Q-Pong

JM Torres

torresjm@fr.ibm.com

2 Décembre 2019

Pong

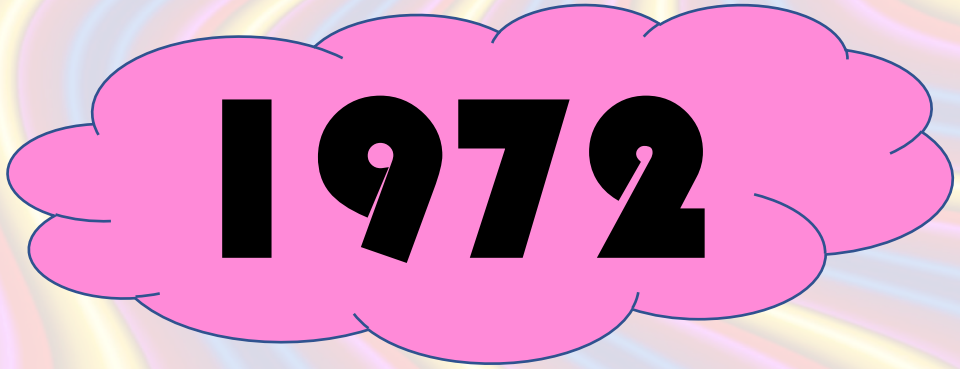
is a two-dimensional sports game that simulates table tennis.

The player controls an in-game paddle by moving it vertically across the left or right side of the screen.

They can compete against another player controlling a second paddle on the opposing side.

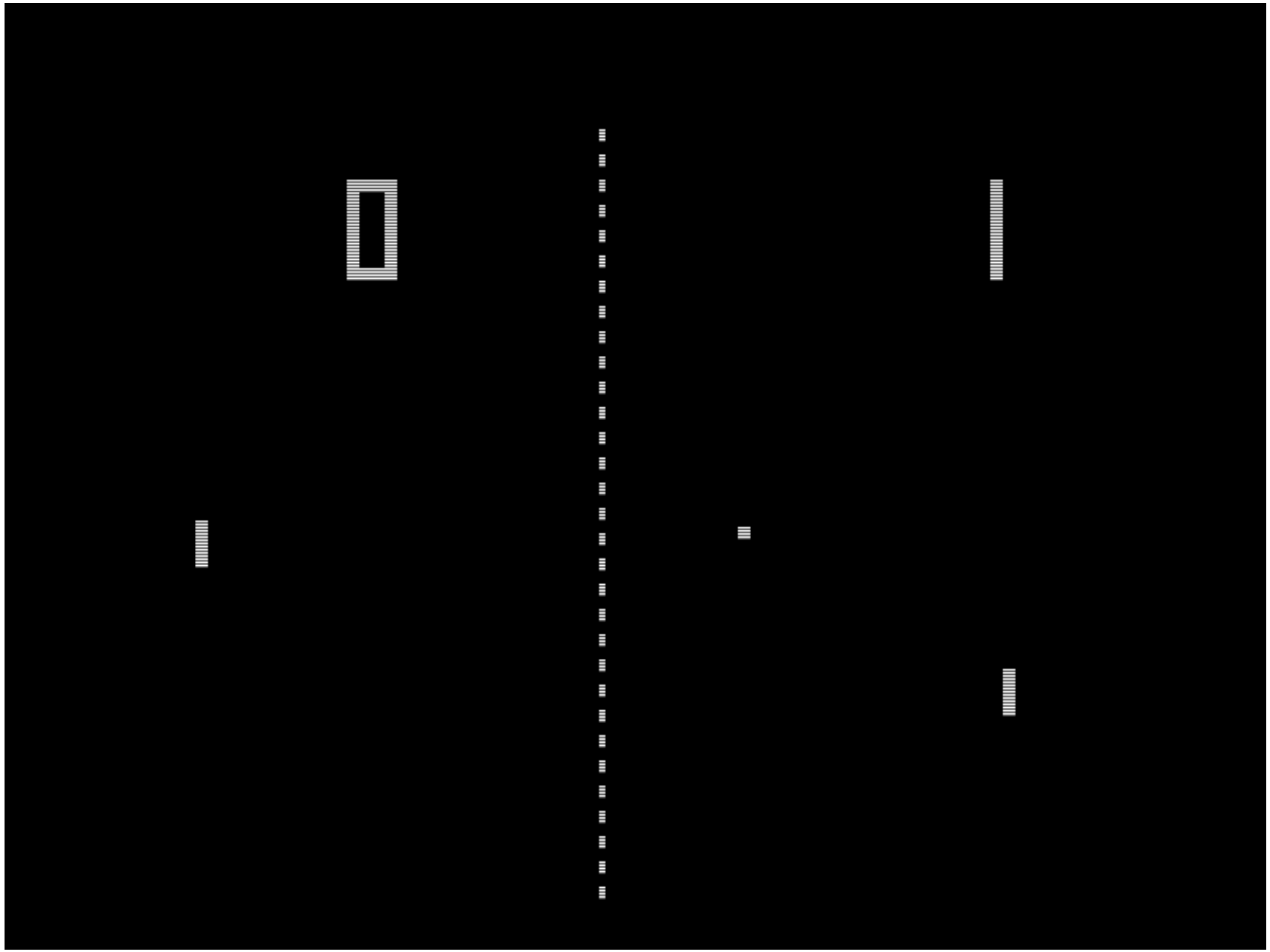
Players use the paddles to hit a ball back and forth.

The goal is for each player to reach eleven points before the opponent; points are earned when one fails to return the ball to the other.



Source Wikipedia.

Écran d'une partie en cours de Pong. Le joueur de droite mène un point à zéro.



Source Wikipedia.

Atari engineer Allan Alcorn designed and built *Pong* as a training exercise.

Source Wikipedia.

QPONG by Huang Junye 黄俊晔

Roundup of Qiskit Hackathon @ Singapore

Anurag Saha Roy Follow

Oct 15 · 4 min read

Qiskit Hackathon @ Singapore Participants & Organisers (Photo: Centre for Quantum Technologies)

QPONG

SELECT DIFFICULTY LEVEL

[A] EASY

[B] NORMAL

[X] EXPERT

CREDITS

MADE BY HUANG JUNYE, JAMES WEAVER, JARROD REILLY AND ANASTASIA JEFFERY

INITIATED AT IBM QISKIT CAMP 2019

POWERED BY JAVAFXPERT/QUANTUM-CIRCUIT-GAME

HuangJunye / QPong

Watch 4 | Unstar 20 | Fork 11

Code | Issues 11 | Pull requests 0 | ZenHub | Actions | Projects 0 | Wiki | Security | Insights

A quantum version of the classic Pong using Qiskit and pygame. Unity version here: https://github.com/HuangJunye/QPong-U...

quantum python game pong

248 commits | 6 branches | 0 packages | 3 releases | 4 contributors | Apache-2.0

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

HuangJunye Update README.md — Latest commit 9e5c993 on 19 Sep

containers	clean up the codes	6 months ago
controls	Move data folder from utils/ to root	5 months ago
data	Move data folder from utils/ to root	5 months ago
model	Remove import *	5 months ago
utils	change winning score to 7	5 months ago
viz	delete circuit_diagram.py	5 months ago

Installation

To play the game, you will need to install Python and three required packages. To do that, you need to use command line tool (Command Prompt for Windows or Terminal for macOS).

