Bachelor Thesis
Decoding the Color Code

Clemens Schumann,
Advised by Peter-Jan Derks

March 22, 2023

Contents

1 Overview of QEC algebra

A quantum computer operates on so-called *qudits*, which can be any multi-level quantum system. Physical implementations of these include particles with spin, as well as controlled EM waves, i.e. lasers.
In this thesis, we will focus on *qubit*-based systems, i.e. two-level quantum systems as base units of computation.

In the following, we will analyze quantum circuit diagrams using the different pictures of quantum mechanics. A quantum circuit diagram is a visual representation of the computation done in a quantum computer, whereby:

- States progress in time along horizontal parallel lines
- Time goes from left to right
- Gates denoted X,Y,Z are the single qubit Pauli operators $\sigma_x, \sigma_y, \sigma_z$
- Gates can act on one or multiple qubits, whereby an X gate on qubit 1 in a 3-qubit system should be interpreted as:
  $(X \otimes \mathbb{I} \otimes \mathbb{I})(|\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle)$

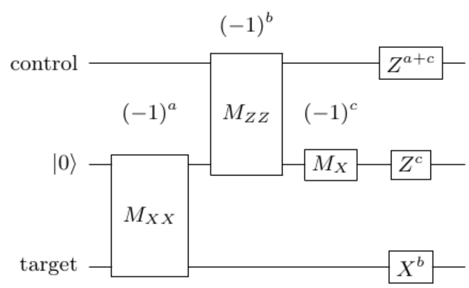

Figure 1: A Quantum Circuit to implement a measurement based Controlled-$X_{|\psi\rangle_{control} \to |\psi\rangle_{target}}$ Gate, where $|0\rangle$ is the $+1$ eigenstate in σ_z-basis.

1.1 Schroedinger picture

In the Schroedinger picture, we focus on the time evolution of qubit states:

$$|\psi\rangle = |\psi(t)\rangle \tag{1}$$

Measurements project these states onto eigenstates of the measurement operators via a projection P, so:

$$P_M^{\pm}|\psi\rangle = \frac{(M \pm \mathbb{I})|\psi\rangle}{2} \tag{2}$$

Where M is a matrix representation of the physical observable to be measured. For example, a measurement of a single qubits spin along the z-axis would be represented as:

$$M_Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{3}$$

And that measurement would perform a projection P_Z:

$$P_Z^+ = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} or P_Z^- = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{4}$$

on the state, depending on whether the measurement result yielded $+1$ or -1.

Therefore, to calculate the output of a quantum circuit in the schroedinger picture, simply apply the measurements and gates on the input states .

As can be seen explicitly calculated in the Schroedinger picture in Appendix 5.1 , the circuit from figure 1 implements a CNOT gate from the control qubit to the target qubit.

We will now analyze this circuit in the Heisenberg picture, finding that it results in an equal output.

1.2 Heisenberg picture and stabilizer formalism

1.2.1 Stabilizer group

We call an operator/gate S, to which the input state is an eigenvector ($S|\psi\rangle = |\psi\rangle$), a *stabilizer* of that input state. For n-qubit systems, we write these stabilizers as n-tensor-products of pauli operators $P \in P_G$, where P_G is the group generated by the Pauli operators and the Pauli operators are the operators on $\mathbb{F}_2$ such that:

$$\forall P \in P_G : P^2 = \mathbb{I}. \tag{5}$$

In the Heisenberg picture, stabilizers are tracked instead of states. The stabilizer group S_G is the group generated by the set of stabilizers:

$$S_G = \langle S_0, .., S_n \rangle : S|\psi_{in}\rangle = |\psi_{in}\rangle \forall S \in S_G \tag{6}$$

So for the example in Figureit is the group of operators to whom $|\psi_{control}\rangle \otimes |0\rangle \otimes |\psi_{target}\rangle$ is an eigenstate, namely $\mathbb{I} \otimes Z \otimes \mathbb{I}$ (and trivially $\mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I}$, which we choose to ignore as a stabilizer since any three-qubit state is stabilized by it, and it can be generated by squaring any stabilizer constructed through tensor products of Pauli matrices).

