

Pannon Egyetem

Képfeldolgozás és Neuroszámítógépek Tanszék



Digitális Rendszerek és Számítógép Architektúrák

1. előadás: Számrendszerek,
Nem-numerikus információ ábrázolása

Előadó: Vörösházi Zsolt
Szolgay Péter

Jegyzetek, segédanyagok:

- Könyvfejezetek:

- <http://www.knt.vein.hu>

- Oktatás → Tantárgyak → Digitális Rendszerek és Számítógép Architektúrák (nappali)

- (chapter02.pdf)

- Fóliák, óravázlatok .ppt (.pdf)

- Feltöltésük folyamatosan

Információ ábrázolás:

■ A) Számrendszerek:

☐ I.) Egész típusú:

- előjel nélküli,
- 1-es komplement,
- előjeles 2-es komplement számrendszerek

☐ II.) Fixpontos,

☐ III.) Lebegőpontos (IBM-32, DEC-32, IEEE-32),

☐ Excess kód (exponens kódolására)

■ B) Nem-numerikus információ kódolása

☐ Hibajavítás és detektálás (Hamming kód)



A) Számrendszerek

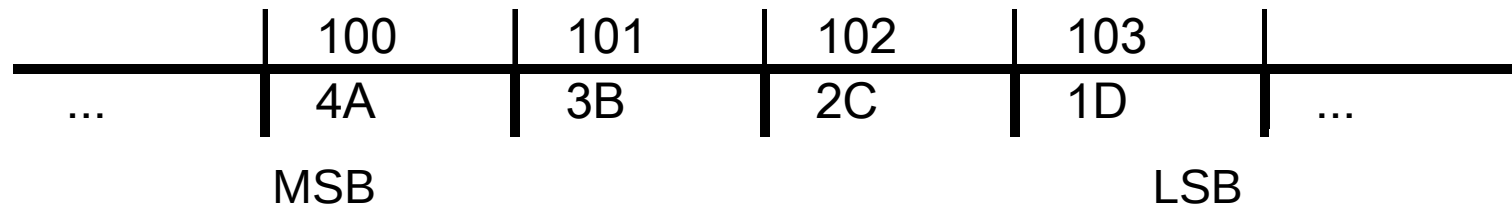
Endianitás (endianness)

- A számítástechnikában, az **endianitás** (byte-sorrend a jó fordítás) az a tulajdonság, ami bizonyos adatok - többnyire kisebb egységek egymást követő sorozata - tárolási és/vagy továbbítási sorrendjéről ad leírást (pl. két protokoll, vagy busz kommunikációja). Ez a tulajdonság döntő fontosságú az integer értékeknek a számítógép memóriájában byte-onként való tárolása (egy memória címhez relatívan), továbbítása esetében
- Byte sorrend megkötés:
 - ☐ Big-Endian formátum
 - ☐ Little-Endian formátum

Háttér: Az eredeti angol kifejezés az *endianness* egy utalás arra a háborúra, amely a két szembenálló csoport között zajlik, akik közül az egyik szerint a lágytojás nagyobb végét (big-endian), míg a másik csoport szerint a lágytojás kisebb végét (little-endian) kell feltörni. Erről Swift ír a *Gulliver Utazásai* című könyvében.

„Nagy a végén” - Big-endian

- A 32 bites egész értéket, (ami legyen „4A 3B 2C 1D”, a 100 címtől kezdve) tároljuk a memóriában, **1 byte-os** elemi tárolókból, 1 byte-onként növekvő címekkel rendelkezik, ekkor a tárolást a következők szerint végzi:

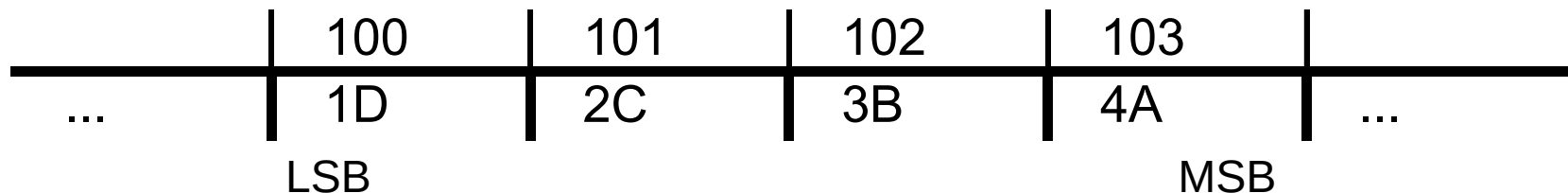


- Ebben az esetben, a „legjellemzőbb” byte - erre általában az ismert angol kifejezést "most significant byte" használják a számítástechnikában (rövidítve *MSB*, ami itt a „4A”) - a memóriában az *legalacsonyabb címen van tárolva*, míg a következő "jellemző byte" (3B) a következő, egyel nagyobb címen van tárolva, és így tovább.
- Bit-reversed format!
- Pl: beágyazott processzorok FPGA-n (MicroBlaze, PowerPC)

„Kicsi a végén” - Little-endian

- Ekkor a 32-bites „4A 3B 2C 1D” értéket a következő módon tárolják
- 100 101 102 103...”1D 2C 3B 4A”...

Így, a kevésbé jellemző ("legkisebb") byte (az angol least significant byte rövidítéséből *LSB* néven ismert) az első, és ez az 1D, tehát a *kis vég kerül előre*:



- Általánosan használt formátum

I.) Egész típusú számrendszer:

- Bináris számrendszer: 1 / 0 (I / H, T / F)
- N biten 2^N lehetséges érték reprezentálható
- Összehasonlító táblázat:

Table 2.1. Number of Representable Values.

<i>Number of Bits</i>	<i>Number of Representable Values</i>	<i>Machines. Uses</i>
4	16	4004, control
8	256	8080, 6800 control, communication
16	65.536	PDP11, 8086, 32020
32	4.29×10^9	IBM 370, 68020, VAX11/780
48	1.41×10^{14}	Unisys
64	1.84×10^{19}	Cray, IEEE (dp)

a.) előjel nélküli egész:

- Unsigned integer: $V_{\text{UNSIGNED INTEGER}} = \sum_{i=0}^{N-1} b_i \times 2^i$
- ahol b_i az i -edik pozícióban lévő '0' vagy '1'
- Reprezentálható értékek határa: 0-tól 2^N-1 -ig
- Helyiértékes rendszer
- Negatív számok ábrázolása nem lehetséges!

- Pl:

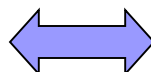
$$101101 \Rightarrow 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ = 45$$

b.) előjeles (kettes komplement) rendszer:

- 2's comp:
$$V_{2'S\ COMPPLEMENT} = -b_{N-1} \times 2^{N-1} + \sum_{i=0}^{N-2} b_i \times 2^i$$
- reprezentálható értékek határa: $-(2^{(N-1)})$ től $2^{(N-1)}-1$ ig
- Ha MSB='1', negatív szám
- Fólia: 2.2 táblázat
- Pl. $101101 \Rightarrow -1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = -19$
- Pl.

$$\begin{array}{r} 10000000 \\ - 01001100 \\ \hline 10110100 \end{array}$$

$$\begin{array}{r} 256 \\ -76 \\ \hline 180 \end{array}$$



$$\begin{array}{r} 01001100 \\ 10110011 \\ + \quad 1 \\ \hline 10110100 \end{array}$$

$$\begin{array}{r} 1's\ kompl. \\ +1 \\ \hline -76 \end{array}$$

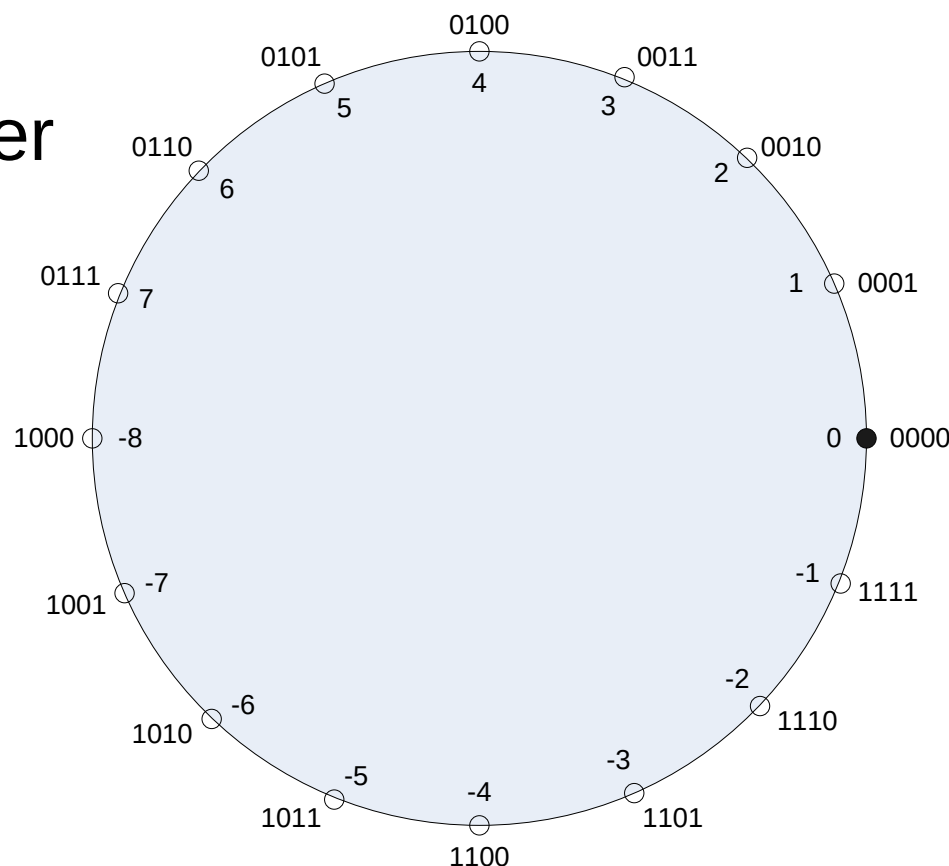
2.2 táblázat:

Table 2.2. 8-Bit Two's Complement Representations.

