Microprocessors

Tuba Ayhan

MEF University

Number systems and codes

Ref. K.J. Breeding, «Digital Design Fundamentals»

Microprocessors

Tuba Ayhan

MEF University

Base conversion

Ref. K.J. Breeding, «Digital Design Fundamentals»

Radix *r* to decimal conversion

- $2x10^2 + 7x10^1 + 6x10^0 + 5x10^{-1} = 267.5$
- Radix *r* to decimal conversion:

$$A_r = (a_n\ a_{n\text{-}1}\ \ldots..a_0.a_{-1}\ldots.a_{-m})_r = \sum_{i=-m}^{n} a_i r^i$$

Ex. $(364.213)_7 =$

$$3x7^2 + 6x7^1 + 4x7^0 + 2x7^{-1} + 1x7^{-2} + 3x7^{-3} = (193.314868)_{10}$$

Decimal to radix *r* conversion

- Convert B_{10} to a number A_r radix *r*.
- Integer part: B_{10} will be divided by r repeately, until no integral part remains and the remainders obtained after each division will constitute the digits of A_r.

$$B_{10} = A_r = (a_n \ a_{n-1} \ \ldots\ldots a_0)_r = a_n r^n + a_{n-1} r^{n-1} + \ldots + a_0$$

$$\text{Int.}\left(\frac{B_{10}}{r}\right) + \text{Frac.}\left(\frac{B_{10}}{r}\right) = \frac{B_{10}}{r}$$

Decimal to radix *r* conversion

For B_{10}=278 and r = 3:

$$\frac{B_{10}}{r} = \left(a_n r^{n-1} + a_{n-1} r^{n-2} + \ldots + a_1\right) + \frac{a_0}{r} = \frac{278}{3} = 92 + \frac{2}{3}$$ → $a_0 = 2$

$$\frac{92}{r} = \left(a_n r^{n-2} + a_{n-1} r^{n-3} + \ldots + a_2\right) + \frac{a_1}{r} = \frac{92}{3} = 30 + \frac{2}{3}$$ → $a_1 = 2$

$$\frac{30}{r} = \left(a_n r^{n-3} + a_{n-1} r^{n-4} + \ldots + a_3\right) + \frac{a_2}{r} = \frac{30}{3} = 10 + \frac{0}{3}$$ → $a_2 = 0$

$$\frac{10}{r} = \left(a_n r^{n-4} + a_{n-1} r^{n-5} + \ldots + a_4\right) + \frac{a_3}{r} = \frac{10}{3} = 3 + \frac{1}{3}$$ → $a_3 = 1$

$$\frac{3}{r} = \left(a_n r^{n-5} + a_{n-1} r^{n-6} + \ldots + a_5\right) + \frac{a_4}{r} = \frac{3}{3} = 1 + \frac{0}{3}$$ → $a_4 = 0$

$$\frac{1}{r} = \left(a_n r^{n-6} + a_{n-1} r^{n-7} + \ldots + a_6\right) + \frac{a_5}{r} = \frac{1}{3} = 0 + \frac{1}{3}$$ → $a_5 = 1$

$(101022)_3 = (278)_{10}$

Decimal to radix *r* conversion

- Conversion of the fractional parts of a decimal number to an equivalent radix r representation:

$$B_{10} = A_r = (0.a_{-1}a_{-2} \ldots\ldots a_{-m})_r = a_{-1}r^{-1} + a_{-2}r^{-2} + \ldots + a_{-m}r^{-m}$$

- Multiplying the result by *r* yields,

$$rB_{10} = a_{-1} + \left(a_{-2}r^{-1} + \ldots + a_{-m}r^{-m+1}\right)$$

from which the integral part becomes a_{-1}.

Repeated multiplication by r yields the succesive digits of the radix r representation of the fractional number B_{10}.

