Use Quantum Mechanics to Compute?

- Weird but useful properties of quantum mechanics:
  - Quantization: Only certain values or orbits are good
    » Remember orbitals from chemistry???
  - Superposition: Schizophrenic physical elements don’t quite know whether they are one thing or another
- All existing digital abstractions try to eliminate QM
  - Transistors/Gates designed with classical behavior
  - Binary abstraction: a “1” is a “1” and a “0” is a “0”
- Quantum Computing:
  Use of Quantization and Superposition to compute.
- Interesting results:
  - Shor’s algorithm: factors in polynomial time!
  - Grover’s algorithm: Finds items in unsorted database in time proportional to square-root of n.
  - Materials simulation: exponential classically, linear-time QM

Quantization: Use of “Spin”

- Particles like Protons have an intrinsic “Spin” when defined with respect to an external magnetic field
- Quantum effect gives “1” and “0”:
  - Either spin is “UP” or “DOWN” nothing between

Kane Proposal II
(First one didn’t quite work)

- Bits Represented by combination of proton/electron spin
- Operations performed by manipulating control gates
  - Complex sequences of pulses perform NMR-like operations
- Temperature < 1° Kelvin
Now add Superposition!

- The bit can be in a combination of “1” and “0”:
  - Written as: $\Psi = C_0|0> + C_1|1>$
  - The C’s are complex numbers!
  - Important Constraint: $|C_0|^2 + |C_1|^2 = 1$
- If measure bit to see what looks like,
  - With probability $|C_0|^2$ we will find $|0>$ (say “UP”)
  - With probability $|C_1|^2$ we will find $|1>$ (say “DOWN”)
- Is this a real effect? Options:
  - This is just statistical - given a large number of protons, a fraction of them ($|C_0|^2$) are “UP” and the rest are down.
  - This is a real effect, and the proton is really both things until you try to look at it
- Reality: second choice! There are experiments to prove it!

A register can have many values!

- Implications of superposition:
  - An $n$-bit register can have $2^n$ values simultaneously!
  - 3-bit example: $\Psi = C_{000}|000> + C_{001}|001> + C_{010}|010> + C_{011}|011> + C_{100}|100> + C_{101}|101> + C_{110}|110> + C_{111}|111>$
- Probabilities of measuring all bits are set by coefficients:
  - So, prob of getting $|000>$ is $|C_{000}|^2$, etc.
  - Suppose we measure only one bit (first):
    - We get “0” with probability: $P_0 = |C_{000}|^2 + |C_{001}|^2 + |C_{010}|^2 + |C_{011}|^2$
    - Result: $\Psi = (C_{000}|000> + C_{001}|001> + C_{010}|010> + C_{011}|011>)$
  - We get “1” with probability: $P_1 = |C_{100}|^2 + |C_{101}|^2 + |C_{110}|^2 + |C_{111}|^2$
    - Result: $\Psi = (C_{100}|100> + C_{101}|101> + C_{110}|110> + C_{111}|111>)$
- Problem: Don’t want environment to measure before ready!
  - Solution: Quantum Error Correction Codes!

Spooky action at a distance

- Consider the following simple 2-bit state: $\Psi = C_{00}|00> + C_{11}|11>$
  - Called an “EPR” pair for “Einstein, Podolsky, Rosen”
- Now, separate the two bits:

Model: Operations on coefficients + measurements

- Basic Computing Paradigm:
  - Input is a register with superposition of many values
    - Possibly all $2^n$ inputs equally probable!
  - Unitary transformations compute on coefficients
    - Must maintain probability property (sum of squares = 1)
  - Looks like doing computation on all $2^n$ inputs simultaneously!
- Output is one result attained by measurement

- If do this poorly, just like probabilistic computation:
  - If $2^n$ inputs equally probable, may be 2$^n$ outputs equally probable.
  - After measure, like picked random input to classical function.
  - All interesting results have some form of “fourier transform” computation being done in unitary transformation.

