THE UNIVERSITY OF CHICAGO

ARCHITECTURAL DESIGN FOR EMERGING QUANTUM TECHNOLOGIES

THESIS PROPOSAL

DEPARTMENT OF COMPUTER SCIENCE

BY

JONATHAN M. BAKER

CHICAGO, ILLINOIS

JUNE 9, 2022

TABLE OF CONTENTS

ABSTRACT

Despite its relative infancy, there are a number of emerging quantum technologies for quantum computation, and it is unclear which will be the clear winner. Evaluation of these technologies at the architectural level, far beyond the small-scale prototypes of 1 to 2 qubits, is critical to producing viable systems capable of executing both near and long-term applications effectively. In order to fairly evaluate new hardware platforms, it is vital to develop technology-specific optimizations and compilation frameworks, pushing hardware closer to its fundamental limits rather than being hindered by ineffective software. Ultimately, our goal is to decide whether or not proposed technologies are able to scale efficiently into the future. Central questions in platform evaluation can be roughly categorized into three themes: 1. Does this technology support fast and economical program execution. 2 Can this technology produce consistent and high-quality program outputs? 3. Does this technology fit the requirements of practical applications now and/or in the future?

In this thesis, I explore the design, optimization, and evaluation of a variety of competing quantum technologies which are each vital when deciphering good candidates for scalable quantum computation, even in its very early stages. This manifests as several case studies serving as examples of a much larger, fundamental architectural questions. First, we consider what are the right abstractions for a quantum computing system by exploring the use of multivalued quantum logic to better make use of hardware capabilities. Second, we consider architectural trade-off spaces by considering neutral atom hardware. Third, we consider application-guided architectural design, evaluating how technology specific architectures can be designed to best fit target applications, like error correction codes. We examine the use of localized memory to enable virtualization of error corrected logical qubits. Finally, we explore a variety of technology specific (and agnostic) compilation optimizations to push proposed architectures and hardwares to their limits to best evaluate their ability to scale beyond prototypes and support large scale applications.

CHAPTER 1
INTRODUCTION

Despite recent booms in quantum information processing research, the field of practical quantum computing is relatively new. As an emerging paradigm, it may seem odd to consider new physical technologies, but the field is already at the horizon of some potentially game-changing capabilities. Although current machines have shown impressive success with devices based upon trapped ions and superconducting transmons, it is unclear what the eventual winning technologies will be and it is imperative to consider how to design and build large-scale architectures based subsets of these physical technologies.

Quantum hardware is still in its relative infancy, boasting tens of qubits as opposed to the thousands to millions needed to execute important algorithms for unordered search and factoring [11, 37]. Most available systems have struggled to scale beyond their prototypes while simultaneously suppressing gate errors and increasing qubit coherence times. This limits the types of programs which can execute effectively, let alone perform error correction. It is unclear whether systems composed of superconducting qubits or trapped ions, the major industry players will take the lead.

Device success is predicated on the increase in the number of qubits, reduction of gate errors below error correction thresholds, and increases in qubit coherence times, or lifetimes. In the near term, this translates into larger (more qubits), deeper programs (more operations per qubit) with improved output distributions (higher likelihood of obtaining the correct answers). In the long term, this translates into qubits which are protected from noise inherent in operating quantum systems which cannot be perfectly isolated from the environment while still having control in the form of encoded logical qubits.

In recent years, there have been a number of improvements throughout the compilation pipeline both close to the hardware and high level circuit optimization. These optimizations aim to reduce gate counts, circuit depth, and communication costs as proxies for increasing the size and quality of programs executable on currently available hardware. These are valuable optimizations and are often well correlated with improved program success rate or reducing physical requirements as we graduate into error corrected regime.

An alternative approach is to evaluate the viability and trade-offs of new quantum technology and associated architectures. This approach is tightly coupled to optimizations along the entire hardware-software stack, where it is much more advantageous to develop technology-specific toolchains. Despite tremendous efforts at both the hardware and software level to minimize the effects of noise, it is unclear if any of the available hardware will be able to scale as needed and no clear winner on underlying hardware has emerged. Even further, it is unknown whether this hardware is best suited to execute the desired programs and while it is often the case that we adapt compilation to the hardware, i.e. transforming applications into the right shape for execution, an alternative approach is explore how to design new architectures which are better suited for the applications we want to run.

Evaluating new hardware technology at the architectural level is decidedly different, though intrinsically coupled to, the development of quantum hardware. Device physicists' goal is often to demonstrate the *existence* of high quality qubits and operations in prototypes, while the architect's goal is to evaluate the systems-level ramifications of this technology, exploring the inherent trade-off spaces to determine viability in the near and long term. Perhaps most critically, this architectural design exploration leads to important insights about what is most important for hardware developers to optimize. For example if some limitations can be effectively mitigated via software, hardware developers can focus on other more fundamental issues. This process of co-design, by evaluating new technology early and often, is central to accelerating scalability.

In this dissertation, we explore several key themes in the design of technology-specific architectural frameworks. While we highlight these at a high level to support a much larger design principle, each chapter that follows aims to be self-contained.

