

II

Fundamentals Of Digital Logic

Our Goals

- Understand
 - Fundamentals and basics
 - Concepts
 - How computers work at the lowest level
- Avoid whenever possible
 - Device physics
 - Engineering design rules
 - Implementation details

Electrical Terminology

- Voltage
 - Quantifiable property of electricity
 - Measure of potential force
 - Unit of measure: *volt*
- Current
 - Quantifiable property of electricity
 - Measure of electron flow along a path
 - Unit of measure: *ampere (amp)*

Analog For Electricity

- Voltage is analogous to water pressure
- Current is analogous to flowing water
- Can have
 - High pressure with little flow
 - Large flow with little pressure

Voltage

- Device used to measure called *voltmeter*
- Can only be measured as difference between two points
- To measure voltage
 - Assume one point represents zero volts (known as *ground*)
 - Express voltage of second point wrt ground

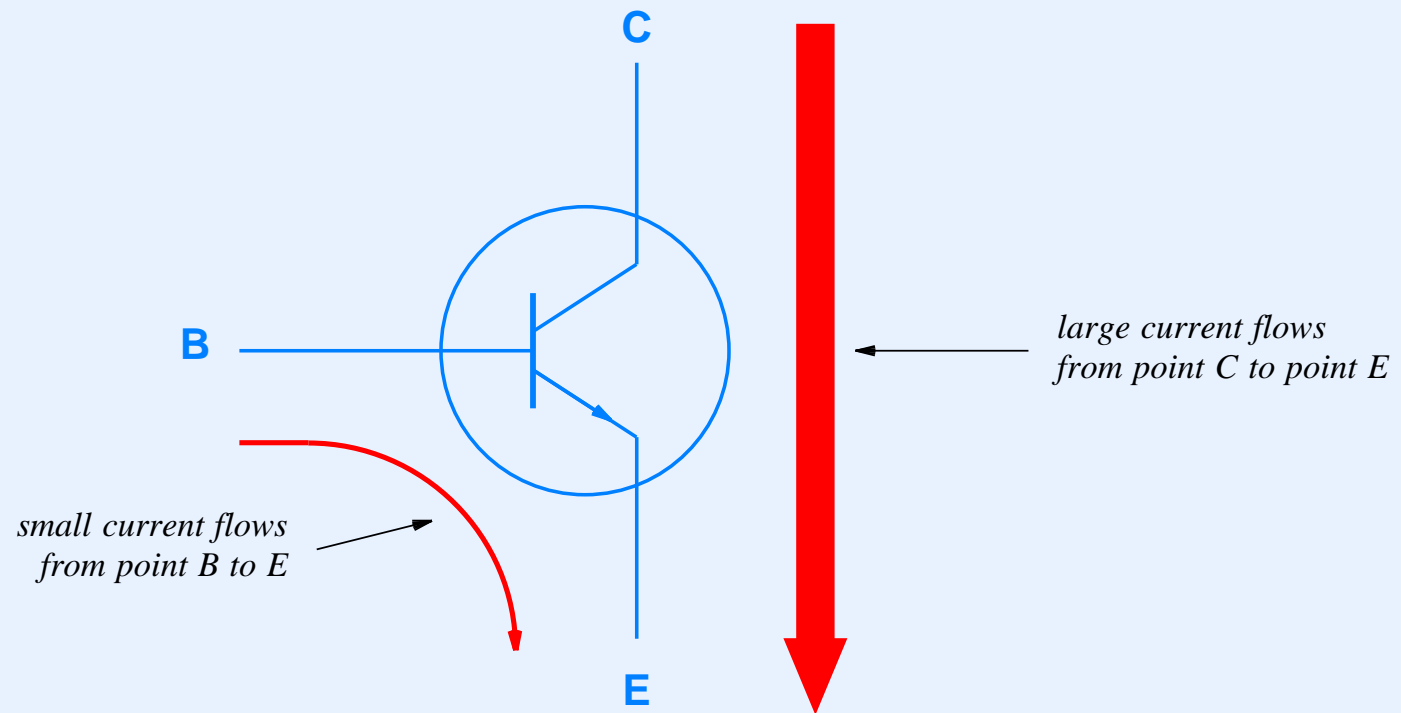
In Practice

- Typical digital circuit operates on five volts
- Two wires connect each chip to *power supply*
 - Ground (zero volts)
 - Power (five volts)
- Digital logic diagrams do not usually show power and ground connections

Transistor

- Basic building block of digital circuits
- Operates on electrical current
- Acts like a miniature switch — small input current controls flow of large current
- Three external connections
 - Emitter
 - Base (control)
 - Collector
- Current between base and emitter controls current between collector and emitter

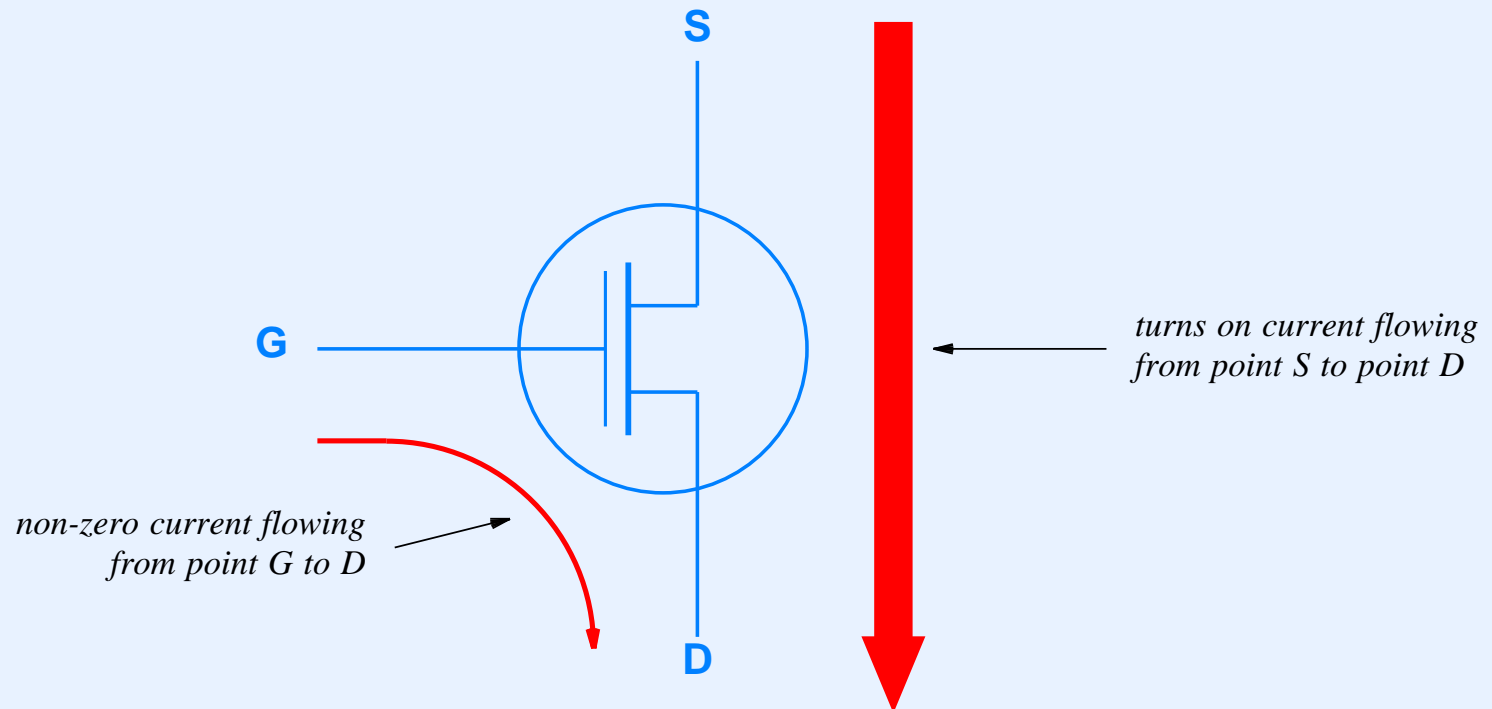
Illustration Of A Traditional Transistor



