Construction of a Superconducting Circuit Simulator and its Applications in Cryogenic Computing

by

O. Murat Onen

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2019

Signature redacted

Author .....
Department of Electrical Engineering and Computer Science
January 31, 2019

Signature redacted

Certified by .....
Karl K. Berggren
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Signature redacted

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Construction of a Superconducting Circuit Simulator and its Applications in Cryogenic Computing

by

O. Murat Onen

Submitted to the Department of Electrical Engineering and Computer Science
on January 31, 2019, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

In this work, I first construct a unified simulation platform, where superconducting electronics can be designed and optimized with high performance and accuracy. For this purpose, I first select numerical simulation methods that can deal with the highly non-linear characteristics of the superconducting devices. I validate the simulated responses with experimental data on device and circuit level examples. Following the implementation of the simulator, I use this framework to analyze existing superconducting nanowire based technologies, and optimize them for wider operation regimes and higher performance metrics. I use nanofabrication processes to realize these devices and conduct liquid helium immersion measurements to characterize them experimentally. Optimized devices show superior characteristics that demonstrate the predictive capabilities of this simulator. Finally, I use this simulator to design a superconducting nanowire based deep neural network training accelerator. I design, implement, and characterize a unit cell for this application. These local processors have significant device-level advantages over the readily available non-volatile memory technologies in realizing mixed-signal architectures. The devices produced throughout this work have immediate and near-term applications, proving the merit of having a high-performance simulator.

Thesis Supervisor: Karl K. Berggren
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

The study reported in this thesis would not have been possible without the support and advice of my colleagues, family and friends. In particular, I would like to thank:

My advisor, Prof. Karl K. Berggren, for his insight, dedication to science and providing a nourishing environment for his students in building independent researching skills. His method of following a wide spectrum of research has initiated and greatly supported this work throughout its course.

Andrew E. Dane, for being my mentor in the field and teaching me the fundamentals of material science, nanofabrication, and even more importantly, scientific rigor. Most of this work would have got stuck on an idea level if he had not chosen to share his knowledge and experience with me.

Brenden A. Butters, Emily Toomey, Marco Colangelo, Di Zhu, and Eugenio Maggiolini for their helpful discussions, collaborations and experimental assistance.

Tayfun Gokmen, Wilfried Haensch and Heike Riel at IBM T.J. Watson Research Center, for giving me the first opportunity to become a part of something bigger.

Prof. Luca Daniel for teaching me advanced numerical simulation techniques that I have used extensively in this work.

Dorothy Fleischer for her kindness, generosity and help with keeping the group organized.

Mark Mondol, James Daley and Kurt Broderick for their technical support with fabrication.

Marco Turchetti for being a true friend and a very professional colleague who has widened my view on a long list of subjects.

My dearest colleague, Sila Deniz Calisgan, for bringing joy, laughter and excitement in my life. I feel very fortunate for having been able to share all the moments of this adventure with her.

My parents, and both grandfathers, who are the very reason behind my determination. I am grateful for their unconditional trust and encouragement. None of this work could be possible if it weren't for their dedication.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Fundamental Properties of Superconductors

The resistance of a normal metal gradually decreases as the temperature is lowered and saturates at very low temperatures[1]. However, some metals, undergo a phase transition to the superconducting state (zero resistance), below a critical temperature (T_c). Some typical T_c values for elemental superconductors are; 9.25 K for niobium, 1.1 K for aluminum and 7.1 K for silicon under high pressure [3]. For transition metal cuprate compounds T_c values of $\geq$120 K have been reported while the record is being held by H_2S with 203 K (at $\approx$ 150 GPa) [13].

In addition to the perfect conductivity, superconductors exclude magnetic flux. This perfect diamagnetism is named as the Meissner effect and can be used to differentiate a perfect electrical conductor and a superconductor. Similar to a critical temperature, there exists a critical magnetic field (H_c) where the phase transition occurs. Typical values for H_c can be given as 198 mT for niobium and 9.9 mT.

As a consequence of perfect conductivity, if a current is somehow injected into a superconducting loop, it continues to flow indefinitely. This property is also limited by a current carrying capacity, known as the critical current density (J_c). Interestingly, the resulting flux due to these currents is found to be quantized in units of $\Phi_o = \frac{h}{2e}$. In this work, these phenomena of 'digitized persistent currents' will be used extensively

[1]The resistance at the absolute 0 is determined by impurity scatterings and defects in the metal.

in Chap.3 and Chap.4.

The aforementioned quantization effect indicates that superconductivity is a macroscopic quantum mechanical phenomenon. This has been followed by a description of superconductivity where all of the superconducting electrons are represented by a single wave function. According to the BCS theory of superconductivity, physical mechanisms behind these effects lie in the pairing of the fermionic particles to form a Bose gas of paired quasiparticles. This process is mediated by electron-phonon coupling and results in the creation of Cooper pairs [10]. The reasoning of such a pairing can be (over)simplified by using minimizing Hamiltonian argument.

The full description of superconductivity deserves an extensive explanation of the physics. For the sake of conciseness, we will skip most of these curious phenomena such as the superconducting energy gap, quasi-particle tunneling, supercurrent equation, the two-fluid model, vortex phases and flux pinning. Instead, we will skip to explaining some of the phenomena and terminology that are of particular importance for the work conducted in this thesis. The reader can use the following references for detailed coverage of the superconductivity: Refs.[42, 33]. The author also wants to cite the lecture notes prepared for the MIT course 6.732 by Prof. Mildred S. Dresselhaus, which are also extensively used in the preparation of this introductory chapter.

1.2 Small Superconductors

Small superconductors refer to superconducting objects with scales smaller than the characteristic lengths of such materials in one or more dimensions [32]. These fundamental length scales are the magnetic penetration depth λ [28], and the coherence length ξ [17]. Former defines the extent of the magnetic field can penetrate into the superconductor while the later indicates the spatial extent over which the superconductivity does not get affected by any local perturbations (e.g. thermal fluctuations).

In this work, we have worked with thin film superconductors of thicknesses varying between 5-30 nm. In such dimensions, it is known that the film's surface and interface

confine the motions of the electrons, leading to the formation of quantum well states [8]. These states heavily influence the overall electronic structure, density of states, and the surface energies.

Understanding these length-scales and the resultant effects have been fundamental for the work conducted in this thesis. For example, the coherence length plays a major role in the switching characteristics (hot-spot dynamics) of a superconducting nanowire. Although we do not cover them in this work, vortex ratchets and flux pinning devices make use of magnetic penetration depth as well. As can be expected, capturing these effects in our simulator will be fundamental in explaining the superconducting device characteristics.

1.3 Superconducting Nanoelectronics and Cryogenic Computing

Following the prediction of tunneling possibility of superconducting Cooper pairs in a superconductor-normal-superconductor (SNS) system, superconducting electronics have been in the focus of research enabling a variety of novel applications in both analog and digital electronics. These applications can be exemplified as superconducting quantum interference devices (SQUIDS), millimeter-wave mixers, voltage standards, high energy magnets, power transmission lines, and most importantly high-performance computing systems [37].

In the absence of phonon scatterings at the cryogenic temperatures, electron transport becomes ballistic. This in return allows the superconducting phase information to be preserved over longer times, that is used as the main method of information encoding in quantum computational systems. In combination with the advancements in superconducting logic families such as single flux quantum (SFQ/RSFQ and eSFQ), reciprocal quantum logic (RQL) and adiabatic quantum flux parametron (AQFP) [41], advantages of Josephson switching devices have been utilized more and more extensively in computing.

In addition to being uniquely situated for quantum computing approaches with their long coherence times, superconducting electronics also have innate characteristics that make them suitable for conventional electronics. The energy spent per switching can be as low as a fraction of an aJ with switching speeds on the order of hundreds of GHz.

Considering the unique and multidimensional physics of superconductors, complicated highly non-linear behavior of devices and demanding nanofabrication techniques, producing a fully-functional superconducting electronics chip can be immensely expensive. To make the field feasible to compete with conventional electronics and allow it to prevail in niche applications of computation, advanced simulation capabilities is an indisputable necessity.

1.3.1 Thesis goal

The main goal of this thesis work is to construct a unified simulation platform where superconducting electronics can be designed and optimized with high accuracy and performance. The simulator will be optimized to be able to deal with the strong nonlinearity of superconducting switching elements and will include a variety of physical disciplines such as quantum mechanics (e.g. flux quantization), thermodynamics (e.g. hot-spot formation and dissipation) and electrical characteristics. Following the verification of the predictive capabilities of the framework, layout and material considerations will be optimized for existing applications. Finally, a novel application will be presented in the heavily researched deep neural network training field, fully exploiting the potential of superconducting nanowire-based devices.

1.3.2 Thesis outline

This thesis will be organized as follows:

Chapter 2 - Construction of the Superconducting Circuit Simulator

The physical phenomena behind the Josephson junctions and the superconducting nanowires will be analyzed. Models of these devices will be incorporated in a simulator that uses advanced numerical techniques to obtain a high-performance and accuracy designing platform. Numerical problems that arise due to the strong non-linearity of the superconducting devices will be analyzed and mitigation techniques will be provided. Validation of the simulated responses will be made by comparisons with experimental data in device and circuit levels.

Chapter 3 - Optimizing Layout and Material Considerations for Superconducting Nanowire-Based Electronics

The simulator will be used to explain the behavior of two superconducting nanowire-based systems, a nanowire memory, and controlled flux shuttling. Operation margins of the nanowire memory will be derived and its sensitivity will be analyzed with respect to its circuit model parameters. Nanowire memories will be optimized for having wider operation margins. These optimal devices are then fabricated using nanofabrication techniques and experimentally characterized in liquid helium immersion measurements. Optimization results obtained by the software will be validated with these experimental results.

Secondly, the software will be used to better understand the controlled flux-shuttling dynamics of shunted nanowires. Simulated responses will be compared with former experimental characterizations as validation. Then, shunted nanowires are optimized by changing material stacks that involve higher thermal sinking of the nanowire, suggested by the simulator we have built. Such devices are realized again by using nanofabrication techniques and measured similarly in liquid helium dewar. Experimental characterization of these optimized shunted nanowires will show superior characteristics which will be used as a basic element in the realization of the system we propose in Chap.4.

Chapter 4 - Design of a Superconducting Deep Neural Network Training Accelerator

The simulator built in this work is finally used in creating a novel application field for the superconducting nanoelectronics, mixed-signal accelerators for deep neural network training. A quick review of DNN fundamentals and crossbar architecture will be provided. Then, the conventional resistive crossbar will be adapted into a superconducting nanowire-based alternative version. This approach mitigates the foregoing problem with the classical non-volatile memory approaches by providing inherently suitable physical characteristics for the application. A cross-point element will be fabricated, making use of the optimized shunted nanowires we have designed in Chap.3.2.3. Experimental characterization of the unit cell will be presented with an overview of system level requirements to realize a full-scale implementation.

Chapter 2

Construction of the Superconducting Circuit Simulator

2.1 Introduction to Superconducting Devices

Superconducting devices that this work will include are Josephson junctions (JJs) and superconducting nanowire (SCNW) based devices. Having very different dynamics and operational principles increase the capability of superconducting electronics solutions when interfaced in hybrid designs. This chapter will discuss the physical foundations of these devices and give proper modeling of them.

2.1.1 Josephson Junctions

Josephson junctions can be defined as a tunneling device, using Cooper pairs as the charge carriers. They are realized by sandwiching a barrier layer in between two superconducting layers. If this barrier is sufficiently thin ($\approx$ few nanometers), the superconducting wavefunction (describing the probability of finding a Cooper pair) can tunnel through this insulating layer. The super-current (Josephson current) flowing through the device is described with the Josephson equation given below:

$$I_s = I_c sin(\gamma), \tag{2.1}$$

$$\gamma = \phi_2 - \phi_1 - \frac{2\pi}{\Phi_\circ} \int_{S1}^{S2} Adl, \tag{2.2}$$

where ϕ is the phase of the macroscopic wave function, γ is the gauge invariant phase difference between the two superconductors and I_c is the critical current of the device [21]. When a direct voltage V is applied to this stack, γ varies as a function of time as follows:

$$\frac{d\gamma}{dt} = \frac{2e}{h}, \tag{2.3}$$

where e is the charge of an electron and h is the Planck constant [22]. As can be seen from the Eq.2.1, the current-voltage characteristics of the Josephson junctions are highly nonlinear.

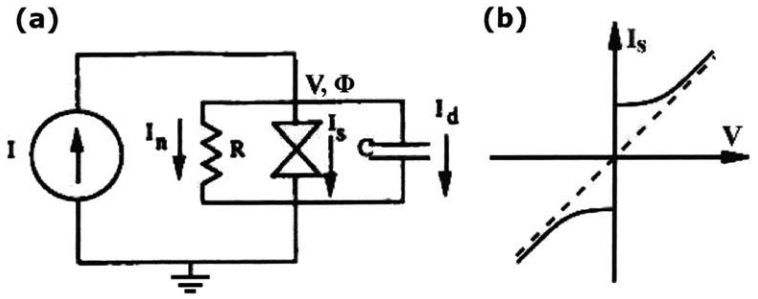

Figure 2-1: RCSJ model for a shunted Josephson junction and its characteristic non-hysteretic current-voltage relation. Figure modified from Ref. [22].

RCSJ Model of Josephson Junctions

A common way to describe the behavior of the Josephson junction is its alternative circuit model: Resistive-Capacitively-Shunted junction (RCSJ). Using the RCSJ model shown in Fig.2-1, Josephson junctions can be described by using the following constitutive equations Eq.2.4 and Eq.2.5, where I_c, R and C are device parameters and ϕ is the phase difference of superconducting order parameter θ between the two terminals of the junction [22]:

$$I = I_s + I_n + I_d = I_c sin(\phi) + \frac{\hbar}{2eR}\frac{d\phi}{dt} + \frac{\hbar C}{2e}\frac{d^2\phi}{dt^2}, \tag{2.4}$$

$$V = \frac{\hbar}{2e}\frac{d\phi}{dt} = \frac{d\Phi}{dt}. \tag{2.5}$$

In steady state, this behavior can be linearized as: $\Phi = L_{J0}I$, where $\Phi = \hbar\phi\backslash 2e$, $L_{J0} = \hbar\backslash 2eI_c$, and ϕ is the superconducting wavefunction phase. Therefore, JJs can be modeled as inductors if the phase is used as the nodal quantity and current as branch quantity.

SFQ Logic Gates

The dominant superconducting integrated circuit technology involving the implementation of Josephson junctions is single flux quantum (SFQ) logic. This technology uses arrangements of inductors and JJs to perform complex computation at clock rates beyond 20 GHz [37]. The inputs and outputs of the SFQ systems are fast voltage pulses that represent logic states, which have the integral of, as the name suggests, a single flux quantum which is $\Phi_o = h\backslash 2e = 2.067\,833\,831$ fWb.

Fig.2-2 illustrates a sample SFQ logic based NAND gate consisting of 23 current sources, 25 inductors, and 35 JJs. The circuit is constructed as an AND gate (bottom left) followed by a NOT gate (bottom right), where the top part is basically a delay line, namely a Josephson Transmission Line (JTL), for the clock signal to arrive at the NOT gate at the same time as AND output. The schematic also includes circuitries that give directionality to the SFQ signal (e.g. J24-J25). These valve structures avoid bleeding of the output signal back to the circuit instead of getting transmitted to the next stage. As will be shown in Sec.2.3 this circuit can be clocked at 500GHz where the output follows the input with one clock cycle delay.

2.1.2 Superconducting Nanowire Based Devices

Nanowire-based devices can be considered as the second main family of superconducting electronics. The main operation of these devices is obtained through switching

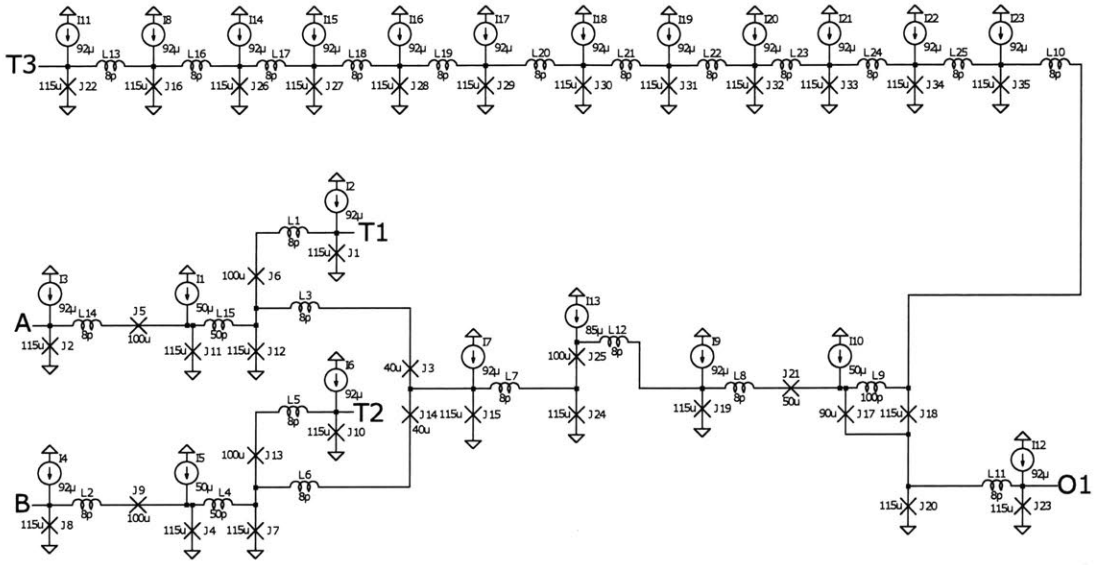

Figure 2-2: Example SFQ logic based NAND gate implementation. High device count is mainly due to the transmission line (top branch) working as delay elements to satisfy pulse timing requirements.

these devices (from the superconducting state to resistive state). These switching events are thermally mediated by the growth of a hot-region, often called as a hot-spot. The growth of these hot-spots is initiated once the current through the device exceeds the switching current of the device. On the other hand, the decay of the hot-spot occurs at lower than this threshold due to Joule-self heating. In other words, the switching characteristics of these devices are hysteretic.

The switched (i.e. normal, resistive) region in a nanowire has high electrical impedance. Therefore, they can potentially fan out to many devices. This property has allowed these devices to perform as interfaces between JJ based circuits and CMOS architectures [48]. However, thermal breakdown of the superconductivity prevents these devices from supporting quantum coherent transport. This behavior is expected as the switching event is essentially a burst of phonons that scrambles the phase information. Furthermore, again due to their thermal characteristics, switching frequencies of nanowire-based devices are lower than the JJs.

The most basic structure in this device family is a nanowire constriction[1]. A

[1]Note that in Chap.3 and Chap.4, we sometimes refer to this constriction geometry as simply nanowire and use the term interchangeably.

nanowire constriction is essentially a narrowing in a nanowire, that has a reduced switching current[2] relative to the surrounding region. As mentioned above, exceeding this threshold leads to hot-spot growth in this constricted region. The area difference between the constricted part and the rest ensures the hot-spot event to occur reliably in a well-defined region. Once the hot-spot blocks the entire cross-section of the nanowire, the device is called to be switched into the normal (resistive) state[3].

The nanowire constriction can be interpreted as a non-linear resistor and a non-linear inductor as a function of current passing through it. The resistance of the constriction is related to the hot-spot through the simple linear relation given below:

$$R_{\mathrm{HS}} = \frac{\rho l}{dw}, \tag{2.6}$$

where ρ and d are the resistivity and thickness of the superconducting film, w is the constriction width and l is the hot-spot length.

The rate at which the hot-spot size changes (growth or decay) is called the hot-spot velocity. The hot-spot velocity is dependent on the current flowing through the nanowire in a hysteretic relation. Nucleation of it only occurs once the current flowing through the device exceeds its switching current. Once it forms, its growth/decay depends on whether the Joule heating ($\propto I^2R$) or the cooling is more dominant. As a definition, the current at which the hot-spot neither grows nor decays is called as the hot-spot current I_{HS} or the retrapping current I_{R}. Note that this current is roughly one-fifth of the switching current (for thin film NbN devices on an insulating substrate), giving rise to the hysteretic current-voltage relation of the nanowire-based devices. The equation for the hot-spot velocity (ν_{HS}) is given as following:

$$\nu_{\mathrm{HS}} = \frac{dl}{dt} = 2\nu_0 \frac{\psi i_{\mathrm{NW}}^2 \backslash I_{\mathrm{SW}}^2 - 2}{\sqrt{\psi i_{\mathrm{NW}}^2 \backslash I_{\mathrm{SW}}^2 - 1}}, \tag{2.7}$$

where ν_0 is the characteristic velocity and ψ is the Stekly parameter [23]. These terms

[2]Here we choose to use the term switching current instead of the critical current to account for all the possible suppression effects over the depairing current.

[3]Before this point current flowing in the wire can bypass the hot region and flow around. The device response does not change until the device is switched into the normal state as the superconducting part short-circuits the rest before.

can be written in terms of material and environment parameters as:

$$\nu_0 = \frac{\sqrt{h_c \kappa \backslash d}}{c}, \tag{2.8}$$

$$\psi = \frac{\rho I_{SW}^2}{h_c w^2 d(T_c - T_s)}, \tag{2.9}$$

where κ is the thermal conductivity of the nanowire, h_c is the thermal contact conductivity, c is the specific heat per unit volume of the nanowire and $T_{c,s}$ are the temperatures of the constriction and the substrate respectively.

Considering that the nanowire is constricted only for a limited extent, l_{HS} has to be less than or equal to the $x_{\text{constriction}}$, which is the physical length of the constriction. On the other hand, the minimum size of a hot-spot is given by the coherence length ξ of the material. This length is a function of temperature as given below:

$$\xi(T) = \frac{\xi(0)}{\sqrt{1 - \frac{T}{T_c}}}. \tag{2.10}$$

For the sake of simplicity, we will ignore this effect of temperature change around the hot-spot region.