- First, we consider how to determine the right abstractions for quantum computation, specifically one of the most fundamental considerations: the computing radix. Most quantum systems, like classical systems, are expressed using a two-level binary abstraction using qubits. However, many available hardware platforms composed of trapped ions or superconducting circuits are not intrinsically binary, instead offering access to a large, sometimes infinite, spectrum of possible logical states. Efficient use of these states could prove extremely useful in expediting quantum computation. For many quantum circuits, we can reduce the total execution time of the program by using additional space in the form of additional qubits or devices. When programs are time-limited it is critical to decompose circuits to minimize depth. This requires that we maintain a delicate balance as using additional space limits the total size of programs which can be executed if a large portion of resources must be reserved. We advocate in particular for the use of *intermediate qudits* (begin and end as qubits) which can be used to obtain the same asymptotic depth as the best known decompositions but can be achieved without the need for any extra space.

  The proposed techniques free up more hardware for computation, rather than dedicating device space as ancilla, limiting the maximum size of computation which can be performed. Current hardware is often calibrated for use only with qubits but has higher-level states available. While classically multivalued and mixed-radix computing is niche, it is important to evaluate the architectural ramifications of these strategies for quantum computation. By temporarily accessing already available higher states, these works can accelerate common circuit components and reduce space overhead, extending the frontier of what can be computed.

- Second, we consider technology-specific architectural tradeoffs by considering one promising platform candidate: neutral atoms. Current hardware implementations face unique and fundamental scalability challenges. While these devices have been useful as proof of concept demonstrations of small-scale, near-term algorithms, it is unclear whether any of them in present form will be able to execute the large-scale computation needed for quantum speedup. Evaluating the viability of new hardware is essential. Architectural studies which fully explore their unique trade-off spaces are key for finding the best way to accelerate beyond prototypes. One such alternative to superconducting qubits or trapped ions is neutral atoms.

There are distinct advantages, for example long distance interactions and native complex instructions, provided by neutral atoms by developing a dedicated compilation pipeline for executing quantum programs on the target hardware. Under similar gate error rates, neutral atom architectures will be capable of executing larger programs sooner than other competing technologies. Unfortunately, neutral atom systems suffer a potentially crippling drawback–atoms can be lost both during and between program execution which usually requires the entire array to be reloaded and the output to be discarded. Fortunately, software solutions can effectively mitigate this increased run time overhead. By developing several compiler solutions which are effective at mitigating this loss we are able to demonstrate viability for scalable quantum computation sooner. This work highlights a key design principle: exploring the use of new hardware serves a critical role in the development of the platform itself. By solving fundamental problems at the systems level with software, hardware developers can focus on solving and optimizing other problems. Co-design of quantum systems is key to accelerate the advancement of quantum computing technology.

- Third we consider architectural design driven by specific applications. In the long term, error correction is one of the most important applications underlying hardware must support and in the past it has been common to develop error correction codes with already available hardware in mind, for example surface codes and color codes which rely on only nearest-neighbor connectivity between devices to implement. While it has been effective, it may not be ideal. The alternative is to consider designing architectures using new hardware components to better fit codes, or generally applications. As an example, we consider using new quantum memory technology to better match surface codes.

In general, current quantum architectures do not tend to make a distinction between memory and processing of quantum information. These architectures are viable, however, as more and more qubits are needed the scalability challenges become apparent. To scale to the millions of qubits needed for error correction, a memory-based architecture can be used to decouple qubit-count from transmon count. We explore a proof-of-concept demonstration of its viability by virtualizing surface code tiles in a 2.5D memory-based architecture. Despite the surface code being designed with currently available 2D architectures in mind, this application is better matched with this 2.5D architecture which allows qubits to be virtualized and stored in local memory. This design reduces physical qubit requirements, enabling fast logical CNOTs, and expediting the creation of special resource states. This highlights a central theme for current quantum architecture design–we must explore the uses of new hardware technology, here the use of resonant cavities to store the information of many qubits, at the systems level.

- Fourth we consider the broad subject of quantum compilation pipelines, a critical component in the evaluation of any emerging quantum technology. Without architecture-specific compilation frameworks, it is very difficulty to effectively compare two technologies, and superficial modifications to preexisting frameworks, while usable, are inefficient. To best understand the limits of new architectures, we must integrate key features and constraints into the mapping, routing, and scheduling stages of our pipelines.

  We discuss several compiler optimizations which outperform generic methods and span several different architectural models. For example, for cluster-based architectures, we can use graph partitioning techniques to inform a timed-sliced routing compiler. Further, we can use effective lookahead heuristics to minimize unnecessary qubit displacement which lead to high latency cross-cluster interactions. These lookahead heuristics have proved essential for other compiler designs for both cavity-based superconducting devices as well as neutral atom devices, ensuring low communication overheads. For each of these architectures, we must adapt compiler design to account for device-specific constraints.

The central message is clear: generic and standard compilation flow can often be ineffective, introducing unnecessary communication overhead thereby reducing output quality. This leads to many device-agnostic optimizations. For example, most available devices cannot execute complex instructions like the Toffoli gate and instead decompose these gates early in the compilation pipeline. Routing and scheduling of these gates tends to be more difficult, requiring higher communication costs. However, by modifying this sequence, specifically by decomposing complex instructions based on target hardware topology and only after routing the interacting qubits together, this communication cost can be cut dramatically.

CHAPTER 2
DETERMINING THE RIGHT ABSTRACTIONS: MULTIVALUED QUANTUM LOGIC

2.1 Introduction

Given the severe constraints on quantum resources, it is critical to fully optimize the compilation of a quantum algorithm in order to have successful computation. Prior architectural research has explored techniques such as mapping, scheduling, and parallelism [6, 13, 18] to extend the amount of useful computation possible. In this chapter, we consider another technique: quantum trits (qutrits) and generally the use of qu*dits* for any number of logical levels d.