- Traditional transistor is an *amplifier*: large output current is proportional to small input current

Field-Effect Transistors

- Used on digital chips (CMOS)
- Configured to act as a switch (on or off)



Boolean Logic

- Mathematical basis for digital circuits
- Three basic functions: *and*, *or*, and *not*

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

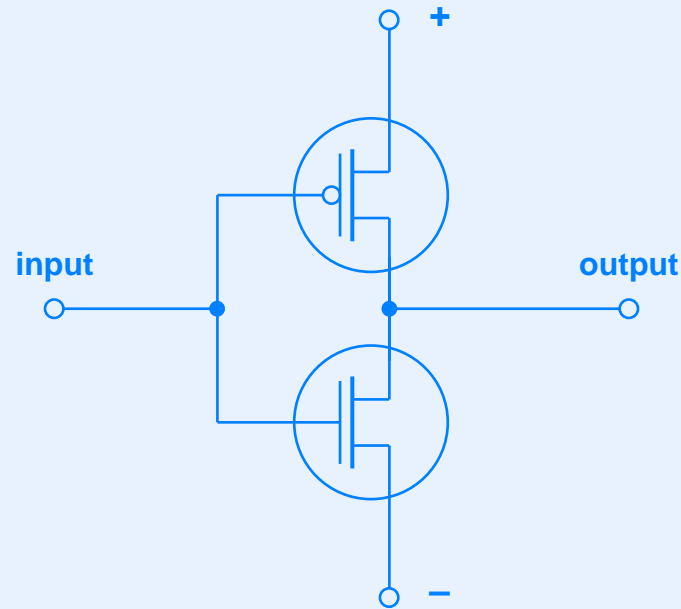
A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

A	not A
0	1
1	0

Digital Logic

- Can implement Boolean functions with transistors
- Five volts represents Boolean *1 (true)*
- Zero volts represents Boolean *0 (false)*

Transistors Implementing Boolean Not



- Transistor with circle on gate turns *on* when input is zero
- When input is zero volts, output is five volts; when input is five volts, output is zero volts

Logic Gate

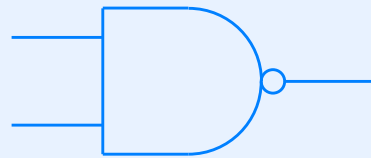
- Hardware component
- Consists of integrated circuit
- Implements an individual Boolean function
- To reduce complexity, provide inverse of Boolean functions
 - Nand gate implements *not and*
 - Nor gate implements *not or*
 - Inverter implements *not*

Truth Tables For Nand and Nor Gates

A	B	A nand B
0	0	1
0	1	1
1	0	1
1	1	0

A	B	A nor B
0	0	1
0	1	0
1	0	0
1	1	0

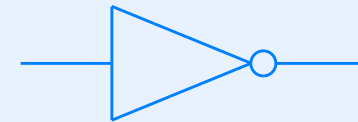
Symbols Used In Schematic Diagrams



nand gate

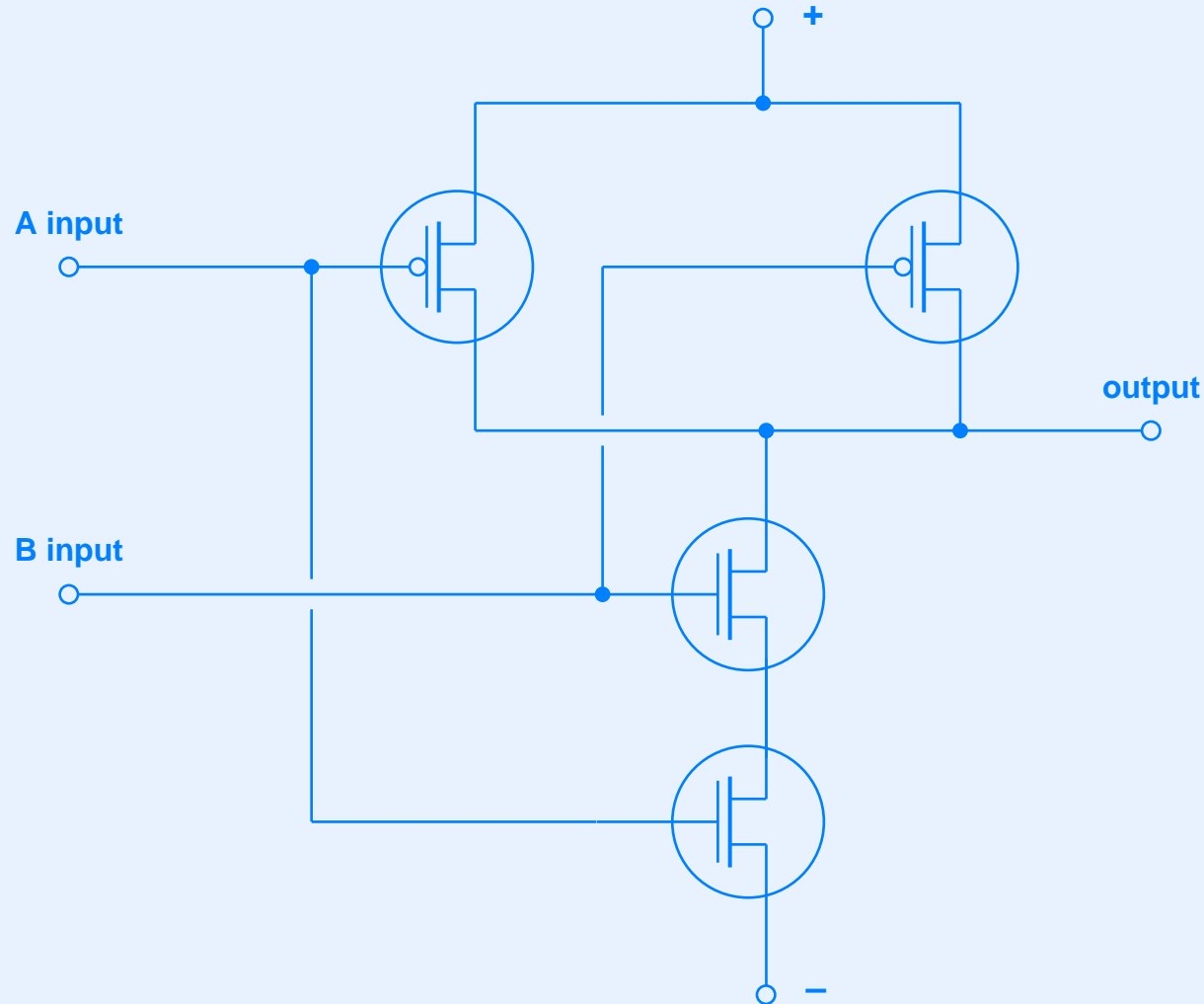


nor gate



inverter

Example Of Internal Gate Structure (Nand Gate)



