1.10.1 БРОЈНИ СИСТЕМИ

Приликом записивања децималног броја користи се позициони бројни систем, где вредност броја зависи од позиције и вредности сваке цифре у броју. У случају децималног броја, свака цифра се множи степеном од 10 у зависности од позиције у броју, као што је представљено у доњем примеру:

$986 = 9 \times 10^2 + 8 \times 10^1 + 6 \times 10^0 = 9 \times 100 + 8 \times 10 + 6 \times 1 = 900 + 80 + 6.$

За целе бројеве, крајња десна позиција цифре представља позицију јединице ($10^0 = 1$). Број у тој позицији показује колико јединица је присутно у броју. Следећа позиција на лево је позиција десетице а затим следи позиција стотине, хиљаде и тако даље. Свака позиција цифре има тежину која је десет пута већа од позиције са десне стране. У децималном бројном систему, постоји десет могућих вредности које се могу појавити у свакој позицији броја, тако да постоји десет бројева који су неопходни за представљање количине у свакој позицији цифре.

У позиционом бројном систему, основа броја се назива корен, тако да је корен или основа децималног бројног система 10.

Када корен није јасан из контекста, обично се користи корен коришћењем индекса. Уколико корен није разумљив, децимални бројеви могу да се напишу у облику као:

568_{10}, 121_{10}, 8873_{10}.

Бинарни бројни систем је исто тако позициони бројни систем, али у овом случају, основа није десет, него је два. Када се пише бинарни број, свака бинарна цифра се множи са одговарајућим степеном од два који је заснован на позицији у броју, као што је илустровано доњим примером:

$1101001 = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$= 1 \times 64 + 1 \times 32 + 0 \times 16 + 1 \times 8 + 0 \times 4 + 0 \times 2 + 1 \times 1$

$= 64 + 32 + 8 + 1 = 105.$

У бинарном бројном систему, постоје само две могуће вредности које се могу појавити у свакој бројној позицији, пре него десетица која може да се нађе у децималном броју. Само цифре 0 и 1 се користе у бинарним бројевима. Израз *"bit"* се користи као скраћеница речи "*binary*" и "*digit*", и када се говори о бинарним бројевима изрази *bit* и *digit* се односе на број бита који се користи за чување или представљање броја. То објашњава број бинарних цифара који су неопходни за записивање броја.

Конверзија из бинарног у децимални број се може приказати на следећи начин, као што је дато у примеру претварања 11000_2 у децимални број:

1 0 1 0

____________ $1 \times 2^1 = 2$

_____________ $1 \times 2^3 = 8$

10

Метод претварања децималног броја у бинарни број се може користити за претварање из децималне у било коју бројну основу. Процес укључује коришћење непрекидног дељења корена све док се не достигне вредност 0. Приликом сваког дељења остатак обезбеђује цифра претвореног броја, почев од најмање значајне цифре.

Конвертовање броја 97_{10} у бинарни се одвија на следећи начин:

97 / 2 = 48 остатак 1 (најмање значајна цифра)

48 / 2 = 24 остатак 0

24 / 2 = 12 остатак 0

12 / 2 = 6 остатак 0

6 / 2 = 3 остатак 0

3 / 2 = 1 остатак 1

1 / 2 = 0 остатак 1 (најзначајнија цифра).

Резултујући бинарни број је: **1100001**.

Следећи пример обухвата: конвертовање броја 43_{10} у бинарни број.

43 / 2 = 21 остатак 1 (најмање значајна цифра)

21 / 2 = 10 остатак 1

10 / 2 = 5 остатак 0

5 / 2 = 2 остатак 1

2 / 2 = 1 остатак 0

1 / 2 = 0 остатак 1 (најзначајнија цифра)

Резултујући бинарни број је: **101011**.

Поред бинарног бројног система доста се користи и хексадецимални бројни систем чија је основа 16, међутим за било коју бројну основу већу од десет, јавља се проблем јер постоји више од неопходних десет симбола за представљање бројева за ту бројну основу. Хексадецимални бројеви су:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Основни разлог за коришћење хексадецималних бројева је веза између бројева 2 и 16. Број шеснаест је степен од броја 2 (16 = 24). Због ове везе, четири броја могу се представити са једном хексадецималном цифром. На тај начин је претварање између бинарних и хексадецималних бројева веома лако, и хексадецимални бројеви се могу користити за писање великих бинарних бројева са много мање цифара.

За претварање бинарног броја у хексадецимални, потребно је дељење у групе од четири цифре почев од цифре на крајњој десној страни. Затим за сваку групу од четири цифре, потребно је претворити 4-битни бинарни број у еквивалентну хексадецималну цифру.