As can be inspected from 2.7, ν_{HS} diverges as the denominator approaches to 0. This creates a mathematical instability for modeling nanowire behavior as each switching event involves that particular point. The simplest solution is to saturate this velocity at a certain level. This approach makes physical sense as well considering that the hot-spot decay (a thermal cooling event) cannot occur at an arbitrarily fast rate (i.e. faster than the phonon escape time). In order to pick the fastest rate, we use the hot-spot size a heat capacity, as it is the over time integration of ν_{HS}. Then, we assume an exponential decay and modify 2.7 as follows:

$$\nu_{HS} = max(\nu_{HS}^{Eq.2.7}, -\frac{l_{HS}}{\tau_{th}}), \tag{2.11}$$

where l_{HS} is the hot-spot size at that time instance.

As can be interpreted from the equation set Eq.2.6 - Eq.2.11, the switching event

with a load attached to the nanowire can be described as: (1) switching ($I_{\mathrm{NW}} \leftarrow I_{\mathrm{SW}}$), (2) current redirection until growth stops (($I_{\mathrm{NW}} \leftarrow I_{\mathrm{R}}$)), (3) decaying and eventually diminishing of the hot-spot ($\int_{I_{\mathrm{NW}}=I_{\mathrm{SW}}}^{I_{\mathrm{NW}}^{final}} d\nu_{\mathrm{HS}} \leq \xi$). The final current in the nanowire at the end of the switching event is therefore determined by: (1) the electrical time constants that determine the rate of change in the current flowing through the nanowire and (2) the thermal time constants that determine how fast hot-spot events can take place. Chapter 3.2 will further discuss the methods and results of modifying these characteristic timescales. Furthermore, one can find a more thorough analysis of modeling nanowire-based devices in Ref.[4].

Most importantly, as the devices are purely classical (i.e. not operated with quantum mechanical principles), the modeling of them can be done with a simple circuit model. In other words, unlike JJs described in Sec.2.1.1, the evolution of phase across the device can be neglected. This factor allows us to apply model order reduction and simplifies the numerical complexity of the system significantly.

Here we want to note that the aforementioned devices make use of a phenomenon called the kinetic inductance[2]. Similar to the conventional definition of self-inductance L in a circuit that is associated with the energy stored in the magnetic field produced by the electrical current, the kinetic inductance, L_{k}, can be defined as the mechanism that stores energy directly related to the motion of the charge carriers. For an electron, the motion equation can be written as:

$$\frac{d\nu}{dt} = -\frac{qE}{m} - \frac{\nu}{\tau}, \tag{2.12}$$

where m is the mass (9.1094×10^{-28} g) and q is the charge of an electron (1.6022×10^{-19} C). Writing the same equation in terms of the current density $J = -qn\nu$:

$$E = (\frac{m}{q^2 n\tau})J + (\frac{m}{qe^2})\frac{dJ}{dt} = \rho J + \Lambda\frac{dJ}{dt}. \tag{2.13}$$

As can be seen from Eq. 2.13, the very motion of the electron gives rise to an effect that can be interpreted as an additional inductive behavior with kinetic inductivity Λ. Kinetic inductance is existent all materials but its effect becomes pronounced in

superconductors due to the zero series DC resistance [22]. This property will be used extensively in Chapters 3 and 4 in designing area-efficient large inductors.

Superconducting nanowires can be used for a variety of interesting applications. For example, the recently demonstrated nanowire cryotron (or nTron) [30], makes use of nanowire constrictions in its operation. Furthermore, in a y-shaped geometry, nanowire devices can utilize the current crowding effect for sensing applications (e.g. as a readout device alternative to SQUIDs)[29]. In this work, we will focus on two main examples to show the simulation and optimization capability of the simulator: (1) a nanowire memory (nMEM) in Sec.3.1, and (2) a shunted nanowire for controlled flux shuttling in Sec.3.2. These applications have been proposed previously in Ref. [49] and Ref. [43], where this work improves their characteristics using the software we build out of scratch. Finally, as a novel application, Chap.4 discusses the usage of superconducting nanowire-based devices in deep neural network training acceleration.

2.2 Superconducting Circuit Simulators

Computer-aided design (CAD) platforms for superconducting electronics have been of interest in the last three decades [34, 47, 16, 15]. The common main goal of these softwares is to create a high-performance design environment, similar to those of semiconductors owe their success to. Unfortunately, direct implementation of existing CAD softwares to superconductor industry is not possible. The reasons behind this incompatibility can be listed as: (1) physical level disparities such as the carrier types (Cooper pairs instead of electrons and holes) and presence phase coherence (for superconducting devices) (2) circuit level discrepancies such as different basic active components (Josephson junctions instead of transistors), passive components (inductors instead of capacitors), and interconnects (Josephson transmission lines instead of metal lines) can be listed as the circuit level reasons behind this incompatibility; (3) logic level differences (SFQ logic instead of CMOS logic) [16]. In order to enable the design of early SFQ logic digital circuits, CAD tools specifically built for superconducting electronics started to appear. These tools consist of circuit simulators (such

as JSpice[47] and [34]), optimizers, layout tools, inductance estimators (post-layout simulators such as Lmeter [5]), and logic simulators [16].

A variety of superconducting circuit simulators are available in two main groups: (1) SPICE-modified ones such as HSpice[40], WRSpice; and (2) original simulators such as NioCAD[35], JSim [14], and PSCAN[34]. Here we want to acknowledge that these design tools and their predecessors (e.g. COMPASS, Jspice3, Spice 3f4) have accomplished an amazing task and immensely improved the design and development of superconducting electronics, and in particular large-scale SFQ-based circuits. However, none of these softwares has libraries designed for superconducting nanowire-based circuits or hybrid applications.

Considering that nano-cryotron based devices have started showing promising characteristics [30, 29] and applications [49, 48, 44], design platforms that are optimized for developing this novel family of devices is necessary. This initiative has been backed up by accurate depiction of the electrothermal dynamics behind the nanowire operations [23]. Recently, a SPICE implementation of this model has demonstrated results for superconducting nanowire-based single photon detectors (SNSPDs) [4]. Similarly, nanowire models in the WRSpice environment have been created to design memory cells [49]. However, due to the absence of low-level optimization to support hot-spot dynamics, the performance of these simulators is limited.

In this work, we have aimed to create a platform that efficiently simulates both Josephson junctions and superconducting nanowires. Furthermore, the simulator built in this work gives the user the capacity to decide between performance and accuracy, which is highly useful for various steps of designing large-scale systems. We provide details of the solver we have constructed and validation results to show the capability of this software. It is further used in optimizing circuit and material level design parameters, for nanowire-specific applications (See Sec.3.1.4 and Sec.3.2.3)

2.2.1 Simulation Framework

The simulator in this work is constructed in MATLAB environment and interfaced with LTspice for the ease of use. The software we build here can support input

sources (voltage/current), resistors, inductors (geometrical and kinetic), Josephson junctions and superconducting nanowires. Currently, it cannot support layout level simulations, but future work certainly intends to address this as well.

The system first inputs a netlist file for the circuit to be simulated (generated and imported from LTSpice) and an input file (programmed in MATLAB). Then, the system matrix is generated following a standard stamping procedure. At this point, the simulator chooses one of the two methods (solvers) depending on the elements in the circuit. Specifically, if there are any Josephson junctions in the netlist, it operates with the 'phase coherent solver'. This solver is capable to capture the phase evolution across the device according to the dynamics discussed in Sec.2.1.1. In the absence of any JJ devices, we simplify the system and use the classical solver. This solver is essentially a standard circuit simulator with an embedded superconducting nanowire model.

As can be expected, both of the solvers are based on solving for Kirchhoff Laws in discrete time steps. The classical solver uses branch currents as the state variables, where the phase-coherent solver uses the superconducting phase as well as the branch currents. Upcoming sections describe the common and different properties of these two solvers and delineate the advantages and tradeoffs of these decisions.

2.2.2 Phase Coherent Circuit Solver

Description of the phase evolution is fundamental to capture the characteristics of the Josephson junctions. In order to achieve this efficiently, we expand the state variables to branch currents and node phases (of the superconducting wavefunction). Describing the JJs with the RCSJ model explained in Sec.2.1.1 requires solving a non-linear system, governed by a 2^{nd} order partial differential equation (PDE) in the form of:

$$F(\Phi_i, \frac{\partial \Phi_i}{\partial t}, \frac{\partial^2 \Phi_i}{\partial^2 t}) = 0. \tag{2.14}$$

Definition of the node phases as a state variable allows us to break this system

into two 1st order PDEs in the form of:

$$F(\Phi_i, V_i, \frac{\partial V_i}{\partial t}) = 0, \tag{2.15}$$

$$\frac{\partial \Phi_i}{\partial t} = V_i. \tag{2.16}$$

Removal of the second derivative will allow us to implement a trapezoidal integration method at the cost of quadrupling the system size as following:

$$\begin{bmatrix} \frac{\partial V}{\partial t} \\ \frac{\partial \Phi}{\partial t} \end{bmatrix} = \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \begin{bmatrix} V \\ \Phi \end{bmatrix} + \begin{bmatrix} \frac{I_S}{C} \\ 0 \end{bmatrix}. \tag{2.17}$$

This operation can be viewed as trading memory off for speed, which is a design choice we made for our application.

The phase-coherent solver we have built uses advanced numerical methods for describing and solving the system. At each time step, we start with producing an initial guess using forward Euler method. This computationally cheap operation is highly beneficial, as we proceed from a meaningful starting point. Obviously, forward Euler method cannot be used for the rest of the system as it would suffer from severe convergence issues. Therefore, we continue with a trapezoidal integration method in our solver.

Considering that the switching behavior of the JJs is highly nonlinear, implementation of Newton's method is selected. Within the Newton's method, we have chosen to use gradual conjugate residual (GCR) technique to solve the linearized systems. Note that an approximate solution is sufficient around the switching point of the devices as the non-linearity is less severe away from the switching point. Therefore, the GCR method is particularly well-suited for our application as it allows us to trade precision for performance. The pseudocode for the implementation is shown below.

In this code $\bar{F}(x)$ represents the problem's set of conservation laws evaluated at x and $\bar{J}(x)$, which is its Jacobian matrix. $\bar{F}(x)$ should not be confused with the trapezoidal method's objective function $\bar{F}_T(\bar{x})$ (with its Jacobian function $\bar{J}_T(\bar{x})$) where

Algorithm 1 Phase Coherent Circuit Solver

import netlist
build connection matrices

1: **while** t $< t_{end}$ **do** ▷ Trapezoidal Method Loop
2: **evaluate** $\bar{F}(\bar{x}^n)$, $\bar{J}(\bar{x}^n)$
3: $\bar{x}^{n+1} \leftarrow$ **Forward Euler** to calculate initial guess
4: **while** $\bar{x}^{n+1}$ - $\bar{x}^n >$ tolerance **do** ▷ Newton's method loop
5: **evaluate** $\bar{F}(\bar{x}^{n+1})$, $\bar{J}(\bar{x}^{n+1})$
6: **construct** $\bar{F}_T(\bar{x}^{n+1})$, $\bar{J}_T(\bar{x}^{n+1})$
7: $\delta\bar{x} \leftarrow$ **GCR** to solve $\bar{J}_T \delta x = -\bar{F}_T$
8: $\bar{x}^{n+1} \leftarrow \bar{x}^{n+1} + \delta\bar{x}$
9: **end**
10: **adjust** time step size
11: **end**
12: **plot** results

$$\bar{F}_T(\bar{x}^n) = \bar{x}^{n-1} - \bar{x}^n - \frac{\Delta t(\bar{F}(\bar{x}^{n-1}) + \bar{F}(\bar{x}^n))}{2}. \tag{2.18}$$

Both of those Jacobian matrices are computed analytically at each time iteration. We want to note that we have observed Jacobian implicit techniques provide surprisingly lower performance. Future work might include diagnosing and addressing the reasons for this unusual behavior to obtain even higher performance. Finally, the author wants to thank and acknowledge the contributions of Marco Turchetti and Brenden A. Butters in realizing the phase coherent solver.

2.2.3 Classical Circuit Solver

Superconducting nanowire-based devices (as well as the conventional electrical circuit components such as resistors, inductors etc.) do not require capturing the phase evolution to describe their operation. Therefore, for the circuits that do not involve any JJs, expanding the state variables to include the phase (See Eq.2.17) is redundant. Nonetheless, quantization of current in a superconducting loop is the same for nanowire-based devices. The mathematical mechanism in the model that ensures this behavior for the JJs is the sinusoidal term in the Josephson equation (Eq.2.4).

This term also ensures single-flux shuttling events due to the time derivative of the gauge invariant phase difference it inputs. Nanowires, that allow multi-flux switching events, can be modeled similarly by higher order sinusoidal terms. In this simulator we have implemented a rounding function (to the closest quantized state), which can be viewed as the limit case of a high order sinusoidal.[4]

However, during a transient, rounding the state at each time step can prevent any change to occur. To exemplify, in order to change the rounded state, the phase progression at a single time step should be larger than $0.5\Phi_o$. In most cases, a rate of change of this size is rejected by the tolerance of the solver to avoid instability issues. In order to overcome this situation, we have chosen to apply flux quantization only in the steady state of the system (computed completely ignoring the quantization of flux along the loop). We have not observed any problem caused by this simplification, as the nanowire-based devices we are concerned with do not change operation as a function of phase progression across them.

The removal of the sinusoidal term and enforcing the quantization of flux only under the steady-state conditions reduces the computational complexity of the system significantly. This method can be interpreted as a model order reduction, as we remove some of the complex dynamics, temporarily, to get a crude result, and then fine-tune it to its final form. Furthermore, in the absence of these hard non-linearities, the system becomes solvable by the built-in backslash operator (\) in MATLAB without having any performance issues. The physical speeds of the nanowire-based devices, that are thermally controlled, are slower than that of the JJs which also support this decision. Therefore, instead of the Newton solver we have constructed for the Josephson junctions in Sec.2.2.2, we simply use (\). This method relieves us from explicitly calculating the Jacobian matrix for the system as well. The pseudocode for the classical circuit solver is given below.

As a further step, one might consider running a full-scale (1D, 2D or 3D) elec-

[4]We have not observed any noticeable differences between this approach and $(sin(\phi))^n$ functions in the steady state results. Considering that the choice for the sinusoidal order, n, would have been arbitrary either ways, the rounding function is found to be adequate for exploring the behavior of the devices we investigated in this work.

Algorithm 2 Classical Circuit Solver

```
1: import netlist
2: build connection matrices
3: while t <t_end do
4:     while x̄^{n+1} − x̄^n > tolerance do
5:         generate system[t_0 + Δt]), state[t_0 + Δt])
6:         x̄^{n+1} ← system\state
7:         adjust time step size
8:     if Steady State then                ▷ No hot-spots and state variables are idle
9:         enforce flux quantization
10:    end
11:    t ← t + Δt
12: end
13: plot results
```

trothermal simulation for each individual hot-spot. This approach might be beneficial for applications where the time registration of each event is crucial such as single photon detectors. An example application can be photon coincidence detection scenarios [50]. However, this approach also requires a better understanding of the hot-spot dynamics, such as how they initiate and fully disappear. Furthermore, it would also lead to a significant increase in simulation times.

2.2.4 Dynamic Time-Stepping

Superconducting circuits involve electrical and thermal events that occur in orders-of-magnitude different time scales. The shorter time constant events (e.g. SFQ pulses $\approx 1\,\text{ps}$) require even smaller time steps to provide sufficient accuracy. However, when the longer time constant events (e.g. electrical time constants $\approx 10\,\mu\text{s}$) are tried to be computed with the same time steps the simulation times become unmanageable. Therefore having a constant time step size is not viable for simulating these systems. This problem can be solved by adjusting the time step using the algorithm below:

The pseudocode given above is used in Algorithm 1 and Algorithm 2 in the lines noted as **adjust** time step size.

In order to control the time step adjuster, we first check if the system converges to a solution or not. If the solution does not converge, a finer time-step is used, until

Algorithm 3 Adaptive Time Scaling

define $\nu_{\Delta t}$: rate of change of Δt
define $a_{\Delta t}$: rate of change of $\nu_{\Delta t}$
define $c^{+,-}$: positive and negative hysteresis counters

```
1: if change < tolerance then                    ▷ Solution accepted: Accelerate
2:     Δt ← Δt × β
3:     increment c⁺                              ▷ Increase positive counter
4:     set c⁻ ← 0                                ▷ Reset negative counter
5:     if c⁺ = Countdown then
6:         accelerate β ← β × α                  ▷ Increase acceleration factor
7:         reset c⁺ ← 0                          ▷ Reset positive counter
8: else                                          ▷ Solution declined: Decelerate
9:     Δt ← Δt/β
10:    increment c⁻
11:    set c⁺ ← 0
12:    if c⁻ = Countdown then
13:        decelerate β ← β/α                    ▷ Increase deceleration factor
14:        reset c⁻ ← 0
```

the convergence is obtained. Additionally, we determine a preset maximum change for the state variables. Once these variables change more than this limit within a single time-step, even though the solution converges, solver rejects the answer and decelerates the simulation speed.

However, it must be noted that changing the time step size rapidly can cause oscillatory and unstable behavior. Using only proportional control (modifying the time step only looking at the current step, and using a constant rate) is observed to be insufficient for our purposes (particularly for JJ-based circuits). Therefore, we have implemented hysteresis counters, operating as an integral controller (in addition to the regular proportional control) over the time step adjuster. In Algorithm 3, these features can be seen as c^+ and c^-, which are the hysteresis counters for changing the rate of change applied with the proportional controller. These terms can also be interpreted as 'momentum terms', disallowing the system change speed at an oscillatory rate.

Finally, we have also made use of the fact that simulator 'knows' the timings of the input pulses. The adaptive time-scaling algorithm reduces the time steps such

that the simulation does not 'skip-over' an input pulse. This simple method allows further acceleration, as the risk to miss an input is completely mitigated.

The structure of the dynamic time stepping algorithm provides the user the flexibility to choose between higher performance and accuracy as well. For example, at early design steps where multiple rapid iterations are required, tolerances can be set high such that the simulator provides rough behavioral estimates. Then, once the design matures, results can be resolved in finer tolerances.

2.3 Demonstrations and Performance Analysis

In order to validate the simulator, we compare the simulated responses of various devices with their experimental characterizations. For this purpose, we start with the current-voltage characteristics of the devices. Fig.2.3 shows the simulated current-voltage relation for an unshunted and a shunted superconducting nanowire.

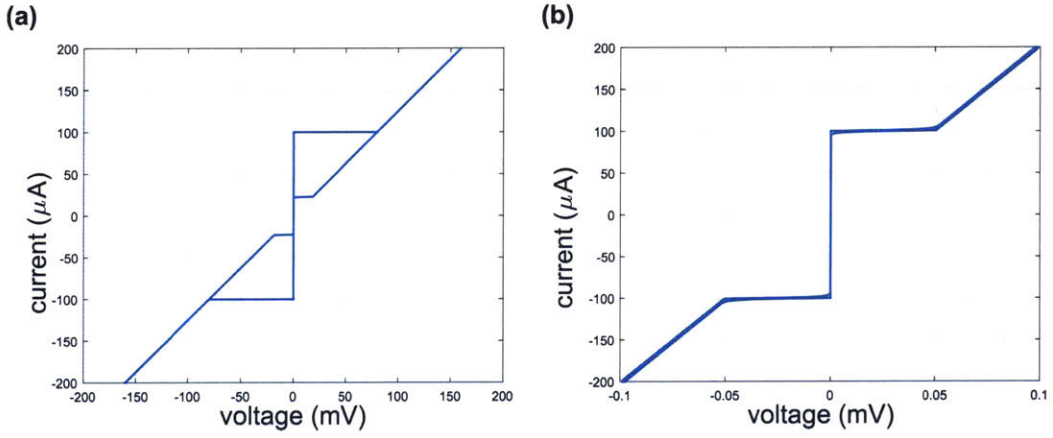

Figure 2-3: Simulated IV characteristics of **(a)** shunted and **(b)** unshunted superconducting nanowires.

It can be seen that the unshunted nanowire shows hysteretic switching characteristics (Fig.2.3a) while the shunted nanowire does not (Fig.2.3b). Circuits involving superconducting nanowires will be shown in more detail in Chapters 3 and 4. On the other hand, the work conducted in this thesis does not involve fabrication or experimental characterization of JJs. However, it certainly is a main feature of the

simulator built in this work, to be able to simulate JJ based circuits. For these reasons, here we show the simulated responses of the Josephson junction and SFQ logic circuits. Fig.2-4 shows the experimentally characterized and the simulated IV curve of a shunted Josephson junction.

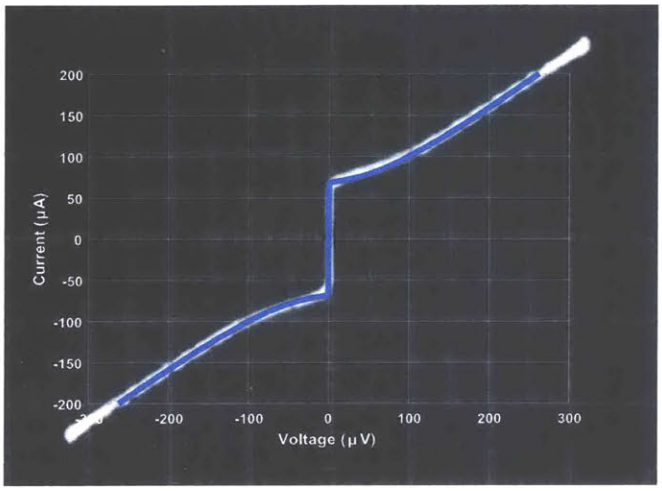

Figure 2-4: Experimental result for a shunted Josephson junction (back) taken from Ref. [46] and simulation result generated by the simulator constructed in this work (front). Switching current of the simulated JJ is directly taken from Ref.[46], while the other circuit parameters (that effectively define the Stewart-McCumber parameter) are modified manually to obtain the same characteristics, as they were not reported in the original experimental data.

It can be seen that the switching characteristics of the device are non-hysteretic as the junction we use here is resistively shunted. As it is a simulation involving a Josephson junction (the characteristics we show here in Fig.2-4 is essentially DC Josephson effect), we have used the phase-coherent solver.

