Converting To/From Binary

Recap: Binary to Decimal

If I want to convert a number from Binary into Decimal, I need to add the powers of 2 at each position the Binary value has a 1 together!

```
1   1   1   0   0   0   0   1
```

Recap: Binary to Decimal

If I want to convert a number from Binary into Decimal, I need to add the powers of 2 at each position the Binary value has a 1 together!

1	1	1	0	0	0	0	1
128	64	32	16	8	4	2	1

Recap: Binary to Decimal

If I want to convert a number from Binary into Decimal, I need to add the powers of 2 at each position the Binary value has a 1 together!

1	1	1	0	0	0	0	1
128	64	32	16	8	4	2	1

Recap: Binary to Decimal

If I want to convert a number from Binary into Decimal, I need to add the powers of 2 at each position the Binary value has a 1 together!

1	1	1	0	0	0	0	1
128	64	32	16	8	4	2	1

128 + 64 + 32 + 1 =

Recap: Binary to Decimal

If I want to convert a number from Binary into Decimal, I need to add the powers of 2 at each position the Binary value has a 1 together!

1	1	1	0	0	0	0	1
128	64	32	16	8	4	2	1

$$128 + 64 + 32 + 1 = 225$$

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 100 from Decimal into Binary.

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 100 from Decimal into Binary.

$$\underset{2^7}{128} > 100 > \underset{2^6}{64}$$

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 100 from Decimal into Binary.

The largest power of 2 that fits into 100 is 64, so we'll mark a 1 as the leftmost value in our binary number.

1						
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 100 from Decimal into Binary.

The largest power of 2 that fits into 100 is 64, so we'll mark a 1 as the leftmost value in our binary number.
Then we want to subtract that power of 2 from our number and repeat the process!

1						
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 100 from Decimal into Binary.

100 - 64 = 36

1						
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 36 from Decimal into Binary.

The largest power of 2 that fits into 36 is 32, so we'll mark a 1 in that position.

1	1					
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 36 from Decimal into Binary.

36 - 32 = 4

1	1					
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 4 from Decimal into Binary.

The largest power of 2 that fits into 4 is 4, so we'll mark a 1 in that position.

1	1			1		
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 4 from Decimal into Binary.

$$4 - 4 = 0$$

1	1			1		
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 4 from Decimal into Binary.

$$4 - 4 = 0$$

Now that we've hit 0, we know our conversion is complete! We can fill in all of the empty slots with 0's!

1	1			1		
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

If we want to convert from Decimal into Binary, we first need to find the largest power of 2 that will fit into the number we have.

Let's say I want to convert 4 from Decimal into Binary.

$$4 - 4 = 0$$

Now that we've hit 0, we know our conversion is complete! We can fill in all of the empty slots with 0's!

1	1	0	0	1	0	0
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

Let's convert the other direction!

Our final result from this process is the binary number `1100100`!

1	1	0	0	1	0	0
2^6 (64)	2^5 (32)	2^4 (16)	2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)

How Computers Do Storage

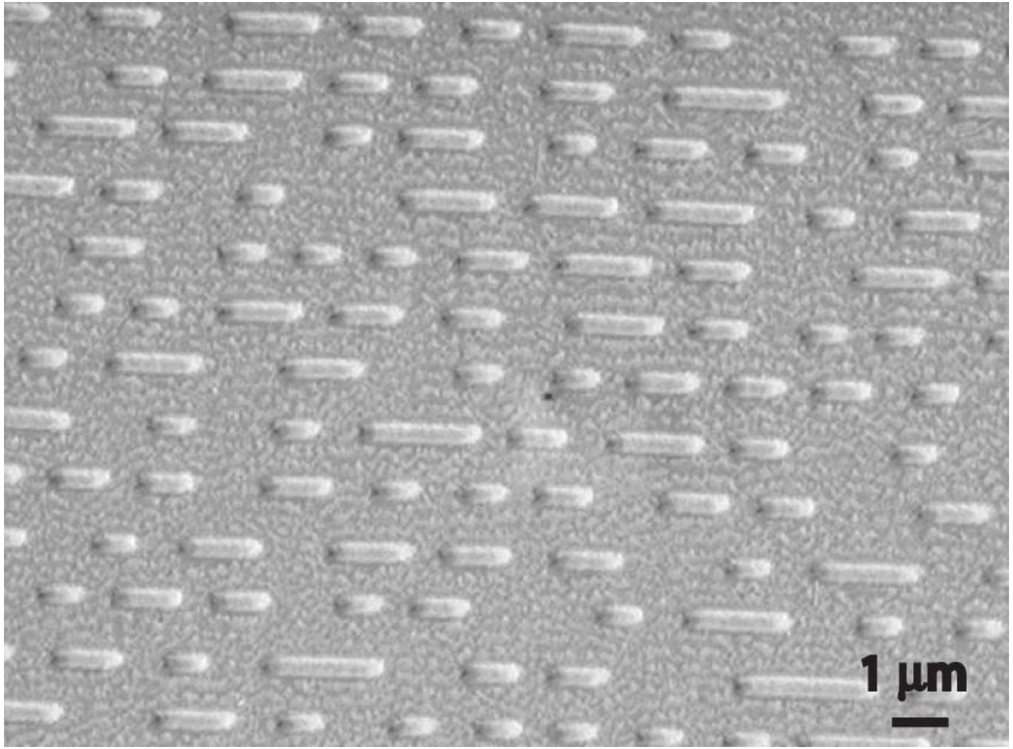
It's All Binary

Like I discussed last class, all digital information is stored in **Binary**.

A single digit in a Binary number (either 0 or 1) is referred to as a "bit".

Computers only use 1's and 0's because that's what is easiest to both read and store! It's significantly easier to determine whether or not you're getting an electrical signal than to read *exactly* how much voltage that signal is carrying.

