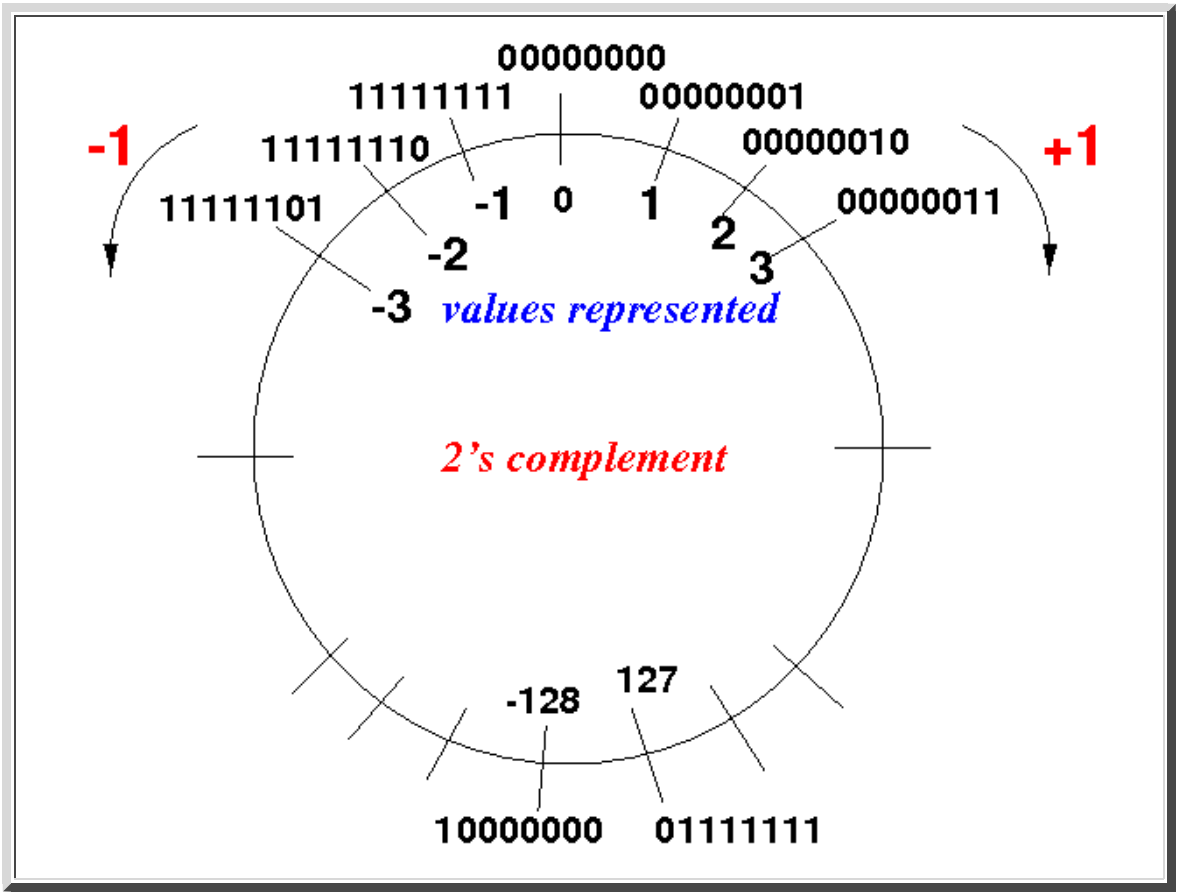
Excess 2^{n-1} encoding

- **2's complement encoding**
  - The **2's complement encoding** makes the **assumption** that:
    - The **representation** for the **value 0** is: `00000000`

    This results in the **following assignment scheme** of **binary numbers** and **values**:



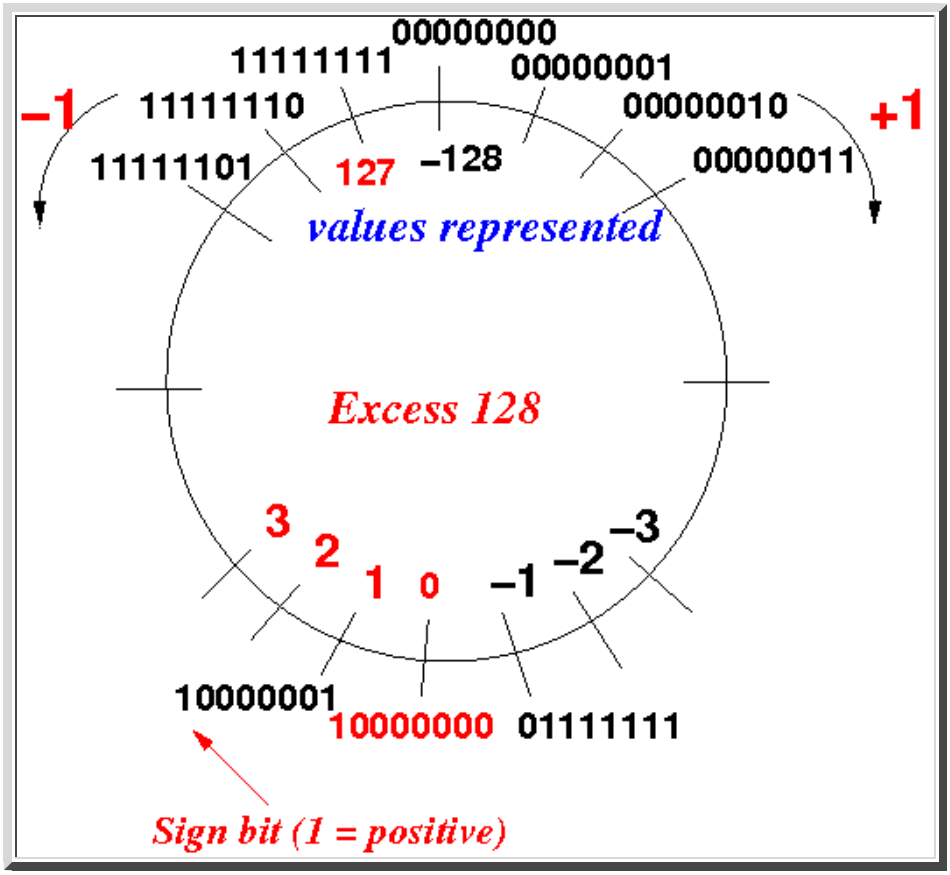
- **Excess 2^{n-1} encoding**
  - There is **another way** to obtain a **representation** of **signed numbers**
  - The **Excess 2^{n-1} encoding** makes the **assumption** that:
    - The **representation** for the **value 0** is: `1000...0000`
      - The **binary number** `1000...0000` of ***n* bits** is the representation for the value 2^{n-1}

        (Hence the encoding is called **Excess 2^{n-1} encoding**)

- **Example: Excess 2^{n-1} encoding with *n* = *8* bits**
  - The **Excess 128 (= 2^{8-1})** encoding scheme makes the **assumption** that:

    - The **representation** for the **value 0** is: `10000000`

    This results in the **following assignment scheme** of **binary numbers** and **values**:



- The **Excess 128 encoding** scheme is as follows:

```
Code      Value
================
00000000  -128  <--- smallest negative value with 8 bits (-2^7)
00000001  -127
.....
01111000    -8
01111001    -7
01111010    -6
01111011    -5
01111100    -4
01111101    -3
01111110    -2
01111111    -1
10000000     0
10000001     1
10000010     2
10000011     3
10000100     4
10000101     5
10000110     6
10000110     6
10000111     7
10001000     8
.....
11111111   127  <--- largest positive value with 8 bits (2^7-1)
```

Note:

> - The **representation** `00000000` represents the **value −128** !!!
>
>   This is **not** a **typo** !!!