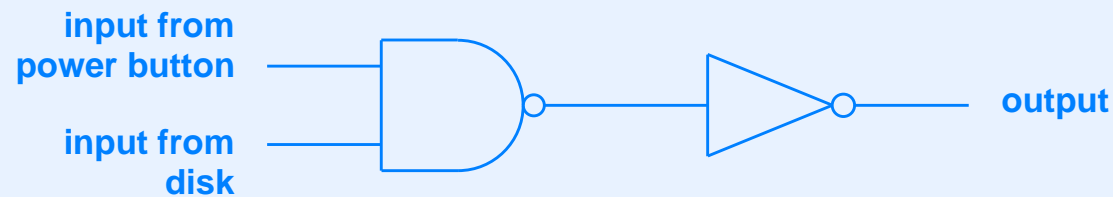
- Solid dot indicates electrical connection

Technology For Logic Gates

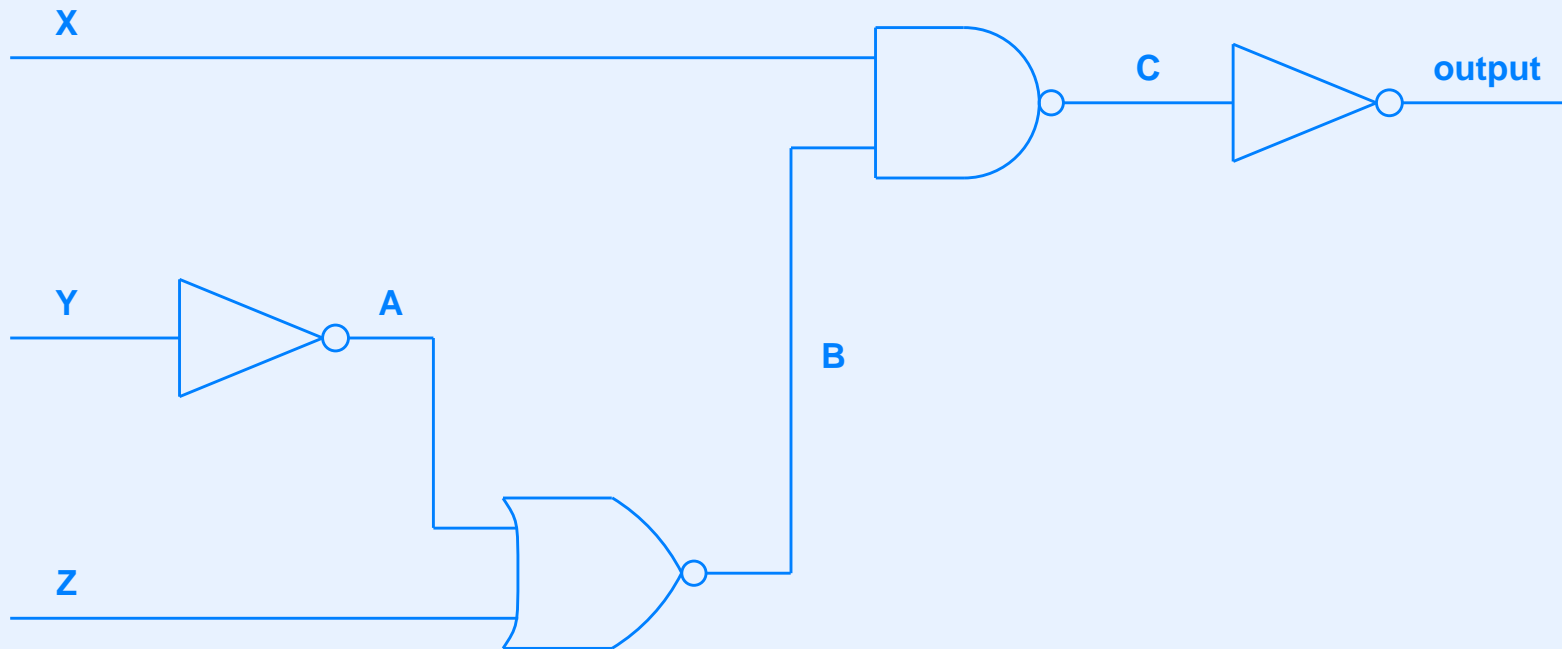
- Most popular technology known as *Transistor-Transistor Logic (TTL)*
- Allows direct interconnection (a wire can connect output from one gate to input of another)
- Single output can connect to multiple inputs
 - Called *fanout*
 - Limited to a small number

Example Interconnection Of TTL Gates

- Suppose we need a signal to indicate that the power button is depressed and the disk is ready
- Two logic gates are needed to form logical *and*
 - Output from nand gate connected to input of inverter



Consider The Following Circuit

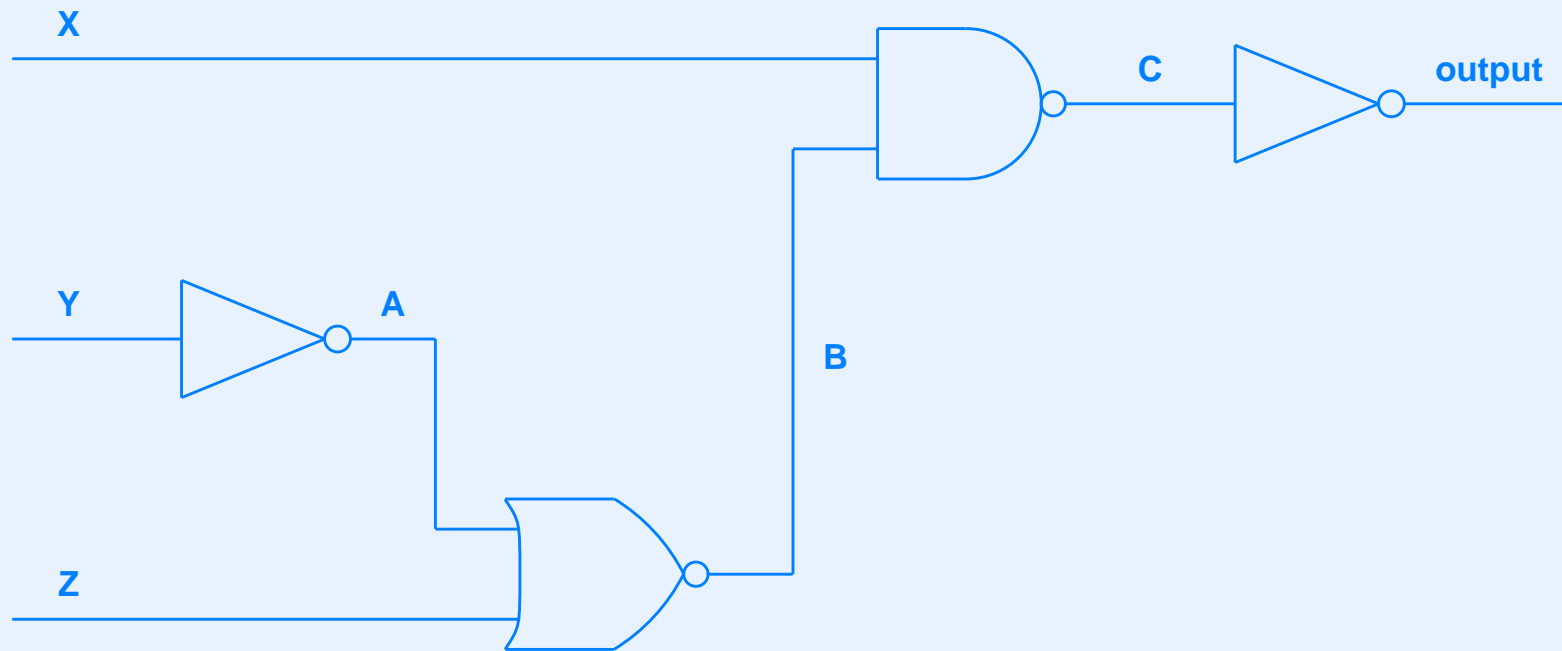


- Question: what does the circuit implement?

