CS140 Lecture 03:
The Machinery of Computation:
Combinational Logic

John Magee

25 January 2017

Some slides credit Aaron Stevens

Overview/Questions

- What did we do last time?
- Can we relate this circuit stuff to something we know something about?
- How can we combine these elements to do more complicated tasks?
- By combining several gates, we create logic-computing circuits.
- Logic-computing circuits can do binary number addition.

What did we talk about last time?

- Circuits control the flow of electricity.
- Gates are simple logical systems.

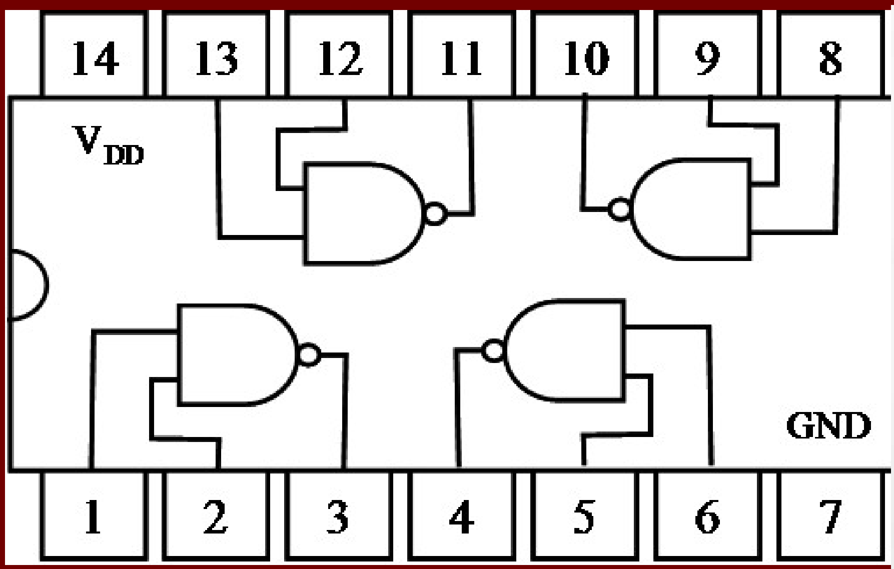
Integrated Circuits

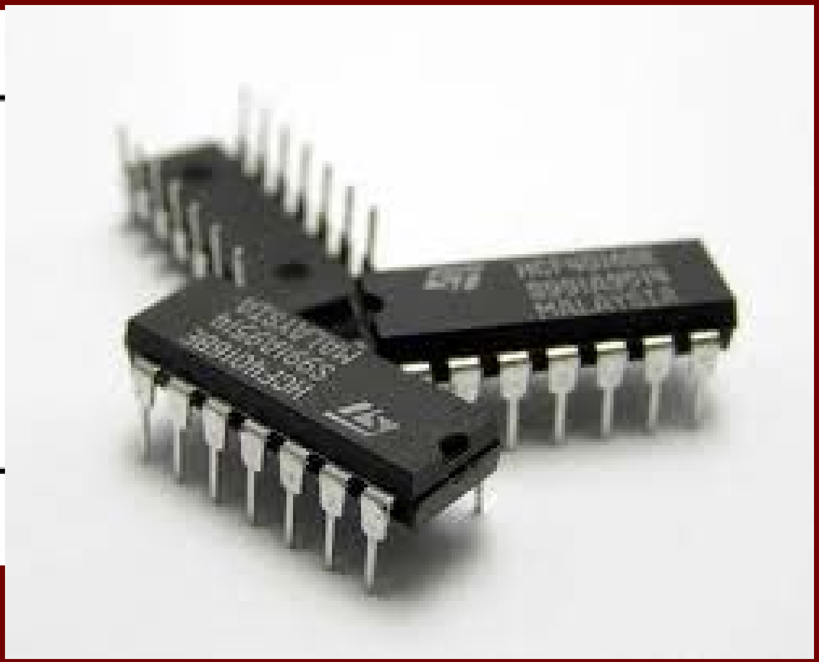
Integrated circuit (also called a *chip*)

A piece of silicon on which multiple (many) gates have been embedded.