Following this validation, we have also examined the area of the voltage pulse, generated by this junction. This value was obtained by integrating the output voltage of a Josephson transmission line (JTL) over time. Simulated response provided this number to be 2.0678 fV s, which is within 0.03 aV s of the expected value of the flux quantum ($\Phi_\circ = \frac{h}{2e}$).

Following the validation of the single device characteristics with our simulator, we have proceeded to test the behavioral level simulations, using well-known SFQ logic circuits. All of the logic gates (AND, OR, buffer, NOT, NOR, XOR etc.), and

fundamental elements (SQUIDs, JTLs etc.) we have tested have produced expected results, compatible with experimentally characterized responses. Here we show only a few of those examples. Fig.2-5 shows the simulated response of the SFQ logic NAND gate, which has the circuit schematic given in Fig.2-2. In the simulation panes, dashed red-lines indicate the clock timings, while the blue lines indicate the timings of the input pulses.

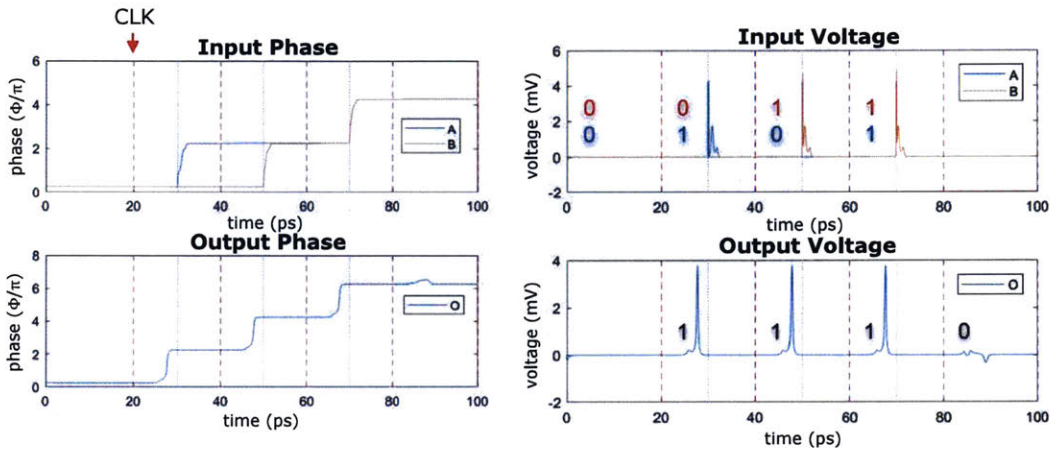

Figure 2-5: Operation of the SFQ logic based NAND gate shown in Fig.2-2. Phase (left) and voltage (right) informations are shown throughout the operation where the binary values of the signals noted on the graphs. It can be seen that output follows the input after a one clock cycle delay.

It can be observed that the SFQ logic NAND gate has one clock cycle delay between its input and output. Similarly, the circuit schematic of an SFQ logic AND gate is given in Fig.2-6. Fig.2-7 shows the input and output voltages, where the gate again shows a one-clock-cycle delay between its input and output.

As a final SFQ logic demonstration, we show a 3-bit counter, which consists of many different SFQ logic elements. This circuit features 428 current sources (for biasing the junctions), 553 JJs, and 458 inductors, creating 549 nodes, leading to a 1098×1098 system matrix.

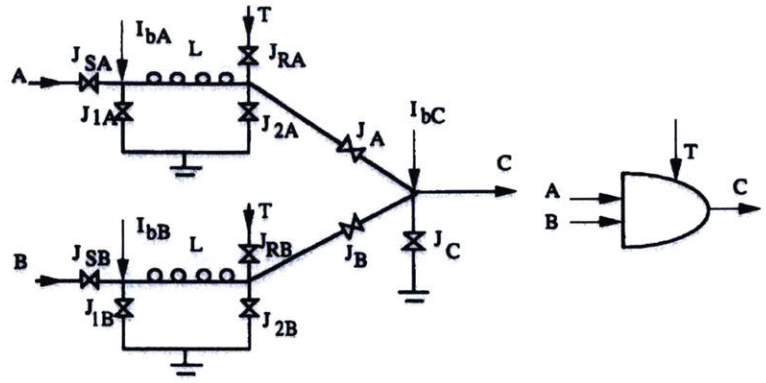

Figure 2-6: Circuit schematic of an SFQ logic AND gate. Figure taken from Ref. [22].

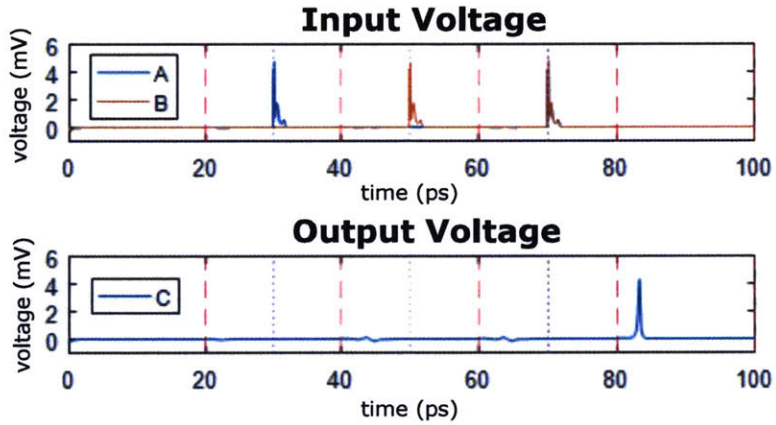

Figure 2-7: Simualted responce of the SFQ logic AND gate shown in Fig.2-6. It can be seen that the output follows the input after one full clock cycle delay.

We have observed that for the solution memory scales with $\mathcal{O}(N)$, and the peak memory with $\mathcal{O}(N^2)$, where N is the number of nodes in the problem. On the other had the computation time tends to scale with $\mathcal{O}(N^{1.5})$. The data points that have led to these conclusions can be found in Table 2.1.

Parameter	NAND	3-bit Counter
Number of Nodes	35	549
Performance	101 ps/s	1.49 ps/s
Memory	4.03 kB/s	63.2 kB/s

Table 2.1: Performance metrics of the simulator, derived from SFQ logic device simulations.

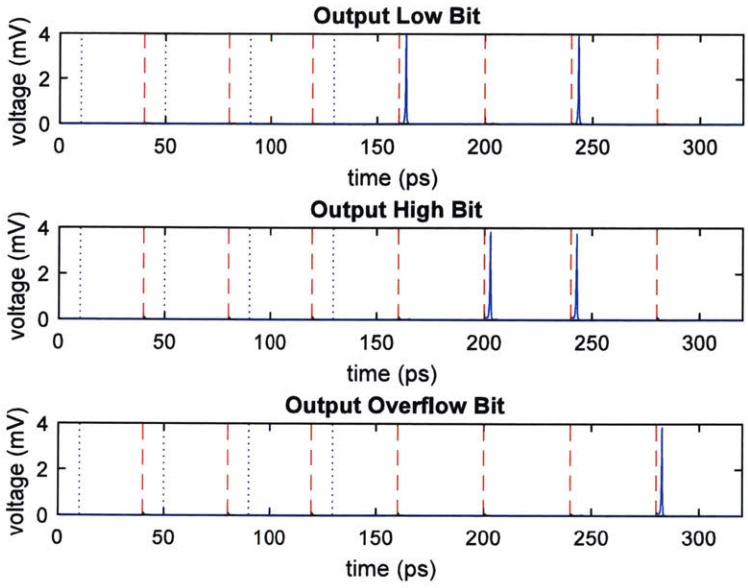

Figure 2-8: Simulated response of a 3-bit SFQ logic counter. After 100 ps, SFQ pulse readings in respective ports read [000], [001], [010], [011] and [110].

Typically, for circuits that operate at very low power and very high speed (such as SFQ logic), operation margins are very tight. Therefore, the ability to perform sensitivity analysis assists the designer greatly in realizing robust circuits. In Chapters 3 and 4 we will show how to use the simulator is utilized inside an optimizer to do so. We will also realize those devices and demonstrate the capability of the software in developing fast, scalable and power-efficient cryogenic computing elements.

Next step with this framework should be the expansion of the simulation capabilities into the layout and logic levels. Our experience has shown that many design malfunctions arise due to the underestimated parasitics, even more importantly, due to inadvertent current crowding. A layout-versus-schematic (LVS) checker would be highly beneficial to prevent such issues. Furthermore, to enable larger scale implementations a hardware description language (HDL) implementation will be required as well. Finally, effective transmission line models to better understand interconnect effects and pulse timing properties would be essential in realizing such systems.

Chapter 3

Optimizing Layout and Material Considerations for Superconducting Nanowire-Based Electronics

Superconducting electronic devices have superior properties such as; long coherence times, high switching speeds, and low energy consumption per switching. Therefore, systems built with these devices have the potential to provide faster and energy efficient architectures. Furthermore, they are uniquely situated to realize quantum information processing. However, such systems often require satisfaction of very strict design considerations. Failure in meeting these requirements result in severe degradation of performance or malfunctioning.

One method to overcome this problem is improving fabrication techniques. A good example of this approach can be nanowire-based technologies, that have easier fabrication processes with respect to their JJ based alternatives. Alternatively, in-plane fabricated JJs are recently under investigation with improved fabrication characteristics. However, in both cases, improving fabrication techniques are slow, and most importantly, immensely expensive. This is the very reason why this work aims to relax these design constraints by optimizing circuits and devices for wider margins. To achieve this goal, we will use the simulator we have constructed in Chap.2.

In this chapter we first demonstrate the complexity of superconducting nanoelec-

tronic systems, using a deceptively simple looking circuit, superconducting nanowire-based memory (nMEM). After simulating its numerous operation regimes, we optimize the unit cell to increase its tolerance to imperfections such as fabrication defects and noise. We prove the merit of the optimizer by fabricating and experimentally characterizing the optimal cell.

Secondly, we investigate another circuit that involves shunted constrictions. We use the simulator to explain device operation and provide design guidelines for future devices. Simulation results are compared with prior experimental characterizations for validation. Finally, we present a new technique which can be used to improved shunting characteristics.

3.1 Superconducting Nanowire Based Memory (nMEM)

Memory is one of the most important elements of any computing system. It has been one of the most fundamental goals of modern electronics to increase the operation speed, while decreasing the footprint and energy consumption of the memory units. Superconducting nanowire-based electronics have been of interest due to their inherent suitability for satisfying these requirements [49]. This section will first analyze the operation of superconducting nanowire-based memories. Then, the circuit design of the memory cells will be optimized using the simulator built in this work. Finally, the optimal devices will be fabricated and experimentally characterized to validate the optimality of the simulation results.

3.1.1 Fundamental Operation of Superconducting Nanowire Memory

Superconducting nanowire memory (nMEM) is essentially a superconducting loop with two constrictions. Its input/output connections are made in a way such that each side has one constriction[1] (Fig.3-1, NW_1, NW_2). Depending on the geometry

[1]Here we use the terms constriction and the nanowire interchangeably. Both of them refer to the constricted geometry of superconducting material, which has lower switching current with respect

(number of squares in the branch) of the branches, each branch is represented with an inductor (L_1, L_2) as shown in Fig.3-1. In order to make the explanation of the circuit easier, assume that the left branch shown in Fig.3-1 is named as the write branch, and the right branch is called as the read branch. This definition also brings two inequalities:

$$L_2 > L_1; \tag{3.1}$$

$$I_{\mathrm{SW},2} > I_{\mathrm{SW},1}, \tag{3.2}$$

where I_{SW} denotes the switching current (i.e. the current beyond which the constriction switches into the resistive state). The reasoning behind these choices will become clear following the explanation of the circuit operation.

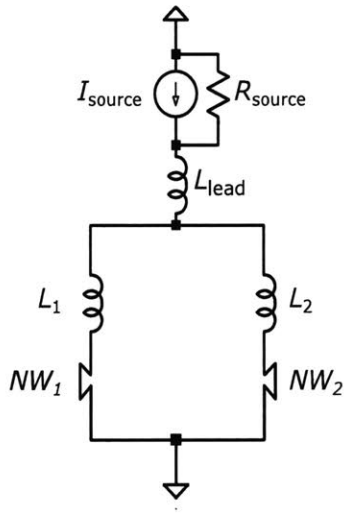

Figure 3-1: Circuit representation of a superconducting nanowire memory (nMEM) in an example setup.

When a current input is applied to the nMEM cell, it gets divided into the two branches with respect to the inductance ratio. Since we defined $L_2 > L_1$; the current flowing in the write arm (define as I_1) is larger than that of the write arm (define as I_2 ,where $I_1 + I_2 = I_{\mathrm{IN}}$). Considering that $I_{\mathrm{SW},2} > I_{\mathrm{SW},1}$ and $I_1 > I_2$, application of I_{IN} can only switch NW_1 (it can also lead to no switching at all but that is not

to the main line it resides in.

of interest here). Assuming $I_{IN} \geq I_{SW,1}\frac{L_2}{L_1+L_2}$; NW_1 switches into the resistive state. Following this switching event, the current in the write arm gets redirected into the read arm. Absence of any current in the write arm allows NW_1 to heal back to its superconducting state. The point at which this event occurs is given as:

$$I_1 = I_{SW,1} - \gamma I_{R1}, \tag{3.3}$$

where I_{R1} is the retrapping current of NW_1 (See Sec.2.1.2), and γ is a proportionality constant slightly greater than unity[2]. This behavior can be explained using the superconducting nanowire basics described in Sec.2.1.2. Note that this analysis assumes that NW_2 does not switch, even after the additional current is shuttled from the switching event of NW_1.

The excess current that is shuttled into the loop gets entrapped in the loop when NW_1 heals. This current is called as the circulating current, I_{circ}, (or persistent current) and is analytically given as:

$$I_{circ} = I_{IN}\frac{L_2}{L_1 + L_2} - (I_{SW,1} - \gamma I_{R,1}). \tag{3.4}$$

After the input disappears, this current continues to flow inside the loop where $-I_1 = I_2 = I_{circ}$. The state of the loop is determined by the value of the I_{circ}, and it is perfectly non-volatile as there are no dissipative elements while the circuit is on standby (this phenomenon is the reason why I_{circ} is also called as the persistent current).

Once this current is set, the effective switching currents $(I'_{SW,1}, I'_{SW,2})$ can be rewritten as:

$$I'_{SW,1} = I_{SW,1} - I_{circ}; \tag{3.5}$$

$$I'_{SW,2} = I_{SW,2} + I_{circ}, \tag{3.6}$$

using superposition. It can be seen that, depending on the sign of the I_{circ}, the

[2]As explained in 2.1.2 ν_{HS} becomes 0 at I_R and decreases steeply for current levels below. Therefore the final current on the branch is smaller than, but close to, I_R

switching current of one of the branches increases, as that of the other one decreases by the same amount. With the help of asymmetrically designed circuit values, this change can be used to detect the state of the cell by applying another (read) input.

Similar to the analysis given above, application of a read input, I_{read} gets divided into the branches, inversely proportional to the branch inductances. This input switches NW_1 if[3],

$$I_{\text{read}} \frac{L_2}{L_1 + L_2} \geq I'_{\text{SW},1}. \tag{3.7}$$

In order to see any DC voltage output across the nMEM, both branches should be in the resistive state. The observation of voltage can be defined as the '1' state, while not observing anything as a response to the read input is defined as the '0' state. The circuit analysis for the memory operation requires calculating the point at which NW_2 switches with the additional current coming from NW_1. The maximum current[4] shuttled from NW_1 to NW_2 is again given as Eq.3.3. Therefore, the switching condition of the read branch is given as:

$$I_{\text{read}} - \gamma I_{\text{R},1} \geq I_{\text{SW},2}. \tag{3.8}$$

As the readout requires switching of at least one constriction, it is a destructive method. Therefore, in a regular operation, each read must be followed by a re-write signal.

The overall nMEM operation can be summarized as follows:

1. Programming pulse switches the weaker (write) constriction to set a certain circulating current in the loop. State of the cell is determined by the sign (i.e. rotation direction) of this I_{circ} and can be controlled by the polarity of the input signal.

[3]Potentially, NW_2 can switch at the same time (or even if NW_1 is not switched), but those cases will be dealt in later sections.

[4]Once NW_1 heals back no additional current is shuttled to NW_2. However, this does not necessarily mean that NW_2 remains un-switched until all the excess current is shuttled. This is why we call this amount as the maximum current.

2. This circulating current effectively modifies the switching currents of both nanowires. If it flows opposite to the read input on the weak (write) constriction, no voltage is read for the read input ('0' state). Otherwise, both nanowires switch consecutively and remain in the resistive state that gives rise to a voltage difference across the nMEM ('1' state).

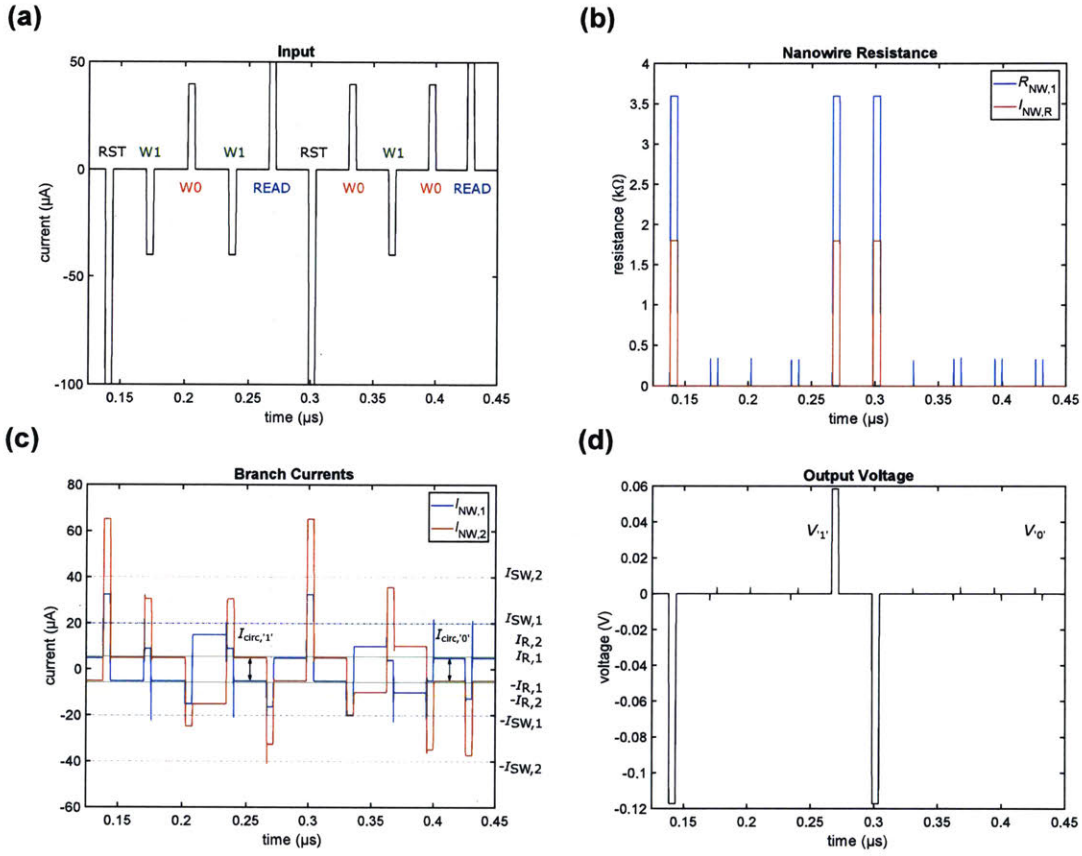

Figure 3-2: Input **(a)**, nanowire resistance **(b)**, branch currents **(c)**, and output voltage **(d)** for a given sample input-output sequence for the nMEM, that checks the two main functionality tests. (1) The output voltage is present only during the readout of the '1' state. (2) The state programming is checked for the overwriting capability. Application of the reset pulse, here shown as a large negative input, is optional. Reset pulse switches both of the constrictions irrespective of the state of the loop and sets a counter-clockwise rotating circulating current inside. The main reason for this pulse is to set the state of the cell to a deterministic level, following the readout operation.

Strictly speaking, there are two basic requirements for an nMEM cell to be accepted as operational. (1) The read input should lead to a voltage state only if the

nMEM is programmed into the '1' state. (2) Both programming inputs should be capable to overwrite the state of the cell, irrespective of its former state. A sample input-output sequence generated by our simulator is shown in Fig.3-2.

3.1.2 Unconventional Operation Modes of the nMEM

In addition to the fundamental mode, we have observed that the nMEM can be operated in other 'unconventional' modes. These modes are indeed functional (i.e. checks the basic requirements defined in the former section), however, they do not necessarily follow the analysis carried out before. We have not yet found any of these regimes providing better performance than the fundamental mode. However, understanding them is crucial to design nMEM devices with the highest possible operational margins.

The source of many of these modes is different switching sequences of NW_1 and NW_2. A list can be made to exemplify such situations as follows:

Case #1: While reading the '1' state, switching of the NW_1 might not shuttle sufficient current into the loop, to make NW_2 switch as well. However, if the read input is high enough, during the rising edge, NW_1 might switch multiple times. The shuttled current at each step builds up and can eventually switch NW_2.

Case #2: The presence of a clockwise I_{circ} increases $I'_{\text{SW},1}$ and decreases $I'_{\text{SW},2}$. In the fundamental mode, fortified $I'_{\text{SW},1}$ defines the '0' state. As an analogue to this operation, weakened $I'_{\text{SW},2}$' defines the '1' state.

Case #3: If the thermal events (e.g. hot spot formation and decay) are faster than the electrical events (e.g. redistribution of the current inside the loop), the nanowire that switched first can heal back after the second one switches (before the current gets redistributed for the second time). In other words, the current might start oscillating back and forth, instead of latching into a voltage state. We have not experimentally observed this mode, therefore it is still unknown to us if it would

qualify as 'functional'.

The presence of these modes will become important in the explanation of the results shown in Fig.3-4.

