Three-Qubit Bit-Flip Error Detection and Correction

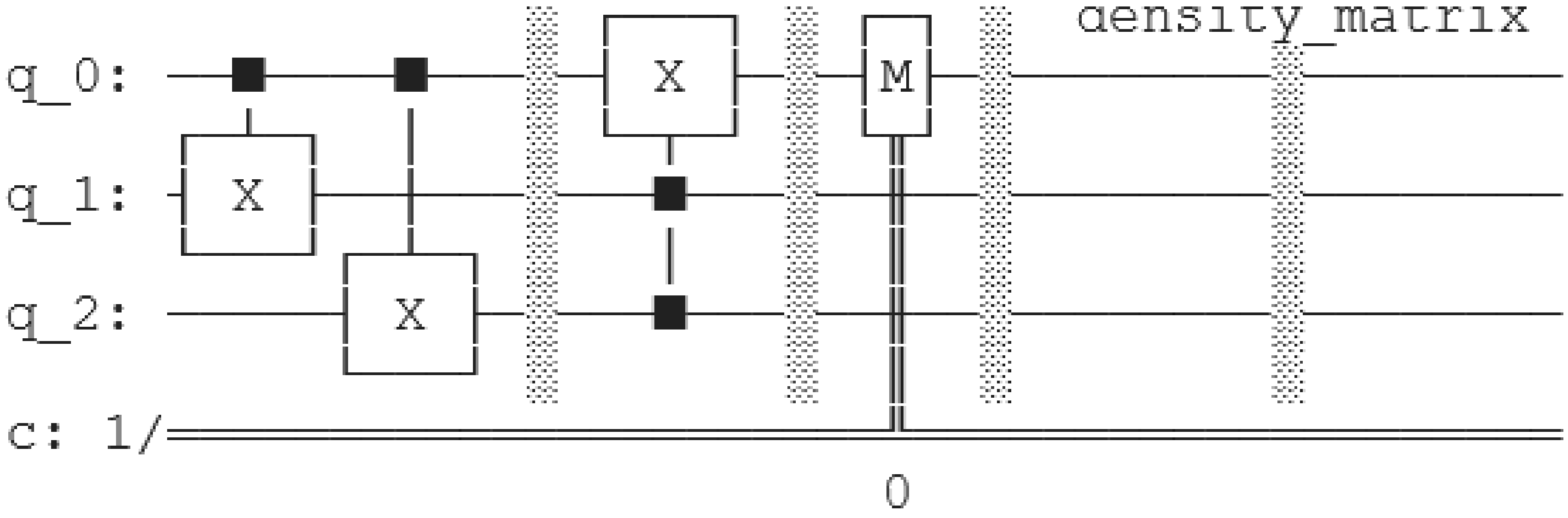
by Landon Greer

Error Correction Overview

- In computing its possible for some kind of interference or noise to alter the state of bits. This is especially true for quantum computing as entanglement is very finnicky and very easy to break.
- Many things can cause noise in quantum systems such electromagnetic frequencies, fluctuations in the temperature, and other parts of the environment. This means that qubits are especially susceptible when being manipulated by gates.
- It is also possible for quantum particle to lose its coherence (quantum properties) from the same causes.

Majority Vote Model

- The majority vote model uses two c not gates as an encoding scheme. Q0 controls Q1 and Q2. Once the information is encoded, a Tofolli gate is used to not Q0 with two control conditions being Q1 and Q2. This is why its called majority vote, it just sets the value Q0 to whichever has more occurrences, 0 or 1.



Majority Vote Pros and Cons

- Pros:
  - Very simple and small circuit
  - Very effective for single bit flip errors
- Cons:
  - Cannot detect where errors occurred
  - Cannot scale easily for different flips
  - Cannot detect multiple errors

