Moduri de reprezentare a datelor utilizate de microprocesorul Intel 8086

1. Elemente de algebră booleană

Algebra logică sau *algebra booleană* a fost concepută ca o metodă simbolică de tratare a funcțiilor logice formale, fiind ulterior extinsă şi în alte domenii. Deoarece între logica formală şi circuitele de comutație (cu două stări) există o corespondență directă (logica formală studiază valoarea de adevăr sau de fals a unor afirmații ce pot fi numai adevărate sau false, iar circuitele de comutație sunt realizate prin interconectarea unor comutatoare ce au două stări: închis sau deschis), algebra booleană s-a impus ca cea mai importantă modalitate de analiză şi sinteză a circuitelor de comutație.

Algebra booleană se defineşte axiomatic astfel: fie (M, x, +) o mulțime cu două operații şi M={0, 1} cu următoarele *proprietăți (axiome)*:

a) pentru orice elemente a1, a2 care apartin lui M există relațiile (*parte stabilă*):
a1 a2 apartine lui M şi a1 + a2 apartine lui M
b) operațiile x şi + sunt *comutative*:
a1 x a2 = a2 x a1 şi a1 + a2 = a2 + a1
c) operațiile x şi + sunt *asociative*:
(a1 x a2) x a3 = a1 x (a2 x a3) şi (a1 + a2) + a3 = a1 + (a2 + a3)
d) operațiile x şi + sunt *distributive una față de alta*:
a1 x (a2 + a3) = a1 x a2 + a1 x a3;
a1 + (a2 x a3) = (a1 + a2) x (a1 + a3);
e) există *element neutru* pentru fiecare operație:
a x 1 = a şi a + 0 = a;
f) există *element invers* (*complementar*) pentru fiecare operație:
a x a\ = a şi a + a\ = 1;

Valorile 0 şi 1 modelează logic stările circuitelor logice (digitale) care au două stări. În continuare operația x va fi denumită *conjuncție*, iar operația + va fi denumită *disjuncție*.

Există următoarele *teoreme (reguli de calcul) asociate algebrei boolene*:

1. a x a = a şi a+a = a (idempotența)
2. a x 0 = 0 şi a+1 = 1 (elementele neutre)
3. a\\ = a (dubla negație)
4. a(a + b) şi a + a x b = a (absorbția)
5. (a+b)\ = a\ x b\ şi (a x b)\ = a\ + b\ (relatiile de Morgan)

Operațiile x şi + pot fi ilustrate cu ajutorul *tabelului de adevăr*:

a	b	a x b	a + b	a\
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Operația + reprezintă *suma modulo 2* (SAU exclusiv) dată de tabelul de adevăr:

a1	a2	a1 + a2
0	0	0
0	1	1
1	0	1
1	1	0

Notațiile utilizate pentru operațiile anterioare sunt: x pentru AND, + pentru OR, + pentru XOR si \ pentru NOT.

2. Baze de numeratie

Reprezentarea unui număr *în baza B* este:

$$(N)_B = \underbrace{c_{n-1} \ldots\ldots\ldots\ldots c_0}_{\textit{parte întreagă}} \;.\; \underbrace{c_{-1} \ldots\ldots\ldots\ldots\ldots c_{-m}}_{\textit{parte fracționară}}$$

unde c_i {0, ... , B-1}, i = n-1, ... , -m.

Numărul N scris in baza B ca mai sus, are *în baza obisnuita 10* valoarea:

$$N = c_{n-1}\, B^{n-1} + \ldots\ldots + c_1\, B^1 + c_0\, B^0 \;.\; c_{-1}\, B^{-1} + c_{-2}\, B^{-2} + \ldots\ldots + c_{-m}\, B^{-m}$$

Valoarea unui numar N scris in baza 10 este de exemplu:

$$N = z_{n-1}\, 10^{n-1} + \ldots\ldots + z_1\, 10^1 + z_0\, 10^0 \;.\; z_{-1}\, 10^{-1} + z_{-2}\, 10^{-2} + \ldots\ldots + z_{-m}\, 10^{-m}$$

Numarul scris N scris in baza 2:

$$(N)_2 = b_{n-1} \ldots\ldots\ldots b_0 \;.\; b_{-1} \ldots\ldots\ldots\ldots b_{-m}$$

are *în baza obisnuita 10* valoarea:

$$N = b_{n-1}\, 2^{n-1} + \ldots\ldots + b_1\, 2^1 + b_0\, 2^0 \;.\; b_{-1}\, 2^{-1} + b_{-2}\, 2^{-2} + \ldots\ldots + b_{-m}\, 2^{-m}$$

Deoarece informatia digitala este memorata sub forma binara, baza de numeratie utilizata este 2. *Cifrele* in *baza 2* se numesc *biti.*

Valorile obisnuite din *baza 10* trebuie deci echivalate in baza 2 pentru efectuarea unor calcule. De aceea este importanta cunoasterea echivalentei valorilor zecimale cu valori binare. In tabelul 1 sunt date *echivalentele primelor 32 de valori zecimale intregi* in binar.

