Master in High Performance Computing

Number Representation

On computer

Nicolas Salles, 13/09/2023

ICTP

Where come from the various number representations

In the position number representation, the base is implicitly contained in the position of number …Thanks to Zero

1932

1000 + (1000-100) + 10 + 10 + 10 + 2

$1*10^3 + 9*10^2 + 3*10^1 + 2*10^0$

The modern number representation can be define as

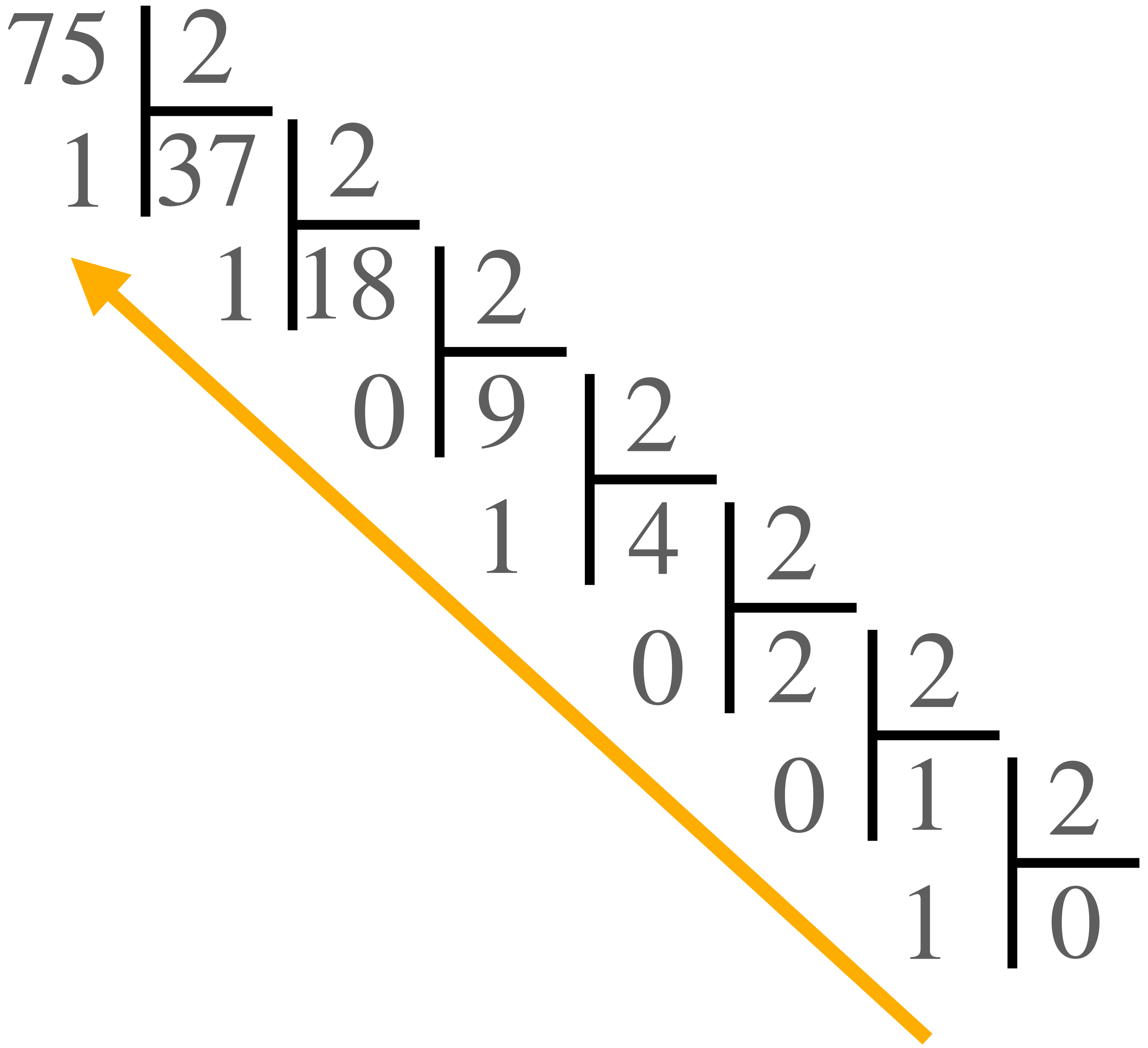
$$d = \sum_{i=-n}^{m} 10^i \text{x} d_i = \sum_{i=-n}^{m} b^i \text{x} d_i$$