Light-Years?
Shor's Factoring Algorithm

- The Security of RSA Public-key cryptosystems depends on the difficulty of factoring a number $N=pq$ (product of two primes)
  - Classical computer: sub-exponential time factoring
  - Quantum computer: polynomial time factoring
- Shor's Factoring Algorithm (for a quantum computer)
  1) Choose random $x: 2 \leq x \leq N-1$.
  2) If $\gcd(x,N) \neq 1$, Bingo!
  3) Find smallest integer $r: x^r \equiv 1 \pmod{N}$
  4) If $r$ is odd, GOTO 1
  5) If $r$ is even, $a \equiv x^{r/2} \pmod{N} \Rightarrow (a-1)(a+1) = kN$
  6) If $a \equiv N-1 \pmod{N}$ GOTO 1
  7) ELSE $\gcd(a \pm 1, N)$ is a non trivial factor of $N$.

Finding $r$ with $x^r \equiv 1 \pmod{N}$

$$\sum_{k} |k\rangle |1\rangle \rightarrow \sum_{k} |k\rangle |x^k\rangle = \sum_{w=0}^{r-1} \sum_{y} |w + r\ y\rangle |x^w\rangle$$

Quantum Fourier Transform

- Finally: Perform measurement
  - Find out $r$ with high probability
  - Get $|y\rangle |a^w\rangle$ where $y$ is of form $k/r$ and $w$ is related

Quantum Computing Architectures

- Why study quantum computing?
  - Interesting, says something about physics
    » Failure to build $\Rightarrow$ quantum mechanics wrong?
  - Mathematical Exercise (perfectly good reason)
    - Hope that it will be practical someday
      » Shor's factoring, Grover's search, Design of Materials
      » Quantum Co-processor included in your Laptop?
- To be practical, will need to hand quantum computer design off to classical designers
  - Baring Adiabatic algorithms, will probably need 100s to 1000s (millions?) of working logical Qubits $\Rightarrow$ 1000s to millions of physical Qubits working together
  - Current chips: ~1 billion transistors!
- Large number of components is realm of architecture
  - What are optimized structures of quantum algorithms when they are mapped to a physical substrate?
  - Optimization not possible by hand
    » Abstraction of elements to design larger circuits
    » Lessons of last 30 years of VLSI design: USE CAD

Quantum Circuit Model

- Quantum Circuit model - graphical representation
  - Time Flows from left to right
  - Single Wires: persistent Qubits, Double Wires: classical bits
    - Qubit - coherent combination of 0 and 1: $\psi = \alpha |0\rangle + \beta |1\rangle$
  - Universal gate set: Sufficient to form all unitary transformations
- Example: Syndrome Measurement (for 3-bit code)
  - Measurement (meter symbol) produces classical bits
- Quantum CAD
  - Circuit expressed as netlist
  - Computer manipulated circuits and implementations
Quantum Error Correction

- Quantum State Fragile ⇒ encode all Qubits
  - Uses many resources: e.g. 3-level [[7,1,3]] code 343 physical Qubits/logical Qubit.
- Still need to handle operations (fault-tolerantly)
  - Some set of gates are simply "transversal."
  - Perform identical gate between each physical bit of logical encoding
  - Others (like T gate for [[7,1,3]] code) cannot be handled transversally
  - Can be performed fault-tolerantly by preparing appropriate ancilla
- Finally, need to perform periodical error correction
  - Correct after every?: Gate, Long distance movement, Long Idle Period
  - Correction reducing entropy ⇒ Consumes Ancilla bits
- Observation: ≥ 90% of QEC gates are used for ancilla production
  ≥ 70-85% of all gates are used for ancilla production

Outline
- Quantum Computing
- Ion Trap Quantum Computing
- Quantum Computer Aided Design
  - Area-Delay to Correct Result (ADCR) metric
  - Comparison of error correction codes
- Quantum Data Paths
  - QLA, CQLA, Qalypso
  - Ancilla factory and Teleportation Network Design
- Error Correction Optimization ("Recorrection")
- Shor's Factoring Circuit Layout and Design

