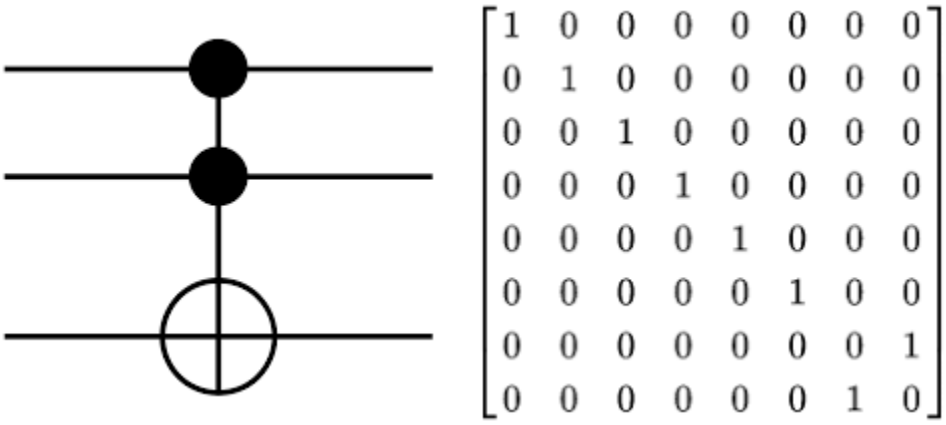
My journey through the IBM Quantum Challenge 2021 - Part 1

May 20th marked the beginning of one of the most awesome challenges in the field of Quantum Computing. The IBM Quantum Challenge commemorated the history of quantum computing, as this year marks the 40th anniversary of the Physics of Computation Conference, where the concept of computing systems based on quantum mechanics was conceived. Each of the five challenges was built on the Field's historical stepping stones.
This blog post will take you on a journey about what challenges there were and how I came up with the solutions that bagged me one of the top places in the Challenge. Unlike athletic kids I've never really liked challenges, but boy was my mind changed with attempting the first Question.

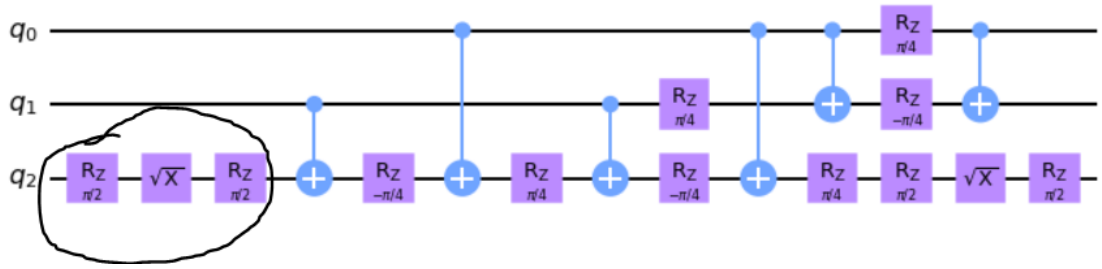
Challenge 1 : Toffoli Gate

Linear algebra is an area of mathematics that forms the basis for quantum computing. Unitary Hermitian matrices can be used to represent all of the Quantum Logic gates. These matrices have a unique property: they are reversible.
In 1981, Thomas Toffoli proposed the Toffoli Gate, a reversible variant of the AND and NAND gates. Reversible gates are more energy efficient and, as previously said, are required for the advancement of Quantum Computation.



Toffoli Gate along with its matrix representation (source1, source2)

The initial task was to build the Toffoli gate using only the most basic quantum gates. This required us to use matrix decomposition to break down the large matrix you see above into smaller pieces. To be honest, I thought the first exercise would be simple, but it took a surprising amount of time.
To be honest, I pushed through with trial and error and was able to complete the circuit shown below in about a day. Another issue was not being able to use the Hadamard gate, but I had already solved that before moving on to the more decomposition of Toffoli gate. So, at a cost of 73, I finally finished the first problem.



The circuit for ccx(the gates marked with pen actually show the Hadamard gate)

Challenge 2 : Shor's algorithm

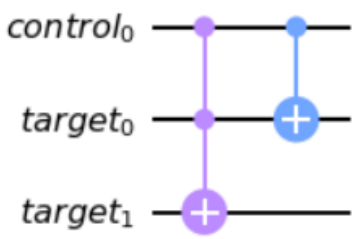
Shor's algorithm was the first algorithm to prove that indeed quantum computers are something special. The approach uses Quantum Phase Estimation to find periods and can be used to factor big prime integers. This will eventually undermine the RSA encryption method, which is based on the difficulty of factoring large prime integers.
The task was to factor the number 35 using only five qubits. This may appear difficult, but half of the materials have already been provided. Our job was to find a unitary U that performs the state transformation; take a look at the question for yourself.

Exercise 2a: Create a circuit (U) that performs the transformation:

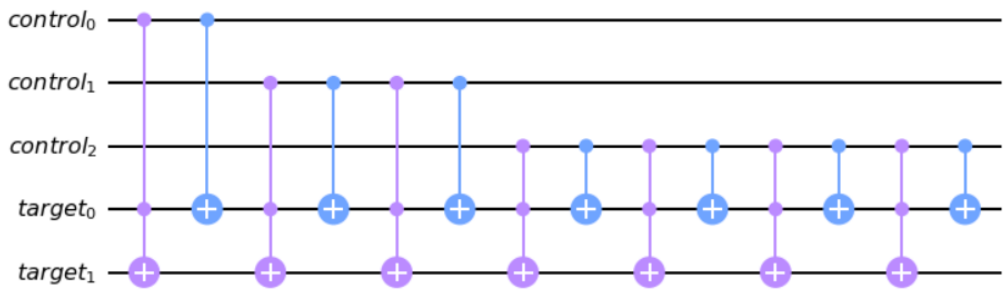
$$U|00\rangle = |01\rangle$$
$$U|01\rangle = |10\rangle$$
$$U|10\rangle = |11\rangle$$
$$U|11\rangle = |00\rangle$$

and is controlled by another qubit. The circuit will act on a 2-qubit target register named 'target', and be controlled by another single-qubit register named 'control'. You should assign your finished circuit to the variable '`cu`'.

This again required me to create a matrix and it took some time but finally I was able to get the unitary which worked fine.



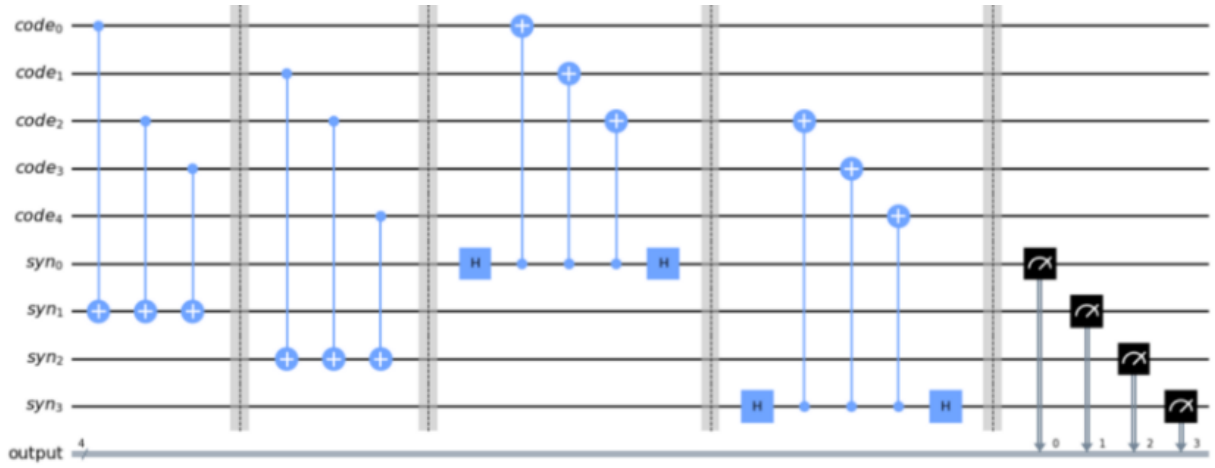
The next step was to get the powers of 2 of this matrix(U, U^2, U^4) and because this question was not graded I simply repeated the circuit 2 and 4 times, and got my final answer.



Although my score sucked(it was 35). I still passed.

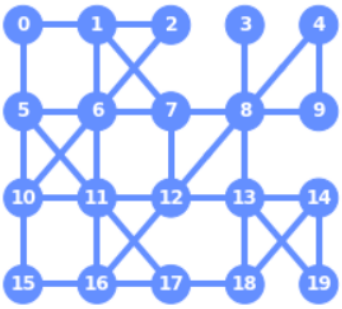
Challenge 3: Error Correction

Quanum Devices are very much prone to noise and error, which means making the hardware to run these devices is very difficult. Quantum Error Correction is the branch of Quantum Computing which actually deals with correcting the error using software, this was also introduced by Peter Shor. The Error Correction works by using error correction codes which use multiple qubits to represent one qubit value, so if one or two qubit's state is changed due to error the overall state that we wanted to represent does not change.
The third challenge was using an actual quantum computer (ibmq_tokyo) to apply surface error correction coding. We already had a functional code from the challenge creators. The only thing left to do was make sure it worked on ibmq_tokyo.



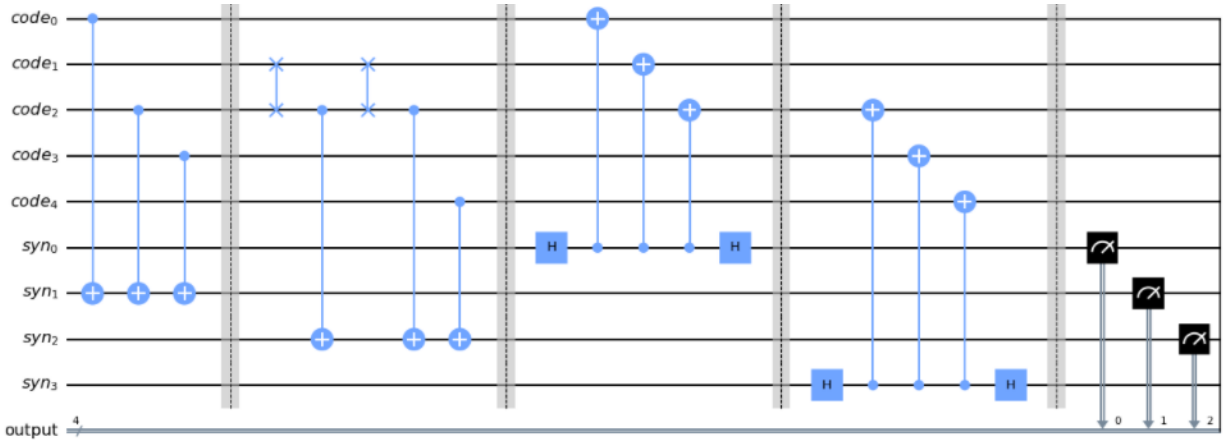
The initial surface code circuit provided by ibm

The first question that comes to mind is why wouldn't the circuit work on ibm's device? When we run a circuit, there is a transpilation mechanism that takes place in the background and maps the circuit to the device. Because all of the qubits on the physical device aren't connected, you can't use a CNOT gate to connect them. To make connectivity possible, the transpiler inserts swap gates, which is exactly what we had to do.



Ibmq_tokyo backend

The image above is a representation of how the device appears. And we'll be using 9 qubits from this(from the surface code provided by ibm) . 0,1,2,5,6,7,10,11, and 12 were the qubits I chose. There are only two qubits which do not have the required connection so I used swap gates in order to overcome this.



The final circuit after the correction I made

Two days were finally over and I had also completed the third section with a score of 266.

……

The blog was getting a bit too long so let's continue in part 2

Resources

1. https://medium.com/qiskit/ibm-quantum-challenge-2021-heres-what-to-expect-65a303753ffb
2. Most of the resources and images come from the jupyter notebook of the tests