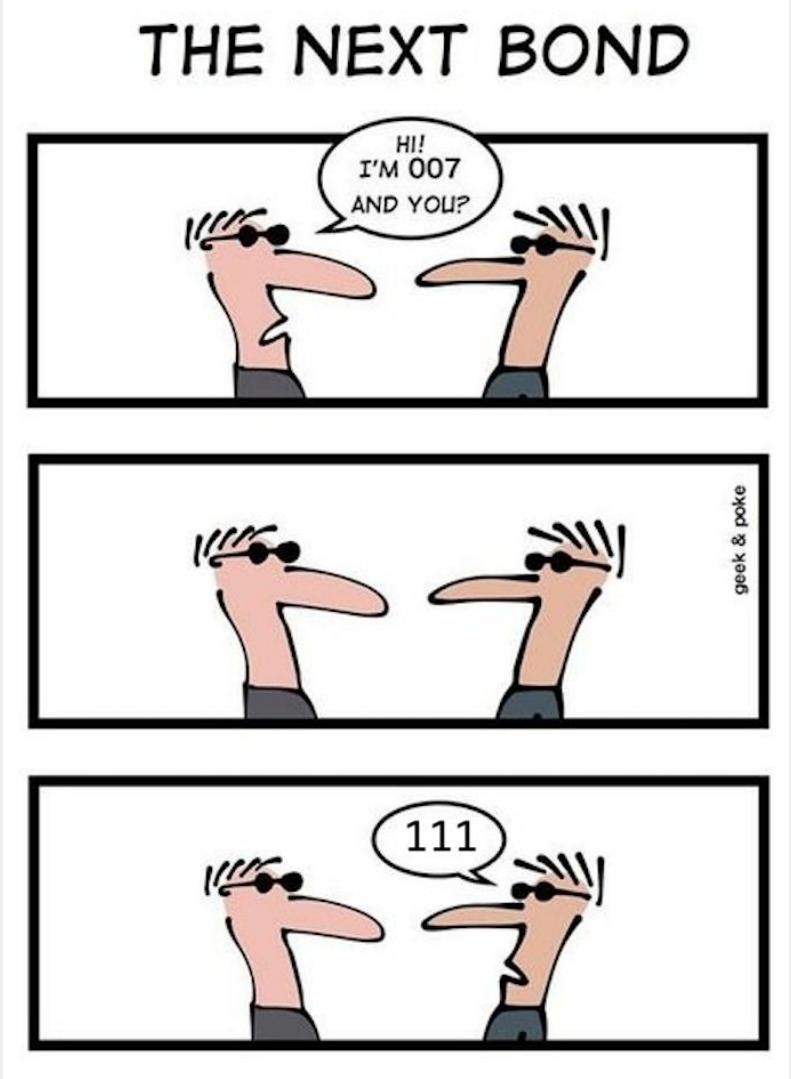
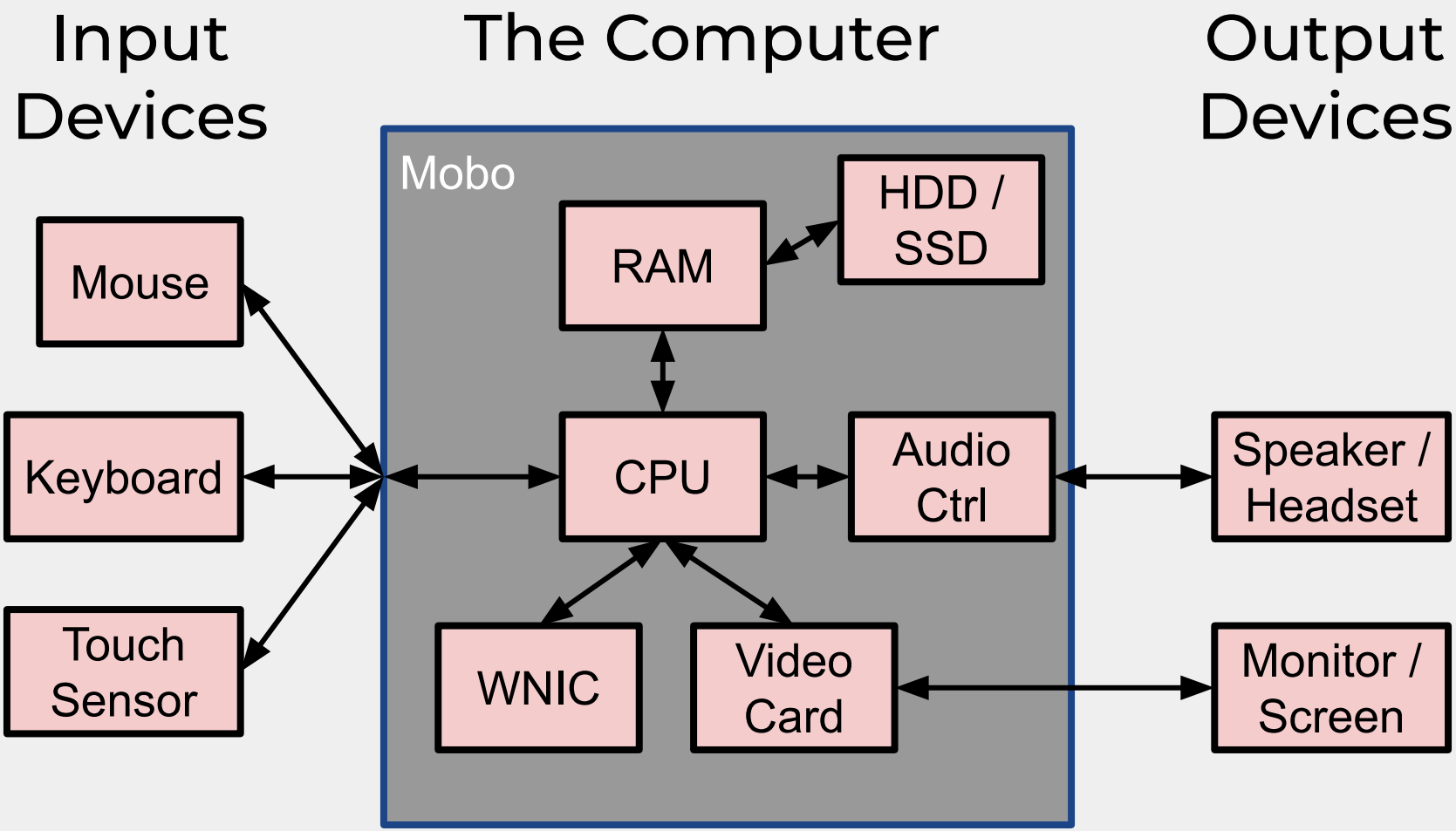
CS 110
Binary