Open command line tool

On Windows, open Command Prompt by typing "Command Prompt" on the search box of Start menu. Check this link if you are not sure about how to do that: https://www.wikihow.com/Open-the-Command-Prompt-in-Windows

On macOS, press Command + Space to open Spotlight. Type "Terminal" on Spotlight to open Terminal. Check this link if you are not sure about how to do that: https://www.wikihow.com/Open-a-Terminal-Window-in-Mac

Install Python

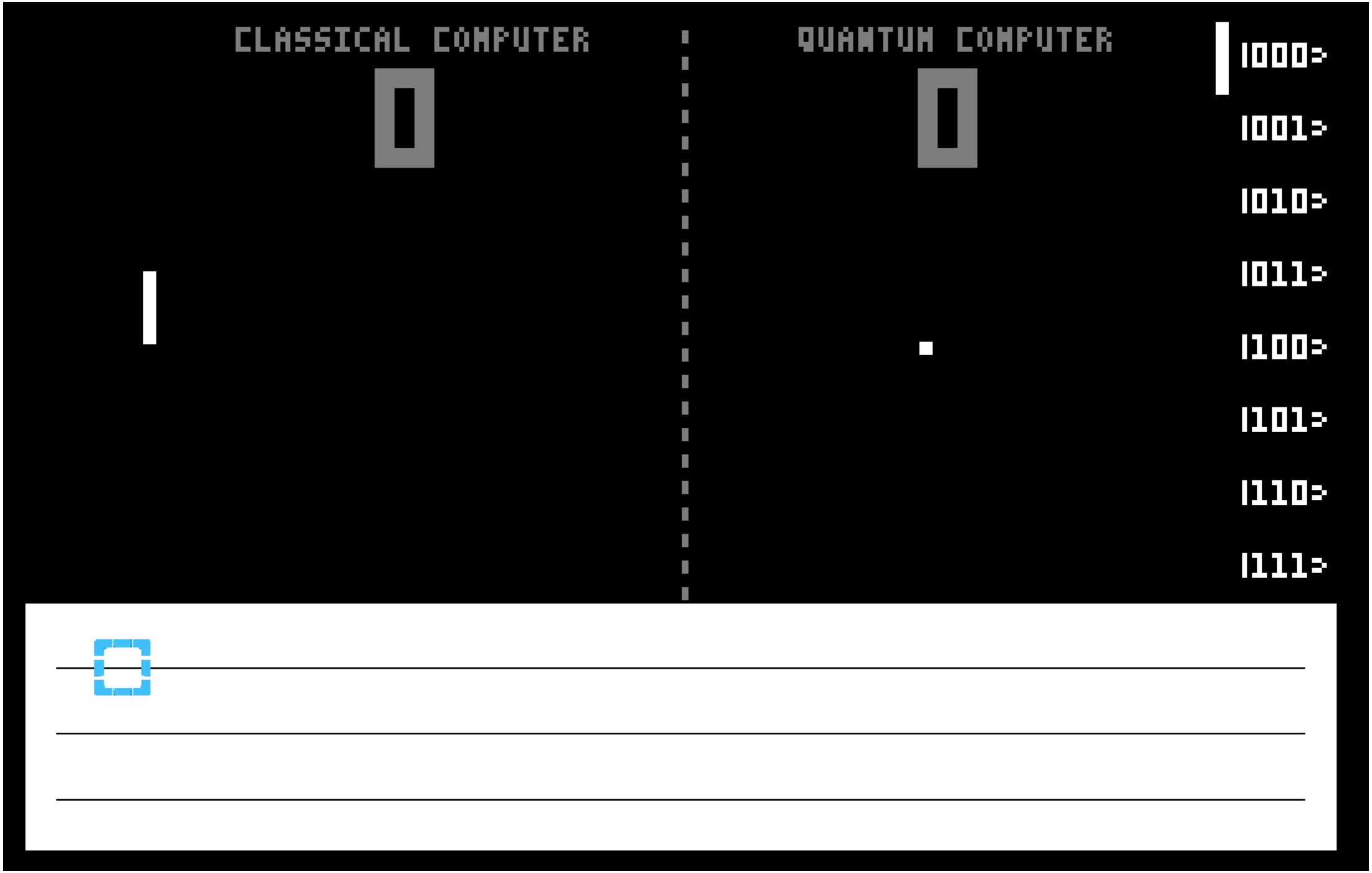
You can install Python from https://www.python.org/ or install Anaconda from https://www.anaconda.com

Install required packages

There are three Python packages required to run the game: PyGame, Qiskit and matplotlib.

Run `pip install [package name]` on command line tool (same for Windows and macOS) to install the packages. For example:

```
pip install pygame
```


CLASSICAL COMPUTER
QUANTUM COMPUTER
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>