<i>Bit Pattern</i>	<i>Value</i>	<i>Note</i>
01111111	127	Largest representable value.
01111110	126	
01111101	125	
...	...	
...	...	
00000010	2	Note that leading zero indicates positive number.
00000001	1	
00000000	0	Unique representation of zero .
11111111	-1	Minus one is always all ones.
11111110	-2	Note that leading one indicates negative number.
11111101	-3	
...	...	
...	...	
10000010	-126	
10000001	-127	
10000000	-128	Smallest (most negative) representable value.

PI: Körkörös számláló (circular nature):

- 4 bites 2's komplement rendszer
- Overflow:
0111 $\rightarrow$ 1000
- Underflow:
1000 $\rightarrow$ 0111



c.) 1-es komplement rendszer:

- V értékű, N bites rendszer: $2^N - 1 - V$.
- „0” lesz ott ahol „1”-es volt, „1”-es lesz ott ahol „0” volt (mivel egy szám negatív alakját, bitjeinek kiegészítésével kapjuk meg).
- csupán minden bitjét negálni, (gyors műveletet)
- Értékhatár: $2^{(N-1)} - 1$ től $-(2^{(N-1)} - 1)$ ig terjed,
- Nem helyiértékes rendszer,
- kétféleképpen is lehet ábrázolni a zérust!! (ellenőrzés szükséges)
- end-around carry: amelyben a részeredményhez kell hozzáadni a végrehajtás eredményét

Példa:

Example 2.3: One's complement arithmetic: Consider a 6-bit one's complement system. Represent 15, -15, 13, and -13 in this system. Then perform the following additions: $15 + 13$, $15 + (-13)$, $13 + (-13)$, and $13 + (-15)$.

The numbers are derived in a simple fashion:

Decimal Value	One's Complement	Comment
15	001111	Positive numbers same as two's complement.
-15	110000	Complement bits to negate.
13	001101	Positive numbers same as two's complement.
-13	110010	Complement bits to negate.

Now for the additions:

$\begin{array}{r} 15 \\ + 13 \\ \hline 28 \end{array}$	$\begin{array}{r} 001111 \\ + 001101 \\ \hline 0\ 011100 \end{array}$	This proceeds just like the two's complement version. No carry out: number is correct, and the result is as we expect.
$\begin{array}{r} 15 \\ + -13 \\ \hline 2 \end{array}$	$\begin{array}{r} 001111 \\ + 110010 \\ \hline 1\ 000001 \\ + 000001 \\ \hline 000010 \end{array}$	The addition is done in the normal fashion, but the result of one <u>is incorrect</u>; however, the presence of a carry says we should add that as a 1 in the LSB which gives the expected result.
$\begin{array}{r} 13 \\ + -13 \\ \hline 0 \end{array}$	$\begin{array}{r} 001101 \\ + 110010 \\ \hline 111111 \end{array}$	This time we will add a positive number to its negative (which is just complement) and end up with all ones — <u>a valid zero</u>.
$\begin{array}{r} 13 \\ + -15 \\ \hline -2 \end{array}$	$\begin{array}{r} 001101 \\ + 110000 \\ \hline 0\ 111101 \end{array}$	Here the positive number is smaller than the negative number, so result is negative; no carry — <u>the value is correct</u>.

end-around: cirkuláris carry, körkörös átvitel

II.) Fixpontos számrendszer

■ Műveletek:

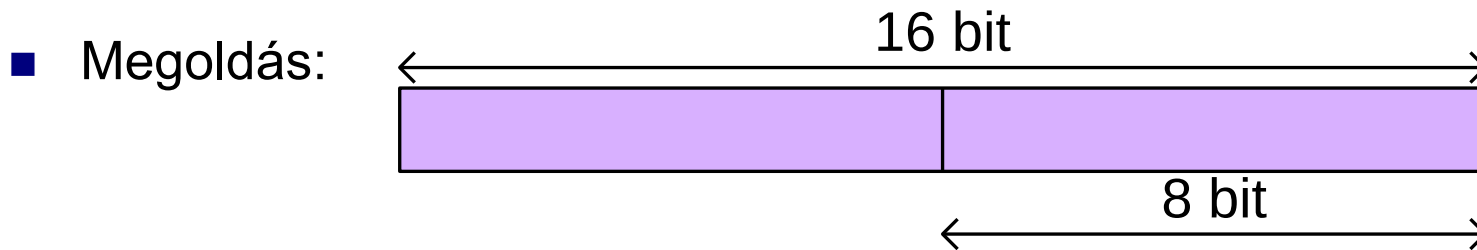
- +, - : ugyanaz, mint az egész szám rdsz. esetén
- *, / : meg kell bizonyosodni arról, hogy a tizedespont helyén maradt-e

$$V_{\text{FIXED POINT}} = -b_{N-1} \times 2^{N-p-1} + \sum_{i=0}^{N-2} b_i \times 2^{i-p}$$

- p: radix (tizedes) pont helye, tizedes jegyek száma
- *differentia*, $\Delta r = 2^{-p}$ (számrendszer finomsága)
 - Ha $p=0 \rightarrow \Delta r=1$, egész rendszer, különben fixpontos
- Alkalmazás: pl. jelfeldolgozás (DSP – Texas Instruments)

Példa: fixpontos rendszer

- Kérdés: Legyen egy 16 bites 2's comp. fixpontos rdsz. ahol $p=8$.
 $V(\text{smallest})=?$, $V(\text{largest})=?$, $\Delta r = ?$ (decimális értékben megadva)



- $V(\text{smallest absolute}) = 00000000.00000001 = 2^{-8} = \underline{0,390625 \cdot 10^{-2}}$
- $V(\text{largest absolute}) = 01111111.11111111 = \underline{\sim 128}$
- $V(\text{largest negative}) = 10000000.00000000 = -2^7 = \underline{-128}$
- Differencia $\Delta r = 2^{-8} = \underline{0,390625 \cdot 10^{-2}}$
- !!DE $V(\text{zero}) = 00000000.00000000$ vagy 11111111.11111111

Excess kód rendszer

- Lebegőpontos számok *kitevőit* (*exponens*-eit) tárolják / kódolják ezzel a módszerrel
- S: a reprezentálni kívánt érték (eredmény), amit tárolnunk
- V: a szám valódi értéke, E: az excess
- $S = V + E$.
- Két számot összeadunk, akkor a következő történik:

$$S1 + S2 = (V1 + E) + (V2 + E) = (V1 + V2) + 2 \times E$$

- pontos eredmény: $[(V1 + V2) + E]$ (ki kell vonnunk E-t!)
- Fólia: 2.4, 2.5, 2.6-os példák

Példa 2.4:

Example 2.4: Number representation in excess codes: What is the representation of $+37_{10}$ in an 8-bit excess 128 code? What is the representation of -23_{10} in an 8-bit excess 128 code? What is the sum of the two numbers, in the 8-bit excess 128 code?

An 8-bit unsigned number can represent values between 0 and 255. The excess representation can then represent values from -128 to +127.

+ 128	10000000	This is the excess.
+ 37	00100101	The value to be represented.
165	10100101	The representation of 37_{10} in excess 128 code.
+ 128	10000000	This is the excess.
- 23	00010111	The value to be represented.
105	01101001	The representation of -23_{10} in excess 128 code.
165	10100101	This is +37 in excess 128.
+ 105	+ 01101001	This is -23 in excess 128.
270	1 00001110	Note the carry out in this operation. 270 is too big to represent in 8 bits; to correct for the $2 \times E$ that is in this sum, subtract 128.
- 128	- 10000000	In binary, is this add or subtract?
142	10001110	This is the representation of 14, the correct result. in excess 128.

Táblázat

2.3: BCD

Table 2.3. Binary Coded Decimal
(BCD) Representations.

<i>Bit Pattern</i>	<i>Value</i>
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	Not valid
1011	Not valid
1100	Not valid
1101	Not valid
1110	Not valid
1111	Not valid

Példa 2.5:

Example 2.5: BCD excess 3 system: Consider a system that works with 3-digit decimal numbers, and it stores the digits in excess 3 format. What is the representation of **573**? What is the representation of **142**? Add the two numbers, and give the correct result in excess 3 format.

The numbers are handled on a digit-by-digit basis, with the excess being included with each digit:

Decimal	Binary	
+ 3 3 3	0011 0011 0011	This is the excess.
5 7 3	0101 0111 0011	And the number to be represented.
8 10 6	1000 1010 0110	The excess 3 representation.
+ 3 3 3	0011 0011 0011	This is the excess.
1 4 2	0001 0100 0010	This is the number to be represented.
4 7 5	0100 0111 0101	The excess 3 representation.
573	1000 1010 0110	Do the addition in decimal and in
+ 142	0100 0111 0101	binary. Correct as needed to
715	1100 1 0001 1011	make output correct.