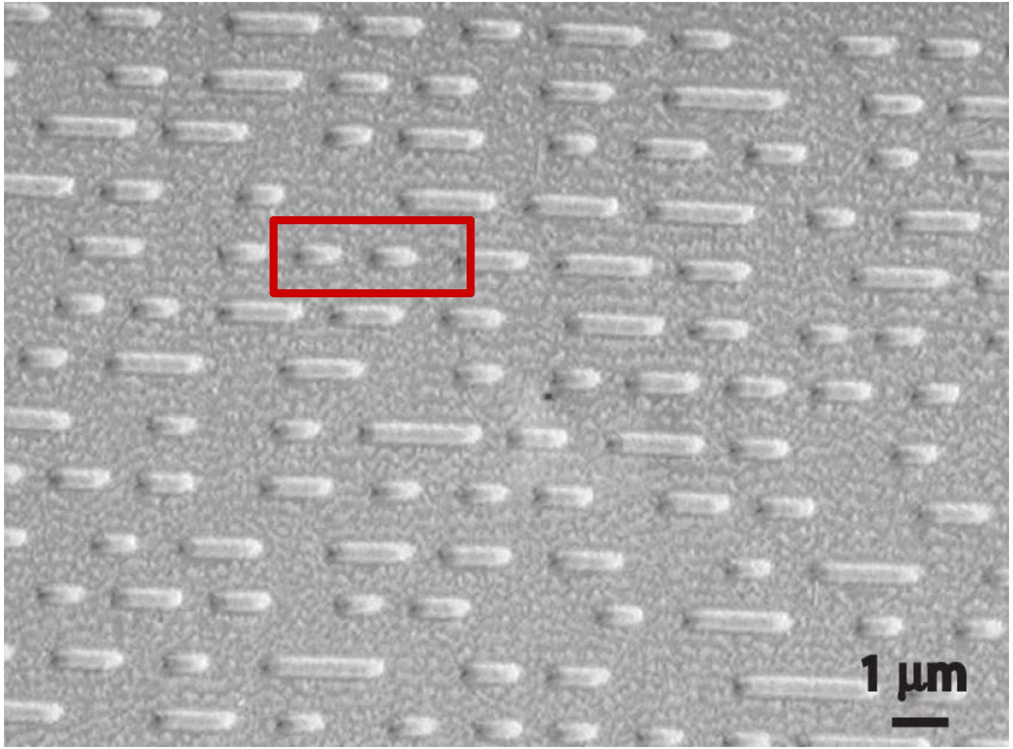
Representing Data



This is the bottom of a CD!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Representing Data

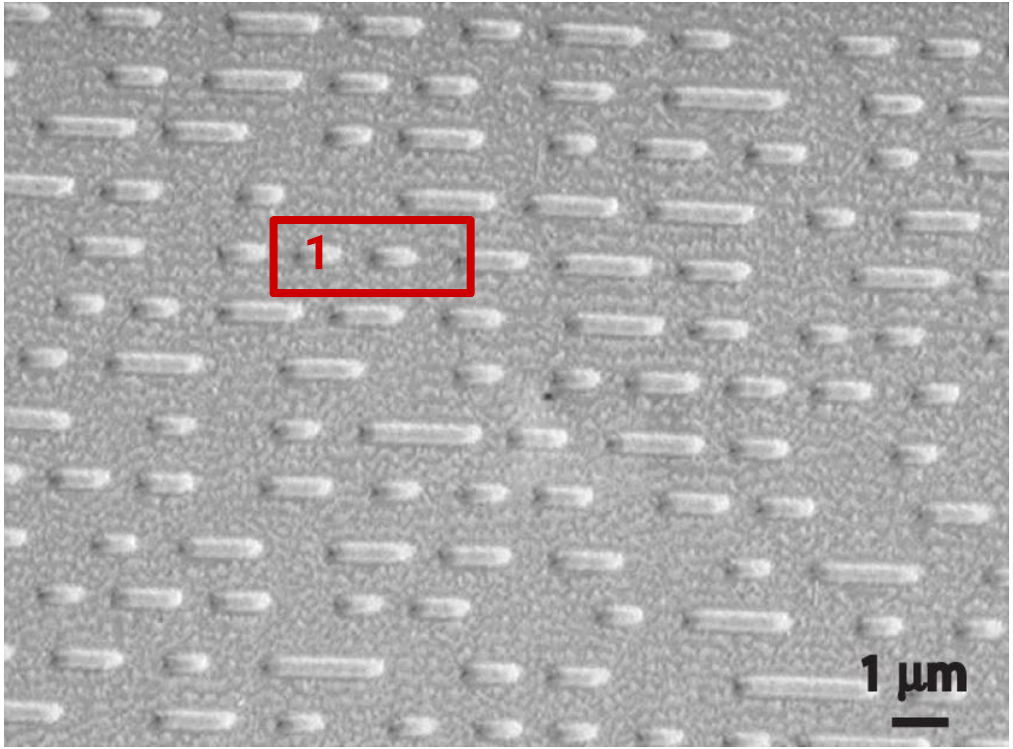


This is the bottom of a CD!