While quantum computation is typically expressed as a two-level binary abstraction of qubits, the underlying physics of quantum systems is not intrinsically binary. Whereas classical computers operate in binary states at the physical level (e.g. clipping above and below a threshold voltage), quantum computers have natural access to an infinite spectrum of discrete energy levels. In fact, hardware must actively suppress higher level states in order to achieve the two-level qubit approximation. Hence, using three-level qutrits is simply a choice of including an additional discrete energy level, albeit at the cost of more opportunities for error.

Prior work on qutrits (or more generally, d-level *qudits*) identified only constant factor gains from extending beyond qubits. In general, this prior work [31] has emphasized the information compression advantages of qutrits. For example, N qubits can be expressed as $\frac{N}{\log_2(3)}$ qutrits, which leads to $\log_2(3) \approx 1.6$-constant factor improvements in run times.

Our approach utilizes qutrits in a novel fashion, *intermediate qutrits*, essentially using the third state as temporary storage, but at the cost of higher per-operation error rates. Under this treatment, the run time (i.e. circuit depth or critical path) is *asymptotically* faster, and the reliability of computations is also improved. Moreover, our approach only applies qutrit operations in an intermediary stage: the input and output are still qubits, which is important for initialization and measurement on real devices [34, 35] and means it can easily be substituted for a binary-only version without needing to modify the remainder of the circuit.

The net result of this work is to extend the frontier of what quantum computers can compute. In particular, the frontier is defined by the zone in which every machine qubit is a data qubit, for example a 100-qubit algorithm running on a 100-qubit machine. In this frontier zone, we do not have room for non-data workspace qubits known as ancilla. The lack of ancilla in the frontier zone is a costly constraint that generally leads to inefficient circuits. For this reason, typical circuits instead operate below the frontier zone, with many machine qubits used as ancilla. We demonstrate that ancilla can be substituted with qutrits, enabling us to operate efficiently within the ancilla-free frontier zone.

We can further improve on this result in a more general way by instead having ancilla, specifically clean ancilla, be *generated* local during the decomposition of an algorithm into a quantum circuit. That is, we propose a new circuit which performs qubit-qudit compression storing the information of many qubits as a small number of qudits at the cost of some gate overhead. These compression circuits produce clean ancilla in the $|0\rangle$ state. The stored data can be retrieved later when needed since all quantum operations are reversible. Essentially, when certain groups of qubits will be unused for a long period of time, we can repurpose them by compressing them and using the produced ancilla. This "compression" is a rearrangement of the stored binary values into higher states. This allows us to store more information into the same number of physical quantum devices and free up qubits for computation.

In this work we present an application of this technique to give logarithmic depth decompositions of quantum arithmetic circuits - a carry lookahead adder and by extension addition by a constant a direct improvement to the more "clever" version we present at first.

We present two compression circuits for qubit-qutrit and qubit-ququart compression and evaluate advantages of various compression schemes in real applications.

We highlight the primary contributions of this work:

1. A Generalized Toffoli circuit construction based on qutrits that leads to asymptotically faster circuits ($633N \rightarrow 38\log_2 N$) than equivalent qubit-only constructions. We also reduce total gate counts from $397N$ to $6N$.

2. Larger arithmetic circuit constructions with intermediate qutrits such as the Incrementer, $A+B$ and $+K$ adders which obtain $O(\log^2 n)$, $O(\log^3 n)$, and $O(\log^3 n)$ depth, respectively.

3. Simulation results using the simulator developed in [9], under realistic noise models, which demonstrate our circuit construction outperforms equivalent qubit circuits in terms of error. For completeness, we also benchmark our circuit against a qubit-only construction augmented by an ancilla and find our construction is still more reliable.

4. The introduction of a novel use of intermediate qu*dits* based on the compression of information on many quantum devices into a small number but with more logical states to directly enable allocation of devices as ancilla which generalizes the use of intermediate qudits for any circuit.

5. FUTURE. Development of control systems to physically implement qudit operations and practically realize these designs.

CHAPTER 3
ARCHITECTURAL TRADE-OFFS IN EMERGING TECHNOLOGY: NEUTRAL ATOM ARCHITECTURES

3.1 Introduction

In the past several years, many leading gate-based quantum computing technologies such as trapped ions and superconducting qubits have managed to build small-scale systems containing on the order of tens of qubits [1, 17, 38, 41]. However, each of these systems have unique scalability challenges [4, 19]. For example, IBM devices have continued to grow in size while error rates have remained high, larger than what is needed for quantum error correction [10]. Trapped ion machines, despite many promising results, have fundamental challenges in controllability [27]. It is unclear whether any of these platforms in present form will be capable of executing large-scale quantum computation needed for algorithms with quantum speedup like Grover's [37] or Shor's [11].

These challenges are fundamental to the underlying technology. Consequently, current approaches which aim to reduce error via software and push the limits of current devices are insufficient for long-term scalability. In recent years there has been a number of improvements to the compilation pipeline stretching across the hardware-software stack [5, 14, 15, 26, 28, 30, 39, 40]. Numerous studies have explored reductions in quantum circuit gate counts, depths, and communication costs via heuristics and optimizations, but cannot overcome the fundamental scalability limitations of the technology.