Пример претварања бинарног броја 10010101 у хексадецимални број, се одвија у неколико фаза:

- претварање у групе од 4 цифре 1101 0111;

- претварање сваке групе у хексадецималну цифру D7;

- $D7_{16.}$

Конвертовање бинарног броја 0111101111110110 у хексадецимални одвија се у следећим фазама:

- поделити број у групе од четири као што је 0111 1011 1111 0110;

- конвертовати сваку групу у хексадецимални број 7 B F 6;

- добија се финално број 7BF6.

За претварање хексадецималног броја у бинарни број, потребно је конвертовати сваку хексадецималну цифру у групу од 4 бинарне цифре.

Претворање хексадецималног броја 374F у бинарни број, одвија се у неколико фаза:

- полази се од броја 3 7 4 F;

- конвертују се хексадецималне цифре у бинарне 0011 0111 0100 1111;

- добија се број 0011011101001111_2.

Постоји неколико начина да се одреди да је дати број хексадецималан. У доста писаног материјала где није јасно шта је корен, користи се индекс 16 пратећи хексадецимални број који се користи. У већини програмских језика, овај метод није применљив, тако да се користи неколико конвенција, као у језицима *C* и *C++*, где су хексадецималне константе представљене са “0x” пре самог броја, као што је: 0x317F, или 0x1234, или 0xAF. У асемблерским програмским језицима, хексадецимална константа почиње бројним карактером тако да се почетна нула “0” користи ако је неопходно. Слово “*h*” се користи као суфикс да би асемблер био упознат да се ради о хексадецималној константи. Асемблерски формати: *371Fh* и *0FABCh* су ваљане хексадецималне константе, док A37h није ваљана хексадецимална константа јер не почиње бројним карактером, и асемблер ће је третирати као променљиву.

У асемблерским програмским језицима који користе *Motorola* стил, хексадецимална константа почиње карактером ‘$’, тако да су $371F или $FABC или $01 ваљане хексадецималне константе.

1.10.2 BCD КОД

Код *BCD* кода (*Binary Coded Decimal*), бројеви су представљени у децималној форми, а свака децимална цифра је енкодована коришћењем бинарног броја од четири бита.

Децимални број 136 би био представљен у *BCD*-у на следећи начин:

136 = 0001 0011 0110.

Претварање бројева између децималног бројног система и *BCD* кода је прилично једноставно. Да би претворили број из децималног у *BCD* код, потребно је написати по шаблону четири бита за сваку децималну цифру. За претварање броја из *BCD* кода у децимални, потребно је поделити број у групе од 4 бита и написати одговарајућу децималну цифру за сваку групу од четири бита.

У примеру је дат 16-битни енкодовани број упакован у *BCD* формат:

01010110 10010011.

Овај број се конвертује у децимални број на следећи начин:

0101 0110 1001 0011

5 6 9 3.

Децимална вредност броја износи 5693.

Уколико се исти број не налази у коду BCD (захтева 32 бита) може се представити услеедећем облику:

00000101 00000110 00001001 00000011

5 6 9 3.

Приликом проучавања бинарних бројева, није било говора о максималној величини броја. У већини рачунарских система бројеви су представљени коришћењем фиксног броја бита чија величина је обично 8, 16, 32, 64 или 80. Ова величина се онда множи са 8, јер је већина компјутерских меморија организована на принципу 8 бита. Бројеви код којих се користи одговарајући број бита за представљање вредности зову се бројеви са фиксном прецизношћу.

Бинарни бројеви који се користе за задржавање позитивних вредности називају се непотписани. У том случају распон позитивних вредности се може представити као *0 -- 2n-1*, где је *n* број бита који се користи. Исто тако је могуће представити потписане (негативне као и позитивне) бројеве као бинарне. У овом случају, користи се део тоталних вредности да би представио позитивне вредности, а остали део опсега представља негативне вредности.

Постоји неколико начина да се потписани бројеви бинарно представе, али најчешће представљање назива се комплемент двојке. Комплемент двојке се добија када се броју сви битови инвертују а затим се дода јединица. Прво се израчунава први комплемент броја тако што се свака бинарна цифра полазног броја промени, нуле постају јединице и обрнуто, а затим се добијеном бинарном броју дода јединица. Сабирање бинарних бројева остварује се на исти начин као и децималних, а пренос на више место се догађа када збир премаши 1 а не 9 као што је то у децималном систему.

