Project Report
LSP-351

Architectural Analysis of Quantum Algorithms for NISQ Hardware: FY21 Quantum System Sciences Line-Supported Program

K.M. Obenland
A. Kurlej
S.B. Alterman

3 February 2022

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001.

This report is the result of studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology. This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering.

Massachusetts Institute of Technology
Lincoln Laboratory

Architectural Analysis of Quantum Algorithms for NISQ Hardware: FY21 Quantum System Sciences Line-Supported Program

K.M. Obenland
A. Kurlej
S.B. Alterman
Group 89

Project Report LSP-351

3 February 2022

This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001.

Lexington Massachusetts

This page intentionally left blank.

TABLE OF CONTENTS

		Page
	List of Illustrations	iii
	List of Tables	v
1.	INTRODUCTION	7
2.	QUANTUM ALGORITHM BENCHMARKS	11
3.	ARCHITECTURE SIMULATION OF NISQ CIRCUITS	15
4.	STUDY OF ION-TRAP QUANTUM ARCHITECTURES	19
5.	CONCLUSIONS AND FUTURE PLANS	25
	References	27

This page intentionally left blank.

LIST OF ILLUSTRATIONS

Figure No.		Page
1	Our approach for evaluating near-term architecture platforms based on quantum algorithmic benchmarks. This approach is executed "top-down" starting from a benchmark that is then compiled to the target gate set and then mapped onto specific hardware devices. Including each of these steps allows us to support a number of diverse benchmarks and helps in the understanding of the interdependencies at each step of the process.	7
2	Design of current superconducting and ion trap quantum computing platforms. The Google Sycamore platform is a 54-qubit superconducting qubit device with 2D nearest neighbor coupling [2]. IBM has also developed a superconducting platform, but has a limited coupling pattern to compensate for their use of fixed frequency qubits [3]. Finally, Honeywell has developed a machine that is an early demonstration of a Quantum Charge Coupled Device (QCCD) architecture [4].	8
3	An adaptive variational method for quantum chemistry. This figure was taken from reference [9]. The method utilizes a pool of operators derived from the Unitary Coupled Cluster Single Double (UCCSD) model of the molecule being simulated. The algorithm proceeds in phases until it reaches convergence by adaptively growing the ansatz with operators that have the largest gradient.	12
4	Circuit scaling for standard and adaptive UCCSD models. For the adaptive model, full simulation was performed for circuits of size up to 16 qubits. The circuit for full UCCSD is fixed by the problem, so the size can be determined by counting the number of gates in the static circuit.	13
5	Our framework for architectural analysis of near-term architectures. The framework consists of three main elements: generation of circuits from implementations of quantum algorithms, compilation of the circuits to target hardware platforms, and simulation of hardware platforms executing the benchmark circuits.	16
6	Functional organization of the architecture simulator. The simulator takes two main inputs: a QASM circuit of the gates to be executed and a configuration file that describes the layout of the architecture. The main output is a set of metrics calculated during the execution of the circuit.	16

LIST OF ILLUSTRATIONS
(Continued)

Figure No.		Page
7	Three candidate ion-trap architectures. The first is a fully connected linear chain, The second a segmented linear chain with 2-qubit interaction zones and 4-qubit storage zones, and the third contains multiple linear traps connected to all others via shuttling channels on either end.	20
8	Total runtime for each of the three candidate architectures. We take as benchmarks VQE circuits for molecules ranging in size from 4 to 80 orbitals, e.g., 4 to 80 qubits. In some cases, there are multiple molecules of the same size. For these we plot the average time.	22
9	Critical path analysis for VQE problems on each of the three architectures. The critical path is the sequence of operations in the parallel execution path that determine the total runtime. We study problems varying in size from 4 to 38 orbitals (qubits) and show the normalized runtime.	23

LIST OF TABLES

Table No.		Page
1	Quantum Benchmark Programs (Circuits)	11
2	Architecture Assumptions Used for Each of the Three Architectures	21

This page intentionally left blank.

1. INTRODUCTION

Emerging quantum computing architectures are at the cusp of showing relevance for real-world problems [1]. However, current quantum computing hardware platforms are not designed with a focus on the problems to be run on them. Rather they are designed from the "bottom-up," where a quantum computing architecture typically consists of a homogeneous replication of a set of low-level devices, i.e., qubit designs. Additionally, demonstrations to-date have mostly been limited to "toy" problems or problems that have little practical value.

The goal of our project is to investigate the architectural trade-offs that naturally occur in the design of emerging quantum computing hardware (see Figure 1). We study these architectures using quantum benchmarks that are implementations of quantum algorithms that have practical impact. At the core of our project is a quantum architecture simulation tool that will allow us to assess the impact that architecture design choices have on the resources and fidelity of the designs. The current target of our work is on systems that are envisioned in the next five years (systems requiring 100-200 qubits).

