Writing negative binary numbers in two's complement format and converting to denary

Example 1

The following three examples show how we can write negative binary numbers in the two's complement format:

−128	64	32	16	8	4	2	1
1	0	0	1	0	0	1	1

By following our normal rules, each time a 1 appears in a column, the column value is added to the total. So, we can see that in denary this is: −128 + 16 + 2 + 1 = −109.

−128	64	32	16	8	4	2	1
1	1	1	0	0	1	0	0

Similarly, in denary this number is −128 + 64 + 32 + 4 = −28.

−128	64	32	16	8	4	2	1
1	1	1	1	0	1	0	1

This number is equivalent to −128 + 64 + 32 + 16 + 4 + 1 = −11.

Note that a two's complement number with a 1-value in the −128 column must represent a negative binary number.

Converting negative denary numbers into binary numbers in two's complement format

Consider the number +67 in 8-bit (two's complement) binary format:

−128	64	32	16	8	4	2	1
0	1	0	0	0	0	1	1

Method 1

Now let's consider the number −67. One method of finding the binary equivalent to −67 is to simply put 1s in their correct places:

−128	64	32	16	8	4	2	1
1	0	1	1	1	1	0	1

−128 + 32 + 16 + 8 + 4 + 1 = −67

Method 2

However, looking at the two binary numbers above, there is another possible way to find the binary representation of a negative denary number:

first write the number as a positive binary value – in this case 67:	0 1 0 0 0 0 1 1
we then invert each binary value, which means swap the 1s and 0s around:	1 0 1 1 1 1 0 0
then add 1 to that number:	1
this gives us the binary for −67:	1 0 1 1 1 1 0 1

Example 2

Convert –79 into an 8-bit binary number using two's complement format.

Method 1

As it is a negative number, we need a 1-value in the –128 column.

–79 is the same as –128 + 49

We can make up 49 from 32 + 16 + 1; giving:

–128	64	32	16	8	4	2	1
1	0	1	1	0	0	0	1

Method 2

write 79 in binary:	0 1 0 0 1 1 1 1
invert the binary digits:	1 0 1 1 0 0 0 0
add 1 to the inverted number	1
thus giving –79:	1 0 1 1 0 0 0 1

–128	64	32	16	8	4	2	1
1	0	1	1	0	0	0	1

It is a good idea to practise both methods.

When applying two's complement, it isn't always necessary for a binary number to have 8 bits:

Example 3

The following 4-bit binary number represents denary number 6:

–8	4	2	1
0	1	1	0

Applying two's complement (1 0 0 1 + 1) would give:

–8	4	2	1
1	0	1	0

in other words: –6

Example 4

The following 12-bit binary number represents denary number 1676:

−2048	1024	512	256	128	64	32	16	8	4	2	1
0	1	1	0	1	0	0	0	1	1	0	0

Applying two's complement (1 0 0 1 0 1 1 1 0 0 1 1 + 1) would give:

−2048	1024	512	256	128	64	32	16	8	4	2	1
1	0	0	1	0	1	1	1	0	1	0	0

In other words: −1676

Activity 1.13

Convert the following negative denary numbers into binary numbers using the two's complement format:

a −18 **c** −47 **e** −88 **g** −100 **i** −16
b −31 **d** −63 **f** −92 **h** −1 **j** −127

Activity 1.14

Convert the following negative binary numbers (written in two's complement format) into negative denary numbers:

a	1	1	0	0	1	1	0	1
b	1	0	1	1	1	1	1	0
c	1	1	1	0	1	1	1	1
d	1	0	0	0	0	1	1	1
e	1	0	1	0	0	0	0	0
f	1	1	1	1	1	0	0	1
g	1	0	1	0	1	1	1	1
h	1	1	1	1	1	1	1	1
i	1	0	0	0	0	0	0	1
j	1	1	1	1	0	1	1	0

1.2 Text, sound and images

1.2.1 Character sets – ASCII code and Unicode

The **ASCII code** system (American Standard Code for Information Interchange) was set up in 1963 for use in communication systems and computer systems. A newer version of the code was published in 1986. The standard ASCII code **character set** consists of 7-bit codes (0 to 127 in denary or 00 to 7F in