IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

PLAYING PONG GAME ON A QUANTUM COMPUTER

[1]Chaitanya Varma, [2]Bikash K. Behera, [3]Prasanta K. Panigrahi
[1]Undergradute Student, [2]PhD Student, [3]Professor
[1]Department of Physics,
[1]St.Stephen's College, University of Delhi, New Delhi-110007, India

Abstract: Pong, an arcade game, was initially designed as a training exercise and became one of the most popular games ever that paved the way for many of the personal-console games in the future. Here, we undertake the task of modeling the above game which can be played on a quantum computer. We create a static 3-space model of the game that follows a sequential approach and includes a user-based choice system on the positioning of the paddles. We develop new quantum circuits, design and simulate those, and verify the results using IBM quantum experience platform**.**

Index Terms **- Quantum Computation, Quantum Logic Gates, Pong Game, Sequential Game Theory, IBM Quantum Computer, Foosball.**

I. INTRODUCTION

Quantum computation and quantum information is the study of the information processing tasks that can be accomplished using quantum mechanical systems [1]. The information presented here will be well understood with a basic understanding of classical computation like what bits are, how logic gates work and basic knowledge of matrices and linear algebra. The basic unit of information processing in a modern-day computer is the bit, which can assume one of the two states: 0 and 1. How quantum computing differs from regular computing is that it uses qubits (short for "Quantum Bits") instead of bits. Unlike a bit, a qubit can exist in more than two states (Fig. 1).

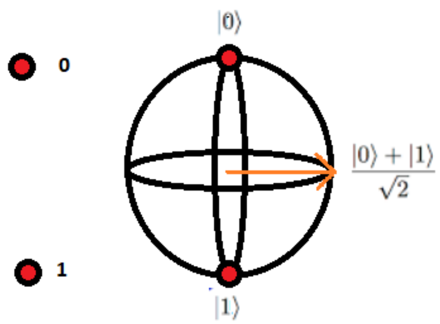


FIG. 1: Comparison between a classical bit and a qubit.

The states that a qubit can exist in are $|0\rangle$, $|1\rangle$, a superposition of these two states. The notation used here can be called a "state" or a "vector" or a "ket". The superposed state is somewhat special. For the superposed state given here [2]:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

Both α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. However, when measured, the qubit is seen either in $|0\rangle$ or in $|1\rangle$ state. Here, $|\alpha|^2$ and $|\beta|^2$ gives the probability of finding $|\psi\rangle$ in $|0\rangle$ or $|1\rangle$ state.

With all this basic information, and taking a brief look over some other papers on quantum games such as quantum Shooting game [3], quantum Bingo [4], quantum Sudoku [5], the solution to the Monty Hall problem [6], Diner's Dilemma [7] and quantum robots [8], quantum go [9], quantum tic-tac-toe [10], one realizes that some further knowledge about the algebra of quantum information is required. The next section will brief you about the basic knowledge of qubits, single-qubit and two-qubit gates.

II. BRA-KET NOTATION AND DIFFERENT TYPES OF PRODUCTS

Bra-Ket notation: The bra-ket notation is also known as Dirac notation. An elementary idea of this notation can be provided as follows: ⟨ c | v ⟩ is a "bracket"; now we may divide it into two parts as ⟨c| and |v⟩. It can be mentioned that a bra-state is a row vector while a ket-state is a column vector.

Different types of products

- An inner product of two states is a bra state (row vector) premultiplied to a ket state (column vector). This results in a single-digit answer as a row vector when premultiplied to a column vector resembles very closely to a dot product.
- An outer product of two states is treated as a ket state (column vector) premultiplied to a bra state (row vector). This results in a matrix with dimensions of the given vectors.
- A tensor product is defined as follows:

$$|c\rangle \bigotimes |v\rangle = |cv\rangle = \begin{pmatrix} w \\ x \end{pmatrix} \bigotimes \begin{pmatrix} y \\ z \end{pmatrix} = \begin{pmatrix} wy \\ wz \\ xy \\ xz \end{pmatrix}$$

It is to be noted that all the above products are carried out in Hilbert space. Before actual application on the IBM QE, here we can go through the following matrices that are used in designing the game.

$$H = \frac{1}{\sqrt{2}} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$CNOT(CX) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$CCNOT(CCX) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Here, H stands for Hadamard, X stands for Pauli-X (Not gate), CNOT (the blue two-qubit gate) stands for control Not, CCNOT/CCX (the purple three-qubit gate) stands for control-control Not gate. The above matrices are representations of the different gates used in our circuits, which have a comprehensive guide given on the documentation page of the IBM quantum experience (IBM QE) platform.