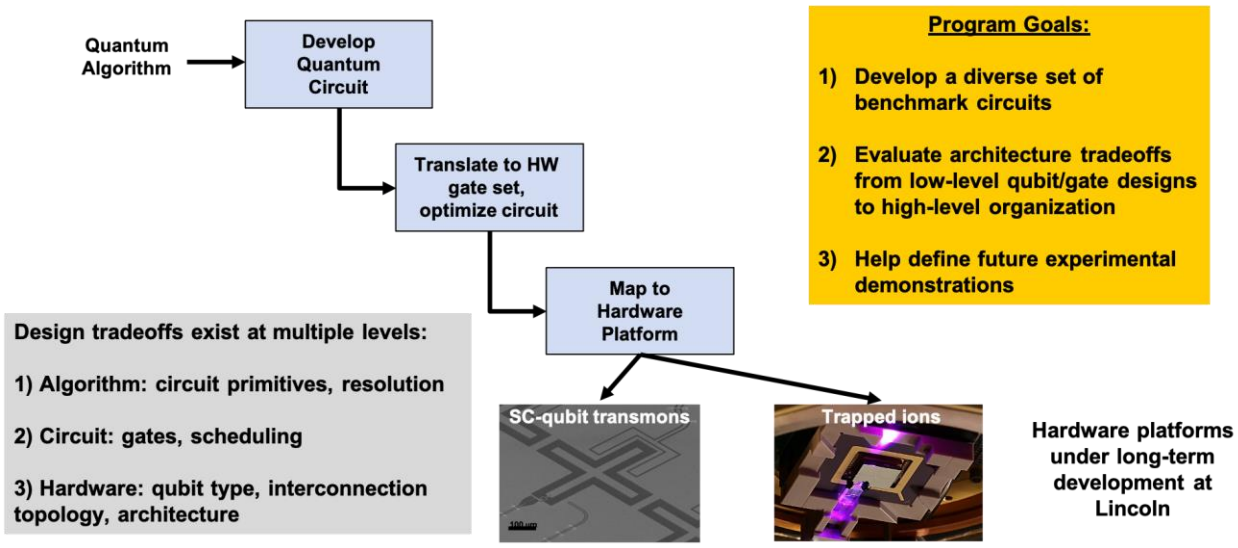


Figure 1. Our approach for evaluating near-term architecture platforms based on quantum algorithmic benchmarks. This approach is executed "top-down" starting from a benchmark that is then compiled to the target gate set and then mapped onto specific hardware devices. Including each of these steps allows us to support a number of diverse benchmarks and helps in the understanding of the interdependencies at each step of the process.

There are many examples of quantum platforms that have been designed using a "bottom-down" approach. In Figure 2, we show three currently available quantum computing platforms: two superconducting devices, and one based on trapped-ion qubits. In each of these, the architecture is motivated by the qubit technology. For Google's Sycamore device [2], the architecture is a simple grid of coupled transmon superconducting qubits. IBM's device [3] uses a limited coupling scheme, which helps reduce frequency collisions due to the use of fixed-frequency qubits. Finally, Honeywell's ion device [4], is based on a scalable QCCD concept [5], however the current demonstration consists of a single linear trap.

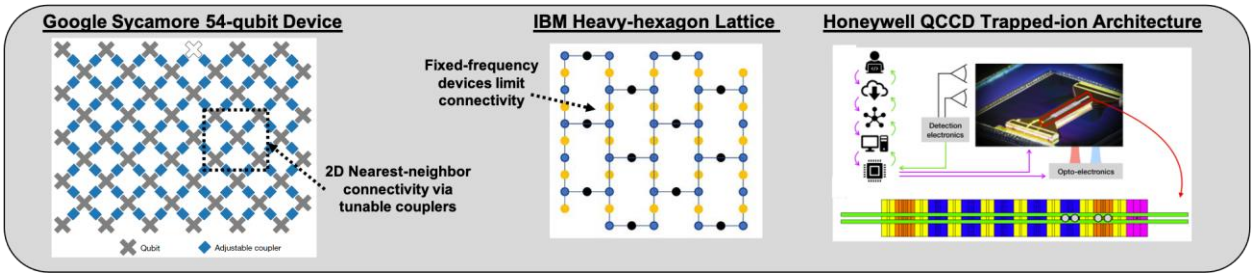


Figure 2. Design of current superconducting and ion trap quantum computing platforms. The Google Sycamore platform is a 54-qubit superconducting qubit device with 2D nearest neighbor coupling [2]. IBM has also developed a superconducting platform, but has a limited coupling pattern to compensate for their use of fixed frequency qubits [3]. Finally, Honeywell has developed a machine that is an early demonstration of a Quantum Charge Coupled Device (QCCD) architecture [4].

Our approach to the analysis and development of quantum architecture platforms differs from the current "bottom-up" design approach in several key ways. Our "top-down" approach is illustrated in Figure 1. This approach is driven by a set of algorithmic benchmarks. These benchmarks are examples of applications that are believed to have utility in the near term. We are developing both compilation tools, which allows us to target a broad set of different algorithms, as well as architecture simulation tools that enables the evaluation of different architectures for performance and fidelity.

The aim of our project is to understand the key architectural tradeoffs for quantum algorithms targeted to near-term quantum processors. The focus in the past year has been on trapped-ion platforms—a technology that has been in development at Lincoln Laboratory for many years [6].

Our project consists of three main activities:

1. Implementation, analysis, and optimization of quantum algorithms to produce quantum benchmark programs

2. Evaluating the architecture tradeoffs by developing a NISQ quantum architecture simulator and compilation support tools

3. Development of phenomenological device error models and using these models in quantum mechanical simulations to understand the impact of error on the performance of the quantum benchmarks

Our project is a multi-year effort that was initiated in FY20. In the following sections, we summarize the work from FY21. In the next section we describe the quantum algorithms used as the basis of our program. In Section 3, we describe the NISQ architecture simulator and show results from our current studies in Section 4. In the final section, we summarize and discuss future work.

This page intentionally left blank.

