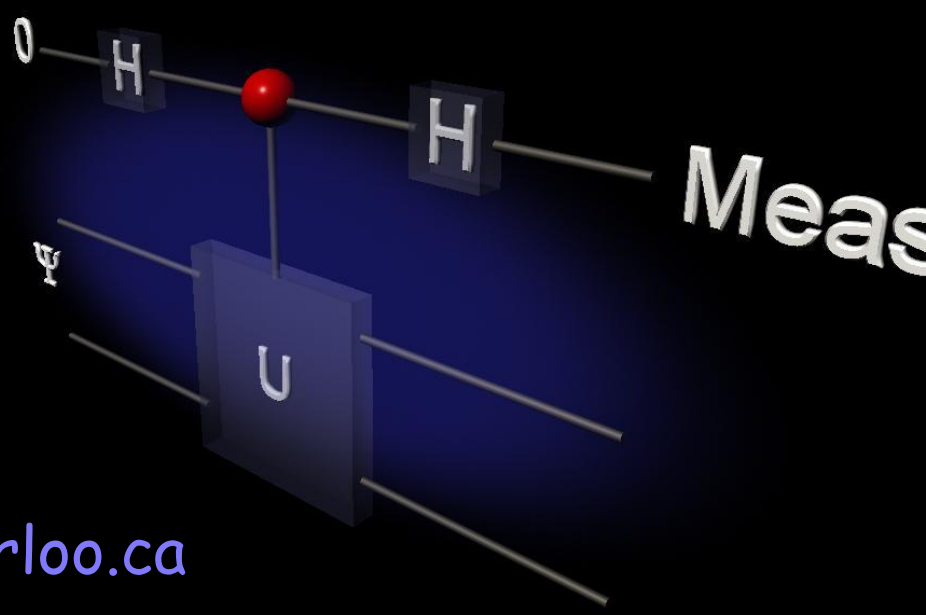
Introduction to Quantum Information Processing



CO481 CS467 PHYS467

Michele Mosca mmosca@iqc.uwaterloo.ca

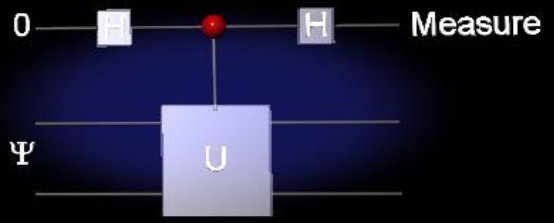
Tuesdays and Thursdays 10am-11:15am

IQC Institute for Quantum Computing

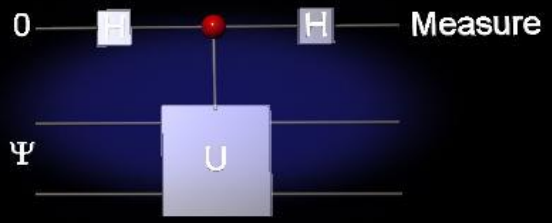
UNIVERSITY OF WATERLOO

PI PERIMETER INSTITUTE FOR THEORETICAL PHYSICS

From quantum physics to a new kind of computer

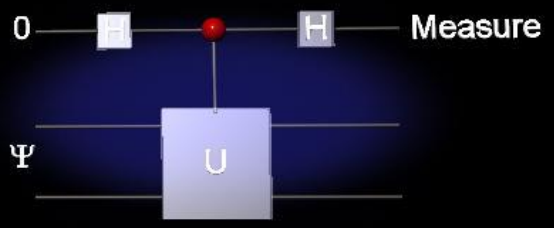


- Classical bits (section 1.4 of text)
- Probabilistic bits (section 1.4 of text)
- Classical Circuit Model (sections 1.3 and 1.5 of text)
- Quantum Circuit Model (parts of sections 2-4 of text)
- No cloning-theorem



Classical (deterministic) bits

Classical (deterministic) bits



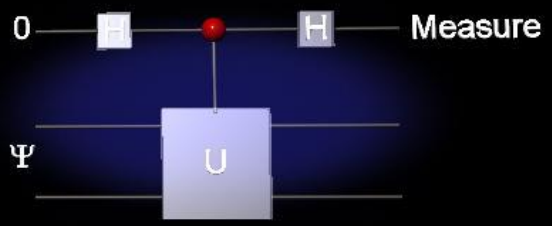
- We usually label the values of a 2-state system A (a “bit”) with 0 or 1.
- If we have n 2-state systems $A_1, A_2, \ldots, A_n$, we usually label the 2^n possible values with 0-1 strings of length n. E.g. for n=3, the possible values are

$$000,001,010,011,100,101,110,111$$

- A more redundant way to represent the value of a bit is by a column vector of 2 numbers, where we put a 1 in position 0 if the value is 0, and we put a 1 in position 1 if the value is 1, and 0s elsewhere.

$$0 \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad 1 \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Classical bits

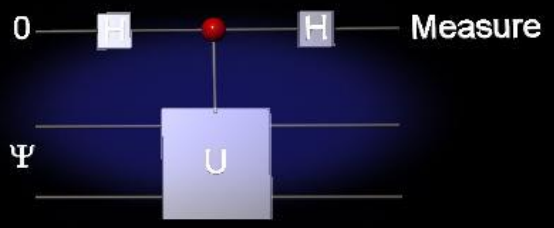


- For n=3 bits, we can represent the 8 possible values as vectors of size 8:

$$000 \equiv \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad 001 \equiv \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \cdots \qquad 111 \equiv \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

- In general, for n bits, we use vectors of length 2^n.

Classical bits



- A process that transforms the value of the bits can be expressed as a function, e.g. $NOT(0) = 1$

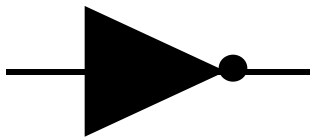
$$NOT(1) = 0$$

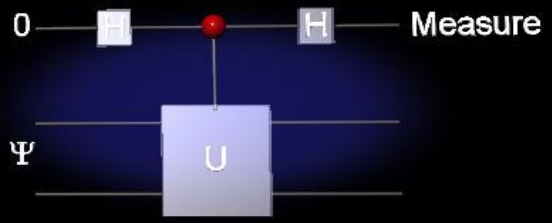
- Using the vector notation, a transformation can be represented by a matrix, e.g. the NOT operation corresponds to the matrix

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

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

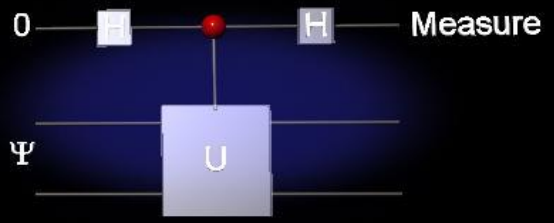
- Gate notation for the process





Tensor products for composite systems

Composite systems

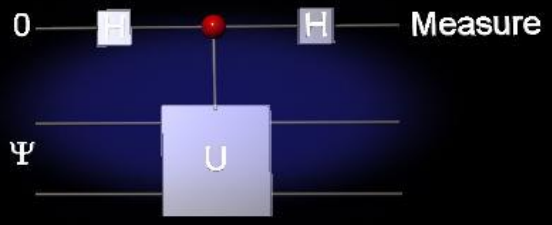


- Suppose the state of two systems A and B are $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ respectively (i.e. “1” and “0” respectively)

- We can treat the two systems as one 4-state system, with states 00,01,10 and 11, described by the vector

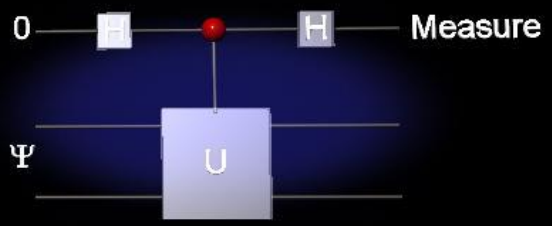
$$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Aside: tensor products (section 2.6)



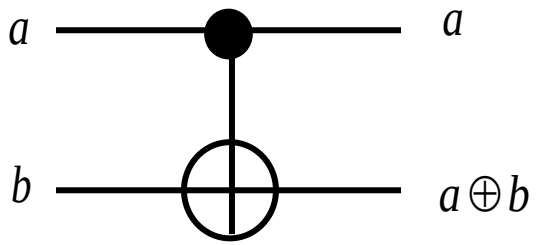
$$\begin{bmatrix} a \\ b \end{bmatrix} \otimes \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} a\begin{bmatrix} c \\ d \end{bmatrix} \\ b\begin{bmatrix} c \\ d \end{bmatrix} \end{bmatrix} = \begin{bmatrix} ac \\ ad \\ bc \\ bd \end{bmatrix}$$

State evolution



- Suppose this 4-state system undergoes an evolution that maps.

$$00 \mapsto 00$$
$$01 \mapsto 01$$
$$10 \mapsto 11$$
$$11 \mapsto 10$$

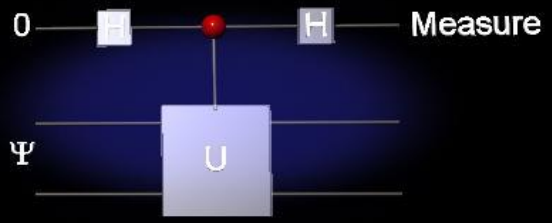


We call this a controlled-NOT gate.

