Binary eg.

- Easy addition / math
- Whole numbers only
- Large numbers can be "broken up" and put through ALU

BCD

- Encode Decimal Digit w/ 4 binary ones
- Reduced Range 4 bits in Base 2 = Range 16, 4 bits BCD = Range 10
- When performing math, results larger than 9 must broken up and carried
  - 39 × 6

9×6=54

```
              0011 (3) 1001 (9)
                       0110 (6)
         ________________________
                       0100
   0001   0101
   0001   1000           X
  ______________________________
   ~~0001   1101~~       0100
   0010   0011             4
      2      3
```

- Packed Decimal Format:
  - Sign 1100 (+), 1101 (-) 1111 (unsigned)
    - Always last nibble (marks number termination)
  - Decimal Precision declared by program.

Sign and Magnitude (Signed Binary)

- high order bit = sign (1 -) (0 +)
- 2 values for Ø (+0, -0)
- Range moves 1/2 into negative space
- 8 bit signed binary
  - Unsigned Range = 256
  - Siged Range = +127 ⟶ -127
  - Unsigned 8 bit: 10101010 = 170_{10}
  - Signed 8bit : 10101010 = -42_{10}
  - First bit matters so # of bits matter! must be known.
- Unintelligable for math
  - 8 + -4 = 4

```
  0000 1000 (8)
  1000 0100 (-4)
  _____________
  1000 1100 (-12)?
```

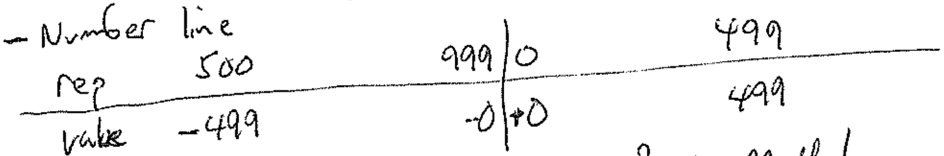
Complementary Representation

- Sign and Magnitude in such a way the computer can perform math!
- 2 methods Radix, Diminished Radix
  - Radix → Value used is the Base Number
  - Diminished Radix → Value is Base Number - 1

Diminished Radix

- 9's Comp (Decimal)
- 1's Comp (Binary)

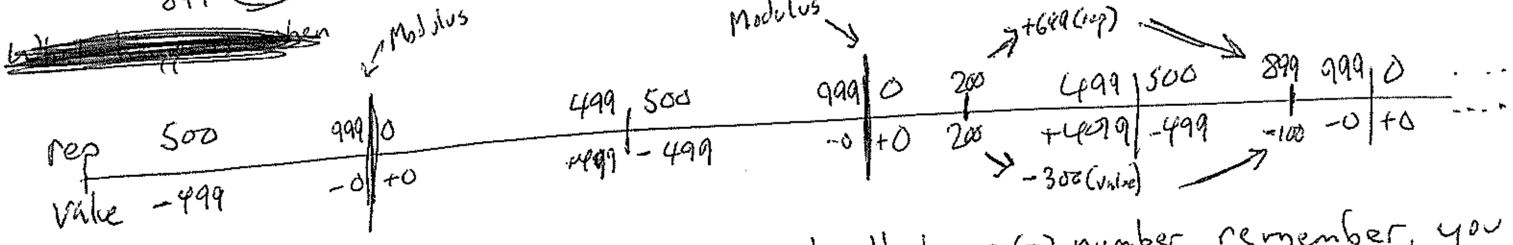
- In all (Radix, Diminished Radix) Sign is not seperated seperately... how?
  - We can assign negative values by subtracting a negative representation from the ~~largest~~ numeral in the radix.
  - 9's Comp Basis → Range of total # of digits − 1
    - 3 digit → 1000 (range) − 1 = 999 ← * we will use this.
    - 5 digit → 100000 − 1 = 99999
    - Range is split in 1/2 for positive/negative
      - First digit 0-4 → + (0-499)
      - First digit 5-9 → − (-499-0)
    - Conversion only happens if negative! positive is untouched!
      - 419 remains 419
      - 519 = 999 − 519 = 480 = -480 (this is called taking the complement) (make sure to add -)
    - Going from - value to representation is the same thing!
      - Since −(−value) = value!
      - So. -480 999 − 480 = 519 = rep 519
    - Number line



- So this representation, why do it? → Math!

649 (-350) + 250 ↳ 649 + 250 = 899 → 999 − 899 = -100 ✓

and in (+) 170 + 250 = 420 (still works!)