How to play

Keyboard

W, A, S, D: Up, Left, Down, Right to move cursor

SPACE: delete gate

X, Y, Z, H: add Pauli-X, Pauli-Y, Pauli-Z, Hadamard gate

C: add CNOT gate

UP, DOWN: move control qubit up or down

Left, Right: add rotation to a gate, at pi/8 step

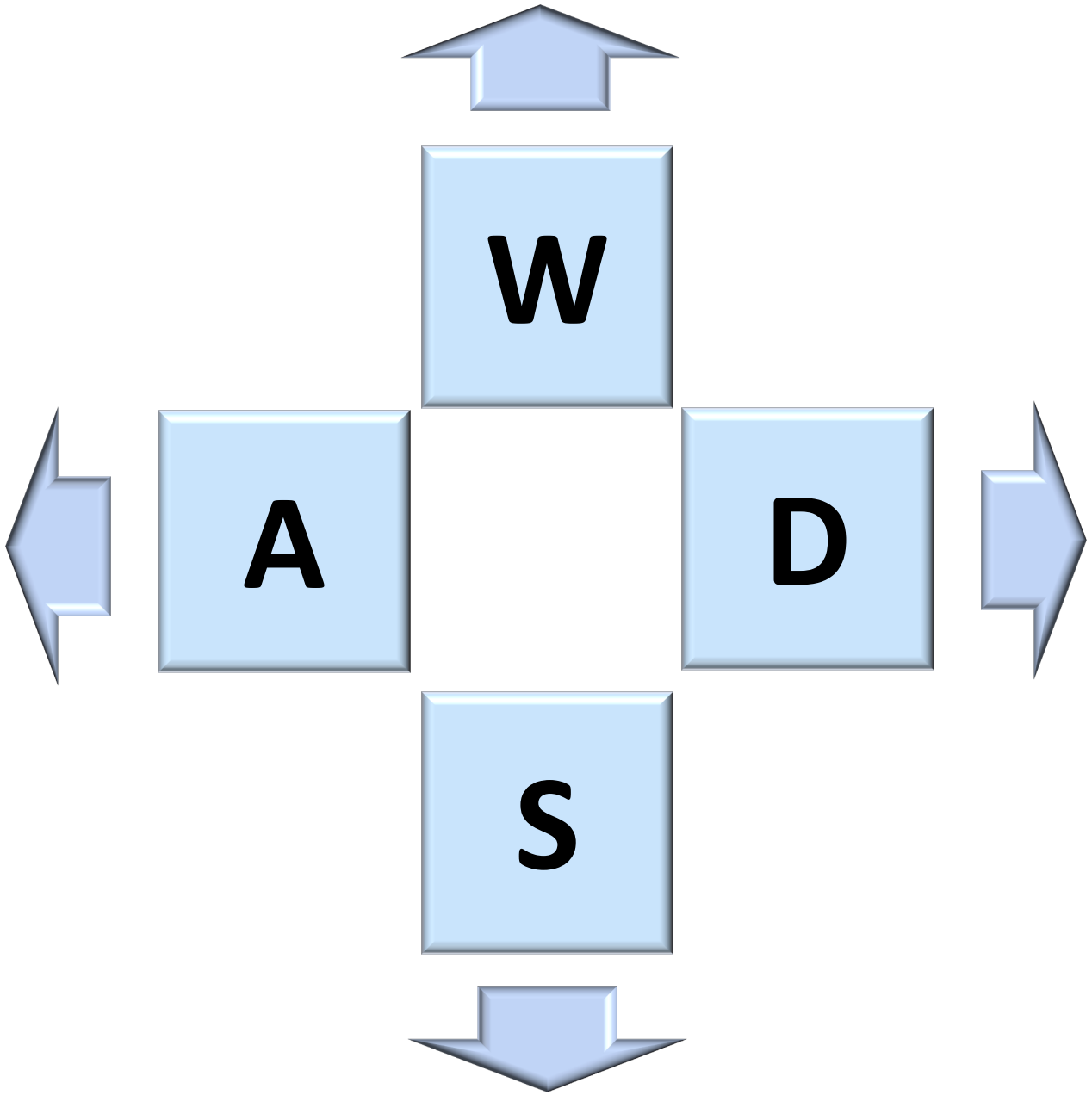
TAB: update visulization

Joystick

Joystick button correspondence depends on the model of joystick. Details will be added later

Credits

Sound effects are made by NoiseCollector from Freesound.org: https://freesound.org/people/NoiseCollector/packs/254/
Font used in the game is Bit5x3 made by Matt LaGrandeur: http://www.mattlag.com/bitfonts/



X Y

Z H

CNOT:

C+⇧/⇩

Déplacer la batte

X « NOT »: $|1\rangle \rightarrow |0\rangle$
$|0\rangle \rightarrow |1\rangle$

Agrandir (mais affaiblir) la batte

H $|0\rangle \rightarrow \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle$
$|1\rangle \rightarrow \frac{1}{2}|0\rangle - \frac{1}{2}|1\rangle$

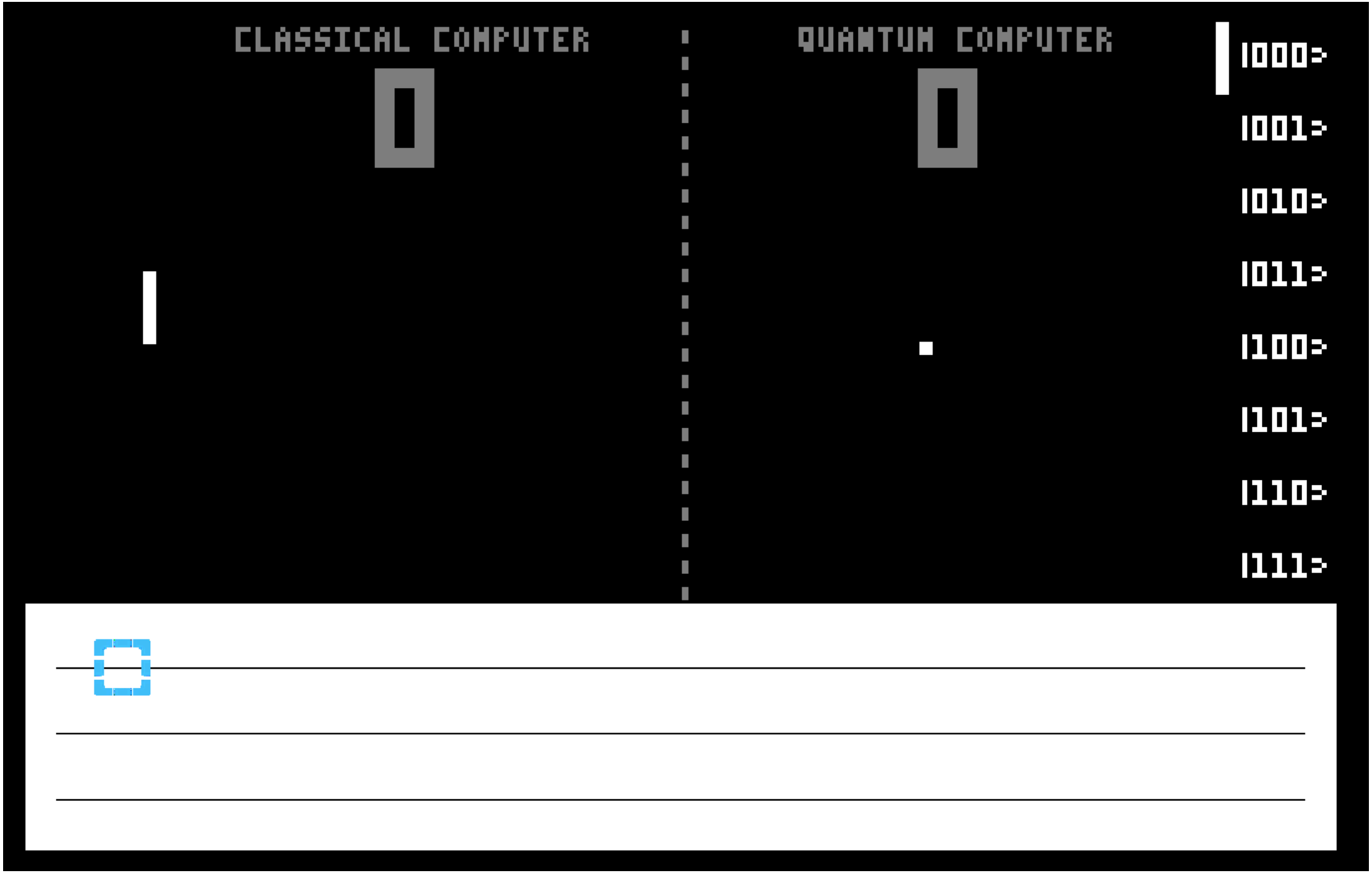
Y « comme X »