2. QUANTUM ALGORITHM BENCHMARKS

We are using and developing a number of quantum algorithm benchmark circuits in our project to use as input to our quantum architecture simulator. A list of these benchmarks is shown in Table 1. These benchmarks represent a wide range of different application areas. Each specific benchmark is parameterized for size and in some cases across different problem instances. Having parameterized circuit instances allows us to study architecture implementations at varying scales.

Table 1

Quantum Benchmark Programs (Circuits)

Each benchmark represents a family of circuits for different instances of the target algorithm varying in size and complexity.

Application area	Algorithm	Description	Circuit Configuration
Quantum simulation	**Quantum signal processing (QSP)**	**Hamiltonian simulation of a Heisenberg spin-chain.**	**Parameterized sized problems. Small problems (n=4) require ~100 gates per iterate. Large problems (n=50) require thousands of gates per iterate.**
Quantum chemistry	**Variational Quantum Eigensolver (VQE)**	**Ground state estimation of molecular Hamiltonians using the 2nd quantization model and STO-3G basis. Generated with Google's OpenFermion tool** [7]**.**	**Circuit size varies with molecule choice. Can run entire VQE process for small problems (<10 orbitals/ 20Q) to generate circuits with ~100s of gates or can generate circuits of arbitrary size, guided by the scaling of UCCSD circuits.**
Optimization	**Quantum Approximate Opt. Alg. (QAOA)**	**Variational optimization of the Max-cut problem for 3-regular graphs, generated through Google's CIRQ framework** [8]**.**	**Circuit size varies with number of nodes in graph, and choice of number of levels *p*.**
Supremacy	**Random quantum circuits**	**Cross Entropy benchmark (XEB) based on Google's quantum supremacy demonstration** [2]**.**	**Circuit can be generated to arbitrary width and depth.**

One main focus of our benchmarking in the past year has been on the Variational-Quantum Eigensolver (VQE) for quantum chemistry. For this benchmark, we have used an adaptive method for building the required ansatz circuit that is based on methods from reference [9]. A schematic, taken from the paper, of the operation of this algorithm is shown in Figure 3. This algorithm attempts to reduce the size of the variational ansatz circuit by adding operators from the model Hamiltonian adaptively. Methods like the Unitary Coupled Cluster Single Double (UCCSD) model use an ansatz with a complete set of operators. In the adaptive approach, we add the operators one at a time based on how much they reduce the cost of the solution.

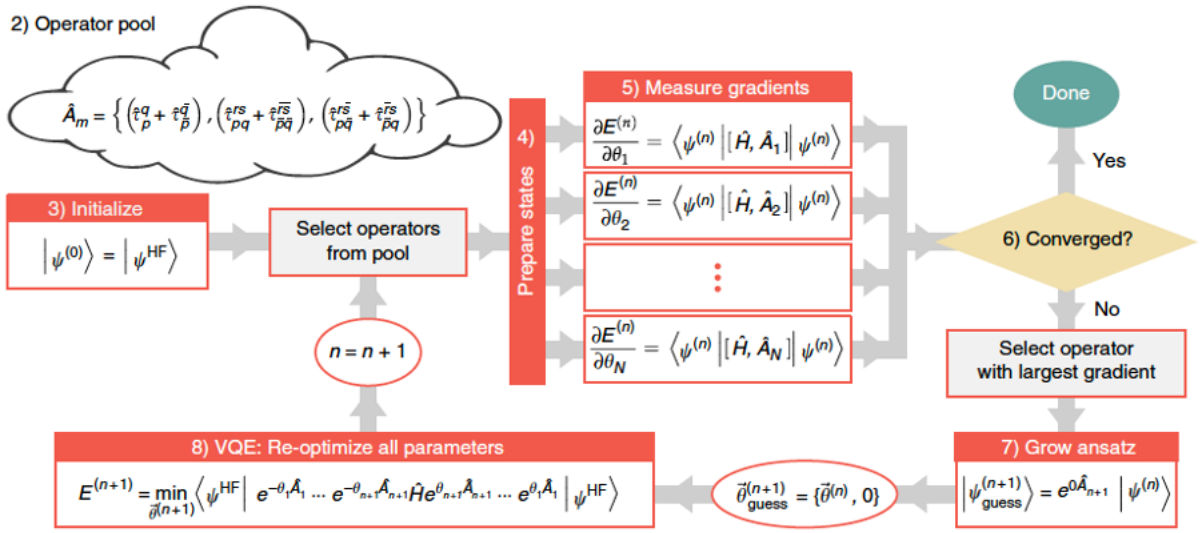
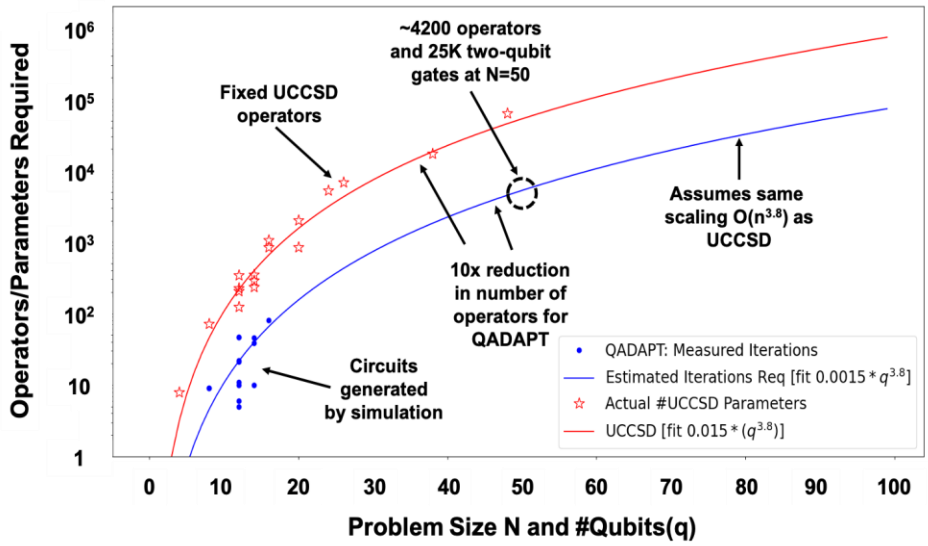