- This evolution corresponds to multiplying the state vector by the matrix

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

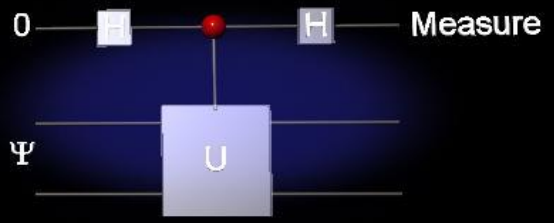
… in matrix notation…



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

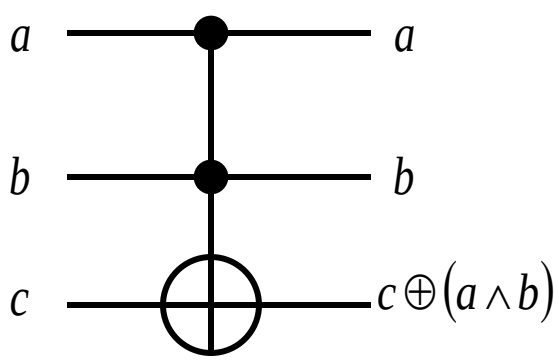
matrix for CNOT

Classical bits



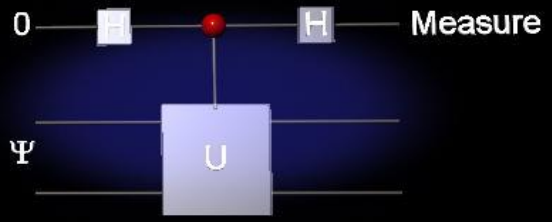
- An n-bit transformation, such as the 3-bit “Toffoli” gate (or controlled-controlled-NOT) can be described by multiplication by an 8x8 matrix

Toffoli gate

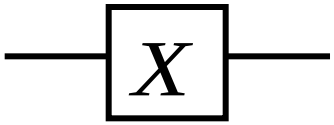


$$
\begin{aligned}
000 &\mapsto 000 \\
001 &\mapsto 001 \\
010 &\mapsto 010 \\
011 &\mapsto 011 \\
100 &\mapsto 100 \\
101 &\mapsto 101 \\
110 &\mapsto 111 \\
111 &\mapsto 110
\end{aligned}
\quad \rightarrow \quad
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
$$

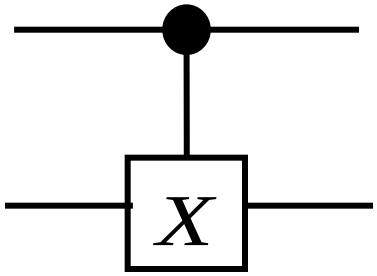
Quantum analogues



"X"-gate or NOT-gate

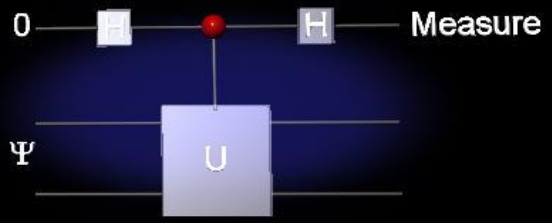


"controlled-NOT" gate



They correspond to the same matrices (which are unitary) in the quantum case

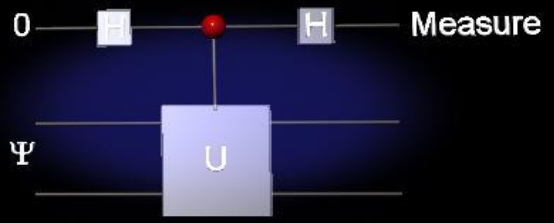
Aside: tensor products (section 2.6)



$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \otimes \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} & a_{12}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \\ a_{21}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} & a_{22}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{bmatrix}$$

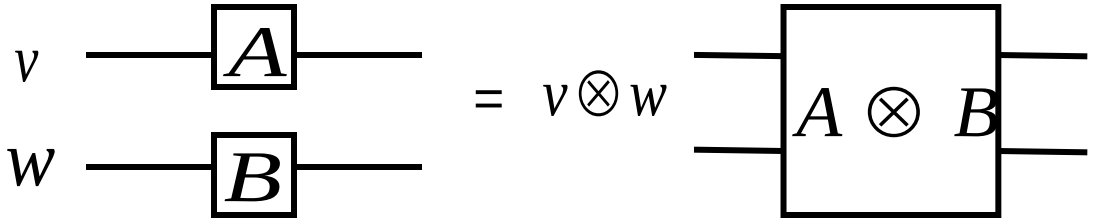
Aside: A basic property of this definition

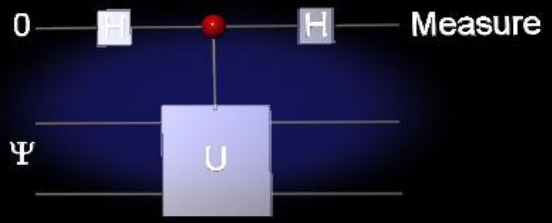


- i.e. applying local operators A and B and then combining the systems is equivalent to combining the systems and then applying the tensor product of A and B

$$\left(\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}\right) \otimes \left(\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}\right) = \left(\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \otimes \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}\right)\left(\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \otimes \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}\right)$$

- Or another way of writing… $(Av) \otimes (Bw) = (A \otimes B)(v \otimes w)$
- Or in gate notation:

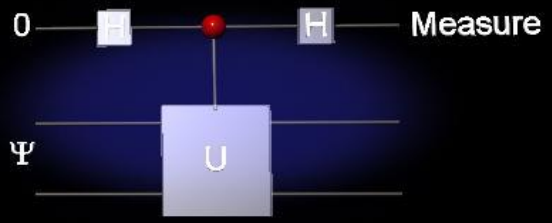




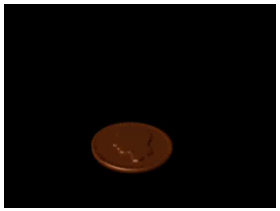
Note: this is not so weird….

Let’s briefly take a look at probabilistic (classical) bits

Probabilistic bits



- Suppose we flip a fair coin to establish the value of a 2-state system A.



- We can describe the state of bit A as

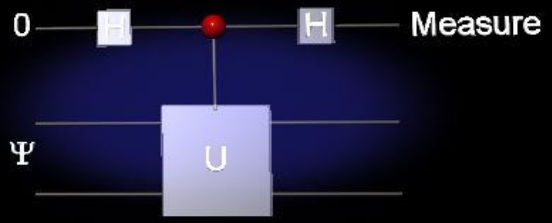
$$50\%\ \text{"0"}, 50\%\ \text{"1"} \qquad \text{or simply} \qquad \begin{pmatrix} .5 \\ .5 \end{pmatrix}$$

(e.g. you can think of these probability vectors as a description of our a priori uncertainty of the outcome, or alternatively as the expected relative frequency of each outcome assuming many repetitions)

- In general, the state of any probabilistic bit can be of the form $\begin{pmatrix} a \\ b \end{pmatrix}$

where $0 \le a, b \le 1, a + b = 1$

Probabilistic bits

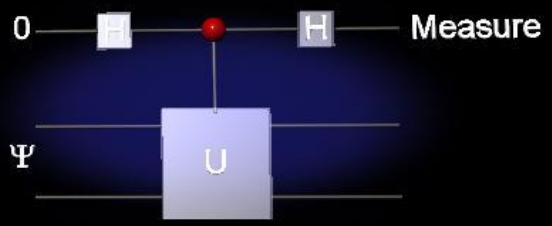


- Suppose the state of two systems A and B are $\begin{pmatrix} .3 \\ .7 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ respectively.

- We can treat the two systems as one 4-state system, with states 00,01,10 and 11, described by the vector

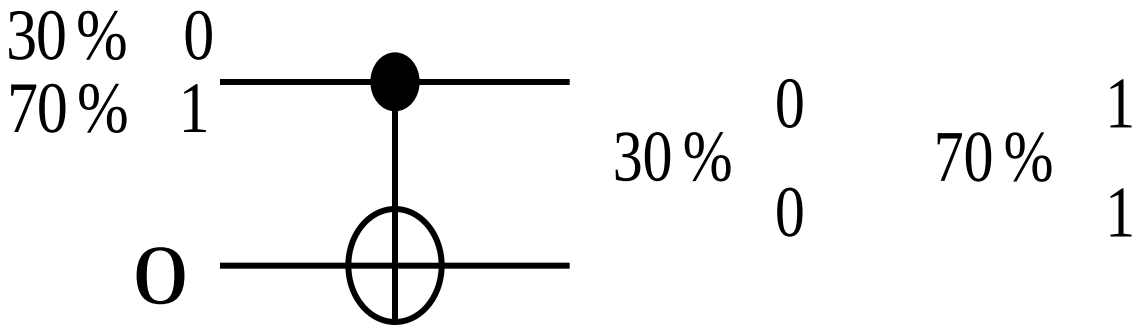
$$\begin{pmatrix} .3 \\ 0 \\ .7 \\ 0 \end{pmatrix} = \begin{pmatrix} .3 \\ .7 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Example



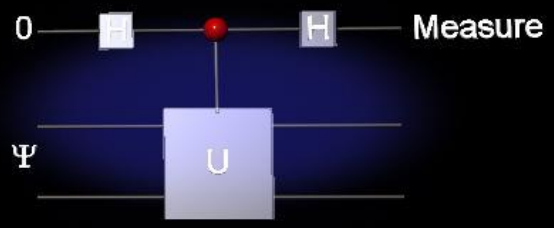
- Suppose the states of systems A and B are $\begin{pmatrix} .3 \\ .7 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$

- Then we apply a controlled-NOT



(Note that bits A and B become "correlated"; we cannot describe them independently)