Carry out of second set of 4 bits indicates that the most significant digit should be incremented by one. Also indicates that this value is **correct** as it stands (since $2 \times E = 6$ and the **carry** out indicates that the number overflowed into the next digit) so we need to add 3. Therefore, the MSD needs to **be** incremented by one and decremented by 3; the middle digit needs to have 3 added; and the LSD needs to be decremented by 3.

$$\begin{array}{r}
 (-3+1) \quad (+3) \quad (-3) \\
 \underline{-0010 + 0011 - 0011} \\
 1010 \quad 0100 \quad 1000 \\
 10 \quad 4 \quad 8
 \end{array}$$

Which is the correct excess 3 representation for **715**₁₀.

MSD: Most significant digit

LSD: Least significant digit

III.) Lebegőpontos rendszer:

- 7 különböző tényező: *a számrendszer alapja, előjele és nagysága, a mantissza alapja, előjele és hosszúsága, ill. a kitevő alapja.*

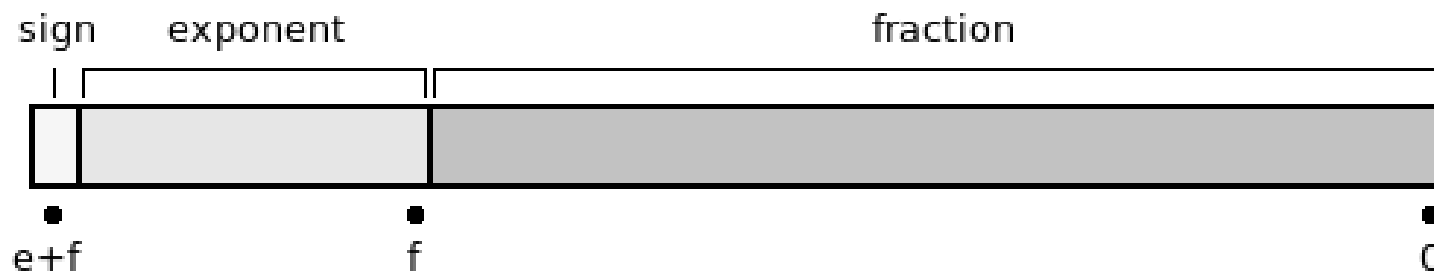
- matematikai jelölés:

$$(\text{előjel}) \text{ Mantissza} \times \text{Alap}^{\text{Kitevő}}$$

- Fixpontosnál nagyságrendekkel kisebb vagy nagyobb számok ábrázolására is mód van:
 - Pl: Avogadro-szám: $6.022 \cdot 10^{23}$
 - Pl: proton tömege $1.673 \cdot 10^{-24}$ g

IEEE 754-1985

- Szabvány bináris lebegőpontos számok tárolásánál, amely tartalmazza még a:
 - negatív zérust: $-0 = 111\dots 1$
 - Normalizálatlan számok
 - NaN: nem szám
 - $-/+$ végtelen



Sign-magnitude format („előjel-hossz” formátum): előjel külön kerül tárolásra (MSB), exponens eltolt (Excess-el kódolt), törtrész utána következik.

Lebegőpontos rendszer jellemzői

- Számrendszer / kitevő alapja: r_b r_e
- Mantissza értéke: $V_M = \sum_{i=0}^{N-1} d_i \times r_b^{i-p}$
 - Maximális: $V_{M_{\max}} = 0.d_m d_m d_m \dots = (1 - r_b^{-N})$
 - Minimális: $V_{M_{\min}} = 0.100 \dots = 1/r_b$
 - Radix pont helye: p
 - Mantissza bitjeinek száma: m
- Exponens értéke (max / min): V_E $V_{E_{\max}}$ $V_{E_{\min}}$
- Lebegőpontos szám értéke: $V_{\text{FPN}} = (-1)^{\text{SIGN}} V_M \times r_b^{V_E}$

Normalizált lebegőpontos rendszer jellemzői

- Ábrázolható maximális érték: $V_{FPN(MAX)} = V_{M(MAX)} \times r_b^{V_{E(MAX)}}$
- Ábrázolható minimális érték: $V_{FPN(MIN)} = V_{M(MIN)} \times r_b^{V_{E(MIN)}}$
- Legális mantisszák száma: $NLM_{FPN} = (r_b - 1) \times r_b^{m-1}$
- Legális exponensek száma: $NLE_{FPN} = V_{E(MAX)} + |V_{E(MIN)}| + 1_{ZERO}$
- Ábrázolható értékek száma: $NRV_{FPN} = NLM_{FPN} \times NLE_{FPN}$
- Normalizálás: mantissza értékét általában $[0... \sim 1]$ közé
- Pl: $32\,768_{10} = 0.32768 \times 10^5 = 3.2768 \times 10^4 = 32.768 \times 10^3 = 327.68 \times 10^2 = 3267.8 \times 10^1$

Példa: normalizált lebegőpontos rendszer

- Adott: Legyen $r_b = 10$, $r_e = 10$, $m = 3$, $e = 2$
- Kérdés: jellemző paraméterek?

- Megoldás: $V_{M(MAX)} = 0.999 = 1.000 - 10^{-3}$

$$V_{M(MIN)} = 0.100$$

$$V_{E(MAX)} = (r_e^e - 1) = 99$$

$$V_{E(MIN)} = -(r_e^e - 1) = -99$$

$$V_{FPN(MAX)} = 0.999 \times 10^{99}$$

$$V_{FPN(MIN)} = 0.100 \times 10^{-99}$$

$$NLM_{FPN} = 9 \times 10 \times 10 = 900 = 9 \times 10^2$$

$$NLE_{FPN} = 99 + |-99| + 1_{ZERO} = 199$$

$$NRV_{FPN} = 2 \times 900 \times 199 = 358,200$$

Normalizált lebegőpontos rdsz.

Table 2.4. 6-Bit Normalized Floating Point System, Base 2.

$$r_b = 2, \quad r_e = 2, \quad m = 4, \quad e = 2$$

				$V_E \rightarrow$	00	01	10	11
				$2^{V_E} \rightarrow$	1	2	4	8
V_M base 2				V_M	$V_M \times 2^{V_E}$			
1	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	1	2	4
1	0	0	1	$\frac{9}{16}$	$\frac{9}{16}$	$1\frac{1}{8}$	$2\frac{1}{4}$	$4\frac{1}{2}$
1	0	1	0	$\frac{5}{8}$	$\frac{5}{8}$	$1\frac{1}{4}$	$2\frac{1}{2}$	5
1	0	1	1	$\frac{11}{16}$	$\frac{11}{16}$	$1\frac{3}{8}$	$2\frac{3}{4}$	$5\frac{1}{2}$
1	1	0	0	$\frac{3}{4}$	$\frac{3}{4}$	$1\frac{1}{2}$	3	6
1	1	0	1	$\frac{13}{16}$	$\frac{13}{16}$	$1\frac{5}{8}$	$3\frac{1}{4}$	$6\frac{1}{2}$
1	1	1	0	$\frac{7}{8}$	$\frac{7}{8}$	$1\frac{3}{4}$	$3\frac{1}{2}$	7
1	1	1	1	$\frac{15}{16}$	$\frac{15}{16}$	$1\frac{7}{8}$	$3\frac{3}{4}$	$7\frac{1}{2}$

$$\text{Smallest fraction} = 0.1000_2 = \frac{1}{2}$$

$$\text{Largest fraction} = 0.1111_2 = \frac{15}{16}$$

$$\text{Smallest number} = 0.1000_2 \times 2^0 = \frac{1}{2}$$

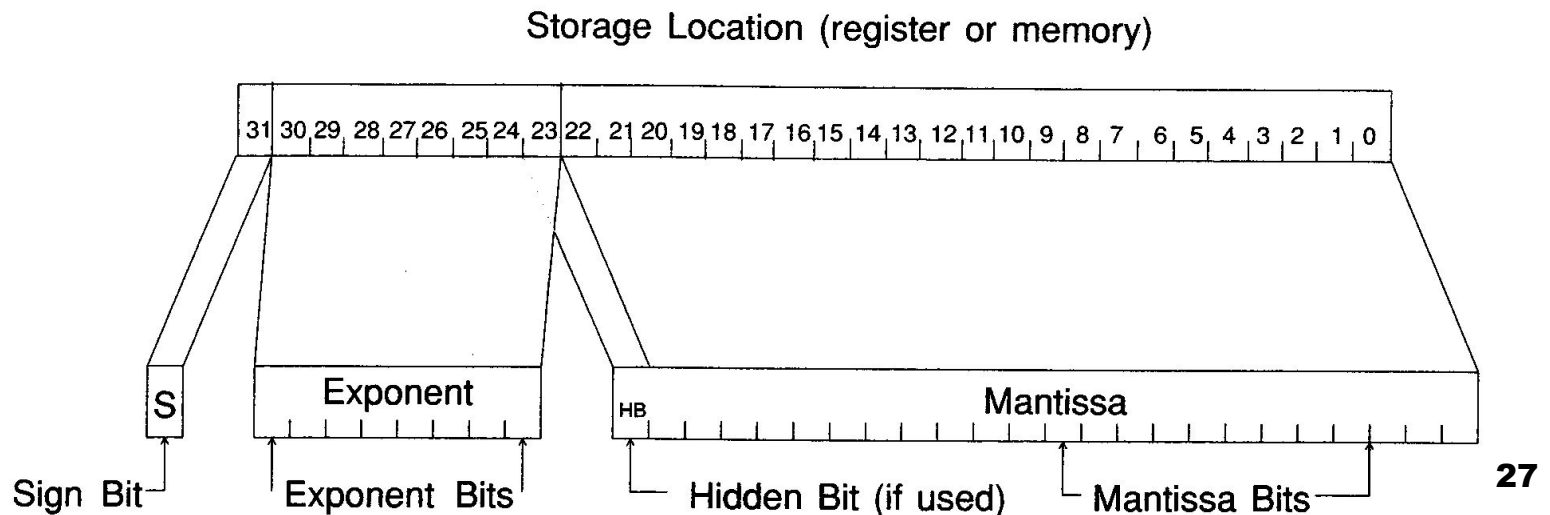
$$\text{Largest number} = 0.1111_2 \times 2^3 = 7\frac{1}{2}$$

$$\text{Number of fractions} = 1 \times 2 \times 2 \times 2 = 8$$

$$\text{Number of values} = 8 \times 4 = 32$$

Lebegőpontos ábrázolás: Rejtett (hidden bit) technika

- „Vezető” egyesek számát a legtöbb rendszer tervezésekor konstans értéként definiálják (1)
 - Ilyen a *mantissa legmagasabb helyiértékű bitje*, amelyet rejtett (*hidden: HB*) bitnek hívunk, amely közvetlenül az exponensbitek mögött helyezkedik el.
 - Ezt beállítva, duplájára nő a legális mantisszák, így az ábrázolható értékek száma is. Másrészt a nullát nem könnyen tudjuk reprezentálni, mivel a legkisebb ábrázolható érték a 00 0000, ami a HB ‘1’-es konstansként való definiálása miatt $0.1000_2 \times 2^0 = \frac{1}{2}$ -nek felel meg ($m=4$, $e=2$, $r_b=r_e=2$ esetén)!