The base of the number representation is define by $\boldsymbol{b}$

With **m** integer and **n** decimal

Integer - Conversion b=10 to b=2

To convert from base 10 to base 2 an integer we pass by the euclidian division



$$75|_{10} = 1001011|_{2}$$

Representation on the machine

The numbers are physically represented in base 2 on the machine - a **bit**

A number is represented by a finit group of 8 bits, a **bytes**

A bit $\longrightarrow$ 0 or 1

A bytes $\longrightarrow$ $255 = 2^8-1$

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

By convention integer are represented on 4 bytes = 32 bits $\longrightarrow$ Max value = $2^{32}-1 \sim 4\text{x}10^{10}$

The negative integer

To represent the negative integer we use the first bit to sign the number

The range of the representation is reduce: $[-2^{31}:2^{31}-1]$

We lose one digit: max val = 2^7-1

0/1	2^6	2^5	2^4	2^3	2^2	2^1	2^0

0: positif
1: negative

The negative integer

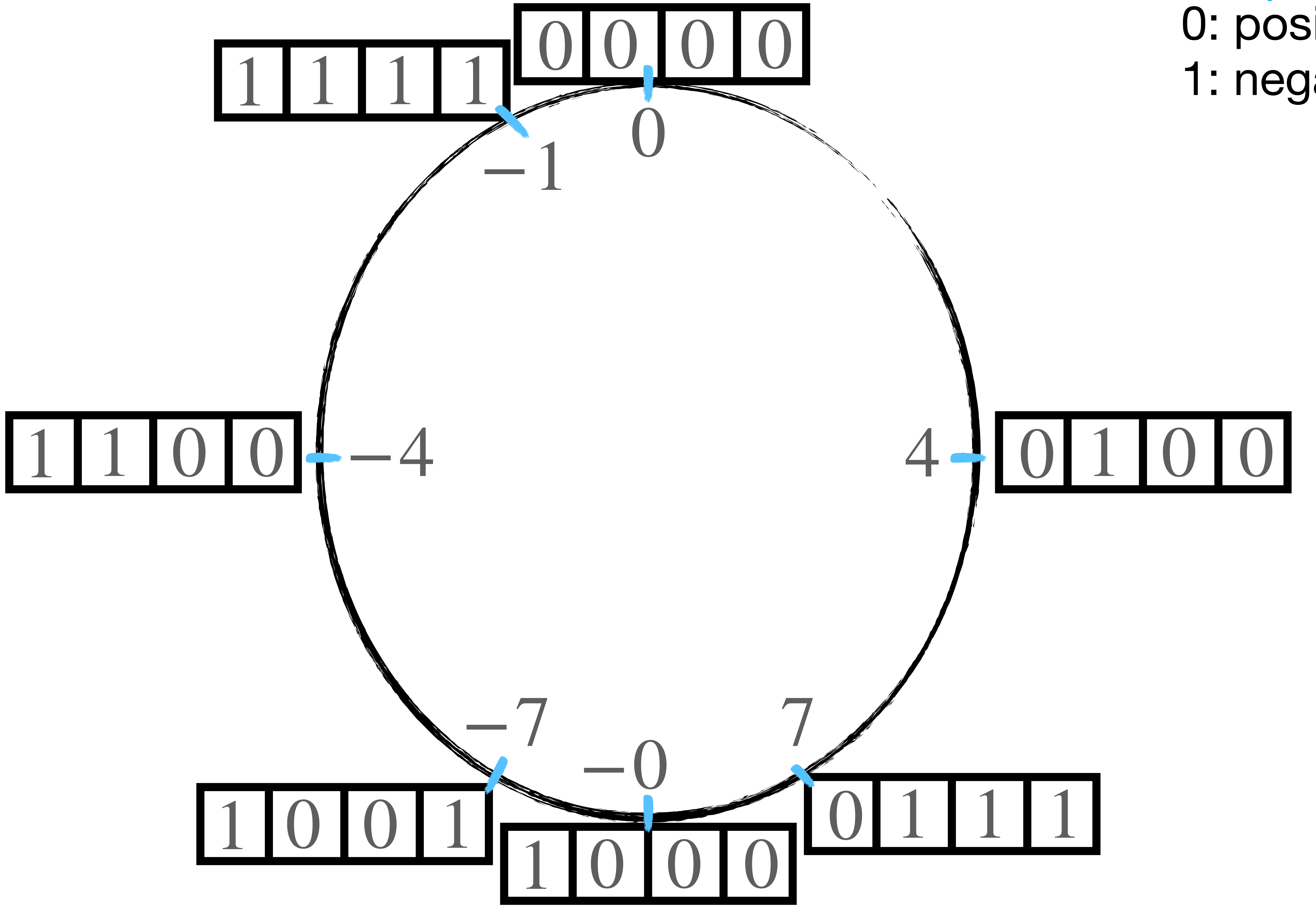
We lose one digit: max val = 2^7-1

0/1	2^6	2^5	2^4	2^3	2^2	2^1	2^0

0: positif
1: negative

To represent the negative integer we use the first bit to sign the number

The range of the representation is reduce: [-2^{31}:$2^{31}-1$]