Tab.1. Echivalenta valorilor zecimale (0..31), binare si hexazecimale

Valoare in baza 10 (zecimala)		Valoare in baza 2 (binara)	Valoare in baza 16 (hexa-zecimala)
Propriu-zisa	Ca suma de puteri ale lui 2		
0		0	0
1	2^0	1	1
2	2^1	10	2
3	$2^1 + 2^0$	11	3
4	2^2	100	4
5	$2^2 + 2^0$	101	5
6	$2^2 + 2^1$	110	6
7	$2^2 + 2^1 + 2^0$	111	7
8	2^3	1000	8
9	$2^3 + 2^0$	1001	9
10	$2^3 + 2^1$	1010	A
11	$2^3 + 2^1 + 2^0$	1011	B
12	$2^3 + 2^2$	1100	C
13	$2^3 + 2^2 + 2^0$	1101	D
14	$2^3 + 2^2 + 2^1$	1110	E
15	$2^3 + 2^2 + 2^1 + 2^0$	1111	F
16	2^4	1 0000	10
17	$2^4 + 2^0$	1 0001	11
18	$2^4 + 2^1$	1 0010	12
19	$2^4 + 2^1 + 2^0$	1 0011	13
20	$2^4 + 2^2$	1 0100	14
21	$2^4 + 2^2 + 2^0$	1 0101	15
22	$2^4 + 2^2 + 2^1$	1 0110	16
23	$2^4 + 2^2 + 2^1 + 2^0$	1 0111	17
24	$2^4 + 2^3$	1 1000	18
25	$2^4 + 2^3 + 2^0$	1 1001	19
26	$2^4 + 2^3 + 2^1$	1 1010	1A
27	$2^4 + 2^3 + 2^1 + 2^0$	1 1011	1B
28	$2^4 + 2^3 + 2^2$	1 1100	1C
29	$2^4 + 2^3 + 2^2 + 2^0$	1 1101	1D
30	$2^4 + 2^3 + 2^2 + 2^1$	1 1110	1E
31	$2^4 + 2^3 + 2^2 + 2^1 + 2^0$	1 1111	1F

Pe de alta parte, pentru usurinta scrierii valorilor care au un numar mare de cifre binare, se utilizeaza *baza 16*. Cifrele in baza 16 se numesc *cifre hexa-zecimale*, sau mai scurt *cifre hexa.*

Convenim sa utilizam terminatia **B** pentru *valori binare* si terminatia **H** pentru *valori hexa-zecimale*, valorile zecimale fiind scrise obisnuit.

Conversia unui număr din baza 2 în baza B=2^m se efectuează astfel: se împarte numărul scris în binar în grupe de cifre de câte *m* biţi şi se reprezintă fiecare grupă printr-o cifră în baza B. Astfel, deoarece 16 este 2^4, rezulta ca se pot grupa 4 cifre binare pentru a forma 1 cifra hexa-zecimala.

Exemplu: 0111 1111 B = 7 F H

Se observa urmatoarele *echivalente*:

1 cifra hexa = 4 cifre binare (4 biti)
2 cifre hexa = 8 cifre binare (1 octet)
4 cifre hexa = 16 cifre binare (1 cuvant de 16 biti)

In tabelul 1 sunt date *echivalentele primelor 32 de valori zecimale intregi* si in hexa-zecimal, iar in tabelul 2 sunt date *echivalentele puterilor lui 2 in binar si hexa-zecimal.*

Tab.2. Echivalenta puterilor lui 2 ca valori zecimale, binare si hexazecimale

Valoare in baza 10 (zecimala)			Valoare in baza 2 (binara)	Valoare in baza 16 (hexa-zecimala)
Propriu-zisa	Aproximativa	Ca putere a lui 2		
1		2^0	1	1
2		2^1	10	2
4		2^2	100	4
8		2^3	1000	8
16		2^4	1 000	10
32		2^5	10 0000	20
64		2^6	100 0000	40
128		2^7	1000 0000	80
256		2^8	1 0000 0000	100
512		2^9	10 0000 0000	200
1 024	1 K	2^{10}	100 0000 0000	400
2 048	2 K	2^{11}	1000 0000 0000	800
4 096	4 K	2^{12}	1 0000 0000 0000	1000
8 192	8 K	2^{13}	10 0000 0000 0000	2000
16 384	16 K	2^{14}	100 0000 0000 0000	4000
32 768	32 K	2^{15}	1000 0000 0000 0000	8000
65 536	64 K	2^{16}	1 0000 0000 0000 0000	10000
131 072	128 K	2^{17}	10 0000 0000 0000 0000	20000
262 144	256 K	2^{18}	100 0000 0000 0000 0000	40000
524 288	512 K	2^{19}	1000 0000 0000 0000 0000	80000
1 048 576	1 M	2^{20}	1 0000 0000 0000 0000 0000	100000

In orice baza *n, produsul cu o putere a bazei* prezinta un interes deosebit.