An alternative approach is to consider emerging quantum technologies and new architectures. In this work, we explore hardware composed of arrays of individually-trapped, ultra-cold neutral atoms which have shown great promise and have unique properties which make them appealing from a software standpoint [36]. These properties include: potential for high-fidelity quantum gates, indistinguishable qubits that enable scaling to many qubits,

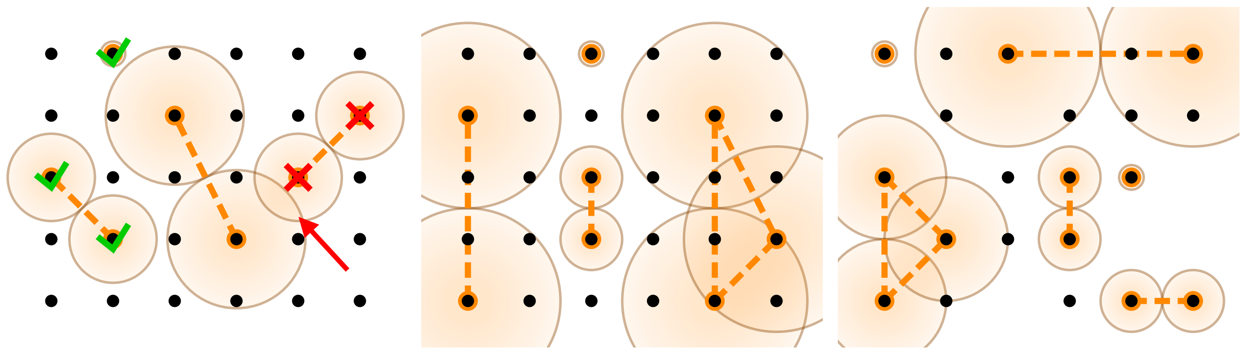
(a) Restriction Zone (b) Max Distance 3 (c) Atom Loss

Figure 3.1: Examples of interactions on a neutral atom device. (a) Interactions of various distances are permitted up to a maximum. Gates can occur in parallel if their zones do not intersect. The interaction marked with green checks can occur in parallel with the middle interaction. (b) The maximum interaction distance specifies which physical qubits can interact. Compiler strategies suited for this variable distance are needed for neutral atom architectures. (c) Neutral atom systems are prone to sporadic atom loss. Efficient adaptation to this loss reduces computation overhead.

long-range qubit interactions that approximate global (or generally high) connectivity, and the potential to perform complex multiqubit (≥ 3 operands) operations without expensive decompositions to native gates [21].

Neutral-atom (NA) architectures also face unique challenges. Long-range interactions induce zones of restriction around the operating qubits which prevent simultaneous operations on qubits in these zones. Most importantly, the atoms in a neutral atom device can be lost via random processes during and between computation. In the worst case, the compiled program no longer fits on the now sparser grid of qubits, requiring a reload of the entire array every cycle. This is a costly operation to repeat for thousands of trials. Coping with this loss in a time-efficient manner is important to minimizing the run time of input programs while not dramatically increasing either gate count or depth, both of which will reduce program success rate. In Figure 3.1 we show a small piece of a NA system with many gates of various sizes and distances being executed in parallel. Restriction zones are highlighted and importantly no pair of gates have intersecting restriction zones. When atoms are lost during computation, rather than a uniform grid wehave a much sparser graph and qubits will be further apart on average. A key to the success of a NA system is resilience to loss of atoms, avoiding expensive reloads.

In this work, we explore the trade-offs in neutral atom architectures to assess both current viability and near future prospects. We extend current compilation methods to directly account for the unique NA constraints like long-range interactions, areas of restriction, and native implementation of multiqubit gates. We propose several coping strategies at the hardware and software level to adapt to loss of atoms during program execution; we evaluate the tradeoff of execution time and resilience to atom loss versus program success rate.

The field of quantum computation is still early in development and no clear winner for underlying hardware has emerged. It is vital to consider new technology and evaluate its potential early and often to determine viability as it scales. In this work, we do just that, considering a neutral atom architecture of size comparable to target hardware sizes for competing technologies. Despite comparably higher gate errors and lack of large scale demonstration, we determine if the unique properties offered by this new technology enable more efficient computation at scale. Perhaps more importantly, this work can serve as a guide for hardware developers. We demonstrate that the fundamental problem of atom loss can be mitigated via software solutions and doesn't need to be highly optimized at the hardware level. This allows hardware engineers to focus on other fundamental problems which cannot easily be mitigated by software, such as gate error rate.

In this work we introduce a scalable neutral atom architecture based on demonstrated physical implementations which permit long range interactions and native multiqubit gates. The specific major contributions of our work are the following:

- Adapt current quantum compiler technology by extending prior work to explicitly account for interaction distance, induced restriction zones, and multiqubit gates.

- Evaluate system-wide implications of these properties, specifically reduced gate counts and depth at the cost of increased serialization. Our compiler exploits the gain while mitigating this cost.

- Demonstrate via simulation based on experimental results and through program error analysis the ability of NA systems to quickly surpass competitors in the intermediate-term despite currently worse gate errors,
- Model sporadic atom loss in NA systems and propose hardware and compiler solutions to mitigate run time and program error rate overheads. We explore each strategy's resilience to atom loss, the effect on expected program success rate, and overall run time.

