b

	0 0 0 1 0 0 1 1
+	0 0 0 1 0 0 0 1
carry	1 1 1
sum	0 0 1 0 0 1 0 0

c

128	64	32	16	8	4	2	1
1	0	0	1	0	0	0	0

d

128	64	32	16	8	4	2	1
0	0	0	0	0	1	0	0

In **c** the result is $36 \times 2^2 = 144$ (which is correct).

In **d** the result of the right shift gives a value of 4, which is incorrect since $36 \div 2^3$ is not 4; therefore, the number of possible right shifts has been exceeded. You can also see that a 1 has been lost from the original binary number, which is another sign that there have been too many right shifts.

Activity 1.11

1 **a** Write down the denary value of the following binary number.

0	1	1	0	1	0	0	0

b Shift the binary number three places to the right and comment on your result.

c Write down the denary value of the following binary number.

0	0	0	0	1	1	1	1

d Shift the binary number four places to the left and comment on your result.

2 **a** Convert 29 and 51 to 8-bit binary numbers.

b Add the two binary numbers in part **a**.

c Shift the result in part **b** three places to the right.

d Convert 75 to an 8-bit binary number.

e Add the two binary numbers from parts **c** and **d**.

f Shift your result from part **e** one place to the left.

1.1.6 Two's complement (binary numbers)

Up until now, we have assumed all binary numbers are positive integers. To allow the possibility of representing negative integers we make use of **two's complement**. In this section we will again assume 8-bit registers are being used. Only one minor change to the binary headings needs to be introduced here:

−128	64	32	16	8	4	2	1

In two's complement the left-most bit is changed to a negative value. For instance, for an 8-bit number, the value 128 is now changed to −128, but all the other headings remain the same. This means the new range of possible numbers is: −128 (10000000) to +127 (01111111).

It is important to realise when applying two's complement to a binary number that the left-most bit always determines the sign of the binary number. A 1-value in the left-most bit indicates a negative number and a 0-value in the left-most bit indicates a positive number (for example, 00110011 represents 51 and 11001111 represents −49).

Writing positive binary numbers in two's complement format

Example 1

The following two examples show how we can write the following positive binary numbers in the two's complement format 19 and 4:

−128	64	32	16	8	4	2	1
0	0	0	1	0	0	1	1
0	0	0	0	0	1	0	0

As you will notice, for positive binary numbers, it is no different to what was done in Section 1.1.2.

Converting positive denary numbers to binary numbers in the two's complement format

If we wish to convert a positive denary number to the two's complement format, we do exactly the same as in Section 1.1.2:

Example 2

Convert **a** 38 **b** 125 to 8-bit binary numbers using the two's complement format.

a Since this number is positive, we must have a zero in the −128 column. It is then a simple case of putting 1-values into their correct positions to make up the value of 38:

−128	64	32	16	8	4	2	1
0	0	1	0	0	1	1	0

b Again, since this is a positive number, we must have a zero in the −128 column. As in part **a**, we then place 1-values in the appropriate columns to make up the value of 125:

−128	64	32	16	8	4	2	1
0	1	1	1	1	1	0	1