3.1.3 Operation Margins of the nMEM

This chapter has so far focused on the working principles and the different operation regimes of the nMEM unit cells. Indeed, our group has formerly been able to realize high-performance single-cell memories without requiring any design optimization [49]. However, the case has not been the same for the nMEM arrays. To briefly summarize, nMEM arrays work the same way as explained above, with the addition of heaters as cell-select devices. The main idea is that unless the heater is not turned on, none of the inputs lead to any change in the cell. When the heater is 'ON', superconductivity of the active layer is suppressed and therefore devices start responding to the electrical signals. This effect is used to choose the cell of interest without disturbing the others.

We have observed that the system-level design requirements are much more challenging to meet than that of the cell-level ones. There are two main different reasons behind this fact: (1) the secondary effects due to the heater-superconducting active layer interaction (e.g. widened I_{SW} distributions, bubble formation in dewar environment etc.), and (2) co-satisfaction requirements of a higher number of (in)equalities. The work done in this thesis does not involve the modeling of these heater devices. Therefore, the physical interactions are still assumed to be ideal, which should be addressed in future studies. Instead, this work covers the maximization of individual nMEM margins such that they can cope with the array requirements.

The operation of an nMEM cell can be described by 7 parameters : L_1, L_2, $I_{\mathrm{SW},1}$, $I_{\mathrm{SW},2}$, $I_{\mathrm{prog},'1'}$, $I_{\mathrm{prog},'0'}$, and I_{read}. For simplicity, we can assume a base inductance (L_{base}) and switching current ($I_{\mathrm{SW,base}}$) and define the counterpart using this base units and two ratios (L_{ratio}, $I_{\mathrm{SW,ratio}}$). To quantify the operational margins, we have defined a performance metric for the nMEM cell. This value is given by "the difference between the currents, that cause a voltage read when the state is '1' and '0'

respectively"[5]. The easiest way to measure these values is to change the shape of the read pulse into a ramp. If the amplitude of this ramp is set high enough, it eventually switches both of the constrictions. Then, measuring the points at which these events happen, indirectly measures the respective current values. We finally represent the difference of these values as a percentile of I_{base}.

There are many other choices, that could have been selected as the performance metric. For example, if we were to be limited by the fabrication yield, tolerance to circuit parameter variation would have been a better choice. What we observed instead is that the devices were suffering from noise and wide switching current distributions. Therefore, we selected the aforementioned metric such that the output is the same for a wider range of effective[6] input conditions. Finally, although we specify it in the readout cycle, it can also be referred to the programming cycle.

3.1.4 Optimization of Superconducting Nanowire-Based Memory

In order to optimize the nMEM design for the performance metric defined in the former section, we have built a closed-loop system around our simulator. We have used the classical solver described in the Sec.2.2.3, as the phase evolution across the device does not make an operational difference (i.e. circuit involves nanowires that do not conserve phase coherence during operation). As explained in Sec.2.2.3, quantization of flux in the superconducting loop is enforced, when the nodal and branch quantities reach steady state (following their computation in the absence of such a constraint).

The four parameters to be optimized are L_{ratio}, $I_{\text{SW,ratio}}$, $I_{\text{prog,'1'}}$, and $I_{\text{prog,'0'}}$,. Base values for L_1 and $I_{\text{SW,1}}$ are selected to be as 100 pH and 20 μA respectively. We have bounded the state-space by constraining L_{ratio} and $I_{\text{SW,ratio}}$ between [1.1, 4]. For the programming levels we took a slightly different approach and defined a changing base

[5]Note that even the state of the device is '0', there exists a reading current to switch the entire device to cause a voltage across. In functional devices, this current should be (much) higher than that of the '1' state. Therefore, we select their difference as the performance metric.

[6]Taking into account of probabilistic events at the time of programming and readout.

programming current as:

$$I_{\text{prog,base}} = I_{\text{SW,1}} \frac{L_2}{L_1 + L_2}, \tag{3.9}$$

such that $I_{\text{prog,base}}$ is the smallest current that switches the NW_1 under the absence of any circulating current. Following this definition we have defined $I_{\text{prog,'1'}}$, and $I_{\text{prog,'0'}}$, as:

$$I_{\text{prog,'1'}} = I_{\text{prog,base}} \kappa_1; \tag{3.10}$$

$$I_{\text{prog,'0'}} = I_{\text{prog,base}} \kappa_2. \tag{3.11}$$

As explained in the former section, the performance metric is obtained by applying a ramping input as the read input. Then, the point that a voltage difference is observed across the nMEM terminals is registered. This point in time is then referred back to a current value. Finally, the difference of these two current values (while the cell is in the '1' state and the '0' state) is normalized to $I_{\text{SW,1}}$, to produce the performance metric.

After the definitions of the cost function (inverse of performance metric) and the space boundaries, the optimization can be started. For this purpose, we have used two well-known optimization techniques: (1) a simplex algorithm (also known as fmincon as a built-in MATLAB function) and (2) a genetic algorithm. We have observed that the genetic algorithm was not efficient in converging, therefore the optimizations were done using the Nelder-Mead simplex algorithm [26].

Fig.3-3 shows the input-output characteristics of two nMEM designs, to extract the performance metric. The points at which the voltage state initiates are shown with blue and red markers on the voltage plots. The current equivalent of these points is shown with the dotted lines in the current plots. The first design, which is also referred as the 'control design', has $I_{\text{SW,ratio}}$ of 1.4 and L_{ratio} of 1.7. It has a reasonably good performance metric of 61.05%. On the other hand, the optimized design has $I_{\text{SW,ratio}}$ of 2 and L_{ratio} of 3.1. This design has a performance metric of 100.1% which is a significant improvement over the former design.

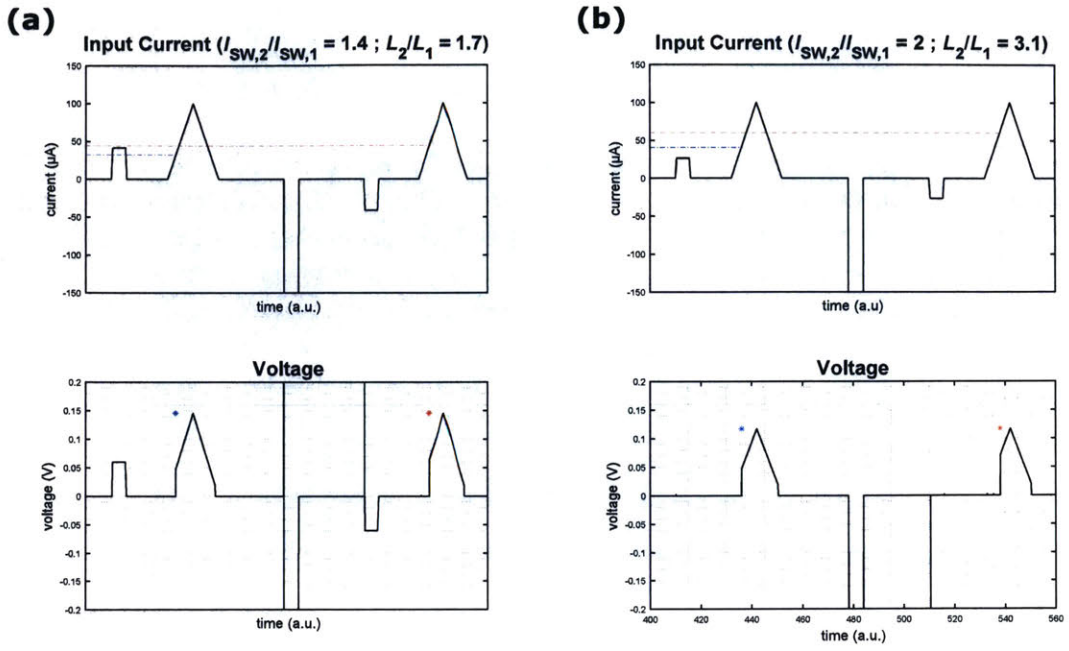

Figure 3-3: Simulation of two nMEM results for **(a)**unoptimized,and **(b)** optimized layout considerations. Red and blue markers in the voltage plots show the time instant that the voltage state is observed. Dotted lines of the same color in the current plots indicate the input current at these time instances. Performance metric for a given nMEM cell is represented by the distance between these red and blue dashed lines. It can be seen that the optimized nMEM design indeed shows superior characteristics over the design that was in use before.

To further visualize the dynamics of the performance metric as a function of the optimization variables, we have also done a coarse scan of the state-space. Results for this sweep is shown in Fig.3-4. To generate this colormap, all 4 variables were cross-swept against one another. Then, the maximum performances for the different κ_1 and κ_2 values were selected for visualization. This translates as plotting the circuit performance under the operation with the best (within the sample set) programming input pair.

It can be seen from Fig.3-4 that the performance metric follows some intrinsic curvatures as a function of $I_{\mathrm{SW,ratio}}$ and L_{ratio}. However, it was observed that for L_{ratio} values above 2.2 these patterns are replaced by random-looking features. We have three main hypotheses to explain this behavior. First of all, it can be an ar-

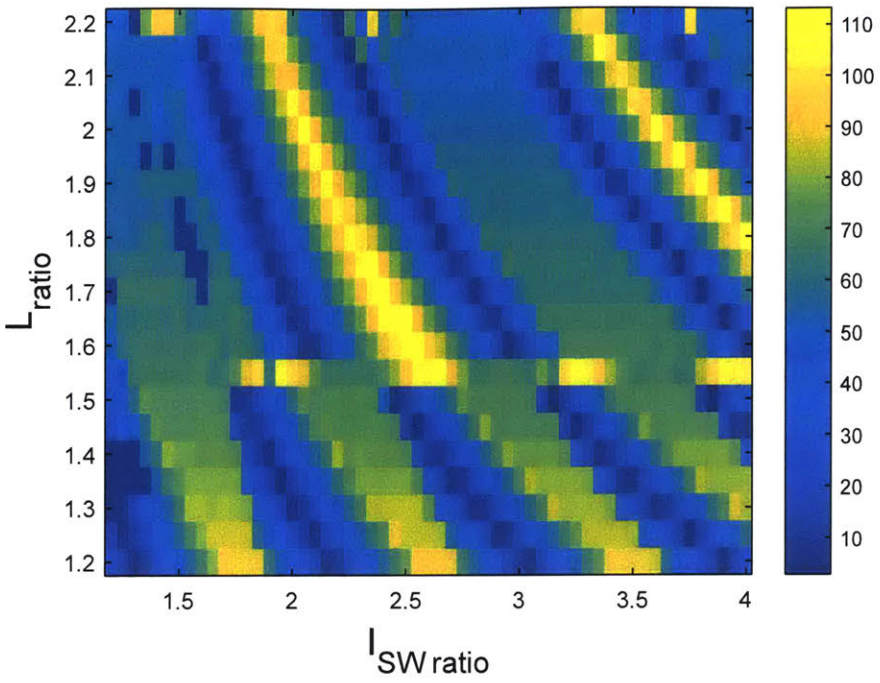

Figure 3-4: Colormap representation of the nMEM performance metric with respect to the cross-swept $I_{\mathrm{SW,ratio}}$ and L_{ratio}. The complex features seen in this plot was observed to disappear for higher L_{ratio} values.

tifact of the simulation algorithm. However, when conducted with finer tolerance values for convergence, results did not change. Second, the ideal programming pulses might not be included in the space we are looking at. To test this idea, we have increased the bounds of the programming input but failed to see any noticeable difference. These results have led us to the third option that despite looking unlikely, those patterns are indeed real. As a matter of fact, the optimal design we are going to verify experimentally in Sec.3.1.6, lies in this (random looking, scattered) region as well. Considering that we have observed the existence of the 'unconventional operation modes' (See Sec.3.1.2), a 'discontinuous' equi-performance planes is a plausible hypothesis. However, we must note that this discussion is still far from conclusion.

Finally, Fig.3-4 shows that our optimizer has missed even better design alternatives. This result is expected considering that the cross-sweep is essentially a 'brute-force-optimizer', that searches the entire state-space for a given set of bounds. However, the missed optimality is absolutely affordable (113.3% with respect to 100.1%),

particularly when the processing time is considered as well.

3.1.5 Fabrication of Superconducting Nanowire-Based Memory

In order to validate the optimization results for the nMEM, devices with various parameters are fabricated (including the optimal ones). The fabrication process is as follows:

- 100 nm Si_3N_4 on Si substrate (laser diced, 1 cm×1 cm, 0.525 mmthick, vendor and Si_3N_4 deposition method unknown) was cleaned via sonication in acetone, methanol and isopropanol (IPA) for 300 s each.

- ≈20 nm of NbN (93 Ω/□, measured by four-point probe, thickness adjusted by sputtering time) was deposited on the chip at room temperature, using reactive DC magnetron sputtering with AJA ATC Orion sputtering system [12].

- Poly-methyl-methyl-acrylate (PMMA, e-beam resist) was spun at 3krpm (diluted to give ≈120 nm, measured by reflectometry) for 60 s and was baked at 180 °C hotplate for 120 s.

- E-beam resist was exposed with 2500 μC/cm^2 dose using 2 nA beam current using Elionix FLS-125 e-beam writer.

- PMMA was developed in 3:1 isopropanol (IPA):methyl-isobutyl-ketone (MIBK) at 0 °C for 90 s followed by N_2 gun dry.

- NbN was etched with reactive ion etching (PlasmaTherm 790) using CF_4 at 10 mTorr, 50 W, for a total period of 360 s partitioned (to avoid burning/reflowing the resist) in 3 steps of 120 s each. Chamber was not opened between successive etches.

- Excess resist was stripped in n-methyl-2-pyrrolidone (NMP) at 70 °C for 1 h.

The micrograph of the resulting devices can be seen in Fig.3-5. The device on the left is the 'control' cell while the one shown on the right is the optimized one. It should be noted that these cells could have been fabricated at much smaller scales (e.g. constriction width at 60 nm). This design choice would have required a different e-beam resist (such as ZEP520A, HSQ, GL2000M or CSAR62) since PMMA is known for having low resolution and high edge-roughness issues. Since these devices were only for validating the simulation performance, such fabrication optimization was not pursued here.

For future references, if one wants to build smaller versions of these cells, we encourage them to use ZEP520A (as it is also a positive tone) or another alternative. A similar fabrication procedure that involves ZEP520A is given in Sec.3.2.4. This resist choice will provide superior resolution and higher (roughly twice) etch resistance for the aforementioned CF_4 reactive ion etching process with respect to that of PMMA. Finally, various other problems such as resist re-flow during etching and residue left after the stripping procedure could also be solved by using ZEP520A (See Sec.3.2.4).

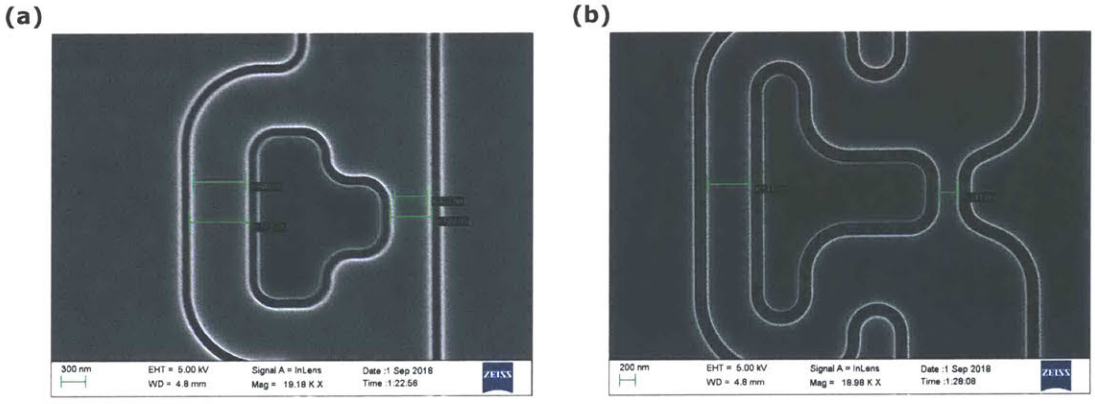

Figure 3-5: Micrographs of two nMEM results for **(a)** unoptimized (control) and **(b)** optimized layout considerations.

3.1.6 Experimental Characterization of Superconducting Nanowire-Based Memory

The experimental characterization of the nMEM devices are made under liquid helium immersion conditions using LeCroy Waverunner 620Zi 2 GHz oscilloscope, and Agilent 33600A Trueform Series arbitrary waveform generator (AWG). The waveforms were programmed using MATLAB, which controlled the AWG and also acquired the output waveform from the oscilloscope[7].

The input waveforms used in the experimental characterization are of the same nature as the ones used for the simulation (see Fig.3-3). First, a programming input ('1' or '0') pulse of 100 µs is applied to the device. Then, a ramping input is applied to read the cell, and the voltage difference across the device is measured. Internal functions of the oscilloscope are used to register the point at which the device switches.

In order to operate the devices under the best input configuration, amplitudes of the programming inputs are optimized using a closed-loop control using MATLAB [7]. This code reads the measured oscilloscope output and evaluates a cost function. Then, it communicates with the AWG again, to modify the programming input to reduce the cost in a recursive way.

Fig. 3-6 shows the experimental results obtained for (a) the unoptimized and (b) the optimized devices. The results are in the form of a histogram, where the x-axis is the triggering time (which directly maps to a read input as explained in Sec.3.1.4) and the y-axis the number of occurrences. In terms of the performance metric defined in Sec.3.1.4, a larger separation means superior performance, as the nMEM is able to produce the correct output for a wider range of input levels. It can be seen that the separation of distributions improved significantly with the circuit designed using simulator's optimization output.

This result is an important step towards achieving the goal of this work, as we intend to provide a high-performance design tool for superconductor-based nanoelectronics. The fact that the software can be used to predict characteristics can be used

[7]The drivers to control these instruments, and the setup optimization codes are written by Brenden A. Butters. The author wants to acknowledge his work and thank him for his contributions.

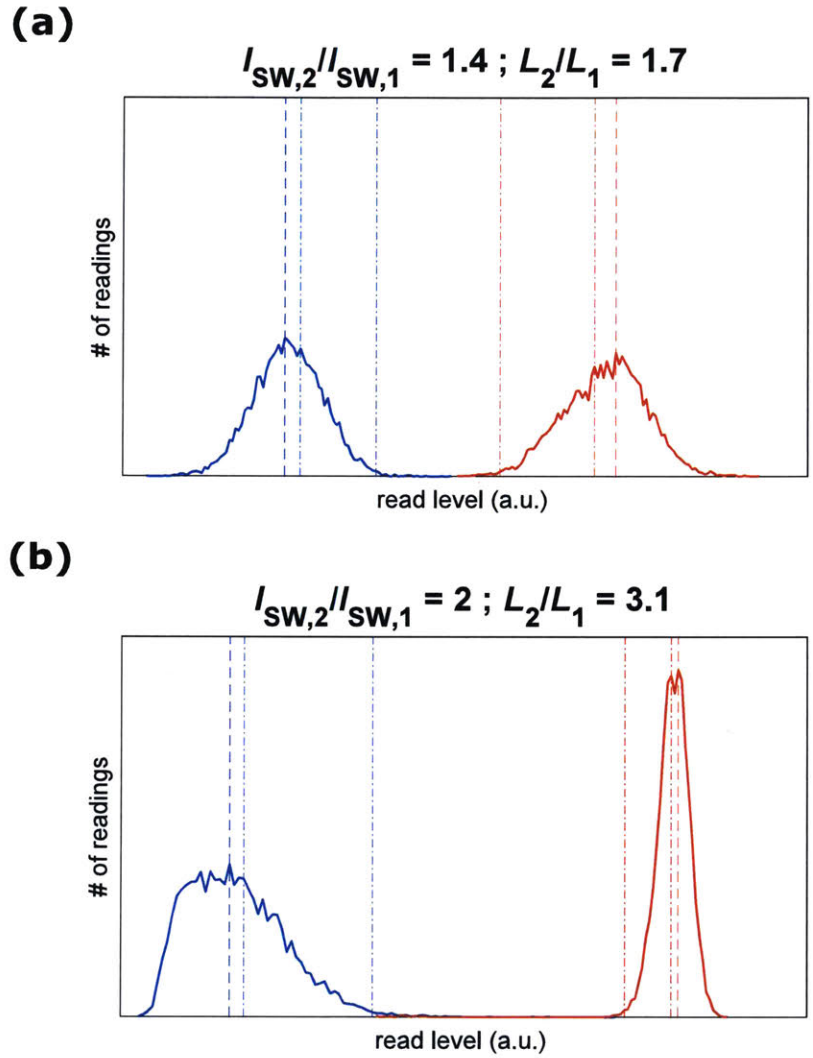

Figure 3-6: Experimental characterization of two nMEM results for **(a)** unoptimized and **(b)** optimized layout considerations. It can be seen that the separation characteristics of the optimized device is superior to its counterpart.

to reduce the chip cost, by reducing the number of fabrication runs, ideally to a single one.

3.2 Multi-level Flux Shuttling with Shunted Nanowires

The behavior of the superconducting nanowires (See Sec.2.1.2) differ greatly from their counterparts, the Josephson junctions (JJs, Sec.2.1.1). Most notably, the switching of the nanowires is mediated thermally, through the formation of a hot-spot. Therefore, the nanowire devices cannot support coherent quantum transport. However, their high-impedance resistive state allows them to achieve fanout numbers well-beyond than that of JJs. Since they can also be triggered by single flux quantum (SFQ) events, nanowire-based devices have been used in interfacing circuits between JJ systems and CMOS based environments [48].

Recently, nanowire-based electronics is extending beyond this complementary role. As shown in Chap.3.1, nanowire memories (nMEMs) already have promising characteristics to become a significant competitor for memory applications. Furthermore, small-scale digital designs, such as half-adders are presented, suggesting that logic families with these devices can be realized in larger scales [48]. One of the main design limitations for these devices come from the abruptness of the switching characteristics. Instead of a gradual change (like a transistor), switching of a nanowire causes a rapid burst of current out of the device.