A stabilizer group is always an abelian group i.e. its elements commute, since if:

$$\forall A, B \in S : AB|\psi\rangle = BA|\psi\rangle = |\psi\rangle \Rightarrow [A, B]|\psi\rangle = 0 \tag{7}$$

1.2.2 Effect of gates on stabilizers

To determine the effect a gate operation A has on a stabilizer, consider the following:

If $S|\psi\rangle = |\psi\rangle$ then:

$$A|\psi\rangle = AS|\psi\rangle = AS\mathbb{I}|\psi\rangle = \underbrace{ASA^\dagger}_{=S'} A|\psi\rangle \tag{8}$$

So we now know that the post-gate state is an eigenstate of S'.

Therefore $S'_G = \langle AS_0A^\dagger, ..., AS_nA^\dagger \rangle$.

1.2.3 Effect of measurements on stabilizers

A Pauli measurement operator M can either commute with all stabilizer operators, in which case M itself is a stabilizer already. In this case the measurement has no effect on the state, since the measurement of a stabilizer projects onto identity. Otherwise it can anticommute with at least one operator in S_G, since Pauli operators as well as their tensor products can only commute or anti-commute with each other. The product of two operators that both anticommute with another operator will then commute with that operator.

So in order to obtain the new stabilizers S'_G:

1. Identify $S \in S_G : \{S, M\} = 0$
2. Remove S from S_G
3. Add M to S_G
4. for $N \in S_G \cup \overline{X} \cup \overline{Z}$: if $\{N, M\} = 0$: replace N with SN

where $\overline{X}$ and $\overline{Z}$ are the sets of logical X and Z operators respectively. (a logical operator is an operator which acts on a systems metastructure which can be treated as its own qubit)

1.2.4 Circuit Analysis in Stabilizer formalism

After a measurement M, an n qubit input state will always collapse into either the +1 or the -1 eigenstate of the measurement operator.

In the first case the acting measurement operator becomes $\mathbb{I}^{\otimes n} + M$, in the second it becomes $\mathbb{I}^{\otimes n} - M$. Therefore, in the circuit shown in figure 1 , the measurements project onto:

$$P_1^{\pm} = \frac{1}{2}\left(\mathbb{I}^{\otimes 3} \pm \mathbb{I} \otimes X \otimes X\right) \tag{9}$$

$$P_2^{\pm} = \frac{1}{2}\left(\mathbb{I}^{\otimes 3} \pm X \otimes X \otimes \mathbb{I}\right) \tag{10}$$

$$P_3^{\pm} = \frac{1}{2}\left(\mathbb{I}^{\otimes 3} \pm \mathbb{I} \otimes X \otimes \mathbb{I}\right) \tag{11}$$

In the following stabilizers will be written without the tensor product symbols, so in our case the stabilizer is initially: $S_G^0 = \langle IZI \rangle$, the logical $\overline{X}$ operator is XXX and the logical $\overline{Z}$ operator is ZIZ.

After the first measurement, the state is stabilized by IXX, since it collapses into an eigenstate of the measurement operator. Notably, if the measurement operator M anticommutes with some element of the stabilizer S:

$$SP_-S^\dagger = \frac{1}{2}S\left(\mathbb{I}^{\otimes 3} - M\right)S^\dagger = \frac{1}{2}\left(\mathbb{I}^{\otimes 3} + M\right)SS^\dagger = P_+ \tag{12}$$

So by applying an anticommuting previous stabilizer operator after the measurement one can ensure that the state is in the P_+ projected state $P_+|\psi_{init}\rangle$ (in short, +1 and -1 eigenstates have the same stabilizers if we add conditional gates accordingly).

For the logical operators, if $\overline{X}$ or $\overline{Z}$ do not commute with the measurement operator, we know that their product with another anticommuting operator from the previous stabilizer must then commute with the measurement operator: $[S_{prev}\overline{X}M, MS_{prev}\overline{X}] = 0$ (recall the previous statement that all Paulis and their tensor products must either commute or anti-commute).

In our case, IZI and IXX anticommute, so now the state is stabilized by $S_G^1 = \langle IXX \rangle$. Both initial logical operators commute with the first measurement operator, so they are left unchanged.