Two Ways To Describe A Circuit

- Boolean expression
 - Often used when designing circuit
 - Can be transformed to equivalent version that takes fewer gates
- Truth table
 - Enumerates inputs and outputs
 - Often used when debugging a circuit

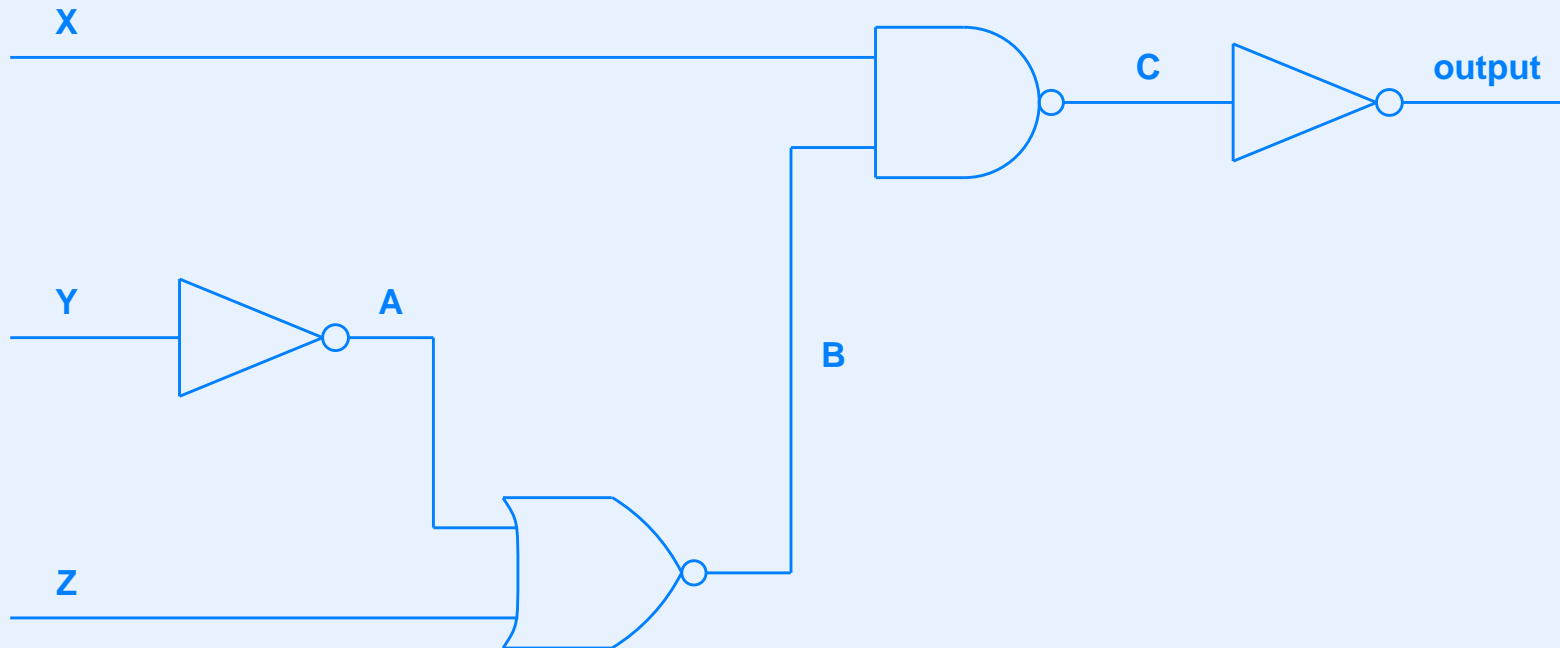
Describing A Circuit With Boolean Algebra



- Value at point *A* is *not Y*
- Value at *B* is:

$$Z \text{ nor } (\text{not } Y)$$

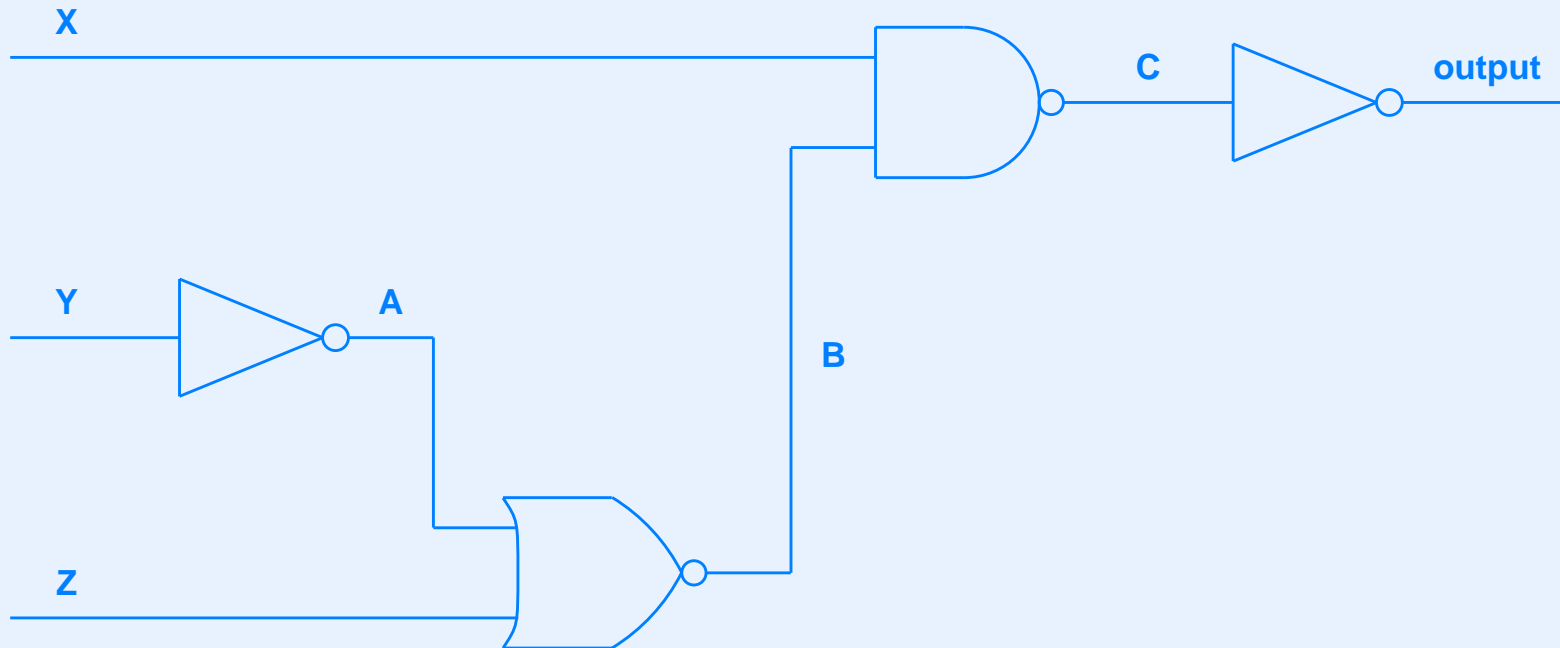
Describing A Circuit With Boolean Algebra (continued)



