Veri Temsili (Data Representation)

- Veri Temsili
  - birden fazla sekilde olabilir.
  - veriler uzerinde yapilan islemlerin sonucuna etkisi yoktur.
  - veriler uzerindeki islemlerin zorluk/kolaylik derecesine etki eder.
- Verilerin temsil edilme sekilleri bilgisayarin dizaynina etki eder.

Infinite Setlerin Finite Setlere donusumu

- Eger bir sembol M degisik deger alirsa, N tane sembolden olusan tum verilerin sayisi M^N dir.
  - Ornek: 3-digitli decimal verilerin toplam sayisi $10^3=1000$ ([0,999])
- Iki farkli durumu birbirinden ayirt eden devrelerin tasarimi son derece basit
  - Bilgisayarlar iki farkli durumu olabilen binary digitlerden (bit) ler kullanilarak design edilmektedir
  - N bit 2^N farkli sayi temsil eder.
- Farkli turdeki sayilar icin genelde farkli temsil sekilleri vardir. (ornek: floating-point, integer)

Binary (ikili) Gosterim

- $s_{n-1}s_{n-2}s_{n-3} \ldots\ldots s_2s_1s_0 = \Sigma_{i=0..n-1}s_i2^i$
- $100101 = 2^0 + 2^2 + 2^5 = 37_{10}$
- Most Significat Bit (MSB)
- Least Significant Bit (LSB)
- Big-Endian
- Little-Endian
- Cogu bilgisayarlar gunumuzde 32bit veya 64 bitlik verilerle calisirlar

.byte 0, 1, 2,3

Byte #			
0	1	2	3

Big Endian

Byte #			
3	2	1	0

Little Endian

1025 00000000 00000000 00000100 00000001

Address	Big Endian	Little Endian
00	00000000	00000001
01	00000000	00000100
02	00000100	00000000
03	00000001	00000000

Integer Sayilar

- Unsigned integer
  - $b_nb_{n-1}b_{n-2} \ldots\ldots b_2b_1b_0 = \Sigma_{i=0..n}b_i2^i$
- Sign magnitude
  - Bir extra bit kullanilmak suretiyle positive sayilar negative sayilardan ayirt edilir (Sign bit).
  - Geri kalan bitler magnitude bitler olarak adlandirilirlar
  - (n+1) bit kullanilarak temsil edilen tamsayilarin araligi $[+(2^n-1) \;\; -(2^n-1)]$.

sign magnitude

X	XXX XXXX

8-bit sign magnitude

Bit Pattern	Decimal Deger
0000 0011	3
1000 0111	-7
1111 1111	-127
0000 0000	0
1000 0000	0

$b_i(-1)^{bn} \sum_{i=0..n-1} b_i 2^i$

Eger MSB 0 ise sayi positive, MSB 1 ise sayi negativedir.

Avantajlari: kolayca anlasilabilmesi

Dezavantajlari: 0 iki farkli sekilde temsil edilir. Toplama ve cikarma islemlerinde kolay degil.

Integerlarin Complement olarak Temsili

- Toplama ve cikarma islemlerini kolaylastirir.
- Positive sayilarin temsili, positive sayilarin sign magnitude olarak temsili ile ayni.
- Negative sayilarin temsili farkli.

One's Complement

$b_n b_{n-1} b_{n-2} \ldots\ldots b_2 b_1 b_0$

$\sum_{i=0\ldots n-1} (b_i . 2^i) - b_n.(2^n-1)$

$b_n.(2^n-1)$: Bias

b_n nin degeri sayinin positive/negative olmasini belirler

Eger b_n 1 ise sayi negative, 0 ise sayi positive dir.

000……000 ve 111……111 nin her ikisi de sifiri gosterir

One's Complement

Ikili Gosterim	Decimal Degeri
0000 0011	3
1111 1100	-3
0001 1111	31
1110 0000	-31
000 0000	0
1111 1111	0
0000 0001	1
1111 1110	-1

(n+1) bit kullanilarak temsil edilen tamsayilarin araligi $[+(2^n-1) \;\; -(2^n-1)]$.

Two's Complement

$b_n b_{n-1} b_{n-2} \ldots\ldots b_2 b_1 b_0$

$\sum_{i=0\ldots n-1} (b_i . 2^i) - b_n.2^n$

$b_n.2^n$: Bias

b_n nin degeri sayinin positive/negative olmasini belirler

Eger b_n 1 ise sayi negative, 0 ise sayi positive dir.

Sifirin tek gosterimi vardir ve 000……000 dir.

Ikili Gosterim	Decimal Degeri
0000 0011	3
1111 1101	-3
1111 1100	-4
0001 1111	31
1110 0001	-31
1110 0000	-32
0000 0000	0
1111 1111	-1

(n+1) bit kullanilarak temsil edilen tamsayilarin araligi $[+(2^n-1) \;\; -2^n]$.