Decimal to radix *r* conversion

- $(0.35)_{10} = (?)_4$

		Integer	Fraction
$0.35 \times 4 = 1.40$	$\rightarrow$	$a_{-1} = 1$	.40
$0.40 \times 4 = 1.60$	$\rightarrow$	$a_{-2} = 1$	.60
$0.60 \times 4 = 2.40$	$\rightarrow$	$a_{-3} = 2$	.40
$0.40 \times 4 = 1.60$	$\rightarrow$	$a_{-4} = 1$	.60
$0.60 \times 4 = 2.40$	$\rightarrow$	$a_{-5} = 2$	.40

$(0.35)_{10} = (0.11212....)_4$, $(0.11212....)_4 = (0.3496....)_{10}$ $\leftarrow$ not exact equivalent, as usually the case

Decimal to binary conversion

- $(65)_{10} = (?)_2$

$65 = 64 + 1 = 2^6 + 2^0 \rightarrow (65)_{10} = (1000001)_2$

Binary to octal / hexadecimal conversion

$$100101011 = 1\text{x}2^8 + 0\text{x}2^7 + 0\text{x}2^6 + 1\text{x}2^5 + 0\text{x}2^4 + 1\text{x}2^3 + 0\text{x}2^2 + 1\text{x}2^1 + 1\text{x}2^0$$

$$= (1\text{x}2^2+0\text{x}2^1+0\text{x}2^0)2^6+(1\text{x}2^2+ 0\text{x}2^1+ 1\text{x}2^0)2^3 + (0\text{x}2^2+1\text{x}2^1+1\text{x}2^0)2^0$$

$$= 4\text{x}(2^3)^2 + 5\text{x}(2^3)^1 + 3\text{x}(2^3)^0$$

$$= 4\text{x}(8)^2 + 5\text{x}(8)^1 + 3\text{x}(8)^0$$

$$= (453)_8$$

Easier way:

$(\underline{100}\ \underline{101}\ \underline{011})_2$

$(\ 4\quad 5\quad 3\ )_8$

Binary to hexadecimal:

$(\underline{1}\ \underline{0010}\ \underline{1011})_2$

$(\ 1\quad 2\quad B\ )_{16}$

Microprocessors

Tuba Ayhan

MEF University

Binary Arithmetics 1

Ref. K.J. Breeding, «Digital Design Fundamentals»

Binary addition

```
          111        → caries from the preceeding bit positions
A  =   10011100
B  =+    110110
   ____________
       11010010
```

Binary multiplication

```
A  =     101100    ← Multiplicand
B  = x     1011    ← Multiplier
   ___________
         101100
        101100
       000000       Shifted Multiplicands
 +    101100
   ___________
      111100100    ← Product
```

Binary subtraction

		0 1 1 1	→	resulting bits after borrow
A	=	1 0 0 0 0	←	Minuend
B	=	- 1 0 1	←	Subtrahend
		1 0 1 1	←	Difference

Microprocessors

Tuba Ayhan

MEF University

Binary arithmetics 2

Ref. K.J. Breeding, «Digital Design Fundamentals»

Radix and diminished radix complements

- A = an n digit number in radix r representation
- The radix complement of A: $A^{*}=r^{n}-A$
- The diminished radix complement of A: $A^{+}=r^{n}-A-1$
- Using complement in subtraction:

$$A^{*}+B=r^{n}-A+B$$

$$A^{*}+B=r^{n}+(B-A)$$

!! r^{n} in radix r is just 1 followed by n zeros.

Binary subtraction using complement

A = 110101, B = 111001 B-A = ?

A* = 1000000-110101 → how can you compute easily?

$A^* = A^+ + 1$, $A^+ = 111111 - 110101$

A* = (111111-110101) + 1

A* = (001010) +1

A* = 001011

$A^* + B = 001011 + 111001 = 1\ \overbrace{000100}^{B-A}$

Note: to find 1's complement, just interchange 1s and 0s. Then, add 1 to find 2's complement.

Question: did we omit the first digit, why?