Benjamin Dicken




Input Devices
The Computer
Output Devices
Mobo
Mouse
Keyboard
Touch Sensor
RAM
HDD / SSD
CPU
Audio Ctrl
WNIC
Video Card
Speaker / Headset
Monitor / Screen

Representing Information

- Computers store information on Hard Drive Disk (HDD) and/or SSD (Solid State Drive)
  - Both HDDs and SSDs are types of ***Hard Drives***
- They also store information on RAM
- Use ***Binary***
- This means that computers can only use ***1s*** and ***0s*** for storing information
  - This includes words, images, programs, etc

Representing Information

- One common type of hard drive today is the SSD (Solid State Drive)
- As solid state drive uses tiny electrical components called ***floating gate transistors (FGT)*** to store each 1 and zero
- A single SSD can have millions, billions, or even trillions of ***FGTs*** in them

Representing Information

- How many bits (1’s and zeros) can a 500 gigabyte hard-drive store?

Representing Information

- How many bits (1’s and zeros) can a 500 gigabyte hard-drive store?

4,294,967,296,000

01001000 01101111 01110111 00100000
01100100 01101111 01100101 01110011
00100000 01100010 01101001 01101110
01100001 01110010 01111001 00100000
01110111 01101111 01110010 01101011
00111111

How does binary work?

Storing things in Binary

Spend some time thinking and develop a methodology of translating **English letters** to **only 1s and 0s**.

How would you go about it?

Storing things in Binary

Spend some time thinking and develop a methodology of translating **A Video** to **only 1s and 0s.**

How would you go about it?

Representing Information

- ***Decimal*** (also called ***base-10***) is the numeric representation that most here are used to
  - In decimal, there are ***ten digits*** to use for representing numeric values: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- ***Binary*** (also called ***base-2***) is just another way of representing numbers
  - In binary, there are ***two digits*** to use for representing numeric values: 0, 1

Representing Information

When we count in ***decimal***

0	7	14	21
1	8	15	22
2	9	16	...
3	10	17	
4	11	18	
5	12	19	
6	13	20	

When we count in ***binary***

0	111
1	1000
10	1001
11	1010
100	1011
101	1100
110	1101

. . . . .

Count

Using the counting technique to determine what the binary representation of the value **19** would be

No computers!

Count

Using the counting technique to determine what the binary representation of the value **223** would be

No computers!

Representing Information

- For every binary number, there is an equivalent decimal number
- When computers retrieve, process, modify, and store information, uses binary representation (ignoring quantum)
- When we talk about information being represented by numbers we will often refer to a ***decimal*** number, but the computer is really using ***binary*** internally