Példa: 2-es alapú **DEC** 32-bites, normalizált lebegőpontos rendszer

- Adott: $r_b=2$, $r_e=2$, $m=23+(1)=24$ együtt, $p=24$ ($m=p!$), $e=8$, az exponenst tároljuk Excess-128 kódolással, és a számokat tároljuk "előjel-hossz" formátumban (~tekintsük a mantisszát pozitívnak).

$$V_{M(MIN)} = \underbrace{0.1000\dots_2}_{24db} = 1/2$$

$$V_{M(MAX)} = 0.1111\dots_2 = 0.999999940395 = 1.0 - 2^{-24}$$

$$V_{E(MIN)} = -(r_e^{e-1} - 1) = -(2^{8-1} - 1) = -127$$

$$V_{E(MAX)} = r_e^{e-1} - 1 = 2^{8-1} - 1 = 127$$

$$V_{FPN(MIN)} = 0.1000\dots_2 \times 2^{-127} = 2.9387 \times 10^{-39}$$

$$V_{FPN(MAX)} = 0.1111\dots_2 \times 2^{+127} = 1.7014 \times 10^{38}$$

$$NLM_{FPN} = 2^{23} = 8,388,608$$

$$NLE_{FPN} = 127 + |-127| + 1_{ZERO} = 255 = (r_e^e - 1) = 2^8 - 1$$

$$NRV_{FPN} = 2^{23} \times (2^8 - 1) = 2.139 \times 10^9$$

Példa: 16-os alapú **IBM-32** bites normalizált lebegőpontos rendszer

- Adott: $r_b=16$, $r_e=2$, $m=6$, $p=6$, $e=7$, az exponenst tároljuk Excess-64 kódolással, és a számokat tároljuk "előjel-hossz" formátumban (tekintsük a mantisszát pozitívnak).

$$V_{M(MIN)} = 0.100000_{16} = 1/16$$

$$V_{M(MAX)} = 0.FFFFFFFF_{16} = 0.999999940395 = 1.0 - 16^{-6}$$

$$V_{E(MIN)} = -(r_e^{e-1} - 1) = -(2^{7-1} - 1) = -63$$

$$V_{E(MAX)} = r_e^{e-1} - 1 = 2^{7-1} - 1 = 63$$

$$V_{FPN(MIN)} = 0.100000 \times 16^{-63} = 8.636 \times 10^{-78}$$

$$V_{FPN(MAX)} = 0.FFFFFFFF_{16} \times 16^{+63} = 7.237 \times 10^{75}$$

$$NLM_{FPN} = 15 \times 16^5 = 15,728,640$$

$$NLE_{FPN} = 63 + |-63| + 1_{ZERO} = 127 = 2^7 - 1$$

$$NRV_{FPN} = 15 \times 16^5 \times (2^7 - 1) = 1.9975 \times 10^9$$

Bővebb tartomány
mint a DEC!

7%-al kevesebb
mint a DEC!! **29**

Példa: IEEE-32 bites normalizált lebegőpontos rendszer

- Adott: $r_b=2$, $r_e=2$, $m=24$, de $p=23!$ (+HB!), , $e=8$, az exponenst tároljuk Excess-127 kódolással, és a számokat tároljuk "előjel-hossz" formátumban (~tekintsük a mantisszát pozitívnak).

$$V_{M(MIN)} = \underbrace{1.000\dots_2}_{23db} = 1$$

$$V_{M(MAX)} = 1.111\dots_2 = 1.99999988 = 2.0 - 2^{-23}$$

$$V_{E(MIN)} = -126 \quad // \text{ Zérus pontosabb ábrázolása miatt}$$

$$V_{E(MAX)} = 127$$

$$V_{FPN(MIN)} = 1.000\dots_2 \times 2^{-126} = 1.1755 \times 10^{-38} \quad // 4 \times \text{DEC!}$$

$$V_{FPN(MAX)} = 1.111\dots_2 \times 2^{+127} = 3.4028 \times 10^{38} \quad // 2 \times \text{DEC!}$$

$$NLM_{FPN} = 2^{23} = 8,388,608$$

$$NLE_{FPN} = 127 + |-126| + 1_{ZERO} = 254$$

$$NRV_{FPN} = 2^{23} \times (2^8 - 1) = 2.131 \times 10^9$$

- *DEC*: zérushoz közelítve szakadás (az első érték aránytalanul messze van)
- Ezt küszöböli ki az *IEEE* rendszer (Normalizálatlan tartományban lineárisan tart a zérushoz): ezért használja a $V_E = 126$

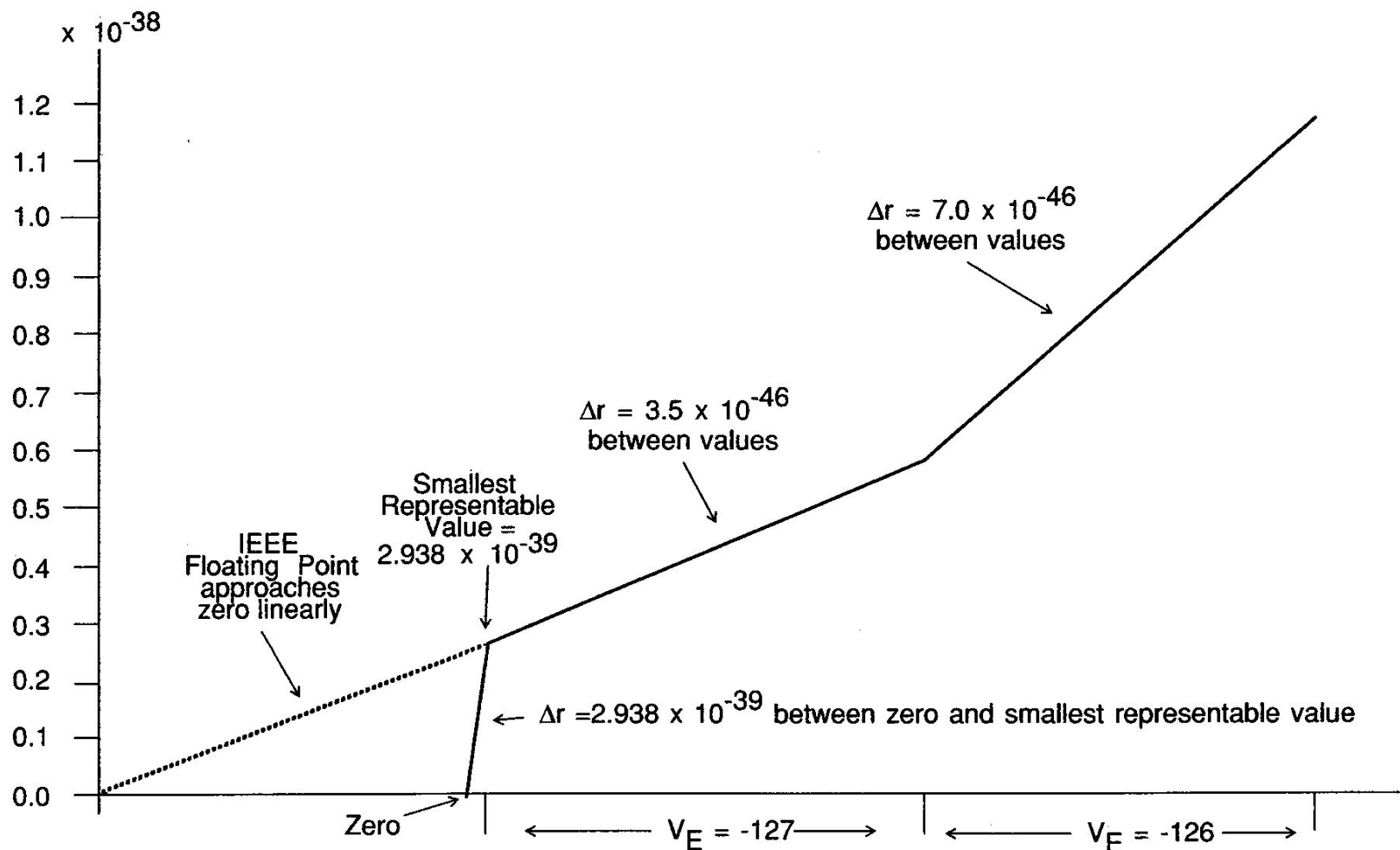


Figure 2.3. Values of the DEC Normalized Floating Point System Near Zero.