Figure 3. An adaptive variational method for quantum chemistry. This figure was taken from reference [9]. The method utilizes a pool of operators derived from the Unitary Coupled Cluster Single Double (UCCSD) model of the molecule being simulated. The algorithm proceeds in phases until it reaches convergence by adaptively growing the ansatz with operators that have the largest gradient.

In an attempt to understand the performance impact of the adaptive VQE approach over the standard UCCSD model [10] [11], we have studied the performance of various molecules over a range of sizes. The results of this study are shown in Figure 4. The full UCCSD method uses an ansatz circuit whose size is fixed for a particular problem. Therefore, we can generate these types of circuits for different problems and measure the number of operators required for each. (Note: the application of an operator in a circuit is implemented by a standard sequence of gates and therefore the number of gates in the circuit is a direct function of the number of operators used.) For the adaptive UCCSD method, the size of the circuit can only be determined by running the circuit and to do this we require a full quantum mechanical simulation of its operation. For the data shown in Figure 4, we perform this full simulation with circuits having up to 16

qubits and then fit the data to a polynomial of the same order as the UCCSD model. This analysis shows about a 10x reduction in the number of gates used by the adaptive method over the full method. Our estimate also indicates that a quantum chemistry problem with N=50 orbitals would require approximately 25,000 two qubit gates. This would be a challenge for today's quantum processors where the two-qubit error rate is close to 10^{-3}.

In the past, we have also studied constructions of the ansatz circuits whose constructions were motivated not by the problem but by the machine they are implemented on. These constructions, like the one described in [12], are optimized for the operations available on the hardware. We have found that these hardware optimized ansatz constructions work well for small problems, but as the problem scales to larger size that the ansatz circuits based on the underlying chemical problem perform better.



Molecules for QADAPT Circuits

Molecule(s)	N and q
H4	8
H6, LiH, CH, BH, NH	12
CH2, BH2, NH2	14
H8	16

Figure 4. Circuit scaling for standard and adaptive UCCSD models. For the adaptive model, full simulation was performed for circuits of size up to 16 qubits. The circuit for full UCCSD is fixed by the problem, so the size can be determined by counting the number of gates in the static circuit.

This page intentionally left blank.

3. ARCHITECTURE SIMULATION OF NISQ CIRCUITS

Our framework for developing and analyzing architecture for near-term architectures is illustrated in Figure 5, and includes the following three elements:

1. **Circuit generation**. Generation of circuits as described in the previous section. As part of this process, we make use of quantum circuit simulation tools to aid in the construction and validation of the circuits.

2. **Circuit compilation**. This is the process of mapping a technology agnostic circuit to the set of gates supported by the target architecture. This includes translating canonical gates, like CNOT, to gates supported by the hardware, e.g., Mølmer-Sørensen gates. It also includes the process of adding hardware specific operations to move qubit states as required to support multi-qubit operations. For example, on a superconducting qubit machine, if we need to perform a CNOT gate on two qubits that are not directly coupled, then the compiler will add SWAP gates to move one (or both) of them to positions where they are directly coupled. On a trapped-ion architecture, chain reordering and split/merge operations are used to move ion qubits as required to support multi-qubit gates.

3. **Architecture simulation.** The final step in the process is a simulation of a target machine that has been configured with specific resources and interconnection topology. This simulation tracks the dynamic execution of the program, e.g., the ordering and timing of gate operations, and the movement of qubit states throughout the machine. Using this simulation tool, we can measure the time required to execute circuits for different architectures and for varying configurations of the same architecture. We can also track utilization of resources on the machine to determine bottlenecks.