Converting from Decimal To Binary

Converting from Decimal To Binary

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

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	64	32	16	8	4	2	1

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Convert 147 to binary

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	64	32	16	8	4	2	1

Convert 147 to binary

147-128 = 19

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Convert 147 to binary

147-128 = 19 **19-16 = 3**

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Convert 147 to binary

147-128 = 19 **19-16 = 3** **3-2 = 1**

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Convert 147 to binary

147-128 = 19 **19-16 = 3** **3-2 = 1** **1-1 = 0**

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	64	32	16	8	4	2	1

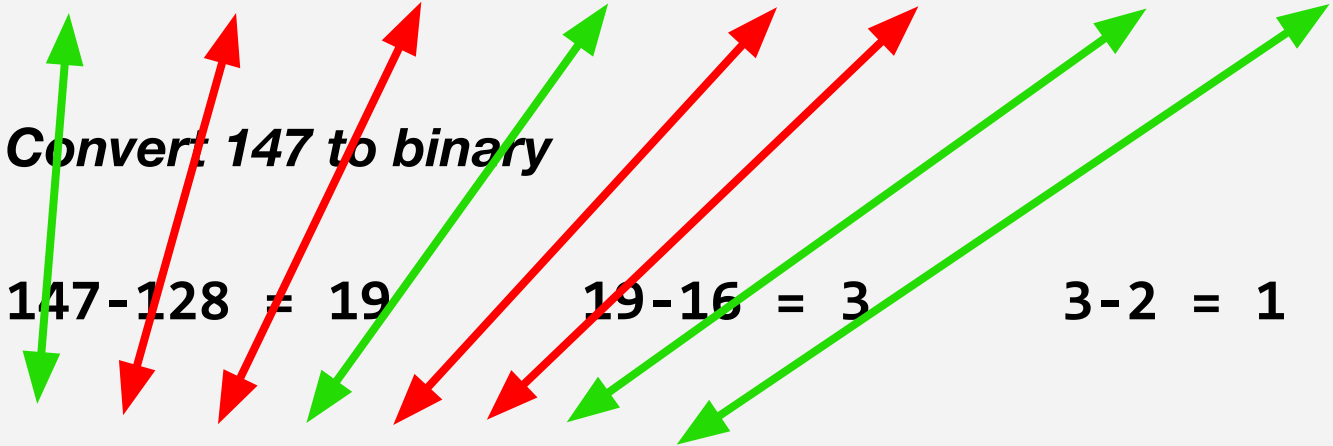
Convert 147 to binary

147-128 = 19 **19-16 = 3** **3-2 = 1** **1-1 = 0**

1 0 0 1 0 0 1 1

Converting from Decimal To Binary

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	64	32	16	8	4	2	1



Convert 147 to binary

147-128 = 19 19-16 = 3 3-2 = 1 1-1 = 0

1 0 0 1 0 0 1 1

Convert to Binary

- Middle: **171**
- Sides: **98**

Convert to Binary

- Middle: **171** **10101011**
- Sides: **98**

Convert to Binary

- Middle: **171** **10101011**
- Sides: **98** **01100010**

How do we go from binary to decimal?

How do we go from binary to decimal?

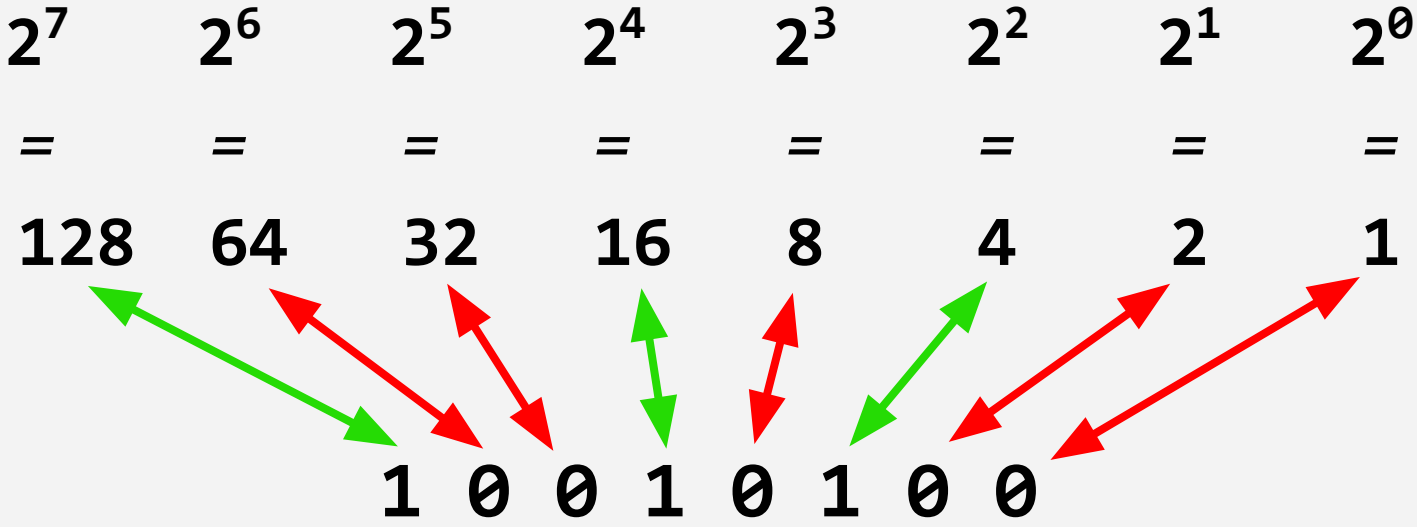
1 0 0 1 0 1 0 0

How do we go from binary to decimal?