- Output is:

X and (Z nor (not Y))

Describing A Circuit With Boolean Algebra (continued)



- Output is (alternative):

X and not (Z or (not Y))

Describing A Circuit With A Truth Table (continued)

X	Y	Z	A	B	C	output
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	1	0
0	1	1	0	0	1	0
1	0	0	1	0	1	0
1	0	1	1	0	1	0
1	1	0	0	1	0	1
1	1	1	0	0	1	0

- Table lists all possible inputs and output for each
- Can also state values for intermediate points

Avoiding Nand / Nor Operations

- Circuits use nand and nor gates
- Sometimes easier for humans to use *and* and *or* operations
- Example circuit or truth table output can be described by Boolean expression:

X and Y and (not Z)

Binary Addition

- How does a computer perform addition?
- Analogous to the method used in elementary school

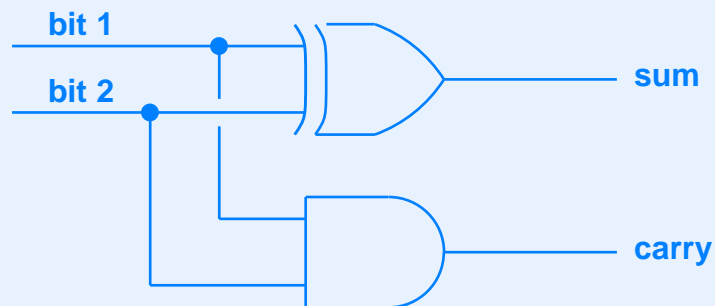
A binary addition diagram illustrating the process of adding two 6-bit numbers. The first number is 10100 and the second is 11101. The sum is 110001. Blue arrows labeled 'carry' show the propagation of carry bits from right to left: from the 2nd bit to the 3rd, from the 3rd to the 4th, and from the 4th to the 5th. A '+' sign is placed to the left of the second number. A horizontal line separates the addends from the sum.

			1	0	1	0	0
+		1	1	1	0	1	
<hr/>							
	1	1	0	0	0	1	

- Note: first bit has no carry input

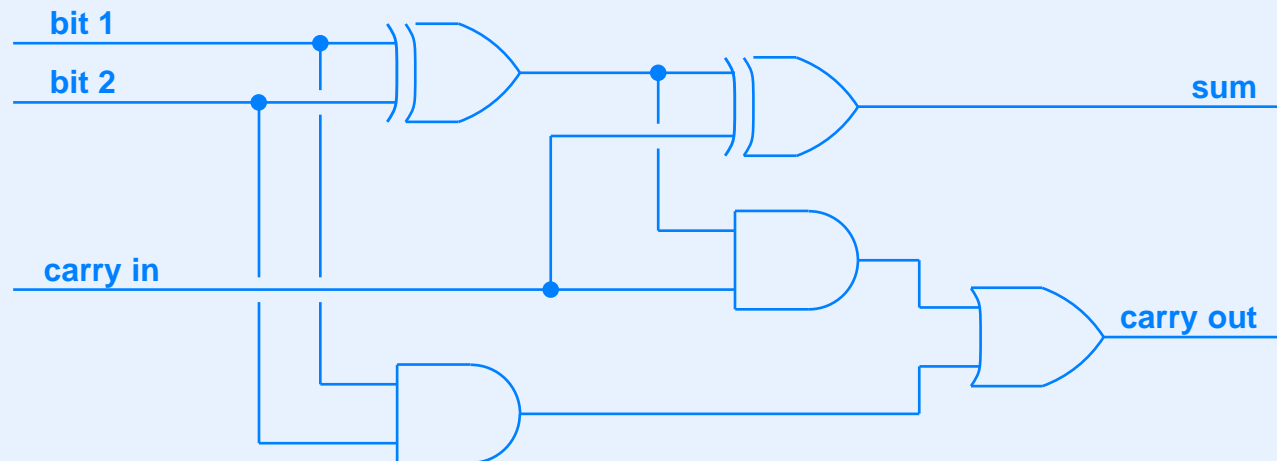
Half-Adder Circuit

- Adds two input bits
- Produces two output bits
 - Sum
 - Carry
- Uses *exclusive or* gate



Full-Adder Circuit

- Input is two bits plus a carry
- Produces two output bits
 - Sum
 - Carry



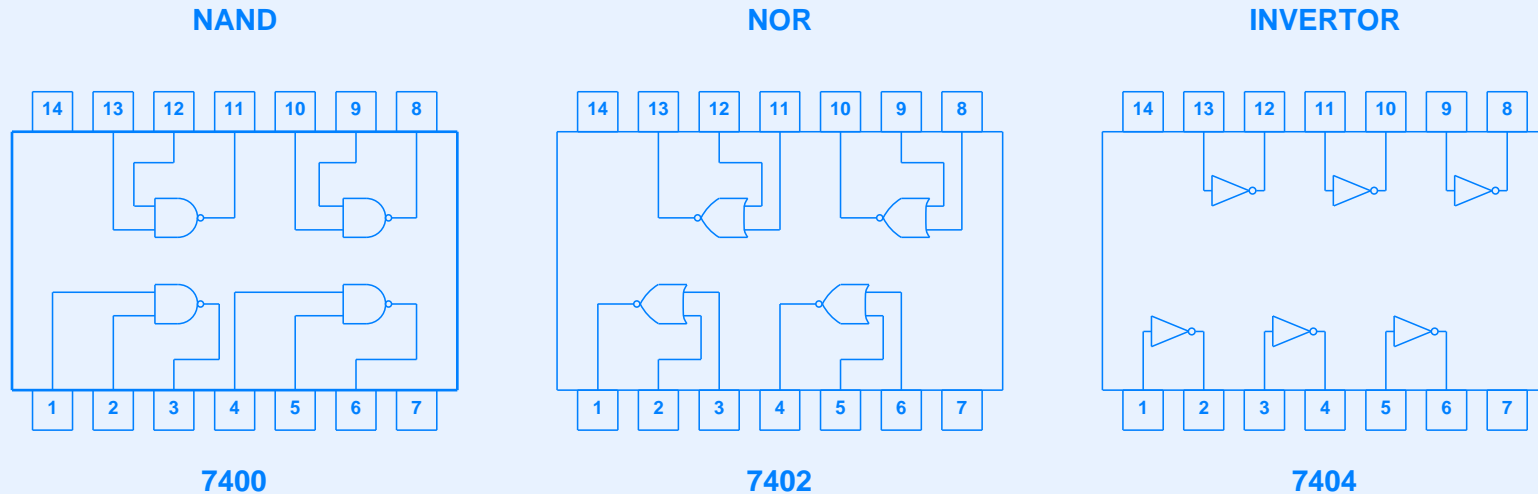
In Practice

- Given gate only has a few connections
- A chip has many pins for external connections
- Result: package multiple gates on each chip
- We will see examples shortly

An Example Logic Gate Technology

- 7400 family of chips
- Package is about one-half inch long
- Implement TTL logic
- Powered by five volts
- Each chip contains multiple gates

Example Gates On 7400-Series Chips



- Pins 7 and 14 connect to ground and power
- Power and ground *must* be connected for the chip to operate

Logic Gates And Computers

Logic Gates And Computers

- Question: how can computers be constructed from simple logic gates?

