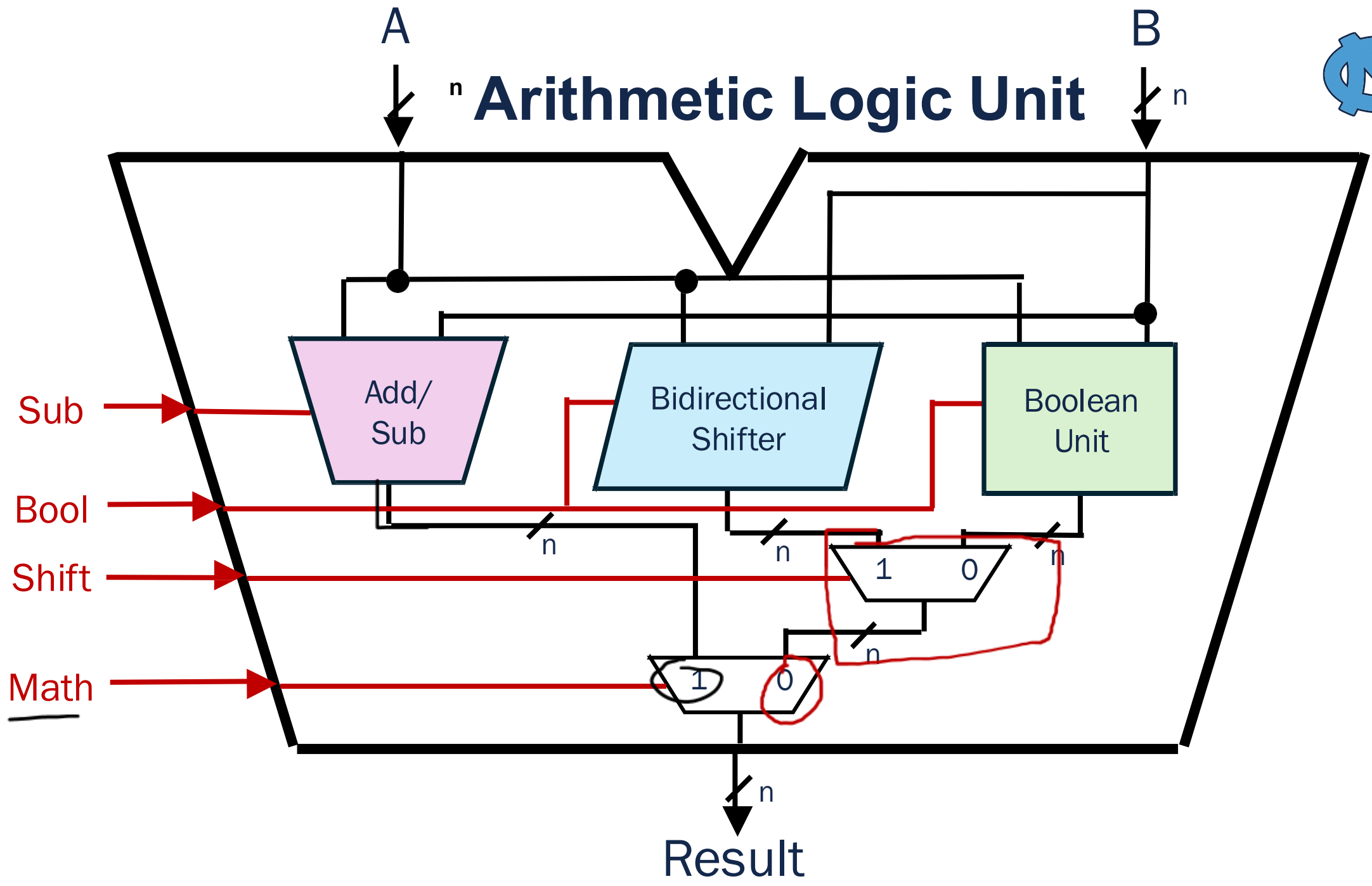
Selecting Outputs





Selecting Outputs





ALU Control Bits

Operation	Expression	Sub
Add	$A + B$	0
Sub	$A - B$	1

Operation	Expression	Bool
Left Shift	$B \ll A$	00
Logical Right Shift	$B \gg A$	10
Arithmetic Right Shift	$B \ggg A$	11

Operation	Expression	Bool
AND	$A \times B$	00
OR	$A + B$	01
XOR	$A \oplus B$	10
NOR	$\overline{A + B}$	11

