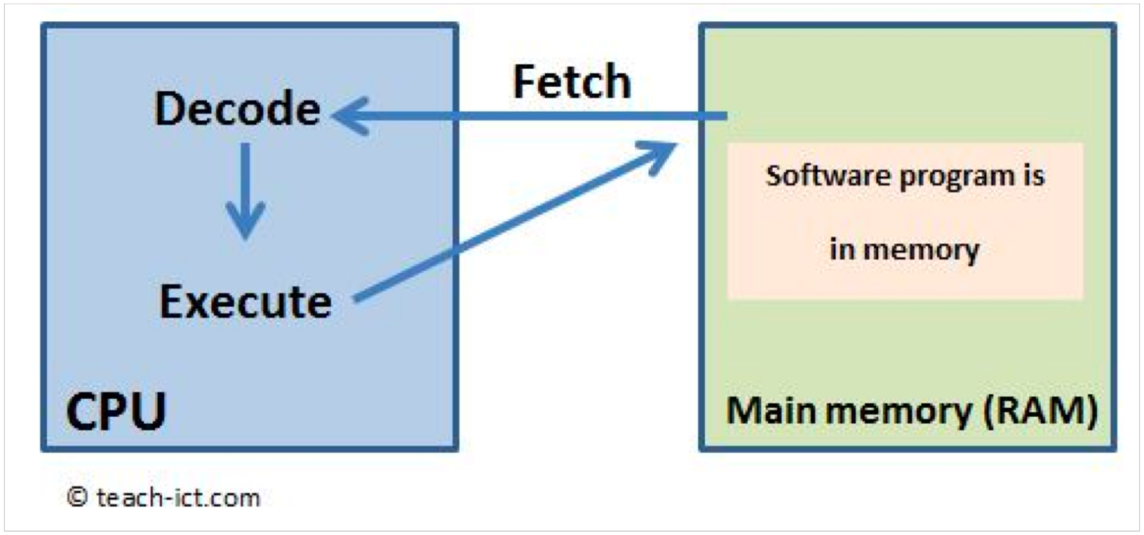
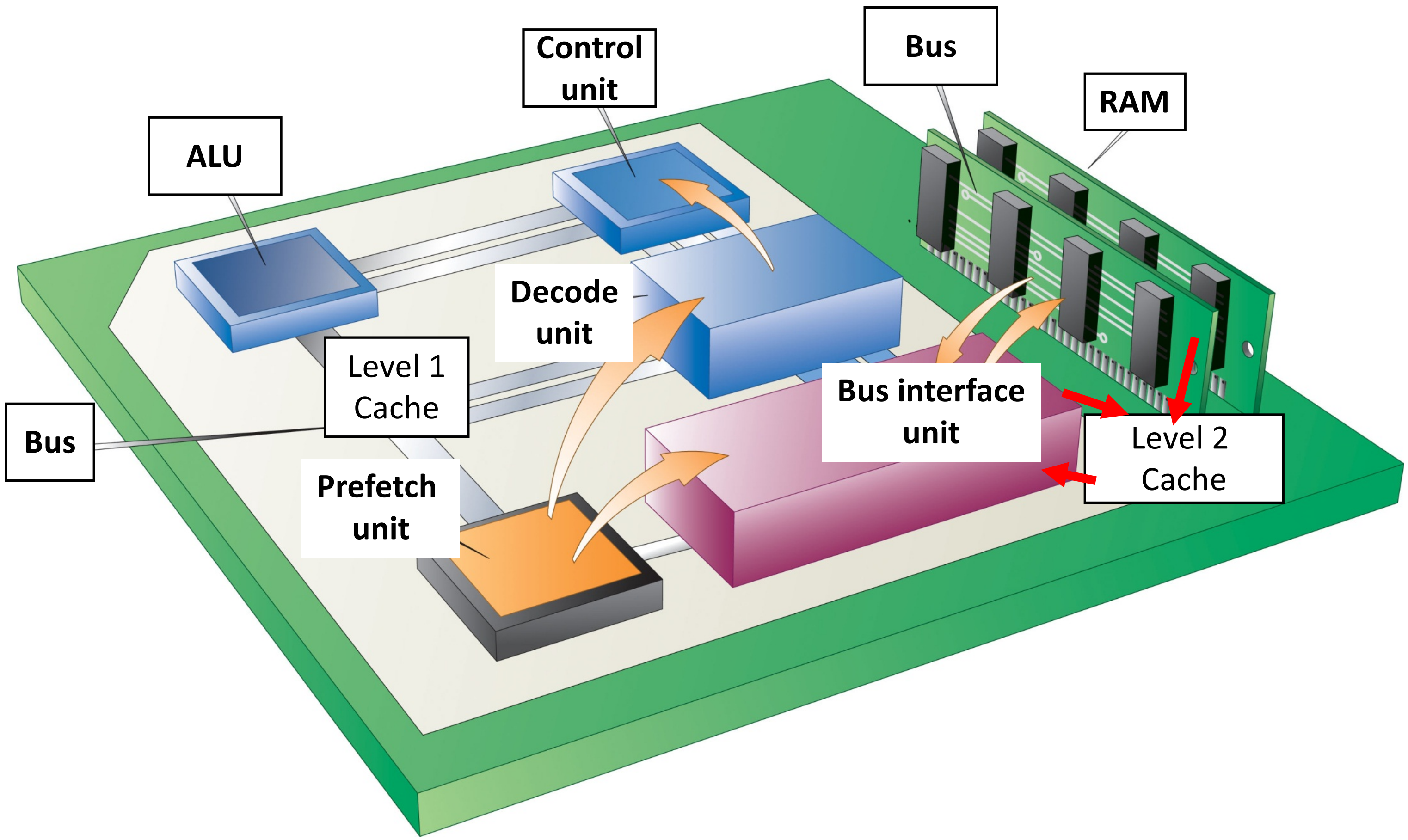
Fetch Decode Execute

