جامعة بيرزيت

BIRZEIT UNIVERSITY

ANSWER BOOKLET

Student: Digital		Number: 1	
Course: Department:		Number:	
Division:		Instructor:	
Date: Day	Month	Year	

For Instructor's Use

Question	Grade
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
Total	

Numbers:-

1991

10^3 10^2 10^1 10^0

binary

~~102456~~

1011101

2^6 2^5 2^4 2^3 2^2 2^1 2^0

(base = radix = 2)
binary

~~radix~~ base = 8 → octal

16 → hexadecimal

$(0.1)_2 \rightarrow (1)(2^{-1}) = (0.5)_{10}$

addition

$1+0 = 0+1 = 1$

$1+1 = 0+0 = 0$

$(53)_{10}$

$(53)/2$

	1
26	0
13	1
6	0
3	1
1	1
0	

$(110101)_2$

32 16 4 1

~~$(47)_{10}$~~ $\equiv$ $(\,??\,)_8$

~~47~~

~~5~~

$(153)_{10} = (\,??\,)_8$

$153/8$

19	1
2	3
0	2

$\Rightarrow (153)_{10} = (231)_8$

Example :- convert

$(0.6875)_{10} \longrightarrow (\quad)_2$

$0.6875 \times 2 = 1.375$ 1 $\downarrow$

$0.375 \times 2 = 0.75$ 0

$0.75 \times 2 = 1.5$ 1

$0.5 \times 2 = 1.0$ 1

$\longrightarrow (0.6875)_{10} = (0.1011)_2$

Example

$\cancel{(0.333)_{10}}$ $(0.3)_{10}$

$\cancel{(0.333) * 2 = 0.666 \quad 0}$

$\cancel{0.666 * 2 =}$

$(0.3)_{10} =$

$(0.3) * 2 = 0.6$ 0

$(0.6) * 2 = 1.2$ 1

$(0.2) * 2 = 0.4$ 0

$(0.4) * 2 = 0.8$ 0

$0.8 * 2 = 1.6$ 1

0.6

0.01001

Octal & hexadecimal & binary

111 010 101 110 . 111 101

$(7 \quad 2 \quad 5 \quad 6 \;.\; 7 \quad 5)_8$

hexadecimal

0 1 2 ... 9 A B C D E F

F = 1111

6 5 2 . 1 3 7

110 101 010 . 001 011 111

$(1111 \quad 1010 \quad 1110)_2$

$(E \quad A \quad E)_H$

complements

9's complement

(2 3 4)

→ ~~complement~~ 9's complement =

999 − 234 = 7 6 5

2 3 7

7 6 5

0 1 2 3 $\xrightarrow{\text{9's complement}}$ = 9 8 7 6

(1 0 1 1 0) 1's compl

= 0 1 0 0 1

~~other~~ 10's complement = 9's + 1

	2	4	6	7	0	0
9's	7	5	3	2	9	9
10's	7	5	3	3	0	0

	0	1	1	0
1's	1	0	0	1
2's	1	0	1	0

In computer hardware it is better to implement the subtract as addition

example:

7 2 5 3 2 − 3 2 5 0

↑ M ↑ N

~~M~~ M

M = 7 2 5 3 2

10's comp of N = + 9 6 7 5 0

Same no. of digits

1 6 9 2 8 2

→ answer = 6 9 2 8 2

opposite

3 2 5 0 − 7 2 5 3 2

M = 0 3 2 5 0

10's complement of N = 2 7 4 6 8

3 0 7 1 8

10's complement = (−) 6 9 2 8 2

same thing for binary

** * section 1→2

1.6 Signed Binary Numbers

* (+) & (−) are not used in hardware ?
* by convention (bit 0 for +ve, 1 for −ve).
* binary ~~number~~ : signed & unsigned

left most for sign

e.g 1 1 0 0 1
unsigned = 25
signed = −9

$\left.\begin{matrix} 11001 \\ \text{unsigned} = 25 \\ \text{signed} = -9 \end{matrix}\right\}$ signed-magnitude

* ~~we can~~ positive numbers (signed) are presented by ~~one~~ one way

e.g 7 = 111

00000111

+ve sign

* There are three different ways to represent negative numbers
  * signed-magnitude +7 = 00000111
    −7 = 10000111
  * signed-1's-complement rep.
    +7 = 00000111
    −7 = 11111000

signed 2's − complement rep → −7 =

* Arithmetic operations
- Addition & subtraction

$9 - 15 = -(15 - 9)$

$= -(6)$

sign This is not the case in the hardware.