III. PONG

The Original Game. Pong is one of the earliest arcade video games. It represents a table tennis sports game using two-dimensional graphics. The game was originally manufactured by Atari and was released it in 1972. It was originally a training exercise for the creator based the idea on an electronic ping-pong game included in the Magnavox Odyssey [12]. A basic run of the program can be seen in the Fig. 2.

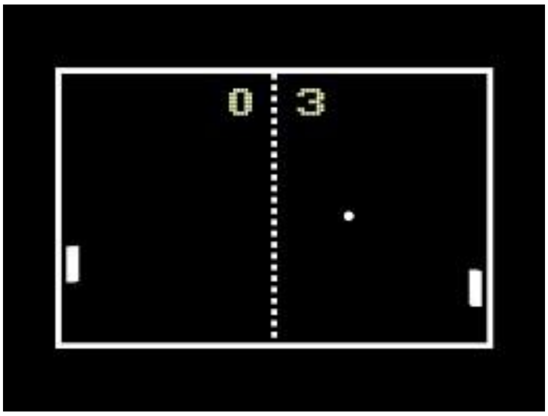


FIG. 2: A snippet of the actual game displaying the scores of the players. The player on the left has scored 0 and the player on the right has scored 3 points.

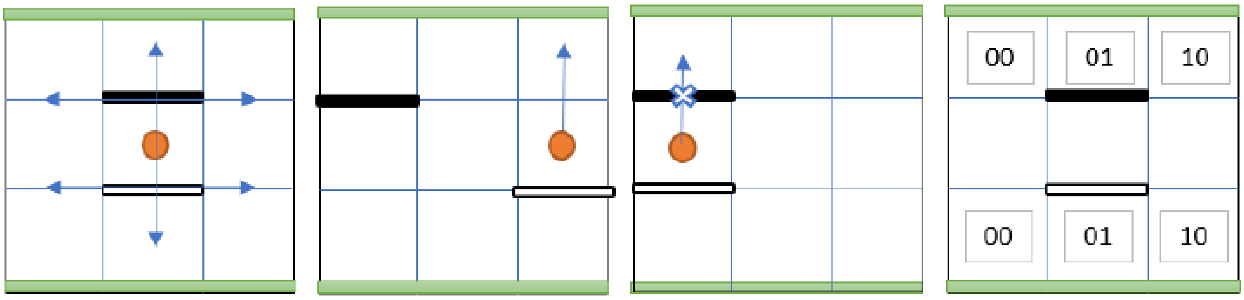
The goal of the game is to score a goal by making the ball cross the line covered by the opponent's paddle. It can also be seen as an oversimplification of the physical dynamic game of Foosball. The way we'll look at the game here will be a simplification of the game into a static sequential game. Even that gets complex as there will still be a large play-court where it's possible for the ball to turn and reflect at multiple angles. We use a 3 × 3 table which has two paddles instead, one white and one black.

Conversion into a sequential game:

Each paddle can move to either of the three spaces horizontally. We assume that the second paddle (the white one here) is controlled by the opponent (as it happens in many multiplayer computer games). That means that it's randomly placed in either of the three places. The aim of the game is to score a goal.
Again, for simplicity, the ball will only move vertically up and down (Fig. 3a). To do that the white paddle is required to shoot in either of the spots the black paddle isn't present. And for the ball to be hit by the paddle it must be at either of the three positions on the top that corresponds to the position of the ball. Now, it may confuse a few readers and include more qubits, so it was fixed that the position of the ball will always correspond to the position of the paddle in turn. Hence, that brings the outcome of a move in the game down to the positions of the two paddles. Therefore, in a particular chance/turn of play the white paddle, it must position itself in either of the two positions that the black paddle is not blocking (Fig. 3b, 3c).

Mathematical Modelling:

We assign the quantum states |00⟩, |01⟩, |10⟩ to the vectors in the 3 respective spaces (left to right) that each paddle will occupy (Fig. 3d). This results in a total of 9 outcomes shown in the Fig. 4.