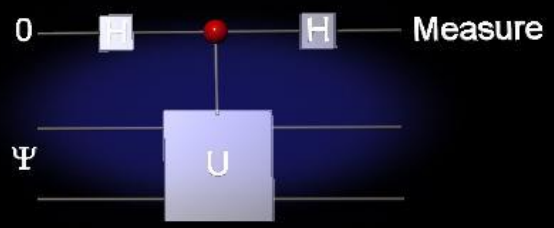
… in matrix notation…



$$\begin{pmatrix} .3 \\ .7 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} .3 \\ 0 \\ .7 \\ 0 \end{pmatrix} \mapsto \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} .3 \\ 0 \\ .7 \\ 0 \end{pmatrix} = \begin{pmatrix} .3 \\ 0 \\ 0 \\ .7 \end{pmatrix}$$

matrix for CNOT

Summary

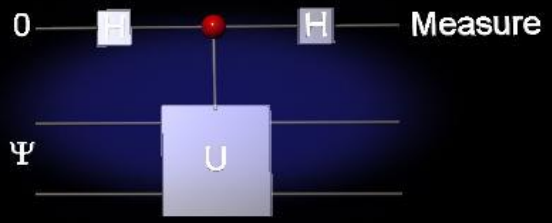


- The state of an N-state probabilistic system can be described by an N-vector of non-negative real numbers

$$\begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{pmatrix}$$

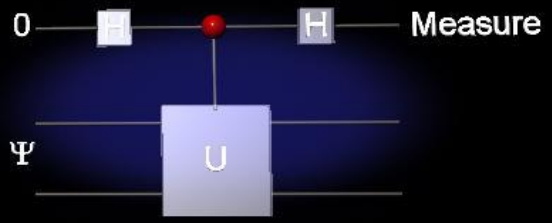
- The evolution of an N-state probabilistic state vector (where the evolution only depends on the current state, i.e. is “Markovian”) can be described by an NxN transition matrix where entry (i,j) corresponds to the probability that the system in state i evolves to state j
- The tensor product is the natural operation for computing state vectors and operations on combined systems of state vectors, from the state vectors and operations on the individual systems.

Relevance to quantum information



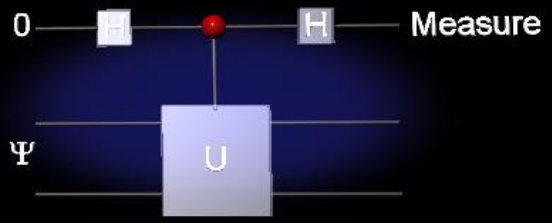
- The structure of pure quantum states and their evolution is very similar (as we will soon see). *Caveat: we use a different norm for the vectors.*

- (In future courses) Classical probabilistic algorithms are analyzed by studying the properties of such transition matrices. Analogous quantum algorithms have been developed and studied in a similar way.



Classical circuit models

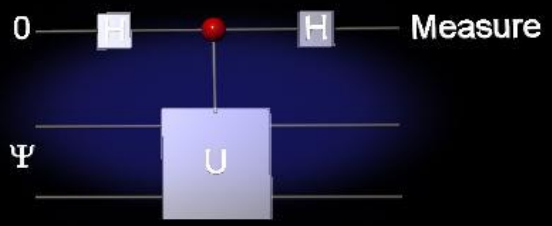
From matrices to circuits…



- We have described some general notation and conventions for describing states of classical bits and transformations on them.

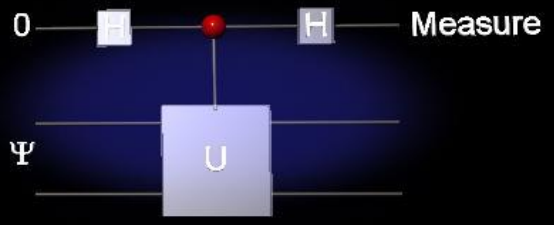
- The circuit model is a useful model for describing transformations on data in terms of basic operations called “gates”. The circuit model is more closely tied to the physical implementation of a computation.

Classical Circuit Model



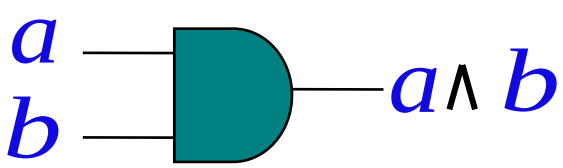
- Definition of gates
- Universal set of gates
- Reversible vs Irreversible gates
- Reversible circuits from irreversible circuits
- New gates and notation for reversible computation

“Classical” Logic Gates

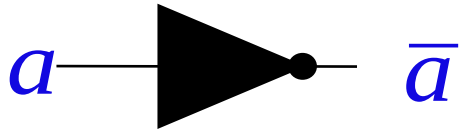


- A gate is a function from m bits to n bits, for some fixed numbers m and n

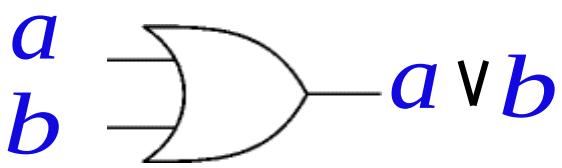
AND



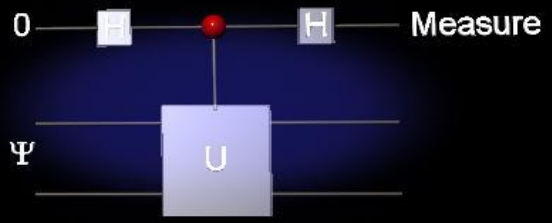
NOT



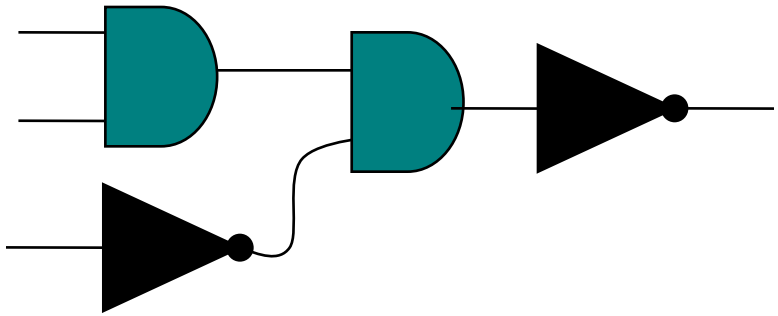
OR



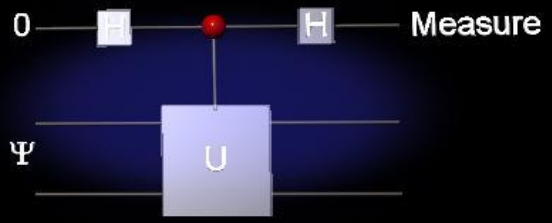
“Classical” Logic Gates



- We “glue” gates together to make “circuits” (or “arrays of gates”) which compute Boolean functions

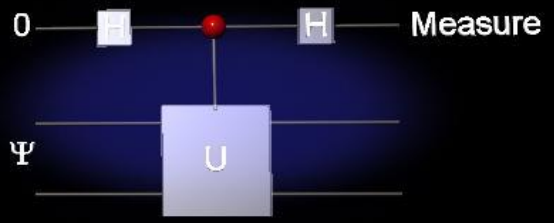


Universal Set of Logic Gates



- A set B of gates is universal if, for any Boolean function F, there is a circuit with gates in B that computes F
- E.g. B = { NOT } is not universal
- E.g B = { AND } is not universal
- E.g. B = {NOT, AND, FANOUT } is universal

Universal Set of Logic Gates

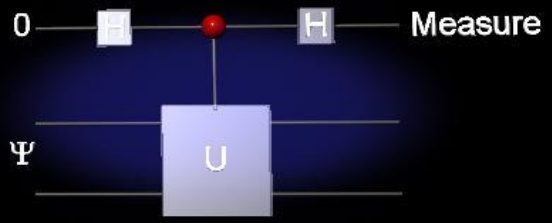


- A circuit designed with one finite set **A** of gates can be efficiently translated into a circuit using gates from a universal set **B**

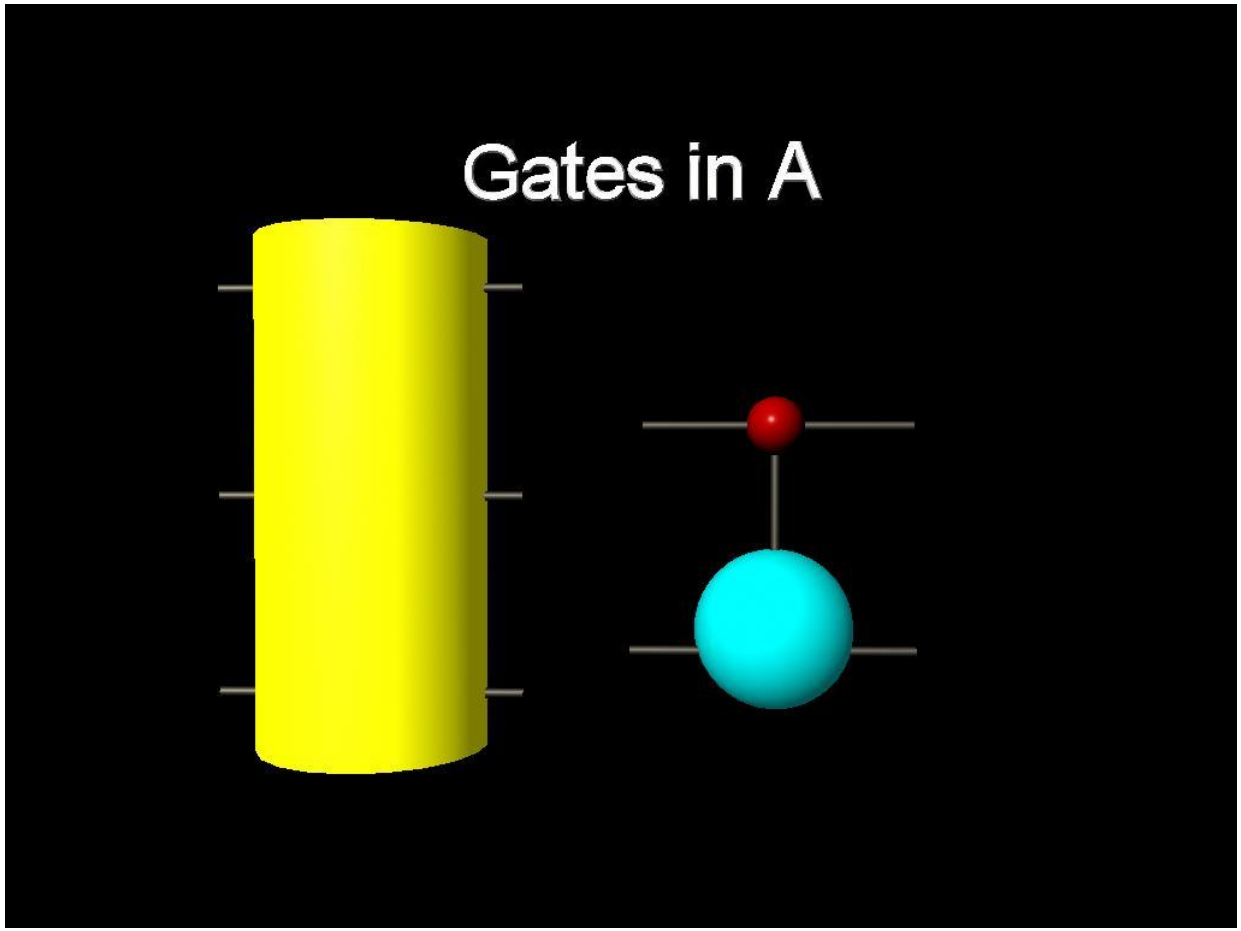
- How? Note that since **B** is universal, every gate in **A** can be realized by a circuit composed of gates from **B**. So we simply replace each gate G in **A** with an appropriate circuit of gates from **B**. The increase in the number of gates will be at most the number of gates in the circuit times a constant. Why?