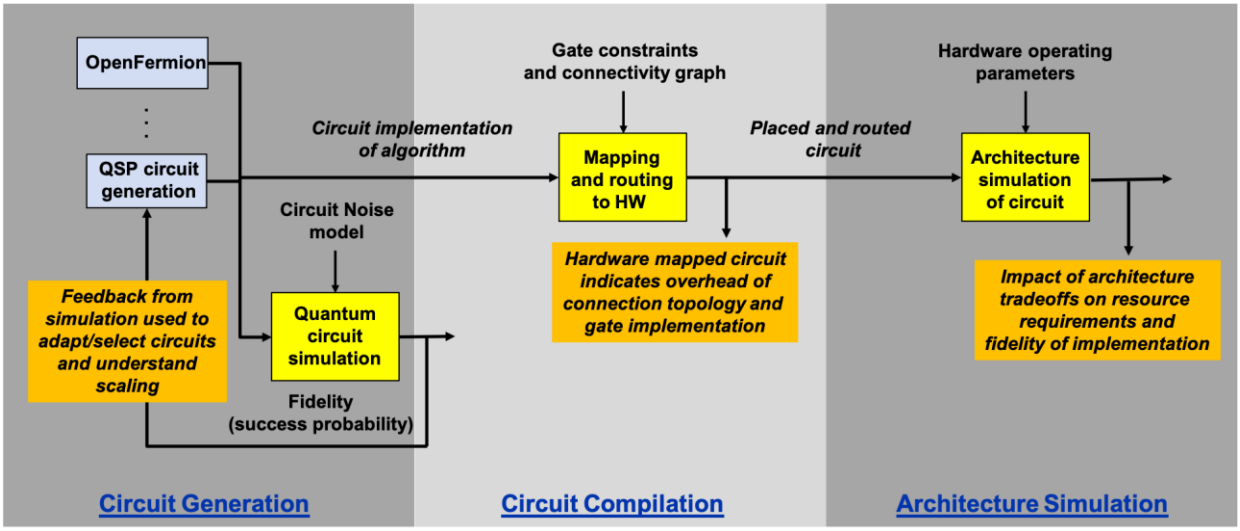


Figure 5. Our framework for architectural analysis of near-term architectures. The framework consists of three main elements: generation of circuits from implementations of quantum algorithms, compilation of the circuits to target hardware platforms, and simulation of hardware platforms executing the benchmark circuits.

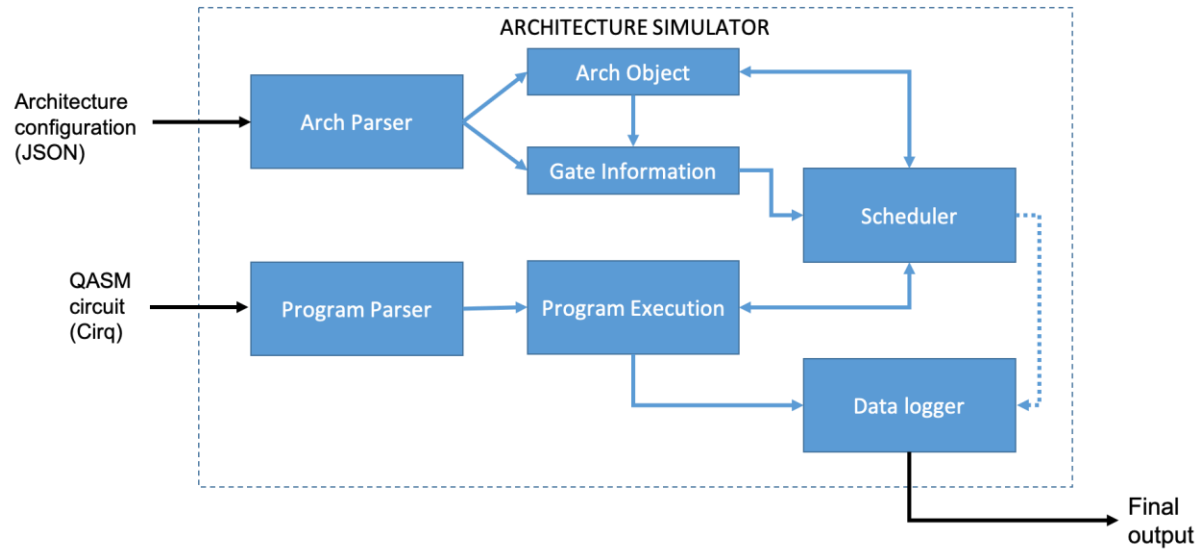


Figure 6. Functional organization of the architecture simulator. The simulator takes two main inputs: a QASM circuit of the gates to be executed and a configuration file that describes the layout of the architecture. The main output is a set of metrics calculated during the execution of the circuit.

The functional organization of the architecture simulator is shown in Figure 6. The simulator takes two inputs: the quantum program (in the form of a QASM program) and an architecture configuration description file (in JSON format). An architecture description allows us to specify a wide range of different architectures and configurations in a convenient and flexible way, allowing us to easily study a wide range of tradeoffs for different architectures. The core of the architecture simulator is the scheduler function that schedules the use and timing of resources of the machine. Resources include objects like: qubits, ion traps, interconnection networks, and coupling resonators. The simulator determines which resources are required by each gate, allocates them when required, and executes the gate for the duration specified in the architecture configuration. Metrics calculated by the simulator as it runs are then logged and output at the completion of a run.

This page intentionally left blank.

4. STUDY OF ION-TRAP QUANTUM ARCHITECTURES

In this section, we use our framework for architectural analysis to study tradeoffs in the design of trapped ion quantum platforms. We selected three different trap architectures as illustrated in Figure 7 and described below:

1. **Single fully connected trap**. This design consists of a single linear chain of ions. Operations can be performed between arbitrary qubits in the chain. Gates are expected to be noisier and slower in this architecture than the others due to the large number of ions trapped in a single crystal. This is motivated by architecture such as the one described in [13].

2. **Multi-zone linear trap**. In this architecture, ions are trapped in a linear chain, but the traps are segmented into isolated zones (i.e., crystals). We assume that each zone is sufficiently separated from the others to support parallel operations without crosstalk. The trap consists of an even number of two-qubit interaction zones that are interleaved between an odd number of four-qubit storage zones. All single-qubit and two-qubit gates are performed in the interaction zones. The storage zones support chain reconfiguration operations like: reordering, split, and join.

3. **Isolated multi-trap ladder architecture**. This design consists of a number of isolated linear traps that are interconnected on each side via shuttling channels. This architecture has greater connectivity than #2, however, ions must be shuttled longer distances and through junctions. This architecture is a simplified example of the QCCD architecture [5].

