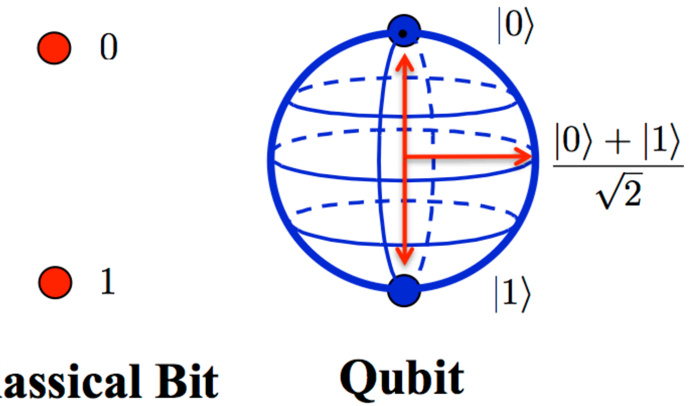
Introduction to Quantum Computing and its Applications to Cyber Security



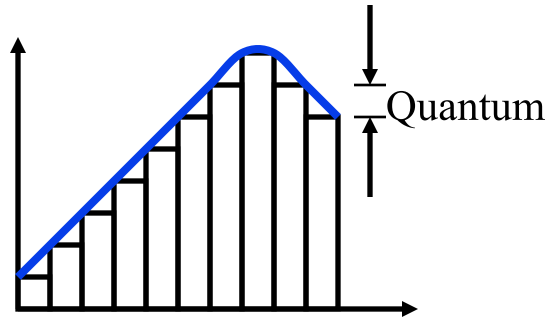
Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@wustl.edu

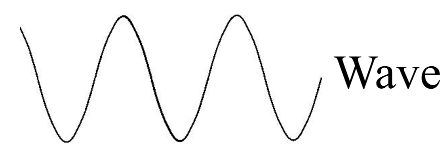
These slides and audio/video recordings of this class lecture are at:
http://www.cse.wustl.edu/~jain/cse570-19/

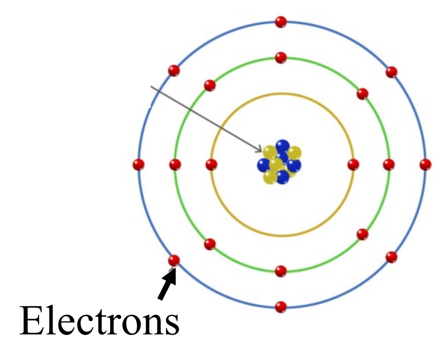
1. What is a Quantum and Quantum Bit?
2. Matrix Algebra Review
3. Quantum Gates: Not, And, or, Nand
4. Applications of Quantum Computing
5. Quantum Hardware and Programming

What is a Quantum?

- Quantization: Analog to digital conversion
- Quantum = Smallest discrete unit
- **Wave Theory**: Light is a wave. It has a frequency, phase, amplitude
- **Quantum Mechanics**: Light behaves like discrete packets of energy that can be absorbed and released
- **Photon** = One quantum of light energy
- Photons can move an electron from one energy level to next higher level
- Photons are released when an electron moves from one level to lower energy level