MEMs-Based Ion Trap Devices
- Ion Traps: One of the more promising quantum computer implementation technologies
  - Built on Silicon
    - Can bootstrap the vast infrastructure that currently exists in the microchip industry
  - Seems to be on a "Moore's Law" like scaling curve
    - 12 bits exist, 30 promised soon, ...
    - Many researchers working on this problem
    - Some optimistic researchers speculate about room temperature
- Properties:
  - Has a long-distance Wire
    - So-called "ballistic movement"
  - Seems to have relatively long decoherence times
  - Seems to have relatively low error rates for:
    - Memory, Gates, Movement

Quantum Computing with Ion Traps
- Qubits are atomic ions (e.g. Be+)
  - State is stored in hyperfine levels
  - Ions suspended in channels between electrodes
- Quantum gates performed by lasers (either one or two bit ops)
  - Only at certain trap locations
  - Ions move between laser sites to perform gates
- Classical control
  - Gate (laser) ops
  - Movement (electrode) ops
    - Complex pulse sequences to cause Ions to migrate
    - Care must be taken to avoid disturbing state
- Demonstrations in the Lab
  - NIST, MIT, Michigan, many others
An Abstraction of Ion Traps

- Basic block abstraction: Simplify Layout

  ![Diagram of ion trap layout]

  - Evaluation of layout through simulation
    - Movement of ions can be done classically
    - Yields Computation Time and Probability of Success
  - Simple Error Model: Depolarizing Errors
    - Errors for every Gate Operation and Unit of Waiting
    - Ballistic Movement Error: Two error Models
      1. Every Hop/Turn has probability of error
      2. Only Accelerations cause error

Outline

- Quantum Computing
- Ion Trap Quantum Computing
- Quantum Computer Aided Design
  - Area-Delay to Correct Result (ADCR) metric
  - Comparison of error correction codes
- Quantum Data Paths
  - QLA, CQLA, Qalypso
  - Ancilla factory and Teleportation Network Design
- Error Correction Optimization ("Recorrection")
- Shor’s Factoring Circuit Layout and Design

Vision of Quantum Circuit Design
Important Measurement Metrics

- Traditional CAD Metrics:
  - Area
    - What is the total area of a circuit?
    - Measured in macroblocks (ultimately \( \mu m^2 \) or similar)
  - Latency (Latency\(_{\text{single}}\))
    - What is the total latency to compute circuit once
    - Measured in seconds (or \( \mu s \))
  - Probability of Success (P\(_{\text{success}}\))
    - Not common metric for classical circuits
    - Account for occurrence of errors and error correction

- Quantum Circuit Metric: ADCR
  - Area-Delay to Correct Result: Probabilistic Area-Delay metric
    - ADCR = Area \times E(\text{Latency}) = \frac{\text{Area} \times \text{Latency\(_{\text{single}}\)}}{P_{\text{success}}}
  - ADCR\(_{\text{optimal}}\): Best ADCR over all configurations
  - Optimization potential: Equipotential designs
    - Trade Area for lower latency
    - Trade lower probability of success for lower latency

How to evaluate a circuit?

- First, generate a physical instance of circuit
  - Encode the circuit in one or more QEC codes
  - Partition and layout circuit: Highly dependant of layout heuristics!
    - Create a physical layout and scheduling of bits
    - Yields area and communication cost

- Then, evaluate probability of success
  - Technique that works well for depolarizing errors: Monte Carlo
    - Possible error points: Operations, Idle Bits, Communications
  - Vectorized Monte Carlo: \( n \) experiments with one pass
    - Need to perform hybrid error analysis for larger circuits
      - Smaller modules evaluated via vector Monte Carlo
      - Teleportation infrastructure evaluated via fidelity of EPR bits
  - Finally - Compute ADCR for particular result
    - Repeat as necessary by varying parameters to generate ADCR\(_{\text{optimal}}\)