Astfel, *in baza 10* produsul unui numar intreg cu 10 inseamna adaugarea unui zero in partea dreapta. Acest lucru este echivalent cu deplasarea cu o cifra la stanga (cifra unitatilor devenea cifra zecilor, cifra zecilor devenea cifra sutelor, etc.).

Produsul cu 100 = 10^2 inseamna adaugare a 2 zerouri in partea dreapta sau deplasarea cu 2 cifre la stanga.

Produsul cu 10^n in baza 10 inseamna deci deplasarea cu *n* cifre (pozitii) la stanga.

Tot astfel, *in baza 2* produsul cu 2^n inseamna deplasarea cu *n* pozitii (biti) la stanga, iar *in baza 16* produsul cu 16^n inseamna deplasarea cu *n* pozitii (biti) la stanga.

Un caz special: *produsul cu 10 H = 10000 B = 16 =* 2^4 este echivalent *in baza 16* cu deplasarea cu *1 pozitie (cifra hexa)* catre stanga, iar *in baza 2* cu deplasarea cu *4 pozitii (biti)* catre stanga.

Exemplu:

(hexa-zecimal)		(binar)		(zecimal)
3A H x	=	0011 1010 B x	=	58 x
1**0** H=		1 **0000** B	=	16
3A**0** H	=	0011 1010 **0000** B	=	928

3. Reprezentarea datelor pe numar finit de biti

In interiorul sistemelor cu microprocesor datele memorate si transferate au in general 2^m biti. De aceea este necesara cunoasterea proprietatilor operatiilor efectuate cu date binare de lungime finita.

Presupunem ca datele utilizate au lungimea de $n = 2^m$ biti (in general n = 8, 16, 32, etc.). Atunci *gama valorilor naturale (intregi pozitive) care pot fi reprezentate cu n biti* este: 0 ..... 2^n - 1. Daca se efectueaza o operatie asupra a doua valori de *n* biti rezultatul exact poate depasi *n* biti (adunarea si scaderea a doua valori de *n* biti au ca rezultat valori de *n+1* biti iar inmultirea a doua valori de *n* biti are ca rezultat o valoare de *2n* biti).

Exemplul 1 (pentru *n = 8*):

```
                    1110 0010 B +        E 2 H +
                    1010 0001 B                A1 H
                    ------------------   ---------
(transport = 1) 1000 0011 B              (1) 8 3 H
```

Exemplul 2 (pentru *n = 16*):

```
                 1001 1100 1110 0010 B -          9 C E 2 H -
                 1011 0110 1010 0001 B            B 6 A1 H
                 -------------------------------  --------------
(imprumut = 1) 1110 0110 0100 0001 B            (-1) E 6 4 1 H
```

Pentru a semnaliza *depasirea de gama de reprezentare a valorilor naturale* la operatiile aritmetice se utilizeaza un bit indicator numit Carry Flag (*indicator de transport*). Acesta indica la adunare "transportul" sau "tinem 1", si indica la scadere "imprumutul" sau "ne imprumutam 1".

Transportul apare atunci cand la o operatie de adunare pe n biti valoarea exacta a rezultatului depaseste n biti, sau cand la o operatie de scadere pe n biti este necesar imprumutul de la al *n-1*-lea bit. Daca se neglijeaza transportul, operatiile aritmetice cu valori reprezentate pe n biti pot fi descrise grafic pe un cerc:

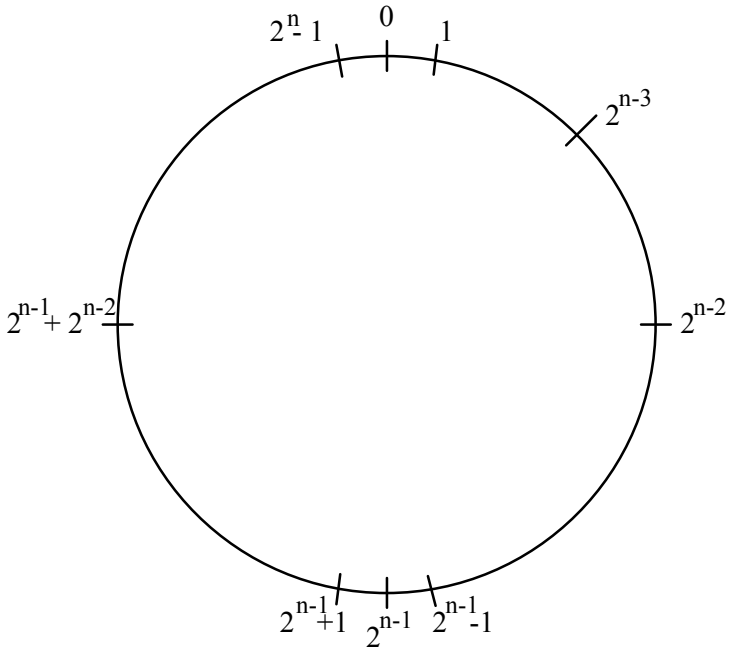

Fig. 1. Cercul valorilor naturale (intregi fara semn) reprezentabile cu n biti

Pentru exemplul 1 anterior (n = 8) acest cerc devine:

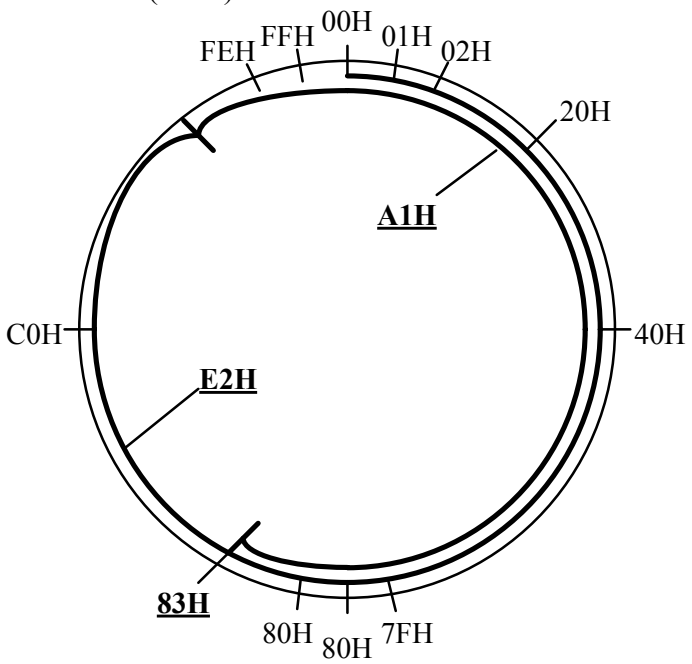

Fig. 2. Cercul valorilor naturale (intregi fara semn) reprezentabile cu 8 biti pentru exemplul 1
Pentru exemplul 2 anterior (n = 16) acest cerc devine:

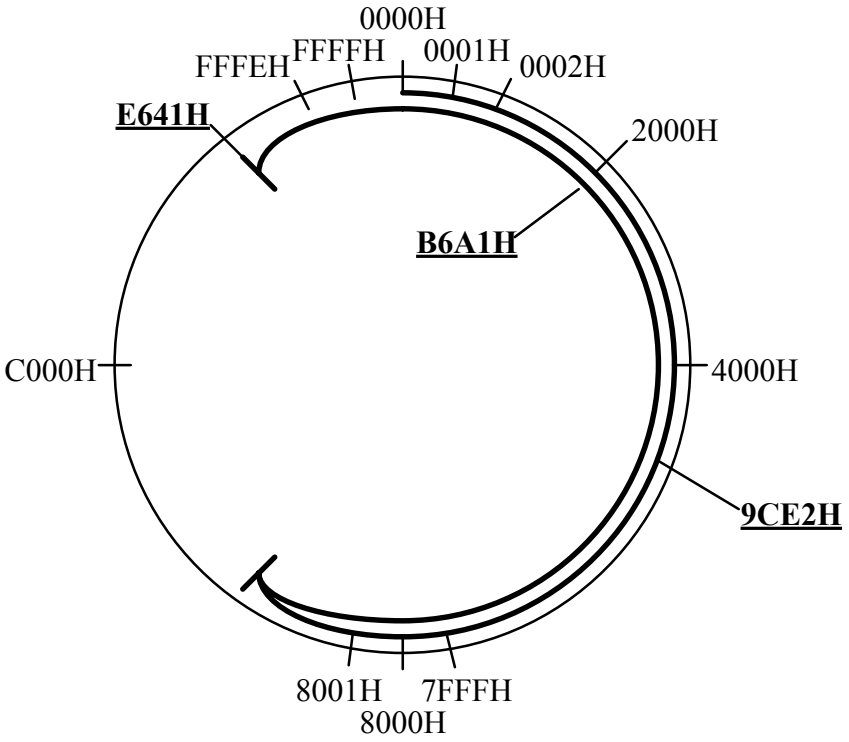

Fig. 3. Cercul valorilor naturale (intregi fara semn) reprezentabile cu 16 biti pentru exemplul 2

Se observa ca operatiile de *adunare pe n biti* sunt echivalente operatiilor de *adunare modulo* 2^n. Pe datele din exemplul 1 (n = 8, $2^n = 2^8$ = 256 = 100H = 1 0000 0000 B) se observa ca:

```
    E 2 H +                 2 2 6 +
    A1 H                    1 6 1
    ---------           ---------
(1) 8 3 H                   3 8 7 = 1 x 2 5 6 + 1 3 1
```

De asemenea, pe datele din exemplul 2 (n = 16, $2^n = 2^{16}$ = 65536 = 10000H = 1 0000 0000 0000 0000 B) se observa ca:

```
    9 C E 2 H -             4 0 1 6 2 -
    B 6 A1 H                4 6 7 5 3
    --------------          ------------
(-1) E 6 4 1 H            - 0 3 4 0 9 = - 1 x 6 5 5 3 6 + 6 2 1 2 7
```

Asadar, corespondenta dintre multimea numerelor naturale si multimea valorilor reprezentabile cu n biti este cea din figura 4.