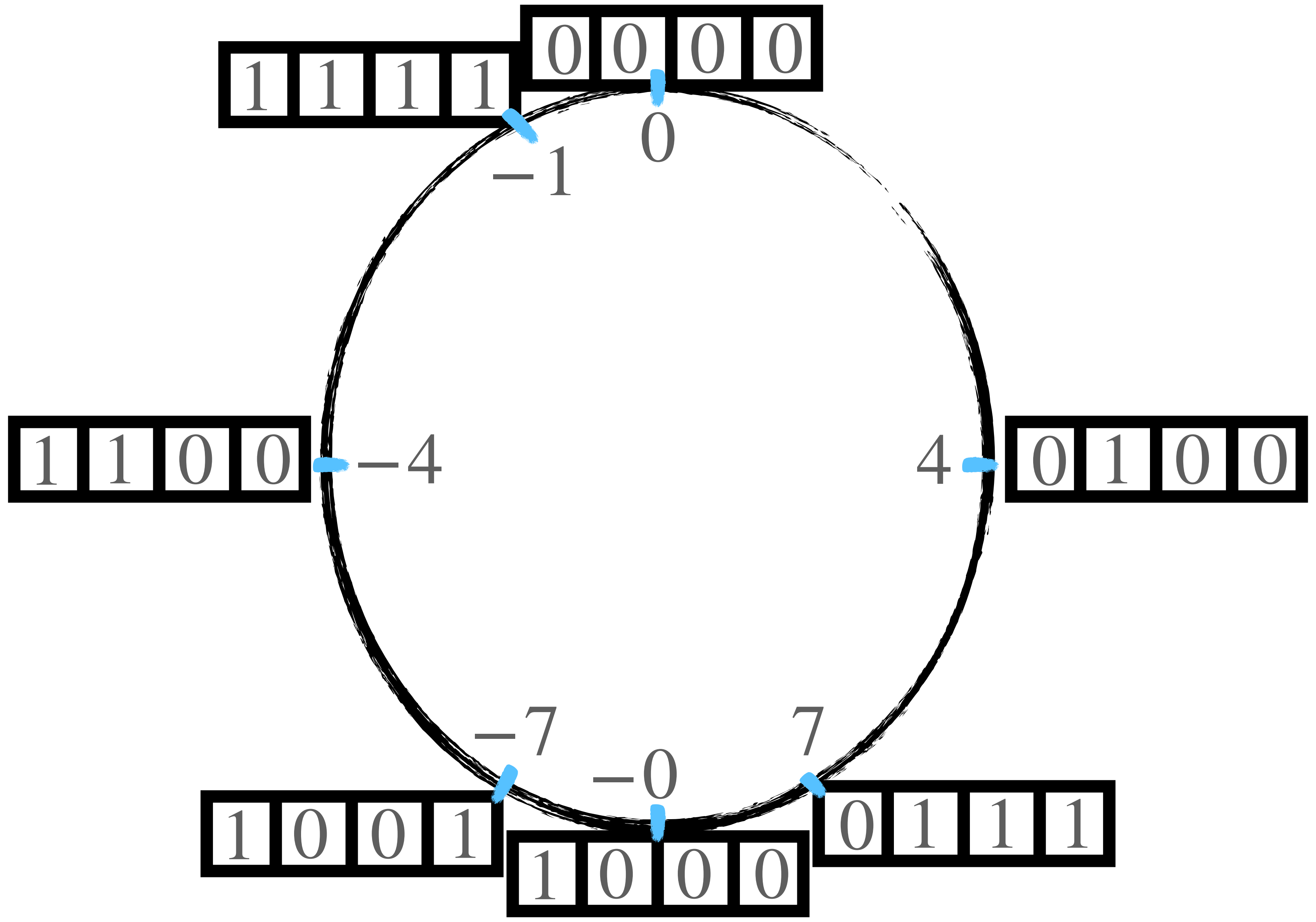
Method of complement at 2

To obtain the opposite value of the a positif integer we use the method of complement at 2