(a) (b) (c) (d)

FIG. 3: Modeling 3×3 Pong into a sequential game. (a) The basic functionality of the elements in the game. (b) The hit from a paddle will be a goal if no obstruction ahead. (c) The hit from a paddle will not be a goal when the other paddle blocks it. (d) Labeling the three spaces the paddle can occupy.

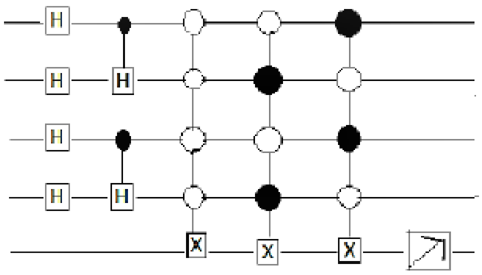
W	B	
00	00	T
00	01	W
00	10	W
01	00	W
01	01	T
01	10	W
10	00	W
10	01	W
10	10	T

B	W	
00	00	T
00	01	W
00	10	W
01	00	W
01	01	T
01	10	W
10	00	W
10	01	W
10	10	T

FIG. 4: The truth table based on the mathematical model.

The second truth table is an outcome of the change of turn which happens as a result of the cases in which the white does not win otherwise seen as a tie. This then leads to the three cases in which the black does not win and the chance/turn is passed onto the white paddle again. Therefore, one can logically say that the chances of winning of a player during his/her turn are higher than losing. This somewhat helps with modelling this into logic gates. Taking a look at the cases that have a lesser chance of winning, it may seem like a reference to Shannon's entropy in quantum information. It is done to design a specific circuit to measure the cases in which the paddle with the turn/chance loses. In all other cases, it is known that the paddle will win. Hence, the basic quantum circuit used to measure such cases is shown in Fig. 5.

FIG. 5: Quantum circuit illustrating the game. By measuring the last qubit, through the Z-measurement box (shown with an arrow



inside) we will know if any of the above combinations were triggered and the qubit was flipped.

Theoretical Circuit

One can divide the quantum circuit into two sections. The first part consisting of the Hadamard gates and control Hadamard gates. The first part creates the three superposed states required for the calculations here: for both W (applied on q[0] and q[1]) and B (applied on q[2] and q[3]). Hence both the paddles have an entangled state of (|00⟩ + |01⟩ + √2|10⟩) / 2, which is a superposition of the three states used to represent the paddle's position. The second part of both the paddles is responsible for checking their positions. It has three subsequent checks for the three-position placements in which the game end in a tie. They are shown in the same order as the cases are present in the truth table. The final circuit made on IBM quantum experience using the provided gates is shown in Fig. 6. It is to be noted that, the IBM QE has CCX gates which is a two-controlled operation, so to accommodate a 4-controlled operation, we use a series of CCX gates along with a few extra qubits. For anti-control operations (the white check in the theoretical circuit diagram), we use X gates on both the sides of the control qubit.

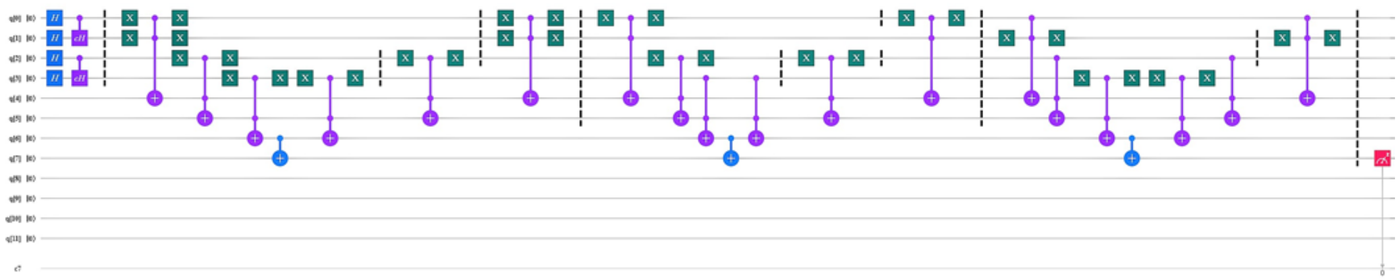


FIG. 6: Quantum circuit on the IBM QE. The extra qubits (q[5], q[6], q[7]) used here occur due to the fact we have 4 control qubits and the gates provided only has a double qubit control at most.