After the second measurement M_2=ZZI, since this measurement anticommutes with the IXX stabilizer, the new stabilizers are: $S_G^2 = \langle ZZI \rangle$. The logical $\overline{X}$ and $\overline{Z}$ operators are unaffected , since they commute with the measurement operator.

After the third measurement M_3 =IXI, since this measurement anticommutes with the stabilizer, the new stabilizers are: $S_G^3 = \langle IXI \rangle$. The logical $\overline{Z}$ operator anticommutes with the measurement, so is replaced by
$\overline{Z_3}$=ZZI $\cdot$ ZIZ = IIZ. The logical $\overline{X}$ is unaffected since it commutes with the measurement operator.

The stabilizer for the control and target qubit is still identity, and logical $\overline{Z} : ZIZ \rightarrow IIZ$.

Since CNOT maps: $Z \otimes Z \mapsto I \otimes Z$, this implements a logical CNOT from the first to the third qubit.

1.3 The Clifford Gates

It has been proven in *reference source* that operators that map a state stabilized by some member of the Pauli-Group to a state stabilized by another member of the Pauli Group can be simulated efficiently on a classical computer. The Group of operators that satisfy this condition is called the Clifford Group.

For the decoder we wish to implement in this thesis, it therefore makes sense to focus on those first and foremost, as applying corrective gates is a computationally/ experimentally expensive task that should be put off until the latest possible moment, and the propagation of an error until then can be simulated efficiently. Also, the pauli-stabilizers utilized to construct our encoding schemes only stabilize clifford and pauli gates, therefore these encoding schemes *can't* protect against non-clifford/pauli operator errors.

The Clifford Group can be generated by:

- The Hadamard-Gate H, which performs single qubit basis changes from eigenstates of X to eigenstates of Z and vice-versa:

  $H|+\rangle = |0\rangle$, $H|0\rangle = |+\rangle$, $H|-\rangle = |1\rangle$, $H|1\rangle = |-\rangle$

- The Phase-Gate P, which performs single qubit sign flips on the state parts which are $|1\rangle$ in the computational basis:

  $P(\alpha|0\rangle \pm \beta|1\rangle) = \alpha|0\rangle \mp \beta|1\rangle$

- The CNOT-Gate, which on a two qubit system performs an X gate on the second qubit if the first qubit is $|1\rangle$, so maps:

  $\alpha|00\rangle + \beta|01\rangle + \delta|10\rangle + \gamma|11\rangle$
  $\mapsto \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$

In the σ_z-basis their matrix representations are:

- $H = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$; $P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$

- $CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

2 Error detection and correction

2.1 Classical codes

The concept of (classical) error-correcting codes (ECC) was introduced by Claude Shannon in 1948. Fundamentally, an ECC encodes *logical* information within a large superset of basic information carriers. In the case of a classical computer, this means encoding a bitstring within a system containing more physical bits than the length of the encoded message, with the goal of message transmission being resilient to some bits being faulty or subject to interference (i.e. EM-interference).

Two such classical ECCs are the repetition and the ring code. We call these codes linear because their graph representations are linear, or one-dimensional. In Quantum error correction, we speak of $[[n, k, d]]$ stabilizer codes if an encoding scheme allows for n physical qubits to encode k logical qubits to an error distance of d, i.e. d arbitrary individual errors being corrigible. In the following, I will refer to linear or classical codes as having a distance of $\frac{1}{2}$, to indicate that they do not protect against an arbitrary single-qubit error, but only against flips in one specific eigenbasis.

2.1.1 Repetition code

For this error code information is encoded by repeating the intended message some amount of times, and then decoding it by performing a majority vote on the transmitted message.

A quantum equivalent of the 3-bit repetition code performed on the message $|1\rangle$ is the $[[3{,}1{,}\frac{1}{2}]]$ repetition code depicted in figure 2 , including so-called *syndrome extraction*. A syndrome is a stabilizer that can be measured to detect whether and where an error has occurred in a multi-qubit system. It is crucial that the measurement of such syndromes occurs without harming the actual quantum information stored in the $data - qubits$. Therefore two additional $ancilla - qubits$ (both initialized to $|0\rangle$) are attached to the circuit via CNOTs. This circuit is stabilized by IZZ and ZZI, measured by ancilla 1/2. The measurement result will therefore be a vector of length two, with each entry either being +1 or -1. To simplify the algebra this will be changed to the binary representation of 0 for +1 and 1 for -1.