Example Place and Route Heuristic: Collapsed Dataflow

- Gate locations placed in dataflow order
  - Qubits flow left to right
  - Initial dataflow geometry folded and sorted
  - Channels routed to reflect dataflow edges
- Too many gate locations, collapse dataflow
  - Using scheduler feedback, identify latency critical edges
  - Merge critical node pairs
  - Reroute channels
- Dataflow mapping allows pipelining of computation!
Comparing Different QEC Codes

- Possible to perform a comparison between codes
  - Pick circuit/Run through CAD flow
  - Result depends on goodness of layout and scheduling heuristic

- Layout for CNOT gate (Compare with Cross, et. al)
  - Using Dataflow Heuristic
    - Validated with Donath’s wire-length estimator (classical CAD)
    - Fully account of movement
    - Local gate model

- Failure Probability results
  - Best: \([23,1,7]\) (Golay), \([25,1,5]\) (Bacon-Shor), \([7,1,3]\) (Steane)
  - Steane does particularly well with high movement errors
    - Simplicity particularly important in regime

- More info in Mark Whitney thesis
  - http://qarc.cs.berkeley.edu/publications

Outline

- Quantum Computing
- Ion Trap Quantum Computing
- Quantum Computer Aided Design
  - Area-Delay to Correct Result (ADCR) metric
  - Comparison of error correction codes
- Quantum Data Paths
  - QLA, CQLA, Qalypso
  - Ancilla factory and Teleportation Network Design
- Error Correction Optimization ("Recorrection")
- Shor’s Factoring Circuit Layout and Design

Quantum Logic Array (QLA)

- Basic Unit:
  - Two-Qubit cell (logical)
  - Storage, Compute, Correction
- Connect Units with Teleporters
  - Probably in mesh topology, but details never entirely clear from original papers
- First Serious (Large-scale) Organization (2005)
  - Tzvetan S. Metodi, Darshan Thaker, Andrew W. Cross, Frederic T. Chong, and Isaac L. Chuang

Details

- Why Regular Array?
  - Distribute Ancilla generation where it is needed
  - Single 2-Qubit storage cell quite large
    - Concatenated \([7,1,3]\) could have 343 or more physical Qubits
    - Size of single logical Qubit \(\Rightarrow\) makes sense to teleport between large logical blocks
  - Regularity easier to exploit for CAD tools!
    - Same reason we have ASICs with regular routing channels
- Assumptions:
  - Rate of ancilla consumption constant for every Qubit
  - Ratio of one Teleporter for every two Qubit gate is optimal
    - (Implicit) Error correction after every move or gate is optimal
  - Parallelism of quantum circuits can exploit computation on every Qubit in the system at same time
- Are these assumptions valid???
Running Circuit at “Speed of Data”

- Often, Ancilla qubits are independent of data
  - Preparation may be pulled offline
  - Very clear Area/Delay tradeoff:
    - Suggests Automatic Tradeoffs (CAD Tool)
- Ancilla qubits should be ready “just in time” to avoid ancilla decoherence from idleness

How much Ancilla Bandwidth Needed?

- 32-bit Quantum Carry-Lookahead Adder
  - Ancilla use very uneven (zero and T ancilla)
  - Performance is flat at high end of ancilla generation bandwidth
    - Can back off 10% in maximum performance and save orders of magnitude in ancilla generation area
- Many bits idle at any one time
  - Need only enough ancilla to maintain state for these bits
  - Many not need to frequently correct idle errors
- Conclusion: makes sense to compute ancilla requirements and share area devoted to ancilla generation

Ancilla Factory Design I

- “In-place” ancilla preparation
- Ancilla factory consists of many of these
  - Encoded ancilla prepared in many places, then moved to output port
  - Movement is costly!

Ancilla Factory Design II

- Pipelined ancilla preparation: break into stages
- Steady stream of encoded ancillae at output port
- Fully laid out and scheduled to get area and bandwidth estimates

- Recycle used correction qubits
The Qalypso Datapath Architecture

- Dense data region
  - Data qubits only
  - Local communication
- Shared Ancilla Factories
  - Distributed to data as needed
  - Fully multiplexed to all data
  - Output ports (←): close to data
  - Input ports (→): may be far from data (recycled state irrelevant)
- Regions connected by teleportation networks

Tiled Quantum Datapaths

Previous: QLA, LQLA
Previous: CQLA, CQLA+
Our Group: Qalypso

- Several Different Datapaths mappable by our CAD flow
  - Variations include hand-tuned Ancilla generators/factories
- Memory: storage for state that doesn’t move much
  - Less/different requirements for Ancilla
  - Original CQLA paper used different QEC encoding
- Automatic mapping must:
  - Partition circuit among compute and memory regions
  - Allocate Ancilla resources to match demand (at knee of curve)
  - Configure and insert teleportation network

Which Datapath is Best?

- Random Circuit Generation
  - f(Gate Count, Gate Types, Qubit Count, Splitting factor)
  - Splitting factor (r): measures connectivity of the circuit
    - Example: 0.5 splits Qubits in half, adds random gates between two halves, then recursively splits results
    - Closely related to Rent’s parameter
- Qalypso clear winner (for all r)
  - 4x lower latency than LQLA
  - 2x smaller area than CQLA+
- Why Qalypso does well:
  - Shared, matched ancilla generation
  - Automatic network sizing (not one Teleporter for every two Qubits)
  - Automatic Identification of Idle Qubits (memory)
- LQLA and CQLA+ perform close second
  - Original datapaths supplemented with better ancilla generators, automatic network sizing, and idle Qubit identification
  - Original QLA and CQLA do very poorly for large circuits

How to Design Teleportation Network

- What is the architecture of the network?
  - Including Topology, Router design, EPR Generators, etc..
- What are the details of EPR distribution?
- What are the practical aspects of routing?
  - When do we set up a channel?
  - What path does the channel take?
Basic Idea: Chained Teleportation

Adjacent T nodes linked for teleportation

• Positive Features
  - Regularity (can build classical network topologies)
  - T node linking not on critical path
  - Pre-purification part of link setup
  - Fidelity amplification of the line
  - Allows continuous stream of EPR correlations to be established for use when necessary

Pre-Purification

• Experiment: Transmit enough EPR pairs over network to meet required fidelity of channel
  - Measure total global traffic
  - Higher Fidelity local EPR pairs ⇒ less global EPR traffic
• Benefit: decreased congestion at T Nodes

Building a Mesh Interconnect

• Grid of T nodes, linked by G nodes
• Packet-switched network
  - Options: Dimension-Order or Adaptive Routing
  - Precomputed or on-demand start time for setup
• Each EPR qubit has associated classical message

Outline

• Quantum Computing
• Ion Trap Quantum Computing
• Quantum Computer Aided Design
  - Area-Delay to Correct Result (ADCR) metric
  - Comparison of error correction codes
• Quantum Data Paths
  - QLA, CQLA, Qalypso
  - Ancilla factory and Teleportation Network Design
• Error Correction Optimization ("Recorrection")
• Shor's Factoring Circuit Layout and Design
Reducing QEC Overhead

- Standard idea: correct after every gate, and long communication, and long idle time
  - This is the easiest for people to analyze
  - Urban Legend? Must do in order to keep circuit fault tolerant!
- This technique is suboptimal (at least in some domains)
  - Not every bit has same noise level!
- Different idea: identify critical Qubits
  - Try to identify paths that feed into noisiest output bits
  - Place correction along these paths to reduce maximum noise

Simple Error Propagation Model

- EDist model of error propagation:
  - Inputs start with EDist = 0
  - Each gate propagates max input EDist to outputs
  - Gates add 1 unit of EDist, correction resets EDist to 1
- Maximum EDist corresponds to Critical Path
  - Back track critical paths that add to Maximum EDist
  - Add correction to keep EDist below critical threshold
  - Example: Added correction to keep EDist_{MAX} ≤ 2

QEC Optimization

- Modified version of retiming algorithm: called “recorrection.”
  - Find minimal placement of correction operations that meets specified MAX(EDist) ≤ EDist_{MAX}
- Probably of success not always reduced for EDist_{MAX} > 1
  - But, operation count and area drastically reduced
- Use Actual Layouts and Fault Analysis
  - Optimization pre-layout, evaluated post-layout

Recorrection in presence of different QEC codes

- 1024-bit QRCA and QCLA adders

- 500 Gate Random Circuit (r=0.5)
  - Not all codes do equally well with Recorrection
    - Both [[23,1,7]] and [[7,1,3]] reasonable candidates
    - [[25,1,5]] doesn't seem to do as well
  - Cost of communication and Idle errors is clear here!
  - However - real optimization situation would vary EDist to find optimal point
Outline

- Quantum Computing
- Ion Trap Quantum Computing
- Quantum Computer Aided Design
  - Area-Delay to Correct Result (ADCR) metric
  - Comparison of error correction codes
- Quantum Data Paths
  - QLA, CQLA, Qalypso
  - Ancilla factory and Teleportation Network Design
- Error Correction Optimization ("Recorrection")
- Shor's Factoring Circuit Layout and Design

Comparison of 1024-bit adders

- 1024-bit Quantum Adder Architectures
  - Ripple-Carry (QRCA)
  - Carry-Lookahead (QCLA)
- Carry-Lookahead is better in all architectures
- QEC Optimization improves ADCR by order of magnitude in some circuit configurations

Area Breakdown for Adders

- Error Correction is not predominant use of area
  - Only 20-40% of area devoted to QEC ancilla
  - For Optimized Qalypso QCLA, 70% of operations for QEC ancilla generation, but only about 20% of area
- T-Ancilla generation is major component
  - Often overlooked
- Networking is significant portion of area when allowed to optimize for ADCR (30%)
  - CQLA and QLA variants didn’t really allow for much flexibility

Investigating 1024-bit Shor's

- Full Layout of all Elements
  - Use of 1024-bit Quantum Adders
  - Optimized error correction
  - Ancilla optimization and Custom Network Layout
- Statistics:
  - Unoptimized version: $1.35 \times 10^{15}$ operations
  - Optimized Version 1000X smaller
  - QFT is only 1% of total execution time
1024-bit Shor’s Continued

- Circuits too big to compute $P_{\text{success}}$
  - Working on this problem
- Fastest Circuit: $6 \times 10^8$ seconds ~ 19 years
  - Speedup by classically computing recursive squares?
- Smallest Circuit: $7659 \text{ mm}^2$
  - Compare to previous estimate of $0.9 \text{ m}^2 = 9 \times 10^5 \text{ mm}^2$

Conclusion

- Quantum Computer Architecture:
  - Considering details of Quantum Computer systems at larger scale (1000s or millions of components)
- Argued that CAD tools may have a place in Quantum Computing Research
  - Presented some details of a Full CAD flow (Partitioning, Layout, Simulation, Error Analysis)
  - New Evaluation Metric: $\text{ADCR} = \text{Area} \times E(\text{Latency})$
  - Full mapping and layout accounts for communication cost
- "Recorrection" Optimization for QEC
  - Simplistic model (EDist) to place correction blocks
  - Validation with full layout
  - Improves ADCR by factors of 10 or more
  - Improves latency and area significantly, can improve probability under some circumstances as well
- Full analysis of Adder architectures and 1024-bit Shor’s
  - Still too long (and too big), but smaller than previous estimates
  - Total circuit size still too big for our error analysis - but have hope that we can improve this