Fetch Decode Execute Cycle



http://www.teach-ict.com/2016/GCSE_Computing/OCR_J276/1_1_systems_architecture/fetch_execute_cycle/miniweb/index.php


Control unit
Bus
RAM
ALU
Decode unit
Level 1 Cache
Bus
Bus interface unit
Level 2 Cache
Prefetch unit

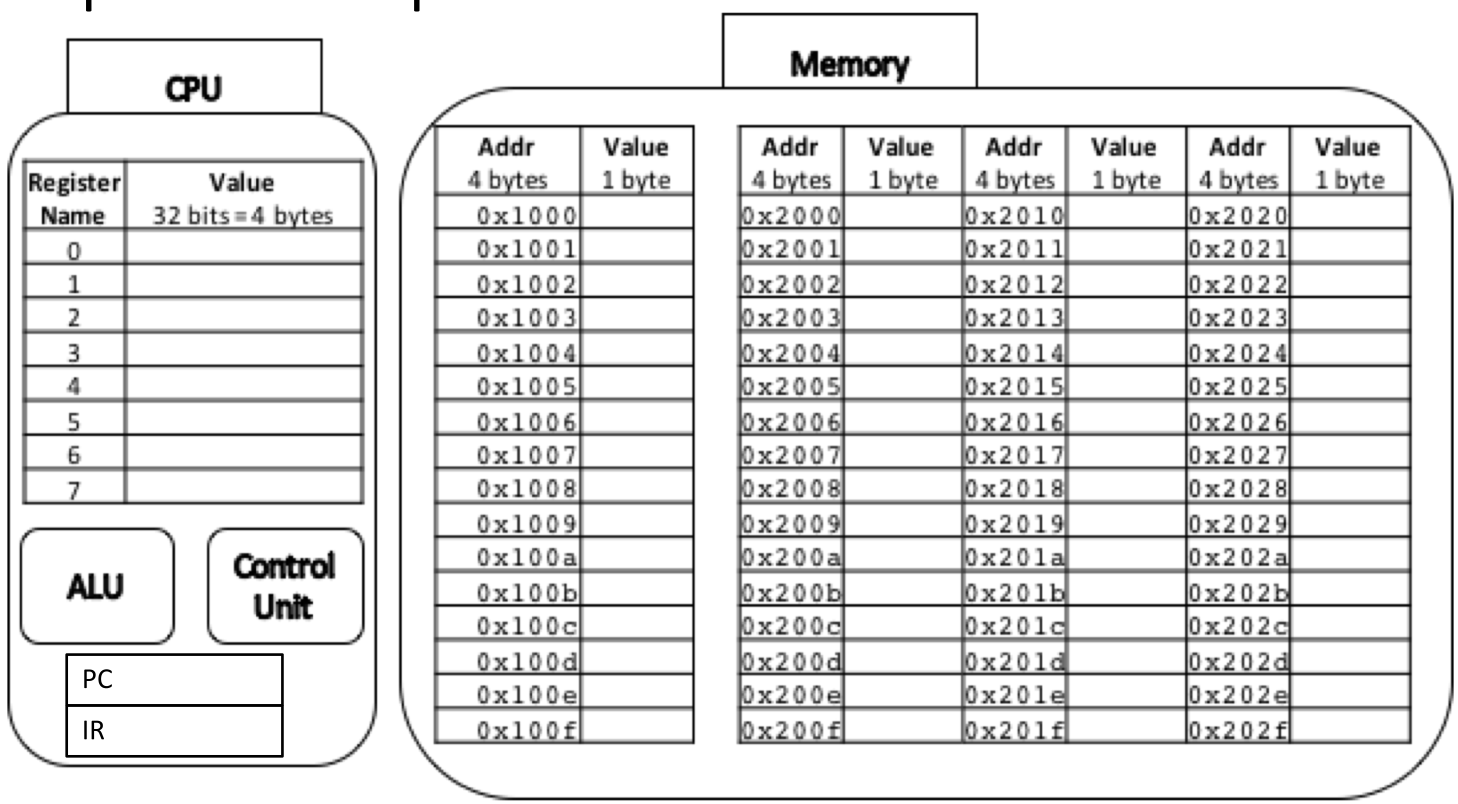
Processor Speed

Factors:

- System clock
  - # of instructions executed per second (ie. 2.5 GHz)
  - MHz is millions of cycles per second, GHz is billions
- Bus Width
  - The number of bits that can be transmitted in one clock cycle
- Word Size
  - preferred size for moving units of info (e.g. 32-bit, 64-bit)
