CS 24

Introduction to Computing Systems

Fixed-Width Integers

Idealized integers can be an **unbounded** number of bits. But, instruction sets work over specific numbers of bytes (e.g., the word size). For example, the `uint8_t` representation of 4 is `0b00000100`.

In general, if the word length is w, then $(b_{w-1} \cdots b_0)_2 = \sum_{i=0}^{w-1} b_i 2^i$.

Poll

What is the largest number representable by 4 bits?

- a 16
- b 15
- c 8
- d 7
- e ???

This takes care of **unsigned** integers, but how do we represent **signed integers**?

In general, if the word length is w, then

$$(b_{w-1}\cdots b_0)_2 = -b_{w-1}2^{w-1} + \left(\sum_{i=0}^{w-2} b_i 2^i\right)$$

$$\sum_{i=0}^{w-1} b_i 2^i$$

$w=4$

1000

$+0$

$-1 \cdot 2^3$

In general, if the word length is w, then

$$(b_{w-1} \cdots b_0)_2 = -b_{w-1} 2^{w-1} + \left(\sum_{i=0}^{w-2} b_i 2^i \right)$$

Poll

Which of these is the 8-bit two's complement representation of -1?

- a 0b11111111
- b 0b01111111
- c 0b10000000
- d 0b00010000
- e ???

$$\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

$$-2^{n+1} + 2^{n+1} - 1$$

In general, if the word length is w, then

$$(b_{w-1} \cdots b_0)_2 = -b_{w-1} 2^{w-1} + \left(\sum_{i=0}^{w-2} b_i 2^i \right)$$

In general, if the word length is w, then

$$(b_{w-1} \cdots b_0)_2 = -b_{w-1} 2^{w-1} + \left(\sum_{i=0}^{w-\mathbf{2}} b_i 2^i \right)$$

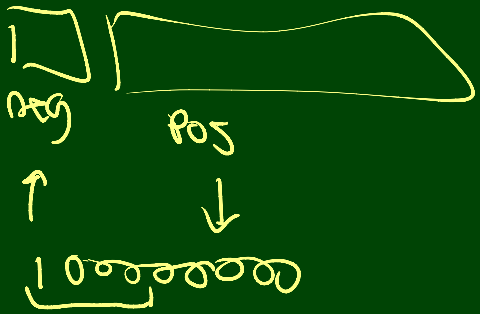
Poll

Which of these is the 16-bit two's complement representation of -1?

a. 0x1000
b. 0xF000
c. 0xFFFF
d. 0xEFFF
e. ???

A Program in x86-64


```
mystery:
    test %edi, %edi
    je   L2
L1:
    imul %edi, %esi
    add  $0xffffffff, %edi
    jne  L1
L2:
    mov  %esi, %eax
    retq
```



Special Integer Values

Base 16	**Unsigned**	**Signed**
Min	`0x0000...`	`0x8000...`
Max	`0xFFFF...`	`0x7FFF...`
-1	Not representable	`0xFFFF...`

Base 10	**Unsigned**	**Signed**
Min	0	-2^{w-1}
Max	$2^{w}-1$	$2^{w-1}-1$