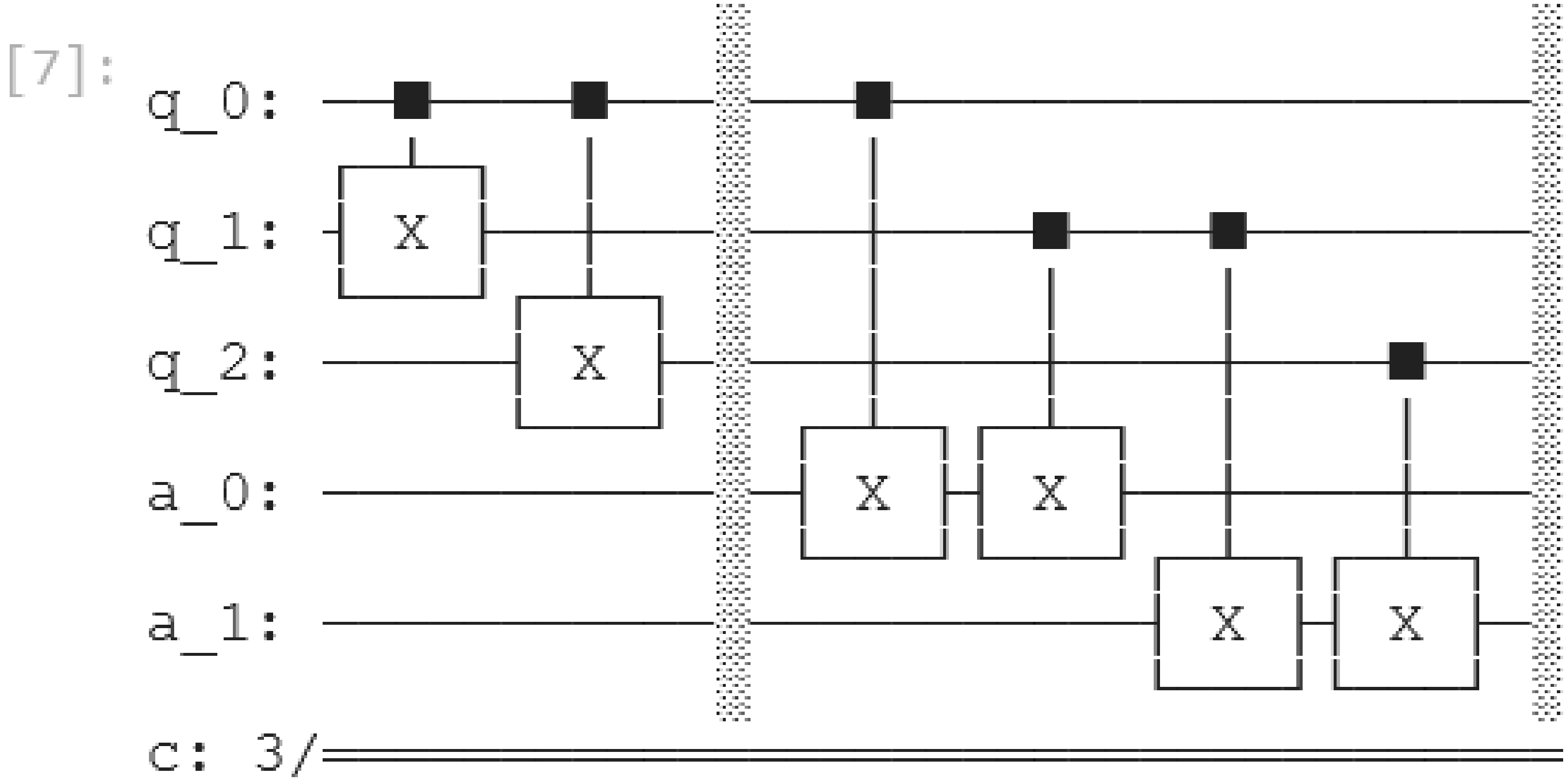
Syndrome Extraction Overview

- The key idea of the syndrome extraction method is that we use additional quantum bits called ancilla to decode error correction
- These are used so that we can get determine locations of errors without destroying the quantum properties of the circuit. I think of them almost like a pointer. Given that they are qubits though, they collapse after measurement.