The manifestation of this behavior can be observed in the nMEM devices. When the programming current switches the nanowire, the current on that branch reduces all the way down to the retrapping current I_R (See Sec.3.1). For a typical constriction with I_{SW} of $100\,\mu A$ ($I_R \approx 0.2 I_{SW} = 20\,\mu A$) and total loop inductance L_{loop} of $1\,nH$ ($I_o = \frac{\Phi_o}{L_{loop}} \approx 2\,\mu A$)[8], each switching event approximately shuttles $40\Phi_0$ into the loop. It is important to note that this value is practically the full capacity of the cell. This behavior limits the cell to binary programming as no intermediate response can be obtained from the nanowire.

Recently, it has been demonstrated that these characteristics can be controlled

[8] Although we use a notation that might indicate it is a quantum entity, this variable is certainly not of a separate quantum nature. It is only a result of quantization of flux, which is a quantum event. Furthermore, unlike Φ_o which is a universal quantity, I_o changes from device to device depending on the loop inductance. The purpose of this definition is to provide ease of representation in our equation set.

by lithographically shunting the nanowire with a resistor [43]. This section first explains the utilization of the simulator built in this work, to understand the operational dynamics of this modified system. Validation of the simulator is performed by comparing it with existing experimental data. Then, the required circuit parameters to obtain single flux quantum level control with these devices is presented. Finally, a modified method is discussed to improve the shunting characteristics of these devices.

3.2.1 Fundamental Operation of Shunted Nanowires

When the current across the nanowire increases beyond its switching current, I_{SW}, it causes the device to switch into a resistive state. The dynamics of this switching event are governed by the electrothermal properties of the hot-spot (See Sec.2.1.2 for details). In summary, the nanowire remains in this resistive state until the current drops down below γI_R, where I_R is the retrapping current and γ is a proportionality constant slightly lower than unity. This behavior is the same for both the shunted and the unshunted cases.

On the other hand, where the excess current gets redirected to, and how fast this occurs differs greatly between the shunted and unshunted nanowires. Consider a loop structure with a single constriction (i.e. Fig.3-1 with practically infinite $I_{SW,2}$). When the nanowire is not shunted (we represent this case with $R_{shunt} = 1\,M\Omega$ for simulation purposes), as there is no other place for it to go, all the excess current floods into the loop. On the other hand, when the nanowire is lithographically shunted (e.g. with $R_{shunt} = 5\,\Omega$[9]), the shunt provides another path for the current to follow. Therefore, the current on the nanowire branch can drop down to the value at which it heals (γI_R), without the entire difference ($I_{SW} - \gamma I_R$) shuttling into the loop. This event can be written in terms as a current division as following:

$$\Delta I_{circ} = (I_{IN} - (I_{SW} - \gamma I_R))\frac{Z_s}{Z_s + Z'_{loop}}, \tag{3.12}$$

where Z'_{loop} and Z_s denote the impedances of the loop and the shunt branch, seen from

[9]The shunt inductance, L_{shunt}, used in these simulations are 200 pH. The value of the L_{shunt} plays an important role in the shunting dynamics and will be covered in detail later.

the nanowire branch respectively. The 'prime' factor in the Z'_{loop} stands for denoting the impedance of the loop, while the nanowire is in resistive state. As previously explained in Sec.2.1.2, the change in the (kinetic) inductance of the nanowire, and the change in the superconducting path (due to the normal region formation) are the reasons for this differentiation.

As can be seen from Eq.3.12, the state of the cell (following the same notation used in Sec.3.1), can now be a fraction of what it was in the nMEM case (Eq.3.4). Note that we have also changed the notation form from I_{circ} to ΔI_{circ} between Eq.3.12 and Eq.3.4 to emphasize that the circulating current does not have to be set to its maximum level but can be tuned gradually. It is also important to underline that these ΔI_{circ} can only occur at quantized levels due to the quantization of flux in superconducting loops (See Chap.2). This flux quantization can be referred to a current equivalent as follows:

$$I_{\circ} = \frac{\Phi_0}{L_{\text{loop}}}, \tag{3.13}$$

where the L_{loop} is the total inductance of the superconducting loop[8]. Any circulating current value that is not an integer multiple of the $I_{\circ}$ is rounded to the nearest energetically favorable state. This analysis clearly shows that the operation dynamics of the shunted nanowires are purely classical (i.e. not quantum mechanically controlled), except for the quantization of flux. This information is important as we can potentially achieve SFQ level control, like a JJ, with a non-phase-coherent device. Therefore, shunted nanowires can alternatively be presented as an intermediate device between the nanowires and JJs as well.

The simulator constructed in this work is used to understand the behavior explained above. We have once more used the classical solver as the evolution of phase can be neglected for the shunted nanowire operation. Fig.3-7 shows the behaviors of the shunted and unshunted nanowires during a simulated switching event. Both cases start with $I_{\text{NW}}(t) = I_{\text{SW}}$ (Fig.3-7a) which switches the nanowires. This switching causes a rapid increase in R_{NW} (Fig.3-7d) and a rapid decrease in I_{NW} (Fig.3-7a)

as the current is shuttled away from the nanowire. This excess current mainly flows into the shunt branch in the shunted case (Fig.3-7b), while it completely flows into the loop in the unshunted one (Fig.3-7c).

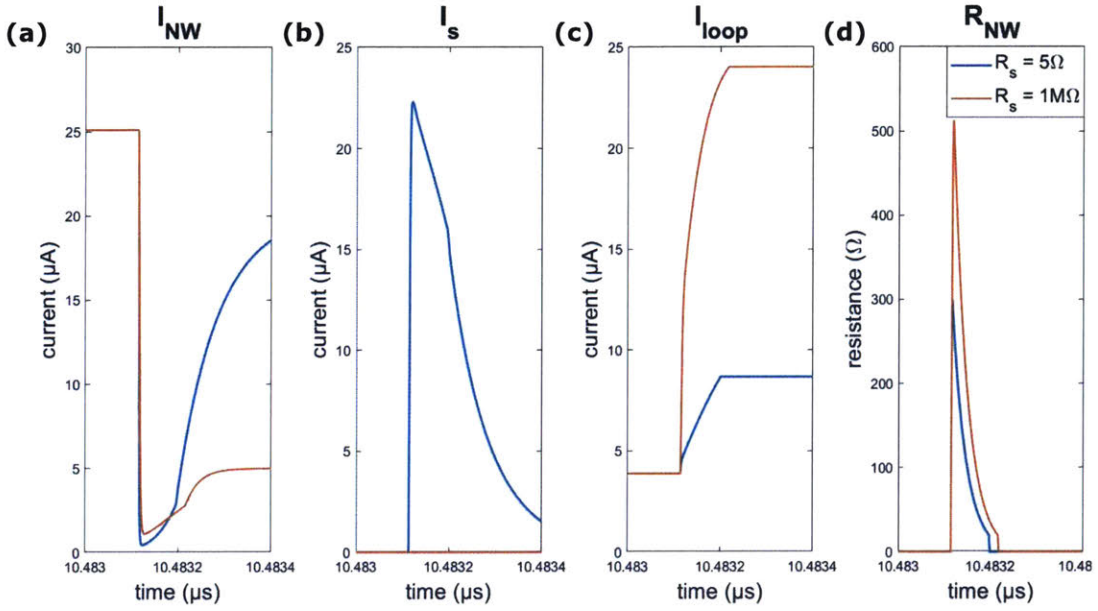

Figure 3-7: Current levels in **(a)** nanowire, **(b)** shunt branch, **(c)** loop inductor, and **(d)** the resistance value of the hot-spot over a switching event for shunted (blue), and unshunted (red) nanowires.

The point at which the final I_{loop}[10] is determined is also different for these two cases. For the unshunted nanowire, current in the loop starts rising with the switching event and settles when the nanowire heals back. On the other hand, when the shunted nanowire switches, almost no current is shuttled into the loop at first. This result is expected as the shunt branch impedance Z_{s} is much smaller than the loop Z_{loop}, seen from the nanowire. Then, the absence of any current allows the nanowire to heal back to its superconducting state. Unlike the switching event, healing takes longer time, meaning that the R_{NW} drops gradually. As can be seen from Fig.3-7, the I_{loop} is defined in this cooling period.

A careful investigation leads to the observation of two more details which will later

[10]Here we are using I_{loop} as a replacement for the regular state definition I_{circ}. Once the bias current is removed, I_{loop} goes to I_{circ}.

become important in the optimization of these systems. Fig.3-7 shows that I_{NW} drops further down for the shunted nanowire with respect to its unshunted counterpart. This difference arises from the competition of the electrical and the thermal time constants of the events. For the unshunted device, the loop impedance is large enough, such that the rate at which the excess current can be expelled out of the nanowire is limited by the electrical time constant. In other words, if the current (somehow) left the branch faster, thermally, nanowire could have healed faster as well. On the other hand, a shunted nanowire can shuttle this current at a much faster rate to the low impedance shunt branch. However, thermally, it cannot cool-down this fast. Therefore, the process is limited by the thermal time constants, it can be observed as an undershoot of I_{NW} (Fig.3-7a).

The second result of the same argument is the lower maximum R_{NW} for the shunted nanowire. This analysis tells that the shunt branch essentially reduces the 'fastest-possible healing time' from the $\tau_{electrical}$ dominated to $\tau_{thermal}$ dominated. It will be of further interest in Sec.3.2.3, where the shunted nanowires are optimized to provide superior shunting characteristics.

Following the explanation of the operation dynamics for the shunted nanowires, we return to their applications. As stated earlier, shunted nanowires can be used to shuttle current in a controlled manner. This behavior can be used to realize multi-level memory devices in a loop configuration, that is similar to the nMEM devices.

To demonstrate this behavior we simulate a superconducting loop with a single shunted constriction in our software. It is important to emphasize once more that we use the classical solver described in Sec.2.2.3, as the circuit behavior can be explained classically. Fig.3-8 shows multi-level programming of this device. In this simulation, we select the input to be a current ramp which is equivalent to a set of pulses with increasing amplitudes. The input applied in Fig.3-8a causes the shunted nanowire to switch 3 times (Fig.3-8d) in this simulation.

Fig.3-8c shows that each switching increases the circulating current, I_{circ}, by $5I_o$. It is important to note here that the increase in the circulating current is not of incremental nature. Instead, each switching event sets the loop to a certain state,

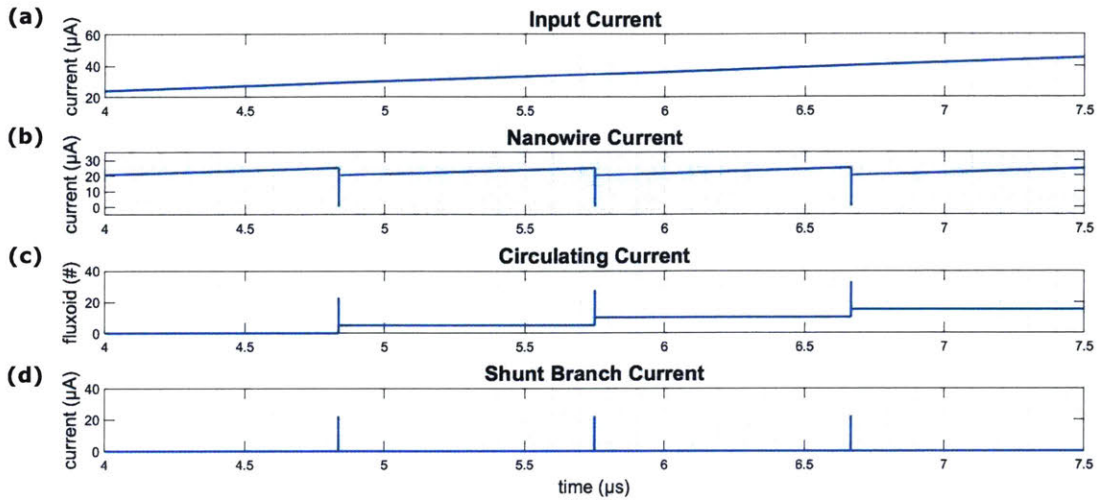

Figure 3-8: **(a)** Input, **(b)** nanowire, **(c)** circulating and **(d)** shunt branch currents for a shunted nanowire based flux shuttling. The input is represented with a ramping current which can be interpreted as pulses of increasing heights. During the simulation window, the shunted nanowire switches 3 times. Each switching sets the circulating current in the loop to a new state. The states are $5I_\circ$ separated from each other, which is defined by its circuit parameters R_s, L_s, L_{loop} and I_{SW}.

which are separated from each other by $5I_\circ$ due to the relations given in Eq.3.12 and Eq.3.13. Since the circulating effectively modifies the effective switching current as Eq.3.5, the switching events also occur at input levels of $5I_\circ$ difference in between.

The number of states that can be resolved using a superconducting loop with a shunted constriction can be calculated as follows:

$$\#\text{ of levels} = \frac{2I_{SW}}{\Delta I_{circ}^{\text{Eq.3.12}}}, \tag{3.14}$$

where $2I_{SW}$ is the full range of the shunted nanowire, $(-I_{SW}, I_{SW})$, and the $\Delta I_{circ}^{\text{Eq.3.12}}$ is the amount of change in the circulating current between adjacent states as defined in Eq.3.12. As stated before in Eq.3.13, this ΔI_{circ} term is quantized by means of $I_\circ$.

3.2.2 Simulator Validation with Prior Experimental Results

In order to validate the simulator's capability of describing shunted constriction behavior, we tried replicating the experimental results with our software. Fig.3-9 shows

the experimental characterization of a shunted nanowire that has been done in our group[11].

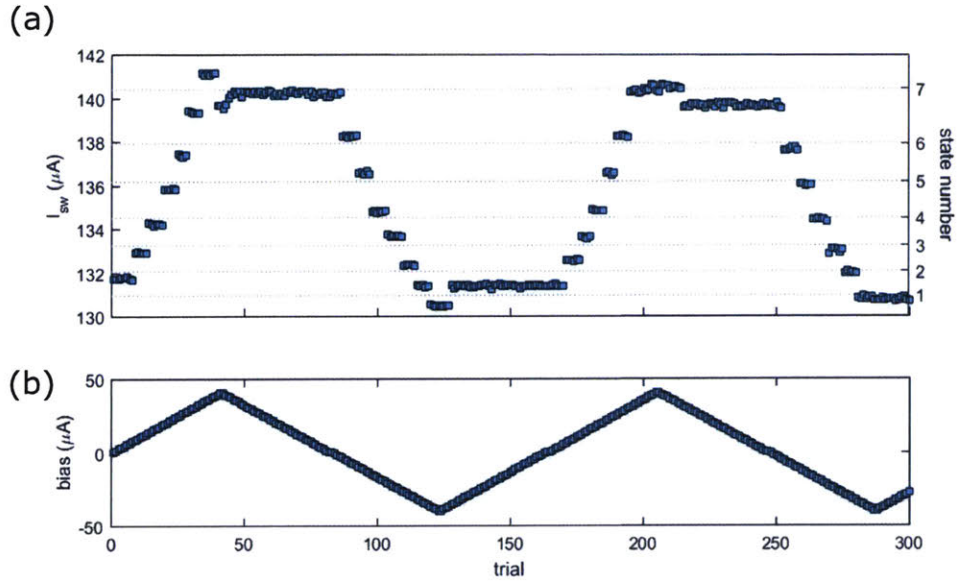

Figure 3-9: Prior experimental characterization for multi-level addressing with shunted nanowire gating. Figure taken from Ref.[43]

The devices that have been used in Fig.3-9 consisted of a superconducting loop what has a shunted constriction as a writing element and a yTron as the reading element. Operational dynamics of yTron will be covered in detail in Chap.4. For the purposes of this section it can be defined as a device that can sense the state of the cell, I_{SW}), without perturbing it (i.e. non-destructive readout element[12]). The experiment is done by sending a set of pulses of increasing amplitude, which look like a ramping Fig.3-9b. The device used in Ref.[43] is characterized to have parameters shown in Table 3.2.2.

It has been observed that Device 1 and 2 (Table 3.2.2) had $\approx 5I_{\circ}$ between its

[11]Author expresses his thanks to Emily Toomey and acknowledges her work in fabrication and experimental characterization of the devices.

[12]Current at the read arm of yTron is ramped up to the point it switches, which is determined by the amount of current flow inside the loop that was previously set by the shunted constriction and programming pulse. Therefore, the output of the yTron readout is the time (skew) it took for yTron to switch while it is ramping up. Current equivalents of these skew values are not reported in the figure.

Parameter	Device 1	Device 2	Device 3	Device 4
$I_{SW,nanowire}$	20 µA	20 µA	20 µA	20 µA
$I_{SW,yTron}$	145 µA	145 µA	145 µA	145 µA
L_1	284 pH	284 pH	284 pH	284 pH
L_2	1.87 nH	1.87 nH	0.66 nH	1.87 nH
R_s	5 Ω	7.8 Ω	7.8 Ω	Unshunted
L_s	50 pH	50 pH	50 pH	Unshunted

Table 3.1: Circuit parameters of experimentally investigated devices in Ref.[43].

programmable states while Device 3 had $\approx 8I_o$. On the other hand Device 4 showed binary state characteristics. From the number of states point of view, this maps to 7, 5, 3, and 2 states for Devices 1, 2, 3, and 4 respectively.

The simulator was able to capture these parameters using the given circuit parameters and the following physical parameters:

Parameter	d	R_{sheet}	J_c	h_c	k	c
Value	20 nm	150 Ω/□	24 GA/cm^2	50 kW/m^{2}K	108 mW/mK	4.4 kJ/m^{3}K

Table 3.2: Physical parameters used for the simulation of the shunted nanowires.

Simulation results shown in Fig.3-8 is obtained by using the circuit parameters of Device 1. The same device is then simulated in an input scheme that was used in Fig.3-9b. Results for this simulation is given in Fig.3-10. First of all, it can be seen that the states are separated with $5I_o$ (4.8 µA) leading to a resolution of 8 states. As the simulator uses a perfect device model, free of noise and fluctuations, the state separation is perfectly $5I_o$ whereas the experimental characterization gives a mean separation of $4.77I_o$ with a standard deviation of $1.23I_o$. This variation is the source of the discrepancy between the experimental data and the simulation results for the number of states (8 states instead of 7).

Secondly, the simulator managed to capture the cell behavior, when the state of the device is approaching its maximum limit. It has formerly been observed that when the $|I_{circ}|$ approaches to the I_{SW}, pulses of even higher amplitudes cause intermediate states (still an integer multiple of I_o but not the nI_o that is characteristically seen by

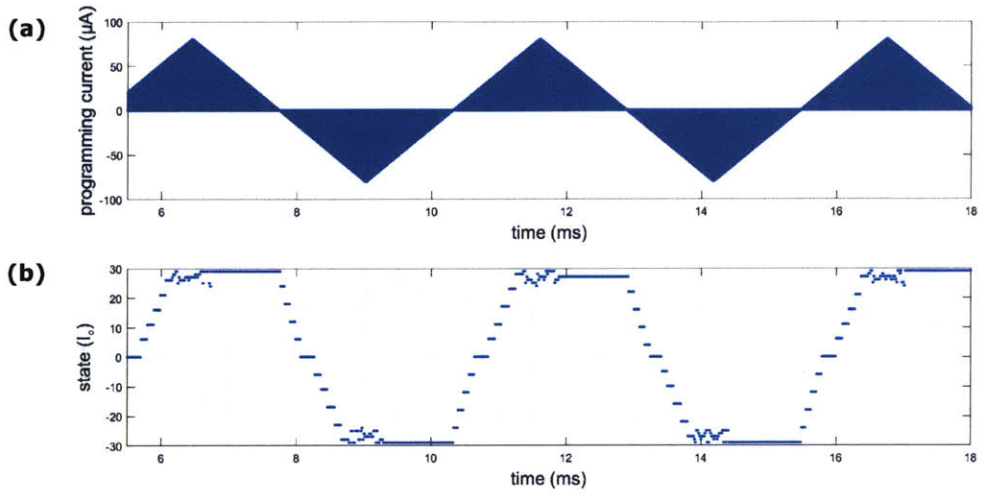

Figure 3-10: Simulation results for multi-level addressing with shunted nanowire gating. **(a)** Input pulse stream is chosen to be similar to the experimental one shown in Fig.3-9. **(b)** Device shows 11 states (instead of 7) due to the higher switching current used in the simulator (28 μA instead of 20 μA). Instability of the state close to the maxima is captured by the simulation.

the cell). This behavior can be seen in the experimental characterization shown in Fig.3-9 and the simulated version shown in Fig.3-10.

3.2.3 Optimization of Shunted Nanowires

Following the validation of the simulator, the software is again used to optimize the performance of the shunted nanowires. The performance metric we choose for these devices is the separation between the programmable states, where the ideal case is $I_\circ$ (equivalent to $\frac{\Phi_\circ}{L_{\text{loop}}}$, which we will refer as single flux quantum (SFQ) level control). In a brute-force searching scheme, we have cross-swept $R_s - L_s$ and $R_s - L_2$ where R_s, L_s are the shunt branch resistance and inductance and L_2 is the inductance of the loop (excluding the nanowire inductance, See Fig.3-1 for reference). Fig.3-11 shows the results obtained in these sweeps with the colormap showing the separation of states in terms of $I_\circ$.