У случају негативног броја, други комплемент се добија тако што се све нуле и прва јединица са десне стране препишу, док се преостале цифре инвертују. У комплементу јединице представљају се бројеви из интервала $-(2^{N-1}-1)$ do $+(2^{N-1}-1)$, док се у комплементу двојке може представити било који целобројни означени број из интервала $-(2^{N-1})$ до $+(2^{N-1}-1)$.

Базичне предности представљања броја у комплементу двојке је да су основне рачунске операције исте као и код неозначених целобројних бројева а улазни подаци су записани у истом броју бита и свако прекорачење преко тог броја битова се искључује из решења.

Други комплемент броја 0010100, одвија се у следећим фазама:

-11010110 прва фаза обухвата инвертовање броја,

+ 00000001 затим додавање јединице,

= 11010111 финално добијени број.

Други комплемент броја 00101001 је 11010111.

Други комплемент броја 10110101, одвија се у следећим фазама:

01001010 прва фаза обухвата инвертовање броја,

+ 00000001 затим додавање јединице,

= 01001011 финално добијени број.

Други комплемент броја 10110101 је 01001011.

Бројна секвенца за осмобитну бинарну вредност коришћењем другог комплемента појављује се у следећем облику:

01111111 7Fh 127 највећи интензитет позитивног броја

01111110 7Eh 126

01111101 7Dh 125

…

00000011 03h

00000010 02h

00000001 01h

00000000 00h

11111111 0FFh -1

11111110 0FEh -2

11111101 0FDh -3

…

10000010 82h -126

10000001 81h -127

10000000 80h -128 највећи интензитет негативног броја.

Степени броја 2:

2^0 1

2^1 2

2^2 4

2^3 8

2^4 16

2^5 32

2^6 64

2^7 128

2^8 256

2^9 512

2^{10} 1024

2^{11} 2048

2^{12} 4096

2^{13} 8192

2^{14} 16384

2^{15} 32768

2^{16} 65536

Хексадецимални бројеви

0 ....0 0000

1 ....1 0001

2 ....2 0010

3 ....3 0011

4 ....4 0100

5 ....5 0101

6 ....6 0110

7 ....7 0111

8 ....8 1000

9 ....9 1001

10 .... A 1010

11 .... B 1011

12 .... C 1100

13 .... D 1101

14 .... E 1110

15 ....F 1111

BCD бројеви

0 0000

1 0001

2 0010

3 0011

4 0100

5 0101

6 0110

7 0111

8 1000

9 1001

Еквивалентни бројеви у децималном, бинарном и хексадецималном бројном систему:

Децимални	**Бинарни**	**Хексадецимални**
0	00000000	00
1	00000001	01
2	00000010	02
3	00000011	03
4	00000100	04
5	00000101	05
6	00000110	06
7	00000111	07
8	00001000	08
9	00001001	09
10	00001010	0A
11	00001011	0B
12	00001100	0C
13	00001101	0D
14	00001110	0E
15	00001111	0F

16	00010000	10
17	00010001	11
31	00011111	1F
32	00100000	20
63	00111111	3F
64	01000000	40
65	01000001	41
127	01111111	7F
128	10000000	80
129	10000001	81
255	11111111	FF
256	0000000100000000	0100
32767	0111111111111111	7FFF
32768	1000000000000000	8000
65535	1111111111111111	FFFF

ASCII Character Set

Low Order Bits		High Order Bits							
		0000 **0**	0001 **1**	0010 **2**	0011 **3**	0100 **4**	0101 **5**	0110 **6**	0111 **7**
0000	**0**	NUL	DLE	Space	0	@	P	`	p
0001	**1**	SOH	DC1	!	1	A	Q	a	q
0010	**2**	STX	DC2	"	2	B	R	b	r
0011	**3**	ETX	DC3	#	3	C	S	c	s
0100	**4**	EOT	DC4	$	4	D	T	d	t
0101	**5**	ENQ	NAK	%	5	E	U	e	u
0110	**6**	ACK	SYN	&	6	F	V	f	v
0111	**7**	BEL	ETB	'	7	G	W	g	w
1000	**8**	BS	CAN	(	8	H	X	h	x
1001	**9**	HT	EM	)	9	I	Y	i	y
1010	**A**	LF	SUB	*	:	J	Z	j	z
1011	**B**	VT	ESC	+	;	K	[	k	{
1100	**C**	FF	FS	,	<	L	\	l	|
1101	**D**	CR	GS	-	=	M	]	m	}
1110	**E**	SO	RS	.	>	N	^	n	~
1111	**F**	SI	US	/	?	O	_	o	DEL

Слика 1.80 Приказ ASCII скупа карактера