To represent the code, Stabilizers can be stacked together to a so-called

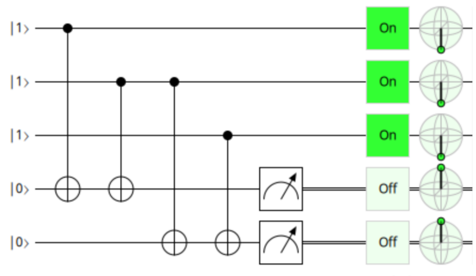

Figure 2: Bitflip Syndrome extractor for [[3,1,$\frac{1}{2}$]] repetition code
+1 measurement result on first ancilla indicates a bitflip error on qubits 1 or 2, +1 result on second ancilla indicates bitflip on second or third qubit

parity-check-matrix, which satisfies:

$$M_{pc} \cdot \vec{v}_{error} = \vec{v}_{syndrome} \tag{13}$$

So e.g. the parity check matrix for the $[3, 1, \frac{1}{2}]$ repetition code would be:

$$M_{pc3} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \tag{14}$$

And the syndrome for an X error on the first qubit would be $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

If we draw a graph to represent this code, with here nodes being ancillas and edges being data qubits, we obtain the following:

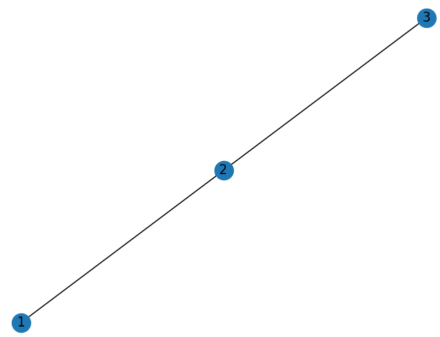

Figure 3: Graph for [[3,1,$\frac{1}{2}$]] repetition code

2.1.2 Ringcode

The ring code's graph essentially just loops around at the repetition code's single-edged ancilla nodes, so for example its edge matrix where the nth row represents which data qubit is connected to the nth ancilla qubit. This matrix for a three-qubit system looks like:

$$M_{pc3} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \tag{15}$$

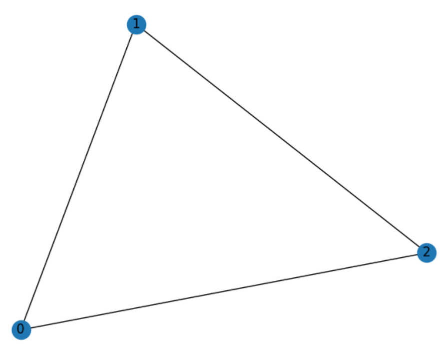

Figure 4: Graph for [[3,1,$\frac{1}{2}$]] ring code

2.2 Quantum Error Model

This way of encoding information however leaves a notable issue:

It only detects bitflip, or Pauli-X, errors occurring on the stored quantum information. While using Hadamard gates one could trivially adapt this code to instead detect Pauli-Z errors, it is not possible to use linear codes like the repetition code to *simultaneously* detect Pauli-X and Pauli-Z errors occurring.

Unlike classical computers, on a quantum computer the type of error is not limited to a bitflip. Even single qubit states have an infinite amount of differing states to it, since when representing a single qubit state as a vector on a Bloch sphere it immediately becomes apparent that there are an infinite number of vectors on that sphere which are different from it. It turns out though, that the change in state from one normalized one to another is merely a sum of two rotations.

Fortunately, noise can therefore be modeled as a sum of Pauli gates. Any single qubit error operator matrix E can be written as:

$$E = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \alpha\mathbb{I} + \beta X + \delta Y + \gamma Z \tag{16}$$

With an apropriate choice of $\alpha, \beta, \gamma, \delta$. In effect, this means that with probability α, the effect of the error $E|\psi\rangle$ will be $\mathbb{I}$; with probability β its effect will be X, and so on.

