FIT 1047

Introduction to computer systems, networks and security

MONASH University

Topics for week 2

- Error detection
- Boolean algebra
- Logical circuits

Error detection and correction

There might be errors in binary data.

Single bits might flip, parts of a word might be missing, etc.

Error detection and correction

- Parity
- Checksum
- CRC – Cyclic Redundancy Check

Parity

What does parity mean?

Parity is just another fancy word for equality

- Needs one additional parity bit
- Decide on even or odd for the complete number
- Set parity bit to 0 or 1 so that number of 1s is even (for even parity), or odd (for odd parity)

Example for even parity:

0	1	0	1	1	1	0	

Example for even parity:

0	1	0	1	1	1	0	

Calculate parity bit to get an even number of 1s:

Example for even parity:

0	1	0	1	1	1	0	

Calculate parity bit to get an even number of 1s:

0	1	0	1	1	1	0	0

Example for even parity:

0	1	0	1	1	1	0	

Calculate parity bit to get an even number of 1s:

0	1	0	1	1	1	0	0

One bit error:

0	1	0	1	1	0	0	0

Example for even parity:

0	1	0	1	1	1	0	

Calculate parity bit to get an even number of 1s:

0	1	0	1	1	1	0	0

One bit error:

0	1	0	1	1	0	0	0

Two errors:

0	0	0	1	1	0	0	0

Summary Parity Bit

- Just add up all bits and add one additional parity bit.
- This bit can be 0 or 1, the result is even for even parity or odd for odd parity.
- A parity bit can only detect a single bit error.

Checksum

- Parity was just about counting. Odd or even.
- Checksums need a bit more processing power

Lets look at a message
43 52 43 30 31 30

1. Pick a number size we want to divide by and agree on it. Lets use 16.
2. Add all numbers. Results in 229
3. Divide sum by the number agreed on. 229 / 16 is 14 with a remainder of 5.
4. Only take the remainder as the checksum.
5. Send the checksum with the message and check on receipt.

Checksum example:

43	52	43	30	31	30	5

Checksum example:

43	52	43	30	31	30	5

One error:

42	52	43	30	31	30	4 □ 5

Checksum example:

43	52	43	30	31	30	5

Two errors:

42	52	43	30	30	30	3 □ 5

Checksum example:

43	52	43	30	31	30	5

Another two errors:

42	52	43	30	32	30	5 = 5

Summary Checksum

- A checksum is the result (just the remainder) of adding up all numbers.
- Can detect multiple errors.
- Errors can get canceled out.

Cyclic Redundancy Check
CRC

Instead of adding up the number, concatenate into one big number:

435243303130	

Divide by a previously established number (we use 16):

435243303130 / 16	

Divide by a previously established number (we use 16) and take the remainder:

435243303130 / 16	10

- In binary, CRCs can work over several bytes.
- The standardised number for division is a polynomial in the ring of polynomials over the finite field GF(2).
- Bits of the message are coefficients of a polynomial.
- The standardised number is the generator polynomial.
- Bits of the CRC are the coefficient of the polynomial derived by dividing the message polynomial by the generator polynomial.

Many standardised CRC codes of different lengths exist.

Summary CRC

- Cyclic redundancy checks CRCs can detect various types of errors.
- CRCs and other redundancy checks are frequently used.
- Example: QR codes can contain up to 30% redundancy.

CRC codes do not provide security!

An attacker could just manipulate the message and compute a new CRC code.

Important: CRC is not a security measure. CRCs are about errors (safety) not malicious attacks (security).

Boolean logic and Boolean algebra

A little bit of history

George Boole,

born November 2, 1815, Lincoln, Lincolnshire, England

died December 8, 1864, Ballintemple, County Cork, Ireland

English mathematician who helped establish modern symbolic logic and whose algebra of logic, now called Boolean algebra, is basic to the design of digital computer circuits.

(Enciclopaedia Britannica)

Boolean logic is probably the simplest possible (useful) logic.

Basic concepts:

- TRUE, FALSE
- AND, OR
- NOT

TRUE and FALSE are values for statements.
Note that not all statements qualify:

- Today, the temperature is over 15 degrees Celsius.
- Today, the weather is good.
- Haggis tastes great.

Usually, TRUE is represented by 1 and FALSE is represented by 0.

Do not confuse binary and Boolean:

	Binary	Boolean
0	Zero	FALSE
1	One	TRUE

A AND B can be represented as
A ∧ B, A × B, AB

A OR B can be represented as
A ∨ B, A+B

NOT A can be represented as
$\overline{A}$

Do not confuse binary and Boolean:

	Binary	Boolean
0+0	0	0 OR 0 = 0
1+1	10	1 OR 1 = 1

AND

Statement A AND Statement B = TRUE only when both statements are TRUE.

Examples

- Donald Duck wears a blue sailor suit AND

Donald Duck does not wear trousers.

Is obviously TRUE

- At the moment we have 15 degrees AND at the moment we have below 5 degrees.

Is obviously FALSE

:ompact

		AB
		0
	1	0
1	0	0
1	1	1

OR

A OR B means that either A or B or both are TRUE

OR different from our usual understanding of or.
Example: On my toast I like bacon or jam.
In Boolean logic, this would mean I also enjoy having both, bacon and jam on my toast, which is actually not true.

Examples

- Today it will be warm OR it will be raining.

Means that it can be warm and dry, cold and raining, or warm and raining.

- Today it is more than 15 degrees OR below 16 degrees.

This is obviously always TRUE.

Truth table for OR

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Now, we only need to add negation and we can construct rather complex expressions.

NOT

If something is TRUE, then, the negation NOT TRUE is obviously FALSE.

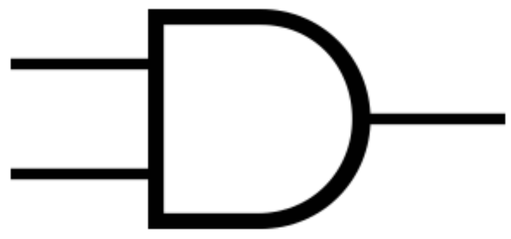
A	$\overline{A}$
0	1
1	0

In electrical circuits, AND, OR, NOT and additional operators (e.g. XOR, NAND, NOR) are realised as so-called logic gates.

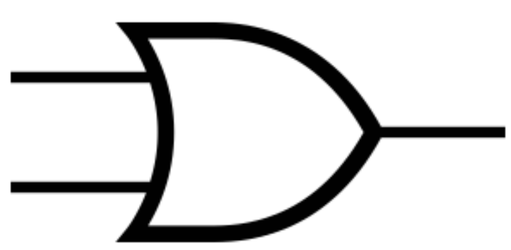
A gate performs one (or several) logical operations on some logical input (i.e. bits) and produces a single logical output.

Examples for gates

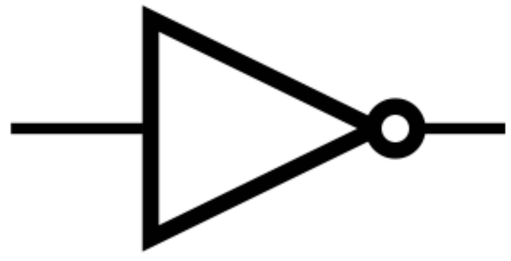
AND



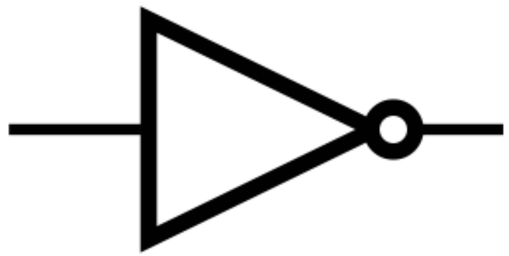
OR



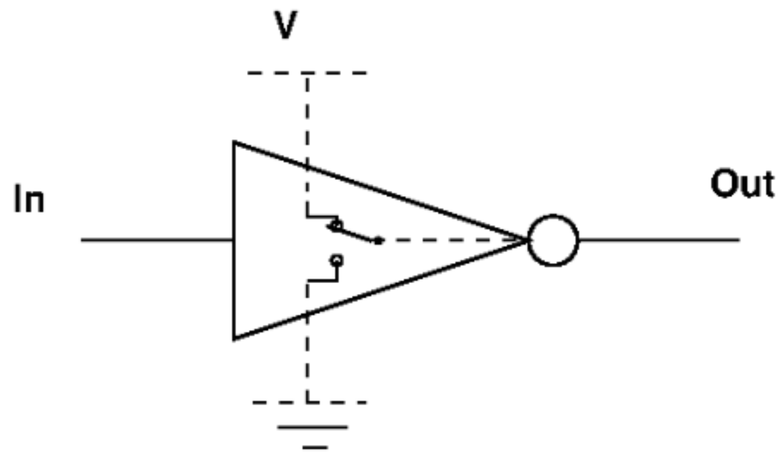
NOT



Logical circuits are not electrical circuits!
Low input can result in high output:



The reason is, that power supply is not shown in the schematic symbols.



- In the lab we will use a tool (Logisim) to simulate logical circuits.
- Lets look at some examples.

Tutorials this week

- Some exercises with numbers (understand why 2's complement works)
- Learn a few rules for boolean algebra
- Use logisim to simulate first logic circuits