1	2	4	8	16	32	64	128	256	512	1024
2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}

10 min to work

Notes

We represent numbers as:
$1028_{10} = 1 \cdot 10^3 + 0 \cdot 10^2 + 2 \cdot 10^1 + 8 \cdot 10^0$
base 10 = decimal

Binary is similar: $1101 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$

1 Unsigned Integers

1.1 If we have an n-digit unsigned numeral $d_{n-1}d_{n-2}\dots d_0$ in *radix* (or *base*) r, then the value of that numeral is $\sum_{i=0}^{n-1} r^i d_i$, which is just fancy notation to say that instead of a 10's or 100's place we have an r's or r^2's place. For the three radices binary, decimal, and hex, we just let r be 2, 10, and 16, respectively.

We don't have calculators during exams, so let's try this by hand. Recall that our preferred tool for writing large numbers is the IEC prefixing system:

Ki (Kibi) = 2^{10}	Gi (Gibi) = 2^{30}	Pi (Pebi) = 2^{50}	Zi (Zebi) = 2^{70}
Mi (Mebi) = 2^{20}	Ti (Tebi) = 2^{40}	Ei (Exbi) = 2^{60}	Yi (Yobi) = 2^{80}

Translation Recommendation graph!

binary ⟷ Hex

binary ⟷ decimal

(a) Convert the following numbers from their initial radix into the other two common radices:

1. 0b$\underbrace{1001}_{8+1=9}\underbrace{0011}_{2+1=3}$ = 147 = 0x93 $\quad 2^0 + 2^1 + 2^4 + 2^7 = 1 + 2 + 16 + 128 = 147$
2. 63 = 0b0011 1111 = 0x3F $\quad$ we know 64 = 0b0100 0000 so 63 = 64 − 1 so 0b$00\underbrace{11}_{3}\ \underbrace{1111}_{F}$. now it's easy to convert it to hex: 0x3F

   tricky
3. 0b$\underbrace{0010}_{2}\underbrace{0100}_{4}$ = 36 = 0x24 $\quad 2^2 + 2^5 = 4 + 32 = 36$
4. 0 = 0b0 = 0x0
5. 39 = 0b0010 0111 = 0x27 $\quad$ I know 32 in binary! (100000) $39 - 32 = 7$ I know 7 in binary (111) combine them: 0b$\underbrace{0010}_{0\times 2}\ \underbrace{0111}_{7}$
6. 437 = 0b0001 1011 0101 = 0x1B5 $\quad$ see final page for how to
7. 0x0123 = 0b0000 0001 0010 0011 = 291 $\quad 2^0 + 2^1 + 2^5 + 2^8$ $\quad 1 + 2 + 32 + 256 = 291$

(b) Convert the following numbers from hex to binary:

1. 0xD3AD = 0b1101 0011 1010 1101 = 54189 $\quad 2^0 + 2^2 + 2^3 + 2^5 + 2^7 + 2^8 + 2^9 + 2^{12} + 2^{14} + 2^{15}$
2. 0xB33F = 0b1011 0011 0011 1111 = 45887 $\quad 2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^8 + 2^9 + 2^{12} + 2^{13} + 2^{15}$
3. 0x7EC4 = 0b0111 1110 1100 0100 = 32452 $\quad 2^2 + 2^6 + 2^7 + 2^9 + 2^{10} + 2^{11} + 2^{12} + 2^{13} + 2^{14}$

(c) Write the following numbers using IEC prefixes:

- 2^{16} = 64 Ki $\quad 2^{10} \cdot 2^6$
- 2^{27} = 128 Mi $\quad 2^{20} \cdot 2^7$
- 2^{43} = 8 Ti $\quad 2^{40} \cdot 2^3$
- 2^{36} = 64 Gi $\quad 2^{30} \cdot 2^6$
- 2^{34} = 16 Gi $\quad 2^{30} \cdot 2^4$
- 2^{61} = 2 Ei $\quad 2^{60} \cdot 2^1$
- 2^{47} = 128 Ti $\quad 2^{40} \cdot 2^7$
- 2^{59} = 512 Pi $\quad 2^{50} \cdot 2^9$

