# Floating Point Representation (1/2)

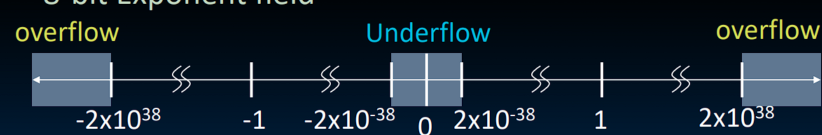
- Normal format: $+1.xxx...x_{two} * 2^{yyy...y}{}_{two}$
- Multiple of Word Size (32 bits)


- **S represents Sign**
- **Exponent represents y's**
- **Significand represents x's**
- **Represent numbers as small as $1.2 \times 10^{-38}$ to as large as $3.4 \times 10^{38}$**

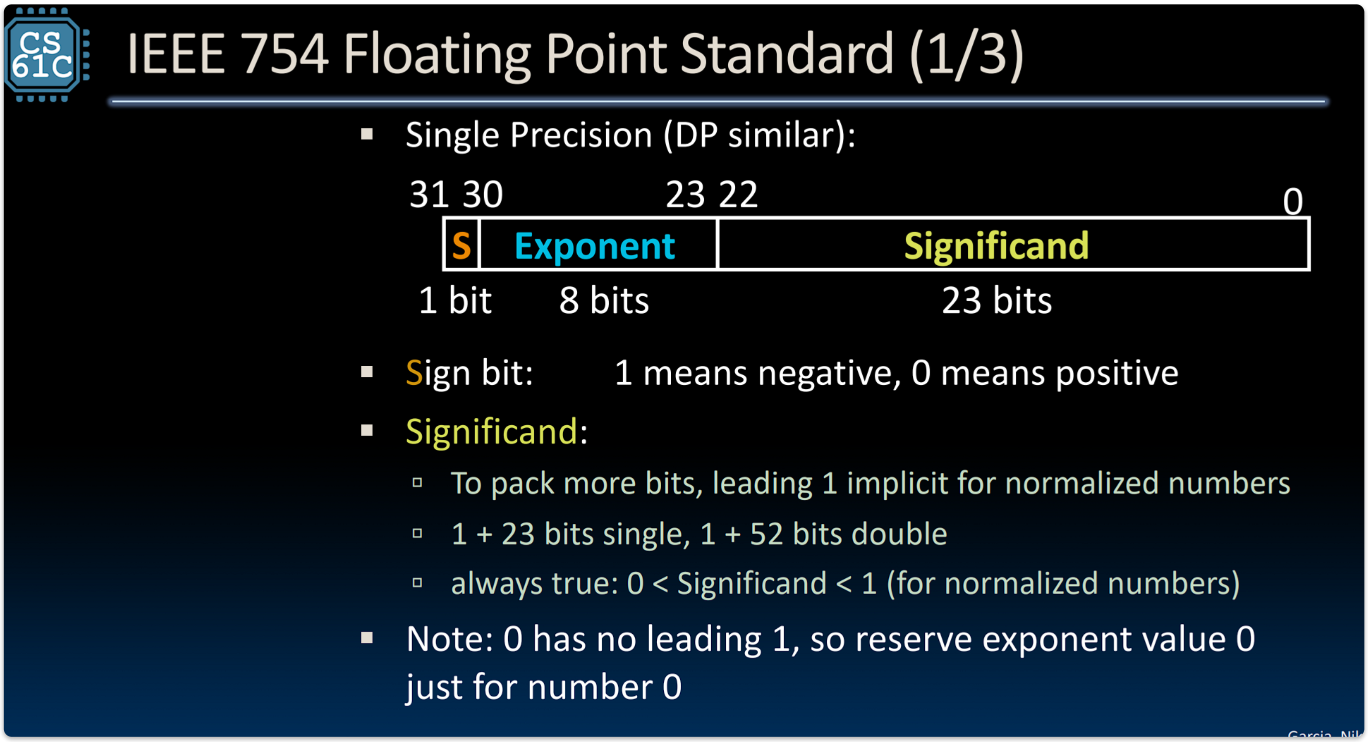



# Floating Point Representation (2/2)

- What if result too large?
  - ($> 3.4 \times 10^{38}$ , $< -3.4 \times 10^{38}$ )
  - Overflow! → Exponent larger than represented in 8-bit Exponent field
- What if result too small?
  - ($>0$ and $< 1.2 \times 10^{-38}$ , $<0$ and $> -1.2 \times 10^{-38}$ )
  - Underflow! → Negative exponent larger than represented in 8-bit Exponent field




- What would help reduce chances of overflow and/or underflow?