It's covered in little bumps, which the computer can read as either a 1 or a 0!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Representing Data

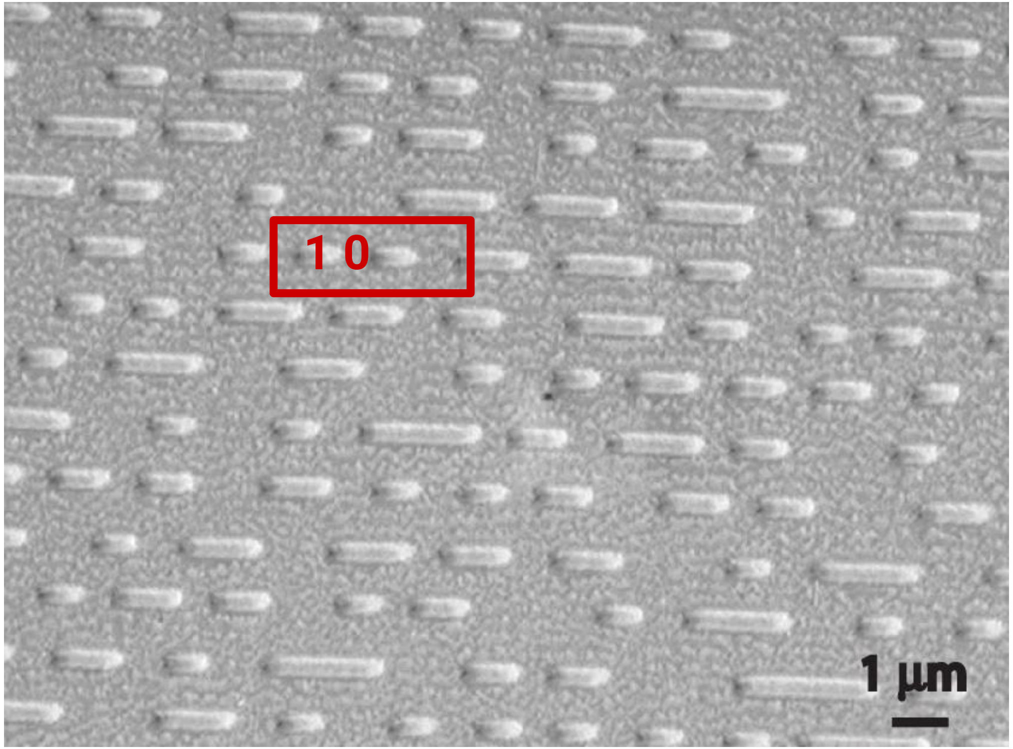


This is the bottom of a CD!

It's covered in little bumps, which the computer can read as either a 1 or a 0!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Representing Data

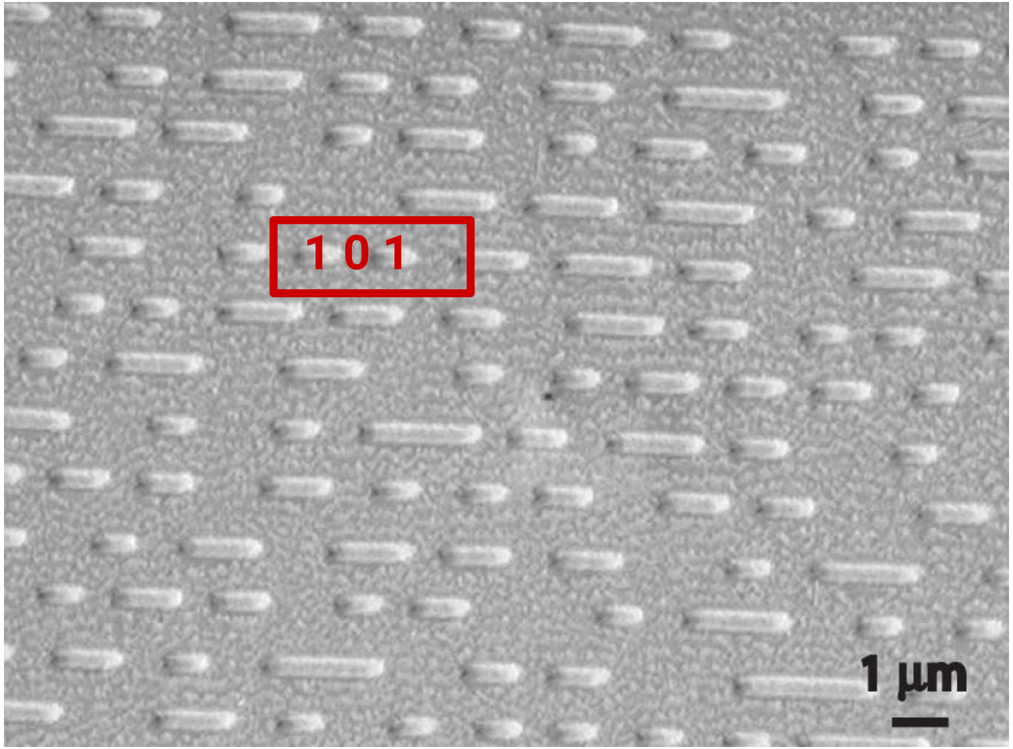


This is the bottom of a CD!

It's covered in little bumps, which the computer can read as either a 1 or a 0!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Representing Data