It is hence sufficient to determine which of these errors $\mathbb{I}$, X, Y or Z has occurred, and we can apply the same operator again to return to the initial state. Since an identity noise occurring is irrelevant to us, and XY as well as ZY (anti-) commute, we need only detect for X and Z errors occurring in order to detect any single qubit errors.

2.3 2D codes

Previous research in computer science provides a toolset for generating valid codes from existing encoding schemes. Hypergraph product codes, introduced by Tillich and Zémor, of two existing codes will always remain a valid detection code.

The parity check matrix H of a hypergraph product code is generated by the parity check matrices of two valid codes in the following

2.3.1 Surface code

We can therefore form a hypergraph product code of two repetition codes for X error detection and Z error detection respectively, obtaining the [[d^2,1,d]] "Surface-Code" which can detect up to d of *both* X and Z errors, and therefore any pauli error happening. We can draw this code as a graph, whereby the code's stabilizers are understood as an adjacency Matrix, the data qubits are on edges of the graph and the ancilla qubits for Z-checks are on faces while the ancilla qubits for X-checks lie on nodes.
Like the repetition code, the Surface code is a code that is regular until it's boundary nodes. INCLUDE LOGICAL OPERATORS INSERT FIGURE SHOWING TORIC CODE

2.3.2 Toric code

Similarly, a hypergraph product code of two ring codes can be generated. We call this code the "Toric code".
Notably, this resembles a donut, or torus. The logical operators on the toric

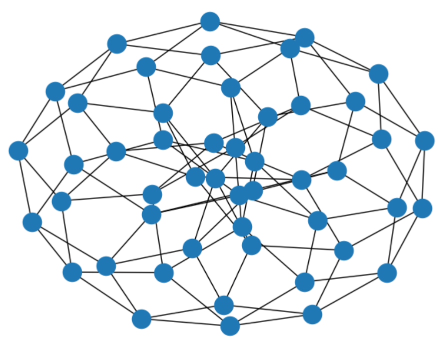

Figure 5: Graph for [[49,1,7]] ring code

code are loops, so a circle of 'errors' on nodes is a logical X operator, and a circle of 'errors' on faces is a logical Z operator.

2.3.3 Color code

The color code is a code whereby the graph whose adjacency matrix's rows are both the code's X stabilizers and Z stabilizers. Any three-colorable graph

represents a valid color code. On the color code, an error is bounded by syndromic faces of all colors.
INSERT FIGURE SHOWING COLOR CODE

2.4 Next steps

- add some visual form of the stabilizer circuit analysis
- explain graphs and hypergraphs
- add visual graph representation of 1D and 2D codes
- add subsection about toric and color code
- add subsection about decoding
- write out xyz matrix representations
- add equation for probabilities in schroedinger picture
- find general commutation property of pauli matrices
- change stabilizer indeces
- change 1.2.3 formulation

3 Decoding Schemes

3.1 Decoders for Surface/Toric codes

The Syndromes on the surface/toric code are a set of nodes and faces on the code graph. The node ancilla syndromes correspond to Z errors, while the face ancilla syndromes correspond to X errors. Since neighboring errors will trigger an ancilla that is between both errors twice, a chain of errors will only appear as two ancilla syndrome bits being flipped at its borders. The task of a decoding scheme for a surface/toric code is thus to find the shortest paths between node pairs/face pairs, since the most likely chain of errors to occur given a $< 50\%$ physical error rate is the shortest one.
In practice, decoders for surface/toric codes only need to be able to match nodes, since the matching of faces is just matching nodes on the dual graphs and the resulting data qubit errors can just be joined (i.e. if an edge is found to have an error on both the X graph as well as the dual Z graph, we know a Y error has occurred on that edge/data-qubit).

ADD FIGURE SHOWING A SYNDROME/CORRECTION!!!!!!!!

3.1.1 MWPM decoding

1. Find a set of unmatched nodes that can be reached from the matching by alternating between matched and unmatched edges. Call these nodes "augmenting nodes".
2. Find an augmenting path starting from each augmenting node, i.e. a path that starts and ends with an unmatched node, and alternates between matched and unmatched edges.
3. If such a path is found, flip all edges along it from matched to unmatched, and vice versa.
4. Repeat until no augmenting path is found.