Silicon pieces are mounted on a plastic or ceramic package with pins along the edges that can be soldered onto circuit boards or inserted into appropriate sockets

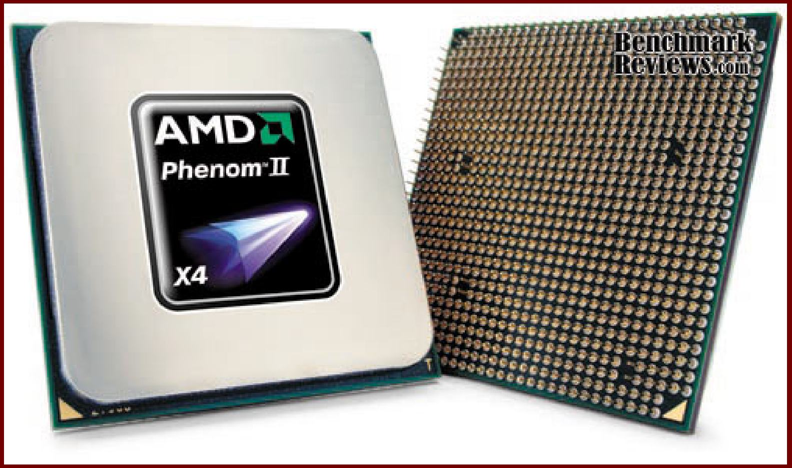
Integrated Circuits





Central Processor Units

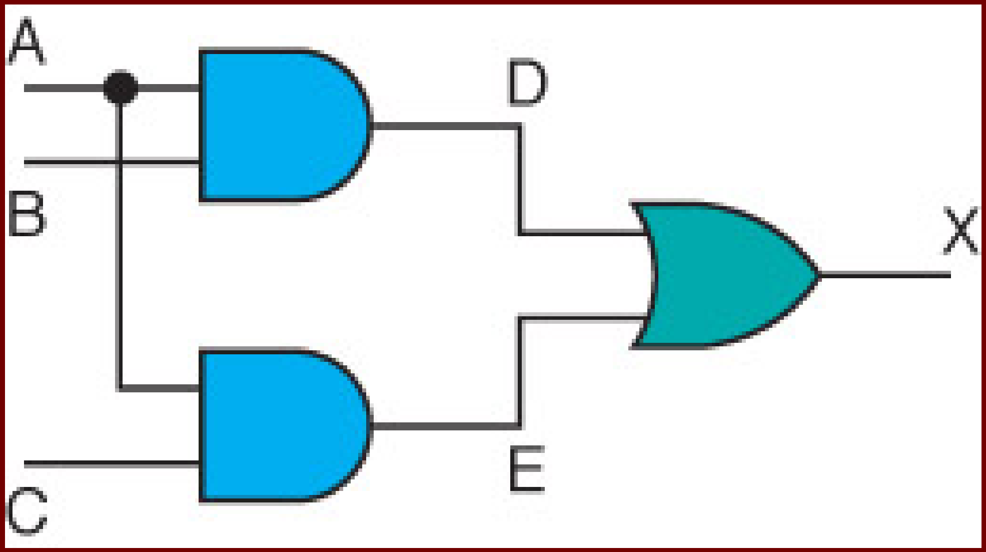
- The most important integrated circuit in any computer is the Central Processing Unit, or CPU.
  - The Intel *Duo Core 2 ®* processor has more than 1.9 billion ($1.9 * 10^9$) gate transistors on one chip.
- The CPU combines many gates, to enable a small number of instructions. Examples:
  - Add/subtract 2 binary inputs
  - Load a value from memory
  - Store a value into memory



Combinational Circuits

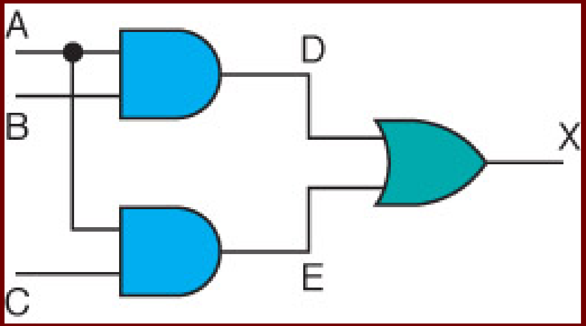
Combines some basic gates (AND, OR, XOR, NOT) into a more complex circuit.