Binary subtraction using complement

MEF UNIVERSITY
Your Freedom in Learning

$A = 10111 \rightarrow (23)_{10}$

$B = 10011 \rightarrow (19)_{10}$ $\quad$ $A>B$

$A^* = 01001$

$A^*+B = 11100 \rightarrow (28)_{10}$

In this case, A >B and 11100 is 2's complement of (A-B)

$A\text{-}B = 00100 \rightarrow (4)_{10}$

$B\text{-}A = -00100 \rightarrow (-4)_{10}$

The operation, (A*+B) yields no carry, if A>B. The sign of (B-A) is (-). Its magnitude is obtained by calculating 2's complement of (A*+B)

Microprocessors

Tuba Ayhan

MEF University

Binary arithmetics 3

Ref. K.J. Breeding, «Digital Design Fundamentals»

Representation of binary numbers

Unsigned numbers		Sign magnitude			Signed 2's complement	
			S	M		
0	0000	**-7**	1	111	**-8**	1000
1	0001	-6	1	110	-7	1001
2	0010	…			…	
3	0011				-2	1110
4	0100	-1	1	001	-1	1111
5	0101	-0	1	000	0	0000
…		0	0	000	1	0001
		1	0	001	2	0010
		2	0	010		
		….			…	
15	1111	**7**	0	111	**7**	0111

Exercise:
Fill the table

Signed 2’s complement representation

- A and B are 8-bit signed two’s complement form.

A = 00111100 (60)

B = 00100100 (36)

A+B = 01100000 (96)

A-B = 00111100+11011100=1 00011000 (24)

-A+B = 11000100+00100100=11101000 (-24)

-A-B = 11000100+11011100=1 10100000 (-96)

- The addition of two n-bit numbers can result in a number whose value requires more than n bits to represent. Such situation is referred to as **overflow** if the result is positive and **underflow** if the result is negative.

Overflow and underflow

- In a signed 2's complement representation, overflow or underflow occurs whenever the sign of the two arguments is the same but different from the sign of the result.

A	=	01100001	(97)
B	= +	00100011	(35)
		10000100	(132)

A	=	10110110	(-74)
B	= +	10000001	(-127)
		1 00110111	(-201)

Since the sum of 97 and 35 is greater than 127, an overflow occurs.

Since the sum of -74 and -127 is smaller than -127, an underflow occurs.

BCD addition

A =		0001 0110 0101	165
B =	+	1000 0011 0010	832
		1001 1001 0111	997

If the sum of the two digits exceeds 9

a. The resulting 4 bits is not a legal BCD code, or
b. Carry occurs out of the 4-bit group

Adding 6 to the wrong result will yield the correct answer.

BCD addition

```
    0101      5
+   1000      8
  ___________
    1101      Not a legal BCD number

     1001      9
+    1000      8
  ___________
    1  0001      carry occurs
```

```
     1101      Answer
+    0110        6
   ___________
  1 0011    13 → correct

  1 0001      Answer
+    0110        6
   ___________
  1 0111    17 → correct
```

Fixed point representation

f: the number of fractional bits

m: the number of magnitude or integer bits.

- **Q***f*: The "Q" prefix is used for fixed point number.
  - Ex: Q7 → a number with 7 fractional bits.
  - Ambiguous notation, if the total word length is not known.
- **Q***m***.***f*:
  - Ex: Q1.30 1 integer bit and 30 fractional bits stored as a 32-bit 2's complement integer.

Floating-Point Numbers

- 32-bit word length computer / signed integer in 2's-complement representation:
  - the range is -2^{31} to$+2^{31-1}$. (in decimal ~-10^{10} to+ 10^{10})
  - the range is -1 to$+1-2^{-31}$ (in decimal ~1 to 10^{-10})
- But, we need both very large integers and very small fractions.
- The binary point float, and the numbers are called floating-point numbers. Example:
- 6.0247×10^{23} $3\ .7291\times10^{-27}$ -1.0341×10^{2} -7.3000×10^{-14},
- 5 significant digits of precision. The scale factors: 10^{23},10^{-27}, 10^{2}, and 10^{-14}

Character Representation

- ASCII (American Standard Code for Information Interchange).
    - Alphanumeric characters, operators, punctuation symbols, and control characters are represented by ***7-bit codes***
- 8-bit byte: [0 – ***7-bit code***]
- 4-bit encoding of numbers is called binary-coded decimal (BCD) code.

Table 1.1 The 7-bit ASCII code.

Bit positions	Bit positions 654							
3210	000	001	010	011	100	101	110	111
0000	NUL	DLE	SPACE	0	@	P	´	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	/	1	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	—	o	DEL