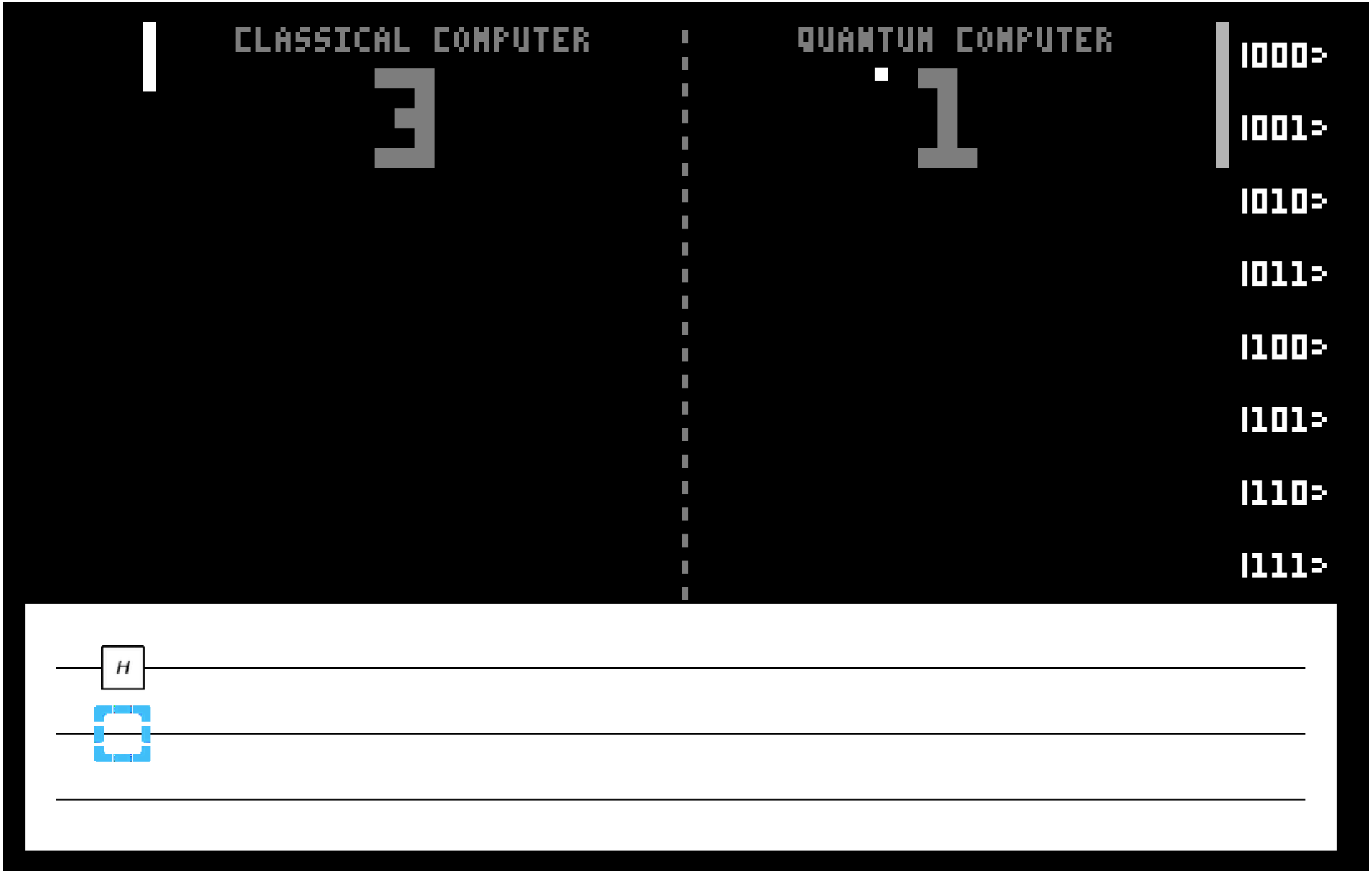
Z « rien »

Morceler la batte

CNOT:

C+⇧/⇩


CLASSICAL COMPUTER
QUANTUM COMPUTER
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>