3.1.2 Union Find decoder

1. Initialize a cluster set for each syndrome node
2. Grow each cluster by one edge in each direction

3. Merge all clusters that share a node
4. For all clusters with an even amount of syndrome nodes, perform MWPM within that cluster. Pop the found error edges from the graph.
5. Repeat until all clusters are merged/discarded.

3.2 Color code decoders

Unlike the surface and toric codes, in the color code the data qubits sit on the graphs nodes, and the ancillas on the graphs faces. Decoding the color code entails matching three differently colored faces to its enclosed nodes. This is a significantly more challenging task than decoding the 2D-codes, since three-colored graph matching is a confirmed NP-hard problem. (reference delfosse paper)

3.2.1 Lookup table decoding

This takes a long time to generate and cannot be done to scale, but is a nice toy.

3.2.2 Lifting decoder

- Create dual of graph
- Generate single-edge-colored subgraphs of the dual
- Decode subgraphs using MWPM/Union-Find
- Unify all edges from subgraph corrections
- Find all loops on this

3.3 Thresholds

To compare different codes and decoding schemes we introduce the concept of thresholds, whereby the threshold of a specific code of scalable distance with a specific decoding scheme is defined as the physical error rate at which the logical error rate becomes greater than 50% in the limit of infinite distance. Thresholds can vary depending on the error model, i.e. some codes can have a higher threshold for X than for Z errors. For simplicity's sake in the following, we will assume equal X, Y and Z error rates of $\frac{per}{3}$, where *per* is the total physical error rate.

4 Conclusion

5 Appendix

5.1 Schroedinger picture calculation of CNOT circuit

In the quantum circuit depicted in figure 1 the input state can be written as $|\psi_{control}\rangle \otimes |0\rangle \otimes |\psi_{target}\rangle$ and the measurement in the first timestep can be expressed as $\mathbb{I} \otimes X \otimes X$.

The initial state $|\phi_{t=0}\rangle = |\psi_{control}\rangle \otimes |\psi_{ancilla}\rangle \otimes |\psi_{target}\rangle$
where
$|\psi_{control}\rangle = \alpha|0\rangle + \beta|1\rangle$
$|\psi_{ancilla}\rangle = |0\rangle$
$|\psi_{target}\rangle = \gamma|0\rangle + \delta|1\rangle$
therefore:

$$|\phi_{t=0}\rangle = \alpha\left(\gamma|000\rangle + \delta|001\rangle\right) + \beta\left(\gamma|100\rangle + \delta|101\rangle\right) \tag{17}$$

If the first measurement result is +1, the state becomes:

$$\begin{aligned}|\phi^{+}_{t=1}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I} + \mathbb{I}\otimes X\otimes X\right)|\phi_{t=0}\rangle \\ &= \alpha\left(\gamma\left(|000\rangle + |011\rangle\right) + \delta\left(|001\rangle + |010\rangle\right)\right) \\ &+ \beta\left(\gamma\left(|100\rangle + |111\rangle\right) + \delta\left(|101\rangle + |110\rangle\right)\right)\end{aligned}$$

if the result is -1, it becomes:

$$\begin{aligned}|\phi^{-}_{t=1}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I} - \mathbb{I}\otimes X\otimes X\right)|\phi_{t=0}\rangle \\ &= \alpha\left(\gamma\left(|000\rangle - |011\rangle\right) + \delta\left(|001\rangle - |010\rangle\right)\right) \\ &+ \beta\left(\gamma\left(|100\rangle - |111\rangle\right) + \delta\left(|101\rangle - |110\rangle\right)\right)\end{aligned}$$

In the case of the +1 Measurement $\rightarrow$ a=0:

$$\begin{aligned}|\phi^{++}_{t=2}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I} + Z\otimes Z\otimes\mathbb{I}\right)|\phi^{+}_{t=1}\rangle \\ &= (|000\rangle\langle000| + |001\rangle\langle001| + |110\rangle\langle110| + |111\rangle\langle111|)|\phi^{+}_{t=1}\rangle \\ &= \alpha\left(\gamma|000\rangle + \delta|001\rangle\right) + \beta\left(\gamma|111\rangle + \delta|110\rangle\right)\end{aligned}$$

$$\begin{aligned}|\phi^{+-}_{t=2}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I} - Z\otimes Z\otimes\mathbb{I}\right)|\phi^{+}_{t=1}\rangle \\ &= (|010\rangle\langle010| + |011\rangle\langle011| + |100\rangle\langle100| + |101\rangle\langle101|)|\phi^{+}_{t=1}\rangle \\ &= \alpha\left(\gamma|011\rangle + \delta|010\rangle\right) + \beta\left(\gamma|100\rangle + \delta|101\rangle\right)\end{aligned}$$

