Unsigned integer representation

- With n bits, max value that can be represented: $2^n - 1$

Binary to Decimal conversion

1	1	0	1	0	0
2^5	2^4	2^3	2^2	2^1	2^0

32 + 16 + 4

=52

6 bits, so max number possible is 2^6-1 = 63

Decimal to Binary (unsigned)

1. Find number of bits required
   $floor(\log_2 number) + 1$
2. For each bit-position, starting from highest, Repeatedly check if number greater or equal to $2^{\text{bit-position}}$,and set bit to 0 or 1 accordingly

Number of bits required (unsigned)

- **52:** $\log_2(52) = 5.7; floor(5.7) = 5; \# \text{ bits} = 6$
  - Check 1: $2^6 - 1 \geq 52; 63 \geq 52$ (YES)
  - Check 2: $2^5 - 1 < 52; 32 < 52$ (YES)
- **102:** $\log_2(102) = 6.67; floor(6.67) = 6; \# \text{ bits} = 7$
  - Check 1: $2^7 - 1 \geq 102; 127 \geq 107$ (YES)
  - Check 2: $2^6 - 1 < 102; 63 < 107$ (YES)
- **276:** $\log_2(276) = 8.10; floor(8.10) = 8; \# \text{ bits} = 9$
  - Check 1: $2^9 - 1 \geq 276; 511 \geq 276$ (YES)
  - Check 2: $2^8 - 1 < 276; 255 < 276$ (YES)

Decimal to Binary (unsigned)

52; # bits = 6

Bit positions start at 0, so 6 bits means 2^0 to 2^5−1

Bit position	Power of 2	Number	>=	Remainder	Bit value
5	32	52	Yes	20	1
4	16	20	Yes	4	1
3	8	4	No	4	0
2	4	4	Yes	0	1
1	2	0	No	0	0
0	1	0	No	0	0

2's complement representation

- Allows representing negative numbers
- **Arithmetic operations can be done by operating on individual bits**
- 0 is always all bits 0
- Most negative number: -2^{n-1}
- Most positive number: $+2^{n-1} - 1$

2's complement range

1 bit	This is weird: -1 to 0		
2 bits	-2	To	+1
3 bits	-4	To	+3
4 bits	-8	To	7
5 bits	-16	To	15
6 bits	-32	To	31
7 bits	-64	To	63
8 bits	-128	To	127
9 bits	-256	To	255
10 bits	-512	To	511

2's complement representation

3 bits	
100	-4
101	-3
110	-2
111	-1
000	0
001	+1
010	+2
011	+3

4 bits	
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1
0000	0
0001	+1
0010	+2
0011	+3
0100	+4
0101	+5
0110	+6
0111	+7

5 bits	
10000	-16
10001	-15
10010	-14
10011	-13
10100	-12
10101	-11
10110	-10
10111	-9
11000	-8
11001	-7
11010	-6
11011	-5
11100	-4
11101	-3
11110	-2
11111	-1
00000	0
00001	+1
00010	+2
00011	+3
00100	+4
00101	+5
00110	+6
00111	+7
01000	+8
01001	+9
01010	+10
01011	+11
01100	+12
01101	+13
01110	+14
01111	+15

6 bits			
100000	-32	000000	0
100001	-31	000001	+1
100010	-30	000010	+2
100011	-29	000011	+3
100100	-28	000100	+4
100101	-27	000101	+5
100110	-26	000110	+6
100111	-25	000111	+7
101000	-24	001000	+8
101001	-23	001001	+9
101010	-22	001010	+10
101011	-21	001011	+11
101100	-20	001100	+12
101101	-19	001101	+13
101110	-18	001110	+14
101111	-17	001111	+15
110000	-16	010000	+16
110001	-15	010001	+17
110010	-14	010010	+18
110011	-13	010011	+19
110100	-12	010100	+20
110101	-11	010101	+21
110110	-10	010110	+22
110111	-9	010111	+23
111000	-8	011000	+24
111001	-7	011001	+25
111010	-6	011010	+26
111011	-5	011011	+27
111100	-4	011100	+28
111101	-3	011101	+29
111110	-2	011110	+30
111111	-1	011111	+31

Hexadecimal Table

0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

Decimal to Binary (2's comp)

- First get number of bits
  $Floor(\log_2 abs(number)) + 2$
- If positive number, then use process we developed before and you are done
- If negative number,
  - First get representation of the absolute value
  - Then invert all bits
  - Then add +1 to the inverted bits

Decimal to Binary 2's complement

- -52

1. # bits = 7
2. Negative number
   a) Representation of +52 = 0110100
   b) Invert all bits: 1001011
   c) Add +1: +0000001
   =1001100

All 7 bits. Note that the MSB will always be zero in this intermediate step

All 7 bits. Note that the MSB will always be ONE for negative numbers at the very end

2's complement binary to decimal

- If MSB is 0, same as unsigned
- If MSB is 1, reverse steps:
  a) Invert all bits
  b) Add +1
  c) Now determine magnitude
     Remember it is a negative number

2's Complement Binary to decimal

- 1001100
- MSB is 1
  a) Invert all bits: 0110011
  b) Add +1: +0000001
  0110100
  $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

= 32+16+4 = 52

c) -52

d) Go back to 2's complement range and check

2's complement arithmetic
It's bitwise addition!

- 52 + (-101) = -49

```
  00110100
 +10011011
 ---------
  11001111
```

Extension rule: 2s complement

1001100	7 bits
11001100	8 bits
111001100	9 bits

- To take a number represented in X bits can get its representation in Y bits, (Y > X), copy the **MSB** into the "new" bit positions

Fixed point

- After the decimal point negative powers of 2
- 0.001

1	.	1	0	1	0
2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}

1.625

Conversion from binary to decimal

- 0.43

Power of 2	Weight	Number	number >= weight	Remainder	Bit value
2	0.5	0.43	No	0.43	0
4	0.25	0.43	Yes	0.18	1
8	0.125	0.18	Yes	0.055	1
16	0.0625	0.055	No	0.055	0
32	0.03125	0.055	Yes	0.02375	1
64	0.015625	0.02375	Yes	0.008125	1
128	0.0078125	0.008125	Yes	0.0003125	1
Represented value	**0.4296875**				

Floating Point Standard
IEEE-754 Standard
Single-Precision Representation

1 bit	8 bits	23 bits
S	Exponent	Fraction

- $N = -1^S * 1.fraction * 2^{exponent-127}$
  $when\ 1 \leq exponent \leq 254$
- $N = -1^S * 0.fraction * 2^{-126}$
  $when\ exponent == 0$

11000000110010001000000000000000

1 10000001 10010000000000000000000

S = 1 ; therefore negative number

exponent = 129

fraction = 10010000000000000000000

$N = -1^1 * 1.1001 * 2^{129-127}$

$N = -1^1 * 1.1001 * 2^2$

$N = -1^1 * 110.01$

$N = 6.25$

743.5

0010 1110 0111.10

= 1.01110 0111.10 * 2^9

1) $fraction = 011100111$

2) Exponent-127=9⇒Exponent = 136

10001000

3) Final representation

0 10001000 0111 0011 1000 0000 0000 000