- Thus, to be able to implement an arbitrary computation, it suffices to be able to realize a universal set of gates.

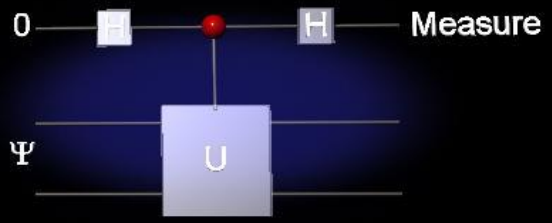
Example of Universality



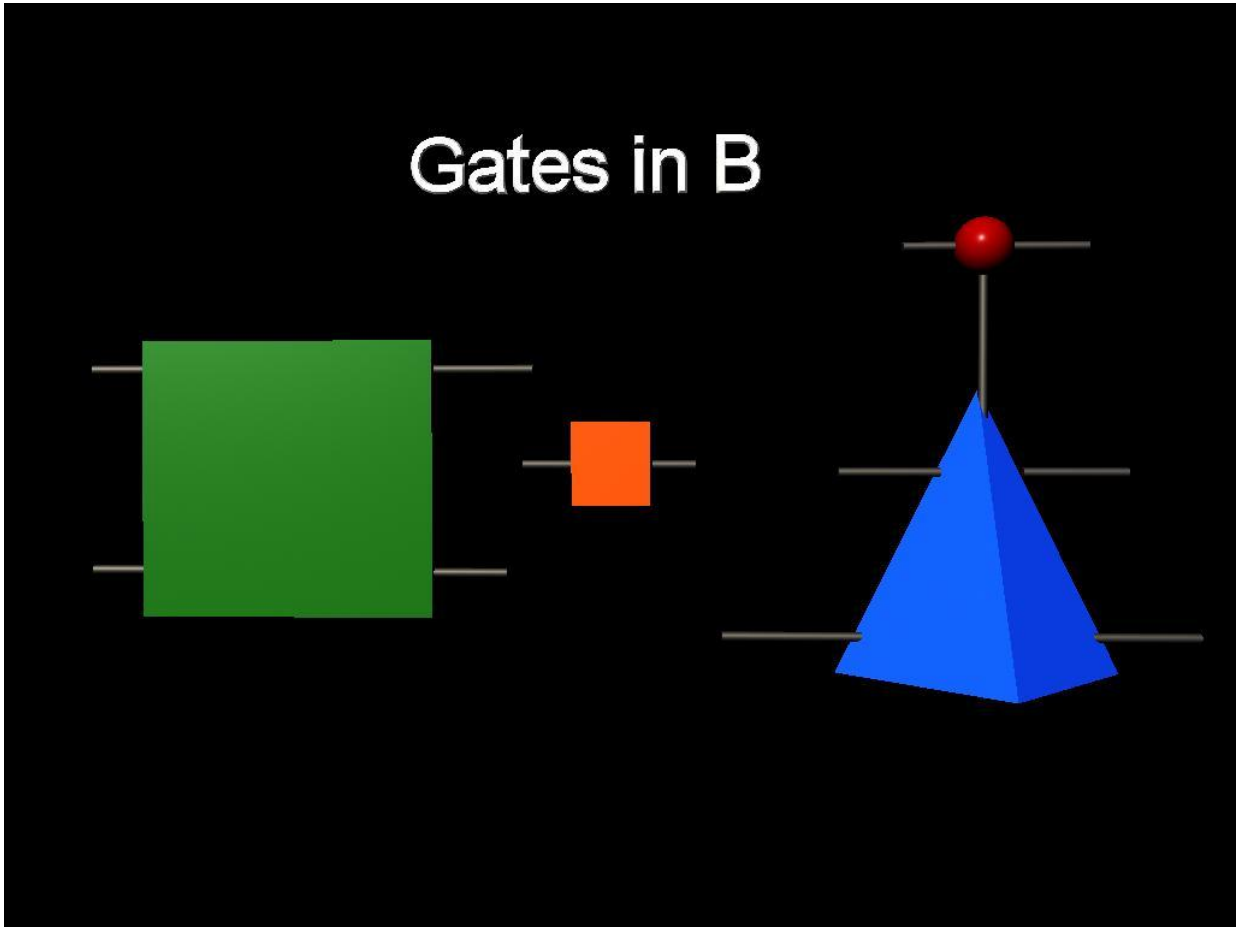
- Imagine the following scenario….



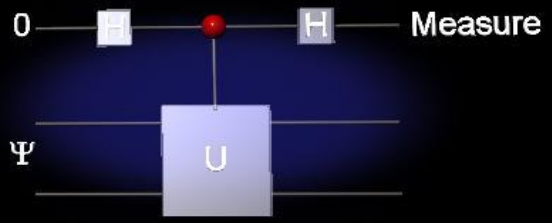
Example of Universality



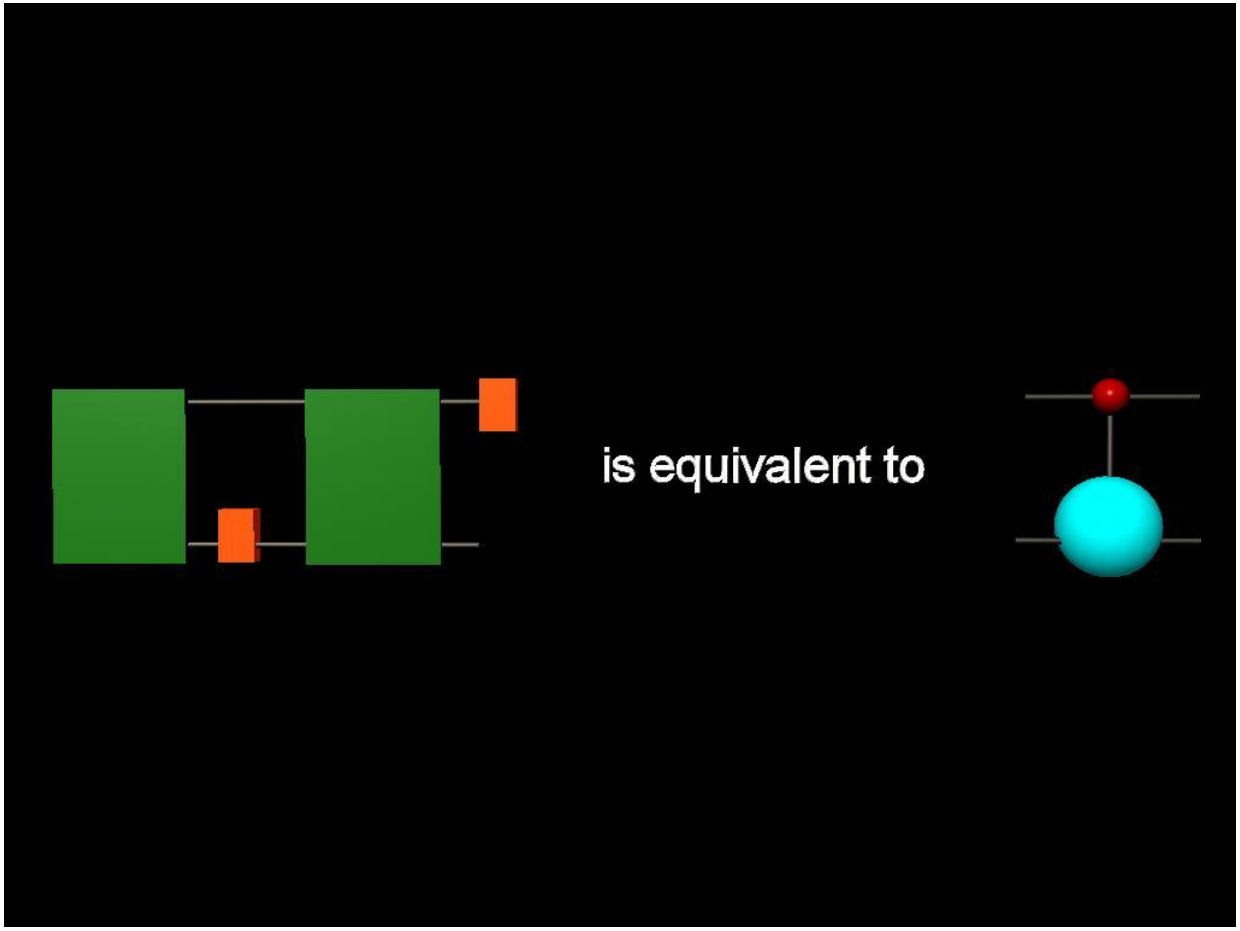
- Imagine the following scenario….