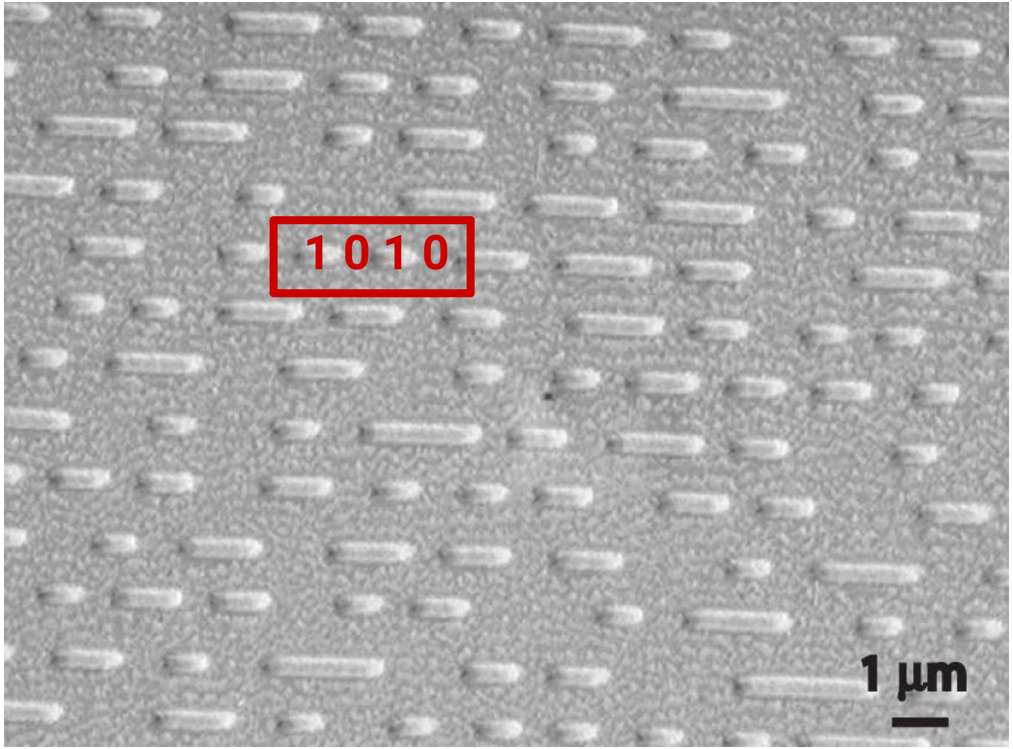


This is the bottom of a CD!

It's covered in little bumps, which the computer can read as either a 1 or a 0!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Representing Data



This is the bottom of a CD!

It's covered in little bumps, which the computer can read as either a 1 or a 0!

Source: https://courses.candelalearning.com/princgenchem/chapter/6-3-atomic-spectra-and-models-of-the-atom/

Groups of Bits

Name	Size	Example
bit	single 0 or 1	`True` / `False`
byte	8 bits	1 character of text
kilobyte (kB)	2^{10} bytes = 1024 bytes	1 paragraph of text
megabyte (MB)	2^{20} bytes = 1024 kB	3MB = 3 minute song
gigabyte (GB)	2^{30} bytes = 1024 MB	16GB = iPad storage
terabyte (TB)	2^{40} bytes = 1024 GB	10TB = Entire Library of Congress
petabyte (PB)	2^{50} bytes = 1024 TB	10 billion facebook photos

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 bit: 2 possible values - 0 and 1.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 bit: 2 possible values - 0 and 1.

2 bits: 4 possible values - 00, 01, 10, and 11.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 bit: 2 possible values - 0 and 1.

2 bits: 4 possible values - 00, 01, 10, and 11.

3 bits: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 **bit**: 2 possible values - 0 and 1.

2 **bits**: 4 possible values - 00, 01, 10, and 11.

3 **bits**: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 **bit**: 2 possible values - 0 and 1.

2 **bits**: 4 possible values - 00, 01, 10, and 11.

3 **bits**: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 **bit**: 2 possible values - 0 and 1.

2 **bits**: 4 possible values - 00, 01, 10, and 11.

3 **bits**: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 **bit**: 2 possible values - 0 and 1.

2 **bits**: 4 possible values - 00, 01, 10, and 11.

3 **bits**: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

Storage Capacity

Let's see how many total values can be stored in any given number of bits:

1 bit: 2 possible values - 0 and 1.

2 bits: 4 possible values - 00, 01, 10, and 11.

3 bits: 8 possible values - 000, 001, 010, 011, 100, 101, 110, and 111.

n bits: 2^n possible values

Mapping Binary Values to Data

In order to store information like text in a computer, I need to map all of the values I want to store to a binary value!

I could, for example, use the following encoding:

`00`	A
`01`	B
`10`	C
`11`	D

Mapping Binary Values to Data

In order to store information like text in a computer, I need to map all of the values I want to store to a binary value!

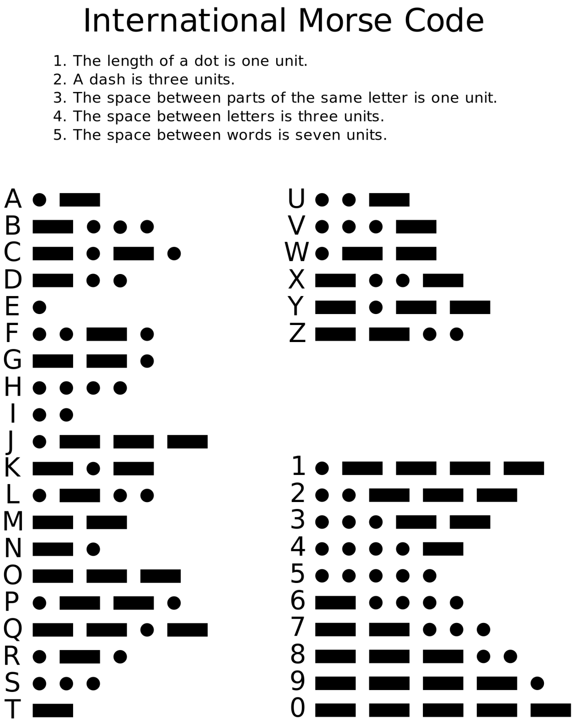
I could, for example, use the following encoding:

`00` - A
`01` - B
`10` - C
`11` - D

Some issues with this approach:

- I can represent, at most, 4 values
- Only I know this encoding!
    - If other people don't know this encoding, they won't be able to decipher a message that uses it!

Morse Code



International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

Back when information was sent digitally via telegraph, Morse Code was invented as a way to encode all of the characters in the alphabet as a sequence of dots and dashes.

Since there's 2 symbols being used, we can replace them with 0's and 1's, and have a passable Binary encoding for every (capital) letter!

