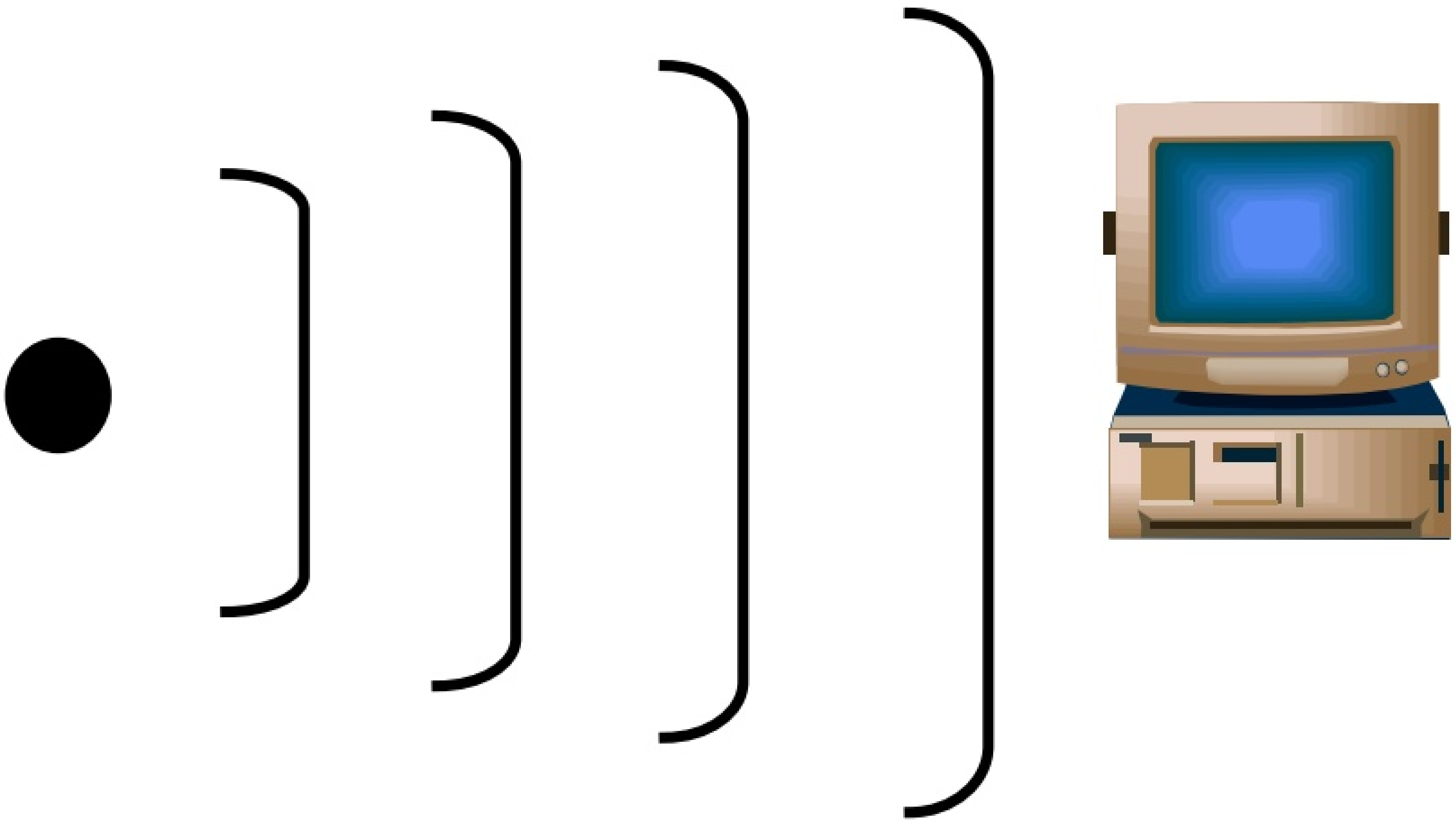
Building a computer



Why build a computer?

- Curiosity
  - What really happens when I hit a key?
- Necessity
  - Prerequisite to parts of the course
- Breadth
  - Should understand what's changing the world

Why NOT build a computer?

- Computers seem really complicated !
  - Pentium III has over 28 MILLION components
- How can we hope to understand them ?

Questions in building a house

- What are the basic components
  - 2”x4”’s, I-beams, plasterboard, ..
  - Light fixtures, plumbing, …
- What is glue for combining them
  - Nails, screws, bolts, pipes, …
- How do we model the process
  - Architectural drawings, building codes, …

Questions in building a computer

- What are the basic components
  - Logic gates
- What is glue for combining them
  - Wires to build circuits
- How do we model the process
  - Architectural drawings, design rules, …
  - Mathematical formulation

Details

- Wires are made of silicon or chemicals
- Crossing wires make transistors
- Electrons either do or don’t flow in wires
  - Think of light switches turning current on or off
- Wire thickness currently about .135 micron
  - 1 micron = 10^{-6} meters
  - Can fit 28 million transistors in less than 1”x1”
- Design must follow fabrication rules
  - Non-crossing wires can’t get too close
- Mathematical abstraction – Boolean algebra

What is Abstraction?

- Ignoring / Hiding details to capture essential common features
- Example for us: We'll ignore whether we're talking about:
    - A Pentium II or a Pentium III
    - A Mac or a PC
- Instead, we'll focus on what **really** makes a computer a computer.

Abstraction (cont.)

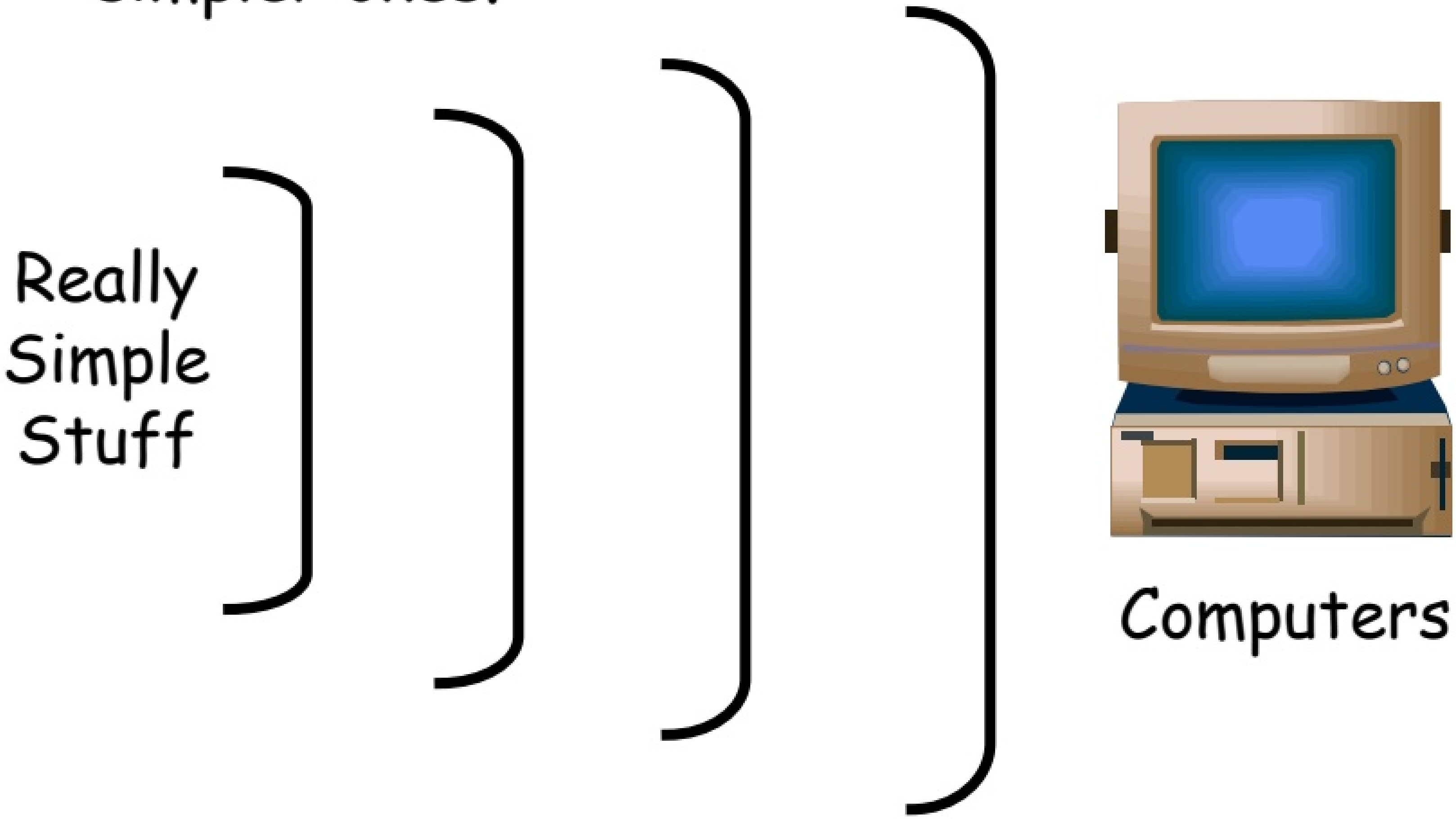
- Real Life Example: "Brush your teeth"
- Here, we ignore:
  - What kind of toothbrush
  - What kind of toothpaste
  - What you're wearing
  - etc.
- These things aren't important!
- Often called *hand waving*

Abstraction (cont.)