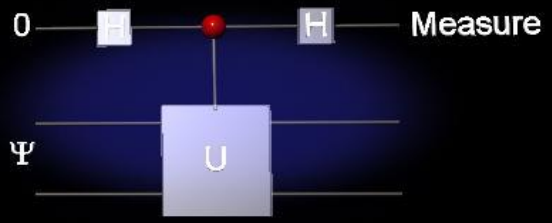
Example of Universality



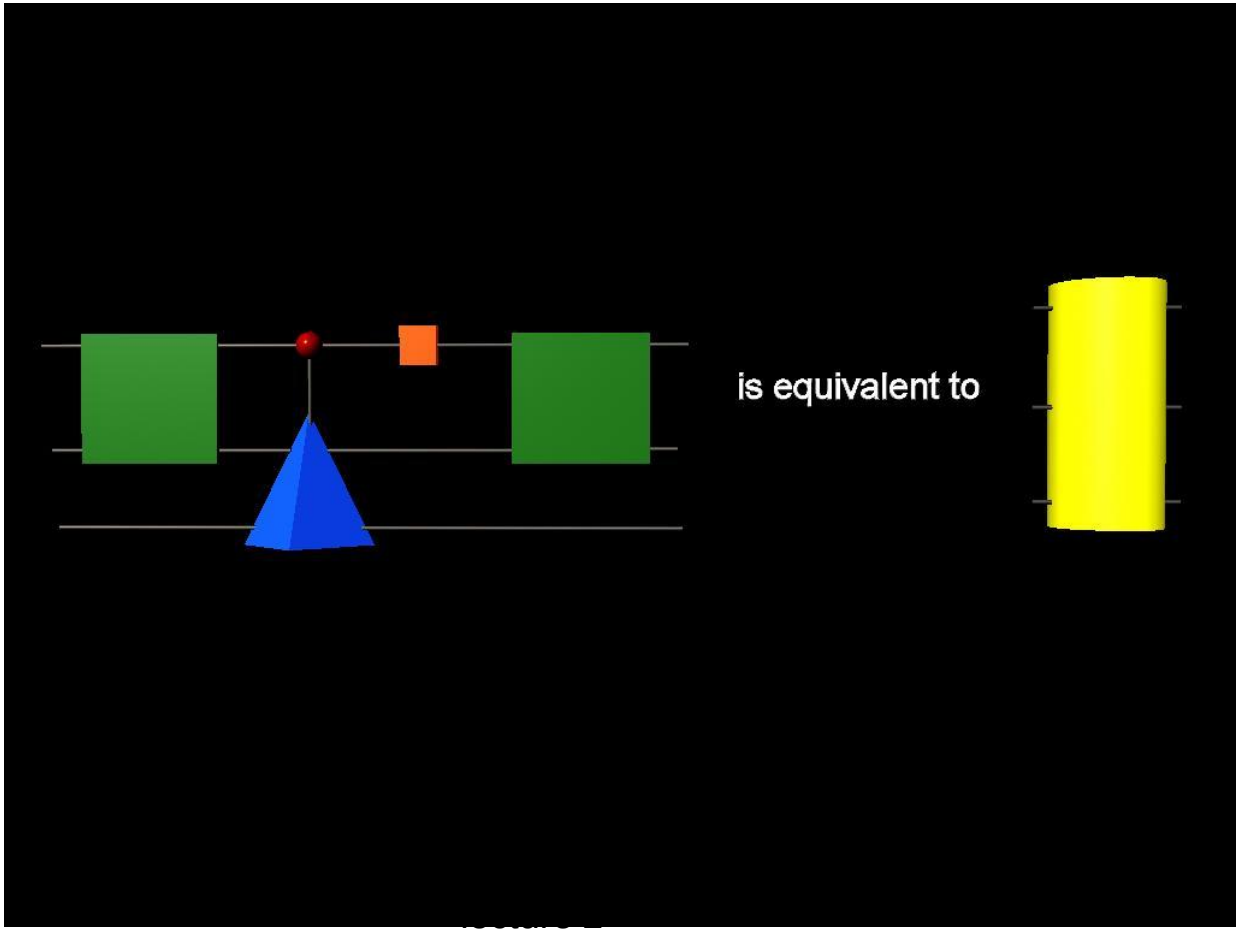
- Imagine the following scenario….



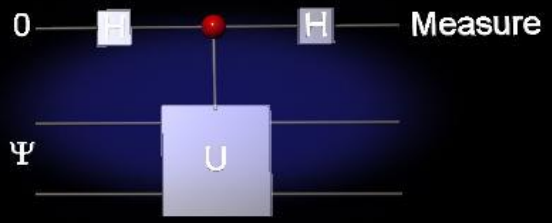
Example of Universality



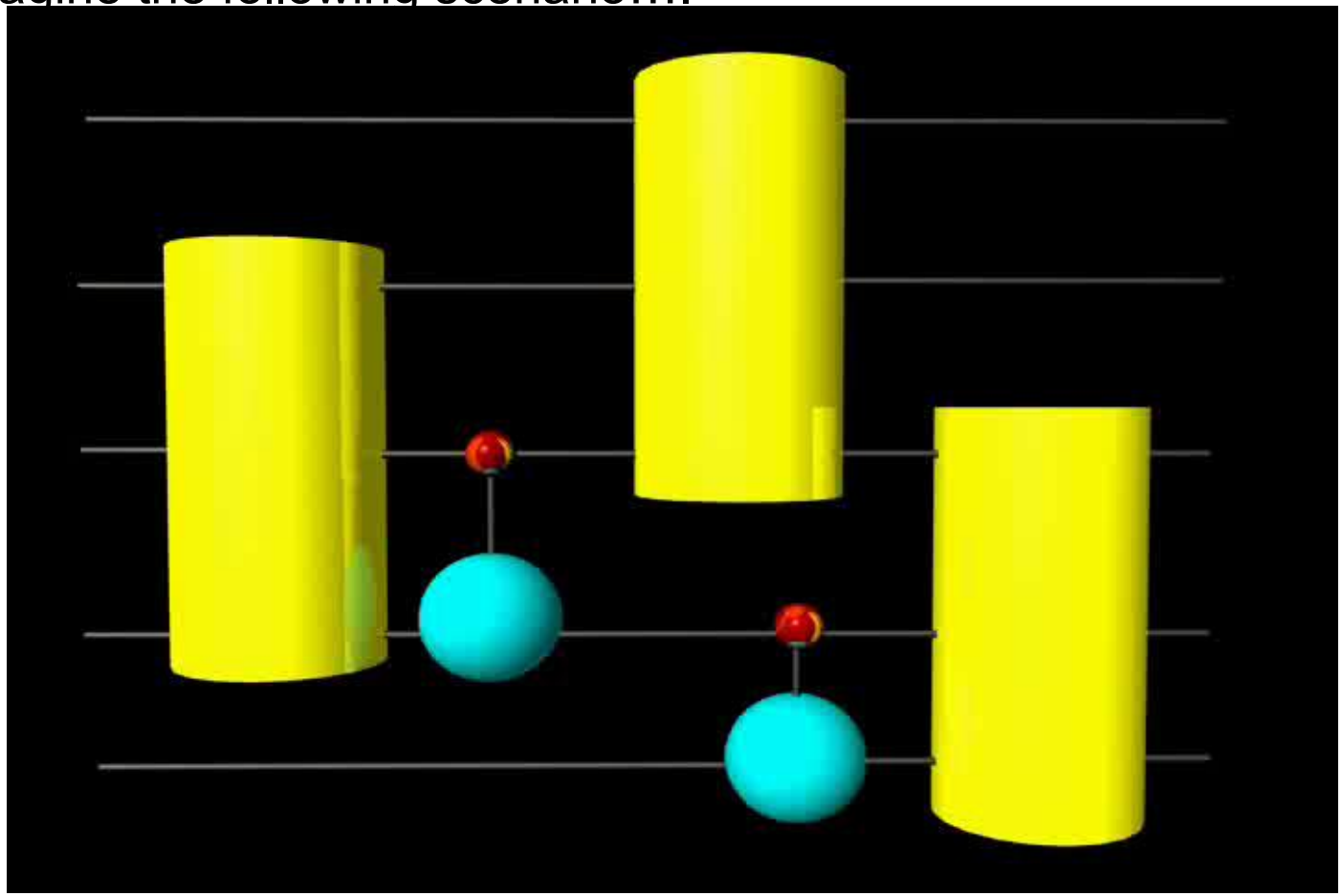
- Imagine the following scenario….



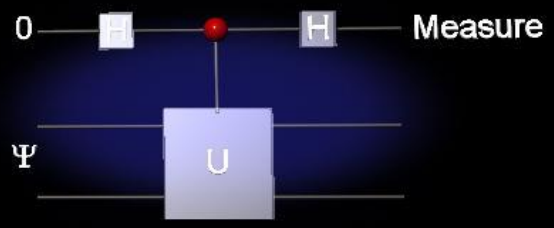
Example of Universality



- Imagine the following scenario….