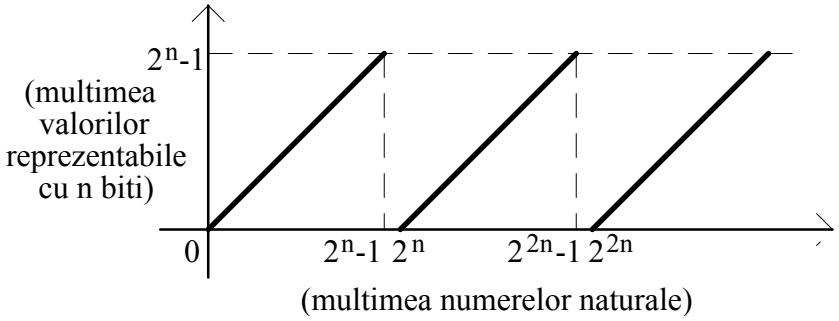

Fig. 4. Corespondenta numerelor naturale cu valorile reprezentabile cu n biti

4. Reprezentarea numerelor negative

Reprezentarea numerelor negative în binar se poate face in mai multe moduri.

Fie $(b_{n-1}\ b_{n-2} \ldots b_0 \,.\, b_{-1} \ldots b_{-m}) = (N)_2$

In mod obisnuit *bitul care indica semnul* este *cel mai semnificativ* bit (b_{n-1}), iar numerele pozitive au $b_{n-1} = 0$ şi numerele pozitive au $b_{n-1} = 1$.

4.1. Reprezentarea în modul şi semn (MS)

In acest caz *semnul numarului N* este **+** daca $b_{n-1} = 0$ si **-** daca $b_{n-1} = 1$, iar *modulul numărului N* în baza 10 este:

$$| N | = b_{n-2}\, 2^{n-2} + \ldots\ldots + b_1\, 2^1 + b_0\, 2^0 + b_{-1}\, 2^{-1} + b_{-2}\, 2^{-2} + \ldots\ldots + b_{-m}\, 2^{-m}$$

Exemplu:

$(9)_{C1}$ = 0000 1001 B = 9 H
$(-9)_{C1}$ = 1000 1001 B = 89 H

Gama dinamică a numerelor reprezentate în MS este +/- (2^{n-1} - 2^{-m}). Valoarea de modul maxim este atinsă pentru bi = 1, i = - m, n-2, adică:

$$2^{n-1} + \ldots\ldots + 2^1 + 2^0 + 2^{-1} + 2^{-2} + \ldots\ldots + 2^{-m} = 2^{n-1} - 2^{-m}$$

Pentru *numerele intregi* m = 0. Numerele întregi reprezentate în MS pe n biţi au *gama dinamică* +/- (2^{n-1} -1).

În multe cazuri domeniul +/- (2^{n-1} -1) se *scalează (normează)* cu 2^{n-1}, adică N' = N/ 2^{n-1}, deci *virgula va fi plasată imediat după bitul de semn.* In aceasta situatii se reprezintă pe n biţi *numere fracţionare cu semn. Gama dinamică* pentru reprezentarea MS a numerelor fracţionare este +/- ($1-1/2^{n-1}$).

4.2. Reprezentarea în complement de 1 (C1)

Reprezentarea *numerelor pozitive in C1* este *identica reprezentarii MS.*

Reprezentarea *unui numar negativ în C1* se obtine *prin complementarea bit cu bit a valorii modulului său.*

Exemplu:

$(9)_{C1}$ = 0000 1001 B = 9 H
$(-9)_{C1}$ = 1111 0110 B = F6 H

Se observa ca F6 H = 15 x 16 + 6 = 246 = 255 - 9 = $(2^n -1) - 9$. Rezulta o a doua regulă pentru obținerea reprezentării valorilor negative a unui număr întreg în C1: *scăderea din 2^n-1 a valorii sale absolute.*

Gama dinamică pentru reprezentarea C1 este +/- $(2^{n-1} - 2^{-m})$ (la fel ca în cazul reprezentării MS).

Dezavantajul reprezentărilor MS şi C1 este acela că *valoarea 0 se exprimă în 2 moduri diferite* (pentru MS valoarea 0 este 000...0 sau 100...0 iar pentru C1 valoarea 0 este 000...0 sau 111...1.