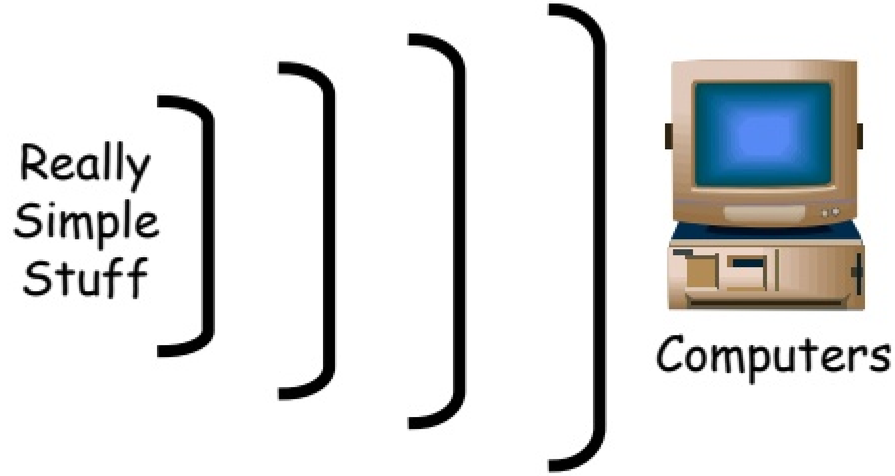
- Abstraction allows us to understand things in a Modular fashion.
- If we had to spell out everything we did, our lives would seem really complicated.
- Same is true for computers.
  To understand them, we use abstraction.
- Working bottom up.

Layers of Abstraction

- Build more and more powerful tools out of simpler ones.

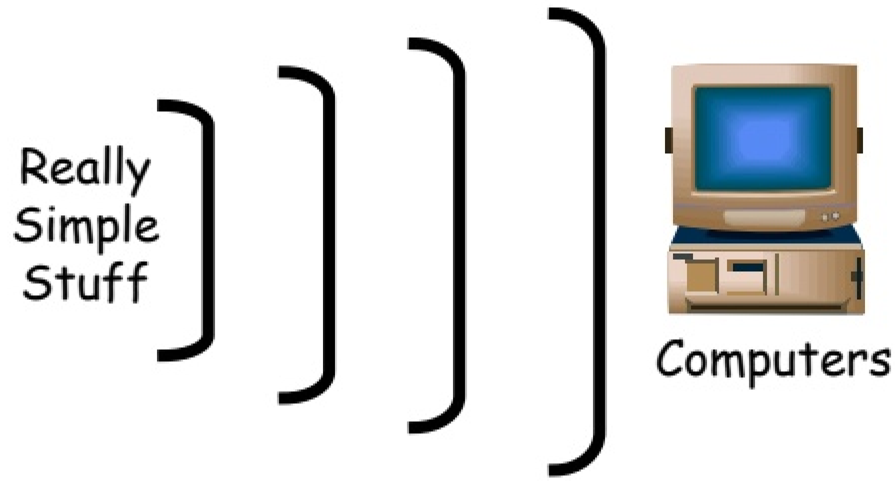


Layers of Abstraction (cont.)



- Each layer corresponds to a beautiful *Big Idea* of computer science.
- These ideas are the foundation for the "digital revolution" that everyone talks about.

Layers of Abstraction (cont.)



- For Computers, What is the "Really Simple Stuff"?
- Answer: " 0's and 1's "

The secret lives of 0's and 1's

A Simple Logic Puzzle

- Ram will go to the party if
  Ramesh goes AND Mukesh does NOT.
- Mukesh will go if Ravi does NOT go OR if Vikas goes.
- Ramesh will go to the party if Alice AND Ravi go.

- Alice and Ravi decide to go,
  but Vikas stays home.
- Will Ram go to the party?

A Simple Logic Puzzle

- Ram will go to the party if
  Ramesh goes AND Mukesh does NOT.
- Mukesh will go if Ravi does NOT go OR if Vikas goes.
- Ramesh will go to the party if Alice AND Ravi go.

- Alice and Ravi decide to go,
  but Vikas stays home.
- Will Ram go to the party?
- Answer: YES

Using 0's and 1's

- What do 0's and 1's mean?
- For now, we'll take "Natural meanings:"

0 = False	1 = True

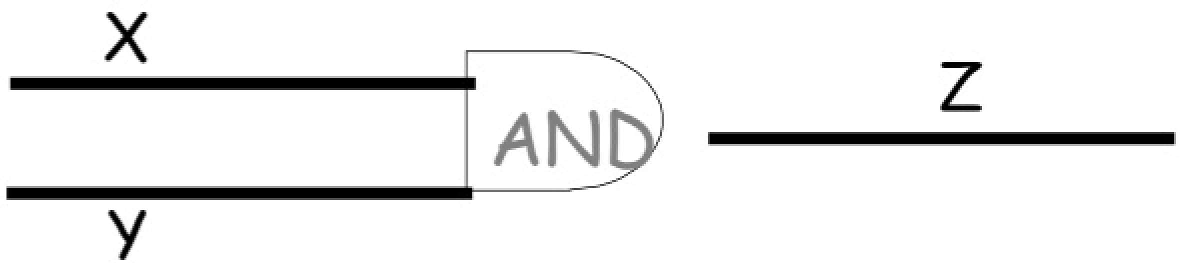
- For example, if we have a variable *Alice* for whether Alice goes to the party,
  - If Alice goes, we write *Alice* = 1
  - If Alice doesn't, we write *Alice* = 0

Logic Gates

- Computers are circuits made of *Logic Gates*.
- Logic gates manipulate 0's and 1's (False and True) by letting electrons flow or not.
- We'll look at three types of Logic Gates:
  - **AND** **are all inputs true?**
  - **OR** **is one input true?**
  - **NOT** **flip the truth value**

AND Gate

- "Zac will go to the party if Xena AND Yanni go."

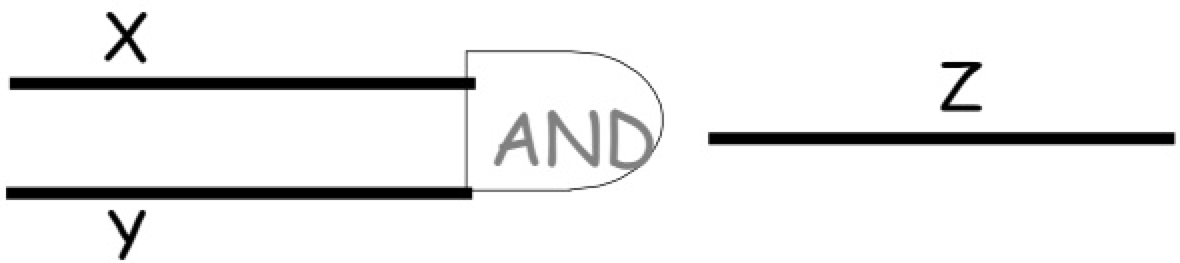


X	Y	Z
F	F	F
F	T	F
T	F	F
T	T	T

"Truth Table"

AND Gate (cont.)

- "Zac will go to the party if Xena AND Yanni go."



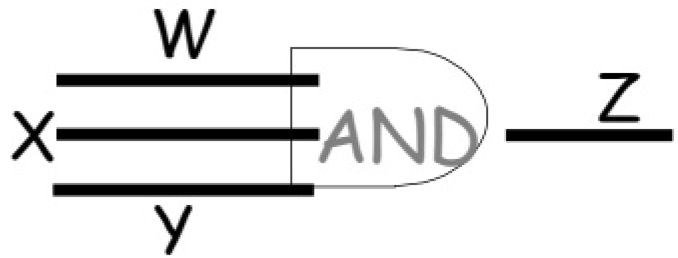
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table

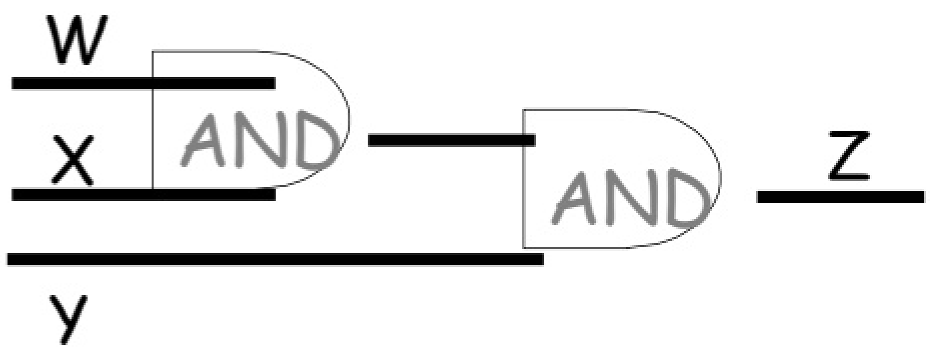
AND Gate (cont.)

- AND *Gate* is a circuit that allows output current to flow if both inputs are flowing.

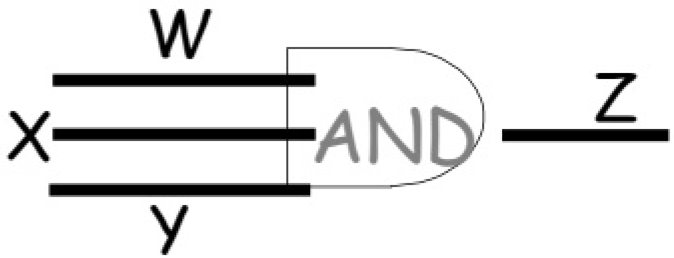
AND Gate (cont.)



is shorthand for:

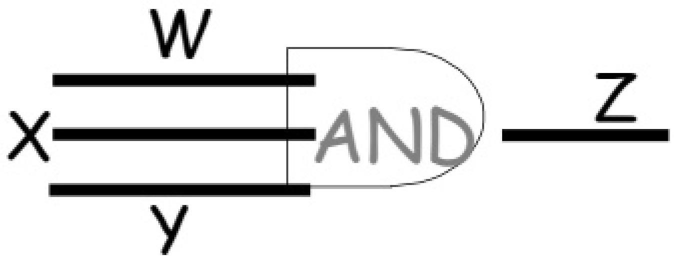


AND Gate (cont.)



W	X	Y	Z
0	0	0	?
0	0	1	?
0	1	0	?
0	1	1	?
1	0	0	?
1	0	1	?
1	1	0	?
1	1	1	?