  - Address, instruction and bus sizes are usually multiples of the word size

  Architecture
  - general design of CPU

A simplified representation…

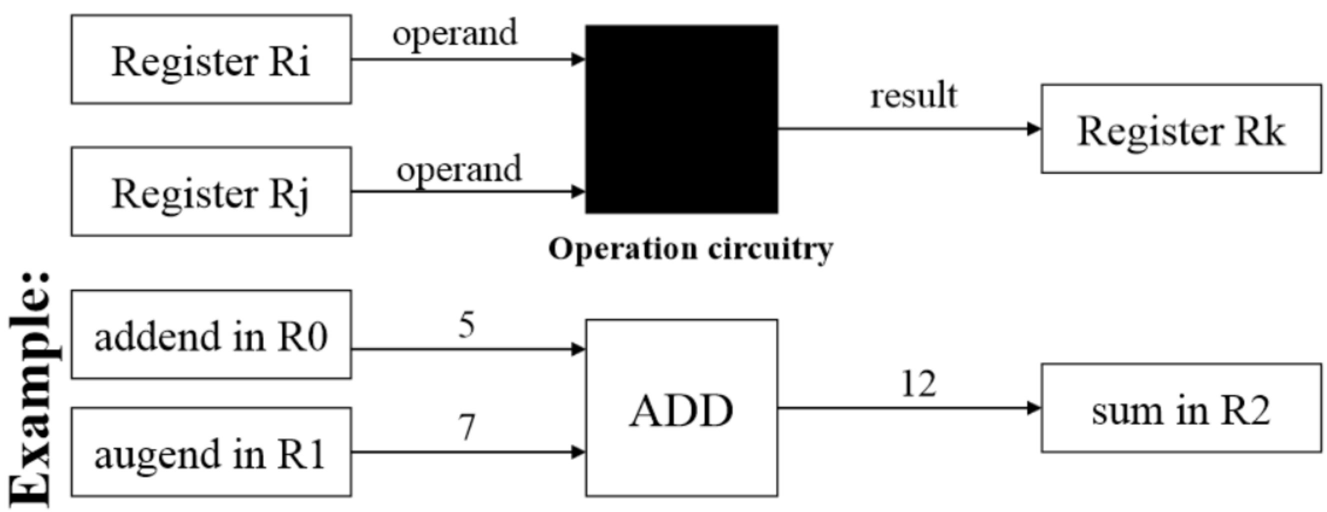


PC – Program Counter: holds the address of the next instruction

IR – Instruction Register: holds the current instruction being executed

How Registers are used in the CPU

- Every arithmetic or logical operation has one or more operands and one result.
- Operands are contained in registers (“source”).
- A “black box” of circuits performs the operation.
- The result goes into a register (“destination”).