**1) 格式化值**

当指数段 exp 的位模式既不全为 0（即数值 0），也不全为 1（即单精度数值为 255，以单精度数为例， 8 位的指数为可以表达 0~255 的 255 个指数值；双精度数值为 2047）的时候，就属于这类情况。如图 2 所示。

| | | |
|---|---|---|
| S | $\neq 0 \& \neq 255$ | f |

图 2

我们知道，指数可以为正数，也可以为负数。为了处理负指数的情况，实际的指数值按要求需要加上一个偏置（Bias）值作为保存在指数段中的值。因此，这种情况下的指数段被解释为以偏置形式表示的有符号整数。即指数的值为：$E=e^{-Bias}$

其中，e 是无符号数，其位表示为 $e^{k-1}...e^1e^0$，而 Bias 是一个等于 $2^{k-1}-1$（单精度是 127，双精度是 1023）的偏置值。由此产生指数的取值范围是：单精度为 -126~+127，双精度为 -1022~+1023。

对小数段 frac，可解释为描述小数值 f，其中 $0 \le f < 1$，其二进制表示为 $0.f_{n-1}...f_1f_0$，也就是二进制小数点在最高有效位的左边。有效数字定义为 M=1+f。有时候，这种方式也叫作隐含的以 1 开头的表示法，因为我们可以把 M 看成一个二进制表达式为 $1.f_{n-1}f_{n-2}...f_0$ 的数字。既然我们总是能够调整指数 E，使得有效数字 M 的范围为 $1 \le M < 2$（假设没有溢出），那么这种表示方法是一种轻松获得一个额外精度位的技巧。同时，由于第一位总是等于 1，因此我们就不需要显式地表示它。拿单精度数为例，按照上面所介绍的知识，实际上可以用 23 位长的有效数字来表达 24 位的有效数字。比如，对单精度数而言，二进制的 1001.101（即十进制的 9.625）可以表达为 $1.001101 \times 2^3$，所以实际保存在有效数字位中的值为：

- Represent 0?
  - exponent all zeroes
  - significand all zeroes
  - What about sign? Both cases valid.

```
+0: 0 00000000 00000000000000000000000
-0: 1 00000000 00000000000000000000000
```

- Reserve exponents, significands:

| Exponent | Significand | Object |
|---|---|---|
| 0 | 0 | 0 |
| 0 | nonzero | Denorm |
| 1-254 | anything | +/- fl. pt. # |
| 255 | 0 | +/- ∞ |
| 255 | nonzero | NaN |

## Sorting Requirement...

Computer Science 61C Fall 2021 — Wawrzynek and Weaver

- We can sort the sign field by just +/-...
  - Makes it easy to separate the two.. But what then?
- We need to sort by exponent + mantissa easily
  - Thus biased notation:
    An unsigned comparison between exponents Just Works
    - Bigger is larger
  - And the exponent is more significant, so it just sorts by exponent
  - And when the exponent is the same, the mantissa sorting Just Works
- So we can sort all positive numbers together just like they were integers
- And also an exponent of 0 isn't actually special...
  - The special exponents are MAX and MIN...

Berkeley EECS



| Exponent | Significand | Object |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | nonzero | Denorm |
| 1-254 | aynthing | Normal Floating Point |
| 255 | 0 | Infinity |
| 255 | Nonzero | NaN |

看一下使用二进制补码时的两种有用的快捷方式。第一种是对二进制补码求相反数的快速方法。简单地把每个 0 都转为 1 以及每个 1 都转为 0，然后对结果加 1。这个捷径是基于以下观察：一个数与其取反表达式的和一定是 $111 \ldots 111_2$，它表示 $-1$。由于 $x + \bar{x} = -1$，因此 $x + \bar{x} + 1 = 0$ 或 $\bar{x} + 1 = -x$。（用符号 $\bar{x}$ 表示 $x$ 按位取反。）