Design of a Modular Kilo-Qubit Scale Ion Trap Quantum Computer

Sam, Nick, Arvid, Jacob, Emile, Colin

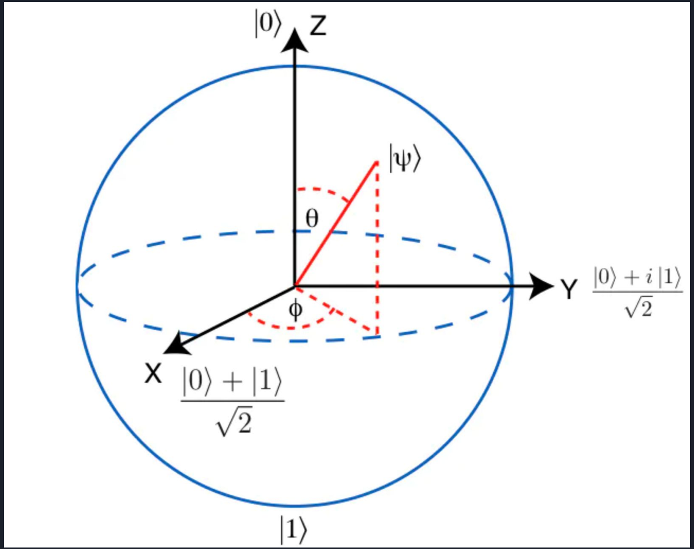
Introduction

Project / Objective

- ❑ Goal: Address the design concerns of a novel medium-scale QC architecture.
- ❑ Software: Quantum operation scheduling and qubit utilization optimizing
- ❑ Hardware: Generate a viable layout and identify any electro-physical concerns by investigating the current state of the art
- ❑ Limitations: Fabrication coordinated by Sandia N.L. with an industry partner at a later stage.

Project Motivation: Quantum Background

Quantum Computing Metrics

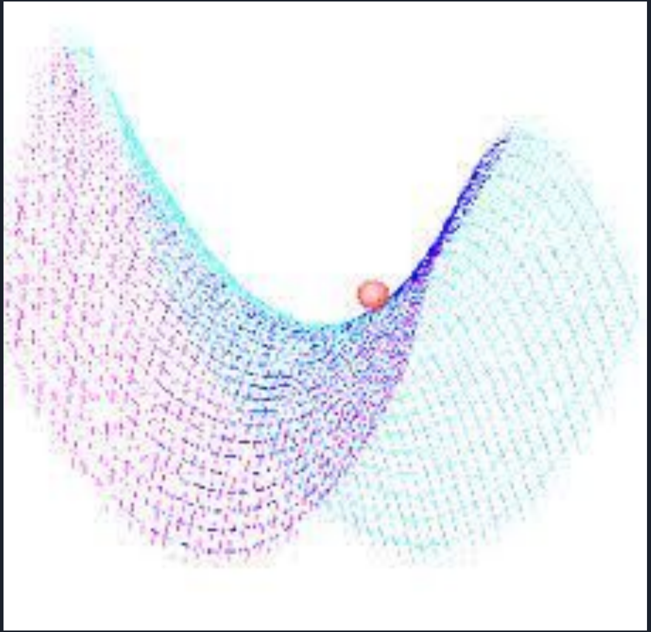


$|0\rangle$
Z
$|\psi\rangle$
θ
ϕ
Y
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

- ❑ Quantum Volume
  - ❑ Introduced by IBM in 2019
  - ❑ Maximum size of square quantum circuits that can be successfully implemented
- ❑ Coherence Time
  - ❑ Measures how long a qubit remains in a reliable superposition
- ❑ Noise / Quantum Fidelity / Error Correction
  - ❑ Quantum Fidelity - How closely you can expect a qubit to behave compared to its Quantum Coding (mathematical) counterpart
  - ❑ Error Correction codes are the main way to combat noise

Quantum Computing Implementations

- Superconducting Qubits
  - Fast Operations and Limited Scale
  - No-resistance current as Qubit
- Ion Trap
  - Slow, reliable operation, scalable
  - Yb^+ Ions suspended in an EMF “trap”
- New: Client’s Ion Trap Junction Design
  - Traps made of segmented linear RF electrodes
  - Linear trap-ends overlap at 90^o with 2x suspension height vertical spacing
  - RF electrode segment at overlap turn on/off to establish control over qubit in the region