In the case of the -1 Measurement → a=1:

$$\begin{aligned}|\phi_{t=2}^{-+}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I}+Z\otimes Z\otimes\mathbb{I}\right)|\phi_{t=1}^{-}\rangle \\ &= \alpha\left(\gamma|000\rangle+\delta|001\rangle\right)-\beta\left(\gamma|111\rangle+\delta|110\rangle\right)\end{aligned}$$

$$\begin{aligned}|\phi_{t=2}^{--}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I}-Z\otimes Z\otimes\mathbb{I}\right)|\phi_{t=1}^{-}\rangle \\ &= -\alpha\left(\gamma|011\rangle+\delta|010\rangle\right)+\beta\left(\gamma|100\rangle+\delta|101\rangle\right)\end{aligned}$$

Now the applied measurement is $\mathbb{I}\otimes X\otimes\mathbb{I}$, which means:

$$\begin{aligned}|\phi_{t=3}^{+++}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I}+\mathbb{I}\otimes X\otimes\mathbb{I}\right)|\phi_{t=2}^{++}\rangle \\ &= \frac{1}{2}((|010\rangle+|000\rangle)\langle 000|+(|011\rangle+|001\rangle)\langle 001| \\ &+(|000\rangle+|010\rangle)\langle 010|+(|001\rangle+|011\rangle)\langle 011| \\ &+(|110\rangle+|100\rangle)\langle 100|+(|111\rangle+|101\rangle)\langle 101| \\ &+(|100\rangle+|110\rangle)\langle 110|+(|101\rangle+|111\rangle)\langle 111|)|\phi_{t=2}^{++}\rangle \\ &= \frac{1}{2}(\alpha\left(\gamma(|000\rangle+|010\rangle)+\delta(|011\rangle+|001\rangle)\right) \\ &+\beta\left(\gamma(|101\rangle+|111\rangle)+\delta(|100\rangle+|110\rangle)\right))\end{aligned}$$

In this case, a,b and c would each be zero, therefore no further gate would be applied.
As intended, this state is equivalent to $CNOT_{|\psi_{Control}\rangle\rightarrow|\psi_{Target}\rangle}|\phi_{t=0}\rangle$.
If the last measurement result is -1:

$$\begin{aligned}|\phi_{t=3}^{++-}\rangle &= \frac{1}{2}\left(\mathbb{I}\otimes\mathbb{I}\otimes\mathbb{I}-\mathbb{I}\otimes X\otimes\mathbb{I}\right)|\phi_{t=2}^{++}\rangle \\ &= \frac{1}{2}((|010\rangle+|000\rangle)\langle 000|+(|001\rangle-|011\rangle)\langle 001| \\ &+(|010\rangle-|000\rangle)\langle 010|+(|011\rangle-|001\rangle)\langle 011| \\ &+(|100\rangle-|110\rangle)\langle 100|+(|101\rangle-|111\rangle)\langle 101| \\ &+(|110\rangle-|100\rangle)\langle 110|+(|111\rangle-|101\rangle)\langle 111|)|\phi_{t=2}^{++}\rangle \\ &= \frac{1}{2}(\alpha\left(TODOTODOTODOTODOTODO\gamma(|000\rangle+|010\rangle)+\delta(|011\rangle+|001\rangle)\right) \\ &+\beta\left(\gamma(|101\rangle+|111\rangle)+\delta(|100\rangle+|110\rangle)\right))\end{aligned}$$

Notably, each measurement sequence has a differing resulting ancilla state, however we do not care since ancillas are meant to be discarded.
For now, the other 7 final computation steps are left as an exercise to the reader, however I probably will still finish that.