(d) Write the following numbers as powers of 2:

- 2 Ki = 2^{11} $\quad 2^1 \cdot 2^{10}$
- 512 Ki = 2^{19} $\quad 2^9 \cdot 2^{10}$
- 16 Mi = 2^{24} $\quad 2^4 \cdot 2^{20}$
- 256 Pi = 2^{58} $\quad 2^8 \cdot 2^{50}$
- 64 Gi = 2^{36} $\quad 2^6 \cdot 2^{30}$
- 128 Ei = 2^{67} $\quad 2^7 \cdot 2^{60}$

minilecture on bias

Bias is taking an unsigned number, n and adding some number, the bias, to it to get the actual value you want to represent. Eg. If I wanted to represent -2 in 4 bits, what should my bias be? Well, it depends on the range! with 4 bits, I have $2^4 = 16$ values so I can set the bias to a lot of values $[-18, -2]$. Lets say I have a bias of $-8 = \left(\frac{-2^n}{2}\right)$, what is the unsigned value n I need to represent -2?

$n + \text{bias} = x$

$n + \cancel{-8} = -2 \; {+8}$

$n = 6$

So the unsigned value 6 represents -2 given a bias of -8!

Problems w/ Bias:

- Complicated Addition + subtraction!

Twos complement minilecture

We want to represent negative numbers AND have easy addition and subtraction! IDea: Make the largest bit negative and keep addition the same! So given 4 bits, each bits value is as follows $-2^3, 2^2, 2^1, 2^0$

$-8, 4, 2, 1$

We can add the bits values like unsigned! How do we get negative numbers? Flip the bits and add 1! So if I want -3, I write 3

0b0011, flip the bits

0b1100, then add one

0b1101 = -3 in 4 bit twos complement. Best part is addition stays the same! $4 - 3 = 4 + (-3)$

$4 =$ 0b0100

$-3 =$ 0b1101

$1 =$ 0b0001

Carry bits are the same so no overflow!

no a, b
give (min) for rest

Given these representations given, what is the value of 0b11001101?

Unsigned: $2^0+2^2+2^3+2^6+2^7$
$1+4+8+64+128$
$=205$

Twos comp: $2^0+2^2+2^3+2^6-2^7$
OR! Flip bits and add one!:
$00110010+1 = 00110011$
so $2^0+2^1+2^4+2^5$
$1+2+16+32=51$
& negate it since we flipped it!
-51

Bias: Take unsigned val + add bias: $205-127=78$
See how the same bits can be anything! And there is nothing indicating its representation!

2 Signed Integers

[2.1] Unsigned binary numbers work for natural numbers, but many calculations use negative numbers as well. To deal with this, a number of different schemes have been used to represent signed numbers, but we will focus on two's complement, as it is the standard solution for representing signed integers.

- Most significant bit has a negative value, all others are positive. So the value of an n-digit two's complement number can be written as $\sum_{i=0}^{n-2} 2^i d_i - 2^{n-1} d_{n-1}$.
- Otherwise exactly the same as unsigned integers.
- A neat trick for flipping the sign of a two's complement number: flip all the bits and add 1.
- Addition is exactly the same as with an unsigned number.
- Only one 0, and it's located at 0b0.

8 bits means 2^8 possible $=256$ numbers

For questions (a) through (c), assume an 8-bit integer and answer each one for the case of an unsigned number, biased number with a bias of -127, and two's complement number. Indicate if it cannot be answered with a specific representation.

(a) What is the largest integer? What is the result of adding one to that number?

1. Unsigned? 255, 0
2. Biased? 128, -127 largest number + Bias $= 255-127=128$
3. Two's Complement? 127, -128 Two's complement has one positive zero. zero takes up the extra positive number.

where n is the unsigned representation and x is the value of the number you wanted

$n + \text{bias} = x$
so $n = x - \text{bias}$

(b) How would you represent the numbers 0, 1, and -1?