Logic Gates And Computers

- Question: how can computers be constructed from simple logic gates?
- Answer: they cannot
 - Logic gates only provide a Boolean combination of inputs

Logic Gates And Computers

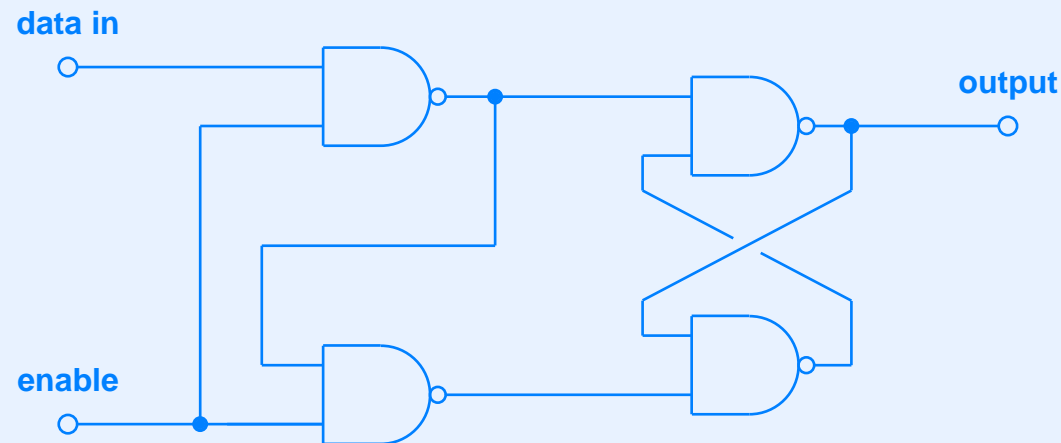
- Question: how can computers be constructed from simple logic gates?
- Answer: they cannot
 - Logic gates only provide a Boolean combination of inputs
- Additional functionality is needed
 - Circuits that maintain state
 - Circuits that operate on a clock

Circuits That Maintain State

- More sophisticated than *combinatorial circuits*
- Output depends on history of previous input as well as values on input lines

Basic Circuit That Maintains State

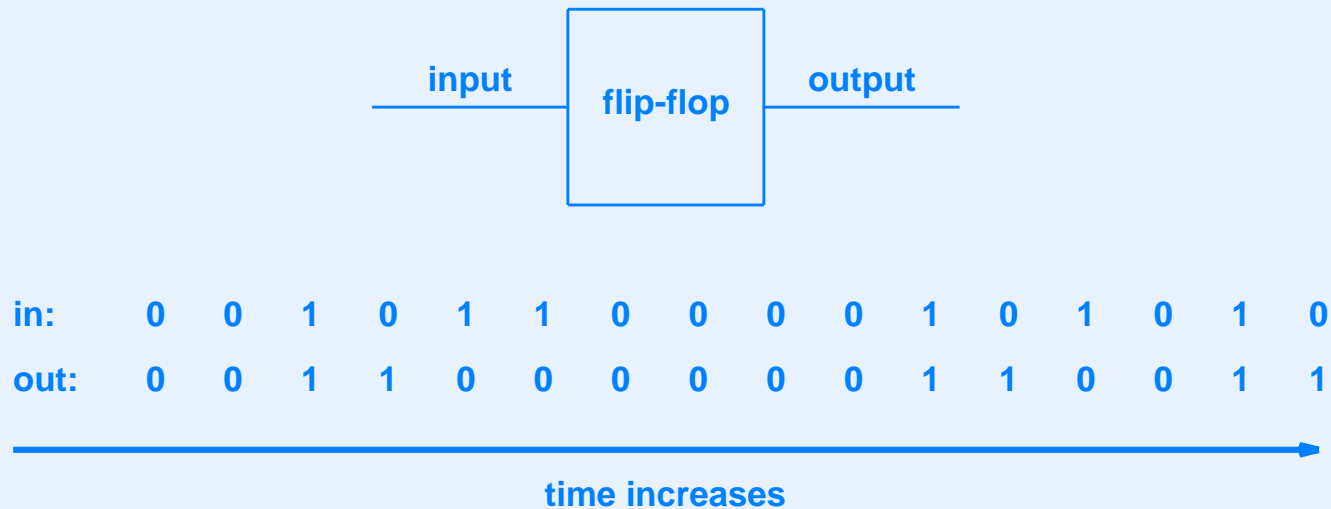
- Known as *latch*
- Has two inputs: *data* and *enable*
- When enable is 1, output is same as data
- When enable goes to 0, output stays locked at current value



A More Complex Circuit That Maintains State

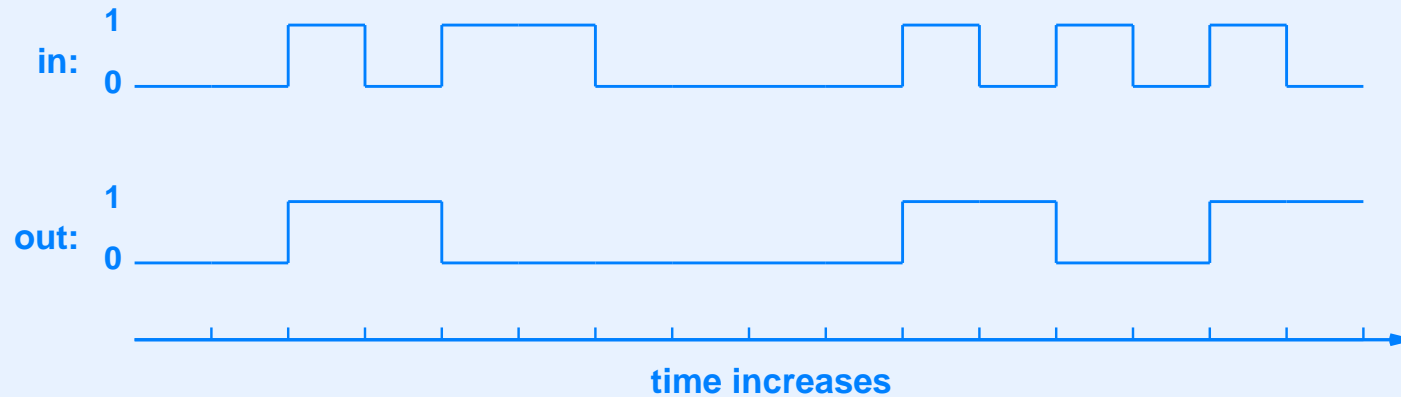
- Basic *flip-flop*
- Can be constructed from a pair of latches
- Analogous to push-button power switch (i.e., push-on push-off)
- Each new 1 received as input causes output to reverse
 - First input pulse causes flip-flop to turn on
 - Second input pulse causes flip-flop to turn off