The main result of this sweep is: finer control of current shuttling can be managed by lowering the shunt impedance. It can be seen in Fig.3-11a that SFQ level control

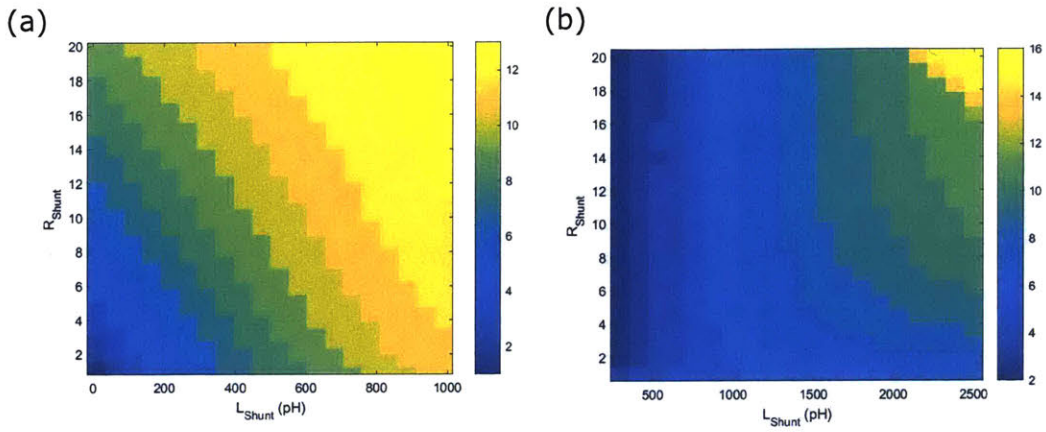

Figure 3-11: Cross-sweeps for flux-per-event for **(a)** shunt resistor and shunt inductor, and **(b)** shunt resistor, loop inductor.

is possible for $R_s = 1\,\Omega$ and $L_s = 50\,\text{pH}$. Moreover, once the shunting is powerful enough, the effect of the loop inductance gets minimized as well (Fig.3-11b). Furthermore, a simplified version of this sweep following the circuit analysis covered in this section can be used to capture the essential behavior of the system[13]. The result of this fitting is shown in Fig.3-12. The main source of the difference between the actual sweep and the simplified fit is the time constants of events, where the simplified fit assumes a certain time constant to produce the impedance of the branch. However, as the system evolves, the nonlinear resistance and inductance of the nanowire lead to varying time-constants.

Furthermore, it has been observed that reducing the thermal time constants resulted with superior shunting characteristics. The initial observations that have led to this conclusion are explained in Sec.3.2.1.

Fig.3-13 shows that even for a shunt resistance as high as $50\,\Omega$, with higher ($100\times$) thermal sinking of the active layer can provide SFQ level control. At cryogenic temperatures, thermal conductivity follows electrical conductivity (due to the absence of crystal motion caused phononic conductivity). For the conventional substrates, such as Si, Si_2, Si_3N_4, Al_2O_3[14] etc. thermal conductivity is very low, as all of them

[13]Author wants to thank Emily Toomey once more here for her contribution in generating the simplified fit, according to the model we have developed together.

[14]Thermal conductivity of Al_2O_3 is considerably high at room temperature due to the stiff crystal structure and high speed of sound of the material. However, this term becomes negligible when the

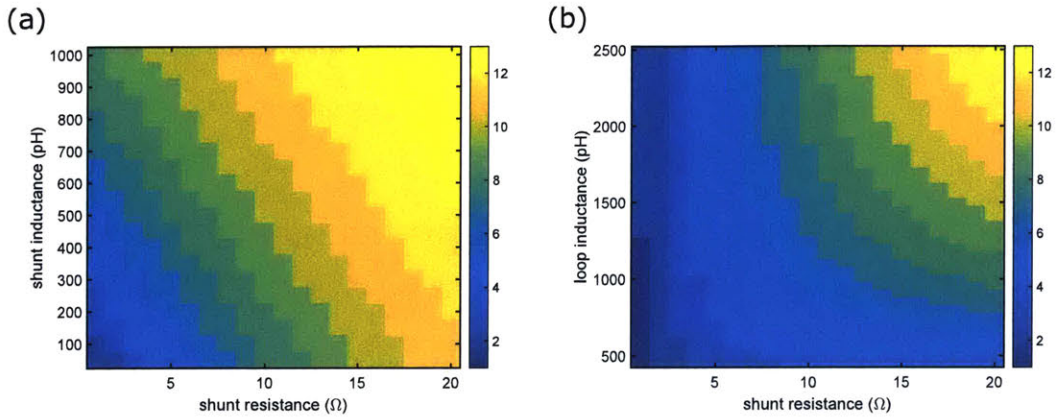

Figure 3-12: Simplified fit for flux-per-event for **(a)** shunt resistor and shunt inductor, and **(b)** shunt resistor loop inductor.

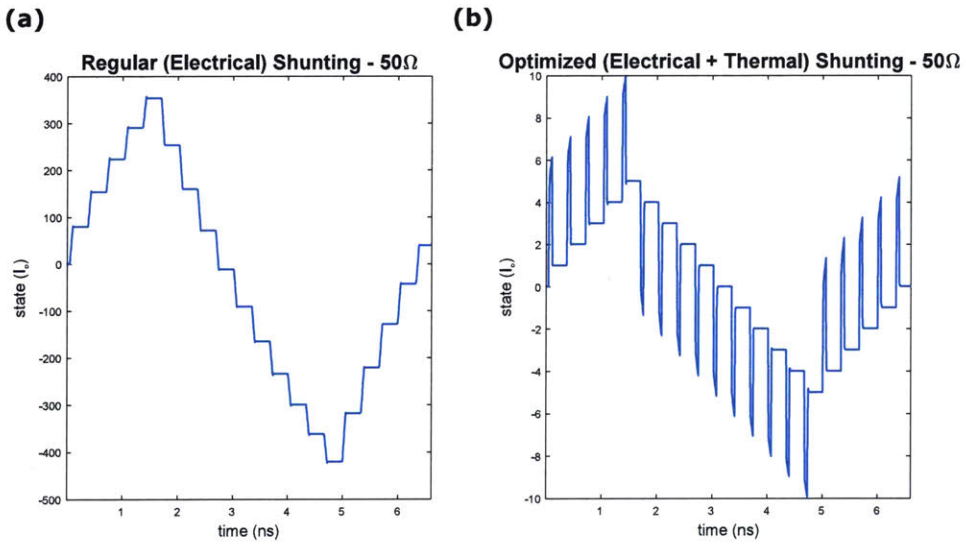

Figure 3-13: Simulation results for two different shunting conditions. **(a)** Regular electrical shunting with a relatively large shunt resistor ($R_s = 50\,\Omega$). **(b)** Optimized, electrical and thermal, shunting with the same shunt resistor but 100 times higher thermal conductivity. Optimized shunting method can achieve SFQ level control even with $R_s = 50\,\Omega$.

are electrical insulators. However, the overall shunting design can be modified by placing the metal layer (shunt resistor) directly beneath the constriction. This method provides three significant advantages at the same time.

1. The thermal sinking increases the hot-spot event speeds by orders of magnitude,

material is cooled down to cryogenic temperatures since the lattice movements disappear.

allowing shorter time-windows for current shuttling (See Sec.3.2.1).

2. Direct contacting of the metal layer practically removes the shunt inductance, which increases the effectiveness of the shunt branch as explained before (See Fig.3-11).

3. The shunt resistance (calculated by $(R_{\text{sheet}}) \times (\#\text{of squares})$) is reduced by reducing the number of squares (seen by the hot-spot) significantly.

It must be noted that this modified shunting method brings some other design concerns. First of all, the direct contact of metal can proximitize the superconductor, leading to suppression of its superconductivity. Secondly, when the current is passing through the shunting branch it can lead to heating of the constriction which can again cause suppression of the active layer. Next section discusses the material choices and thicknesses engineered to overcome these problems and obtain optimized characteristics from the shunted nanowires.

3.2.4 Fabrication of Optimized Shunted Nanowires

In order to obtain the best characteristics from the in-contact-shunting, a variety of material stacks have been tested. Table 3.2.4 shows these stacks with details of the deposition and patterning methods.

Table 3.2.4 can be divided into three main categories: (1) Ti-based devices, (2) Al-based devices, and (3) Lift-off based devices. The titanium-based devices were initially selected due to its availability in our sputtering chamber, and close to unity residual resistance ratio. However, we immediately realized that Ti cannot be the top layer due to the very fast oxidation process. This problem can be mitigated by placing the Ti layer beneath the NbN layer (sputtering them one after another without disturbing the vacuum conditions inside the chamber). As can be expected, this method similarly leads to oxidation of the top NbN layer (instead of the Ti layer as before). However, oxidation of NbN forms a thin stopping layer which can be tolerated since the active layer of these devices are $\approx 25\,\text{nm}$ thick. Unfortunately, etching

Stack	Deposition	Patterning	Comments
	(1) Sputtering	(1) RIE	
	(2) Evaporation	(2) Ar Milling	
		(3) Liftoff	
		(4) Not removed	
NbN + Ti	(1) + (1)	(1) + (1)	Ti oxidizes
Ti + NbN	(1) + (1)	(1) + (1)	Requires thick resist
Ti + NbN	(1) + (1)	(1) + (4)	yTron is shunted
NbN + Al	(1) + (1)	(2) + (1)	Milling melts Al
Ti + Pt + NbN	(1) + (2 + 2)	(3 + 3) + (1)	Works well

Table 3.3: Material stacks that have been investigated for realizing optimized shunted nanowire characteristics. Note that the orders of deposition and patterning methods are written as the order of application, not the order of the material stack listed in column 1. Sputtering in the deposition column refers to magnetron sputtering with AJA sputtering system at room temperature, whereas the second option stands for electron beam evaporation. Patterning column discusses the selective removal of the layer while RIE stands for reactive ion etching with. Ar milling is done with a Kaufman ion source inside another AJA system located at the EML at MIT.

of Ti using reactive ion etching (RIE) requires very thick resist choices. We have characterized that using CF_4 as the etchant at 50 W plasma, non-patterned Ti film had an etch rate of 25.6 pm/s (18.5 $\Omega/\square$ of Ti, secured from oxidation by thin layer NbN, took 1500 s to fully etch, time for etching NbN is subtracted). For a patterned film, we have observed that this number is approximately halved (i.e. etching is twice as slow). Quantitatively, to pattern the Ti film (of $\approx R_{\text{sheet}} = 20\,\Omega/\square$), $\approx$550 µm thick PMMA or equivalently, $\approx$200 µm thick ZEP520A is required. Both of those options reduce the resolution of the transfer, while using PMMA in particular results in severe pattern degradation. Alternatively, the etchant can be changed or power can be increased to increase the etch rates. We have tried chlorine chemistry but realized the native titania layer protected the metal from Cl_2 etching. For future references, this problem can potentially be solved by ballistically breaking the 'crust' first and then performing the RIE using CF_4. On the other hand, increasing the plasma power amplifies the etching rate for the resist as well, therefore does not solve the problem in a meaningful way.

We have also considered having a metal layer everywhere, practically shorting the entire circuit. Although for a semiconductor-based device it would be fatal, superconductors with zero DC resistance cannot be shorted by a metal layer with low, but-nonetheless non-zero, resistance. However, we have realized that this method reduces the response amplitude of the readout device, yTron. To briefly explain this behavior, one must know that we commonly use the sudden increase of resistance (when the device is switched) to perform 'readout'. The yTron devices are of the same nature as well. Therefore, shunting them with an underlying metal layer reduces the effective resistance in a readout event. This behavior will be explained in further detail in Chapter 4. To summarize, due to readout circuitry related reasons, having a metal layer everywhere is not a viable option.

The problems mentioned above have led to the investigation of aluminum-based devices. Al does not suffer from oxidation as much as Ti and has even higher conductivity. In addition to being present in our AJA sputtering chamber, Al brings the option to implement milling as the patterning method. We have sputtered Al on top of the active NbN layer first and tried to pattern Al by milling and NbN by RIE. The aluminum film with sheet resistance of 18.2 Ω was characterized to get milled in 500 s, using the Kaufman ion source (Ar) inside the AJA system located at the EML at MIT. Unfortunately, this system does not have active sample cooling, which leads to heating of the sample. This factor has caused the aluminum to meltdown and re-freeze when cooled back in spherical geometries (Fig.3-14a). We have also observed that even when the metal is not melting (using Ti instead of Al), PMMA got severely damaged due to both the heating and the ion bombardment (Fig.3-14b).

Failure of these trials have led us using a trilayer stack of NbN, Ti and Pt. NbN layer is sputtered in our AJA system where the Ti and Pt layers evaporated later. The fabrication process for this stack is given in detail below:

- 100 nm Si_3N_4 on Si substrate (laser diced, 1 cm$\times$1 cm, 0.525 mmthick, vendor and Si_3N_4 deposition method unknown) is cleaned via sonication in acetone, methanol and isopropanol (IPA) for 300 s each.

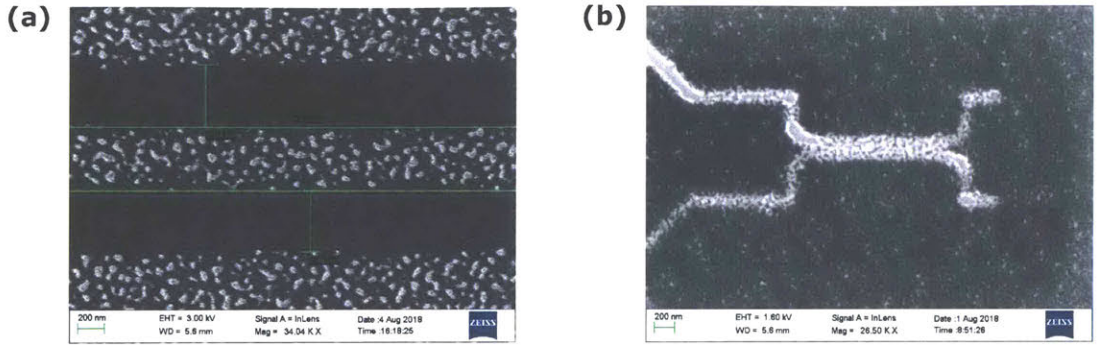

Figure 3-14: Micrographs for failed fabrication attempts. **(a)** Melted aluminum due to the heated environment inside the milling chamber. **(b)** Degradation of the pattern transfer due to resist reflow while Ar milling.

- Polymethylglutarimide (PMGI - SF9) is spun at 4.5krpm for 60 s and baked at 180 °C for 90 s.
- Microposit S1813 photoresist is spun at 4.5krpm for 60 s and baked at 100 °C for 90 s.
- Alignment markers and metal islands (for shunting) are defined by direct photolithography. Direct photolithography is performed using Heidelberg μPG101 with 7 mW beam at 25% duty cycle.
- Photoresist is developed in MF® CD-26 for 60 s and rinsed in DI water.
- 20 nm of Ti and 5 nm Pt ($R_{\text{sheet}} = 46.5\ \Omega/\square$, measured by four-point-probe) is deposited on the chip using e-beam evaporator in MIT NanoStructures Laboratory facilities.
- Liftoff is performed via sonication in acetone for 180 s, followed by N_2 gun dry, 120 s dip in CD-26 (to remove excess PMGI) and finally rinsing in DI water (to clean from CD-26).
- $\approx$25 nm of NbN ($R_{\text{sheet}} = 93\,\Omega/\square$, measured by four-point probe, thickness adjusted by sputtering time) is deposited on the chip using magnetron sputtering at room temperature with AJA ATC Orion sputtering system [12].

- Undiluted ZEP520A (e-beam resist) is spun at 3krpm for 60 s and baked at 180 °C for 120 s.
- Using Elionix FLS-125 (e-beam writer), active layer pattern is aligned to the alignment markers using back-scattered electron detector (BSE[15]).
- E-beam resist is exposed with 500 $\mu C/cm^2$ dose using 500 pA beam current for small features (devices) and 40 nA for larger features (pads).
- ZEP520A is developed in o-Xylene at 5 °C for 90 s followed by 30 s dip in iso-propanol and N_2 gun dry.
- NbN is etched with reactive ion etching (PlasmaTherm 790) using CF_4 at 50 W at 10 mTorr, for a total period of 360 s partitioned (to avoid burning/reflowing the resist) in 3 steps of 120 s each. Chamber was not opened between successive etches.
- Excess resist is stripped in n-methyl-2-pyrrolidone (NMP) at 70 °C for 1 h.

Images of the chip at various points of fabrication procedure can be seen in Fig.3-15[16]. The resolution of this process has been 60 nm but can certainly be optimized further (See Fig.4-6).

3.2.5 Experimental Characterization of Optimized Shunted Nanowires

Experimental characterizations of the shunted nanowires were made under liquid helium immersion conditions using LeCroy Waverunner 620Zi 2 GHz oscilloscope, and Agilent 33600A Trueform Series arbitrary waveform generator (AWG). Waveforms were programmed using MATLAB, which controlled the AWG and also acquired the output waveform from the oscilloscope[17].

[15]Since alignment markers defined in Pt/Ti layer are buried under NbN active layer, they are invisible to any other detectors.

[16]Author wants to take Andrew E. Dane, Di Zhu, Eugenio Maggiolini and Mark Mondol for providing helpful comments that made this fabrication process possible. The author also wants to acknowledge James Daley for his contributions in providing e-beam evaporated material and Kurt Broderick for his contributions with exploring the milling processes.

[17]Drivers to control these instruments, and the setup optimization codes are written by Brenden A. Butters. The author wants to acknowledge his work and thank him for his contributions.

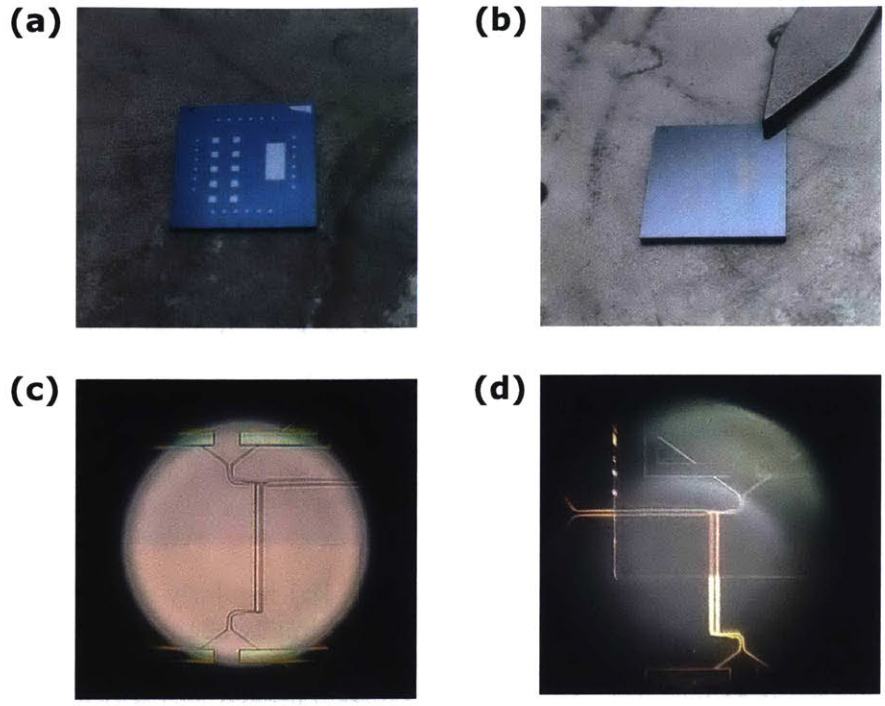

Figure 3-15: Fabrication steps of optimized shunted nanowire process. **(a)** Patterned Pt/Ti layer before NbN deposition. Alignment markers can be seen around the chip where the big windows are where constrictions will be placed on top for realizing shunted nanowires. **(b)** Chip right after the NbN sputtering. Pt/Ti windows are visible to the eye but they are invisible to secondary electron detectors inside the Elionix (e-beam writer), which is the reason for using back-scattered electron sensors. **(c)** Optical microscope image of the chip after exposure and development. The presence of the metal layer can be seen as a contrast difference in the middle. **(d)** Dark-field microscope image of the chip after etching and stripping steps. Presence of the shunting layer beneath is again visible as the reflectivity changes.

First of all, we characterized the I-V relations of the optimized shunted nanowires. Figure 3-16 shows four I-V curves for (a) an unshunted nanowire, (b) a regular (electrically) shunted nanowire, (c) a curious failed attempt, and (d) an optimized shunted nanowire. The switching characteristics of the unshunted nanowires are known to be hysteretic (Fig.3-16a, See Sec.2.1.2). It has also been demonstrated before that shunted nanowires, similar to shunted JJs, show non-hysteretic I-V curves instead (Fig.3-16b) [44]. Therefore, we expect the optimized shunted nanowires to have perfectly non-hysteretic switching as well.

Prior to examining the results, we want to note that these non-hysteretic switching

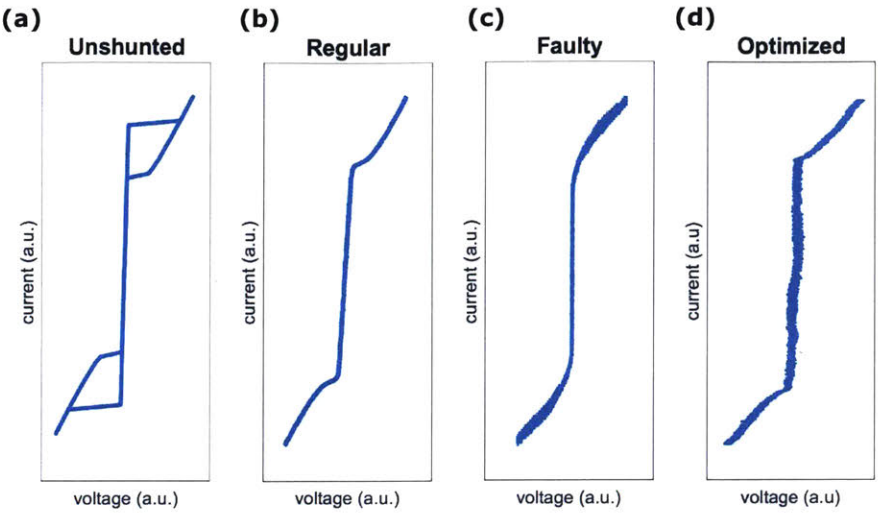

Figure 3-16: I-V characteristics of **(a)** unshunted, **(b)** regularly shunted (lithographically defined side resistor), **(c)** faulty and, **(d)** optimally shunted (by Pt capped Ti beneath the active layer) nanowires. Plots shown in **(a)** and **(b)** belong to characterizations performed by Andrew E. Dane, which are designed and fabricated by Emily Toomey. More detailed results for comparison of unshunted and regularly shunted nanowires can be found in Ref.[43].

characteristics can also arise from thermal suppression of superconductivity. We have observed this situation in a device, where the shunting layer suffered from low quality and bad contact with the active layer. Following many discussions with colleagues[18], we have attributed the downward curving of the I-V characteristics in the resistive regime (Fig.3-16c) to shunting layer behaving as a heater, instead of a heat-sink. This conclusion is also supported by the lower-than-expected switching current of the device shown in Figure 3-16c. Finally, Figure 3-16d shows the I-V characteristics of the optimized shunted nanowire. It can be seen that the switching is non-hysteretic with the resistive region bending upward similar to that of a regular shunted nanowire.