Slide Credit: Dr. Sudhakar Ganti

Digital Logic Building Blocks

- Logic functions are implemented using basic building blocks of AND, OR, NOT, XOR

- Order of operations: BNAO

AND $\wedge$	NAND
OR $\vee$	NOR
NOT $\neg$	
XOR $\oplus$	

Logic

- I bring my umbrella when it is raining,
  or when the forecast is bad

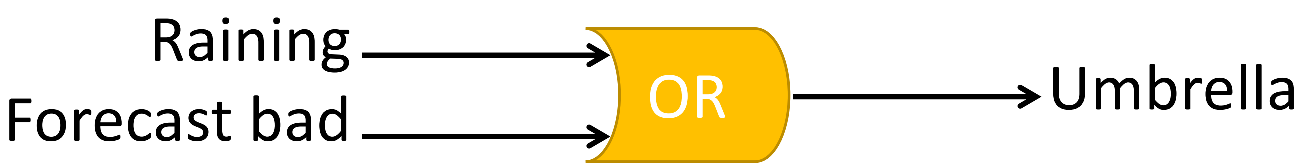
  - raining, forecast bad -> umbrella
  - raining, forecast good -> umbrella
  - not raining, forecast bad -> umbrella
  - not raining, forecast good -> no umbrella

adapted from http://www.doc.ic.ac.uk/~dfg/hardware/HardwareLecture01.pdf

OR Truth Table (+ or V)

1 = true, 0 = false

Raining?	Forecast bad?	Umbrella (raining or forecast bad)
0	0	0
0	1	1
1	0	1
1	1	1



Logic

- **Or** is a logical operation
  - it means one or the other or both must be true
- **And** is also a logical operation
  - it means both must be true

- **Boolean value**: TRUE or FALSE (1 or 0)
- **Boolean expression**: An expression that is either TRUE or FALSE (1 or 0)

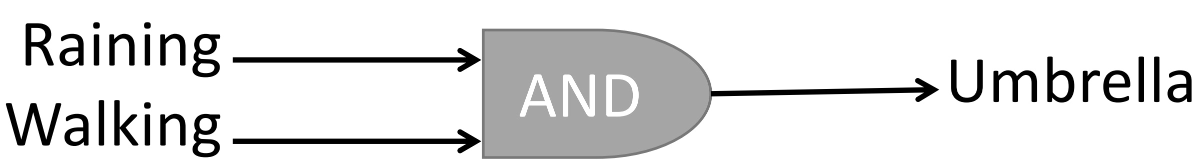
Logic

- I bring my umbrella if I am walking to work **and** it is raining

AND Truth Table (• or ∧)

1 = true, 0 = false

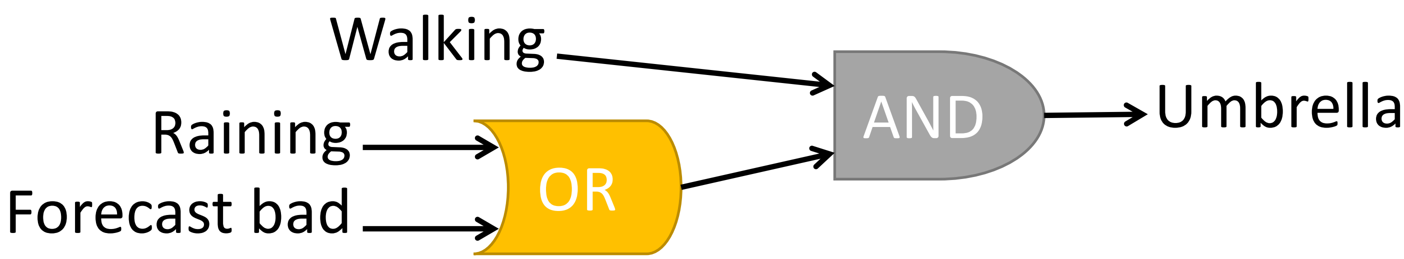
Raining?	Walking to work?	Umbrella (raining and walking)
0	0	0
0	1	0
1	0	0
1	1	1