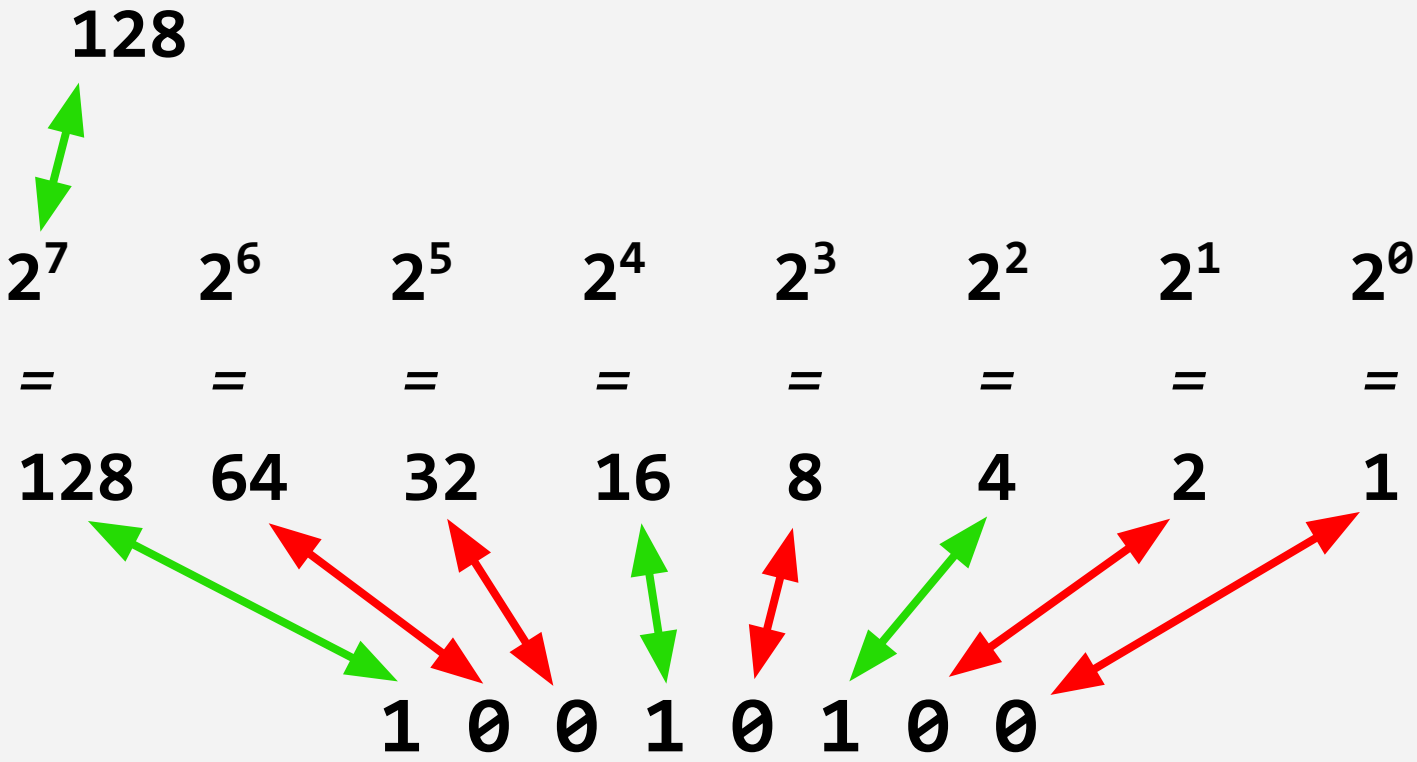
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
=	=	=	=	=	=	=	=
128	64	32	16	8	4	2	1

1 0 0 1 0 1 0 0

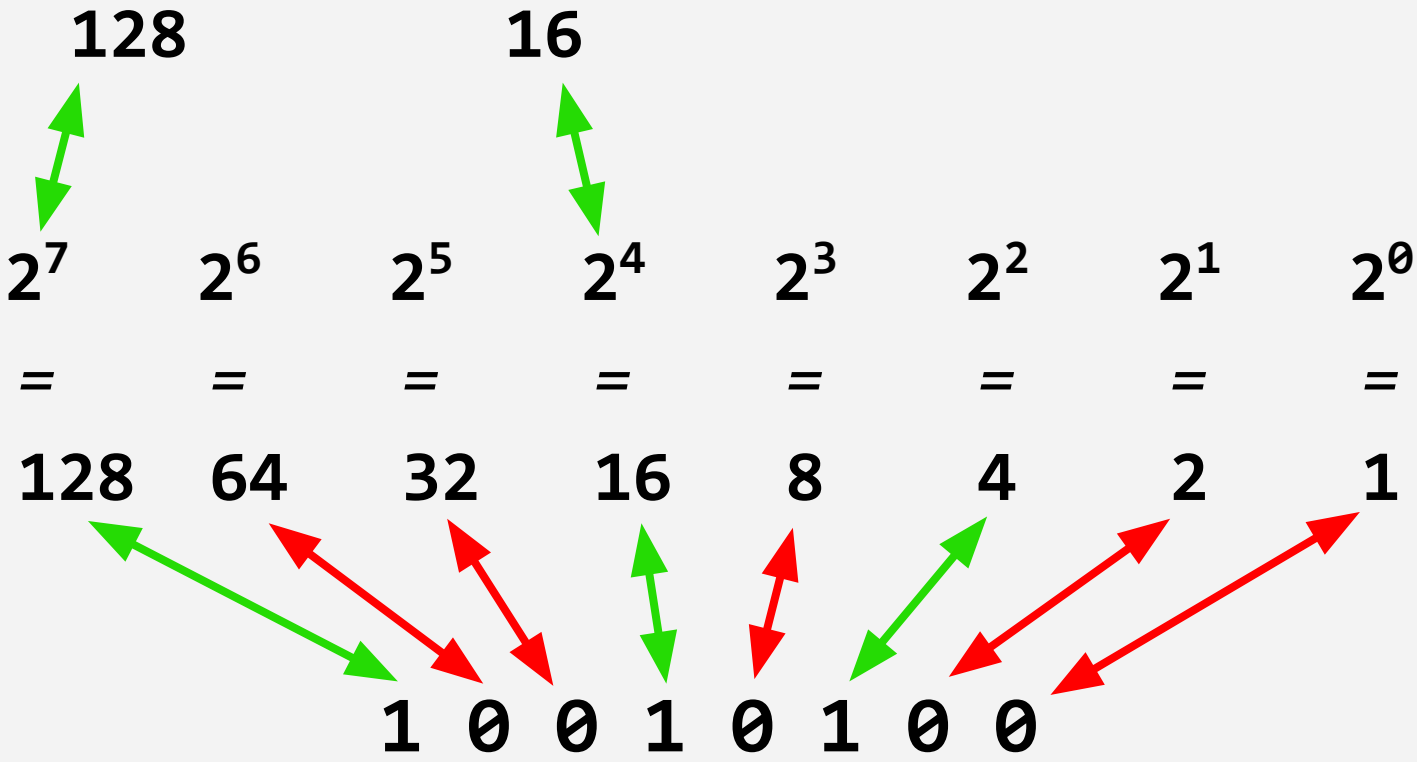
How do we go from binary to decimal?