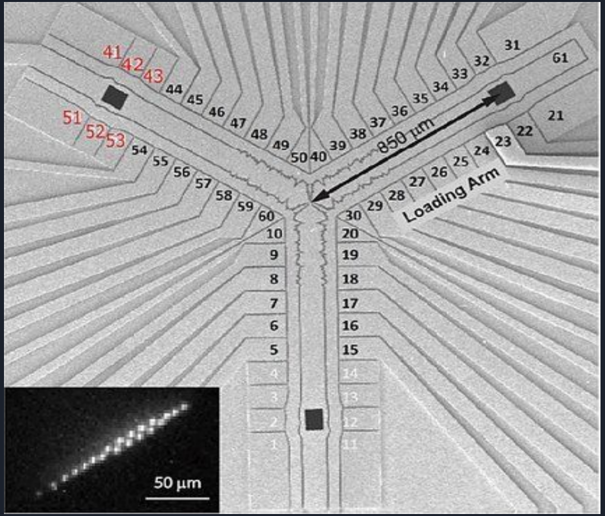


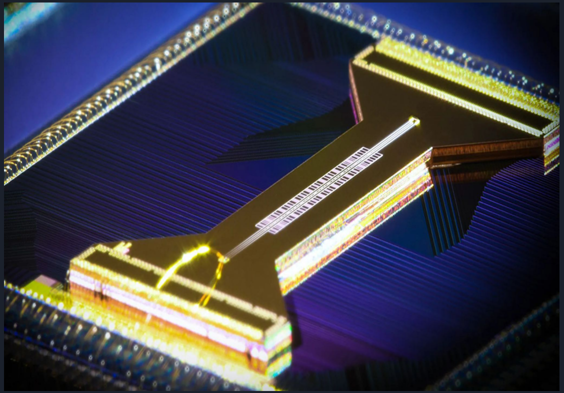
Design Architecture

Design Architecture: Background

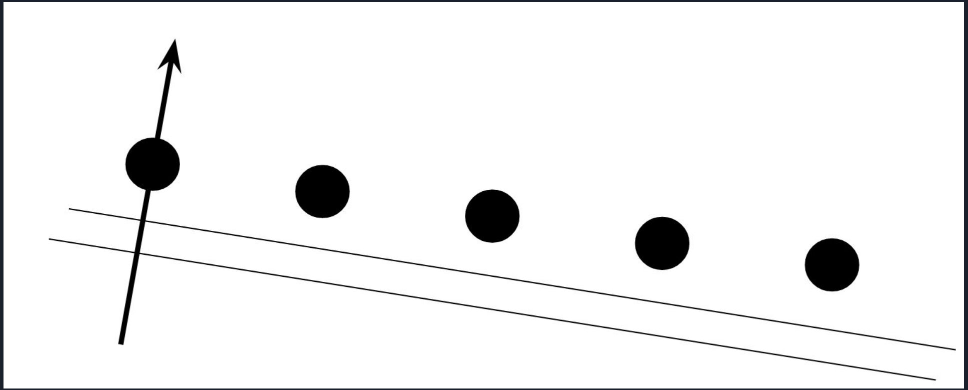
- Current leading (public) model:
  - Y intersection
  - All electrodes facing “up”
  - Limited surface area at intersection -> Qubit conflict

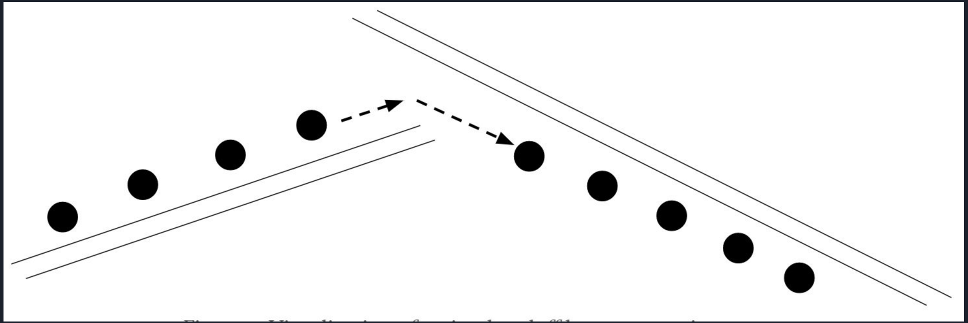
- How can we improve the leading design?
  - Increase stability of Qubits
  - Maximize efficiency of intersections
  - Modular -> Scalable