- Outputs from one circuit flow into the inputs of another circuit.
- The **input values** explicitly determine the output values.



Combinational Circuits

Three inputs require eight rows to describe all possible input combinations ($2^3 = 8$):



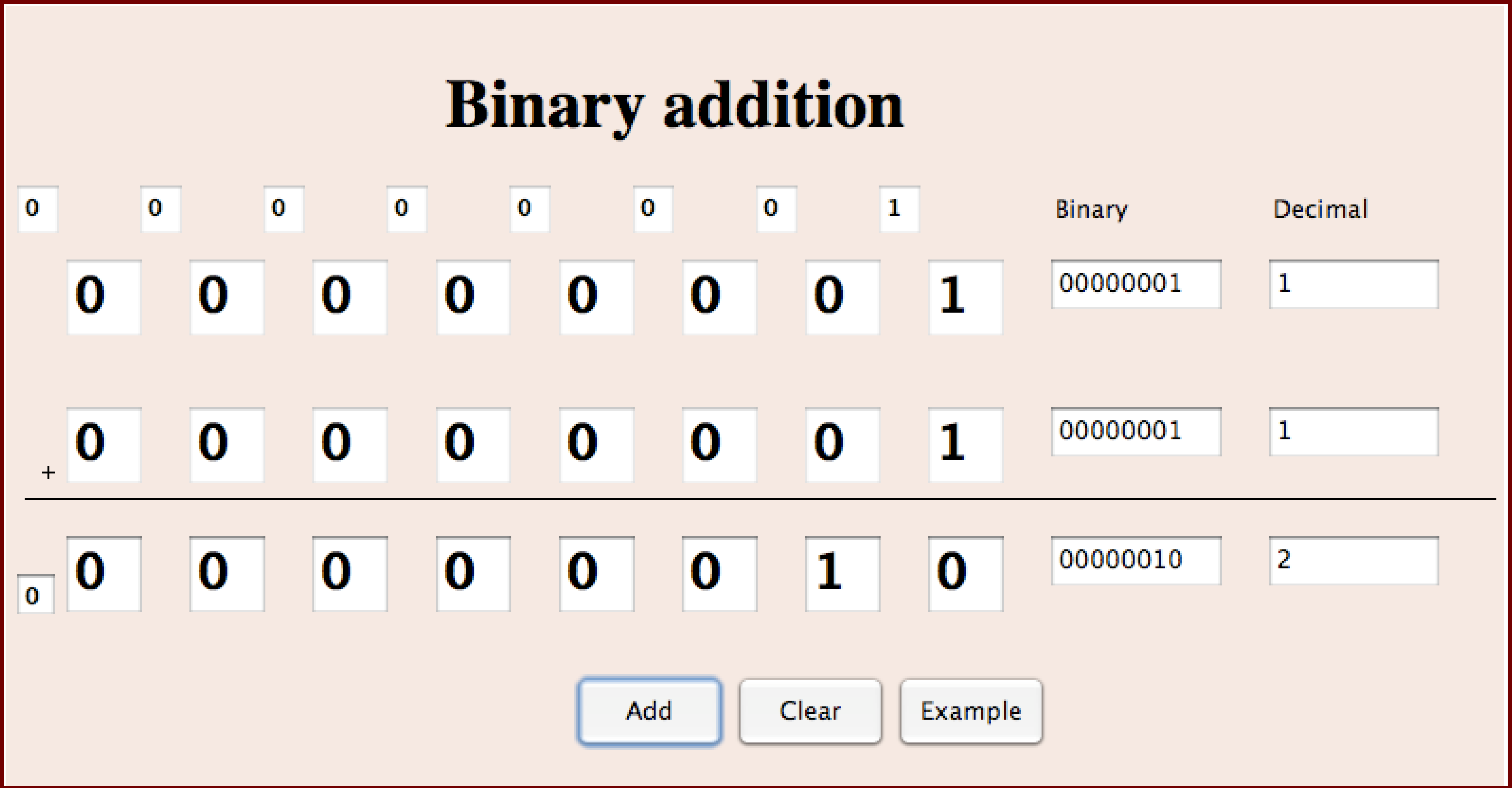
A	B	C	D	E	X
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