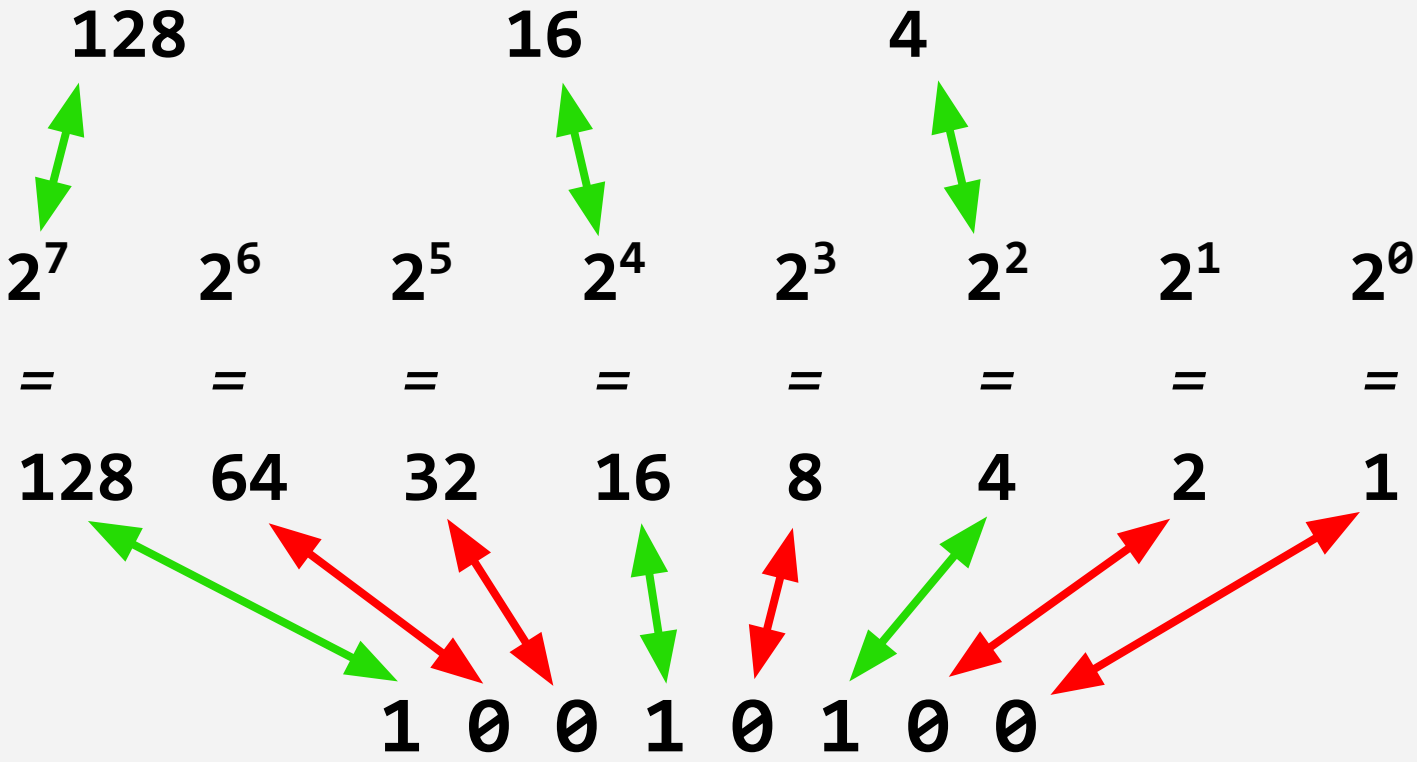
How do we go from binary to decimal?