Design Architecture: How It Works

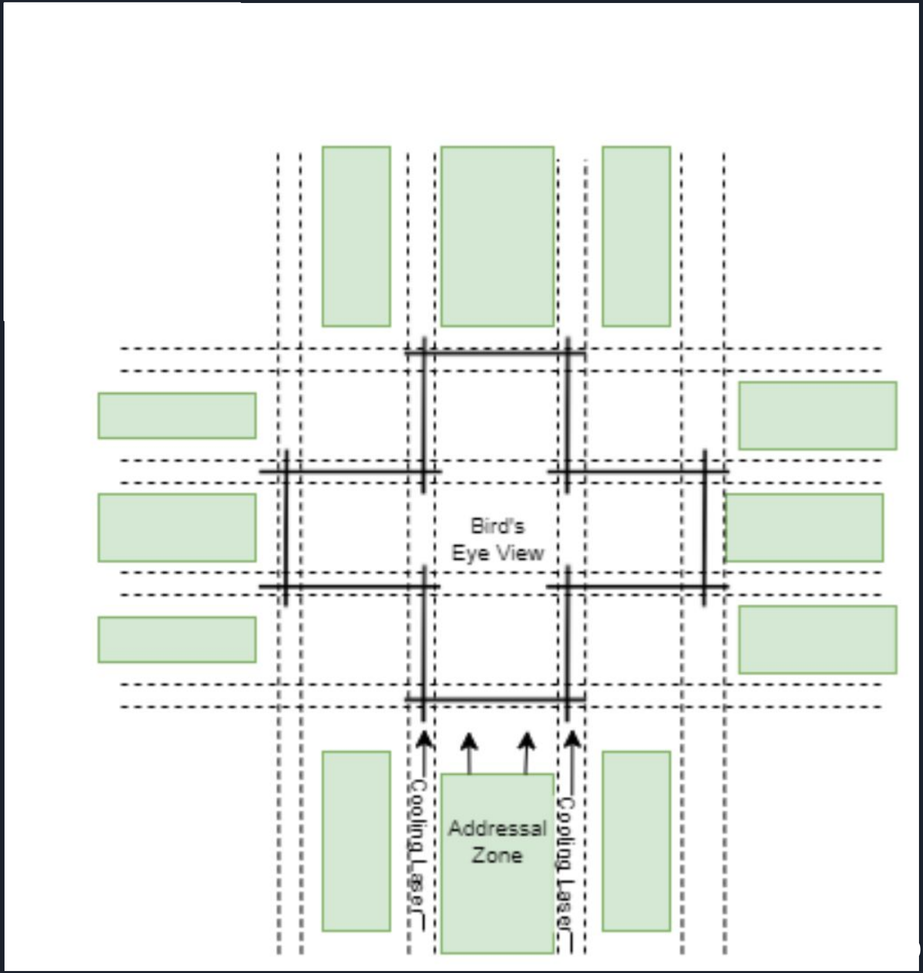




- ❑ Based on an ion trap **similar to** the Honeywell H1
- ❑ Uses multiple ion traps
  - ❑ Storage
  - ❑ Computing
  - ❑ ~10 ions per trap
- ❑ Ions will be handed off between traps
  - ❑ Hold the ion in place at the end
  - ❑ Turn off trap #1 while turning on trap #2

Design Architecture: Visual

- Primary Design Outlook
  - 12 traps in one node: ~120 qubits
  - ~2^{13} QV after error correction
  - Addressal Zone is over-simplified for visualization purposes
- Communication between nodes
  - Successfully interfacing and parallelizing multiple nodes could produce a kilo-qubit machine with capabilities on the order of ~2^{xxx} QV
- Note on simplification:
  - Addressal lasers may be located above/below traps rather than out-set from the node. Details unclear at this time



Hardware Team: Challenges, Solutions, Contributions and Accomplishments

Hardware Team Overview: Goals and Challenges

Goals: To Figure Out:

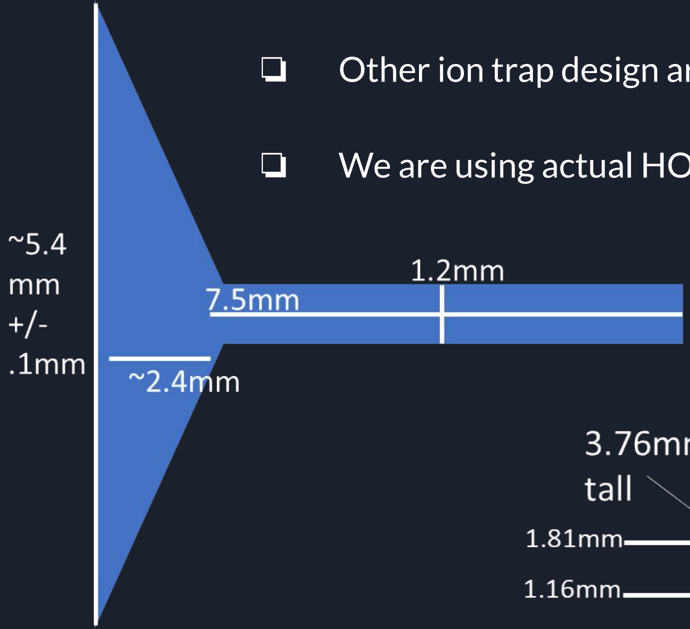
- ❑ What components will be needed? How many components? How much space will each take up?
- ❑ How large this computer will be, and is that within reason?
- ❑ Will any components “run into each other” - physically or otherwise?

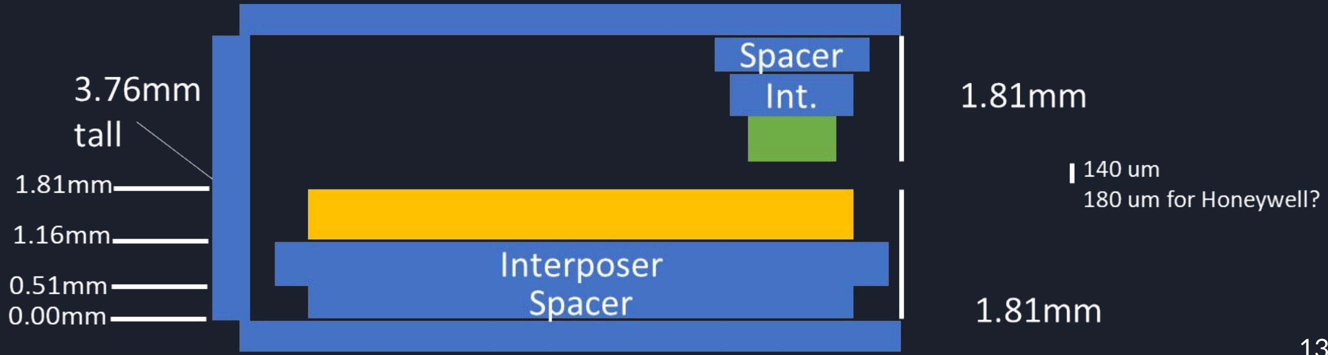
Challenges:

- ❑ ✔Traps won’t fit together on a reasonably sized chip / Overall design will be too large
- ❑ ✔Too many ancillary components
- ❑ ✔Ancillary components may overlap, physically or otherwise
- ❑ Won’t be able to use an “off-the-shelf” trap like the HOA

Our Design: Trap Measurements

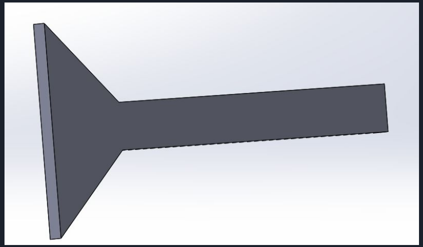
- The HOA trap is a “standard” trap -> Halfway through the semester, we learned the HOA trap would not work
- Other ion trap design are not public, so we didn’t have access to their measurements
- We are using actual HOA trap measurements as a proxy

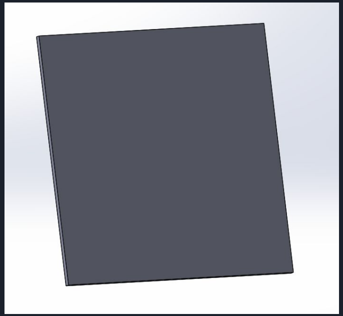




Our Design: SolidWorks Design

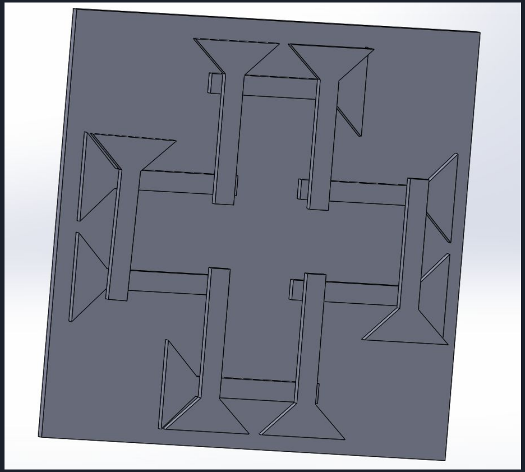
- To visualize our design with the previous measurements, we created a 3D models