Output Of A Flip-Flop



- Note: output only changes when input makes a transition from zero to one

Flip-Flop Action Plotted As Transition Diagram

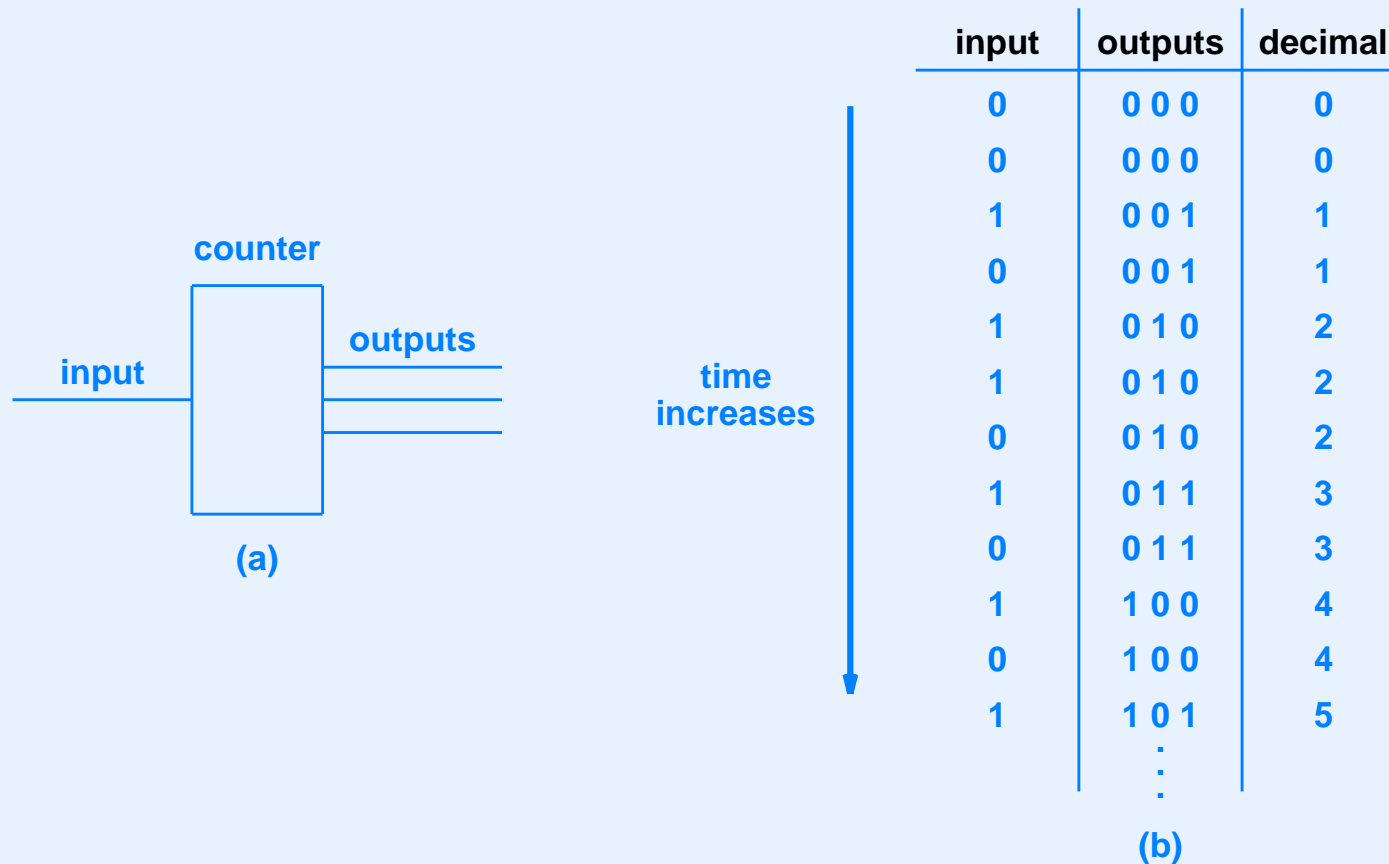


- Output changes on *leading edge* of input
- Also called *rising edge*

Binary Counter

- Counts input pulses
- Output is binary value
- Includes *reset line* to restart count at zero
- Example: 4-bit counter available as single integrated circuit

Illustration Of Counter



- Part (a) shows the schematic of a counter chip
- Part (b) shows the output as the input changes

Clock

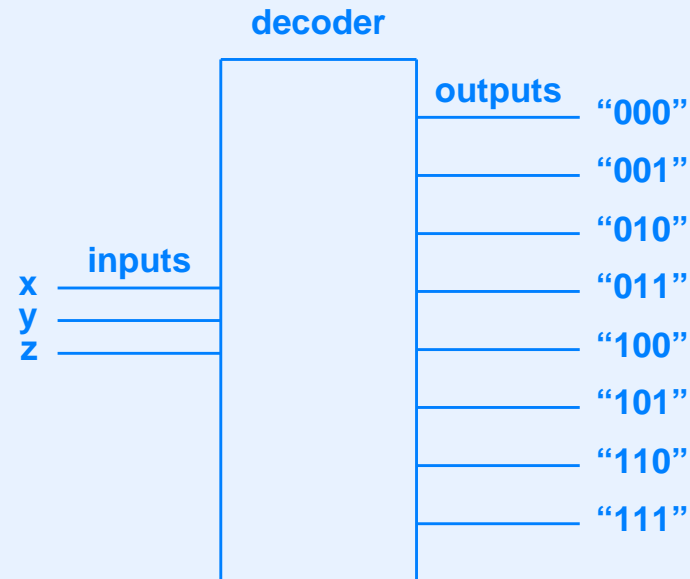
- Electronic circuit that pulses regularly
- Measured in cycles per second (Hz)
- Digital output of clock is sequence of 0 1 0 1 ...
- Permits active circuits

Decoder/Demultiplexor

- Takes binary number as input
- Uses input to select one output
- Technical distinction
 - *decoder* simply selects one output
 - *demultiplexor* feeds a special input to the selected output
- In practice: engineers often use the term “demux” for either, and blur the distinction

Illustration Of Decoder

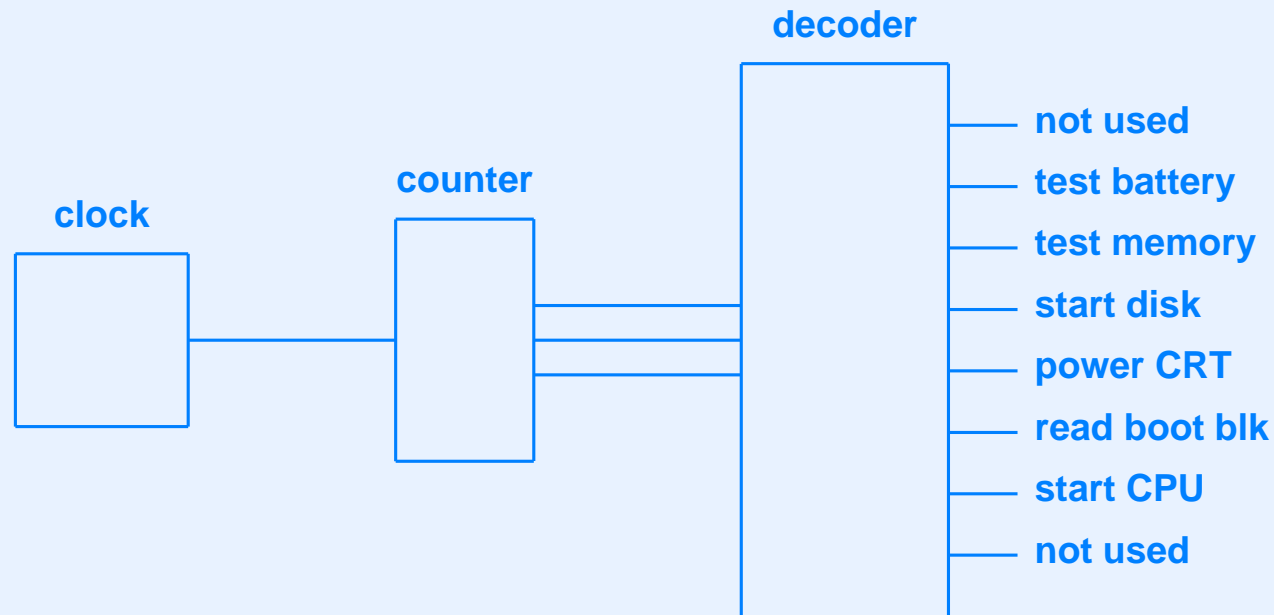
- Binary value on inputs determines which output is active