Sequencing

As mentioned earlier, we want to have a sequential approach at the game. Therefore, to incorporate that, we need to give the user choice on where they want to place their paddle. To perform this, the previously created 3 states have to be converted back to a position where we can place the paddle at |00⟩, and apply quantum gates on it to change the position to |01⟩ or |10⟩. We make use of the identity property of the Hadamard and control Hadamard gates, i.e. when reapplied on the qubits in a mirrored sequence, they reach back to the states they were initially. This is achieved in the quantum circuit shown in Fig. 7.

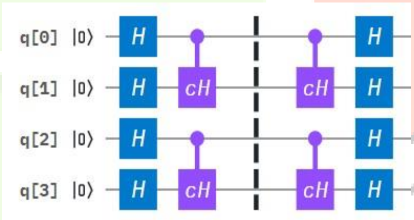
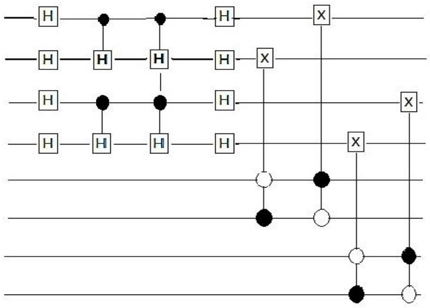


FIG. 7: The operations applied before the provision of choice. Here, Hadamard and control Hadamard operations on the qubits q[0], q[1], and q[2], q[3] help bring the state to |00i for the upper and lower paddle respectively.

Now that the states are back to |00⟩, we can use quantum controlled operations to set the conditions for positioning the paddle. This requires additional two qubits per paddle. The part of the circuit, that will be used to check the positioning of the paddles, is shown in Fig. 8.

FIG. 8: The choice provision circuit. The first CCX checks for the position |01⟩ and the second CCX checks for the position |10⟩ for both



the paddles.

Now, the paddles are at |00⟩ and the circuit diagram shown above (Fig. 8) takes care of |01⟩ and |10⟩. The full quantum circuit, demonstrating the modelling of the game is illustrated in Fig. 9.

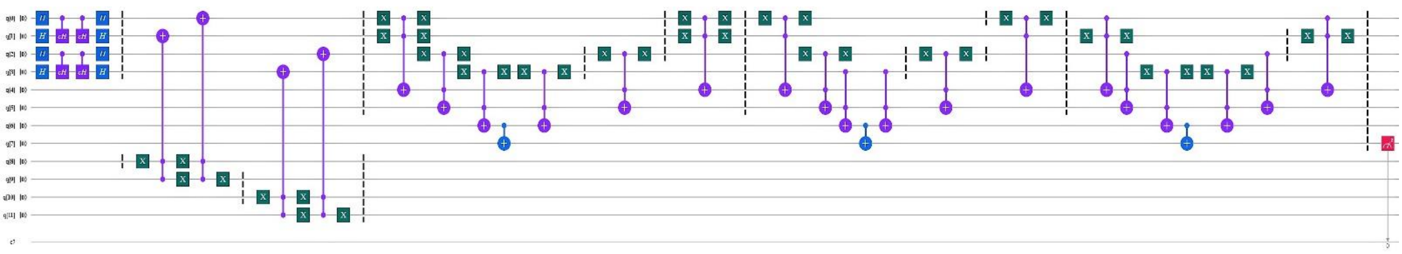


FIG. 9: The full quantum circuit. Here, qubits q[8] and q[9] hold the choice provided for the positions of the 1st paddle and qubits q[10] and q[11] hold the choice provided for the 2nd paddle.

To decide whether the game ended in a tie or not, we have to check the value of the (trigger) qubit used to provide the signal for the trigger in either of the three cases ending in a tie. This means that if the value of the qubit we measure comes out to be |1⟩ (Fig. 11), then there has been a tie else the player whose paddle is supposed to be in turn wins the point.

Circuit Examples

Here is a circuit in which the output will not be a tie as the first paddle is at |10⟩ and the second at |01⟩ (Fig. 10).