CLASSICAL COMPUTER
3
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
H

CLASSICAL COMPUTER

5

QUANTUM COMPUTER

1

|000>

|001>

|010>

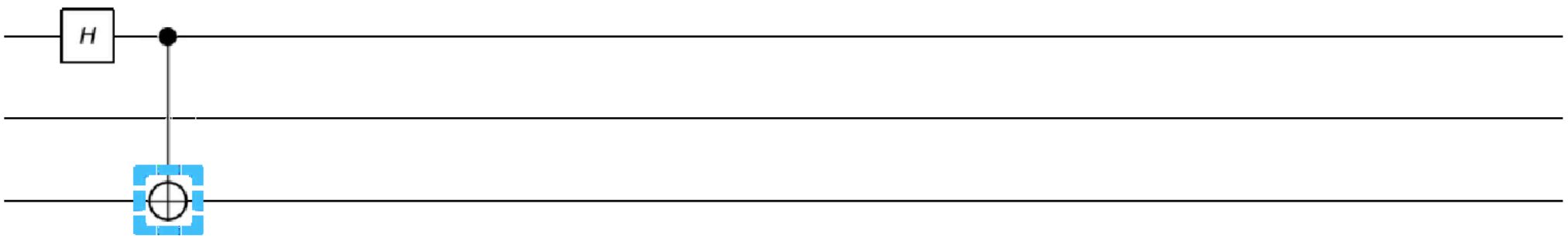
|011>

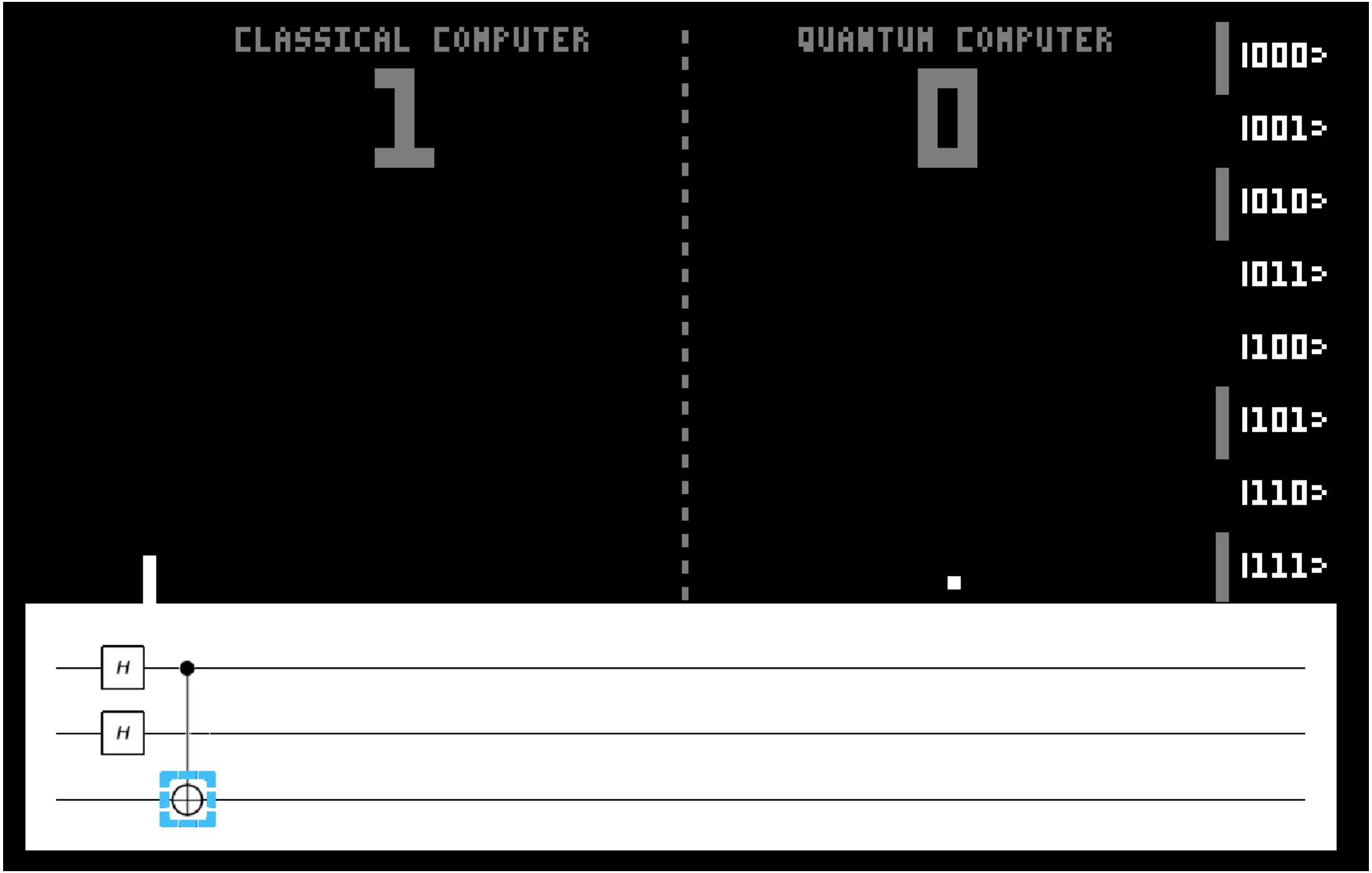
|100>

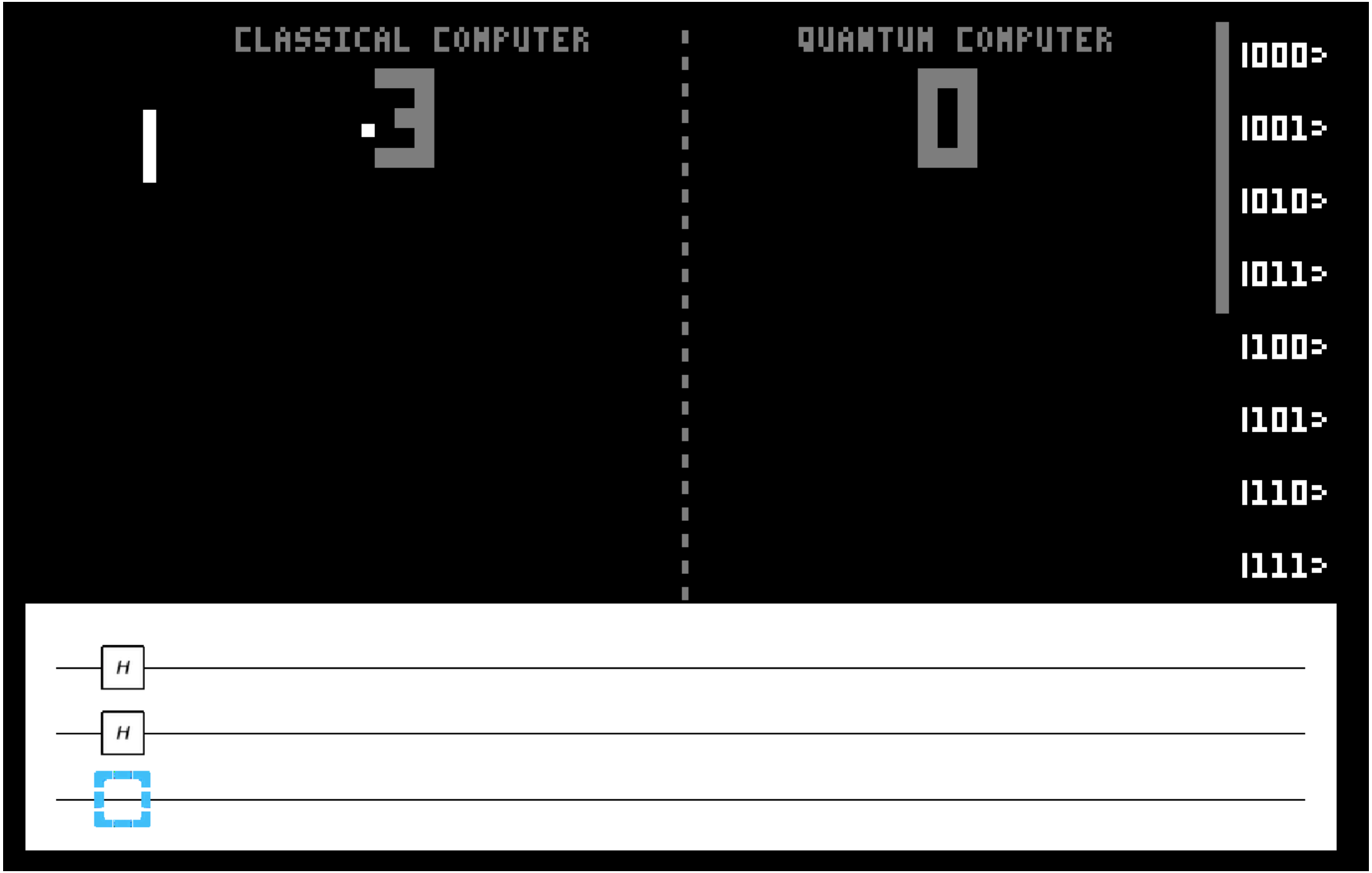
|101>

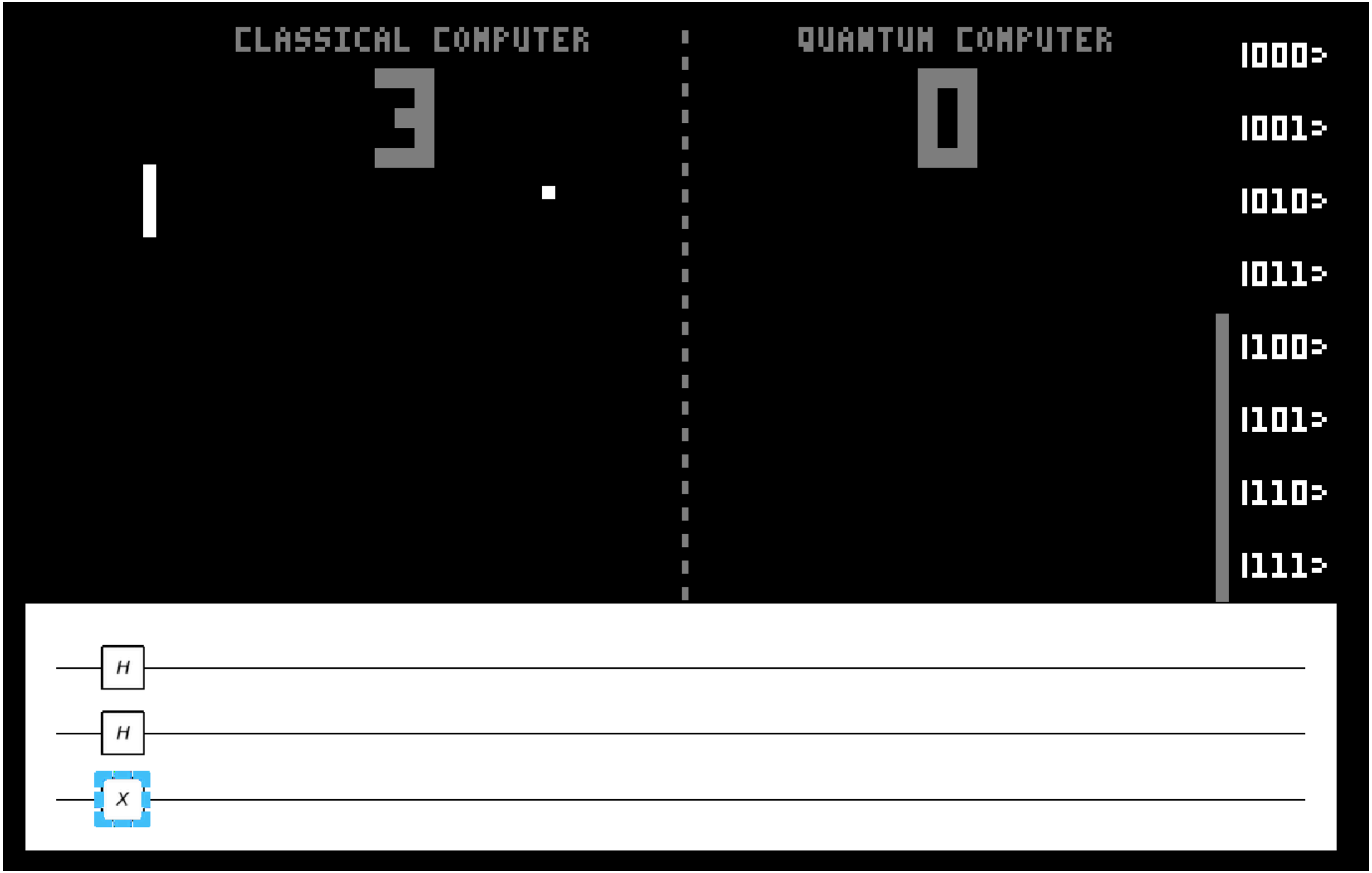
|110>

|111>




CLASSICAL COMPUTER
1
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
H
H


CLASSICAL COMPUTER
.3
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
H
H


CLASSICAL COMPUTER
3
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
H
H
X


Good
luck

Déplacer la batte

X « NOT »: $|1\rangle \rightarrow |0\rangle$
$|0\rangle \rightarrow |1\rangle$

Agrandir (mais affaiblir) la batte

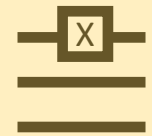
H $|0\rangle \rightarrow \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle$
$|1\rangle \rightarrow \frac{1}{2}|0\rangle - \frac{1}{2}|1\rangle$

Morceler la batte

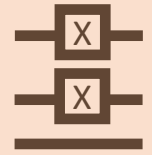
C CNOT:

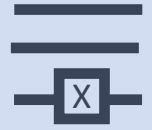
+⇧/⇩

$|000\rangle$ 

$|001\rangle$ 

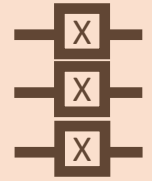
$|010\rangle$ 

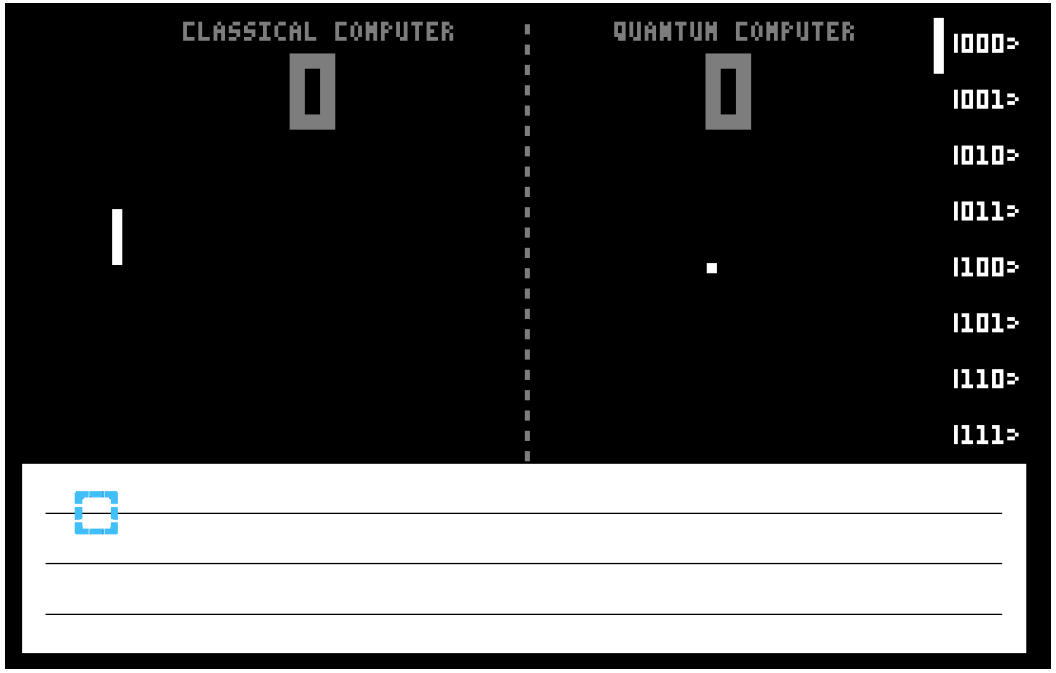
$|011\rangle$ 

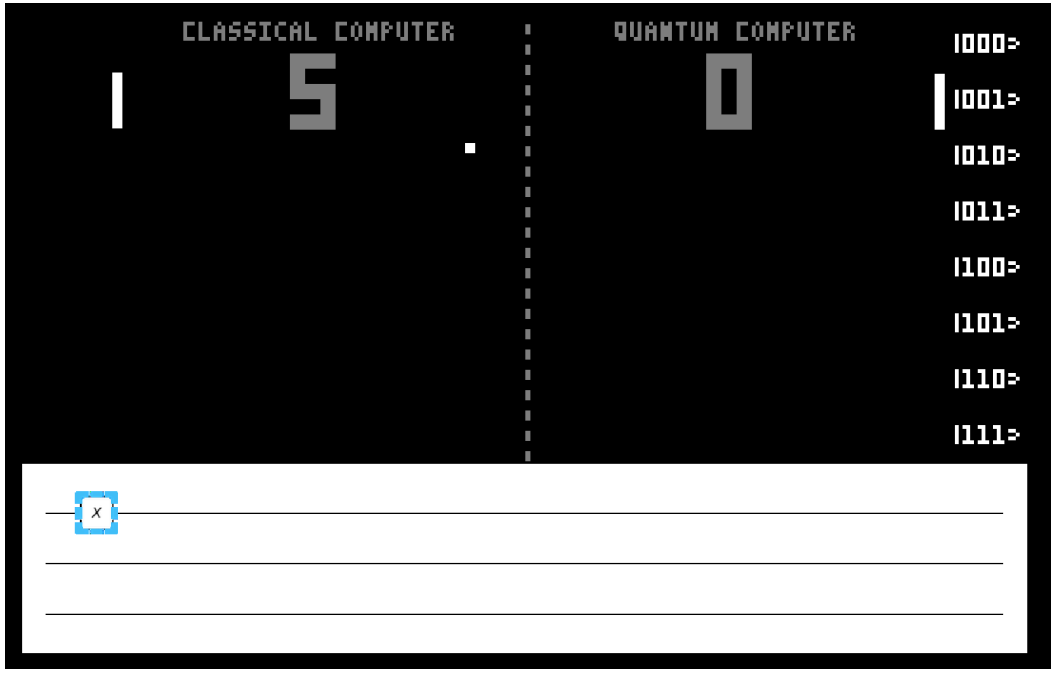
$|100\rangle$ 

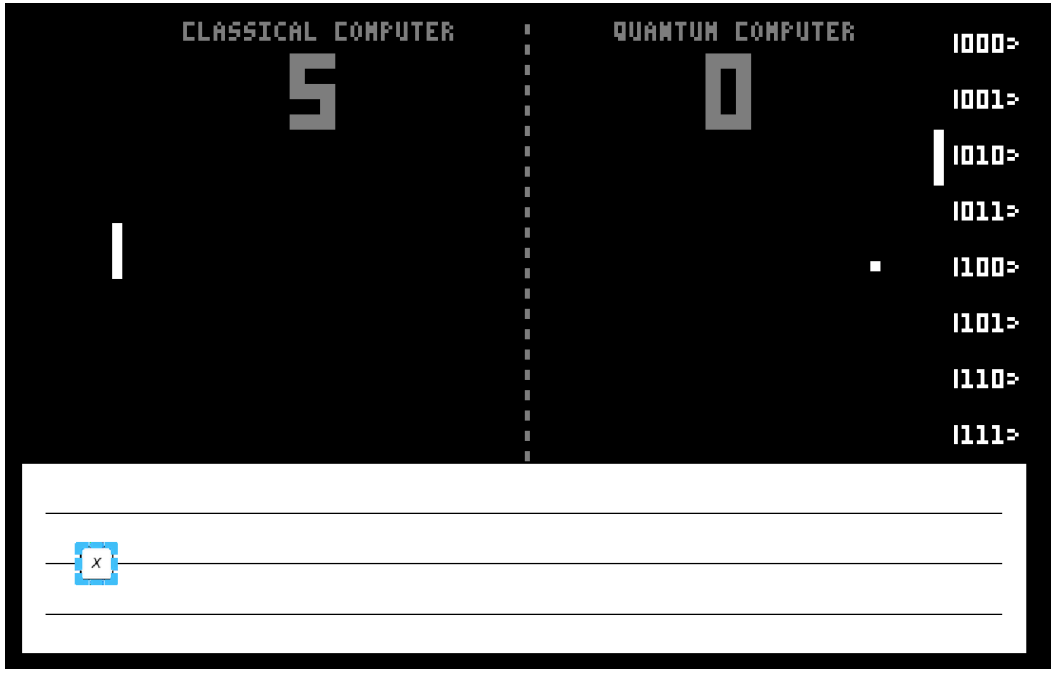
$|101\rangle$ 

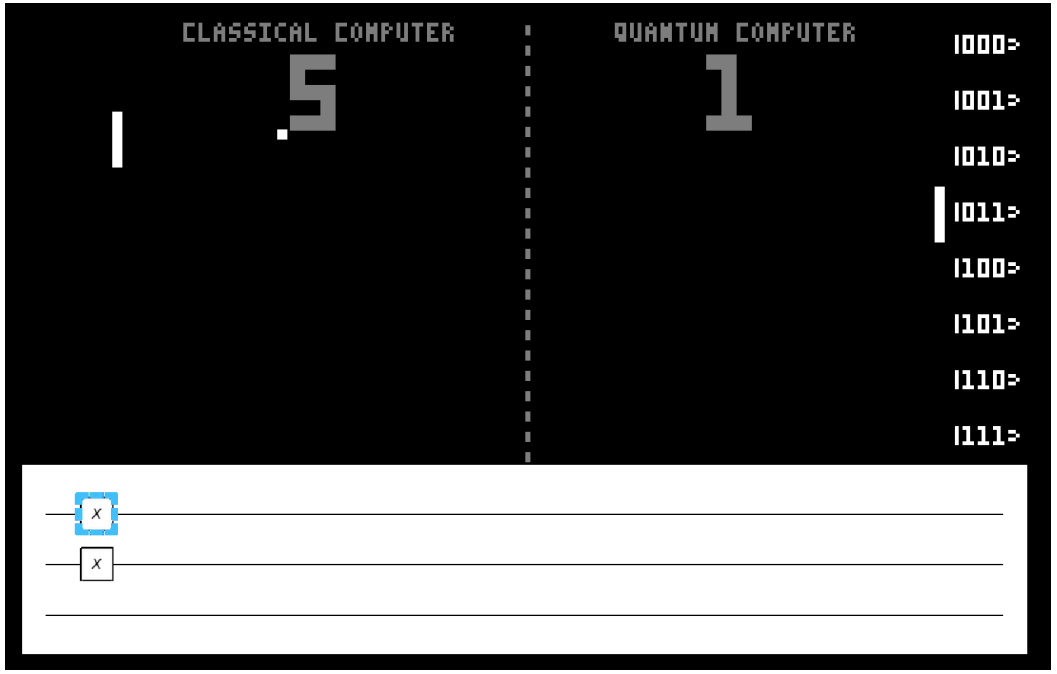
$|110\rangle$ 

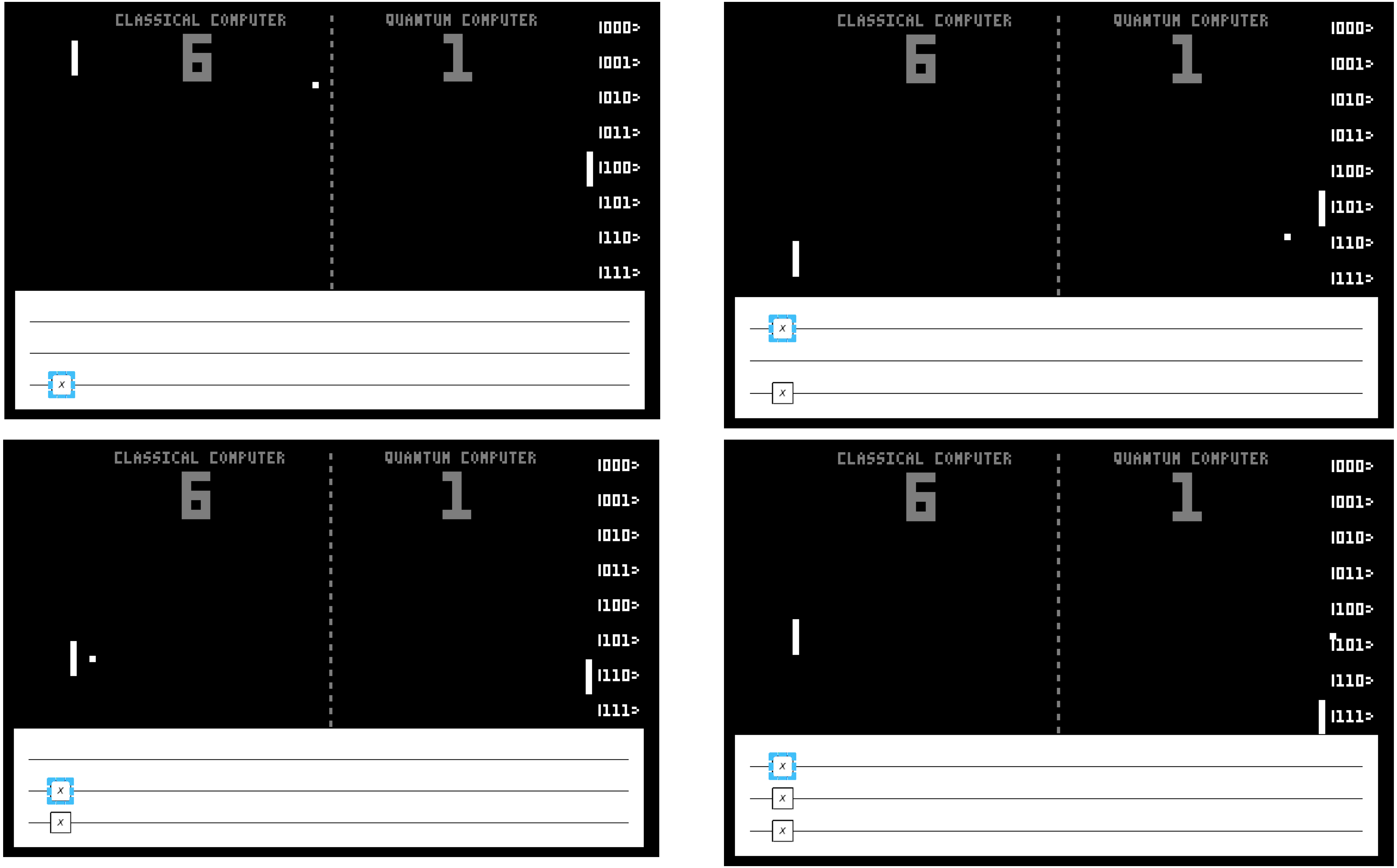
$|111\rangle$ 


CLASSICAL COMPUTER
0
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>


CLASSICAL COMPUTER
5
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X


CLASSICAL COMPUTER
5
QUANTUM COMPUTER
0
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X


CLASSICAL COMPUTER
5
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X
X


CLASSICAL COMPUTER
6
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X
CLASSICAL COMPUTER
6
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X
X
CLASSICAL COMPUTER
6
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X
X
CLASSICAL COMPUTER
6
QUANTUM COMPUTER
1
|000>
|001>
|010>
|011>
|100>
|101>
|110>
|111>
X
X
X