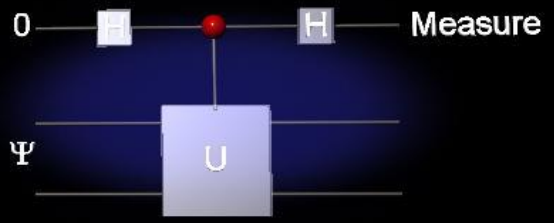


“Classical” Logic Gates



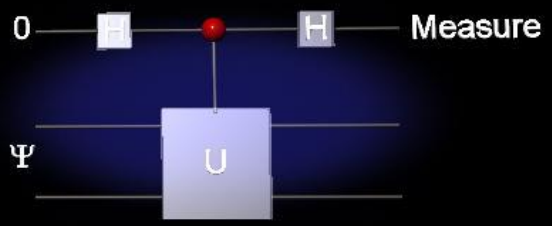
- If all physical processes are unitary (and thus reversible), a complete description of a physical process implementing the AND gate should be reversible.
- However the AND gate is not logically reversible.
- Therefore, the (non-reversible) AND gate “throws away” or “erases” information that would make it reversible.

“Classical” Logic Gates



- Landauer’s Principle (N&C,3.2.5): To erase a single bit of information dissipates at least kT log(2) amount of energy into the environment
- It was thought that dissipation of energy implied fundamental limits on real computation

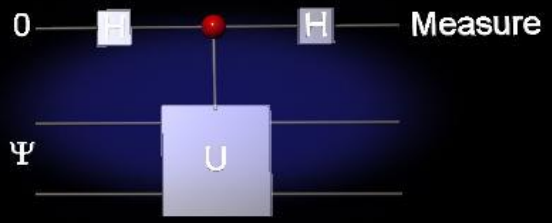
“Classical” Logic Gates



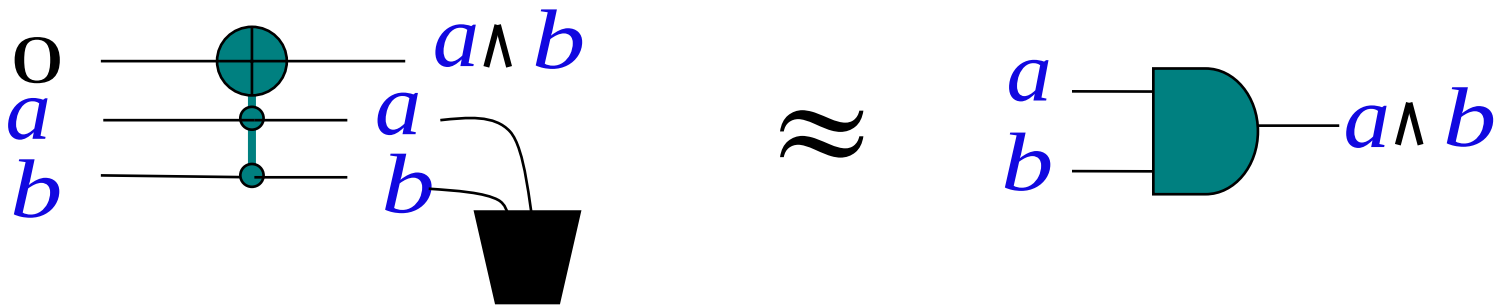
- However Bennett showed that any computation can be made reversible and therefore doesn’t in principle require energy dissipation

- Method: Replace each irreversible gate with a reversible generalization

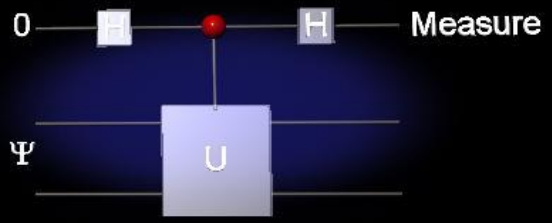
Irreversible gates from reversible ones



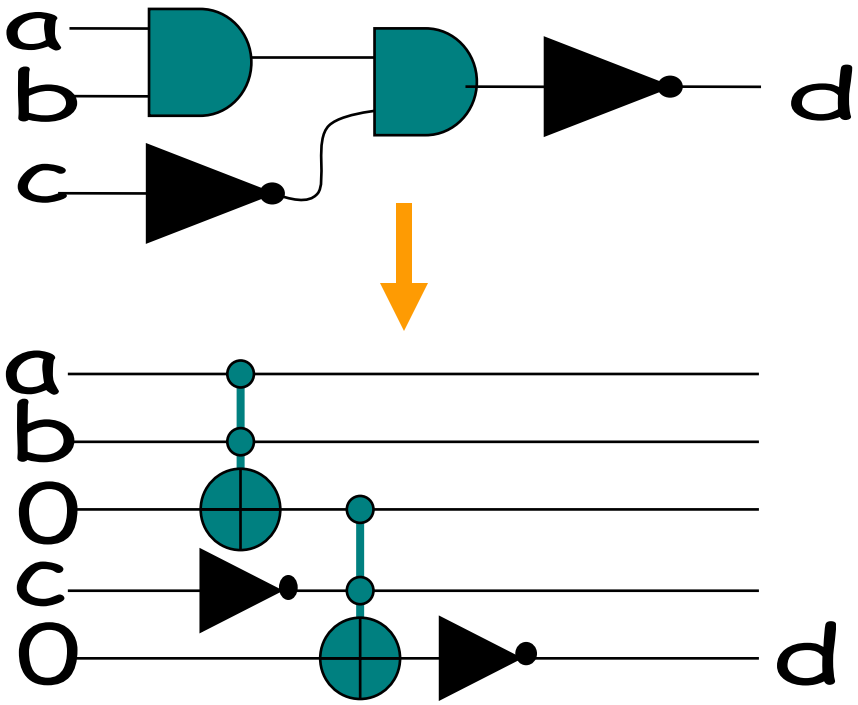
- Note that irreversible gates are really just reversible gates where we hardwire some inputs and throw away some outputs



Making reversible circuits

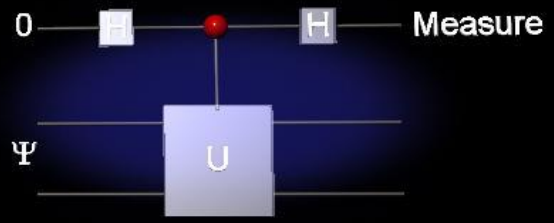


- Replace irreversible gates with their reversible counterparts



Making reversible circuits

(see Fig. 1.6 in text)



- One problem is that there will be junk left in the extra bits
- Bennett showed how to “uncompute” the junk

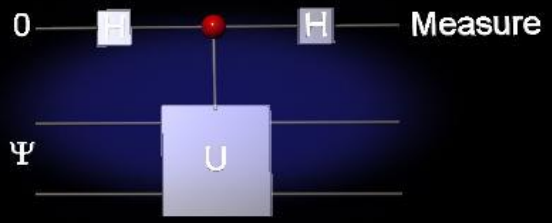
$$|x\rangle|0\rangle|0\rangle|0\rangle$$

$$\xrightarrow{compute\ f(x)} |x\rangle|f(x)\rangle|junk(x)\rangle|0\rangle$$

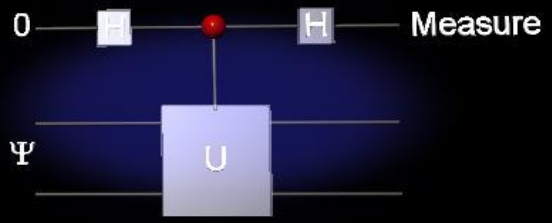
$$\xrightarrow{copy\ f(x)} |x\rangle|f(x)\rangle|junk(x)\rangle|f(x)\rangle$$

$$\xrightarrow{uncompute\ f(x)} |x\rangle|0\rangle|0\rangle|f(x)\rangle$$

Making reversible circuits

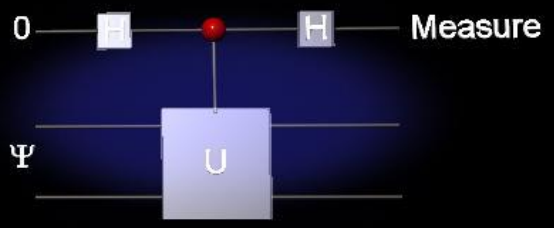


- An irreversible circuit with space S and depth (or “time”) T can thus be simulated by a reversible circuit with space in O(S+T) and time O(T)
- Bennett also showed how to implement a reversible version with time $O(T^{1+\varepsilon})$ and space O(S log(T)) or time O(T) and space $O(ST^{\varepsilon})$.