Following the verification of the fundamental behavior, we proceed to show the optimized shunted nanowires in shuttling controlled amounts of current. For this purpose, we have fabricated the same shunted constriction, connected to two loops with different sizes; $L_{\mathrm{loop,small}} = 526.6\,\mathrm{pH}$ and $L_{\mathrm{loop,large}} = 3.94\,\mathrm{nH}$. Note that the values given here are calculations using the following set of equations:

[18]Author wants to thank Di Zhu and Emily Toomey for their helpful comments.

$$T_c = 9\,\mathrm{K};\ R_{\mathrm{sheet}} = 97.2\,\Omega/\square; \tag{3.15}$$

$$L_k = 1.38\frac{R_{\mathrm{sheet}}}{T_c} = 14.9\,\mathrm{pH}/\square; \tag{3.16}$$

$$L_{\mathrm{loop,small}} = (14.9\,\mathrm{pH}/\square)(35.3\square) = 526.6\,\mathrm{pH}, \tag{3.17}$$

$$L_{\mathrm{loop,large}} = (14.9\,\mathrm{pH}/\square)(264.5\square) = 3.94\,\mathrm{nH}. \tag{3.18}$$

These loop inductances give rise to quantization of circulating current with $I_{\mathrm{o,small}}$ = 3.93 µA and $I_{\mathrm{o,large}}$ = 525.16 nA. As explained before, this quantization limits the smallest ΔI_{circ} that can be programmed in the loop. Figure 3-17a show that the small loop has 33 discrete levels. Considering that the shunted nanowire has a switching current of 123 µA, calculations suggest that the control is at SFQ level. Furthermore, for the larger loop, we have observed a continuum of states. From a functional point of view, the large loop device is not as useful, as the definition of a 'state' is hard to make. However, it suggests that the same shunting mechanism can potentially shuttle current finer than 3.93 µA. This results also show that the limitation of obtaining devices with many programmable levels is now limited by the readout circuitry instead of the programming part.

The optimized shunted nanowire presented here has promising characteristics and can potentially be used as a multi-level memory. We note that we do not yet have any memory-specific measurements such as read/write speed or error rates. In Chap. 4 we will use these devices in a novel application for superconducting nanoelectronics: acceleration of deep neural network training.

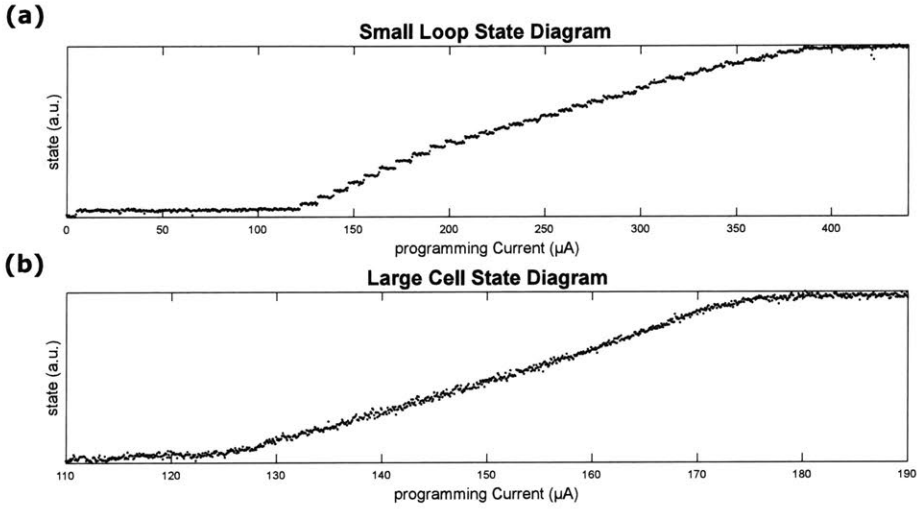

Figure 3-17: State diagrams for **(a)** the small loop and **(b)** the large loop. Quantization of the states can be observed with the small loop where the loop shows 33 well-defined quantized levels. Large cell shows a continuum of states which makes it hard to define as 'levels'. However, it proves that the limitation is not coming from shunting at that point where all the in-between levels can be addressed.

Chapter 4

Design of a Superconducting Deep Neural Network Training Accelerator

4.1 Introduction to Deep Neural Networks (DNNs)

Deep neural networks (DNNs) [27] are powerful tools that are widely used in image recognition [25], natural language processing [9] and big data analytics [31]. Training of these structures involve a high number of matrix multiplications and require massive volumes of data transfer. As a result, time and power costs become unmanageable for larger implementations, which limits the scalability of the approach.

Acceleration of DNN training has attracted great attention in both software and hardware research. Hardware implementations (i.e. neuromorphic engineering) are of particular interest to the nanoelectronics community as a new application field. These approaches aim to build non-volatile memory (NVM) based local processors to reduce the high volumes of data transfer suffered by conventional von Neumann architectures [6]. Implementations involving phase change memory (PCM) [1], resistive random access memory (RRAM), [7] and memristors [36, 24] have recently emerged to realize mixed-signal accelerators.

The most common framework employed in the aforementioned implementations is the crossbar architecture [39]. The main idea behind this method is to store the matrix entries (weights) in cross-point elements and perform multiplications locally.

These frameworks reduce the $\mathcal{O}(N^3)$[1] computational complexity of matrix-matrix multiplication for digital processors to $\mathcal{O}(N)$. Recently, an all-parallel update scheme has been demonstrated for crossbars, that combines computation and application of the weight updates to obtain $\mathcal{O}(N)$ complexity as well [20]. This method extends the application of crossbars to training accelerators, on top of inference machines. However, it also requires a set of strict device properties for the unit cell that have not yet been achieved by any approaches to-date. Imperfections of the unit cells lead to heavy training performance degradation and limit the approach to inference applications only.

In this chapter, we delineate the unit-cell requirements and then provide a superconducting nanowire-based design. Devices demonstrated in here have many programmable non-volatile states that can be used in analog multiplication. Furthermore, their switching characteristics are inherently symmetric due to the quantized nature of flux (equivalently current) in superconducting loops. The results shown in this paper suggest that this new family of devices can unlock the full potential of crossbar-based DNN training accelerators.

4.1.1 Fundamentals of Deep Neural Networks

Deep neural networks (DNNs) are a large family of classifiers that have been greatly successful over the last decade[2]. They are essentially a stack of layers that process a given input through a series of linear and non-linear operations [27]. In most cases, the input-output relationship of DNNs can be interpreted as a classification response for a given sample. Therefore, DNN operations can be analyzed in two main divisions: (1) training DNNs to produce accurate classification results and (2) using already trained DNNs to classify samples (inference).

Training of the DNNs is often conducted by stochastic gradient descent (SGD)

[1]Using schoolbook matrix multiplication. It can be optimized to $\mathcal{O}(N^{2.373})$ using advanced algorithms [11].

[2]We want to note that in this work we use the term exclusively for artificial neural networks. In other words, biomimetic approaches, such as spiking neural networks are not considered in this thesis.

method. This method first defines a cost (error) function which is inversely proportional to the network's performance. Then, the error gradient with respect to each parameter is calculated using the backpropagation algorithm [38]. This algorithm consists of three main operational cycles, namely: (1) forward pass, (2) backward pass, and (3) update. Forward pass is essentially the inference step, where a given sample ($\vec{x}$) is processed by the network to generate a response. In the backward pass, the network response (both the final and the intermediate ones) is propagated in the reverse way (from end to the beginning). This step generates the second set of vectors (often shown as $\vec{\delta}$) which are used to generate the update matrix. Finally, in the update cycle, the outer product of $\vec{x}$ and $\vec{\delta}$ is computed, which is the update matrix ($\Delta\overline{\overline{W}}$), and subtracted from the present weight matrix ($\overline{\overline{W}}$).

Unfortunately, the operation described above is highly computation intensive. First of all, matrix multiplication on conventional von Neumann architectures has the computational complexity of $\mathcal{O}(N^3)$. Moreover, the entire process requires massive volumes of data transfer, which creates another bottleneck with these architectures. The dissonance between the algorithm and the architecture creates unmanageable costs of time and energy. This fact is of further importance for training as it involves many inference operations and further. To quantify, an inference task can take less than a ms whereas training the same network can take up to multiple weeks on high-end digital processors. These factors eventually limit the size of the DNNs that can be trained with the conventional digital processors[3].

4.1.2 Crossbar Architecture

Mixed-signal approaches that are based on crossbar architectures have the potential to mitigate the problems listed above [39]. The main idea behind the crossbar is representing the matrix entries with the cross-point elements where each of them

[3]For clarification purposes, we want to note that crossbar itself does not perform the non-linear functions used in DNN training. These functions include activation functions (e.g. sigmoid, tanh, ReLu etc.) and cost functions (e.g. softmax), and are designed to be handled in digital processors, or separate analog ASIC. This is the main reason behind the naming of these structures as mixed-signal accelerators.

performs local multiplications. The outputs of individual cells are then integrated at the ends of rows/columns to get the resultant product. A sample resistive crossbar is shown in Fig.4-1.

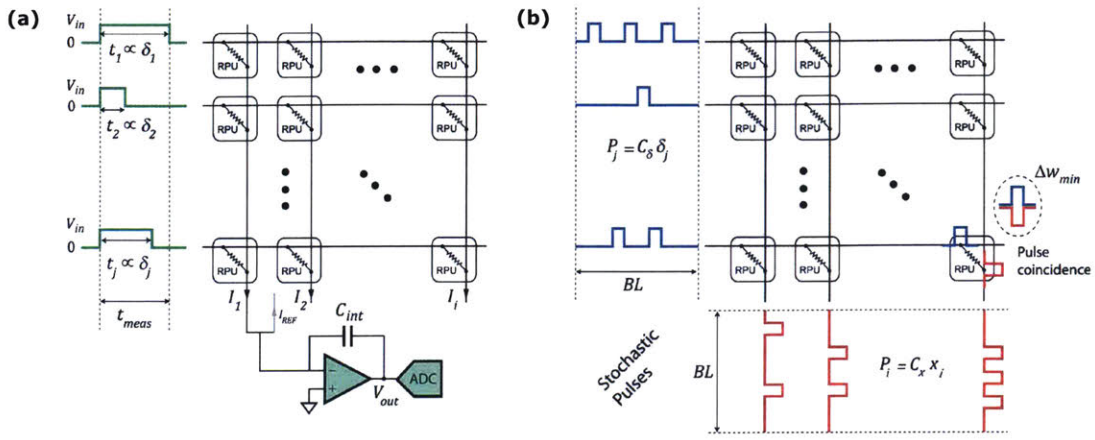

Figure 4-1: Schematics of a resistive crossbar in **(a)** forward pass (vector-matrix multiplication), and **(b)** update with the recently proposed stochastic update scheme. Figure taken from Ref. [18].

In the forward and backward passes, the inputs are represented as voltage pulses of different pulse-widths and fed into the crossbar array from rows (or columns for the backward pass). Through Ohm's law, each cross-point element (a programmable resistor in this example), performs analog multiplication between the input and the state of the device (resistance). The resultant current output coming from each cell is then integrated at the ends of columns (or rows for the backward pass). Considering that the complete vector-matrix product is computed in parallel, crossbar architecture ensures having $\mathcal{O}(\mathrm{N})$ complexity for the matrix-matrix multiplication.

However, to accelerate the complete training operation, it is also required to tune the weights all in parallel. A method called stochastic updating is recently proposed to achieve this task [20]. To briefly summarize, vectors of which the outer product gives the update is represented by random bit-streams where the probability of having a high pulse is proportional to the value it corresponds to. Then, these bit streams are fed into the system from rows and columns such that only the coinciding pulses lead to an incremental update on that particular node (See Fig.4-1b). This method allows

implicit calculation and application of the update all in parallel as well. Therefore, the complete training operation can be performed in $\mathcal{O}(1)$ complexity using crossbar architectures.

Unfortunately, it is observed that due to the imperfections in the unit cell, the chips are often limited to inference applications. Considering that the bottleneck of the large-scale DNNs is to train them, search for the unit cell is an open field with utmost importance. Next section discusses the unit cell requirements in further detail.

4.1.3 Crosspoint Element Requirements

A crosspoint element can be defined as a local multiplier with a tunable multiplication factor. The number of its programmable states define the resolution of the multiplier. A high ON/OFF ratio[4] and a high number of states are desirable in order to have devices with higher controllability. These states are also required to be non-volatile, with retention times longer than the complete training procedure in order to enable incremental and implicit updating.

As previously emphasized by [18], the switching dynamics between states should be symmetric with low inter-device variation. Here we define symmetry as the state of the cell remaining effectively unchanged, following an arbitrary sequence of increment and decrement pulses of equal total number. Considering that the DNN training consists of numerous small updates, the presence of any asymmetry creates drift terms in weight programming and leads to significant deterioration of the training performance[5]. Unlike asymmetric updates (that are of drift nature), random noise (of diffusion nature) in readout and/or update can be tolerated due to the intrinsic dynamics of neural network training [20, 45]. Finally, to realize the aforementioned update scheme, switching of the device requires a thresholding mechanism to allow state changing only under concurrent pulses on the row and column (See Sec. 4.2.3).

[4]Ratio of the highest multiplication factor to the lowest one.

[5]Some requirements with the crosspoint element can be relaxed by using simple algorithmic manipulations as explained in [18]. However, in a realistic scenario, all cells (N^2) will bear different asymmetries. In a parallelized update scheme, only ($2N$) variables can be accessed to. Therefore, none of the current algorithm techniques have been able to relieve the symmetry requirement. This justifies that the need for symmetry is absolute.

In conclusion, to successfully implement an efficient DNN training accelerator, a symmetric switching multi-level programmable non-volatile unit cell is essential. These properties should be maintained at the device level since compensation techniques limit the applicability of the approaches only up to a certain problem size. Methods that involve serial accessing to individual devices reduce the acceleration factors significantly. On the other hand, approaches that introduce compliance circuitries around the device impair the scalability of the approach with inefficient device counts.

Upon the satisfaction of these strict criteria, mixed-signal frameworks can potentially accelerate DNN training 30.000× with respect to the state-of-the-art digital processors [20] which is the very reason behind the sublime efforts in crosspoint element research in the nanoelectronics community.

4.2 Superconducting Nanowire-Based Processor

In this section, we will first discuss the operation dynamics of the superconducting nanowire-based cross-point element. Then, a new method to perform analog multiplication with these devices will be explained. Finally, system-level details of the crossbar architecture will be shown with connection diagrams under different steps of the backpropagation algorithm and an analysis of the periphery requirements.

4.2.1 State Representation and Programming

The superconducting nanowire-based processing element is essentially a superconducting loop that employs a shunted constriction [43] as the writing element and a y-shaped current combiner (yTron) [29] as the reading element (Fig.4-2a). As the current flowing in the constriction arm is increased above nanowire's switching current (I_{SW}), it switches into resistive (normal) state (Fig.4-2b-I). This event redirects the current on the constriction branch into the loop (Fig.4-2b-II). Under the absence of current flow, the nanowire restores its superconducting state and entraps the excess current shuttled into the loop (Fig.4-2b-III). The state of the unit cell can be repre-

sented with this circulating persistent current (I_{circ}) and can be controlled through the constriction.

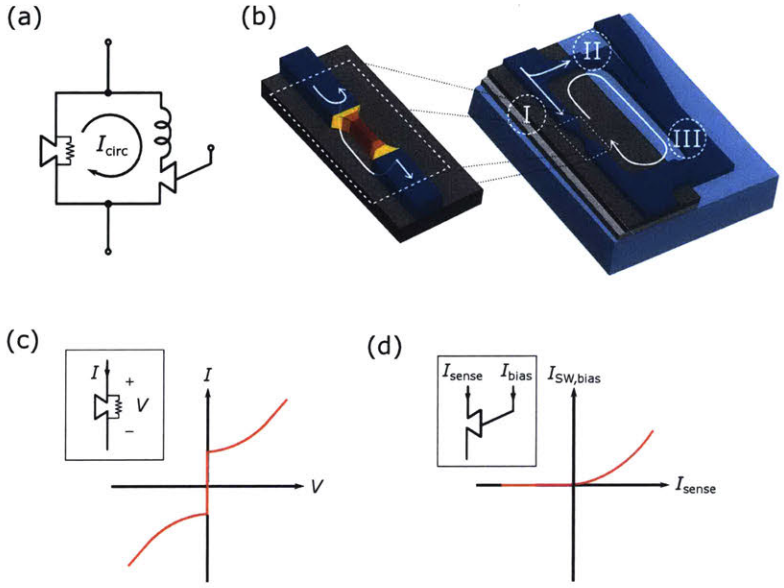

Figure 4-2: Programming of the cell and the fundamental characteristics of the subcomponents: **(a)** Circuit schematic of the unit cell. **(b)** Constriction switches into the resistive state as a response to the increased current density on it (I). Due to the resistance of the constriction, current in the loop redistributes, which allows constriction heal back to its superconducting state (II). At the point when the loop is fully superconducting again, the excess current gets entrapped inside the loop (III). **(c)** Notional I-V curve of a shunted nanowire which shows non-hysteretic characteristics [43]. **(d)** Characteristic graph for yTron, showing modulation of the switching current of the bias arm as a function of the current flowing in the sense arm [29]. Insets in **(b)** and **(c)** represent the circuit symbols for the shunted nanowire and yTron.

However, for an unshunted nanowire, this switching event is abrupt, meaning that even a single event loads the loop to its maximum capacity (See Sec.3.1). To have a fine control over the current shuttled into the loop, the constriction can be locally shunted by a resistor [43]. This shunt branch dampens the switching characteristics of the constriction and avoids excess current to flood into the loop all at once (See Sec.3.2). If the impedance of the shunt branch is sufficiently low, current shuttling can be as fine as a single flux quantum (SFQ, Φ_o)[6][43]. Considering that there are no dis-

[6]Circulating currents in a superconducting loop can only occur in integer multiples of Φ_o/L_{loop},

sipative elements on the loop, this circulating current remains unchanged indefinitely, which means that the state is perfectly non-volatile. Furthermore, the inherently discretized nature of the system (flux quantization) ensures symmetric state switching under SFQ control, since there are no stable intermediate states.

The number of programmable states, N, is given by $N = I_{SW}/\Delta I$, where I_{SW} is the switching current of the narrowest part of the loop (which is the constriction for a properly designed device) and, ΔI is the circulating current difference between adjacent states (equivalent to $L_{loop}\Phi_o$ under SFQ control). Therefore, a higher number of programmable states can be achieved by increasing the loop inductance. Unlike normal metals, some superconductors possess a high kinetic inductance, L_k (arising from the inertia of the charge carriers [2]). For such materials, the L_k can be orders of magnitude higher than the magnetic inductance. This property allows having large inductors without excessive area consumption.

Finally, to read the state of these devices, the circulating current, I_{circ}, should be sensed. The current flowing in the sense arm modulates the switching current of the bias arm as illustrated by Fig.4-2c. The reason behind this effect is a simple geometric phenomenon known as current crowding [29]. Therefore, the point at which the bias arm switches indirectly measures the current flowing in the sense arm (I_{circ} in a cell). Since the path of the circulating current does not experience any switching event, the state of the loop remains unchanged and the readout is non-destructive.

The features mentioned above make this approach a perfect candidate for a multi-level programmable non-volatile unit cell with inherently symmetric switching characteristics.

4.2.2 Analog Multiplication

In a conventional crossbar with resistive cross-point elements, analog multiplication is simply carried by Ohm's law where the weight is represented by the conductance of the element. Then, the contribution from each element is combined at the ends

where $\Phi_o = h/2e = 2.07 \times 10^{-15}$ V s is the flux quantum, and L_{loop} is the device loop inductance. Therefore the finest control over the current is limited by this quantization

of columns via Kirchoff's law, forming the vector-matrix product. Superconducting devices, with infinite conductance, need a new approach to implement multiplication.

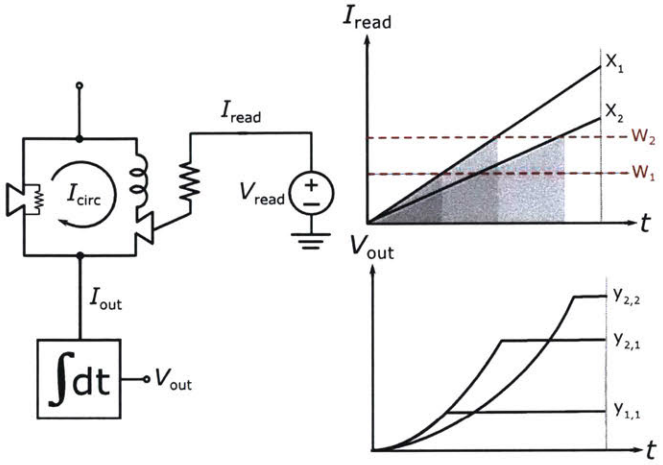

Figure 4-3: Analog multiplication scheme for superconducting nanowire based processor. Circuit schematic for the multiplication operation (left). Input current (I_{READ}) and output voltage (V_{OUT}) for 3 different cases (right). State of the loop determines the integration time while the ramp height (or slope) determines the integrand. Integrated output is proportional to W^2x^{-1} where W is the state of the cell and x is the input.

Multiplication system adopted in this works involves representation of the input signals as voltage ramps (instead of constant voltage levels). This signal is then converted into current at each yTron (bias arm) with the use of a series resistor[7]. As a response to this ramping input, yTron's bias arm switches at a certain level that is determined by the state, I_{circ}, of that unit cell (Fig.4-3). After this switching event, the large induced resistance of the bias arm (many kΩ) will result in a precipitous drop of the output current. The output of the multiplication operation is obtained by the integration of this output current. For a unit cell that is set to a higher state (larger I_{circ}), bias arm switches at a later time, producing a larger integrated output. It can be analytically calculated that the integrated output is proportional to the product of the cell state and the input with the relation given as W^2x^{-1}.