4.3. Reprezentarea în complement de 2 (C2)

Pentru *numere pozitive* reprezentarea este *identică cu cea din C1 şi MS.* Pentru *numere negative* $(-N)_{C2} = 2^n -N$.

Dacă numărul binar are reprezentarea în C2: b_{n-1}, b_{n-2}, ..., $b_0.b_{-1}...b_{-m}$, atunci valoarea sa în zecimal este dată de relația:

$$N = (-1)\, b_{n-1}\, 2^{n-1} + b_{n-2}\, 2^{n-2} + ... + b_1\, 2^1 + b_0\, 2^0 + b_{-1}\, 2^{-1} + b_{-2}\, 2^{-2} + ... + b_{-m}\, 2^{-m}$$

Gama dinamică pentru reprezentarea în C2 este: -2^{n-1} ..... $2^{n-1} - 2^{-m}$

Deoarece pentru N>=0 avem b_{n-1}=0, valoarea maximă se obține pentru b_i=1, i = -m, n-2:

$$N_{max} = 2^{n-2} + ... + 2^1 + 2^0 + 2^{-1} + 2^{-2} + ... + 2^{-m} = 2^{n-1} - 2^{-m},$$

iar pentru N<0 avem b_{n-1}=1 valoarea minimă se obține pentru b_i=0, i = -m,n-2:

$$N_{min} = -2^{n-1}$$

Prin *scalare* cu 2^{n-1} se obține reprezentarea în C2 a *numerelor fracționare.* Gama dinamică (pentru m=0) este: -1 ..... $1- 1/2^{n-1}$

Există două *moduri de obținere a reprezentării in C2* a lui -N:

1) Se adaugă 1 la reprezentarea binară în C1 alui (-N).

2) Se scade din 2^{n-1} valoarea zecimala lui N şi apoi se reprezintă în binar (ca număr fără semn).

Exemplu (pentru n = 8 şi m = 0):

a) *Din C1*:

$(\mathbf{103})_{C2}$ =	0110 0111 B	=	**67 H**	
$(\mathbf{-103})_{C2}$ =	1001 1000 B +	=	98 H +	
	1 B		1 H	
	---		---	
	1001 1001 B	=	**99 H**	

b) *Prin scadere*:

$(\mathbf{-103})_{C2}$ =	1 0000 0000 B -	=	100 H -	= 2^8 - 103
	0110 0111 B		67 H	
	---		---	
	1001 1001 B	=	**99 H**	= **153**

Asadar C2 pe 8 biti al valorii zecimale 103 este valoarea zecimala 153 = 256 - 103.

Daca se efectueaza o operatie asupra a doua valori reprezentate in C2 pe *n* biti *rezultatul exact poate depasi gama de reprezentare* pe *n* biti (adunarea a doua valori pozitive de *n* biti poate avea ca rezultat valori negative de *n* biti iar adunarea a doua valori negative de *n* biti poate avea ca rezultat valori pozitive de *n* biti).

Exemplul 1 (pentru *n = 8*):

	1110 0010 B +	E 2 H +	adica:	- 1 E H +
	1001 0001 B	9 1 H		- 6 F H
	---	---		---
(transport = 1)	0111 0011 B	7 3 H (depasire = 1)		- 8 D H

Pentru interpretarea corecta a rezultatului trebuie semnalata asadar depasirea de gama. Acest lucru se face printr-un bit indicator numit Overflow Flag (*indicator de depasire*). Rezultatul obtinut, 73 H, se interpreteaza ca -8D H, valoare care nu poate fi reprezentata pe 8 biti in C2.

Exemplul 2 (pentru *n = 16*):

	1001 1100 1110 0010 B +	9 C E 2 H +	adica:	- 6 3 1 E H +
	1011 0110 1010 0001 B	B 6 A1 H		- 4 9 5 F H
	---	---		---
(imprumut = 1)	1110 0110 0100 0001 B	5 3 8 3 H (depasire = 1)		+ 5 3 8 3 H

Reprezentarea in C2 a valorilor intregi de n biti poate fi descrisa grafic ca in figura 5.