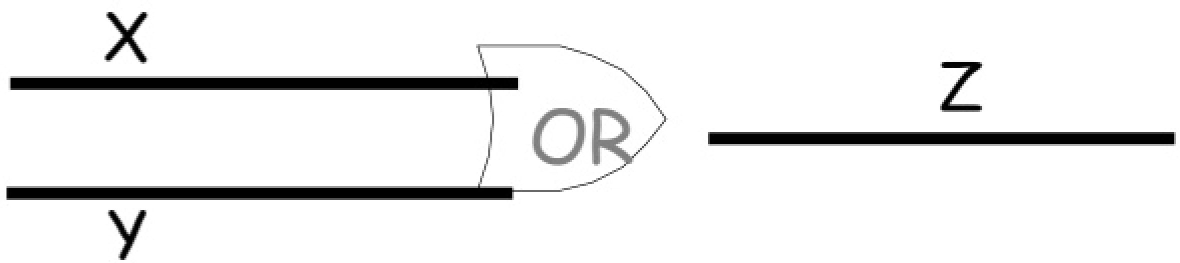
AND Gate (cont.)



w	x	y	z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

OR Gate

- "Zac will go to the party if Xena OR Yanni go."

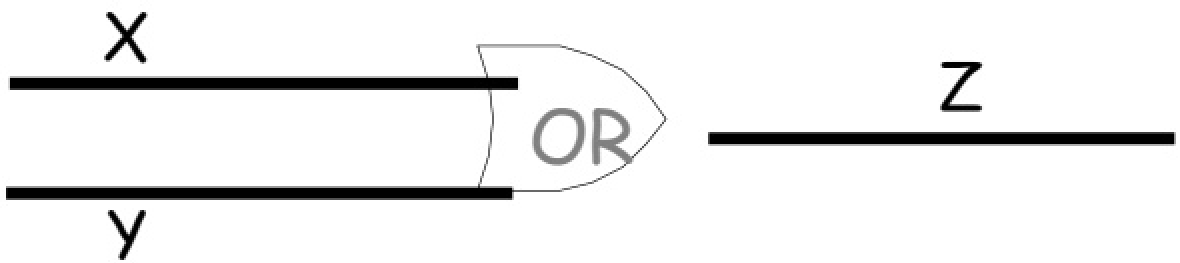


X	Y	Z
F	F	F
F	T	T
T	F	T
T	T	T

Truth Table

OR Gate (cont.)

- "Zac will go to the party if Xena OR Yanni go."



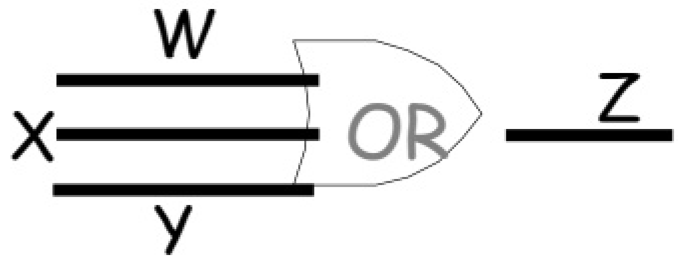
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

Truth Table

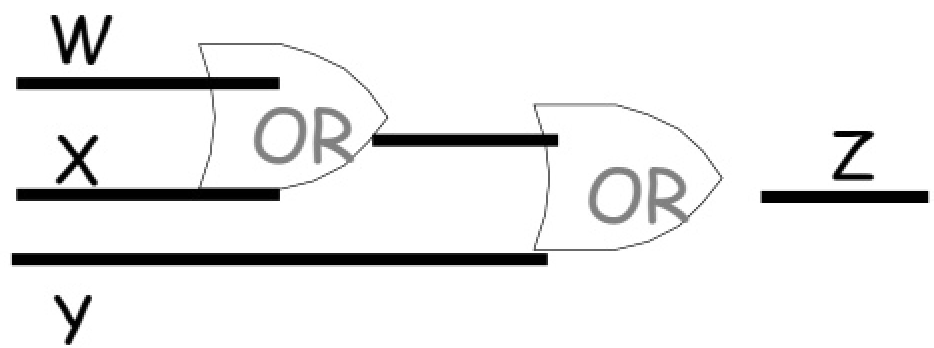
OR Gate (cont.)

- OR Gate is a circuit that allows output current to flow if either input is flowing.

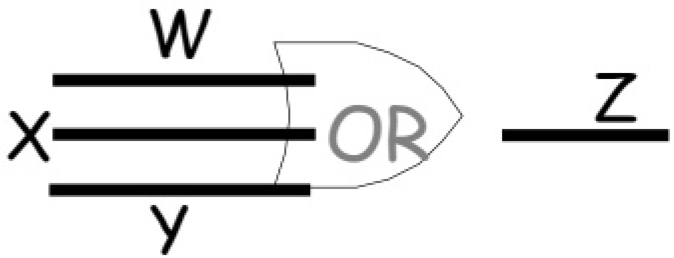
OR Gate (cont.)



is shorthand for:

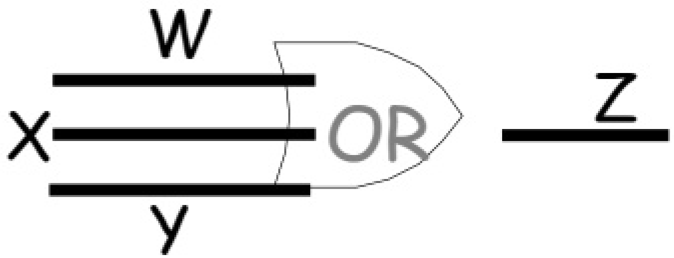


OR Gate (cont.)



W	X	Y	Z
0	0	0	?
0	0	1	?
0	1	0	?
0	1	1	?
1	0	0	?
1	0	1	?
1	1	0	?
1	1	1	?

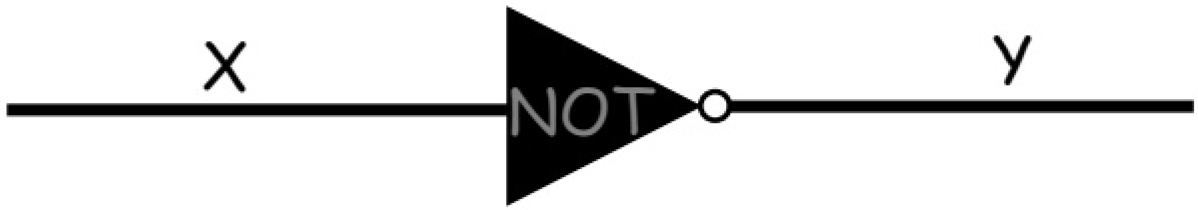
OR Gate (cont.)



w	x	y	z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

NOT Gate

- "Yanni will go to the party if Xena does NOT go."



Shorthand:

x	y
0	1
1	0

Truth Table

NOT Gate (cont.)

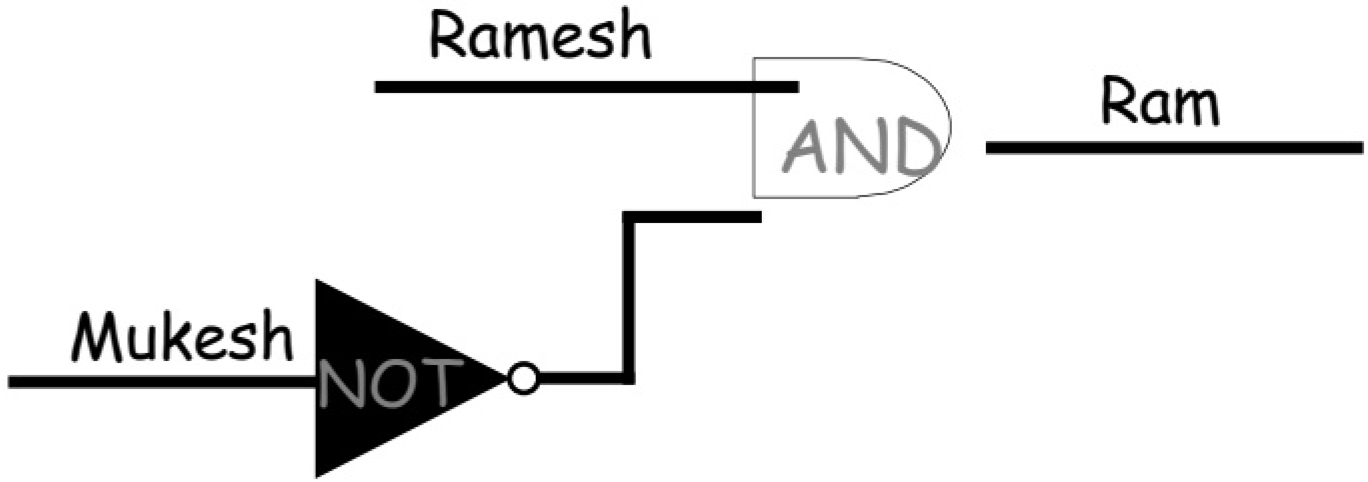
- NOT Gate is a circuit that reverse the sense of a flow.
- Logical complement.

A Simple Logic Puzzle

- Ram will go to the party if
  Ramesh goes AND Mukesh does NOT.
- Mukesh will go if Ravi does NOT go OR if Vikas goes.
- Ramesh will go to the party if Alice AND Ravi go.