Logic

- I bring my umbrella if I am walking to work **and** it is raining **or** the forecast is bad

$$\text{walking} \land (\text{raining} \lor \text{forecast\_bad})$$



Truth Table

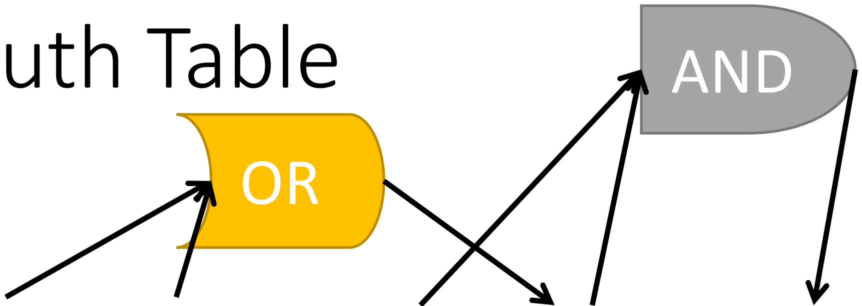
1 = true, 0 = false

Raining	Forecast bad	Walking
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

All of the possible assignments of binary values to 3 variables

Corresponds to the binary encoding of 0…7 (top to bottom)

Truth Table



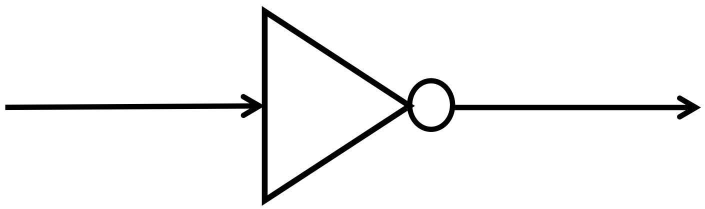
Raining	Forecast bad	Walking	Rain or forecast bad	Umbrella
0	0	0	0	0
0	0	1	0	0
0	1	0	1	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	1
1	1	0	1	0
1	1	1	1	1

Not (x or)

— ¬

1 = true, 0 = false

A	¬ A
0	1
1	0



XOR Truth table ( ) $\oplus$

- XOR – only one or the other can be true

A	B	A XOR B
0	0	
0	1	
1	0	
1	1	

Test your understanding

(X AND Y) OR Z

Which assignment will make the logical expression true?

a) X=1, Y=0, Z=0
b) X=0, Y=0, Z=1
c) X=0, Y=0, Z=0

Test your understanding

$$(X \wedge \neg Y) \vee \neg Z$$

Which assignment will make the logical expression true?

a) X=1, Y=0, Z=1
b) X=0, Y=0, Z=1
c) X=0, Y=0, Z=0
d) a and b
e) a and c

Test your understanding

$$(X \wedge \neg Y) \vee \neg Z$$

Which assignment will make the logical expression true?

Lets make a truth table…

Test your understanding

$$(X \land \neg Y) \lor \neg Z$$

Which assignment will make the logical expression true?

a) X=1, Y=0, Z=1
b) X=0, Y=0, Z=1
c) X=0, Y=0, Z=0
d) a and b
e) a and c

Compare for Equality (1 bit)

- Return 1 if two bits are equal
- Use only AND, OR and NOT

Note the assignments that work out to true (1)

A	B	A=B?
0	0	1
0	1	0
1	0	0
1	1	1

Desired Output

Shared by I. Dufour from:
University of Victoria - CSC 106 – Fall 2018

Compare for Equality

A	B	A∧B	A=B?
0	0	0	1
0	1	0	0
1	0	0	0
1	1	1	1

Compare for Equality

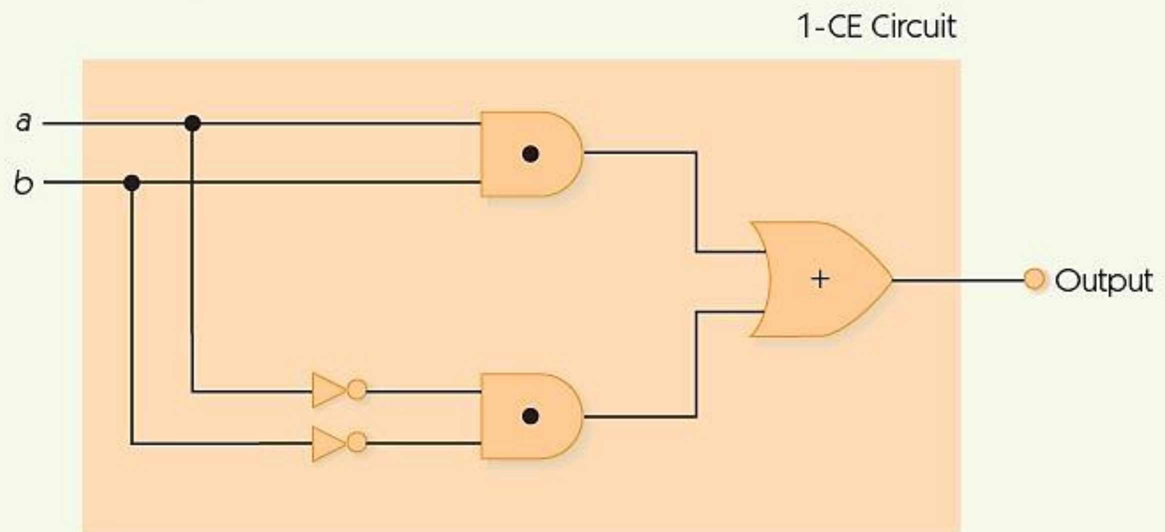
A	B	A∧B	¬A∧¬B	A=B?
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

