

1. Convert each of the following binary representations to decimals.

(i)  $1101.101 = 13.625$

(ii)  $111010011 = 256 + 128 + 64 + 16 + 2 + 1 = 467$

(iii)  $0.00001110101 = 1110101 * 2^{-11} = 117/2048 = 0.05712890625$

(iv)  $1101.001101 = 13 + \frac{1}{8} + \frac{1}{16} + \frac{1}{64} = 13 + \frac{13}{64} = 13.203125$

(v)  $111010011.00011010111 = 467 + \frac{215}{2^{11}} = 467.10498046875$

2. Find the binary representation for the following numbers and set out your computations in convenient tables.

(i) One possible table lay-out:

Rem.	4825	729	729	729	217	217	89	25	25	9	1	1	1	0
$2^n$	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
$b$	0	1	0	0	1	0	1	1	0	1	1	0	0	1

The binary representation is 1001011011001

Notice that the algorithm might be described as follows:

if the number in the second row of the next column is less than the number in the first row, place a 1 in the third row of the next column, otherwise place a zero there. Then subtract the product of the numbers in the second and third rows of the next column from the number in the first row to obtain the number in the first row of the next column. Diagrammatically:

$N$	$N - P \times B$	... proceed until $N$ is reduced to 0
-	$P$	... the place values
-	$B = (N > P)$	... a binary decision(0 or 1)

This is a repetitive process. Some of you will be able to set it up in a spreadsheet.

(ii) Repetitive doubling will find how many halves, quarters, eighths etc. are contained in the number and removed.

Integer part	0	1	1	0	1	0	0	1	0	0	1
Fractional part	8217	6434	2868	5736	1472	2944	5888	1776	3552	7104	4208

The binary representation of 0.8217 to 10 binary places is 0.1101001001. This binary number is actually equal to the decimal 0.8212890625 so if this accuracy is not sufficient more binary places would be needed.

-	$I = (1 < (2 \times F))$	... proceed until required accuracy is obtained
$F$	$(2 \times F) - I$	... keep only fractional parts

(iii) The successive doubling will work just as well if we use rational number arithmetic:

I	0	0	1	0	1	1	0	0	1	1	1
F	$\frac{13}{37}$	$\frac{26}{37}$	$\frac{52-37}{37} = \frac{15}{37}$	$\frac{30}{37}$	$\frac{23}{37}$	$\frac{9}{37}$	$\frac{18}{37}$	$\frac{36}{37}$	$\frac{35}{37}$	$\frac{33}{37}$	$\frac{29}{37}$

Thus the binary expansion to ten places for  $\frac{13}{37}$  is 0.0101100111 — which is the exact representation for 0.3505859375. The decimal expansion for  $\frac{13}{37}$  is 0.3513513513...

- (iv) We will find the binary representation for 81.65 by following the algorithm in part 2(i) through into negative powers of 2.

$n$	$p = 2^n$	$b$	$n - p \times b$
7	128	0	81.65
6	64	1	17.65
5	32	0	17.65
4	16	1	1.65
3	8	0	1.65
2	4	0	1.65
1	2	0	1.65
0	1	1	0.65
-1	0.5	1	0.15
-2	0.25	0	0.15
-3	0.125	1	0.025
-4	0.0625	0	0.025
-5	0.03125	0	0.025
-6	0.015625	1	0.009375
-7	0.0078125	1	0.0015625
-8	0.00390625	0	0.0015625
-9	0.001953125	0	0.0015625
-10	0.0009765625	1	0.000585938

It is not convenient to do this by hand but it is more convenient than having to change the procedure (algorithm) on reaching the binary point. The doubling technique is only convenient when one has to work by hand and would like to avoid fractions (decimal or otherwise).

We have 81.65 in binary (to 10 places) as 1010001.1010011001 (which in fact is the exact representation for 81.6494140625).

- (v) It would be nice to have a user defined function (in whatever environment) to convert 2101.0226 to binary. It would then be a simple matter to compute it to 40 binary places and get:

100000110101.0000010111001001000111010001010011100011

(which, in fact is the exact representation for 2101.022599999999).