In each of these architectures, we assume the need for cooling operations and allocate ions throughout the chains to enable this. Additionally, in our studies we tailor specific designs of each architecture as required by the number of qubits in a problem growing, or shrinking, the size of the trap(s) to the minimum size required by the problem and the architecture.

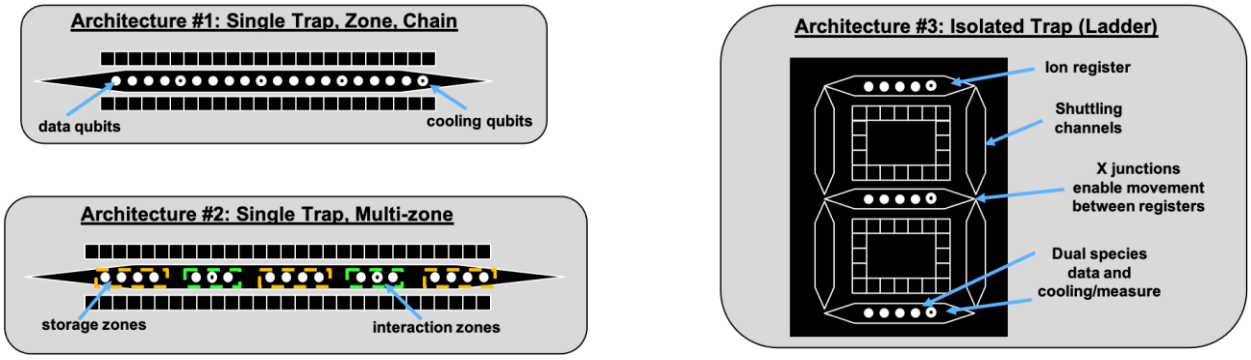


Figure 7. Three candidate ion-trap architectures. The first is a fully connected linear chain, The second a segmented linear chain with 2-qubit interaction zones and 4-qubit storage zones, and the third contains multiple linear traps connected to all others via shuttling channels on either end.

In Table 2, we detail the assumptions made for each architecture. These assumptions include the configuration of the different zones as well as timings of the gates used for logic and reconfiguration operations. Most notable in these assumptions are the timing of multi-qubit gates and for cooling operations. We assume faster two-qubit gates for short chains (100 μs) and slower gates for chains of length 5 or greater (400 μs). We also assume that a cooling gate of duration 650 μs is applied before each two-qubit operation, and a cooling gate of duration 1400 μs is applied before a move operation.

Table 2

Architecture Assumptions Used for Each of the Three Architectures

The majority of the physical gate timing assumptions were derived from reference [4].

	Architecture #1	Architecture #2	Architecture #3
Max Qubits	"Interaction Zone" = up to 80Q/Zone *Single Trap composed of one "Interaction Zone"*	"Storage Zone" = up to 4Q/Zone "Interaction Zone" = up to 2Q/Zone *Single Trap composed of multiple "Zones"*	"Interaction Zone" = up to 4Q/Zone *Multiple Traps, where each Trap is composed of one "Interaction Zone"*
1QB Gate Time [us]	10		
1QB Gate Parallelism	*Each Zone capable of executing an arbitrary number of single qubit gates simultaneously*		
Multi-QB Gate Time [us]	100 *for 2QB chain* 200 *for 3-5QB chain* 400 *for 5+QB chain*		
Multi-QB Gate Pre-Cool Time [us]	650 [4]		
Multi-QB Gate Parallelism	*Each Zone can execute only a single multi-QB gate*		
Split / Join Time [us]	*Not Applicable*	130 [4]	
Chain Rotate Time [us]	*Not Applicable*	200 [4]	
Move Pre-Cool Time [us]	1400 [4]		
Split/Join/Chain Rotate Parallelism	*Each trap can perform an arbitrary number of ops simultaneously per independent chain*		
Inter/Intra-Trap Move Speed [um/us]	*Not Applicable*	1.1 [4]	
Inter-Trap Vertical Spacing [um]	*Not Applicable*		840 [4]
Zone Size [um]	*Not Applicable*	440 [4]	

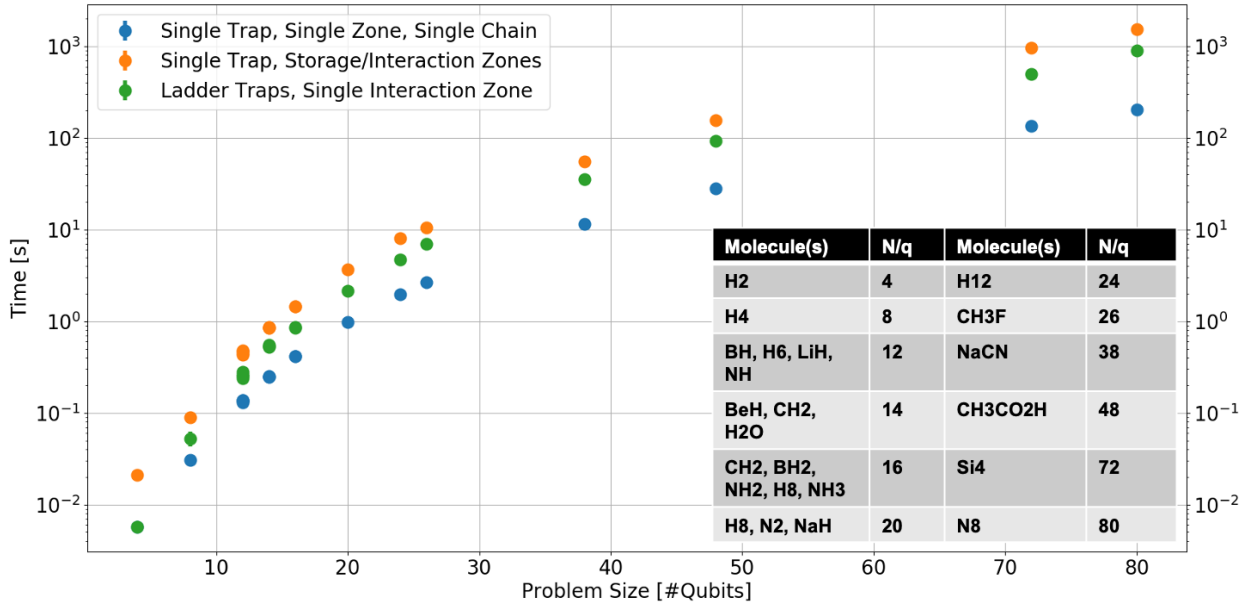
In our first study, we look at the total runtime for each architecture running adaptive VQE circuits for molecules ranging in size from 4 to 80 orbitals. (Each orbital is encoded in a single qubit bit, so these benchmarks range in size from 4 to 80 qubits.) The results from this study are shown in Figure 8. The figure shows the total runtime for each of the circuits. For all cases, the runtime is lowest for the single linear trap followed by the ladder architecture (#3) and finally the single trap multi-zone architecture (#2). This seems to indicate that the extra overhead required to split, move, and join ions in these segmented architectures has a significant impact on the total runtime.