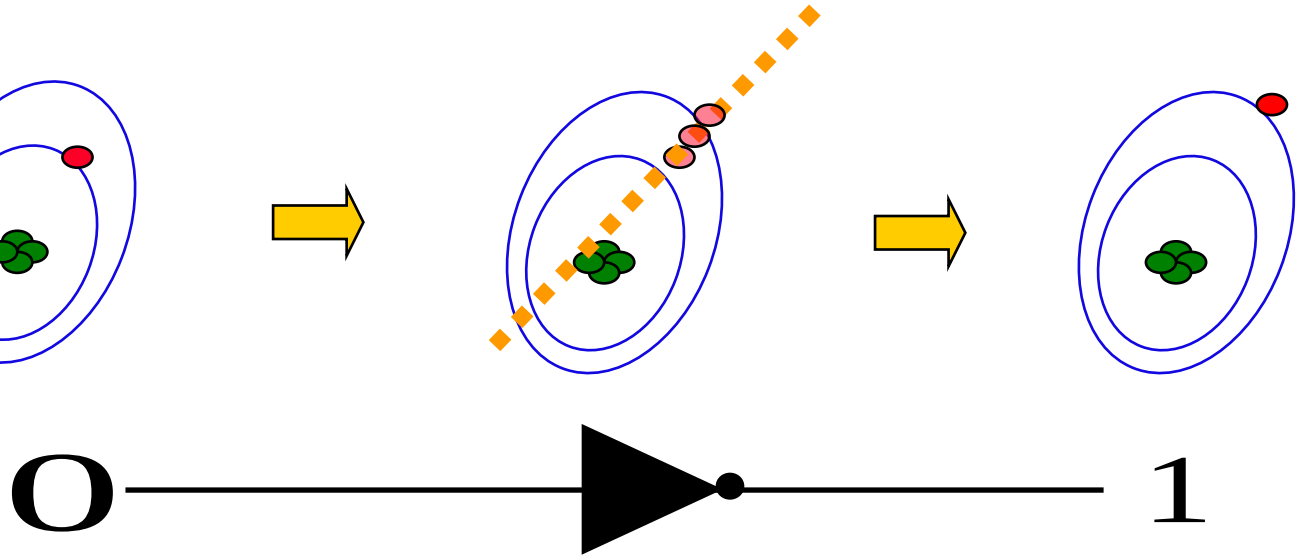


Quantum circuit models

Moving towards Quantum… A small reversible computer

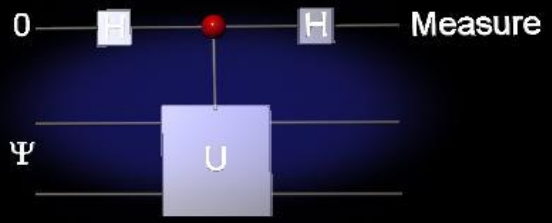


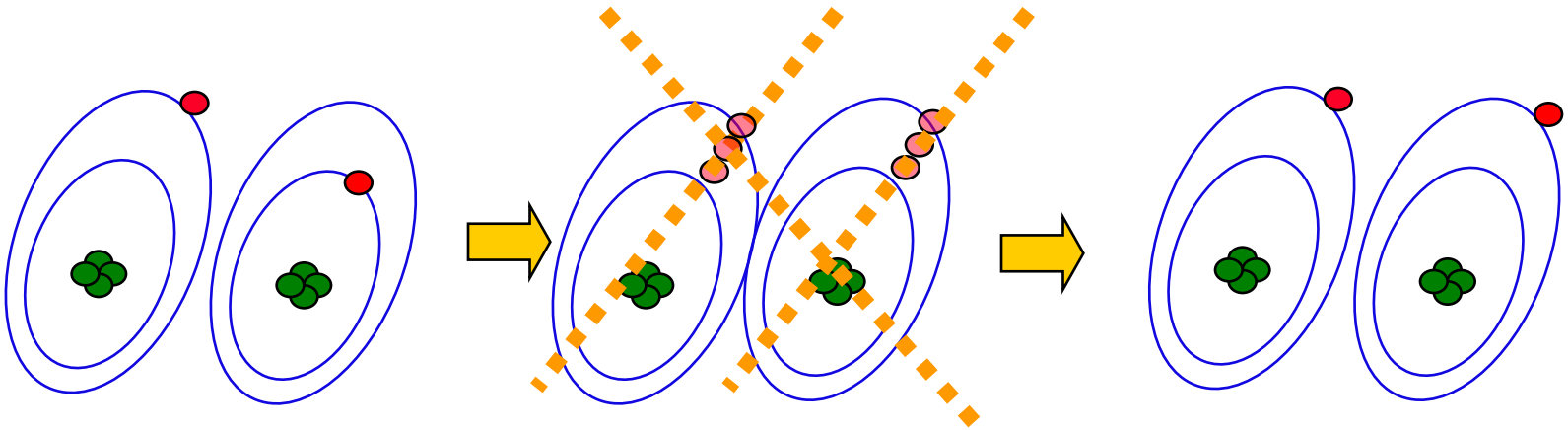
- From Classical reversible circuits to quantum circuits, we first imagine a single physical system with two discernable levels…an atom with its outer electron in either the ground or first excited state.

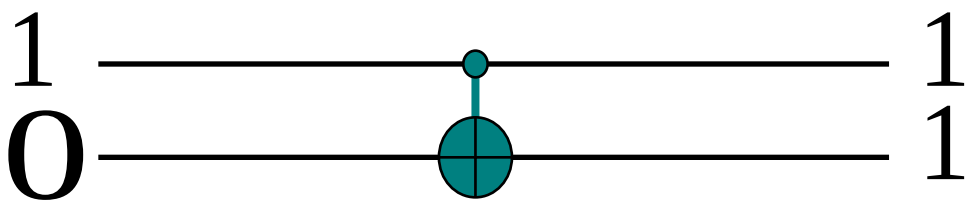


(negligible coupling to the environment)

A small reversible computer

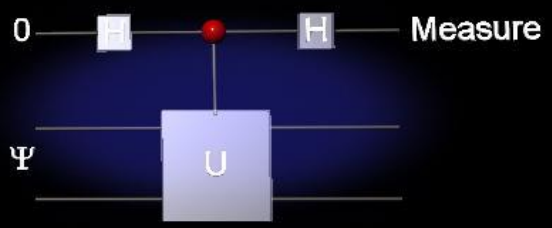


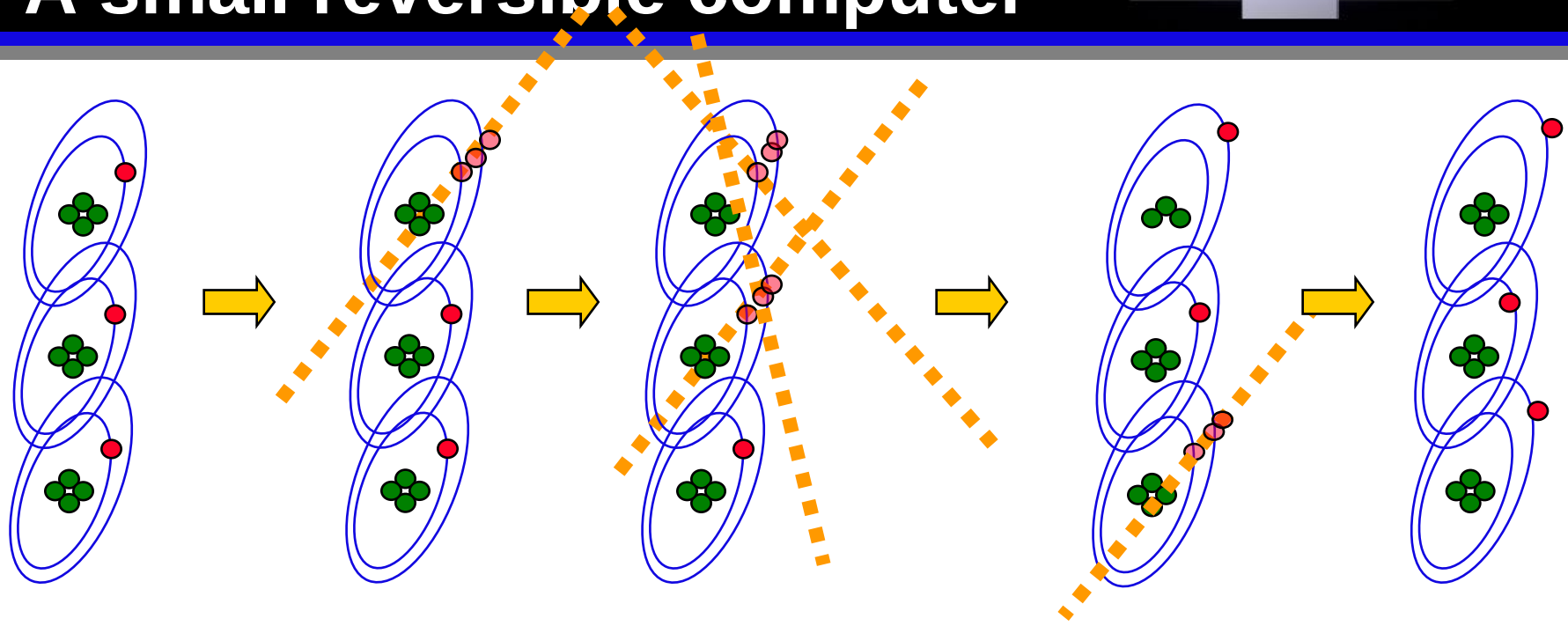


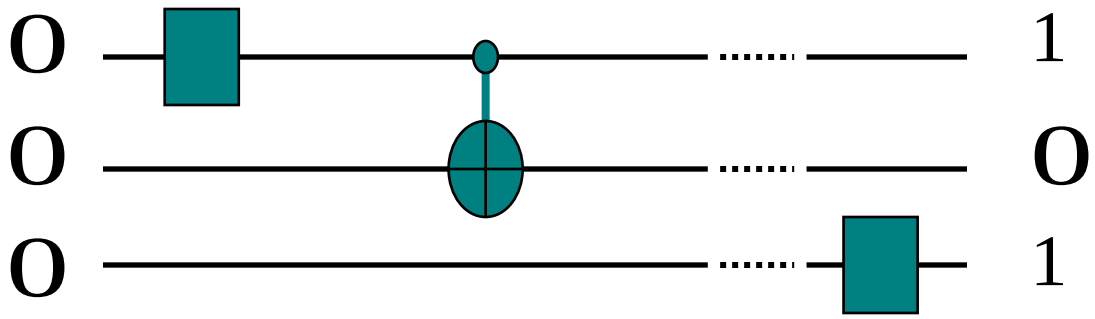


(negligible coupling to the environment)

