MULTIMEDIA UNIVERSITY

PDS0101 Introduction to Digital Systems

Tutorial 2

Tutorial outcomes

By the end of today's lab, you should be able to

- apply arithmetic operations to binary, hex and octal numbers
- convert between binary, hex and octal numbers
- expressed signed binary numbers in in signed- magnitude, 1's and 2's complement

Theory based questions

BINARY NUMBERS

Perform the conversions from binary to decimal number or vice versa

a) 1011_2
b) 110101101_2
c) 0.1101_2
d) 0.00111_2
e) 101101.101_2
f) 10111.1101_2
g) 24_{10}
h) 15_{10}
i) 0.246_{10}
j) 0.0981_{10}
k) 56.625_{10}
l) 110.75_{10}

What is the highest decimal number that can be represented by each of the following number of bits

a) three
b) four
c) seven
d) eight

What is the minimum number of bits required to represent the following decimal numbers?

a) 17
b) 35
c) 205
d) 132

Perform the following arithmetic operations on binary numbers (unsigned)

a) 101+11
b) 1001+101
c) 1100-1001
d) 110-101
e) 11×11
f) 1001×110
g) 111×101
h) 1001÷11
i) 1100÷100

Signed binary numbers

There are several ways to represent signed binary numbers. Usually the MSB in a signed number is recognised as the sign bit and tells you if the number is positive or negative. Computers (and nearly all digital systems) use a *modified* 2's complement for signed numbers. Positive numbers are stored in *true* form (with a 0 for the sign bit) and negative numbers are stored in *complement* form (with a 1 for the sign bit) → i.e. the sign bit does not flip with the complement

Note: The examples in lectures show 2's complement for UNsigned numbers and is for reference to show concept of obtaining the 1's and 2's complement of binary numbers.

Express each decimal number below in binary as an 8-bit *sign-magnitude number* as well as 1's and 2's complement form for use in arithmetic operations

	Signed binary	1's complement	2's complement
a) -80			
b) +80			
c) -123			
d) -34			
e) +101			
f) -125			
g) +60			

Signed binary addition/subtraction

Basic rules of addition of signed binary numbers are the same as unsigned numbers but if any of the values are negative (i.e. subtraction or negative value), the 2's complement is used instead and/or the subtraction becomes addition instead. If the result of the operation is negative, the reverse process of obtaining 2's complement is done to obtain the result.

The rules for detecting overflow in a two's complement sum are simple:

If the sum of two positive numbers yields a negative result, the sum has overflowed.

If the sum of two negative numbers yields a positive result, the sum has overflowed.

Otherwise, the sum has not overflowed.

Perform the following arithmetic functions using ***signed*** 8-bit 2's complement form of each decimal number

a) 101 – 34
b) 60 – 125
c) -125 + 80
d) –34 – 123
e) 60 + 80
f) 80 – 80
g) -80 – 34

Perform the arithmetic functions of the ***unsigned*** binary numbers below in 2's complement form (Note: with unsigned binary numbers, MSB represents a number value)

a) 10001100+00111001
b) 11011001+11100111
c) 00110011-00010000
d) 01100101-11101000

OCTAL NUMBERS

Convert the octal numbers below to decimal and decimal to octal

a) 12_8
b) 73_8
c) 56_8
d) 163_8
e) 1024_8
f) 85_{10}
g) 103_{10}
h) 1024_{10}
i) 98_{10}
j) 999_{10}

Convert the following decimal fractions to its octal fraction equivalents and vice-versa

a) 28.175
b) 59.080
c) 88.888
d) 180.01_8
e) 407.304_8
f) 345.135_8

Convert each octal below to binary and each binary into octal

a) 13_8
b) 13271_8
c) 1100_2
d) 111100010111_2

Perform the calculations for the following octal values as shown below

a) $555_8 + 574_8$
b) $711_8 - 45_8$
c) $456_8 + 123_8$
d) $77714_8 + 76_8$
e) $765_8 - 444_8$
f) $44_8 - 6_8$

HEXADEMICAL NUMBERS

Perform the following conversions from hexadecimal number to binary, octal and decimal numbers

	Binary	Octal	Decimal
a) $A034B_{16}$			
b) $666FA_{16}$			
c) 66_{16}			
d) 191_{16}			

Perform the calculation for the following hex numbers as shown below:

a) 15h+32h
b) 12h+EBh
c) $AAA_{16} + 111_{16}$
d) $DDF_{16} + 11_{16}$
e) 16Fh + 4A2h
f) 9EFh + 9EFh
g) $C_{16} - 2_{16}$
h) $BB_{16} - C1_{16}$
i) 1586h – 243h
j) $576A_{16} - AB_{16}$
k) $1234_{16} - 4321_{16}$
l) $FD19_{16} - AC_{16}$

Applied knowledge questions

Digital systems represent characters of the roman/english alphabet using 7-bits of a byte. Using the ASCII conversion table below, translate the numbers below the table into its corresponding decimal value and decode the words below. You may have to first determine what numerical base is being used before attempting to decode. Note that only characters are used and there are no symbols in the encoded words.

Regular ASCII Chart (character codes 0 - 127)

000	(nul)	016 ►	(dle)	032	sp	048	0	064	@	080	P	096	`	112	p
001 ☺	(soh)	017 ◄	(dc1)	033	!	049	1	065	A	081	Q	097	a	113	q
002 ☻	(stx)	018 ↕	(dc2)	034	"	050	2	066	B	082	R	098	b	114	r
003 ♥	(etx)	019 ‼	(dc3)	035	#	051	3	067	C	083	S	099	c	115	s
004 ♦	(eot)	020 ¶	(dc4)	036	$	052	4	068	D	084	T	100	d	116	t
005 ♣	(enq)	021 §	(nak)	037	%	053	5	069	E	085	U	101	e	117	u
006 ♠	(ack)	022 ▬	(syn)	038	&	054	6	070	F	086	V	102	f	118	v
007 •	(bel)	023 ↨	(etb)	039	'	055	7	071	G	087	W	103	g	119	w
008 ◘	(bs)	024 ↑	(can)	040	(	056	8	072	H	088	X	104	h	120	x
009	(tab)	025 ↓	(em)	041	)	057	9	073	I	089	Y	105	i	121	y
010	(lf)	026	(eof)	042	*	058	:	074	J	090	Z	106	j	122	z
011 ♂	(vt)	027 ←	(esc)	043	+	059	;	075	K	091	[	107	k	123	{
012 ♀	(np)	028 ∟	(fs)	044	,	060	<	076	L	092	\	108	l	124	|
013	(cr)	029 ↔	(gs)	045	-	061	=	077	M	093	]	109	m	125	}
014 ♫	(so)	030 ▲	(rs)	046	.	062	>	078	N	094	^	110	n	126	~
015 ☼	(si)	031 ▼	(us)	047	/	063	?	079	O	095	_	111	o	127	⌂

- **0100001001100001011011100110000101101110011000 01**
- **6910810111210497110116**
- **1041511471511641411 54**
- **0x000x4d0x750x6c0x740x690x6d0x650x640x690x61**