(same thing for signed-magnitude)

but in signed complement no need for comparison between the numbers
(it is only addition process

* A carry out of the sign-bit position is discarded

Examples (from book):

+ 6	0 0 0 0 0 1 1 0
+ 13	0 0 0 0 1 1 0 1
+ 19	0 0 0 1 0 0 1 1

- 6	1 1 1 1 1 0 1 0
+ 13	0 0 0 0 1 1 0 1
+ 7	1 0 0 0 0 0 1 1 1

$$
\begin{array}{rl}
+6 & 0\,0\,0\,0\,0\,1\,1\,0 \\
-13 & 1\,1\,1\,1\,0\,0\,1\,1 \\
\hline
-7 & 1\,1\,1\,1\,1\,0\,0\,1
\end{array}
$$

$-(0\,0\,0\,0\,0\,1\,1\,1) = -7$

2's complement since the sign is $\underline{\underline{-ve}}$

* * *

Binary Codes:

n-bit binary code is a group of n bits $a_1\ a_2\ a_3 \ldots a_n$

a_i either $\underline{\underline{0}}$ or $\underline{\underline{1}}$

- 2^n distinct combination can be presented with n bits.

- to represent m combination we need $\lceil \log_2 m \rceil$ bits

e.g by 3 bits we can make 8 combination

for ~~7~~ ~~8~~ ~~6~~ 5 combination we need $\lceil \log_2 5 \rceil = \underline{\underline{3}}$ (at least)

$\log_a x = \dfrac{\log_{10} x}{\log_{10} a}$??

~~→ by n bits~~

from 0 to 2^n-1 can be presented by n- bits

* BCD Code (binary coded decimal).

- BCD coding is a way to represent the decimal digits by means of a code that contains 1's & 0's, by this, it will be possible to perform the arithmetic operations directly with decimal numbers when they are stored in the computer in coded form.

- There are some unassigned bit combinations if the number of elements in the set is not a multiple power of 2 (as in BCD (in BCD 6 states are not assigned)

Decimal	BCD digit
0	0000
1	0001
2	0010
3	0011
⋮	
9	1001

(as normal binary) up to now but

If we want to represent more than 1 digit then it will be different from binary conversion

e.g. $(9)_{10} = (1001)_{BCD} = (1001)_2$

but $(15)_{10} = (0001\ 0101)_{BCD} = (1111)_2$

BCD is decimal not binary.

* BCD Addition

example 1

$$\begin{array}{rr} 2 & 0010 \\ +\ 3 & 0011 \\ \hline +5 & 0101 \end{array}$$

(addition result is OK as well as the sum is less than or equal 9).

Example 2:

$$\begin{array}{rr} 7 & 0111 \\ +\ 3 & 0011 \\ \hline 10 & 1010 \\ & 0110 \\ \hline & 10000 \end{array}$$

(10 in binary but this is not BCD.)

6

$(0001\ 0000)_{BCD}$

Example

	1			1	
	3 7	0 0 1 1	0 1 1 1		
+	9 6	1 0 0 1	0 1 1 0		
	1 3 3	1 1 0 1	1 1 0 1		
		0 1 1 0	0 1 1 0		
		1 0 0 1 1	0 0 1 1		
✓		1 3	3		

** Section 8.9

Example

	3 7	0 0 0 1 1	0 1 1 1	
−	9 6	0 0 0 0	0 1 0 0	
		1 0 1 0 0	1 0 1 1	
			0 1 1 0	
			0 0 0 1	

− sign

1 0 1 0 0 0 0 0 1

0 0 1 0 1 1 0 0 1 = −59

other decimal codes

- BCD 8 4 2 1 ✓

- 2 4 2 1 code

0	0	0	0	→ 0
1	1	1	1	9
1	0	0	0	2
0	0	1	0	2

} ~~more~~ than one possibility for coding

(BCD & 2421 codes are example of weighted codes): adding the weight of digits will 1 binary

e.g

2	4	2	1
1	0	1	1

= 2 + 2 + 0 + 1
= 5

- The advantage of 2421 is self complementing code

e.g 7

2	4	2	1
0	1	1	1
1	0	0	0

→ comple

→ 2 complement of 7

- Excess 3 code
0 → 00 11
1 → 01 00
⋮
9 → 11 00

advantage: self complement code

e.g.
1 → 0 1 0 0

8 ← 1 0 1 1

- 8 4 -2 -1 code
(include negative)

in all codes there are some unused states.

* * *
Section ~~120~~ 1:00 - 2:00 6

Gray code

~~0000 0001 0011 0010~~

0000 0001 | 0011 0010 | 0110, 0111, 0101, 0100

0 1 2 3 4 5 6 7

Advantage: one change only for consequent numbers.

it will be used ~~take~~ alot in this ~~future~~ course.

ASCII character code

128 char 7 bits

e.g A 1000001

B

⋮

ASCII (American standard code for Information Interchange). is

* Also control chars are included e.g DEL) for delete

Error Detecting Code

- it mainly used for error-check in data communication (parity check

- two types of parity check: even & odd.

even

ASCII A = 1 000 000 1
with even A will be sent like
0 1 000 000 1
with odd parity it will be sent
1 1 000 000 1

* even parity is more common

* error detection (e.g. even parity)
A will be sent 0 1 000 000 1
if it received as 0 0 000 000 1
→ error will be detected since parity is odd.
but even number of errors will not be detected (only odd no. will be detected).