Errors:

- Last lecture we saw that any qubit will have a finite lifetime.

- Due to spontaneous emission/thermalization, qubit population will decay at rate Γ, or with a lifetime $T_1 \sim \frac{1}{\Gamma}$. (Diagonal elements of $\hat{\rho}$.)

- Qubit coherence, or the off-diagonal elements of $\hat{\rho}$, which carry the phase information, will at best survive for coherence time $T_2 \leq 2T_1$ (for a true 2-level system) but in reality for most qubits $T_2 < T_1$. This is known as dephasing, or decoherence.

- Often, both from a master equation formalism and experimentally,

$$\text{population: } \rho_{ii} \propto e^{-t/T_1}$$

$$\text{coherence: } \rho_{ij} \propto e^{-(t/T_2)^{\alpha}}$$

with α as some positive real number, and $\alpha \geq 1$.

- This means any quantum computation run on real, physical qubits is working against a ticking clock.

- In addition, other things can go wrong in a computation and result in errors.

Q. For class: Like what?

A:
- State preparation errors
  (Don't perfectly initialize qubits.)
- Measurement errors.
  (All measurements have noise.)
- Gate errors
  (Single qubit gates are rotations, can over or under rotate, or rotate about the wrong axis. Two-qubit gates are just conditional rotations and suffer from the same issue.
- Qubits can be lost
  (Atoms or ions can escape their traps.)
- Can end up in a state outside of the computational subspace.
  (Most physical qubit candidates are not actually two-level systems.)

We'll set these last two aside for now and come back to them later.

- Gate errors are especially scary. Any useful quantum algorithm will require many gates (circuit depth.)

- For example, the most efficient implementation of Shor's algorithm that has been put forward (Gidney, 2025) requires $\sim 10^{10}$ T gates to factor an 2048-bit number.
(with $\sim 10^3$ logical qubits.)

- A single bit or phase flip error can ruin the computation completely.

- The probability of <u>no</u> error on N gates given a single gate error probability of p is:

$$P = (1-p)^N \sim e^{-Np}$$

(Assuming errors on each gate are independent.)

So for $N \sim 10^{10}$, we need $p < 10^{-10}$-10^{-11} for $P \sim O(1)$.

- This is extremely daunting. Current gate error rates are at best $p \sim 10^{-3}$ level for single qubit gates and 10^{-4} for two-qubit gates, and vary by platform.

- Even with perfect gate calibration and no technical issues at all, the fidelity of a gate is always limited by the finite time it takes due to the finite T_2. If the gate takes time T_g, then:

$$p \leq 1 - e^{-T_g/T_2} \approx \frac{T_g}{T_2}$$

- Many of the quantum computing platforms are approaching or at this limit already.

- In the end, you can only reduce T_g or increase T_2 by so much before you run into pretty insurmountable issues.

- We need some other, more scalable solution. (Error correction!)

Error correction:

Intro:

- Useful quantum computers (i.e. quantum computers that out-perform classical ones) will require algorithms w/ hundreds (or many more) qubits, and long sequences of gates (circuit depth.)

This poses a challenge because as we have just seen:

1) The finite T_1, T_2 times of physical qubits mean there will be qubit flips and phase errors.

2) No quantum gates are perfect, the fidelity is always finite (<1), and the more gates performed the more errors are likely introduced.

- To make quantum computing work, we need quantum error correction.

Classical error correction:

- Classical computers also require error correction to function reliably, as classical hardware is not perfect. As one example, consider information transmitted on a noisy channel. There will always be some probability a bit is flipped or lost. ∴ All (essentially) communications protocols use error correction to mitigate this.

- The insight that imperfect hardware can be used to sythesize reliable machines is not obvious. The idea dates back to:

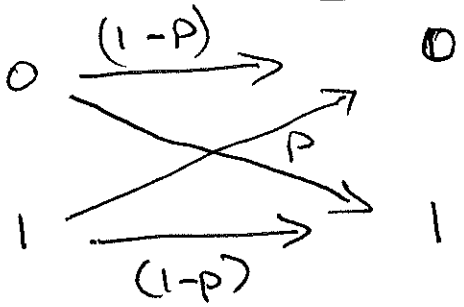
> John Von Neumann, "Probabilistic logics and the synthesis of Reliable organisms from unreliable components." – 1956 (see slide.)

- To understand quantum error correction it is helpful to first consider classical error correction.

Classical bit flip error:

- Suppose we have some probability p of a bit flip in unit time:
  (for each bit, assume probabilities are independent)
  What happens?

Input state: output state:



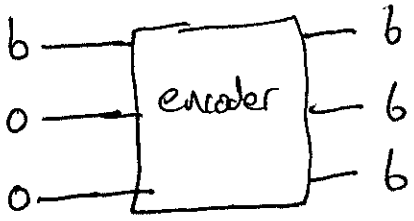
with probability p $\begin{cases} 0 \to 1 \\ 1 \to 0 \end{cases}$

probability $1-p$ $\begin{cases} 0 \to 0 \\ 1 \to 1 \end{cases}$

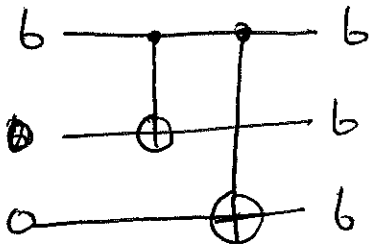
How can we reduce the probability of an error?

By encoding the information in more than one bit:

$b = 0$ or 1:



→ The logic circuit is: (we can think of this as a classical circuit)



codeword

$0 \to \overbrace{000}$ = logical 0

$1 \to 111$ = logical 1

What happens now?

Each bit is subject to a bit Flip error w/ probability p:

Input	output	Probability:
000 →	000	$(1-p)^3$
	001	$p(1-p)^2$
	010	$p(1-p)^2$
	100	$p(1-p)^2$
	011	$p^2(1-p)$
	101	$p^2(1-p)$
	110	$p^2(1-p)$
	111	p^3

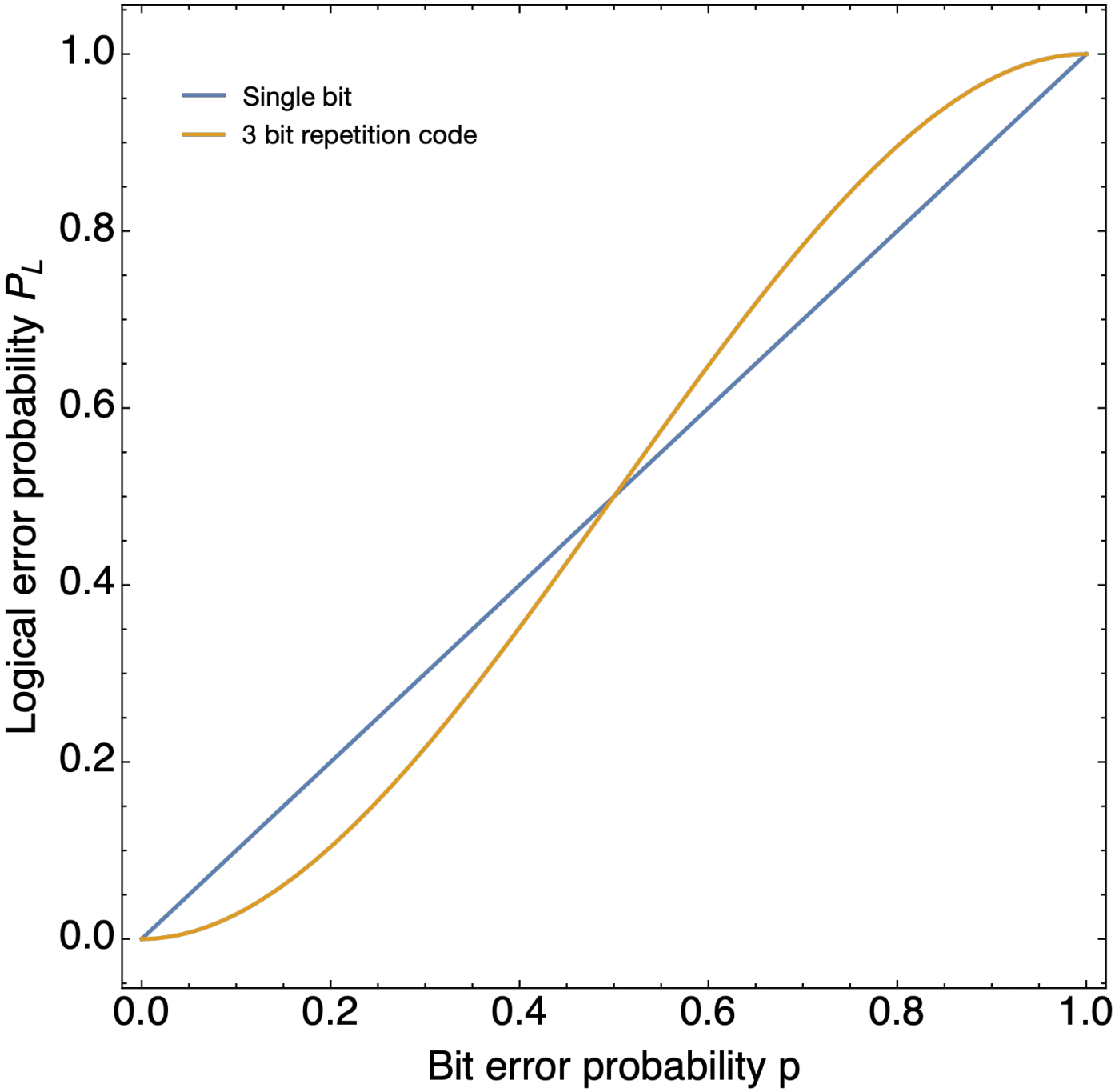
$\Sigma = 1$

How can we correct errors?

Majority rules! Measure all 3 bits and restore bene to the majority value. Suppose again we started with 000. The first four cases in the above table will succesfully produce the correct value:

$$\left.\begin{aligned} P(3\ 0's) &= 1-p^3 \\ P(2\ 0's) &= 3p(1-p)^2 \end{aligned}\right\} P(2\ 0's) + P(3\ 0's) = (1+2p)(1-p)^2$$

$(1+2p)(1-p)^2 > (1-p)$ if $p < \frac{1}{2}$, so majority vote will tend to reduce the error if $p < 1/2$.

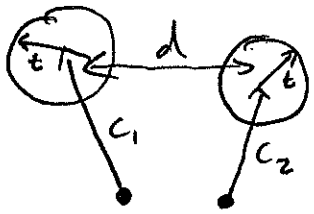


~ Error rate is reduced when $p < 0.5$.

p_{th}

The code distance is the "Hamming" distance between the codewords, which is the ~~distance~~ number of places where two vectors of binary values differ. Codewords c_1, c_2 subject to t errors will end up in regions of radius t. For errors to be distinguishable,

$$t = \lfloor (d-1)/2 \rfloor .$$



A distance d code can detect up to $d-1$ bit errors and correct up to $(d-1)/2$ errors.

- When discussing error correction codes, we will use the following notation:

$[[n, k, d]]$ code

- n: # of physical bits
- k: # of encoded logical bits
- d: code distance

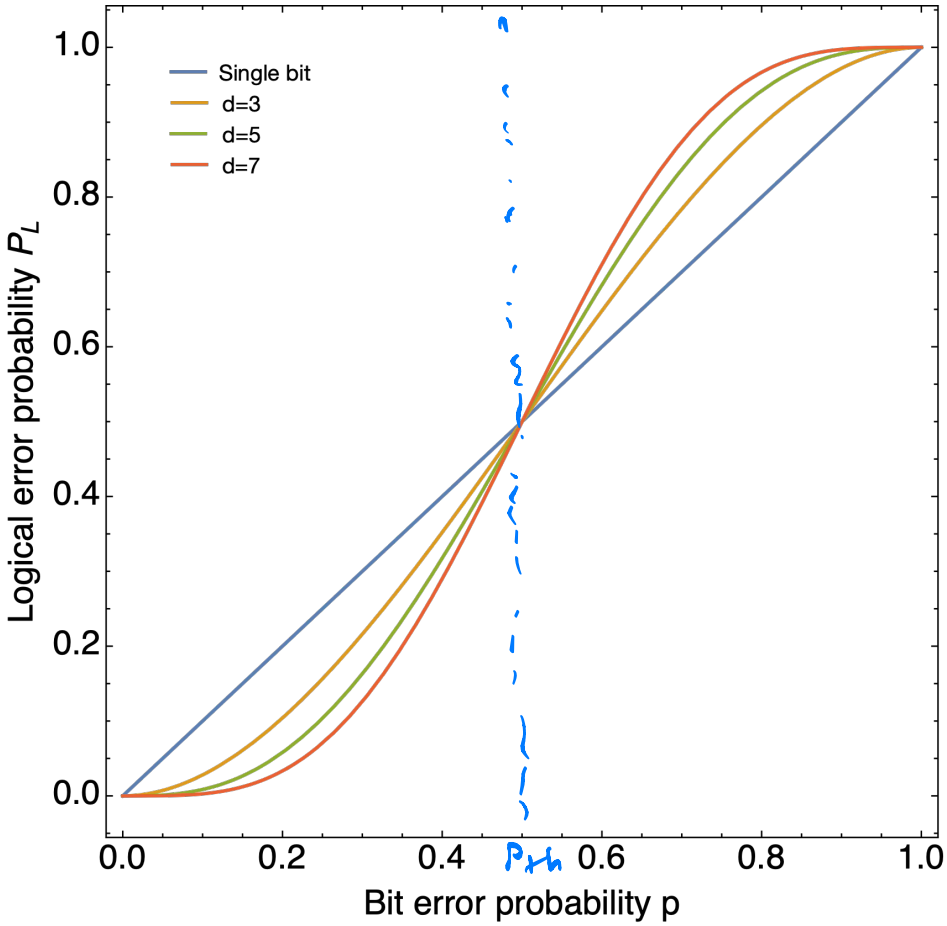
The three-bit code we just considered is a

$[[3, 1, 3]]$ code. (Classical)

3 bits encode 1 logical bit and can correct 1 error.

- For the repetition code, d just scales with number of bits.

$d = 5$: $0 \to 00000$, $1 \to 11111$ (can correct up to two bit flips.)



How much does it help?

For $p \ll 1$, the lowest order term that can't be corrected dominates. Since we can correct up to $\frac{d-1}{2}$ errors, this means the probability of an uncorrected error goes as

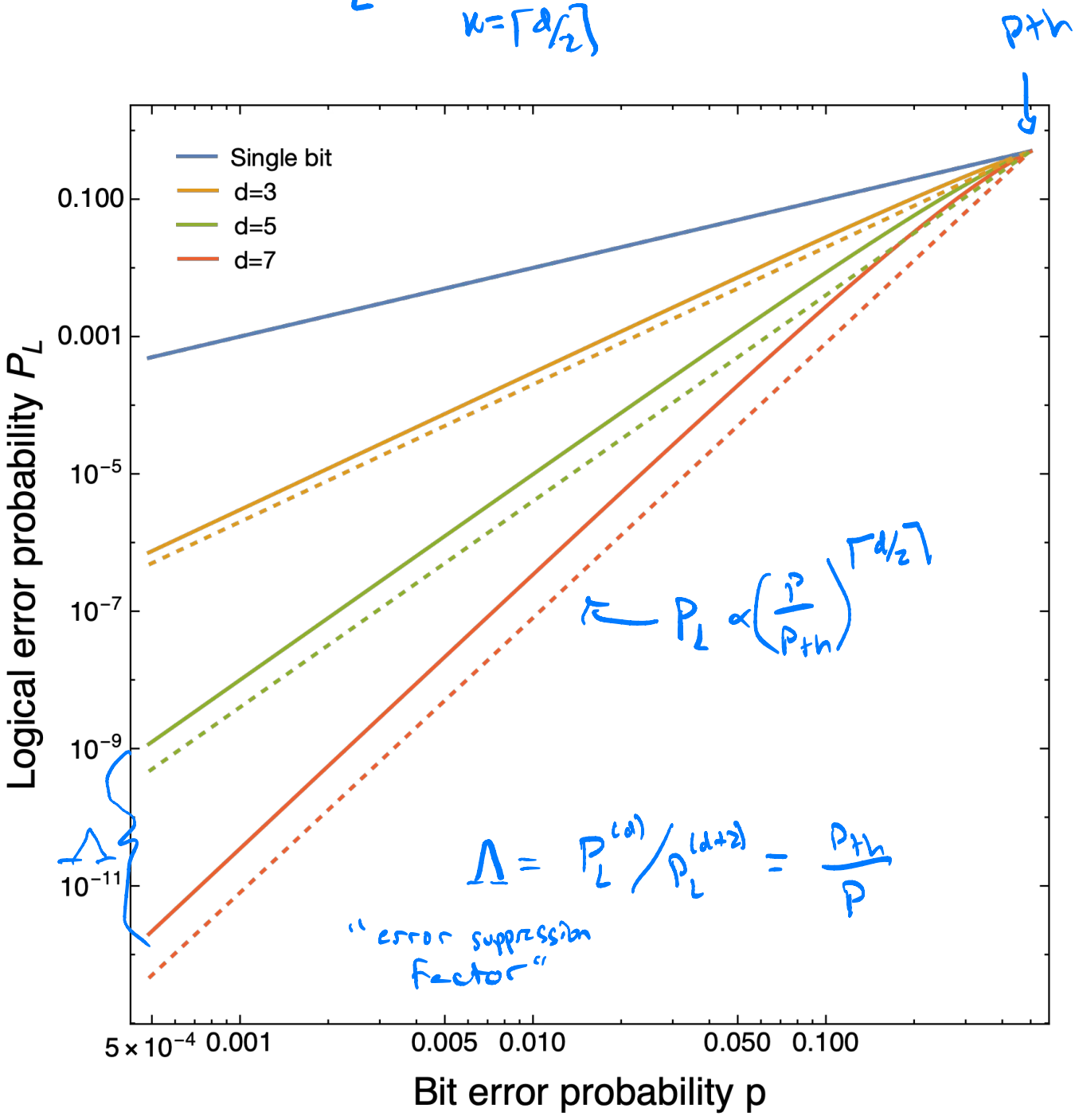
$$P_L \propto p^{\lceil d/2 \rceil}$$

This only works below p_{th}, so it is helpful to write as

$$P_L \propto \left(\frac{p}{p_{th}}\right)^{\lceil d/2 \rceil}$$

- To be rigorous,

$$P_L = \sum_{k=\lceil d/2 \rceil}^{d} \binom{d}{k} p^k (1-p)^{d-k}$$



Can we do this w/ qubits? (will this work to correct errors on a QC.?) Why not? } Questions for class.

Yes and no...

First, why not:

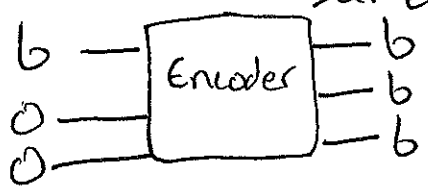
- Can't copy unknown quantum states (no cloning theorem) so how do we build the "encoder"?
- Measuring the qubits to do "majority rules" will project them and collapse superposition/destroy entanglement, so we can't do it mid-calc.
- More errors than just bit flips on a quantum computer... Phase flips, and "analog" as opposed to digital errors.

For these very reasonable sounding reasons, even after Peter Shor proposed "Shor's algorithm" (1994) many people thought implementing it was impossible, and remained uninterested in quantum computing. But in 1995, Peter Shor published a paper showing that a quantum analog to to classical error correction IS possible. (PRA 52, R2493 (1995), cited >5,000 times)

- How is this possible?

Let's consider a slightly different classical error correction approach:

Same encoding as before:



(we will worry about copying quantum states a little later.)

However, now we perform a different measurement on the codewords

here after some possible error has occurred.

We perform the "syndrome measurements:"

XOR_{12} and XOR_{13}, or in other words we measure the parity of bits 1 and 2 and of bits 1 and 3:

input codewords (with errors):	error Syndrome: XOR_{12}	XOR_{13}	(recovery operation) Corrective action:	output codeword
000	0	0	Nothing	000
001	0	1	flip b_3	000
010	1	0	flip b_2	000
100	1	1	flip b_1	000
011	1	1	Flip b_1	111
101	1	0	Flip b_2	111
110	0	1	flip b_3	111
111	0	0	Nothing	111

Notice that here our syndrome measurement does not distinguish between the two different output codewords, or even tell us the values of any of the bits. It corrects for any single bit flip, but fails for 2 or 3 bit flips. This is the same success probability as for majority voting. So success probability is $(1+2p)(1-p)^2$, compared to $1-p$ for no error correction. So works as long as $p < 1/2$.

(This is the best you can hope for, as $p = 1/2$ completely randomizes the info.)

What circuit will implement this?

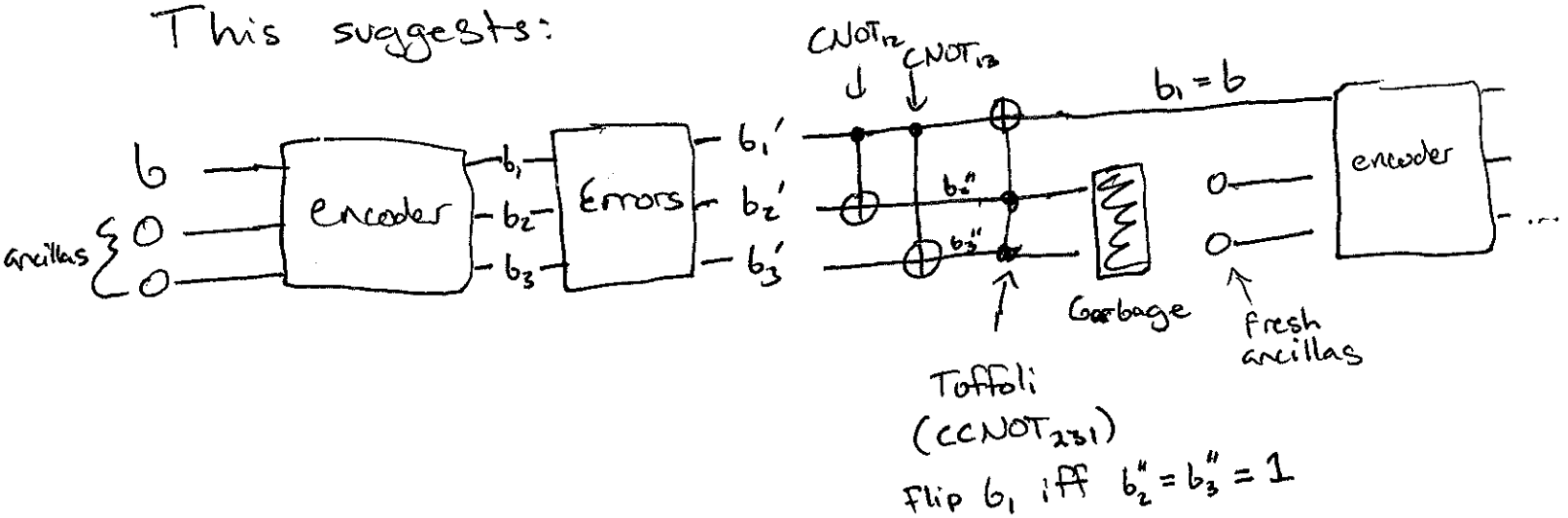
Note that:

Input		XOR
0	0	0
0	1	1
1	0	1
1	1	0

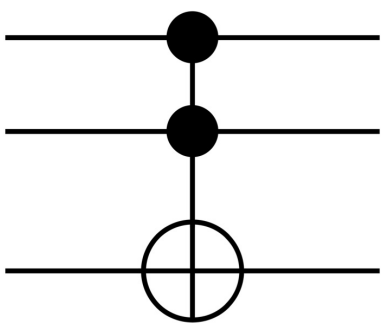
CNOT	
00 →	00
01 →	01
10 →	11
11 →	10

XOR output the same as the target bit after CNOT.

This suggests:



This works for at most one bit flip. Need to be smarter (and add more ancillas) to correct for more.



Truth table

INPUT			OUTPUT		
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Permutation matrix form

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

Quantum error correction:

- We appear to have potentially solved one of the problems we faced, that of protective measurements destroying our quantum information (maybe?). But we are still faced w/ several daunting issues:
  - No cloning
  - Quantum errors are analog
  - Phase-flips?
  - In addition, between decoding, correcting, and recoding the data is unprotected. And what about errors on the gates for decoding and correcting errors?

    This still looks hard!

However, it's not quite so bad as it might first appear. A general error on a single qubit can be discretized:

$$\text{qubit } |\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

$$\text{Environment } |E\rangle$$

The following interactions between the qubit and the environment can occur:

$$|0\rangle|E\rangle \rightarrow \beta_1|0\rangle|E_1\rangle + \beta_2|1\rangle|E_2\rangle$$

$$|1\rangle|E\rangle \rightarrow \beta_3|1\rangle|E_3\rangle + \beta_4|0\rangle|E_4\rangle$$

$E_1 \ldots E_4$ are states of the environment.

$\beta_1 \ldots \beta_4$ are complex amplitudes.

$\Rightarrow |\psi\rangle|E\rangle = (\alpha_0|0\rangle + \alpha_1|1\rangle)|E\rangle \rightarrow$

$\rightarrow \alpha_0\beta_1|0\rangle|E_1\rangle + \alpha_0\beta_2|1\rangle|E_2\rangle + \alpha_1\beta_3|1\rangle|E_3\rangle + \alpha_1\beta_4|0\rangle|E_4\rangle$

$= \frac{1}{2}(\alpha_0|0\rangle + \alpha_1|1\rangle)(\beta_1|E_1\rangle + \beta_3|E_3\rangle) + \ldots$

$+ \frac{1}{2}(\alpha_0|0\rangle - \alpha_1|1\rangle)(\beta_1|E_1\rangle - \beta_3|E_3\rangle) + \ldots$

$+ \frac{1}{2}(\alpha_0|1\rangle + \alpha_1|0\rangle)(\beta_2|E_2\rangle + \beta_4|E_4\rangle) + \ldots$

$+ \frac{1}{2}(\alpha_0|1\rangle - \alpha_1|0\rangle)(\beta_2|E_2\rangle - \beta_4|E_4\rangle)$

Then note that:

$\alpha_0|0\rangle + \alpha_1|1\rangle = |\psi\rangle$

$\alpha_0|0\rangle - \alpha_1|1\rangle = Z|\psi\rangle$

$\alpha_0|1\rangle + \alpha_1|0\rangle = X|\psi\rangle$

$\alpha_0|1\rangle - \alpha_1|0\rangle = XZ|\psi\rangle$

So:

$|\psi\rangle|E\rangle \rightarrow \frac{1}{2}|\psi\rangle(\beta_1|E_1\rangle + \beta_3|E_3\rangle)$

$+\frac{1}{2}Z|\psi\rangle(\beta_1|E_1\rangle - \beta_3|E_3\rangle)$

$+\frac{1}{2}X|\psi\rangle(\beta_2|E_2\rangle + \beta_4|E_4\rangle)$

$+\frac{1}{2}XZ|\psi\rangle(\beta|E_2\rangle - \beta_4|E_4\rangle).$

- Remarkably, any qubit interaction w/ the environment, including our control system and gate errors, can be decomposed into one of four discrete qubit operations (I, Z, X, XZ) and some change to the environment.

- If the state of the environment factors out of the qubit state, the qubit is not entangled w/ the environment, and this is a coherent error.
  - If it does get entangled, then when the environment is traced out it's no longer a pure state and it's an incoherent error.

Let's return now to the 3-qubit repetition code.

- Because of no-cloning we can't implement

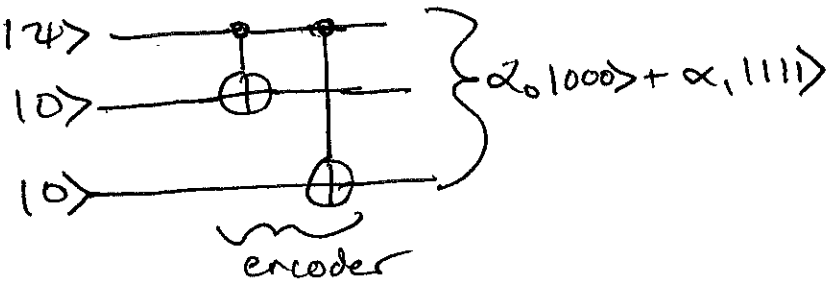
$$|\psi\rangle|0\rangle|0\rangle \rightarrow |\psi\rangle|\psi\rangle|\psi\rangle \quad \text{for unknown } |\psi\rangle.$$

But instead, as foreshadowed by our use of CNOT gates, we do:

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

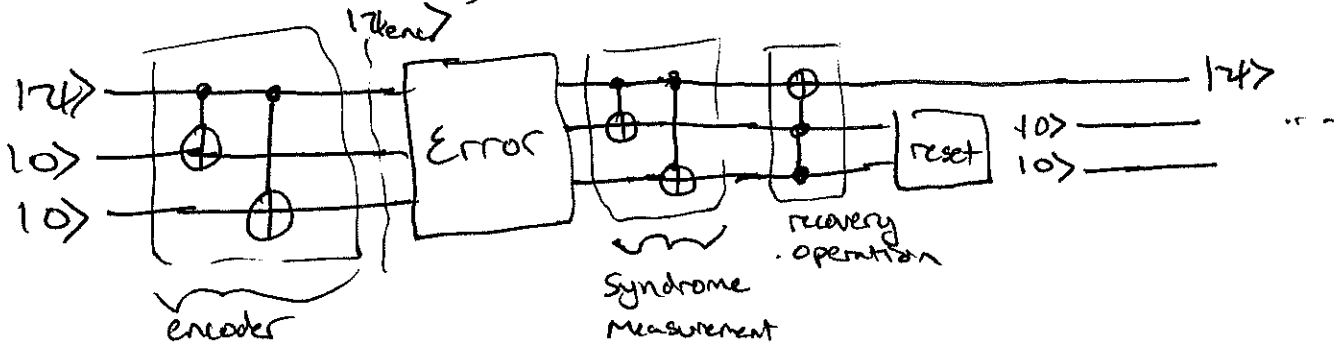
$$|\psi\rangle|0\rangle|0\rangle \rightarrow U_{enc}|\psi\rangle|0\rangle|0\rangle = \alpha_0|000\rangle + \alpha_1|111\rangle$$

$\alpha_0|0\rangle + \alpha_1|1\rangle =$



This 3-qubit encoding cannot protect against arbitrary errors, but it is sufficient to protect against a single type of error X, or Z, or XZ.

Just as before, the full circuit is:

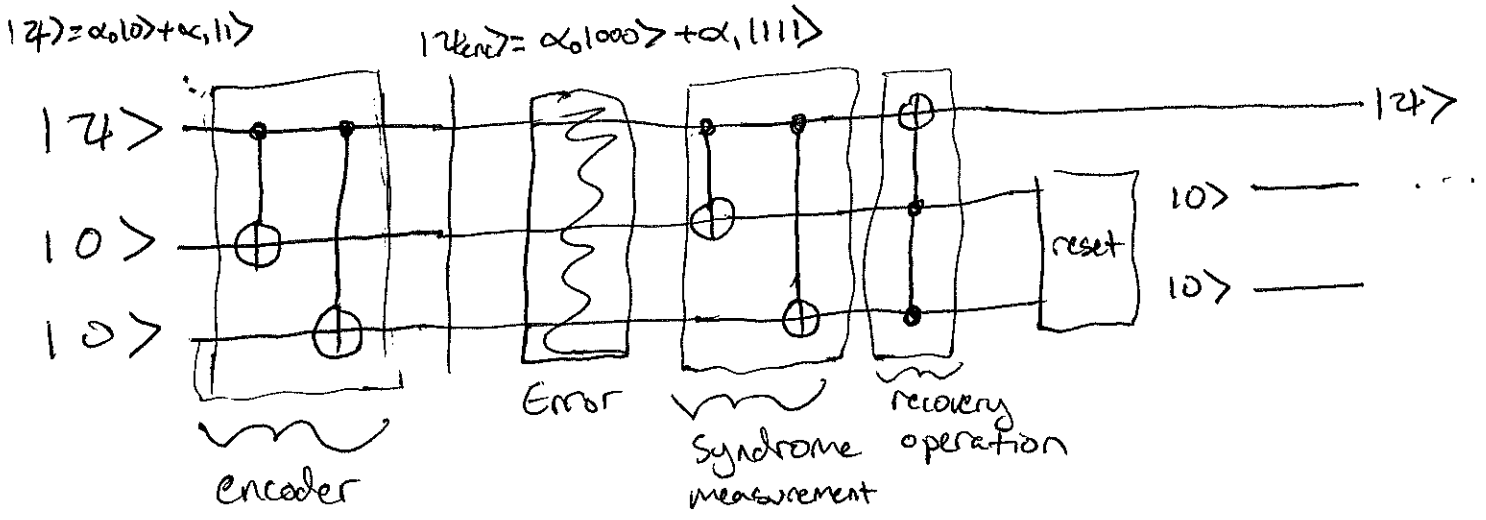


3 qubit bit flip code continued:

- We are trying to find a quantum analog to classical error correction, and identified the following challenges:

- Projective measurements can destroy quantum computation ✓
- No-cloning ✓
- Quantum errors are analog ✓
- Phase-flips?
- Decoding, correcting, and recoding leaves the data unprotected. Also, the gates used to decode and recode will have errors too.