1. Inverse the value of all bit
2. Add 1

$$
\begin{array}{r}
2 = 0010 \\
1101 \\
+0001 \\
\hline
1110 = -2
\end{array}
$$

$$
\begin{array}{r}
0010 \\
+1110 \\
\hline
0000
\end{array}
$$



Floating point - Conversion b=10 to b=2

To convert from base 10 to base 2 the decimal part of real number we pass by a multiplication

$$
\begin{aligned}
0.375 \times 2 &= 0.75 \\
0.75 \times 2 &= 1.5 \\
0.5 \times 2 &= 1.0
\end{aligned}
$$

$$0.375\,|_{10} = 0.011\,|_{2}$$

Floating point - Conversion b=10 to b=2

To convert from base 10 to base 2 the decimal part of real number we pass by a multiplication

$$
\begin{aligned}
0.375 \times 2 &= 0.75 \\
0.75 \times 2 &= 1.5 \\
0.5 \times 2 &= 1.0
\end{aligned}
$$

$$0.375\,|_{10} = 0.011\,|_{2}$$

$$
\begin{aligned}
0.3 \times 2 &= 0.6 \\
0.6 \times 2 &= 1.2 \\
0.2 \times 2 &= 0.4 \\
0.4 \times 2 &= 0.8 \\
0.8 \times 2 &= 1.6 \\
0.6 \times 2 &= 1.2 \\
&\ldots
\end{aligned}
$$

$$0.3\,|_{10} = 0.01001[1001]\,|_{2}$$

Floating point representation on machine

$$-254.603 = -\,2.54603 \text{x} 10^{2}$$

Sign

Mantisse

Exposant

Float is coded on 4 bytes

Double is coded on 8 bytes

1 bit	8 bits	23 bits	In memory
	11 bits	52 bits	

$$r = (-1)^{s} * \left(1 + \sum_{i=1}^{N_m} m_i 2^{-i}\right) * 2^{e_b - e_0}$$

The bit of sign is associate to the mantisse but the sign of the exponent follow another rule

For 8 bits associate to the exponent we define $e_0 = 2^{8-1} - 1 = 127$

And we store in memory an bias exponent: $e_b = e + e_0 \in \{-2^{7-1} + 1 : 2^{7-1}\}$

Floating point representation on machine

$$r = (-1)^s * (1 + \sum_{i=1}^{N_m} m_i 2^{-i}) * 2^{e_b - e_0}$$

$$A = \boxed{1}\ \boxed{10000010}\ \boxed{00110000000000000000000}$$

Sign

Exposant

Mantisse

$S = 1$ (Negatif)

$$e = (2^7 + 2^1) - 127 = 130 - 127 = 3$$

$$m = 2^{-3} + 2^{-4} = 0.125 + 0.0625 = 0.1875$$

$$A = -1.1875 \times 2^3 = -9.5$$

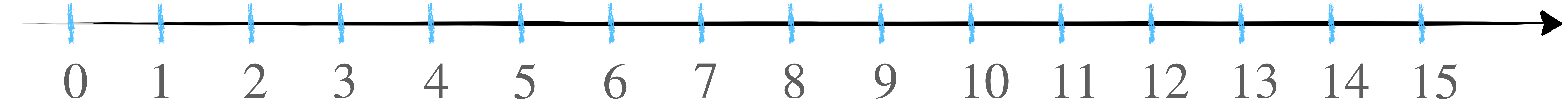
Range of number representation

Example of **positive** real number stored in 4 bits

$$1.\underbrace{\boxed{}}_{2^{-1}}\underbrace{\boxed{}}_{2^{-2}} \times 2^{\underbrace{\boxed{}}_{2^{1}}\underbrace{\boxed{}}_{2^{0}}}$$

$$\varepsilon = 2^{-2} = 25\,\%$$

	exp	0 0 $2^0 = 1$	0 1 $2^1 = 2$	1 0 $2^2 = 4$	1 1 $2^3 = 8$
1.	0 0 0 1 1 0 1 1				
	Δ ε				