CHAPTER 4
APPLICATION-GUIDED ARCHITECTURAL DESIGN. VIRTUALIZING ERROR CORRECTED QUBITS

4.1 Introduction

Quantum devices have improved significantly in the last several years both in terms of physical error rates and number of usable qubits. For example, IBM and others have made accessible via the cloud several devices with 5 to 53 qubits with moderate error rates [2]. Concurrently, great progress has been made at the software level such as improved compilation procedures reducing required overhead for program execution. These efforts are directed at enabling NISQ (Noisy Intermediate-Scale Quantum) [32] algorithms to demonstrate the power of quantum computing. Machines in this era are expected to run some important programs and have recently been used to by Google to demonstrate "quantum supremacy" [3].

Despite this, these machines will be too small for error correction and unable to run large-scale programs due to unreliable qubits. The ultimate goal is to construct fault-tolerant machines capable of executing thousands of gates and in the long-term to execute large-scale algorithms such as Shor's [37] and Grover's [11] with speedups over classical algorithms. There are a number of promising error correction schemes which have been proposed such as the color code [20] or the surface code [7, 8, 16]. The surface code is a particularly appealing candidate because of its low overhead, high error threshold, and its reliance on few nearest-neighbor interactions in a 2D array of qubits, a common feature of superconducting transmon qubit hardware. In fact, Google's next milestone is to demonstrate error corrected qubits [3, 24].

Current architectures for both NISQ and fault-tolerant quantum computers make no distinction between the memory and processing of quantum information (represented in qubits). While currently viable, as larger devices are built, the engineering challenges of scaling up to hundreds of qubits becomes readily apparent. For transmon technology used by Google, IBM, and Rigetti, some of these issues include fabrication consistency and crosstalk during parallel operations. Every qubit needs dedicated control wires and signal generators which fill the refrigerator the device runs in. To scale to the millions of qubits needed for useful fault-tolerant machines [8], we need to a memory-based architecture to decouple qubit-count from transmon-count.

In this work, we use a recently realized qubit memory technology which stores qubits in a superconducting cavity [29]. This technology, while new, is expected to become competitive with existing transmon devices. Stored in cavity, qubits have a significantly longer lifetime (coherence time) but must be loaded into a transmon for computation. Although the basic concept of a compute qubit and associated memory has been demonstrated experimentally, the contribution of our work is to design and evaluate a system-level organization of these components within the context of a novel surface code embedding and fault-tolerant quantum operations. We provide a proof of concept in the form of a practical use case motivating more complex experimental demonstrations of larger systems using this technology.

Our proposed 2.5D memory-based design is a typical 2D grid of transmons with memory added as shown in Figure 4.1. This can be compared with the traditional 2D error correction implementation in Figure **??**, where the checkerboards represent error-corrected logical qubits. The logical qubits in this system are stored at unique virtual addresses in memory cavities when not in use. They are loaded to a physical address in the transmons and made accessible for computation on request and are periodically loaded to correct errors, similar to DRAM refresh. This design allows for more efficient operations such as the transversal CNOT between logical qubits sharing the same physical address i.e. co-located in the same cavities. This is not possible on the surface code in 2D which requires methods such as braiding or lattice surgery for a CNOT operation.

We introduce two embeddings of the 2D surface code to this new architecture that spread

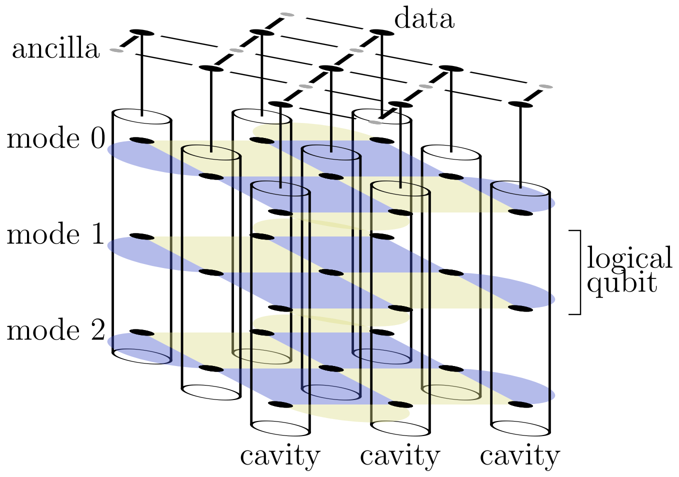

Figure 4.1: Our fault-tolerant architecture with random-access memory local to each transmon. On top is the typical 2D grid of transmon qubits. Attached below each data transmon is a resonant cavity storing error-prone data qubits (shown as black circles). This pattern is tiled in 2D to obtain a 2.5D array of logical qubits. Our key innovation here is storing the qubits that make up each logical qubit (shown as checkerboards) across many cavities to enable efficient computation.

logical qubits across many cavities. Despite serialization due to memory access, we are able to store and error-correct stacks of these logical qubits. Furthermore, we show surface code operations via lattice surgery can be used unchanged in this new architecture while also enabling a more efficient CNOT operation. Similarly, we are able to use standard and architecture-specific magic-state distillation protocols [23] in order to ensure universal computation. Magic-state distillation is a critical component of error-corrected algorithms so any improvement will directly speed up algorithms including Shor's and Grover's.