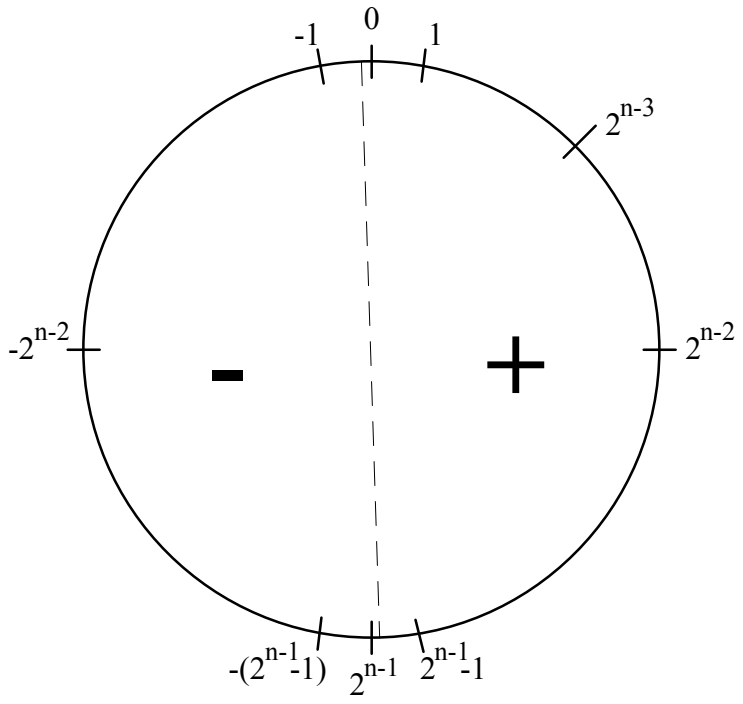

Fig. 5. Cercul valorilor intregi (cu semn) reprezentabile cu n biti

Pentru exemplul 1 anterior (n = 8) acest cerc devine:

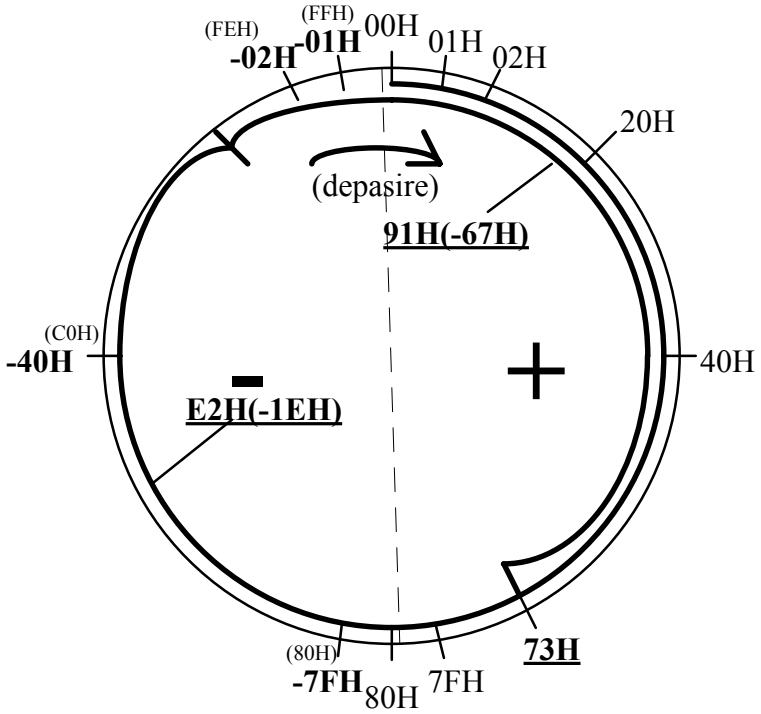

Fig. 6. Cercul valorilor intregi (cu semn) reprezentabile cu 8 biti pentru exemplul 1

Pentru exemplul 2 anterior (n = 16) acest cerc devine:

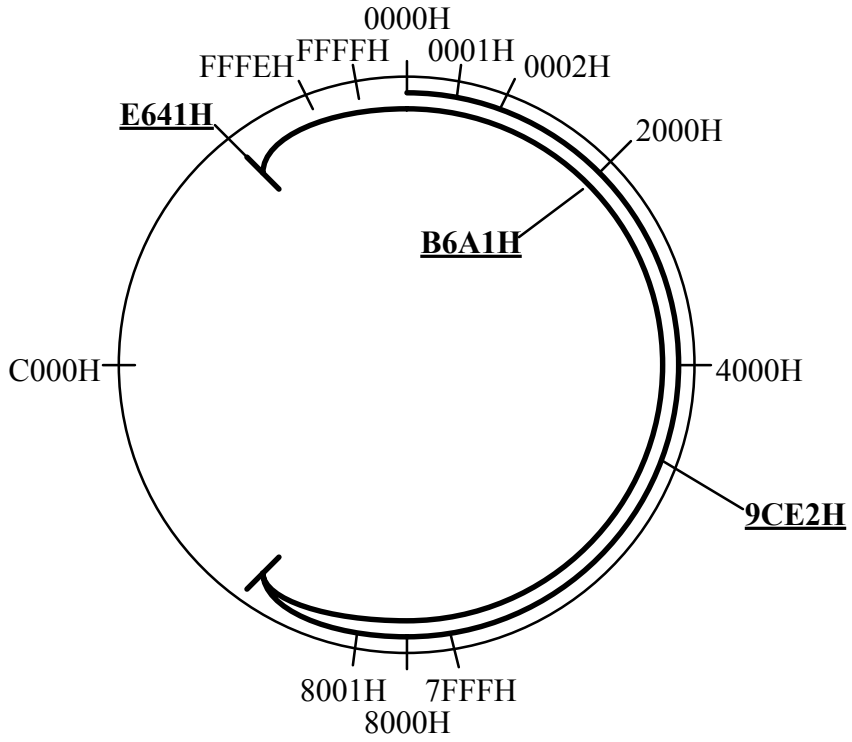

Fig. 7. Cercul valorilor intregi (cu semn) reprezentabile cu 16 biti pentru exemplul 2

