• Negating 2’s complement numbers
  • Complement each bit and then add 1.

• Value of an N-bit 2’s complement number $b_{n-1}b_{n-2}...b_2b_1b_0$

$$-b_{n-1}2^{n-1} + \sum_{k=0}^{n-2} b_k2^k$$

• Sign extension of 2’s complement numbers
  • Replicate the most significant bit (MSB) to make numbers longer
  • For example, going from 4-bit to 8-bit numbers:
    • 0101 (+5) should become 0000 0101 (+5).
    • But 1100 (−4) should become 1111 1100 (−4).

• Subtraction: implement by negating the 2nd input and then adding.

• Overflow occurs when:
  – If you add two positive numbers and get a negative result.
  – If you add two negative numbers and get a positive result.

If 0100 is the 5-bit representation for 11, what is the 2’s complement representation for -11?

<table>
<thead>
<tr>
<th>0100</th>
<th>+4_{10} (a positive number in 4-bit two’s complement)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>= (invert all the bits)</td>
</tr>
<tr>
<td></td>
<td>= -4_{10} (and add one)</td>
</tr>
</tbody>
</table>

We have 2 unsigned 8-bit words: $a_7a_6a_5a_4a_3a_2a_1a_0$
And we want the 8-bit word: $b_7b_6b_5b_4b_3b_2b_1b_0$

We have 2 unsigned 8-bit words: $a_7a_6a_5a_4a_3a_2a_1a_0$
And we want the 8-bit word: $b_7b_6b_5b_4b_3b_2b_1b_0$

We have 2 unsigned 8-bit words: $a_7a_6a_5a_4a_3a_2a_1a_0$
And we want the 8-bit word: $b_7b_6b_5b_4b_3b_2b_1b_0$

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>+ (-4)</td>
</tr>
</tbody>
</table>

- | 1 | 1 | 1 | 0 |
- | 0 | 0 | 1 | 1 |