המרכז
לחינוך סייבר
CYBER EDUCATION CENTER

Assembly for Reverse Engineering

Bases and Logical Functions

```
call   2084
jmp    15CA
call   17B4
mov    si,7160
xor    di,di
mov    es,[7606]
mov    bx,000F
xor    cx,cx
mov    bp,0001
mov    dx,ds
```

Barak Gonen

ארגון המחשב ושפת הסף

ברק גונן

מטה הסייבר הצה"לי

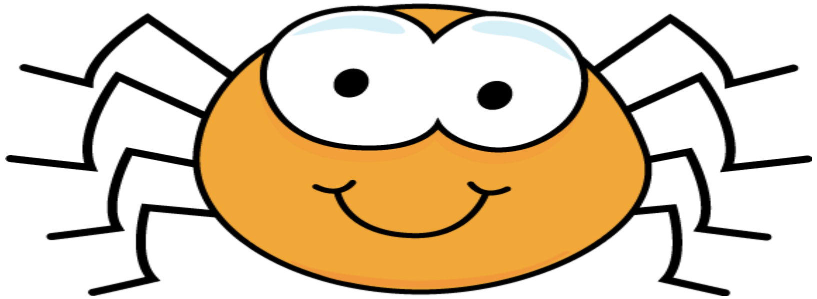
המרכז לחינוך סייבר
CYBER EDUCATION CENTER

Assembly Book

- https://data.cyber.org.il/assembly/assembly_book.pdf

Bases

- Base 10:
  - Digits: 0,1,2,3,4,5,6,7,8,9
  - Representing 10 requires 2 digits
- Base N:
  - Digits: 0,…, N-1
  - Representing N requires 2 digits
- How many legs do I have in base 3?



Try it:

Write down the first 22 numbers in base 7

(0, 1, …)

Bases Conversion

- Convert 199 in base 5-

Operation	Reminder
199:5= 39	4
39:5= 7	4
7:5= 1	2
1:5= 0	1

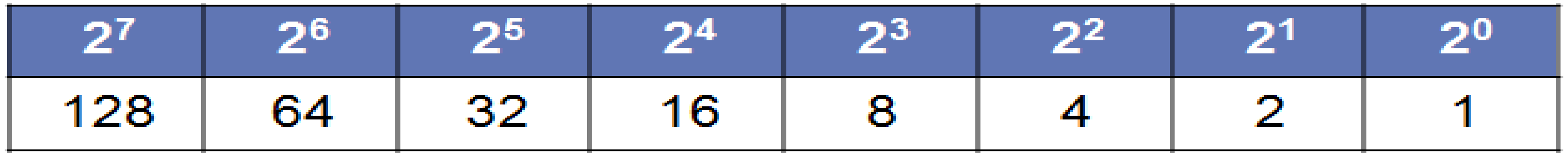
Try it:

Convert 300 to base 4

Base 2

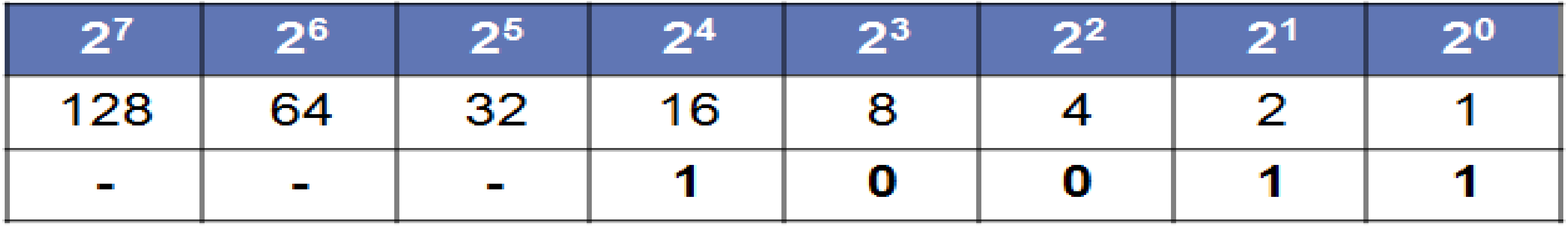
- Only 0, 1

- The value of each digit is according to it’s position:



2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

- For example = $16+2+1=19_{10}$ 10011_2

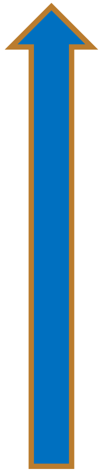


2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
-	-	-	1	0	0	1	1

Decimal to Binary

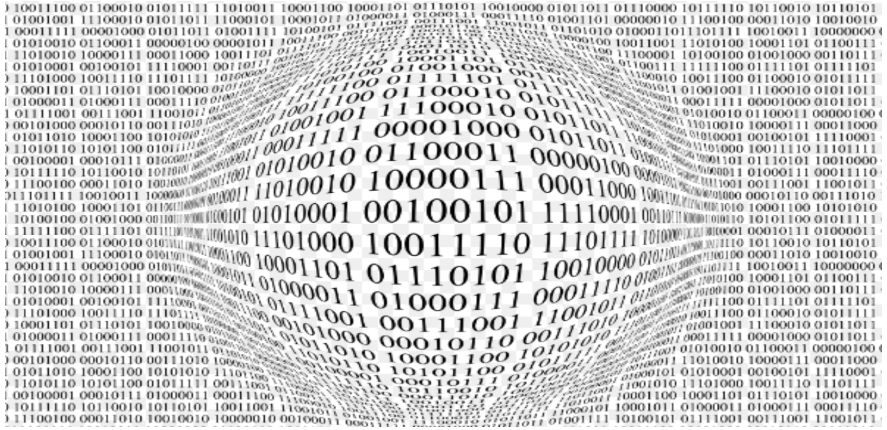
- Let’s reverse 19_{10} back to base 2:

Operation	Reminder
19:2= 9	1
9:2= 4	1
4:2= 2	0
2:2= 1	0
1:2= 0	1



Try it:

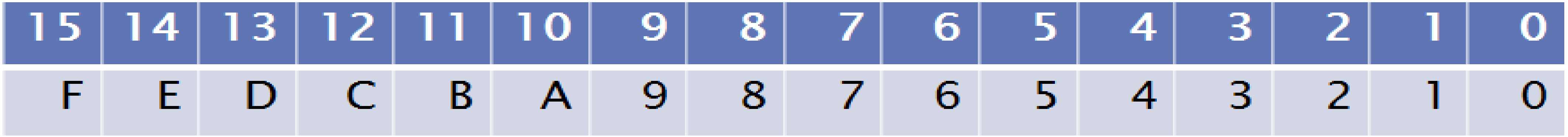
Convert 10011110 to base 10



Try it:

Convert 199 to base 2

Base 16 - Hexdecimal

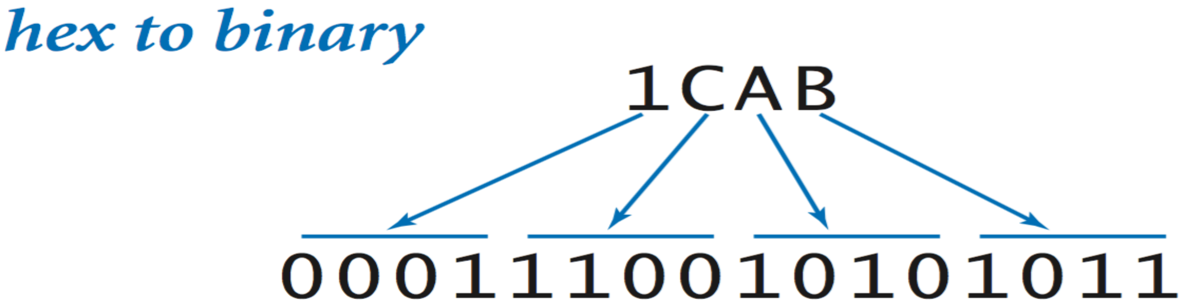


15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

- Some valid numbers, sometimes used as magic numbers (initialize memory, default passwords):
  - 0x4B1D
  - 0xC0DE
  - 0xC0FFEE
  - 0xBADF00D
  - 0xDEADBEAF