To better understand the factors that are driving the performance of the benchmarks on the three architectures, we have also performed a critical path analysis of the execution time. The critical path, in the parallel execution of a program, is the sequence of instructions that determine the total runtime. By observing the mixture of instructions that constitute the critical path, we can better understand which operations dominate the overhead of each architecture. The results of this analysis are shown in Figure 9. In this study, we determine the critical path for benchmarks ranging in size from N=4 to N=38. We plot a histogram of normalized operation time for each instruction type.

We can draw a few interesting conclusions from the data shown in Figure 9. First of all, the cooling time is a dominate factor for all architectures. Heating of ions is a critical factor affecting the fidelity of two-qubit gates. We used the assumptions from the experiment described in reference [4] to determine when and for how long to apply cooling pulses in our simulation. The experiment described in [4] is much

smaller than our benchmarks and it is likely that they are being overly conservative in their use of cooling. Our study indicates that one needs to take a closer look at when cooling needs to be applied, otherwise, this will drive the execution time.

Another interesting conclusion shown by the data in Figure 9 is that other than cooling, the dominate operation for each architecture is different. For the single trap architecture (#1), the two-qubit MS gate is the dominate operation. This is because this architecture makes little use of chain reorganization operations like: split/join or chain reorder. For the single chain multi-zone architecture (#2) we see that chain reorganization operations dominate the critical path. This is due to the fact that moving ions between distant zones requires multiple hops to move thru intermediate zones. Finally for architecture #3, we see that the dominate operation is the time required to move (or shuttle) a qubit between traps. In this architecture, ions don't need to move between intermediate zones, like in architecture #2, but the zones are separated by greater distances.



Molecule(s)	N/q	Molecule(s)	N/q
H2	4	H12	24
H4	8	CH3F	26
BH, H6, LiH, NH	12	NaCN	38
BeH, CH2, H2O	14	CH3CO2H	48
CH2, BH2, NH2, H8, NH3	16	Si4	72
H8, N2, NaH	20	N8	80

Figure 8. Total runtime for each of the three candidate architectures. We take as benchmarks VQE circuits for molecules ranging in size from 4 to 80 orbitals, e.g., 4 to 80 qubits. In some cases, there are multiple molecules of the same size. For these we plot the average time.

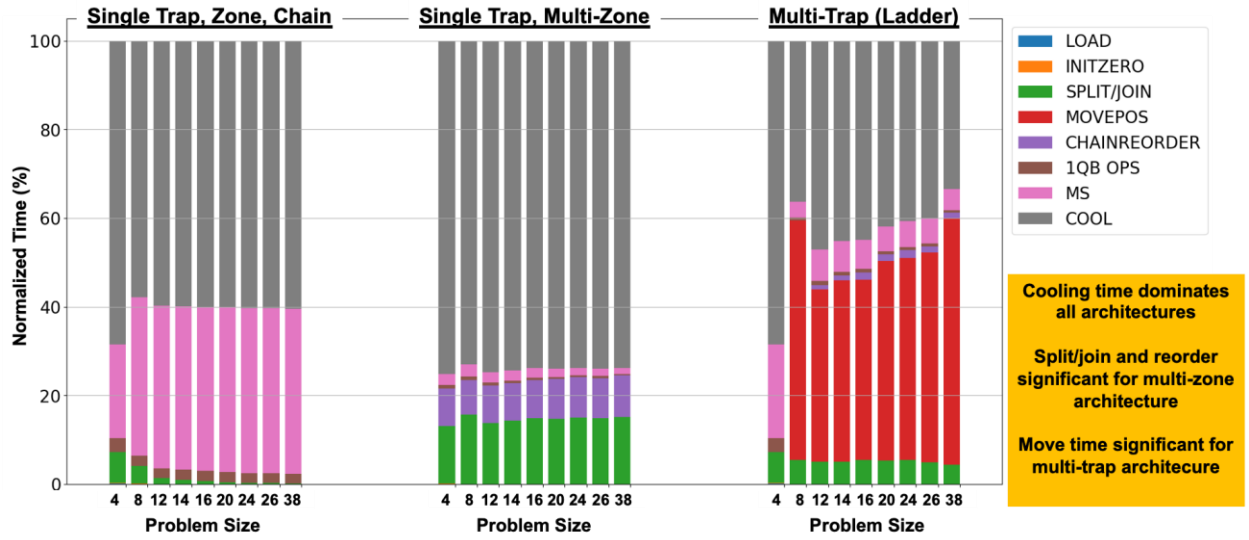


Figure 9. Critical path analysis for VQE problems on each of the three architectures. The critical path is the sequence of operations in the parallel execution path that determine the total runtime. We study problems varying in size from 4 to 38 orbitals (qubits) and show the normalized runtime.