Syndrome Extraction Method: Detecting Errors

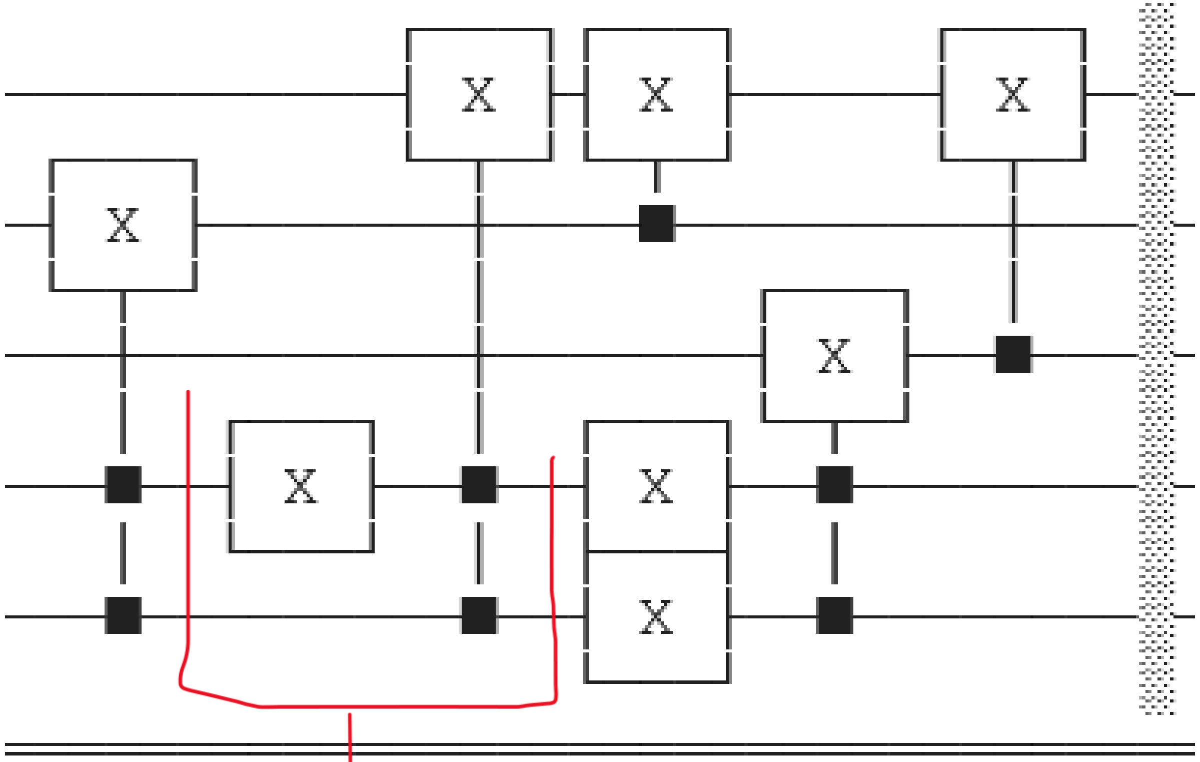
- Data is Q0, Q1 and Q2 are used to ensure no data was altered.
- The syndrome extraction method also uses two cnots to encode Q1 and Q2 just like majority vote.
- It then cnots Ancilla0 based on Q0 and Q1 and Ancilla1 based on Q1 and Q2. This means we can locate where bits flip based on the syndrome(value of the ancillas).

A look at detecting errors



If q1 flipped then a0 would become 1 and a1 would become 1. 11 informs us that q1 flipped. Similarly, 01 would mean q2 flipped and 10 would mean q1 flipped. As you might have guessed, this method breaks when multiple bits flip as you cannot locate the error. This is actually the same problem error correction faces in classical computing.

Correcting Errors based on Syndrome



~A0&&A1

To correct the bits that are flipped, you really can just use classical AND operations. In a quantum circuit that looks like this where Toppoli gate is used as an AND gate. Flip gates are used here to not for the proper operation:
To flip q2, the first ancilla should be 0 and the second ancilla should be 1. To do ~A0&&A1, we flip A0 bit, it then needs to be flipped back for the next check.

Syndrome Extraction Pros and Cons

- Pros:
  - Can determine where the error was:
  - Can be extended for both bit flip and phase flip
  - Alterations are used for more efficient error detection and detecting multiple errors. Algorithms like Shor’s are built on this.

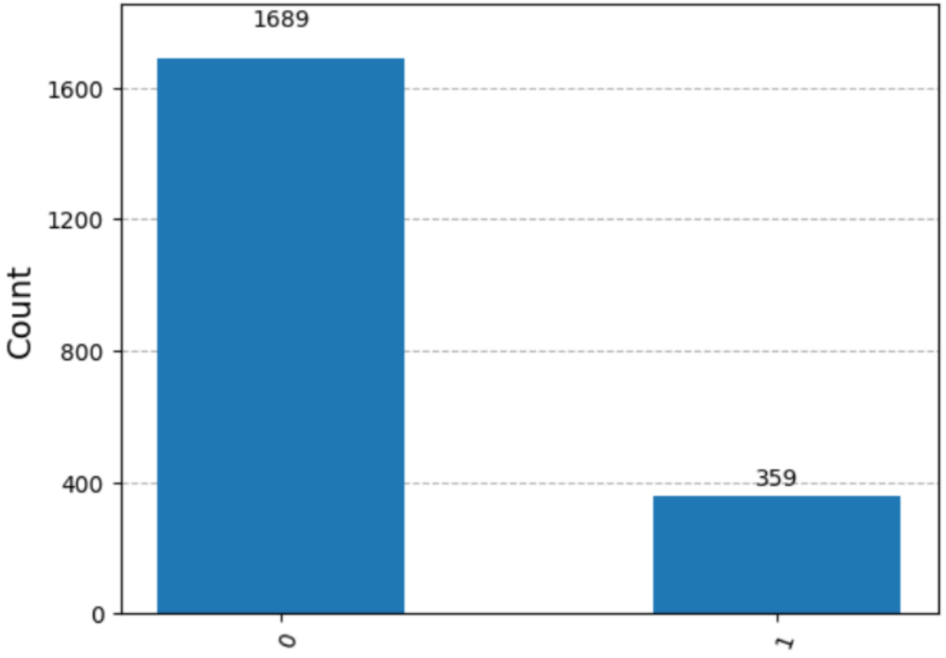
- Cons:
  - Much more complex circuit. Higher amount of gates means higher chance of decoherence.
  - This implementation can only fix one bit-flip error