Range of number representation

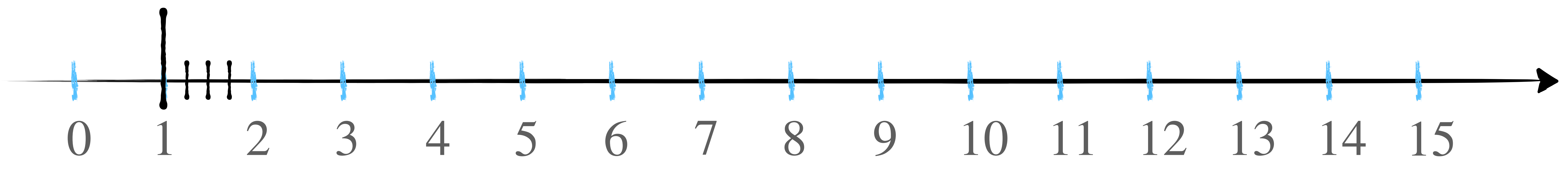
Example of **positive** real number stored in 4 bits

$$1.\underset{2^{-1}\,2^{-2}}{\boxed{}\boxed{}} \times 2^{\underset{2^1\,2^0}{\boxed{}\boxed{}}}$$

$$\varepsilon = 2^{-2} = 25\,\%$$

exp	0 0 $2^0 = 1$	0 1 $2^1 = 2$	1 0 $2^2 = 4$	1 1 $2^3 = 8$
1. 0 0	1			
1. 0 1	1.25			
1. 1 0	1.5			
1. 1 1	1.75			
Δ	0.25			
ε	0.25/1			

Min = 1



Range of number representation

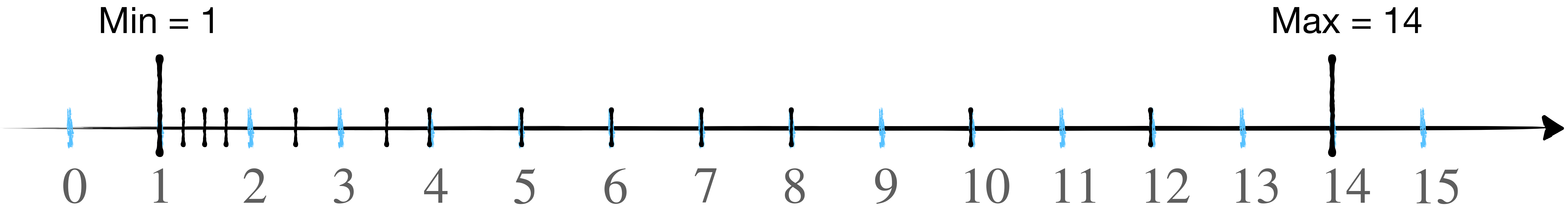
Example of **positive** real number stored in 4 bits

$$1.\underset{2^{-1}\,2^{-2}}{\boxed{}\boxed{}} \times 2^{\underset{2^1\;2^0}{\boxed{}\boxed{}}}$$

$$\varepsilon = 2^{-2} = 25\,\%$$

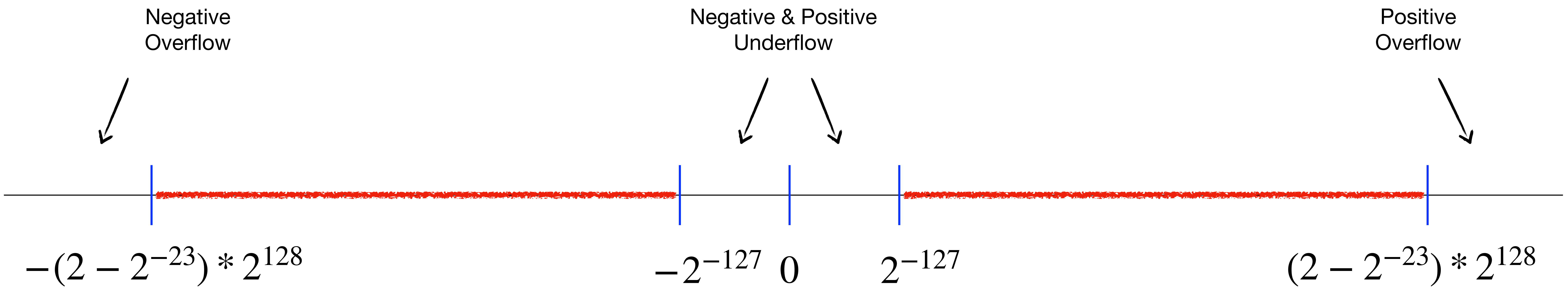
exp	0 0 ($2^0 = 1$)	0 1 ($2^1 = 2$)	1 0 ($2^2 = 4$)	1 1 ($2^3 = 8$)
1. 0 0	1	2	4	8
1. 0 1	1.25	2.5	5	10
1. 1 0	1.5	3	6	12
1. 1 1	1.75	3.5	7	14
Δ	0.25	0.5	1	2
ε	0.25/1	0.5/2	1/4	2/8

$= 25\,\%$ max



Range of number representation

Simple precision (4 bytes)



Character

One character is stored in 1 bytes (8 bits). The character representation follows the ASCII norme.

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]