Operation	Sub	Bool	Shift	Math
$A + B$	0	~~XX~~	~~XX~~	L
$A - B$	↓	~~XX~~	~~X~~	L
$B \ll A$	X	00	1	0
$B \gg A$	^	10	1	0
$B \ggg A$	X	L1	1	0
$A \text{ AND } B$				
$A \text{ OR } B$				
$A \text{ XOR } B$				
$A \text{ NOR } B$				

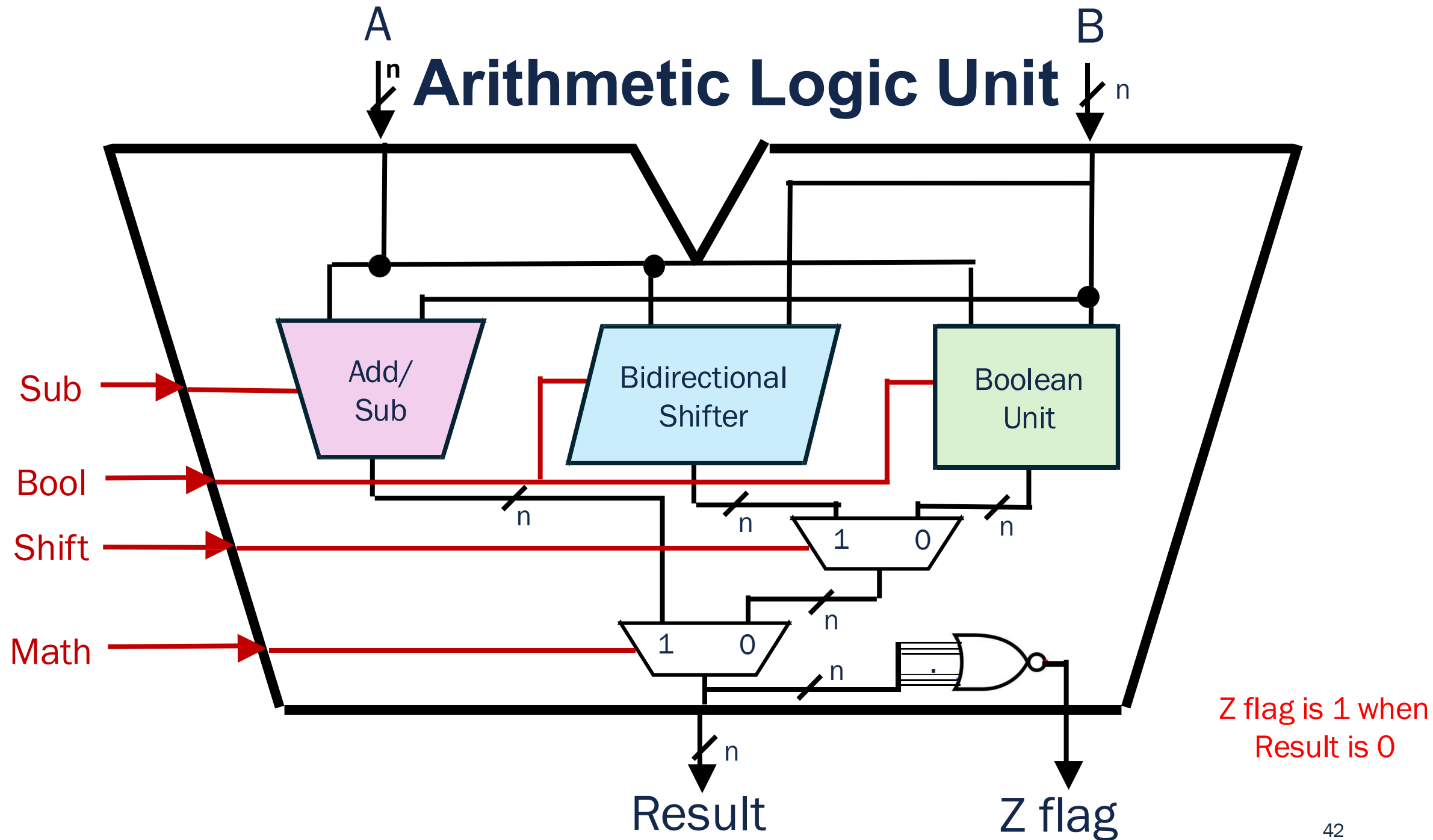