A = 0 1

B = 1 0 0 0

C = 1 0 1 0

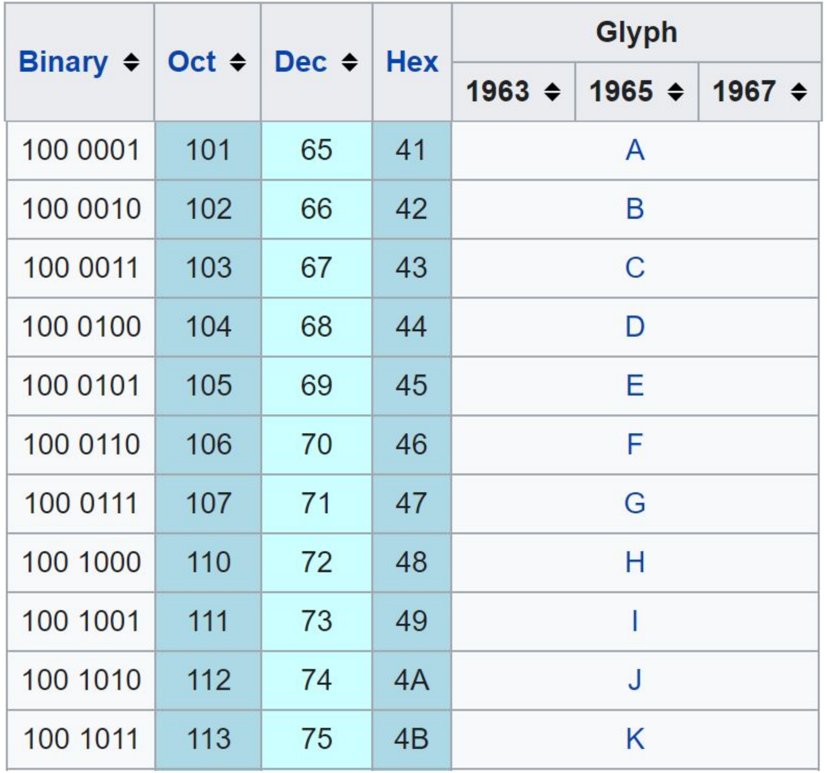
Image Source: https://en.wikipedia.org/wiki/Morse_code#/media/File:International_Morse_Code.svg

Moving On From Morse

As digital data becomes increasingly complex, we need to be able to encode more than just capital letters and digits.

It turns out that we can encode all of the characters in the English language, both upper and lowercase, as well as a large number of symbols, using only 1 **byte** for each!

The ASCII Table

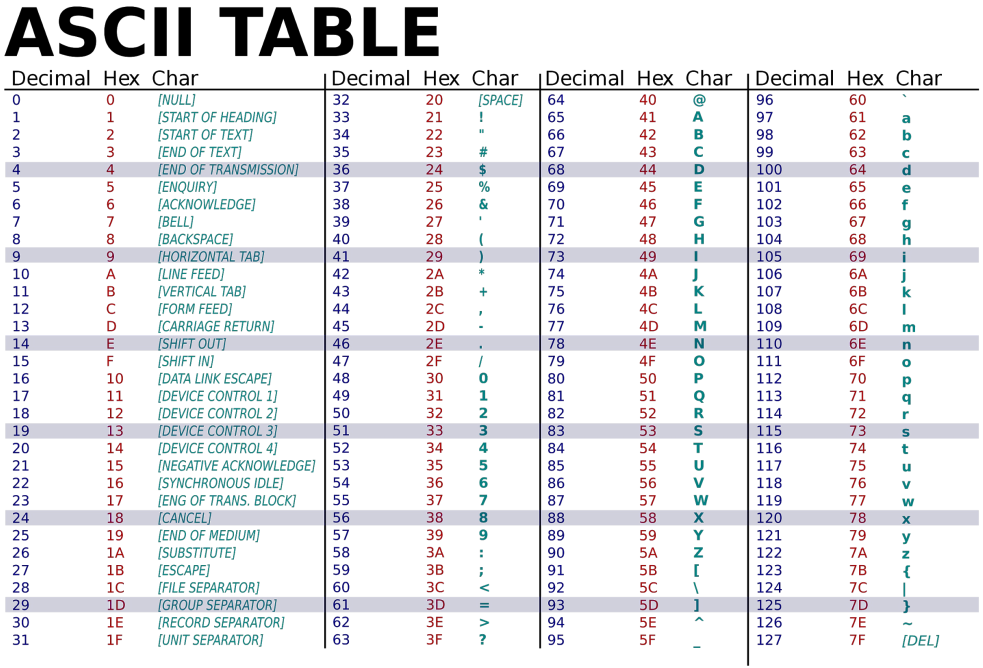


Binary	Oct	Dec	Hex	Glyph		
				1963	1965	1967
100 0001	101	65	41	A		
100 0010	102	66	42	B		
100 0011	103	67	43	C		
100 0100	104	68	44	D		
100 0101	105	69	45	E		
100 0110	106	70	46	F		
100 0111	107	71	47	G		
100 1000	110	72	48	H		
100 1001	111	73	49	I		
100 1010	112	74	4A	J		
100 1011	113	75	4B	K		

ASCII is the standard encoding for text used by computers all around the world.

ASCII Table Image Source: https://en.wikipedia.org/wiki/ASCII

Let's Decode!



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]

If I have the binary sequence

010010000110010101111001

What does it mean?

Let's decode it!