This same circuit using a Boolean expression is (AB + AC)

Recall Binary Number Addition

Adding two 1-bit numbers together produces

- A sum bit
- A carry bit



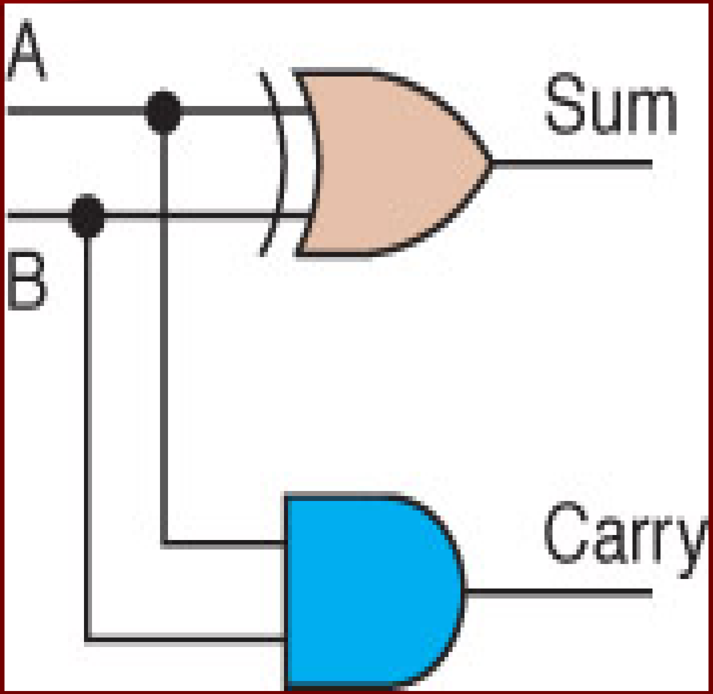
Binary Number Addition

Look closely at the values for Sum and Carry...

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Do they look like any of the gates we've seen?

A Circuit for Binary Addition



Sum = A XOR B
Carry = A AND B

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

This circuit is called a half-adder.

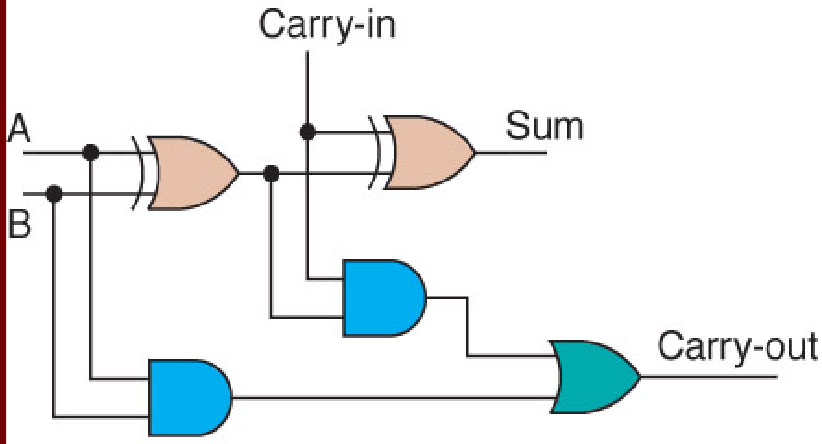
(It doesn’t take a carry-in.)

Full Adder Circuit

The full adder takes 3 inputs:

– A, B, and a carry-in value

Logic Diagram



Truth Table

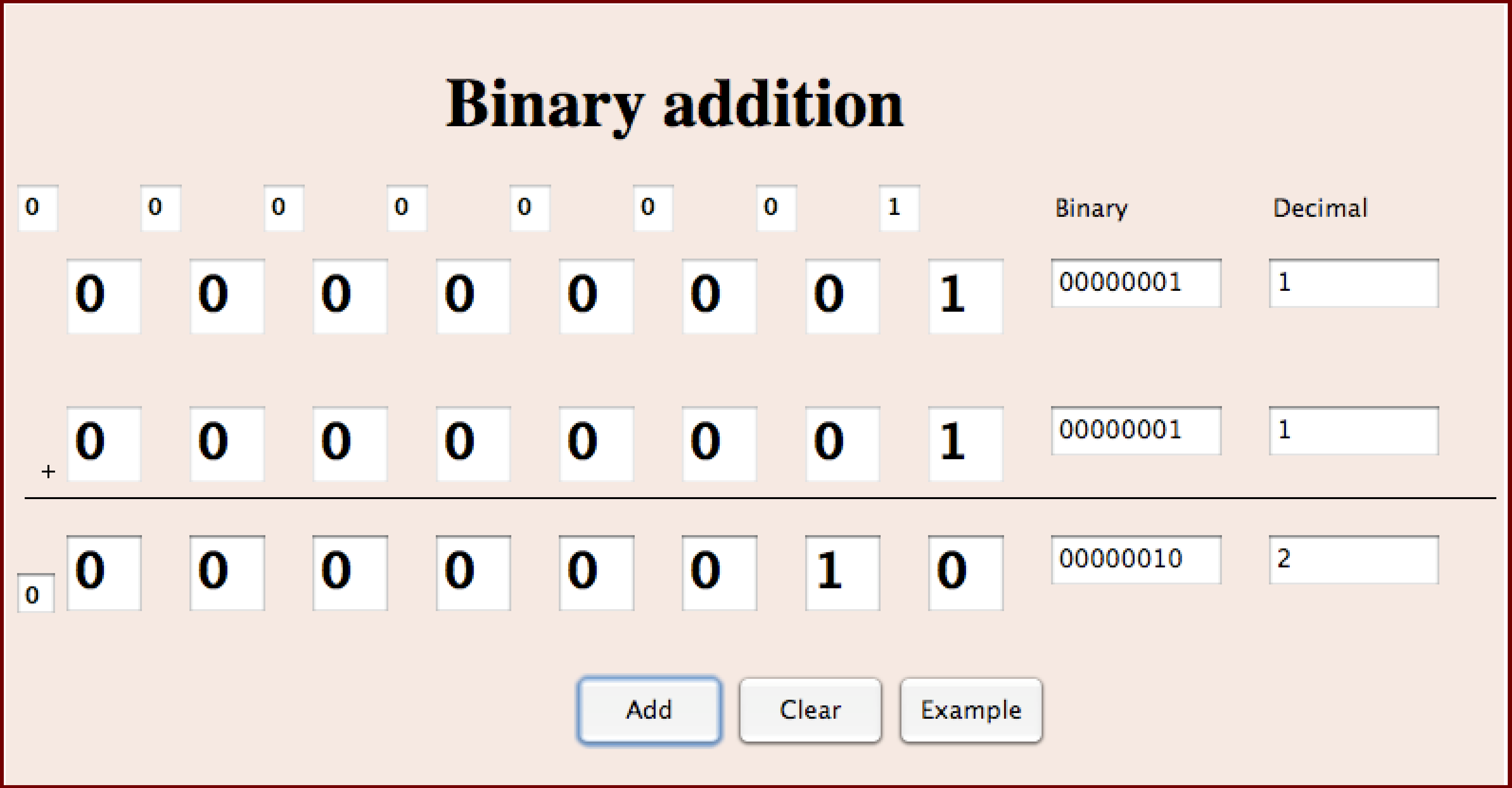
A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 4.10 **A full adder**