Biased Gosterim

$b_n b_{n-1} \ldots\ldots b_2 b_1 b_0$

$\sum_{i=0..n} (b_i .\ 2^i) - B$ B: bias ($B=2^n$ veya $B=2^n-1$)

Decimal	Biased Gosterim	
	Decimal	Binary
-4	0	000
-3	1	001
-2	2	010
-1	3	011
0	4	100
1	5	101
2	6	110
3	7	111

n=3

Bias: $2^{3-1}=4$

binary	unsigned	sign mag.	two's comp.	one's comp.	biased-127	biased-128
0000 0000	0	0	0	0	-127	-128
0000 0001	1	1	1	1	-126	-127
0000 0010	2	2	2	2	-125	-126
0111 1111	127	127	127	127	0	-1
1000 0000	128	0	-128	-127	1	0
1000 0001	129	-1	-127	-126	2	1
1111 1101	253	-125	-3	-2	126	125
1111 1110	254	-126	-2	-1	127	126
1111 1111	255	-127	-1	0	128	127

$b_n b_{n-1} b_{n-2} \ldots\ldots b_3 b_2 b_1 b_0$

Format	Range	Value
unsigned	$0 <= I <= 2^{n+1}$	$I = \sum_{i=0}^{n} b_i.2^i$
sign magnitude	$-2^n < I < 2^n$	$I = (-1)^{b_n} \sum_{i=0}^{n-1} b_i.2^i$
one's complement	$-2^n < I < 2^n$	$I = \sum_{i=0}^{n-1} b_i.2^i - b_n(2^n - 1)$
two's complement	$-2^n <= I < 2^n$	$I = \sum_{i=0}^{n-1} b_i.2^i - b_n.2^n$
bias-B	$-B <= I < 2^{n+1} - B$	$I = \sum_{i=0}^{n-1} b_i.2^i - B$

Sign Extension

- 8/16 bitlik sayilar 32 bitlik sayi olarak gosterilebilir.
- Original sayinin degerinde degisiklik olmaz
- Carpma islemlerinde bazen gereksinim duyulur.

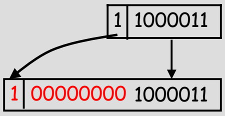
Unsigned Integers

$\boxed{10000011}$

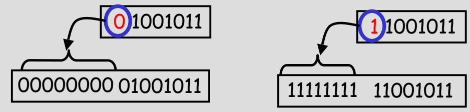
$\downarrow$

$\boxed{\overbrace{00000000}\ 10000011}$

sign magnitude



Complement (one's /two's)



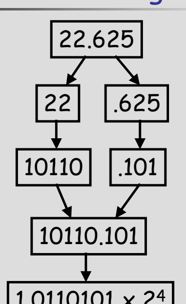
Character

- Alphanumerical
- Gosterim Standardi
  - ASCII (American Standard for Computer Information Exchange)
    - Her bir character 7 bit olarak kodlanir (128 tane)
    - Non-printing karakters (esc)
- Soru: Character ler ASCII standardinda 7 bitle kodlanir. Oysa C dilinde uzunkulkari 1 byte (8) bit. 8-7=1 (1 bite ne oldu?)


```
            .data
ch:         .byte
int:        .word
digit:      .word
            .text

            move    int,0
            get     ch
procch:     bgt     ch, '9', notadigit
            sub     digit, ch, '0'
            bltz    digit, notadigit
            mul     int, int, 10
            add     int, int, digit
            get     ch
            b       procch
notadigit:
```


Floating Point (FP)



.625 *2 = 1.250 1
.250 *2 = 0.5 0
.5 *2 = 1.0 1

IEEE Floating Point Standard (IEEE 17.15)

S (1 bit)	E (8 bit)	F (23 bit)	single precision
S (1 bit)	E (11 bit)	F (52 bit)	double precision

Single Precision (IEEE 17.15)

S (1 bit)	E (8 bit)	F (23 bit)

single precision

$0 \leq E \leq 255 \quad e = E - 127$ (Biased-127)

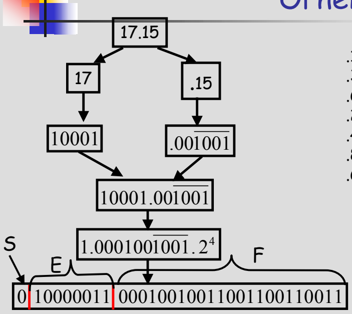
S	E	F	Number
0/1	0	0	0
0	255	0	+∞
1	255	0	-∞
0/1	255	≠ 0	NaN

Single Precision (IEEE 17.15)

$$z_{31} z_{30} z_{29} \ldots\ldots z_0$$

$$I = (-1)^{z_{31}} (1 + 2^{-23} . \sum_{i=0}^{22} z_i 2^i) . 2^{(\sum_{i=0}^{7} z_{i+23} 2^i) - 2^7 - 1}$$

Ornek



.15 *2 = 0.3 0
.3 * 2 = 0.6 0
.6 * 2 = 1.2 1
.2 * 2 = 0.4 0
.4 * 2 = 0.8 0
.8 * 2 = 1.6 1
.6 * 2 = 1.2 1