Transmission Noise Results

- Initially I implemented a model that applied noise to all gates. This was a bit counterintuitive because the gates for solving the bit flip were then getting flipped. To fix this I created a separate model that only applies noise during transmission phase. This was represented by identity gates. Here is the resulting data:

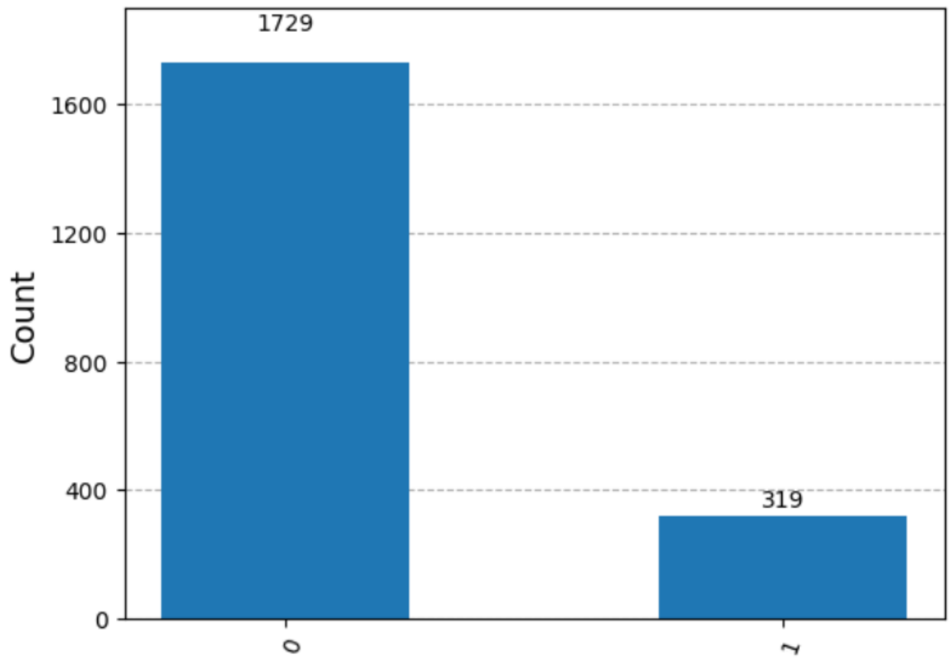
This is the value of q0. 0 is what is desired here and a 1 means that error was not corrected. This probably means that multiple bits were flipped.

The flip chance was 10 percent per qubit during transmission

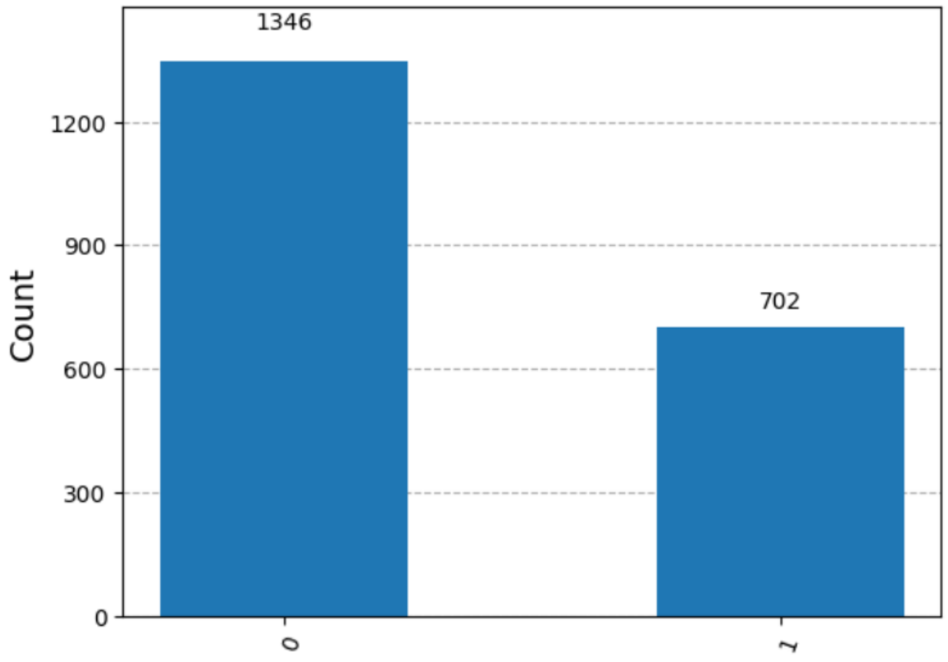


Results of noise on all gates:

Majority Vote Results



Syndrome Extraction results



End of Presentation