ALU Control Bits

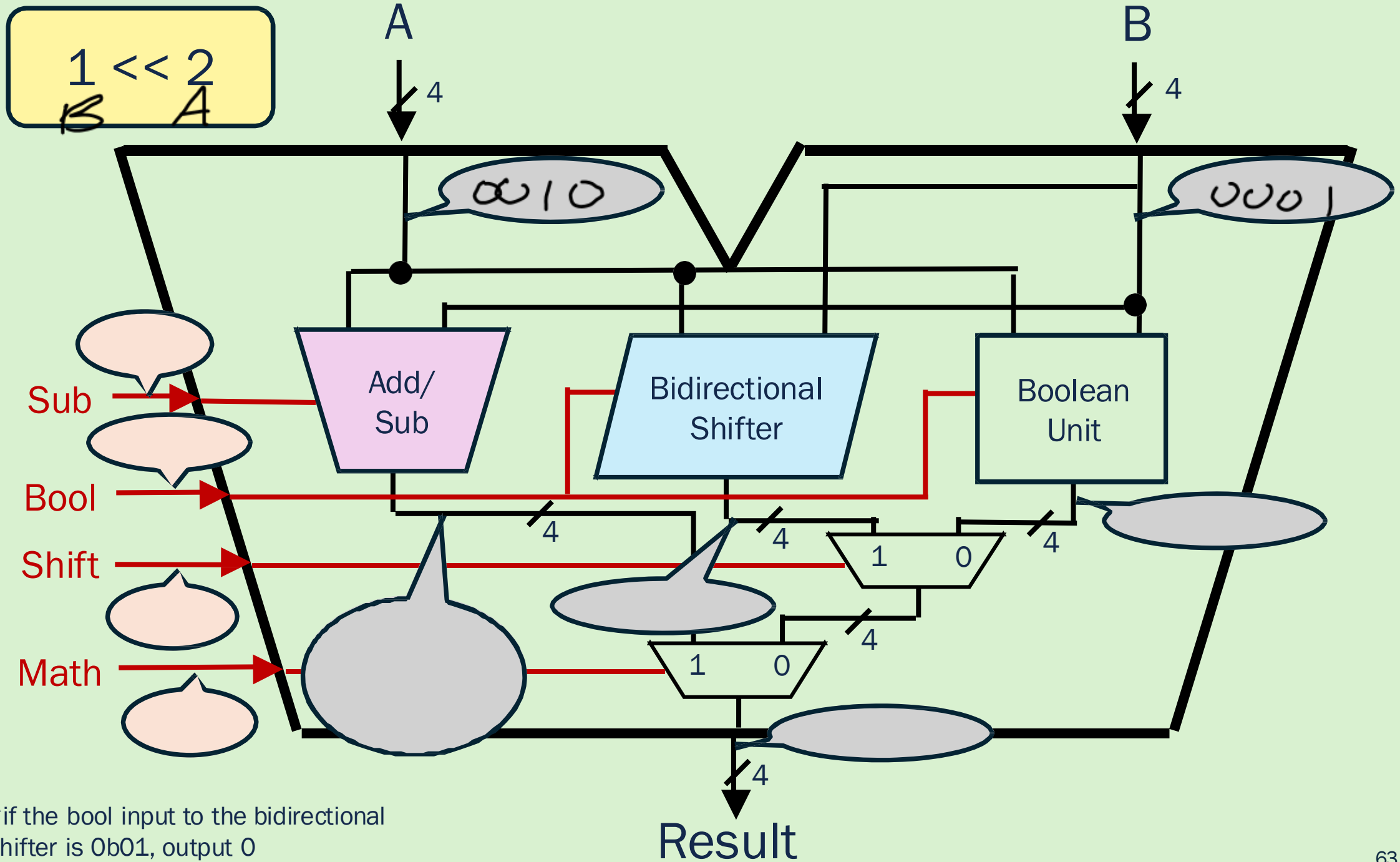
Operation	Expression	Sub
Add	$A + B$	0
Sub	$A - B$	1

Operation	Expression	Bool
Left Shift	$B \ll A$	00
Logical Right Shift	$B \gg A$	10
Arithmetic Right Shift	$B \ggg A$	11