We discuss several important features of any proposed error correction code, such as the threshold error rate (below which the code is able to correct more errors than its execution causes), the code distance, and the number of physical qubits to encode a logical qubit. In many codes, the number of physical qubits can be quite large. We develop an embedding from the standard representation to this new architecture which reduces the required number of physical transmon qubits by a factor of approximately k, the number of resonant modes per cavity. We also develop a Compact variant saving an additional 2x. This is significant because we can obtain a code distance $\sqrt{2k}$ times greater or use hardware with only $\frac{1}{2k}$ the required physical transmons for a given algorithm. In the near-to-intermediate term, when qubits are a highly constrained resource this will accelerate a path towards fault-tolerant computation. In fact, the smallest instance of Compact requires only 11 transmons and 9 cavities for k logical qubits.

We evaluate variants of our architecture by comparing against the surface code on a larger 2D device. Specifically, we determine the error correction threshold rates via simulation for each and find they are all close to the baseline threshold. This shows the additional error sources do not significantly impact the performance. We explore the sensitivity of the threshold to many different sources of error, some of which are unique to the memory used in this architecture. We end by evaluating magic-state distillation protocols which have a large impact on overall algorithm performance and find a 1.22x speedup normalized by the number of transmon qubits.

In summary, we make the following contributions:

- We introduce a 2.5D architecture where qubit-local memory is used for random access to error-corrected, logical qubits stored across different memories. This allows a simple virtual and physical address scheme. Error correction is performed continuously by loading each from memory.

- We give two efficient adaptations of the surface code in this architecture, Natural and Compact. Unlike a naive embedding, both support fast transversal CNOTs in addition to lattice surgery operations with improved connectivity between logical qubits.

- We develop an error correction implementation optimized for Compact and designed to maximise parallelism and minimize the spread of errors.

- Via simulation, we determine the surface code adapted to our 2.5D architecture is still an effective error correction code while greatly reducing hardware requirements.

CHAPTER 5
EVALUATING ARCHITECTURES AT THEIR LIMITS: IMPROVED COMPILATION METHODS

5.1 Introduction

In the last few years, quantum computing has taken major steps forward towards becoming practical, useful for more than distant, untenable algorithms [**? ?**]. More and more devices are becoming available with on the order of 10s of quantum bits, albeit noisy. One of the most important emerging problems is scalability of current hardware in order to execute interesting problems in the current, Noisy-Intermediate Scale Quantum (NISQ) [33], era with the goal to demonstrate realizable quantum advantage or even quantum supremacy [3].

There are several competing technologies for the underlying implementation of qubits such as trapped ions or superconducting circuits. Each of these technologies present unique challenges to scalability, at least without error correction. For example, in superconducting technology with limited connectivity between hardware qubits, numerous additional operations called SWAPs must be added to an input program in order to execute it. For example, in Linear-Nearest-Neighbor and a 2D grid topologies on the order of thousands of additional operations must be inserted for execution, many of which are derived from increased average distance between qubits, as shown in Figure 5.1. With so many additional operations, the input programs are almost guaranteed to fail. These added operations dramatically increase the execution time of these programs, moving far beyond the coherence limit of current qubits (approximately the lifetime of qubits).

Past efforts have focused primarily on the compilation problem to these small, near-term devices, such as variation aware mappings or SWAP reductions [12, 22, 25, 42, 43]. These techniques have improved the ability of programs to succeed on currently available devices, but do not provide a path to scalability. There are inherit limits in current quantum architectures,

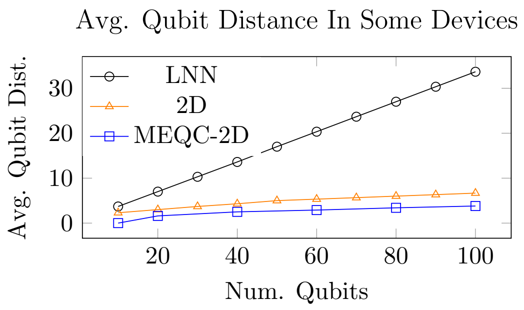

Figure 5.1: Average qubit distance on several instances of near-term target architectures. Average qubit distance approximates how many SWAPs are necessary to interact an arbitrary pair of qubits. LNN architectures scale extremely poorly in this metric resulting in a large number of added gates and depth. 2D and MEQC architectures scale much better. We show this translates into reduced number of gates and reduced depth as we scale into the future.

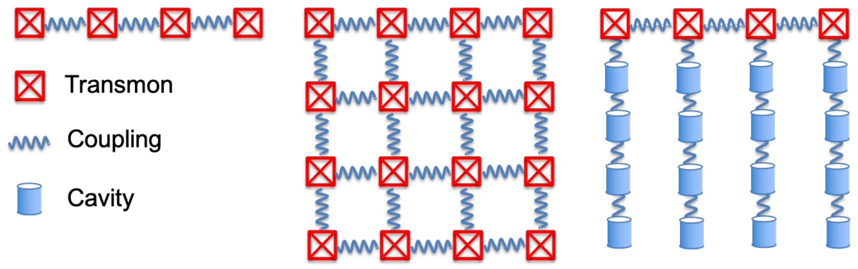

Figure 5.2: Both current and the proposed MEQC device. On the left is a LNN device where adjacent superconducting transmons are coupled enabling two qubit interactions. In the center is a 2D mesh architecture common among current manufacturers. On the right is the proposed MEQC architecture with transmons arranged in a line. Each transmon has an attached cavity which stores in memory multiple qubits. To operate on the qubits, they must first be loaded into the transmons.

such as gate error rates or qubit coherence times. Even with improved error rates or longer qubit lifetimes, the large gate and depth overheads induced by limited connectivity are too cumbersome.