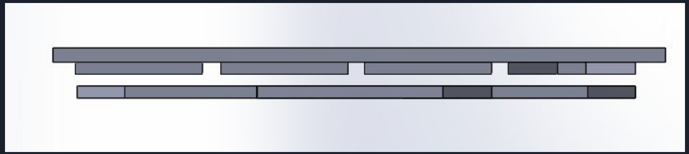




- Components:

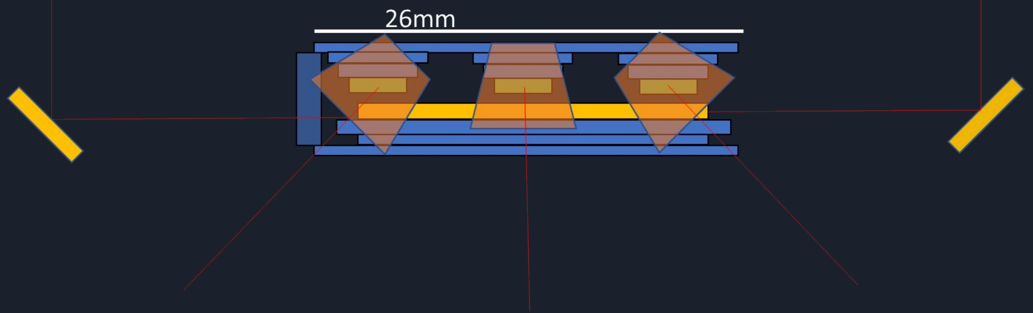
- Assembly:

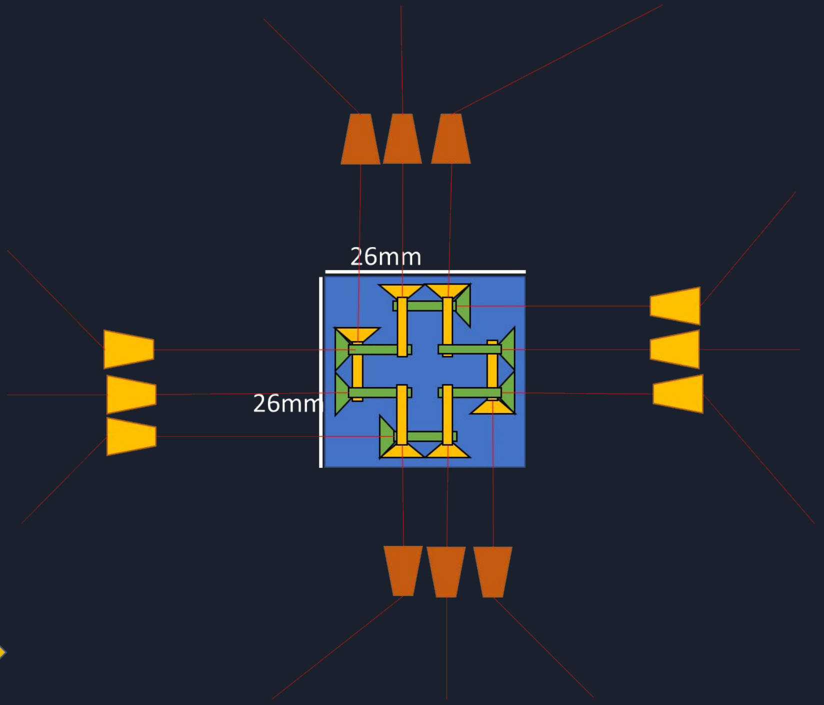




Our Design: Chip and Other Hardware

- ❑ Ancillary hardware can be located outside of QC's operating environment,
- ❑ Lasers can be reflected onto chip surfaces using mirrors
- ❑ **Exact number and size of ancillary components is unknown**





Our Design: Trap Circuitry

- Basic Ion Trap Circuit
  - Inner DC electrode
  - RF Electrode Pair
  - Outer DC electrode

- Issues with our Design
  - DC electrode routed to the four corners of the Chip
  - Need to space electrode to avoid interference
  - Minimize additional space needed to complete routing for all traps

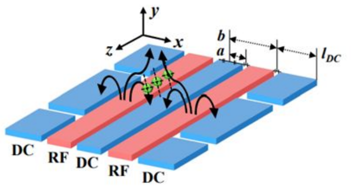


Figure 2. Schematic of a surface ion trap in a symmetric five-rail geometry. The red and blue rectangles indicate the RF and DC electrodes, respectively. The curved arrows denote the direction of the electric field when the RF voltage is positive. The dashed black lines indicate the principal axes of ion motions, which are tilted, for Doppler cooling with a single laser.