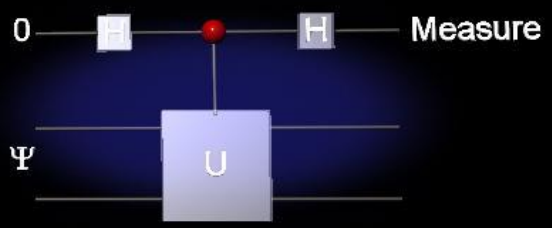
A small reversible computer

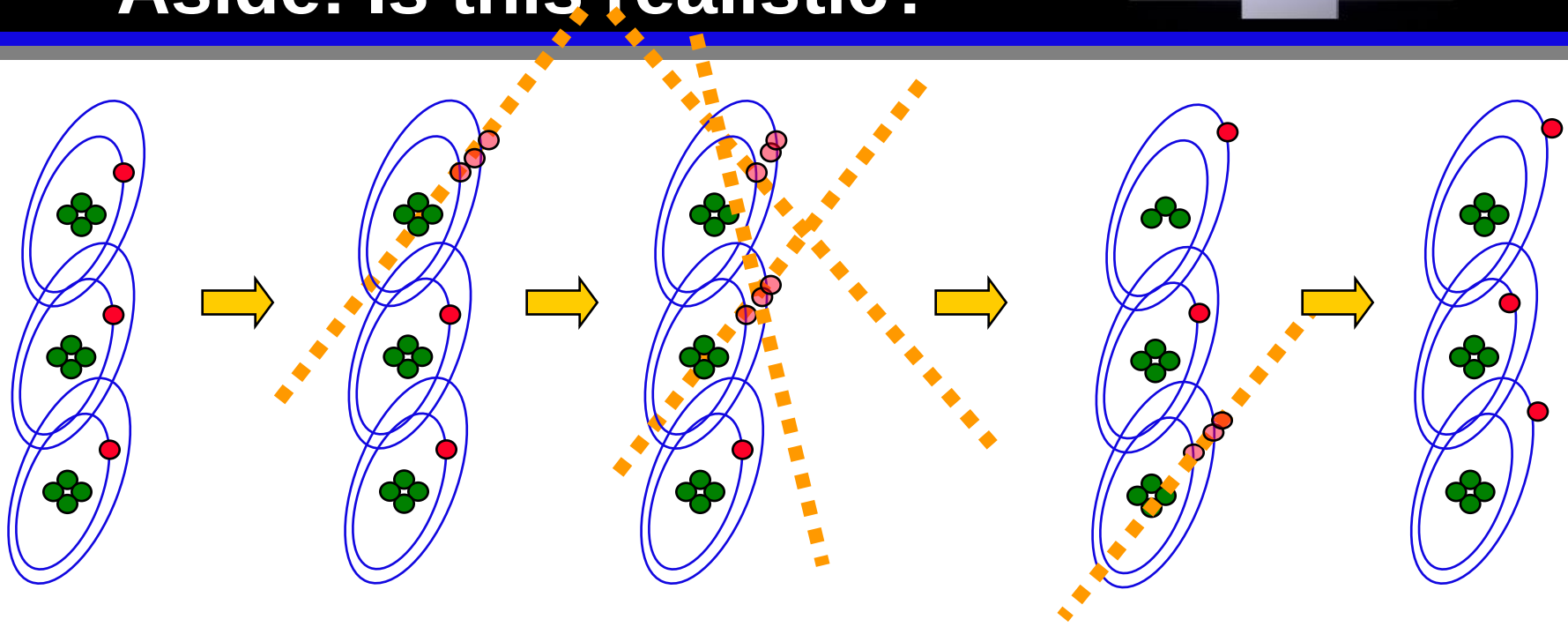






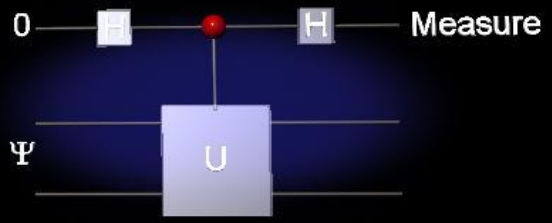
Aside: Is this realistic?





- We do have a theory of classical linear error correction. But before we worry about stabilizing this system, let's push forward its capabilities.

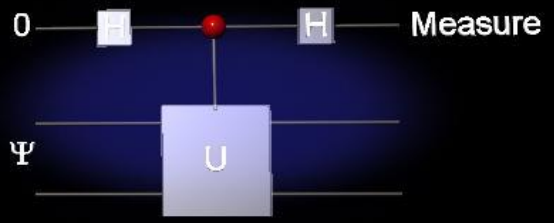
Quantum Mechanics and Information Processing



- Since physics is quantum mechanical, we need to recast the theory of information processing in a quantum mechanical framework.

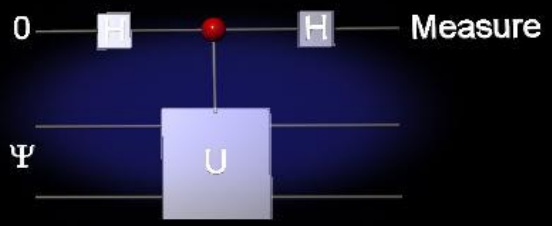
Any physical medium capable of representing 0 and 1 is in principle capable of being in a state described by $\alpha_0|0\rangle + \alpha_1|1\rangle$

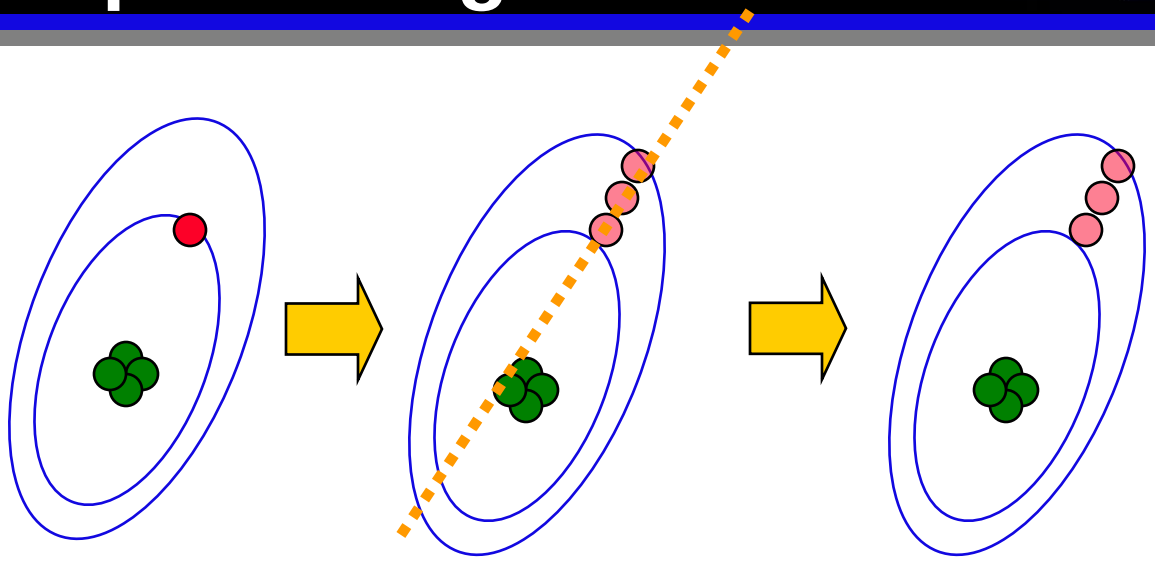
Quantum mechanics and information



- What does $\alpha_0|0\rangle + \alpha_1|1\rangle$ really mean?
- It's a "mystery". THE mystery. We don't understand it, but we can tell you how it works. (Richard Feynman)

A quantum gate

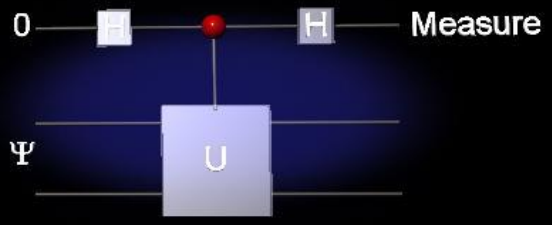




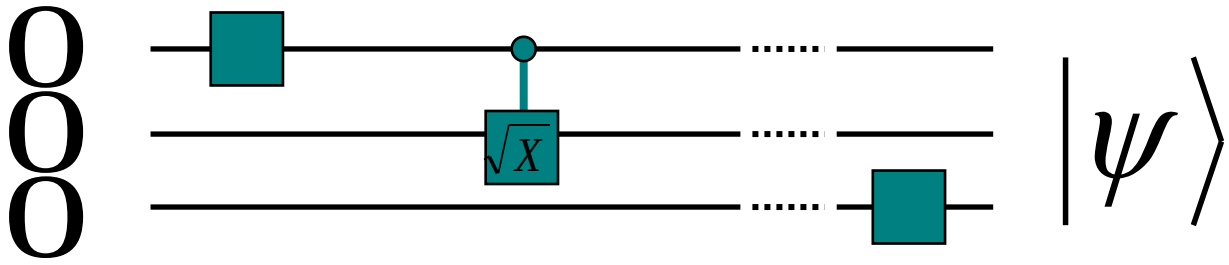
$$|0\rangle \;\text{—}\boxed{\sqrt{X}}\text{—}\; \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$$

$$|1\rangle \;\text{—}\boxed{\sqrt{X}}\text{—}\; \frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

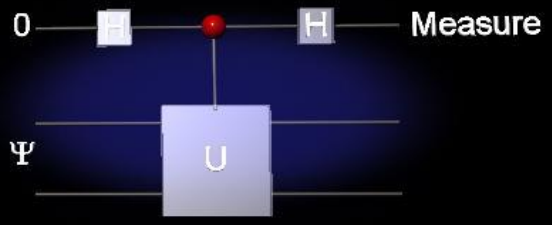
“Quantum Circuit Model”



- For this course, we will mostly focus on “reversible acyclic circuits of quantum gates”



“Quantum Circuit Model”



- This model closely resembles the model of reversible acyclic (deterministic) circuits, except we also have unitary quantum gates that create superpositions of two or more distinguishable states