We instead propose a new architecture, Memory-Equipped Quantum Computing (MEQC). We use superconducting qubits equipped with resonator cavities which can store qubits in their modes. Recently, small, physical prototypes of this have been realized and experimented with; however, the architectural implications of this new technology are almost entirely unexplored. For example, in the original experiments, the proposed benefit of this technology was to store less frequently used qubits in long coherent memory. Qubits stored in the modes of the attached cavity are expected to have an order of magnitude longer coherence times than typical transmon coherence times. This directly will reduce the frequency of idle errors on qubits which are unused. Idle errors are due inability to isolate qubits perfectly from the environment while still being able to manipulate them. In other superconducting devices, all qubits are roughly equally subject to these errors. In the proposed architecture, stored qubits are more isolated, resulting in protection from these idle errors.

This benefit is secondary to another hugely important feature for near- and intermediate-term and that is connectivity. In the proposed architecture, the transmon-cavity technology enables a gate to be applied, via transmon mediation, to any pair of qubits stored in the same cavity. This random access to stored qubits greatly improves local connectivity between qubits in these devices. This translates into significant reduction in compilation overhead, reducing the need for large numbers of SWAPs, but only if the compilation procedure properly accounts for these well connected regions. While experimentalists anticipated the coherence times of cavities to be the critical advantage, we find in our proposed MEQC architecture the primary benefit is this random access. For our proposed architecture, we provide a complete compilation framework transforming input quantum circuits to ones executable in hardware utilizing transmon-local memory. Our framework explicitly maximizes the advantages of both of these features to minimize compilation overhead.

Neither of these gains come for free. In the proposed architecture, in order to execute a gate we must first load the qubit from memory into a transmon which can be manipulated. Therefore, operations on this architecture require on average two additional operations in the form of Loads and Stores. For small programs, this architecture will require more operations than others. Furthermore, since there is only a single operational transmon per cavity, this prevents gates from being executed in parallel on qubits located in the same cavity. The gains of this new architecture in terms of scalability outweigh these downsides, specifically by reducing the number of total operations required to execute input programs.

In summary, we make the following contributions:

- We introduce a new scalable quantum architecture, MEQC, which takes advantage of recent quantum memory-like hardware increasing local qubit connectivity and protects infrequently used qubits from idle errors.

- We develop a full compilation framework for MEQC devices which includes heuristics to maximize time spent in long coherent memory when unused and heuristics for mapping and routing of qubits during execution which minimizes the total number of SWAP operations inserted.

- We demonstrate our system reduces required gate and depth overhead substantially over other competitive options and subsequently increases the chance to succeed.

- We analyze different architecture-specific design choices such as number of modes per cavity, and top-level transmon connectivity. Furthermore, our system is able tolerate up to 12x worse transmon-transmon interconnect error which is expected to be the dominant source of error in systems utilizing transmon-cavity technology in near-term. We conclude our architecture presents a path towards scalability in the near and intermediate term.

CHAPTER 6
TIMELINE FOR COMPLETION

This thesis will be comprised of the following works that I've completed in the last four years:

- Exploiting Long Distance Interactions and Tolerating Atom Loss in Neutral Atom Quantum Architectures, ISCA 2021
- Virtual Logical Qubits: A Compact Architecture for Fault-Tolerant Quantum Computing, MICRO 2020
- Orchestrated Trios: Compiling for Efficient Communication in Quantum Programs with Three-Qubit Gates, ASPLOS 2021
- Memory-Equipped Quantum Architectures: The Power of Random Access, PACT 2020
- Improved Quantum Circuits via Intermediate Qutrits, TQC 20220
- Time-Sliced Quantum Circuit Partitioning for Modular Architectures, CF 2020
- Efficient Quantum Circuit Decompositions via Intermediate Qudits, ISMVL 2020
- Asymptotic Improvements to Quantum Circuits via Qutrits, ISCA 2019
- Noise-Adaptive Compiler Mappings for Noisy Intermediate-Scale Quantum Computers, ASPLOS 2019

For completion of this thesis my primary focus is on unifying the message of these works to best capture the major theme of my work: exploration of emerging quantum technologies and the development of technology specific compilation frameworks to best evaluate their potential. I plan to finish writing by the end of July.

REFERENCES

[1] A preview of bristlecone, google's new quantum processor, Mar 2018.

[2] Quantum devices & simulators, Jun 2018.

[3] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[4] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews*, 6(2):021314, 2019.

[5] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah. On the qubit routing problem. *arXiv preprint arXiv:1902.08091*, 2019.

[6] Y. Ding, A. Holmes, A. Javadi-Abhari, D. Franklin, M. Martonosi, and F. Chong. Magic-state functional units: Mapping and scheduling multi-level distillation circuits for fault-tolerant quantum architectures. In *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 828–840. IEEE, 2018.

[7] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3):032324, 2012.

[8] C. Gidney and M. Ekerå. How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. *arXiv preprint arXiv:1905.09749*, 2019.

[9] P. Gokhale, J. M. Baker, C. Duckering, N. C. Brown, K. R. Brown, and F. T. Chong. Asymptotic improvements to quantum circuits via qutrits. In *Intl. Symp. on Computer Architecture (ISCA)*, June 2019.