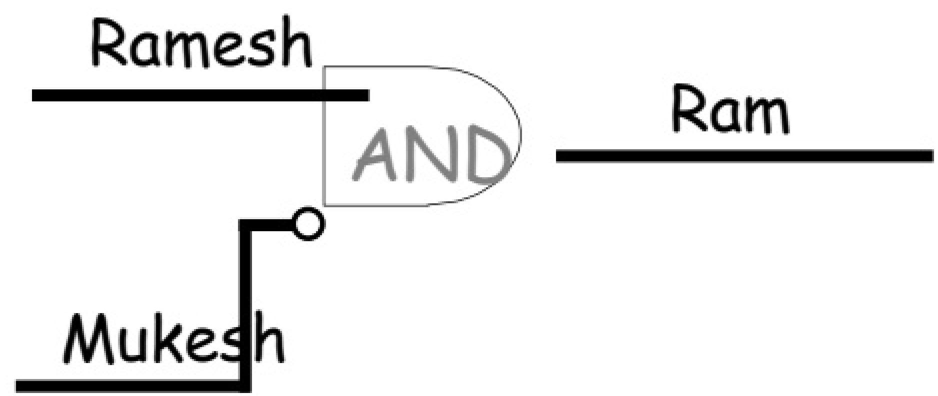
- Alice and Ravi decide to go,
  but Vikas stays home.
- Will Ram go to the party?

Logic Puzzle Circuit



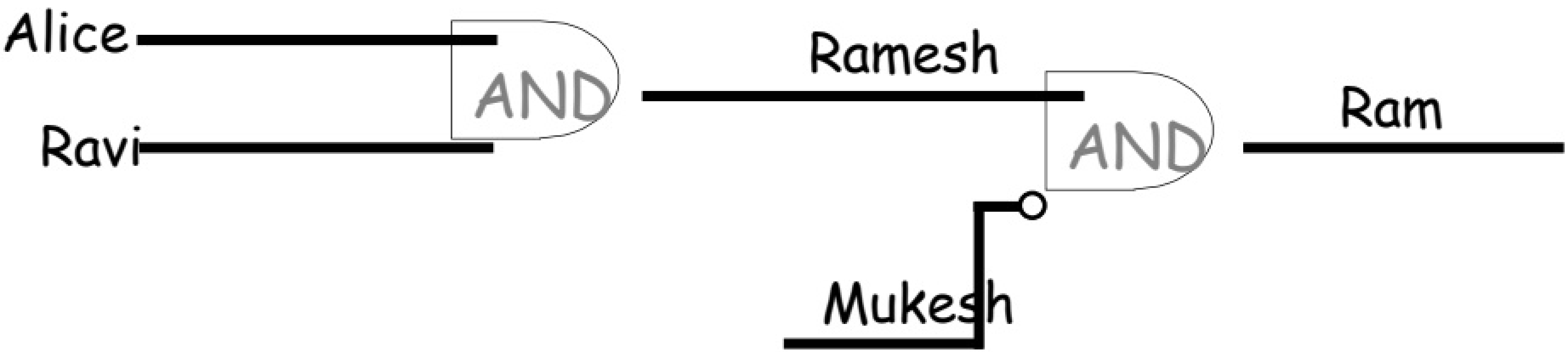
Ram will go to the party if
Ramesh goes AND Mukesh does NOT.

Logic Puzzle Circuit (cont.)



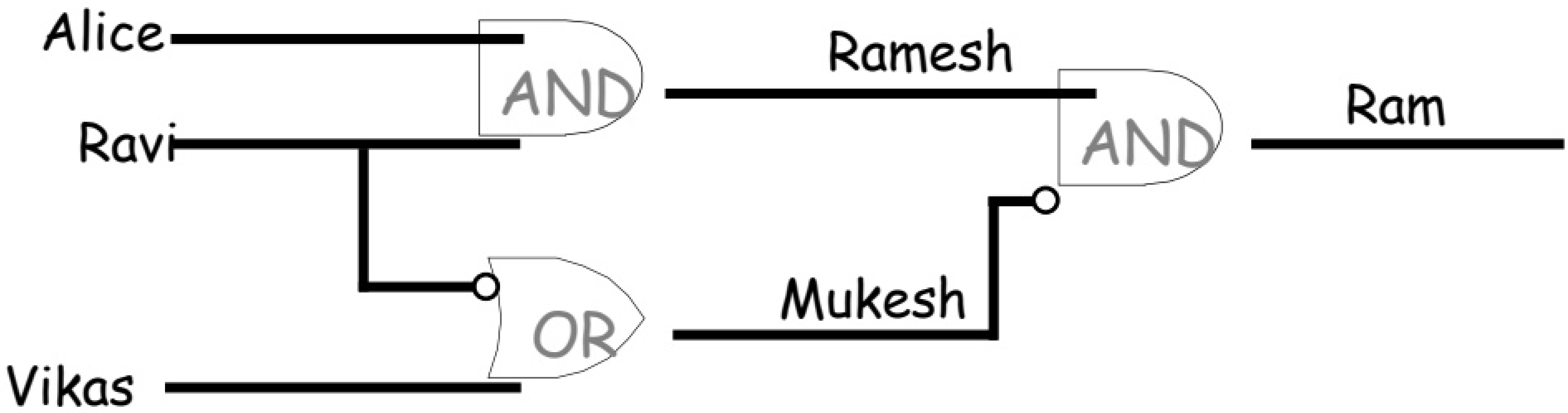
Ram will go to the party if
Ramesh goes AND Mukesh does NOT.

Logic Puzzle Circuit (cont.)



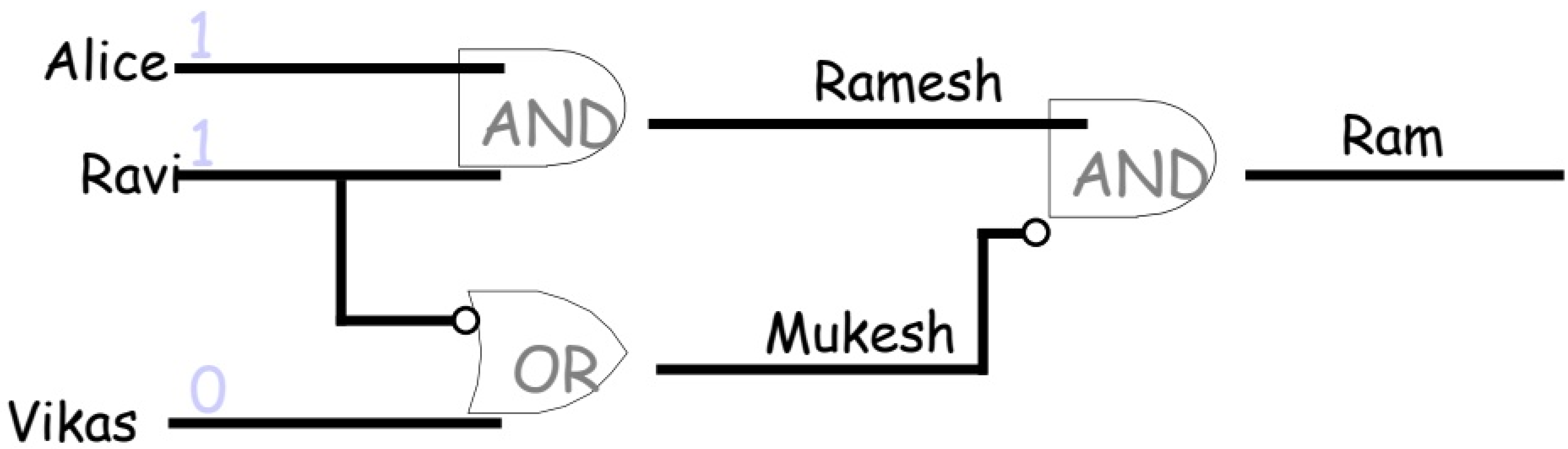
Ramesh will go to the party if Alice AND Ravi go.

Logic Puzzle Circuit (cont.)



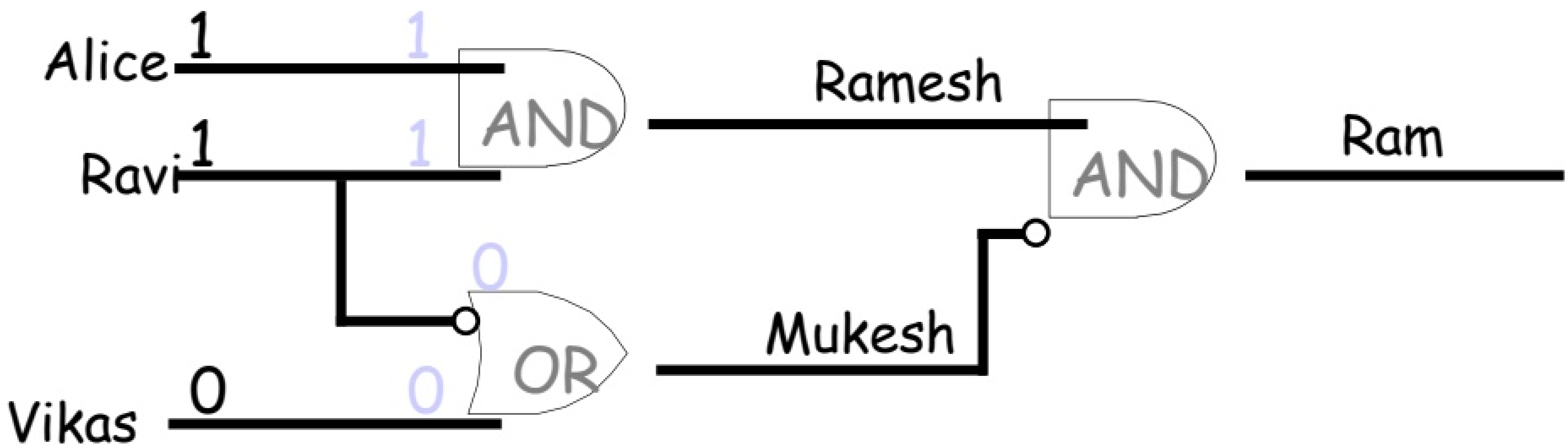
Mukesh will go if Ravi does NOT go OR if Vikas goes.

Logic Puzzle Circuit (cont.)