- **Wikipedia** calls this scheme ***offset binary***: click here

---

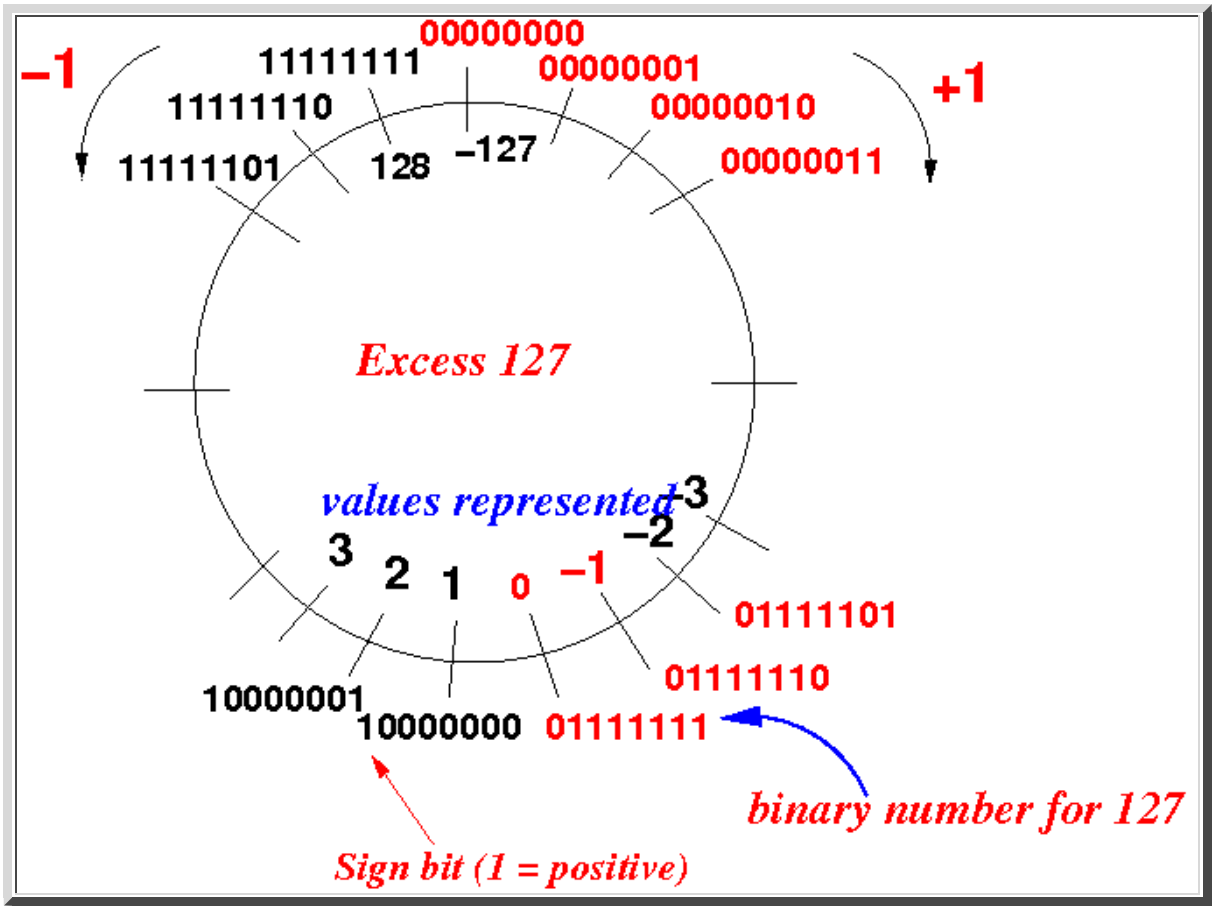
- **Another possible offset: $2^{n-1} - 1$**
  - There is **another commonly used** choice for the **representation** of the **value 0**
  - The **Excess 2^{n-1} −1 encoding** makes the **assumption** that:

> - The **representation** for the **value 0** is: `011111...11111`
>
>> - The **binary number** `011111...11111` of ***n* bits** is the representation for the value **2^{n-1} −1**
>>
>>   (Hence the encoding is called **Excess 2^{n-1} −1 encoding**)

- **Example: Excess 2^{n-1}−1 encoding with *n* = *8* bits**
  - The **Excess 127 (= 2^{8-1}−1)** encoding scheme makes the **assumption** that:

- The **representation** for the **value 0** is: `01111111`

This results in the **following assignment scheme** of **binary numbers** and **values**:



- The **Excess 127 encoding** scheme is as follows:

```
Code      Value
================
00000000  -127  <--- smallest negative value with 8 bits (-2^7)
00000001  -126
.....
01111000    -7
01111001    -6
01111010    -5
01111011    -4
01111100    -3
01111101    -2
01111110    -1
01111111     0
10000000     1
10000001     2
10000010     3
10000011     4
10000100     5
10000101     6
10000110     7
10000110     8
10000111     9
10001000    10
.....
11111111   128  <--- largest positive value with 8 bits (2^7-1)
```

Note:

- The **representation** `00000000` represents the **value** **−127** !!!

  This is **not** a **typo** !!!

- **Note:**

  - We will see the **excess 127** encode again when we discuss **floating point number representation**
  - **Excess 127** is used to represent the **exponent** of the **single precision** IEEE 754 standard.