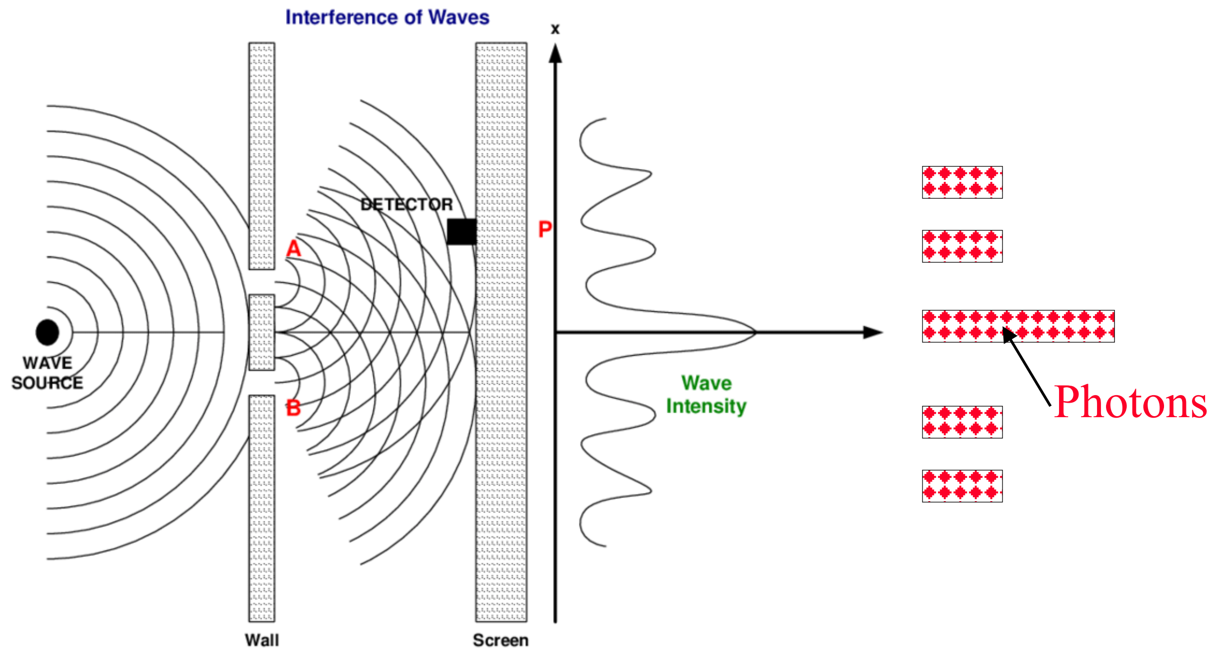






Probabilistic Behavior

- Young's Double-Slit Experiment 1801



- The two waves exiting the slits interfere.
- Interference is constructive at some spots and destructive at others ⇒ Probabilistic

Quantum Bits

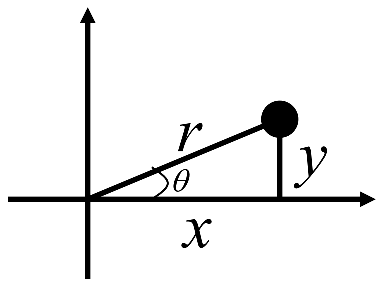
1. Computing bit is a binary scalar: 0 or 1
2. Quantum bit (**Qubit**) is a 2×1 **vector**: $\begin{bmatrix}1\\0\end{bmatrix}$ or $\begin{bmatrix}0\\1\end{bmatrix}$
3. Vector elements of Qubits are **complex numbers** $x+iy$
4. **Modulus** of a complex Number $|x+iy| = \sqrt{(x+iy)(\underbrace{x-iy}_{\text{Conjugate}})} = \sqrt{x^2+y^2}$

   Example: $|(1+2i)| = \sqrt{(1+2i)(1-2i)} = \sqrt{1+4} = \sqrt{5}$
5. Probability of each element in a qubit vector is proportional to its modulus squared $\begin{bmatrix}a_0\\a_1\end{bmatrix} \Rightarrow \begin{matrix}P = |a_0|^2/(|a_0|^2+|a_1|^2)\\ P = |a_1|^2/(|a_0|^2+|a_1|^2)\end{matrix}$

$$\begin{bmatrix}1+2i\\1-i\end{bmatrix} \Rightarrow \frac{|1+2i|}{|1-i|} = \frac{\sqrt{(1+2i)(1-2i)}}{\sqrt{(1-i)(1+i)}} = \frac{\sqrt{5}}{\sqrt{2}} \Rightarrow P = \begin{cases}5/(5+2) & = & 5/7\\ 2/(5+2) & = & 2/7\end{cases}$$

Polar Representation

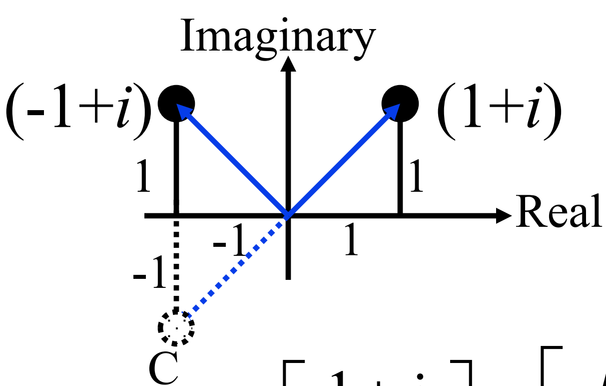
- Complex numbers in polar coordinates:



$$(x+iy)=re^{i\theta}=r\left(cos(\theta)+isin(\theta)\right)$$

$$r=\sqrt{x^2+y^2}$$

$$\theta=tan^{-1}(y/x)$$

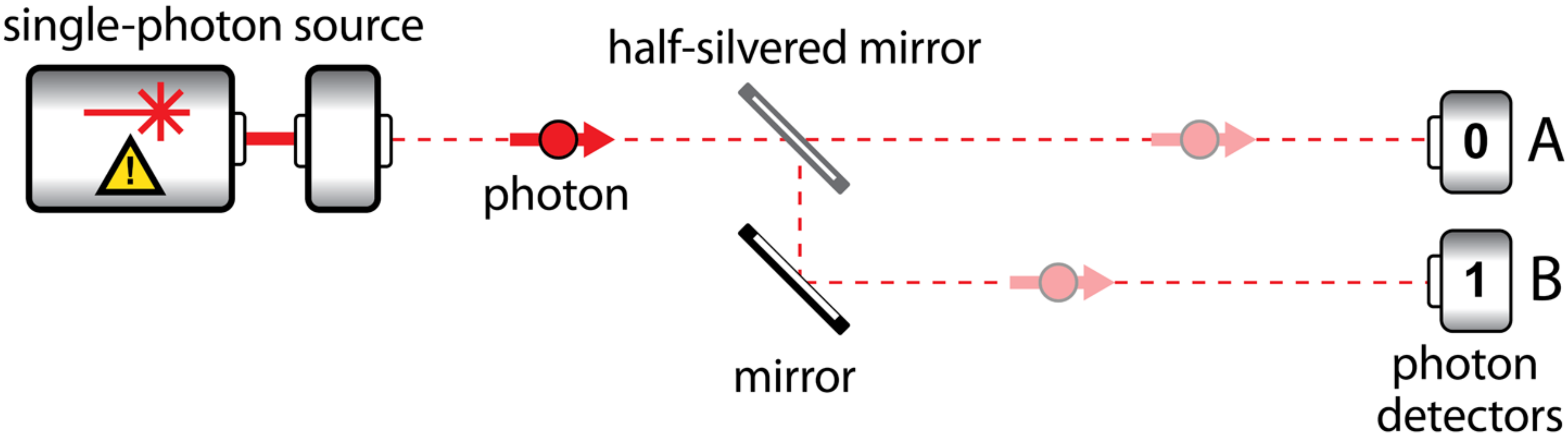


$2\pi = 360^o$
$\pi/4=45^o$

$$\begin{bmatrix}1+i\\-1+i\end{bmatrix}=\begin{bmatrix}\sqrt{2}e^{i\pi/4}\\\sqrt{2}e^{3\pi/4}\end{bmatrix}=\begin{bmatrix}\sqrt{2}\left(\cos(\pi/4)+i\sin(\pi/4)\right)\\\sqrt{2}\left(\cos(3\pi/4)+i\sin(3\pi/4)\right)\end{bmatrix}$$

- **Exercise**: Find the complex and polar representation of C

Qubit Interpretation



[Source: Johnston, et al. 2019]

- If a single photon is emitted from the source, the photon reaches position A or B with some probability
  $\Rightarrow$ Photon has a *superposition* (rather than position)
- Each position has a different path length and, therefore, different amplitude and phase

Ref: E. R. Johnston, N. Harrigan, and M. Gimeno-Segovia, "Programming Quantum Computers," O'reilly, 2019, ISBN:9781492039686, 320 pp.

Bra-Ket Notation

- The vector ψ is denoted in bra-kets $|\psi>$
- Brackets: { }, [ ], < >
- Bra $<a|$
- Ket $|a>$
- Example: Ket-zero and ket-one

$$\begin{bmatrix}1\\0\end{bmatrix} = |0> \quad \begin{bmatrix}0\\1\end{bmatrix} = |1>$$

- Bra is the transpose of the complex-conjugate of a Ket.
  Example: Bra-zero and Bra-one

$$\begin{bmatrix}1 & 0\end{bmatrix} = <0| \quad \begin{bmatrix}0 & 1\end{bmatrix} = <1|$$

Matrix Multiplication

- Matrix multiplication ×:

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \times \begin{bmatrix} b_{01} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix}$$

$$= \begin{bmatrix} a_{00}b_{00} + a_{01}b_{10} + a_{02}b_{20} & a_{00}b_{01} + a_{01}b_{11} + a_{02}b_{21} \\ a_{10}b_{00} + a_{11}b_{10} + a_{12}b_{20} & a_{10}b_{01} + a_{11}b_{11} + a_{12}b_{21} \end{bmatrix}$$

- Example:

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix}$$

3×2 **2×3** **3×3**

Tensor Product

- **Tensor Product** $\otimes$: $m \times n \otimes k \times l$ results in $mk \times nl$ matrix

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} B = \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix}$$

$$A \otimes B = \begin{bmatrix} a_{00}\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix} & a_{01}\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix} \\ a_{10}\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix} & a_{11}\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} a_{00}b_{00} & a_{00}b_{01} & a_{00}b_{02} & a_{01}b_{00} & a_{01}b_{01} & a_{01}b_{02} \\ a_{00}b_{10} & a_{00}b_{11} & a_{00}b_{12} & a_{01}b_{10} & a_{01}b_{11} & a_{01}b_{12} \\ a_{00}b_{20} & a_{00}b_{21} & a_{00}b_{22} & a_{01}b_{20} & a_{01}b_{21} & a_{01}b_{22} \\ a_{10}b_{00} & a_{10}b_{01} & a_{10}b_{02} & a_{11}b_{00} & a_{11}b_{01} & a_{11}b_{02} \\ a_{10}b_{10} & a_{10}b_{11} & a_{10}b_{12} & a_{11}b_{10} & a_{11}b_{11} & a_{11}b_{12} \\ a_{10}b_{20} & a_{10}b_{21} & a_{10}b_{22} & a_{11}b_{20} & a_{11}b_{21} & a_{11}b_{22} \end{bmatrix}$$

Tensor Product (Cont)

- Example 1:

$$\begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \end{bmatrix} \otimes \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix} = \begin{bmatrix} a_{00} \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix} \\ a_{10} \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix} \\ a_{20} \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_{00}b_{00} \\ a_{00}b_{10} \\ a_{10}b_{00} \\ a_{10}b_{10} \\ a_{20}b_{00} \\ a_{20}b_{10} \end{bmatrix}$$

3×1 **2×1** **6×1**

- Example 2: $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$

2×2 **1×3** **2×6**

Multiple Qubits and QuBytes

One Qbit: $|0> = \begin{bmatrix}1\\0\end{bmatrix}$ $|1> = \begin{bmatrix}0\\1\end{bmatrix}$

Two Qbits: $|00> = |0> \otimes |0> = \begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$ $|01> = \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$ $|10> = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$ $|11> = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

Tensor Product

Three Qbits:

$$\begin{array}{cccccccc}
\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} & \begin{bmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} & \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} & \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{bmatrix} & \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{bmatrix} & \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} & \begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{bmatrix} & \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{bmatrix} \\
|000> & |001> & |010> & |011> & |100> & |101> & |110> & |111>
\end{array}$$

- In a k-qubit register, each of the 2^k positions can be any complex number
- QuByte=8-Qubits = 256-element vector

Homework 19A

- Given two matrices:

$$A = \begin{bmatrix} 1+i & 1 \\ 1-i & i \end{bmatrix} B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

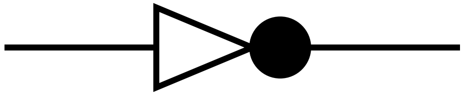
- Compute: $A \times B,\ A \otimes B$

- Compute the probabilities of each element of $A \times B$

Quantum Gates

1. Quantum NOT Gate
2. Quantum AND Gate
3. Quantum OR Gate
4. Quantum NAND Gate
5. Quantum √NOT Gate

Quantum NOT Gate

- **NOT:** 

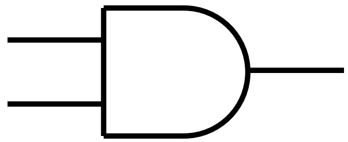
$$NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} ? \\ ? \end{bmatrix}$$

$$NOT\,|1> \;=\; |0> \qquad NOT\,|0> \;=\; |?>$$

- **Exercise:** Fill in the ?'s

Quantum AND Gate

- **AND:** 

$$\text{AND} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

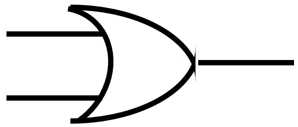
$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} ? \\ ? \end{bmatrix}$$

$$\text{AND} \quad |00> \quad |01> \quad |10> \quad |11> = |0> \quad |0> \quad |0> \quad |?>$$

$$2\times 4 \qquad 4\times 1 \qquad 2\times 1$$

- **Exercise:** Fill in the ?'s

Quantum OR Gate

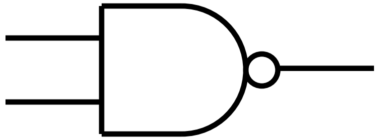
- **OR:** 

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

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\text{OR} \qquad |00> \quad |01> \quad |10> \quad |11> = \quad |0> \quad |1> \quad |1> \quad |1>$$

Quantum NAND Gate

- **NAND:** 

$$\text{NAND} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\text{NAND} \qquad |00> \quad |01> \quad |10> \quad |11> = \quad |1> \quad |1> \quad |1> \quad |0>$$

Quantum √NOT Gate

- **√NOT:** √NOT × √NOT = NOT

$$\sqrt{\text{NOT}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \times \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{|1>} = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

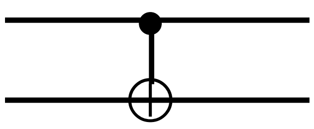
$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \times \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{|0>} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \times \frac{1}{\sqrt{2}}\begin{bmatrix} -1 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} -1 \\ 0 \end{bmatrix}}_{|0>}$$

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \times \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{|1>}$$

Controlled NOT Gate

- **CNOT:** If the control bit is 0, no change to the 2nd bit
  If control bit is 1, the 2nd bit is complemented



$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\underset{\text{CNOT}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}} \times \underset{|00>}{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \quad \underset{|01>}{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} \quad \underset{|10>}{\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}} \quad \underset{|11>}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}} = \underset{|00>}{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \quad \underset{|01>}{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} \quad \underset{|11>}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}} \quad \underset{|10>}{\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}}$$

- Controlled NOT gate can be used to produce two bits that are **entangled** ⇒ Two bits behave similarly even if far apart
  ⇒ Can be used for teleportation of information

Quantum Gates: Summary

$$\left.\begin{array}{ll}
\text{NOT}=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \text{AND}=\begin{bmatrix}1 & 1 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\\[2ex]
\text{OR}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 1 & 1\end{bmatrix} & \text{NAND}=\begin{bmatrix}0 & 0 & 0 & 1\\ 1 & 1 & 1 & 0\end{bmatrix}
\end{array}\right\}\text{Classical}$$

$$\left.\begin{array}{ll}
\sqrt{\text{NOT}}=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix} & \text{CNOT}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}
\end{array}\right\}\text{Non-Classical}$$

- The first 4 gates above are similar to the classical gates. The last two are non-classical gate.
- There are many other classical/non-classical quantum gates, e.g., Rotate, Copy, Read, Write, …
- Using such gates one can design **quantum circuits**

Quantum Applications

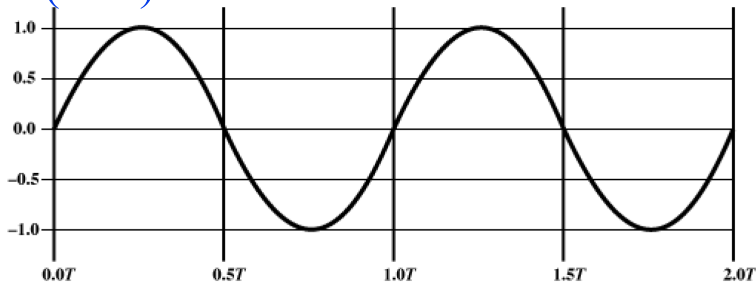
- It has been shown that quantum computation makes several problems easy that are hard currently. Including:
  - **Fourier Transforms**
  - **Factoring large numbers**
  - Error correction
  - Searching a large unordered list
- There are some new methods:
  - Quantum Key Exchange
  - Quantum Teleportation (transfer states from one location to another)

Quantum-Safe Cryptography is being standardized

Fourier Transforms

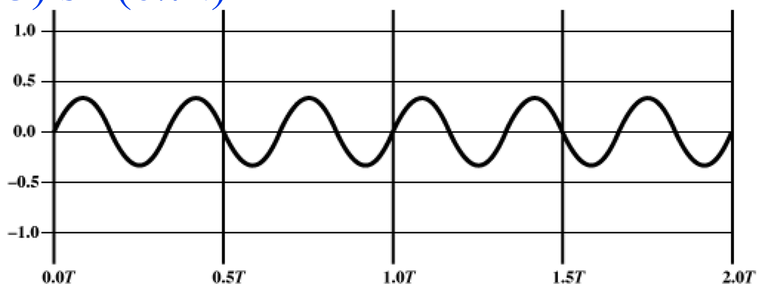
Time Domain

A sin(2πft)



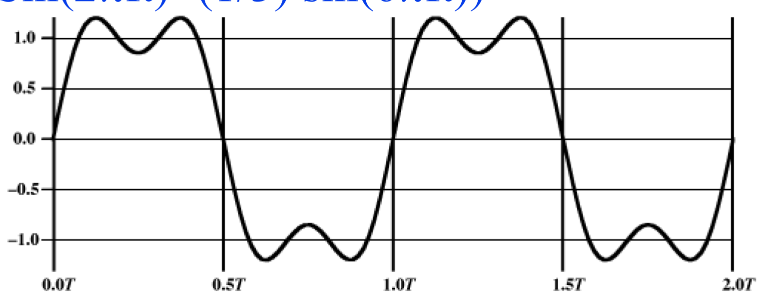
(a) sin (2πft)

(A/3) sin(6πft)



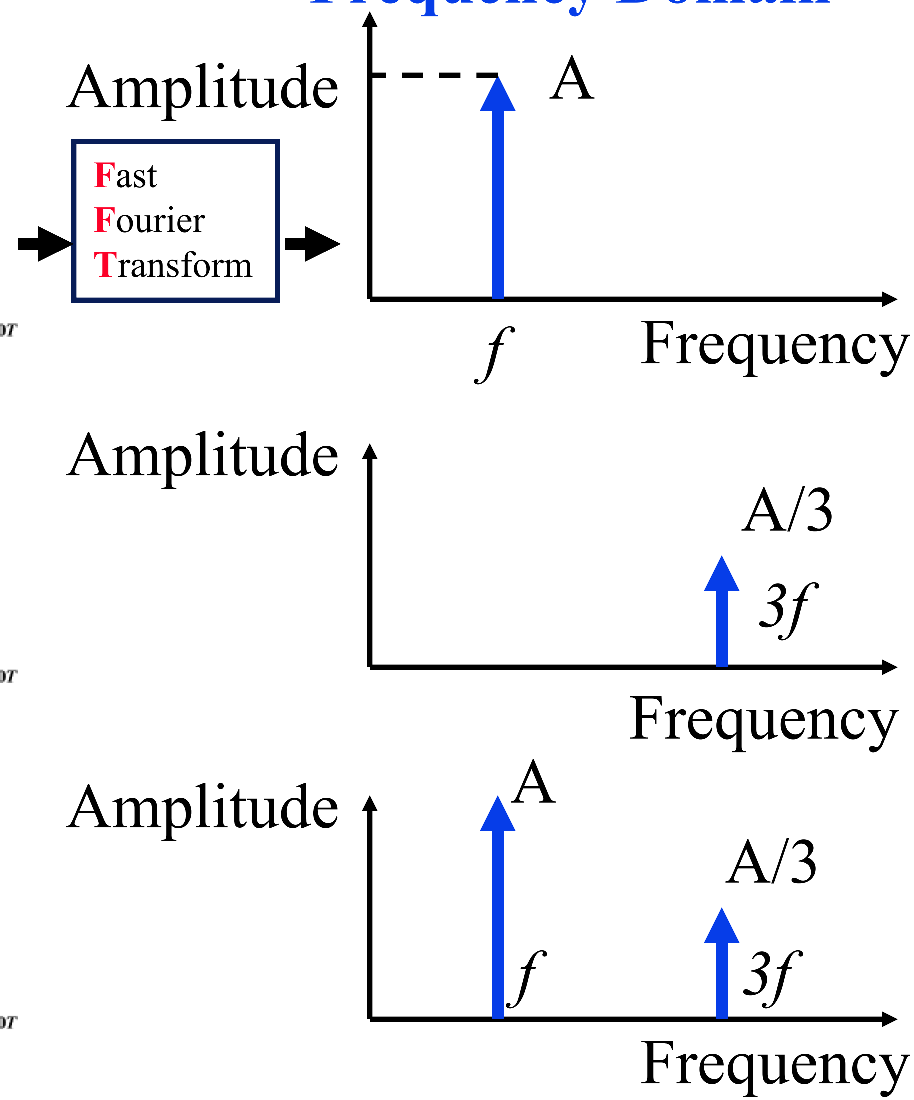
(b) (1/3) sin (2π(3f)t)

A(Sin(2πft)+(1/3) sin(6πft))



(c) (4/π) [sin (2πft) + (1/3) sin (2π(3f)t)]

Frequency Domain



Quantum Fourier Transform (QFT)

- Fourier transform is used to find periodic components of signals
- Conventional computing requires O($n2^n$) gates,
  n = # of bits in the input register = Size of input numbers
  $\Rightarrow$ Exponential in n
- Quantum computing allows Fourier transforms using O(m^2) quantum gates, m = # of qubits in the q-registers
  $\Rightarrow$ Polynomial in m
- QFT is faster than classical FT for large *inputs*

GCD

- Greatest Common Divisor of any two numbers
  - Divide the larger number with the smaller number and get the remainder less than the divisor
  - Divide the previous divisor with the remainder
  - Continue this until the remainder is zero.
    The last divisor is the GCD

15) 35 (2
30
05) 15 (3
15
gcd
0

Shor's Factoring Algorithm

- Peter Shor used QFT and showed that Quantum Computers can find prime factors of large numbers exponentially faster than conventional computers
- **Step 1:** Find the period of a^i mod N sequence. Here a is co-prime to $N \Rightarrow a$ is a prime such that gcd(a, N) = 1 $\Rightarrow$ a and N have no common factors.
  - Example: N=15, a=2; 2^i mod 15 for i=0, 1, 2, … = 1, 2, 4, 8, 1, … $\Rightarrow p=4$
  - This is the classical method for finding period. QFT makes it fast.
- **Step 2:** Prime factors of N might be $gcd(N, a^{p/2}+1)$ and $gcd(N, a^{p/2}-1)$
  - Example: $gcd(15, 2^2-1) = 3$; $gcd(15, 2^2+1) = 5$;

Homework 19B

- Find factors of 35 using Shor's algorithm. Show all steps.
- Optional: Try factoring 407 (Answer: 11×37)

Quantum Machine Learning (QML)

- Quantum for solving systems of linear equation
- Quantum Principal Component Analysis
- Quantum Support Vector Machines (QSVM)
  - Classical SVM has runtime of O(poly(m,n)), m data points, n features
  - QSVM has runtime of O(log(mn))
    - Currently limited to data that can be represented with small number of qubits
- QML can process data directly from Quantum sensors with full range of quantum information

Ref: E. R. Johnston, N. Harrigan, and M. Gimeno-Segovia, "Programming Quantum Computers," O'reilly, 2019, ISBN:9781492039686, 320 pp.

Building Quantum Computers

1. **Neural Atom**: Group of cesium or rubidium atoms are cooled down to a few degree Kelvin and controlled using lasers
2. **Nuclear Magnetic Resonance** (NMR)
3. **Nitrogen-Vacancy Center-in-Diamond**: Some carbon atoms in diamond lattice are replaced by nitrogen atoms
4. **Photonics**: Mirrors, beam splitters, and phase shifters are used to control photons
5. **Spin Qubits**: Using semiconductor materials
6. **Topological Quantum Computing**: Uses Anyon which are quasi-particles different from photons or electrons
7. **Superconducting Qubits**: Requires cooling down to 10mK

Ref: J. D. Hidary, "Quantum Computing: An Applied Approach," Springer, 2019, 380 pp.

Quantum Hardware

- IBM Q Experience: 5-Qubit quantum processor
  Open to public for experiments using their cloud,
  https://www.ibm.com/quantum-computing/technology/experience/

Ref: https://www.ibm.com/blogs/research/2018/04/ibm-startups-accelerate-quantum/

Quantum Hardware (Cont)

- Google's Quantum computer in Santa Barbara Lab

Ref: https://www.nbcnews.com/mach/science/google-claims-quantum-computing-breakthrough-ibm-pushes-back-ncna1070461

Quantum Simulators

- QCEngine: https://oreilly-qc.github.io/
- Qiskit, https://qiskit.org/
  - Qiskit OpenQASM (Quantum Assembly Language), https://github.com/QISKit/openqasm/blob/master/examples/generic/adder.qasm
- Q# (Qsharp), https://docs.microsoft.com/en-gb/quantum/?view=qsharp-preview
- Cirq, https://arxiv.org/abs/1812.09167
- Forest, https://www.rigetti.com/forest
- List of QC Simulators, https://quantiki.org/wiki/list-qc-simulators
- See the complete list at: https://en.wikipedia.org/wiki/Quantum_programming

Ref: E. R. Johnston, N. Harrigan, and M. Gimeno-Segovia, "Programming Quantum Computers," O'reilly, 2019, ISBN:9781492039686, 320 pp.

Quantum Supremacy

- Quantum Supremacy: Solve a problem on quantum computer that can not be solved on a classical computer
- Google announced it has achieved Quantum Supremacy on October 23, 2019
  - Google built a 54-qubit quantum computer using programmable superconducting processor
- Vendors: IBM, Microsoft, Google, Alibaba Cloud, D-Wave Systems, 1QBit, QC Ware, QinetiQ, Rigetti Computing, Zapata Computing
- Global Competition: China, Japan, USA, EU are also competing

Ref: F. Arute, K. Arya, R. Babbush, *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature* **574,** 505–510 (Oct. 23, 2019), https://www.nature.com/articles/s41586-019-1666-5

Summary

1. Qubits are two element vectors. Each element is a complex number that indicate the probability of that level
2. Multi-qubits are represented by tensor products of single-qubits
3. Qbit operations are mostly matrix operations. The number of possible operations is much larger than the classic computing.
4. Shor's factorization algorithm is an example of algorithms that can be done in significantly less time than in classic computing
5. Quantum computing is here. IBM, Microsoft, Google all offer platforms that can be used to write simple quantum computing programs and familiarize yourself.
6. Quantum-Safe Crypto is in standardization

Reading List

- J. D. Hidary, "Quantum Computing: An Applied Approach," Springer, 2019, 380 pp.
- Mercedes Gimeno-Segovia, Nic Harrigan, Eric R. Johnston, "Programming Quantum Computers," O'Reilly Media, Inc., July 2019, ISBN:9781492039686 (Safari Book). **Recommended.**
- N. S. Yanofsky and M. A. Mannucci, "Quantum Computing for Computer Scientists," Cambridge, 2008, 380 pp.
- N. D. Mermin, "Quantum Computer Science: An Introduction," Cambridge, 2007, 220 pp.

References

- Gerd Leuchs, Dagmar Bruss, "Quantum Information," 2 Volume Set, 2nd Edition, Wiley-VCH, June 2019, ISBN:9783527413539 (Safari Book).
- Vladimir Silva, "Practical Quantum Computing for Developers: Programming Quantum Rigs in the Cloud using Python, Quantum Assembly Language and IBM QExperience," Apress, December 2018, ISBN:9781484242186 (Safari Book).
- Mingsheng Ying, "Foundations of Quantum Programming," Morgan Kaufmann, March 2016, ISBN:9780128025468 (Safari Book).
- F.J. Duarte, "Quantum Optics for Engineers," CRC Press, November 2017, ISBN:9781351832618 (Safari Book).
- Quantum Algorithm Zoo, (Compiled list of Quantum algorithms), http://quantumalgorithmzoo.org/

Wikipedia Links

- https://en.wikipedia.org/?title=Inner-product&redirect=no
- https://en.wikipedia.org/wiki/Bra%E2%80%93ket_notation
- https://en.wikipedia.org/wiki/Complex_number
- https://en.wikipedia.org/wiki/Controlled_NOT_gate
- https://en.wikipedia.org/wiki/Dot_product
- https://en.wikipedia.org/wiki/Fourier_transform
- https://en.wikipedia.org/wiki/Greatest_common_divisor
- https://en.wikipedia.org/wiki/List_of_quantum_processors
- https://en.wikipedia.org/wiki/Matrix_multiplication
- https://en.wikipedia.org/wiki/Polar_coordinate_system
- https://en.wikipedia.org/wiki/Quantum
- https://en.wikipedia.org/wiki/Quantum_algorithm
- https://en.wikipedia.org/wiki/Quantum_computing
- https://en.wikipedia.org/wiki/Quantum_entanglement
- https://en.wikipedia.org/wiki/Quantum_error_correction
- https://en.wikipedia.org/wiki/Quantum_Fourier_transform

Wikipedia Links (Cont)

- https://en.wikipedia.org/wiki/Quantum_logic_gate
- https://en.wikipedia.org/wiki/Quantum_machine_learning
- https://en.wikipedia.org/wiki/Quantum_mechanics
- https://en.wikipedia.org/wiki/Quantum_simulator
- https://en.wikipedia.org/wiki/Quantum_supremacy
- https://en.wikipedia.org/wiki/Quantum_technology
- https://en.wikipedia.org/wiki/Quantum_teleportation
- https://en.wikipedia.org/wiki/Qubit
- https://en.wikipedia.org/wiki/Shor%27s_algorithm
- https://en.wikipedia.org/wiki/Superconducting_quantum_computing
- https://en.wikipedia.org/wiki/Sycamore_processor
- https://en.wikipedia.org/wiki/Tensor_product
- https://en.wikipedia.org/wiki/Timeline_of_quantum_computing
- https://en.wikipedia.org/wiki/Category:Quantum_gates

Classic Papers on Quantum Computing

- R. P. Feynman, “Simulating Physics with Computers,” *International journal of theoretical physics* 21.6 (1982): 467-488, http://www.springerlink.com/index/t2x8115127841630.pdf
- D. E. Deutsch, “Quantum theory, the Church-Turing principle and the universal quantum computer,” *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 400.1818 (1985): 97-117, , https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1985.0070
- D. E. Deutsch, “Quantum Computational Networks,” *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 425.1868 (1989), 73-90. , https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.1989.0099 (subscribers only)
- P. W. Shor, “Algorithms for Quantum Computation: Discrete Log and Factoring,” Proceedings of the 35th Annual Symposium on the Foundations of Computer Science, IEEE, 1994, p. 124
- A. Barenco et al., “Elementary gates for quantum computation,” Physical Review A, March 22, 1995, https://arxiv.org/pdf/quant-ph/9503016

Classic Papers (Cont)

- L. K. Grover, "A fast quantum mechanical algorithm for database search," Proceedings, STOC 1996, Philadelphia PA, USA, pp. 212-219, https://arxiv.org/pdf/quant-ph/9605043
- G. Brassard et al., "Quantum Counting," 1998, https://arxiv.org/pdf/quant-ph/9805082
- G. Brassard et al., "Quantum Amplitude Amplification and Estimation," 2000, https://arxiv.org/pdf/quant-ph/0005055
- S. Lloyd, "Quantum Algorithm for Solving Linear Systems of Equations," American Physical Society, APS March Meeting 2010, March 15-19,2010

Scan This to Download These Slides

Raj Jain

http://rajjain.com

Related Modules

CSE567M: Computer Systems Analysis (Spring 2013),
https://www.youtube.com/playlist?list=PLjGG94etKypJEKjNAa1n_1X0bWWNyZcof

CSE473S: Introduction to Computer Networks (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypJWOSPMh8Azcgy5e_10TiDw

Wireless and Mobile Networking (Spring 2016),
https://www.youtube.com/playlist?list=PLjGG94etKypKeb0nzyN9tSs_HCd5c4wXF

CSE571S: Network Security (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypKvzfVtutHcPFJXumyyg93u

Video Podcasts of Prof. Raj Jain's Lectures,
https://www.youtube.com/channel/UCN4-5wzNP9-ruOzQMs-8NUw