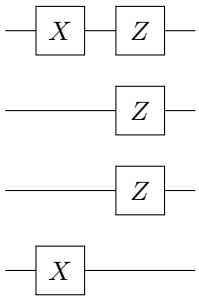
Encoding a Message

- **Description.** We saw in the last activity that Bob can encode two bits of data by appropriately using the NOT gate and the Z gate. However, in the real world Bob would need to send far more information to Alice than a simple two bits of data. For example, in the case of our bike sharing network, it might be necessary to transmit the name of the user who is trying to borrow a bicycle. So, how do we encode letters using bits? A common solution is ASCII, which uses eight bits to represent the Latin alphabet and other common symbols. Each combination of eight bits corresponds to a different letter, as illustrated by the following table.

$a \mapsto 01100001$	$b \mapsto 01100010$	$c \mapsto 01100011$	$d \mapsto 01100100$
$e \mapsto 01100101$	$f \mapsto 01100110$	$g \mapsto 01100111$	$h \mapsto 01101000$
$i \mapsto 01101001$	$j \mapsto 01101010$	$k \mapsto 01101011$	$l \mapsto 01101100$
$m \mapsto 01101101$	$n \mapsto 01101110$	$o \mapsto 01101111$	$p \mapsto 01110000$
$q \mapsto 01110001$	$r \mapsto 01110010$	$s \mapsto 01110011$	$t \mapsto 01110100$
$u \mapsto 01110101$	$v \mapsto 01110110$	$w \mapsto 01110111$	$x \mapsto 01111000$
$y \mapsto 01111001$	$z \mapsto 01111010$		

  In this challenge, you are asked to first pick your favourite letter. We will then pretend that Alice has sent us four qubits from her Bell pairs, so that we can transmit back the eight-bit representation of our favourite letter. For example, the following circuit encodes 11010110. (Note that this is not a valid ASCII letter).



- **Submission.** The letter you have picked and the circuit which encodes the letter (either hand-drawn or in a simulator).