Példa: **IEEE-32** bites normalizált lebegőpontos rendszer (folyt.)

- $V_E = [-127, 126] = [1, 254]$ eltolt Excess-127 tartomány
- Speciális jelentőség:
 - $V_E = 0$ értékénél (zérus ábrázolása)
 - $V_E = 255$ értékénél lehetőség van bizonyos információk tárolására:

V_E	Előjel	Ábrázolás jelentése
$\neq 0$	1,0	Nem egy szám (NaN)
0	0	$+\infty$
0	1	$-\infty$



B) Nem-numerikus információ kódolása

Nem-numerikus információk

- Szöveges,
- Logikai (Boolean) információt,
- Grafikus szimbólumokat,
- és a címeket, vezérlési karaktereket értjük alattuk

Szöveges információ

- Minimális: 14 karakterből álló halmazban: számjegy (0-9), tizedes pont, pozitív ill. negatív jel, és üres karakter.
- + ábécé (A-Z), a központosítás, címkék és a formátumvezérlő karakterek (mint pl. vessző, tabulátor, (CR: Carriage Return) kocszi-vissza, soremelés (LF:Line Feed) , lapemelés (FF: From Feed), zárójel)
- Így elemek száma 46: 6 biten ábrázolható
$$\lceil \log_2 46 \rceil = 6 \text{ bit}$$
- De 7 biten tárolva már kisbetűs, mind pedig a nagybetűs karaktereket is magába foglalja

Szöveges információ kódolás

- BCD (Binary Coded Decimal): 6-biten
 - nagybetűk, számok, és speciális karakterek
- EBCDIC (Extended Binary Coded Decimal Interchange Code): 8-biten (A. Függelék)
 - + kisbetűs karaktereket és kiegészítő-információkat
 - 256 értékből nincs mindegyik kihasználva
 - Továbbá I és R betűknél szakadás van!
- ASCII (American Standard Code for Information Interchange): (A függelék) – alap 7-biten / extended 8-biten
- UTF-n (Universal Transformation Format): változó hosszúságú karakterkészlet (többnyelvűség támogatása)

EBCDIC

HEX DIGITS 1ST → 2ND ↓	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	SP SP010000	& SM030000	= SP100000	ø LO010000	Ø LO020000	° SM190000	μ SM170000	€ SC040000	{ SM110000	}	\	0
-1	9SP SP300000	é LE110000	/	Ê LE120000	a LA010000	j LJ010000	~	£	A 000	J LJ020000	÷	1
-2	â LA190000	ê LE190000	Â LA180000	Ê LE180000	b LB010000	k LX010000				K LX020000	S	2
-3	ä LA170000	ë LE170000	Ä LA180000	Ë LE180000	c LC010000	l LL010000				L LL020000	T	3
-4	à LA190000	è LE190000	À LA140000	È LE140000	d LD010000	m LM010000				M LM020000	U	4
-5	á LA110000	í LI110000	Á LA120000	Í LI120000	e LE010000	n LN010000	ı LV010000	ş SM040000	ı LE020000	N LN020000	V	5
-6	ä LA190000	î LI190000	Ã LA200000	Î LI180000	f LF010000	o LO010000	w LW010000	ŕ SM030000	F LF020000	O LO020000	W	6
-7	â LA270000	ï LI170000	Ä LA280000	Ï LI180000	g LG010000	p LP010000	x LX010000	Œ LO020000	G LG020000	P LP020000	X	7
-8	ç LC410000	ì LI130000	Ç LC420000	Ï LI140000	h LH010000	q LQ010000	y LY010000	œ LO010000	H LH020000	Q LQ020000	Y	8
-9	ñ LN190000	ß LS010000	Ñ LN200000	` SD130000	i LI010000	r LQ010000	z LZ010000	ÿ LY180000	I LI020000	R LR020000	Z	9
-A	ý LY120000	! SP020000	Š LS220000	:	« SP170000	® SM210000	ı SP030000	ı SM060000	ı SP020000	ı ND011000	ı ND021000	ı ND031000
-B	· SP110000	š SC030000	, SP080000	# SM010000	» SP180000	® SM200000	ı SP180000	š LS210000	ô LO150000	û LU150000	ô LO160000	û LU160000
-C	< SA030000	* SM040000	% SM020000	@ SM050000	ð LD060000	æ LA010000	Đ LD040000	ı SD010000	ö LO170000	ü LU170000	ö LO180000	ü LU180000
-D	(SP060000	) SP070000	= SP090000	ı SP050000	ý LY110000	ž LZ110000	[SM060000	]	ò LO130000	ù LU130000	ò LO140000	ù LU140000
-E	+ SA010000	ı SP140000	> SA050000	= SA040000	þ LT010000	Æ LA020000	Þ LT040000	Ž LZ020000	ó LO110000	ú LU110000	ó LO120000	ú LU120000
-F	 SM130000	^ SD150000	? SP150000	" SP040000	± SA020000	€ SC200000	® SM030000	×	õ LO190000	ÿ LY170000	ö LO200000	ı

Extended ASCII (1 byte)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0										Š	SP	°	À	Á	à	á
1										š	‘	±	¢	Ä	Å	ä
2										’	¢	z	Ã	ä	å	â
3										”	£	³	Ä	Å	ä	å
4										”	¤	´	Å	Ä	å	ö
5										•	¥	µ	İ	Ö	ö	”
6										–	‚	¶	Ÿ	Ÿ	Ÿ	Ÿ
7										–	Š	•	İ	Ç	÷	÷
8										~	”	‚	İ	È	é	”
9										™	©	¹	É	Ê	ë	ù
A										š	ª	¿	Ê	Ë	ê	°
B										+	;	«	»	Ë	ë	û
C										,	<	¼	¼	Ë	ë	ü
D										–	=	½	½	Ë	ë	ü
E										.	>	¾	¾	Ë	ë	ü
F										/	?	¿	¿	Ë	ë	ü

Legend



Code point contains a non-printable control character.



Code point is unoccupied; reserved for future use.

Unicode

Nyelvkészletek: Alap, - Latin 1/2, görög, cirill, héber, arab stb.

Általános írásjelek, matematikai, pénzügyi, mértani szimbólumok stb.

