**Negative Binary Numbers Task**

|  |
| --- |
| *At National 5 we only used positive integers. Most numbering systems have both positive and negative numbers therefore we need a method of representing these. This is known as* ***Two’s Complement.***  *In Two’s complement, the most significant bit (the leftmost bit) indicates whether the number is positive or negative. If the most significant bit is a 1 then it is a negative number, if it is a 0 then it is positive.*  *If we were using 8-bits then using two’s complement we would know Example 1 is a positive number (because the leftmost bit is a 0) and Example 2 is a negative number (because the leftmost bit is a 1).*  128  64  32  16  8  4  2  1  0  0  1  0  0  0  0  0  Example 1  128  64  32  16  8  4  2  1  1  1  1  0  1  1  1  1  Example 2 |

*How to convert binary numbers to denary numbers?*

**METHOD 1**

Negative binary numbers are represented in the following way:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |

The example above is the denary number -100. A zero means that the number above will NOT be added. A one means that the number will be added. So, -128 + 16 + 8 + 4 = -100.

Notice that the first number is no longer 128 but -128. This indicates whether or not the number will be a negative number. So if the first number is a 1, it will be a negative number. If the first number if a 0, it will be a positive number.

**EXERCISE 1**

Try the following examples:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans 1** |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans 2** |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans 3** |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans 4** |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans 5** |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |  |  |

**METHOD 2**

Method 1 works well for 8-bit binary numbers, but what if we have more bits?

Let’s look at an example past paper question below.

|  |
| --- |
| *Convert the following 16-bit two’s complement number into denary.*  *1111 1110 1110 1011* |

Method 1 isn’t a good option to use here. An easier method is to:

* Flip all of the bits
* Add 1 to the number

Flip the bits means 1 becomes 0 and 0 becomes 1.

So 1111 1110 1110 1011 flipped looks like:

0000 0001 0001 0100

Remember, we need to add one. So we are now left with:

0000 0001 0001 0101

We can ignore the first seven 0s, and that gives us the number 1 0001 0101.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **256** | **128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |  | **Ans** |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |  | **277** |

The last thing we have to do is look at the original number (1111 1110 1110 1011). As this number started with a 1, we need to remember to make the number negative. So the answer is -277.

**EXERCISE 2**

Try the following examples:

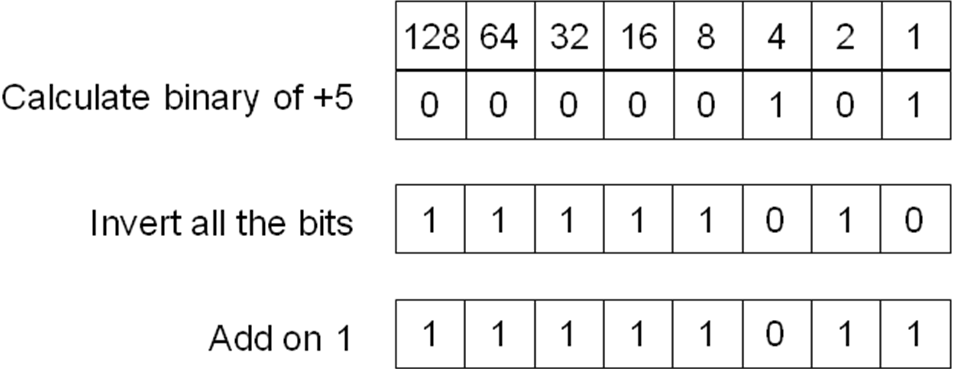
|  |  |
| --- | --- |
| 1. 1111 1110 0010 1011 |  |
| 2. 1111 1110 1110 1001 |  |
| 3. 1111 1110 1010 1011 |  |
| 4. 1110 1000 1011 |  |
| 5. 1111 1111 1111 1110 0000 1011 |  |

*How to convert denary numbers to binary numbers?*

To convert a negative denary number into its two complement representation you:

* work out the positive binary number
* invert all the bits (so a 1 becomes a 0 and a 0 becomes a 1)
* then add 1.

To show how -5 would be represented as an 8-bit binary number:



Therefore the two’s complement representation of -5 would be 11111011.

**EXERCISE 3**

Try the following examples:

|  |  |
| --- | --- |
| 1. -100 |  |
| 2. -30 |  |
| 3. -77 |  |
| 4. -81 |  |
| 5. -200 (Careful with this one, an extra bit is needed) |  |