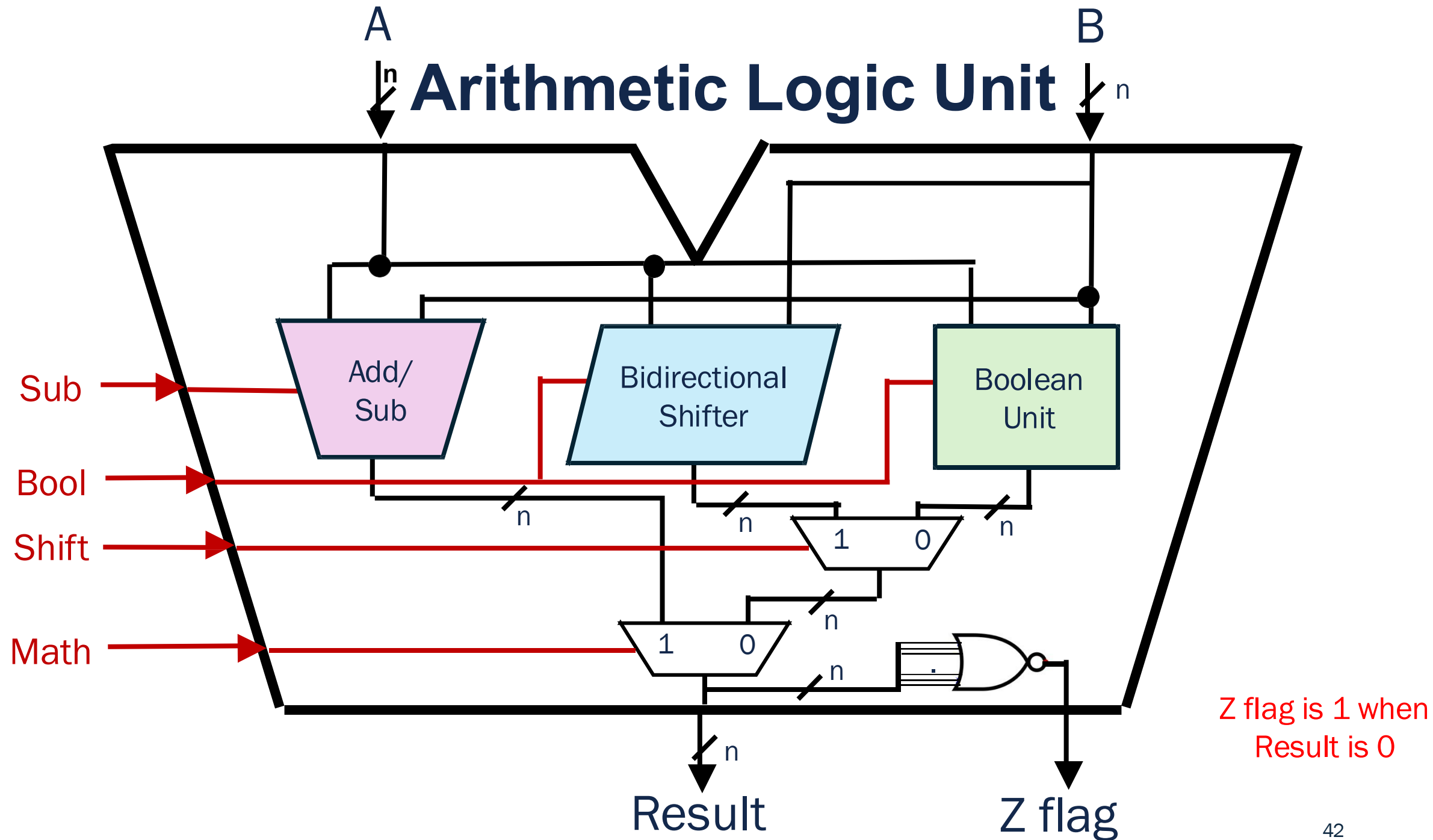
Operation	Expression	Bool
AND	$A \times B$	00
OR	$A + B$	01
XOR	$A \oplus B$	10
NOR	$\overline{A + B}$	11

Operation	Sub	Bool	Shift	Math
$A + B$	0b0	0bXX	0bX	0b1
$A - B$	0b1	0bXX	0bX	0b1
$B \ll A$	0bX	0b00	0b1	0b0
$B \gg A$	0bX	0b10	0b1	0b0
$B \ggg A$	0bX	0b11	0b1	0b0
$A \text{ AND } B$	0bX	0b00	0b0	0b0
$A \text{ OR } B$	0bX	0b01	0b0	0b0
$A \text{ XOR } B$	0bX	0b10	0b0	0b0
$A \text{ NOR } B$	0bX	0b11	0b0	0b0





*if the bool input to the bidirectional shifter is 0b01, output 0



Some Takeaways



- This is ***one possible*** ALU implementation based on MIPS-style instruction set
- The building blocks (addition, subtraction, shifting, Boolean operations, and multiplexers) are also used in RISC-V processors
 - You will build in Lab 3!! 😊
- Bits in the control table are implementation choices, not specified directly by the ISA
- **The ISA specifies what an instruction does. The ALU design determines how the hardware implements it.**