ASCII Table Image Source: https://simple.wikipedia.org/wiki/ASCII#/media/File:ASCII-Table-wide.svg

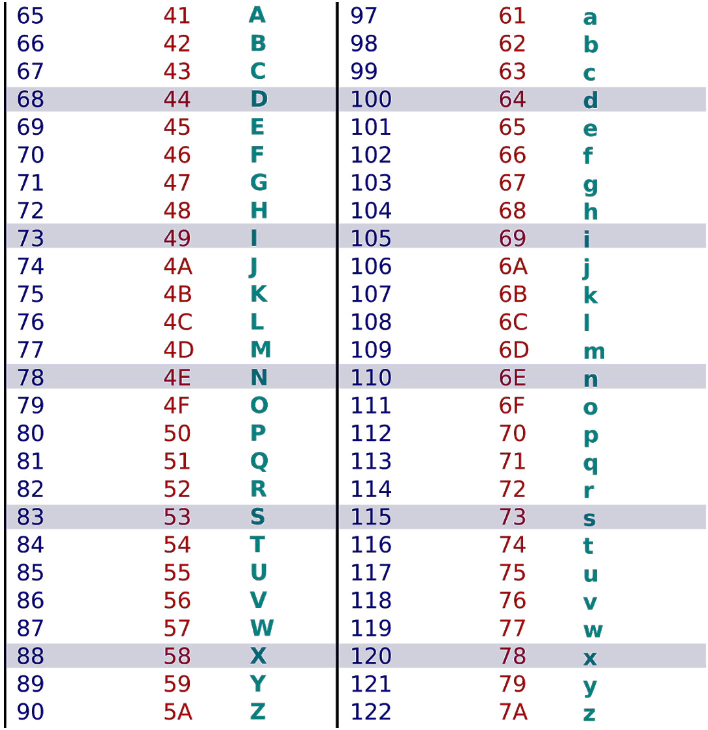
Let's Decode!

010010000110010101111001

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

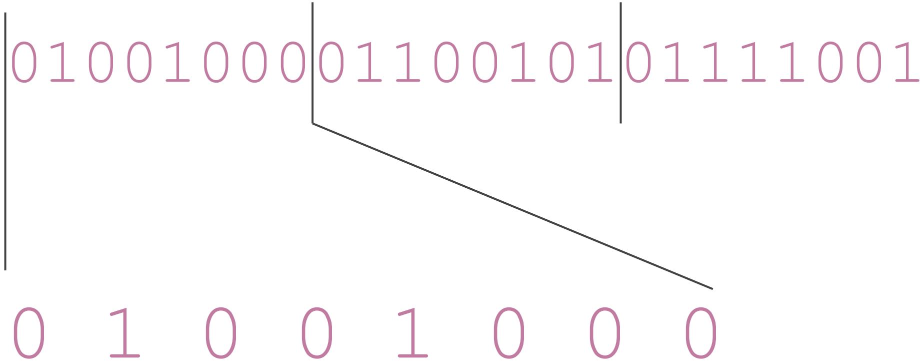
Let's Decode!