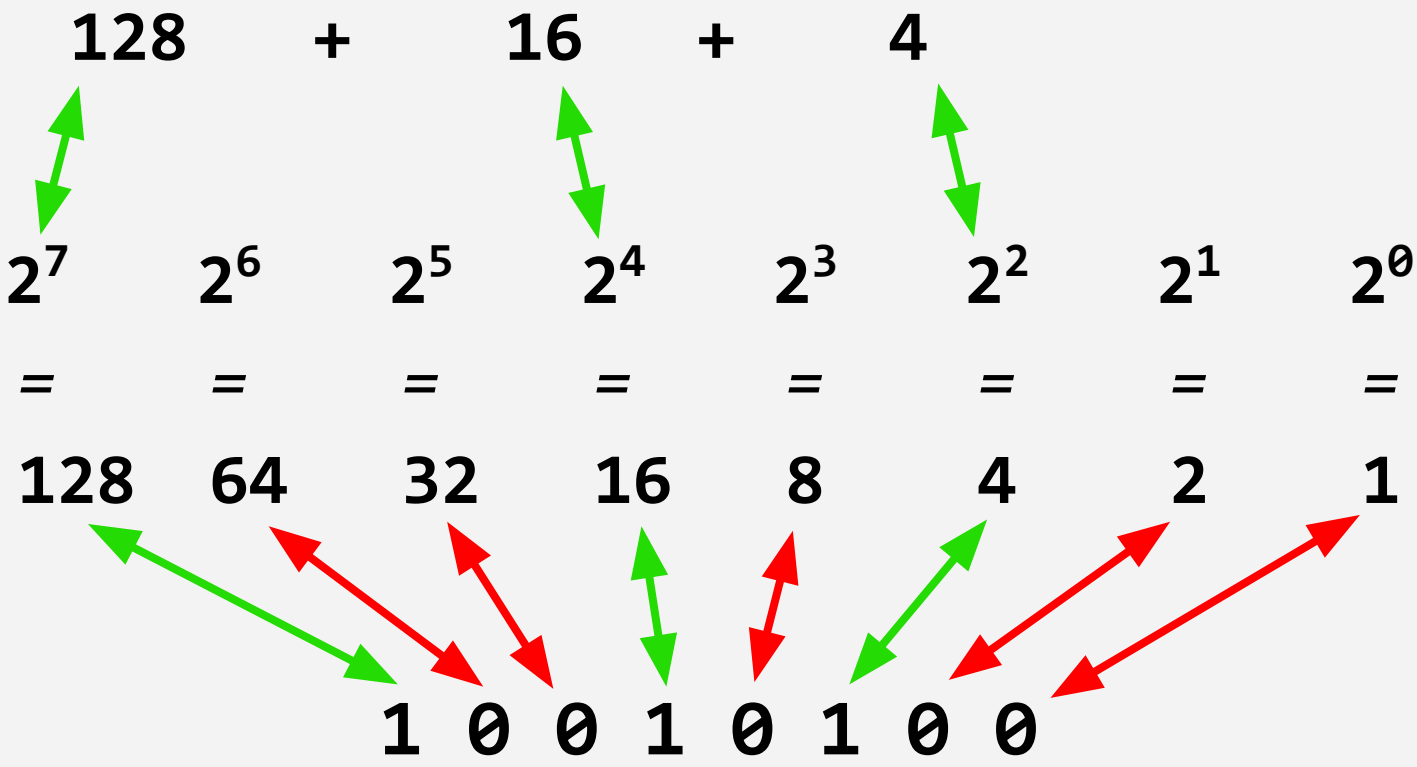
How do we go from binary to decimal?