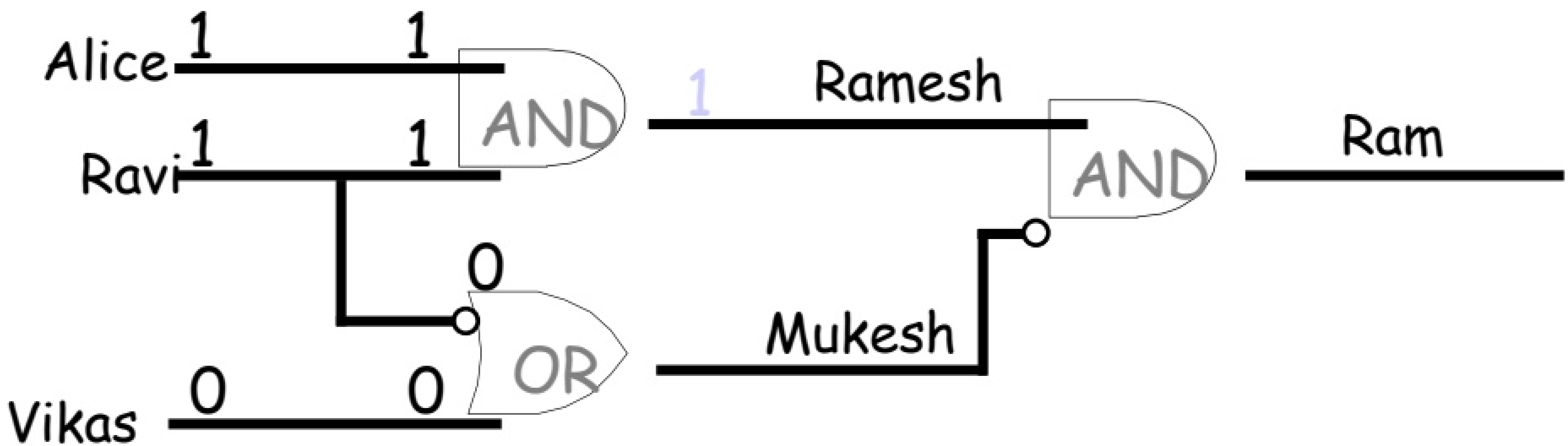
Alice and Ravi decide to go,
but Vikas stays home.

Logic Puzzle Circuit (cont.)



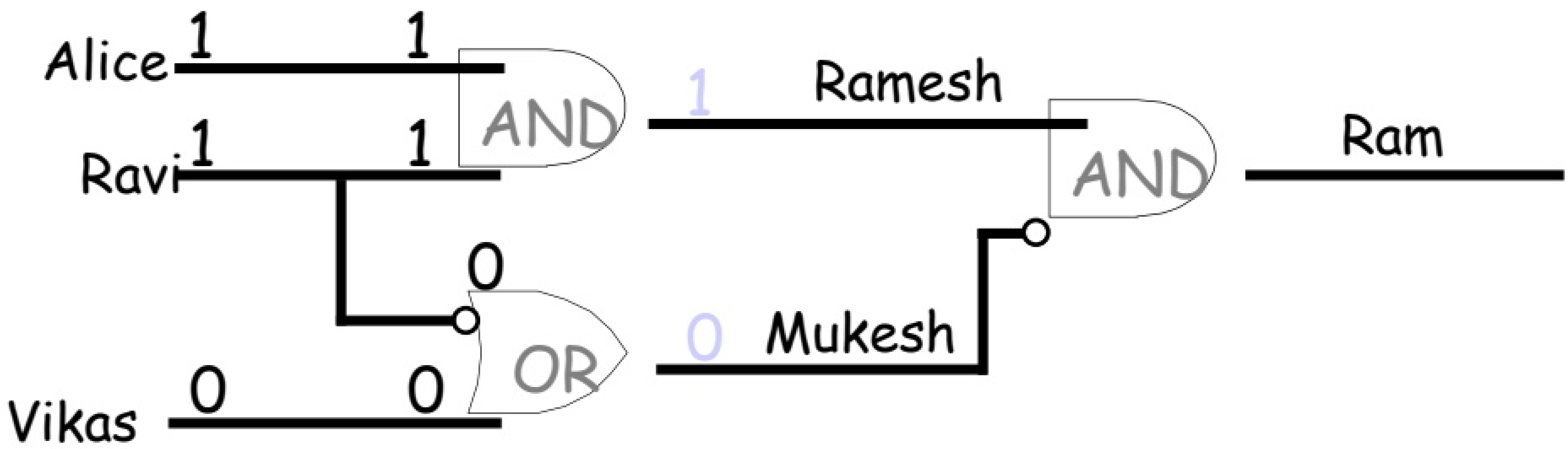
Evaluation…

Logic Puzzle Circuit (cont.)



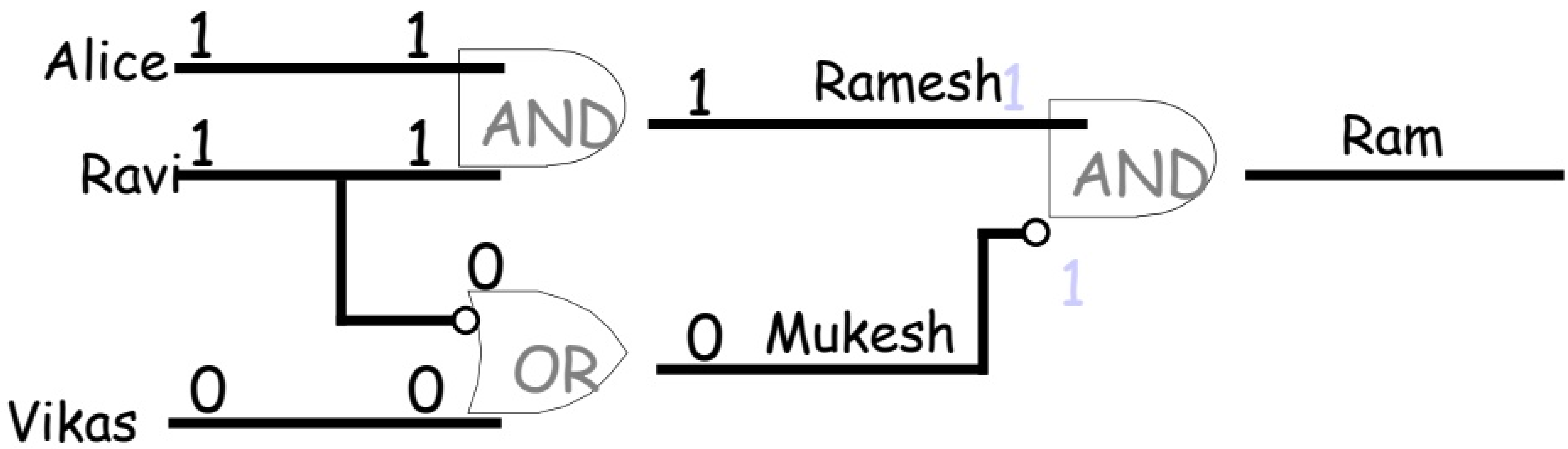
Evaluation…

Logic Puzzle Circuit (cont.)



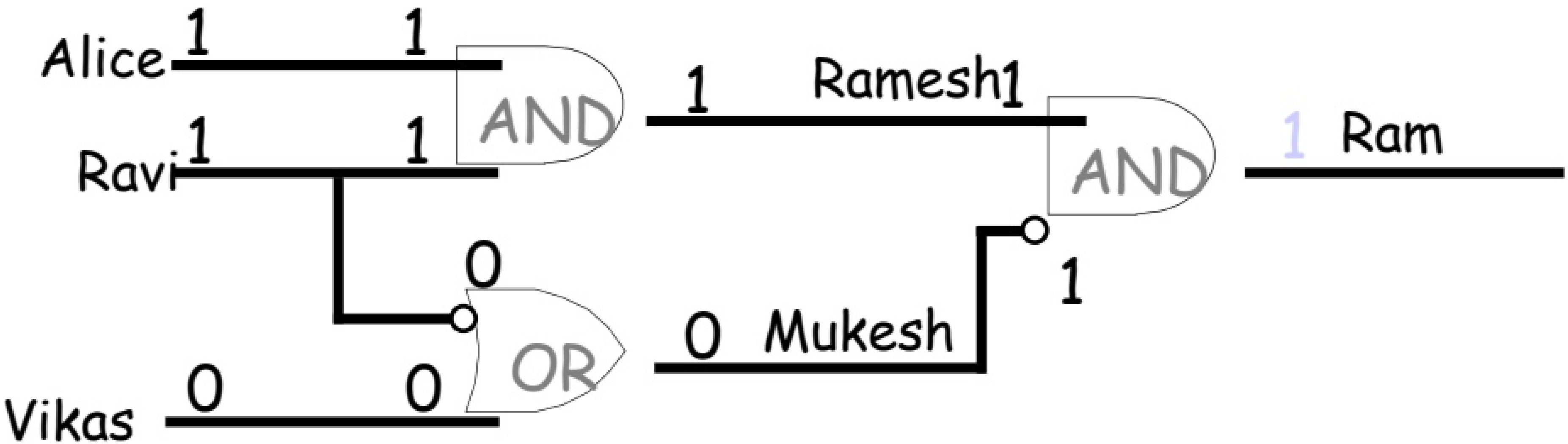
Evaluation...

Logic Puzzle Circuit (cont.)