Recall Binary Number Addition

Adding two 1-bit numbers together produces

- A sum bit
- A carry bit



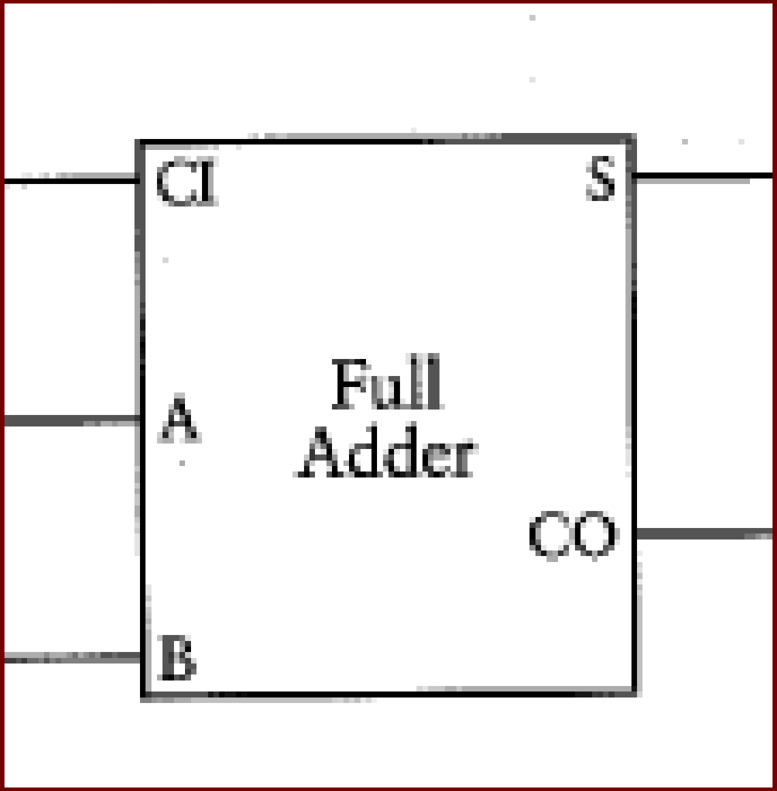
http://www.cs.bu.edu/courses/cs101/labs/ECS_2e/Applets/APPLETS/BINARYADD/applet_frame.htm

The Full Adder

Here is the Full Adder, with its internal details hidden (an abstraction).

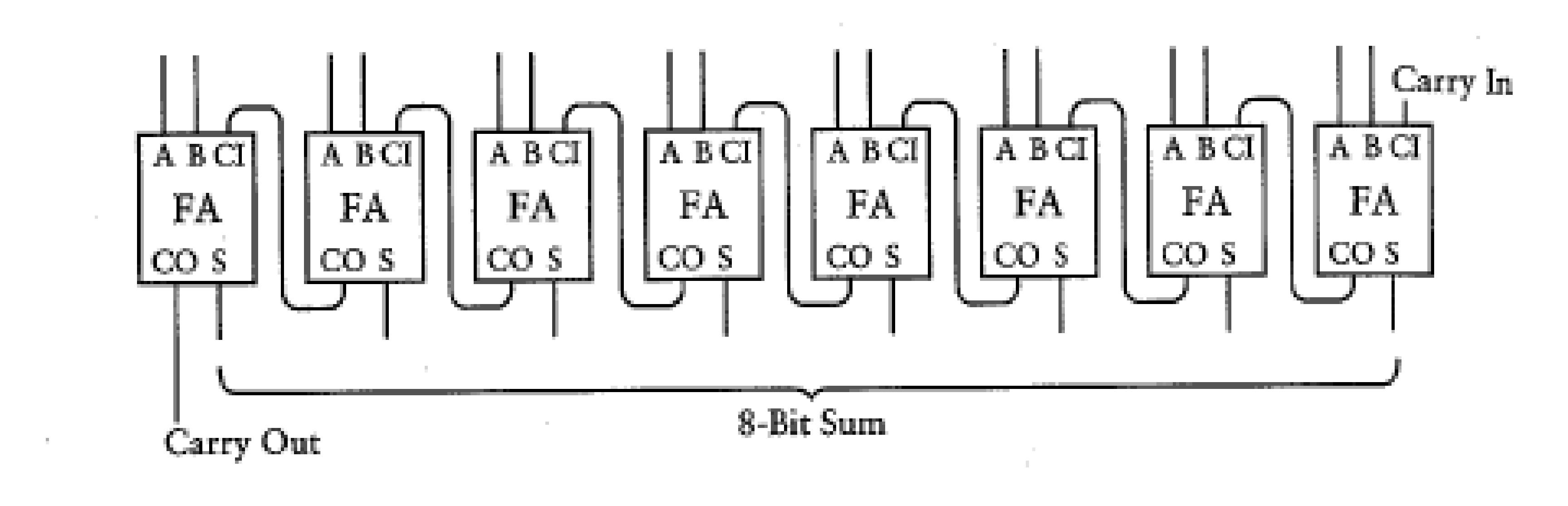
What matters now are:

- inputs are A, B, and CI.
- outputs are S and CO



An 8-bit Adder

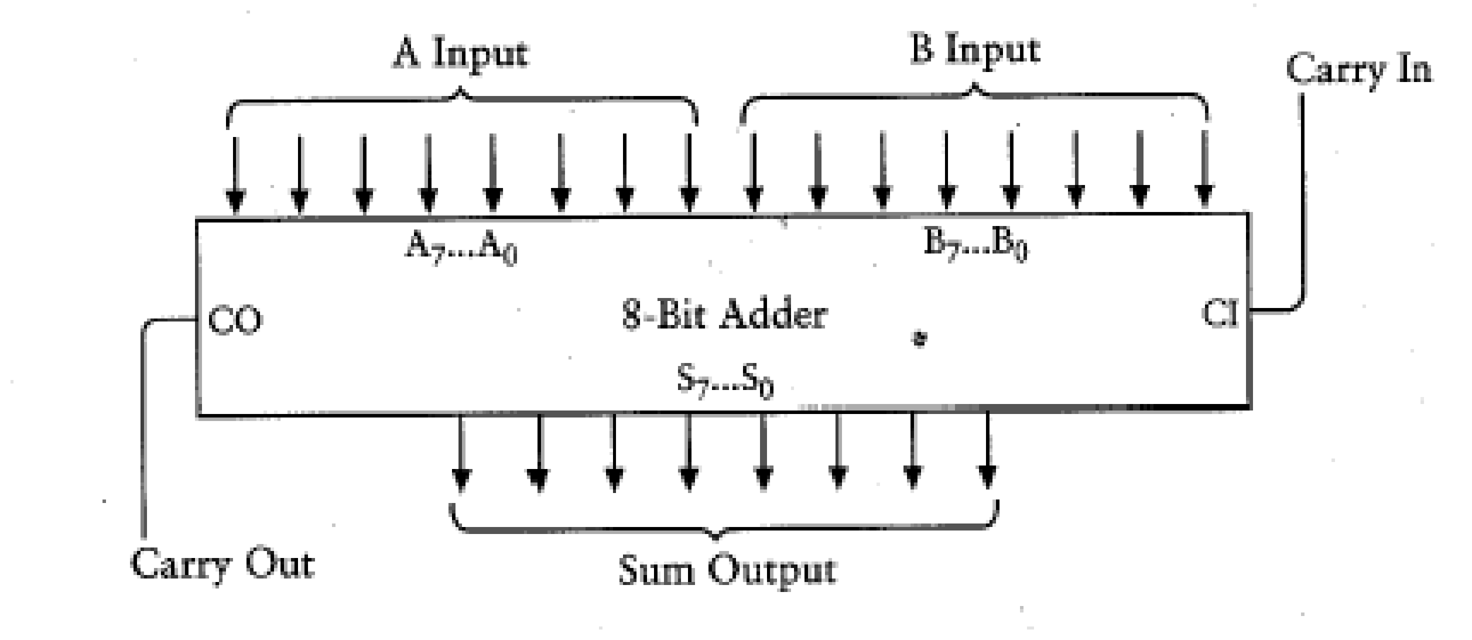
To add two 8-bit numbers together, we need an 8-bit adder:



Notice how the carry out from one bit's adder becomes the carry-in to the next adder.

An 8-bit Adder

We can abstract away the 1-bit adders,
And summarize with this diagram:



Notice the inputs and outputs.

Output from the Adder

The adder produces 2 outputs:

– Sum (multi-bit), Carry Out (1-bit)