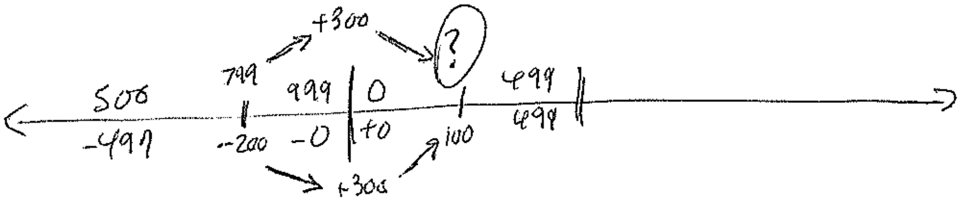
Wraparound: If we start in (+) and add larger (−) number, remember, you are adding the representation, not the value, you go right on the line!

200 + (-300) = ?

999 − 300 = 699 → 200 + 699 = 899 → 999 − 899 = -100

End Around Carry

- But what about crossing the modulus?
  - Thoose 2 Ø's will mess with us!



(-200) + 300 = <u>100</u>

$$\begin{array}{r} 999 \\ -200 \\ \hline 799 \end{array} \nearrow \begin{array}{r} 799 \\ +300 \\ \hline 1099 \end{array}$$

1099 ← Looks like we broke it!
- Double Ø can be countered by adding 1!
- The extra digit means we crossed the modulus
- The extra digit means we cross the modulus we cross 2 Ø's!
- Every time we cross the modulus use it to add 1!
- Everytime we cross

$$\begin{array}{r} 1099 \\ \hookrightarrow 1 \\ \hline 100 \end{array}$$ ✓ ⟸ This is end around carry.

Overflow

- We have explored going from positive space to negative and negative to pos.
- What if? 350 + 300 = 650 $\hookrightarrow \begin{array}{r} 999 \\ -650 \\ \hline 349 \end{array}$ → <u>Garbage!</u> <u>overflow!</u>

- How do you know?
  - Start with 2 of the same sign
  - end with the opposite!

1's Binary (Diminished Radix)

Still Range of total digits − 1 = basis value

- 4 bit range = 10000 (16) − 1 = 1111 (15)
- 8 bit range = 100000000 (256) − 1 = 11111111 (255)

4 bit Example → 0101 = 5

1101 = $\begin{array}{r} 1111 \\ 1101 \\ \hline 0010 \end{array}$ = -2 → But... Look at the and value..

1101 → rep
0010 → value notice anything?

<u>Flip it and forget it!</u>

Now 8 bit

$\overset{32}{0}0\overset{8}{1}0|1\overset{2}{0}10$ = 42

how would we get -42's rep?

$$\begin{array}{r} 11111111 \\ -00101010 \\ \hline 11010101 \end{array} = -42$$

- Flipping is just as easy (even easier) for the computer!

1's Comp math

1) 45 + 58 = 103

$$\begin{array}{rl} 00101101 & (45) \\ 00111010 & (58) \\ \hline 01100111 & (103) \end{array}$$

everything normal

2) 45 + (-58) = -13

first 58 = 00111010 → rep: 11000101 (this is what would be stored)

$$\begin{array}{rl} \hookrightarrow 00101101 & (45) \\ +\ 11000101 & \\ \hline 11110010 & \rightarrow \text{rep.} \end{array}$$

flip

00001101 (-13)

(wraparound)

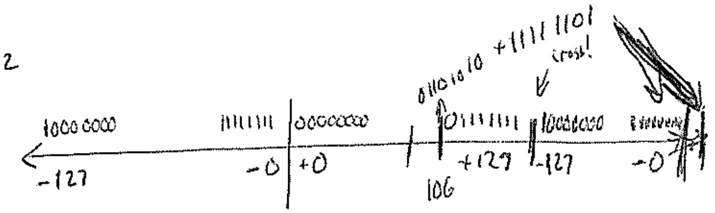
1's end around carry

106 + (-2) = 104

01101010 = 106 , 2 = 00000010
11111101 ← rep -2

$$\begin{array}{r} 01101010 \\ +\ 11111101 \\ \hline 101100111 \\ \xrightarrow{\quad} 1 \\ \hline 01101000 \end{array}$$

= 104 ✓



→ This means that subtraction can be handled by flipping the value and adding!

Overflow

64 + 65 = 129 (+) → 0100 0000 (64)

(+) → 0100 0001 (65)

(-) → 1000 0~~0~~01

overflow! throw it away!

