Introduction

In digital computers, data is ultimately stored as bits that take on the value 1 or 0. Later in this course, you’ll see how the ability to manipulate data in this format enables us to more easily build complex, reliable computer systems. For now, we’re concerned with understanding how bits can represent data.

Before you start, complete the form below to assign a role to each member. If you have 3 people, combine speaker and reflector.

<table>
<thead>
<tr>
<th>Team</th>
<th>Date</th>
<th>Team Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Recorder</strong></td>
<td></td>
<td>records all answers &amp; questions, and provides copies to team &amp; facilitator.</td>
</tr>
<tr>
<td><strong>Speaker</strong></td>
<td></td>
<td>talks to facilitator and other teams.</td>
</tr>
<tr>
<td><strong>Manager</strong></td>
<td></td>
<td>keeps track of time and makes sure everyone contributes appropriately.</td>
</tr>
<tr>
<td><strong>Reflector</strong></td>
<td></td>
<td>considers how the team could work and learn more effectively.</td>
</tr>
</tbody>
</table>

Part A: Bits and binary numbers

Consider a device that samples a sound wave at regular intervals to convert it to a sequence of numbers, each stored digitally as bits. The resolution of the sampling is how many bits each sample gets. The plot below shows a wave, where the vertical lines are points in time where the wave is sampled. The y-axis has no label because we are going to consider different resolutions. We'll call the number of ticks on the y-axis the number of levels.
1. Fill in the following table. The first row is already finished.

<table>
<thead>
<tr>
<th>Resolution</th>
<th>The list of distinct levels at this resolution</th>
<th>Number of levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 bit</td>
<td>0, 1</td>
<td>2</td>
</tr>
<tr>
<td>2 bits</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 bits</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N bits</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

2. How did you determine the last row?

3. How many more levels are available when you add 1 bit of resolution?

4. How many more levels are available when you double the number of bits of resolution?

5. Suppose we converted the sampled wave back to real audio using a speaker. Describe how a low-resolution wave might sound compared to a high-resolution wave.

An ordered sequence of bits can be used to represent integers. To represent an integer in **base two (aka, binary)** means to write its "digits" as bits (1's or 0's) indicating the powers of two that comprise the number.

\[
193_{10} = \begin{array}{ccc}
    \text{Hundreds} & \text{tens} & \text{ones} \\
    1 & 9 & 3 \\
\end{array} \\
\]

\[
101_2 = \begin{array}{ccc}
    \text{fours} & \text{twos} & \text{ones} \\
    1 & 0 & 1 \\
\end{array} \\
\]
6. List the numbers zero to eight in order, representing them in base two.

7. Translate the following base two numbers to our typical number base: base ten (i.e., decimal).

\[1111_2 = \_\_\_10\]

\[10101_2 = \_\_\_10\]

8. Write a formula for translating a binary number \(x_{n-1} \ldots x_3 x_2 x_1\) to decimal. The \(x_i\) means the \(i^{th}\) digit of the binary number (either a 1 or 0). Your formula may contain decimal constants in it.

9. Translate the following base ten numbers to base two.

\[8_{10} = \_\_\_2\]

\[25_{10} = \_\_\_2\]

\[100_{10} = \_\_\_2\]

10. Write an algorithm (as pseudocode or detailed steps is acceptable) to compute each bit in the binary representation of a decimal number.

11. Use what you've learned about base 10 and base 2 numbers to translate numbers between other bases.

\[25_{10} = \_\_\_\_3\]

\[21_{16} = \_\_\_\_8\]
Part B: Including negative integers

On the left is a number wheel for positive integers that can be represented using 3 bits. Notice that +1 takes you to the next position going clockwise and -1 takes you to the next position counter-clockwise. At the lightning bolt, we "overflow".

1. On the following number line, write out 3-bit binary integers, using the following rule: the leftmost bit indicates sign (0 positive, 1 negative) and the remaining bits indicates the absolute value.

<table>
<thead>
<tr>
<th>binary</th>
<th>101</th>
<th>000</th>
<th>001</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

2. Did you use all available 3-bit numbers? Why or why not?

3. Turn your number line into a number wheel by attaching the endpoints. What do you observe that is interesting?
Let’s try a different approach to negative integers by starting with the 3-bit number wheel.

4. Complete the wheel to include both positive and negative integers. The overflow point and 0 have been provided as hints. Use the following additional constraints:
   • the most significant (left-most) bit must be 1 for negatives and 0 for positives and zero
   • no integer may appear twice (i.e., no integer may have two different 3-bit representations)

![3-bit number wheel](image)

The most common way to represent signed integers using bits is **two's complement**. A correct number wheel above demonstrates 3-bit two's complement. In n-bit two's complement, the binary digits \( x_{n-1} \ldots x_3 \ x_2 \ x_1 \) can be interpreted using the expression

\[ x_{n-1} \times -(2^{n-1}) + \sum_{i=0}^{n-2} x_i \times 2^i \]

5. Convert the following values from 4-bit two's complement to decimal.

1011

1111

6. For an n-bit two's complement integer, what is the smallest value that can be represented? The largest? Why?

---

**Two's complement algorithm:**
To convert a negative integer, \(-x\), to \(n\)-bit two's complement:

1. Convert \(x\) to its \(n\)-bit binary representation
2. Complement each bit of \(x\) (\(\bar{x}\)), that is, change 1's to 0's and 0's to 1's
3. Add one to \(\bar{x}\)

7. Convert the following to 8-bit two's complement.

-59

-128

-1

8. What is the value of \(x + \bar{x} + 1\)?

9. Solve the above equation for \(\bar{x} + 1\). How does this new equation relate to the two's complement algorithm?

Other than binary and decimal, another common base that we will use is 16 or hexadecimal (or hex for short). In hexadecimal, we represent digits larger than 9 as follows: A=10, B=11, C=12, D=13, E=14, F=15.

10. Convert the following numbers to the given base.

\(A1_{16} = \text{______}_{10}\)

\(56_{10} = \text{______}_{16}\)

\(267_{10} = \text{______}_{16}\)

Derived from CS-POGIL ORG_02_TWOS