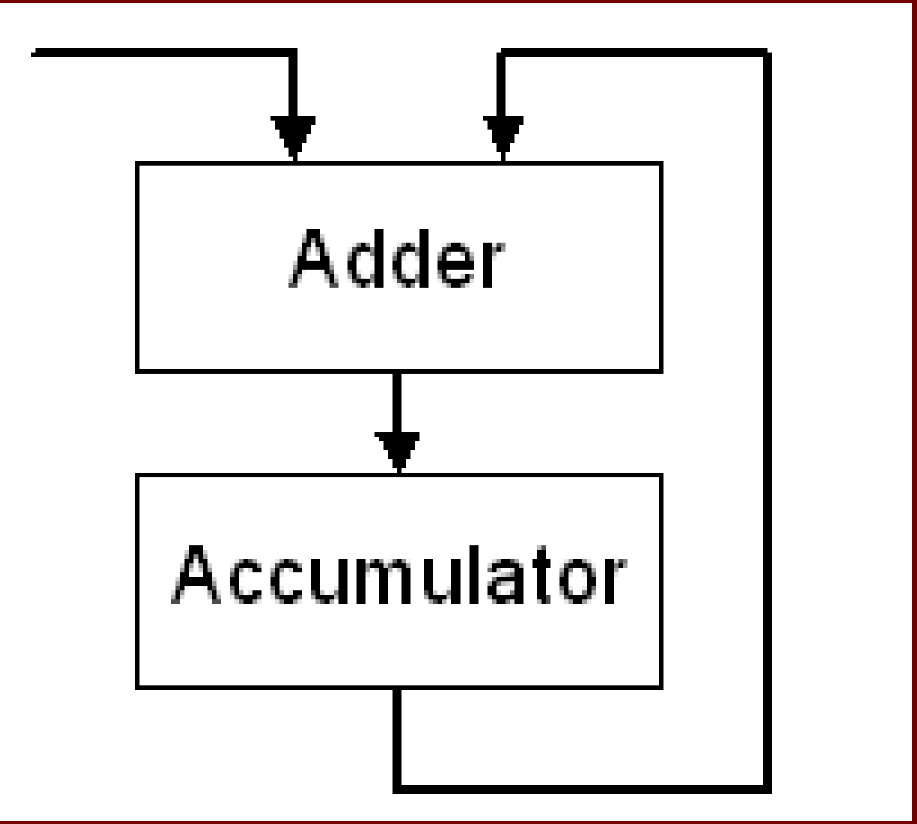
Where does the output go from here?

Accumulator

A circuit connected to an adder, which stores the adder's result.

Putting it Together

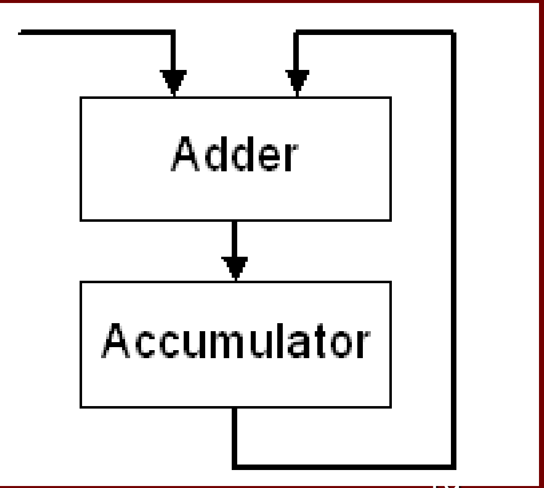
The accumulator is a memory circuit, and is wired as both an output from the adder and an input back into the adder.



Accumulator Example

Suppose we want to add 3 numbers:

1) Clear the accumulator (set to all 0s)
2) Load the first input into the adder
3) Compute the sum of accumulator + input
4) Result flows back into accumulator
5) Go to step 2 with next input



Input to the Adder

The adder takes inputs

- A, B are two binary numbers
- (Carry-in should be 0)
  - Our nand2tetris adder will be slightly different

How do we feed numbers into the adder?

Random Access Memory

A large memory unit which stores data before/after processing.

What about Subtraction?

2s complement

Recall that binary subtraction is accomplished by adding the 2s complement of a number.

Inverter

A circuit built using NOT gates, which inverts all bits – turning 1s into 0s and 0s into 1s.

- The inverter creates a 1s complement of its input.
- Adding 1 to this gives a 2s complement number, suitable for doing subtraction.
- (How could we add 1 to the inverted number?)

What about Subtraction?

In nand2tetris world:

Note that $x - y = -(-x + y)$

Since we don't need to store intermediate results, this can be done in 1's complement:

- Flip the bits of x (this computes –x in 1's complement)
- Add y
- Flip the bits of the result

From Adding Machine...

What we've got is an machine that can do addition/subtraction in circuitry.

It can read data from memory, and write data back to memory.

We haven't dealt with how to:

- Specify from which address memory to read.
- Specify which operation to perform (add/subtract).
- Specify to which address to write.

Take-Away Points

- Combination gates
- Half-Adder
- Full Adder
- Adder
- Inverter
- Accumulator