Evaluation…

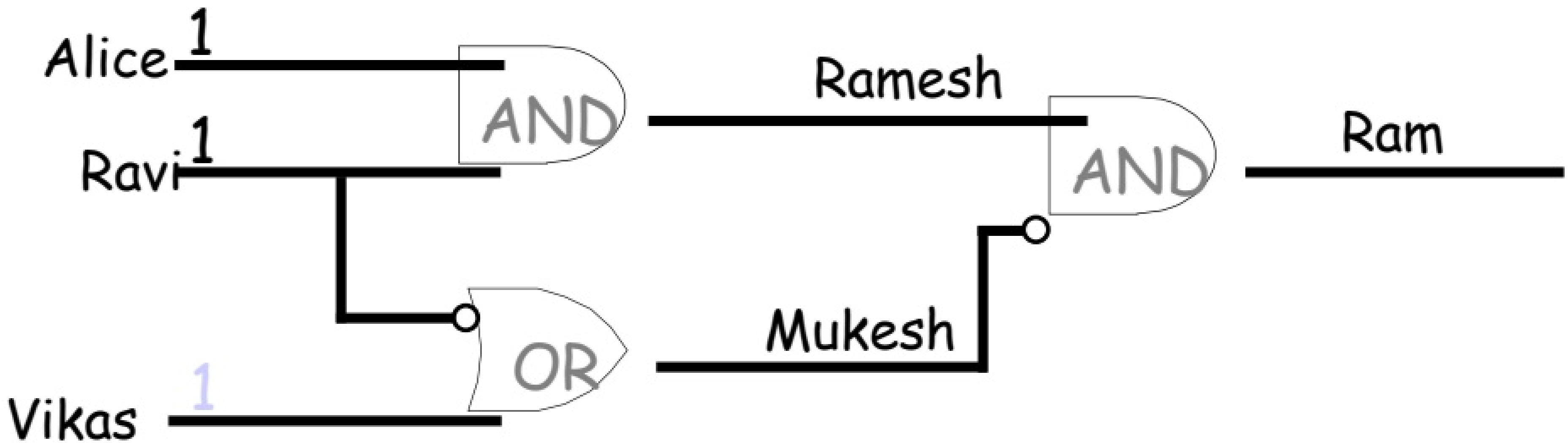
Logic Puzzle Circuit (cont.)



Evaluation Complete!

Answer: Ram goes to the party.

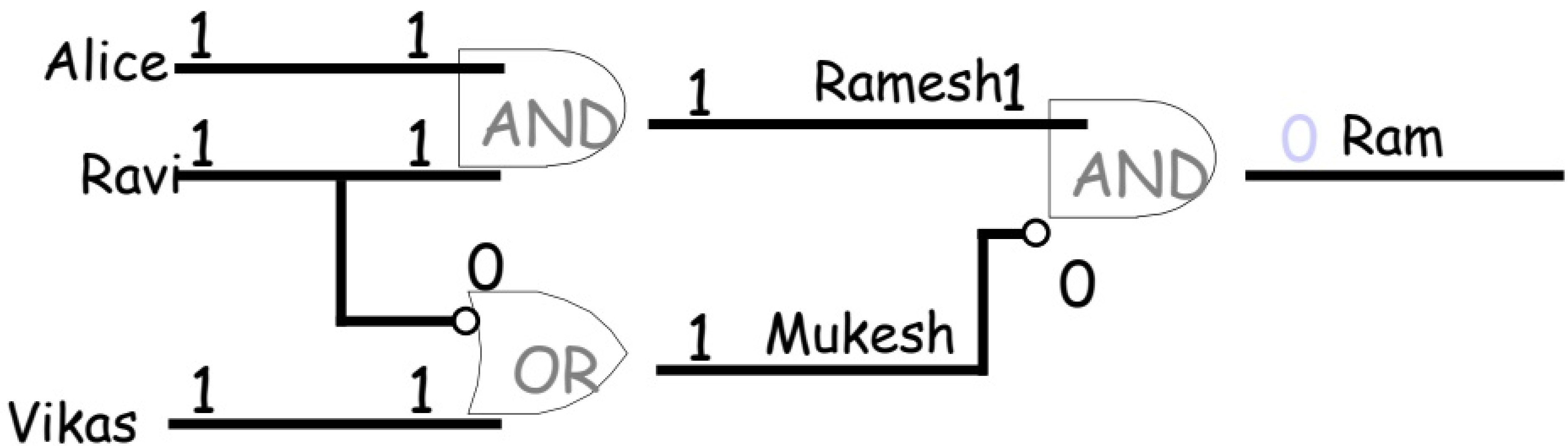
Logic Puzzle Circuit (cont.)



What if:

Alice, Ravi, and Vikas all go to the party?

Logic Puzzle Circuit (cont.)



What if:

Alice, Ravi, and Vikas all go to the party?

Answer: Ram does NOT go to the party!

- Is it all clear?
- Should/Could we do another such problem?
  - Light controllers
    - Light fixture has 3 switches
    - Light is on if an odd number of the switches are on

Building Circuits

- Suppose someone gives us an arbitrary Truth Table.

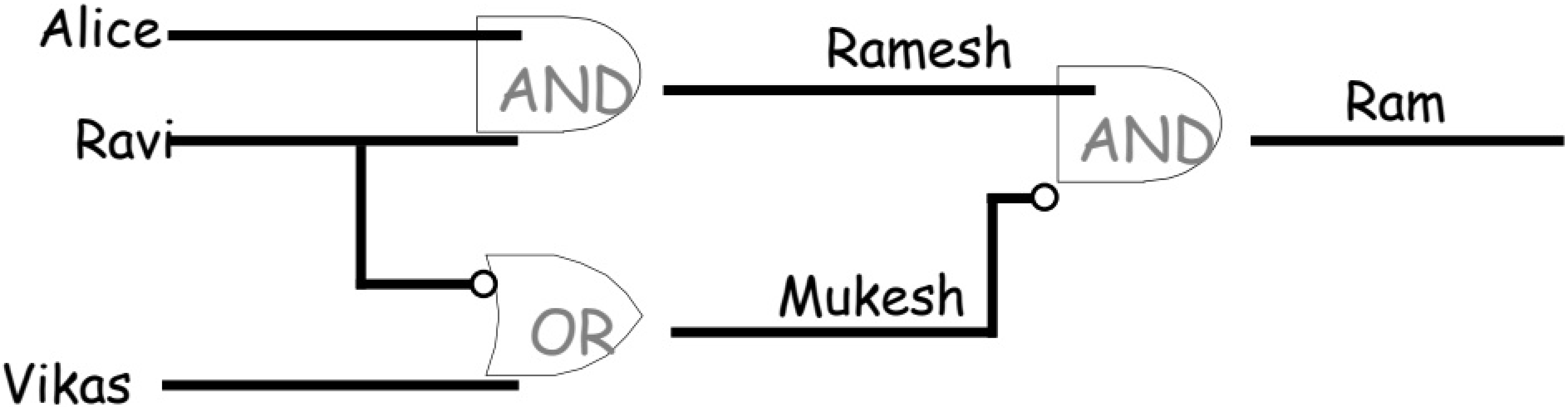
- Can we build a circuit which satisfies exactly that Truth Table?

Our Logic Puzzle

- Ram will go to the party if
  Ramesh goes AND Mukesh does NOT.
- Mukesh will go if Ravi does NOT go OR if Vikas goes.
- Ramesh will go to the party if Alice AND Ravi go.

- Suppose we made the truth table for this puzzle.

Logic Puzzle Circuit



The full circuit for the Logic Puzzle.

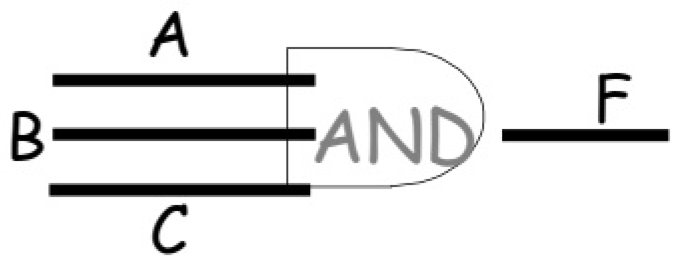
Logic Puzzle Circuit (cont.)

Alice	Ravi	Vikas	Ram
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Note: Ram goes only if A=B=1 and C=0

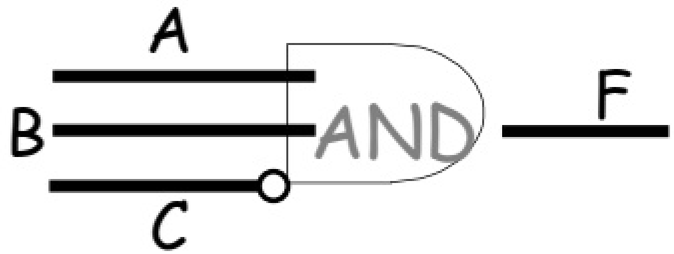
Truth Table for Logic Puzzle

Recall: AND Gate



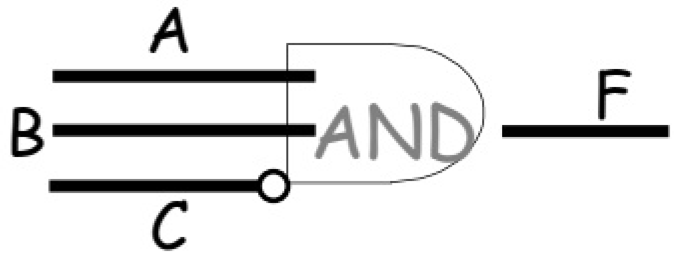
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Modified AND Gate



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

Modified AND Gate

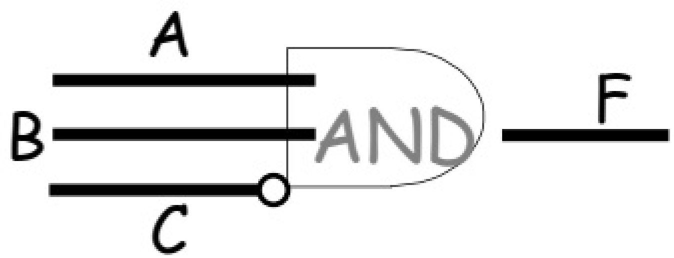


Note: Ram goes only if A=B=1 and C=0.

The modified AND also solves the Logic Puzzle!