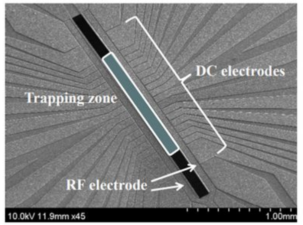
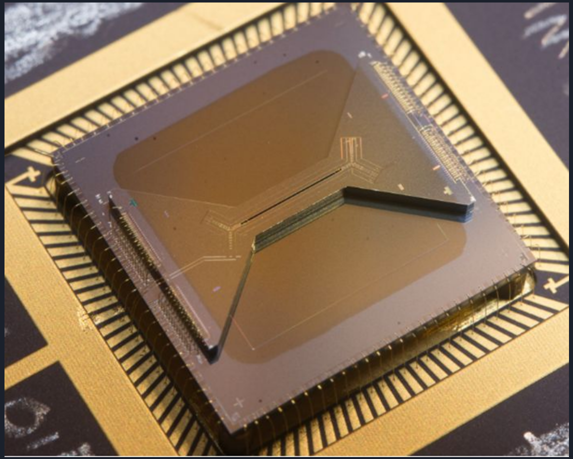
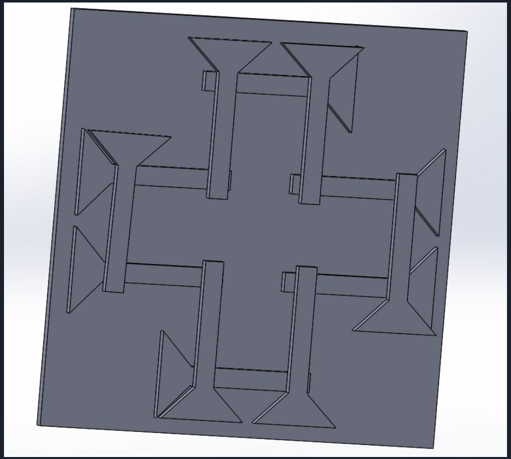


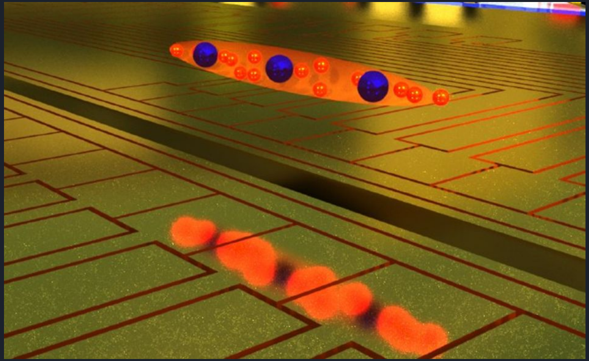
Figure 1. Scanning electron micrograph image of a surface ion trap fabricated by our group [11]. (The design layout of the ion trap uses the trap chip of Sandia National Laboratory [12] as the benchmark.)

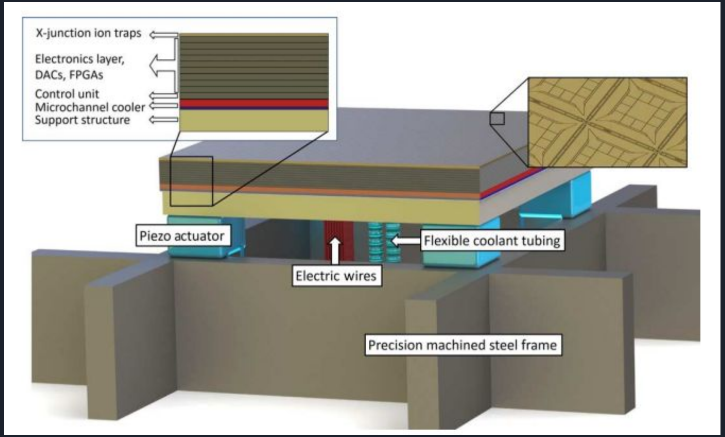




Our Design: Trap Circuitry

- ❏ Electrode Width Reduction
  - ❏ Reduce the size of the electrode before routing
  - ❏ Chosen size shouldn't affect the performance
- ❏ Connect electrode with same voltage
  - ❏ Reduce the amount of electrode to bring to the bonding
  - ❏ Complexe and Reduces flexibility on traps individual usage
- ❏ Sub-wafer electronics
  - ❏ Need more ancillary
  - ❏ Will also solve other issues