This page intentionally left blank.

5. CONCLUSIONS AND FUTURE PLANS

In this second year of this project, we have been able to use the tools developed by the program to study the architectural tradeoffs in the design of ion-trap architectures. We have looked at three different architectures and found that the performance drivers for each are different. We have also developed four different families of algorithmic benchmarks. The studies shown in this report focus on one of the benchmarks, adaptive VQE applied to chemistry problems, however, our automated compilation and simulation tools will enable us to benchmark the remaining problems in the coming year.

One important aspect that is missing from our current analysis is the impact that architecture has on the fidelity of the executed benchmark. Fidelity is likely to be a more important factor in the performance of an architecture than execution time. This is why we are currently developing error models that incorporate the important aspects of the architectural configurations used in our study. One important factor is the impact that heating has on multi-qubit gate fidelity. We have developed a model to track the heating that accumulates as a result of the execution of a benchmark. We can then use this heating information in a quantum mechanical simulation of the benchmark to determine the fidelity. This capability will allow us to determine the best use of cooling operations to mitigate the impact of this heating.

This page intentionally left blank.

REFERENCES

1. J. Preskill, "Quantum Computing in the NISQ era and beyond," Quantum 2, vol. 79, 2018.

2. F. Arute and et. al., "Quantum supremacy using a programmable superconducting processor," Nature, no. 574, pp. 505-510, 2019.

3. IBM, 2019. [Online]. Available: https://www.ibm.com/blogs/research/2019/09/quantum-computation-center/.

4. J. D. J. F. C. e. a. Pino, "Demonstration of the trapped-ion quantum CCD computer architecture.," Nature, vol. 592, pp. 209-213, 2021.

5. D. Kielpinski, C. Monroe and D. Wineland, "Architecture for a Large-Scale Ion-Trap Quantum Computer," Nature, vol. 417, 2002.

6. R. J. Niffenegger, J. Stuart, C. Sorace-Agaskar, D. Kharas, S. Bramhavar, C. Bruzewicz, W. Loh, R. Maxson, R. McConnell, D. Reens, G. West, J. Sage and J. Chiaverini, "Integrated multi-wavelength control of an ion qubit," Nature, vol. 586, pp. 538-542, 2020.

7. J. McClean and et. al, "OpenFermion: The Electronic Structure Package for Quantum Computers," 2017. [Online]. Available: https://arxiv.org/abs/1710.07629.

8. Google, "Google Cirq," Google, [Online]. Available: https://cirq.readthedocs.io/en/stable/#.

9. H. R. Grimsley, S. E. Economou, E. Barnes and N. J. Mayhall, "An adaptive variational algorithm for exact molecular simulations on a quantum computer," Nature Communications, vol. 10, 2019.

10. A. G. Taube and R. J. Bartlett, "New perspectives on unitary coupled-cluster theory," Int. J. Quant. Chem, vol. 106, pp. 3393-3401, 2006.

11. J. Romero, R. Babbush, J. McClean, C. Hempel, P. Love and A. Aspuru-Guzik, "Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz.," Quantum S&T, vol. Vol. 4, no. Num. 1, 2018.

12. A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. Chow and J. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," Nature, vol. 549, pp. 242-246, 2017.

13. C. Monroe and J. Kim, "Scaling the Ion Trap Quantum Processor," Science, vol. 339, no. 6124, pp. 1164-1169, 2013.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
03-02-2022	Project Report	

4. TITLE AND SUBTITLE

Architectural Analysis of Quantum Algorithms for NISQ Hardware: FY21 Quantum System Sciences Line-Supported Program

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

K.M. Obenland, A. Kurlej, S.B. Alterman

5d. PROJECT NUMBER
2230-18

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02421-6426

8. PERFORMING ORGANIZATION REPORT NUMBER

LSP-351

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02421-6426

10. SPONSOR/MONITOR'S ACRONYM(S)
MIT LL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES

13. ABSTRACT

Emerging quantum computing architectures are at the cusp of showing relevance for real-world problems. However, current quantum computing hardware platforms are not designed with a focus on the problems to be run on them. Rather they are designed from the "bottom-up," where a quantum computing architecture typically consists of a homogeneous replication of a set of low-level devices, i.e., qubit designs. Additionally, demonstrations to-date have mostly been limited to "toy" problems or problems that have little practical value. The goal of our project is to investigate the architectural trade-offs that naturally occur in the design of emerging quantum computing hardware. We study these architectures using quantum benchmarks that are implementations of quantum algorithms that have practical impact. At the core of our project is a quantum architecture simulation tool that will allow us to assess the impact that architecture design choices have on the resources and fidelity of the designs. The current target of our work is on systems that are envisioned in the next five years (systems requiring 100-200 qubits).

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	None	34	**19b. TELEPHONE NUMBER** *(include area code)*
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED			