CSCI 4140 Qubits and Gates

Mark Green

Faculty of Science

Ontario Tech

Introduction

- Examine the gate based model of quantum computing
- There are other approaches, but this is a good starting point:
  - The most common model
  - Somewhat similar to classical computing
  - Needs the least knowledge of quantum mechanics
- Most books on quantum computing concentrate on this model, and is the basis for the software we are using

Introduction

- In classical computing the gates are the physical resource, the bits come along for free
- We have a fixed number of gates in each computer, but we have a limitless supply of bits
- The opposite is true with quantum computing
- We have a fixed number of qubits, and an unlimited number of gates
- In classical computing there are only a few types of gates
- In quantum computing there is an infinite number of gate types

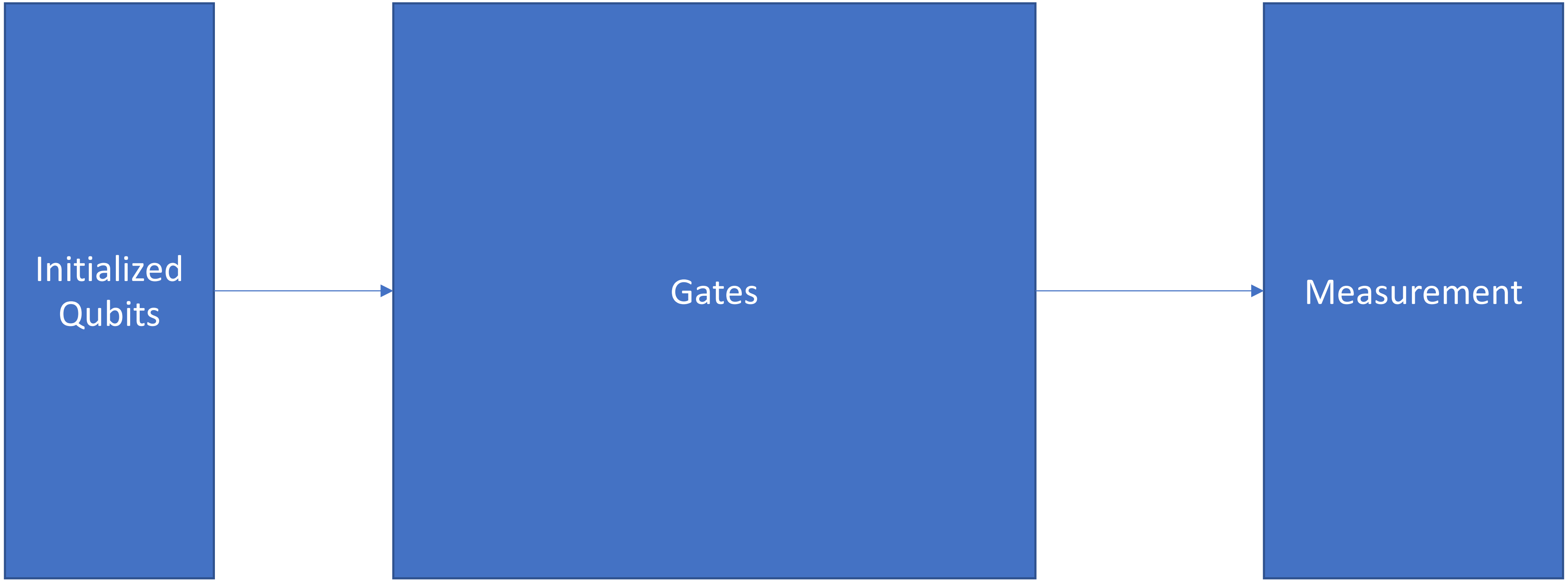
Introduction

- So far there are no real high level programming languages for quantum computing
- Programming is done at the gate level
- This isn’t as bad as it sounds, since quantum gates are more powerful than classical gates
- Most of the quantum computing algorithms require only a relatively small number of gates, so this isn’t a major problem

Model

- Our model is qubit driven
- Start with a fixed number of qubits, initialized to the input to the algorithm
- This is followed by a number of gate applications, gates are applied in parallel to all the qubits in the program, at least theoretically
- After the gates have been applied the qubits are measured to get the final result
- In some cases a classical computer will be involved after the measurement

Model



Qubits

- We’ve seen that a qubit can be in state |0> or state |1>, or anything in-between
- We can write it as:

$$q = \alpha|0\rangle + \beta|1\rangle$$

- Where:

$$|\alpha|^2 + |\beta|^2 = 1$$

- And

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad and\ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Qubits

- Note that |0> and |1> for an orthonormal basis for the space of qubits, we will come back to this later, computational basis
- Recall that both α and β are complex and give the probabilities of observing |0> and |1> when the qubit is measured
- According to our model we need to put the qubits in an initial state
- By default qubits start out in state |0>, which is usually the ground state of the quantum system
- We need some way of getting a |1>, which can be done using the X gate, similar to the NOT gate in digital logic

Qubits

- Gates are represented by matrices, in this case 2x2 matrices since our qubits are 2D vectors
- The matrix for the X gates is:

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

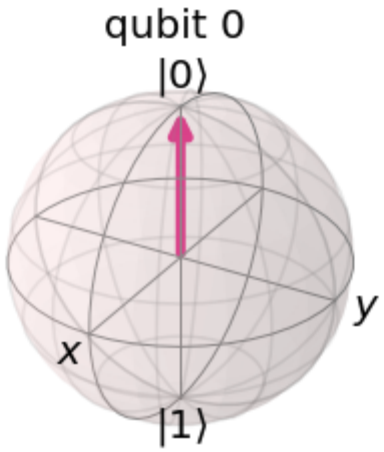
- Let's see what happens when we apply X to |0>:

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

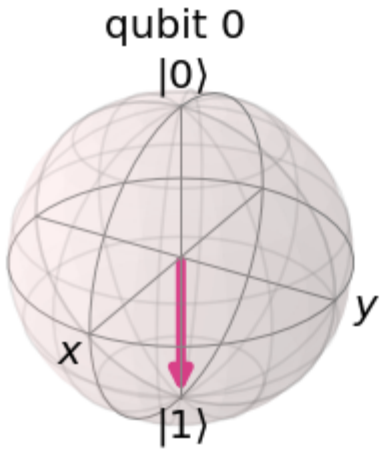
- So applying an X gate to a |0> qubit puts it into the |1> state

Qubits

|0>



X|0>



- Our qubits live on the Bloch sphere.
- They are unit length, so we can view them as two angles, given the direction to a point on the sphere.
- Note that |0> and |1> are at the poles
- This visualization is used extensively in quantum computing

Qubits

- We can initialize qubits, how can we measure them?
- Measuring a qubit brings us back to the digital world, we will get a real value that is interpreted as a 0 or 1
- In this case |0> maps onto 0, and |1> maps onto 1
- Now α and β determine the probabilities of getting a 0, or a 1
- This gives us a way formally defining what we mean by measurement
- Note, that this is always going to be a probability

Qubits

- Given a qubit q, and a state |x> that we want to measure, the probability of measuring |x> is given by:
$$p(|x\rangle) = |\langle q|x\rangle|^2$$
- Example: $q = \alpha|0\rangle + \beta|1\rangle$ and we want to measure |0>, we have
$$|\langle q|0\rangle|^2 = \left|\begin{pmatrix}\alpha^* & \beta^*\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\right|^2 = |\alpha|^2$$
- This gives us what we would expect, similarly if we measure q with respect to |1> we get $|\beta|^2$
- Note, that we can measure with respect to any state we like, at least in theory

Qubits

- One of the interesting properties of qubit is phase, this is something we need to have a good handle on, since its important later
- It turns out there are two types of phase, global phase and local phase
- We can’t measure global phase, it just disappears when we measure, but we still need to understand it
- Consider the following state:

$$i|1\rangle = \begin{pmatrix} 0 \\ i \end{pmatrix}$$

Qubits

- If we measure with respect to |1> we get the probability of the qubit being in state |1>
- What happens if we measure with respect to i|1>?

$$|\langle q|(i|1\rangle)\rangle|^2 = |i\langle q|1\rangle|^2 = |\langle q|1\rangle|^2$$

- On measurement i disappears, it’s as if it never was there, we have no way of telling if its part of the state
- This is true for any overall factor, γ, such that $|\gamma|^2$=1
- This is called the global phase

Qubits

- The only phase we can measure is local phase, the phase difference between |0> and |1>
- With this is mind, we can rewrite our qubit as:

$$|q\rangle = \alpha|0\rangle + e^{i\phi}\beta|1\rangle$$

- In this case α, β and ϕ are all real numbers, and we still have that:

$$\sqrt{\alpha^2 + \beta^2} = 1$$

- Also recall that

$$\sqrt{sin^2x + cos^2x} = 1$$

Qubits

- From this we can conclude that:

$$\alpha = cos\frac{\theta}{2} \quad and\ \beta = sin\frac{\theta}{2}$$

- For some angle θ, we can now write our qubit as:

$$|q\rangle = cos\frac{\theta}{2}|0\rangle + sin\frac{\theta}{2}e^{i\phi}|1\rangle$$

- From this we can interpret θ and ϕ as spherical coordinates, which brings us back to the Bloch sphere
- We can represent any qubit by a point on the Bloch sphere

Qubits

- We’ve been working in the computational basis, or the Z basis
- We could use any orthonormal basis, there is an infinite number of them
- Another basis that we will find useful is the X basis given by:

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

Gates

- There are an infinite number of gates, we will not examine all of them
- Instead, examine the common ones
- The first set of common gates are the Pauli gates:

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

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

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

Gates

- These gates perform rotations by π radians about the X, Y and Z axis
- Superposition is required to get more complex states, the easiest way to do this is with the Hadamard gate:

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

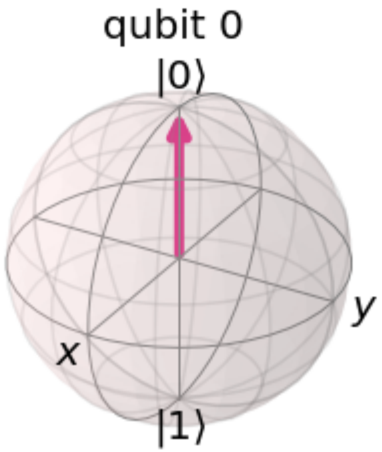
- It works in the following way:

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

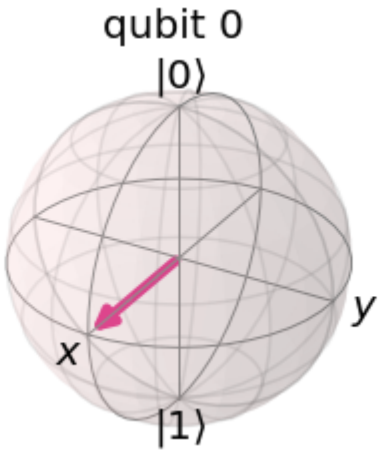
- This gate also changes basis for us

Gates

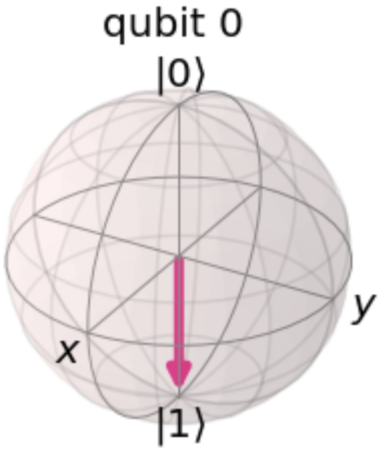
|0>



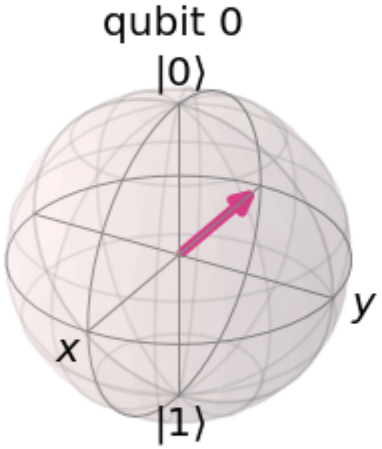
H|0>



|1>



H|1>

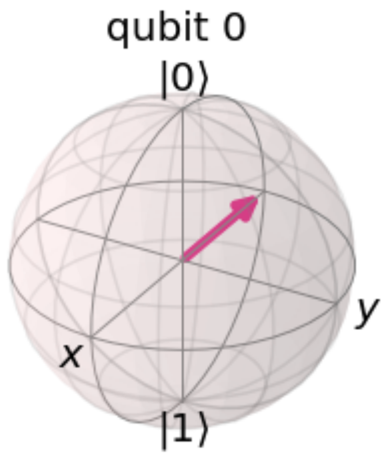


Gates

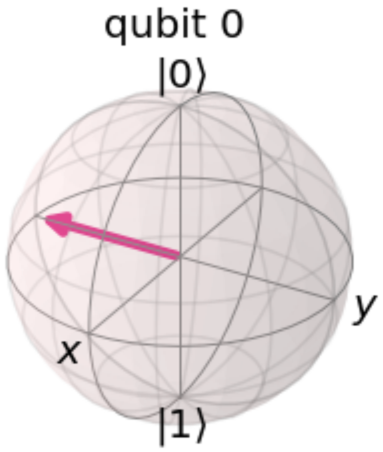
- The R_ϕ gate performs a rotation of ϕ radians about the Z axis:
$$R_\phi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$$
- This is a more general gate, since it is parameterized
- Note that a Z gate is the same as the R_π gate
- There are a number of specialized versions of this gate:
$$S = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{2}} \end{pmatrix} \quad S^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{-i\pi}{2}} \end{pmatrix}$$
- Note that S isn't its own inverse so we need $S^\dagger$

Gates

|->



S|->



Gates

- The T gate is a very useful gate, it is given by:

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{pmatrix} \quad T^{\dagger} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{-i\pi}{4}} \end{pmatrix}$$

- This is our standard set of one qubit gates
- There is a universal one qubit gate given by:

$$U_3(\theta, \phi, \lambda) = \begin{pmatrix} cos\frac{\theta}{2} & -e^{-i\lambda} sin\frac{\theta}{2} \\ e^{i\phi} sin\frac{\theta}{2} & e^{i\lambda+i\phi} cos\frac{\theta}{2} \end{pmatrix}$$

Gates

- There are two specializations of this gate:

$$U_2(\phi, \lambda) = U_3\left(\frac{\pi}{2}, \phi, \lambda\right) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -e^{i\lambda} \\ e^{i\phi} & e^{i\lambda + i\phi} \end{pmatrix}$$

$$U_1(\lambda) = U_3(0,0,\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix}$$

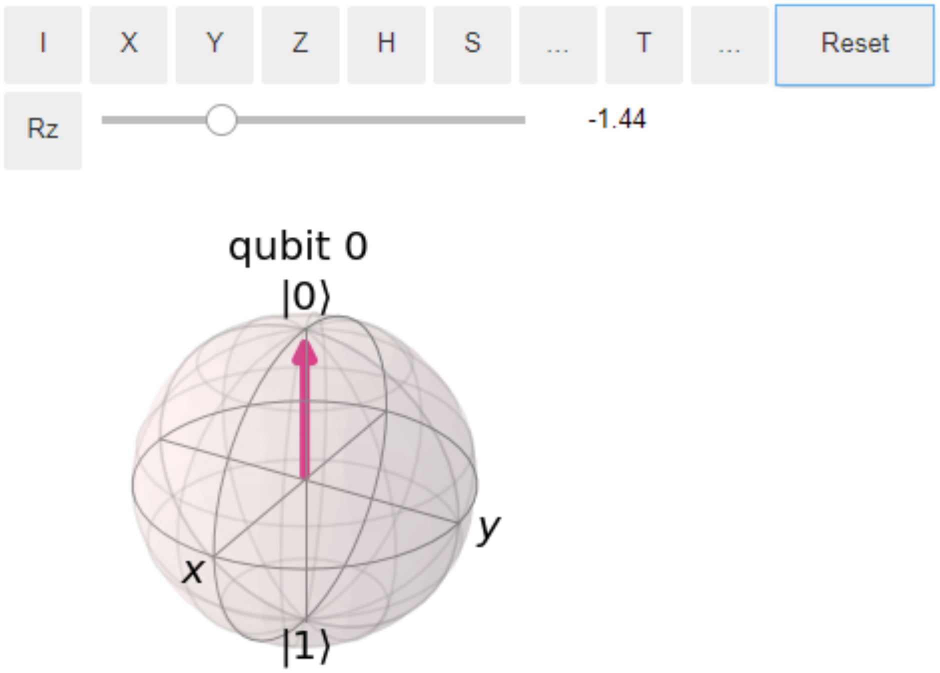
- Note that U_1 is the same as the R_ϕ gate
- The U gates are the gates that are implemented in hardware on the IBM quantum computers

Gates

- Note that all the 1 qubit gates are essentially rotating the qubit on the Bloch sphere
- Since a qubit is always a vector of length 1, there is nothing else we can really do with it
- We need to have more than 1 qubit to do anything interesting

Demo

```
from qiskit_textbook.widgets import gate_demo
gate_demo()
```



Multiple Qubits

- Can't compute anything interesting with one qubit, we need to put multiple qubits together
- We need to pay attention to the order of the qubits, otherwise things don't work out correctly
- Let's start with just two qubits, a and b
- In this case the basis consists of the following four vectors:

$$|00\rangle, \qquad |01\rangle, \qquad |10\rangle, \qquad |11\rangle$$

$$|q\rangle = q_{00}|00\rangle + q_{01}|01\rangle + q_{10}|10\rangle + q_{11}|11\rangle$$

Multiple Qubits

- We must still have:

$$|q_{00}|^2 + |q_{01}|^2 + |q_{10}|^2 + |q_{11}|^2 = 1$$

- We do measurement in the same way, except we now have a larger state vector
- Consider two qubits, given by

$$|a\rangle = \binom{a_0}{a_1} \quad |b\rangle = \binom{b_0}{b_1}$$

- We can combine the two qubits with the tensor product, with b being the higher order bit

Multiple Qubits

- This gives us the following:

$$|ba\rangle = |b\rangle \otimes |a\rangle = \begin{pmatrix} b_0 a_0 \\ b_0 a_1 \\ b_1 a_0 \\ b_1 a_1 \end{pmatrix}$$

- Given just the 4D vector, in general there is no way to factor out the two qubits, if we can it's called a product state
- This extends to three qubits, etc., in the following way:

$$|cba\rangle = |c\rangle \otimes |b\rangle \otimes |a\rangle$$

Multiple Qubits

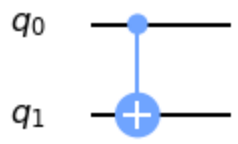
- With a two qubit pair, two single qubit gates can be applied to the qubits separately
- But, we now have a 4D state vector, we need a 4x4 matrix
- Our single qubit gates are 2x2 matrices, how do we get a 4x4 matrix
- We use the tensor product, in the case of |ba>, with M_b applied to |b> and M_a applied to |a>, we have the following:
$$M = M_b \otimes M_a$$
- The matrices don’t commute, so the order is important

Multiple Qubits

- If only one single qubit gate is applied, we use the identity matrix for the other qubit
- We need gates that operate on multiple qubits, the most important of these is CNOT, controlled NOT or CX gate
- This gates has 2 input and 2 outputs, called the control and target
- For now the control input isn’t changed by the gate
- A NOT or X gate is applied to the target qubit if the control qubit is |1>

Multiple Qubits

- In a circuit the CX gate looks like this:

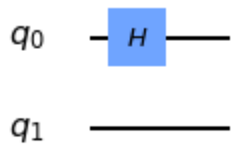


- In this case q_0 is the control and q_1 is the target, the simple truth table is:

Input (t,c)	Output (t,c)
00	00
01	11
10	10
11	01

Multiple Qubits

- If there is no superposition, CX acts similar to a NOT, the interesting behavior occurs when there is superposition
- We can do this using an H gate on the control:

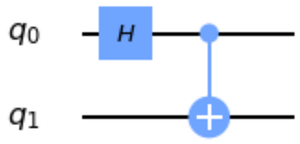


- This give us the following state vector:

$$\text{Statevector} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \\ 0 \end{bmatrix}$$

Multiple Qubits

- Now lets try the following circuit:



- This give us the following state vector

$$\text{Statevector} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

- This is what’s called the Bell state and is famous in quantum mechanics

Multiple Qubits

- What’s special about the Bell state?
- Take a close look at the state vector, there are two non-zero entries, one for |00> and one for |11>
- The qubits are in the same state, either both |0> or both |1>
- If we measure one qubit we instantly know the value of the other
- As soon as one qubit is measured the other one instantly goes into the measured state, there is no longer a superposition of states
- This is an example of entanglement, and was first discovered in the 1930s

Multiple Qubits

- This puzzled physicists for many years, it looks like it breaks the theory of relatively
- The two qubits could be separated by a considerable distance, but it appears that communications is instantaneous when measurement occurs
- The actual communications is when the qubits travel, which must be at less than the speed of light, so there is no contradiction
- Note: an entangled state can’t be factored into the tensor product of two single qubit states

Multiple Qubits

- There is another way of viewing entanglement
- In the entangled state we know everything about the composite state, but we know nothing about the individual qubits
- If we measure one of the qubits in the entangled state we essentially get a random result
- The possible values of the qubit are equally likely
- We can see that in the state vector, both have an equal chance of measuring $|0\rangle$ or $|1\rangle$

Multiple Qubits

- We can produce controlled versions of our other single qubit gates, the control qubit controls whether the gate functions, or just passes the target through unchanged
- Return to the CNOT and the |+> and |-> basis
- We have the following:

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$$

Multiple Qubits

- We also have that:

H|0> = |+>

H|1> = |->

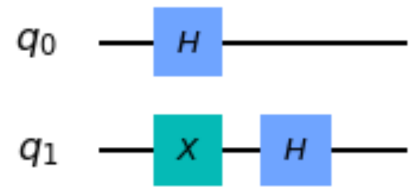
- How does the CNOT effect |++>?

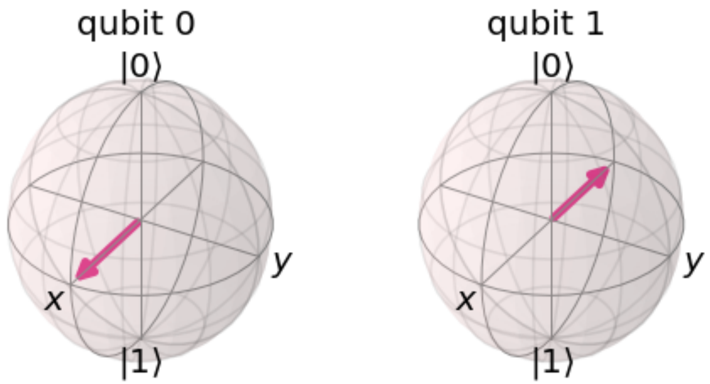
$$|++\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + 11\rangle)$$

- The CNOT swaps the coefficients of |01> and |11>, but they are the same so it has no impact on the qubits

Multiple Qubits – Phase Kickback

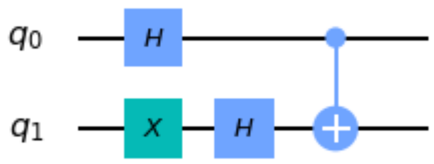
- Now let’s turn to |-+>, we can construct this using the following circuit:

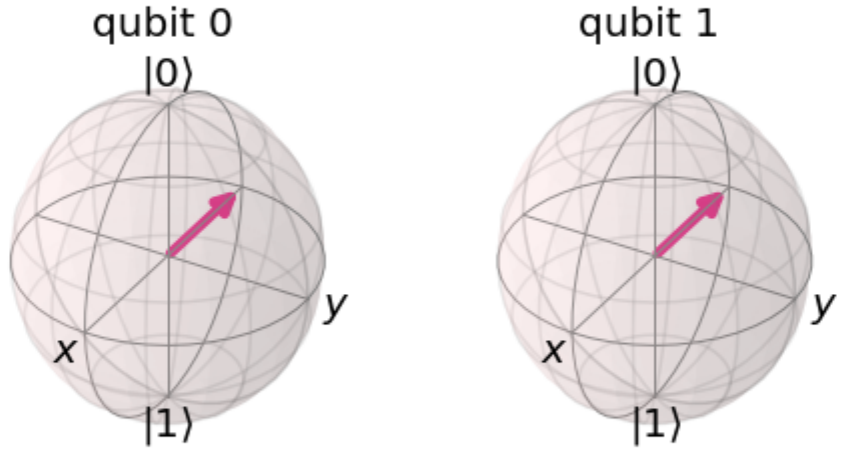




Multiple Qubits – Phase Kickback

- Now add the CNOT to the circuit





Multiple Qubits – Phase Kickback

- Remember qubit 0 is the control qubit, it’s the one that changed value, not the target qubit
- This is called a phase kickback, the modification to the target qubit is kicked to the control qubit
- Phase kickback is important for a number of the quantum algorithms, it is one of our major algorithmic tricks
- We will see it used extensively next week

Multiple Qubits – Phase Kickback

- We can do the same sort of thing with the controlled T gate
- Recall that the T gate is a rotation about the Z axis:

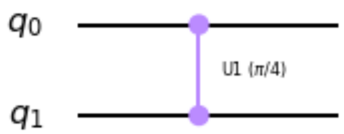
$$\mathbf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$$

- The matrix for the controlled T gate is

$$\text{Controlled-T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\pi/4} \end{bmatrix}$$

Multiple Qubits – Phase Kickback

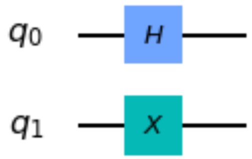
- Now lets build a circuit with a controlled T in it:



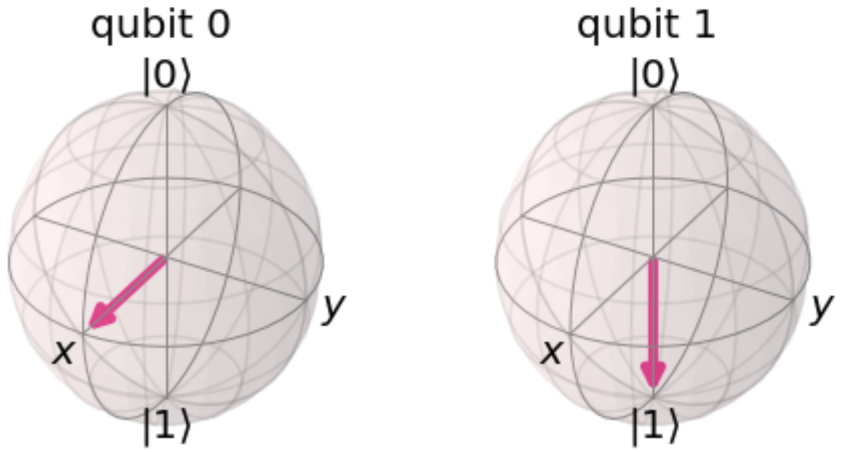
- Note, that it looks like both qubits are controlled, we'll see why this is the case later
- If we stick to just |0> and |1> this doesn't do anything unexpected, it applies the rotation if the control qubit is |1>
- But that's kind of boring, so lets put our control qubit into the |+> state

Multiple Qubits – Phase Kickback

- We can use the following circuit for this:

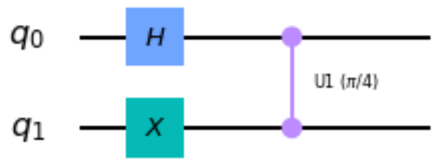


- In terms of a Bloch sphere we get:

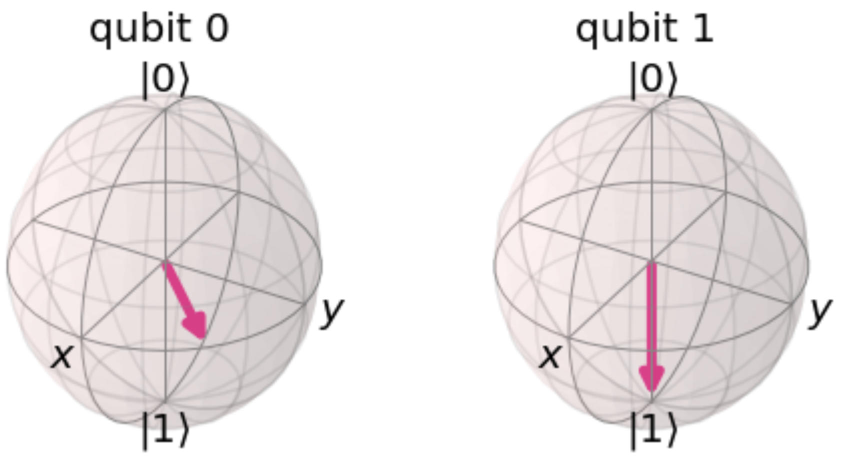


Multiple Qubits – Phase Kickback

- Now we add the controlled T to the picture:

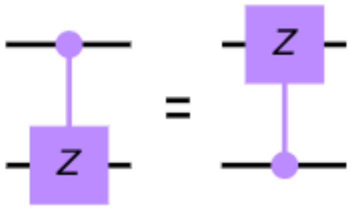


- When we run this we get:



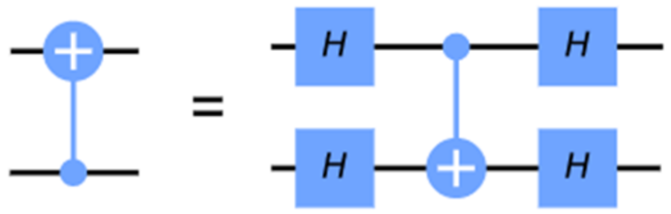
Multiple Qubits – Phase Kickback

- Again it's the control qubit that has been changed, which can be viewed as another example of phase kickback
- Alternatively we can view the controlled Z gates (Z, S and T) as being symmetric:



Multiple Qubits

- Given that H moves between the X and Z basis, we have the following identity:



- Why would we want to do this?
- If the hardware only allows a CNOT in one direction (between qubits), we can use this to reverse the direction

Multiple Qubits

- Remember we have an unlimited supply of gates, but a fixed number of qubits, so this make sense
- There are many interesting ways to combine gates, which convert them to other types of gates
- These are called circuit identities
- A given hardware device may only implement a subset of the gates, the circuit identities allow us to convert our circuit into something that will run on these devices

Circuit Identities

- We have the following two interesting identities:

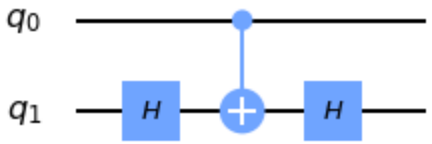
$$HXH = Z$$

$$HZH = X$$

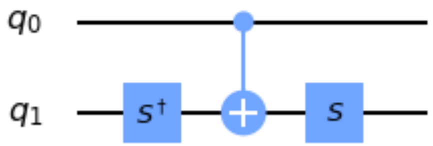
- How does this help us?
- The only two qubit gate the IBM hardware has is the CNOT or cx, if we want to do a controlled Z it looks like we are out of luck
- But we can use the above identities to save the day, all we need to do is use H to change the basis

Circuit Identities

- We can use the following circuit for a controlled Z:



- We can do a similar thing for a controlled Y:



Summary

- In this lecture we have examined the basics of qubits and gates
- We’ve shown how gates can be combined to simulate other gates
- We’ve discuss phase kickback that is important for some of the quantum algorithms
- The video lectures for this week show how quantum circuits can be built with Qiskit and provide more examples
- Make sure to view the eigenvalues and eigenvectors video before next week’s lecture