1. Unsigned? 0b0000 0000, 0b0000 0001, N/A ← cannot represent signed numbers in unsigned
2. Biased? 0b0111 1111, 0b1000 0000, 0b0111 1110 0: $0-\text{Bias} = -(-127)=127$ | 1: $1-(-127)=128$ | -1: $-1-(-127)=126$
3. Two's Complement? 0b0000 0000, 0b0000 0001, 0b1111 1111

$\rightarrow -2^7+2^6+2^5+2^4+2^3+2^2+2^1+2^0 = -128+127=-1$
-or- flip bits of 1 and add 1:
$0\ 0000001 \rightarrow 1111\ 1110+1 = 1111\ 1111$

(c) How would you represent 17 and -17?

1. Unsigned? 0b0001 0001, N/A $17=16+1=2^0+2^4$
2. Biased? 0b1001 0000, 0b0110 1110 $17+(127)=144$ | $-17+127=110$
3. Two's Complement? 0b0001 0001, 0b1110 1111

like unsigned ↗ $-17=-2^7+2^6+2^5+2^3+2^2+2^1+2^0$ or 17: 0001 0001
17: $11101110+1=11101111$

(d) What is the largest integer that can be represented by *any* encoding scheme that only uses 8 bits?

There is no such integer. For example, an arbitrary 8-bit mapping could choose to represent the numbers from 1 to 256 instead of 0 to 255.

(e) Prove that the two's complement inversion trick is valid (i.e. that x and $\overline{x}+1$ sum to 0).

Note that for any x we have $x+\overline{x} = 0\text{b}1\ldots1$. Adding 0b1 to 0b1...1 will cause the value to overflow, meaning that $0\text{b}1\ldots1 + 0\text{b}1 = 0\text{b}0 = 0$. Therefore, $x+\overline{x}+1 = 0$

Ex: -17: $x = 1110\ 1111$ $\overline{x} = 0001\ 0000$

$x+\overline{x} = 1111\ 1111$ & $x+\overline{x}+1 =$

```
 1111 1111
 0000 0001
overflow 0000 0000
```
← all zeros.

A straightforward hand calculation shows that $0b1\ldots1 + 0b1 = 0$.

(f) Explain where each of the three radices shines and why it is preferred over other bases in a given context.

Decimal is the preferred radix for human hand calculations, likely related to the fact that humans have 10 fingers.

Binary numerals are particularly useful for computers. Binary signals are less likely to be garbled than higher radix signals, as there is more "distance" (voltage or current) between valid signals. Additionally, binary signals are quite convenient to design circuits, as we'll see later in the course.

Hexadecimal numbers are a convenient shorthand for displaying binary numbers, owing to the fact that one hex digit corresponds exactly to four binary digits.

Reet
10min

3 Arithmetic and Counting

overflow examples: + & + = − ; − & − = +

cannot overflow if you do a + plus − in twos complement!

Overflow can be seen if the carry in ≠ carry out

3.1 Addition and subtraction of binary/hex numbers can be done in a similar fashion as with decimal digits by working right to left and carrying over extra digits to the next place. However, sometimes this may result in an overflow if the number of bits can no longer represent the true sum. Overflow occurs if and only if two numbers with the same sign are added and the result has the opposite sign.

$$\begin{array}{r} 011001 \\ -\ 000111 \\ \hline 010010 \end{array} = 2^4 + 2^1 = 16 + 2 = 18$$

(a) Compute the decimal result of the following arithmetic expressions involving 6-bit Two's Complement numbers as they would be calculated on a computer. Do any of these result in an overflow? Are all these operations possible?

1. 0b011001 − 0b000111

flip bits and add 1! so 0b011001 + 0b111001

same so no overflow

$$\begin{array}{r} 011001 \\ +\ 111001 \\ \hline 1010010 \end{array}$$

← truncate to 8 bits so 0b10010

0b010010 = 18, No overflow.

differ so overflow

0b100011 = −29
+ 0b111010 = −6
0b011101 = 29! overflow!