So far we have found possible solutions for the first 3 issues, in the form of the 3-qubit bit flip code:



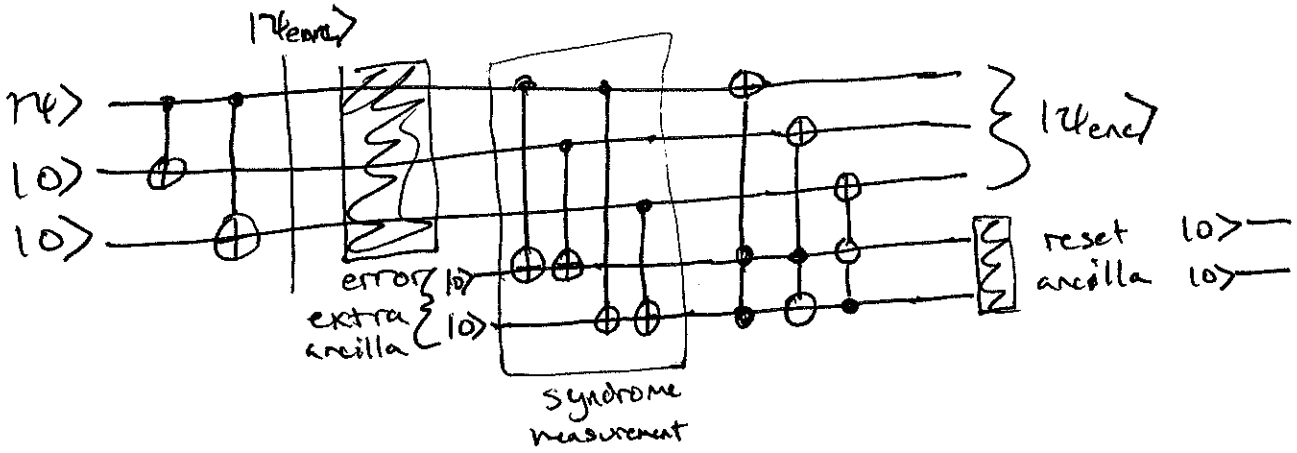
Let's check that this corrects a bit flip on any of the 3 qubits:

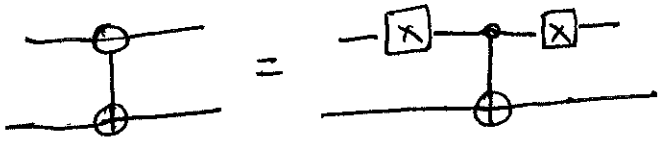
$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, $|\psi_{enc}\rangle = \alpha_0|000\rangle + \alpha_1|111\rangle$

error op	result	Syndrome	recovered state
I	$\alpha_0\|000\rangle+\alpha_1\|111\rangle$	$\alpha_0\|000\rangle+\alpha_1\|100\rangle$	$\alpha_0\|000\rangle+\alpha_1\|100\rangle$ $=\|\psi\rangle\|00\rangle$
X_1	$\alpha_0\|100\rangle+\alpha_1\|011\rangle$	$\alpha_0\|111\rangle+\alpha_1\|011\rangle$	$\alpha_0\|011\rangle+\alpha_1\|111\rangle$ $=\|\psi\rangle\|11\rangle$
X_2	$\alpha_0\|010\rangle+\alpha_1\|101\rangle$	$\alpha_0\|010\rangle+\alpha_1\|110\rangle$	$\alpha_0\|010\rangle+\alpha_1\|110\rangle$ $=\|\psi\rangle\|10\rangle$
X_3	$\alpha_0\|001\rangle+\alpha_1\|110\rangle$	$\alpha_0\|001\rangle+\alpha_1\|101\rangle$	$\alpha_0\|001\rangle+\alpha_1\|101\rangle$ $=\|\psi\rangle\|01\rangle$

In each case we recover $|\psi\rangle|anc\rangle$, then reset $|anc\rangle \to |00\rangle$ and continue.

- This circuit restores $|\psi\rangle$ but not $|\psi_{enc}\rangle$, leaving it vulnerable to errors before we reencode. We can fix that using two ancilla qubits:



Where the symbol  =

- We also need to protect against phase flips, a problem that does not exist classically.
- To see how to do this, recall that bit flips and phase flips are equivalent in different bases.

$$X|0\rangle = |1\rangle \qquad X|1\rangle = |0\rangle$$
$$Z|0\rangle = |0\rangle \qquad Z|1\rangle = -|1\rangle$$

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \qquad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

$$X|+\rangle = |+\rangle \qquad X|-\rangle = -|-\rangle$$
$$Z|+\rangle = |-\rangle \qquad Z|-\rangle = |+\rangle$$

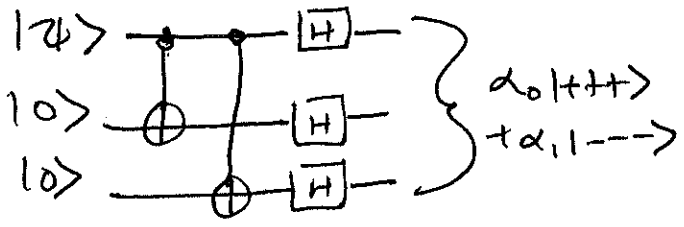
In other words in X basis, Z looks like X and X like Z.

So if we change basis before and after the error, the bit flip correction code fixes phase flips instead.

The operator that switches between Z and X is H. (Hadamard)

$$H|0\rangle = |+\rangle, \quad H|1\rangle = |-\rangle$$
$$H|+\rangle = H^2|0\rangle = |0\rangle, \quad H|-\rangle = H^2|1\rangle = |1\rangle$$

Encoding circuit:



Syndrome + recovery