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

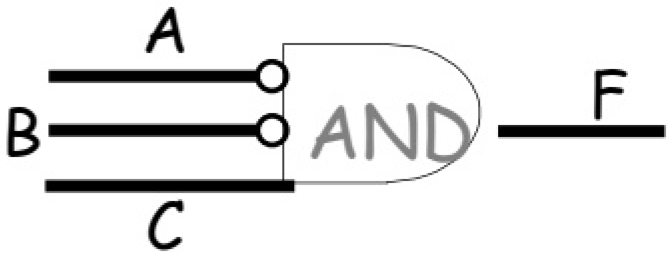
Modified AND Gate



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

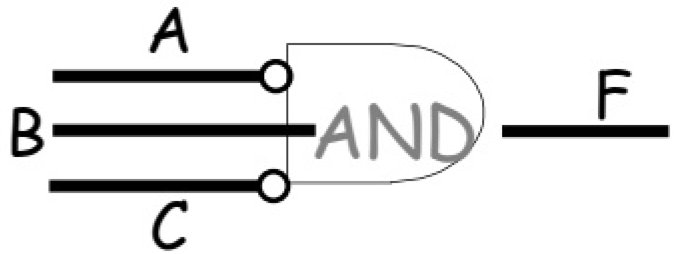
?

Sure …



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

Similarly...



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

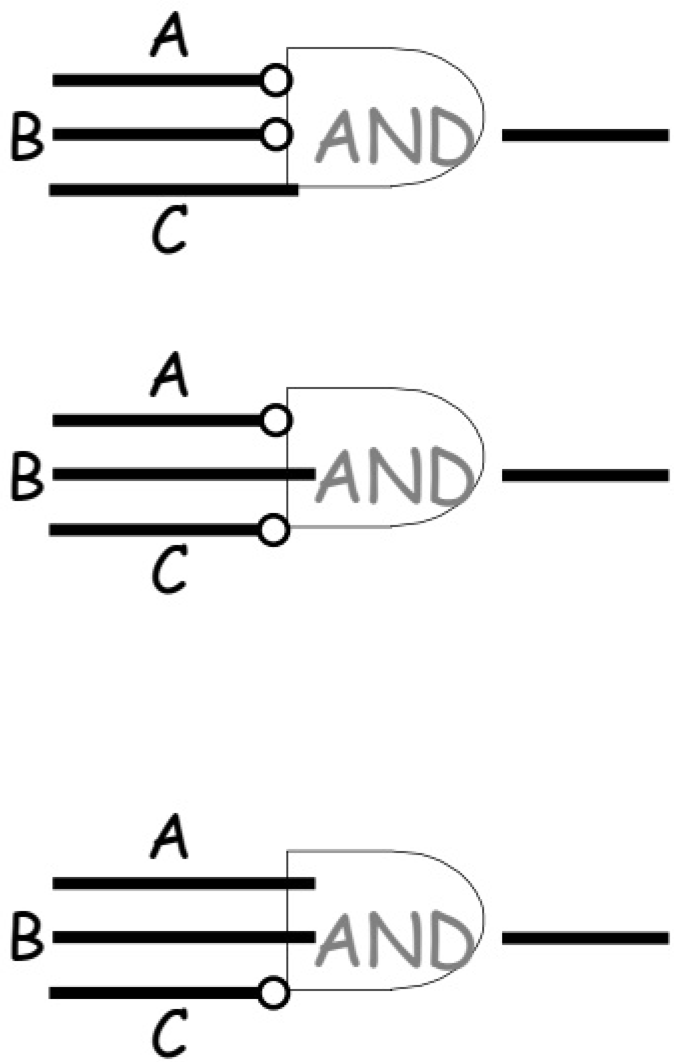
Similarly, we can make a circuit for any Truth Table with only a single 1 in its output.

Given Any Truth Table…

First, make circuits for each row in Truth Table with a 1 in the output.

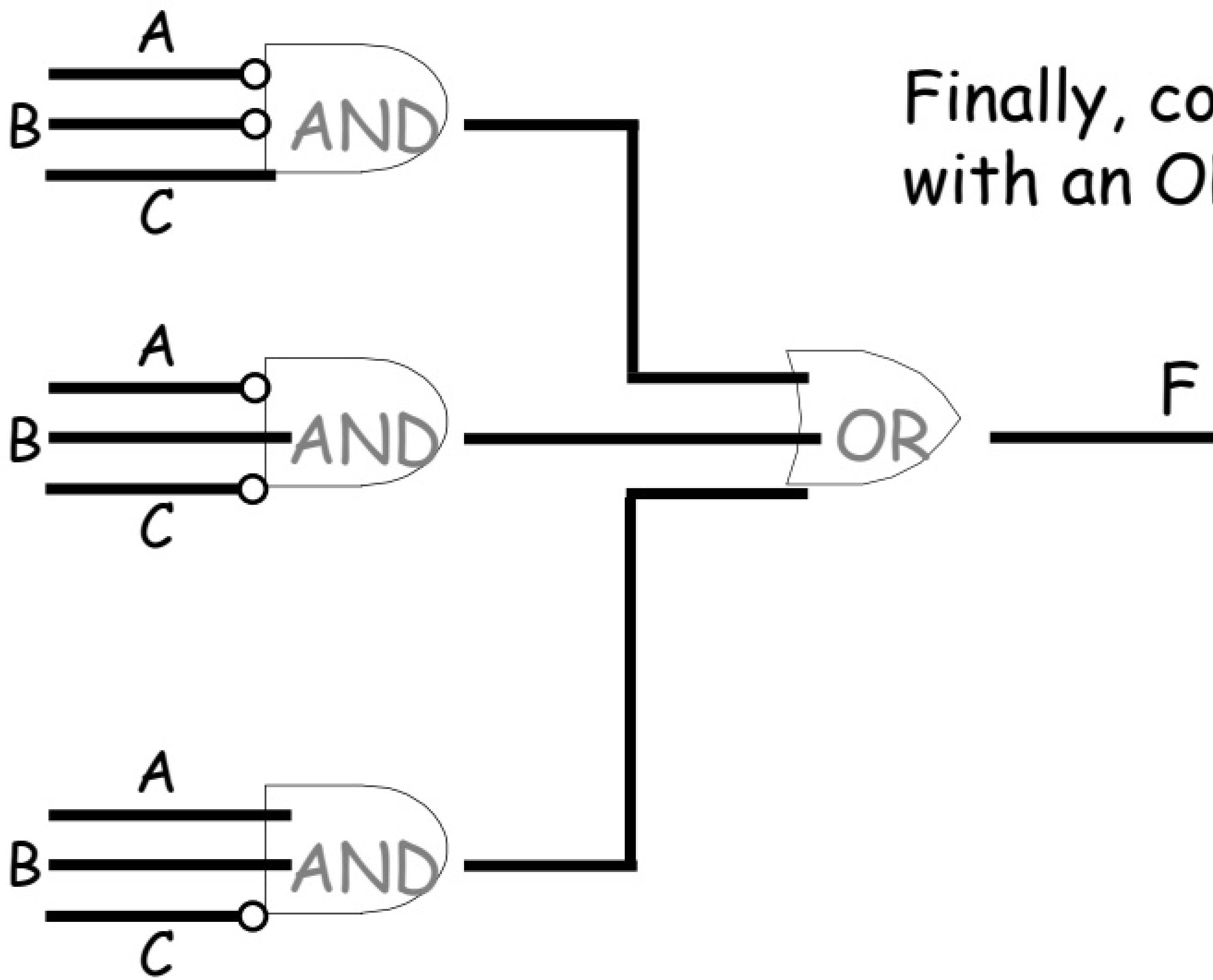
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Any Truth Table (cont.)



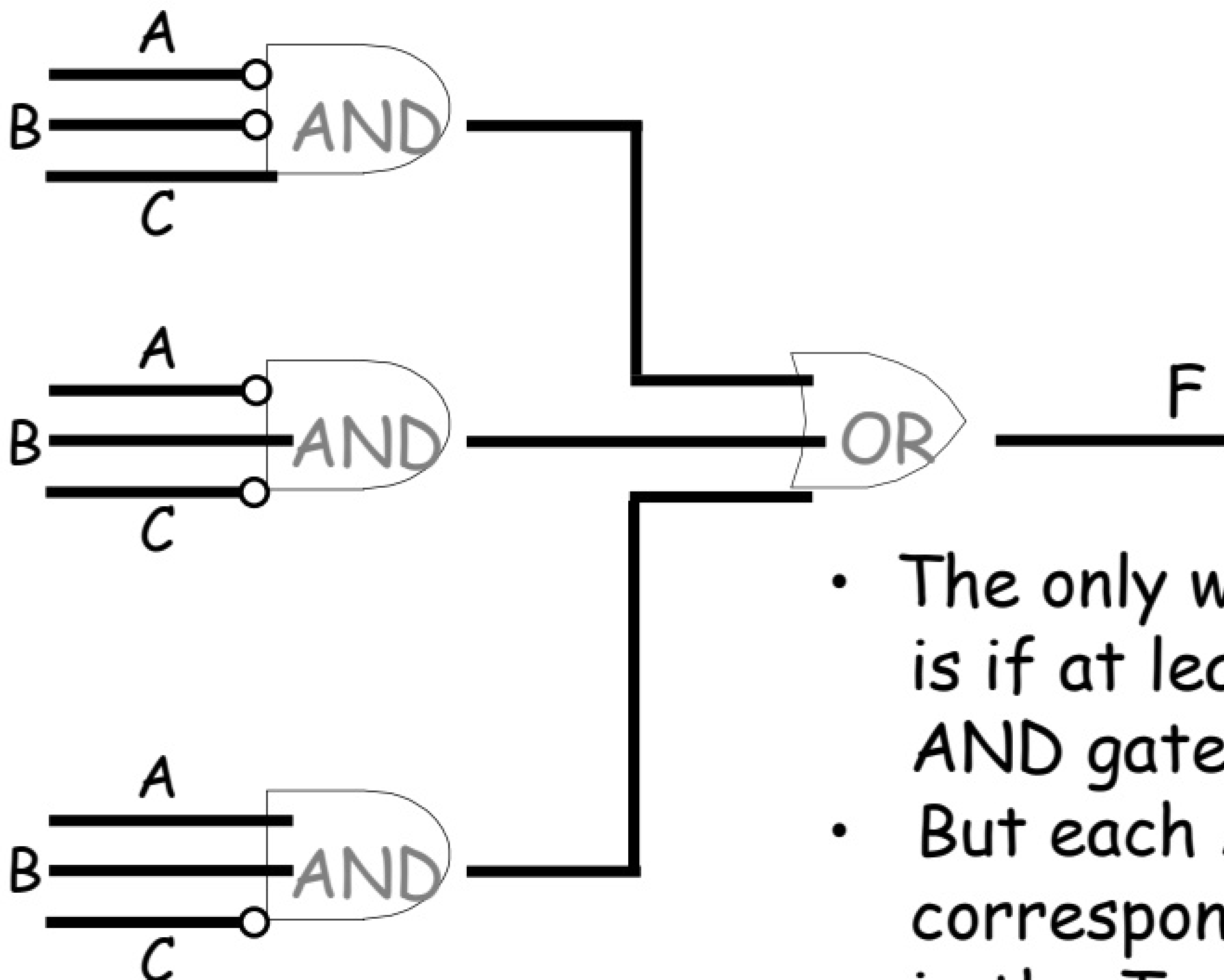
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Any Truth Table (cont.)