[10] D. Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. In *Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics*, volume 68, pages 13–58, 2010.

[11] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Annual ACM Symposium on Theory of Computing*, pages 212–219. ACM, 1996.

[12] G. G. Guerreschi and J. Park. Two-step approach to scheduling quantum circuits, 2017.

[13] G. G. Guerreschi and J. Park. Two-step approach to scheduling quantum circuits. *Quantum Science and Technology*, 3(4):045003, jul 2018.

[14] G. G. Guerreschi and J. Park. Two-step approach to scheduling quantum circuits. *Quantum Science and Technology*, 3(4):045003, 2018.

[15] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima. An efficient conversion of quantum circuits to a linear nearest neighbor architecture. *Quantum Information and Computation*, 11(1):142, 2011.

[16] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter. Surface code quantum computing by lattice surgery. *New Journal of Physics*, 14(12):123011, 2012.

[17] IBM Quantum Devices. `https://quantumexperience.ng.bluemix.net/qx/devices`. Accessed: 2020-11-24.

[18] A. Javadi-Abhari, P. Gokhale, A. Holmes, D. Franklin, K. R. Brown, M. Martonosi, and F. T. Chong. Optimized surface code communication in superconducting quantum computers. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-50 '17, pages 692–705, New York, NY, USA, 2017. ACM.

[19] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11:369–395, 2020.

[20] A. J. Landahl, J. T. Anderson, and P. R. Rice. Fault-tolerant quantum computing with color codes. *arXiv preprint arXiv:1108.5738*, 2011.

[21] H. Levine, A. Keesling, G. Semeghini, A. Omran, T. T. Wang, S. Ebadi, H. Bernien, M. Greiner, V. Vuletić, H. Pichler, et al. Parallel implementation of high-fidelity multiqubit gates with neutral atoms. *Physical review letters*, 123(17):170503, 2019.

[22] G. Li, Y. Ding, and Y. Xie. Tackling the qubit mapping problem for nisq-era quantum devices, 2018.

[23] D. Litinski. A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum*, 3:128, 2019.

[24] J. Martinis. Quantum supremacy using a programmable superconducting processor, 11 2019. Institute for Quantum Information and Matter Seminar at the California Institute of Technology.

[25] P. Murali, J. M. Baker, A. J. Abhari, F. T. Chong, and M. Martonosi. Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers, 2019.

[26] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi. Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1015–1029, 2019.

[27] P. Murali, D. M. Debroy, K. R. Brown, and M. Martonosi. *Architecting Noisy Intermediate-Scale Trapped Ion Quantum Computers*, page 529–542. IEEE Press, 2020.

[28] P. Murali, D. C. McKay, M. Martonosi, and A. Javadi-Abhari. Software mitigation of crosstalk on noisy intermediate-scale quantum computers. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1001–1016, 2020.

[29] R. Naik, N. Leung, S. Chakram, P. Groszkowski, Y. Lu, N. Earnest, D. McKay, J. Koch, and D. Schuster. Random access quantum information processors using multimode circuit quantum electrodynamics. *Nature communications*, 8(1):1904, 2017.

[30] Y. Nam, N. J. Ross, Y. Su, A. M. Childs, and D. Maslov. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information*, 4(1):1–12, 2018.

[31] A. Pavlidis and E. Floratos. Arithmetic circuits for multilevel qudits based on quantum fourier transform. *arXiv preprint arXiv:1707.08834*, 2017.

[32] J. Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, Aug. 2018.

[33] J. Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, Aug. 2018.

[34] J. Randall, A. M. Lawrence, S. C. Webster, S. Weidt, N. V. Vitanov, and W. K. Hensinger.

Generation of high-fidelity quantum control methods for multilevel systems. *Phys. Rev. A*, 98:043414, 10 2018.

[35] J. Randall, S. Weidt, E. D. Standing, K. Lake, S. C. Webster, D. F. Murgia, T. Navickas, K. Roth, and W. K. Hensinger. Efficient preparation and detection of microwave dressed-state qubits and qutrits with trapped ions. *Phys. Rev. A*, 91:012322, 01 2015.

[36] M. Saffman. Quantum computing with atomic qubits and rydberg interactions: progress and challenges. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 49(20):202001, 2016.

[37] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, Oct. 1997.

[38] R. S. Smith, M. J. Curtis, and W. J. Zeng. A practical quantum instruction set architecture. *arXiv preprint arXiv:1608.03355*, 2016.

[39] S. S. Tannu and M. Qureshi. Ensemble of diverse mappings: Improving reliability of quantum computers by orchestrating dissimilar mistakes. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 253–265, 2019.

[40] R. Wille, L. Burgholzer, and A. Zulehner. Mapping quantum circuits to ibm qx architectures using the minimal number of swap and h operations. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2019.

[41] K. Wright, K. Beck, S. Debnath, J. Amini, Y. Nam, N. Grzesiak, J.-S. Chen, N. Pisenti, M. Chmielewski, C. Collins, et al. Benchmarking an 11-qubit quantum computer. *Nature communications*, 10(1):1–6, 2019.

[42] X. Zhang, H. Xiang, T. Xiang, L. Fu, and J. Sang. An efficient quantum circuits optimizing scheme compared with qiskit, 2018.

[43] A. Zulehner, A. Paler, and R. Wille. An efficient methodology for mapping quantum circuits to the ibm qx architectures, 2017.