2. 0b100011 + 0b111010

Adding together we get 0b1011101, however since we are working with 6-bit numbers we truncate the first digit to get 0b011101 = 29. Since we added two negative numbers and ended up with a positive number, this results in an overflow.

no overflow

$$\begin{array}{r} 0x3B = 111011 \\ +\ 0x06 = 000110 \\ \hline 1\,000001 \end{array}$$

drop this bit as we can only use 6 bits to rep numbers!

= 0b000001 = 1

3. 0x3B + 0x06

Converting to binary, we get 0b111011 + 0b000110 = (after truncating) 0b000001 = 1. Despite the extra truncated bit, this is not an overflow as -5 + 6 indeed equals 1!

4. 0xFF − 0xAA

0xFF = 0b11111111 & 0xAA = 10011001 both 8 bits!

Trick question! This is not possible, as these hex numbers would need 8 bits to represent and we are working with 6 bit numbers.

7min

(b) What is the least number of bits needed to represent the following ranges using any number representation scheme.

so need to find the smallest power of 2 (since we are looking for number of bits needed) which can have more than n representable values.

1. 0 to 256

$1 \to 256 = 256 + \underset{0}{1} > 257$ values

$256 = 2^8$ but we need 257 values so we need one more bit so 9 bits!

In general n bits can be used to represent at most 2^n distinct things. As such 8 bits can represent $2^8 = 256$ numbers. However, this range actually contains 257 numbers so we need 9 bits.

2. -7 to 56

$[1 \to 56] = 56 + \overbrace{[-7, 0]}^{8} = 56 + 8 = 64$ power of 2!

Range of 64 numbers which can be represented through 6 bits as $2^6 = 64$

so 2^6 so 6 bits

3. 64 to 127 and -64 to -127

$[64, 127]$ need 64 bits as do $[-127, -64]$ so

$64 \cdot 2$

$2^6 \cdot 2^1 = 2^7$ so 7 bits

We are representing 128 numbers in total which requires 7 bits.

4. Address every byte of a 12 TiB chunk of memory

Since a TiB is 2^{40} and the factor of 12 needs 4 bits, in total we can represent using 44 bits as 2^{43} bytes $<$ 12 TiB $<$ 2^{44} bytes

$TiB = 2^{40}$ $\quad 12 < 16$

2^4

so need 44 bits!

1a6) ex: Convert from 10 → 2. Division!

437:

$$2\overline{)437} = 218,\ 437 - 436 = 1 \quad 2\overline{)218} = 109,\ 218 - 218 = 0 \quad 2\overline{)109} = 054,\ 109 - 108 = 1 \quad 2\overline{)54} = 27,\ 54 - 54 = 0$$

$$2\overline{)27} = 13,\ 27 - 26 = 1 \quad 2\overline{)13} = 6,\ 13 - 12 = 1 \quad 2\overline{)6} = 3,\ 6 - 6 = 0 \quad 2\overline{)3} = 1,\ 3 - 2 = 1 \quad 2\overline{)1} = 0,\ 1 - 0 = 1$$

LSB ← to → MSB

Finally get a zero.

We use floor division to get the remainder. Read from
This is general idea from going from higher radix to lower radix.

So

$$0b\ \underbrace{1}\underbrace{1011}\underbrace{0101}$$

Hex → Binary: Group into 4 ones & convert directly.
&
Binary → Hex

A = 10 D = 13
B = 11 E = 14
C = 12 F = 15

1010 = 8+2 = 10 = A
B = 11 = 1011

Alternative for 437:

512 − 437 = 75

$$\downarrow \qquad\qquad \downarrow$$

$$\underbrace{1}\underbrace{0000}\underbrace{0000} \qquad \underbrace{100}\underbrace{1011}$$

$$\begin{array}{r} 1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ -\ 0\,0\,0\,1\,0\,0\,1\,0\,1\,1 \\ \hline 0\,1\,1\,0\,1\,1\,0\,1\,0\,1 \end{array}$$