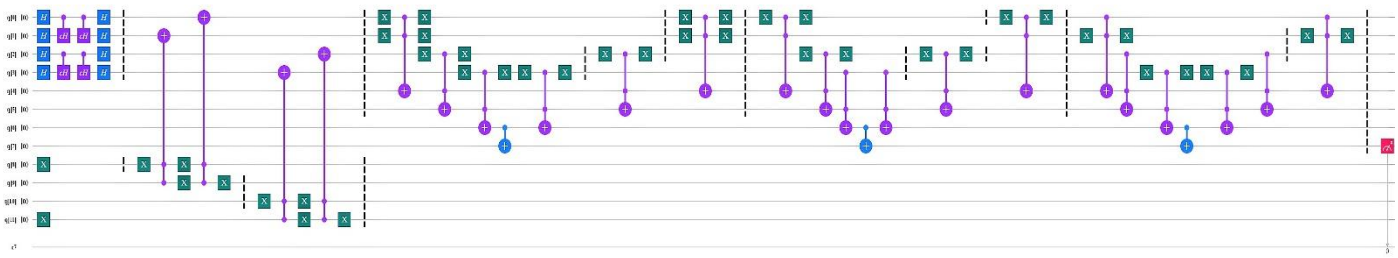


FIG. 10: Full quantum circuit with a variable choice. Here, q[8] and q[9] set the position |10i for the 1st paddle and q[10] and q[11] set the position |01⟩ for the 2nd paddle.

Results

IG. 11: A result snippet from the tie case. Output: |1⟩

FIG. 12: A result snippet from the goal case. Output |0⟩

IV. Conclusion

To conclude, we have modelled a static 3 × 3 Pong game by developing a quantum circuit which can be designed on a quantum computer to be played with. Pong, although is a dynamic non-sequential game, here it was converted into a sequential game to be easily modelled with the help of quantum circuits. We were able to provide the user with the choice to position the paddle themselves. We have designed the quantum circuits on the IBM quantum experience platform and verified the performance of the game. Our approach presents the scope for further research where it can be modified to be a random position checker on this 3 × 3 rendition of Pong. The same approach then can be applied to extend the model to n × n system.

V. Acknowledgments

C.V. would like to thank IISER Kolkata for providing the right push and support for being able to gain some research experience. The author would like to acknowledge his guardians, teachers, and friends (some of them). B.K.B. acknowledges the financial support of IISER-K Institute fellowship. The authors would like to acknowledge the support of IBM Quantum Experience for helping produce experimental results. The views expressed do not represent IBM's policies.

References

[1] A. Nielsen and I. L.Chuang, Quantum Computing and Quantum Information by Michael and Nielsen, (2011).
[2] D. McMahon, Quantum Computing Explained by David, (2008).
[3] B.B.Roy, B. K. Behera, and P. K. Panigrahi, Modelling A Simple Shooting Game Using Quantum Computation, DOI: 10.13140/RG.2.2.30976.07680.
[4] V. Singh, B. K. Behera, and P. K. Panigrahi, Design of Quantum Circuits to Play Bingo Game in a Quantum Computer, DOI: 10.13140/RG.2.2.22727.34720.
[5] A.Pal, S.Chandra, V. Mongia, B. K. Behera, and P. K. Panigrahi, Solving Sudoku Game Using Quantum Computation, DOI: 10.13140/RG.2.2.19777.86885.
[6] S. Paul, B. K. Behera, and P. K. Panigrahi, Playing Quantum Monty Hall Game in a Quantum Computer, DOI: 10.13140/RG.2.2.22315.49442.
[7] A. Anand, B. K. Behera, and P. K. Panigrahi, Solving Diner's Dilemma Game, Circuit Implementation and Verification on the IBM Quantum Simulator, DOI:10.13140/RG.2.2.28940.05765.
[8] S. Mahanti, S. Das, B. K. Behera, and P. K. Panigrahi, Quantum Robots Can Fly; Play Games: An IBM Quantum Experience, DOI: 10.13140/RG.2.2.28795.62241.
[9] A.Ranchin, Quantum Go, arXiv:1603.04751 [quant-ph].
[10] A. Goff, Quantum tic-tac-toe: A teaching metaphor for superposition in quantum mechanics, DOI: 10.1119/1.2213635.
[11] A. Pathak, Elements of Quantum Computations and Quantum Communication by Anirban, (2013).
[12] URL: https://en.wikipedia.org/wiki/Pong.