Finally, combine them with an OR gate.

Any Truth Table (cont.)



- The only way for F=1 is if at least ONE of AND gates outputs 1.
- But each AND gate corresponds to a row in the Truth Table with a 1 in the output!

Any Truth Table (cont.)

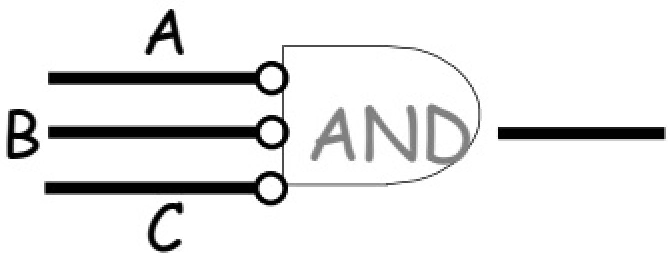
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

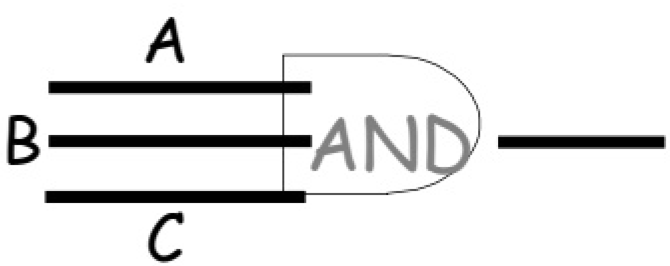
Another Example

First, make circuits for each row in Truth Table with a 1 in the output.

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

Another Example (cont.)

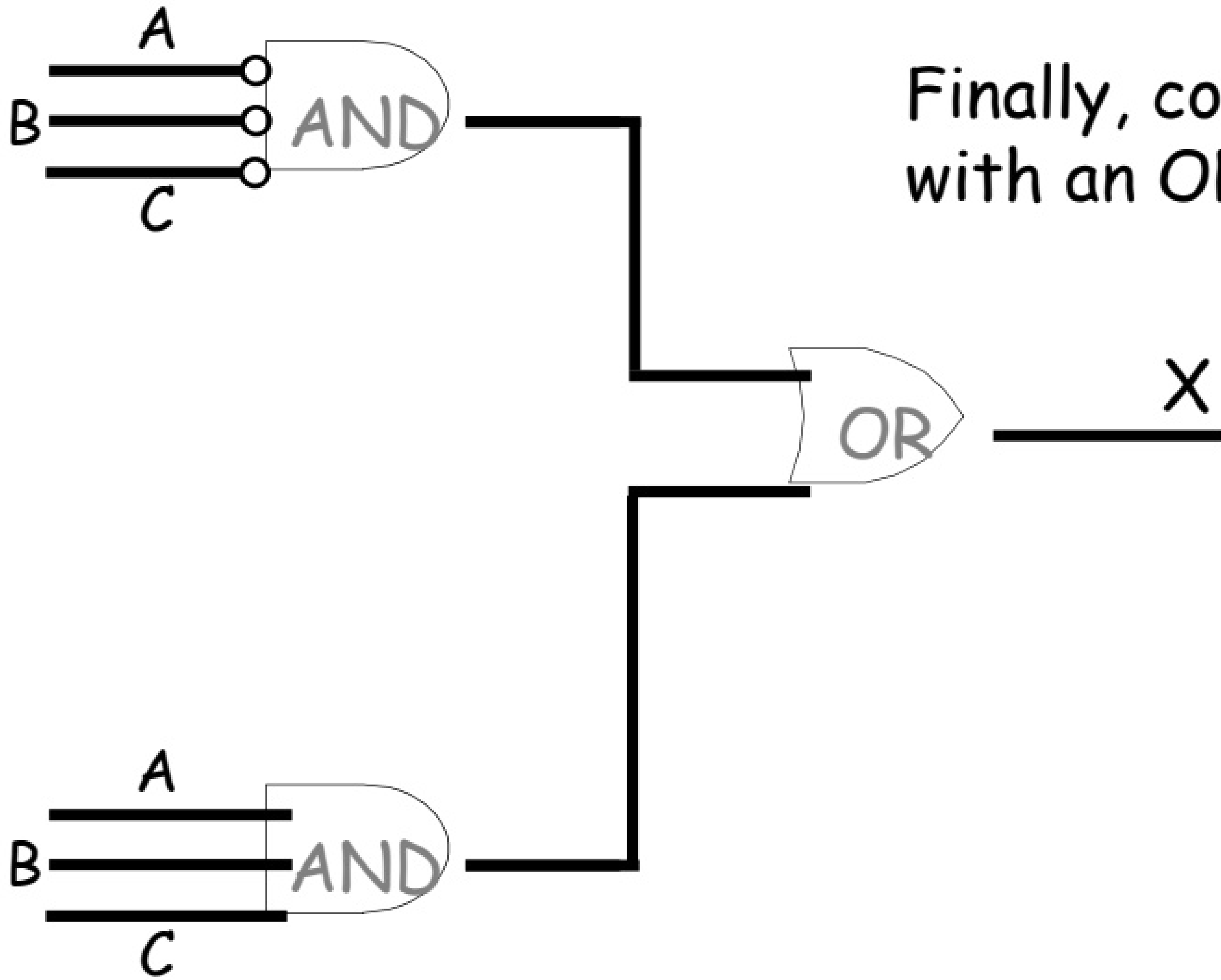




A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Another Example (cont.)

Finally, combine them with an OR gate.



Another Example

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

Universality

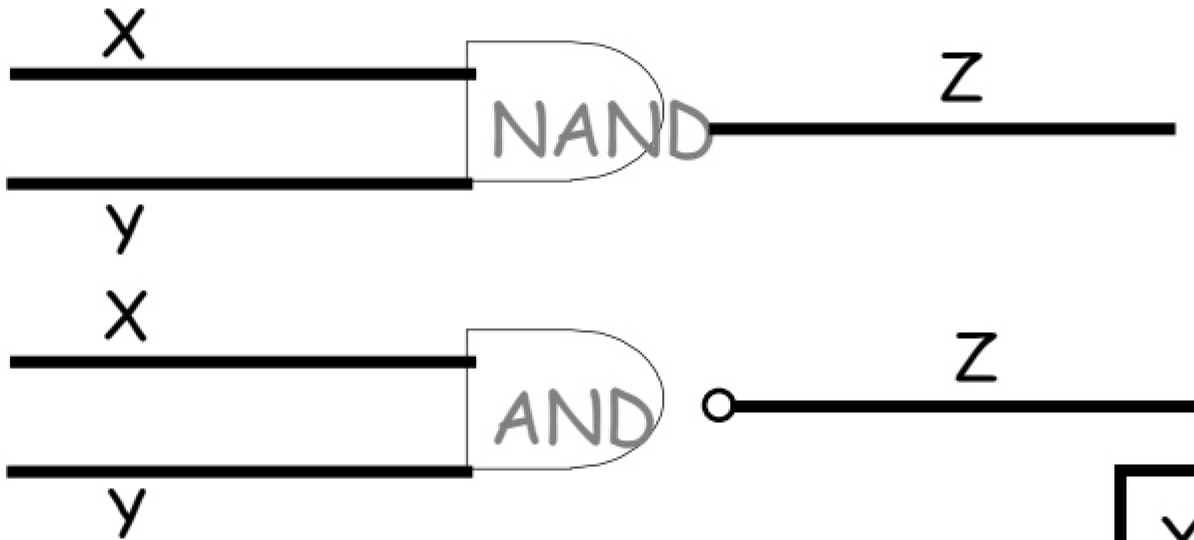
- Note same idea works no matter how many input variables.
- So for ANY Truth Table we can write down, we can make a circuit for it using only 3 Logic Gates: AND, OR, NOT
- This gives us a very powerful tool !
- Our first technical use of *abstraction*: "Make a circuit for that Truth Table." is something we can abstract and understand.

Further Issues

Some issues to think about on your own

- We know that AND,OR, and NOT are "universal" – we can make a circuit for any Truth Table using just these gates ! What else is universal?
- ***Surprising answer***: There is a *single* gate called "NAND" which is universal all by itself !

NAND Gate



X	y	Z
0	0	1
0	1	1
1	0	1
1	1	0

Truth Table

Next Time:
Why are 0's and 1's all we need?