Software Team: Challenges, Solutions, Contributions and Accomplishments

Software Team Overview: The Digital Twin

- Modeling Computation on a New Hardware Paradigm

- ❑ Motive:
  - ❑ Address new scheduling, control, and communication concerns
- ❑ Goals:
  - ❑ Schedule qubit movements and interactions between traps
  - ❑ Automate and optimize logical to physical qubit mapping
  - ❑ Take advantage of scalability in the hardware paradigm
- ❑ Benefits:
  - ❑ Demonstrates the computational benefits of our hardware design
  - ❑ Begin developing software for our hardware design

Simplifying the Domain:

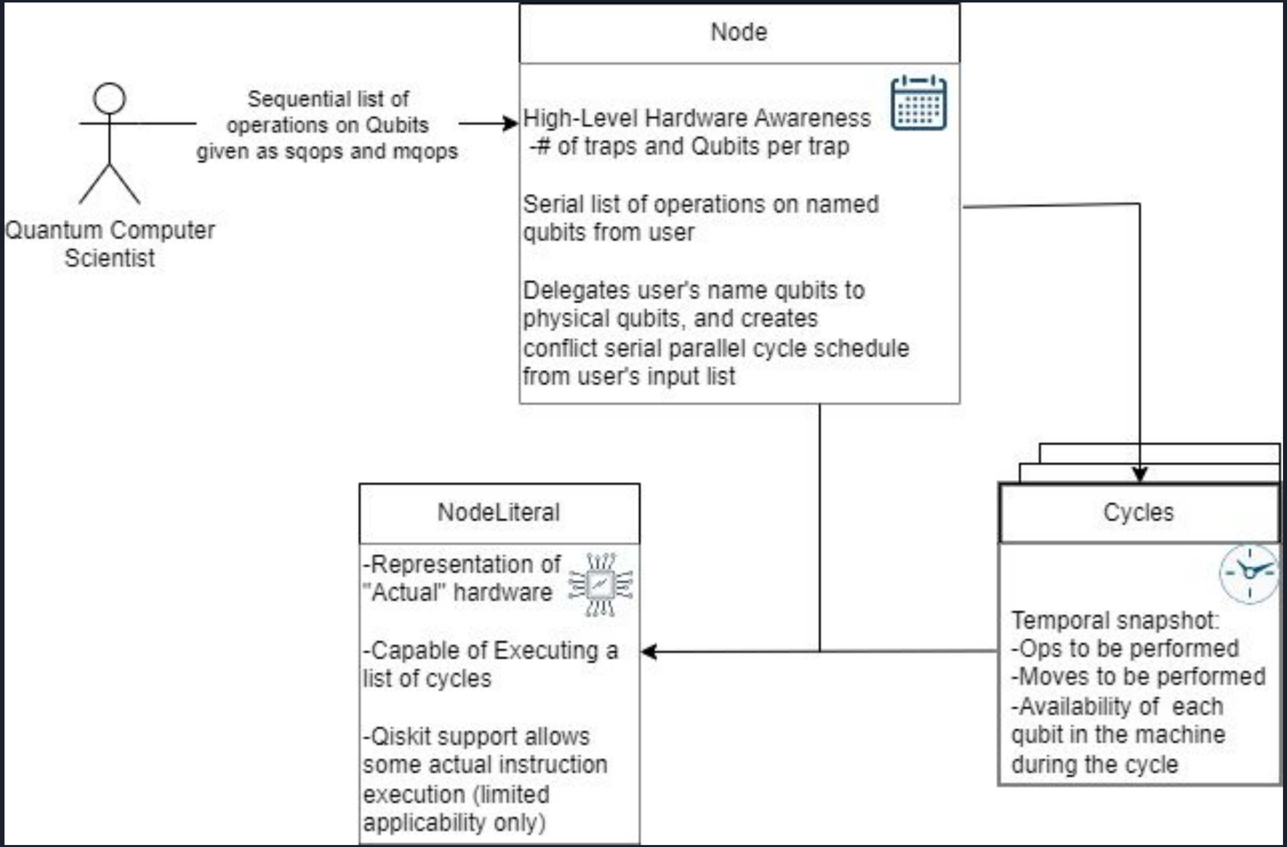
- Quantum Computer Design is a large domain:
  - Precision laser control
  - Timing
  - Electrical interferences
  - Ions cooling
  - Etc.