Radix

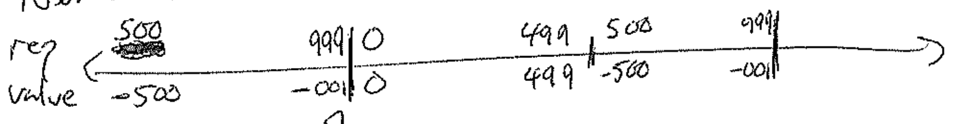
Almost everything is the same! But Basis is the Range

3 digit Decimal → 1000 (10's Comp)

8 digit Binary → 10000 0000 (2's Comp)

10's Complement

Numberline



No double 0!

Conversion:

247 = 247! (positive)

```
          9 9 10
          X 0 0 0
503 rep = - 5 0 3     -497
            4 9 7

            9 9
           X 0 0 0
-17 value =    - 1 7
             9 8 3 rep
```

Is subtracting from 1000 annoying?
 - you can take 1 from the radix and add it back later!

```
-17 =  999
       -17
       982
        +1
       983
```

Why?

End Around carry (no double 0's)

-200 + 320 = 120

```
 999
-200
 799+1=800
      +320
      X120
  Ignore!
```

Easiest for the ALU (ignores the overflow)
 - But only if signs were different!
 otherwise what is it? (Overflow)

2's Comp

Flip it and +1!

```
42 → 00101010
      ↳ Flip → 11010101
                     +1
             11010110 → -42
```

2's Example

```
32 + (-42)                 32 16 8 4 2 1
                           00101010
32 → 0010 0000             11010101
-42 → 1101 0111                  +1
      ---------            --------
      1111 0111            11010110
      0000 1000            00100000
            +1             --------
          1000             11110110
                           00001001
                                 +1
                           --------
                           00001010 → 10 ✓
```

2's End Around

```
106 + (-8) = 98
                            -8 = 00001000
106 → 0110 1010                  11110111
      1111 1000                        +1
      ---------                  --------
    1 0110 0010                  11111000
      64 32 16 8 4 2 1
       96       → 98  ✓
```

Exponential Notation (Scientific Notation)

4 components
- Sign
- magnitude or mantissa
- Sign of exponent
- Magnitud of exponent

$+.12345 \times 10^{-4}$

Sign

Mantissa

Sign of exp.

Magnitude of exp.

Base of exponent (assumed)

Location of radix (assumed)

Assumed
- Base (Decimal)
- Decimal (Radix) Point Location.

Floating Point Format (8 digit)

SEEMMMMM

Sign of mantissa — magnitude of exponent — mantissa

S = 0(+) or 5 (-)

What is missing? (~~magnitude~~ Sign of exponent)

- Excess-N notation → Pick middle value as offset where N is the middle value

```
rep        0     49|50    99
exp. value -50    -1|0     49
```

So...
$10^0 \to 50$
$10^3 \to 53$
$10^{-3} \to 47$

- Floating Point Addition + sub
  - Exponent and Mantissa done separately
  - Exponents must be the same (or made so)
    - Remember to keep decimal points aligned
    - Percision must be maintained (least significant digits may be lost)

$05199520 = +.99520 \times 10^{1}$

$04967850 = +.67850 \times 10^{-1}$

Bring lower exponent up!

$.0067850 \times 10^{1} \leftarrow +2$

now add matissas

$$\begin{array}{r} .99520 \\ .006785 \\ \hline 1.001985 \end{array} \times 10^{1}$$

Note decimal not all the way left!

Fixing is called "Normalizing the mantissa"

round

$.10019(85) \times 10^{2} \leftarrow$

$.10020 \times 10^{2}$

Multiplication and Division

- Mantissas → mult or divided!
- Exponents → added or subtracted
- Normalization → required to restore location of decimal point and maintain percision of result.
- Adjustment of Excess-N by (-50) because both were added with it!
  - Ex $53 + 52 = 105\ (-50) = 55$ ✓

Ex. $05220000 \rightarrow +.200000 \times 10^{2}$

$04712500 \rightarrow +.471250 \times 10^{-3}$

Add exponents: $52 + 47 = 99 - 50 = 49$

multiply mantissas → $$\begin{array}{r} .20000 \\ \times\ .12500 \\ \hline .02500 \end{array}$$ → Normalize ↪ $.25000 \times 10^{-2}$

Back to SEEMMMMM → 04825000 ✓

Pull up slides 62 + 63 for IEEE 754 standard