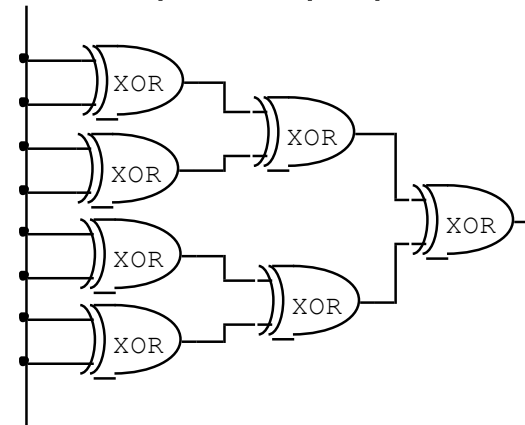
٢	٣	٤	٥	٦	٧	٨	٩	١٠	١١	١٢	١٣	١٤	١٥	١٦	١٧	١٨	١٩	٢٠	٢١	٢٢	٢٣	٢٤	٢٥	٢٦	٢٧	٢٨	٢٩	٣٠	٣١	٣٢	٣٣	٣٤	٣٥	٣٦	٣٧	٣٨	٣٩	٤٠	٤١	٤٢	٤٣	٤٤	٤٥	٤٦	٤٧	٤٨	٤٩	٥٠	٥١	٥٢	٥٣	٥٤	٥٥	٥٦	٥٧	٥٨	٥٩	٦٠	٦١	٦٢	٦٣	٦٤	٦٥	٦٦	٦٧	٦٨	٦٩	٧٠	٧١	٧٢	٧٣	٧٤	٧٥	٧٦	٧٧	٧٨	٧٩	٨٠	٨١	٨٢	٨٣	٨٤	٨٥	٨٦	٨٧	٨٨	٨٩	٩٠	٩١	٩٢	٩٣	٩٤	٩٥	٩٦	٩٧	٩٨	٩٩	١٠٠	١٠١	١٠٢	١٠٣	١٠٤	١٠٥	١٠٦	١٠٧	١٠٨	١٠٩	١١٠	١١١	١١٢	١١٣	١١٤	١١٥	١١٦	١١٧	١١٨	١١٩	١٢٠	١٢١	١٢٢	١٢٣	١٢٤	١٢٥	١٢٦	١٢٧	١٢٨	١٢٩	١٣٠	١٣١	١٣٢	١٣٣	١٣٤	١٣٥	١٣٦	١٣٧	١٣٨	١٣٩	١٤٠	١٤١	١٤٢	١٤٣	١٤٤	١٤٥	١٤٦	١٤٧	١٤٨	١٤٩	١٥٠	١٥١	١٥٢	١٥٣	١٥٤	١٥٥	١٥٦	١٥٧	١٥٨	١٥٩	١٦٠	١٦١	١٦٢	١٦٣	١٦٤	١٦٥	١٦٦	١٦٧	١٦٨	١٦٩	١٧٠	١٧١	١٧٢	١٧٣	١٧٤	١٧٥	١٧٦	١٧٧	١٧٨	١٧٩	١٨٠	١٨١	١٨٢	١٨٣	١٨٤	١٨٥	١٨٦	١٨٧	١٨٨	١٨٩	١٩٠	١٩١	١٩٢	١٩٣	١٩٤	١٩٥	١٩٦	١٩٧	١٩٨	١٩٩	٢٠٠	٢٠١	٢٠٢	٢٠٣	٢٠٤	٢٠٥	٢٠٦	٢٠٧	٢٠٨	٢٠٩	٢١٠	٢١١	٢١٢	٢١٣	٢١٤	٢١٥	٢١٦	٢١٧	٢١٨	٢١٩	٢٢٠	٢٢١	٢٢٢	٢٢٣	٢٢٤	٢٢٥	٢٢٦	٢٢٧	٢٢٨	٢٢٩	٢٣٠	٢٣١	٢٣٢	٢٣٣	٢٣٤	٢٣٥	٢٣٦	٢٣٧	٢٣٨	٢٣٩	٢٤٠	٢٤١	٢٤٢	٢٤٣	٢٤٤	٢٤٥	٢٤٦	٢٤٧	٢٤٨	٢٤٩	٢٥٠	٢٥١	٢٥٢	٢٥٣	٢٥٤	٢٥٥	٢٥٦	٢٥٧	٢٥٨	٢٥٩	٢٦٠	٢٦١	٢٦٢	٢٦٣	٢٦٤	٢٦٥	٢٦٦	٢٦٧	٢٦٨	٢٦٩	٢٧٠	٢٧١	٢٧٢	٢٧٣	٢٧٤	٢٧٥	٢٧٦	٢٧٧	٢٧٨	٢٧٩	٢٨٠	٢٨١	٢٨٢	٢٨٣	٢٨٤	٢٨٥	٢٨٦	٢٨٧	٢٨٨	٢٨٩	٢٩٠	٢٩١	٢٩٢	٢٩٣	٢٩٤	٢٩٥	٢٩٦	٢٩٧	٢٩٨	٢٩٩	٣٠٠	٣٠١	٣٠٢	٣٠٣	٣٠٤	٣٠٥	٣٠٦	٣٠٧	٣٠٨	٣٠٩	٣١٠	٣١١	٣١٢	٣١٣	٣١٤	٣١٥	٣١٦	٣١٧	٣١٨	٣١٩	٣٢٠	٣٢١	٣٢٢	٣٢٣	٣٢٤	٣٢٥	٣٢٦	٣٢٧	٣٢٨	٣٢٩	٣٣٠	٣٣١	٣٣٢	٣٣٣	٣٣٤	٣٣٥	٣٣٦	٣٣٧	٣٣٨	٣٣٩	٣٤٠	٣٤١	٣٤٢	٣٤٣	٣٤٤	٣٤٥	٣٤٦	٣٤٧	٣٤٨	٣٤٩	٣٥٠	٣٥١	٣٥٢	٣٥٣	٣٥٤	٣٥٥	٣٥٦	٣٥٧	٣٥٨	٣٥٩	٣٦٠	٣٦١	٣٦٢	٣٦٣	٣٦٤	٣٦٥	٣٦٦	٣٦٧	٣٦٨	٣٦٩	٣٧٠	٣٧١	٣٧٢	٣٧٣	٣٧٤	٣٧٥	٣٧٦	٣٧٧	٣٧٨	٣٧٩	٣٨٠	٣٨١	٣٨٢	٣٨٣	٣٨٤	٣٨٥	٣٨٦	٣٨٧	٣٨٨	٣٨٩	٣٩٠	٣٩١	٣٩٢	٣٩٣	٣٩٤	٣٩٥	٣٩٦	٣٩٧	٣٩٨	٣٩٩	٤٠٠	٤٠١	٤٠٢	٤٠٣	٤٠٤	٤٠٥	٤٠٦	٤٠٧	٤٠٨	٤٠٩	٤١٠	٤١١	٤١٢	٤١٣	٤١٤	٤١٥	٤١٦	٤١٧	٤١٨	٤١٩	٤٢٠	٤٢١	٤٢٢	٤٢٣	٤٢٤	٤٢٥	٤٢٦	٤٢٧	٤٢٨	٤٢٩	٤٣٠	٤٣١	٤٣٢	٤٣٣	٤٣٤	٤٣٥	٤٣٦	٤٣٧	٤٣٨	٤٣٩	٤٤٠	٤٤١	٤٤٢	٤٤٣	٤٤٤	٤٤٥	٤٤٦	٤٤٧	٤٤٨	٤٤٩	٤٥٠	٤٥١	٤٥٢	٤٥٣	٤٥٤	٤٥٥	٤٥٦	٤٥٧	٤٥٨	٤٥٩	٤٦٠	٤٦١	٤٦٢	٤٦٣	٤٦٤	٤٦٥	٤٦٦	٤٦٧	٤٦٨	٤٦٩	٤٧٠	٤٧١	٤٧٢	٤٧٣	٤٧٤	٤٧٥	٤٧٦	٤٧٧	٤٧٨	٤٧٩	٤٨٠	٤٨١	٤٨٢	٤٨٣	٤٨٤	٤٨٥	٤٨٦	٤٨٧	٤٨٨	٤٨٩	٤٩٠	٤٩١	٤٩٢	٤٩٣	٤٩٤	٤٩٥	٤٩٦	٤٩٧	٤٩٨	٤٩٩	٥٠٠	٥٠١	٥٠٢	٥٠٣	٥٠٤	٥٠٥	٥٠٦	٥٠٧	٥٠٨	٥٠٩	٥١٠	٥١١	٥١٢	٥١٣	٥١٤	٥١٥	٥١٦	٥١٧	٥١٨	٥١٩	٥٢٠	٥٢١	٥٢٢	٥٢٣	٥٢٤	٥٢٥	٥٢٦	٥٢٧	٥٢٨	٥٢٩	٥٣٠	٥٣١	٥٣٢	٥٣٣	٥٣٤	٥٣٥	٥٣٦	٥٣٧	٥٣٨	٥٣٩	٥٤٠	٥٤١	٥٤٢	٥٤٣	٥٤٤	٥٤٥	٥٤٦	٥٤٧	٥٤٨	٥٤٩	٥٥٠	٥٥١	٥٥٢	٥٥٣	٥٥٤	٥٥٥	٥٥٦	٥٥٧	٥٥٨	٥٥٩	٥٦٠	٥٦١	٥٦٢	٥٦٣	٥٦٤	٥٦٥	٥٦٦	٥٦٧	٥٦٨	٥٦٩	٥٧٠	٥٧١	٥٧٢	٥٧٣	٥٧٤	٥٧٥	٥٧٦	٥٧٧	٥٧٨	٥٧٩	٥٨٠	٥٨١	٥٨٢	٥٨٣	٥٨٤	٥٨٥	٥٨٦	٥٨٧	٥٨٨	٥٨٩	٥٩٠	٥٩١	٥٩٢	٥٩٣	٥٩٤	٥٩٥	٥٩٦	٥٩٧	٥٩٨	٥٩٩	٦٠٠	٦٠١	٦٠٢	٦٠٣	٦٠٤	٦٠٥	٦٠٦	٦٠٧	٦٠٨	٦٠٩	٦١٠	٦١١	٦١٢	٦١٣	٦١٤	٦١٥	٦١٦	٦١٧	٦١٨	٦١٩	٦٢٠	٦٢١	٦٢٢	٦٢٣	٦٢٤	٦٢٥	٦٢٦	٦٢٧	٦٢٨	٦٢٩	٦٣٠	٦٣١	٦٣٢	٦٣٣	٦٣٤	٦٣٥	٦٣٦	٦٣٧	٦٣٨	٦٣٩	٦٤٠	٦٤١	٦٤٢	٦٤٣	٦٤٤	٦٤٥	٦٤٦	٦٤٧	٦٤٨	٦٤٩	٦٥٠	٦٥١	٦٥٢	٦٥٣	٦٥٤	٦٥٥	٦٥٦	٦٥٧	٦٥٨	٦٥٩	٦٦٠	٦٦١	٦٦٢	٦٦٣	٦٦٤	٦٦٥	٦٦٦	٦٦٧	٦٦٨	٦٦٩	٦٧٠	٦٧١	٦٧٢	٦٧٣	٦٧٤	٦٧٥	٦٧٦	٦٧٧	٦٧٨	٦٧٩	٦٨٠	٦٨١	٦٨٢	٦٨٣	٦٨٤	٦٨٥	٦٨٦	٦٨٧	٦٨٨	٦٨٩	٦٩٠	٦٩١	٦٩٢	٦٩٣	٦٩٤	٦٩٥	٦٩٦	٦٩٧	٦٩٨	٦٩٩	٧٠٠	٧٠١	٧٠٢	٧٠٣	٧٠٤	٧٠٥	٧٠٦	٧٠٧	٧٠٨	٧٠٩	٧١٠	٧١١	٧١٢	٧١٣	٧١٤	٧١٥	٧١٦	٧١٧	٧١٨	٧١٩	٧٢٠	٧٢١	٧٢٢	٧٢٣	٧٢٤	٧٢٥	٧٢٦	٧٢٧	٧٢٨	٧٢٩	٧٣٠	٧٣١	٧٣٢	٧٣٣	٧٣٤	٧٣٥	٧٣٦	٧٣٧	٧٣٨	٧٣٩	٧٤٠	٧٤١	٧٤٢	٧٤٣	٧٤٤	٧٤٥	٧٤٦	٧٤٧	٧٤٨	٧٤٩	٧٥٠	٧٥١	٧٥٢	٧٥٣	٧٥٤	٧٥٥	٧٥٦	٧٥٧	٧٥٨	٧٥٩	٧٦٠	٧٦١	٧٦٢	٧٦٣	٧٦٤	٧٦٥	٧٦٦	٧٦٧	٧٦٨	٧٦٩	٧٧٠	٧٧١	٧٧٢	٧٧٣	٧٧٤	٧٧٥	٧٧٦	٧٧٧	٧٧٨	٧٧٩	٧٨٠	٧٨١	٧٨٢	٧٨٣	٧٨٤	٧٨٥	٧٨٦	٧٨٧	٧٨٨	٧٨٩	٧٩٠	٧٩١	٧٩٢	٧٩٣	٧٩٤	٧٩٥	٧٩٦	٧٩٧	٧٩٨	٧٩٩	٨٠٠	٨٠١	٨٠٢	٨٠٣	٨٠٤	٨٠٥	٨٠٦	٨٠٧	٨٠٨	٨٠٩	٨١٠	٨١١	٨١٢	٨١٣	٨١٤	٨١٥	٨١٦	٨١٧	٨١٨	٨١٩	٨٢٠	٨٢١	٨٢٢	٨٢٣	٨٢٤	٨٢٥	٨٢٦	٨٢٧	٨٢٨	٨٢٩	٨٣٠	٨٣١	٨٣٢	٨٣٣	٨٣٤	٨٣٥	٨٣٦	٨٣٧	٨٣٨	٨٣٩	٨٤٠	٨٤١	٨٤٢	٨٤٣	٨٤٤	٨٤٥	٨٤٦	٨٤٧	٨٤٨	٨٤٩	٨٥٠	٨٥١	٨٥٢	٨٥٣	٨٥٤	٨٥٥	٨٥٦	٨٥٧	٨٥٨	٨٥٩	٨٦٠	٨٦١	٨٦٢	٨٦٣	٨٦٤	٨٦٥	٨٦٦	٨٦٧	٨٦٨	٨٦٩	٨٧٠	٨٧١	٨٧٢	٨٧٣	٨٧٤	٨٧٥	٨٧٦	٨٧٧	٨٧٨	٨٧٩	٨٨٠	٨٨١	٨٨٢	٨٨٣	٨٨٤	٨٨٥	٨٨٦	٨٨٧	٨٨٨	٨٨٩	٨٩٠	٨٩١	٨٩٢	٨٩٣	٨٩٤	٨٩٥	٨٩٦	٨٩٧	٨٩٨	٨٩٩	٩٠٠	٩٠١	٩٠٢	٩٠٣	٩٠٤	٩٠٥	٩٠٦	٩٠٧	٩٠٨	٩٠٩	٩١٠	٩١١	٩١٢	٩١٣	٩١٤	٩١٥	٩١٦	٩١٧	٩١٨	٩١٩	٩٢٠	٩٢١	٩٢٢	٩٢٣	٩٢٤	٩٢٥	٩٢٦	٩٢٧	٩٢٨	٩٢٩	٩٣٠	٩٣١	٩٣٢	٩٣٣	٩٣٤	٩٣٥	٩٣٦	٩٣٧	٩٣٨	٩٣٩	٩٤٠	٩٤١	٩٤٢	٩٤٣	٩٤٤	٩٤٥	٩٤٦	٩٤٧	٩٤٨	٩٤٩	٩٥٠	٩٥١	٩٥٢	٩٥٣	٩٥٤	٩٥٥	٩٥٦	٩٥٧	٩٥٨	٩٥٩	٩٦٠	٩٦١	٩٦٢	٩٦٣	٩٦٤	٩٦٥	٩٦٦	٩٦٧	٩٦٨	٩٦٩	٩٧٠	٩٧١	٩٧٢	٩٧٣	٩٧٤	٩٧٥	٩٧٦	٩٧٧	٩٧٨	٩٧٩	٩٨٠	٩٨١	٩٨٢	٩٨٣	٩٨٤	٩٨٥	٩٨٦	٩٨٧	٩٨٨	٩٨٩	٩٩٠	٩٩١	٩٩٢	٩٩٣	٩٩٤	٩٩٥	٩٩٦	٩٩٧	٩٩٨	٩٩٩	١٠٠٠	١٠٠١	١٠٠٢	١٠٠٣	١٠٠٤	١٠٠٥	١٠٠٦	١٠٠٧	١٠٠٨	١٠٠٩	١٠١٠	١٠١١	١٠١٢	١٠١٣	١٠١٤	١٠١٥	١٠١٦	١٠١٧	١٠١٨	١٠١٩	١٠٢٠	١٠٢١	١٠٢٢	١٠٢٣	١٠٢٤	١٠٢٥	١٠٢٦	١٠٢٧	١٠٢٨	١٠٢٩	١٠٣٠	١٠٣١	١٠٣٢	١٠٣٣	١٠٣٤	١٠٣٥	١٠٣٦	١٠٣٧	١٠٣٨	١٠٣٩	١٠٤٠	١٠٤١	١٠٤٢	١٠٤٣	١٠٤٤	١٠٤٥	١٠٤٦	١٠٤٧	١٠٤٨	١٠٤٩	١٠٥٠	١٠٥١	١٠٥٢	١٠٥٣	١٠٥٤	١٠٥٥	١٠٥٦	١٠٥٧	١٠٥٨	١٠٥٩	١٠٦٠	١٠٦١	١٠٦٢	١٠٦٣	١٠٦٤	١٠٦٥	١٠٦٦	١٠٦٧	١٠٦٨	١٠٦٩	١٠٧٠	١٠٧١	١٠٧٢	١٠٧٣	١٠٧٤	١٠٧٥	١٠٧٦	١٠٧٧	١٠٧٨	١٠٧٩	١٠٨٠	١٠٨١	١٠٨٢	١٠٨٣	١٠٨٤	١٠٨٥	١٠٨٦	١٠٨٧	١٠٨٨	١٠٨٩	١٠٩٠	١٠٩١	١٠٩٢	١٠٩٣	١٠٩٤	١٠٩٥	١٠٩٦	١٠٩٧	١٠٩٨	١٠٩٩	١١٠٠	١١٠١	١١٠٢	١١٠٣	١١٠٤	١١٠٥	١١٠٦	١١٠٧	١١٠٨	١١٠٩	١١١٠	١١١١	١١١٢	١١١٣	١١١٤	١١١٥	١١١٦	١١١٧	١١١٨	١١١٩	١١٢٠	١١٢١	١١٢٢	١١٢٣	
---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	--