01001000|01100101|01111001



65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

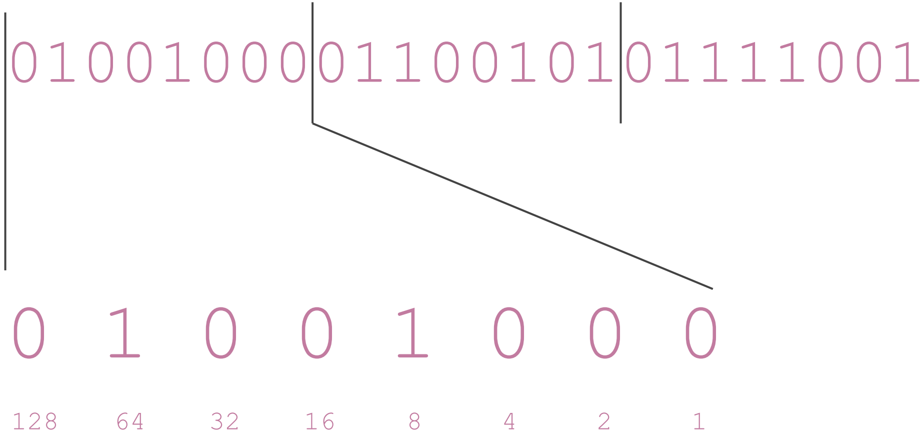


010010000110010101111001

0 1 0 0 1 0 0 0

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



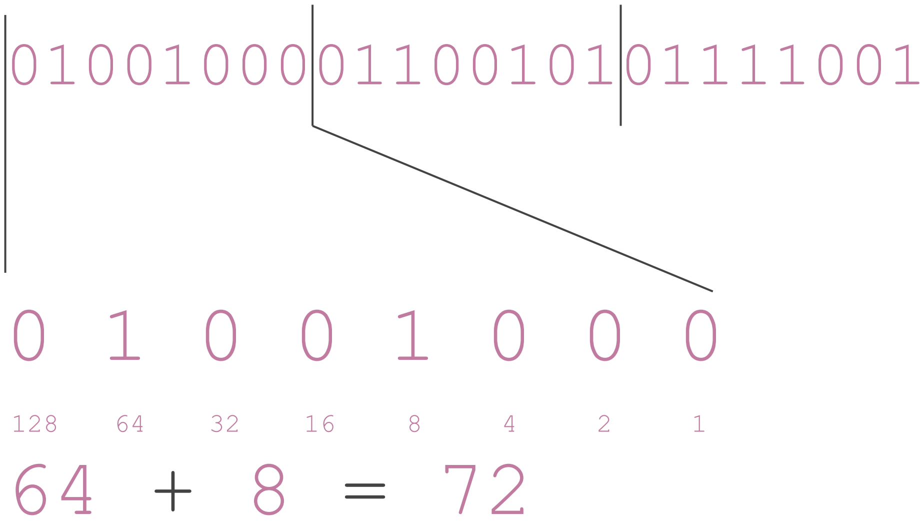
01001000|01100101|01111001

0 1 0 0 1 0 0 0

128 64 32 16 8 4 2 1

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



01001000 01100101 01111001

0	1	0	0	1	0	0	0
128	64	32	16	8	4	2	1

64 + 8 = 72

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

01001000 01100101 01111001

0	1	0	0	1	0	0	0
128	64	32	16	8	4	2	1

64 + 8 = 72

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

01001000 01100101 01111001

H

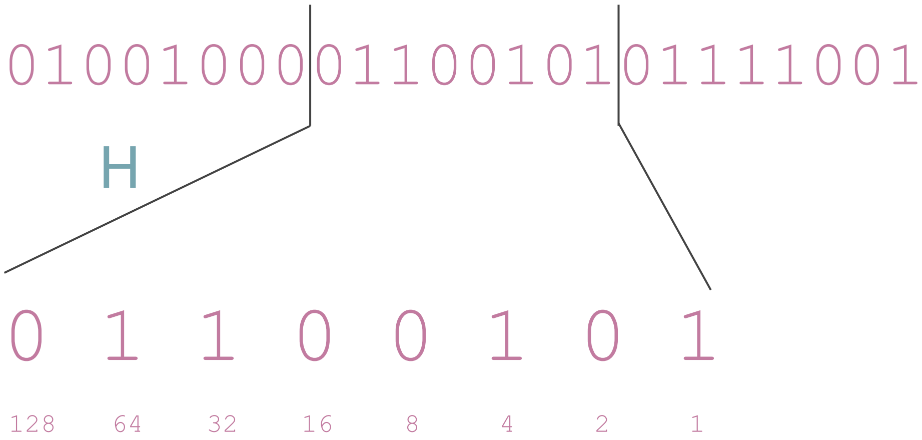
0 1 0 0 1 0 0 0

128 64 32 16 8 4 2 1

64 + 8 = 72

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



010010000110010101111001

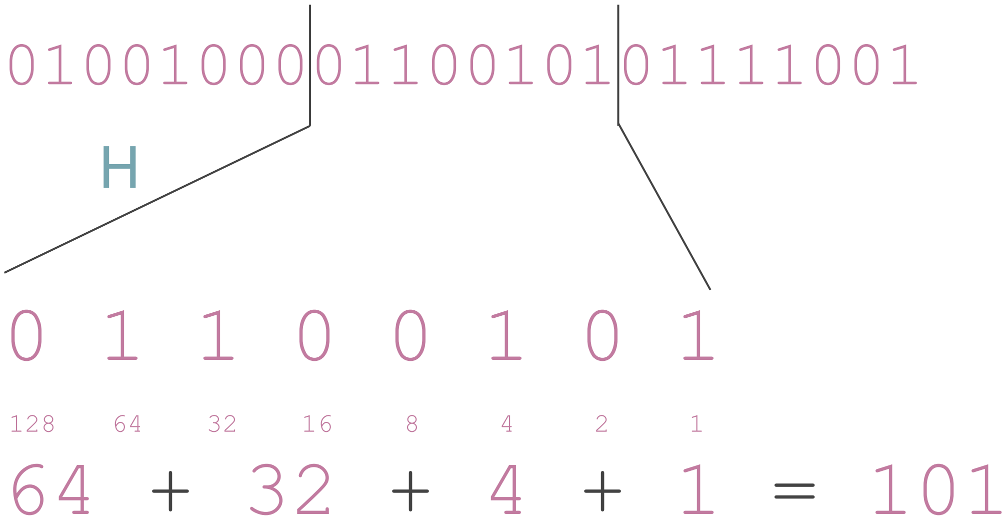
H

0 1 1 0 0 1 0 1

128 64 32 16 8 4 2 1

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



010010000110010101111001

H

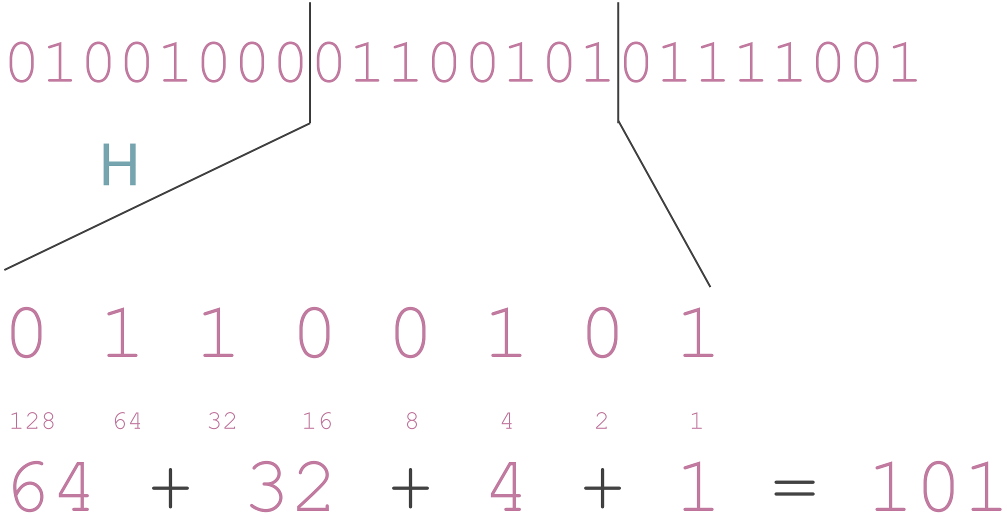
0 1 1 0 0 1 0 1

128 64 32 16 8 4 2 1

64 + 32 + 4 + 1 = 101

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



010010000110010101111001

H

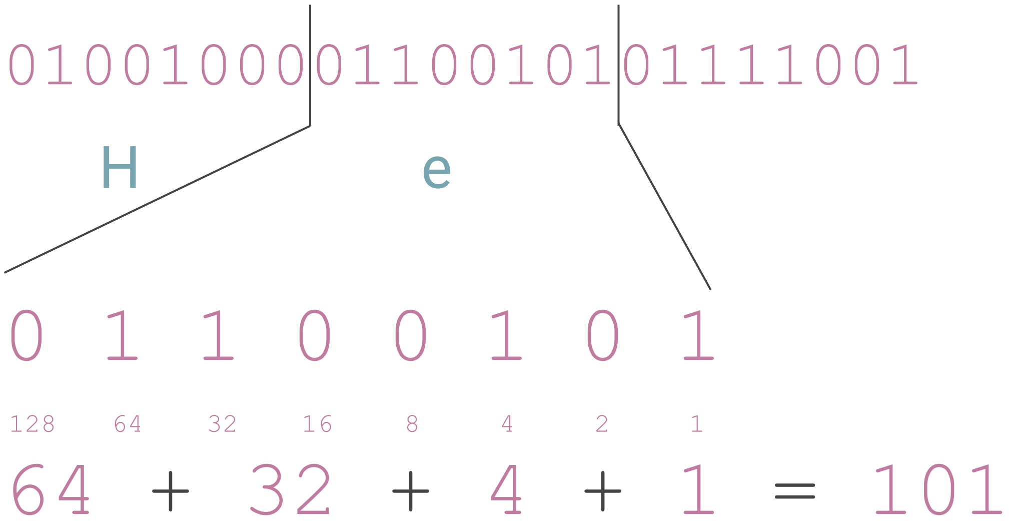
0 1 1 0 0 1 0 1

128 64 32 16 8 4 2 1

$64 + 32 + 4 + 1 = 101$

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



010010000110010101111001

H e