Compare for Equality

A	B	A∧B	¬A∧¬B	(A∧B) V (¬A∧¬B)	A=B?
0	0	0	1	1	1
0	1	0	0	0	0
1	0	0	0	0	0
1	1	1	0	1	1

1-CE

FIGURE 4.27

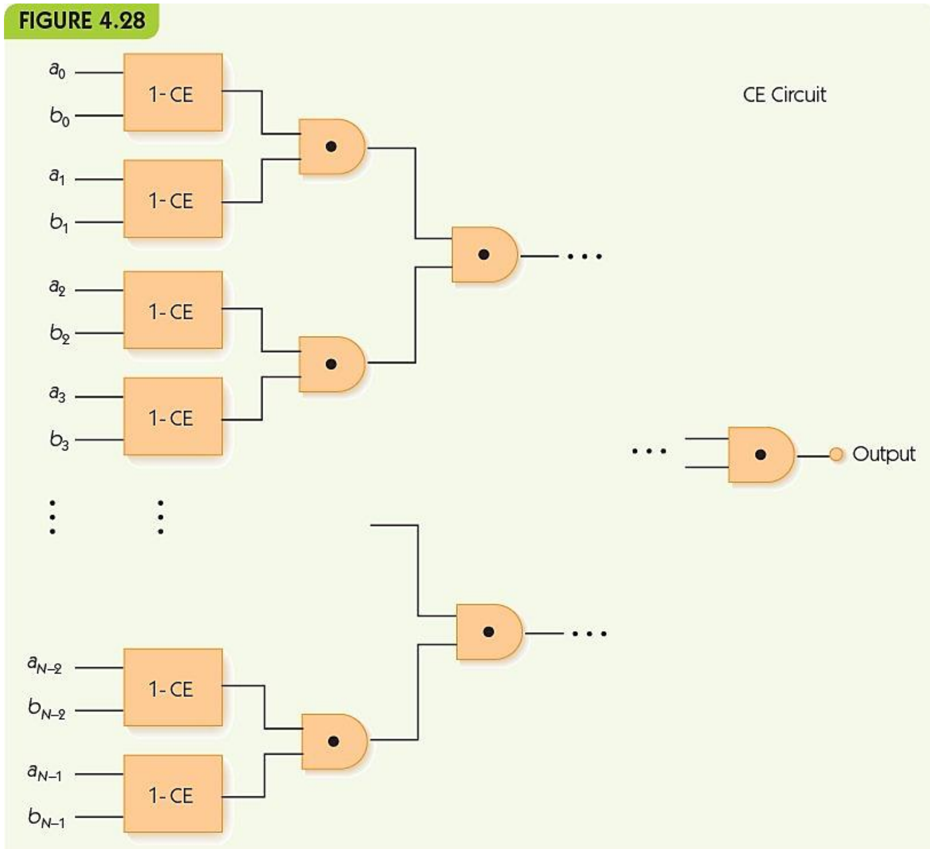


One-bit compare-for-equality circuit

Compare for Equality (CE)

- N-bit CE circuit
- Input:
  - $a_0a_2\ldots a_{n-1}$
  - $b_0b_2\ldots b_{n-1}$

  (a_i and b_i are individual bits)
- Pair up corresponding bits:
  - a_0 with b_0
  - a_1 with b_1
  - etc.
- Run a 1-CE circuit on each pair
- Perform AND on all of the 1-CE outputs



N-bit compare-for-equality circuit