Hibakódolás - Hibadetektálás és Javítás

- N bit segítségével 2^N különböző érték, cím, vagy utasítás ábrázolható
- 1 bittel növelve (N+1) esetén: 2^N -ről 2^{N+1} -re, megduplázódik
- Redundancia, detektálás, hibajavítás

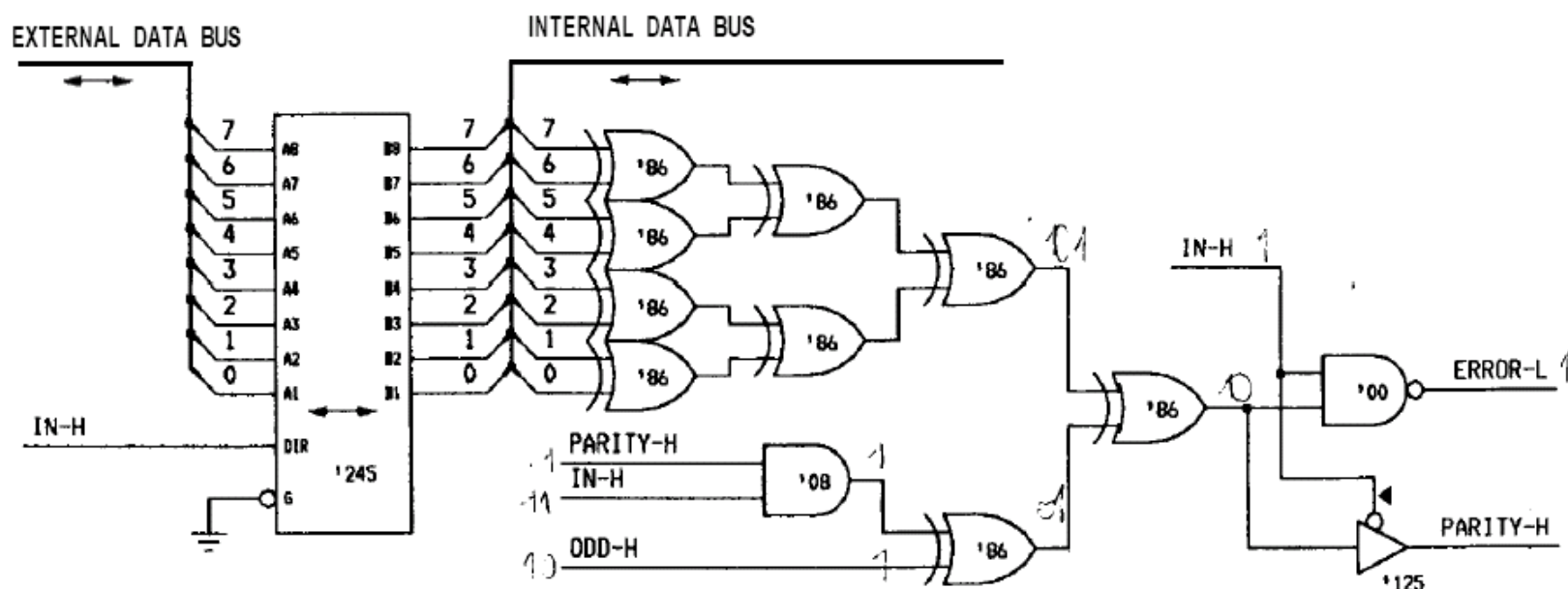
Paritás bit ellenőrzés

- Páros v. páratlan paritás: N bites információ egy kiegészítő bittel bővül → egyszeres hiba felismerése (bitek számát párosra v. páratlanra egészíti ki)
 - Pl: leragadásból: '0'-ból '1'-es lesz, vagy fordítva
 - Pl: ideiglenes, tranziens jellegű hiba
 - 8 adatbithez paritásbit generálás (XOR) (IC '280)