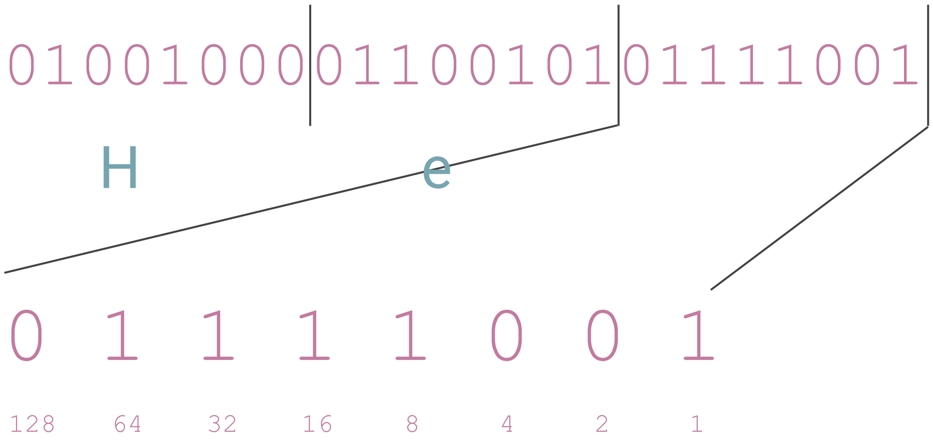
0 1 1 0 0 1 0 1

128 64 32 16 8 4 2 1

$64 + 32 + 4 + 1 = 101$

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

01001000 | 01100101 | 01111001

H

e

0 1 1 1 1 0 0 1

128 64 32 16 8 4 2 1

64 + 32 + 16 + 8 + 1 = 121

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

01001000 | 01100101 | 01111001

H e

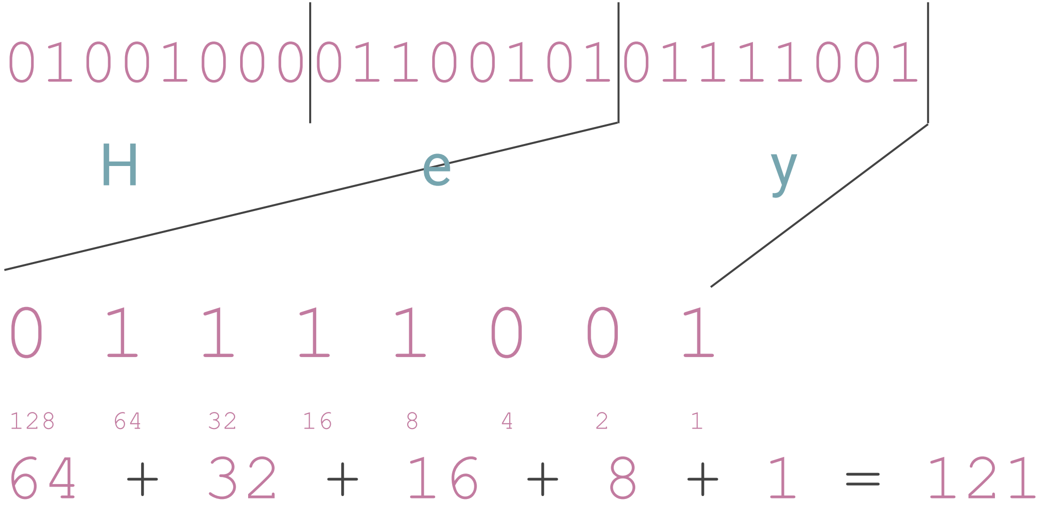
0 1 1 1 1 0 0 1

128 64 32 16 8 4 2 1

64 + 32 + 16 + 8 + 1 = 121

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!



01001000 01100101 01111001

H e y

0 1 1 1 1 0 0 1

128 64 32 16 8 4 2 1

64 + 32 + 16 + 8 + 1 = 121

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

Let's Decode!

01001000 | 01100101 | 01111001 |

H e y

65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z

ASCII in Python

If we want to translate a character to its ASCII value in Python, we can use the `ord()` function on it!

```
print(ord("a"))
```

```
97
```

If we want to translate an ASCII value to a character, we can use the `chr()` function!

```
print(chr(65))
```

```
A
```

Code to translate a binary sequence to text

```
def bin_to_text(binary_sequence):
    translated_text = ""
    for i in range(0, len(binary_sequence), 8):
        current_byte = binary_sequence[i:i+8]
        ascii_value = bin_to_dec(current_byte)
        character = chr(ascii_value)
        translated_text += character

    return translated_text
```