Example: Execute A Sequence Of Steps

- Desired sequence
 - Test the battery
 - Power on and test the memory
 - Start the disk spinning
 - Power up the display
 - Read boot sector from disk into memory
 - Start the CPU

Circuit To Execute A Sequence



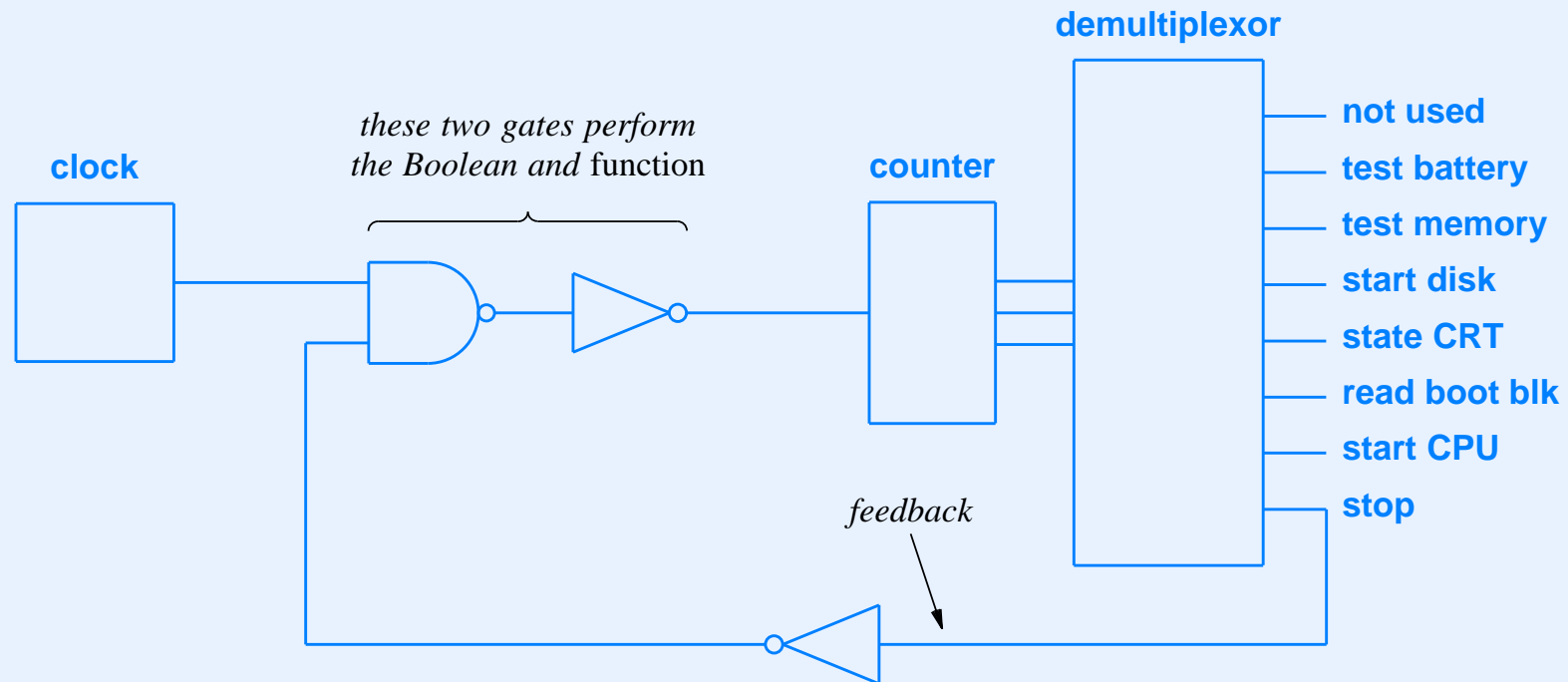
- Technique: count clock pulses and use decoder to select an output for each possible counter output
- Note: counter will wrap around to zero, so this is an infinite loop

Feedback

- Output of circuit used as an input
- Called *feedback*
- Allows more control
- Example: stop sequence when output F becomes active
- Boolean algebra

CLOCK *and* (*not* F)

Illustration Of Feedback For Termination



- Note additional input needed to restart sequence

Spare Gates

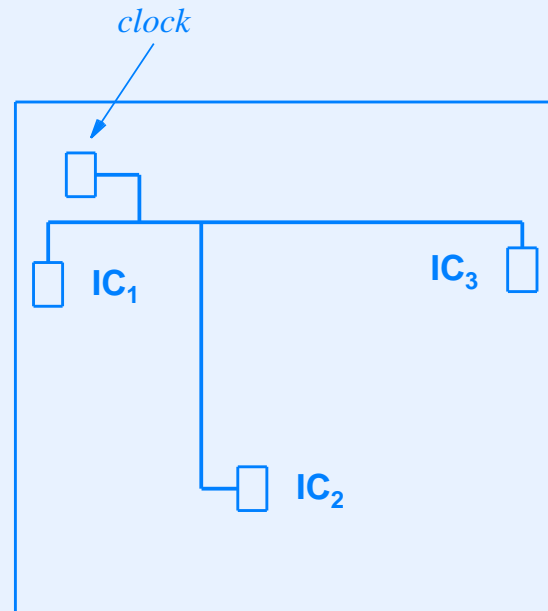
- Note: because chip contains multiple gates, some gates may be unused
- May be able to substitute spare gates in place of additional chip
- Example uses spare nand gate as inverter by connecting one input to five volts:

$$1 \text{ nand } x = \text{not } x$$

Practical Engineering Concerns

- Power consumption (wiring must carry sufficient power)
- Heat dissipation (chips must be kept cool)
- Timing (gates take time to settle after input changes)
- Clock synchronization (clock signal must reach all chips simultaneously)

Illustration Of Clock Skew



- Length of wire determines time required for signal to propagate

Classification Of Technologies

Name	Example Use
Small Scale Integration (SSI)	The most basic logic such as Boolean gates
Medium Scale Integration (MSI)	Intermediate logic such as counters
Large Scale Integration (LSI)	More complex logic such as embedded processors
Very Large Scale Integration (VLSI)	The most complex processors (i.e., CPUs)

Levels Of Abstraction

Abstraction	Implemented With
Computer	Circuit board(s)
Circuit board	Components such as processor and memory
Processor	VLSI chip
VLSI chip	Many gates
Gate	Many transistors
Transistor	Semiconductor implemented in silicon

Reconfigurable Logic

- Alternative to standard gates
- Allows chip to be configured multiple times
- Can create
 - Various gates
 - Interconnections
- Most popular form: *Field Programmable Gate Array (FPGA)*

Summary

- Computer systems are constructed of digital logic circuits
- Fundamental building block is called a *gate*
- Digital circuit can be described by
 - Boolean algebra (most useful when designing)
 - Truth table (most useful when debugging)
- Clock allows active circuit to perform sequence of operations
- Feedback allows output to control processing
- Practical engineering concerns include
 - Power consumption and heat dissipation
 - Clock skew and synchronization