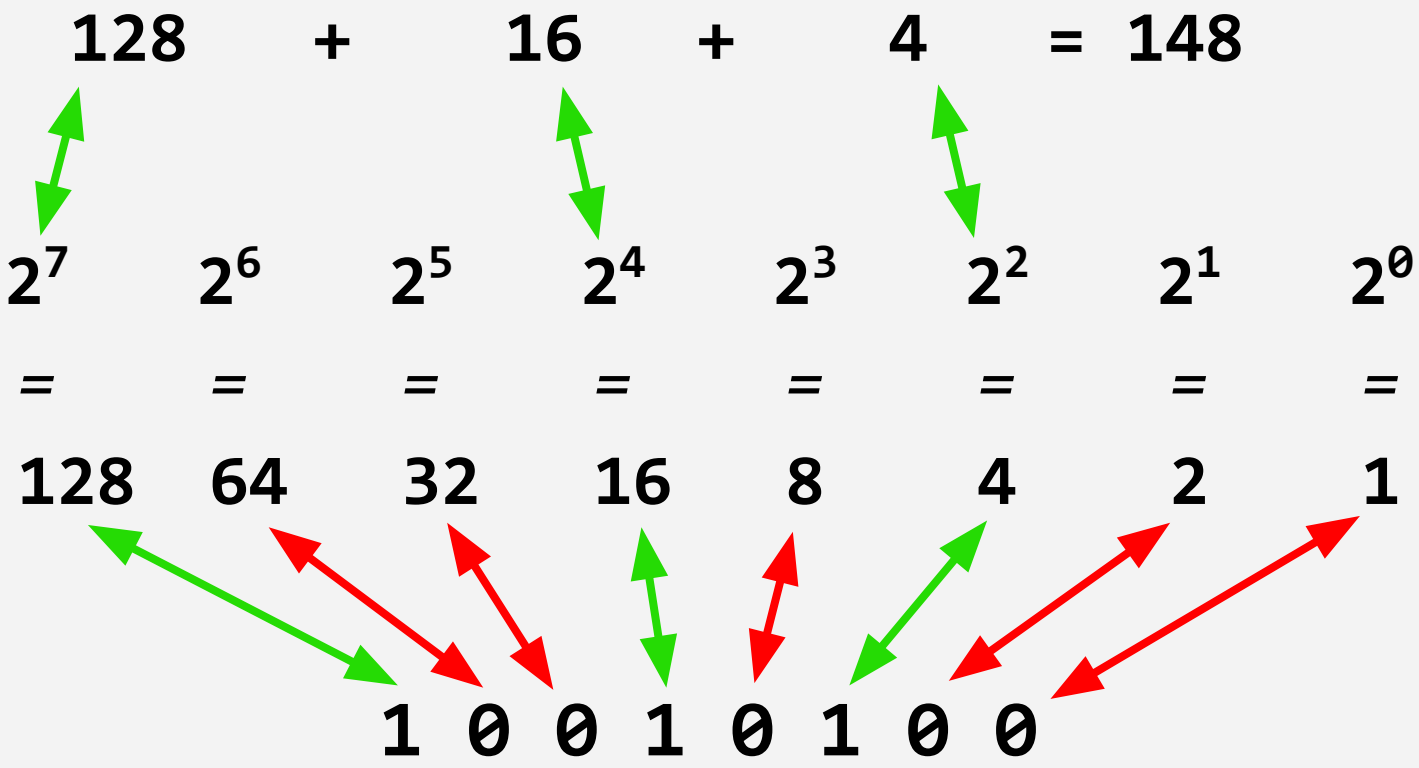
How do we go from binary to decimal?



How do we go from binary to decimal?



How do we go from binary to decimal?



Convert from Binary

- Middle: **`1010111`**
- Sides: **`0011111`**

Convert from Binary

- Middle: **`1010111`** **`87`**
- Sides: **`0011111`**

Convert from Binary

- Middle: **1010111** **87**
- Sides: **0011111** **31**

Binary Conversion

- What if we want to convert a larger number?

Binary Conversion

- What if we want to convert a larger number?
  - **Middle:** 787 to binary
  - **Sides:** 515 to binary

Binary Conversion

- What if we want to convert a larger number?
  - **Middle:** 787 to binary `1100010011`
  - **Sides:** 515 to binary

Binary Conversion

- What if we want to convert a larger number?
  - **Middle:** 787 to binary `1100010011`
  - **Sides:** 515 to binary `1000000011`

Large conversion

Middle: `1010101011011101`

Sides: `1110110111100100`

Large conversion

Middle: `10101 01010 11101`

Sides: `11101 10111 00100`

Large conversion

Middle: `10101 01010 11101`

Sides: `11101 10111 00100`

`21853`

Large conversion

Middle: `10101 01010 11101`

Sides: `11101 10111 00100`

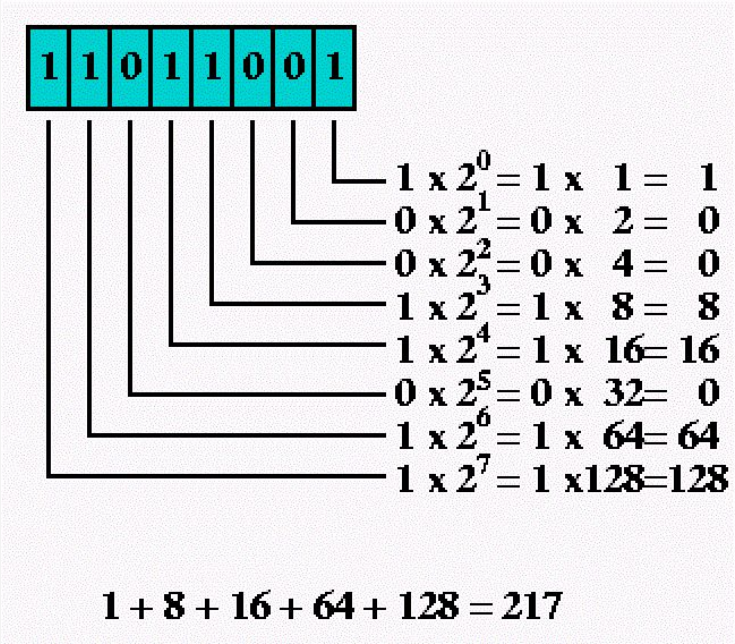
`21853`

`30436`

Converting from Binary

- Take an 8-bit binary string as input
- Print out the resulting decimal number
- For instance:

```
Enter binary number:
10101010
Decimal: 170
```



```
binary_string = input('Enter binary number:\n')
```

```
binary_string = input('Enter binary number:\n')

decimal_number = 0
i = ???
while i >= 0:
    ## What goes here?
```

```
binary_string = input('Enter binary number:\n')

decimal_number = 0
i = ???
while i >= 0:
    ## What goes here?

print('Decimal:', decimal_number)
```