Hex 2 Bin

- Easy! Simply convert each Hex digit to 4 Binaries
- 4 Binaries – “Nibble”



hex to binary

1CAB

0001 1100 1010 1011

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

Try it:

- Search "Flippy bit"
- Reach score 15

Multiplying by Base

- In base 10, how do we multiply a number by 10?
- In base 2, multiply by 2 = add zero
- In base 16, multiply by 16 = add zero

- 101011 x 102 = 1010110
- 0xABCD x 0x10 = 0xABCD0

Storing numbers in memory

- Computers use fixed number of bits to store numbers
- Otherwise, how can we tell if 1111 is 15 or 3,3?
- Next slides shall use byte (8 bits) size numbers

2’s Complement

- Problem: represent negative numbers
- Method: negate and add 1
- Example: 6 is 0000 0110. How about -6?

```
   11111001
 +        1
 ----------
   11111010
```

- Advantage: sum is always zero

```
    00000110
 +  11111010
    --------
 (1)00000000
```

Signed Binary to Decimal

- If the number is positive (left bit == 0):
  - “Normal” conversion
  - Each bit has the value of it’s index^2
- If the number is negative (left bit == 1):
  - Find the 2’s complement (a positive number)
  - Convert like a positive number
  - Change sign to minus

Try it:

Convert -12 to base 2

Try it:

Convert to base 10:

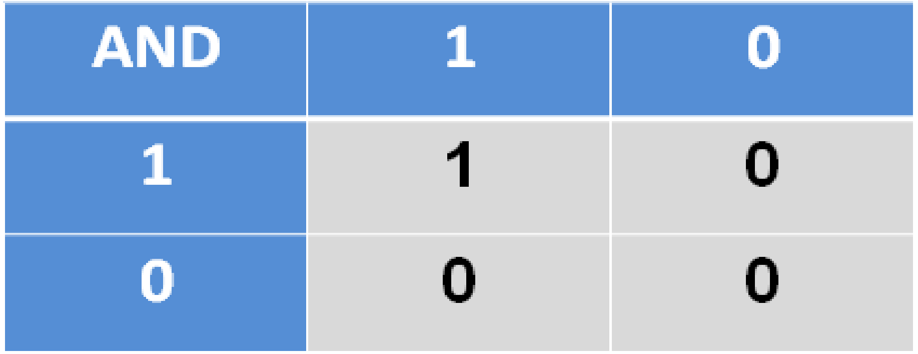
11001000 as unsigned

11001000 as signed

Logical Functions

- AND
- OR
- XOR
- NOT

AND



AND	1	0
1	1	0
0	0	0

0000 0111 and
1001 0110

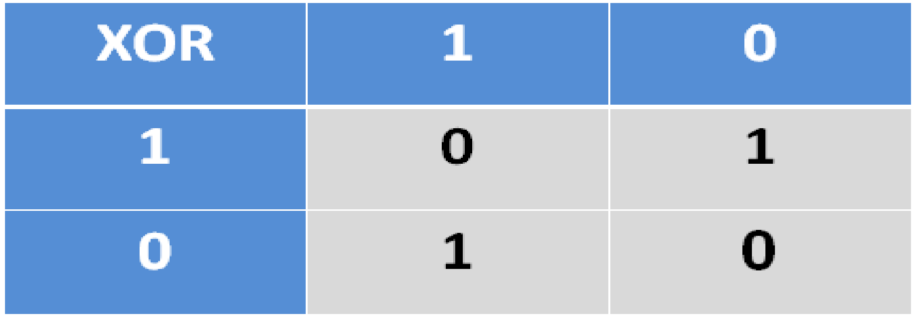
0000 0110

AND

- Using AND, how can we-
  - Check if a number is even?
  - Check if a number divides by 4?
  - A signed number is negative?

- ‘Mask’ – A set of bits used for isolating and operating on certain bits

XOR



XOR	1	0
1	0	1
0	1	0

- XOR is equal to ADD modulo 2
- A set of bits XOR the same set of bits == 0
- Useful in encryption

XOR - Encryption

Message : 1001 0011

Key : 0101 0100

xor : 1001 0011
0101 0100

Encrypted :1100 0111

xor : 1100 0111
0101 0100

Decrypted: 1001 0011