8 bites adatút kétirányú paritásbit ellenőrzéssel

- $IN-H=1 \rightarrow PARITY-H=1$ paritás ell.
- Ha $ODD-H=1$ páratlan paritást használ
- Hiba esetén: $ERROR-L=1$

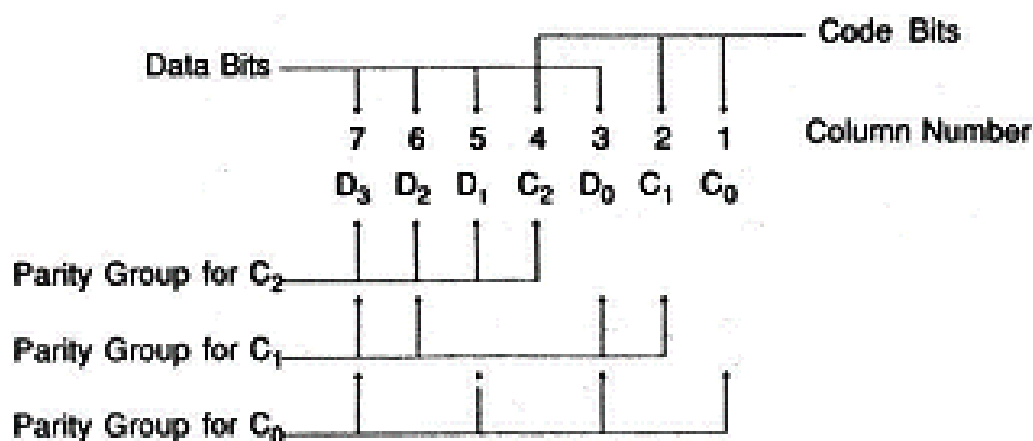


Hamming kód

- Több redundáns bittel nemcsak a hiba meglétét, és helyét tudjuk detektálni, hanem a hibás bitet javítani is tudjuk
- Hamming kód: egy biten tároljuk a bitmintázatok azonos helyiértékű bitjeinek különbségét, tehát egybites hibát lehet vele *javítani*.

Hamming kódú kódszó konstruálása (pl. 7 – bites kódszóra)

- 2^N-1 bites Hamming kód: N kódbit, 2^N-N-1 adatbit
- Összesen 7 biten **4 adatbitet** (D_0, D_1, D_2, D_3), **3 kódbittel** (C_0, C_1, C_2) kódolunk
- C_i kódbitek a bináris súlyuknak megfelelő bitpozíciókban
- A maradék pozíciókat rendre adatbitekkel töltjük fel (D_i)

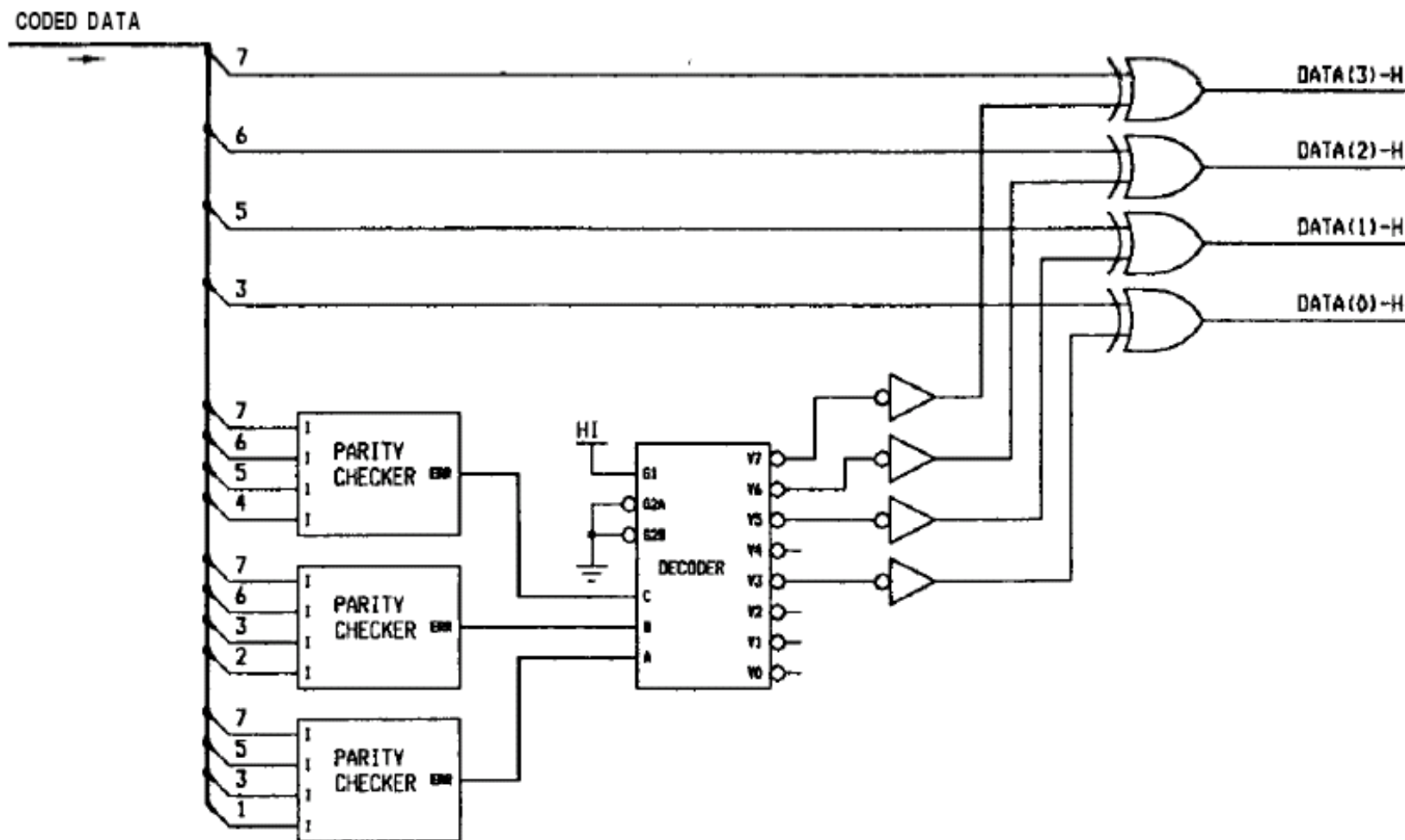


Paritás-csoportok	Bit pozíciók	Bitek jelölései
0	1, 3, 5, 7	C_0, D_0, D_1, D_3
1	2, 3, 6, 7	C_1, D_0, D_2, D_3
2	4, 5, 6, 7	C_2, D_1, D_2, D_3

Hamming kódú hibajavító áramkör tervezése

- 3 kódbitünk van, így írásnál 3 paritásbit generáló-, míg olvasásnál 3 paritás-ellenőrző áramkör kell.
- Példa: bemeneti adatbit-mintázatunk 0101.
 - Mivel páratlan paritást alkalmazunk, a megfelelő helyen szereplő kódbitekkel kiegészítve a következő szót kapjuk: 0100110. Ha nincs hiba, a paritásellenőrzők (C0, C1, C2) kimenete 000, minden egyes paritáscsoportra.
 - Hiba esetén például, ha az input mintázat 0100010, akkor a paritásellenőrző hibát észlel. Ugyan C2. paritásbitcsoport rendben, de a C1. és C0. hibás:
 - 011 az azonosított minta a harmadik oszlopban (**D0** helyén).
 - Javításként *invertáljuk* a 3. bitpozícióban lévő bitet. 0100010 $\Rightarrow$ 0100110. Ekkor a kódbitek a következőképpen módosulnak a páratlan paritásnak megfelelően: C0=1, C1=1 és C2=0.

7-bites Hamming kódú hibajavító áramkör felépítése



Példa:

Hamming kód (DEB-el) 8 adatbitre: mi a helyes ábrázolása 8 biten a 01011100 adatbit mintázatnak. Szükséges 8 adatbit (D0-D7), 4 kódbit (C0-C3) és egy kettős hibajelző bit (DEB). Páratlan paritást alkalmazunk. (BW-binary weight jelenti az egyes oszlopok bináris súlyát, 1,2, 4 ill 8 biten).

13	12	11	10	9	8	7	6	5	4	3	2	1	Oszlopszám
DEB	D7	D6	D5	D4	C3	D3	D2	D1	C2	D0	C1	C0	
	1	1	1	1	1	0	0	0	0	0	0	0	BW, 8bit
	1	0	0	0	0	1	1	1	1	0	0	0	BW, 4 bit
	0	1	1	0	0	1	1	0	0	1	1	0	BW, 2 bit
	0	1	0	1	0	1	0	1	0	1	0	1	BW, 1 bit

Paritáscsoportok	Bit pozíciók	Bitek jelölései
0	1, 3, 5, 7, 9, 11	C0, D0, D1, D3, D4, D6
1	2, 3, 6, 7, 10, 11	C1, D0, D2, D3, D5, D6
2	4, 5, 6, 7, 12	C2, D1, D2, D3, D7
3	8, 9, 10, 11, 12	C3, D4, D5, D6, D7

13	12	11	10	9	8	7	6	5	4	3	2	1	Oszlopszám
DEB	D7	D6	D5	D4	C3	D3	D2	D1	C2	D0	C1	C0	
–	0	1	0	1	–	1	1	0	–	0	–	–	Adatbitek
1	0	1	0	1	1	1	1	0	1	0	0	0	Hozzáadott

kódbitek. Tehát a helyes ábrázolása 01011100-nek a következő:
1010111101000.