- We focussed on what’s new and manageable:
  - Perpendicular trap junctions and ion transfer
  - Parallel classical-scale scheduling at a node level

Software Accomplishments: Prototype Scheduler

- ❑ Node class
  - ❑ Given instructions, computes a dependency mapping between operations and resources
  - ❑ It constructs cycles to be used in the NodeLiteral class
- ❑ Cycle class
  - ❑ The cycle class simulates a cycle of a quantum computer cycle
- ❑ NodeLiteral class
  - ❑ The NodeLiteral class simulates the hardware level node, it directly interacts with qubits, there are no optimizations or controls here
- ❑ Scheduling
  - ❑ Gates execute as late as possible to minimize decoherence
  - ❑ Gates execute in a correct order

Software Accomplishments: Scheduling Prototype


Quantum Computer Scientist
Sequential list of operations on Qubits given as sqops and mqops
Node
High-Level Hardware Awareness
-# of traps and Qubits per trap
Serial list of operations on named qubits from user
Delegates user's name qubits to physical qubits, and creates conflict serial parallel cycle schedule from user's input list
NodeLiteral
-Representation of "Actual" hardware
-Capable of Executing a list of cycles
-Qiskit support allows some actual instruction execution (limited applicability only)
Cycles
Temporal snapshot:
-Ops to be performed
-Moves to be performed
-Availability of each qubit in the machine during the cycle

Software Team Individual Contributions

- ❑ Sam: Communication and Documentation Lead
- ❑ Arvid: Software Development Lead
- ❑ Jacob: Research Lead

Software Challenges and Solutions

- A Brand-New Device: What software contributions were both feasible and necessary with an 8-month timeline?
    - Sought advice from the senior design faculty on how to define a senior design project in the context of our client’s situation
    - Hosted weekly client meetings to collectively review overall project concept and our team’s role

- Defining a deliverable
    - Nearly “ground-floor” on the client’s new hardware paradigm definition
    - Knowledge debt and project goal ambiguity caused discussions about expectations to be slow

Software Challenges and Solutions Cont.

- Tempered Expectations
  - We prioritized outlining and addressing the most fundamental concerns of operation, and placed high importance on clear documentation for project continuation

- Scheduling Complexity:
  - Node-level qubit movement, concurrent quantum operations, and the do-as-late-as-possible policy are all more-or-less novel concepts
  - Except for the loose similarity to classical schedulers, pertinent work is limited, highly technical, and may not all be publically accessible
  - Addressing this wound up being the bulk of the technical work

Future Work and Conclusion

Hardware Conclusions / Future Work

- Primary Next Step: Review and verification of design with someone in the quantum computing space
  - Can this design be feasibly constructed?
  - Are we overlooking any elements (space, power, cooling, EM forces)?

- Physical implementation of our quantum computer node design
  - Can not use off the shelf components
  - Construction and testing must be done off-site (Sandia Labs)
  - Once physical computer is built, simple testing can begin

- Further research into cluster-scale connections must be done
  - How should nodes communicate with each other?
  - Can we get better returns to scale regarding ancillary hardware?

Software Conclusions / Future Work

- Outcome: Prototype Scheduler
- Next Steps:
  - Smarter qubit movement planning
  - Incorporating a stabilizer circuit (“memory”) paradigm
  - Addressing quantum-level noise concerns
  - Developing a multi-node operational ontology
- Further Testing
  - Classifying types of jobs based on resources and operations involved
  - Implement end-to-end tests with different classes of jobs as cases
- Client Feedback
  - Need to optimize our scheduling by cost of moving qubits
  - Keep software flexible
    - Modular trap count and Memory are two areas where high flexibility is needed

Overall Conclusion Summary

- Goal: Design hardware and software implementing a novel quantum computer architecture.
- Obstacles:
  - Unclear/changing project goals
  - Knowledge debt & novelty
- Outcomes:
  - Curation of a large project KB
  - Software: quantum job scheduler with as-late-as-possible and high proximity policies
  - Hardware: Global-level SolidWorks model. Notional models of circuitry implementation + identification of electro-physical concerns
- Future (for other project maintainers):
  - 3-6 months: previously mentioned next-steps for each team
  - 6-24 months: Paper, invention disclosure, fabrication assessment and provisional patent