4.4. Echivalenta valorilor binare in diferite moduri de reprezentare

Microprocesorul Intel 8086 utilizeaza date de tip octet (8 biti) si cuvant (16 biti). De aceea este necesara cunoasterea echivalentei valorilor fara semn (naturale) reprezentabile cu 8 si 16 biti cu valorile cu semn (intregi).

Echivalentele pentru date de tip octet sunt date in tabelul 3, iar cele pentru date de tip cuvant in tabelul 4.

Tab.3. Echivalenta valorilor de 8 biti fara semn (naturale) cu cele cu semn (intregi) in diferite moduri de reprezentare (MS, C1, C2)

Valoarea fara semn a numarului in			Valoarea codului binar obținută prin interpretarea ca număr cu semn în		
zecimal	binar	hexazecimal	(MS)	(C1)	(C2)
0	0000 0000	0 0	0	0	0
1	0000 0001	0 1	1	1	1
2	0000 0010	0 2	2	2	2
3	0000 0011	0 3	3	3	3
4	0000 0100	0 4	4	4	4
5	0000 0101	0 5	5	5	5
6	0000 0110	0 6	6	6	6
...	...	...	...	...	...
...	...	...	...	...	...
63	0011 1111	3 F	63	63	63
64	0100 0000	4 0	64	64	64
65	0100 0001	4 1	65	65	65
...	...	...	...	...	...
...	...	...	...	...	...
127	0111 1111	7 F	127	127	127
128	1000 0000	8 0	0	- 127	- 128
129	1000 0001	8 1	1	- 126	- 127
...	...	...	...	...	...
...	...	...	...	...	...
191	1011 1111	B F	-63	-64	-65
192	1100 0000	C 0	-64	-63	-64
193	1100 0001	C 1	-65	-62	-63
...	...	...	...	...	...
...	...	...	...	...	...
250	1111 1110	F A	-122	-5	-6
251	1111 1110	F B	-123	-4	-5
252	1111 1100	F C	-124	-3	-4
253	1111 1101	F D	-125	-2	-3
254	1111 1110	F E	-126	-1	-2
255	1111 1111	F F	-127	0	-1

Tab.4. Echivalenta valorilor de 16 biti fara semn (naturale) cu cele cu semn (intregi) in diferite moduri de reprezentare (MS, C1, C2)

Valoarea fara semn a numarului in			Valoarea codului binar obținută prin interpretarea ca număr cu semn în		
zecimal	binar	hexa	(MS)	(C1)	(C2)
0	0000 0000 0000 0000	0 0 0 0	0	0	0
1	0000 0000 0000 0001	0 0 0 1	1	1	1
2	0000 0000 0000 0010	0 0 0 2	2	2	2
3	0000 0000 0000 0011	0 0 0 3	3	3	3
4	0000 0000 0000 0100	0 0 0 4	4	4	4
5	0000 0000 0000 0101	0 0 0 5	5	5	5
6	0000 0000 0000 0110	0 0 0 6	6	6	6
...	...	...	...	...	...
...	...	...	...	...	...
16383	0011 1111 1111 1111	3 F F F	16383	16383	16383
16384	0100 0000 0000 0000	4 0 0 0	16384	16384	16384
16385	0100 0000 0000 0001	4 0 0 1	16385	16385	16385
...	...	...	...	...	...
...	...	...	...	...	...
32767	0111 1111 1111 1111	7 F F F	32767	32767	32767
32768	1000 0000 0000 0000	8 0 0 0	0	- 32767	-32768
32769	1000 0000 0000 0001	8 0 0 1	- 1	- 32766	-32767
...	...	...	...	...	...
...	...	...	...	...	...
49151	1011 1111 1111 1111	B F F F	- 16383	- 16384	- 16385
49152	1100 0000 0000 0000	C 0 0 0	- 16384	- 16383	- 16384
49153	1100 0000 0000 0001	C 0 0 1	- 16385	- 16382	- 16383
...	...	...	...	...	...
...	...	...	...	...	...
65531	1111 1111 1111 1110	F F F A	-122	-5	-6
65532	1111 1111 1111 1110	F F F B	-123	-4	-5
65533	1111 1111 1111 1110	F F F C	-124	-3	-4
65534	1111 1111 1111 1110	F F F D	-125	-2	-3
65535	1111 1111 1111 1110	F F F E	-126	-1	-2
65536	1111 1111 1111 1111	F F F F	-127	0	-1