Activity

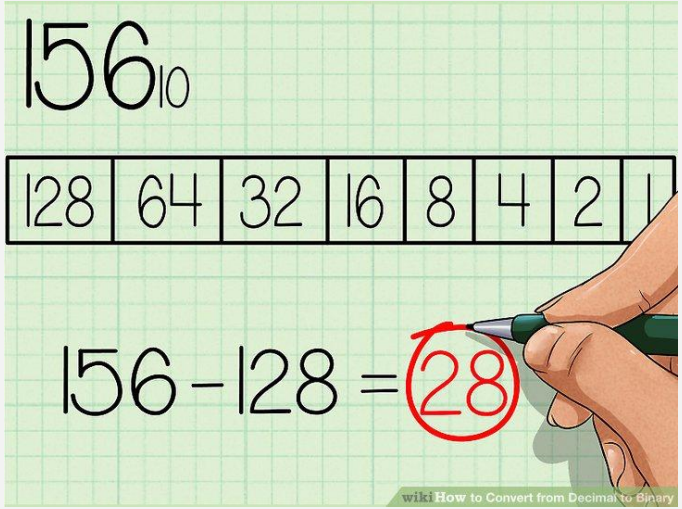
```
binary_string = input('Enter binary number:\n')

decimal_number = 0
i = ???  ## What should i start at ?
while i >= 0:
    ## What goes here?

print('Decimal:', decimal_number)
```

```
binary_string = input('Enter binary number:\n')

decimal_number = 0
i = len(binary_string) - 1
pow = 0
while i >= 0:
    if binary_string[i] == '1':
        decimal_number += 2 ** pow
    pow += 1
    i -= 1

print('Decimal:', decimal_number)
```

Converting from Decimal

- Take a decimal integer as input (max 255)
- Print out the resulting binary string data
- For instance:

```
Enter Decimal number less than 256:
125
Binary: 01111101
```



```
decimal_number = int(input('Enter Decimal number less than 256:\n'))
power = 7
binary_string = ''
while power >= 0:
    # What goes here?
    power -= 1
print(binary_string)
```

Activity

```
decimal_number = int(input('Enter Decimal number less than 256:\n'))
power = 7
binary_string = ''
while power >= 0:
    # What goes here?
    power -= 1
print(binary_string)
```

```
decimal_number = int(input('Enter Decimal number less than 256:\n'))
power = 7
binary_string = ''
while power >= 0:
    power_val = 2 ** power
    if decimal_number >= power_val:
        binary_string += '1'
        decimal_number -= power_val
    else:
        binary_string += '0'
    power -= 1
print(binary_string)
```

```
decimal_number = int(input('Enter Decimal number less than 256:\n'))
power = 7
binary_string = ''
while power >= 0:
    power_val = 2 ** power
    if decimal_number >= power_val:
        binary_string += '1'
        decimal_number -= power_val
    else:
        binary_string += '0'
    power -= 1
print(binary_string)
```

What would happen if the input was 2000 ?

Write the code on the whiteboard

- Write a program that
  - Accepts a binary number (not just 8-bit ones, any length) as input
  - Reports to the user how many 1's and 0's were in the string
  - For example:

```
Enter binary number:
1101011101010000010
    1s: 9
    0s: 10
```

```
binary_string = input('Enter binary number:\n')
count_0 = 0
count_1 = 0

### What goes here?

print('    0s:', count_0)
print('    1s:', count_1)
```

```
binary_string = input('Enter binary number:\n')
count_0 = 0
count_1 = 0
i = 0
while i < len(binary_string):
    if binary_string[i] == '0':
        count_0 += 1
    elif binary_string[i] == '1':
        count_1 += 1
    i += 1

print('    0s:', count_0)
print('    1s:', count_1)
```

```
binary_string = input('Enter binary number:\n')
count_0 = 0
count_1 = 0
i = 0
while i < len(binary_string):
    if binary_string[i] == '0':
        count_0 += 1
    elif binary_string[i] == '1':
        count_1 += 1
    i += 1
    # LOCATION
print('    0s:', count_0)
print('    1s:', count_1)
```

Loop table for **i** , **count_0**, and **count_1** based on this location, for input:

1010

```
binary_string = input('Enter binary number:\n')
count_0 = 0
count_1 = 0
i = 0
while i < len(binary_string):
    if binary_string[i] == '0':
        count_0 += 1
    elif binary_string[i] == '1':
        count_1 += 1
    i += 1
    # LOCATION
print('    0s:', count_0)
print('    1s:', count_1)
```

i	count_0	count_1
1	0	1
2	1	1
3	1	2
4	2	2

Write the code on the whiteboard

- Write a program that
  - Accepts a string of digits as input
  - Outputs them like so:

```
Enter string of digits
12501103
1 -> 2
5 -> 0
1 -> 1
0 -> 3
```