Zeptojoule Computing: Superconducting adiabatic logic for scalable energy-efficient computing

by

L. Camron Blackburn

B.A. Physics, New York University, 2017
M.S. Massachusetts Institute of Technology, 2021

Submitted to the Program in Media Arts and Science,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2026

Authored by: L. Camron Blackburn
Program in Media Arts and Science
May 15, 2026

Certified by: Neil Gershenfeld
Director of MIT Center for Bits and Atoms, Thesis Supervisor

Certified by: Karl K. Berggren
Julius A. Stratton Professor in Electrical Engineering and Physics, Thesis Supervisor

Accepted by: Joseph A. Paradiso
Academic Head
Program in Media Arts and Science

Zeptojoule Computing: Superconducting adiabatic logic for scalable energy-efficient computing

by

L. Camron Blackburn

Submitted to the Program in Media Arts and Science,
School of Architecture and Planning
on May 15, 2026 in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

ABSTRACT

The world's artificial intelligence data centers are on track to consume more electricity than many industrialized nations, but the silicon transistors powering them dissipate energy 100,000 times the physical minimum dictated by thermodynamics. In contrast, the Adiabatic Quantum Flux Parametron (AQFP) is a superconducting digital logic device capable of switching at energies near this fundamental thermodynamic limit: AQFP dissipates about 10^{-21} J per operation at 5 GHz. Even accounting for a $\sim$1000 W/W cryogenic cooling overhead to maintain superconducting operation at 4 K, this represents roughly 100$\times$ lower energy dissipation than the $\sim 10^{-16}$ J switching energy of modern CMOS transistors. Yet superconducting digital logic has long struggled to translate device-level efficiency into practical system-level gains, hindered by the absence of dense superconducting memory, complex clocking networks that limit scalability, and, until recently, the continued dominance of Moore's law CMOS scaling.

This dissertation demonstrates how AQFP circuits can move beyond device-level promise toward system-level viability through contributions at four levels of the hardware abstraction stack. First, I develop synchronizer circuits that relax AQFP's rigid timing requirements, enabling scalable multi-clock domain designs. Second, I design, fabricate, and test a compact AQFP SR-Loop register file for low-level on-chip memory. Third, I introduce a fine-grained multithreaded accelerator design that exploits AQFP's gate-level clocking to achieve high throughput and low-power compute on linear algebra workloads. Fourth, I adapt established CMOS accelerator modeling methods for superconducting electronics, enabling the first quantitative full-stack design space exploration tool for AQFP systems on real-world AI workloads. Taken together, these contributions chart a practical path toward scalable superconducting computing with the potential for orders-of-magnitude improvement in energy performance.

Thesis supervisor: Neil Gershenfeld
Title: Director of MIT Center for Bits and Atoms

Thesis supervisor: Karl K. Berggren
Title: Julius A. Stratton Professor in Electrical Engineering and Physics

THESIS COMMITTEE

Thesis Supervisors

Neil Gershenfeld
Director of MIT Center for Bits and Atoms
Program in Media Arts and Science

Karl K. Berggren
Faculty Head of Electrical Engineering
Julius A. Stratton Professor in Electrical Engineering and Physics
Department of Electrical Engineering and Computer Science

Thesis Reader

Vivienne Sze
Professor
Department of Electrical Engineering and Computer Science

for Kathleen Blackburn,
and the whole Blackburn family.

Acknowledgments

The path through my PhD was shaped by so many people, some providing only brief interactions with lasting impact and others that were in the grind with me through thick and thin. Everyone knows, this is the hardest dissertation section to write and simultaneously the most widely read. It's impossible to capture it all, but what follows is an attempt to put into words the profound gratitude I feel for the opportunity, support, collaboration, and friendship that I've had over these years.

First and foremost, I would like to thank my PhD advisors, both Neil Gershenfeld and Karl Berggren.

When deciding to go to graduate school, I applied exclusively to physics PhD programs, except for one unconventional path at the MIT Center for Bits and Atoms. I'd like to thank Neil for convincing me that forever needing to explain why I have a PhD in media arts and science from the school of architecture and planning instead of physics or electrical engineering, was going to be worth the journey. It truly was. Neil, thank you for exponentially broadening my understanding of the world, teaching me how to make almost anything, pushing me to think big, training us all in how to frame our research with narrative to make an impact, and most importantly, giving us the freedom to explore and have fun through it all. The CBA is an inspiring place that I've been privileged to be a part of, with access to and freedom with some of the coolest tools and equipment I could've imagined.

I'd like to thank Karl for giving me the PhD experience that I had looked up to since I was an undergrad. The diligence and care you put into mentoring, teaching, and advising high-caliber research is a constant reminder of why I chose to get a PhD in the first place. Your support and encouragement have shaped me into a much stronger scientist and engineer, and have made it possible for me to produce work I can feel proud of. It's been an honor to learn from you over these years, and I'm walking away with so many lessons, both technical and non-technical, that I feel lucky to have learned.

I want to thank Vivienne Sze for serving on my committee and for providing invaluable guidance as my work expanded up the computer hardware stack. I've been inspired by your work since I first arrived at MIT. Thank you for writing the textbook on the topic that motivates the application layer of my work - it has been an honor to have you on my committee.

There are three people, without which many of the key results of this thesis would not have been possible:

I'd like to thank Alex Wynn. Without Alex, I probably would not know what an Adiabatic Quantum Flux Parametron is, let alone get from designing simple logic gates in my master's to a full accelerator by the end of my PhD. It's truly insane what we've accomplished as a team over the last 5+ years, and so much of that can be attributed to your leadership, drive, and determination. I'm excited for the next chapter, continuing to build this technology together. Thank you for shaping this journey in profound ways.

I'd like to thank David Russo for teaching me the practical tools I needed to design a superconducting integrated circuit, for significantly improving many of my original circuit designs, and for being an impressively diligent and organized engineer.

And finally, Tanner Andrulis for building the open source tools that enabled the architecture design portion of this thesis and, more importantly, for teaching me how to use them and collaborating over the last year.

I owe a significant acknowledgment to MIT Lincoln Laboratory for the funding and infrastructure that supported this entire body of work. The results and experiments of my PhD would not have been possible without the MITLL SFQ5ee fabrication process, and access to the lab for all of the experimental testing and chip design that I was able to do on site. I'd like to thank the Group 89 leadership for giving me the opportunity to spend so much of my PhD at Lincoln Laboratory, and the many collaborators and researchers in Group 89 for the countless scientific discussions and for always being willing to take a moment to answer questions or help me find equipment I needed: Evan Golden, Neel Parmar, David Goldfinger, Sergey Tolpygo, Andrew Wagner, David Russo, Matt Guyton, Leonard Johnson, Dan Oates, Richard Younger, and Alex Wynn.

Thank you to Evan Golden, first at Lincoln, for teaching me how to test my circuits in liquid helium, and then proceeding to join QNN and be one of the best office mates I've had - thanks for the long research discussions, the endless supply of interesting candies and caffeine, and for always bringing excellent lighting and ambiance to our windowless office.

To all past and present members of the Quantum Nanostructures and Nanofabrication group: Alejandro Simon, John Simonaitis, Owen Medeiros, Emma Batson, Adina Bechhofer, Matteo Castellani, Reed Foster, Francesca Incalza, Stewart Koppell, Tareq El Dandachi, Dip Joti Paul, Tony Zhou, Gian Luca Dolso, Evan Golden, Ben Mazur, Davide Mondin, Malick Sere. Thank you for the scientific discussion and companionship through this grad school journey. And specifically, thank you to Reed for always providing such thoughtful and insightful feedback on presentations and papers, to Matteo for the support and company through classes and thesis push, to Dip Joti for encouraging me to join the group, to Emma for always making me feel welcomed, and to John for building a great community.

To Kara, Marissa, Rinske, and Dorothy, thank you for handling all of the purchases, travel, and admin logistics over the years. Especially thank you to Kara for always abstracting away the bureaucratic overhead and making our life as a student so much easier.

To all of the students in the Center for Bits and Atoms: Alfonso Parra Rubio, Eyal Perry, Jake Read, Alan Han, Zach Fredin, Erik Strand, Jack Forman, Jiaming Lui, Dimitar Dimitrov,

Kat Kormilitsyna, Miana Smith, Quentin Bolsee, Sara Fernandez, Amira Adbel-Rahman, David Preiss - thank you for sharing the unique experience that is pursuing a degree in CBA. It's been an inspiration to learn from you all. Special call out for Zach, Alfonso, Eyal, and Jack for kicking off this journey together as the 2019 incoming cohort; and to Erik and Amira for the fruitful research discussions and for being the bits-leaning fourth floor office mates over the years.

Thank you to the broader Media Lab community, especially those in the golden age of studcom - Samantha, Alfonso, Eyal, Rob, Naana, Tobin, Hope, Valdemar - thank you for all of the effort that you've put into making the Media Lab a fun and exciting place to be. And of course the larger 99Fridays crew - party prep, planing, and building were some of the highlights of my time at the lab.

And of course, I must thank all of the amazing friends and community that I've found in Camberville. This is a really special place full of brilliant, playful, and inspiring people that I've been so lucky to feel supported by. Specifically, to Alfonso, Jurgis, Christina, Eve, Jan, and Eyal for the establishment of Joseph and Sons - it's been a special type of friendship how much we've all shown up for one another over the years, always going above and beyond, committed to the bit. Christina, thank you for being there through the thick and thin, truly picking me up whenever I'm down. Eve, thanks for being my confidant through all of the adventures and stresses. Jurgis, thank you for literally pulling me across this dissertation finish line, and for capturing so much of our wonderful lives on film. Ariel, thank you for the endless support and positivity, for building intentional community together, and for inspiring me to take risks.

To Dani, thank you for the unconditional support through the end of this journey, for the endless pep-talks, home cooked meals, and patience. You've made me feel safe and loved in more ways than you know, thank you for being a recharging home base to fight the burnout.

To Alfonso, this is now the fourth time your name comes up, so we obviously did a lot together, but thank you first and foremost for being the best flatmate I could've asked for and building a wonderfully supportive home together. You've been like family through this journey, from the early days of covid to the final push, I've been so lucky to share a house, a lab, and cats with you over these years.

And thanks to the cats, Kelvin and Joules, for the endless cuddles.

And finally, to my family - Mom, Jeff, Evie, Danny, Lauren, and Brian - thank you for always believing in me. Lo, thank you for all the support and encouragement. Brian, for being Boston buddies through the majority of this PhD, it was always comforting to know there's family close by. Danny, thanks for always being ready to nerd out with me on whatever new topic I was learning, and for just getting it, in the life sense. Evie, thank you for being there to give me the reality check pep talk I always needed when times were tough, I'm so grateful to have a sister like you. Jeff, thanks for always looking out for me and helping make this path possible by setting me up to dream big. Mom, thanks for raising me by example to be a strong, confident, and determined woman - you inspire me all the

time and I couldn't have done this without you.

And of course, to the family members who passed away before this journey started, but were nonetheless critical to the pursuit of this degree. To my grandpa, who supported me more than he could have known in these years, thank you. And to my grandma, to whom this dissertation is dedicated, thank you for inspiring a curiosity for science and learning, for being such an interesting lady, and for believing in me always. To my dad, who would have loved the media lab, thanks for paving the road as a curiosity-driven visionary, in so many ways this journey made me feel closer to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Modern compute is hitting a power wall. The rapid scaling of AI has reached a point where the question is *"how do we deliver enough power to the chips?"* rather than *"how do we build more chips?"* [1]. The 2024 LBNL Data Center Energy Usage Report estimates that U.S. data centers consumed 176 TWh in 2023, amounting to 4.4% of total U.S. electricity consumption, and projects this figure to grow to between 325 and 580 TWh by 2028 [2]. For perspective, a typical 1 GW nuclear reactor operating at a 90% capacity factor generates approximately 8 TWh per year, so meeting even the lower bound of the 2028 projection would require the output of roughly 40 dedicated reactors. This is not an arbitrary comparison: industry hyperscalers are actively contracting nuclear capacity for their data centers. Google, for example, recently signed an agreement with Kairos Power to deploy a fleet of small modular reactors totaling 500 MW by 2035, dedicated to Google's datacenter load [3]. At an even larger capital extreme, there is growing interest in sending data centers to space to access continuous solar power [4],[5]. As AI inference scales and foundation models grow, the bottleneck is increasingly the ability to deliver and dissipate power in dense, high-throughput networks of chips. The work in this thesis is motivated by the prospect of a new computing hardware stack that can deliver orders of magnitude more compute here on earth with today's power grid.

The fundamental energy dissipation required to perform a single bit erasure is $k_B T \ln(2)$, known as Landauer's limit [6]. At room temperature, this limit is 2.8×10^{-21} J, or 2.8 zJ. Meanwhile, advanced node CMOS transistors dissipate on the order of 10^{-16} J (0.1 fJ) per switching event, roughly $10^5\times$ above the theoretical limit. Concurrently, Moore's law is approaching it's end, meaning the rate at which this switching energy is decreasing with fabrication and material advances is slowing, and projected to stop decreasing with future

nodes [7]. In contrast, the Adiabatic Quantum Flux Parametron (AQFP) is a superconducting digital logic device capable of switching at energies near this fundamental thermodynamic limit. At 4.2 K and a 5 GHz clock frequency, the switching energy per Josephson junction has been measured at approximately 1.4×10^{-21} J, equivalent to $24\, k_B T$ [8]. Even accounting for a $\sim$1000 W/W cryogenic cooling overhead to maintain superconductivity at 4 K, this represents roughly two orders of magnitude lower energy dissipation than the $\sim 10^{-16}$ J switching energy of modern CMOS transistors. This device-level energy efficiency is what motivates scaling AQFP toward full computing systems. With AQFP, the goal is not a marginal transistor improvement, but rather operation near the physical limits of computation itself.

However, superconducting electronics have promised high performance and low energy for decades, and large-scale systems have repeatedly fallen short of delivering on that promise. Despite differences in how they encode and propagate logic, superconducting logics families converge on a common set of obstacles: the absence of dense cryogenic memory, biasing and clocking networks whose overhead consumes the device-level energy advantage, and practical issues such as flux trapping and fabrication yield [9],[10],[11]. For AQFP logic in particular, multiphase excitation clocks force gate-level pipelining, which leads to long latencies in large-scale architectures [12]. Additionally, the AQFP devices have low drive strength, which constrains interconnect length and fanout and necessitates the insertion of splitters and buffers that further compound timing delay [13]. Compounding the device-level challenges, the field also lacks the high-level system design exploration tools that are foundational in CMOS VLSI, where architectural trade-offs are evaluated long before fabrication. Without an analogous workflow for AQFP, architectural choices remain difficult to evaluate beyond isolated demonstrations, and many of the most promising AQFP designs to date have inherited their architectural assumptions from CMOS rather than designing for the natural strengths and weaknesses of the superconducting substrate. Therefore, this work is motivated by the question: *how do we design computing systems that exploit AQFP device physics to preserve as much device-level efficiency as possible at the system scale?*

1.2 Historical Context

There are three pillars of historical context that this work can be viewed from: superconducting electronics, theoretical physics thought experiments on thermodynamics and information theory, and early computing in Japan. The next three sections walk through each of these intertwined histories.

1.2.1 Superconducting Electronics

Researchers have explored superconducting logic devices for high-speed energy-efficient computation since the 1950s when the cryotron device was invented by Dudley Buck and explored as an alternative to vacuum tubes [14],[15]. It is worth emphasizing up front that "superconducting logic" is not a single technology. Many distinct logic families have been developed over the decades, each with different trade-offs in speed, energy dissipation, and circuit scalability.

The first sustained attempt at a superconducting computer came from IBM. Beginning in 1964 and running until its cancellation in September 1983, IBM's Josephson program built voltage-state logic from lead-alloy Josephson junctions, where the junction switch between superconducting and resistive states encoded the logical 1 and 0 [16],[17]. The program was ambitious, well-funded, and ultimately unsuccessful, costing roughly a quarter billion dollars in funding before being shut down. At the device level, problems arose because the lead-alloy junctions were unreliable under thermal cycling and the resistive state logic dissipated enough energy that the cryogenic cooling overhead substantially eroded the efficiency advantage IBM had originally projected. Additionally, the circuits suffered from "punch through" failures where the junctions failed to reset between clock cycles compounding in bit error rate as systems got more complex [17]. But the program's deepest difficulty was timing rather than physics. IBM was building superconducting computers in the same window during which silicon CMOS was entering its most dramatic period of scaling, and the moving target proved impossible to catch. By the time Bell Labs introduced reliable Nb/Al-AlO_x/Nb junctions in 1983 [18], IBM had already canceled the Josephson computer efforts.

However, the Nb junctions did not stay unused. Japanese groups, working under MITI funding through the 1980s, adopted the new fabrication process and seeded a research community that has continued to drive superconducting electronics into the present [19].

Then, in the 1990s and early 2000s, superconducting electronics gained a lot of traction again for the Rapid Single Flux Quantum (RSFQ) logic family. The RSFQ logic family was originally introduced by Likharev in 1985 [20]. RSFQ encodes a bit not as a voltage level, but instead as the presence or absence of a single quantum of magnetic flux, commonly referred to as a fluxon or SFQ[1], propagating through a superconducting circuit. The promise of RSFQ was speed. Switching is gated by Josephson junction dynamics rather than by charging a capacitor, and a 1999 demonstration reached 770 GHz in a T-flip-flop test circuit [21]. That

[1]"SFQ" is used to refer to both the logic family and the fluxon pulse. This can be confusing at times because we can talk about an SFQ pulse in a circuit, but not be referring to the SFQ logic family (which itself is shorthand to refer to all falvors of RSFQ logic like ESFQ, eeSFQ, DSFQ, xSFQ, etc.). For example, AQFP circuits operate with SFQ circulating currents, but they are not within the SFQ logic family.

number became the headline figure for the field, and it is still cited today.

As RSFQ work moved from isolated demonstrations to larger circuits, however, the picture got more complicated. Reliable SFQ pulse propagation requires resistively shunted Josephson junctions to tune the damping characteristics of the junction, and these shunt resistors turn the circuit into a static and dynamic power sink. Once cooling overhead is included, RSFQ per-operation energy is comparable to advanced CMOS [11]. While there are energy efficient variants of RSFQ logic [22],[23], that address the static power loss and minimize dynamic switching costs, they still run into the larger issue of scaling a parallel current power network. SFQ circuits are powered by supplying the junctions with a direct current bias, but current sources split in parallel across a large scale circuit quickly become energetically expensive and cannot scale infinitely because the superconducting wires carring the power supply have a critical current at which they will no longer superconduct. Additionally, RSFQ also has a sensitivity to ambient magnetic field. Stray fluxons trapped in the circuit during cooldown can perturb the state of nearby loops and produce bit errors, while this sensitivity grows with circuit size as more area becomes available for trapping [24].

More recently, the adiabatic quantum flux parametron (AQFP) has gained traction within the superconducting electronics community for its extreme energy efficiency. The AQFP descends from a lineage of parametron computing (covered in section 1.2.3), but its modern adiabatic form was first demonstrated by Takeuchi et al. in 2013 [25]. It attracted wider attention in 2021, when Ayala et al. demonstrated MANA, a 4-bit hybrid RISC and dataflow microprocessor built entirely in AQFP logic and operating at 1.4 zJ per junction [12]. As discussed earlier, the AQFP switching energy sits near the thermodynamic limit of energy required for bit erasure, which is generally two orders of magnitude lower than equivalent switching energies in RSFQ. AQFP is the first superconducting logic family for which the cost of cryogenic operation is convincingly outweighed by the energy savings at the device level.

However, AQFP also has it's share of difficulties. Cascading logic in AQFP circuits depend on a mutual inductance transformer at the device's output, as will be discussed in Chapter 2. But the consequence is that the footprint of an AQFP device is large, $\mathcal{O}(100)$ μm^2, due to the geometric inductance required for coupling the output transformer. Additionally, its very low switching energy means the current it generates is also small, so it cannot drive long interconnects, which is an issue for scaling to larger circuits [13]. AQFP also requires a multi-phase AC excitation that doubles as both clock and power supply [26]. This multi-phase clocking network makes a large-scale AQFP circuit sensitive to clock skew and confines computation to fine-grained pipelining at the gate level, leading to high latency in complex circuit design.

So if superconducting electronics has been investigated for energy-efficient computing for nearly seventy years, the obvious question is: why now? The short answer is that several converging trends have changed the calculation. Energy is now the primary constraint on computing systems in a way it simply was not in the case the IBM era. AI workloads in particular are pushing data center power demand into a regime where the cooling overhead of cryogenic operation is no longer prohibitive relative to the energy saved. This is happening at the same time in which CMOS scaling is slowing to the point where per-transistor switching energy is no longer decreasing significantly between transistor fabrication nodes. Meanwhile, superconducting fabrication has matured. MIT Lincoln Laboratory's SFQ5ee process and successors offer high-density, high-yield niobium fabrication that resolves the materials issues that hampered earlier programs [27],[28]. Additionally, cryogenic infrastructure, driven by a decade of investment in superconducting quantum computing, is now accessible at price points and form factors that classical superconducting electronics could not have justified on their own.

1.2.2 Fundamental physical limits of compute

There's a rich history of physics theory that proves information is fundamentally physical. It spans about a century of research, beginning with a thought experiment that physicists initially treated as a paradox, and ending with the modern understanding that information is a thermodynamic quantity with its own accounting. The result, the Landauer limit, sets the irreducible energy cost per erased bit, and it is the standard against which all computing technologies are ultimately measured.

Starting in December 1867 with a letter from James Clerk Maxwell to Peter Guthrie Tait, in which Maxwell sketched what would later be called Maxwell's demon [29]. Maxwell imagined a chamber of gas in equilibrium, divided in two by a partition with a small door. A demon at the door observes each approaching molecule and opens or closes the gate to let fast molecules through in one direction and slow ones in the other. Over time, one side of the chamber heats up and the other cools down, producing a temperature gradient out of a gas that began in equilibrium. A gradient sustained without work apparently violates the second law of thermodynamics, and Maxwell's demon thus posed a paradox: if information about the molecules is enough to undo equilibrium, then the second law must depend on assumptions about what observers can do, not just on the underlying mechanics.

Leo Szilard resolved the paradox in 1929 by decomposing the demon to its simplest possible form in his "Szilard engine" thought experiment [30],[31]. In the Szilard engine, a single gas molecule is place in a chamber. The demon observes which half of the chamber the

molecule is in, inserts a partition, attaches a piston to the side with the gas particle, and lets the gas isothermally expand, therefore extracting thermodynamic work. Since the chamber is split into two sides, there are two possible states that the demon can decide the particle is in, i.e. two bits of information. Szilard determined that the work extracted by this decision is $k_B T \ln 2$. It's worth noting that this value originates with Szilard, not Landauer, despite the bound carrying Landauer's name. Szilard preserved the second law by arguing that the demon must dissipate at least $k_B T \ln 2$ per measurement to acquire the information in the first place.

Rolf Landauer corrected the location of the cost in 1961 [6]. Measurement and storage, Landauer argued, are in principle free; what costs $k_B T \ln 2$ is the *erasure* of a bit. Any logically irreversible operation maps multiple input states to a single output state and therefore reduces the entropy of the computing system, and that entropy reduction must be released to the bath as heat. The demon does not pay for looking; it pays at the end of the cycle when it resets its memory to be ready for the next observation. The Landauer limit applies to every physical implementation of irreversible computation and is the relevant lower bound for any logic family that overwrites its working state.

A brief tangent must be made to point out that a computation that never erases anything has, in principle, no fundamental thermodynamic cost. This is the foundation of reversible computing, formalized by Charles Bennett in 1973 [32]. A reversible machine accumulates intermediate results rather than overwriting them, and any computation it performs can in principle be unwound, such that the Hamiltonian is reversible and the system is conservative. The catch is memory: a real machine cannot accumulate state indefinitely, and any operation that frees memory pays the Landauer cost at that moment. Reversible computing defers the limit rather than abolishing it, trading dissipation for storage.

The remaining question is whether any physical device can come close to the Landauer limit in practice. Robert Keyes and Landauer addressed this in 1970, in a paper titled *Minimal Energy Dissipation in Logic* [33]. Their hypothetical device is modulated in time between a single-well configuration, where no logical state is encoded, and a double-well configuration, where the system can be in either the left or right well, therefore encoding a binary digit. A small input bias breaks the symmetry of the double well as it forms, so the system settles into the well that encodes the desired logical state. The device dissipates almost no energy because the state coordinate is always at or near a local minimum of its potential. The work of switching comes from transforming the potential landscape itself rather than from accelerating the state particle across a barrier. This is the structural distinction between adiabatic switching, where the state follows a slowly varying potential, and conventional transistor switching, which requires charging and discharging a capacitor to drive the device

coordinate out of equilibrium and over potential barriers all while dissipating energy.

The Keyes–Landauer hypothetical device is dynamically almost identical to the AQFP. Keyes and Landauer even cited Eiichi Goto's Esaki-diode parametron as a candidate physical implementation, which was an early inspiration for the eventual superconducting quantum flux parametron as discussed in the next section.

For most of the century in which these arguments were developed, the limits were theoretical. They have since been measured. Bérut et al. provided the first direct experimental verification of Landauer's principle in 2012, using a single colloidal particle in a modulated optical double-well trap as a one-bit memory and observing the dissipated heat per erasure saturate at $k_B T \ln 2$ in the slow-cycle limit [34]. Koski et al. realized Szilard's engine experimentally in 2014, using a single-electron box at sub-Kelvin temperatures to extract approximately $k_B T \ln 2$ of work per bit of information [35]. Therefore, this limit is real and reachable in carefully constructed experiments and the remaining question is how close to it can a practical and scalable logic family be made to operate.

1.2.3 The Parametron

It was an oversimplification to say that the AQFP was first established in 2013. The 2013 work introduced the modern adiabatic form, but the underlying device concept has a much longer history that threads together Japanese parametron computing of the 1950s, von Neumann's late patent work, Likharev's parametric quantron in the 1970s, and the original quantum flux parametron in the 1980s.

The parametron was first conceived of as a room temperature digital computing element in 1954 by Eiichi Goto [36]. The original parametron is an LC circuit with a nonlinear reactive element (Goto used a ferrite-core inductor) excited at twice its resonance frequency. Under parametric pumping, the circuit settles into a subharmonic oscillation at the resonance frequency, and the phase of that oscillation can lock to either of two stable values π radians apart. The two stable phases encode the logical 1 and 0, with logic operations are performed by phase synchronization between coupled parametrons. Goto's parametron offered substantial advantages over vacuum tubes for the postwar Japanese computing environment: it was cheap, made from passive components, ran on lower power, and was substantially more reliable than vacuum tubes or the unstable early point-contact transistors. Most of Japan's first generation of digital computers were built from parametrons. The MUSASINO-1, completed at NTT's Electrical Communication Laboratories in March 1957, was the first parametron computer, using approximately 5,400 parametron elements [37]. Parametron computers remained in commercial production through the early 1960s before being displaced by transistor logic.

Von Neumann was working on a very similar idea in the same time period. In April 1954, the same year Goto announced the parametron, von Neumann filed a patent titled *Non-linear Capacitance or Inductance Switching, Amplifying, and Memory Organs* with IBM, granted posthumously in December 1957 [38]. The patent describes logic implemented using parametrically pumped nonlinear reactive elements, with majority-gate logic built from coupled subharmonic oscillators, structurally equivalent to Goto's parametron. Von Neumann's motivation is speculated to have been speed: parametrons could be driven at frequencies well above what vacuum tubes could support, and he was looking for ways to build computers faster than the von-Neumann architecture machines he had been designing with vacuum tube components [39]. The patent was rendered largely irrelevant by the rapid growth of silicon transistors at Bell Labs over the next several years, and the parametron fell out of favor in both Japan and the United States by the mid-1960s.

However, Goto did not abandon the device. He continued exploring parametron variants and, in 1960, proposed a parametron based on the Esaki tunnel diode rather than a ferrite core, exploiting the diode's negative differential resistance region to achieve the nonlinear parametric response in a more compact form with a direct current output, rather than a phase offset [40][2]. This Esaki diode parametron is the device Keyes and Landauer cited in 1970 as a candidate physical realization of their minimum-dissipation logic [33], discussed in the previous section. Goto's group later moved to superconducting implementations. In 1986, Goto and Loe described the DC flux parametron, the first superconducting parametron based on Josephson junctions [41], which was renamed the quantum flux parametron (QFP) and developed more fully by Hosoya, Goto, and colleagues in 1991 [42]. In the QFP, the parametric drive is implemented by an inductively coupled AC magnetic flux that modulates the critical current of the Josephson junctions, transitioning the device's potential energy between a single-well and a double-well shape, with the logical state encoded in the polarity of circulating current in the loop.

A parallel line of development ran through Konstantin Likharev's group in Moscow. In 1977, Likharev proposed the parametric quantron: an rf-SQUID whose Josephson junction critical current is modulated by an external flux so that the device's potential energy transitions adiabatically between single-well and double-well shapes [43]. The parametric quantron and the QFP are closely related devices, differing in their specific circuit topology and biasing, but both implement the Keyes and Landauer minimum dissipation dynamics in a superconducting circuit. Likharev followed in 1982 with an extended analysis tying

[2]Interestingly, after re-reading this paper for this citation, one of the ways that Eiichi proposed implementing memory in the Esaki diode circuits was with "serial delay-line memory", a similar conclusion that my research discusses in Chapter 4.

the parametric quantron explicitly to the Bennett reversible computing framework, showing that the device could approach the Landauer limit in principle [43]. Likharev's parametric quantron never became a working logic family on its own, but the underlying device concept was preserved and developed in Japan through Goto's QFP work.

The QFP itself, in its 1980s form, was not adiabatic. The Josephson junctions available in that era had relatively low critical current densities and the QFP was designed with shunt resistors, which forced the device to operate in a parameter regime where the single-well to double-well transition was not smooth. The system had to hop over a small residual barrier between the two configurations rather than transition continuously, and the resulting non-adiabatic switching dissipated energy well above the Landauer floor [25],[26]. By the 1990s the program supporting much of the work around superconducting circuits in Japan came to a close and development on QFP circuits slowed. However, the QFP remained an intriguing device in the literature as a highly sensitive comparator circuit with the potential to approach quantum-limited energy sensitivity [44].

The modern adiabatic form of the QFP, the AQFP, was demonstrated by Takeuchi, Yamanashi, and Yoshikawa in 2013 [25]. The change from QFP to AQFP is not entirely topological but rather based on the fabrication and device parameters. By using Josephson junctions with higher J_c (critical current density) and a larger McCumber parameter, β_c, by removing the shunt resistor, the device is able to operate in a regime where the transition from single-well-to-double-well is smooth, without any residual barrier of before. The AQFP is the device Keyes and Landauer proposed in 1970, made possible by fifty years of accumulated progress in superconducting fabrication. The device physics is developed in more detail in Chapter 2.

1.3 Thesis Goals

The goal of this thesis is to provide a roadmap for advancing AQFP from circuit-level demonstration towards scalable, energy-efficient computing systems. To that end, the chapters that follow contribute circuit primitives, memory, microarchitecture designs, and a full-stack modeling framework that together have the potential to close several specific gaps between current AQFP and system-level integration. The rest of this dissertation is organized as follows:

Chapter 2 develops the device physics, circuit construction techniques, and design conventions used throughout the dissertation. It derives the AQFP equations of motion from a Lagrangian treatment of the underlying superconducting circuit, describes how the multi-phase clock enforces directional data propagation, and introduces the cell library used in the

subsequent chapters. The material is established background rather than original research, but is required context for the chapters that follow.

Chapter 3 develops the phase synchronizer circuit [45]. The chapter begins by introducing the weak constant, a sub-cell modification of the AQFP buffer, and then constructs the synchronizer circuit, which allows data arriving on any clock phase to be propagated to a known phase, rather than erased. The weak constant is measured on fabricated test chips, and the phase synchronizer is simulated. The chapter closes with an analysis of the remaining circuit-level work required to demonstrate the full phase-synchronizer circuit.

Chapter 4 develops the SR-Loop memory architecture. The chapter motivates the use of circulating storage for AQFP, demonstrates a serial SR-Loop storage cell, and then composes the cell into a register file. Two iterations of the register file design are discussed. The v1 register file organizes the storage cells in a two-dimensional array which incurs significant routing overhead cost. The v2 design optimizes the layout by implementing a bit-slicing organization and modifying the topology of the storage loops.

Chapter 5 develops the LinCore microarchitecture. The chapter motivates a closed-loop execution-storage pattern that hides the inherent latency of AQFP circuits. The chapter describes the LinCore scheduling and microarchitectural block diagram, then analyzes a first attempt at fabricating the LinCore accelerator.

Chapter 6 develops the AccelForge framework for superconducting accelerator design space exploration. The chapter motivates the modeling approach, presents the component models for AQFP and RQL compute, cryogenic memory, and temperature-stage interconnect. Then the framework is applied to a sequence of design studies that build progressively from a bare processing element array through an on-chip global buffer architecture to a full TPU-like organization with realistic interconnect models.

Chapter 7 summarizes the contributions of the dissertation and briefly discusses a far-forward looking design path toward a cryogenically cooled compute system at the datacenter scale.

Chapter 2

Background

The contributions of this thesis rest on two characteristics of the AQFP device: (i) its switching energy can approach the Landauer limit, and (ii) it requires a multi-phase clock to enforce directional data flow when propagating logic. This chapter reviews the device physics behind these facts. We start from superconductivity and Josephson junction dynamics, derive the AQFP equations of motion and energy landscape from a Lagrangian treatment of the circuit, and then describe how digital logic primitives are constructed from the resulting device and introduce the cell library design used throughout this thesis work. The material throughout this chapter is not novel research, as it draws on the pedagogical treatment of the AQFP tutorial review from Takeuchi et al. [26], as well as standard references on superconducting circuits.

2.1 Superconductivity

Superconductivity is a quantum mechanical property exhibited by some materials below a critical temperature, T_c, which is specific to the material itself. It is defined by two properties: (i) perfect conductivity, which means zero electrical resistance when carrying a DC current; and (ii) perfect diamagnetism, exhibited by a superconductor's expulsion of magnetic field when transitioning below T_c, commonly referred to as the Meissner effect [46]. Below T_c, the free electrons in a superconductor condense into Cooper pairs, which are a bosonic quasiparticle that can settle to a single ground state and, therefore, traverse the material lattice without scattering [47]. This means that a DC current established in a closed superconducting loop will, in principle[1], circulate indefinitely. Notably, zero resistance is a strictly DC phenomenon. At finite frequency, unpaired quasiparticles will respond to the

[1] and, arguably, in practice - the measured lower bound for the supercurrent to decay is 10^5 years [46]

time-varying field and dissipate energy.

Superconductivity has been demonstrated in various materials with transition temperatures ranging from milliKelvin to hundreds of Kelvin. The mechanism for superconductivity is well understood for materials with a T_c below 10 K, commonly referred to as low temperature superconductors (LTS). On the other hand, the superconducting properties of high temperature superconductors (HTS) are less understood and the materials tend to be difficult to fabricate useful thin-film devices from. Therefore, this thesis focuses on low-temperature superconductors. In particular, we will discuss integrated circuits fabricated in niobium (Nb), which has a T_c of approximately 9.2 K. The circuits are then operated at temperatures near 4 K, well below T_c, where the quasiparticle density is exponentially suppressed and the dominant circuit physics is set by the superconducting condensate.

2.2 Superconducting circuits

A circuit, in its most abstract form, is a graph. The edges are **branches**, each defining a circuit element with its branch variables (typically a current i_b and a voltage v_b); and the vertices are **nodes**, the points at which circuit elements meet. The nodes impose topological constraints through Kirchhoff's laws, while the branches describe the device physics through their constitutive relations. Together they are enough to design, analyze, and ultimately fabricate a circuit that does something useful. This is the standard picture taught in an introductory electronics course [48]. What follows is concerned with where that picture needs to be modified or extended for superconducting electronics.

2.2.1 Flux quantization and the superconducting phase

Beyond the phenomenology of zero resistance and the Meissner effect, a defining feature of superconductivity is the existence of a many-particle condensate wavefunction that maintains phase coherence across the superconductor [46]. The condensate, made up of Cooper pairs, can be described by a single complex order parameter

$$\psi(\mathbf{r}) = |\psi(\mathbf{r})|\, e^{i\varphi(\mathbf{r})} \tag{2.1}$$

whose amplitude $|\psi(\mathbf{r})|$ is related to the local Cooper pair density and whose phase $\varphi(\mathbf{r})$ is well-defined over macroscopic length scales. The macroscopic nature of the phase has a direct consequence whenever the superconductor forms a loop. While traversing the superconducting loop, $\psi(\mathbf{r})$ must return to the value at which it started, so the accumulated change in phase must be an integer multiple of 2π. This condition is the same argument that quantizes orbital

angular momentum in an atom, and applied to a superconducting ring, it forces the fluxoid threading the loop to take only discrete values[2],

$$\Phi = n\Phi_0, \qquad \Phi_0 \equiv \frac{h}{2e} \approx 2.07 \times 10^{-15}\ \text{V}\cdot\text{s} \tag{2.2}$$

where n is an integer, h is the Planck constant, and the factor of $2e$ in the denominator refers to the charge of the Cooper pair [49]. Therefore, we define a quantity called the flux quantum, Φ_0, which is the smallest indivisible packet of fluxoid that can enter or leave a superconducting loop. The flux quantum is the basic information carrier in flux-based logic families in superconducting electronics.

Magnetic flux is a quantity in ordinary non-superconducting circuits, defined as the time integral of the voltage across a branch, $\phi_b(t) = \int_{-\infty}^{t} v_b(t')\,dt'$. Flux is to voltage, what charge is to current. Although, it's less commonly referenced in semiconducting digital circuits because voltages are easy to measure and reason about directly when there are resistive elements. On the other hand, in a superconducting circuit, voltage is a poor primary variable. A bare superconducting wire develops no voltage drop in steady state, regardless of the current it carries, so treating voltage as the independent branch variable obscures the relevant physics. It is more natural to work with branch flux, ϕ_b, when discussing superconducting circuit components. In this formalism, constitutive relations are written accordingly: an inductor obeys $\phi_b = L\,i_b$; a capacitor obeys $i_b = C\,\ddot{\phi}_b$; a resistor obeys $\dot{\phi}_b = R\,i_b$; etc.

Branch flux has the further advantage that it naturally normalizes by the flux quantum, therefore defining the branch phase

$$\varphi_b \equiv \frac{2\pi}{\Phi_0}\,\phi_b \tag{2.3}$$

This is the same phase that appears in the condensate wavefunction, but at the circuit level it is most usefully interpreted as as a normalized flux. Branch phase, φ_b, is the accumulated flux, ϕ_b, across a branch expressed as a fraction of Φ_0, with $\varphi_b = 2\pi$ corresponding to one flux quantum.

2.2.2 The Josephson junction

The Josephson junction (JJ) is the basic active device of most superconducting electronics. A JJ is formed when two superconductors are weakly coupled such that they share the same order parameter but can be offset in phase [49]. In superconducting integrated circuits, the

[2]the fluxoid is a quantity that describes the flux contribution from the magnetic flux and the kinetic inductance ($\Lambda \int J_s \cdot d\ell$), but in our circuits of interest made from low kinetic inductance materials, the magnetic flux dominates and fluxoid $\approx$ flux.

weak link typically takes the form of a thin oxide tunnel barrier between two superconducting electrodes, forming a superconductor–insulator–superconductor (SIS) junction.

In 1962, Brian Josephson predicted that the supercurrent and voltage on this weak link are governed entirely by the gauge-invariant phase difference, $\varphi_j = \varphi_2 - \varphi_1$, between the two superconductors, defining the two Josephson relations [50]

$$I = I_c \sin \varphi_j \tag{2.4}$$

$$\frac{d\varphi_j}{dt} = \frac{2eV}{\hbar} \tag{2.5}$$

Equation (2.4) is known as the DC Josephson effect: at zero voltage, the junction sustains a dissipationless supercurrent up to a maximum value I_c, the critical current, set entirely by the phase. Equation (2.5) is the AC Josephson effect: under a finite voltage, the phase winds in time and the supercurrent oscillates at the Josephson frequency $f_J = 2eV/h$ [49].

At the circuit level, equations (2.4) and (2.5) together describe a nonlinear, phase-dependent inductive circuit element. Differentiating (2.4) and combining with (2.5) gives $V = L_J\, dI/dt$ with the Josephson inductance

$$L_J = \frac{\Phi_0}{2\pi I_c \cos \varphi} \tag{2.6}$$

and a corresponding stored, non-dissipative coupling energy

$$E_J(\varphi) = \frac{\Phi_0 I_c}{2\pi} (1 - \cos \varphi) \tag{2.7}$$

The ideal JJ is a lossless circuit element, $E_J(\varphi)$ is a conservative potential, and the supercurrent flows without voltage drop.

2.2.2.1 RCSJ model

A real junction departs from the ideal picture in two ways: (i) above I_c the junction develops a voltage and quasiparticle normal current flows, and (ii) the SIS geometry contributes a parasitic capacitance to the device. These non-idealities are captured by the resistively and capacitively shunted junction (RCSJ) model [49]. In this model, the ideal JJ element is placed in parallel with a resistor, R, and a capacitor, C.

Additionally, there is always some small, but nonzero, subgap conductance from quasiparticle tunneling. We model this conductor as a piecewise linear voltage-dependent resistor in the shunt: $R = R_{sg}$ below the superconducting gap voltage, V_g, where tunneling is suppressed, and $R = R_n$ above V_g where quasiparticle tunneling dominates. These parameters are

partially constrained by the superconducting fabrication process. For example, in the MITLL SFQ5ee process $R_{sg} \sim 9R_n$ [27].

Under a current bias, $I_x(t)$, current conservation across the three RCSJ parallel branches gives the equation of motion,

$$I_x(t) = \frac{\Phi_0}{2\pi} C \ddot{\varphi} + \frac{\Phi_0}{2\pi} \frac{1}{R} \dot{\varphi} + I_c \sin \varphi \tag{2.8}$$

We see that the current-driven RCSJ model is a nonlinear, damped, driven oscillator, like a pendulum [49]. Equivalently, the phase moves as a particle in the tilted washboard potential

$$U(\varphi) = -\frac{\Phi_0}{2\pi} I_x \varphi - \frac{\Phi_0 I_c}{2\pi} \cos \varphi \tag{2.9}$$

where the bias current sets the tilt and the wells are the metastable zero voltage states ($\varphi = n2\pi$) of the junction. The small amplitude oscillation frequency in a well is referred to as the Josephson plasma frequency,

$$\omega_p = \sqrt{2\pi I_c / (\Phi_0 C)} \tag{2.10}$$

which describes a characteristic response time of the JJ. For example, in the MITLL SFQ5ee process, an unshunted 50 μA Josephson junction has $\omega_p = 2.08$ THz.

Finally, a dimensionless damping parameter conventionally used in superconducting electronics is the McCumber parameter

$$\beta_c = Q^2 = 2\pi I_c R^2 C / \Phi_0 \tag{2.11}$$

In SFQ-style circuits, an external shunt resistor sets $\beta_c \approx 1$, so junctions are near critical damping and each 2π phase slip produces a single, well-defined voltage pulse. This is the SFQ pulse that encodes logic state [51]. The cost of this regime is that the shunt resistor dissipates the junction's stored coupling energy $E_J \sim I_c \Phi_0$ on every switching event. AQFP circuits avoid this entirely because the junctions are unshunted, $\beta_c \gg 1$, and the JJs never undergo a 2π phase slip. Therefore, the only energy dissipation is from subgap leakage through R_{sg}, with the current set by the speed at which the AQFP is clocked. For example, an AQFP switching at 5 GHz fabricated in the SFQ5ee process ($R_{sg} = 9R_n$), has a switching energy $0.001 I_c \Phi_0$, following $E_{sw} \approx 2 I_c \Phi_0 \frac{\tau_j}{\tau_x}$.

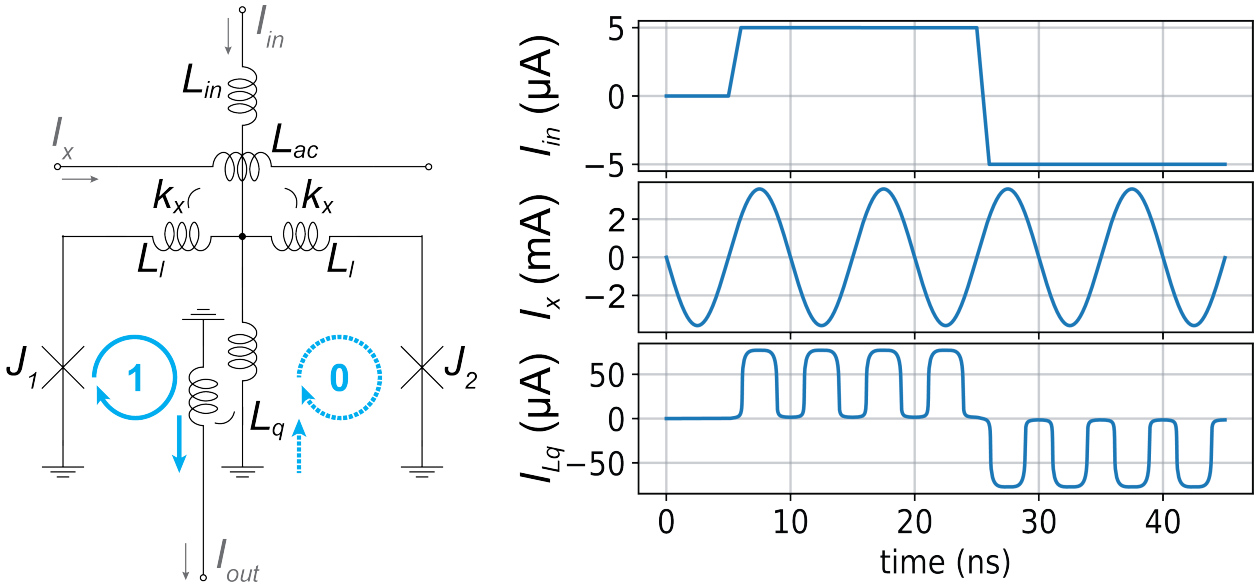


Figure 2.1: Left: schematic of the AQFP. Right: SPICE simulated device operation. Junctions have 50 μA critical currents with a MITLL SFQ5ee process model, $L_q = 10\,\text{pH}$ and $L_l = 1\,\text{pH}$.

2.3 The Adiabatic Quantum Flux Parametron (AQFP)

The AQFP is a Josephson junction based superconducting digital logic device. The topology of the AQFP can be described as a DC SQUID with a shunt inductor through the center, L_q. Figure 2.1 shows a schematic of the AQFP. The parametron has distinct ON and OFF states controlled by an excitation signal, I_x, which serves simultaneously as power supply and clock for AQFP circuits. When I_x induces a phase shift of π across the loops, the device is ON, and a circulating current is induced in either the left or right loop. A small current on the input, I_{in}, while I_x is transitioning ON will favor the current to the left or right loop, and the resulting circulating current through L_q is either positive or negative. The polarity of the current across L_q encodes the logical 1 or 0, which is passed along an output transformer to I_{out} and propagated to the next device.

The right panel of Figure 2.1 shows a SPICE simulation of the device operation. Once I_x crosses the π phase threshold, a small perturbation on I_{in} is amplified into a well-defined current pulse along L_q.

The "adiabatic" qualifier in the device's name is used in the classical dynamics context of an adiabatic invariant on a physical system. To be adiabatic refers to the fact that I_x is ramped slowly compared to the damping rate, $\Gamma \sim \omega_p/Q$. As we noted in the previous section, the Josephson plasma frequency is in the terahertz range, and β_c is around 100 for an unshunted junction, so we can comfortably clock the AQFP circuits around a GHz and maintain adiabaticity.

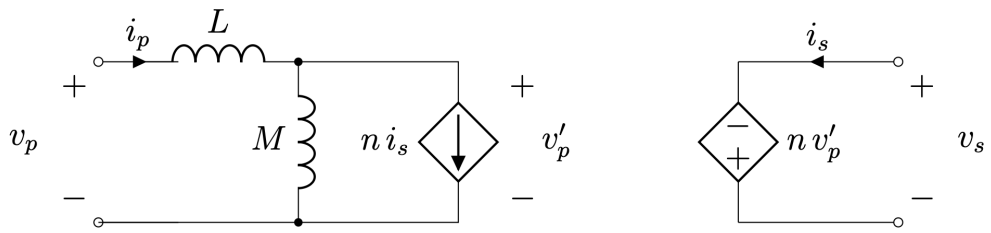


Figure 2.2: Equivalent circuit model for an inductive transformer.

The sections that follow develop this picture more formally. We first work through a circuit analysis of the AQFP to derive its equations of motion [52], from which we then obtain the full device Hamiltonian. The resulting potential energy landscape provides the working intuition for AQFP circuit behavior.

2.3.1 Circuit analysis

The goal of this subsection is to derive the equations of motion for the AQFP circuit in a form that exposes its two independent degrees of freedom.

Before writing down Kirchhoff's laws, we make three simplifications to the circuit of Figure 2.1. First, we remove the output transformer. Output loading does affect the dynamics and energy dissipation of a cascaded device, but for a first order derivation we will ignore this effect. Second, we replace the excitation transformer with its equivalent circuit model. A mutual inductance transformer can be described with a coupled current-controlled current source and voltage-controlled voltage source, as shown in Figure 2.2. The branch fluxes on the two sides of the transformer are related by

$$\begin{pmatrix}\phi_p\\ \phi_s\end{pmatrix} = \begin{pmatrix}L_p & M\\ M & L_s\end{pmatrix}\begin{pmatrix}i_p\\ i_s\end{pmatrix} \tag{2.12}$$

where $M = k\sqrt{L_p L_s}$ and $k \in [0,1]$ is the coupling coefficient of the transformer. In the AQFP, the excitation transformer's primary winding carries I_x and supplies power, while each loop inductor, L_{li}, acts as a secondary winding receiving this power. The constitutive relation for each loop inductor therefore picks up an I_x-dependent contribution,

$$\phi_{li} = L_{li}\, i_{li} + M I_x \tag{2.13}$$

Third, we redraw the schematic with the input source, I_{in}, split across two degenerate loops, one for each side of the device. This re-casting does not alter the topology, but it preserves the vertical symmetry of the AQFP explicitly in the diagram, which simplifies

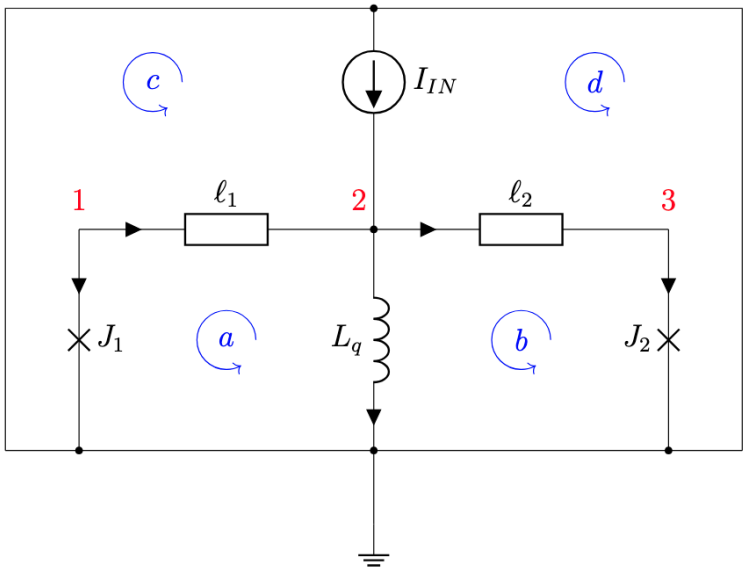


Figure 2.3: Simplified schematic of AQFP circuit labeled for circuit analysis.

the algebra. The simplified circuit, with nodes labelled $1, 2, 3$ and loops labelled a, b, c, d, is shown in Figure 2.3.

Applying Kirchhoff's current law at each of the three nodes gives

$$-i_{j1} - i_{l1} = 0 \tag{2.14a}$$
$$i_{l1} - i_{in} - i_{Lq} - i_{l2} = 0 \tag{2.14b}$$
$$-i_{l2} - i_{j2} = 0 \tag{2.14c}$$

and Kirchhoff's voltage law around each of the four labelled loops gives

$$-\varphi_{Lq} - \varphi_{l1} + \varphi_{j1} = 0 \tag{2.15a}$$
$$\varphi_{Lq} - \varphi_{l2} - \varphi_{j2} = 0 \tag{2.15b}$$
$$-\varphi_{in} + \varphi_{l2} + \varphi_{j2} = 0 \tag{2.15c}$$
$$\varphi_{in} + \varphi_{l1} - \varphi_{j1} = 0 \tag{2.15d}$$

where the last two equations are degenerate by construction (due to the symmetric redrawing

of I_{in}). Combined with the branch constitutive relations:

$$i_{in} = -I_{in} \tag{2.16a}$$

$$i_{ji} = I_c \sin(\varphi_{ji}) + \frac{\Phi_0}{2\pi}\frac{1}{R_i}\dot{\varphi}_{ji} + \frac{\Phi_0}{2\pi}C_i\ddot{\varphi}_{ji} \tag{2.16b}$$

$$\phi_{li} = L_{li}i_{li} + MI_x \tag{2.16c}$$

$$\phi_{Lq} = L_q i_{Lq} \tag{2.16d}$$

these provide a complete system from which the dynamics of the AQFP can be extracted.

We now want to identify the independent degrees of freedom of the circuit. The natural variables are the branch phases across the two junctions, φ_{j1} and φ_{j2}, and the phase across the shunt inductor, φ_{Lq}. The KVL constraints in Eq. 2.15 pin φ_{Lq} to the mean of the two junction fluxes, $\varphi_{Lq} = \frac{1}{2}(\varphi_{j1} + \varphi_{j2})$, so it does not move independently. At this point the same symmetry that motivated the redrawing of the schematic motivates a coordinate change. Just as a differential amplifier is most naturally analyzed in terms of common-mode and differential-mode signals at its two inputs, the AQFP is most naturally analyzed in terms of common-mode and differential-mode combinations of its two junction fluxes:

$$\varphi_+ = \tfrac{1}{2}(\varphi_{j1} + \varphi_{j2}) \tag{2.17}$$

$$\varphi_- = \tfrac{1}{2}(\varphi_{j1} - \varphi_{j2}) \tag{2.18}$$

Substituting these equations into the constraint equations and simplifying yields the AQFP equations of motion:

$$C\ddot{\phi}_+ = \frac{2L_qL_l}{\xi}I_{IN} + \left(\frac{2L_q}{\xi} - \frac{1}{L_l}\right)\phi_+ - 2I_c \sin\varphi_+ \cos\varphi_- - \frac{1}{R}\dot{\phi}_+ \tag{2.19a}$$

$$C\ddot{\phi}_- = \frac{2M}{L_l}I_X - \frac{1}{L_l}\phi_- - 2I_c \sin\varphi_- \cos\varphi_+ - \frac{1}{R}\dot{\phi}_- \tag{2.19b}$$

where $\xi = L_l^2 + 2L_lL_q$ and we have assumed both junctions and both loop inductance transformers are identical. To avoid carrying $\frac{\Phi_0}{2\pi}$ terms, we've expressed some terms in flux rather than phase, assuming Eqn. 2.2 relationship.

The structure of Eqs. 2.19 reveals the AQFP as a system of two coupled nonlinear oscillators. The results are similar to the current-biased RCSJ equation derived in Section 2.2.2.1, but with an additional inductive term and an extra cosine term on the JJ contribution to capture the coupled phases. From this, the role of the bias currents are clear, with I_{in} driving a current for the common-mode junction and I_x driving the differential-mode junction.

2.3.2 AQFP Hamiltonian

Finally arriving at the single-well-to-double-well potential-energy framing for the AQFP, we can derive the potential-energy landscape from the equations of motion, Eqs. 2.19. Applying the Euler-Lagrange formalism [52] to the common-mode and differential-mode equations of motion yields the kinetic and potential energies of the AQFP,

$$T = \frac{1}{2}C(\dot{\phi}_+^2 + \dot{\phi}_-^2) \tag{2.20a}$$

$$V = -\frac{2L_qL_l}{\xi}I_{in}\phi_+ - \frac{2M}{L_l}I_x\phi_- - \frac{1}{2}\left(\frac{2L_q}{\xi} - \frac{1}{L_l}\right)\phi_+^2 + \frac{1}{2}\frac{1}{L_l}\phi_-^2 + 2I_c(1 - \cos\varphi_+\cos\varphi_-) \tag{2.20b}$$

The AQFP is commonly discussed as a 1D bistable system (single-well opening into a double-well) but this is a projection. The true potential $V(\varphi_+, \varphi_-)$ is a 2D surface, and the device's operation is most cleanly understood by reading the structure of that surface directly from Eq. 2.20b. Three contributions build the landscape:

- The $\cos\varphi_+\cos\varphi_-$ term from the two coupled Josephson junctions contributes an egg-carton potential: a periodic array of wells and saddle points spaced by 2π in each phase coordinate.
- The ϕ_+^2 and ϕ_-^2 terms from the inductive energy storage contribute a parabolic bowl that grows large as the phases stray from the origin, confining the system to the region near $\phi_\pm = 0$.
- The linear terms $I_{in}\phi_+$ and $I_x\phi_-$ tilt the surface along the common-mode and differential-mode axes respectively, similar to a current bias tilting the washboard potential of a current-biased RCSJ model.

Figure 2.4 visualizes this 2D landscape as the excitation current is ramped. Panels (a)-(c) show contour maps of $V(\varphi_+, \varphi_-)$ at $I_x = 0$, 1 mA, and 2 mA. As I_x increases, the tilt along the differential-mode axis grows, and the single minimum at $\varphi_+ = \varphi_- = 0$ separates into a pair of degenerate minima displaced along φ_+. This is the bifurcation that underlies AQFP logic operation: with I_x off, the system has one stable state and with I_x on, it has two, all while a small I_{in} tilt along the common-mode axis selects which of the two the system settles into.

To extract a more conventional 1D view, we can project the 2D potential onto either of its two axes by minimizing the other coordinate. To plot the 1D potential along the

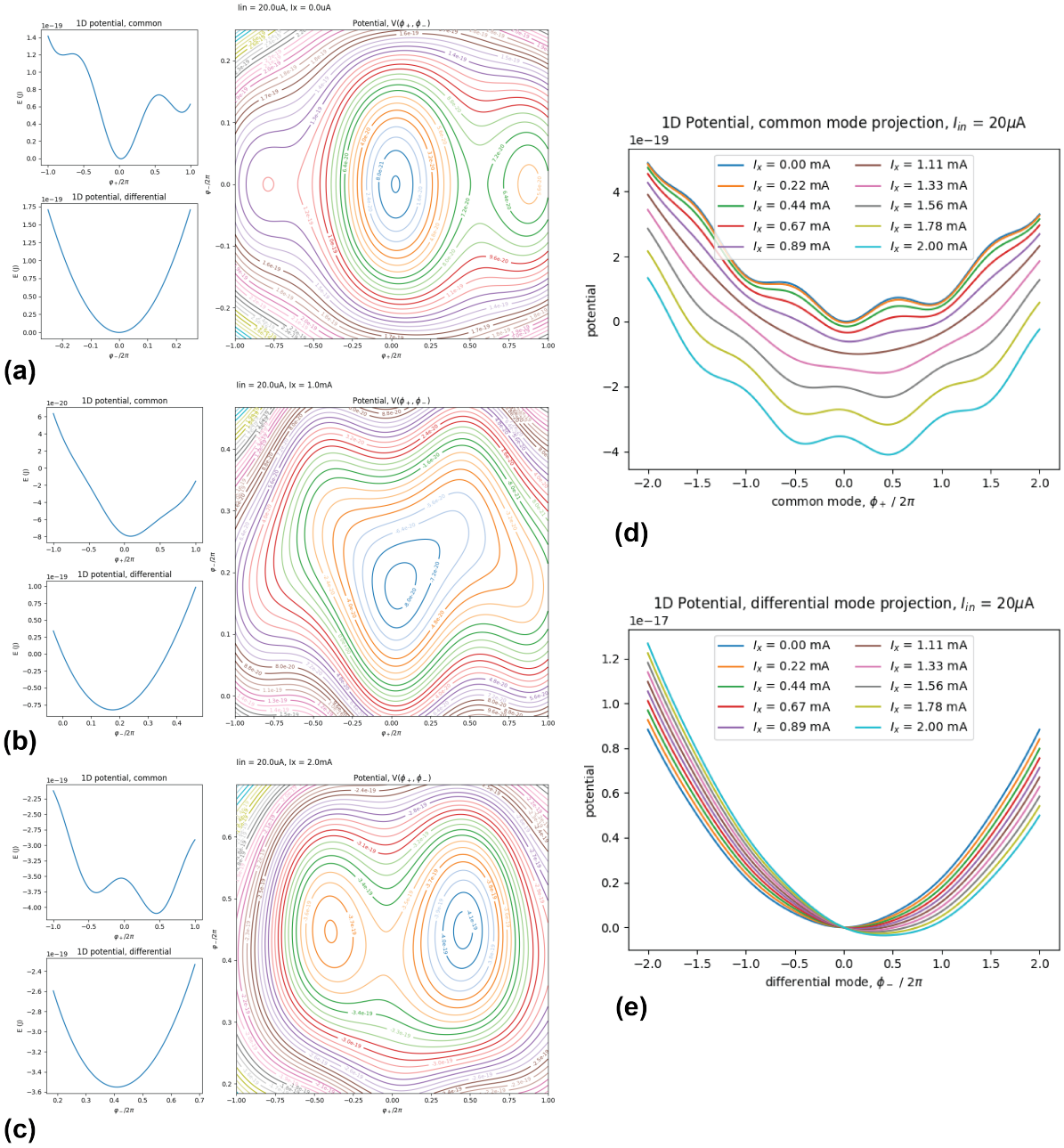


Figure 2.4: AQFP Potential energy contours

common-mode axis, we minimize V with respect to φ_-:

$$\frac{\partial}{\partial \varphi_-} V(\varphi_+, \varphi_-) = -\frac{2M}{L_l} I_x + \frac{\Phi_0}{2\pi} \frac{1}{L_l} \varphi_- + 2I_c \cos\varphi_+ \sin\varphi_- = 0$$

Assuming φ_- remains small near the minimum so $\sin\varphi_- \approx \varphi_-$, we find

$$\varphi_{-,min} = \frac{\frac{2M}{L_l} I_x}{\frac{\Phi_0}{2\pi L_l} + 2I_c \cos\varphi_+} \tag{2.21}$$

Substituting this expression back into Eq. 2.20b gives $V_{1D}(\varphi_+)$, the 1D potential along the common-mode axis with the differential-mode taking on its minimum energy value at each ϕ_+. Repeating the procedure with the roles of φ_+ and φ_- swapped gives

$$\varphi_{+,min} = \frac{\frac{2L_q L_l}{\xi} I_{in}}{\frac{\Phi_0}{2\pi}\left(\frac{1}{L_l} + \frac{2L_q}{\xi}\right) + 2I_c \cos\varphi_-} \tag{2.22}$$

which, substituted back into Eq. 2.20b, yields $V_{1D}(\varphi_-)$.

Figure 2.4(d) and (e) plot $V_{1D}(\varphi_+)$ and $V_{1D}(\varphi_-)$ for a range of I_x. The common-mode projection in (d) is the familiar single-well-to-double-well transition discussed in Section 1.2.3. The new content here is the quadratic decrease in overall energy as I_x grows. This drop records the fact that, as the differential-mode is being tilted by I_x, the system is rolling down the φ_- direction at the same time that it is splitting along φ_+. The shift of the global minimum along both axes is also visible directly in the contour plots (a)-(c) as the range of φ_- over which the double-well minima are located changes with I_x.

2.4 Building digital circuits with AQFP

2.4.1 Majority gate combinational logic

The basic AQFP cell described in above acts as a buffer. When clocked, it reproduces the sign of its input current on the output. Two small modifications to this cell generate the rest of the primitive logic set.

First, reversing the polarity of the output transformer flips the sign of I_{out} relative to the current on L_q, so that a positive input produces a negative output. The resulting cell is an inverter. Second, introducing an asymmetry between the left and right loops biases the potential so that the system preferentially settles into one of the two wells. The result is a hard-coded constant that does not require I_{in}. The asymmetry can be implemented either by sizing one loop inductor, L_l larger than the other or by changing the excitation coupling constant, k_x, on one side. The buffer, inverter, and the two constant devices together form the complete single-input primitive set of the AQFP cell library.

Multi-input gates are then constructed by inductively summing the outputs of several

devices to the input of a downstream device. The downstream device sees $I_{in} = I_1 + I_2 + I_3$ and propagates the sign of that sum to its output. This is equivalent to a three-input majority function MAJ(A, B, C). As shown in Figure 2.5, connecting one of the three inputs to a constant-0 or constant-1 cell reduces the majority to an AND or OR, respectively. Universal logic in AQFP is therefore built on a foundation of buffer, inverter, constant, and majority gates.

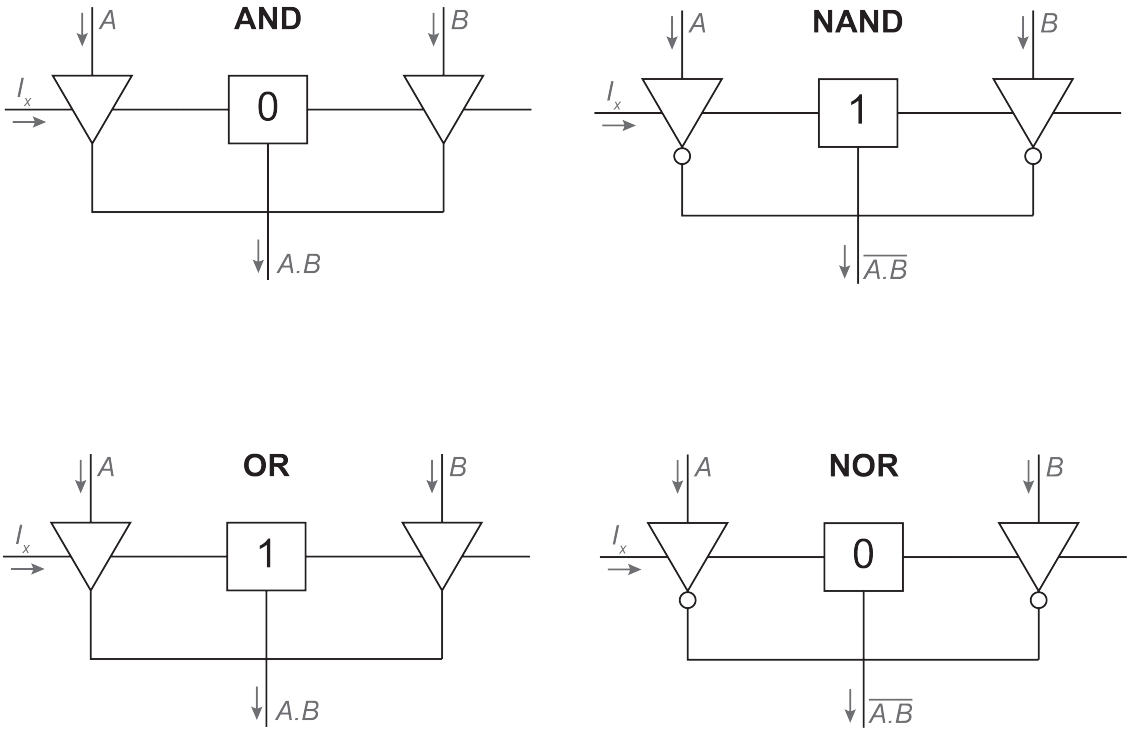


Figure 2.5: Combinational logic gates implemented in AQFP through current summation at the input of a majority cell. AND and OR are obtained by tying one input to a constant 0 or 1, respectively; NAND and NOR modify AND and OR with inverters on each of the data inputs.

2.4.2 Multiphase clock for sequential logic

The AQFP is a two terminal device, with one for data coupling and the other for clock and power. Because the input and output of an AQFP are galvanically connected at the center node, any current along L_q propagates through both the output transformer and "backwards" along the input. Therefore, to confine the data propagation to a definitive direction, a multiphase excitation scheme is needed. A minimum of three phases is required to interleave the receiving, propagating, and blocking states of each cell so that data moves unidirectionally through a pipeline without back-propagation [12],[26].

In practice, a four-phase clock is used for clean isolation between data stages and a lower

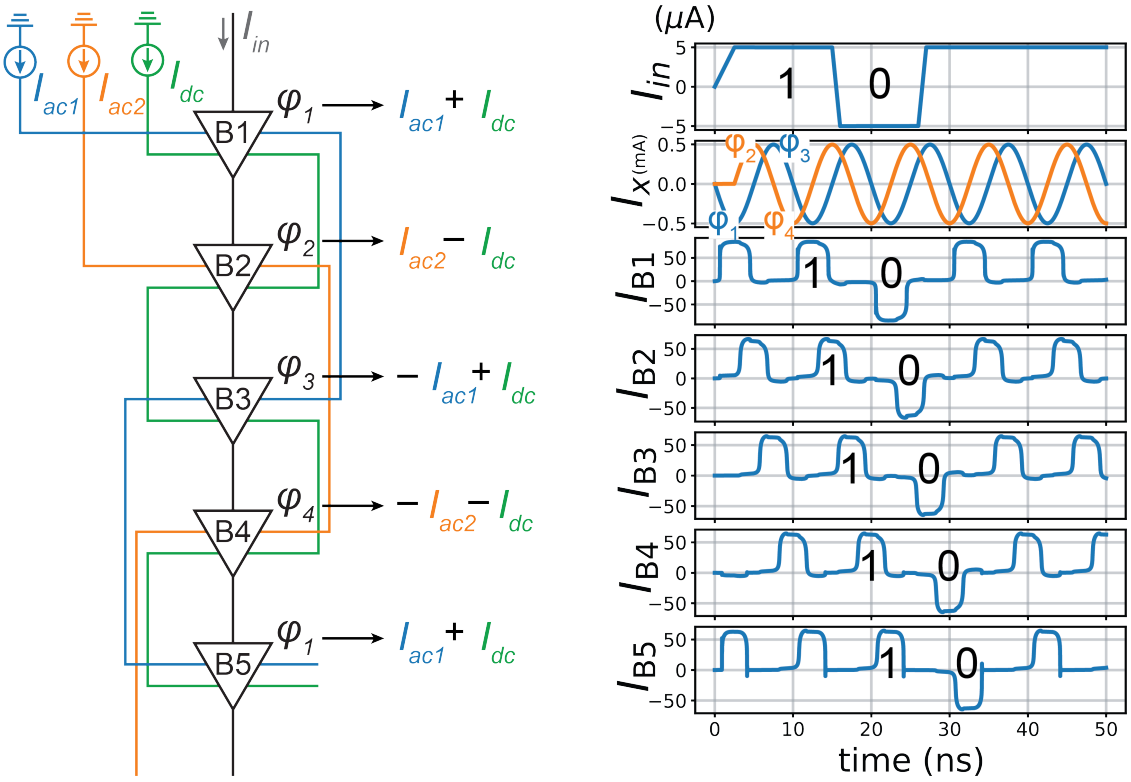


Figure 2.6: Four-phase clocking generated from two AC excitation signals in quadrature summed with a DC offset, meandered around a chain of AQFP buffers. Simulation traces show data propagating one stage per phase along the chain.

AC signal count. The four phases are generated from two AC excitation lines 90 degrees out of phase, combined with a DC bias. The excitation lines meandered around the buffer chain so that adjacent cells see successive phases, as shown in Figure 2.6. Because data can only transfer during the overlap of adjacent active phases, the multiphase clock also defines a precise timing window within which input data must arrive. An AQFP samples its input as the excitation signal ramps up, and any value present on the input line at that moment is amplified to the output. Ensuring that valid data is aligned with the receiving phase of the downstream cell is therefore a central constraint in AQFP circuit design, and it motivates the phase synchronizer introduced in Chapter 3.

2.4.3 MITLL cell library

Throughout this thesis, we use an AQFP cell library developed at MIT Lincoln Laboratory (MITLL) by David Russo and Andrew Wagner. Several AQFP cell libraries have been proposed in the literature, such as: minimalist cell library from Takeuchi et al. [53] with 30 μm $\times$ 40 μm buffer size in AIST STP2 process; or the compact logic cell design from He et

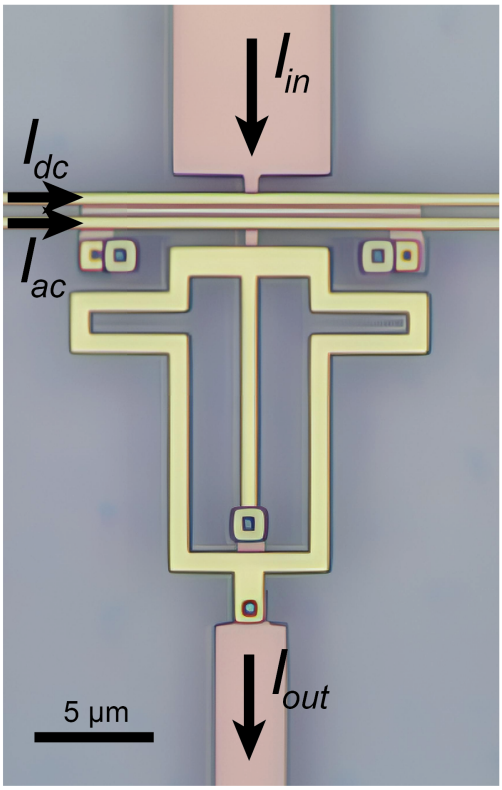


Figure 2.7: Optical micrograph of an AQFP buffer in the MITLL AQFP cell library, fabricated in the MITLL SFQ5ee process. 20 μm × 20 μm footprint

al. [54] with 15μm × 20μm on 8 metal layers in the MITLL SFQ5ee process. The MITLL cells used here provide a competitive footprint on 4 metal layers.

Figure 2.7 shows an optical micrograph of an AQFP buffer fabricated in the MITLL SFQ5ee process [27]. This particular device was fabricated without ground planes to make the underlying structure visible to imaging. The output transformer is laid out as a T on metal layers M5 and M6, and the AC excitation and DC bias lines are coupled across the top of the cell. The full cell footprint is 20 μm × 20 μm, or 400 μm^2.

2.4.4 Simulated energy dissipation

The expected switching energy of an AQFP cell can be extracted from SPICE simulation, as shown in Figure 2.8. We calculate the energy by integrating the instantaneous power across the excitation transformer over multiple excitation cycles and dividing by the number of cycles to get the average energy dissipation per cycle:

$$E_{\text{disp}} = \frac{1}{N}\int_0^{NT} I_{\text{ac1}}\left(V_{\text{out}} - V_{\text{in}}\right) dt \tag{2.23}$$

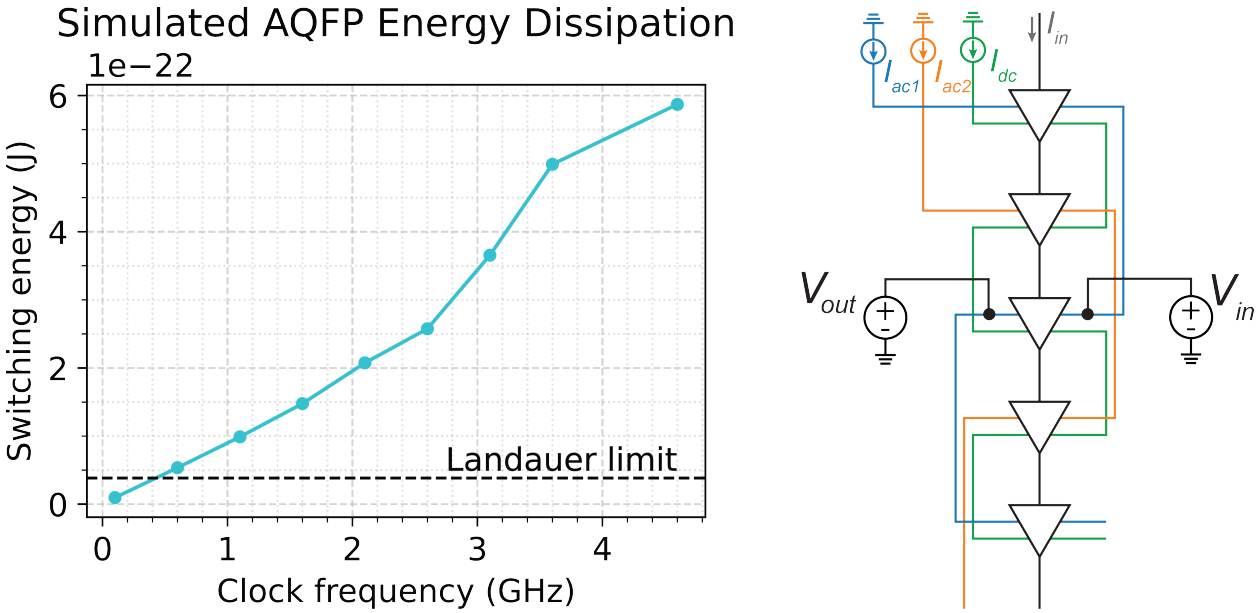


Figure 2.8: Cadence SPICE simulation of the switching energy dissipation for a single AQFP buffer in the MITLL cell library, swept across clock frequency. Calculated as the time integral of the product of voltage drop and current across the AC excitation line, averaged over many clock cycles.

where T is the period of clock, N is the number of cycles averaged over, and the current and voltage labeled match the schematic in Figure 2.8. Because the junctions in AQFP cells are unshunted and operate without 2π phase slips, the only dissipative element in the device itself is the subgap leakage resistance of each JJ. This calculation captures the intrinsic device-level dissipation but does not account for dielectric losses in the excitation lines, which can contribute a significant additional fraction of the total on-chip power budget [55].

Figure 2.8 shows the simulated energy trace for a single AQFP buffer in the MITLL cell library across a range of excitation frequencies. At 5 GHz, the dissipation is 0.58 zJ per switch, in reasonable agreement with the analytical result on the simplified AQFP model from the previous section, as well as with prior reports of simulated AQFP switching energy at $\mathcal{O}(10^{-22})$ J near 5 GHz [26].

Note that the device under test is balanced, with equal buffer loads on the input and the output, which yields the lowest possible switching energy [56],[57]. An unbalanced load would drive additional oscillations as the cell transitions between its single-well and double-well states, incurring slightly higher dissipation.

We separately attempted to measure the AQFP switching energy experimentally using the AC-clock amplitude differential method of [8], but the resolution of the available measurement setup was insufficient to extract an accurate value. The measurement established only an

upper bound of 5.4 fJ per switch of on-chip dissipation. More measurement details are given in Appendix A.

2.5 Prior art

Over the past few decades, AQFP research has progressed from individual device demonstrations to circuit scale prototypes, with a publication cadence that has accelerated as AQFP design flow has matured and interest in ultra-low-energy compute has drawn attention from outside fields. The pedagogical review by Takeuchi et al. [26] gives a comprehensive survey of the field as of 2022. Here we briefly highlight the demonstrations that frame the contributions of this thesis.

The most ambitious AQFP circuit demonstration to date is the MANA microprocessor by Ayala et al. [12], a 4-bit hybrid RISC-dataflow architecture fabricated and measured at 100 kHz, with the execution stage breakout operational up to 2.5 GHz. MANA is the first demonstration of a complete AQFP microprocessor, integrating an instruction decoder, ALU, register file, and execution stage on a single die. It remains the canonical proof point for system-level AQFP feasibility. Other circuit demonstrations include carry look-ahead adder families [58], a serializer/deserializer interface implemented with hybrid AQFP/RSFQ circuits [59], and reversible computing implementations that could potentially outperform the Landauer limit [60],[61]. The broader landscape of superconducting electronics tooling has also expanded, with cell libraries, layout-extraction software, and circuit-level simulators developed under the IARPA SuperTools program and its successors [62],[63],[64],[65].

2.5.1 Contributions

This body of prior work has established that AQFP circuits can reliably be fabricated and scale to meaningful circuit sizes while maintaining its ultra-low-power performance. The chapters that follow operate in the layers above this foundation, addressing the architectural primitives, memory structures, and design abstractions that any future large-scale AQFP system will need. They are organized around four key challenges: data synchronization, on-chip memory, gate-level clocking, and system design exploration. The phase synchronizer of Chapter 3 addresses the absence of a native synchronizer primitive in AQFP. Without a synchronizer, circuit size and multi-chip integration is limited by the excitation clocks which cannot be distributed with phase precision across arbitrarily large interconnects. The SR-Loop register file of Chapter 4 addresses the lack of a compact on-chip memory in AQFP by implementing an AQFP delay line storage element, rather than strictly importing the

random-access memory organization of CMOS into a logic family for which it is poorly suited. The LinCore microarchitecture of Chapter 5 addresses the throughput cost of AQFP constant baseline energy dissipation by using fine-grained multithreading to keep every clocked stage productive on every cycle. Finally, the AccelForge modeling framework of Chapter 6 addresses the absence of a system-level design space exploration tool for superconducting logic. Without such a tool, architectural choices have had to be evaluated either by simulating each candidate circuit from the component level up, or by extrapolating from single circuit demonstrations which cannot resolve architectural trade-offs or identify where device-level efficiency may be lost at scale.

The common thread across the first three circuit contributions is that each is designed around the physics of the AQFP rather than fitting a CMOS-style abstraction to the superconducting logic family. The multiphase clock is repurposed as a synchronization primitive, then as a memory structure, then as a scheduling mechanism. The AQFP's baseline per-cycle dissipation, rooted in the two-terminal nature of the device, drives the architecture design choice to keep every stage productive on every cycle. While the final contribution closes the loop by extending system-level design tools to superconducting electronics so these device-aware design choices can be quantified against CMOS on real meaningful workloads.

Chapter 3

Synchronizer for AQFP

In AQFP circuits, digital logic is encoded as the polarity of current pulses on superconducting wires, and these pulses propagate only when driven by a multiphase AC excitation clock. This creates a strict timing model where data must arrive during the precise excitation window of the receiving stage. Small deviations due to interconnect delay, clock skew, or mismatched clock domains, can cause data to be mis-sampled or dropped entirely. As circuits scale in physical size or span multiple chips, distributing perfectly phase-aligned clocks becomes increasingly difficult, constraining both layout and system architecture.

In CMOS circuits, synchronizers are a common circuit element used to safely transfer data between regions of a chip operating on different clocks, known as clock domain crossing (CDC), or mis-aligned clock phases. The form a CMOS synchronizer takes depends on the relationship between the source and destination clocks. When the two clocks are fully asynchronous, meaning they share no fixed frequency or phase relationship, the receiving flip-flop can be clocked while its input is changing, leaving its output in an unstable intermediate state known as metastability. The standard mitigation is a two-flip-flop synchronizer, which cascades a second flop to allow the metastable state time to resolve before its value is consumed downstream. When the two clocks instead share a nominal frequency, a simpler class of synchronizer can be used: a mesochronous synchronizer handles a fixed but unknown phase offset between the source and destination clocks, while a plesiochronous synchronizer additionally tolerates a small frequency mismatch that causes the phase relationship to drift slowly over time (i.e. clock skew). These operate by detecting which phase the source data arrives on and steering it to a known phase of the destination clock, rather than by resolving an unbounded metastable state.

The AQFP synchronizer presented here is closest in spirit to a mesochronous synchronizer. AQFP does not suffer from logic-level metastability in the CMOS sense. Although the parametron has an energy maximum between its two logical wells, this maximum is unstable,

and thermal noise drives the cell into one well or the other within a single excitation cycle. The timing problem in AQFP is instead one of data capture. The device can only sample data while it is activated, so data that arrives outside the excitation window of the receiving cell is lost entirely. The phase synchronizer resolves this by sampling the input across all excitation phases and propagating the captured value to a known reference phase of the next cycle. No equivalent circuit existed natively for AQFP prior to this work.

The text that follows includes reprints from a work that was originally published in *IEEE Transactions on Applied Superconductivity* [45]. Note that this text uses an earlier implementation and naming convention for AQFP circuits. The circuits are designed with a three-phase clock rather than the four-phase clock used in the rest of this thesis, and we refer to the clock as the *activation* signal rather than the *excitation* signal.

3.1 Weak Constant AQFP

An AQFP, schematic shown in Figure 3.1, consists of two superconducting loops, each with a single Josephson junction, that share a backbone and are inductively coupled to an AC activation signal. When the activation signal is on, a single flux quantum will be stored in one of the two loops, corresponding to a logical "1" or "0" state, directed by the polarity of the input current passing through each of the loops. A conventional AQFP buffer does not have a logically deterministic state for a null input condition; in the case of no current on the input line, it will amplify thermal noise and flux-bias offsets to generate pseudorandom data output [66]. This is an issue for a bit-level synchronizer which must sample the input across all phases and selectively propagate the meaningful data value. Therefore, to design a phase synchronizer, we first develop the notion of a weak-constant AQFP cell.

In such a gate, a constant logic value will be given on the output if the signal on the input is null or weak, but in the presence of a data signal larger than a predetermined threshold, the constant output will be overwritten by the data bit. As shown in Figure 3.1, the weak constant is created by introducing a slight asymmetry between the activation coupling coefficients, leading the AQFP to repeatedly favor one state. The coupling coefficient is manipulated, as opposed to the size of the branch inductors, L_α, because this has a greater effect on the parametron state [67]. The strength of this offset can be controlled by the magnitude of the of asymmetry, while the default state of the weak constant QFP can be controlled by selecting which loop is larger.

This can be further understood by examining the current dynamics of the basic AQFP device. Writing Kirchhoff's Current Law on each of the nodes of the AQFP results in the

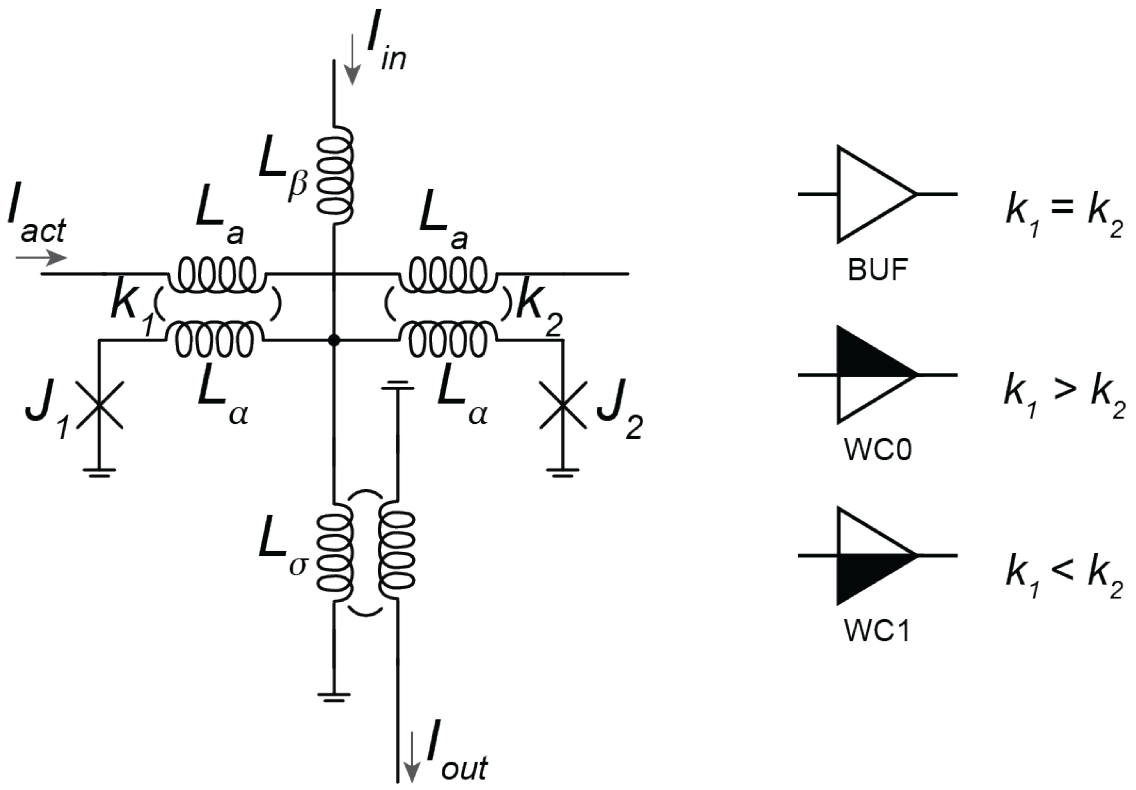


Figure 3.1: (a) Generic AQFP schematic. (b) Circuit symbols for buffer (BUF), weak constant 0 (WC0), and weak constant 1 (WC1), and their associated modification to the schematic. A buffer has perfectly symmetric coupled loops, while the weak constant has unequal coupling constants. © 2023 IEEE.

following three equations

$$I_c \sin\phi_1 + \ell_\alpha(\phi_1 - \sigma - n_a k_1 \alpha) = 0 \tag{3.1}$$

$$\begin{aligned}\ell_\alpha(\phi_1 + \phi_2 - 2\sigma + n_a(k_2 - k_1)\alpha) \\ + \ell_\beta(\beta - \sigma) - \ell_\sigma \sigma = 0\end{aligned} \tag{3.2}$$

$$I_c \sin\phi_2 + \ell_\alpha(\phi_2 - \sigma + n_a k_2 \alpha) = 0 \tag{3.3}$$

where I_c is the critical current of the junctions, $\ell = \frac{\Phi_o}{2\pi}\frac{1}{L}$ for each inductor, ϕ_1 and ϕ_2 are the phase differences across each junction, β, α, and σ are the phase differences across the input, loop, and output inductors, respectively; n_a^2 is the activation transformer ratio, which we've assumed to always be symmetric in both loops, and k_1 and k_2 are the coupling constants

which can vary. We can reorganize these equations to be expressed in terms of $\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$ and $\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$, and solve Eq. 3.2 for σ to reduce the three equations to two and get the following result.

$$2I_c \sin\phi_+ \cos\phi_- + \ell_+ \left(\phi_+ + \frac{n_a(k_2 - k_1)}{2}\alpha - \frac{\ell_\beta}{\ell_\alpha(k_\sigma - 2)}\beta\right) = 0 \tag{3.4}$$

$$2I_c \cos\phi_+ \sin\phi_- + \ell_\alpha(2\phi_- - n_a(k_2 + k_1)\alpha) = 0 \tag{3.5}$$

where $\ell_+ = \frac{2\ell_\alpha(k_\sigma - 2)}{k_\sigma}$ and $k_\sigma = (2\ell_\alpha + \ell_\beta + \ell_\sigma)/\ell_\alpha$. These equations guide the operation of the parametron.

The AQFP will be storing information when it's in the double potential state, *i.e.* when the activation phase difference $\alpha = \pm\frac{\pi}{n_a(k_2+k_1)}$ [42]. This corresponds to a solution for Eq. 3.5 when $\phi_- = \pm\pi$ and we can therefore reduce our analysis to Eq. 3.4 when $\phi_- = 0, \pm\pi$. As shown in Figure 3.2 the solution of Eq. 3.4 can be visualized as the intersection of a sine function with a straight line. Provided the slope, ℓ_+, is less than one, there are up to three possible solutions: high or low ϕ_+ corresponding to a logical 1 or 0, and the solution around 0 which corresponds to an energy maxima.

The state that the parametron settles in depends on the linear shift provided by the α and β term in the line equation. If $k_1 = k_2$, as is the case for the buffer, then the state of the gate depends entirely on the input value. If no input value is given, it will randomly slip into either high or low state. However, if the coupling constants are not equal, then the parametron state also depends on the activation signal, and it is this asymmetry that produces the biased weak constant cell. In other words, the extra coupling between the activation transformer behaves as a current source in one of the parametron loops and causes the output to favor one side. Additionally, optimization of the constant cell from a potential energy perspective has been done by Ando [67].

Figure 3.3 demonstrates simulation of both weak constant cells compared to a conventional AQFP buffer. The SPICE level simulation was performed with Cadence SPECTRE simulator with support for Josephson junction dynamics. A 1μA Gaussian noise source was included in the input signal to highlight the distinction between the weak constant cells and the buffer: the buffer amplifies the noise source (highlighted in red), while the weak constant maintains its constant output. For an input magnitude of 15μA, the operational range of WC0 was found to be $k_2/k_1 = [1.01, 1.36]$; a ratio larger than this will result in a full constant cell. The inverse is true for the WC1 cell.

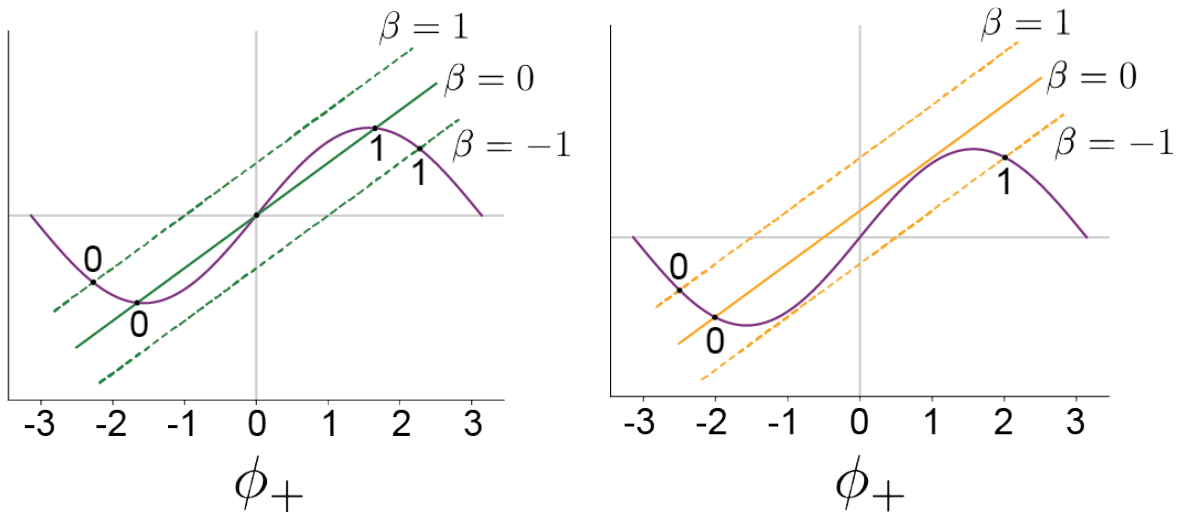


Figure 3.2: Solutions to Eq. 3.4 for (a) the buffer when $k_1 = k_2$ and (b) the weak constant 0 when $k_1 > k_2$. The logical value that each solution corresponds to is labeled on the graphs. Dashed lines above and below correspond to input 1 ($\beta = 1$) and 0 ($\beta = -1$), respectively; and the solid line is state of the parametron when activated with no input ($\beta = 0$). © 2023 IEEE.

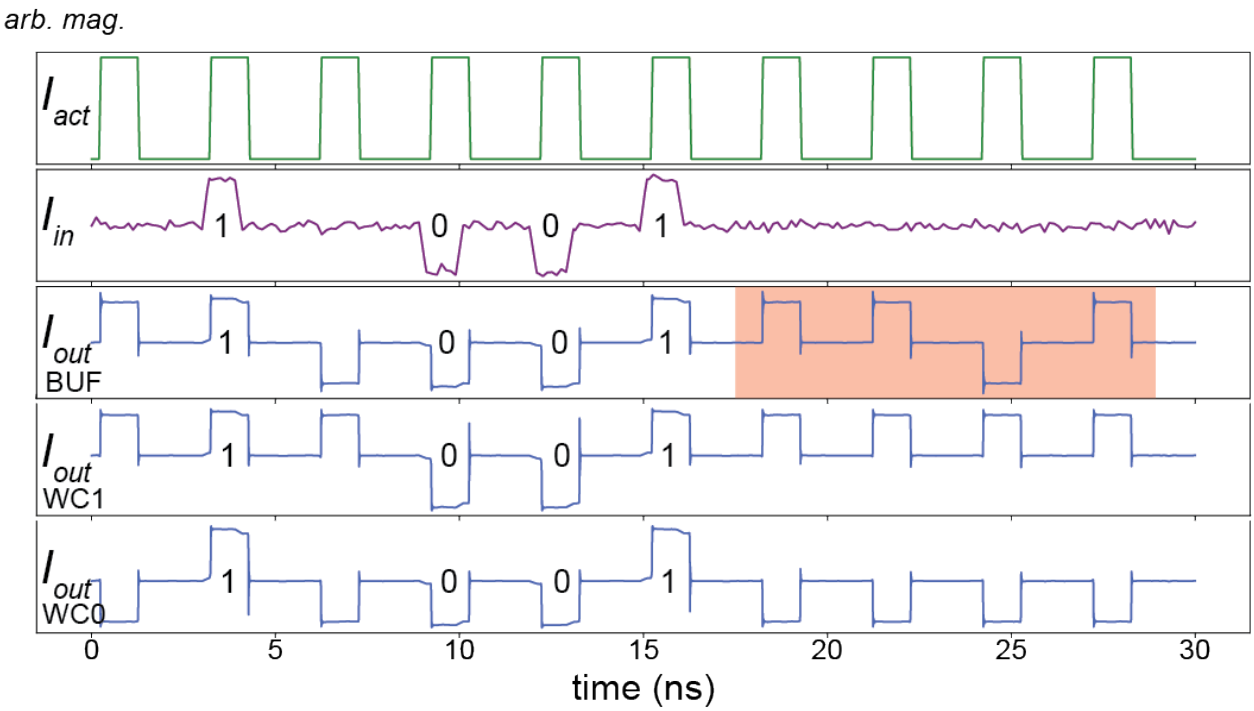


Figure 3.3: Simulated output of a single buffer, weak constant 1 (WC1), and weak constant 0 (WC0). Gaussian random noise has been added to the input signal to highlight proper operation of each cell: the buffer amplifies the pseudorandom noise when no data value is passed, while WC0 and WC1 default to 0 or 1 respectively. © 2023 IEEE.

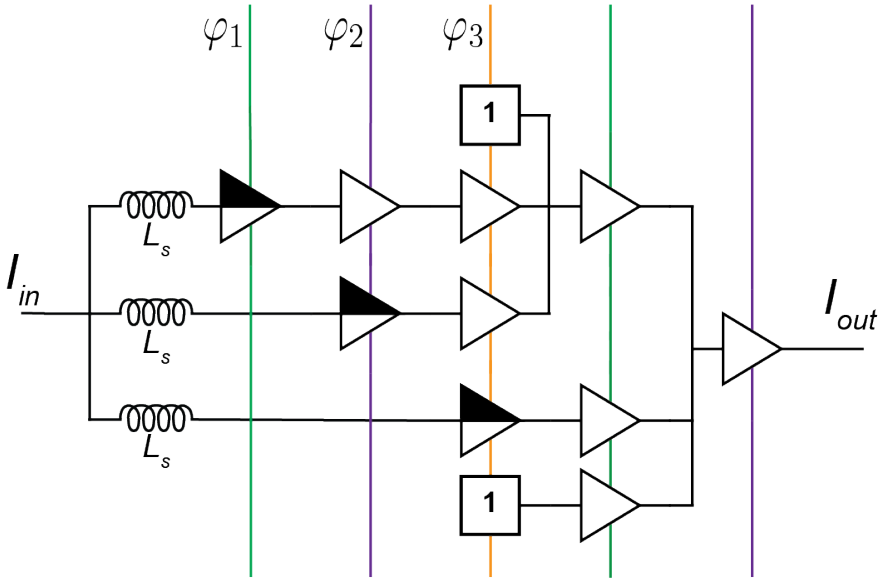


Figure 3.4: Phase synchronizer circuit schematic. Majority operations are indicated by the connection of three nodes, thus a constant 1 joined with two inputs results in an OR operation. The activation signal for each cell is labeled with φ_1, φ_2, φ_3 and cycles through each stage from left to right, as the color indicates. Each QFP cell requires 2 JJs, the circuit has 24 JJs in total. © 2023 IEEE.

3.2 Phase synchronizer

In this section we will discuss the requirements for a synchronizer circuit in AQFP broadly, and then present the details of our design. Followed by simulation of the full synchronizer and measurement of the individual weak constant cells.

3.2.1 Requirements

Due to the two-terminal nature of the parametron device, AQFP logic requires a multiphase activation signal with a minimum of 3 phases. In a circuit network, a single QFP device cycles through states of (i) **receiving** data while the activation signal is brought high, (ii) **propagating** data at the output/input while the activation signal remains high, and (iii) **blocking** data while the activation signal is low to prevent back propagation of data. Therefore, the arrival of data must align with a QFP in the receiving state.

If the data arrival phase is unknown, then a bit-level phase synchronizer must perform the following tasks: (i) accept data input during any time of the activation cycle (whether it aligns with a single phase or is spread between multiple); (ii) remove temporal uncertainty associated with the input bit by propagating the value to an output signal of a known phase; (iii) output a predetermined value in the absence of an input signal.

We designed a phase synchronizer with a multiplexer topology that samples each input phase through a weak constant zero cell and outputs the logical OR of all input phases, shown in Figure 3.4. The circuit meets the phase synchronizer requirements for a three-phase clock, although the general design could be extended to a larger phase count if needed.

3.2.2 Simulation

Operation of the phase synchronizer was verified in simulation with Cadence SPECTRE tools, shown in Figure 3.5. Notably, the current input signal to the phase synchronizer must be large enough to split across all of the feedback inductors and overcome the threshold of the weak constant values. In simulation with 18 pH inductors on each of the input branches, a 60 μA input amplitude was required; compared to the 15 μA logic-level currents typically used in AQFP circuits. Therefore, to drive the phase synchronizer with AQFP logic, a multi-turn transformer could be added to the input to passively amplify the current. Alternatively, if area is more of a concern than energy, a DC-bias amplifier, such as a SQUID amplifier [68] or nanocryotron [69], could be added to the input.

For VLSI design, the phase synchronizer is an important circuit to include at I/O ports of AQFP logic units and could demand a higher current level as a design requirement; similar to I/O versus logic voltage levels in CMOS IC design.

3.2.3 Circuit Measurement

We demonstrate initial fabrication and testing of the weak constant cell. The circuits were fabricated in the SFQ5ee process [27], and tested with a liquid helium dunk probe at kHz frequencies, using an Octopux measurement system [70]. Figure 3.7 shows a micrograph of our AQFP buffer (3.7a) alongside a WC0 cell (3.7b). The asymmetry between the coupling constants was achieved by briefly routing the activation line (red highlighted line in Figure 3.7) away from the right arm inductor of the QFP.

The output of the gates was measured by directly coupling a DC-bias SQUID to each of the output lines. The results from testing are shown in Figure 3.6. Ideally, we would expect the buffer to output pseudorandom values when activated with 0 μA on the input line, as discussed in section 3.1; however, in reality, the gray zone of the buffer is not located perfectly at 0μA, presumably due to flux trapping noise in the circuit and fabrication tolerances [44]. In this sense, all buffer cells can indirectly behave as weak constants, but a weak constant is distinct in its purposeful asymmetric design and repeatable bias.

The data threshold for our fabricated buffer cell is around 5μA, and 90μA for the weak constant. Given that the current level used in AQFP logic is around 15 - 30μA, the 90μA

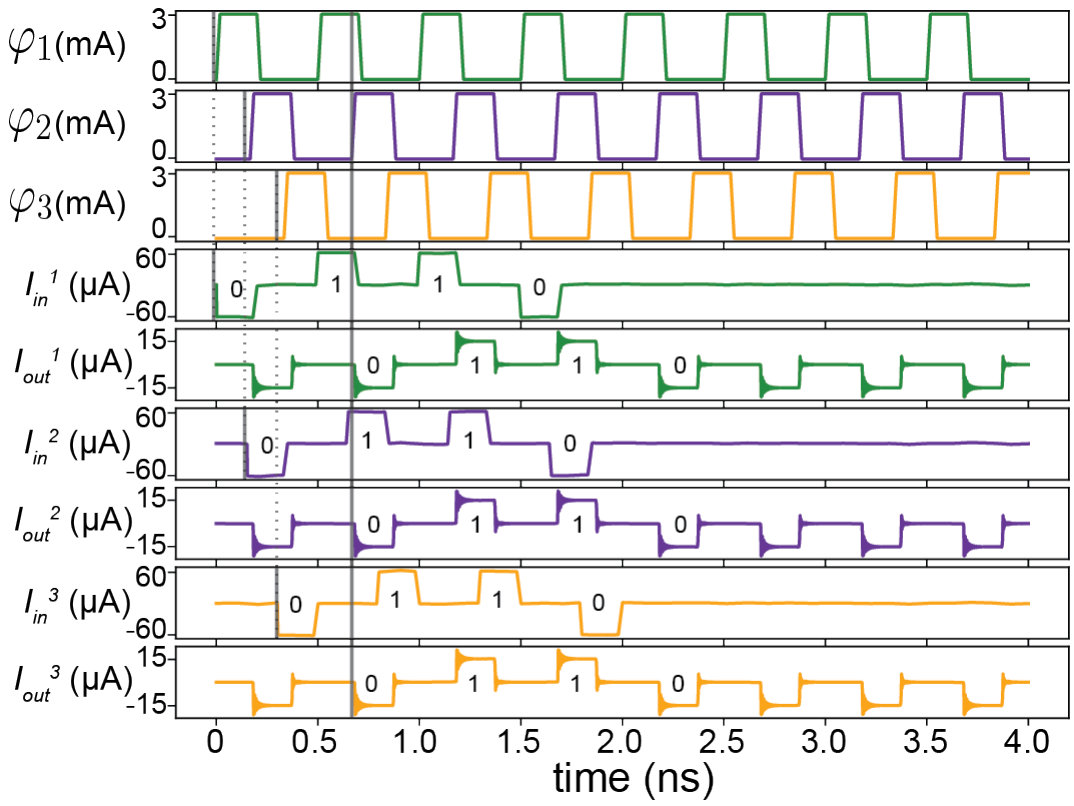
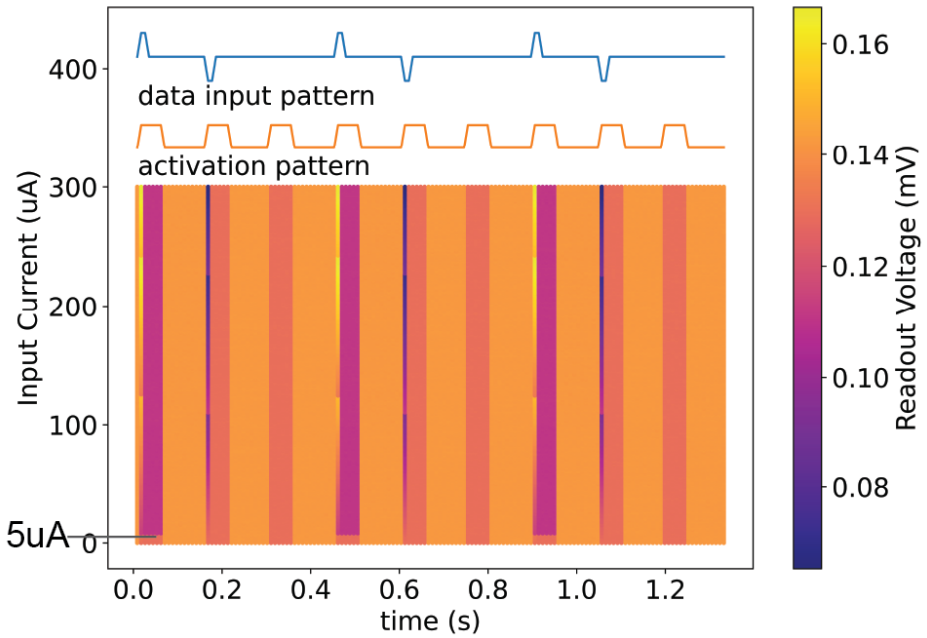
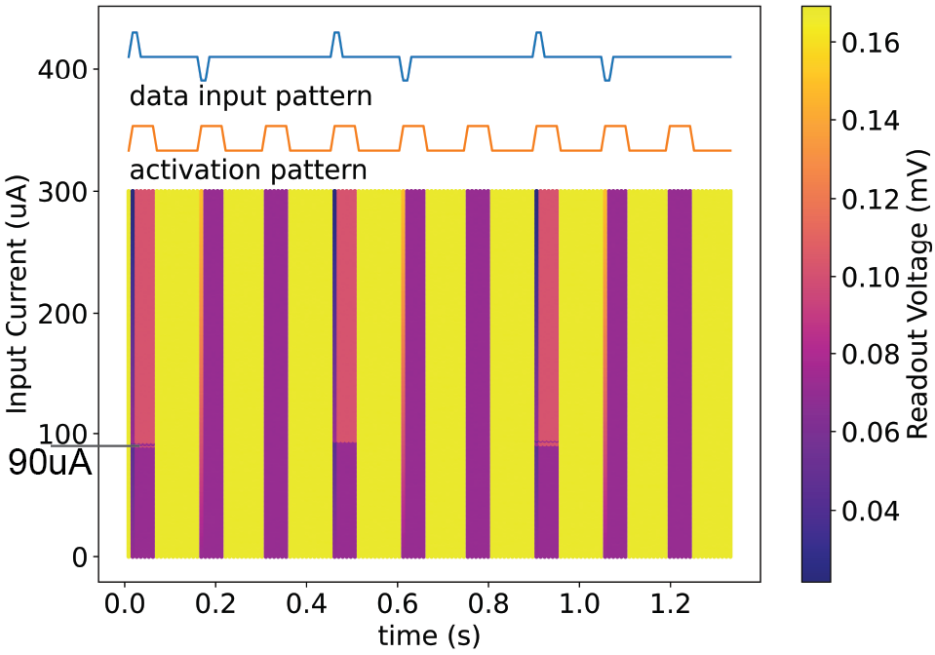


Figure 3.5: SPICE level simulation of the phase synchronizer circuit at 1 GHz activation cycle. The input was shifted to align with the first, second, and third phase, I_{in1}, I_{in2}, I_{in3} respectively, while the output remains aligned at the second phase of the next activation cycle. A Gaussian noise source was included on the input signal. L_s was minimized to 18pH for isolation of back propagation noise. The operational margin on the activation amplitude is [-16%, +8%] centered around 3mA.© 2023 IEEE.

threshold is much too high of an operating point for the weak constant. This most likely arises from too large of an asymmetry between the activation couple constants and further optimizations are reserved for future work.

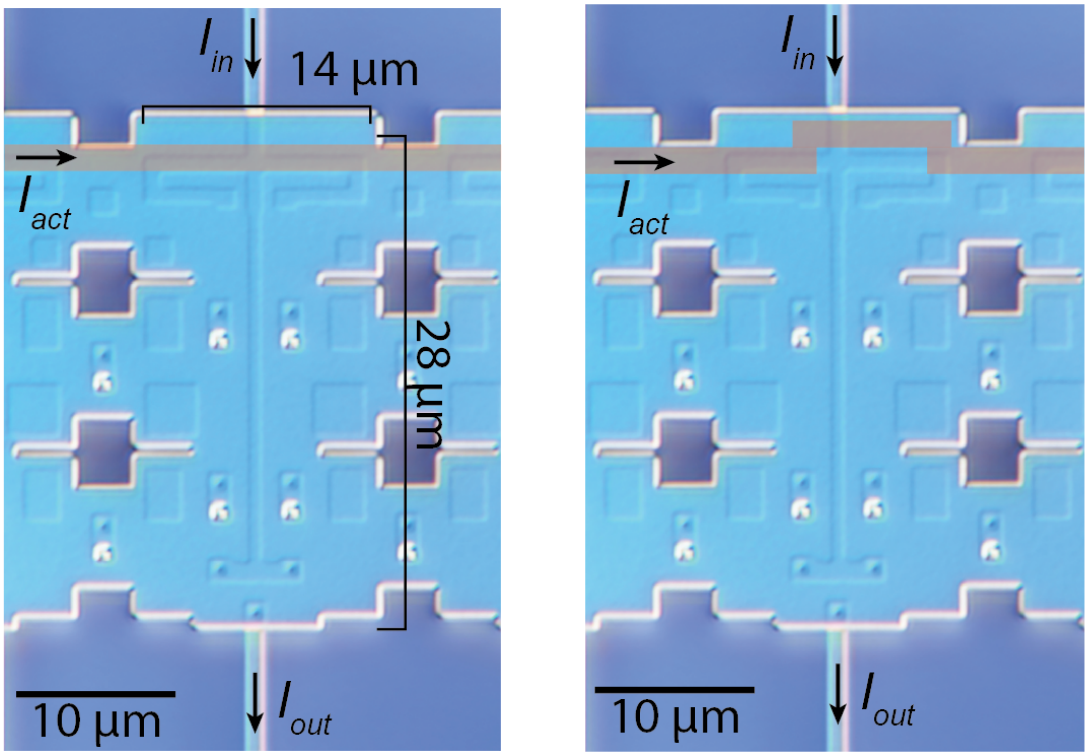


(a) Buffer



(b) Weak Constant 0

Figure 3.6: Input margin measurements of the (a) AQFP buffer and (b) weak constant. The data and activation pattern are shown across the top. When the gate's data threshold is reached, the output voltage begins switching with the data pattern: 5 μA for the buffer, 90 μA for the weak constant 0 cell. © 2023 IEEE.



(a) Buffer (b) Weak Constant

Figure 3.7: A polarized optical micrograph of (a) the buffer and (b) the weak constant is shown next to each device's test data. The activation signal is coupled through an inductor segment which is not pictured on the micrograph metal layer, but drawn in red to highlight the coupling asymmetry which is critical to the weak constant operation. © 2023 IEEE.

3.3 Outlook

The phase synchronizer is the first AQFP primitive that allows data to be captured without strict phase alignment to the local excitation clock. Its most important application is as a building block for clock domain crossing. By sampling the input across all excitation phases and propagating the captured value to a known reference phase, the synchronizer absorbs the timing uncertainty between a source and destination domain. This means that a designer no longer needs to guarantee phase-matched delivery of every signal between them.

This capability becomes increasingly important as AQFP circuits scale. Distributing a single phase-aligned excitation clock across a large die, let alone across multiple chips, becomes progressively harder as the physical extent of the circuit grows and interconnect delays accumulate. Partitioning a large design into independently clocked regions, joined at their boundaries by synchronizers, relaxes this constraint and is a prerequisite for the kind of multi-chip module integration that any large-scale superconducting system will require. Additionally, having different regions of an AQFP circuit run at different frequencies is an

important design space to have available when scaling AQFP to large-scale designs, as we will see in the system-level architecture studies in Chapter 6.

Chapter 4

AQFP SR-Loop Register File

In this chapter, we introduce the AQFP SR-Loop storage cell and discuss how it can be organized into a register file. We present two versions of the register file architecture, each with significant improvements from the prior work. Section 1.2 includes material reprinted from work originally published in *IEEE Transactions on Applied Superconductivity* [71]. I would like to acknowledge David Russo for his contributions to the v2 design optimization and development of the directly coupled AQFP for a more compact storage component.

4.1 Circulating Storage for Superconductors

High-density, low-power memory remains one of the greatest obstacles to practical superconducting computing [10],[11],[72]. A modern processor relies on a memory hierarchy of mature CMOS primitives: DRAM for main memory, SRAM for caches, and latch-based register files in the datapath. Superconducting analogs have been demonstrated and proposed for each of these levels, but demonstrations remain at much smaller scales and lower densities than their CMOS counterparts, with none reaching the capacity required for general-purpose computing.

There have been many attempts to build storage cells from superconducting circuits, including vortex transition RAM [73], Josephson static RAM (JSRAM) [74], and superconducting nanowire memory [75]. However, the JJ-based designs have struggled to scale due to their larger footprint and difficulty driving addressable arrays with low fanout devices, while the less mature nanowire-based memories have promising footprint and fanout, but at the cost of a much large power dissipation. The largest superconducting random-access memory demonstrated with measurement to date remains a 4 Kbit Josephson RAM from 1995 [76].

The fundamental difficulty is density. Superconducting storage elements are built from loops of supercurrent and therefore inherently occupy more area than a transistor storing data

as charge on a small capacitor. Compounding this, most superconducting logic families have limited fanout, so scaling these storage cells into a conventional addressable array requires many splitters and buffers along wordlines and bitlines. Compact register files are particularly scarce in AQFP logic. To our knowledge, only a single AQFP register file design has been previously demonstrated [77], which was later incorporated into the MANA microprocessor as the 16-word by 4-bit register file. Within the MANA design, the register file was the largest pipeline stage by both JJ count (8,142) and footprint (4.7×6.6 mm^2) [12].

In most superconducting memory designs, a JJ-based circuit provides a single-bit storage cell and the memory architecture is organized into a randomly accessible array. But the cost of moving data to and from a storage cell is high. In AQFP, addressable arrays require decoders and demultiplexers with large networks of splitters and timing buffers to fan out a data value across all of the storage cells. A single-bit storage cell pays the full addressing cost to deposit one bit, leaving the addressing infrastructure the dominant area and power cost. The register file design introduced in this chapter instead stores multiple bits in a shift-register feedback loop, so that the addressing overhead is amortized across the bits in the storage cell.

The design takes inspiration from delay line memories used in early digital computers, like EDVAC and EDSAC with mercury acoustic delay line memories [78],[79], as well as the more recent implementation of superconducting delay line memory built on SFQ-driven passive transmission lines [23]. The shared idea is to store data across time with sequential access pattern, rather than across space with random access pattern. Pulses propagate along a delay medium and are made available for reading on every pass around a loop. Circulating storage of this form trades latency for area and addressing complexity. However, the additional latency incurred from sequential access in the circulating memory can be hidden with appropriate multithreading (Chp. 5) and workload scheduling (Chp. 6). In the AQFP version presented here, the delay medium is not a passive transmission line but an active chain of AQFP buffers configured in a loop with a splitter to readout values.

4.2 SR-Loop Memory Cell

The SR-Loop is the storage primitive on which the register files in the rest of this chapter are built. This section presents its design, experimental measurements of 3-bit and 8-bit implementations, and an analysis of how energy and latency scale with bit depth.

4.2.1 Design

The SR-Loop memory cell is based on a majority gate set-reset latch [80], but modified with additional buffers in the feedback loop to store any desired length of data, rather than a single bit. As shown in Table 4.1, when the Set and Reset signal values match, the previously stored data value, Q_{n-1}, is held. When the Set and Reset signals are different, the stored data is overwritten to match the Set signal.

S	R	Q_n	Action
0	0	Q_{n-1}	Hold
0	1	0	Write 0
1	0	1	Write 1
1	1	Q_{n-1}	Hold

Table 4.1: Logic truth table for Set-Reset Latch. © 2025 IEEE.

The SR-Loop can be thought of as a delay line memory element where the delay element is a loop of AQFP buffers which can store a sequence of bits by indefinitely passing them along the loop. A schematic diagram for the SR-Loop is shown in Figure 4.1b. The bits are addressable with the SR latch at the start of the loop and read out with a 1-to-2 splitter at the end of the loop. The bit depth is the number of bits simultaneously held in the storage loop, which is set by the number of buffers in the loop. To allow proper read and write operation, each bit must be temporally separated by a single clock cycle; therefore, in the case of a four-phase clock, the bit is spatially spread across 4 buffers. With the number of buffers drawn in Figure 4.1b, it takes 12 clock phases for a single bit to propagate from the start to the end of the storage loop and 1 bit is written every 4 phases, so this design can hold 3 bits of information with a 3 clock cycle read latency. We define the read latency as the minimum time delay for a bit written to the loop to be available on the output, Q_n. However, the ellipsis at the turning point of the loop indicate that any number of additional buffers could be added to increase the storage capacity of the cell. For example, an 8-bit storage cell requires 20 additional buffers, for a total of 32 clock phases from start to end and therefore a 8 clock cycle read latency.

The SR-Loop has a non-destructive readout and an output bit is always available on the splitter. Therefore, one must keep track of timing to ensure that the bit on the output aligns with the bit intended to be read. The readout timing can be handled by a memory controller or hardcoded in the microarchitecture design, as will be discussed in more detail in Chapter 5

4.2.2 Measurement

Two versions of the SR-Loop storage cell were fabricated and tested: a 3-bit cell that demonstrates basic circuit operation and a larger 8-bit cell that explores layout optimizations

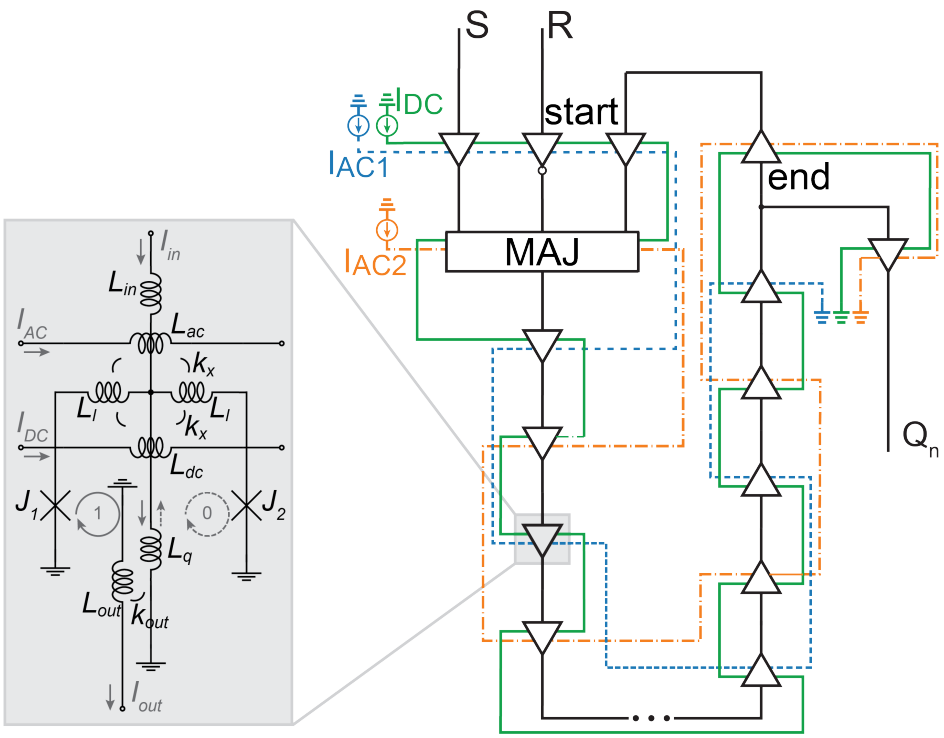


Figure 4.1: **(a)** Schematic of a single AQFP buffer. The 1 (0) logic state is represented by a single fluxon in the left (right) loop which generates a positive (negative) current along the center shunt inductor, L_q. The device values are: L_{in}=350fH, L_{ac}=L_{dc}=7.4pH, k_x=0.18, L_l = 1.3pH, L_q=10pH, L_{out}=8pH, k_{out}=0.3 **(b)** Schematic diagram of SR-Loop memory cell. A majority gate SR Latch at the input receives the Set (S), Reset (R), and previously stored bit (Q_{n-1}) and propagates the output along a shift register storage loop. A four-phase clock is generated from two AC signals 90 degrees out of phase, I_{AC1} and I_{AC2}, and a DC bias, I_{DC}, and 1 bit of information is stored across every four buffers. With the buffers shown, this memory cell would store 3 bits, but the ellipsis indicate additional buffers can be added for deeper storage capacity. © 2025 IEEE.

for area. This subsection discusses experimental testing of both designs.

4.2.2.1 3-bit Cell

The 3-bit SR-Loop has a 240 μm × 100 μm footprint with 34 JJs, although it was not designed for aggressive area optimization. The experimental results are shown in Figure 4.2.

Figure 4.2a demonstrates the SR-loop writing every possible 3-bit sequence and storing it for two complete passes around the loop. As expected, the data takes 3 clock cycles to propagate to the output, and the output is aligned with the fourth clock phase, *i.e.* the negative peak of I_{AC2}. The output is read from the splitter buffer into a weakly coupled DC-SQUID amplifier. The readout SQUID is biased such that it is always in its voltage state, so the polarity of the output current pulse from the AQFP shifts the SQUID voltage in the

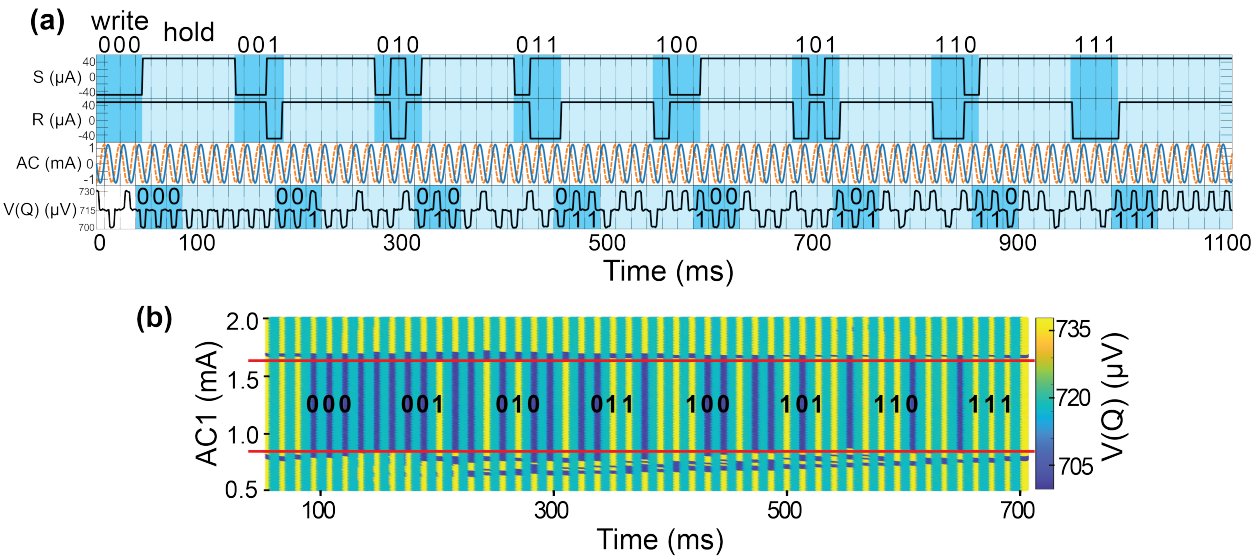


Figure 4.2: **(a)** Basic circuit operation is verified at 70 Hz by writing and storing every possible 3-bit sequence. The top two axes show the Set and Reset signal supplied to the circuit. The third axis plots both I_{AC1}(solid, blue) and I_{AC2}(dashed, orange) and verifies that the output is aligned with phase four, the minimum peak on I_{AC2}. The bottom axis plots the SR-Loop readout, $V(Q)$, amplified to the Octopux at room temperature with a DC-SQUID voltage readout. As expected, we see each bit sequence repeated 3 times - once when it is written and twice while storing. **(b)** The margins are obtained by individually sweeping the amplitude of each AC signal and the DC bias for all possible storage sequences (same data sequences as part (a), but modified for only one storage of each sequence to decrease testing time). The clock margin plot for only I_{AC1} is shown here as an example. Horizontal slices describe a single run of the data pattern for the AC amplitude specified on the y-axis. The output voltage state is read from the color map - blue regions represent a 0 logic state and yellow is 1. The limits of the correct operating region are indicated with a red line, *e.g.* the circuit works for I_{AC1} values between 0.859 mA and 1.633 mA. Margins for I_{AC2} and I_{DC} were extracted in the same way. The I_{AC1}, I_{AC2}, and I_{DC} margins are ±31%, 32%, and 15%, respectively. © 2025 IEEE.

positive or negative direction, which is then detected at room temperature by the Octopux.

There is an asymmetry between the 0 and 1 readout because we are reading the AQFP output along the sinusoidal DC-SQUID modulation curve. Therefore, depending on where the SQUID is biased, equal and opposite AQFP current pulses can result in unequal voltage shifts. The asymmetry could be removed by selecting a readout SQUID bias that is along a linear region of the SQUID modulation curve, or including an additional bias line to linearize the sinusoidal modulation. Additionally, in this chip, the SQUID bias line was shared between multiple AQFP circuits, so the smaller noisy pulses which align with the minimum peak of I_{AC2} (the second clock phase) is unwanted feedthrough from a neighboring circuit.

Figure 4.2b displays the clock amplitude margins. A single data pattern was evaluated for

proper operation while the amplitudes of I_{AC1}, I_{AC2} and I_{DC} were varied individually. The nominal operating point and clock margins were 1.25 mA ± 31%, 1.27 mA ± 32%, and 1.29 mA ± 15% for I_{AC1}, I_{AC2}, and I_{DC}, respectively. The circuit was measured at only 70 Hz due to limitations of the testing apparatus, so an analysis of clock margin dependency on frequency is left to future work.

4.2.2.2 8-bit Cell

The 8-bit SR-Loop storage cell has a 30 buffer feedback loop and 78 Josephson junctions in total. Three layout variants of this cell were fabricated, each with a more aggressively optimized footprint, as shown in Figure 4.3. The area of each design is: (a) 0.110 mm^2, (b) 0.084 mm^2, and (c) 0.064 mm^2.

The footprint reductions across versions come from progressively more compact clock routing. Because the feedback loop must close on itself and AQFP buffers cannot drive long wires, a long shift register with all of the data propagating in the same direction (i.e. a straight line) is not viable. The buffer chain must be able to propagate AQFP data in multiple directions to make a viable feedback loop. However, in our four-phase clocking scheme, the direction of data propagation is set by the polarity of the clock currents entering each buffer so the data direction cannot be rotated without correspondingly rotating the clock routing.

The three layout variants differ in how they reconcile the rotating data path with the clock routing. Version (a) follows the schematic directly: the clock lines meander with the buffer chain, rotating with the devices at every turn so that the clock currents always match the local data propagation direction. This works for a standalone cell, but it does not compose well into larger designs. At scale, AQFP clock signals are routed as a grid, with all buffers along a single clock wire, such that they are biased in series. A storage cell that forces the clock to follow the data path complicates integration into that grid.

Version (b) addresses this by holding the clock placement relative to the AQFP devices fixed, but flipping the order of them between adjacent columns of buffers, so that data propagates downward in one column and upward in the next. The swap is implemented in an extra ground tile between columns. The schematic on the right-hand side of Figure 4.3 shows the swapping scheme. The numbers on each side of the buffers indicate the clock phase on which that buffer fires. This change reduces the cell width from 240 μm to 200 μm and, more importantly, makes the cell easier to embed in a larger AQFP system with uniform grid clocking.

Version (c) compacts the width further by moving the clock swap from a dedicated ground tile to a lower metal layer, M2, beneath the M4 ground plane. The M4 ground plane is

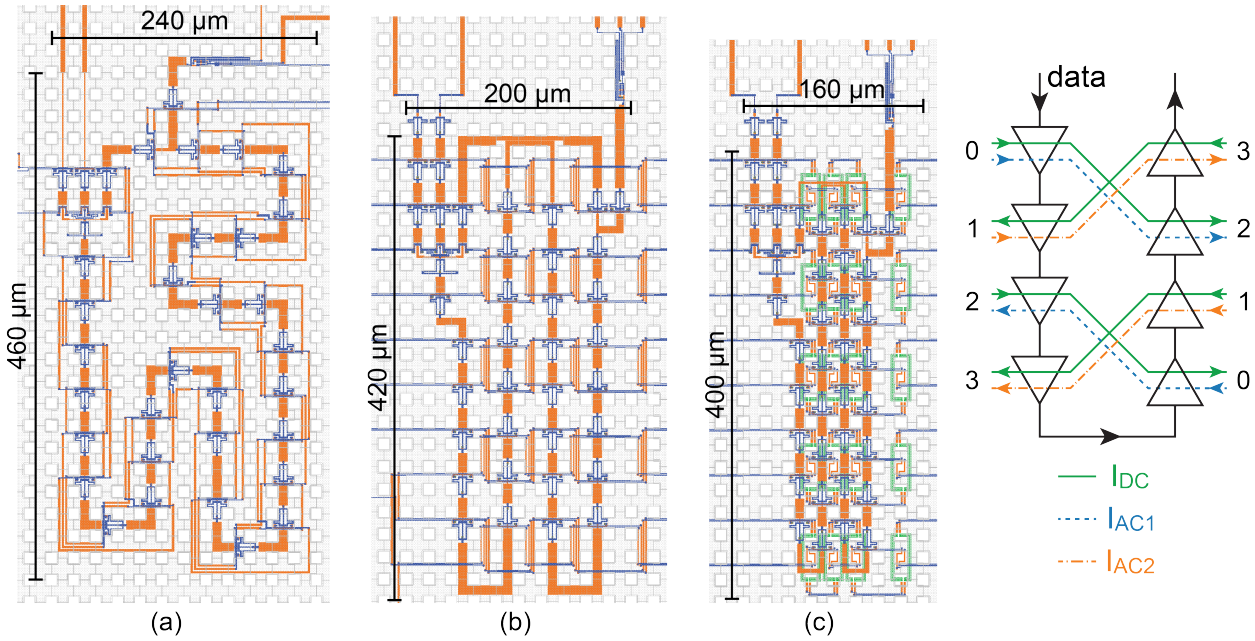


Figure 4.3: Layout variations for an 8-bit SR-Loop storage cell. (a) follows the schematic diagram exactly, snaking the clock routing with the buffers; (b) shrinks the footprint by swapping neighboring clock signals to flip the direction of data propagation between buffer turns; (c) shrinks the footprint further by routing the clock swapping to a lower metal layer (M2, below a M4 ground plane). The schematic on the right shows the scheme for the clock signal swapping so that data can propagate down on one side and up on the other; arrows indicate direction of current flow.

intended to isolate the high-current clock lines (∼1 mA) from the low-current data path (15 μA) above, while the clock swap is overlapping with the devices rather than costing a full ground tile of area. This brings the width down to 160 μm.

Of the three layouts, only version (a) operated correctly. Figure 4.4 shows measurement data of an 8-bit pattern written to the storage loop and read back across two full rotations around the loop, confirming both the write and the circulating storage behavior. Versions (b) and (c) did not function correctly. In both cases, we predict that the failure is coming from mutual coupling between the clock traces and the data path, but the affected geometry differs. In version (b), the clock swap is implemented in a ground tile between columns and the data overlaps clock traces only at the 180° turns. In version (c), one clock runs on M2 and the other on M6 while the data propagates on M5 between them, so the data overlaps clock at every point along propagation rather than only at the turns. Given the disparity in current levels between the clock (1 mA) and the data (15 μA), even a small mutual inductance is sufficient for clock feedthrough to disturb the stored bit. The schematic-level SPICE simulations used during design do not include extracted parasitic mutual inductances

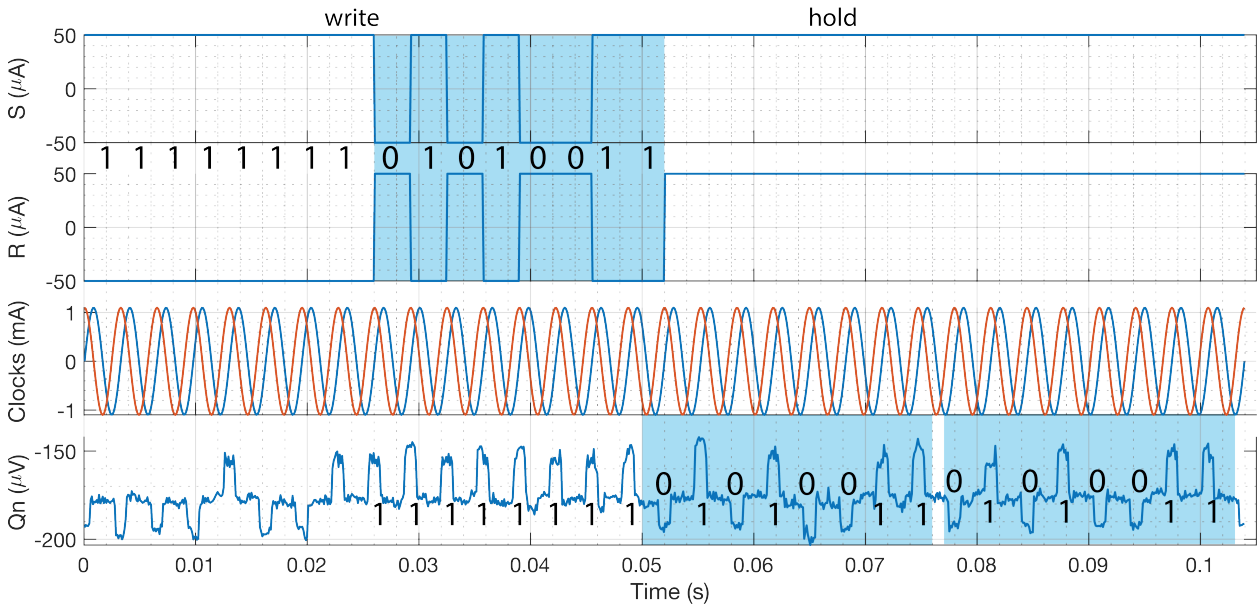


Figure 4.4: Write and storage operation of the 8-bit SR-Loop cell, layout version (a) in Figure 4.3. The cell is first initialized with all ones, then the pattern `01010011` is written into the storage loop and read out across two full rotations around the loop. Testing is performed at 300 Hz, limited by the measurement apparatus.

between routing layers, so the coupling was not predicted. Incorporating parasitic extraction into the design flow could prevent this class of failure in future layouts.

We have generated updated layouts that mitigate this coupling by modifying how the clock traces are flipped between alternating data-path directions, so that the clock and data do not overlap on adjacent metal layers. Testing of these revised designs is left to future work. In parallel, we have designed a test structure to directly measure the mutual coupling between a clock-carrying wire and a data path on an adjacent metal layer. This will provide a quantitative constraint on allowable crossings in future AQFP layouts. This test chip is currently in fabrication, with return expected in the coming months.

4.2.2.3 Summary of Metrics

Table 4.2 provides a summary of the measured and simulated metrics from the 3-bit and 8-bit SR-Loop storage cell. The circuits were fabricated in the MITLL SFQ5ee process [27] and tested at 4 K in a liquid helium immersion probe using an Octopux measurement system [70]. Although the circuits are simulated up to 5 GHz, the experimental testing was constrained to frequencies on the order of 100 Hz due to limitations in testing equipment. The measurements were performed in a magnetically shielded environment with ambient field measured to be around 100 nT. Circuits were designed and simulated in the Cadence Virtuoso ADE using

Table 4.2: Summary of measured (meas.) and simulated (sim.) metrics for the 3-bit and 8-bit SR-Loop cell circuits.

	3-bit cell	**8-bit cell**
JJ count	34	78
Area (mm^2)	0.024	0.110
meas. frequency (Hz)	70	300
sim. frequency (GHz)	5	5
sim. energy dissipation† per cycle @ 5 GHz (aJ)	29.9	68.8

† includes 1000 W/W cooling overhead for room temperature energy dissipation

the Spectre circuit simulator with a custom MITLL SFQ5ee PDK.

4.2.3 Scaling

The bit depth of an SR-Loop cell is set by the length of the shift-register feedback loop, but it should not be increased indefinitely. Because the loop is actively clocked, every buffer switches once per cycle whether or not its bit is being accessed, and the read latency grows with the loop length. This makes the per-bit energy of an SR-Loop scale super-linearly with bit depth, in contrast to passive superconducting storage where energy could be dissipated only on access. The crossover between active and passive storage sets a practical upper bound on SR-Loop bit depth.

We can estimate this crossover with an order-of-magnitude comparison against SFQ-based passive storage, such as NDRO register files [81] or delay-line memories from passive transmission lines [23]. In these implementations, a persistent current loop stores a single bit per fluxon. Assuming a single SFQ write costs approximately $\Phi_0 I_c \approx 1 \times 10^{-19}$ J for $I_c = 50\ \mu$A, the SFQ storage energy is linear in bit depth x:

$$E_{\text{SFQ}} = x \cdot 100 \text{ zJ}. \tag{4.1}$$

The SR-Loop energy per access scales differently, because the read latency is equal to the bit depth (one bit per clock cycle) and every buffer in the loop switches each cycle. With a four-phase clock and 1.4 zJ/cycle per QFP [12], the SR-Loop energy is quadratic in bit depth:

$$E_{\text{AQFP}} = x \cdot (4 \text{ QFP/bit} \cdot 1.4 \text{ zJ/cycle/QFP} \cdot x \text{ cycle}) = 5.6\, x^2 \text{ zJ}. \tag{4.2}$$

Setting $E_{\text{AQFP}} = E_{\text{SFQ}}$ gives a crossover at $x \approx 18$ bits. Below this depth, the active SR-Loop is more energy-efficient; beyond it, a passive SFQ storage cell could be favorable.

4.3 SR-Loop Register File

This section presents two architectures for organizing SR-Loop memory cells into a register file (RF). The first version (v1) follows a row-column addressable array, while the second version (v2) uses a bit-sliced organization with SR-Loop output tapped at the middle of the loop and a denser design with direct-coupled AQFPs. We compare both designs against the D-latch register file used in the MANA microprocessor.

4.3.1 The Original Design

The first implementation of a register file with SR-Loop storage cells organized the cells in a row-column addressable array. We designed, simulated, and fabricated a 4×4 array of the 8-bit SR-Loop storage cells. We will refer to SR-Loop register files as having an $N \times M \times K$-bit configuration, where N is the width, M is the depth, and K is the storage loop depth. Therefore, our v1 design is a $4 \times 4 \times 8$-bit register file[1], with the organization shown in Figure 4.5.

The register file is organized to store 4-bit words. The four columns of the array encode the four bit positions of a 4-bit word. The 8-bit storage loop stores eight successive copies of that bit position with a new value broadcast on the loop output every cycle. On any given clock cycle, the four rows are sending four complete words to the readout mergers. Over the loop's 8-cycle rotation period, all 32 stored words (4 rows $\times$ 8 time slots) are exposed in sequence. Although the register file holds 32 words, only the 4 rows are directly addressable on any given cycle, so the write/read address is only 2 bits wide. The register file is designed with two readout ports to support two-operand instructions.

Write addressing uses a single decoder to select which word to write into. To write a 4-bit value, the S and R signals are driven across the four columns simultaneously, with each column receiving the S and R pair appropriate for its bit position. For example, to write `b1010` into word 2, we set $S =$ `b1010`, $R =$ `b0101`, and $W =$ `b10`.

The register file has two readout ports, QA and QB, addressed independently by RA and RB using the same 2-bit encoding as W. The selected word's bit positions are propagated through a tree of OR mergers to produce the 4-bit output word on the corresponding port.

[1]Note that 8-bit does not refer to the word length of the register file, unless the words are stored sequentially in a single storage loop, but that's not the case in our design, which holds 4b words

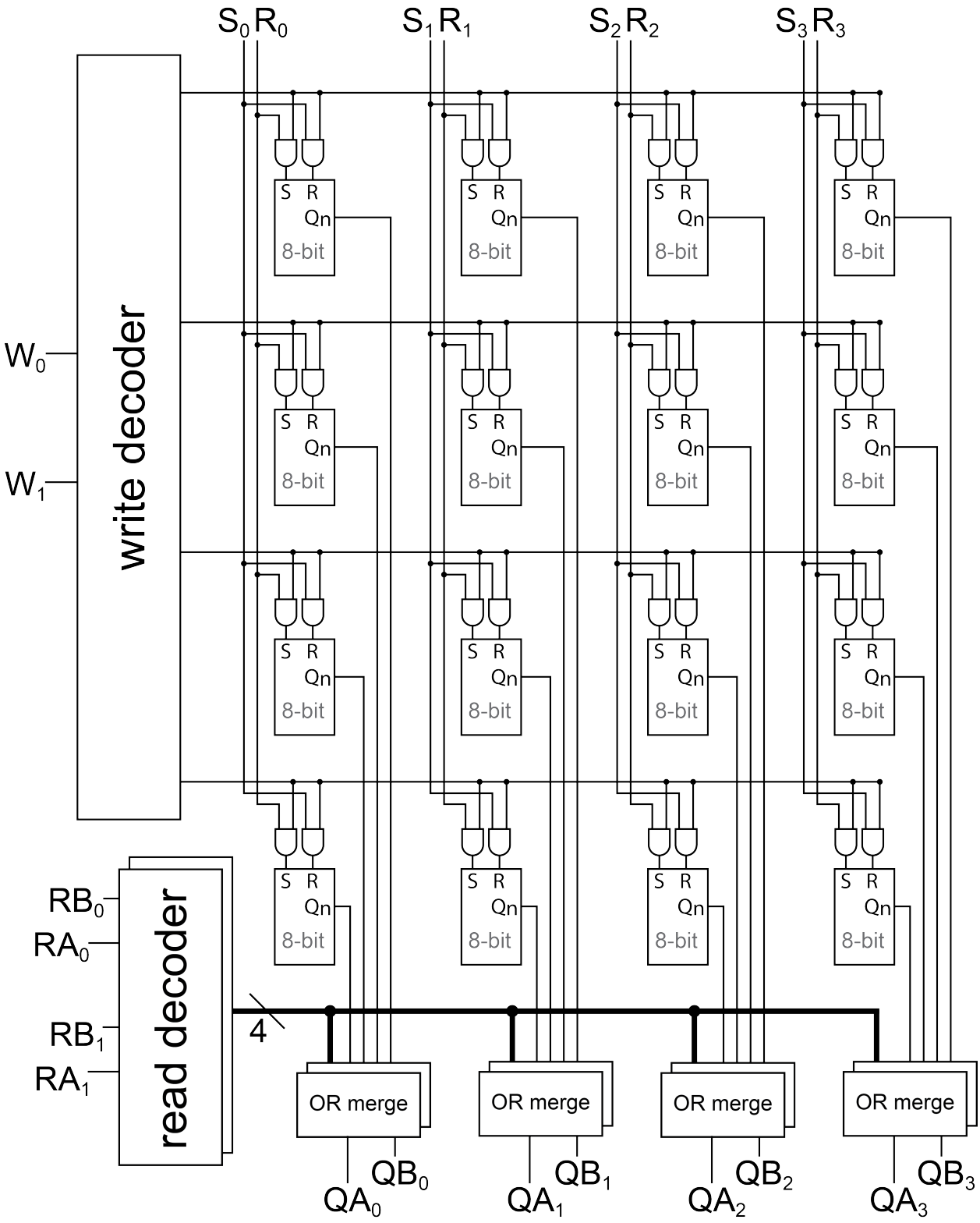


Figure 4.5: Block diagram of SR-Loop register file v1. The 8-bit storage cells are organized into a 4 × array with row and column addressing shown. A breakout of the demux and decoder circuits are shown. The clock signals are omitted in the register file for clarity, but shown in gray on the demux and decoder schematic.

Figure 4.6 shows a Cadence SPICE simulation demonstrating write and read operations across every cell in the array. Color shading identifies the word address (blue = 00, orange = 01, green = 10, pink = 11), and bit positions are arranged left to right across the columns. The top row plots the S and R signals applied during the write phase, with each color band writing a different word into the array. The middle row plots the Q_n output of every SR-Loop cell; rows correspond to words 0–3 and columns to bit positions 0–3. The bit written into a given cell appears at its output 9 clock cycles later, reflecting the 8-bit storage loop's read latency plus decoder overhead, and then circulates around the loop indefinitely. Therefore, the written values can appear more than once on the cell's output and the subsequent rotations around the storage loop are indicated with a lighter shaded region in the figure. The bottom row plots the two readout ports. QA shows successive reads of word 0 (RA = 00, blue) and word 2 (RA = 10, green); QB shows reads of word 1 (RB = 01, orange) and word 3 (RB = 11, pink).

The simulated results in Figure 4.6 were generated prior to layout, from schematics that closely matched the diagram in Figure 4.5. The translation from that schematic to a physical layout proved substantially more involved than the equivalent step would be in CMOS. As emphasized throughout this chapter, AQFP buffers cannot drive long superconducting wires, and the wires themselves act as large inductors. Propagating data spatially across an addressable array therefore requires inserting buffer chains along every routing path, and the lengths of those chains must be matched so that bits traveling hundreds of micrometers arrive in the same clock phase as bits routed across a few micrometers. This means that each added buffer alters the timing budget seen by every parallel data path, and the schematic must be revisited to keep all phases aligned.

This had a direct impact on device count. The pre-layout schematic contained 3,152 Josephson junctions while the final layout contains 10,062. This is a roughly 3× inflation driven almost entirely by the buffer and timing-matching overhead that the schematic did not capture. The lesson is, in AQFP the buffer count and clock-phase assignment depend so strongly on physical routing distances that the schematic cannot be finalized independently of the layout. This is unlike a typical synthesized digital CMOS flow, where standard cell abstraction, automated buffer insertion, and post-layout timing closure decouple the two steps on a first pass. In AQFP, every inserted buffer consumes a clock cycle, so layout-driven buffer insertion changes the logical timing of the design rather than perturbing it, and the schematic must be revised accordingly. Therefore, design flow must instead alternate between the two, with the schematic continuously updated to reflect the buffers and routing delays imposed by a realistic layout. We refer to this as layout-aware schematic design.

The final v1 layout is shown in Figure 4.7. The footprint is 2.1 mm × 3.5 mm (7.35 mm^2),

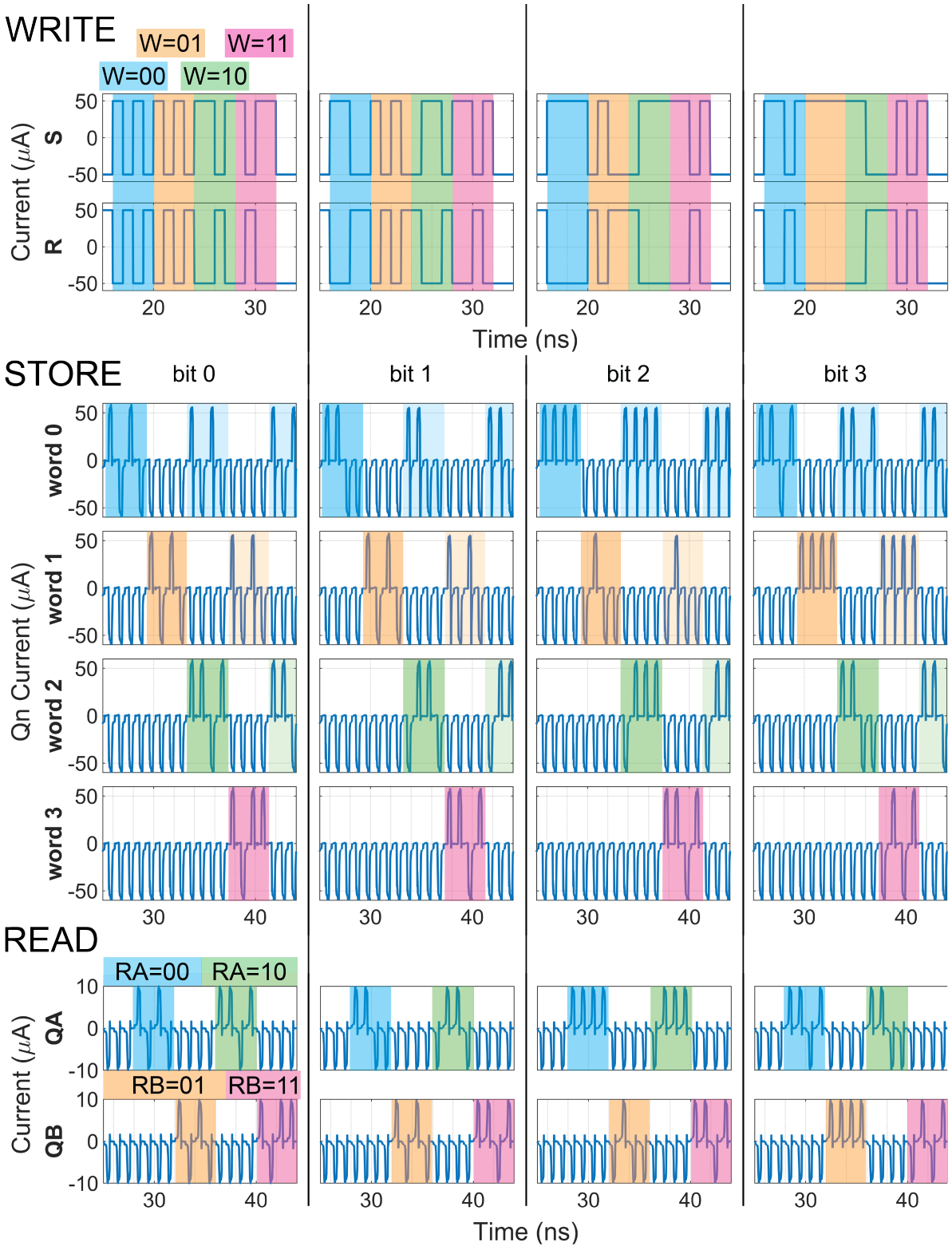


Figure 4.6: Cadence SPICE simulation of the v1 SR-Loop register file at 5 GHz. Top: write data signals S and R applied to each of the four bit columns. Middle: Q_n output of every SR-Loop cell in the 4×4 array (rows = words, columns = bit positions). Bottom: data on the two readout ports, QA and QB. Color shading identifies the word address being written or read: blue = 00, orange = 01, green = 10, pink = 11.

and the storage cells occupy a small fraction of that area with the remainder dedicated to routing and addressing. The design uses the most compact 8-bit storage cell variant, Figure 4.3(c). As discussed in Section 4.2.2.2, that variant did not operate correctly in measurement due to clock-to-data mutual coupling, so, although the v1 register file has not yet been tested, it is not expected to operate correctly either. The compact variant was selected because the v1 layout was submitted for fabrication before the chip containing the storage cell variants had returned from fabrication for testing. We committed to a storage cell whose functional status was unknown at submission because the tape-out timelines prevented a closed-loop design cycle.

Despite the fact that the design is not expected to function as a complete register file, v1 was instructive as a first attempt at a large-scale AQFP circuit. It was placed almost entirely by hand, without the aid of automated place-and-route tools, and the experience exposed how strongly the physical routing constraints feed back into the schematic. Additionally, simple breakout circuits from the design, remain to be characterized and may produce useful results once tested. However, the principal takeaway was a much clearer understanding of how layout constraints couple back into the register file architecture, which directly motivated the v2 design discussed in the next section.

4.3.2 An Improved Design

The lessons learned from v1 motivated a second version of the register file. The v2 design is 5.4× smaller in area than v1 while holding 8× more bits, resulting in a 44× improvement in effective bit density. Three modifications enable this improvement: an architectural change to where the readout port sits within the storage loop, a device-level swap to a newer AQFP cell library, and a improved, denser design of the SR-Loop storage cell.

In v2, the storage cell readout is relocated from the end of the shift-register feedback loop to its midpoint. Writes still enter at the start of the loop, but the read now sits halfway around. This splits the loop into two equal-length halves, allowing the loop to be folded as a hairpin with two 180-degree turns rather than snaking around a central area as in v1. The resulting cell is long and narrow rather than square, and write and read addressing can be separated onto opposite sides of the loop rather than crowded at a single point. Additionally, the storage cells in v2 are organized as a single row of bit-sliced columns rather than a 4×4 array of independent cells, so that address values no longer need to be dispatched across a two-dimensional array. The v2 architecture is shown in Figure 4.8.

The bit-slicing organization groups bits at the same bit position across all words into a single slice. On the write path, a single demux routes each incoming bit to the appropriate

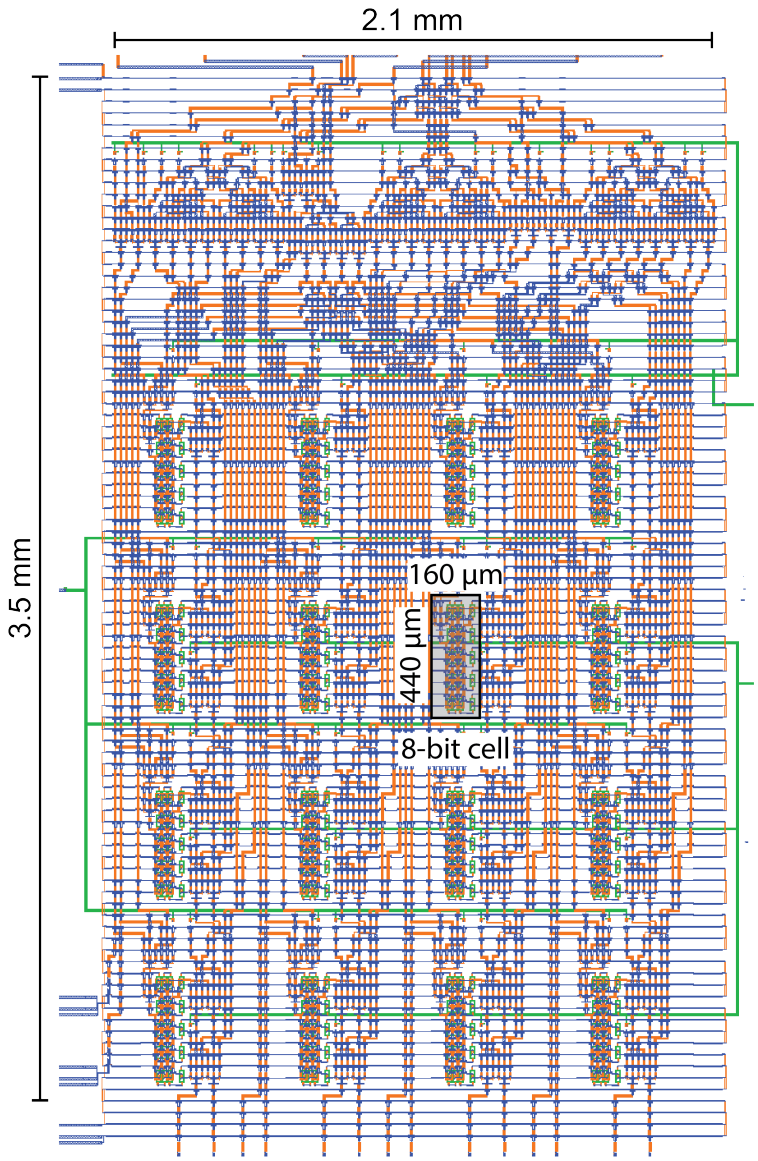


Figure 4.7: GDS layout of the v1 SR-Loop register file. The 8-bit storage cell is small relative to the full array; most of the area is taken up by routing and addressing overhead. The total JJ count is 10,062, placed almost entirely by hand, and the footprint is 2.1 mm × 3.5 mm (7.35 mm^2).

slice. On the read path, it is more space-efficient to select the word within each slice rather than rejoin the slices into a single decoder. Therefore, so the read address is fanned out to every slice and a copy of the word-select decoder is instantiated in each one. The v2 register file is configured as a 4 × 4 × 64-bit array, meaning there are four 64-bit loops per slice and four slices in total. This organization scales cleanly in two directions: (i) the datawidth can be increased by adding more slices, and (ii) the depth can be increased by adding more loops to each slice.

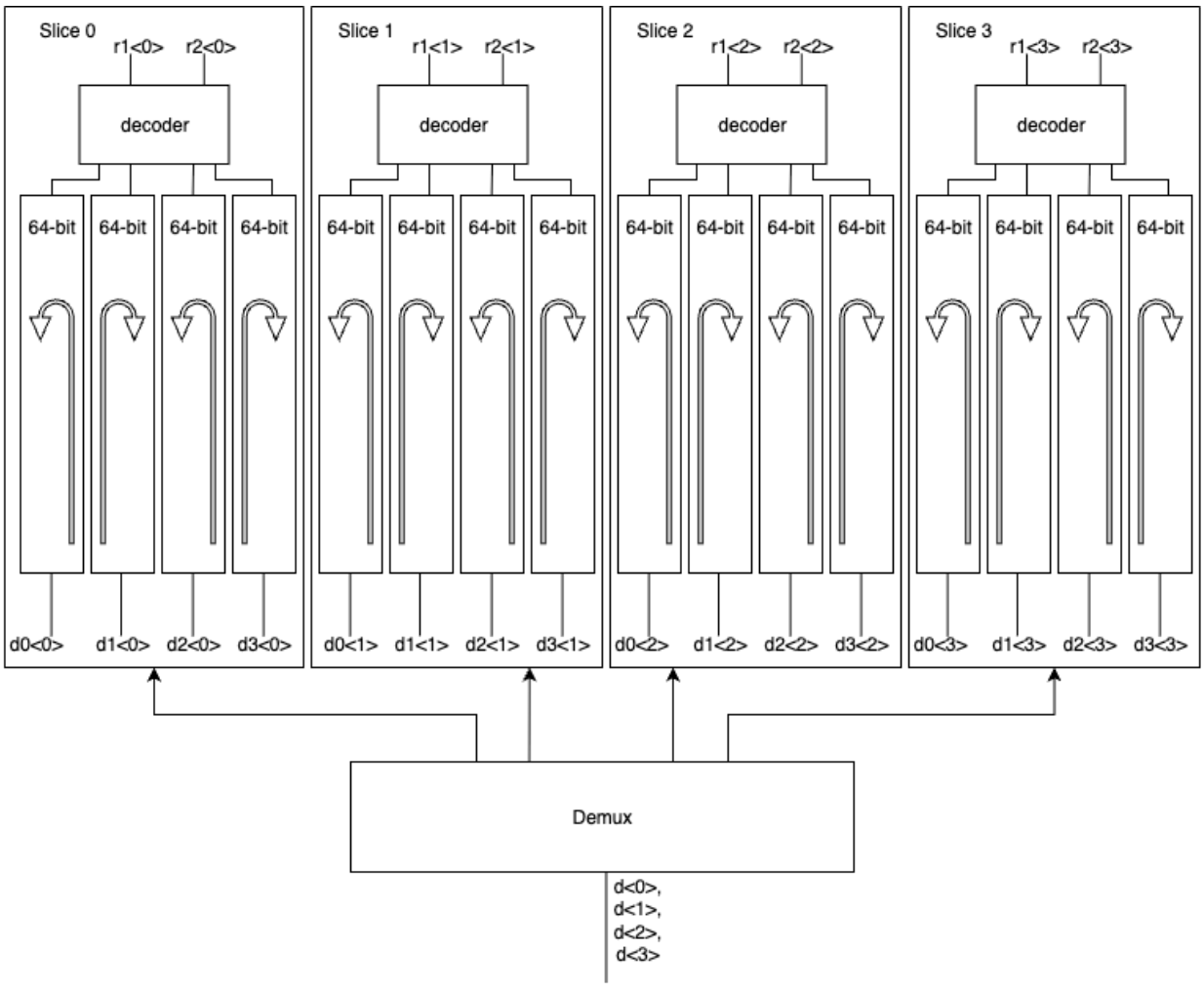


Figure 4.8: Architecture of the v2 SR-Loop register file, 4 × 4 × 64-bit configuration with two readout ports (r1 and r2). Each slice holds one bit position across all stored words, with the number of feedback loops per slice setting the register file depth and the number of slices setting the datawidth. Bit positions are indicated with `<i>`.

The second modification is at the device level. In the time between v1 and v2 of the register file, a new version of the MITLL AQFP cell library was developed by David Russo and Andrew Wagner. The v2 library shrunk the standard AQFP devices from a 20 μm by 20 μm footprint to 8.2 μm by 27.5 μm, through denser routing and a new ground tile design. Therefore, providing a 1.8× shrink in area purely by switching device libraries.

A further modification was made in the design of the storage cell itself. The buffers in the shift-register feedback loop are replaced with directly coupled AQFPs[2]. Directly coupled AQFPs eliminate the output transformer of the standard AQFP cell and replace it with a T-inductive network that galvanically couples neighboring devices [82]. Removing the

[2]The directly coupled AQFP for the MITLL AQFP cell library were designed by David Russo

transformer reduces the buffer footprint by a factor of 4.6×: the standard AQFP buffer in the v2 library is 8.2 μm by 27.5 μm, while the directly coupled AQFP is 12.8 μm by 3.8 μm.

The denser storage loop also permitted a larger per-loop capacity. The v2 register file holds 1024 bits (1 Kib) in a 4 × 4 × 64-bit configuration. It achieves this at roughly the same JJ count as v1 (10,774 vs. 10,062), so the per-cycle energy dissipation is comparable between the two designs even though the storage capacity has grown by 8×. As in the first version, the design has two readout ports to support two-operand instructions.

The design has recently returned from fabrication and will be measured shortly, with better prospects for functional operation than v1.

4.3.3 Summary of Metrics

Table 4.3 summarizes the simulated metrics for the v1 and v2 SR-Loop register files alongside the D-latch based register file used in the MANA microprocessor [12].

The SR-Loop register file is substantially more compact than the D-latch design. The effective bit density includes the area of the addressing and routing overhead, while the raw bit density reflects only the storage cells. The raw bit density for the MANA register file is not reported in the published literature. On effective bit density, v2 achieves a 380× improvement over the MANA register file.

The energy dissipation value reported in MANA was extrapolated from the 1.4 aJ per JJ per clock cycle measured value [8] (11.39 fJ/op / 8142 JJs = 1.4 aJ / JJ). We report energy values for the SR-Loop register file the same way, assuming that our AQFP cell library dissipates similarly. All energy values include a 1000 W/W cryogenic cooling overhead to be reported at the room-temperature equivalent. Note that this is a conservative upper-bound assumption for the SR-Loop register file, since our simulated per-device energy is roughly 4× lower than the measured value. Under this assumption, the SR-Loop register files dissipate more energy per cycle than MANA simply because they have more JJs. However, when normalized for storage capacity (energy per bit), the v2 SR-Loop register file is 12× more energy-efficient per bit than the MANA register file.

The principal drawback of the SR-Loop design is sequential access. Unlike a conventional addressable array, a read from an SR-Loop cell requires waiting for the requested bit to circulate to the readout port. This adds a variable latency of up to one loop depth on top of the addressing latency reported in the table, which in the worst case for v2 would be 64 cycles. The 7-cycle read latency in Table 4.3 reflects only the addressing path and not this sequential access delay. This penalty is manageable when memory access patterns are predictable or can be hidden by a well-matched microarchitecture, as developed in subsequent

chapters. For general-purpose processors or workloads with irregular access patterns, the SR-Loop approach is less well suited.

Table 4.3: Summary of metrics for v1 and v2 implementations of the SR-Loop register file and comparison to MANA microprocessor register file.

	v1	**v2**	**MANA RF** [12]
Capacity (bits)	128	1,024	64
Configuration	4 × 4 × 8-bit	4 × 4 × 64-bit	16 × 4b
Number of readout ports	2	2	2
JJ count	10,062	10,774	8,142
Area (mm^2)	7.35 (2.1 × 3.5 mm)	1.35 (0.65 × 2.07 mm)	31.0 (4.7 × 6.6 mm)
Effective bit density (bit/mm^2)	17.4	760	2
Raw bit density (bit/mm^2)	113	1,777	–
Read latency†(clock cycles)	12	7	8
Energy‡per cycle @ 5 GHz (fJ)	14.1	15.1	11.40
Energy‡per bit (aJ)	110	14.7	178

† delay between issuing a read request to receiving the data, does not account for sequential access delay on SR-Loop storage cells.
‡ includes 1000 W/W cooling overhead for room temperature energy dissipation

4.4 Outlook

Two key details require future work. The 8-bit storage cell has only been demonstrated in its least compact layout variant; the two denser variants failed in measurement and the v1 register file inherits that failure. The revised storage cell layouts and the dedicated clock-to-data coupling test structure described in Section 4.2.2.2 are the immediate next experimental steps. Additionally, it's clear that implementing parasitic extraction tools to the AQFP simulation workflow before a tapeout at this density would be beneficial in the future.

On the architecture side, the v2 register file has returned from fabrication and characterization is in progress; the metrics reported in Table 4.3 remain simulated until that measurement closes. Demonstrating an SR-Loop memory at intermediate scale, between the 1 Kb register file fabricated here and the Mb-class on-chip memory this design is projected to support in Chapter 6, is the additional experimental work needed to validate these projections for larger scale system design.

The primary contribution of this chapter is a register file primitive that departs from the CMOS-derived random-access organization that has shaped prior superconducting memory designs. The SR-Loop register file achieves a compact design by embracing the propagation-native nature of AQFP and storing data as bits circulating in a shift-register feedback loop. Compared to the D-latch register file in the MANA microprocessor, the SR-Loop register file has a 380× larger effective bit density at comparable read latency and a 12× reduction in energy per bit.

The cost of this density is sequential access. A read incurs a variable latency of up to one loop depth on top of the addressing path, which makes the SR-Loop a poor match for workloads with irregular access patterns and general-purpose processors. The bet of the design, and the premise that the rest of this dissertation tests, is that this access penalty can be absorbed by a microarchitecture that schedules computation around the storage's natural circulation period rather than fighting it. The chapters that follow develop that argument.

Chapter 5

LinCore: High throughput linear algebra accelerator

In this chapter, we introduce LinCore, a high throughput linear algebra accelerator for AQFP.

In Section 5.1, we motivate the architectural design by examining the constraints AQFP imposes on conventional pipelined microarchitectures, using the MANA microprocessor as an example, and we introduce fine-grained multithreading and the barrel-processor scheduling pattern as the architectural response. In Section 5.2, we describe the LinCore microarchitecture: a closed-loop execution-storage co-design that combines barrel scheduling with the AQFP SR-Loop register file (Chapter 4), and we analyze how the placement of the read port along the storage loop produces two scheduling variants, v1 and v2. In Section 5.3, we describe the v1 implementation, which has been fabricated but is unlikely to function as designed due to defects identified after tape-out. The v2 implementation is in active development at the time of writing and is not described in detail in this dissertation; a full account, including measured results, will appear in forthcoming publications. We close with future directions in Section 5.4.

I would like to acknowledge Alex Wynn for his design of the ALU component, David Russo for contributing to the v2 design improvements, and both of them for numerous discussions on scheduling patterns.

5.1 Motivation and Background

5.1.1 Propagation-native architecture

AQFP departs from CMOS in a way that is fundamental to architecture, not just to circuit design. In CMOS, data is naturally stored: logical values are encoded as voltages on transistor

gates, driven by a constant power supply, they remain there until overwritten. Computation is layered into this storage with a clock signal that switches transistors when new operands are ready. AQFP has no analogous resting state. A logical value exists only as a current pulse propagating through a chain of buffers under a multiphase excitation signal that behaves as both power and clock. Data is always in motion, and storage is achieved only by continued propagation. Computation occurs on top of this propagation as the data passes through majority gates and inverters.

The consequence is that architectures imported directly from CMOS will leave the deeply pipelined AQFP circuits idle most of the time as data are shuttled through the compute. And importantly, an idle AQFP still incurs an energy cost. Because clock is also power, every AQFP device dissipates energy on every cycle, whether or not useful work is being executed. The MANA microprocessor, the largest AQFP microprocessor demonstrated to date [12], illustrates the cost. MANA is a hybrid RISC-dataflow design, intended as a proof of concept rather than a throughput-optimized implementation. From a control perspective it has two pipeline stages, because all scheduling decisions are made and buffered in the first stage: (1) the instruction buffer-decode-issue (IDI) stage, which fits within a single clock cycle (four logic operations under the four-phase clock); and (2) the rest of the datapath through the register file (RFX), execution (EX), and writeback (WB) stages, which together incur a 26-cycle latency. In theory, the AQFP gate-level clocking allows the second stage to be pipelined such that 26 instructions could be executed "in-flight" [12]. Achieving that throughput requires the in-flight instructions to be free of data dependencies, and even then the four-word instruction buffer in the MANA demonstration caps the burst length at four. In practice, the demonstrated test programs all contained either a data or control hazard, and a 27-cycle hardware stall was inserted after every instruction. The reported energy of 30 fJ/op (including a 1000× cooling overhead) assumes the architecture's intended IPC of 1 in burst mode, computed as 21,460 junctions × 1.4 zJ per junction switching energy at 5 GHz. However, under the demonstrated execution pattern, with the 27-cycle stall applied to every instruction, the effective energy per useful operation in those programs is 27× higher, or 810 fJ/op. While MANA was a pivotal proof of concept to inspire the field of AQFP computing, the authors themselves note that under four-phase clocking it is "extremely difficult or perhaps impossible to implement hazard resolution techniques to make the overall pipeline stall free" [12].

Stepping back from MANA's specific design choices, the lesson is that fitting AQFP into the storage-native shape of a CMOS pipeline forces the architecture to spend most of its clock cycles shuttling data rather than computing. To use AQFP efficiently, the architecture must accept that data is always propagating and arrange for useful computation to happen

during that propagation rather than waiting for it to arrive. A fine-grained multithreaded microarchitecture matches this constraint directly.

5.1.2 Fine-grained multithreading and barrel processors

Fine-grained multithreading is a microarchitectural pattern for exploiting thread-level parallelism by switching between hardware threads every clock cycle, resulting in a temporal interleaving of instructions from independent threads [83]. Because consecutive instructions in the pipeline come from independent threads, they cannot have register dependencies on one another, and the hazard detection and forwarding logic that a conventional pipelined processor requires can be removed entirely. The pattern was introduced in the 1980s for parallelizable supercomputing applications, where it was used to hide memory latency. HEP (Heterogeneous Element Processor) [84], Horizon [85], and its successor, the Tera Computer System [86], all achieved high throughput for their time by "switching context (instruction stream) each machine cycle" [85]. The same mechanism has since become standard in graphics processors: each NVIDIA streaming multiprocessor maintains many resident warps and switches between them on a per-cycle basis to hide both arithmetic and memory latency [87].

A *barrel processor* is a strict form of fine-grained multithreading in which each clock cycle issues the next thread in a fixed cyclical order [88]. A barrel processor does not allow for multiple instructions to be issued in the same clock cycle (simultaneous multithreading), so its scheduling logic is trivial and typically implemented with just a counter modulo the thread count. The scheduler and architectural simplicity is a main motivating factor for barrel processors. The pattern dates to the peripheral processors of the CDC 6600, where ten threads shared a ten-stage pipeline in fixed rotation [89], and was used in HEP and Tera at the single-processor level beneath the more general fine-grained multithreading those systems exposed to software [86]. More recent examples include the XMOS XS1, an embedded barrel processor whose deterministic per-thread timing is exploited for real-time I/O processing [90], and the RISC-V barrel processor of AskariHemmat et al., which serves as a tightly coupled controller for an array of DNN processing elements [88].

These properties are valuable in CMOS, where the cost of barrel scheduling is throughput per thread traded for simplicity and predictability. In AQFP circuits, as the next section develops, the barrel scheduling pattern becomes attractive because it eliminates the idle pipeline cycles that would otherwise dissipate energy without producing useful work. LinCore borrows this scheduling pattern from the barrel processor lineage but applies it to a fixed-function linear algebra accelerator rather than a general-purpose CPU.

5.2 LinCore microarchitecture

LinCore is a fine-grained multithreaded linear algebra accelerator that uses strict round-robin scheduling to share a single datapath across N hardware threads. The microarchitecture is designed to keep the AQFP datapath fully occupied by embracing the gate-level clocking and propagation-native nature of the logic. The intended workload is matrix multiplication, which dominates AI inference and training, but the ISA also supports more general elementwise and reduction operations over the register file. We begin with an overview of the architecture, parameterized to describe both versions, and then discuss the improvements made from v1 to v2.

5.2.1 Closed-loop execution-storage design

LinCore is composed of two main datapaths: the **execution path**, which has an ALU with an adder and a multiplier to perform basic linear algebra operations; and the **storage path** which is the AQFP SR-Loop register file (see Chapter 4). Computation is performed by propagating data in a continuous loop from execution path to storage path. LinCore borrows the strict round-robin scheduling method from the barrel processor lineage [84],[86],[88],[89], but applies it to a fixed-function accelerator rather than a general-purpose CPU. Unlike these previous implementations, the LinCore processor has a cyclical serial memory. The SR-Loop register file combines addressable data storage with long feedback-loop shift registers, so the storage itself is pipelined with a rotating access pattern. This is similar to one of the first digital computers, EDSAC from 1948, that used mercury-based delay-line memories. However, in this early design a more traditional control pipeline stage was timed to the delay of the mercury tubes and the computer was limited in performance by the latency of the delay lines. Blending delay-line memory with barrel processor scheduling, the LinCore accelerator realizes an architecture method that can hide the compute and memory latency of deeply pipelined AQFP circuits.

LinCore is hardcoded to support N hardware threads, where N is the number of cycles for an instruction to fully complete. N is set by the length of the shift-register feedback loop in the SR-Loop register file. We refer to the cycle-level pipeline stage within a datapath as a "slot". Slots are stationary positions along the datapath, while hardware threads propagate through them. The execution path is co-designed so that the number of slots aligns with the correct read and write points along the storage loop. The result is a closed-loop datapath where round-robin scheduling is a result of the storage path's natural circulation period, rather than requiring additional scheduling or control logic.

Figure 5.1 shows a frame-by-frame snapshot of the LinCore dataflow. Threads propagate

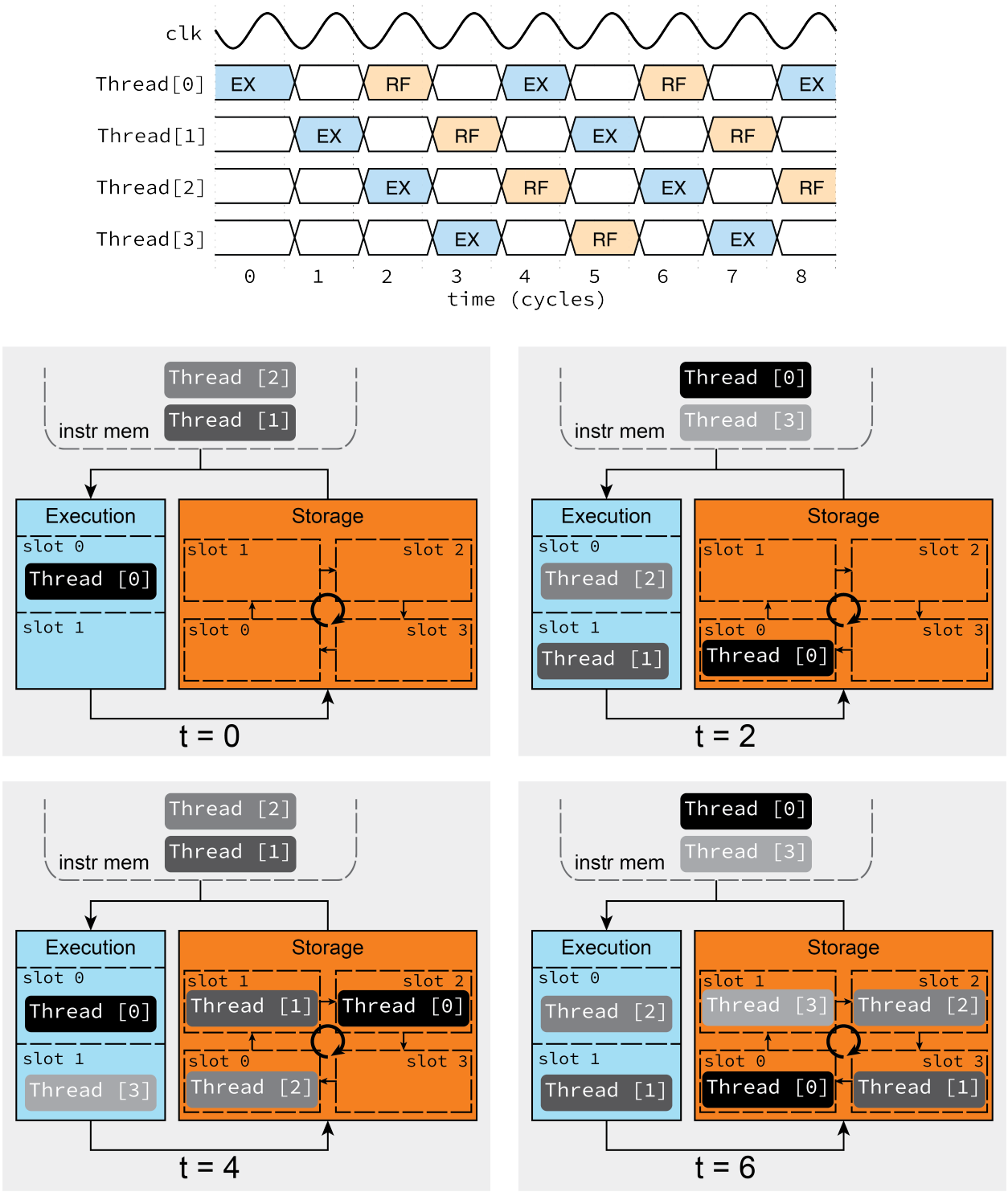


Figure 5.1: Frame-by-frame walkthrough of LinCore dataflow for a four-thread instance. The top timing diagram shows the schedule of execution (EX) and register file (RF) operations issued in the execution and storage paths, respectively. After two clock cycles, the pipeline is full, and an EX and RF operation are issued on every cycle. The bottom panels show where each thread is located along the execution path and storage loop at successive timesteps. Threads rotate counter clockwise through the outer datapath, and rotate clockwise within the storage loop.

Table 5.1: LinCore instruction set.

Instruction	Format	Description
ADD	`OPCODE[2:0] DST[1:0] RA[1:0] RB[1:0]`	DST = RA + RB
MUL	`OPCODE[2:0] DST[1:0] RA[1:0] RB[1:0]`	DST = RA × RB
LI	`OPCODE[2:0] DST[1:0] IMM[3:0]`	Load immediate
NOP	—	no operation, stall

counter-clockwise from instruction memory to the execution path to the storage path and so on. Within the storage path, threads circulate clockwise through the feedback loop. In this example, the storage loop has a depth of four clock cycles, meaning there are four slots that can each hold a thread. The architecture therefore supports four hardware threads running simultaneously. The execution path has a pipeline depth of two ($N/2$), because we've assumed the read port of the SR-Loop register file is at the midpoint of the storage loop, as in the v2 design (Section 4.3.2).

From LinCore's perspective, once all slots are filled, a single instruction completes every cycle. From the perspective of a single hardware thread, an instruction completes once every N cycles. The timing diagram at the top of Figure 5.1 makes this concrete. Each colored clock period marks when an execution (EX) or register file (RF) operation is issued for a given thread. The unshaded cycles correspond to moments when the operation's data is in flight through the datapath and, therefore, inaccessible. Without multithreading, these unshaded cycles would be pipeline stalls. Once the execution pipeline is full, after $N/2$ cycles in this case, an execution and a storage operation are issued on every clock cycle.

Because the slots of the storage loop are stationary while the threads propagate through them, thread i's register state is at the storage path's read/write ports during cycle $i \bmod N$, and only during that cycle. Threads are therefore physically prevented from accessing each other's register state, since a thread's registers are not present at the read/write ports during another thread's slot. Inter-thread communication, if needed, must be arranged through external memory and is not accounted for in this early design.

5.2.2 Datapath components and instruction set

We describe two versions of a 4-bit LinCore accelerator: v1 has been fabricated; v2 is in active development at the time of writing. We have designed and simulated two versions of a 4-bit LinCore accelerator. Figure 5.2 shows a block-level diagram representative of both versions. LinCore runs on a 9-bit instruction in the `op dest src src` format, with `src` replaced for `imm` in the load immediate case. The four supported instructions are listed in Table 5.1.

The execution path is comprises four blocks:

The **instruction decoder** receives the full instruction sequence from an off-chip signal and splits it into the opcode, destination address, source register addresses, and immediate value.

The **ALU** consists of an FP4 adder and an FP4 multiplier on separate paths with the opcode selecting which operation produces the result. The current implementation does not have a fused multiply-accumulate (MAC) primitive, so a MAC operation requires two instruction cycles, one each for the multiply and the add. This is why we refer to LinCore as a linear algebra accelerator, rather than specifically a matrix-multiply accelerator. However, adding a fused MAC unit would easily fit into the LinCore microarchitecture and is a straightforward design task left to future work.

A **write-only buffer**, implemented as a 4-bit wide chain of AQFP buffers, carries the immediate value along the execution path with the correct timing for write-immediate (`LI`) operations.

Finally, a **merge** on the output selects between the ALU result and the immediate value and forwards a single 4-bit result to the storage path.

The storage path is the AQFP SR-Loop register file: a 4×4×8-bit (128 b) register file in v1 (Section **??**) and a 4×4×64-bit (1 Kb) register file in v2 (Section 4.3.2). Both versions store four 4-bit words organized as a 4×4 array; the difference is the loop depth, which grows from 8 bits in v1 to 64 bits in v2. Each hardware thread therefore has four addressable registers. From the LinCore scheduling perspective, the primary difference between the two register file versions is the placement of the read port along the feedback loop.

5.2.3 Read port placement and pipeline timing

When an instruction enters the pipeline, the full instruction word is routed to the execution path and the source register addresses are dispatched to the register file. For the operands read from the register file to arrive at the ALU on the cycle that the instruction decoder issues the corresponding opcode, the instruction decoder's path must be buffered to match the read latency of the storage path. The operands read in the first slot of the execution path therefore come from the thread that occupied the read slot of the storage path one cycle earlier. To put this plainly, we will assume the readout port for the storage path in Figure 5.1 is at `slot 1`. Then, at $t = 6$ `Thread [2]` occupies `slot 0` of the execution path and is carrying the operands that were read on its behalf from storage `slot 1` at $t = 5$. In

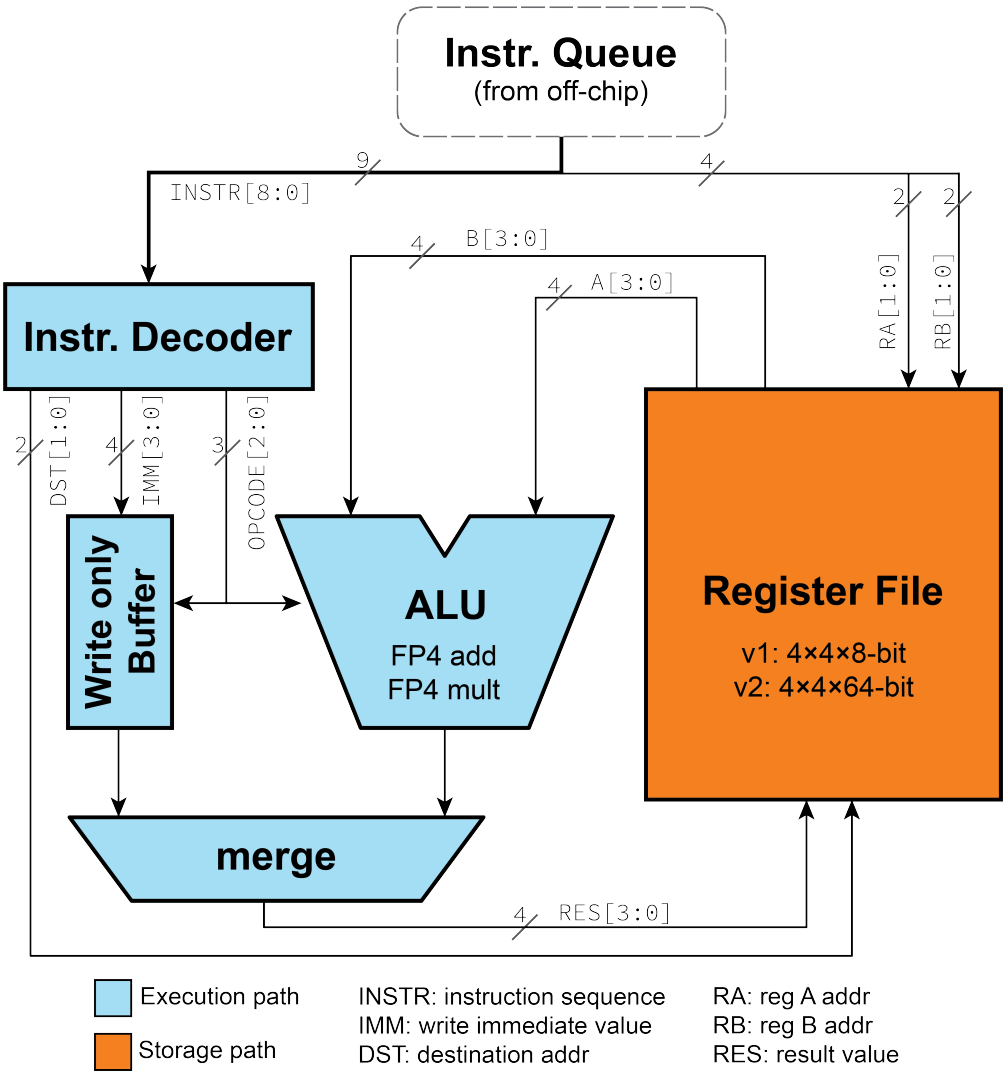


Figure 5.2: Block diagram of the LinCore implementation. The execution path (blue) and storage path (orange) form a closed loop. Instructions arrive from off-chip instruction memory and are routed to the necessary components by the instruction decoder. The ALU produces a 4-bit result via either FP4 addition or multiplication; the merge unit selects between the ALU output and the write-only buffer carrying the immediate value, forwarding the chosen result (`RES`) to the register file. The register file is configured as $4 \times 4 \times 8$-bit in v1 and $4 \times 4 \times 64$-bit in v2.

the timing diagram, the RF label marks the cycle on which a write operation is issued to the storage path. The read operation is not labeled and is assumed to always be issued one cycle before the execution operation (EX).

In the v1 register file, the read and write ports occupy the same slot of the storage loop. Every time a thread reaches the register file port, it is simultaneously written to and read from. The execution-path and storage-path operations are therefore no longer interleaved as in

Figure 5.1. Instead, the storage-path write must be issued one cycle before the execution-path operation, since the simultaneous read at that slot must occur in time to deliver operands to the ALU. The pipeline depth of the execution path is correspondingly N, rather than $N/2$, to match the delay between read and write.

Figure 5.3 shows a digital timing diagram for a four-thread instance of LinCore v1. The color coding is the same as in Figure 5.1, blue for the execution path and orange for the storage path, but the slot labels are replaced with the identity of the instruction (A, B, C, ...) being issued by each datapath. Taking `Thread[0]` as an example, instruction A is issued on the execution path at $t = 0$ and then written to the storage path at $t = 3$. Then, instruction B is immediately issued on the execution path at $t = 4$, but this is before A has completed a full rotation through the storage loop.

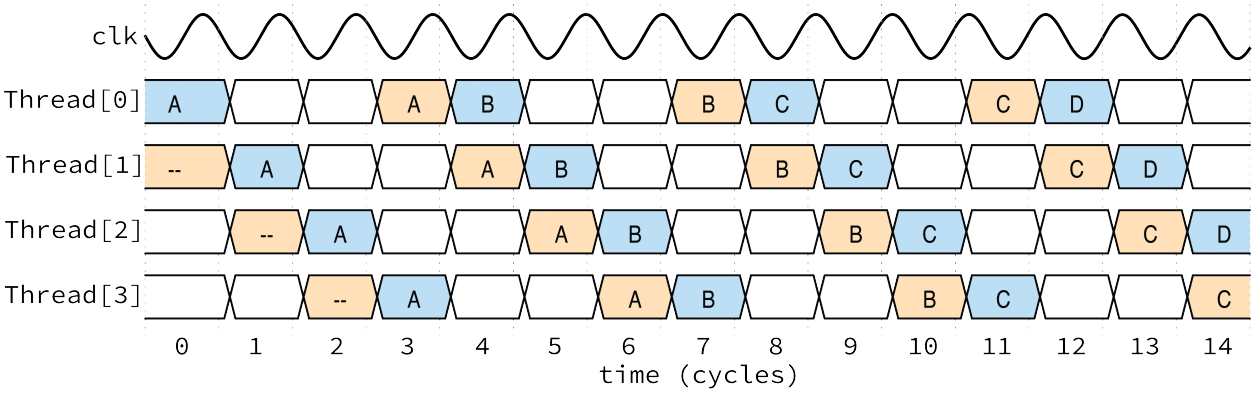


Figure 5.3: Timing diagram for a four-thread instance of the LinCore v1 design. The read and write ports of the register file now occupy the same slot of the storage loop. The color coding follows Figure 5.1, with blue marking execution path issues and orange marking storage path issues, but the slot labels are replaced with the identity of the instruction (A, B, C, ...) being issued by each datapath.

This overlap has a subtle but important consequence for single-thread instruction dependencies. The total round trip latency from a write issue to the resulting value being available at the read port is now $2N$, decomposed as N cycles through the execution path and N cycles around the storage loop. Data written by instruction A are therefore not available to be read until instruction C, two instructions later in the same thread's stream. Programs with back to back read-after-write (RAW) dependencies on a single thread incur an effective stall of one instruction issue, which must be filled with a `NOP`. Because all instructions in a thread share the same register addresses, we will not refer to this as introducing an additional hardware thread. Instead, the stall is a per-thread hazard that a compiler could mitigate by reordering instructions to separate dependent reads and writes (multithreading in the software). For matrix multiplication, the dominant compute pattern is the MAC sequence where a multiply is immediately followed by an add that consumes its result, exactly the RAW dependency

that triggers the v1 stall. Avoiding a stall on every MAC requires the compiler to interleave independent MACs, an intra-thread scheduling burden that motivates the v2 design.

The v2 design removes this hazard. The read port is moved to the midpoint of the storage loop, and the execution path and storage path return to the interleaved schedule of Figure 5.1. The execution path pipeline depth is $N/2$, matching the offset between the write port and the read port along the storage loop. The round trip latency from a write issue to read availability is therefore N, decomposed as $N/2$ cycles through the execution path and $N/2$ cycles from write to read on the storage path. Successive instructions in a single thread are spaced exactly N cycles apart, so the data written by one instruction is available to be read by the next, and the RAW hazard of v1 disappears.

5.3 Implementation

The four thread instances used in Figures 5.1 and 5.3 are illustrative, a physical implementation of LinCore will have a larger thread count to overcome the large latencies on large-scale AQFP circuits. This section describes an initial implementation of the LinCore processor, v1, which has been fabricated, but not yet tested. At the time of writing, development of an improved design is underway and we will discuss general design concepts for these improvements, but detailed specification, fabrication, and measurement will be reported in future work.

v1 implementation. The v1 LinCore accelerator has been fabricated and fits within a 5 $\times$ 5 mm die; its annotated layout is shown in Figure 5.4. The register file is a $4 \times 4 \times 8$-bit (128 b) SR-Loop, with read and write ports occupying the same slot of the storage loop. This is the same design described in Section 4.3.1. The ALU is a 4-bit floating-point unit in the E2M1 format, with separate adder and multiplier paths selected by the opcode. The execution path has a pipeline depth of $N = 64$, matched to combined latency of the 8-bit storage loop and addressing overhead. The design was implemented in v1 of the MIT LL AQFP standard cell library.

v1 defects. Two defects were identified after tape-out that make the fabricated v1 die unlikely to function as designed in measurement. First, the 8-bit storage cell used in the v1 register file exhibits coupling between data and clock signals that corrupts stored values; the failure mode is characterized in Section 4.2.2.2. Second, the 4×4 array layout introduces unequal routing distances between the active storage area and the read/write ports, since data exiting different rows on the same clock cycle must traverse different distances before reaching the merge with the execution path. The v1 design attempted to compensate for this

in layout by hopping data across five clock phases, but the same approach had already failed in the standalone 8-bit register file due to coupling and delay introduced at each crossing.

The underlying issue is that we followed a conventional schematic-to-layout design workflow, and the layout-aware buffers required for timing and splitting signals were inserted only after the schematic was complete, without sufficient modeling to verify the final layout-versus-schematic modifications. For AQFP at this density, this workflow is inadequate; layout-aware schematic design is needed to discover routing-driven timing constraints and place buffers concurrently with the schematic.

v2. The v2 design aims to address these defects. The register file is replaced with the improved design v2, as described in Section 4.3.2. In this version, the read and write ports are separated along the storage loop, with write at the start of the loop and read at the midpoint, which removes the v1 RAW hazard as described in Section 5.2.3. The ALU is mostly unchanged in the v2 design, however the execution path is padded with fewer timing buffers to match the $N/2$ delay from the register file. Additionally, the v2 LinCore is implemented in v2 of the MIT LL AQFP standard cell library, which reduces the underlying device footprint by 1.8×.

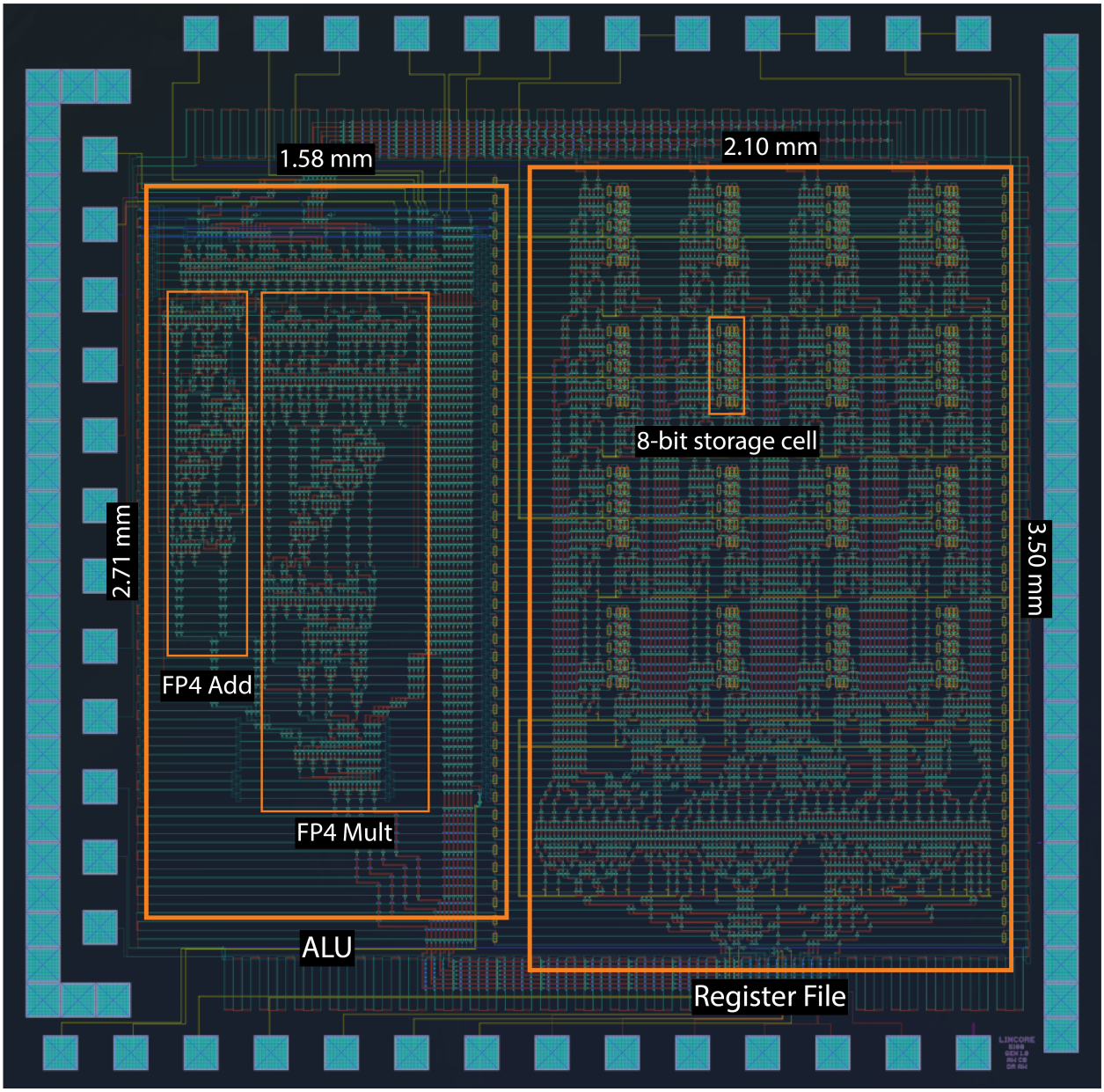


Figure 5.4: Annotated layout of LinCore v1 design. ALU on the left is a 4-bit floating-point unit in the E2M1 format, with separate adder and multiplier paths selected by the opcode. The register file on the right is the $4 \times 4 \times 8$-bit SR-Loop design described in Section 4.3.1.

5.4 Outlook

The immediate next step is bringing the v2 LinCore through simulation, fabrication, measurement. Beyond that, two directions extend naturally from the v1 design experience and the v2 register file pairing. First, the v1 layout exposed the limits of a conventional schematic to layout workflow for AQFP at this density. Developing a layout-aware schematic design

methodology, in which routing-driven timing constraints feed back into the schematic during design rather than after, is a prerequisite for scaling LinCore to deeper pipelines and larger thread counts. Second, the v1 ALU does not include a fused multiply-accumulate primitive, so each MAC operation consumes two instruction cycles. Adding a fused MAC unit fits cleanly into the LinCore microarchitecture and would roughly double throughput on the matrix multiplication workloads LinCore is designed for. Both directions are left to future work.

The primary contribution of this chapter is a microarchitecture that translates the propagation-native and gate-level clocked nature of AQFP into an asset rather than a constraint. By co-designing the execution path with the SR-Loop register file and applying strict round-robin scheduling, LinCore keeps the deeply pipelined AQFP datapath fully occupied without additional control logic. The closed-loop execution-storage organization and the barrel scheduled compute pattern are independent of the specific ALU functionality and could be extended to more general-purpose AQFP compute. The system-level energy and performance projections that follow in Chapter 6 build on the LinCore microarchitecture as the compute unit.

Chapter 6

Full-stack Architecture Modeling with AccelForge

This chapter develops the case for, and the realization of, a full-stack architecture modeling capability for superconducting accelerators. Section 6.1 motivates why such a tool is needed and outlines the scope and limits of what it can deliver. Section 6.2 reviews the analytical modeling lineage that the work builds on, covering AI workloads, accelerator organization, mapping, and the AccelForge framework that we leverage for superconductors. Section 6.3 describes how we fit superconducting electronics into AccelForge: the treatment of cryogenic cooling overhead and the component models developed for AQFP and RQL compute, cryogenic memory, and temperature-stage interconnects. Section 6.4 applies the tool in a sequence of design studies that move from a bare PE array, to a PE array with an on-chip global buffer, to a TPU-like memory hierarchy, and finally to a system model that includes the 4 K to room-temperature interconnect; it closes with a cryostat sizing check for the largest configuration evaluated.

The work discussed in this chapter was developed in collaboration with Tanner Andrulis. This chapter represents work in final preparation for publication and future readers interested in the authoritative reference should refer to the final published paper.

6.1 The need for full-stack architecture modeling in superconducting electronics

Early existence proofs and small fabricated subsystems of digital superconducting circuits show promise to rival or exceed CMOS in energy performance [12] and speed [91], but the path to get from these building blocks to a full system is less explored. Research in superconducting

electronics has historically concentrated at the device- and circuit-level, with comparatively little attention paid to the architectural and system-level questions that determine whether these advantages translate into useful computing applications. A few recent efforts have addressed this gap with workload-specific evaluations of superconducting digital architectures [92],[93], but each of these efforts target a single pre-determined architecture design.

A flexible and fast modeling tool is critical for evaluating architecture design decisions before investing time, fabrication cost, and engineering effort into end-to-end design. This type of design exploration tool would allow superconducting circuit designers to make informed decisions about component organizations, memory hierarchy, interconnect/packaging strategies, etc. at the earliest stage of design.

In this chapter, we will review how this type of analytical early-stage architecture modeling is done for tensor algebra accelerators implemented in traditional semiconductor technologies, leverage a state-of-the-art modeling framework, AccelForge [94], for superconducting systems, and explore early results with the tool. The goal of the tool described here is to provide research-grade estimates of full-stack performance for superconducting accelerators. Its component models are parameterized from SPICE simulations, published results, and measured results where available. However, the system-level projections it produces should not be interpreted as verified or benchmarked results because there is no superconducting system at the scale targeted by the design tool to date and therefore validation against measured data are not possible.

Within these limits, the tool is intended as a guide for designing the next generation of large-scale superconducting electronics. It is most useful for extracting trends and quantifying trade-offs between architecture components, identifying which design directions are worth pursuing, and flagging where device- or circuit-level advances would have the largest system-level impact. Establishing the absolute accuracy of system projections will require future verification against fabricated systems and is outside the scope of this dissertation.

6.2 Accelerator modeling basics

Architecture modeling for AI accelerators is a mature subfield in conventional semiconductor chip design, with established frameworks, workload conventions, and validation methodologies. The remainder of this chapter draws on that body of work, applying it to superconducting systems where no comparable tooling exists. This section sets up the necessary background: the structure of modern AI workloads, the organization of accelerators designed to run them, the role of mapping in evaluating a candidate design, and the analytical modeling framework that we leverage for superconducting accelerators in Section 6.3.

6.2.1 AI Workloads

Modern AI inference workloads are dominated by tensor algebra. Large language model (LLM) or deep neural network (DNN) layers include sequences of tensor operations, such as matrix multiplications, convolutions, and scalar operations. Systems running AI workloads spend most of their time and energy moving data for both transfer and storage, and performing computations which are often multiply-accumulates (MAC) operations.

To describe such workloads precisely, we adopt Einsum (Einstein summation) notation [95], as is common within the field [96],[97],[98]. In Einsum notation, a tensor algebra computation is expressed as an equation between indexed tensors, where summation is implied over any rank variable that appears more than once on the same side of the equation. For example, matrix multiplication $\boldsymbol{Z} = \boldsymbol{A} \times \boldsymbol{B}$ is written

$$Z_{m,n} = A_{m,k} \times B_{k,n} \tag{6.1}$$

where m, n, and k are indices for the corresponding ranks of the matrices. The contraction over k is implicit, given that k appears twice on the right side of the equation. From this, AI workloads can be described as a cascade of Einsums and data dependencies.

Workload selection is not incidental to architecture modeling, it is half of the problem. An accelerator is, by definition, a piece of hardware specialized to a particular class of computation, so the same architecture will look efficient or wasteful depending on what it is asked to run. Reporting performance numbers for an accelerator without specifying the workload conflates two independent design choices and makes comparison between architectures meaningless.

LLM workloads In our design studies, we will first focus on simple matrix-matrix multiply workloads to evaluate the modeling tool, and then later expand to LLM inference workload based on the GPT3 transformer architecture [99]. A transformer layer consists of a multi-head self-attention block followed by a feed-forward network (MLP layer), each surrounded by residual connections and layer normalization. Both blocks are dominated by dense matrix multiplications. A full inference run naturally splits into two phases: the *prefill* phase processes the user's prompt in a single forward pass that produces the key and value caches for every layer; then the *decode* phase generates output tokens one at a time with each new token requiring a full forward pass that reads the model weights and the accumulated KV-cache from memory [98],[100].

These two phases place very different demands on the hardware. The distinction is most cleanly expressed in terms of *operational intensity*: the number of arithmetic operations performed per byte of data moved from memory [101],[102]. When operational intensity is

high, performance is often bounded by the peak compute throughput of the accelerator, and the workload is said to be *compute-bound*. When it is low, performance is often bounded by the rate at which operands can be delivered from memory, and the workload is *memory bandwidth-bound*. Prefill processes many tokens in parallel through the same weights, so each weight matrix is reused across the full prompt and operational intensity is typically high, therefore compute-bound. Decode, by contrast, generates one token at a time and each forward pass reads the entire model and the growing KV-cache from memory, but performs only the operations required for a single token, so operational intensity is low and decode is typically memory bandwidth-bound. Moving a byte of data from main memory to compute costs significantly more energy and time compared to performing a MAC operation [96],[103], so a key objective for accelerator design is to minimize costly main memory accesses in both regimes.

6.2.2 Accelerator architecture organization

An important challenge for an AI accelerator is data movement. In general, there are two architectural knobs that can be tuned to address this: (i) temporal reuse, where data is reused across multiple timesteps without re-fetching from an outer memory; and (ii) spatial reuse, where data is reused across multiple spatial instances without refetching from memory. For example, temporal reuse can be exploited by fetching a value from expensive off-chip memory and keeping it in a cheaper on-chip memory to use across computations. While spatial reuse can take the form of fetching a value from memory and using it across multiple parallel processing elements (PEs). When well matched, these mechanisms push the workload towards a compute-bound regime and amortize each expensive memory access over many cheap arithmetic operations [104].

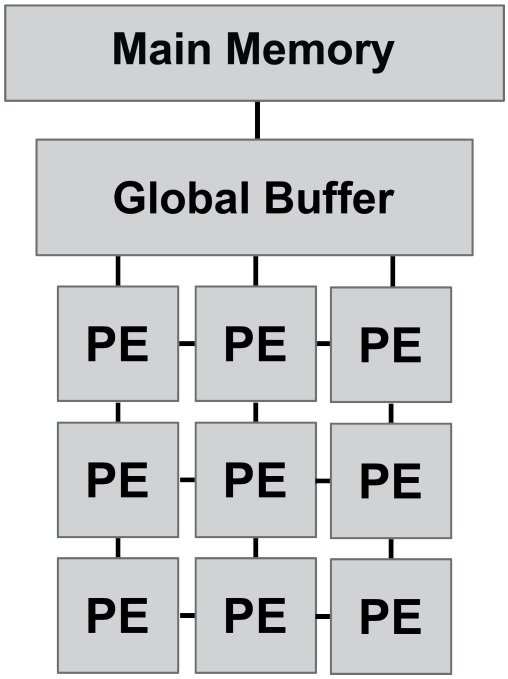


Figure 6.1: Typical template for accelerator architecture.

An example organization is shown in Figure 6.1. Off-chip main memory holds the full set of model weights and activations for the workload. An on-chip global buffer (GLB) sits one level closer to the compute, with capacity typically in the megabyte range, and is used to stage tiles of the workload that are actively being consumed by the PE array.

The PE array is a spatial collection of compute units, each of which typically contains the arithmetic needed for a MAC and a small amount of local storage. This accelerator captures the important attributes of temporal reuse, spatial reuse, and parallelism. To scale this up to full accelerators such as TPU [105] and Eyeriss [104], we may add multiple levels of memory hierarchy and fanout, along with other compute units.

6.2.3 Mapping

Given a workload and an architecture, the *mapping* specifies how data movement and operations are scheduled in the accelerator architecture. Mapping determines which tiles are stored in each level of the memory hierarchy, in what order tiles are processed, and how computations are distributed across the PE array [96]. The mapping roughly plays the role that the compiler plays for a CPU. The mapping also determines how much reuse is leveraged by the accelerator, because it determines which tensor is held at each level of the memory hierarchy and for how long.

Because the mapping determines the data movement pattern, it has a direct effect on both energy and throughput. Different mappings of the same workload onto the same architecture can produce radically different energy efficiency even when they deliver comparable throughput. Parashar et al. [106] demonstrate this on the `VGG conv3_2` layer mapped to a 1024-MAC architecture: among the 480,000 mappings that achieve within 5% of peak performance, energy efficiency varies by nearly 19×, and only ten mappings are within 1% of the energy-optimal choice. The implication for a design space exploration tool is that the quality of its architecture comparisons is bounded by the quality of the mapper it uses. An evaluation that compares two architectures using sub-optimal mappings cannot distinguish a poor architecture from a poorly-mapped one, and may attribute to architecture what is in fact an artifact of the mapper's search constraints.

The mapspace for a single Einsum on a multi-level memory hierarchy is combinatorially large, with 10^{23} to 10^{37} valid mappings for typical transformer and CNN workloads on commercial-scale accelerators [97]. Exhaustive search is therefore infeasible, and prior mappers manage this by restricting the mapspace along one or more axes: fixing the dataflow, constraining tile shapes, considering only layer-by-layer (unfused) mappings, or trading optimality guarantees for runtime with heuristics [106],[107],[108]. These restrictions make search tractable, but they also place a ceiling on the mappings the tool can find. Recent work shows that the cost of these restrictions is large in practice: the Fast and Fusiest Mapper (FFM) [109] finds fused mappings with 1.3–37× lower latency for the resulting mapping when compared to prior mappers; and the Turbo-Charged Mapper (TCM) [97],

which achieves comparable speedups in the unfused setting through a new pruning scheme based on *dataplacement*. Together, TCM and FFM provide the mapper used in this work and make it tractable to evaluate each candidate architecture against its true Pareto-optimal mapping rather than against an arbitrary representative one.

6.2.4 AccelForge modeling framework

To quickly evaluate early-stage accelerator design decisions, analytical modeling is preferable to cycle-accurate or circuit-level simulation because it doesn't require the underlying microarchitecture or circuit implementations to be specified. For an optimized design process, we would like system-level design results to guide focus on lower level circuit design and vice versa.

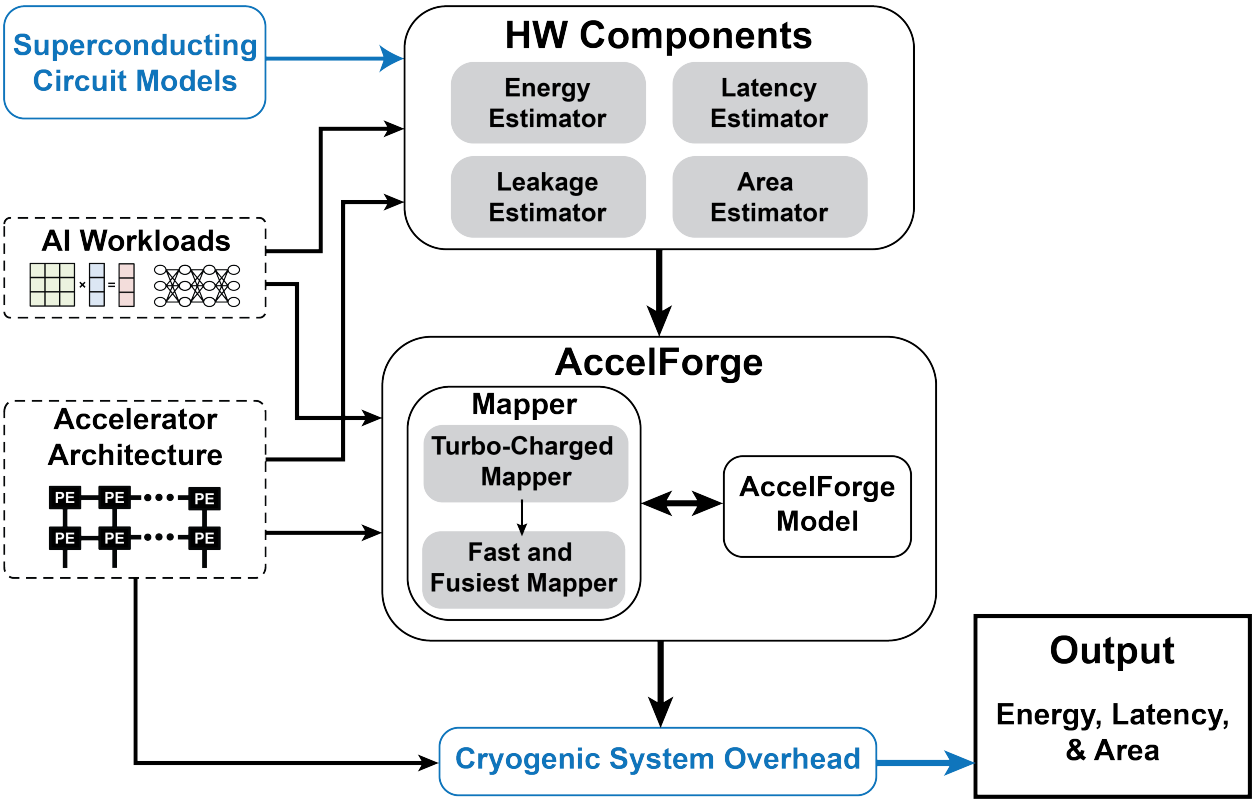


Figure 6.2: Block diagram of AccelForge modeling tool; superconducting contributions shown in blue. Quick mapping is done with Turbo-Charged Mapper (TCM) [97] and Fast and Fusiest Mapper (FFM) [109]. Each mapping is then evaluated with AccelForge analytical model [110],[111], with component level projections informed from hardware components [112],[113]. The tool is leveraged for modeling superconducting electronics with new superconducting hardware component models and cryogenic overhead factored in to the output.

Our superconducting design exploration leverages the AccelForge modeling framework and hardware components [94],[112], the successor to Timeloop+Accelergy [106],[113]. The

principal advance over Timeloop+Accelergy is that AccelForge can search *fused* mapspaces in tractable runtime, by virtue of its improved mapper and model. A block diagram overview of the tool is shown in Figure 6.2. A workload and architecture specification (as .yaml input) is passed to the HW components and AccelForge, along with a set of optimization goals. Then, the energy, latency, and area of the architecture under the optimal mapping it can find for that workload is returned. The tool is composed of three high-level components: the hardware components, the mapper, and the model. Technology-specific energy and area specs are supplied to AccelForge by the hardware component models [112],[113], which are parameterized architecture building blocks, agnostic to mapping and dataflow. TCM [97] and FFM [109] are the mappers, as discussed in Section 6.2.3. They evaluate candidate mappings with results from the AccelForge model [110],[111], which derives the per-component data transfer counts and compute counts for each mapping on the specified architecture.

The AccelForge model has been validated against published accelerator designs spanning a range of dataflows, with worst-case modeling error below 4% relative to the reference results [111]. HW component models inherit the validation status of their underlying library. For example, CiMLoop HW components have been validated against both simulated and silicon-measured results and demonstrated an average energy error of 4% for discrete components, and a maximum 7% error in energy efficiency for workload-specific projections[112]. Therefore, in regimes where the tool can validate accelerators which have already been fabricated, AccelForge produces performance metrics consistent with measured outcomes. When leveraging the tool for superconducting systems, in a regime where large-scale systems do not exist to validate against, we are optimistic that results should be accurate enough to rank and prioritize design choices correctly.

The AccelForge model rests on a set of explicit assumptions that constrain what it can faithfully represent [94]. These are detailed in the tool's documentation, but we will highlight a few key assumptions:

- Per-action energy and latency are taken to be constant per Einsum and independent of component state; therefore, per-bit memory access costs and per-operation compute costs do not vary with addresses, stored values, or access history.
- Scheduling is static and exhaustive, meaning all memory is explicitly allocated by the mapping, following memory hierarchy order from the architecture, and it's assumed that all data arrive at their destination exactly at the cycle it's consumed.
- Perfect pipelining is assumed such that within each Einsum, the latencies of compute and memory accesses are expected to overlap and the total latency is the maximum of the per-component latencies rather than their sum.

6.3 AccelForge for superconducting electronics

In the semiconductor accelerator community, tools like Timeloop+Accelergy, and now AccelForge, are an established part of the early-stage design workflow. No equivalent tool exists for superconducting electronics, and the absence of one is a significant gap in the field's ability to evaluate architectural choices before committing to large scale design and fabrication efforts. The remainder of this section describes how we've leveraged AccelForge to close that gap. We will first discuss how the cryogenic cooling cost is handled in the model. Then, we will walk through the hardware component models that have been developed for compute logic, memory, and interconnects across cryogenic temperature stages.

6.3.1 Cooling overhead

Superconducting circuits require a cryogenic environment to operate. The Nb-AlOx-Nb Josephson junction technology for digital superconducting electronics needs to be around 4 K to operate reliably [27]. The following text describes the cooling assumptions used throughout the design studies in this chapter and how the cryogenic system overhead is incorporated in the AccelForge framework.

Cryocooler specific power. The cost of cryogenic cooling is parameterized by the *specific power*, the ratio of room-temperature input power to delivered cooling power at the cold stage:

$$\eta_{\text{cooling}} \equiv \frac{P_{\text{hot}}}{P_{\text{cold}}} \quad [\text{W/W}]. \tag{6.2}$$

The specific power is the inverse of the coefficient of performance and is bounded below by the Carnot limit, $(T_H - T_L)/T_L$, which evaluates to 74 W/W for a 4 K cold stage with a 300 K heat sink. Real cryocoolers fall well above this bound. The IRDS Cryogenic Electronics and Quantum Information Processing report [9] surveys commercial systems and reports box plot statistics for specific power at cryogenic temperatures of interest. For large capacity refrigerators ($\geq$ 10 W cooling power) operating near 4 K, the reported median specific power is 577 W/W. While small capacity cryocoolers (less than 10 W cooling power at 4 K) have an average specific power an order of magnitude worse at 6,970 W/W. The split between the two regimes reflects the loss of efficiency for small cooling capacities and motivates designing large-scale AQFP systems for the large capacity cryocooler regime.

Implementation in AccelForge. To start, we assume that the cryogenic system is characterized by a single representative specific power. As is commonly used in the field, we take this value to be 1000 W/W at 4 K. Cooling overhead is applied as a post-processing multiplier on AccelForge energy results, rather than as an input to the component models. We've added a temperature specification to each of our HW component models and wrapped AccelForge calls with a cryogenic system overhead check that will apply a specific power multiplicative factor to component energies based on their temperature stage. For example, components on a 4 K stage have their energies multiplied by $\eta_{\text{cooling}} = 1000$ W/W and components at room temperature pass through unchanged with $\eta_{\text{cooling}} = 1$ W/W.

However, a single specific power, is a coarse abstraction of what a real cryogenic system looks like. The compressor design and operation can tune how much cooling capacity is available at each stage and how much wall-plug power is required to sustain it. One could imagine a system in which the cryocooler design itself is tightly coupled to the architecture and workload of the computing system it cools, with cooling capacity allocated across stages to match the dissipation profile of the accelerator. Modeling these cryocooler design choices, and folding the resulting $\eta_{cooling}$ relationships into AccelForge results, is left to future work. The reason that cooling overhead is applied as a post-processing multiplier on component energies, rather than baked into the component models themselves, is precisely to keep this degree of freedom open. Along with the fact that it allows the same set of AccelForge results to be re-evaluated under any cryocooler model by changing the per-stage multipliers.

6.3.2 Component models

AccelForge hardware components provide a parameterized model for the energy, latency, and area of accelerator building blocks, implemented as Python classes [112],[113].

Each component model exposes an `area` and a `leakage power`, together with a set of `actions` the component can perform. Each action returns the `energy` and `latency` associated with performing it. These values scale with parameters set in the architecture specification or passed in from AccelForge. An adder component model, for example, exposes a datawidth parameter that selects the bit-width of the adder (e.g., 8-bit versus 32-bit), with area and leakage power returned accordingly, and an `add` action that returns the energy and latency of a single addition. There may also be a technology node parameter further scaling these values through the appropriate scaling functions.

Our superconducting component models are extrapolated from published results, measured data, or internal circuit designs.

AQFP Models In AQFP logic, every device is driven through a full switching event by the AC clock-power each clock period, and each switch dissipates energy whether or not the device sits on the active data path. This stands in contrast to CMOS, where dynamic energy is dissipated only when an input transition propagates through a gate, i.e. a CMOS adder that receives no operands in a given cycle dissipates negligible dynamic energy. CMOS circuits also have leakage power, which is dissipated by transistors in the nominally "off" state and is independent of clock frequency or switching activity. In CMOS, the clock frequency therefore governs dynamic energy while V_{DD} has the dominant effect on leakage. However, in AQFP, the clock signal is also the power supply and every device is energized every cycle, regardless of activity. This means that an AQFP adder dissipates the energy required to perform a single addition each clock cycle whether or not an addition is actually being performed[1]. There is a baseline power dissipation associated with every AQFP component whenever the circuit is clocked.

From the HW Component modeling perspective, this means that AQFP energy is most naturally represented as leakage power rather than a per-action dynamic energy. This description is not exact, since AQFP cycle dissipation depends on clock frequency, whereas CMOS leakage does not. But for the purpose of fitting AQFP behavior into a transistor-native tool, all energy in an AQFP component model is reported as leakage and the dynamic energy of every action is set to zero. The frequency dependence of the leakage value is accounted for at model construction time, given the operating frequency specified in the architecture description.

The architectural consequence of AQFP energy dissipation being a frequency-dependent, constant baseline is that energy and latency are tightly coupled. Because every device dissipates on every cycle whether utilized or not, energy efficiency requires keeping the available compute fully utilized and completing the computation as quickly as possible, minimizing the total number of clock cycles and therefore the total dissipation. Keeping the compute fully utilized means that the memory hierarchy must be fast enough to feed all the available compute without bottlenecking. However, the on-chip memory hierarchy is also composed of AQFP devices, which will be most energy efficient when running at as low a frequency as possible. Therefore, as we will see in subsequent sections, the trade-off between memory bandwidth and compute utilization is a key parameter for AQFP design.

All AQFP component models follow the same construction. The model first determines the Josephson junction (JJ) count of the circuit implementing the component. Per-device

[1] assuming the adder is pipelined for one addition per cycle, so that the total energy dissipated per cycle is equal to a single addition. If the addition took 2 cycles, the add would cost double the energy, but the baseline AQFP power per cycle would remain the same.

power dissipation is then extracted from the operating frequency specified in the architecture. The component's leakage power is set to the product of JJ count and per-device dissipation and all action energies are set to zero. AccelForge expects action latency to be the time between issuing each operation, so for a fully pipelined unit, this should be equal to one clock period. As discussed in Chapter 5, the LinCore dataflow microarchitecture enables all AQFP components to be fully pipelined, so in the context of AccelForge, our AQFP models will have an action latency equal to the clock period. Area is computed as the JJ count multiplied by the cell node area, i.e. the footprint of a single AQFP device, and a routing-overhead multiplier. The routing-overhead value is a single scalar applied uniformly across all AQFP component models, extracted from large-scale AQFP circuits we have designed (on the order of 10^4 JJs, e.g. the 1 Kb register file of Chapter 4).

RQL Models Reciprocal Quantum Logic (RQL) is another superconducting logic family [114], for which we have built a parallel set of AccelForge component models. Including RQL alongside AQFP highlights that superconducting logics are not monolithic in their performance metrics.

Like RSFQ, RQL encodes digital 0s and 1s in the presence or absence of single flux quantum (SFQ) pulses, but it augments this scheme with antifluxon pulses that invert fluxons, allowing logical state to propagate under an AC current bias rather than a DC bias. The AC bias eliminates the static dissipation that DC-biased shunted junctions incur in RSFQ, making RQL substantially more energy efficient. It is not, however, as energy efficient as AQFP, because RQL still relies on shunted Josephson junctions whose switching diverts current into the shunt resistors.

The RQL component models are extrapolated from Dorojevets et al. [115], which presents cell-level designs for the key components needed to scale RQL circuits. The tables in that work report JJ count, energy, and latency values for compute and memory components, derived from layout-aware VHDL simulation.

The authors note that their simulations account for a small standby (leakage) power contribution arising from parasitic coupling of the clock lines to the shunt resistors. Most of their tables report a single total energy per component that lumps standby and dynamic contributions together, but Table V breaks out standby energy alongside total JJ count for the memory and register file designs. From Table V, the average standby energy per JJ works out to 0.469 zJ, which we've assumed to be per clock cycle dissipation since no operation is stated in the table. AccelForge requires leakage power and dynamic action energy as separate inputs, so we subtract this per-JJ average from each reported action energy and reassign it to the leakage channel.

Table 6.1: 8-bit adder specs across superconducting and CMOS technologies

	AQFP		RQL		CMOS 7 nm
Metric	intAdd	fpAdd	intAdd	fpAdd	intAdd
Area (μm^2)	4,941	20,142	3,606	35,018	3.05
Latency (ns)	1.0	1.0	0.154	1.40	0.075
Dynamic Energy (fJ)*	0.0	0.0	14.3	127	4.10
Leakage (μW)	0.0328	0.134	3.55	34.5	0.0469
Total Energy/op at 4 K (fJ)*	0.0328	0.134	14.8	175	4.10

* 1000 W/W cooling overhead applied to cryogenic components

The paper provides JJ counts for each component, which we convert to component area using a per-JJ footprint set by the assumed cell node and a routing multiplier extracted from other RQL layout demonstrations.

6.3.2.1 Compute Models

An overview of all of the superconducting compute component models to date are given in the subsequent tables:

- Table 6.1 compares 8-bit FP and integer adders from AQFP, RQL, and CMOS[2].
- Table 6.2 does the same for 8-bit multiplier.
- Table 6.3 shows compound component models for multiply-accumulate (MAC) operations in each of the technologies.

AQFP The AQFP component library includes integer and floating-point versions of an adder, a multiplier, and a fused multiply-accumulate (MAC), extrapolated from 4-bit reference circuits that we designed and fabricated in the MIT-LL SFQ5ee process. The 4-bit ripple-carry integer adder has 183 AQFP devices, and the 4-bit floating-point adder has 746. The 4-bit integer multiplier has 1,052 devices, and the 4-bit floating-point multiplier has 317. The floating-point multiplier is smaller than the integer multiplier at this width because a 4-bit FP format has only a 1-bit mantissa, so the device count is dominated by the small exponent adder rather than by a mantissa multiplier subcomponent. Integer multipliers are particularly expensive in AQFP. For a multiplier with n-bit datawidth, the n^2 partial products generated from the operand bits must be fanned out to multiple downstream adders in the accumulation

[2]The CMOS adder, multiplier, and MAC models are the Aladdin models [116] provided by AccelForge.

Table 6.2: 8-bit multiplier specs across superconducting and CMOS technologies

	AQFP		**RQL**		**CMOS 7 nm**
Metric	intMult	fpMult	intMult	fpMult	intMult
Area (μm^2)	56,808	17,118	13,649	18,905	17.5
Latency (ns)	1.0	1.0	0.432	0.703	0.075
Dynamic Energy (fJ)*	0.0	0.0	50.7	69.7	61.9
Leakage (μW)*	0.377	0.114	13.4	18.6	0.391
Total Energy/op (fJ)*	0.377	0.114	56.5	82.8	61.9

* 1000 W/W cooling overhead applied to cryogenic components

tree and, in AQFP, fanout requires a dedicated splitter cell. Therefore, the partial-product distribution network contributes substantially to the total device count.

Larger datawidths are projected from these reference points using topology-appropriate scaling rules:

- $\Theta(n)$ for the integer ripple-carry adder
- $\Theta(n \log n)$ for the floating-point adder (dominated by the mantissa-alignment and normalization shifters)
- $\Theta(n^2)$ for both multipliers (dominated by partial-product generation)

The floating-point multiplier projection carries an additional caveat: at the 4-bit reference width, device count is dominated by exponent addition rather than by the mantissa multiplier, while at the larger widths (FP16, FP32) the mantissa multiplier dominates. The $\Theta(n^2)$ rule captures the asymptotic regime correctly, but absolute projections at intermediate widths should be expected to carry larger error than the integer-multiplier projection. The scaling rules are derived from the circuit topology of each component rather than fit to fabricated data, since only one datawidth has been fabricated for each component. Direct validation of the AQFP scaling rules against fabricated AQFP circuits at multiple widths is left to future work.

Per-component energy, area, and latency are then computed from the projected cell counts using the construction described in the previous section, with the operating clock frequency taken to be 1 GHz unless overridden by the architecture specification.

RQL The RQL compute models are extrapolated from the 32-bit floating-point and integer adder and multiplier designs reported by Dorojevets et al. for the MIT-LL 10 kA/cm^2 [115].

Table 6.3: 8-bit MAC compound component models across superconducting and CMOS technologies. Compound component composed of 8-bit multiplier and 16-bit adder for accumulator.

	AQFP		**RQL**		**CMOS 7 nm**
Metric	intMAC	fpMAC	intMAC	fpMAC	intMAC
Area (μm^2)	61,749	37,260	14,871	53,923	20.5
Latency (ns)	1.0	1.0	0.470	2.102	0.150
Dynamic Energy (fJ)*	0.0	0.0	56.8	196.6	66.0
Leakage (μW)*	0.410	0.248	14.6	53.1	0.437
Total Energy/op (fJ)*	0.410	0.248	63.6	308.1	66.1

* 1000 W/W cooling overhead applied to cryogenic components

The integer adder reported in Table 6.1 uses the 32-bit sparse-tree integer adder, so that we can compare scaling to its 64-bit counterpart. Smaller and larger datawidths are projected from the 32-bit references using topology-appropriate scaling rules. Device count and energy scale as $\Theta(n \log n)$ for both adders, and as $\Theta(n^2)$ for both multipliers.

It is assumed that the RQL compute modules are not fully pipelined, as in AQFP, so we need to account for latency as well. The clock frequency is assumed to be 12.1 GHz, matching the operating frequency reported in the Dorojevets paper, unless overridden by the architecture specification. Latency scales as $\Theta(\log n)$ for both adder and multiplier, reflecting the depth of the parallel-prefix carry tree and the partial-product compression tree, respectively.

Because the Dorojevets paper also reports values for 64-bit sparse-tree adders and multipliers, we can validate the parameterized model by projecting from the 32-bit reference and comparing against the 64-bit reported values. Figure 6.3 shows this comparison for device count, energy, and latency across the 64-bit components; the parameterized models agree with the Dorojevets values within 10% across all metrics.

6.3.2.2 Memory Models

The memory models are organized into two categories: on-chip buffer memory, which encompasses register files and buffer caches, and main memory technologies. Table 6.4 reports 1 Kb storage blocks for the on-chip memory technologies, and Table 6.5 reports the main memory technologies. Within each category, the models are further separated by operating temperature, distinguishing cryogenic from room-temperature technologies.

The AQFP SR-Loop memory model is based on the v2 SR-Loop register file design described

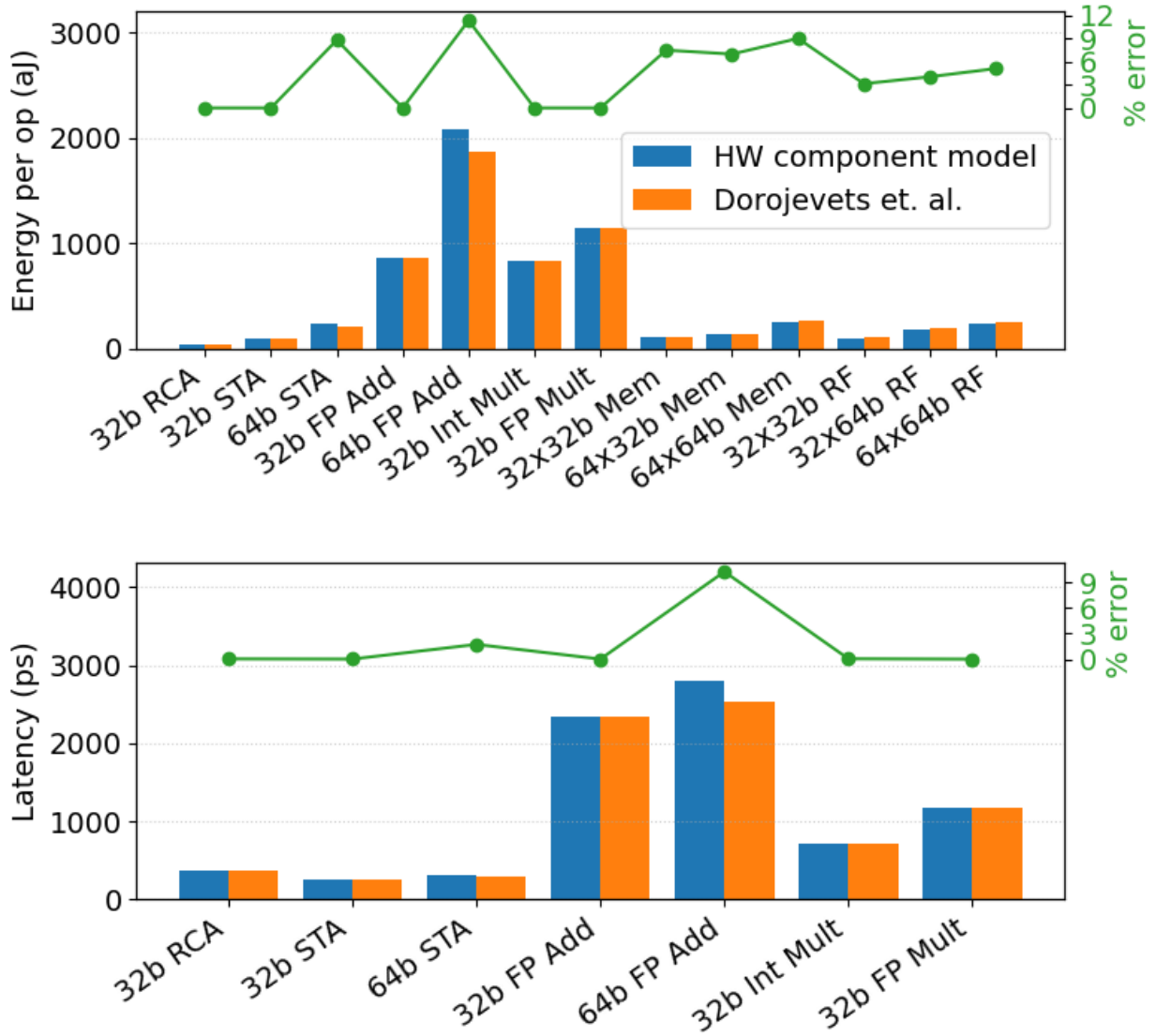


Figure 6.3: Verification of RQL HW component models, comparing to simulated values reported in [115]. The models are extrapolated from the 32-bit circuits.

in Chapter 4. The 1 Kb entry in Table 6.4 corresponds to the 4 × 4 × 64-bit configuration designed and fabricated in Section 4.3.2.

The RQL NDRO model is derived from the register file designs in Dorojevets et al. [115]. The NDRO storage cell is the simplest single-bit storage element available in RQL and is functionally analogous to a flip-flop or latch register in CMOS. Table V of [115] provides a component-level breakdown of six register file configurations using NDRO storage cells. For each subcomponent, we extracted scaling laws as a function of register file width and depth to produce a parameterized model that accepts arbitrary width and depth values from

the architecture specification. Energy comparisons for our models relative to values given in Dorojevets et al. are shown in Figure 6.3.

The nMem model is based on nanowire memory measurements reported in [75],[117]. The values used in the model are taken directly from circuit measurements of a first-generation, deliberately conservative design intended as a proof of concept. Several straightforward layout improvements would yield substantially better per-cell energy, including replacing the bulk resistor with a superconducting wire and confining the resistive heater to the area where heating is actually required, and more aggressive feature sizes to improve cell density. However, the published characterization reports only single-cell performance and does not account for the energy cost of drivers or address decoding. We therefore retain the unoptimized per-cell values as a conservative proxy that implicitly absorbs the unaccounted peripheral overhead.

The CryoSRAM model is extracted from [118], where a 40 nm CMOS array was characterized for cryogenic operation. The model metrics are scaled from Table II of that work, which reports read and write energies for a 32×32 array, including drivers and addressing, at 4.2 K. The reported energy at 4.2 K is comparable to the 7 nm CACTI results at room temperature, which suggests that further optimization, particularly migration to a smaller technology node, would yield additional gains in cryogenic CMOS performance.

The room-temperature SRAM model is based on the CACTI cache modeling tool [119],[120],[121], which is provided as a built-in hardware component model within the AccelForge framework [112],[113].

Table 6.4: Component model output for 1 Kb register across cryogenic and room-temperature memory technologies.

	Cryogenic				**Room Temp.**
Metric	AQFP SRLoop	RQL NDRO	nMem	CryoSRAM	CACTI SRAM
Area (mm^2)	0.124	0.267	0.00102	0.00356	0.00004
Leakage (μW)*	0.528	263	0.0	0.0	0.0210
Read Latency (ns)	1.0	0.0232	0.125	0.0368	0.0968
Dynamic Read Energy (fJ)*	0.0	28.6	1864.0	154,500	50.8
Total Energy / read (fJ)*	0.528	34.7	1864.0	154,500	50.8
Write Latency (ns)	1.0	0.0232	0.125	0.0368	0.0968
Dynamic Write Energy (fJ)*	0.0	67.1	1.04×10^7	118,250	105
Total Energy / write (fJ)*	0.528	73.2	1.04×10^7	118,250	105

* 1000 W/W cooling overhead applied to cryogenic components

Scaling AQFP SR-Loop Memory Large on-chip memory capable of holding intermediate activations and weights is critical to the energy efficiency of an AI accelerator, as the design studies in the next section will make explicit. The largest superconducting random-access memory demonstrated to date, however, is 4 Kb, fabricated in SFQ-style latches [76]. Scaling on-chip superconducting memory to the Mb to 100s Mb range required by modern accelerators is therefore extrapolation beyond any existing demonstration. With AQFP as the focus of this work, and the SR-Loop register file serving as the on-chip global buffer, the remainder of this section describes how the SR-Loop circuit is extrapolated from the 1 Kb demonstrated scale to Mb-class storage in the AccelForge component model.

The central modeling decision concerns how AQFP energy dissipation scales with memory capacity. Because every active AQFP device dissipates a baseline power on every clock cycle, the total power of a memory built from active AQFP storage cells (rather than passive supercurrent loops) scales directly with device count, and therefore with capacity. At a GHz-range clock frequency, this scaling makes large capacity AQFP memory prohibitively expensive in power. The key observation that makes large AQFP memories feasible is that on-chip memory does not in general need to be clocked at the same rate as compute. Reducing the clock frequency of the memory relative to the compute clock trades memory bandwidth for power reduction. AI workloads have predictable access patterns, and we can leverage reuse to reduce accesses to large memories. In general, large memories far from compute are accessed less frequently than smaller memories near compute. This means that large memories can have a lower bandwidth and be clocked at a lower rate.

The SR-Loop memory model is parameterized by four quantities: `width`, `loop depth`, `size`, and `clock derate`. `Width` sets the word length across the memory bank, i.e., the number of slices organized in each row. `Loop depth` sets the number of bits each storage cell can hold in its shift-register feedback loop. `Size` is the overall capacity of the memory; given size and width, the depth of the register file (the number of loops in each slice) is determined. `Clock derate` is the factor by which the memory clock is reduced relative to the compute clock, capturing the bandwidth vs. power trade described above.

The model further organizes the memory architecture into banks, following the convention used in SRAM design. A single bank has a maximum width of 32 slices and a maximum loop depth of 64 bits per slice. The per-bank width and depth set the size of the decoder and demux required to address a single bank. When the architecture-level width or depth parameters exceed these per-bank maxima, the memory is partitioned across multiple banks with an additional layer of decoder and demux addressing overhead proportional to the number of banks.

Table 6.5: Main memory Component models

	Cryogenic		Room Temp.	
Metric	CryoHBM	nMem	HBM3	LPDDR4
Energy / bit (pJ)*	4,330	233 (read) 1,300 (write)	4.05	8
Datawidth	1024	1024	1024	64
Latency / bit (ns)	0.0182	–	0.0182	0.297
Throughput (Gbps)	6.4	–	6.4	0.050

* 1000 W/W cooling overhead applied to cryogenic components

6.3.2.3 Interconnect and I/O

How a superconducting computing system communicates with the room-temperature world, whether for main memory, CPU routing, or networking I/O, can make or break overall system performance. In this section we evaluate a set of temperature stage interconnect technologies and compare their throughput and energy per bit.

For efficiency, 4 K cryostats are typically built with two stages: a first stage cooling 300 K to roughly 40–50 K, and a second stage from 40 K to 4 K. The lower stage has a much higher specific power, so the cooling budget at 4 K is the binding constraint. For example, a typical system might allow 1 W at 4 K and 50 W at 40 K, all for the same 1000 W/W overhead cost. We therefore focus interconnect modeling on the 4 K-to-40 K transition and treat the 40 K-to-300 K jump as effectively free.

Each component model below accounts for cable heat load at the 4 K stage and the dynamic and static power of the cryogenic transmitter and receiver circuits. Models accept a `channel count` parameter that scales the number of parallel data lanes.

We separate paths by direction: *ingress* carries data from room-temperature (or 40 K) down to the 4 K stage; *egress* carries data from 4 K outward. We examine electrical, optical, and wireless cabling. Some component models are based on full transceiver circuit designs with measured throughput, while others are more speculative. We evaluate components against a target on the order of ∼1 Tb/s, to be on the order of high-bandwidth memory throughput.

Table 6.6 and Tables 6.7 summarize the circuit performance for each model and Figure 6.4 plots throughput versus power as the channel count is swept (1, 2, 4, 8, . . .) for each model. Points at the same ordinal index across traces correspond to the same channel count labeled in the bottom trace.

Across the models, an optical link gives the lowest energy per bit for ingress once optical

Table 6.6: Ingress component models. All values are reported at 4 K, i.e. no cryogenic cooling taken into account.

Model	Fiber-CMOS-PAM4	Wireless-CMOS	Stripline-RSFQ/AQFP	Coax-CMOS	Fiber-AQFP
References	[122]	[123]	[59],[124]	[122],[125]	–
Data rate (Gbps)	56.0	4.4	4.5	154	20.0
Cable heat load (mW)	negl.	0.0	0.125	0.355	negl.
Dynamic energy (fJ/bit)	410	34.5	0.000138	410	1.0
Leakage power (mW)	15.2	0.0	0.125	0.355	0.0003
Total energy per bit (fJ/bit)	681	34.5	27.8	412	1.02

Table 6.7: Egress component models. All values are reported at 4 K, i.e. no cryogenic cooling taken into account.

Model	Fiber-CMOS-PAM4	Wireless-CMOS	Fiber-CMOS-MonoEO	Coax-CMOS	Stripline-AQFP
References	[122]	[123]	[126]	[125],[127]	[124]
Data rate (Gbps)	56.0	4.0	1.0	40.0	5.0
Cable heat load (mW)	negl.	0.0	negl.	0.355	0.125
Dynamic energy (fJ/bit)	390	214	74.9	2460	0.0
Leakage power (mW)	30	0.0	0.0251	0.0	0.125
Total energy per bit (fJ/bit)	926	214	100	2469	25.0

absorption, cable heat load, and receiver circuitry are accounted for. For egress, however, an electrical link is most power efficient because generating or modulating a light source at 4 K is currently less efficient than the heat load of the flexible stripline cables. We assume the latency of each interconnect is the inverse of its throughput as data passes through the link without additional pipelining or buffering. We also assume that power and circuit complexity scale linearly with channel count. Real implementations may share infrastructure such as clocking, equalization, and laser sources across channels, reducing the per-channel cost at higher channel counts, but our linear assumption provides an upper-bound approximation. A summary of each of the models follows.

`Fiber-CMOS-PAM4`: The CO-QLink is a high-speed cryogenic optical transceiver transmitting 56 Gbps with PAM4 modulation [122]. The receiver is a co-packaged PIN diode with cryoCMOS analog front end. Because VCSELs are unreliable at cryogenic temperatures, the transmitter drives an external Mach-Zehnder modulator (MZM) through a voltage-mode driver with three-tap feed-forward equalization. Cable heat load is negligible because optical fiber has very low thermal conductivity, but the cryoCMOS drivers and receivers dominate the energy per bit.

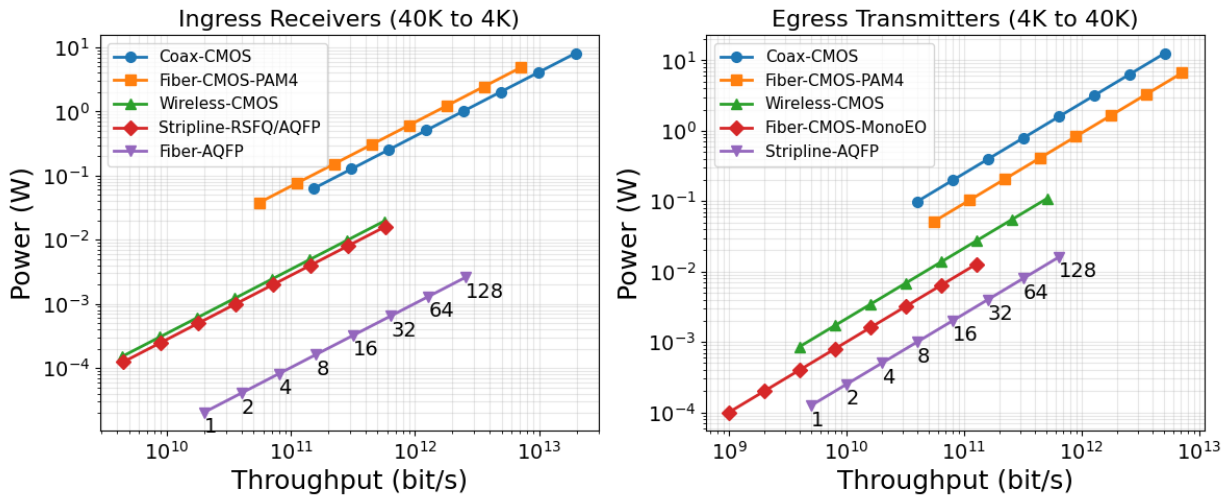


Figure 6.4: throughput versus power sweeping channel count for all of the interconnect models

`Wireless-CMOS`: The wireless terahertz interconnect eliminates conductive cable heat load entirely because the signal propagates through vacuum between the room-temperature horn antenna and the 4 K chip [123]. The reported link operates near the fundamental thermodynamic limit of heat-to-information transfer for a chip based 4 K to 300 K interface. In practice, however, the on-chip THz transceiver requires power-hungry RF circuitry that sets its power dissipation comparable to electrical cable connections. The demonstrated data rate is approximately 4 Gbps.

`Stripline-RSFQ/AQFP` (ingress): This ingress model pairs an AQFP/RSFQ hybrid SerDes [59] with a Delft Circuits Cri/oFlex flex cable [124]. The flex cable contributes 125 μW of heat load per channel at 4 K. The hybrid SerDes uses RSFQ shift registers to capture high-speed serial data and converts it to lower-rate AQFP signals; the demonstrated SerDes test chip operates up to 4.5 GHz with 621 pW total dissipation [59].

`Coax-CMOS` (ingress): This ingress model represents a semi-rigid coax cable option that can send higher frequency data than the flexible stripline cables. Raicu et al. provides a meticulous thermal model for the SC-086/50-SCN-CN coaxial cable, finding that it has a heat load of 0.355 mW at 4 K [125]. From Coax Co., Ltd., the cable is expected to operate up to 124 GHz [128]. We pair the semi-rigid coax with the cryoCMOS receiver from CO-QLink [122], removing the PIN diode and TIA front-end so that the modeled receiver consumes 0.41 pJ/bit, representative of a cryoCMOS digital back-end needed to recover data from the coax cable.

`Fiber-AQFP` (ingress): This model represents projected performance of an optical ingress

link with a cryogenic photodetector drives a superconducting electronics receiver front-end. Cable heat load is treated identically to other fiber-based models in this section. Receiver power is modeled from first-principles bounds on the photodetector and front-end circuit; detailed circuit implementation is omitted pending patent disclosure.

`Fiber-CMOS-MonoEO` (egress): This egress model is based on the most optimistic operating point of a monolithic electro-optical transmitter developed in Yin's thesis [126]. The transmitter takes input from SFQ voltage pulses directed and drives a microring modulator with a laser-forwarded coherent receiver at room temperature. Replacing the cryoCMOS driver chain of CO-QLink [122] with a monolithic superconducting electronics photonic transmitter substantially reduces the cryogenic-side power.

`Coax-CMOS` (egress): The cryoCMOS egress model is based on Fakkel et al. 40 GB/s PAM4 wireline transmitter [127]. This DAC-based transmitter was fabricated in 40 nm CMOS, so improved design could achieve better energy efficiency at a smaller fabrication node. We've attached the transmitter to the SC-086/50-SCN-CN semi-rigid coax, following [125], because flexible stripline cables cannot support the 40 GB/s bandwidth.

`Stripline-AQFP` (egress): Finally, the AQFP latch with stripline egress model is based on a latching AQFP amplifier sending signal through the Delft Cri/oFlex cable [124]. These amplifiers produce output swings large enough to be recovered at room temperature without an intermediate amplification stage and can be clocked up to 5 GHz, similar to the AQFP devices themselves.

6.4 Design studies

In the following studies, all workloads have 8 bits per value, and datawidth is set to 8 on compute components. In comparing CMOS to superconducting hardware implementations we always keep the same architecture spec, but allow the AccelForge mapper to find the optimal mapping (given the different HW components). We do this to highlight performance and efficiency changes that arise from the device technology, providing an apples to apples comparison for superconducting versus CMOS. While the optimal architecture design may be different for superconducting versus CMOS, we save that for future work.

6.4.1 Bare Processing Element (PE) array

We begin with the system most representative of what can be built today: an array of LinCore-style processing elements. Each PE consists of a single MAC unit and an adjacent 1 Kb register file. The full architecture, shown in Figure 6.5, is an array of these PEs connected directly to main memory. For the superconducting technologies, we assume an ideal interface between 4 K and room temperature, that is, the cold compute can read and write room-temperature HBM with no additional latency or energy attributable to inter-stage communication. This assumption is intentionally optimistic and isolates the cost of the compute fabric from the cost of cryogenic-to-room-temperature data movement, which we revisit in Section 6.4.3.

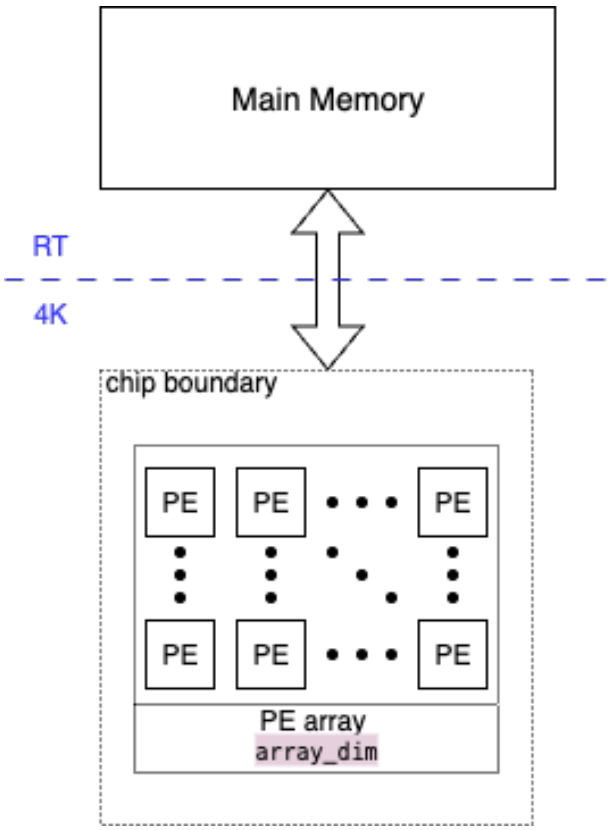


Figure 6.5: Architecture of the bare superconducting PE array, with no on-chip memory and an idealized 4 K to room-temperature (RT) memory interface.

The workload for this study is a single dense matrix multiply of an $M \times K$ matrix by a $K \times N$ matrix, illustrated in Figure 6.6. We size this matmul to the largest dense GEMM in the GPT3 157B model fully-connected layer. In the fully-connected layer, the up-projection and down-projection both take the form of a matrix multiply with shapes set by d_{ff}, d_{model}, and the token count $B \cdot S$, where B is the batch dimension and S is the sequence length. For the GPT-3 175B parameter model, $d_{\text{model}} = 12288$, $d_{\text{ff}} = 4 \cdot d_{\text{model}} = 49152$, and we take $B = 1$, $S = 2048$ (a single user with a 2048-token prompt), giving a matmul with $M \times K \times N = 49152 \times 12288 \times 2048$.

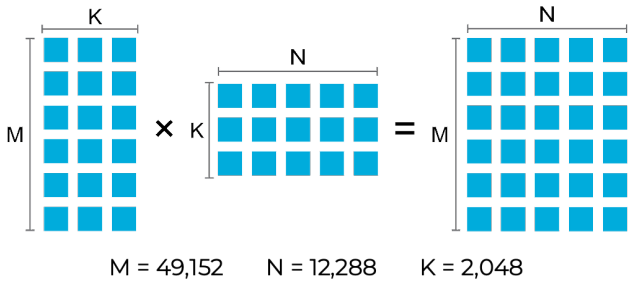


Figure 6.6: Matrix-multiply workload with dimensions set to a single fully-connected layer of GPT-3 175B.

This PE array architecture exposes only one sweepable parameter: the dimension of the array. Sweeping the array dimension while holding the workload fixed produces the energy

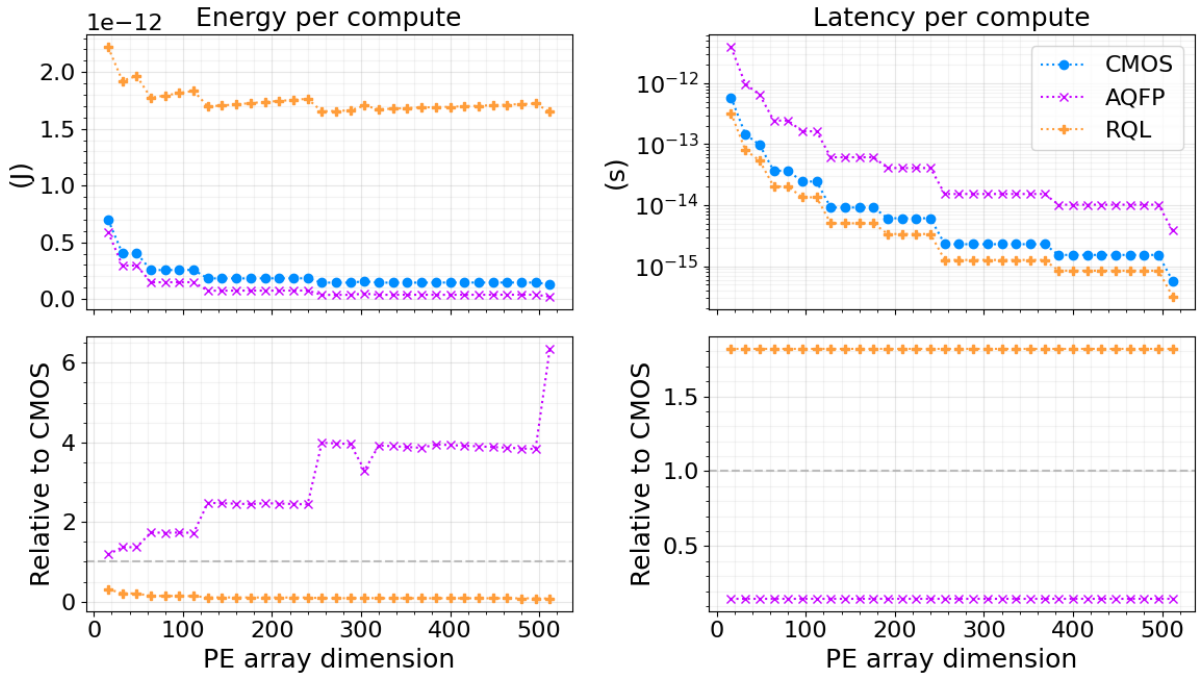


Figure 6.7: Energy and latency results for the PE array architecture (no global buffer) running the matrix-multiply workload, as a function of PE array dimension. Top row: absolute energy (J) and latency (s) per compute, by technology. Bottom row: energy and latency reduction relative to CMOS; the gray horizontal line marks reduction = 1, so traces above the line indicate an improvement over CMOS. Averaged across array dimensions: AQFP achieves 3.08× energy reduction at 6.6× longer latency; RQL increases energy by ∼ 9× and reduces latency by 1.8×.

and latency results in Figure 6.7. Across all array dimensions, AQFP achieves an average energy reduction of 3.08× relative to CMOS, while RQL increases energy by approximately 9×. The trend reverses on latency: RQL achieves a 1.8× latency reduction relative to CMOS, while AQFP increases latency by 6.6×. We note that the 1 GHz clock assumed for AQFP here is conservative and a lower latency could be easily achievable. AQFP operation has been demonstrated at 5 GHz [12], and projections from circuit simulation extend to 10 GHz [129].

The bare PE array results show that the system-level energy advantage of AQFP is at most ∼ 5× energy reduction relative to CMOS at the best operating point in the sweep, despite the underlying AQFP devices providing on the order of 100× lower energy per operation than CMOS. Figure 6.8 breaks down the per-component energy and latency contributions for the 256 × 256 array running the matrix multiply workload. The MAC component itself is 108× lower energy in AQFP than in CMOS, consistent with device-level expectations, but this gain does not propagate to the total because main memory dominates the workload energy.

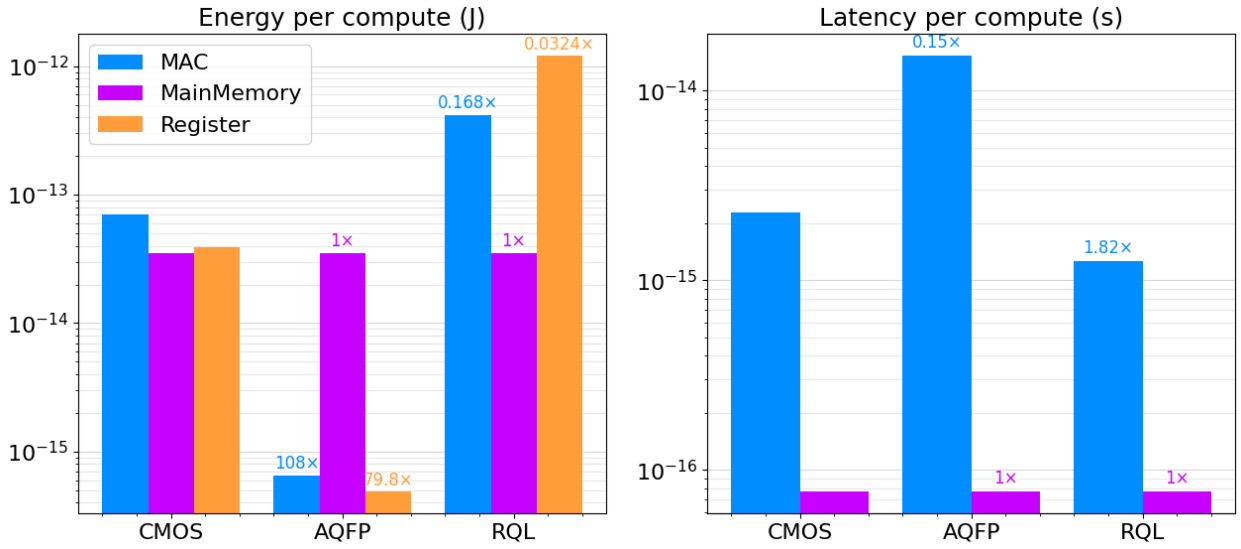


Figure 6.8: Per-component energy and latency breakdown for the 256 × 256 PE array (no global buffer) running the matrix-multiply workload. Labels above each bar give the per-component reduction relative to the CMOS baseline; e.g., the AQFP MAC component dissipates 108× less energy than the CMOS MAC. The system-level total is dominated by main memory, which is identical in both cases.

Since main memory is identical in the CMOS and superconducting cases (room-temperature HBM in both), the component that dominates the energy budget receives no improvement, and the system-level reduction is bounded.

We can think of this as the energy analog of Amdahl's law. Amdahl's law states that the achievable speedup from parallelizing a workload is bounded by the serial fraction that cannot be parallelized. The same bound applies to energy reduction: if only a subset of the system's components is made dramatically more efficient, the total system energy reduction is bounded by the energy fraction of the components that were *not* improved. Concretely, if a fraction f of the baseline workload energy is dissipated in components that are made $r\times$ more efficient and the remaining $(1 - f)$ is unchanged, the total system energy reduction is

$$\frac{E_{\text{baseline}}}{E_{\text{improved}}} = \frac{1}{(1 - f) + f/r}. \tag{6.3}$$

Figure 6.9 plots this relationship for the case $r = 100$, corresponding to the per-bit-operation energy gap between AQFP and CMOS at the device level. The curve is steep near $f = 1$: a workload with 50% of its energy in the AQFP-accelerated path achieves only $\sim 2\times$ system-level reduction. To realize even half of the device-level advantage at the system level, the architecture must drive nearly all of the workload's energy into the cold compute path: 99% is required to reach $\sim 50\times$. The next section examines on-chip memory hierarchy

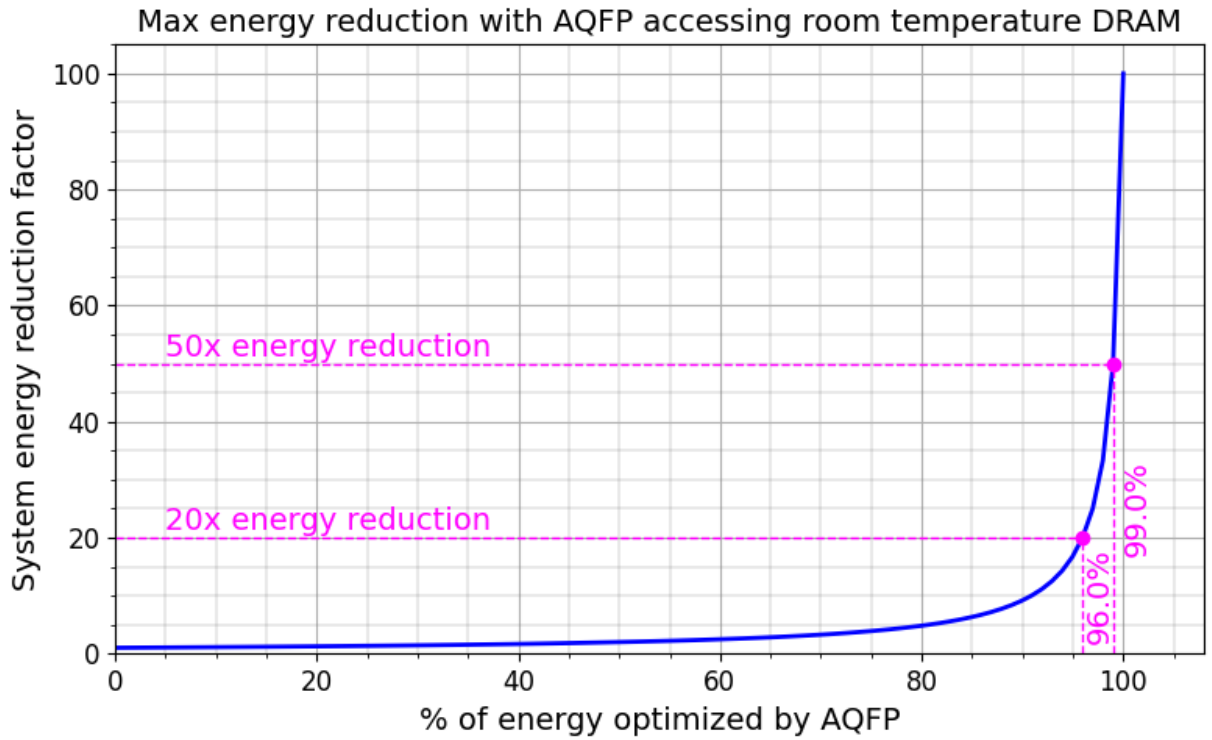


Figure 6.9: Amdahl's law for energy. Assuming any fraction f of the workload energy is dissipated in components that are made 100× more efficient, the total system energy reduction follows Eq. 6.3. Realizing a 50× system-level reduction requires $f \gtrsim 99\%$.

architectural choices that shift the energy fraction toward the AQFP path and bring the system-level reduction closer to the device-level limit.

6.4.2 On-chip memory hierarchy

For the remainder of this chapter, we focus on AQFP and CMOS only. Noting that RQL is a promising technology for high-speed circuit operation, it has comparable or higher per-component energy than CMOS in the bare PE array study and the energy-efficiency questions that motivate this work are not well served by it. The bare PE array results identified main memory as the dominant energy contributor and the bottleneck preventing the AQFP device-level advantage from reaching the system level. One possible way to address this is to introduce a large on-chip memory that holds weights and intermediate values across many compute operations, amortizing each main-memory access over the compute it enables. For AI workloads of the scale considered here, the on-chip memory needs to be tens of MB to capture fusion effectively for modern LLMs, while needing hundreds of MBs to capture all reuse for the same models. Memory of this size has

not be fabrication with superconducting devices before, but our model follows the SR-Loop scaling considerations described in Section 6.3.2.2 to project into this large-memory regime. Our 1 GB upper bound holds roughly 20% of a single GPT-3 175B fully-connected layer weight matrix; smaller buffers reduce the achievable amortization proportionally.

The architecture is shown in Figure 6.10. Two changes from the LinCore-style PE array architecture deserve emphasis. First, the array now shares a single on-chip global buffer (GLB) sized in the tens of Mb to 1 GB range. Second, the PE itself is simplified: rather than a 1 Kb register file alongside each MAC, each PE holds only a single AQFP feedback-loop register sized to the weight value, enabling a weight-stationary dataflow. The workload is unchanged from the LinCore PE array study (Figure 6.6).

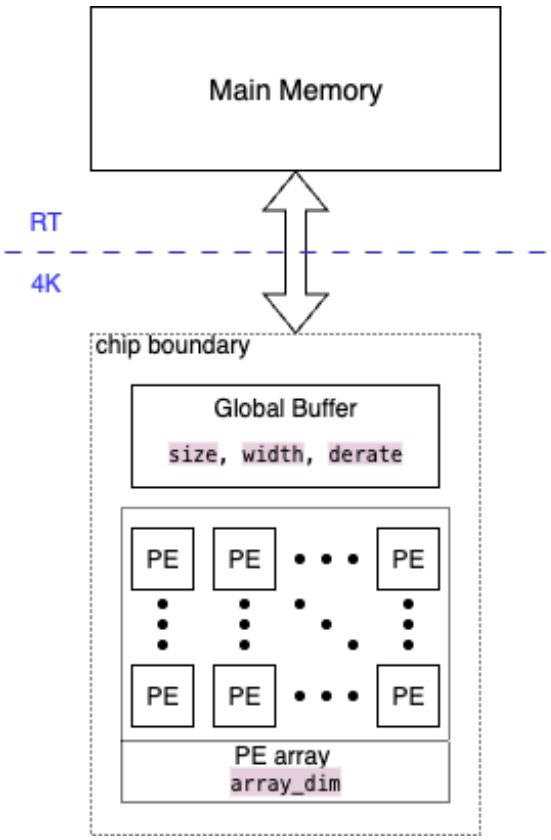


Figure 6.10: Architecture of the PE array with on-chip global buffer. PEs are composed of a MAC and a single weight-sized register; the global buffer is shared across the array.

This architecture exposes four sweepable parameters: PE array dimension, GLB width (the word length of the buffer), GLB loop depth (the number of bits per SR-Loop storage cell, defined in Section 6.3.2.2), and GLB clock derate (the factor by which the memory clock is reduced relative to the compute clock, also defined in Section 6.3.2.2). Of these, only the array dimension and GLB width are meaningful for the CMOS baseline; loop depth and clock derate are AQFP-specific properties of the SR-Loop memory.

Figure 6.11 shows the AQFP energy reduction relative to CMOS across the parameter sweeps. The dominant trend is that the AQFP energy advantage is set by the balance between GLB throughput and compute throughput. When the GLB cannot supply data fast enough to keep the PE array busy, the array stalls. Because every active AQFP device dissipates baseline power on every clock cycle, idle MACs continue to dissipate energy while waiting, eroding the per-operation efficiency. The energy advantage is therefore maximized in the regime where GLB delivered bandwidth is matched to PE array consumption.

Figure 6.12 makes this clear. For the 1 GB configuration with derate $= 8$, the realized MAC energy reduction is only $\sim 3\times$, an order of magnitude short of the $108\times$ device-level

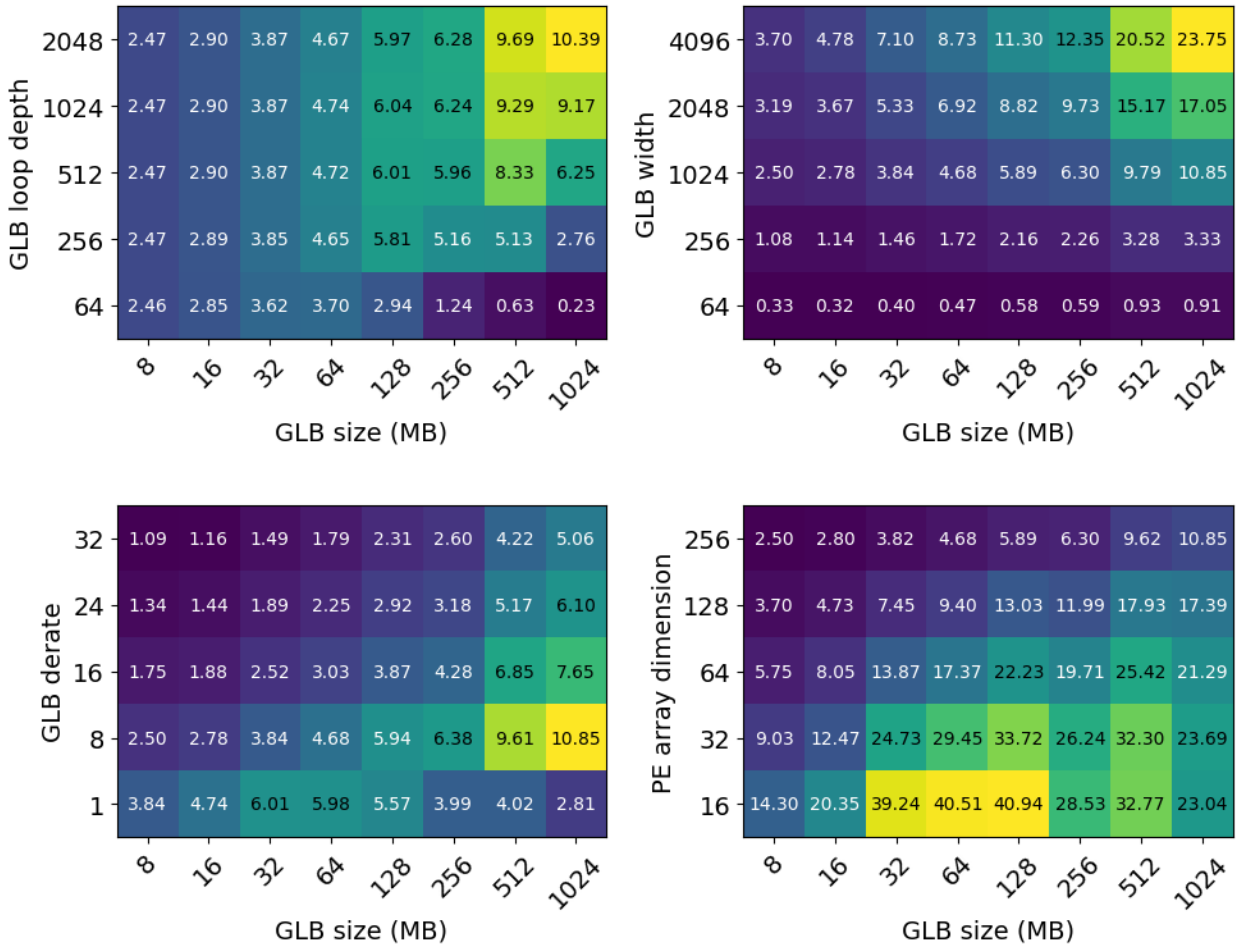


Figure 6.11: AQFP energy reduction relative to CMOS for the matmul workload on the PE array with GLB architecture, swept across each of the four architecture parameters. Each cell reports the AQFP energy reduction relative to CMOS at that sweep point. The CMOS baseline is swept jointly with AQFP across the parameters that apply to both technologies (array dimension, GLB width); for AQFP-only parameters (loop depth, GLB derate), the CMOS baseline is held at the default. Default values when not swept: GLB width = 1024, loop depth = 1024, GLB derate = 8, array dimension = 256.

reduction reported in the LinCore PE array breakdown. The gap is not a property of the MAC itself; the MAC component model is unchanged. Rather, the GLB's bandwidth at derate = 8 is too low to feed the 256×256 array, so the array runs many cycles with no useful data, and the idle-cycle baseline dissipation accumulates into the dynamic per-MAC energy attribution. Adding on-chip memory is a necessary step toward maintaining device-level efficiency gains, but it is not sufficient: the memory's delivered bandwidth must also be matched to the array's compute throughput.

The sweeps in Figure 6.11 are an apples-to-apples comparison: every parameter that

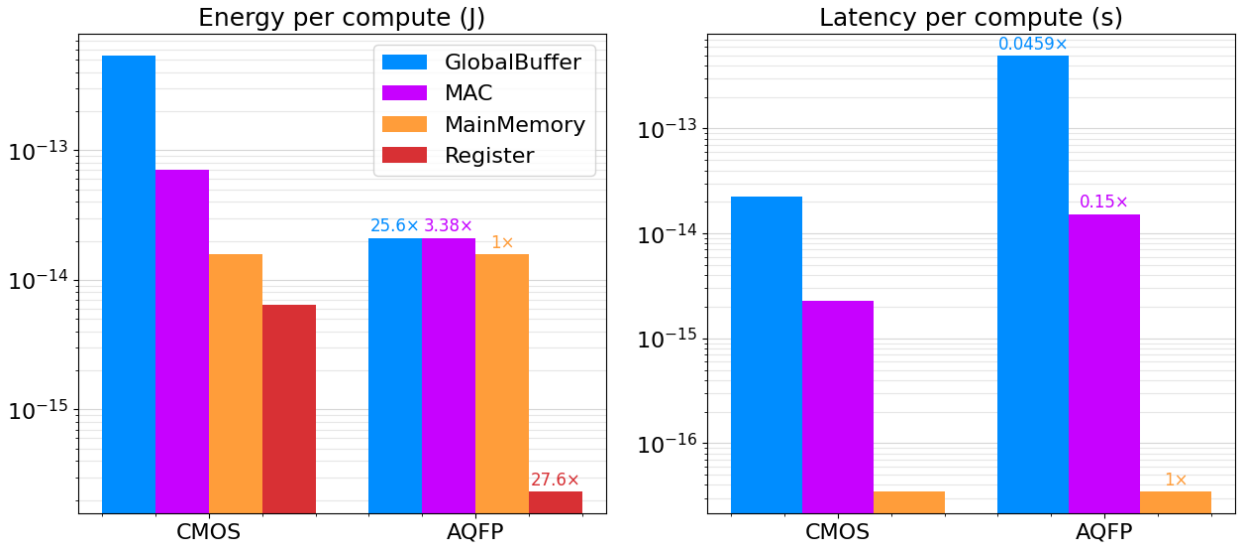


Figure 6.12: Per-component energy and latency breakdown for the PE array with 1 GB global buffer (GLB width = 1024, loop depth = 1024, GLB derate = 8, array dimension = 256) running the matmul workload. The MAC component achieves only $\sim 3\times$ energy reduction relative to CMOS, well below the device-level $\sim 100\times$, because data movement stalls at the GLB and idle MACs continue to dissipate baseline AQFP power.

applies to both technologies is swept identically for AQFP and CMOS. The two AQFP-specific parameters (loop depth, GLB derate) have no CMOS counterpart, so the CMOS baseline is held at the default value when those axes are swept. This comparison answers the question "in what region of the architecture parameter space does AQFP provide a system-level energy benefit over CMOS?", but it deliberately does not answer the question "in what region does AQFP beat the best-known CMOS design?".

The latter requires a fixed, well-optimized CMOS reference, which we discuss in future work.

TPU-like memory hierarchy. The single-level GLB PE array shows that on-chip memory throughput, not capacity, sets the achievable AQFP energy advantage in this architecture. A natural next step is to add another level to the on-chip memory hierarchy so that delivered bandwidth at the array can be increased without scaling the entire on-chip memory at the same access rate. We follow the TPUv4i organization [105]: a single on-chip global buffer fans out to four 128×128 MAC arrays, each backed by a local buffer co-located with the array. Our configuration deviates from TPUv4i in one respect: we use a 256 MB global buffer rather than the 128 MB CMEM of TPUv4i. The architecture is therefore TPU-like rather than a faithful TPUv4i model.

The added on-chip capacity also lets us do more fusion, therefore running the workload

more efficiently. We move from the single matmul of the previous studies to GPT-3 6.7B parameter inference, evaluating both the prefill step (compute-bound, full prompt processed in parallel) and a single decode step from the KV-cache regime (memory-bound, one new token per batch entry). Batch size is 100 in both cases; the prefill prompt is 2048 tokens and the decode workload generates a single new token per batch entry on top of the cached prefix. We continue to assume an idealized 4 K to room-temperature memory interface; this assumption is increasingly speculative as we move toward a more complete system model and is revisited in the next Section.

The results in Figure 6.14 show two features worth noting. First, the AQFP on-chip memories themselves outperform their CMOS counterparts by more than their device-level advantage of $\sim$ 100×. The AQFP global buffer at derate = 16 has a 326× energy reduction per access relative to the CACTI-modeled SRAM baseline, and the AQFP local buffer realizes nearly 200×.

A possible mechanism behind the AQFP memory advantage is worth stating explicitly because it is not just a consequence of low device energy. CMOS SRAM stores data passively in cross-coupled inverters and pays an active access cost on every read or write to drive bitlines, sense amplifiers, and decoders. AQFP SR-Loop memory stores data by continuously circulating it through clocked feedback loops; because AQFP devices are inherently clocked and their data motion is part of normal operation, the per-access cost of moving stored data into the array is largely absorbed into baseline circuit operation rather than incurred as a discrete access energy. Clock derating amplifies this advantage by reducing per-cycle dissipation in proportion to the bandwidth reduction. Whether this scaling holds for very large memories is an open question: maintaining signal integrity across a multi-bank superconducting memory will require additional buffering and likely additional device types for long-distance data motion,

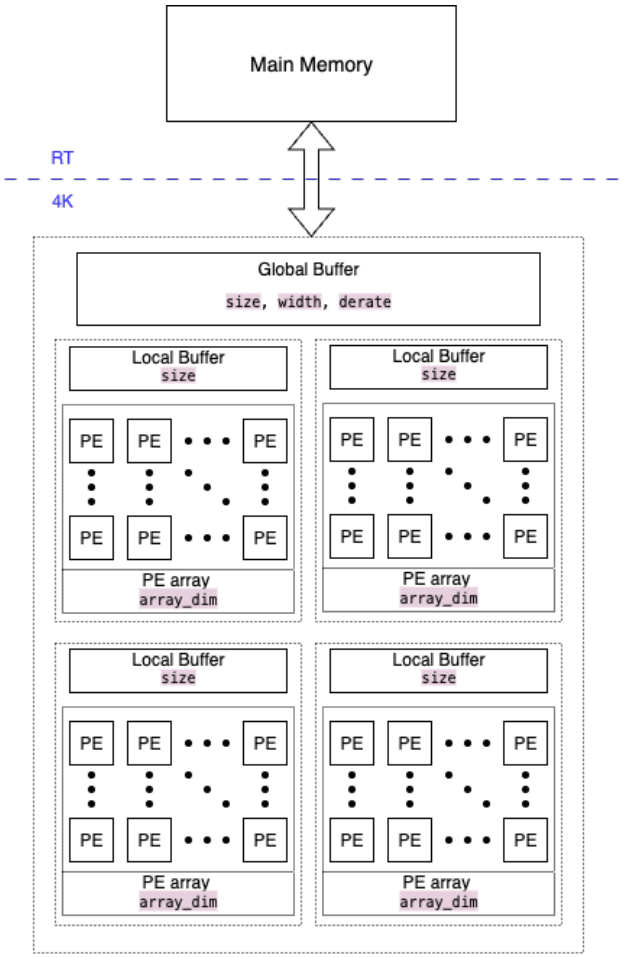


Figure 6.13: TPU-like architecture with on-chip memory hierarchy: a shared global buffer feeds four 128 × 128 MAC arrays, each with a co-located local buffer.

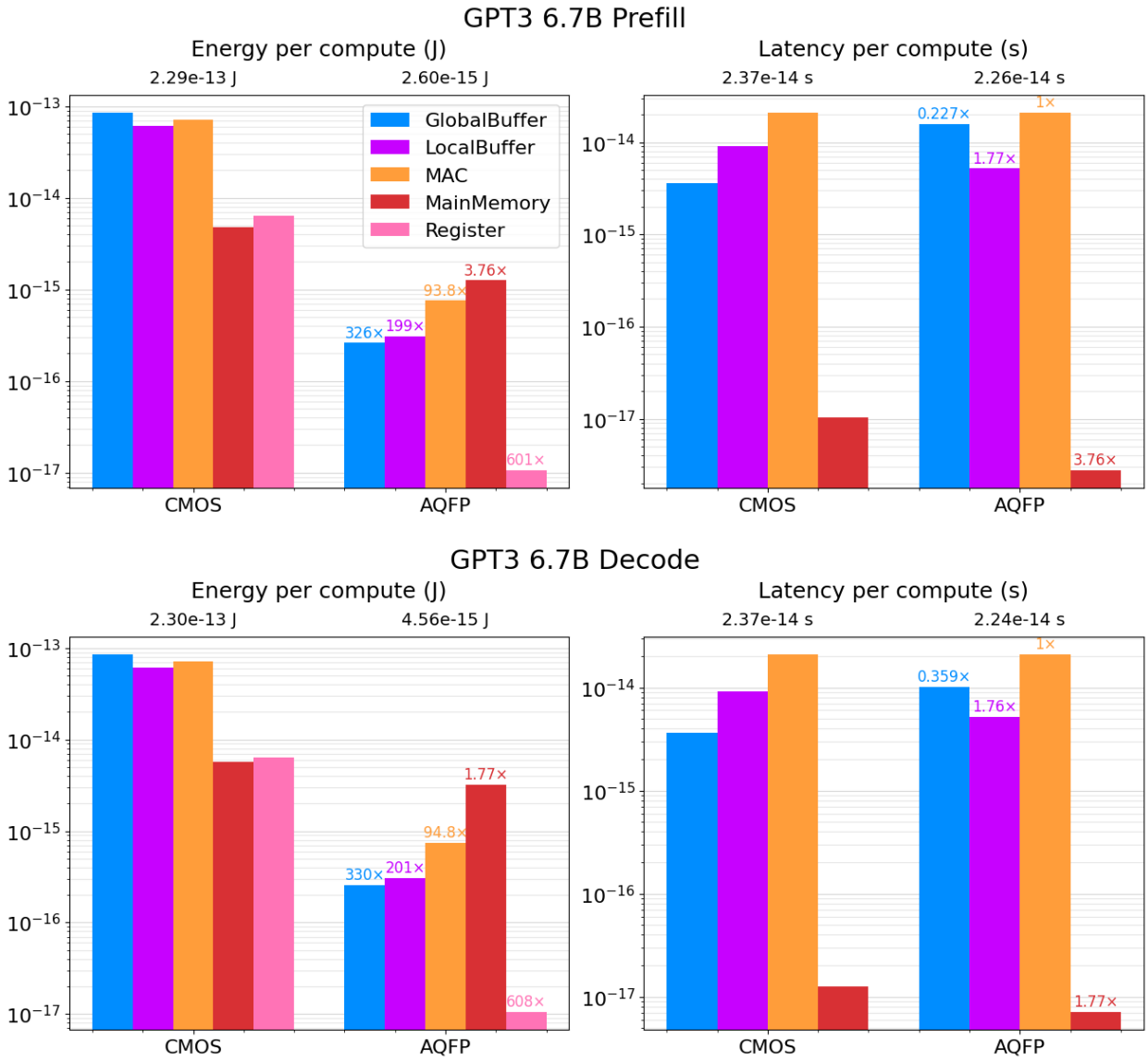


Figure 6.14: TPU-like architecture (Figure 6.13) running GPT-3 6.7B prefill and KV-cache decode workloads. Batch size = 100; decode advances by one token per batch entry. Configuration: 256 MB global buffer, 4 MB local buffer per MAC array, MAC array dimension = 128, AQFP global buffer derate = 16. AQFP achieves 88× system-level energy reduction relative to CMOS in the compute-bound prefill step and 50× in the memory-bound decode step.

both of which would erode the per-bit energy advantage. The model here treats the SR-Loop scaling as well-behaved, and we take the resulting projection as an upper bound on what AQFP on-chip memory can deliver rather than a guarantee.

Second, AQFP shows a larger energy advantage in the compute-bound prefill workload (88×) than in the memory-bound decode workload (50×), as expected. In prefill, weights

loaded from main memory are reused across all $100 \times 2048 = 2.05 \times 10^5$ tokens in the batch, so per-MAC arithmetic intensity is high and the cold compute path carries most of the workload energy. In decode, each weight is reused across only the 100 tokens of the batch, so main-memory traffic per arithmetic operation is $\sim 2000\times$ higher. The difference in main memory energy and latency for CMOS versus AQFP is the mapper finding different optimal mappings for each technology.

6.4.3 Temperature stage interconnect

Finally, to make the system model more realistic, we incorporate the interconnect component models of the previous section into AccelForge runs, accounting for the cost of moving data between the 4 K stage and room temperature.

We first ask how interconnect bandwidth, set by the channel count, affects overall workload performance. Figure 6.15 shows energy and latency per compute as the ingress channel count is swept for the Stripline-RSFQ/AQFP ingress model, with no interconnect modeled on egress. We use this model as a baseline because it is the most readily realizable configuration today. The workloads are GPT-3 6.7B parameter prefill and decode, batch size 100.

In prefill, when the accelerator is compute-bound, per-compute energy is minimized at one ingress channel and grows with channel count, since additional channels add interconnect leakage and dynamic power without relieving any bottleneck. Latency per compute drops from $\sim$0.6 ns at one channel to the CMOS baseline of $\sim$0.25 ns by eight channels, indicating that even in a compute-bound workload there is enough ingress traffic at very low channel counts to incur a measurable latency penalty.

In decode, the accelerator is memory bandwidth-limited, and the energy trade-off changes character. The minimum per-compute energy occurs at four channels rather than one. The mechanism is the same one identified in the global buffer analysis (Section 6.4.2). When AQFP compute units stall waiting for memory, the constant per-cycle leakage of the AQFP datapath continues to accumulate energy without producing work. Adding ingress channels reduces these stalls and the leakage cost falls faster than the interconnect cost rises, until the two trends cross at four channels. The latency penalty at low channel count is also larger in decode than in prefill ($\sim$1.6 ns at one channel versus $\sim$0.6 ns), reflecting the workload's greater sensitivity to ingress bandwidth.

System-level results with interconnect and memory Holding the ingress channel count at 8 to maintain comparable throughput to CMOS with minimal decrease in energy efficiency, we now compare the AQFP architecture across a range of interconnect and main memory configurations. Figure 6.16 shows per-compute energy for the GPT-3 6.7B prefill

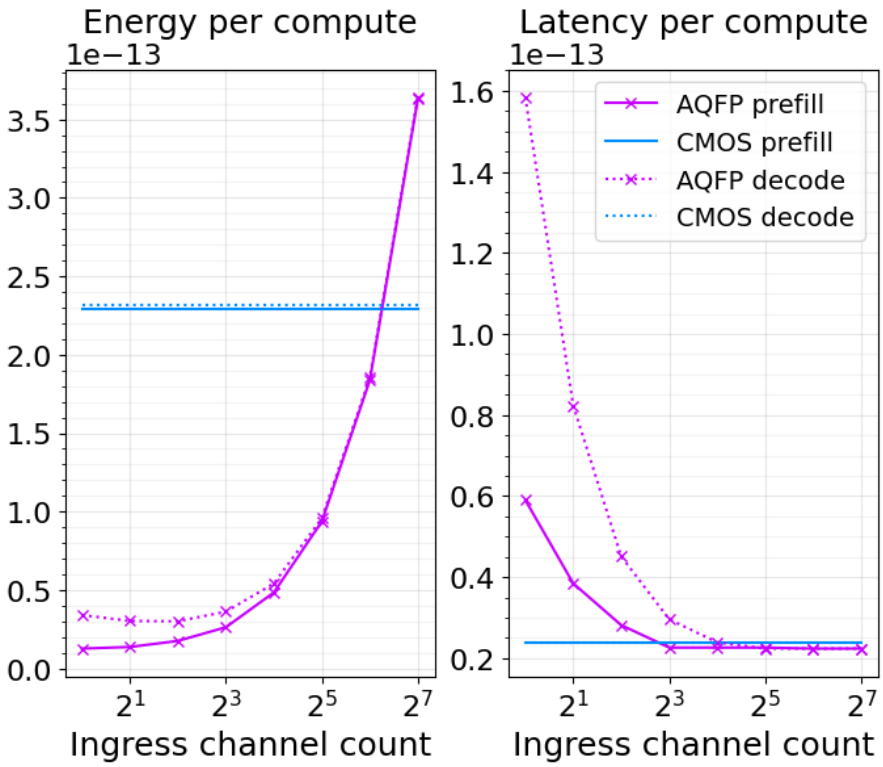


Figure 6.15: Per-compute energy and latency for TPUv4-like architecture running GPT-3 6.7B prefill and decode, batch size 100, sweeping ingress channel count for the Stripline-RSFQ/AQFP model. AQFP architecture uses the interconnect on ingress only; CMOS baseline shown for reference.

and decode workloads on the TPUv4-like architecture, with energy efficiency relative to a CMOS baseline reported above each cluster. The accelerator architecture is identical across all configurations, following the TPUv4-like design with a 256 GB global buffer fanning out to 4 MB local buffers, each with 128-dimension PEs. Configurations differ only in the temperature stage interconnect or the main memory technology.

Reading from left to right: the first two clusters show the CMOS and AQFP ideal baselines from Figure 6.14. The next four clusters include cryostat interconnect models: `Fiber-AQFP` and `Stripline-RSFQ/AQFP` on ingress only (no egress cost), followed by the same two ingress models paired with the `Stripline-AQFP` egress. The final two clusters replace the room-temperature main memory with the nMem and CryoHBM cryogenic main memory models, eliminating the need for ingress and egress cryostate cabling entirely because all data movement now occurs at the 4 K stage.

The AQFP ideal sets the upper bound on what any realistic configuration can achieve, since it captures only the AQFP datapath energy. We see that the choice of interconnect technology is the largest single determinant of system-level efficiency. Restricting attention

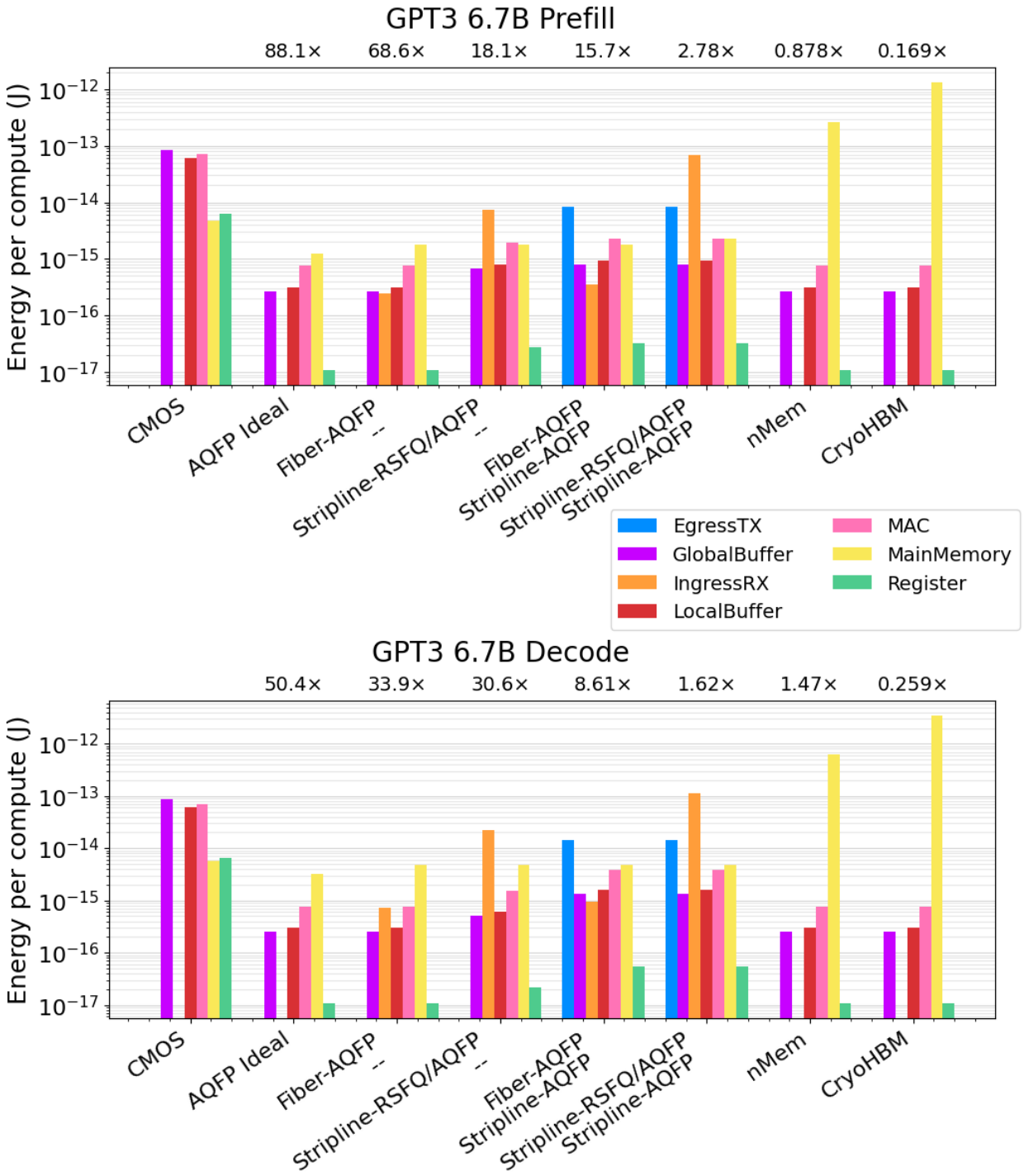


Figure 6.16: GPT3 6.7B parameter prefill and decode workload on the base TPUv4-like accelerator with different interconnect and main memory technology configurations.

to ingress-only configurations to isolate the effect of the interconnect from egress overhead, we focus on the two most energy-efficient ingress models: `Fiber-AQFP`, an optical link based on projected performance of an AQFP-compatible receiver, and `Stripline-RSFQ/AQFP`, an electrical link derived from demonstrated circuits [59]. `Fiber-AQFP` retains 68.6× in prefill and 33.9× in decode, while `Stripline-RSFQ/AQFP` retains 18× in prefill and 30× in decode. Adding the `Stripline-AQFP` egress drops both ingress configurations by 4–6×, an overhead dominated by the stripline cable heat load.

The two cryogenic main memory configurations, nMem and CryoHBM, both fall at or below the CMOS baseline in prefill (0.878× and 0.169× respectively) and remain near it in decode. The on-chip power dissipation of cryogenic main memory is comparable to that of its room-temperature counterpart (see Table 6.5), but at 4 K that dissipation incurs the 1000× cooling overhead, which dominates the energy budget. CryoHBM is worse than nMem because nMem uses a superconducting memory technology that is intrinsically more energy efficient than the cryoCMOS DRAM storage cells. The architectural conclusion is that, given today's superconducting memory capabilities, main memory should remain at room temperature, and the system should pay the cost of crossing the temperature stage interconnect rather than bringing the memory to 4 K.

Taken together, these results identify a clear engineering path. Optical AQFP-compatible interconnects already approach the AQFP ideal on the ingress side and represent the leading direction for closing the remaining gap to the ceiling. The active bottleneck is egress, where a low-power optical egress matched to AQFP signal levels has not yet been demonstrated. The `Fiber-MonoEO` model from [126] suggests a viable direction, but in that design the CMOS amplifier and laser coupling introduce more power dissipation than the electrical stripline connection. A complementary approach is to power-gate the egress interconnect. In workloads where all Einsums can be fused into a single dataflow, egress is active only at the end of the computation, so a galvanic latch or similar low-leakage gating mechanism could eliminate egress overhead during the bulk of the workload. In the meantime, the `Fiber-AQFP` ingress paired with `Stripline-AQFP` egress already delivers a 15.7× energy reduction on prefill and 8.6× on decode.

For context, this efficiency range is comparable to the gains delivered by major shifts in commercial accelerator generations. The original TPUv1 delivered 30–80× better performance per watt than contemporary CPUs and GPUs on Google's production inference workloads [101]. This result justified the deployment of domain-specific accelerators across Google's datacenters. The present work shows that AQFP, evaluated at the full system level with realistic interconnect, delivers 8–15× system-level efficiency over a recent TPU baseline with technology today, and a path towards 34–69× in the future. The fact that a new compute

Table 6.8: JJ count and power dissipation for AQFP accelerator architectures.

Architecture	JJ Count	GLB Derate	Dissipation at 4 K (mW)
256×256 LinCore Array	1.6 B	—	0.0745
256×256 PE Array 128 Mb Global Buffer	10.92 B	16×	0.077
		8×	0.080
		1×	0.489
TPUv4-like 256 Mb Global Buffer 4 Mb Local Buffer 128 PE array	40.02 B	16×	1.00
		8×	1.11
		1×	1.79

substrate can match the architectural-generation efficiency gap while running on the same workload is the central result of this chapter.

6.4.4 Cryostat sizing

It is worth briefly checking how large the architectures we have been discussing are, in terms of JJ count and total on-chip dissipation. Table 6.8 summarizes each of the accelerator architectures evaluated in the previous sections, listing their JJ count and on-chip dissipation at various levels of GLB derate where applicable.

The smallest configuration, a 256×256 LinCore array with no global buffer, dissipates 75 μW. The PE array variant with a 128 Mb global buffer, discussed in Section 6.4.2, dissipates between 77 μW and 0.5 mW depending on the buffer derate factor. The largest configuration evaluated, a TPUv4-style organization with a 256 Mb global buffer, 4 Mb local buffers, and a 128-PE array totalling 40 billion Josephson junctions, dissipates 1.79 mW at the most aggressive (1×) derate setting and under 1.1 mW with larger derate factors.

These dissipation levels fit comfortably within the cold-stage thermal budget of a small commercially available rack-mount cryocooler. The attocube attoCMC, for example, is a self-contained 19-inch rack-mountable system (10U) that delivers 50 mW of cooling power at 2.8 K. It runs from a single-phase wall outlet and consumes less than 1.4 kW of wall-plug power [130]. The largest of the evaluated architectures uses less than 4% of this 50 mW budget. To put this in perspective, a typical microwave uses ∼1 kW of power, so the largest TPUv4-like AQFP accelerator could easily be cooled by the power required to run a common kitchen appliance.

6.5 Future work

The design studies in this chapter are early-stage results and only begin to exercise what AccelForge can do for superconducting systems. Several directions stand out.

First, the comparisons here have been apples-to-apples between AQFP and CMOS in architecture specification, with the mapper finding optimal mapping for each technology implementation. This guarantees a fair comparison of the underlying technologies but means that the CMOS baseline is not necessarily its own best design. Future work should hold a well-optimized CMOS reference fixed and explore the AQFP architecture design space on its own terms.

Second, interconnect emerged as one of the most consequential design choices for system-level efficiency, and it is also the area where our component models rest on the thinnest evidence. Most of the interconnect references are drawn from the quantum computing literature, where power consumption can be a secondary concern. Validating these models and designing cryogenic drivers and receivers specifically for AI inference workloads, with power as the primary metric, is a clear next step.

Third, the architectures evaluated in this chapter are well beyond the area of a single fabricated reticle, yet our cost models assume monolithic integration with no superconducting chip to chip overhead. Future work should incorporate chip to chip energy and latency costs into AccelForge's component models and revisit the partitioning of the PE array and global buffer under that constraint. Promising packaging substrates, such as superconducting multi-chip modules [131] and 3D volume mount devices [132], should be explored for their different bandwidth, energy, and density.

Additionally, the component models developed in this chapter, spanning AQFP, RQL, cryogenic memory, and temperature-stage interconnects, lay the groundwork for evaluating heterogeneous systems that combine technologies in a single system. A natural target is a hybrid architecture in which compute-bound operations are dispatched to an AQFP accelerator at 4 K, memory bandwidth-bound operations to a room-temperature CMOS accelerator, and potentially latency-sensitive operations to an RQL accelerator that trades energy efficiency for throughput. The component models are ready and the AccelForge framework has the ability to explore this type of heterogeneous system design.

Finally, the cryostat itself should become a design space dimension rather than a fixed overhead ratio. The TPUv4-like AQFP architecture in Section 6.4.4 uses less than 4% of the cooling budget of the small rack-mount attoCMC cryocooler [130]. A single accelerator therefore underutilizes the available cooling capacity, and the natural question is how to fill a given cryostat with a mix of architectures and workloads. Incorporating cryostat sizing into

an AccelForge post-processing step would let the tool answer this question directly.

Underlying all of these directions is the need for end-to-end measured validation. The motivation for this tool, as described in Section 6.1, is that superconducting electronics has lacked the kind of early-stage architectural modeling infrastructure long standard in the CMOS accelerator community. The work in this chapter closes that gap on the modeling side, but the projections it produces cannot yet be calibrated against measured systems at scale. The natural next phase of this project is to identify the most promising architectures, build them, measure them, and feed the results back into the component models. That feedback loop is what would turn superconducting computing from a collection of circuit-level demonstrations into advanced computing systems that can rival CMOS performance and efficiency.

Chapter 7

Conclusion and Outlook

The premise of this dissertation, stated in Chapter 1, was that AQFP device-level energy advantage over CMOS is well established but has not yet translated into a credible path to system-level computing. The intervening chapters set out to close that gap with contributions distributed across the hardware abstraction stack, from circuit primitives to a full-stack modeling tool.

With this work in place, AQFP technology now has:

a **phase synchronizer** (Chapter 3), the first AQFP primitive that captures data without strict phase alignment to the local excitation clock. By absorbing timing uncertainty between source and destination clocks, the synchronizer removes a prerequisite obstacle to clock domain crossings in AQFP or multi-chip module integration.

an **SR-Loop register file** (Chapter 4) that departs from the CMOS-derived random-access organization inherited by prior superconducting memories, and in turn achieves higher density. The v2 design stores 1 Kib in a $4 \times 4 \times 64$-bit feedback-loop configuration at 760 bit/mm^2 effective density, a 380× improvement over the D-latch register file of the MANA microprocessor at comparable read latency and 12× lower energy per bit. The cost is sequential access with longer latencies; however, the rest of the dissertation posits that this penalty can be absorbed by a microarchitecture co-designed with the storage's circulation period.

a design for a **high-throughput linear algebra accelerator** (Chapter 5), LinCore, that does exactly that. By combining barrel scheduled fine-grained multithreading with the SR-Loop register file in a closed execution-storage loop, LinCore keeps a deeply pipelined AQFP datapath fully occupied at one operation per cycle without additional control logic. The closed-loop execution-storage pattern is independent of the specific

ALU and provides a template for AQFP compute beyond linear algebra.

a **full-stack architecture design space exploration tool** (Chapter 6), AccelForge, for superconducting electronics, which adapts established CMOS accelerator modeling methods to AQFP and RQL through cooling-aware component models for compute, cryogenic memory, and temperature-stage interconnects. Under idealized assumptions that exclude cryogenic to room temperature interconnect cost, AccelForge modeling found that a TPU-like AQFP architecture with 256 MB of on-chip memory achieves up to 88× system-level energy reduction over a 7 nm CMOS baseline on the compute-bound prefill phase and 50× on memory bandwidth-bound decode. Once interconnect costs are included, the architecture retains 8 - 15× efficiency gains, with headroom to improve further from better mapping and modeling exploration. For context, this matches the architectural-generation efficiency gap that justified Google's deployment of its first custom AI accelerator across its data centers.

7.1 Next steps

Each of the contributions above leaves immediate experimental and design work to be done. The weak constant variant of the synchronizer fabricated in Chapter 3 has a data threshold $\sim 90\,\mu$A, well above the 15–30 μA operating range of the surrounding AQFP logic; reducing the excitation signal asymmetry to bring the threshold into range is the remaining circuit-level task. The v2 SR-Loop register file has returned from fabrication and characterization is in progress. Furthermore, demonstrating an SR-Loop memory at intermediate scale between the 1 Kib register file and the MB size buffers projected in Chapter 6 is an important next step needed to validate the on-chip memory hierarchy projections from AccelForge. Additionally, the v1 LinCore die is unlikely to function as designed because of the storage cell coupling and routing-asymmetry defects identified after tape-out; the v2 design addresses both, and bringing it through simulation, fabrication, and measurement is the next milestone. Beyond v2, two LinCore directions extend naturally: a fused multiply-accumulate unit, which would roughly double matrix-multiplication throughput, and a layout-aware schematic design methodology.

Finally, AccelForge itself rests on thin evidence in the interconnect models, which are drawn largely from quantum-computing references where power is a secondary metric. Validating these models and designing cryogenic drivers and receivers specifically for AI inference workloads, with power as the primary metric, is the most consequential modeling-side gap. End-to-end measured validation of any of the projected architectures remains the open

problem that the modeling effort was built to enable.

7.2 Forward looking

The contributions of this dissertation are sized to fit on a single fabricated die. The energy problem motivating the dissertation is sized to a national power grid. The vision animating this work is a cryogenic compute substrate that delivers orders of magnitude more compute from today's power grid by approaching the fundamental energy limits of computation. Together the prior work and contributions from the preceding chapters provided a foundation, but what follows is the path from a single AQFP accelerator die to a cryogenically cooled datacenter.

The first engineering frontier needed to realize this vision is in demonstrating the cluster of dies that together would comprise a single AQFP accelerator. The largest configuration evaluated in Chapter 6 totals roughly 40 billion Josephson junctions, which is several orders of magnitude beyond the largest AQFP circuit demonstrated to date. The route from current fabrication scales, at millions of junctions, to billions of AQFP is not a single chip but a multi-chip integration problem. A multi-die AQFP module, with each die running its own excitation clock and data crossing between dies through synchronizers, is the natural unit of compute at the next scale up from the single die. Establishing how many dies can be co-packaged before substrate-level thermal or interconnect constraints bind, and how an SR-Loop memory hierarchy partitions cleanly across that boundary, is the architectural problem at this scale.

The second frontier is the temperature stage boundary. The AQFP compute operates at 4 K, but the world it serves runs at 300 K. Every byte that crosses this boundary carries a thermal cost from the cabling and a power cost from the drivers and receivers on either end. The interconnect study in Section 6.4.3 identified this as the single most consequential design choice for system-level efficiency, and the bottleneck on the egress side, where a low-power optical link matched to AQFP signal levels has not yet been demonstrated, remains the open hardware problem. A cryogenic datacenter is not a single 4 K stage with a wide I/O cable to room temperature; instead, it is a thermal hierarchy, with intermediate stages at $\sim$40 K and $\sim$77 K that can be exploited to amortize the conduction load of the interconnect or distribute the compute. Designing the rack-scale interconnect fabric for a cryogenic compute system is therefore a co-design problem across the temperature stages, and it is where the cooling hierarchy, the interconnect material stack, and the signaling protocol all meet.

Additionally, a large-scale system can play with heterogeneity. The AccelForge studies make a clear architectural case for AQFP on compute heavy matrix-multiply operations,

but it has limited comparative advantage on the long tail of irregular, control-flow-heavy, sparse, or otherwise non-tensor workloads that conventional silicon handles well. A deployable cryogenic compute system is therefore not an AQFP datacenter but a heterogeneous one, with AQFP accelerators positioned as domain-specific inference engines attached to a CMOS host through the temperature-stage interconnect fabric described above. The component models developed in Chapter 6 already support this kind of heterogeneous reasoning across AQFP, RQL, and CMOS in a single system specification. The design question is now how to partition a real production workload across the cryogenic and room-temperature tiers, and how the resulting data movement reshapes the rack-scale interconnect requirements.

Finally, we come to the cryocooler itself. Existing commercial cryocoolers fall into two regimes: small cooling capacity (< 10 W) cryostats, or large capacity (> 10 W) helium reliquefier systems. Small capacity cryostats have a higher specific power, but the benefit of modularity and low integration overhead. Meanwhile, large capacity helium reliquefier systems can reach impressive specific powers on the scale of 500 W, but they are designed for monolithic cold-volume loads with a lot of infrastructure overhead. These large systems are intended for use cases where the cold volume is held across the operational lifetime of the system. However, a datacenter must be repairable at the granularity of a single rack on a maintenance shift. Warming up a centralized cryoplant to service a single failed cable, then cooling the entire installation back down over the days or weeks such a cycle requires, is incompatible with the operational model that makes datacenter compute economically deployable. Therefore, the right cryocooler for a cryogenic datacenter sits between these regimes: large enough in aggregate cooling capacity to approach the IRDS specific-power median, but modular at the granularity of a single rack or a small cluster of racks, so that any individual cold stage can be warmed and serviced independently of the rest of the installation. Closing this gap is a cryogenic engineering problem that should be co-designed with the architecture.

The opening of this dissertation observed that today's silicon transistors dissipate energy roughly five orders of magnitude above the Landauer limit, while the world's AI inference workloads are hitting a power wall. The AQFP device sits within roughly two orders of magnitude of the Landauer limit at room-temperature equivalent energy, leaving three orders of magnitude of headroom between where superconducting compute can reach today and where the laws of physics will eventually stop it. The contributions of this dissertation provide initial steps for this trajectory.

Appendix A

Measuring AQFP energy dissipation

This appendix documents an attempt to measure the on-chip energy dissipation of the MITLL AQFP cell library. The measurement technique is not novel; analogous experiments have been performed on other AQFP processes by Takeuchi and colleagues [8],[133], and our goal was to reproduce that approach for the MITLL SFQ5ee process. The attempt was ultimately unsuccessful, as the instrumentation available to us lacked the amplitude resolution required to resolve the predicted signal. We include this memo here for two reasons: as a pedagogical walkthrough of the AC return-power differential method applied to a small AQFP shift register, and as a documentation of the practical sensitivity limits that future measurements on this cell library will need to overcome. The remainder of this appendix presents the measurement approach, the equipment configuration, the post-processing pipeline, and an analysis of why the existing setup falls short of the required noise floor.

Goal: Measure on-chip energy dissipation of an AQFP shift register and extrapolate the energy per device per switching event

Approach: Exploiting the fact that AQFP switching is controlled by both an AC clock bias *and* DC clock bias, we can observe the AC clock return power with and without the devices switching by toggling the DC bias. We compare the AC return signals with DC ON (AQFPs switching) versus DC OFF (AQFPs not switching) and any change in the amplitude/power is attributed to energy absorbed by the active AQFPs.

Key expectation: The chip we've tested has ∼2,300 AQFPs, so the predicted change in AC return peak amplitude is on the order of ∼0.5 nV when clocking at 1 MHz.

What we did: Collected long averaged scope data (15 min and 1 hr) at 1, 5, and 10 MHz; then implemented coherent mixing + low pass filtering to extract tiny amplitude

steps

Result: The measurement is not resolvable with the current equipment setup because the scope data are effectively quantized at 2.44 μV (averaged analog signal is digitized for storage to 16-bit word over a 160 mV scope range). This is about 10^3–10^4 times larger than the expected nV-scale amplitude modulation from the energy dissipation, so the desired signal is lost before post-processing.

Conclusion: With the current instrumentation and acquisition format, we can't measure the AQFP energy via AC return amplitude difference. **We need to repeat the measurement with a spectrum analyzer** that can preserve the nV-scale amplitude modulation on the ∼60 mV peak clock return amplitude.

A.1 Prior Work

Takeuchi et al. [8] reported an experimental energy measurement on an AQFP 8-bit carry look-ahead adder by modulating whether switching occurs with the DC clock line and observing the resulting change in clock-line power using a spectrum analyzer (Anritsu MS2830A). Their method is conceptually similar to ours: switching activity is controlled by the biasing condition, and dissipation is inferred from the difference in returned clock power between "active" and "inactive" states.

They collect the energy dissipation at 5 GHz because at this high-speed it was absorbing enough power to be detectable on their spectrum analyzer. They validated correct operation of the circuit at 1 GHz, but could not measure the energy at this lower frequency because the power was too small. Above 1 GHz frequency, there were more logic errors in the circuit output, but the paper states "all the AQFP gates in the 8-bit CLA were expected to be performing switching operations during the power measurement because the excitation currents supply magnetic flux equally to all the gates".

The paper measures a difference of less than 0.05 dBm on very small resolution bandwidth. The resulting power dissipation of the entire chip is 97 nW, which they extrapolate to 7.5 nW for the 8b CLA, and therefore 1.4 zJ energy dissipation per junction.

Herr et al. [134] demonstrated an experimental power measurement for an RQL 8-bit CLA, also using clock return signals, but with a modulation technique that explicitly creates measurable spectral features. Rather than a static "on/off" comparison, they periodically alternate the input data between an all-zeros pattern (minimal switching) and a pseudo-random pattern (high switching activity). This chopping modulates the clock amplitude at the chopping frequency, producing sidebands around the clock carrier that can be measured on a

spectrum analyzer. The inferred dissipation is proportional to the ratio of sideband power to carrier power. Because sidebands can arise from both amplitude modulation (AM, associated with true power draw) and phase modulation (PM, associated with switching-dependent impedance and propagation delay from the Josephson junctions coupled to the clock line), they treat the purely-AM interpretation as an upper bound and then apply a correction based on prior measurements indicating comparable AM and PM contributions. Using this method, they report a dissipation of ∼510 nW for the CLA core at ∼6.2 GHz.

$$\frac{\Delta P}{P_0} = 2\pi\sqrt{\frac{P_{SSB}/2}{P_0}} = 8 + \frac{1}{2}(SSB - 3) \text{ (in dB)} \tag{A.1}$$

Finally, both of these studies achieved on-chip dissipation measurements without requiring special "energy test structures" on chip, which is desirable for our use case since our chips are already fabricated and future designs have a long turn around time. However, if we are willing to dedicate chip area to a measurement structure, resonator-coupled dissipation measurements demonstrated in earlier AQFP work [133] provide an additional path that may improve sensitivity at lower operating frequencies.

A.2 Measurement approach and protocol

The measurement idea is to exploit the fact that the AQFPs only "switch" when the DC bias is present in addition to the AC clock. The clock lines are routed off-chip and terminated at room temperature, allowing us to measure returned carrier power. Therefore, we can monitor the return AC waveform while toggling the DC condition. With **DC on**, the AQFPs absorb energy from the clock network; with **DC off**, the AQFPs remain in the single-well regime and are assumed to dissipate negligibly. Under that model, the AC return amplitude should be slightly smaller with DC on than with DC off, and the difference between the two states is what we're trying to measure. As an exaggerated example, we expect to see something like this where DC is on until 10 μs and then once turned off, we see a large amplitude:

The experiment was performed in a liquid helium dip probe using 50 Ω coax from the PCB (wirebonds + connectors) to room-temperature equipment: a Tektronix AWG5208 for clock and data generation and a Teledyne LeCroy WavePro 254HD for acquisition. The chip we used is from SFQ5B5 with AQFPs from the v1 cell library design and no RF tapers on pads. The clock lines in the v1 design are closer to ∼30 Ω on chip, so we expect impedance mismatch relative to the 50 Ω supply cables. Our assumption is that mismatch/reflections exist in both DC states, so the *difference* between DC on/off should still isolate power absorbed by switching, even if absolute delivered power is not well defined due to the impedance

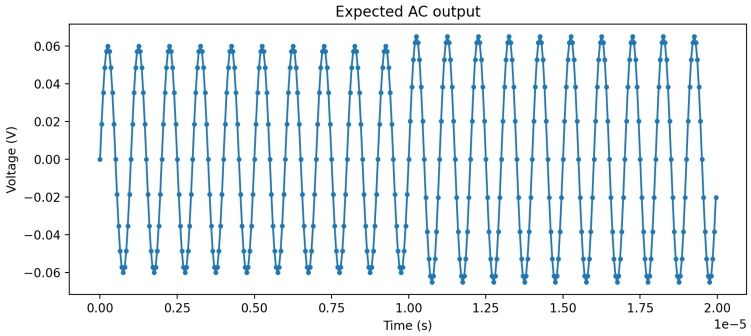


Figure A.1: Exaggerated example of expected AC return amplitude change when toggling DC bias.

mismatches.

A schematic overview of the testing setup is shown below.

There are two conceptual concerns to keep on the radar. First, switching AQFPs can in principle change the effective impedance seen by the clock line, which could create an amplitude difference that is not purely dissipation. Second, if there is non-negligible power absorbed from the DC line itself, then an AC-only return comparison misses part of the energy budget. Neither issue changes the core conclusion below (we are nowhere near the needed sensitivity), but they matter for how we interpret results once we *can* measure in this regime.

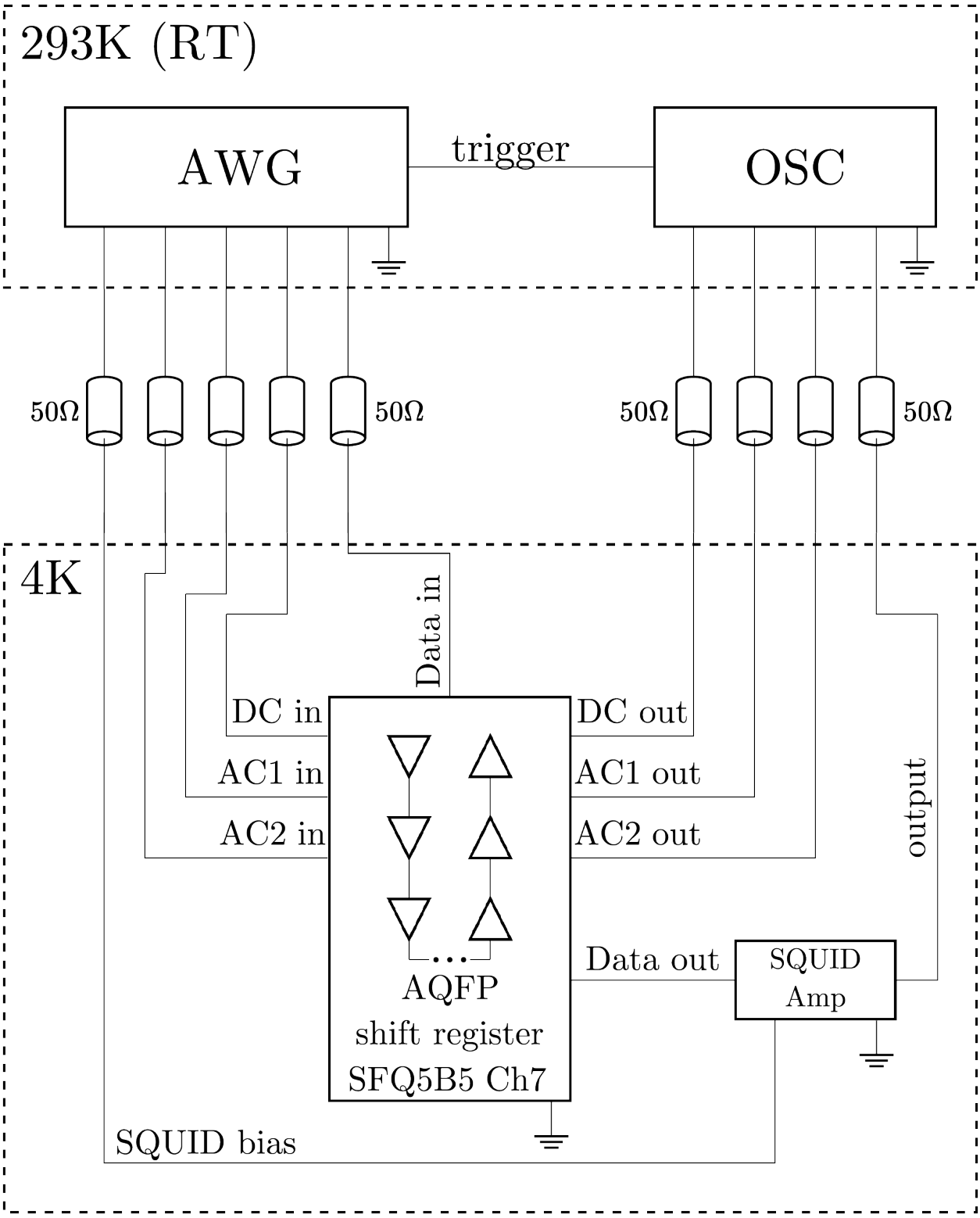


Figure A.2: Schematic overview of the energy measurement testing setup.

A.3 Expected signal size

From Cadence SPICE simulation, a single one of our AQFPs dissipates 6.4e-22 J = 0.64 zJ per switching event. Previously published AQFP measurements report 1.4zJ energy dissipation per JJ, so we can assume 2.8 zJ per device [8]. We will measure the energy dissipated from all of the AQFPs on chip (∼2,300).

Early on, we estimated the size of the amplitude difference to be a few μV by converting "energy per AQFP × frequency" into a power and then mapping that power directly to a voltage using $P = V^2/50\,\Omega$. That logic would be appropriate if we were measuring the entire delivered clock power, but in our case we are measuring a tiny perturbation on top of a large continuous return signal. The correct quantity is the *difference* in power between DC off and DC on, which corresponds to a *small change* in the amplitude of a much larger carrier.

We model the return as $V_{AC} = V_0 \sin(2\pi f_c t + \phi)$ and define V_{ON} and V_{OFF} as the carrier amplitudes with DC ON/OFF. With $\Delta P = P_{OFF} - P_{ON}$ and $\Delta V = V_{OFF} - V_{ON}$, we have:

$$\Delta P = \frac{1}{R}(V_{OFF}^2 - V_{ON}^2) = \frac{1}{R}(\Delta V^2 + 2\Delta V \cdot V_{ON}) \tag{A.2}$$

Since $\Delta V \ll V_{ON}$ (given that it's a small perturbation on a big signal), the ΔV^2 term is negligible and we get

$$\Delta V \approx \frac{R\,\Delta P}{2V_{ON}} \tag{A.3}$$

Assuming a single AQFP dissipates $\sim 10^{-21}$ J per event at 1 MHz gives $\Delta P \sim 10^{-15}$ W = 1 fW per device. In our setup, the return carrier amplitude is ∼60 mV, so the implied amplitude change per AQFP is only ∼0.41 pV. So for about 2,300 devices with each AC line coupled to roughly half of them, we expect

$$\Delta V_{AC1} = \Delta V_{AC2} \approx 0.41 \text{ pV} \times (2300/2) \approx 0.5 \text{ nV} \tag{A.4}$$

So **we are trying to measure sub-nV amplitude difference on a 60 mV peak carrier amplitude.**

A.3.1 Data processing method

Since the clock tone is generated by our AWG, we know f_c very precisely and can do coherent detection to extract the tiny amplitude change. The basic workflow is to multiply the measured return waveform by a complex reference at f_c (i.e., mix to baseband), which produces a component near DC proportional to the instantaneous amplitude and a component at $2f_c$. Low-pass filtering removes the $2f_c$ term and higher-frequency noise, leaving a slow

time trace that should show any step correlated with the DC toggling.

To sanity-check the workflow, I tested the method on synthetic signals in Python. When I embed a 1 nV amplitude step into a sinusoid and add white noise at or below that scale, coherent mixing plus appropriate filtering recovers the step as expected, shown below. Once the noise exceeds the step magnitude, detection fails, also as expected.

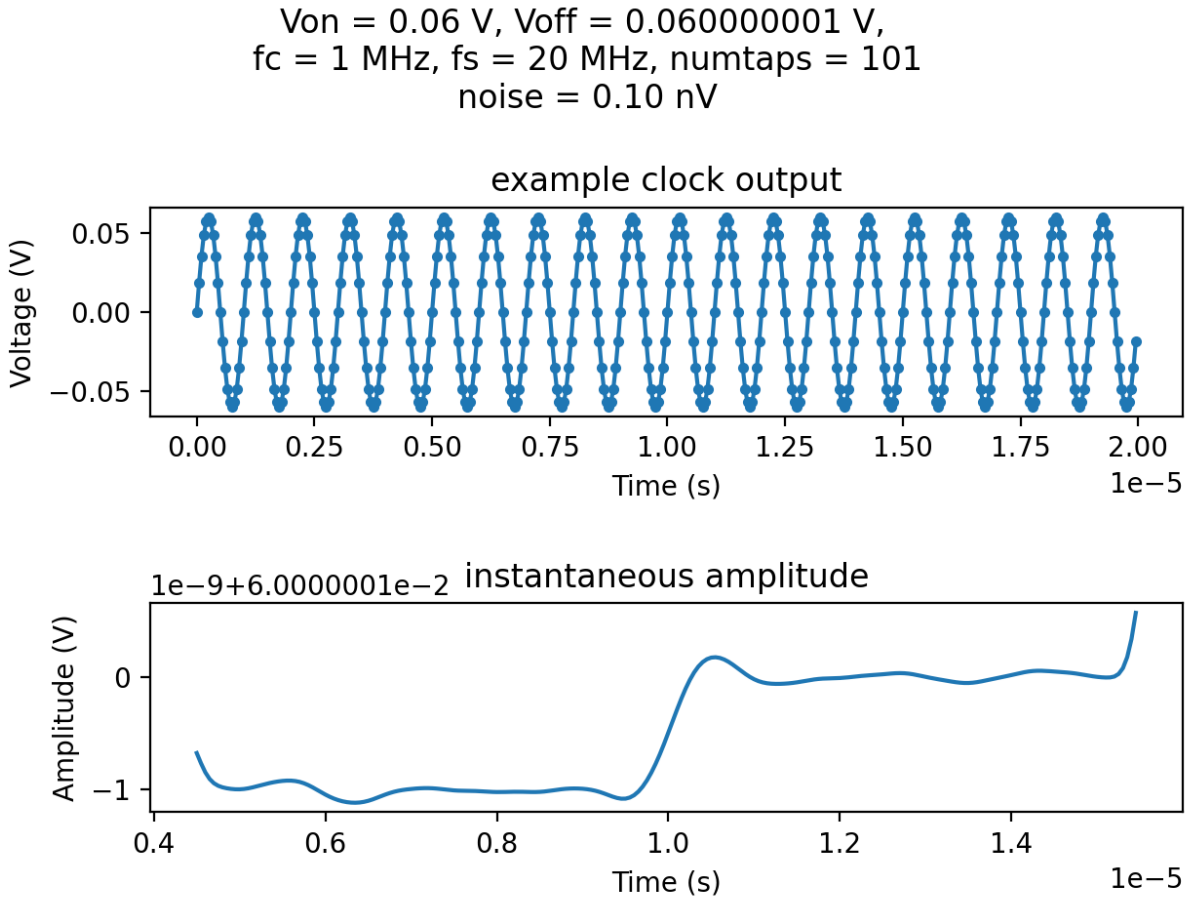


Figure A.3: Synthetic signal test of coherent demodulation pipeline, showing recovery of a 1 nV amplitude step with small additive noise.

A.4 Results

The best dataset we have (lowest clock feedthrough / most stable conditions) was collected in October. We have runs at 1, 5, and 10 MHz and at two long averaging settings: 900 s (15 min) and 3600 s (1 hr), with three repeats per parameter setting. At 1 MHz the AQFP output is present when DC is on and absent when DC is off, which confirms we are toggling the intended operating condition, but the data pulses are difficult to detect due to the clock coupling. At 5 MHz and above the output is not detectable due to the overpowering clock

coupling. The dataset also shows slow drift and "charging-like" behavior related to the DC clock line, the data input, and triggering; this drift is visible in the small AQFP output signals and is likely also present in any attempt to resolve subtle amplitude changes on the clock return.

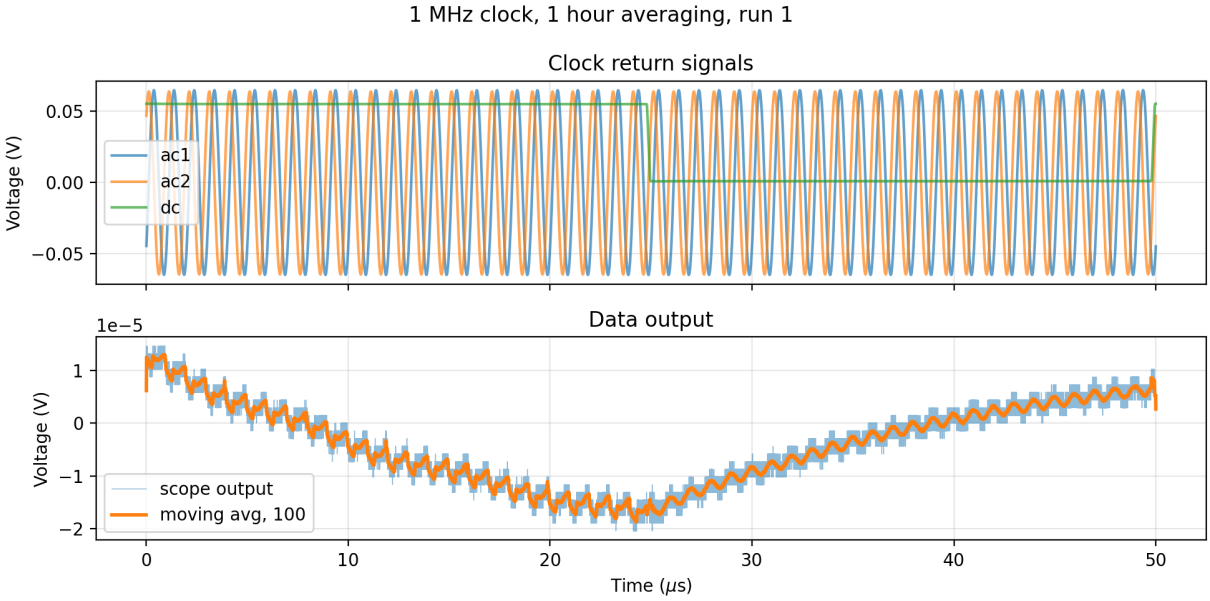


Figure A.4: 1 MHz, 1 hr averaging, run 1.

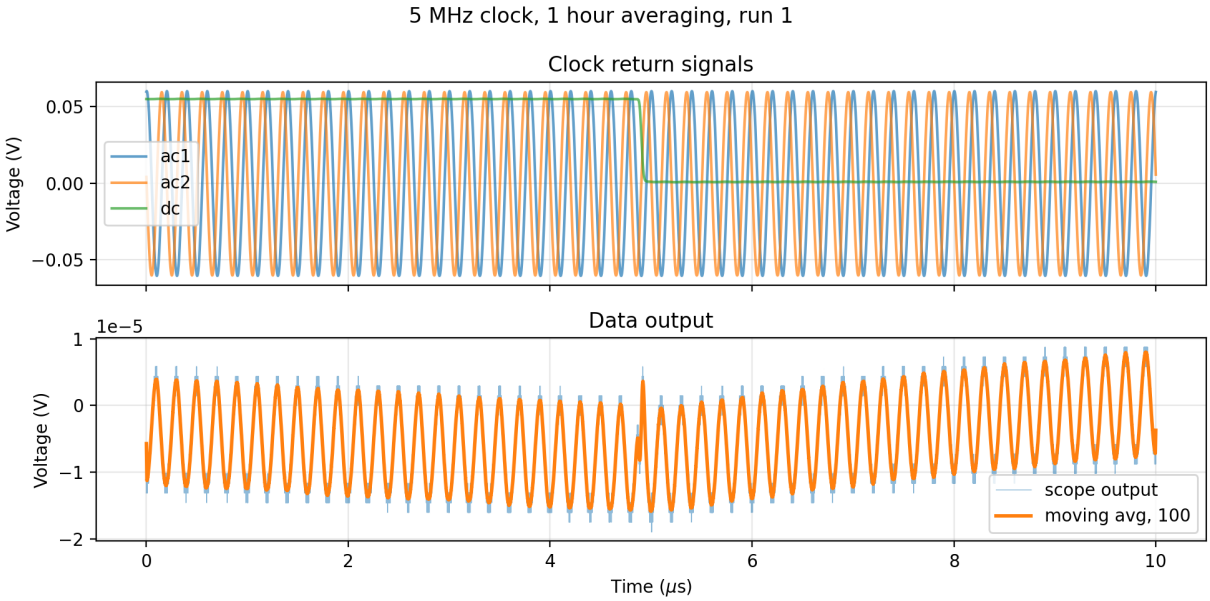


Figure A.5: 5 MHz, 1 hr averaging, run 1.

There are two practical issues that kill us before drift even becomes the dominant problem. First, the variance of the return signal after 900 s averaging is still $\sim$10 μV, and after 3600 s

averaging is $\sim$3 μV. Both of these are obviously many orders above a 0.5 nV target. Second, and more fundamentally, the noise pattern shows clear binning around $\sim$3 μV, which is consistent with data quantization, not just analog noise.

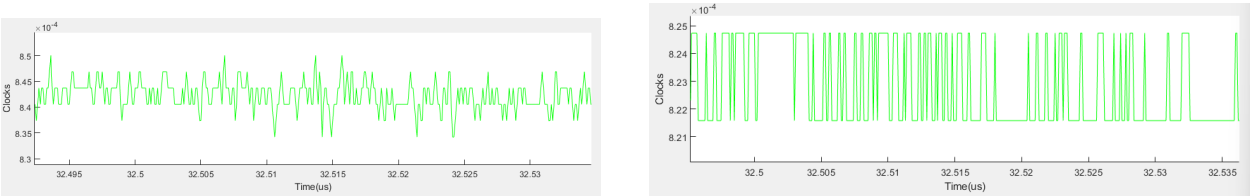


Figure A.6: Left: Zoom in on DC OFF trace after 900 s (15 min) of averaging. Right: Zoom in on DC OFF traces after 3600 s (1 hr) of averaging.

At 20 mV/div (160 mV full scale), the scope's 12-bit ADC LSB is $\sim$39 μV, so long averaging legitimately beats the instantaneous ADC resolution. However, the scope stores the waveform as a **16-bit word**, which at 160 mV full scale corresponds to 160 mV / $2^{16} \approx$ **2.44 μV**. That matches the quantization we see in the saved data. In other words: even if the front-end noise were smaller, the stored samples are effectively snapped to a $\sim$2.44 μV grid, and that destroys any sub-nV perturbation before the coherent demodulation step can do anything useful.

As a final check, I ran the same coherent demodulation pipeline on the real dataset (after decimating for tractable processing). The demodulated amplitude trace shows a large transient when DC toggles, but it does not show a stable, measurable baseline separation between DC on and DC off that would correspond to AQFP dissipation. Given the quantization limit, that is exactly what we should expect.

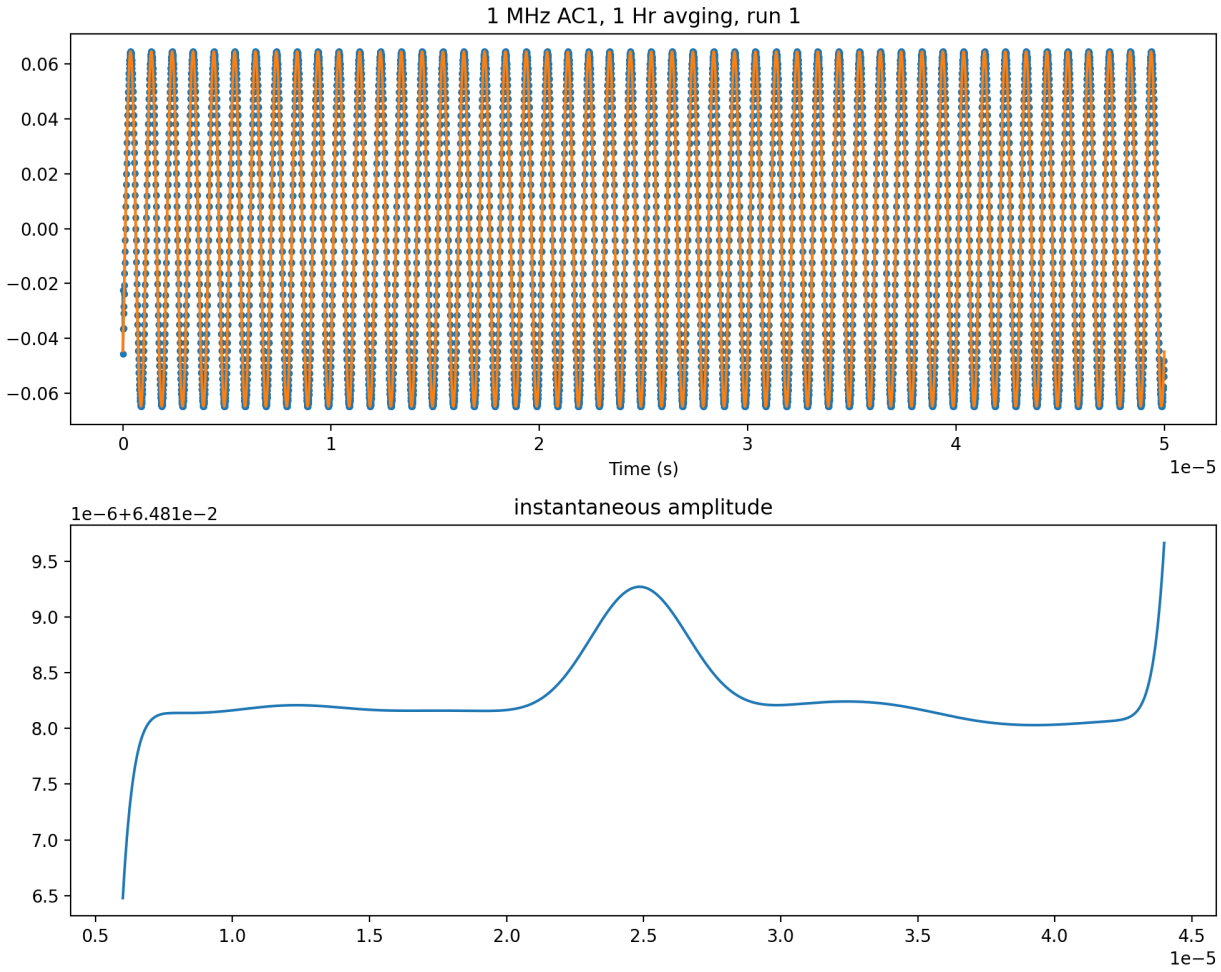


Figure A.7: Coherent demodulation output for AC1, 1 hr averaging, run 1.

A.5 Conclusion

With the current configuration, **we cannot measure AQFP energy dissipation via AC return amplitude difference**. The expected effect is ∼0.5 nV, while the stored waveform resolution is ∼2–3 μV at the vertical scale required to capture the carrier—already ∼3–4 orders of magnitude too coarse.

We could attenuate the clock return and measure it on the finest voltage resolution: 1 mV/div. With this, we would expect to see the bit binning around 8 mV / 2^{16} = 122 nV, which still isn't small enough to resolve the amplitude difference at this low of AQFP frequency. Additionally, when looking at the data output trace (which was collected at 1mV/div), the steps in noise are ∼1.5 μV even after 1 hr of averaging, which is consistent with the 12-bit ADC resolution (8 mV / 2^{12} = 1.9 μV), so we would need to add a noise source to better dither the output during the averaging.

All in all, I don't think digitally processing raw time domain traces is the best way to collect this tiny measurement.

A.6 Next steps

To make this measurement possible, we need an acquisition approach that either (a) preserves amplitude perturbations far below the scope's stored-waveform quantization, or (b) suppresses the large carrier before digitization so the ADC range is used on the residual. In practice, that means shifting away from time-domain amplitude differencing on the scope and toward a narrowband power measurement approach (spectrum analyzer / lock-in style) or an analog cancellation strategy.

The prior on-chip energy measurements in AQFP [8] and RQL [134] suggest two practical takeaways for our setup. First, from AQFP, at MHz clock rates the total absorbed power is extremely small, so we also need to solve the issue of clock feedthrough on our output (see section below) in order to verify data operation at higher frequencies. Second, spectrum analyzer based techniques that convert switching activity into a narrowband power observable (either a direct return-power difference at high frequency as in [8], or sideband-based modulation as in [134]) are required for this measurement.

The most direct next experiment is to replicate the Takeuchi AQFP adder measurement [8] using a spectrum analyzer to measure the returned clock tone power at f_c while toggling the DC bias. In parallel, we can adapt the RQL sideband approach [134] by "chopping" switching activity (e.g., periodically toggling DC bias at a known modulation frequency) and measuring the sidebands around the clock carrier. Both approaches avoid relying on microvolt-quantized time-domain samples and instead use instrumentation designed for small relative power changes on a strong carrier.

Task: Re-run the energy experiment using a spectrum analyzer as the primary readout. Try both (1) direct ON/OFF carrier power differencing and (2) DC-bias chopping with sideband readout.

A.6.1 Hardware improvements

Since testing the SFQ5B5 chips, newer runs have come back with long shift registers in the v2 AQFP library that include impedance-matched clock lines and pad tapers—overall a cleaner RF platform with fewer reflections and a more predictable clock environment. This should reduce spurious coupling paths and simplify interpretation of any measured clock-power changes.

Task: Use a V2 long shift register as the primary test vehicle for the next measurement round.

A.6.2 Characterize and mitigate clock feedthrough on data

We observe visible feedthrough of the AC clock signals onto the data output that is strongly frequency-dependent and worsens at higher frequency. Practically, this is why the AQFP output becomes difficult or impossible to resolve above ∼5 MHz in the current setup (and can be worse under unfavorable grounding conditions). This issue is also noted in Herr et al. for RQL, where clock feedthrough to the outputs is attributed to pickup in the cryoprobe/package and is large compared to the true output signal levels [134].

A useful next step is to quantify this feedthrough as a coupling curve versus frequency. One direct method is to take the FFT/power spectrum of the output channel and measure the clock-tone component that appears at the output as the clock frequency is swept. This provides a direct "clock→output" coupling metric in dB. The measurement should probably be performed with DC clock OFF since the data output should contain components near the same frequency of the clock; however, given the output observed so far, this contribution would be small.

In our previous measurements, we supplied -7.72 dBm clock signals (0.13 V peak amplitude) and measured -14.4 dBm on the return. The data output had a swing of about 2.5 μV, so even a reasonably well isolated coupling between clock and data would disrupt the -102 dBm data output signal. However, it should be noted that in octopux measurements we observe a data peak to peak swing of up to 50 μA on a well bias SQUID (-82 dBm), so perhaps the small data pulse on the high-speed test setup could also be improved with better SQUID biasing.

Separately, we can test fixture-level mitigation strategies (grounding changes, return-current paths, PCB-to-die grounding at 4 K, shielding changes) to see whether the coupling is dominated by package/PCB rather than the chip itself. I think that a first step here should be to short die ground to PCB ground at 4 K, rather than returning it to room temperature for 50 Ω termination.

Task: Perform "clock feedthrough vs frequency" measurement to the next test cycle and use it to guide grounding/fixture changes before repeating energy measurements.

A.6.3 Future tapeout designs

If we are willing to sacrifice I/O pads and chip area to improve measurement observability, differential readout could resolve some of the clock feedthrough on high-speed measurements.

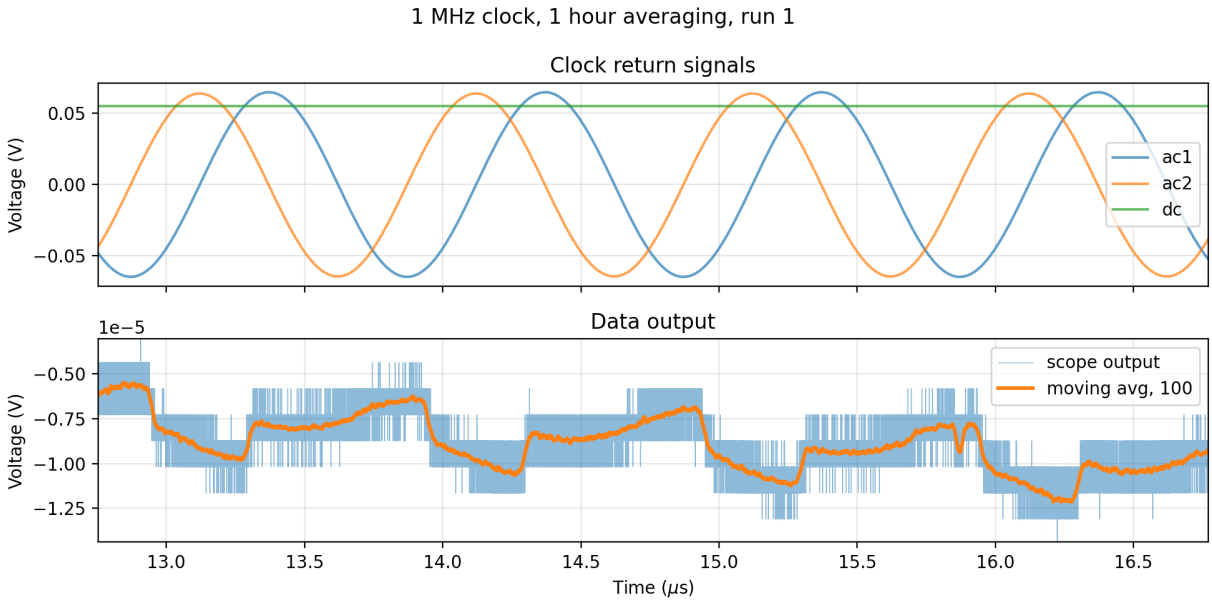


Figure A.8: AQFP output signal scale comparison.

A minimal differential test structure could be implemented by splitting the final output signal into two paths, inverting one path, and reading both with separate SQUID amplifiers. Subtracting the two channels would reject common-mode clock feedthrough without requiring repeated runs for post processing subtraction.

In addition, if we decide to include a dedicated energy test structure, a resonator-coupled dissipation measurement (as demonstrated in earlier AQFP work [133]) is an option that may improve sensitivity, at the cost of operating the circuit at a fixed frequency and additional RF design complexity. (maybe this method could be implemented with existing designs by adding a capacitor between clock signals at room temperature?)

Task: Include at least one **differential-readout test structure** (preferably long shift registers) **on the upcoming tapeout**. Then, consider adding a resonator-coupled energy structure only if spectrum-analyzer readout still cannot reach the needed sensitivity at our operating regime.

References

[1] B. Gerstner and B. Gurley, *All things AI w @altcap @sama & @satyanadella. A Halloween Special.* Oct. 31, 2025. URL: https://youtu.be/Gnl833wXRz0.

[2] A. Shehabi, S. J. Smith, A. Hubbard, A. Newkirk, N. Lei, M. A. Siddik, B. Holecek, J. G. Koomey, E. R. Masanet, and D. A. Sartor, "2024 United States Data Center Energy Usage Report," Lawrence Berkely National Laboratory, 2025. DOI: 10.71468/P1WC7Q. Accessed: May 11, 2026. URL: https://escholarship.org/uc/item/32d6m0d1.

[3] "Google, Kairos Power, TVA announce collaboration," World Nuclear News, Accessed: Oct. 8, 2025. URL: https://world-nuclear-news.org/articles/google-kairos-power-tva-announce-collaboration.

[4] G. Brumfiel, "Big tech's next move is to put data centers in space. Can it work?" *NPR*, (), Apr. 3, 2026. Accessed: May 11, 2026. URL: https://www.npr.org/2026/04/03/nx-s1-5718416/ai-data-centers-in-space-spacex-elon-musk.

[5] A. McCalip. "Economics of Orbital vs Terrestrial Data Centers," Andrew McCalip, Accessed: May 11, 2026. URL: https://andrewmccalip.com/space-datacenters.

[6] R. Landauer, "Irreversibility and Heat Generation in the Computing Process," *IBM Journal of Research and Development*, **vol. 5**(no. 3), pp. 183–191, Jul. 1961, ISSN: 0018-8646. DOI: 10.1147/rd.53.0183. Accessed: May 11, 2025. URL: https://ieeexplore.ieee.org/document/5392446/.

[7] IEEE, "International Roadmap for Devices and Systems 2024 Edition: More Moore," Piscataway, NJ, Industry Roadmap, 2024. URL: https://irds.ieee.org/editions/irds2024/.

[8] N. Takeuchi, T. Yamae, C. L. Ayala, H. Suzuki, and N. Yoshikawa, "An adiabatic superconductor 8-bit adder with 24kBT energy dissipation per junction," *Applied Physics Letters*, **vol. 114**(no. 4), p. 042 602, Jan. 29, 2019, ISSN: 0003-6951. DOI: 10.1063/1.5080753. Accessed: Sep. 22, 2024. URL: https://doi.org/10.1063/1.5080753.

[9] IEEE, "International Roadmap for Devices and Systems 2023 Edition: Cryogenic Electronics and Quantum Information Processing," Institute of Electrical and Electronics Engineers, Industry Roadmap, 2023, pp. 1–93. DOI: 10.60627/042B-J892. Accessed: May 3, 2026. URL: https://doi.org/10.60627/042b-j892.

[10] F. Bedard, N. K. Welker, G. R. Cotter, M. A. Escavage, and J. T. Pinkston, "Superconducting Technology Assessment," National Security Agency Office of Corporate Assesments, Aug. 2005, p. 257. URL: https://www.nitrd.gov/pubs/nsa/sta.pdf.

[11] D. S. Holmes, A. L. Ripple, and M. A. Manheimer, "Energy-Efficient Superconducting Computing—Power Budgets and Requirements," *IEEE Transactions on Applied Superconductivity*, **vol. 23**(no. 3), pp. 1 701 610–1 701 610, Jun. 2013, ISSN: 1558-2515. DOI: 10.1109/TASC.2013.2244634.

[12] C. L. Ayala, T. Tanaka, R. Saito, M. Nozoe, N. Takeuchi, and N. Yoshikawa, "MANA: A Monolithic Adiabatic iNtegration Architecture Microprocessor Using 1.4-zJ/op Unshunted Superconductor Josephson Junction Devices," *IEEE Journal of Solid-State Circuits*, **vol. 56**(no. 4), pp. 1152–1165, Apr. 2021, ISSN: 0018-9200, 1558-173X. DOI: 10.1109/JSSC.2020.3041338. Accessed: May 7, 2021. URL: https://ieeexplore.ieee.org/document/9295318/.

[13] N. Mizushima, N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Adiabatic quantum-flux-parametron boosters for long interconnection and large fanouts," *Superconductor Science and Technology*, **vol. 36**(no. 11), p. 115 021, Nov. 1, 2023, ISSN: 0953-2048, 1361-6668. DOI: 10.1088/1361-6668/acef67. Accessed: Dec. 6, 2023. URL: https://iopscience.iop.org/article/10.1088/1361-6668/acef67.

[14] D. A. Buck, "The Cryotron-A Superconductive Computer Component," *Proceedings of the IRE*, **vol. 44**(no. 4), pp. 482–493, Apr. 1956, ISSN: 2162-6634. DOI: 10.1109/JRPROC.1956.274927.

[15] D. C. Brock, "Dudley Buck's Forgotten Cryotron Computer," *IEEE Spectrum*, (), Mar. 19, 2014. Accessed: Aug. 24, 2021. URL: https://spectrum.ieee.org/dudley-bucks-forgotten-cryotron-computer.

[16] W. Anacker, "Josephson Computer Technology: An IBM Research Project," *IBM Journal of Research and Development*, **vol. 24**(no. 2), pp. 107–112, Mar. 1980, ISSN: 0018-8646. DOI: 10.1147/rd.242.0107.

[17] C. C. M. Mody, "Between research and development: IBM and Josephson computing," *Physics Today*, **vol. 69**(no. 10), pp. 32–38, Oct. 1, 2016, ISSN: 0031-9228. DOI: 10.1063/PT.3.3327. Accessed: May 7, 2024. URL: https://doi.org/10.1063/PT.3.3327.

[18] M. Gurvitch, M. A. Washington, and H. A. Huggins, "High quality refractory Josephson tunnel junctions utilizing thin aluminum layers," *Applied Physics Letters*, **vol. 42**(no. 5), pp. 472–474, Mar. 1, 1983, ISSN: 0003-6951. DOI: 10.1063/1.93974. Accessed: May 11, 2026. URL: https://pubs.aip.org/aip/apl/article/42/5/472/48168/High-quality-refractory-Josephson-tunnel-junctions.

[19] A. I. Braginski, "Superconductor Electronics: Status and Outlook," *Journal of Superconductivity and Novel Magnetism*, **vol. 32**(no. 1), pp. 23–44, Jan. 1, 2019, ISSN: 1557-1947. DOI: 10.1007/s10948-018-4884-4. Accessed: Dec. 3, 2021. URL: https://doi.org/10.1007/s10948-018-4884-4.

[20] K. K. Likharev, O. A. Mukhanov, and V. K. Semenov, "Resistive single flux quantum logic for the Josephson-junction digital technology," in *SQUID'85: Superconducting Quantum Interference Devices and Their Applications*, 1985, pp. 1103–1108.

[21] W. Chen, A. Rylyakov, V. Patel, J. Lukens, and K. Likharev, "Rapid single flux quantum T-flip flop operating up to 770 GHz," *IEEE Transactions on Applied Superconductivity*, **vol. 9**(no. 2), pp. 3212–3215, Jun. 1999, ISSN: 1558-2515. DOI: 10.1109/77.783712. Accessed: May 11, 2026. URL: https://ieeexplore.ieee.org/abstract/document/783712.

[22] O. A. Mukhanov, "Energy-Efficient Single Flux Quantum Technology," *IEEE Transactions on Applied Superconductivity*, **vol. 21**(no. 3), pp. 760–769, Jun. 2011, ISSN: 1558-2515. DOI: 10.1109/TASC.2010.2096792.

[23] J. Volk, A. Wynn, E. Golden, T. Sherwood, and G. Tzimpragos, "Addressable superconductor integrated circuit memory from delay lines," *Scientific Reports*, **vol. 13**(no. 1), p. 16 639, Oct. 3, 2023, ISSN: 2045-2322. DOI: 10.1038/s41598-023-43205-8. Accessed: Mar. 14, 2025. URL: https://www.nature.com/articles/s41598-023-43205-8.

[24] R. T. Kapur et al., "Flux-trapping characterization for superconducting electronics using a cryogenic widefield N- V diamond microscope," *Physical Review Applied*, **vol. 25**(no. 4), p. 044 073, Apr. 24, 2026, ISSN: 2331-7019. DOI: 10.1103/3zcp-lsrd. Accessed: May 18, 2026. URL: https://link.aps.org/doi/10.1103/3zcp-lsrd.

[25] N. Takeuchi, D. Ozawa, Y. Yamanashi, and N. Yoshikawa, "An adiabatic quantum flux parametron as an ultra-low-power logic device," *Superconductor Science and Technology*, **vol. 26**(no. 3), p. 035 010, Mar. 1, 2013, ISSN: 0953-2048, 1361-6668. DOI: 10.1088/0953-2048/26/3/035010. Accessed: Aug. 16, 2022. URL: https://iopscience.iop.org/article/10.1088/0953-2048/26/3/035010.

[26] N. Takeuchi, T. Yamae, C. L. Ayala, H. Suzuki, and N. Yoshikawa, "Adiabatic Quantum-Flux-Parametron: A Tutorial Review," *IEICE Transactions on Electronics*, **vol. E105.C**(no. 6), pp. 251–263, Jun. 1, 2022, ISSN: 0916-8524, 1745-1353. DOI: 10.1587/transele.2021SEP0003. Accessed: Jun. 28, 2022. URL: https://www.jstage.jst.go.jp/article/transele/E105.C/6/E105.C_2021SEP0003/_article.

[27] S. K. Tolpygo, V. Bolkhovsky, T. J. Weir, A. Wynn, D. E. Oates, L. M. Johnson, and M. A. Gouker, "Advanced Fabrication Processes for Superconducting Very Large-Scale Integrated Circuits," *IEEE Transactions on Applied Superconductivity*, **vol. 26**(no. 3), pp. 1–10, Apr. 2016, ISSN: 1558-2515. DOI: 10.1109/TASC.2016.2519388.

[28] S. K. Tolpygo, J. L. Mallek, V. Bolkhovsky, R. Rastogi, E. B. Golden, T. J. Weir, L. M. Johnson, and M. A. Gouker, "Progress Toward Superconductor Electronics Fabrication Process With Planarized NbN and NbN/Nb Layers," *IEEE Transactions on Applied Superconductivity*, **vol. 33**(no. 5), pp. 1–12, Aug. 2023, ISSN: 1558-2515. DOI: 10.1109/TASC.2023.3246430. Accessed: May 11, 2026. URL: https://ieeexplore.ieee.org/document/10049082.

[29] J. C. Maxwell, "Maxwell, j.c. (1867) letter to tait, p.g.," in *The Scientific Letters and Papers of James Clerk Maxwell Volume 2*, P. M. Harman, Ed., vol. 2. 1862-1873, 2 vols., Cambridge, UK: Cambridge University Press, Jun. 1990, pp. 331–332, ISBN: 0 521 25625 9. Accessed: May 11, 2026. URL: https://www.cambridge.org/universitypress/subjects/history/history-science-and-technology/scientific-letters-and-papers-james-clerk-maxwell-volume-2.

[30] L. Szilard, "Über die Entropieverminderung in einem thermodynamischen System bei Eingriffen intelligenter Wesen," *Zeitschrift für Physik*, **vol. 53**(no. 11), pp. 840–856, 1929.

[31] L. Szilard, "On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings," *Behavioral Science*, **vol. 9**(no. 4), pp. 301–310, 1964, ISSN: 1099-1743. DOI: 10.1002/bs.3830090402. Accessed: May 11, 2026. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/bs.3830090402.

[32] C. H. Bennett, "Logical Reversibility of Computation," *IBM Journal of Research and Development*, **vol. 17**(no. 6), pp. 525–532, Nov. 1973, ISSN: 0018-8646, 0018-8646. DOI: 10.1147/rd.176.0525. Accessed: Apr. 12, 2024. URL: http://ieeexplore.ieee.org/document/5391327/.

[33] R. W. Keyes and R. Landauer, "Minimal Energy Dissipation in Logic," *IBM Journal of Research and Development*, **vol. 14**(no. 2), pp. 152–157, Mar. 1970, ISSN: 0018-8646. DOI: 10.1147/rd.142.0152. Accessed: May 11, 2025. URL: https://ieeexplore.ieee.org/document/5391667/.

[34] A. Bérut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider, and E. Lutz, "Experimental verification of Landauer's principle linking information and thermodynamics," *Nature*, **vol. 483**(no. 7388), pp. 187–189, 7388, Mar. 2012, ISSN: 1476-4687. DOI: 10.1038/nature10872. Accessed: Jan. 20, 2023. URL: https://www.nature.com/articles/nature10872.

[35] J. V. Koski, V. F. Maisi, J. P. Pekola, and D. V. Averin, "Experimental realization of a Szilard engine with a single electron," *Proceedings of the National Academy of Sciences*, **vol. 111**(no. 38), pp. 13 786–13 789, Sep. 23, 2014, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1406966111. Accessed: Jan. 23, 2023. URL: https://pnas.org/doi/full/10.1073/pnas.1406966111.

[36] E. Goto, "The Parametron, a Digital Computing Element Which Utilizes Parametric Oscillation," *Proceedings of the IRE*, **vol. 47**(no. 8), pp. 1304–1316, Aug. 1959, ISSN: 2162-6634. DOI: 10.1109/JRPROC.1959.287195. Accessed: Nov. 13, 2023. URL: https://ieeexplore.ieee.org/document/4065825.

[37] S. Muroga and K. Takashima, "The Parametron Digital Computer MUSASINO-1," *IRE Transactions on Electronic Computers*, **vol. EC-8**(no. 3), pp. 308–316, Sep. 1959, ISSN: 0367-9950. DOI: 10.1109/TEC.1959.5222689. Accessed: Nov. 26, 2025. URL: https://ieeexplore.ieee.org/document/5222689/.

[38] J. Von Neumann, "Non-linear capacitance or inductance switching, amplifying, and memory organs," U.S. Patent 2815488A, Dec. 3, 1957. Accessed: May 11, 2026. URL: https://patents.google.com/patent/US2815488A/en.

[39] R. L. Wigington, "A New Concept in Computing," *Proceedings of the IRE*, **vol. 47**(no. 4), pp. 516–523, Apr. 1959, ISSN: 2162-6634. DOI: 10.1109/JRPROC.1959.287311. Accessed: May 11, 2025. URL: https://ieeexplore.ieee.org/document/4065705/.

[40] E. Goto, K. Murata, K. Nakazawa, K. Nakagawa, T. Moto-Oka, Y. Matsuoka, Y. Ishibashi, H. Ishida, T. Soma, and E. Wada, "Esaki Diode High-Speed Logical Circuits," *IRE Transactions on Electronic Computers*, **vol. EC-9**(no. 1), pp. 25–29, Mar. 1960, ISSN: 0367-9950. DOI: 10.1109/TEC.1960.5221600. Accessed: May 11, 2026. URL: https://ieeexplore.ieee.org/abstract/document/5221600.

[41] E. Goto and K. F. Loe, *Dc Flux Parametron: A New Approach To Josephson Junction Logic*. World Scientific, Aug. 1, 1986, 212 pp., ISBN: 978-981-4513-64-7. Google Books: VngGCwAAQBAJ.

[42] M. Hosoya, W. Hioe, J. Casas, R. Kamikawai, Y. Harada, Y. Wada, H. Nakane, R. Suda, and E. Goto, "Quantum flux parametron: A single quantum flux device for Josephson supercomputer," *IEEE Transactions on Applied Superconductivity*, **vol. 1**(no. 2), pp. 77–89, Jun. 1991, ISSN: 1558-2515. DOI: 10.1109/77.84613.

[43] K. Likharev, "Dynamics of some single flux quantum devices: I. Parametric quantron," *IEEE Transactions on Magnetics*, **vol. 13**(no. 1), pp. 242–244, Jan. 1977, ISSN: 1941-0069. DOI: 10.1109/TMAG.1977.1059351.

[44] D. Feld, P. Sage, K. Berggren, and A. Siddiqui, "Measurement of the energy sensitivity of a superconductive comparator," *IEEE Transactions on Applied Superconductivity*, **vol. 9**(no. 2), pp. 4361–4366, Jun. 1999, ISSN: 1558-2515. DOI: 10.1109/77.783991. Accessed: May 11, 2026. URL: https://ieeexplore.ieee.org/abstract/document/783991.

[45] L. C. Blackburn, E. Golden, A. Wynn, A. Wagner, and N. Gershenfeld, "Design and Simulation of Phase Synchronizer for Adiabatic Quantum Flux Parametron Circuits," *IEEE Transactions on Applied Superconductivity*, **vol. 33**(no. 5), pp. 1–5, Aug. 2023, ISSN: 1558-2515. DOI: 10.1109/TASC.2023.3243186. Accessed: Oct. 27, 2025. URL: https://ieeexplore.ieee.org/document/10040715.

[46] M. Tinkham, *Introduction to Superconductivity*, 2nd ed. Garden City, New York: Dover Publications, Jun. 14, 2004, ISBN: 0-486-43503-2. URL: http://www.worldcat.org/isbn/0486435032.

[47] J. Bardeen, L. N. Cooper, and J. R. Schrieffer, "Theory of Superconductivity," *Physical Review*, **vol. 108**(no. 5), pp. 1175–1204, Dec. 1, 1957. DOI: 10.1103/PhysRev.108.1175. Accessed: Jul. 5, 2021. URL: https://link.aps.org/doi/10.1103/PhysRev.108.1175.

[48] O. Bose and K. Stevens, *Introductory Network Theory* (Harper International Student Reprints). Harper, 1965. URL: https://books.google.com/books?id=RY85nQEACAAJ.

[49] A. M. Kadin, *Introduction to Superconducting Circuits* (Superconducting Circuits). New York: Wiley, 1999, 382 pp., ISBN: 978-0-471-31432-5. Accessed: May 14, 2026. URL: https://catalog.hathitrust.org/Record/004027781.

[50] B. Josephson, "Supercurrents through barriers," *Advances in Physics*, **vol. 14**(no. 56), pp. 419–451, Oct. 1, 1965, ISSN: 0001-8732. DOI: 10.1080/00018736500101091. Accessed: May 14, 2026. URL: https://doi.org/10.1080/00018736500101091.

[51] K. Likharev and V. Semenov, "RSFQ logic/memory family: A new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Transactions on Applied Superconductivity*, **vol. 1**(no. 1), pp. 3–28, Mar. 1991, ISSN: 1558-2515. DOI: 10.1109/77.80745.

[52] C. Z. Pratt, "Extracting equations of motion from superconducting circuits," *Physical Review Research*, **vol. 7**(no. 1), 2025. DOI: 10.1103/PhysRevResearch.7.013014.

[53] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Adiabatic quantum-flux-parametron cell library adopting minimalist design," *Journal of Applied Physics*, **vol. 117**(no. 17), p. 173 912, May 7, 2015, ISSN: 0021-8979, 1089-7550. DOI: 10.1063/1.4919838. Accessed: Sep. 8, 2021. URL: http://aip.scitation.org/doi/10.1063/1.4919838.

[54] Y. He, C. L. Ayala, N. Takeuchi, T. Yamae, Y. Hironaka, A. Sahu, V. Gupta, A. Talalaevskii, D. Gupta, and N. Yoshikawa, "A compact AQFP logic cell design using an 8-metal layer superconductor process," *Superconductor Science and Technology*, **vol. 33**(no. 3), p. 035 010, Mar. 1, 2020, ISSN: 0953-2048, 1361-6668. DOI: 10.1088/1361-6668/ab6feb. Accessed: Dec. 1, 2021. URL: https://iopscience.iop.org/article/10.1088/1361-6668/ab6feb.

[55] S. K. Tolpygo, "Scalability of Superconductor Electronics: Limitations Imposed by AC Clock and Flux Bias Transformers," *IEEE Transactions on Applied Superconductivity*, **vol. 33**(no. 2), pp. 1–19, Mar. 2023, ISSN: 1558-2515. DOI: 10.1109/TASC.2022.3230373. Accessed: Jan. 13, 2025. URL: https://ieeexplore.ieee.org/document/9992087/?arnumber=9992087.

[56] T. Yamae, N. Takeuchi, and N. Yoshikawa, "Minimum energy dissipation required for information processing using adiabatic quantum-flux-parametron circuits," *Journal of Applied Physics*, **vol. 135**(no. 6), p. 063 902, Feb. 14, 2024, ISSN: 0021-8979, 1089-7550. DOI: 10.1063/5.0187756. Accessed: Apr. 12, 2024. URL: https://pubs.aip.org/jap/article/135/6/063902/3262821/Minimum-energy-dissipation-required-for.

[57] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Reversibility and energy dissipation in adiabatic superconductor logic," *Scientific Reports*, **vol. 7**(no. 1), p. 75, 1, Mar. 6, 2017, ISSN: 2045-2322. DOI: 10.1038/s41598-017-00089-9. Accessed: Dec. 19, 2022. URL: https://www.nature.com/articles/s41598-017-00089-9.

[58] C. L. Ayala, N. Takeuchi, Y. Yamanashi, T. Ortlepp, and N. Yoshikawa, "Majority-Logic-Optimized Parallel Prefix Carry Look-Ahead Adder Families Using Adiabatic Quantum-Flux-Parametron Logic," *IEEE Transactions on Applied Superconductivity*,

vol. 27(no. 4), pp. 1–7, Jun. 2017, ISSN: 1558-2515. DOI: 10.1109/TASC.2016.2642041. Accessed: May 14, 2026. URL: https://ieeexplore.ieee.org/document/7792227/.

[59] Y. Hironaka, T. Yamae, C. L. Ayala, N. Yoshikawa, and N. Takeuchi, "Low-Latency Adiabatic Quantum-Flux-Parametron Circuit Integrated With a Hybrid Serializer/Deserializer," *IEEE Access*, **vol. 10**(), pp. 133 584–133 590, 2022, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3230447. Accessed: Mar. 19, 2026. URL: https://ieeexplore.ieee.org/document/9991989/.

[60] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Reversible logic gate using adiabatic superconducting devices," *Scientific Reports*, **vol. 4**(no. 1), p. 6354, Sep. 15, 2014, ISSN: 2045-2322. DOI: 10.1038/srep06354. Accessed: May 14, 2026. URL: https://www.nature.com/articles/srep06354.

[61] T. Yamae, N. Takeuchi, and N. Yoshikawa, "A reversible full adder using adiabatic superconductor logic," *Superconductor Science and Technology*, **vol. 32**(no. 3), p. 035 005, Jan. 2019, ISSN: 0953-2048. DOI: 10.1088/1361-6668/aaf8c9. Accessed: May 14, 2026. URL: https://doi.org/10.1088/1361-6668/aaf8c9.

[62] C. J. Fourie et al., "Results From the ColdFlux Superconductor Integrated Circuit Design Tool Project," *IEEE Transactions on Applied Superconductivity*, **vol. 33**(no. 8), pp. 1–26, Nov. 2023, ISSN: 1558-2515. DOI: 10.1109/TASC.2023.3306381. Accessed: May 14, 2026. URL: https://ieeexplore.ieee.org/abstract/document/10224066.

[63] C. J. Fourie, "Electronic Design Automation tools for superconducting circuits," *Journal of Physics: Conference Series*, **vol. 1590**(no. 1), p. 012 040, Jul. 2020, ISSN: 1742-6596. DOI: 10.1088/1742-6596/1590/1/012040. Accessed: May 14, 2026. URL: https://doi.org/10.1088/1742-6596/1590/1/012040.

[64] R. Freeman, J. Kawa, and K. Singhal, "Synopsys' Journey to Enable TCAD and EDA Tools for Superconducting Electronics," ().

[65] L. Schindler, R. van Staden, C. J. Fourie, C. L. Ayala, J. A. Coetzee, T. Tanaka, R. Saito, and N. Yoshikawa, "Standard Cell Layout Synthesis for Row-Based Placement and Routing of RSFQ and AQFP Logic Families," in *2019 IEEE International Superconductive Electronics Conference (ISEC)*, Jul. 2019, pp. 1–5. DOI: 10.1109/ISEC46533.2019.8990962. Accessed: May 14, 2026. URL: https://ieeexplore.ieee.org/abstract/document/8990962.

[66] H. Ko and G. Lee, "Noise analysis of the quantum flux parametron," *IEEE Transactions on Applied Superconductivity*, **vol. 2**(no. 3), pp. 156–164, Sep. 1992, ISSN: 1558-2515. DOI: 10.1109/77.160155.

[67] T. Ando, N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Adiabatic quantum-flux-parametron constant cells using asymmetrical structures," *IEEJ Trans. Fundam. Mater*, **vol. 136**(no. 12), pp. 747–752, 2016.

[68] K. Irwin and M. Huber, "SQUID operational amplifier," *IEEE Transactions on Applied Superconductivity*, **vol. 11**(no. 1), pp. 1265–1270, Mar. 2001, ISSN: 1558-2515. DOI: 10.1109/77.919580.

[69] A. N. McCaughan and K. K. Berggren, "A Superconducting-Nanowire Three-Terminal Electrothermal Device," *Nano Letters*, **vol. 14**(no. 10), pp. 5748–5753, Oct. 8, 2014, ISSN: 1530-6984. DOI: 10.1021/nl502629x. Accessed: May 16, 2022. URL: https://doi.org/10.1021/nl502629x.

[70] D. Zinoviev and Y. Polyakov, "Octopux: An advanced automated setup for testing superconductor circuits," *IEEE Transactions on Applied Superconductivity*, **vol. 7**(no. 2), pp. 3240–3243, Jun. 1997, ISSN: 1558-2515. DOI: 10.1109/77.622039.

[71] L. C. Blackburn, A. Wynn, K. K. Berggren, and N. Gershenfeld, "A Compact Bit Serial Memory Cell for Adiabatic Quantum Flux Parametron Register Files," *IEEE Transactions on Applied Superconductivity*, **vol. 35**(no. 5), pp. 1–5, Aug. 2025, ISSN: 1558-2515. DOI: 10.1109/TASC.2025.3540048. Accessed: Mar. 12, 2025. URL: https://ieeexplore.ieee.org/document/10882861/?arnumber=10882861.

[72] S. Alam, M. S. Hossain, S. R. Srinivasa, and A. Aziz, "Cryogenic memory technologies," *Nature Electronics*, **vol. 6**(no. 3), pp. 185–198, Mar. 2023, ISSN: 2520-1131. DOI: 10.1038/s41928-023-00930-2. Accessed: May 10, 2026. URL: https://www.nature.com/articles/s41928-023-00930-2.

[73] V. K. Semenov, Y. A. Polyakov, and S. K. Tolpygo, "Very Large Scale Integration of Josephson-Junction-Based Superconductor Random Access Memories," *IEEE Transactions on Applied Superconductivity*, **vol. 29**(no. 5), pp. 1–9, Aug. 2019, ISSN: 1558-2515. DOI: 10.1109/TASC.2019.2904971. Accessed: Oct. 29, 2025. URL: https://ieeexplore.ieee.org/document/8667388/.

[74] Q. Herr, T. Josephsen, and A. Herr, "Superconducting pulse conserving logic and Josephson-SRAM," *Applied Physics Letters*, **vol. 122**(no. 18), p. 182 604, May 4, 2023, ISSN: 0003-6951. DOI: 10.1063/5.0148235. Accessed: May 10, 2026. URL: https://doi.org/10.1063/5.0148235.

[75] O. Medeiros, M. Castellani, V. Karam, R. Foster, A. Simon, F. Incalza, B. Butters, M. Colangelo, and K. K. Berggren, "A scalable superconducting nanowire memory array with row–column addressing," *Nature Electronics*, **vol. 9**(no. 1), pp. 69–77, Jan. 2026, ISSN: 2520-1131. DOI: 10.1038/s41928-025-01512-0. Accessed: Apr. 9, 2026. URL: https://www.nature.com/articles/s41928-025-01512-0.

[76] S. Nagasawa, Y. Hashimoto, H. Numata, and S. Tahara, "A 380 ps, 9.5 mW Josephson 4-Kbit RAM operated at a high bit yield," *IEEE Transactions on Applied Superconductivity*, **vol. 5**(no. 2), pp. 2447–2452, Jun. 1995, ISSN: 1558-2515. DOI: 10.1109/77.403086. Accessed: Nov. 25, 2025. URL: https://ieeexplore.ieee.org/document/403086/.

[77] N. Tsuji, C. L. Ayala, N. Takeuchi, T. Ortlepp, Y. Yamanashi, and N. Yoshikawa, "Design and Implementation of a 16-Word by 1-Bit Register File Using Adiabatic Quantum Flux Parametron Logic," *IEEE Transactions on Applied Superconductivity*, **vol. 27**(no. 4), pp. 1–4, Jun. 2017, ISSN: 1558-2515. DOI: 10.1109/TASC.2017.2656128. Accessed: Nov. 25, 2025. URL: https://ieeexplore.ieee.org/document/7828038.

[78] M. Williams, "The origins, uses, and fate of the EDVAC," *IEEE Annals of the History of Computing*, **vol. 15**(no. 1), pp. 22–38, 1993, ISSN: 1934-1547. DOI: 10.1109/85.194089. Accessed: May 10, 2026. URL: https://ieeexplore.ieee.org/document/194089/.

[79] "Cambridge Laboratory EDSAC Report," Cambridge University Mathematical Laboratory, Cambridge, UK, May 1948. Accessed: May 8, 2026. URL: https://www.billp.org/ccs/Edsac/Report/index.html.

[80] N. Tsuji, N. Takeuchi, Y. Yamanashi, T. Ortlepp, and N. Yoshikawa, "Majority Gate-Based Feedback Latches for Adiabatic Quantum Flux Parametron Logic," *IEICE Transactions on Electronics*, **vol. E99.C**(no. 6), pp. 710–716, 2016. DOI: 10.1587/transele.E99.C.710.

[81] A. F. Kirichenko, A. Sahu, T. V. Filippov, O. A. Mukhanov, A. V. Dotsenko, M. Dorojevets, and A. K. Kasperek, "Demonstration of an 8×8-bit RSFQ multi-port register file," in *2013 IEEE 14th International Superconductive Electronics Conference (ISEC)*, Jul. 2013, pp. 1–3. DOI: 10.1109/ISEC.2013.6604257. Accessed: May 9, 2026. URL: https://ieeexplore.ieee.org/document/6604257/.

[82] N. Takeuchi, K. Arai, and N. Yoshikawa, "Directly coupled adiabatic superconductor logic," *Superconductor Science and Technology*, **vol. 33**(no. 6), p. 065 002, May 2020, ISSN: 0953-2048. DOI: 10.1088/1361-6668/ab87ad. Accessed: May 10, 2026. URL: https://doi.org/10.1088/1361-6668/ab87ad.

[83] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Fifth Edition: A Quantitative Approach*, 5th ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011, ISBN: 0-12-383872-X.

[84] B. J. Smith, "Architecture and applications of the hep multiprocessor computer system," *SPIE Proceedings*, **vol. 298**(), pp. 241–248, Jul. 30, 1982, ISSN: 0277-786X.

[85] M. R. Thistle and B. J. Smith, "A processor architecture for horizon," in *Proceedings of the 1988 ACM/IEEE Conference on Supercomputing*, ser. Supercomputing '88, Washington, DC, USA: IEEE Computer Society Press, Nov. 1, 1988, pp. 35–41, ISBN: 978-0-8186-0882-7. Accessed: May 7, 2026. URL: https://dl.acm.org/doi/10.5555/62972.62977.

[86] R. Alverson, D. Callahan, D. Cummings, B. Koblenz, A. Porterfield, and B. Smith, "The Tera computer system," in *Proceedings of the 4th International Conference on Supercomputing*, ser. ICS '90, New York, NY, USA: Association for Computing Machinery, Jun. 1, 1990, pp. 1–6, ISBN: 978-0-89791-369-0. DOI: 10.1145/77726.255132. Accessed: May 6, 2026. URL: https://dl.acm.org/doi/10.1145/77726.255132.

[87] D. Patterson, "The top 10 innovations in the new NVIDIA Fermi architecture, and the top 3 next challenges," *Nvidia Whitepaper*, **vol. 47**(), 2009. URL: https://www.nvidia.com/content/PDF/fermi_white_papers/D.Patterson_Top10InnovationsInNVIDIAFermi.pdf.

[88] M. AskariHemmat, O. Bilaniuk, S. Wagner, Y. Savaria, and J.-P. David, "RISC-V Barrel Processor for Deep Neural Network Acceleration," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2021, pp. 1–5. DOI: 10.1109/ISCAS51556.2021.9401617. Accessed: May 7, 2026. URL: https://ieeexplore.ieee.org/document/9401617/.

[89] J. E. Thornton, *Design of a Computer—the Control Data 6600*. Scott Foresman & Co, 1970.

[90] D. May, *The XMOS XS1 Architecture*. Bristol: XMOS Limited, 2009, 261 pp., ISBN: 978-1-907361-01-2 978-1-907361-04-3.

[91] M. Dorojevets, C. L. Ayala, and A. K. Kasperek, "Data-Flow Microarchitecture for Wide Datapath RSFQ Processors: Design Study," *IEEE Transactions on Applied Superconductivity*, **vol. 21**(no. 3), pp. 787–791, Jun. 2011, ISSN: 1558-2515. DOI: 10.1109/TASC.2010.2087410. Accessed: May 7, 2026. URL: https://ieeexplore.ieee.org/document/5634068/.

[92] P. Gonzalez-Guerrero, K. Huch, N. Patra, T. Popovici, and G. Michelogiannakis, "Toward Practical Superconducting Accelerators for Machine Learning Using U-SFQ," *J. Emerg. Technol. Comput. Syst.*, **vol. 20**(no. 2), 7:1–7:22, Jun. 7, 2024, ISSN: 1550-4832. DOI: 10.1145/3653073. Accessed: Apr. 26, 2026. URL: https://dl.acm.org/doi/10.1145/3653073.

[93] J. Kundu et al., "A System Level Performance Evaluation for Superconducting Digital Systems," in *2025 Design, Automation & Test in Europe Conference (DATE)*, Mar. 2025, pp. 1–7. DOI: 10.23919/DATE64628.2025.10992879. Accessed: Apr. 26, 2026. URL: https://ieeexplore.ieee.org/document/10992879/.

[94] T. Andrulis and M. Gilbert, *AccelForge*. URL: https://github.com/Accelergy-Project/accelforge.

[95] A. Einstein, "Die Grundlage der allgemeinen Relativitätstheorie," *Annalen der Physik*, **vol. 354**(no. 7), pp. 769–822, 1916, ISSN: 1521-3889. DOI: 10.1002/andp.19163540702. Accessed: Apr. 27, 2026. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.19163540702.

[96] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, *Efficient Processing of Deep Neural Networks* (Synthesis Lectures on Computer Architecture). Cham: Springer International Publishing, 2020, ISBN: 978-3-031-00638-8 978-3-031-01766-7. DOI: 10.1007/978-3-031-01766-7. Accessed: Apr. 27, 2026. URL: https://link.springer.com/10.1007/978-3-031-01766-7.

[97] M. Gilbert, T. Andrulis, V. Sze, and J. S. Emer. "The Turbo-Charged Mapper: Fast and Optimal Mapping for Accelerator Modeling and Evaluation." arXiv: 2602.15172 [cs], Accessed: Mar. 5, 2026. URL: http://arxiv.org/abs/2602.15172, pre-published.

[98] J. Austin et al., "How to scale your model," (), 2025.

[99] T. Brown et al., "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. Accessed: Apr. 27, 2026. URL: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[100] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré. "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness." arXiv: 2205.14135 [cs], Accessed: Apr. 27, 2026. URL: http://arxiv.org/abs/2205.14135, pre-published.

[101] N. P. Jouppi et al. "In-Datacenter Performance Analysis of a Tensor Processing Unit." arXiv: 1704.04760 [cs], Accessed: Mar. 16, 2026. URL: http://arxiv.org/abs/1704.04760, pre-published.

[102] S. Williams, A. Waterman, and D. Patterson, "Roofline: An insightful visual performance model for multicore architectures," *Commun. ACM*, **vol. 52**(no. 4), pp. 65–76, Apr. 1, 2009, ISSN: 0001-0782. DOI: 10.1145/1498765.1498785. Accessed: Apr. 27, 2026. URL: https://dl.acm.org/doi/10.1145/1498765.1498785.

[103] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, San Francisco, CA, USA: IEEE, Feb. 2014, pp. 10–14, ISBN: 978-1-4799-0920-9 978-1-4799-0918-6. DOI: 10.1109/ISSCC.2014.6757323. Accessed: Apr. 27, 2026. URL: http://ieeexplore.ieee.org/document/6757323/.

[104] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, **vol. 52**(no. 1), pp. 127–138, Jan. 2017, ISSN: 1558-173X. DOI: 10.1109/JSSC.2016.2616357. Accessed: Mar. 12, 2025. URL: https://ieeexplore.ieee.org/document/7738524/?arnumber=7738524.

[105] N. P. Jouppi et al. "TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings." arXiv: 2304.01433 [cs], Accessed: Mar. 5, 2026. URL: http://arxiv.org/abs/2304.01433, pre-published.

[106] A. Parashar, P. Raina, Y. S. Shao, Y.-H. Chen, V. A. Ying, A. Mukkara, R. Venkatesan, B. Khailany, S. W. Keckler, and J. Emer, "Timeloop: A Systematic Approach to DNN Accelerator Evaluation," in *2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Mar. 2019, pp. 304–315. DOI: 10.1109/ISPASS.2019.00042. Accessed: Mar. 5, 2026. URL: https://ieeexplore.ieee.org/document/8695666.

[107] K. Hegde, P.-A. Tsai, S. Huang, V. Chandra, A. Parashar, and C. W. Fletcher, "Mind mappings: Enabling efficient algorithm-accelerator mapping space search," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '21, New York, NY, USA: Association for Computing Machinery, Apr. 17, 2021, pp. 943–958, ISBN: 978-1-4503-8317-2. DOI: 10.1145/3445814.3446762. Accessed: Apr. 26, 2026. URL: https://dl.acm.org/doi/10.1145/3445814.3446762.

[108] Q. Huang, M. Kang, G. Dinh, T. Norell, A. Kalaiah, J. Demmel, J. Wawrzynek, and Y. S. Shao, "CoSA: Scheduling by Constrained Optimization for Spatial Accelerators," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, Jun. 2021, pp. 554–566. DOI: 10.1109/ISCA52012.2021.00050. Accessed: Apr. 27, 2026. URL: https://ieeexplore.ieee.org/document/9499855/.

[109] T. Andrulis, M. Gilbert, V. Sze, and J. S. Emer. "Fast and Fusiest: An Optimal Fusion-Aware Mapper for Accelerator Modeling and Evaluation." arXiv: 2602.15166 [cs], Accessed: Mar. 5, 2026. URL: http://arxiv.org/abs/2602.15166, pre-published.

[110] M. Gilbert, Y. N. Wu, A. Parashar, V. Sze, and J. S. Emer, "LoopTree: Enabling Exploration of Fused-layer Dataflow Accelerators," in *2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Apr. 2023, pp. 316–318. DOI: 10.1109/ISPASS57527.2023.00038. Accessed: Mar. 5, 2026. URL: https://ieeexplore.ieee.org/document/10158176.

[111] M. Gilbert, "LoopTree: Enabling Systematic and Flexible Exploration of Fused-layer Dataflow Accelerators," Thesis, Massachusetts Institute of Technology, Sep. 2023. Accessed: Apr. 28, 2026. URL: https://dspace.mit.edu/handle/1721.1/152743.

[112] T. Andrulis, J. S. Emer, and V. Sze, "CiMLoop: A Flexible, Accurate, and Fast Compute-In-Memory Modeling Tool," in *2024 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, May 5, 2024, pp. 10–23. DOI: 10.1109/ISPASS61541.2024.00012. arXiv: 2405.07259 [cs]. Accessed: Sep. 17, 2025. URL: http://arxiv.org/abs/2405.07259.

[113] Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Westminster, CO, USA: IEEE, Nov. 2019, pp. 1–8, ISBN: 978-1-7281-2350-9. DOI: 10.1109/ICCAD45719.2019.8942149. Accessed: Mar. 5, 2026. URL: https://ieeexplore.ieee.org/document/8942149/.

[114] Q. P. Herr, A. Y. Herr, O. T. Oberg, and A. G. Ioannidis, "Ultra-low-power superconductor logic," *Journal of Applied Physics*, **vol. 109**(no. 10), May 15, 2011, ISSN: 0021-8979. DOI: 10.1063/1.3585849. Accessed: Apr. 29, 2026. URL: https://pubs.aip.org/aip/jap/article/109/10/103903/984942/Ultra-low-power-superconductor-logic.

[115] M. Dorojevets, Z. Chen, C. L. Ayala, and A. K. Kasperek, "Towards 32-bit Energy-Efficient Superconductor RQL Processors: The Cell-Level Design and Analysis of Key Processing and On-Chip Storage Units," *IEEE Transactions on Applied Superconductivity*, **vol. 25**(no. 3), pp. 1–8, Jun. 2015, ISSN: 1558-2515. DOI: 10.1109/TASC.2014.2368354. Accessed: Sep. 16, 2025. URL: https://ieeexplore.ieee.org/document/6949079/.

[116] Y. S. Shao, B. Reagen, G.-Y. Wei, and D. Brooks, "Aladdin: A pre-RTL, power-performance accelerator simulator enabling large design space exploration of customized architectures," in *2014 ACM/IEEE 41st International Symposium on Computer Ar-*

chitecture (ISCA), Jun. 2014, pp. 97–108. DOI: 10.1109/ISCA.2014.6853196. Accessed: Apr. 29, 2026. URL: https://ieeexplore.ieee.org/document/6853196/.

[117] O. A. Medeiros, "Superconducting Nanowire Integrated Circuits for Scalable Cryogenic Memory," Thesis, Massachusetts Institute of Technology, May 2025. Accessed: Apr. 29, 2026. URL: https://dspace.mit.edu/handle/1721.1/164062.

[118] R. A. Damsteegt, R. W. J. Overwater, M. Babaie, and F. Sebastiano, "A Benchmark of Cryo-CMOS Embedded SRAM/DRAMs in 40-nm CMOS," *IEEE Journal of Solid-State Circuits*, **vol. 59**(no. 7), pp. 2042–2054, Jul. 2024, ISSN: 1558-173X. DOI: 10.1109/JSSC.2024.3385696. Accessed: Apr. 29, 2026. URL: https://ieeexplore.ieee.org/document/10500491.

[119] P. Shivakumar and N. P. Jouppi, "Cacti 3.0: An integrated cache timing, power, and area model," (), 2001.

[120] S. Wilton and N. Jouppi, "CACTI: An enhanced cache access and cycle time model," *IEEE Journal of Solid-State Circuits*, **vol. 31**(no. 5), pp. 677–688, May 1996, ISSN: 1558-173X. DOI: 10.1109/4.509850. Accessed: Apr. 29, 2026. URL: https://ieeexplore.ieee.org/document/509850.

[121] R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee, and V. Srinivas, "CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories," *ACM Trans. Archit. Code Optim.*, **vol. 14**(no. 2), 14:1–14:25, Jun. 28, 2017, ISSN: 1544-3566. DOI: 10.1145/3085572. Accessed: Apr. 29, 2026. URL: https://dl.acm.org/doi/10.1145/3085572.

[122] Z. Chang et al. "CO-QLink: Cryogenic Optical Link for Scalable Quantum Computing Systems and High-Performance Cryogenic Computing Systems." arXiv: 2511.22920 [quant-ph], Accessed: Mar. 17, 2026. URL: http://arxiv.org/abs/2511.22920, pre-published.

[123] J. Wang, I. Harris, M. Ibrahim, D. Englund, and R. Han, "A wireless terahertz cryogenic interconnect that minimizes heat-to-information transfer," *Nature Electronics*, **vol. 8**(no. 5), pp. 426–436, May 2025, ISSN: 2520-1131. DOI: 10.1038/s41928-025-01355-9. Accessed: Mar. 17, 2026. URL: https://www.nature.com/articles/s41928-025-01355-9.

[124] Delft Circuits, "Cri/oFlex 3," Delft Circuits, Data Sheet, May 19, 2020. URL: https://cryocoax.com/wp-content/uploads/2020/06/Datasheet-CrioFlex-3.pdf.

[125] N. Raicu, T. Hogan, X. Wu, M. Vahidpour, D. Snow, M. Hollister, and M. Field, "Cryogenic Thermal Modeling of Microwave High Density Signaling," *EPJ Quantum Technology*, **vol. 12**(no. 1), p. 124, Dec. 2025, ISSN: 2662-4400, 2196-0763. DOI: 10.1140/epjqt/s40507-025-00427-1. arXiv: 2502.01945 [quant-ph]. Accessed: May 4, 2026. URL: http://arxiv.org/abs/2502.01945.

[126] B. Yin, "Cryogenic Optical Link: Device, Circuit, and System," UC Berkeley, 2024. Accessed: Mar. 19, 2026. URL: https://escholarship.org/uc/item/1h85w8df.

[127] N. Fakkel, M. Mortazavi, R. W. J. Overwater, F. Sebastiano, and M. Babaie, "A Cryo-CMOS DAC-Based 40-Gb/s PAM4 Wireline Transmitter for Quantum Computing," *IEEE Journal of Solid-State Circuits*, **vol. 59**(no. 5), pp. 1433–1446, May 2024, ISSN: 1558-173X. DOI: 10.1109/JSSC.2024.3364968. Accessed: Mar. 17, 2026. URL: https://ieeexplore.ieee.org/document/10441816.

[128] COAX CO., LTD. "SC-086/50-SCN-CN," Accessed: May 4, 2026. URL: http://www.coax.co.jp/en/product/sc/086-50-scn-cn.html.

[129] C. L. Ayala, N. Yoshikawa, Y. Hoshika, and Y. Omori, "Multi-GHz Zeptojoule Computing Using Emerging Adiabatic Superconductor Circuits," in *2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Jul. 2024, pp. 557–564. DOI: 10.1109/ISVLSI61997.2024.00106. Accessed: Oct. 29, 2025. URL: https://ieeexplore.ieee.org/document/10682681/.

[130] "attoCMC: Compact, rack mountable cryostat system," attocube, Accessed: May 6, 2026. URL: https://www.attocube.com/en/products/cryostats/compact-mobile-cryogenics/attoCMC.

[131] R. N. Das et al., "Extremely large area (88 mm × 88 mm) superconducting integrated circuit (ELASIC)," *Scientific Reports*, **vol. 13**(no. 1), p. 11 796, Jul. 21, 2023, ISSN: 2045-2322. DOI: 10.1038/s41598-023-39032-6. Accessed: May 18, 2026. URL: https://www.nature.com/articles/s41598-023-39032-6.

[132] A. Han, "Volume Mount Devices," Massachusetts Institute of Technology, May 2025. HDL: 1721.1/164144. Accessed: May 18, 2026. URL: https://hdl.handle.net/1721.1/164144.

[133] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Measurement of 10 zJ energy dissipation of adiabatic quantum-flux-parametron logic using a superconducting resonator," *Applied Physics Letters*, **vol. 102**(no. 5), p. 052 602, Feb. 4, 2013, ISSN: 0003-6951, 1077-3118. DOI: 10.1063/1.4790276. Accessed: Nov. 2, 2021. URL: http://aip.scitation.org/doi/10.1063/1.4790276.

[134] A. Y. Herr, Q. P. Herr, O. T. Oberg, O. Naaman, J. X. Przybysz, P. Borodulin, and S. B. Shauck, "An 8-bit carry look-ahead adder with 150 ps latency and sub-microwatt power dissipation at 10 GHz," *Journal of Applied Physics*, **vol. 113**(no. 3), p. 033 911, Jan. 21, 2013, ISSN: 0021-8979, 1089-7550. DOI: 10.1063/1.4776713. arXiv: 1212.2994 [quant-ph]. Accessed: Dec. 19, 2025. URL: http://arxiv.org/abs/1212.2994.