# Bits, bytes and friends

Michela Ceria
University of Trento

Giancarlo Rinaldo
Affiliation not available

Massimiliano Sala
University of Trento

September 21, 2018

# Contents

## 0.1 Bits standard operations and logic

A *bit* (binary digit) is a unit of measure for information, introduced by C. Shannon in 1948.
It might be seen as the minimal amount of information needed in order to distinguish among two events occurring with the same probability.
Bits are used in Information Theory and - in general - in most of Computer Science applications.
They are denoted with the constants 0 and 1, representing the two events with the same probability. Their set is often denoted with $\{0, 1\}$, but we need another notation, that is,

$$\mathbb{F}_2 = \{0, 1\}$$

Indeed, with this different notation we mean that we can perform operations on bits. More precisely, we want to introduce two operations, *sum* and *multiplication*, retaining some similarity with the usual operations of sum and product for numbers. The numbers we are interested in are integers, which we collect in a set called $\mathbb{Z} = \{\ldots, -1, 0, 1, 2, \ldots\}$, non-negative integers, which we collect in a set called $\mathbb{N} = \{0, 1, 2, \ldots\}$, and rational numbers, which we collect in a set called $\mathbb{Q} = \{\ldots, \frac{-5}{101}, 0, \frac{1}{2}, 3, \ldots\}$.
Since $\mathbb{F}_2$ is so small, we can use the following table to show how to sum and multiply bits.

| $a$ | $b$ | $a + b$ | $a \times b$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

Table 1: Sum and product

The first column and the second contain the values of two input variables, $a$ and $b$, representing the two bits we sum or multiply; the third and the fourth, respectively, contain the sum and the product of $a$ and $b$.

**Exercise 1.** *Compute the following operations in* $\mathbb{F}_2$*:*

- $(1 + 1) + 0$
- $(1 + 1) + 1$
- $(1 + 1) \cdot 1$
- $(0 + 0) + 1 + (1 + 0) \cdot 1$

Operations on bits share properties with those involving integer or rational numbers.

Let us start with a comparison between the sum in $\mathbb{F}_2$ and the sum in $\mathbb{Z}$.

We can sum $2, 3$ and $4$ in $\mathbb{Z}$ and it happens that

$$(2+3)+4 = 5+4 = 9 = 2+7 = 2+(3+4),$$

so we can write also $9 = 2+3+4$, omitting the parentheses. Actually, this holds for any three numbers in $\mathbb{Z}$.
We claim that this holds *also* for the sum of bits. For example

$$(1+0)+1 = 1+1 = 0 = 1+(0+1) = 1+0+1\,.$$

The reader can verify our claim for any $a, b, c \in \mathbb{F}_2$.
Any time we have a set and a sum operation which satisfies the general property

$$\forall a, b, c, \quad (a+b)+c \;=\; a+(b+c) \;=\; a+b+c\,,$$

we say that the operation is *associative*.
Therefore, we can conclude that both the sum in $\mathbb{Z}$ and the sum in $\mathbb{F}_2$ are associative operations.

Consider now $2, 3 \in \mathbb{Z}$. We know that $2+3 = 3+2 = 5$ and this holds for any pair of integers. We claim that this holds also for bits, as for example

$$0+1 \;=\; 1+0 \;=\; 1\,.$$

We leave to the reader the verification of this statement for each $a, b \in \mathbb{F}_2$.
We can formalize this property by saying that both the sum in $\mathbb{Z}$ and the sum in $\mathbb{F}_2$ are *commutative* operations. In a more general setting, a sum operation on a set is called commutative if for any $a, b$ in that set we have

$$a+b \;=\; b+a\,.$$

**Exercise 2.** *Compute the following operations*

*a.* $1+1+1+0+0+0+0+1$

*b.* $1+1+1+1+0+0+0+0$

*May you argue the result of b. once knowing a.? If so, which properties of bits are you using?*

Now we take a close look at $0 \in \mathbb{Z}$. Considering any integer $a$, we have that $a+0 = 0+a = a$, as for example $3+0 = 0+3 = 3$. In other words, summing a number with zero leaves the number unchanged (and this happens *only* for 0).
The same happens with $0 \in \mathbb{F}_2$, since in $\mathbb{F}_2$ $0+0 = 0$ and $0+1 = 1+0 = 1$.
In a general setting, if we have a sum operation with a special element $e$ such that $a+e = e+a = a$ for any $a$ in the set, then we say that $e$ is the *neutral element* of the operation.
Therefore, we can say that $0 \in \mathbb{F}_2$ is the neutral element of the sum in $\mathbb{F}_2$, while $0 \in \mathbb{Z}$ is the neutral element in $\mathbb{Z}$.

**Remark 1.** *When there is a set with a sum operation, it is usual to call* 0 *the neutral element. This is unfortunate because for example the* 0 *in* $\mathbb{F}_2$ *and the* 0 *are in* $\mathbb{Z}$ *are two totally different objects. We are confident that the readers will soon be accustomed to this abuse of notation.*

Another important property of the sum in $\mathbb{Z}$ is that, given an element $a \in \mathbb{Z}$, we can always find an element $b \in \mathbb{Z}$ such that $a + b = b + a = 0$. For example, for $a = 3$, we have $b = -3$, getting $3 + (-3) = (-3) + 3 = 0$.
In the general case, when there is a set with a sum, if for any element $a$ there is another element $b$ such that $a + b = 0$ and this element is unique, then we say that $b$ is the *opposite* of $a$ and we write $b = -a$.
Opposites exist also for the sum in the set of bits, indeed

$$0 + 0 = 0 \quad \text{and} \quad 1 + 1 = 0$$

and so in $\mathbb{F}_2$:

- the opposite of 1 is 1 (and $-1 = 1$),
- the opposite of 0 is 0 (and $-0 = 0$).

**Exercise 3.** *What is the opposite of*

- $(1 + 0) \cdot 1$*?*
- $(0 + 0) \cdot 0$*?*
- $1 + 1 + 1 + 1 + 0 + 1 + (1 \cdot 1)$*?*

It is time that we introduce more formalism. We can consider any set $G$ endowed with an operation $*$ such that for any $a, b$ in $G$ the operation outputs another element $a * b$ of $G$. If the operation satisfies the following stringent properties

i) $*$ is associative;

ii) $*$ is commutative;

iii) $*$ has a neutral element;

iv) each element of $G$ has an opposite w.r.t. $*$

then $G$ (with the operation $*$) is called an *abelian group.*

**Remark 2.** *The adjective "abelian" indicates that ii) holds, i.e. that the operation* $*$ *is commutative. If only i), iii) and iv) hold,* $G$ *is only a group.*

Our previous observations about the sum in $\mathbb{Z}$ and the sum in $\mathbb{F}_2$ lead us to claim that both $\mathbb{Z}$ and $\mathbb{F}_2$ are *abelian groups w.r.t. their sum.*

Having dealt with the sum, we pass now to compare the multiplication of integers and that of bits.
Consider again $2, 3, 4 \in \mathbb{Z}$. We have that

$$(2 \cdot 3) \cdot 4 = 6 \cdot 4 = 24 = 2 \cdot (3 \cdot 4) = 2 \cdot 12,$$

so, as it was for the sum, we can remove the parentheses and write

$$2 \cdot 3 \cdot 4 = 24.$$

The same holds for bits, for example

$$(0 \cdot 1) \cdot 1 = 0 \cdot 1 = 0 = 0 \cdot (1 \cdot 1) = 0 \cdot 1 = 0 \cdot 1 \cdot 1.$$

We leave to the reader the verification that this holds for each bit triplet.
This property can be formalized by saying that both the multiplication in $\mathbb{Z}$ and the multiplication in $\mathbb{F}_2$ are *associative*. In formulas, we say that for any $a, b, c$ (which are elements of $\mathbb{Z}$ or of $\mathbb{F}_2$), we have

$$(a \cdot b) \cdot c \; = \; a \cdot (b \cdot c) \; = \; a \cdot b \cdot c \, .$$

Consider now $2, 3 \in \mathbb{Z}$. We know that $2 \cdot 3 = 3 \cdot 2 = 6$ and this holds for every pair of integers. We claim the same for bits, since for example

$$0 \cdot 1 \; = \; 1 \cdot 0 \; = \; 0 \, .$$

We leave, as an exercise to the reader, to verify this for each $a, b \in \mathbb{F}_2$.
Again, this property can be formalized by saying that both the product in $\mathbb{Z}$ and the product in $\mathbb{F}_2$ are *commutative*. In formulas, we say that for any $a, b$ (which are elements of $\mathbb{Z}$ or of $\mathbb{F}_2$), we have

$$a \cdot b = b \cdot a.$$

We examine now the behaviour of $1 \in \mathbb{Z}$. Taken any other integer number $a$, we have that $a \cdot 1 = 1 \cdot a = a$, for example $3 \cdot 1 = 1 \cdot 3 = 3$, so multiplying a number with 1 gives, as result, again that number.
Also the bit $1 \in \mathbb{F}_2$, has the same behaviour, since $1 \cdot 0 = 0 \cdot 1 = 0$ and $1 \cdot 1 = 1$.
We can formalize this statement by saying that 1 is the *neutral element* of the product, both in $\mathbb{Z}$ and in $\mathbb{F}_2$.
Once we have introduced a neutral element, we might wish to go on and define a group, as we did with the sum. Unfortunately, this cannot always be done, as we will see from now on.

Given $2 \in \mathbb{Z}$, we know that we *cannot* find any integer $b$ such that $2 \cdot b = b \cdot 2 = 1$. It is still possible to find such a number $b$, but we need to move to *fractions* i.e. consider the set of *rational numbers* $\mathbb{Q}$ instead of $\mathbb{Z}$. The set $\mathbb{Q}$ is not so different from $\mathbb{Z}$, in particular all the properties previously stated for $\mathbb{Z}$ also hold for $\mathbb{Q}$. Still, if we consider $2 \in \mathbb{Q}$ we can easily find $b = \frac{1}{2} \in \mathbb{Q}$, for which $2 \cdot \frac{1}{2} = \frac{1}{2} \cdot 1 = 1$.
We can easily find a similar rational for each *nonzero* element of $\mathbb{Q}$ (while we cannot for 0).
It is of utmost interest to observe that bits behave like rational numbers rather

than integers. Indeed, $1 \cdot 1 = 1$ in $\mathbb{F}_2$, but we cannot find any bit $b$ such that $1 \cdot b = b \cdot 1 = 1$.
We can formalize this property, saying that each nonzero element of $\mathbb{Q}$ (respectively $\mathbb{F}_2$) has a *multiplicative inverse*, i.e.
$\forall a \in \mathbb{Q} \setminus \{0\}$( resp $\mathbb{F}_2 \setminus \{0\}$), $\exists a^{-1} \in \mathbb{Q} \setminus \{0\}$( respectively $\mathbb{F}_2 \setminus \{0\}$) s.t. $a \cdot a^{-1} = a^{-1} \cdot a = 1$. In $\mathbb{F}_2$, the (multiplicative) inverse of 1 is 1 and 0 has no (multiplicative) inverse (but keep in mind that the opposite of 0 exists: $-0 = 0$ in $\mathbb{F}_2$).
Finally, we show how sum and multiplication interact. Taken $2, 3, 4 \in \mathbb{Q}$, we have that

$$2 \cdot (3 + 4) = 2 \cdot 7 = 14 = (2 \cdot 3) + (2 \cdot 4) = 6 + 8.$$

This holds in general for three elements of $\mathbb{Q}$ and we can notice the same good property also in $\mathbb{F}_2$, for example:

$$1 \cdot (0 + 1) = 1 \cdot 1 = 1 = (1 \cdot 0) + (1 \cdot 1) = 0 + 1.$$

We can formalize this property, saying that, both in $\mathbb{Q}$ and in $\mathbb{F}_2$, the multiplication is *distributive* w.r.t. the sum. In formulas

$$\forall a, b, c \in \mathbb{Q} \quad (\text{respectively } \mathbb{F}_2) \quad a \cdot (b + c) \; = \; (a \cdot b) + (a \cdot c)\,.$$

**Exercise 4.** *Apply the distributive property to the following expressions:*

- $1 \cdot (1 + 0)$
- $1 \cdot (0 + 1) + 0 \cdot (1 + 1)$

A set $G$, endowed with two operations (sum and multiplication), denoted by $+, \cdot$, is called *field* if

i) $G$ is an abelian group w.r.t. $+$; let 0 be the neutral element;

ii) $G \setminus \{0\}$ is an abelian group w.r.t. $\cdot$;

iii) $\cdot$ is distributive w.r.t. $+$.

According to the above definition, we can say that both $\mathbb{Q}$ and $\mathbb{F}_2$ are fields, whereas $\mathbb{Z}$ is **not** a field.

The notation $\mathbb{F}_2$ for the set of bits (that - from now on - we will call the *field of bits*), reflects this fact, since $\mathbb{F}$ stands for "field" and the subscript 2 stands for its size.

Bits can also be seen as a way to represent the logical values TRUE/FALSE. The usual notation is 0 = FALSE and 1 = TRUE.
Once fixed this notation, we have the following well known truth tables:

We conclude this section with proposing a few exercises.

**Exercise 5.** *What can you observe by comparing tables 1 and 2? Compare with Table 1.*

| a | b | a OR b | a AND b | a XOR b |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

Table 2: Logic

| a | NOT a |
|---|---|
| 0 | 1 |
| 1 | 0 |

**Exercise 6.** *Can you represent* NOT *by some expression? Hint: Try to complete the following expression*

$$\text{NOT } a = a\_\_$$

*where $a$ is in $\mathbb{F}_2 = \{0, 1\}$ and using $0, 1, +, \cdot$.*

**Exercise 7.** *Can you represent* OR *using some expression on bits?*
*Hint: consider $a \,\text{OR}\, b = \text{NOT}\,[(\text{NOT}\, a)\,\text{AND}\,(\text{NOT}\, b)]$ and use the results in 6.*

**Exercise 8.** *Is $\mathbb{Q}$, the set of rational numbers, an abelian group w.r.t. the sum? Give detailed comments.*

**Exercise 9.** *Is $\mathbb{Q}$ an abelian group with respect to the multiplication? If so, prove it; if not, provide a counterexample.*

## 0.2 Polynomials on bits

In many cryptographic applications, as for example in Linear Feedback Shift Registers (LFSRs for short), cryptographers have to deal with polynomials whose coefficients are *bits*. In this section we will introduce these polynomials.

Let us start with *terms*. A *term* in the variable $x$ is a power of $x$, i.e. $x^a$ for some non-negative integer $a \in \mathbb{N}$. For example, $x^2$ or $x^{100}$, but also $x = x^1$ and $1 = x^0$. With terms we can define polynomials, since a *polynomial* in the variable $x$ and coefficients in $\mathbb{F}_2$ (i.e. a polynomial *over the field of bits*) is any expression of the form $a_0 + a_1 x + a_2 x^2 + ... + a_n x^n$, where $a_0, ..., a_n \in \mathbb{F}_2$ are bits and $x^i, i = 0, ...n$ are terms.
For example, $x + x^5$ and $x^3 + x^{10} + x^{21}$ are polynomials over $\mathbb{F}_2$, while $\frac{1}{2}x + \frac{2}{11}x^{10} - x^{11}$ is not.

The set $\mathbb{F}_2[x]$ contains all polynomials in the variable $x$ whose coefficients are bits:

$$\mathbb{F}_2[x] \;=\; \{1, x, x+1, x^2, x^2+x, ....\}\,.$$

The notation is the analogue of that for real polynomials, whose set is usually denoted by $\mathbb{R}[x]$.
In analogy with $\mathbb{R}[x]$, we can define the *degree* of a monomial $x^\alpha \in \mathbb{F}_2[x]$ as the *positive integer* $\alpha$. For example, the degree of $x^5$ is 5 and the degree of 1 is 0.
The degree of a polynomial $f$ is the maximal degree of the monomials appearing in $f$ with **nonzero** coefficient. Since the monomials in $\mathbb{F}_2[x]$ are only power of $x$, the degree of $f$ is actually the maximal power of $x$ appearing in $f$. The degree of the polynomial $f = 0$ is conventionally set to $-\infty$. As an example, the degree of $0 \cdot x^7 + 1 \cdot x^5 + x^4 + x^2 + x$ is 5, since the coefficient of $x^7$ is zero and the coefficient of $x^5$ is 1.
Two polynomials are called *equal* if they have the same degree $d$ and, for each $i = 0, ..., d$, term $x^i$ has the same coefficient in both.
For example $x^2 + x + 1 \;=\; 0x^4 + x^2 + x + 1$.
Having understood polynomials, we pass to perform operations with them.
First, we explain how to sum two polynomials.
This operation is defined exactly as that with real polynomials, but one has to take into account that the coefficients are bits, so they have to be summed with the "+" operation we defined for bits.
For example, if $x^3 + x^2$ and $x^2 + x + 1$ are in $\mathbb{F}_2[x]$, then

$$(x^3 + x^2) + (x^2 + x + 1) \;=\; x^3 + (1+1)x^2 + x + 1\,.$$

Since $1 + 1 = 0$ in $\mathbb{F}_2$, we finally get

$$(x^3 + x^2) + (x^2 + x + 1) \;=\; x^3 + x + 1\,.$$

**Exercise 10.** *Compute the following sums:*

- $(x^3 + x^2 + x + 1) + (x^4 + x^2 + x)$
- $(x^{10} + x^7 + x + 1) + (x^5 + x) + (x^7 + 1)$
- $x + (x^8 + x^5 + 1) + (x^6 + x^2 + x)$

As with the sum, the *product* of two polynomials is defined as the usual product of real polynomials, but taking into account that their coefficients are bits. For example

$$(x^3 + x^2) \cdot (x^2 + x) \;=\; x^5 + x^4 + x^4 + x^3 \;=\; x^5 + (1+1)x^4 + x^3 \;=\; x^5 + x^3\,.$$

We observe that $\mathbb{F}_2[x]$ is a group with the sum, as you can check by yourself. You can even show that $\mathbb{F}_2[x]$ is very close to being a field, since multiplication and addition distribute and multiplication would give rise to a group, except that, for example, there is no multiplicative inverse of $x$ in $\mathbb{F}_2[x]$. So the only missing property for $\mathbb{F}_2[x]$ is the existence of multiplicative inverse for any nonzero element, which is exactly the same situation for $\mathbb{Z}$.

**Exercise 11.** *Find the degree of the following polynomials*

- $(x^4 + x^3 + 1) \cdot (x^3 + x^2 + 1)$
- $(x^3 + x^2 + x) + (x^4 + x^3)$
- $(x^2 + x + 1) + (x^2 + x)$

*Give your comments on the degree of $f \cdot g$ and $f + g$, with $f, g \in \mathbb{F}_2[x]$.*

Since we can multiply polynomials, we can also raise them to powers, with the usual meaning

$$f^n = \underbrace{f \cdots f}_{n}$$

Consider for example the polynomial $f(x) = x + 1 \in \mathbb{F}_2[x]$ and suppose to compute its square power $(x + 1)^2$. This means multiplying $f$ by itself so

$$f(x)^2 = f(x) \cdot f(x) = (x+1) \cdot (x+1) = x^2 + x + x + 1 = x^2 + (1+1)x + 1 = x^2 + 1.$$

We notice that $x^2 + 1 = x^2 + (1)^2$, so we can write $(x + 1)^2 = x^2 + (1)^2$.

**Exercise 12.** *Check by examples that in general raising a polynomial in $\mathbb{F}_2$ to the power 2 is the same as raising to the power 2 its monomials and summing them.*

The property shown in exercise 12 is not shared by the analogous operation for real polynomials. Indeed, if we consider $q(x) = x + 1 \in \mathbb{R}[x]$, we have - as it is well-known since school - $q(x)^2 = (x + 1)^2 = x^2 + 2x + 1$, which is different from $x^2 + 1$.

**Example 1.** *When we learn at school how to raise the binomial $x+1$ to a power $\alpha$, we learn the construction of the so-called Pascal's (or Tartaglia's) Triangle. We can repeat the same construction on $\mathbb{F}_2[x]$:*
$(x + 1)^0 = 1 \to \mathbf{1}$
$(x + 1)^1 = x + 1 \to \mathbf{1\ 1}$
$(x + 1)^2 = x^2 + 1 \to \mathbf{1\ 0\ 1}$
$(x + 1)^3 = x^3 + x^2 + x + 1 \to \mathbf{1\ 1\ 1\ 1}$
$(x + 1)^4 = x^4 + 1 \to \mathbf{1\ 0\ 0\ 0\ 1}$
*and so on.*
*We have then constructed the Triangle:*

$$11\ 11\ 0\ 11\ 1\ 1\ 11\ 0\ 0\ 0\ 1 \tag{1}$$

**Exercise 13.** *Compute the following powers over $\mathbb{F}_2[x]$*

- $(x^2 + x)^2$
- $(x^3 + x + 1)^2$
- $(x + 1)^4$

- $(x^2 + x + 1)^3$

*Give your comments about powers of polynomials with even exponents.*

**Exercise 14.** *Provide a link between the Pascal's Triangle in $\mathbb{Z}[x]$ and the new version of the triangle defined here.*

Each polynomial $f \in \mathbb{F}_2[x]$ can be seen as a function $f : \mathbb{F}_2 \to \mathbb{F}_2$, such that

$$0 \mapsto f(0) \qquad 1 \mapsto f(1)$$

where $f(i), i \in \mathbb{F}_2$ is the bit we get by substituting $x$ with $i$ in $f$ and simplifying the obtained expression.

**Example 2.** *If $f_1 = x^3 + x + 1$ we get*

$$f_1(0) = 0^3 + 0 + 1 = 0 + 0 + 1 = 1$$

*and*

$$f_1(1) = 1^3 + 1 + 1 = 1 + 1 + 1 = 1.$$

*On the other hand, if we consider $f_2 = x^3 + x$ then*

$$f_2(0) = 0^3 + 0 = 0$$

*and*

$$f_2(1) = 1^3 + 1 = 0.$$

*Finally, if we take $f_3 = x^4 + 1$,*

$$f_3(0) = 0^4 + 1 = 1$$

*and*

$$f_3(0) = 1^4 + 1 = 0.$$

As shown by the above example, the evaluation of a polynomial $p$ in $i \in \mathbb{F}_2$ can return 0 or 1. If it returns 0 we say that $i$ is a *root* of $p$.
With the notation of example 2, neither 0 nor 1 are roots of $f_1$, while both 1 and 0 are roots of $f_2$ and only 1 is a root of $f_3$.

**Exercise 15.** *Compute the roots in $\mathbb{F}_2$ of the following polynomials in $\mathbb{F}_2[x]$:*

- $x^4 + x^3 + x$
- $x^3 + x + 1$
- $x^6 + x^5 + 1$
- $x^4 + x^3 + x^2 + x$.

**Exercise 16.** *How many functions $\mathbb{F}_2 \to \mathbb{F}_2$ exist?*
*To answer this question, we will follow the following steps:*

1. *as we already know, given a polynomial $p(x) \in \mathbb{F}_2[x]$, we can evaluate it at the elements of $\mathbb{F}_2$, i.e. we can compute $p(0)$ and $p(1)$, which clearly belong to $\mathbb{F}_2$, by mere substitution. Can you find two different polynomials $f, g \in \mathbb{F}_2[x]$ s.t. they have the same evaluations, i.e. $f(0) = g(0)$ and $f(1) = g(1)$?*

2. *Let us think now about the possible values we can have via evaluation. Given $f \in \mathbb{F}_2[x]$, we evaluate it at 0 and so $f(0)$ can be either 0 or 1 and, in the same way, $f(1)$ can be either 0 or 1.*
   *We summarize in the following table the values a polynomial can assume if evaluated in $\mathbb{F}_2$: As an example, if$p = x^2 + x$, then $p(0) = 0$ and $p(1) = 0$*

| a | p(a) | q(a) | r(a) | s(a) |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |

Table 3: All the possible evaluations

   *as in column 2 of the table; if $q = x^2 + x + 1$, then $q(0) = q(1) = 1$ as in the third column. If $r = x^3$, we have $r(0) = 0$ and $r(1) = 1$ as in column four. Finally, if $s = x^2 + 1$ we have $s(0) = 1$ and $s(1) = 0$, as in the last column of the table.*
   *Can you find other polynomials $p', q', r', s' \in \mathbb{F}_2[x]$* ***of minimal degree*** *such that, evaluated in $0, 1$ behave exactly as $p, q, r$ and $s$?*

3. *Looking at Table 3, guess how many functions $\mathbb{F}_2 \to \mathbb{F}_2$ exist.*

There is a property for polynomials that plays a special role in cryptography. We introduce it formally in the following definition and then we discuss it.

**Definition 1.** *A nonzero polynomial $f \in \mathbb{F}_2[x]$ of degree $\geq 1$ is called* reducible *if there exist $q, r \in \mathbb{F}_2[x]$ of* ***positive*** *degree such that their product is $f$, i.e.*

$$f = q\,r\,.$$

*If no polynomials of this form exist, then $f$ is an* irreducible *polynomial.*

Polynomials of degree 1 are clearly irreducible.

**Example 3.** *Consider the polynomial $f = x^4 + x^2 + x \in \mathbb{F}_2[x]$. Clearly $f = x(x^3 + x + 1)$ and so $f$ is reducible.*
*We also note that $f(0) = 0$, so 0 is a root of $f$.*

A famous theorem due to Ruffini ensures that a polynomial $p \in \mathbb{F}_2[x]$ has a root $a \in \mathbb{F}_2$ if and only if it can be written as $p = (x+a)q$ for a certain $q \in \mathbb{F}_2[x]$. A special case is when $f = (x + a)$, in this case $q = 1$. In all other cases, $p$ is then irreducible.
For low-degree polynomials we can strenthen Ruffini's theorem in the following theorem.

**Theorem 1.** *A polynomial $p \in \mathbb{F}_2[x]$ of degree 2 or 3 is reducible if and only if it has at least a root in $\mathbb{F}_2$.*
Or, equivalently,
*A polynomial $p \in \mathbb{F}_2[x]$ of degree 2 or 3 is irreducible if and only if it has no roots in $\mathbb{F}_2$.*

**Example 4.** *Consider the polynomial $p(x) = x^3 + x + 1$. Its degree is 3 and we have $p(0) = p(1) = 1$ so $p$ has no roots in $\mathbb{F}_2$. By Theorem 1 it is irreducible. On the other hand $q(x) = x^2 + 1$ has degree 2 and it holds $p(0) = 1$, $p(1) = 0$, so 1 is a root of $q(x)$. Indeed $q(x)$ is reducible and it can be decomposed as $q(x) = (x + 1)(x + 1) = (x + 1)^2$.*

**Exercise 17.** *Factorize the following polynomials in $\mathbb{F}_2[x]$:*

- $p_1 = x^5 + x^4 + x + 1$
- $p_2 = x^4 + x^2 + 1$

*Hint: try all possible irreducible factors of degree 1 and 2.*

**Exercise 18.** *Find all the irreducible polynomials of degree 2 in $\mathbb{F}_2[x]$.*
*Use this result to prove that $f = x^4 + x + 1$ is irreducible in $\mathbb{F}_2[x]$.*

In order to complete our study on polynomial operations, we have to define one more operation: *the division of polynomials.*
A consequence of more advanced theory for polynomials with coefficients in a field is the following. If we consider two polynomials $f$ and $g$ in $\mathbb{F}_2[x]$, we can find two other polynomials, say $q, r$, such that $f = g \cdot q + r$ and $\deg(r) < \deg(g)$. The polynomial $q$ is called *quotient*, whereas $r$ is the **remainder**. If $r = 0$ then we say that $g$ *divides* $f$, or that we have performed an *exact division* between $f$ and $g$.

**Example 5.** *Let us consider $f = x^2 + 1$, $g = x$ in $\mathbb{F}_2[x]$; we clearly have $q = x$ and $r = 1$:*

$$x^2 + 1 = x \cdot x + 1.$$

*If we consider $f = 1 + x^2$, $g = x + 1$ in $\mathbb{F}_2[x]$, we note that $x^2 + 1 = (x + 1) \cdot (x + 1)$, so $q = g$ and $r = 0$.*

**Example 6.** *Let us consider $f = x$, $g = x^2 + 1$ in $\mathbb{F}_2[x]$; this case is special since we cannot perform any division and we obviously have $q = 0$ and $r = f$.*

Let us now see how to compute the quotient and the reminder; we will use $f = x^2 + 1$, $g = x$ in $\mathbb{F}_2[x]$ as a simple example.
We start reordering the monomials in $f$ and the monomials in $g$ in decreasing order by degree.

$$f = x^2 + 1,\ g = x$$

Then we put the polynomials in a table, as follows

| $x^2$ | $+0x$ | $+1$ | $x$ |
|---|---|---|---|
| | | | |

If in $f(x)$ a term of degree smaller than $\deg(f)$ does not appear, we put it in the table with coefficient **zero**.
Now we divide the term of maximal degree of $f$ by that of $g$, getting a new monomial $m$; in our particular case $m = x$

| $x^2$ | $+0x$ | $+1$ | $x$ |
|---|---|---|---|
| | | | $x$ |

Then, we multiply $m \cdot g$ and we insert the result in the table under $f$, so that the terms of same degree are lined-up.

| $x^2$ | $+0x$ | $+1$ | $x$ |
|---|---|---|---|
| $x^2$ | $+0x$ | $+0$ | $x$ |

We sum the lined-up monomials (remember that the coefficients belong to $\mathbb{F}_2$, so $1 + 1 = 0$!!!!); the obtained sum is the *partial reminder* of our division.

| $x^2$ | $+0x$ | $+1$ | $x$ |
|---|---|---|---|
| $x^2$ | $+0x$ | $+0$ | $x$ |
| $0x^2$ | $+0x$ | $+1$ | $x$ |

If the partial remainder's degree is greater than or equal to the degree of $g$, *then* repeat the procedure; otherwise, stop and observe that

- the polynomial under $g$ is the quotient $q$
- the partial remainder is the (final) remainder $r$

In our example $q = x$ and $r = 1$.

**Exercise 19.** *Perform the division between the following polynomials in* $\mathbb{F}_2[x]$

- $f = x^5 + 1$, $g = x^3 + x + 1$
- $f = x^3 + x^2 + 1$, $g = x^3 + 1$;
- $f = x^4 + x$, $g = x + 1$;
- $f = x^5 + x^3 + 1$, $g = x^2 + x + 1$;
- $f = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$, $g = x^3 + x^2 + 1$.

*Which of them are exact divisions?*

## 0.3 Vectors of Bits

Usually, computers, have to work with sequences of bits, that are "ordered set". These sequences, called *vectors* by Mathematicians, are commonly defined *arrays* or *strings* by Computer Scientists.
Some of these strings are particularly important for applications, so they have their own name. They are:

- *nibbles*: vectors of 4 bits;
- *bytes*: vectors of 8 bits

In general we call *stream* a vector whose exact size is not known *a priori*
Streams can be visualized as blocks of one-bit bricks:

| 0 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|

or they can be represented between "(" and ")" with the bits separated by ",". That is

| 0 | 1 | 1 |
|---|---|---|

is $(0, 1, 1)$. The set of all vectors of a fixed length $n$ is

$$(\mathbb{F}_2)^n.$$

We will use both of these representations for convenience.
The operations on $\mathbb{F}_2$-vectors can be extended componentwise, that is if we sum two vectors bit by bit with respect to the position we have:

$$(0, 1, 1) + (1, 0, 1) = (1, 1, 0). \tag{2}$$

**Exercise 20.** *Every vector has an opposite with respect to the sum. Are you able to find it? Try first with* $(0, 0, 0, 0)$, $(0, 1, 0, 1), (1, 1, 1, 1), (1, 1, 0, 0)$ *and then derive a general rule.*

We can also extend componentwise the idea of *multiple*: the multiple of a vector $b$ with respect to an integer $a$ is obtained by multiplying by $a$ each component of $b$. Here is an example

$$3 \cdot (1, 0, 1) = (3 \cdot 1, 3 \cdot 0, 3 \cdot 1) = (1, 0, 1).$$

**Exercise 21.** *Compute* $5 \cdot (1, 1, 0)$ *and* $6 \cdot (0, 1, 0)$.
*Can you derive a rule to compute the multiple of a vector? (see exercise* **??***).*

Another structure useful to manage ordered sets, like vectors, is the set of polynomials described in Section 0.2. We recall that the sum of two polynomials, $x + 1, x^2 + 1 \in \mathbb{F}_2[x]$, is

$$(x + 1) + (x^2 + 1) \; = \; x^2 + x \, . \tag{3}$$

**Exercise 22.** *Do you see any connections between (2) and (3)?*

As you have seen in the previous exercise, nice and meaningful operations on vectors can be interpreted as operations on polynomials. In the rest of the section we give other examples.

One of the most important and fast operations for a computer (and more precisely in its Central Processing Unit, CPU) consists in *shifting* to the left or to the right of one position a vectors of bits (register). For example shifting to the right (or to the left) we have:

$$(1,1,0,1,0) \Longleftrightarrow (0,1,1,0,1). \tag{4}$$

The polynomials associate to (4) are

$$x^4 + x^3 + x \Longleftrightarrow x^3 + x^2 + 1. \tag{5}$$

That is, shifting to the left a vector seems to correspond to the multiplication of its polynomial by $x$, while shifting to the right seems to correspond to the division of its polynomial by $x$.

**Exercise 23.** *Suppose you want to double a vector of $n$ bits. For example let $n = 3$ and we want that $(1,0,1)$ becomes $(1,0,1,1,0,1)$. Using the polynomial notation for the vector and multiplying by another polynomial $p$ you can obtain the result expected. Who is $p$?*

Suppose that the fourth bit (the one with "?" symbol) in a vector $v$,

$$v = (?, 1, 1, 0),$$

is computed by the sum of the first and the third. In languages as "C" we can say

$$v[3] = v[2] + v[0];$$

where $v[0]$ represents the first entry of the vector and $v[2]$ the third one. In a polynomial notation this "function" becomes

$$x^3 = x^2 + x^0$$

with the exponents representing the positions of the vector entries.

Let us consider the vector $(0,1,1) \in (\mathbb{F}_2)^3$, associated to the polynomial $f = x^3+x+1$ as explained above. We observe that this polynomial is irreducible.

We can construct a Linear Feedback Shift Register (LFSR) over three bits, **using** $f = x^3 + x + 1$.
First of all, we start with an initial vector (called *the initial state*), for example $(1,0,1)$, inserting it in the following structure:

| ? | $\longrightarrow\longrightarrow$ | 1 | 0 | 1 |
|---|---|---|---|---|

We can associate a monomial to each position in the structure

| $x^3$ | $\longrightarrow\longrightarrow$ | $x^2$ | $x$ | 1 |
|---|---|---|---|---|

We aim to compute the mysterious bit, that is, the element we indicated with ?. This bit is in correspondence to the monomial $x^3$ and since we want to use the polynomial $f$, we must consider the portion of f without its leading term $x^3$, that is, $f - x^3 = x + 1$.
Therefore, we interpret $x + 1$ as the following operation:

- we read the bits in positions x and 1, which are 0 and 1, respectively,
- we sum these two bits, obtaining $0 + 1 = \mathbf{1}$,
- we finally insert this new bit **1** in the place of the mysterios bit ?.

| **1** | $\longrightarrow\longrightarrow$ | 1 | 0 | 1 |
|---|---|---|---|---|

Now, we shift the vector to the right:

| ? | $\longrightarrow\longrightarrow$ | **1** | 1 | 0 |
|---|---|---|---|---|

so we have a new three-bit vector, that we call again state.
We now need to compute the new mysterious value ?, and so we proceed with the same algorithm, by taking the two bits at positions $x$ (i.e., 1) and 1 (i.e., 0). We will then get $1 + 0 = 1$ and so

| 1 | $\longrightarrow\longrightarrow$ | 1 | 1 | 0 |
|---|---|---|---|---|

which becomes, after another shift to the right, the following

| ? | $\longrightarrow\longrightarrow$ | 1 | 1 | 1 |
|---|---|---|---|---|

We repeat again the mysterious bit computation and we get $1 + 1 = 0$, meaning

| 0 | $\longrightarrow\longrightarrow$ | 1 | 1 | 1 |
|---|---|---|---|---|

and the shift to the right, as follows

| ? | $\longrightarrow\longrightarrow$ | 0 | 1 | 1 |
|---|---|---|---|---|

and again the mysterious bit computation $1 + 1 = 0$, that is,

| 0 | $\longrightarrow\longrightarrow$ | 0 | 1 | 1 |
|---|---|---|---|---|

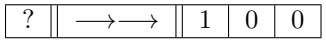
and again the shift to the right, obtaining

| ? | $\longrightarrow\longrightarrow$ | 0 | 0 | 1 |
|---|---|---|---|---|

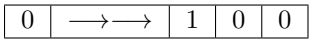
and again the mysterious bit computation $0 + 1 = 1$ with

| 1 | $\longrightarrow\longrightarrow$ | 0 | 0 | 1 |
|---|---|---|---|---|

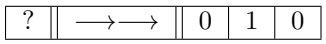
and the shift to the right



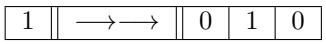

and again the mysterious bit computation $0 + 0 = 0$



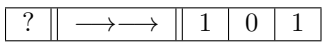

and again the shift to the right




and, finally $1 + 0 = 1$ and




with the last shift




We notice the following facts:

(1) After having performed the above algorithm seven times, we get again the initial vector $(1, 0, 1)$.

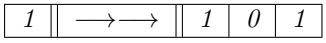
(2) While performing the above algorithm, we find all seven nonzero 3-bit strings

Fact (1) is a property that is shared by many LFSR's. It is enough to use any polynomial $f \in \mathbb{F}_2[x]$ of the form $f = x^n + \ldots + 1$ to construct an LFSR producing $n$-bit vectors and such that any initial state is got again after a finite number of iterations.
Fact (2) is more difficult to generalize. If we want an LFSR that can start from any initial nonzero vector and obtain all other nonzero vectors, then we must use very special polynomials, called *primitive* polynomials.

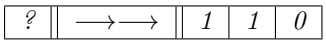
**Definition 2.** *We call primitive any polynomial $f \in \mathbb{F}_2[x]$ of degree $n$ that, used as feedback polynomials of a LFSR, can generate all the non-zero $n$-tuples of bits,i.e. all the $n$-tuples but $(0, 0, 0, ..., 0, 0)$, using as initial state any of these.*

An example of primitive polynomial was that used in the previous LFSR: $x^3 + x + 1$.

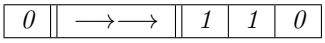
**Example 7.** *We see now an example of non-primitive polynomial. Let us consider $f = x^3 + 1$ and we start again from $(1, 0, 1)$, and we perform the algorithm*



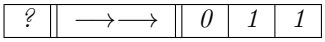

*getting*



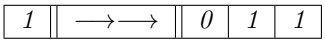

*Then*



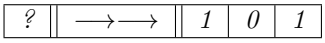

*from which*




*Then*



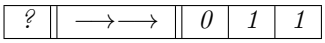

*finally getting*



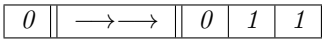

**Example 8.** *Let us consider a polynomial which is* ***not*** *of the form* $f = x^n + \ldots + 1$*, for example*

$$f = x^3 + x^2\,,$$

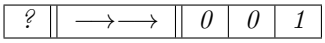
*using it to construct an LFSR. The portion of* $f$ *without its leading term is only formed by* $x^2$*, and so no computation is needed to obtain the mysterious bit. We take the following initial state*



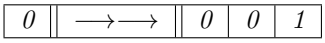

*and we take the bit in the position represented by* $x^2$*, i.e.* 0 *as the mysterious bit:*



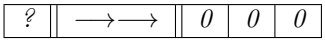

*After a shift to the right, we obtain*




*The mysterious bit turns out to be* 0 *again, so we have*




*and, after a shift to the right, we get*




*From now on, we are stuck; indeed the mysterious bit will always be equal to* 0 *and so, after the shift to the right, we will always obtain* $(0, 0, 0)$*.*

We recall the notion of irreducible polynomial (Definition 1). An irreducible polynomial cannot be written as product of two non-trivial factors. It turns out that there is a close links between primitive polynomials and irreducible polynomials, as shown in the following theorem.

**Theorem 2.** *If* $p \in \mathbb{F}_2[x]$ *is primitive then* $p$ *is irreducible.*

The previous theorem cannot be inverted. For example, the polynomial $x^4 + x^3 + x^2 + x + 1$ is irreducible but it is not primitive.

**Exercise 24.** *Verify that* $x^4 + x^3 + x^2 + x + 1$ *is not primitive via a LFSR.*

## 0.4 Multivariate polynomials on bits

In many cryptographic applications, polynomials in only one variable are not enough and we need to deal with polynomials in two or more variables. In this section, we define such polynomials and we see how to sum and multiply them.

We begin with two variables, $x$ and $y$.
A *term* or *monomial* in the two variables $x$ and $y$ is a product of powers, i.e. $x^i y^h$, for some $i, h$ in $\mathbb{N}$.
For example, we can consider the following monomials

$$x^2y^3\,(i=2,h=3), \quad x^4\,(i=4,h=0), \quad y^7\,(i=0,h=7), \quad 1\,(i=h=0).$$

Be careful that for example $x^{-4}$ and $xy^{-1}$ are **not** monomials.
With monomials, we can define polynomials in the two variables $x$ and $y$ and coefficients in $\mathbb{F}_2$ (i.e. multivariate polynomials in the field of bits) as sums of monomials (without coefficients). For example, the following are polynomials

$$x^2y^3 + x + x^{10} + y, \quad xy + x^{11}y^2 \quad \text{and} \quad x^3y^5$$

(a monomial is also a polynomial).
Formally, a polynomial in the two variables $x$ and $y$ and coefficients in $\mathbb{F}_2$ is any expression of the form

$$f(x,y) = \sum_{i,h\in\mathbb{N}} a_{i,h} x^i y^h$$

such that

- for each $i, h$ in $\mathbb{N}$, $a_{i,h} \in \mathbb{F}_2$,
- only a **finite** number of coefficients is nonzero.

The set containing all polynomials in $x, y$ with bits as coefficient is denoted by $\mathbb{F}_2[x, y]$.

The $x$-degree of a term $x^i y^h$ in the two variables $x, y$ is the value $i$, whereas $h$ is its $y$-degree. To be more precise

$$\deg_x(x^i y^h) = i,\ \deg_y(x^i y^h) = h\,.$$

The *total degree* (or, simply, degree) of a term $x^i y^h$ in the two variables $x, y$ is the sum of its $x$-degree and its $y$-degree, i.e.

$$\deg(x^i y^h) \;=\; i + h\,.$$

If we consider $xy^5$, then $\deg_x(xy^5) = 1$, $\deg_y(xy^5) = 5$, and $\deg(xy^5) = 6$.
The *degree of a polynomial* $f \in \mathbb{F}_2[x, y]$ is the highest degree of the monomials appearing in $f$ with nonzero coefficient, so, if $f = x^3y + xy^6 - y^2$, then $\deg(f) = \deg(xy^6) = 7$, and if $g = x^3 + y$ then $\deg(g) = \deg(x^3) = 3$.

**Exercise 25.** *In $\mathbb{F}_2[x,y]$ what is...*

- *the $y$-degree of $xy^6$?*
- *the degree of $xy^5+1$?*
- *the degree of $x^7+x^4y^3+y^7$?*

Two polynomials $f,g \in \mathbb{F}_2[x,y]$ are called *equal* if

$$\begin{gathered} \textit{a term } x^iy^j \textit{ appears in } f \textit{ with nonzero coefficient } a_{i,j} \\ \textit{if and only if} \\ x^iy^j \textit{ appears in } g \textit{ with coefficient } a_{i,j}. \end{gathered}$$

Then, we can see that, in $\mathbb{F}_2[x,y]$, $x^3+0y^{12}=x^3=x^3+0xy^{32}$.
Similarly to what presented in section 0.2 for univariate polynomials, the sum and the product of polynomials in $\mathbb{F}_2[x,y]$ are defined analogously to real polynomials, but of course we must take into account that their coefficients are bits. The following examples will clarify this point.

**Example 9.** *In $\mathbb{F}_2[x,y]$, let $f_1=x^3y+xy+1$, $f_2=y^3+xy+1$ and $f_3=xy+1$. It holds*

- $f_1+f_2=(x^3y+xy+1)+(y^3+xy+1)=x^3y+y^3+2xy+2=x^3y+y^3$
- $f_3f_1=(xy+1)(x^3y+xy+1)=x^4y^2+x^2y^2+x^3y+1$

**Exercise 26.** *Compute the following sums and products in $\mathbb{F}_2[x,y]$ and find the degree of the final result:*

- $(xy^3+y)+(xy^3+x^2)$
- $(x+y+y^3)(x+y+y^3)$
- $\big((x+y)(xy^3+y)\big)+(xy+y^2+1)$
- $\big((x+1)(y+y^3x)\big)+\big((xy+y^2+y+1)(x+1)\big)$

Since we can multiply polynomials in $x,y$, we can raise them at a power $m$, with the usual meaning: $f \mapsto f^m$.
If, for example, we take the polynomial $f=x^3+y+1 \in \mathbb{F}_2[x,y]$, we have that

$$f^2=(x^3+y+1)^2=f\cdot f=(x^3+y+1)(x^3+y+1)=$$

$$x^6+x^3y+x^3+x^3y+y^2+y+x^3+y+1=x^6+y^2+1=(x^3)^2+(y)^2+(1)^2.$$

As in the univariate case, raising a multivariate polynomial in $\mathbb{F}_2[x,y]$ to the power 2 is the same as raising its monomials to the power 2 and summing them.

We consider now the case of three variables, $x$, $y$ and $z$.
A *term* or *monomial* in the three variables $x$, $y$ and $z$ is a product of powers,

i.e. $x^i y^h z^l$, for some $i, h, l$ in $\mathbb{N}$.
For example, we can consider the following monomials

$$x^2y^3z\,(i=2,h=3,l=1),\quad x^4\,(i=4,h=l=0),\quad z^9\,(i=h=0,l=9),\quad 1\,(i=h=l=0).$$

With monomials, we can define polynomials in the variables $x$, $y$ and $z$ and coefficients in $\mathbb{F}_2$ as sums of monomials (without coefficients). For example, the following are polynomials

$$x^2y^3 + x + x^{10} + z,\quad xy + z^{11}y^2,\quad xyz + z^2 + 1 \text{ and } x^3y^5z$$

(a monomial is also a polynomial).
Formally, a polynomial in the three variables $x$, $y$ and $z$ and coefficients in $\mathbb{F}_2$ is any expression of the form

$$f(x,y,z) = \sum_{i,h,l\in\mathbb{N}} a_{i,h,l}x^i y^h z^l$$

such that

- for each $i, h, l$ in $\mathbb{N}$, $a_{i,h,l} \in \mathbb{F}_2$,
- only a **finite** number of coefficients is nonzero.

The set containing all polynomials in $x, y, z$ with bits as coefficient is denoted by $\mathbb{F}_2[x, y, z]$.

The $x$-degree of a term $x^i y^h z^l$ in the three variables $x, y, z$ is the value $i$, whereas $h$ is its $y$-degree and $l$ its $z$-degree. To be more precise

$$\deg_x(x^i y^h z^l) = i,\ \deg_y(x^i y^h z^l) = h,\ \deg_z(x^i y^h z^l) = l\,.$$

The *total degree* (or, simply, degree) of a term $x^i y^h z^l$ in the three variables $x, y, z$ is the sum of its $x$-degree, its $y$-degree and its $z$-degree, i.e.

$$\deg(x^i y^h z^l) \;=\; i + h + l\,.$$

If we consider $xy^5z^3$ then $\deg_x(xy^5z^3) = 1$, $\deg_y(xy^5z^3) = 5$, $\deg_z(xy^5z^3) = 3$, and $\deg(xy^5z^3) = 9$.
The *degree of a polynomial* $f \in \mathbb{F}_2[x, y, z]$ is the highest degree of the monomials appearing in $f$ with nonzero coefficient, so, if $f = x^3z + xy^8 - z^2y$, then $\deg(f) = \deg(xy^8) = 9$, and if $g = xz^3 + xyz$ then $\deg(g) = 4$.

**Exercise 27.** *In $\mathbb{F}_2[x, y, z]$ what is...*

- *the $y$-degree of $xy^6$?*
- *the $z$-degree of $xyz^3$*
- *the degree of $xy^5 + z + y^4 + 1$?*

- *the degree of* $z^6 + x^3y^3z + z^7 + y^2 + x + 1$*?*

Two polynomials $f, g \in \mathbb{F}_2[x, y, z]$ are called *equal* if

$$\begin{gathered} \textit{a term } {x_1}^i y^h z^l \textit{ appears in } f \textit{ with nonzero coefficient } a_{i,h,l} \\ \textit{if and only if} \\ {x_1}^i y^h z^l \textit{ appears in } g \textit{ with coefficient } a_{i,h,l}. \end{gathered}$$

Then, $x^3 + 0y^{12} = x^3 = x^3 + 0xyz^{32}$ and $xyz = 0x^3 + xyz + 0y^4 + 0z$.

The sum and the product of polynomials in $\mathbb{F}_2[x, y, z]$ follow the usual rules.

**Example 10.** *In* $\mathbb{F}_2[x, y, z]$*, let* $f = x^3z + 1$*,* $g = y^3 + xyz + 1$ *and* $h = xyz$*. It holds*

- $f + g = (x^3z + 1) + (y^3 + xyz + 1) = x^3z + y^3 + xyz$
- $hf = (xyz)(x^3y + xy + 1) = x^4y^2z + x^2y^2z + xyz$

**Exercise 28.** *Compute the following sums and products in* $\mathbb{F}_2[x, y, z]$ *and find the degree of the final result:*

- $(xy^3z + y + z) + (xy^3 + x^2z)$
- $(x + yz + y^3)(xz + y + y^3)$
- $\big((xz + y)(xyz^3 + y)\big) + (xy + y^2 + 1)$
- $\big((x + z + 1)(y + y^3x)\big) + \big((xy + y^2z + y + 1)(xz + 1)\big)$

Since we can multiply polynomials in $x, y, z$, we can raise them at any power. If, for example, we take the polynomial $f = x^3z + y + z + 1 \in \mathbb{F}_2[x, y, z]$, we have that

$$f^2 = (x^3z + y + z + 1)^2 = f \cdot f = (x^3z + y + z + 1)(x^3z + y + z + 1)$$

$$x^6z^2 + y^2 + z^2 + 1 = (x^3z)^2 + (y)^2 + (z)^2 + 1^2.$$

As in the univariate case, raising a multivariate polynomial in $\mathbb{F}_2[x, y, z]$ to the power 2 is the same as raising to the power 2 its monomials and summing them.

Now we are ready to tackle the generic case of $n$ variables: $x_1, x_2, ..., x_n$
A *term* or *monomial* in the $n$ variables $x_1, ..., x_n$ is a product of powers of $x_1, ..., x_n$, i.e. ${x_1}^{i_1} \cdots x_n^{i_n}$ for some $i_1, ..., i_n \in \mathbb{N}$.
For example, for three variables $x_1, x_2, x_3$, it is clear that ${x_1}^3{x_2}^8{x_3}^6$, ${x_2{x_3}^4} = {{x_1}^0x_2{x_3}^4}$ ,${{x_1}^4} = {{x_1}^4{x_2}^0{x_3}^0}$ are all terms.
With terms, we can define polynomials in $n$ variables $x_1, ..., x_n$ and coefficients in $\mathbb{F}_2$ (i.e. multivariate polynomials in the field of bits) as expressions of the form

$$f(x_1, ..., x_n) = \sum_{i_1,...,i_n \in \mathbb{N}} a_{i_1,...,i_n} {x_1}^{i_1} \cdots x_n^{i_n}$$

such that

- for each $i_1, ..., i_n \in \mathbb{N}$, $a_{i_1,...,i_n} \in \mathbb{F}_2$,
- only a finite number of coefficients is nonzero.

The set containing all polynomials in $n$ variables $x_1, ..., x_n$ with coefficients in $\mathbb{F}_2$ is denoted by $\mathbb{F}_2[x_1, ..., x_n]$.
Note that

- ${x_2}^2 x_3 + x_1$ is a polynomial in $\mathbb{F}_2[x_1, x_2, x_3]$; but it is **not** a polynomial in $\mathbb{F}_2[x_1, x_2]$;
- $x_1 - \frac{1}{2}x_2$ is not a polynomial in $\mathbb{F}_2[x_1, x_2]$;
- $x_1 x_2 {x_3}^3 + 1$ is a polynomial in $\mathbb{F}_2[x_1, x_2, x_3]$.

When (as we have done before) we will deal with two or three variables, we may denote them as $x, y$ or $x, y, z$.
For $1 \leq j \leq n$, the *$x_j$-degree* of a term in $n$ variables ${x_1}^{i_1} \cdots x_n^{i_n}$ is the value $i_j$.
To simplify the expression we can use $\deg_j = \deg_{x_j}$ i.e.

$$\deg_{x_j}({x_1}^{i_1} \cdots x_n^{i_n}) = \deg_j({x_1}^{i_1} \cdots x_n^{i_n}) = i_j.$$

The *total degree* (or, simply, degree) of a term $t$ in $n$ variables $t = {x_1}^{i_1} \cdots x_n^{i_n}$ is the sum of all $\deg_j(t)'$s for all $1 \leq j \leq n$, i.e.

$$\deg({x_1}^{i_1} \cdots x_n^{i_n}) = \sum_{j=1}^{n} \deg_j({x_1}^{i_1} \cdots x_n^{i_n}) = i_1 + ... + i_n.$$

If we consider $x_1 {x_3}^5 {x_4}^2$, then $\deg_1(x_1 {x_3}^5 {x_4}^2) = 1$, $\deg_2(x_1 {x_3}^5 {x_4}^2) = 0$, $\deg_3(x_1 {x_3}^5 {x_4}^2) = 5$, $\deg_4(x_1 {x_3}^5 {x_4}^2) = 2$ and $\deg(x_1 {x_3}^5 {x_4}^2) = 8$.
The *degree of a polynomial* $f \in \mathbb{F}_2[x_1, ..., x_n]$ is the highest degree of the monomials appearing in $f$ with nonzero coefficient, so, if $f = {x_1}^3 x_2 + x_1 {x_2}^6 - {x_3}^{10} x_2$, then $\deg(f) = \deg({x_3}^{10} x_2) = 11$.

**Exercise 29.** *In $\mathbb{F}_2[x_1, x_2 x_3, x_4]$ what is...*

- *the 2-degree of ${x_2}^3 x_3$?*
- *the 4-degree of ${x_2}^3 x_3$?*
- *the degree of ${x_2}^3 x_3 + x_2 x_4$?*
- *the degree of ${x_1}^7 + {x_2}^4 {x_3}^3 + {x_4}^3 x_1$?*

Two polynomials $f, g \in \mathbb{F}_2[x_1, ..., x_n]$ are called *equal* if

*a term ${x_1}^{i_1} \cdots x_n^{i_n}$ appears in $f$ with nonzero coefficient $a_{i_1 ... i_n}$*
*if and only if*
*${x_1}^{i_1} \cdots x_n^{i_n}$ appears in $g$ with coefficient $a_{i_1 ... i_n}$ as well.*

Note that $x^3 + 0y^{12} = x^3 = x^3 + 0xy^{32}$ and $x_5x_4 - x_2 = x_5x_4 + 0x_3 - x_2$.
The sum, the product and the powers of polynomials in $\mathbb{F}_2[x_1, ...x_n]$ follow the usual rules.
If, for example, we take the polynomial $f = x_3x_1 + x_4 + 1$, we have that

$$f^2 = (x_3x_1 + x_4 + 1)^2 = f \cdot f = (x_3x_1 + x_4 + 1)(x_3x_1 + x_4 + 1)$$

$$x_3{}^2x_1{}^2 + x_4{}^2 + 1 = (x_3x_1)^2 + (x_4)^2 + 1^2.$$

**Exercise 30.** *In* $\mathbb{F}_2[x_1, x_2, x_3, x_4, x_5]$ *compute*

- $(x_1{}^{12}x_5 + x_3{}^3 + x_4 + x_2{}^6x_3) + (x_4{}^5 + x_3{}^3 + x_2{}^6x_3 + x_5{}^6x_1)$
- $(x_1{}^{10}x_2{}^6 + x_3x_4x_5 + x_2{}^5x_5)(x_1{}^3x_4 + x_2x_5 + x_3{}^5)$
- $(x_1 + x_5)^3$

## 0.5 Boolean Functions

In this section, we will introduce some functions that are very important in information theory and in cryptography, the so-called *Boolean functions*.

First we consider some examples. We will need polynomials with three variables and bits as coefficients, that we call $\mathbb{F}_2[x, y, z]$.

**Example 11.** *For a given vector of bits $S$, for example $S = (\bar{x}, \bar{y}, \bar{z}) \in (\mathbb{F}_2)^3$, we define the "parity bit" as a new bit which takes the values 1 or 0 depending on the number of bits in $S$ holding value 1. If this number is odd, then the parity bit takes the value 1, 0 otherwise. To compute the parity bit we can use the following polynomial function $p$*

$$p : (\mathbb{F}_2)^3 \to \mathbb{F}_2, \quad (\bar{x}, \bar{y}, \bar{z}) \mapsto \bar{x} + \bar{y} + \bar{z},$$

$$p \in \mathbb{F}_2[x, y, z], \qquad p = x + y + z.$$

**Exercise 31.** *Verify that the function $p(x, y, z)$ returns the parity bit.*

**Example 12.** *We are interested in a function $f$ that returns 1 if a vector of bits is null, 0 otherwise. This is a useful Boolean function, since it recognizes whether a vector of bits is the null vector $(0, \ldots, 0)$.*
*We now show how to obtain a polynomial representing this function. We observed in Exercise 5 that the multiplication of bits has the same truth table of the* AND *operator. In particular we have 1 if each bit is 1, 0 otherwise. This is the opposite of what we want and so we can add 1 to have the sought-after function (see Exercise 6). For example, if $n = 3$ the function becomes*

$$f : (\mathbb{F}_2)^3 \to \mathbb{F}_2, \quad (\bar{x}, \bar{y}, \bar{z}) \mapsto \bar{x}\bar{y}\bar{z} + 1,$$

$$f \in \mathbb{F}_2[x, y, z], \qquad f = xyz + 1.$$

**Example 13.** *In Exercise 5 we observed that the operator* OR *corresponds to neither the sum in $\mathbb{F}_2$ nor the multiplication in $\mathbb{F}_2$. We would like to define a Boolean function*

$$o : (\mathbb{F}_2)^2 \to \mathbb{F}_2,\ o \in \mathbb{F}_2[x, y],\ (x, y) \mapsto o(x, y),$$

*that corresponds to the* OR *operator. We need the De Morgan law, which is used in logic:*

$$\mathrm{NOT}(x \,\mathrm{OR}\, y) = \mathrm{NOT}(x) \,\mathrm{AND}\, \mathrm{NOT}(y)$$

*From this we deduce that*

$$o(x, y) = \mathrm{NOT}(\mathrm{NOT}(x \,\mathrm{OR}\, y)) = \mathrm{NOT}(\mathrm{NOT}(x) \,\mathrm{AND}\, \mathrm{NOT}(y)).$$

*Since the operator* AND *corresponds to multiplication, the above formula becomes*

$$o(x, y) = \big((x + 1)(y + 1)\big) + 1,$$

*which is (after simplifications)*

$$o(x, y) = xy + x + y.$$

**Exercise 32.** *Find the Boolean function corresponding* $x \,\mathrm{OR}\, (y \,\mathrm{OR}\, z)$.

Let $f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$ be a polynomial function such that $f(x, y, z) = xy + yz$. Evaluating $f$ at the 3-tuple $(0, 0, 0)$ we get $f(0, 0, 0) = 0+0 = 0$, and evaluating it in $(1, 1, 1)$ we get $f(1, 1, 1) = 1 \cdot 1 + 1 \cdot 1 = 1 + 1 = 0$.
Note that the input of $f$ is a 3-tuple of bits and its output is only one bit.
Let us now consider also another function $g : (\mathbb{F}_2)^3 \to \mathbb{F}_2$, $g(x, y, z) = x^2y + yz$. We evaluate $g$ at the same 3-tuples $(0, 0, 0)$ and $(1, 1, 1)$ and we obtain that $g(0, 0, 0) = f(0, 0, 0) = 0$ and $g(1, 1, 1) = f(1, 1, 1) = 0$. Indeed, you can check that $f$ and $g$ take the same values in *all* vectors of $(\mathbb{F}_2)^3$, so that

$$f(\bar{x}, \bar{y}, \bar{z}) = g(\bar{x}, \bar{y}, \bar{z}),\ \forall (\bar{x}, \bar{y}, \bar{z}) \in (\mathbb{F}_2)^3.$$

Hence, $f$ and $g$ are **distinct** as polynomials in $\mathbb{F}_2[x, y, z]$, but they are **equal** as Boolean functions, since given an input, they return the same output!

The unexpected behaviour of the above functions $f$ and $g$ comes from the fact that $u^2 = u$ for any $u \in \mathbb{F}_2$. Therefore, every polynomial function from $(\mathbb{F}_2)^3$ to $\mathbb{F}_2$ can be written as a polynomial $h \in \mathbb{F}_2[x, y, z]$ where $0 \leq \deg_x(h), \deg_y(h), \deg_z(h) \leq 1$, for example, $x^3y^2 + z^{10}xy^2 = xy + zxy$, as a Boolean function.

**Definition 3.** *A* squarefree *monomial in* $\mathbb{F}_2[x, y, z]$ *is a monomial where each variable appears with exponent at most one.*

Even more generally, it is possible to prove that **every function** from $(\mathbb{F}_2)^3$ to $\mathbb{F}_2$ can be written in a polynomial form, called the Absolute Normal Form (ANF), where every monomial is squarefree.

**Definition 4.** *Let* $f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$. *The* Absolute Normal Form *of* $f$ *is*

$$f(x, y, z) = axyz + bxy + cxz + dzy + \alpha x + \beta y + \gamma z + \delta,$$

*where* $a, b, c, d, \alpha, \beta, \gamma, \delta$ *are in* $\mathbb{F}_2$.

In the ANF of any $f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$ there are at most:

- one monomial of degree $3 \to xyz$;
- three monomials of degree $2 \to xy, \quad xz, \quad zy$;
- three monomials of degree $1 \to x, \quad y, \quad z$;
- one monomial of degree $0 \to 1$;

where a monomial appears if and only if its corresponding coefficient is nonzero.

We show now that every function from $(\mathbb{F}_2)^3$ to $\mathbb{F}_2$ can be written in ANF. Consider the set $B_3 = \{f : (\mathbb{F}_2)^3 \to \mathbb{F}_2\}$ of **all** functions from $(\mathbb{F}_2)^3$ to $(\mathbb{F}_2)$.

Since the functions from $(\mathbb{F}_2)^3$ to $(\mathbb{F}_2)$ with ANF are a **special kind** of functions from $(\mathbb{F}_2)^3$ to $(\mathbb{F}_2)$, then we have the following set inclusion

$$\{f : (\mathbb{F}_2)^3 \to \mathbb{F}_2 \text{ with ANF}\} \subseteq \{f : (\mathbb{F}_2)^3 \to \mathbb{F}_2\}\,.$$

By Definition 4, in the ANF of a Boolean function $f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$ there are 8 coefficients $a, b, c, d, \alpha, \beta, \gamma, \delta \in \mathbb{F}_2$. Each of them may hold value 0 or 1, depending on the function $f$, so we have **two** choices for each coefficient. Thus, we have at most $2^8$ functions in $\{f : (\mathbb{F}_2)^3 \to \mathbb{F}_2 \text{ with ANF}\}$. The number of functions in $B_3$ with ANF may be less than $2^8$ only in the case when two different choices on the coefficients of the ANF lead to the same function. This case is impossible since

$$\begin{gathered}\text{if } f, g \in \mathbb{F}_2[x, y, z] \text{ are in ANF and } f \neq g \text{ in } \mathbb{F}_2[x, y, z], \text{ then } f \neq g \text{ in } B_3, \text{ i.e.}\\ \text{there is at least a 3-tuple of bits } (u, v, w) \in (\mathbb{F}_2)^3 \text{ such that}\\ f(u, v, w) \neq g(u, v, w)\end{gathered}$$

(we do not prove this last claim, leaving it as a useful exercise to the reader). Now, pointing out that the size of $B_3$ is $|B_3| = |(\mathbb{F}_2)^3| = 2^8$, so there are as many functions in $B_3$ as the number of possible ANF's, we get that

$$\{f : (\mathbb{F}_2)^3 \to \mathbb{F}_2 \text{ with ANF}\} \;=\; B_3\,,$$

hence every function from $(\mathbb{F}_2)^3$ to $\mathbb{F}_2$ can be written in Absolute Normal Form.

**Example 14.** *Consider the function $f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$ defined by*
$f(0,0,0) = 0$, $f(0,0,1) = 0$, $f(0,1,0) = 1$, $f(0,1,1) = 1$,
$f(1,0,0) = 1$, $f(1,0,1) = 1$, $f(1,1,0) = 0$, $f(1,1,1) = 1$.
*We want to derive the ANF of $f$. We know that it is something like*

$$f(x, y, z) = axyz + bxy + cxz + dyz + \alpha x + \beta y + \gamma z + \delta,$$

*for some $a, b, c, d, \alpha, \beta, \gamma, \delta \in \mathbb{F}_2$. By evaluating the ANF at $(0,0,0)$ we see that $f(0,0,0) = \delta$. By definition of $f$, $f(0,0,0) = 0$, so we have identified $\delta$ as $\delta = 0$.*
*By evaluating the ANF at $(1,0,0)$ we see that $f(1,0,0) = \alpha + \delta = \alpha$. By definition of $f$, $f(1,0,0) = 1$, so we have identified $\alpha$ as $\alpha = 1$. Similarly we deduce $\beta = 1$ and $\gamma = 0$. By evaluating the ANF at $(1,1,0)$ we see that $f(1,1,0) = b + \alpha + \beta + \delta = b$. By definition of $f$, $f(1,1,0) = 0$, so we have identified $b$ as $b = 0$. Similarly $c = 0$ and $d = 0$. Finally, by evaluating the ANF at $(1,1,1)$ we see that $a = 1$. The final result is $f = xyz + x + y$.*

**Exercise 33.** *Determine the ANF of the function defined by*
$f(0,0,0) = 0$, $f(0,0,1) = 0$, $f(0,1,0) = 1$, $f(0,1,1) = 1$,
$f(1,0,0) = 1$, $f(1,0,1) = 1$, $f(1,1,0) = 1$, $f(1,1,1) = 1$.

The following function, often used in cryptography, is called the *majority function*:

$$f : (\mathbb{F}_2)^3 \to \mathbb{F}_2$$

$$(\bar{x}, \bar{y}, \bar{z}) \mapsto \bar{x}\bar{y} + \bar{x}\bar{z} + \bar{y}\bar{z}.$$

This function returns value 1 if and only if at least two among the input bits $\{\bar{x}, \bar{y}, \bar{z}\}$ hold value 1. Obviously, it returns 0 otherwise, i.e. if and only if at least two among the three input bits holds value 0.

**Exercise 34.** *Verify the above statement, by evaluating $f$ at all 3-tuples of bits.*

Until now we have only given results and examples where a function has at most 3 bits in input. We would like to consider the most general case, where the number of bits in input is an arbitrary integer $n \geq 1$.
To do that, we need to consider $\mathbb{F}_2[x_1, ..., x_n]$ and so we can generalize definition 3 to the following

**Definition 5.** *A* squarefree *monomial* $\mathbf{t}$ *in* $\mathbb{F}_2[x_1, ..., x_n]$ *is a monomial where each variable appears with exponent at most one, that is,*

$$0 \leq \deg_{x_1}(\mathbf{t}), \deg_{x_2}(\mathbf{t}), ..., \deg_{x_n}(\mathbf{t}) \leq 1.$$

We can now conclude this section with the most general formulation of the Absolute Normal Form Theorem.

**Theorem 3** (Absolute Normal Form Theorem)**.** *Let $n \in \mathbb{N}$. Any Boolean function*

$$f : (\mathbb{F}_2)^n \to (\mathbb{F}_2)$$

*can be written as a polynomial in $\mathbb{F}_2[x_1, .., x_n]$. More precisely, $f$ can be written as the sum of some* squarefree monomials *of degree from 0 to $n$, i.e.*

$$f = a_0 + a_1x_1 + ... + a_nx_n + a_{1,2}x_1x_2 + ...a_{n-1,n}x_{n-1}x_n + ... + a_{1,2,...,n}x_1x_2 \cdots x_n\,,$$

*where $a_0, ..., a_{1,2,...,n} \in \mathbb{F}_2$.*

**Example 15.** *Consider the function of Example 12, returning 1 if a vector of bits is 1 and 0 otherwise. We can see that if we consider, for example, vectors of 4 bits, it can be represented as $f = x_1x_2x_3x_4 + 1$.*

## 0.6 Bytes

The polynomials in $\mathbb{F}_2[x]$ are infinite. However, if we bound the degree, we find a finite set, as for example

$$S := \{p \in \mathbb{F}_2[x] | \deg(p) < 3\} = \{0, 1, x, x + 1, x^2, x^2 + x, x^2 + 1, x^2 + x + 1\}.$$

This set contains 8 polynomials; indeed, $f \in S$ if and only if $f$ is of the form

$$f = a_0 + a_1x + a_2x^2,\ a_0, a_1, a_2 \in \mathbb{F}_2$$

Since there are two choices for each coefficient $a_0, a_1, a_2$ the polynomials are $2^3 = 8$. If we take two polynomials $f, g$ in $S$, then also $f + g$ belongs to $S$,

since when we sum two polynomials, the degree cannot grow (see Exercise 11), according to the rule in $\mathbb{F}_2[x]$

$$\deg(f+g) \leq \deg(f), \deg(g); \qquad \deg(f+g) < \deg(f), \deg(g) \iff \deg(f) = \deg(g).$$

However, if we multiply $f, g$, their product $fg$ can be outside $S$; for example

$$f = x^2 + 1,\ g = x + 1 \quad \Longrightarrow \quad (x^2 + 1)(x + 1) = x^3 + x^2 + x + 1\,.$$

We describe a way to make $S$ "closed with respect to multiplication",that is, we want to define a special operation on the polynomials in $S$ that allows their product to **remain** in $S$. We do that by what is called a "polynomial relation". For example, we can define the following relation $x^3 = x + 1$, or equivalently $x^3 + x + 1 = 0$. With this relation in mind, any time we find a monomial of degree greater than or equal to 3 we substitute $x^3$ with $x + 1$. We iterate this substitution until we obtain a polynomial of degree strictly less than 3. For example

$$x^5{+}x \;=\; x^2(\underline{x^3}){+}x \;=^{\text{substitution}}\; x^2(\underline{x+1}){+}x \;=\; \underline{x^3}{+}x^2{+}x \;=^{\text{substitution}}\; \underline{x+1}{+}x^2{+}x \;=\; x^2{+}1\,.$$

The crucial observation here is that we can obtain the same result by dividing the polynomial $x^5 + x$ by the polynomial $g = x^3 + x + 1$: the remainder is $x$ (see Exercise 19).
Indeed, we can consider a set $T$, which collects the remainders of the divisions of all polynomials in $F_2[x]$ by $g$. It is easy to see that $T$ contains $S$, because when we divide a polynomial $f$ of degree less than 3 by $g$, the remainder is $f$ itself (see Example 6), so

$$\{0, 1, x, x + 1, x^2, x^2 + x, x^2 + 1, x^2 + x + 1\} \subset T$$

On the other hand, if we consider any polynomial $f \notin S$, this has $\deg(f) \geq 3$ and when we divide it by $g$, we will get a remainder of degree strictly less than $\deg(g) = 3$, and so $T$ can only contain polynomials of degree at most 2, therefore

$$T = S = \{0, 1, x, x + 1, x^2, x^2 + x, x^2 + 1, x^2 + x + 1\}\,.$$

We can perform this construction in general. Let $g$ be in $F_2[x]$, with $\deg(g) = n$. We consider the set $A$ of polynomials that collects all remainders of the division by $g$

$$A \;=\; \{\text{remainders by } g\}\,.$$

This set is finite, because the degree of its polynomials is at most $n - 1$, and again it actually contains all polynomials with degree at most $n - 1$

$$A \;=\; \{0, 1, x, x + 1, \ldots, x^{n-1}, x^{n-1} + 1, \ldots, x^{n-1} + x^{n-2} + \cdots + x + 1\}\,.$$

Also in this general setting, we can sum and multiply. The sum is obvious, the multiplication may require some divisions by $g$ before we can arrive at a small-degree polynomial.

**Example 16.** *Let us consider the polynomial* $g = x^2 + x + 1 \in \mathbb{F}_2[x]$ *and define the set* $A_1 = \{\text{remainders by } g\}$. *For this particular* $g$, *we have* $A_1 = \{0, 1, x, x+1\}$. *We want to show that* $A_1$ *is a field. We recall the definition of field, namely that* $A_1$ *should satisfy the following properties*

*i)* $A_1$ *is an abelian group w.r.t. the sum of polynomials;*

*ii)* $A_1 \setminus \{0\} = \{1, x, x+1\}$ *is an abelian group w.r.t. the product of polynomials;*

*iii)* $(f+g)\cdot h = fh + gh$, *for any* $f, g, h \in A_1$.

*Since* $A_1$ *is formed by polynomials, properties i) and iii) are obvious. As regards ii), we need only to prove that each nonzero element in* $A_1$ *has an inverse:*

- $1 \cdot 1 = 1$ *so* $1$ *is the inverse of itself;*
- $x(x+1) = \underline{x^2} + x =^{\text{substitution}} \underline{(x+1)} + x = x + 1 + x = 1$, *so in* $A_1$,

$$\frac{1}{x+1} = x, \qquad \frac{1}{x} = x + 1.$$

The polynomial $g = x^2 + x + 1$ of the above example is irreducible (prove it using Theorem 1); in the next example, we see what happens if the polynomial we consider is reducible.

**Example 17.** *Consider the polynomial* $h = x^2 + 1$. *We can easily note that it is reducible, since* $h = (x+1)^2$. *Let us call* $A_2$ *the set of remainders modulo* $h$. *We see that* $A_2 = \{0, 1, x, x+1\}$, *which is apparently the same* $A_1$ *of the previous example. However we must distinguish* $A_1$ *from* $A_2$, *because the relation imposed on* $A_2$ *by* $h$ *does* **not** *make it a field. We prove it by showing that* $x+1 \in A_2 \setminus \{0\}$ *has no inverse. Indeed*

- $(x+1)\cdot 1 = x + 1 \neq 1$;
- $(x+1)\cdot x = \underline{x^2} + x =^{\text{substitution}} \underline{1} + x \neq 1$;
- $(x+1)(x+1) = \underline{x^2} + 1 =^{\text{substitution}} \underline{1} + 1 = 0 \neq 1$.

For cryptographic reasons we are especially interested in the case of $g_{256} = x^8 + x^4 + x^3 + x^2 + 1 \in \mathbb{F}_2[x]$. We define $\mathbb{F}_{256} := \{p \in \mathbb{F}_2[x] \mid \deg(p) < 8\}$. As explained above $\mathbb{F}_{256}$ is the set of the remainders of divisions by $g_{256}$, i.e.

$$\mathbb{F}_{256} = \{0, 1, , x, x+1, \ldots, x^8, \ldots, x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1\}.$$

The following fundamental theorem explains the different behaviour of the polynomials sets in the two previous examples:

**Theorem 4.** *Let* $g$ *be in* $\mathbb{F}_2[x]$. *We consider the set* $A$ *of polynomials that collects all remainders of the division by* $g$

$$A = \{\text{remainders by } g\}.$$

*Then* $A$ *is a* <u>field</u> *if and only if* $p$ *is an* <u>irreducible</u> *polynomial.*

**Exercise 35.** *List all irreducible polynomials of degree* 2 *and* 3*, proving irreducibility by showing that the remainders' set is a field.*

Thanks to Theorem 4 and the fact that $g_{256}$ is irreducible, the following corollary holds

**Corollary 1.** $\mathbb{F}_{256}$ *is a field.*

The field $\mathbb{F}_{256}$ is so important that it deserves a name: the byte field. Its elements are called *bytes*. You may have met bytes already in computer science courses and they looked different from polynomials, so in the next subsection we will show you the connection between the representation of bytes in polynomial form (which is that used in cryptography) and more common representations (hex values and bit strings).

### 0.6.1 Notations for bytes

In Section 0.3 we have introduced nibbles and bytes, respectively, as vectors of four bits and of eight bits.
So a nibble could be 0100 and a byte could be 00111010. We first have a special look at nibbles, then we relate them to bytes, and finally we compare this approach with the polynomial notation.

In computer science it is very useful to represent long strings of bits in a compact way. For this aim, we use hexadecimal notation, i.e. the notation for numbers in base 16. This notation system uses sixteen symbols, i.e. the usual symbols 0-9 to represent numbers from 0 to 9 and then the letters A, B, C, D, E, F for the numbers from ten to sixteen.
Each hexadecimal symbol represents a *nibble*, i.e. four bits, as shown in the following table

| Decimal | Bits | Polynomials | Hex |
|---|---|---|---|
| 0 | 0000 | 0 | 0 |
| 1 | 0001 | 1 | 1 |
| 2 | 0010 | $x$ | 2 |
| 3 | 0011 | $x+1$ | 3 |
| 4 | 0100 | $x^2$ | 4 |
| 5 | 0101 | $x^2+1$ | 5 |
| 6 | 0110 | $x^2+x$ | 6 |
| 7 | 0111 | $x^2+x+1$ | 7 |
| 8 | 1000 | $x^3$ | 8 |
| 9 | 1001 | $x^3+1$ | 9 |
| 10 | 1010 | $x^3+x$ | $A$ |
| 11 | 1011 | $x^3+x+1$ | $B$ |
| 12 | 1100 | $x^3+x^2$ | $C$ |
| 13 | 1101 | $x^3+x^2+1$ | $D$ |
| 14 | 1110 | $x^3+x^2+x$ | $E$ |
| 15 | 1111 | $x^3+x^2+x+1$ | $F$ |

In order to convert from binary to hexadecimal notation we perform the following steps:

- group the bits in nibbles from right to left;
- convert each nibble in an hexadecimal digit as in the table above.

On the other hands, to convert from hexadecimal to binary notation

- using the table above, write each hexadecimal digit as a nibble;
- juxtapose the results.

**Example 18.** *The vector* 00010101 *is formed by the two nibbles* 0001 0101 *represented as* $0x15$*, corresponding to the decimal* 21*. The vector* 11100111 *is formed by the nibbles* 1110 0111*, with the notation* $0xE7$*, corresponding to the decimal* 231*.*

**Exercise 36.** *Find the hex representation of the following strings:*

- 11110100*;*
- 01100010*;*
- 11100010*.*

*Find the binary representation of the following hexadecimal numbers:* $AF$*,* $A0$ *and* $B5$*.*

A byte is a vector of 8 bits, so it is formed by two nibbles. Then, we can represent it with two "hex digits", as we have done in Example 18.
We can also represent bytes as polynomials: in particular if $a_7a_6a_5a_4a_3a_2a_1a_0$ is the binary vector representing a byte, we have the following representation

$$a_7a_6a_5a_4a_3a_2a_1a_0 \to a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0.$$

**Example 19.** *The byte* 10101011*, corresponding to the hex notation* $0xAB$*, can also be denoted by the polynomial* $x^7 + x^5 + x^3 + x + 1$*.*

The polynomial notation for bytes is important because we can multiply bytes by means of polynomial multiplication, which turns out to be important for cryptographic applications. Let us see an example.

**Example 20.** *Suppose we want to multiply* $0xAB$ *by* $0x4F$*. Then, we first convert these numbers from the hex notation to the binary notation:*

$$0xAB = 10101011, \qquad 0x4F = 01001111.$$

*From the binary notation, we can get the corresponding polynomials:*

$$0xAB = x^7 + x^5 + x^3 + x + 1 \qquad 0x4F = x^6 + x^3 + x^2 + x + 1.$$

*Multiplying the polynomials is now an easy task; we only have to remember that, since we are dealing with bytes we need to take the reminder by* $g_{256} = x^8 + x^4 + x^3 + x^2 + 1$. *So*

$$0xAB \cdot 0x4F \to (x^7 + x^5 + x^3 + x + 1)(x^6 + x^3 + x^2 + x + 1) =$$

$$x^{13} + x^{11} + x^{10} + x^7 + x^6 + x^3 + 1 = x^7 + x^6 + x^4 + x$$

*The result is* $x^7 + x^6 + x^4 + x = 11010010 = 0xD2$.

**Example 21.** *Suppose we want to perform the bytes multiplication* $88 \cdot 88$, *where each byte is represented in decimal form. In its binary form,* 88 *is* 01011000, *whereas its hex form is* $0x58$ *and it can be represented as the polynomial* $f = x^6 + x^4 + x^3$. *An easy computation shows that*

$$f^2 = (x^6+x^4+x^3)^2 = \underline{x^{12}}+x^8+x^6 \ =^{\text{substitution}} \ \underline{x^8 + x^7 + x^6 + x^4}+x^8+x^6 = x^7+x^4\,,$$

*whose representation in binary form is* 10010000, *in hex notation is* $0x90$ *and in decimal form is* 144.

**Exercise 37.** *Perform the following byte multiplications and write the results in the notations: hexadecimal, binary, polynomials, decimal.*

- $0xCA \cdot 0x1C$;
- $01010000 \cdot 10100100$;
- $0x4A \cdot 0x6D$
- $11 \cdot 49$

## 0.7 Vectorial Boolean functions

**Definition 6.** *A vectorial Boolean function is a function*

$$F : (\mathbb{F}_2)^n \to (\mathbb{F}_2)^m,$$

*with* $m, n \in \mathbb{N}$.

We can represent a vectorial Boolean function as a vector of Boolean functions:

$$F : (\mathbb{F}_2)^n \to (\mathbb{F}_2)^m,$$
$$(x_1, ..., x_n) \mapsto \begin{bmatrix} F_1(x_1, ..., x_n) \\ \vdots \\ F_m(x_1, ..., x_n) \end{bmatrix},$$

**Example 22.** *The function* $F : (\mathbb{F}_2)^3 \to (\mathbb{F}_2)^2$, *given by* $(x, y, z) \mapsto (xy + y, y + z)$ *is a vectorial Boolean function.*

We notice that every component of a vectorial Boolean function is a Boolean function, so each component can be represented in its absolute normal form.

**Theorem 5.** *Let*

$$F : (\mathbb{F}_2)^n \to (\mathbb{F}_2)^m, n, m \in \mathbb{N}$$

*be a vectorial Boolean function. Then* $F = \begin{bmatrix} F_1(x_1, ..., x_n) \\ \vdots \\ F_m(x_1, ..., x_n) \end{bmatrix}$, *with*

$$F_i = \sum_{S \subset \{1,...,n\}} a_S^{(i)} x_S^{(i)},$$

*where* $a_S^{(i)} \in \mathbb{F}_2$ *and* $x_S^{(i)} = \prod_{j \in S} x_j$ *for every* $j \in \{1, ..., m\}$.

Since the size of the set of all functions from a set $A$ to a set $B$ i.e. $\mathfrak{F} = \{f : B \to B\}$ is $|\mathfrak{F}| = |B|^{|A|}$, then the number of all vectorial Boolean functions from $(\mathbb{F}_2)^n$ to $(\mathbb{F}_2)^m$ is $(2^m)^{2^n} = 2^{m2^n}$. We observe that the same number comes from counting all vectors of length $m$ with components which are Boolean functions in $x_1, ..., x_n$.

A special case of Boolean functions arises when $m = n$, i.e. the domain and the codomain of the function coincide:

$$F : (\mathbb{F}_2)^n \to (\mathbb{F}_2)^n, n \in \mathbb{N}$$

Some of these could be permutations (i.e. bijections). Since a vectorial Boolean function is a function between two finite sets, then it is a invertible if and only if it is injective or surjective.

**Theorem 6.** *Let* $n \in \mathbb{N}$. *If* $F : (\mathbb{F}_2)^n \to (\mathbb{F}_2)^n$ *is a permutation, then*

$$\deg(F_i) \leq n - 1, \forall i \in \{1, ..., n\}.$$

## 0.8 Friends

There exist particular phenomena as those related with time that have a kind of *ciclicity*. We focus on two examples

1. The days of the week, or if you like music, the musical notes, which are 7;
2. The hours of a day that we assume are 12, as in an analog clock.

The two clocks in the picture represent exactly these two cases. These examples are similar in some sense. In fact we have that the first day of a week is Monday, the $8^{th}$ is Monday again. The seventeenth is Wednesday. In this case we have 2 tours around the clock and the remainder is 3. That is

$$17 = 2 \times 7 + 3.$$

The remainder, namely 3, is the number of interest for our purpose. This is nothing but an application of division algorithm. Focus on the second clock, the one with 12 hours, and suppose we want to find what is the $26^{th}$ hour. We have also in this case two tours around the clock and then the remainder is 2, that is

$$26 = 2 \times 12 + 2.$$

Obviously the remainder depends on how many are the hours the clock. Hence every number has a unique representation after fixing the number of hours. Because of this reduction we always assume that the labelling of the hours is exactly

$$\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$$

where $n$ is the number of the hours of the clock, and $\mathbb{Z}_n$ is the set representing the labelling of the hours. Over these clocks we can define a very nice arithmetic that is called the clock arithmetic, or modular arithmetic. That is we can enrich the set of "hours" with addition and multiplication in a natural way. Thus we can add or multiply the numbers in the standard way and then apply the division with respect to the fixed hours and find the remainder observing that the result belongs to $\mathbb{Z}_n$. As for the case $\mathbb{F}_2$ we represent the tables of the two operations on $\mathbb{Z}_7$ and $\mathbb{Z}_{12}$. First of all we consider the sum.

In both clocks the sum is nothing but a rotation of the first row by one element. For example adding 3 to each element of the first row we have a rotation by 3 elements. in particular the 4 in the first row becomes

$$3 + 4 = 7 \mod 7 = 0 \mod 7.$$

Where mod represents the remainder of the division by 7. The other follows according to this rule.

Now focus on the multiplication's tables.

The two clocks behave in different ways. In $\mathbb{Z}_7$ the multiplication produce a much random table, but, in each row we find al the elements of $\mathbb{Z}_7$. If we study the table of $\mathbb{Z}_{12}$ there are some rows that are complete some other not. That is the second clock has a pathological behavior that the first clock does not have. For example suppose we want to solve the following equation

$$9x = 0 \mod 12.$$

There are 3 numbers in the set of hours that give as a solution and these are 0, 4 and 8. That is not so nice. Now suppose we want to solve the following equation

$$9x = 1 \mod 12.$$

If you consider each multiple of 9 modulo 12 is in the set of hours $\{0, 3, 6, 9\}$ hence the equation has no solution. In this case we say that the equation do not satisfy the "cancellation" property. In fact in a "standard" equation similar to the first one we can multiply 9 by its inverse, getting rid of the 9, and obtaining the unique solution 0. In fact by the table we observe that there are not multiple of 9 that gives 1. That is 9 is not invertible and has not a regular behaviour. Moreover if we have the equation

$$7x = 0 \mod 12.$$

there exists the unique solution $x = 0$, and with respect to the equation

$$7x = 9 \mod 12.$$

there is a unique solution, that is $x = 3$. It is not difficult to observe that the elements that are not regular in the set $\mathbb{Z}_{12}$ are the ones that are not coprime with 12, namely the elements in $a \in \mathbb{Z}_{12}$ such the greatest common divisor with 12 is not 1. They are $\{2, 3, 4, 6, 8, 9, 10, 12 = 0\}$. The remaining elements, the ones whose greatest common divisor with 12 is 1, are invertible: $\{1, 5, 7, 11\}$. Moreover in $\mathbb{Z}_7$ they are all but 0. The number of invertible elements in $\mathbb{Z}_m$ is a fundamental information. The function that given as input $m$, the number of elements of $\mathbb{Z}_m$, give as output the number of invertible elements in $\mathbb{Z}_m$ is called Euler function, namely

$$\phi(m) = |\{a \in \mathbb{Z}_m \text{ and a is invertible}\}|.$$

Hence $\phi(7) = 6$ and $\phi(12) = 4$. In general if $m$ is a prime number $\phi(m) = m-1$. But it is hard for a given number $m$ to compute $\phi(m)$.

By observation above, we may deduce that when $m$ is a prime number the set $\mathbb{Z}_m$ is a field. In fact all the non-zero elements are invertible, as in $\mathbb{Z}_7$. Moreover multiplying two non-zero elements we obtain another non-zero element.

Another particular fact about these fields, is that exist elements whose powers cover all the non-zero elements of thield itself. As an example consider the elements 3 and 5 in $\mathbb{Z}_7$. The powers of 3 are

$$\begin{aligned}
&3\\
&3^2 \equiv 9 \equiv 2\\
&3^3 \equiv 3^2 \cdot 3 \equiv 2 \cdot 3 \equiv 6\\
&3^4 \equiv 3^3 \cdot 3 \equiv 6 \cdot 3 \equiv 4\\
&3^5 \equiv 3^4 \cdot 3 \equiv 4 \cdot 3 \equiv 5\\
&3^6 \equiv 3^5 \cdot 3 \equiv 5 \cdot 3 \equiv 1
\end{aligned}$$

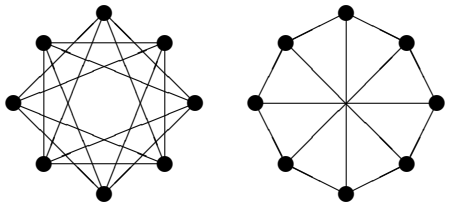


Figure 1: $C_8(\{2,3\})$ e $C_8(\{1,4\})$.

The powers of 5 are

$$\begin{aligned}
&5\\
&5^2 \equiv 25 \equiv 4\\
&5^3 \equiv 5^2 \cdot 5 \equiv 4 \cdot 5 \equiv 6\\
&5^4 \equiv 5^3 \cdot 5 \equiv 6 \cdot 5 \equiv 2\\
&5^5 \equiv 5^4 \cdot 5 \equiv 2 \cdot 5 \equiv 3\\
&5^6 \equiv 5^5 \cdot 5 \equiv 3 \cdot 5 \equiv 1
\end{aligned}$$

**Exercise 38.** *Prove that all the elements of $\mathbb{Z}_7$ different from 3 and 5 are not primitive.*

## 0.9 Some cryptographic applications

This is not a book about cryptography, it is rather a book that helps the reader face a course focused on cryptography, so we will not explain the general theory of cryptography. However, after having seen some mathematical background in the previous sections, it is possible to define rigorously a few cryptographic systems in an easy way. We believe these systems can motivate you to start a deeper subject of cryptography on your own.

In this section we will see:

- the Diffie-Helman key-exchange
- the Rivest-Shamir-Adleman encryption/decryption system
- the classical El Gamal encryption/decryption system
- a classical stream cipher

### 0.9.1 DH

The cryptographic problem solved by the Diffie-Helman key-exchange (DH) has been a long-standing problem in all cryptographic operations since the dawn of time.

Whenever two peers agreed on a system to encrypt their messages, sooner or later they needed to find a way to change it, in order to prevent the enemy from understanding it. The most efficient way to do so is to design a cryptosystem whose exact working depends on a short string, called a **cryptographic key**. Since the key is short (only 16 bytes even in modern times), it can be communicated easily. We will see an example of such a system in Section **??**.
Unfortunately, if there are thousands of users of the systems (as for example the units of an army moving into enemy territories) and the communication channel can be heard by the enemy (radio communication being the main example in history), it becomes very complex to send millions of keys in a secure way.

The DH system allows two peers (Alice and Bob) to agree on a key (or any other short secret), even when communicating over an insecure channel that can be heard by the enemy. Alice and Bob agree on a prime $p$ and a primitive element $g$ of $\mathbb{Z}_p$. This is public information, so for example $p$ and $g$ could be told on the air by the General to all units in his command and it does not matter if the enemy hears these.
The system works as follows:

- Alice chooses secretly a positive integer $a$ less than $p$ and she does not share it with anyone,
- Bob chooses secretly a positive integer $b$ less than $p$ and he does not share it with anyone,
- Alice computes the exponentiation $g^a$ in $\mathbb{Z}_p$ and she sends the result to Bob (the enemy may intercept it)
- Bob computes the exponentiation $g^b$ in $\mathbb{Z}_p$ and he sends the result to Alice (the enemy may intercept it)
- Alice has received $g^b$ and so she can compute another exponentiation $(g^b)^a$ in $\mathbb{Z}_p$, the results being $g^{(ab)}$, that she keeps secret and does not share with anyone,
- Bob has received $g^a$ and so he can compute another exponentiation $(g^a)^b$ in $\mathbb{Z}_p$, the results being $g^{(ab)}$, that he keeps secret and does not share with anyone,
- now Alice and Bob have the same value, $g^{ab}$, and they start using it as a key.

Thanks to the research in finite fields developed in the last 30 years, we observe that, even if the enemies simultaneously collect $p$, $g$, $g^a$ and $g^b$, then they have a negligible probability to reconstruct $g^{ab}$ and so the key remains hidden from them.

The system we have described is still used today a lot, especially on the Internet. We complete this subsection with some observations on the DH protocol as it used nowadays:

- the prime number $p$ is huge, at least $p > 2^{2048}$,
- the prime number $p$ has to be chosen with some specific algebraic properties, for example the factorization of $p-1$ in prime numbers should contain one huge prime,
- there are some primes that have been standardized and are used by many applications, as for instance those proposed by the National Institute of Standards in the U.S.

### 0.9.2 RSA

The key-exchange is only one of the numerous problems that cryptographers face in modern times.
Another interesting problem is the so-called **public-key encryption**, as follows:

- Alice wants to send Bob a secret, but now they are far apart and they can communicate only in an insecure way (e.g., by standard email);
- unfortunately, they did not agree beforehand on a system that uses cryptographic keys and so the DH protocol is useless.

A solution to this problem is the RSA algorithm, summarized below:

- Bob chooses secretly two distinct prime numbers, $p$ and $q$, and perform their multiplication $N = pq$;
- Bob chooses secretly a positive integer $d$ smaller than $N$ and coprime with $\phi(N) = (p-1)(q-1)$;
- Bob computes the inverse $e$ of $d$ modulo $\phi(N)$;
- Bob sends to Alice both $N$ and $e$ (the enemies may intercept them);
- Alice has received $N$ and $e$; she has a message $m$ to send to Bob and so she compute the exponentiation $m^e$ in $\mathbb{Z}_N$, the results being the ciphertext $c$;
- Alice sends $c$ to Bob (the enemies may intercept it);
- Bob has received $c$ and so he can compute another exponentiation $c^d$ in $\mathbb{Z}_p$, the results being $m$ thanks to Theorem ??.

Thanks to the research in number theory developed so far, even if the enemies simultaneously collect $N$, $e$ and $c$, then they have a negligible probability to reconstruct $m$ and so the message remains hidden from them.

The system we have described is still used today, especially for electronic payments. We complete this subsection with some observations on the RSA protocol as it used nowadays:

- the prime numbers $p$ and $q$ are huge, at least $p, q > 2^{1024}$,

- the prime numbers $p, q$ has to be chosen with care, for example their difference $|p - q|$ must exceed both their square roots,
- the private exponent $d$ must be large, at least $\sqrt[4]{N}$,
- the public exponent $e$ should be at least $e \geq 65537$.

### 0.9.3 El Gamal

We now describe another cryptographic algorithm that provides public-key encryption:

- Bob chooses a prime $p$ and a primitive element $g$ of $\mathbb{Z}_p$.
- Bob chooses secretly a positive integer $x$ smaller than $p - 1$.
- Bob computes the exponentiation: $h = g^x$ in $\mathbb{Z}_p$.
- Bob sends $p$, $g$ and $h$ to Alice (the enemies may intercept them);
- Alice has received $p$,$g$ and $h$; she has a message $m$ to send to Bob; she chooses secretely a positive integer $y$ smaller than $p - 1$;
- Alice computes two preliminary exponentiations in $\mathbb{Z}_p$: $c_1 = g^y$ and $s = h^y$ (obviously $h = g^{xy}$).
- Alice encrypts her message $m$ by computing $c_2 = ms$; finally, she sends $c_1$ and $c_2$ to Bob (the enemies may intercept them);
- Bob computes $s$ by an exponentiation in $\mathbb{Z}_p$, since $s = {c_1}^x$;
- Finally, Bob computes the message $m$ by another exponentiation in $\mathbb{Z}_p$, i.e. $m = c_2 \cdot s^{-1}$

Thanks to the research in algebra developed so far, we observe that, even if the enemies simultaneously collect $p$, $g$, $h = g^x$ and $c_1 = g^y$, $s = h^y$ and $c_2 = ms$, then they have a negligible probability to reconstruct $m$ or $x$ and so the message remains hidden from them.

The system we have described is still used. We complete this subsection with some observations on the El-Gamal protocol as it used nowadays:

- the prime number $p$ is huge, at least $p > 2^{2048}$,
- the prime number $p$ has to be chosen with some specific algebraic properties, for example the factorization of $p-1$ in prime numbers should contain one huge prime,
- there are some primes that have been standardized and are used by many applications, as for instance those proposed by the IETF.

### 0.9.4 Stream ciphers

A stream cipher is a cryptographic algorithm that allows two peers to encrypt and decrypt messages, once a secret key has been securely shared (as for example with DH).

In this section we describe the easiest possible stream ciphers, which were used only briefly at about the same time of World War II. We call this design a "classical stream cipher".

To build a classical stream cipher we need some LFSR's (see Section 0.3), say $n$, and a Boolean function with $n$ inputs (and one output). We illustrate the working with an example, which the reader will be able to generalize.
We consider three LFSR's, with polynomials: $x^3 + x + 1$, $x^4 + x^3 + x^2 + x + 1$ and $x^2 + x + 1$.
We consider the Bf $x_1 * x_2 + x_1 + x_2$, and we call it the *combining function*.
The secret key is the initial state (initial vector), which in this case must have $3 + 4 + 2$ bits.

We are ready to describe encryption and decryption:

- Alice wants to encrypt seven bits $m = (0100110)$. Alice generates one bit from each of the three LFSR's and uses the bits as input to the Bf; in other words, if $b_1$,$b_2$ and $b_3$ are the bits generated respectively by the LFSR's, then Alice computes $z_0 = b_1 * b_2 + b_1 + b_2$. The bit $z_1$ is the first bit of the so-called *keystream*;

- Alice adds $z_0$ to the first bit of $m$, which is $m_0 = 0$, and sends $c_0 = z_0 + m_0$; observe the LFSR's have changed their state and so they may now generate another bit;

- similarly, Alice computes $z_1, \ldots, z_6$, and performs the XOR $c_1 = z_1 + m_1, \ldots, c_6 = z_6 + m_6$; she will send $c_1, \ldots, c_6$;

- Bob receives $c_0, \ldots, c_6$;

- his LFSR's are the same of those of Alice and have the same initial state; also the combining function is the same; so, Bob can compute the same bits of the keystream that Alice computed, that is, $z_0, \ldots, z_6$;

- Bob computes the XOR's $c_0 + z_0, \ldots, c_6 + z_6$, obtaining $m_0, \ldots, m_6$, since $c_i + z_i = (z_i + m_i) + z_i = (z_i + z_i) + m_i = 0 + m_i = m_i$.

Stream ciphers used nowadays have a structure which is much more complex, however they still use normally LFSR's. The initial state goes from 64 bits to 256 bits being 128 the minimum length giving acceptable security.
As examples of strea ciphers in use:

- E0, which is used in Blootooth;

- RC4, which is used on the Internet;

- A5/3, which is used in all mobile communications (GSM).

## 0.10 Solutions and hints for exercises

**Exercise 1.**

- $(1+1)+0=(0)+0=0$;
- $(1+1)+1=(0)+1=1$;
- $(1+1)\cdot 1=(0)\cdot 1=0$;
- $(0+0)+1+(1+0)\cdot 1=(0)+1+[(1)\cdot 1]=0+1+1=0$.

**Exercise 2.**

a. $1+1+1+0+0+0+0+1=(1+1)+(1+0)+(0+0)+(0+1)=0+1+0+1=0$

b. $1+1+1+1+0+0+0+0=1+1+1+0+0+0+0+1=0$; the first equality comes from the commutativity of the sum in $\mathbb{F}_2$. Thanks to this property one sees that the expression in b. is exactly the same as in a., up to the ordering of the summands, so the result is the same.

**Exercise 3.**

- $(1+0)\cdot 1=1$ so the opposite is 1;
- $(0+0)\cdot 0=0$ so the opposite is 0;
- $1+1+1+1+0+1+(11)=0$ so the opposite is 0.

**Exercise 4.**

- $1\cdot(1+0)=(1\cdot 1)+(1\cdot 0)=1+0=1$
- $1\cdot(0+1)+0\cdot(1+1)=(1\cdot 0)+(1\cdot 1)+(0\cdot 1)+(0\cdot 1)=0+1+0+0=1$

**Exercise 5.**
First of all, we notice that XOR behaves as the sum of bits, whereas AND behaves as multiplication.
We cannot say the same for OR. Moreover:

- NOT ((NOT $a$) AND (NOT $b$)) = $a$ OR $b$
- NOT ((NOT $a$) OR (NOT $b$)) = $a$ AND $b$
- (NOT $a$) XOR (NOT $b$) and $a$ XOR $b$ have the same truth table.

**Exercise 6.**
NOT $a=a+1$
**Exercise 7.**
$a$ OR $b=(a+1)(b+1)+1$
**Exercise 8.**
Yes, $\mathbb{Q}$ is an abelian group w.r.t. the sum. Indeed the sum is associative and commutative, $0\in\mathbb{Q}$ is the neutral element and if $a\in\mathbb{Q}$, its opposite is $-a$; for example, for $a=\frac{1}{2}$, its opposite is $-\frac{1}{2}$.
**Exercise 9.**
Well, not exactly; $\mathbb{Q}^*=\mathbb{Q}\setminus\{0\}$ is an abelian group w.r.t. the multiplication. Indeed the multiplication is associative and commutative, $1\in\mathbb{Q}$ is the neutral element and if $a\in\mathbb{Q}^*$, its opposite is $\frac{1}{a}$; for example, for $a=\frac{1}{2}$, its opposite is $\frac{1}{\frac{1}{2}}=2$. The element $0\in\mathbb{Q}$ is not invertible.
Actually, $\mathbb{Q}$ is a *field*.
**Exercise 10.**

- $(x^3+x^2+x+1)+(x^4+x^2+x)=x^4+x^3+1$;

- $(x^{10}+x^7+x+1)+(x^5+x)+(x^7+1)=x^{10}+x^5$;
- $x+(x^8+x^5+1)+(x^6+x^2+x)=(x^8+x^6+x^5+x^2+1)$

**Exercise 11**

- $(x^4+x^3+1)(x^3+x^2+1)=x^7+x^5+x^4+x^2+1$, so its degree is 7;
- $(x^3+x^2+x)+(x^4+x^3)=x^4+x^2+x$ so its degree is 4;
- $(x^2+x+1)+(x^2+x)=1$ so it has degree 0.

We can desume that $\deg(fg)=\deg(f)+\deg(g)$ and $\deg(f+g)\leq\max(\deg(f),\deg(g))$.
**Exercise 12**
Consider for example $f=a_0+a_1x+a_2x^2$. We have $f^2=(a_0+a_1x+a_2x^2)=(a_0+a_1x+a_2x^2)(a_0+a_1x+a_2x^2)=a_0^2+a_0a_1x+a_0a_2x^2+a_0a_1x+a_1^2x^2+a_1a_2x^3+a_0a_2x^2+a_1a_2x^3+a_2^2x^4=a_0^2+a_1^2x^2+a_2^2x^4$
**Exercise 13**

- $(x^2+x)^2=x^4+x^2$
- $(x^3+x+1)^2=x^6+x^2+1$
- $(x+1)^4=x^4+1$
- $(x^2+x+1)^3=(x^2+x+1)^2*(x^2+x+1)=(x^4+x^2+1)(x^2+x+1)=x^6+x^5+x^4+x^4+x^3+x^2+x^2+x+1=x^6+x^5+x^3+x+1$

Observe that $(x+1)^4=x^4+1$, so raising a polynomial to an exponent $b$ which is a power of two means raising its monomials to the power $b$ and sum the results. Notice that this is not true for an even exponent that is not power of 2. For example

$$(x+1)^6=x^6+x^4+x^2+1\neq x^6+1$$

Here there are some more steps to do, indeed, since $6=2\cdot 3$, we can write $(x+1)^6=((x+1)^3)^2$, then we compute $(x+1)^3=x^3+x^2+x+1$ and then we have to raise it to the power two, so here we can raise its monomial to the power two and sum them

$$(x+1)^6=((x+1)^3)^2=(x^3+x^2+x+1)^2=x^6+x^4+x^2+1.$$

**Exercise 14**
Let us recall that in $\mathbb{F}_2[x]$ we have
$(x+1)^0=1\rightarrow\mathbf{1}$
$(x+1)^1=x+1\rightarrow\mathbf{1\ 1}$
$(x+1)^2=x^2+1\rightarrow\mathbf{1\ 0\ 1}$
$(x+1)^3=x^3+x^2+x+1\rightarrow\mathbf{1\ 1\ 1\ 1}$
$(x+1)^4=x^4+1\rightarrow\mathbf{1\ 0\ 0\ 0\ 1}$
and so on.
So, we have the triangle:

$$11\ 11\ 0\ 11\ 1\ 1\ 11\ 0\ 0\ 0\ 1 \tag{6}$$

Now, in $\mathbb{Z}[x]$
$(x+1)^0=1\rightarrow\mathbf{1}$
$(x+1)^1=x+1\rightarrow\mathbf{1\ 1}$
$(x+1)^2=x^2+2x+1\rightarrow\mathbf{1\ 0\ 1}$
$(x+1)^3=x^3+3x^2+3x+1\rightarrow\mathbf{1\ 3\ 3\ 1}$
$(x+1)^4=x^4+4x^3+6x^2+4x+1\rightarrow\mathbf{1\ 4\ 6\ 4\ 1}$

and so on.
We have then the Pascal's (Tartaglia's!) Triangle:

$$11\ 11\ 2\ 11\ 3\ 3\ 11\ 4\ 6\ 4\ 1 \tag{7}$$

We can notice that if in the Pascal's Triangle we have an even number, the corresponding position in the Triangle for $\mathbb{F}_2$ we have 0, whereas if in the Pascal's Triangle we have an odd number, the corresponding position in the Triangle for $\mathbb{F}_2$ we have 1. See the chapter **??** for further explanations on this behaviour.

**Exercise 15**

- $x^4 + x^3 + x \to 0$
- $x^3 + x + 1 \to \emptyset$
- $x^6 + x^5 + 1 \to \emptyset$
- $x^4 + x^3 + x^2 + x \to 0, 1$

**Exercise 16**

1. Take the polynomials $f = x + 1$ and $g = x^2 + 1$; it holds $f(0) = g(0) = 1$, $f(1) = g(1) = 0$
2. $p(x) = 0$, $q(x) = 1$, $r(x) = x$, $s(x) = x + 1$
3. there are 4 functions from $\mathbb{F}_2$ to $\mathbb{F}_2$.

**Exercise 17**

- $p_1 = x^5 + x^4 + x + 1 = (x + 1)^5$;
- $p_2 = x^4 + x^2 + 1 = (x^2 + x + 1)^2$.

**Exercise 18** The only irreducible polynomial of $\mathbb{F}_2[x]$ with degree 2 is $x^2 + x + 1$; now since it does not divide $f = x^4 + x + 1$ and $f$ has no roots in $\mathbb{F}_2$, we can desume that it is irreducible.

**Exercise 19**

**Divisioni: solo il risultato o mettiamo proprio tutto il disegnino??**

**Exercise 20**

$(0, 0, 0, 0) + (0, 0, 0, 0) = (0, 0, 0, 0)$
$(0, 1, 0, 1) + (0, 1, 0, 1) = (0, 0, 0, 0)$
$(1, 1, 1, 1) + (1, 1, 1, 1) = (0, 0, 0, 0)$
$(1, 1, 0, 0) + (1, 1, 0, 0) = (0, 0, 0, 0)$
The opposite of a vector of bits is itself.

**Exercise 21**

$5 \cdot (1, 1, 0) = (5 \cdot 1, 5 \cdot 1, 5 \cdot 0) = (1, 1, 0)$
$6 \cdot (0, 1, 0) = (6 \cdot 0, 6 \cdot 1, 6 \cdot 0) = (0, 0, 0)$
Multiplication for an odd number is the same as multiplication by 1, so it keeps the given vector unchanged; multiplication for an even number is the same as multiplication by 0, so it annihilates the given vector. **Exercise 22**
The two equations give two representations of the same operation, in the sense that we can identify a polynomial $f \in \mathbb{F}_2[x]$ with the list of its coefficients, decreasingly ordered by degree, for example

$$(0, 1, 1) \to 0 \cdot x^2 + 1 \cdot x + 1 \cdot 1.$$

Then the sum of polynomials can be performed as the sum of vectors.

**Exercise 23**

**Exercise**

$1^2 \equiv 1$
$2^2 \equiv 4$, $2^3 \equiv 1$
$4^2 \equiv 2$, $4^3 \equiv 1$
$6^2 \equiv 1$.