[7]Voltage ramps are provided across the forward input (or backward input for backward pass) lines where the other end of the yTron is connected to integrators, which are at ground level (4-4)

4.2.3 Crossbar Architecture and Operation

The architecture used in this work is an implementation of the conventional crossbar approach [39], modified to be compatible with stochastic update scheme. It is devised to support training with stochastic gradient descent (SGD) using the backpropagation algorithm to compute the derivatives [38] in all 3 operation cycles: forward pass, backward pass, and update.

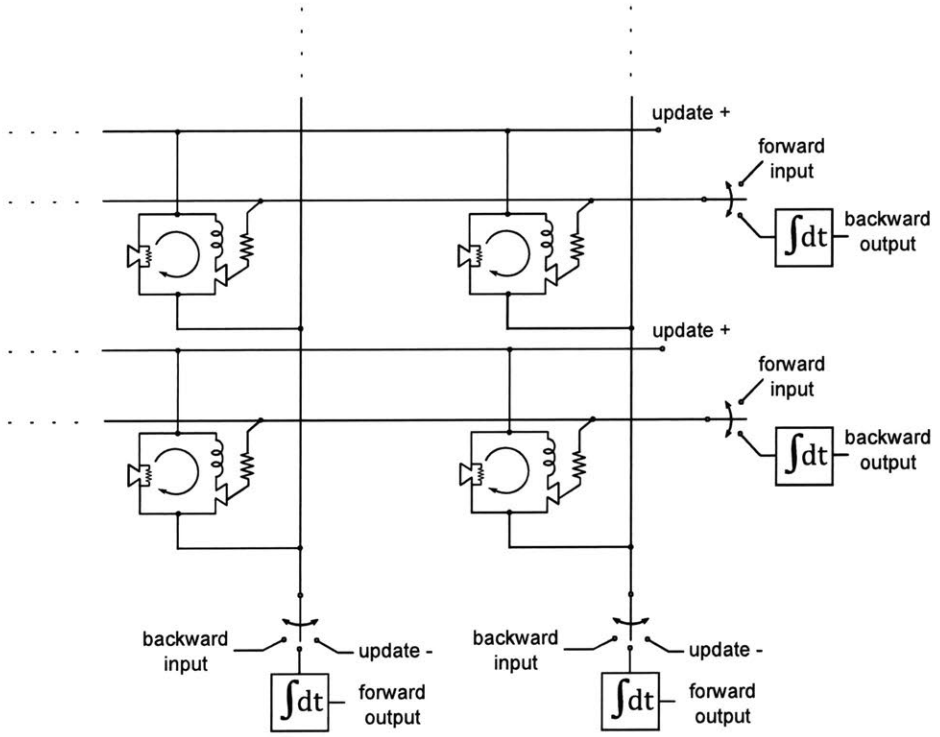

Figure 4-4: Circuit schematic of a section of crossbar array and connection schemes under different operations. Forward pass involves the application of inputs from the rows and the integration of the product at the ends of columns. The backward pass is computed the reverse way to use the transpose of the matrix. Update pulses are sent from both rows and columns from separate lines which include Josephson Transmission Lines (JTLs)

The forward pass involves computing the product of the input vector with the weight matrix. As explained in Sec.4.2.2 the transfer function of the unit cell for a given state and output is proportional to $W^2.x^{-1}$. This transfer function can be used as the "multiplication" in DNN training with minor algorithmic changes. Division

term can be removed by representing inputs as x^{-1} instead of x. On the other hand, the W^2 term leads to a weight dependent learning rate for each cell. Non-linear devices have formerly been investigated in actual neural network training emulations in Ref.[20, 18, 19]. None of those studies have observed any significant performance deterioration due to the non-linearity of the unit cell transfer function.

Calculation of the backward pass can then be done by employing the same method, only with the difference of inverted input signs for convention purposes and interchanged input/output terminals to represent the transpose of the weight matrix.

Updating of the cell follows the stochastic update scheme with the application of pulses from rows and columns with opposite signs (Fig.4-3a). Pulses that are coincident in time provide sufficient current to switch the shunted constriction into the normal state. This switching shuttles an incremental current into, or out of, the loop depending on the input polarity as explained in 4.2.1. Individual pulses are thresholded by the switching current of the constriction and do not lead to any change. This satisfies the AND operation requirement of the method for implicit and all-parallel calculation of the update [20].

Crossbar architecture requires operation with voltages as the programming input. However yTrons operate with current and therefore the input voltage levels are required to be locally translated into a current, with a resistor in series with the bias arm. Constrictions, on the other hand, do not suffer from this problem since the pulses are transmitted with active transmission lines. This technology allows providing the pulses without suffering from deterioration of the signal or sneak-paths, therefore, it is suitable.

Peripheral circuitry for this crossbar can be built using readily available technologies such as cryo-compatible CMOS and Josephson Junctions (JJs). The elements required for a full-scale integration can be listed as Josephson transmission lines, integrators, analog to digital converters (ADCs), and non-linear function evaluators (e.g. a decent CPU or ASIC). Note that placing processors in the cryogenic environment as well would reduce the volume of data transfer between 4K and room temperature. This would drastically reduce the heat load that would arise from high-bandwidth ca-

bles otherwise. The architecture can then be used to accelerate training of Restricted Boltzmann Machines (RBMs), fully connected neural networks[20], convolutional neural networks (CNNs) [18], recurrent neural networks (RNNs), and LSTM Networks [19].

4.2.4 Simulation Results for Superconducting Nanowire-Based Processor

Simulations of the crosspoint element are done using the simulator constructed in this work. We have used the classical solver described in the Sec.2.2.3, as the phase evolution across the device does not make an operational difference (i.e. circuit involves nanowires that do not conserve phase coherence during operation). As explained in Sec.2.2.3, quantization of flux in the superconducting loop is enforced, when the nodal and branch quantities reach steady state (following their computation in the absence of such a constraint). The circuit schematic used to represent the cross-point devices is given in Fig.4-5[8].

Through the simulations, it was observed that programming the cell in incremental mode requires short pulse-widths. This behavior can be explained as cutting the input before the cell reaches to its steady-state, such that the I_{circ} is programmed to only a fraction of the level that it would normally be set to. This allows the repeated application of the same input in incremental programming as shown in Fig.4-5. As expected, the states were inherently discrete and perfectly symmetric under the SFQ control.

However, the delivery of such fast pulses ($\approx 10\,\text{ps}$) to the cross-point cells require repeating active transmission lines (Josephson Transmission Lines, JTL). Design and implementation of JTLs is a well-known and very effective technique that is widely used in the field but was not pursued in this work for simplicity. Therefore, all experimental demonstrations of multi-level programming are done with different height

[8]yTron on the right branch is simply represented as an inductor since under proper operation it should never switch. In the simulation the current that circulates in the loop can be read without a circuit element, therefore it is simplified to the circuit shown here.

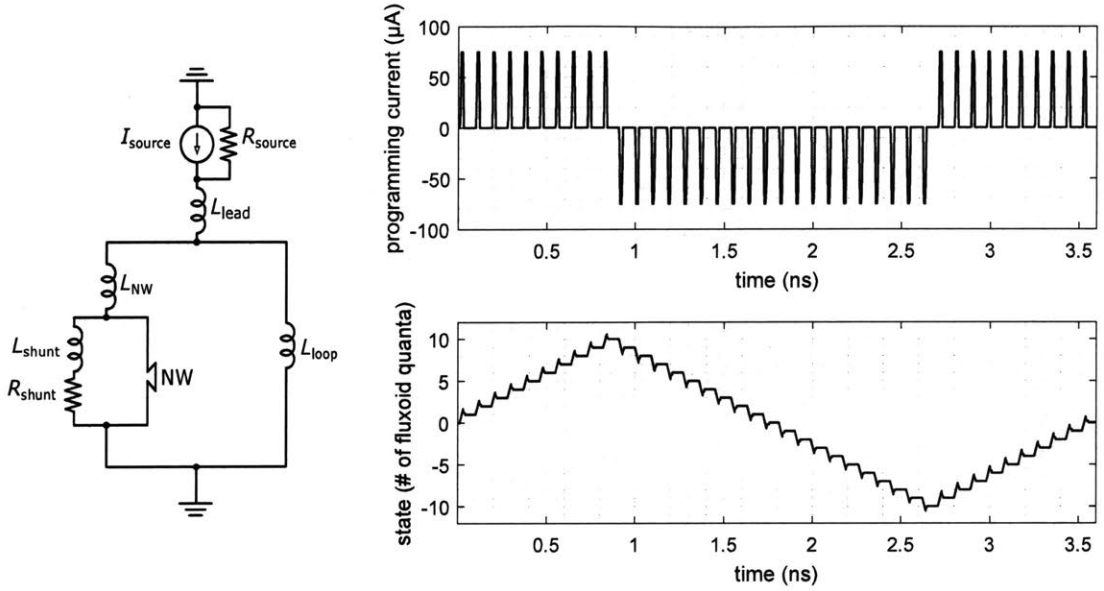

Figure 4-5: Electro-thermal circuit simulations for the unit cell under incremental programming mode. The circuit schematic of the unit cell (left) and simulation results showing incremental state control (right). Pulses of the same height (for each polarity) are applied to the cell. In this simulation amplitude of the programming pulse is twice the switching current of the constriction and pulse-widths are 100 ps. It can be seen that the state of the cell can be controlled incrementally, with SFQ steps in a perfectly symmetric way.

inputs (i.e. the devices are programmed in set mode instead of increment mode).

4.2.5 Design and Fabrication of Unit Cell

Increasing the number of programmable states for the unit cell requires fine control over the current shuttled per switching event. As shown in Sec.3.2.3 this can be done by lowering the shunt branch impedance (R_{shunt}, L_{shunt}). Furthermore, simulated results indicate that the increased thermal sinking of the active layer improves the shunting characteristics by providing additional dampening over the thermal response (See Fig.3-13). Considering that at cryogenic temperatures thermal conductivity follows electrical conductivity, design can easily be improved by placing the metal layer directly in contact with the constriction. This method significantly reduces L_{shunt} while functioning as a heat sink for the constriction. When a shunted nanowire switches, the voltage response is reduced due to the lower effective normal resistance

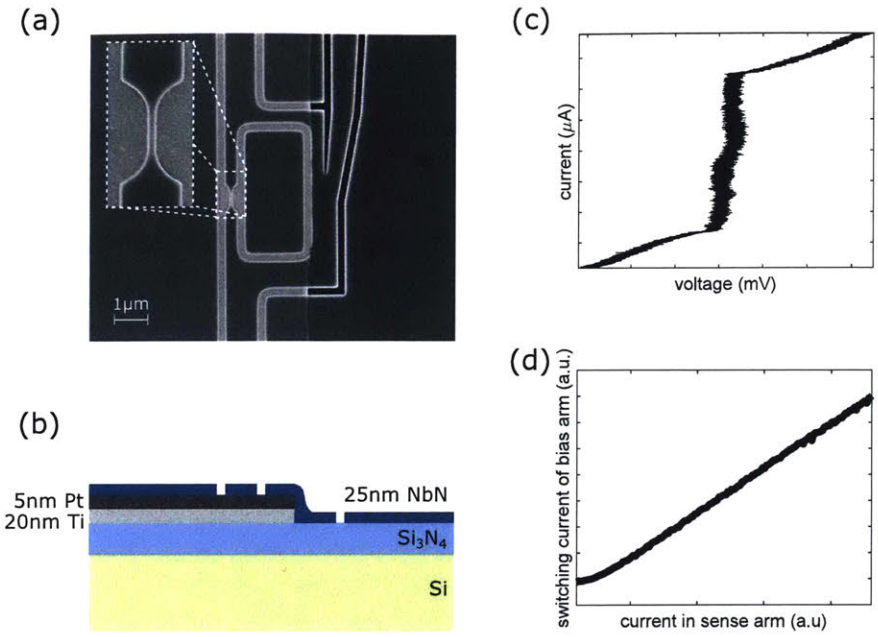

Figure 4-6: Characterizations of the fabricated unit cell and its subcomponents. **(a)** SEM micrograph of the unit cell with constriction width of 80 nm. The brighter background is a manifestation of the presence of the Pt/Ti shunting layer behind the constriction side. **(b)** Material stack used for the fabrication of the devices. **(c)** Experimental I-V curve for the shunted nanowire. The absence of hysteresis indicates the shunting is effective. This data is the same as the one shown in Fig.3-16c **(d)** Characteristic graph of yTron that shows the modulation of the switching current of the bias arm as a function of the current flowing in its sense arm.

($R_{NW}//R_{shunt}$). While this effect is desirable for the constriction; it makes the reading of the yTron difficult. For this reason, the yTron bias arms were not shunted.

The design mentioned above was fabricated on 25 nm thin film NbN with metal shunting stack of 20 nm Ti capped with 5 nm Pt to protect Ti layer. This chip is indeed the same one which we have described the fabrication steps in Sec.3.2.4. Fig.4-6a shows a scanning electron micrograph of a cross-point element, where the presence of the metal layer can be seen via the contrast difference. The loop inductance of the unit cell shown in Fig.4-6a is calculated to be ≈ 526.6 pH which makes circulating current quantized in 3.93 μA steps. The constriction on the same device has a measured switching current of ≈ 123 μA.

4.2.6 Experimental Characterization of Superconducting Nanowire-Based Inductive Processing Unit

Experimental characterizations of these devices were made under liquid helium immersion conditions using LeCroy Waverunner 620Zi 2 GHz oscilloscope, and Agilent 33600A Trueform Series arbitrary waveform generator (AWG). Waveforms were programmed using MATLAB, which controlled the AWG and also acquired the output waveform from the oscilloscope. The state diagram for the unit cell in Fig. 4-6a is shown in Fig.4-7a.

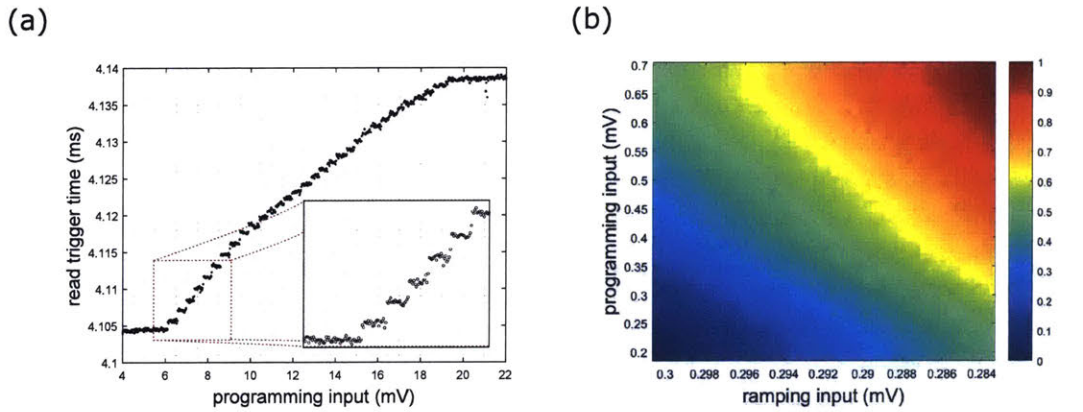

Figure 4-7: State diagram and analog multiplication results for the fabricated unit cell. **(a)** Read trigger time (state of the cell) as a function of programming input. First, a programming input was sent to the device to set the state of the cell. Then, readout was performed by sending a ramping input to the yTron's bias arm and recording the time (current) at which it switched. This operation is repeated for a range of programming inputs to obtain the state diagram of the unit cell. 33 discrete steps were observed which can be thought of as a 5-bit memory. The non-linearity of the state diagram is attributed to the yTron characteristics. **(b)** Analog multiplication results for the unit cell. Similar to the generation of the state diagram, the cell was first set to a certain state using 50 different programming inputs. Then, 50 different ramping inputs were used to perform the analog multiplication operation described in Sec.4.2.2, by again recording the read trigger time. Results are presented as a colormap where the output of the multiplication is represented with the color versus cross-swept programming and ramping inputs. Input ranges are adjusted such that the upper right and lower left corner values are roughly the same.

It can be seen that 33 discrete states could be resolved with this device which is very similar to the calculated number of states at the single-flux-quantum limit. This

result verifies effective shunting of the constriction as the programming control is at the quantization limit. However, it can also be seen that the non-linearity and the noise of the yTron readout limit the resolution of the states. Therefore, in order to achieve a higher number of states using these devices, the yTron design should be further optimized.

The device is then used in a demonstration of the proposed multiplication scheme. Fig.4-7b shows the normalized integrated output as a function of input level and cell state (programming level). It can be seen that the output is proportional to the product of input and weight with the expected characteristics. On the other hand, it can be seen that the sensitivity to the ramping input is still higher than that to the programming input. This behavior manifests itself in the constant output contours shown in Fig.4-7b, where the curvatures are less than expected. One way to analyze this issue is to consider it as a limited dynamic range problem as any multiplication contour-map would appear linear, when the axes are not balanced. Therefore, improved yTron designs might be able to address this issue. Nonetheless, for the application of DNN training, similar to the quadratic behavior, this imperfection can be tolerated by the algorithm.

4.2.7 Discussions for Superconducting Nanowire-Based DNN Training Accelerator

In this work, we have designed, fabricated, and tested superconducting nanowire-based inductive processing element as a cross-point device. Programming of these devices at the single flux quantum (SFQ) limit allowed resolution of 33 discrete, perfectly non-volatile states. We have proposed and demonstrated a method to perform analog multiplication of the input with the state of these devices. Furthermore, using our electro-thermal circuit simulator, we presented the requirements to use the cell in incremental programming mode which leads to perfectly symmetric switching states. Operation of these devices in a crossbar is described for full DNN training that implements stochastic gradient descent using the back-propagation algorithm for

calculation of the gradients.

Former system level analysis in Ref.[20, 18] has shown that having symmetric switching cross-point elements is crucial for building crossbar based DNN training accelerators without causing significant performance degradation. The devices made in this work provides this feature, unlike other non-volatile memory alternatives. The current number of states might be sufficient for small scale applications, however, it needs to be increased to address more complex training tasks. We do not see any fundamental reasons before increasing the number of programmable levels by optimizing the readout mechanism and the device layout. Addressing 100 states with these devices is realistic in the near future where another order of magnitude might become challenging without increasing the device sizes.

We have also provided designs of a full-scale crossbar implementation using our crosspoint elements. The new analog multiplication process we have implemented is suitable for the purposes of the application, although it is not exactly 'multiplication'. The main limitation we see before a large-scale implementation is the routing of the signals, particularly in the update cycle. As the crossbar in this approach is operated with currents, the signaling paths require directional elements to propagate the pulses correctly. On the other hand, forward and backward pass cycles do not suffer from this problem as they can be controlled using voltage pulses, which are then locally converted to current pulses at each yTron. A superconducting nanowire-Josephson junction hybrid architecture can potentially overcome these challenges to provide a large-scale (e.g. 1000×1000) crossbar array.

The devices presented in this work are designed in a particular way such that their weak points can be tolerated by the application and vice versa. Future work includes scaling up the unit cells to increase memory capacity and integrating JTLs to allow using the cells in the incremental operation mode. Ultimately, the concept presented in this work shows promising characteristics and has the potential to realize the acceleration of DNN training without introducing network performance degradation.

Chapter 5

Conclusions and Future Work

In this work, we have constructed a unified simulation platform where superconducting electronics can be designed and optimized with high accuracy and performance. For these purposes, we have implemented two solvers, a phase-coherent solver, and a classical solver. Following this differentiation, we have included phase as a nodal quantity for devices that rely on the evolution of phase information across the device (e.g. Josephson junctions) at the expense of increasing the system size. We have chosen not to do the same for the superconducting nanowire-based devices, where the time dynamics of the phase does not play a major role. In order to be able to deal with the heavy non-linearity of these devices, we have implemented a Newton solver that makes use of gradual conjugate residue method in an iterative scenario. Furthermore, we have made use of adaptive time scaling in order to improve the performance of the simulator by increasing time-steps whenever there is 'dead-time' detected in the simulation. The simulator we have built includes phenomena from a variety of physical disciplines such as quantum mechanics (e.g. quantization of flux), thermodynamics (e.g. hot-spot growth and decay), as well as basic electrical characteristics. Josephson junctions and superconducting nanowires are modeled using existing literature and implemented in the simulator. Resulting simulated responses are validated with the experimental data. We have demonstrated device level verifications such as: the switching characteristics of superconducting nanowires (Fig.2.3), the IV characteristics of a shunted JJ (Fig.2-4), and an accurate representation of

a single flux quantum pulse (Sec.2.3). Then, circuit level simulations are conducted for SFQ logic electronics such as: NAND gate (Fig.2-5), AND gate (Fig.2-7) and a 3-bit counter (Fig.2-8). Simulator proved to be able to produce accurate results while keeping a high-performance metric of computational time scaling with $\mathcal{O}(N^{1.5})$. The framework of this simulator is particularly built such that it allows designer flexibility to trade between accuracy and simulation time as well. Therefore, the software developed in this simulator can be considered as successful in providing what it was initially intended for.

Following the verification of the simulator's capability of portraying superconducting devices, we have started using it to optimize the layout and material considerations for devices in an existing application. First, we have analyzed a nanowire memory (nMEM) in Sec.3.1 to show its fundamental operation principles (Fig.3-2). Then, we have used the simulator to run a sensitivity analysis of the operation margins with respect to the circuit parameters (Fig.3-4). We have fabricated the optimal devices (Fig.3-5) and experimentally characterized them in liquid helium immersion measurements. Results have shown superior performance metrics, further validating the capability of the simulator (Fig.3-6).

The second family of devices we have presented results is the shunted nanowires in controlled flux shuttling applications. We have first shown that the simulator was able to explain the fundamental properties of these devices (Fig.3-7), and replicate former experimental characterizations (Fig.3-10). Following these verifications we have again analyzed the parameter space to have optimal devices (Fig.3-11). More importantly, we have observed that instead of changing the circuit parameters, altering the material stack could provide bigger impacts on the device performance (Fig.3-13). This technique required providing thermal sinking to the nanowire, dampening its switching response and providing controllable shuttling characteristics. These results were concurrently able to realize the circuit parameter sweep suggestions as well, therefore chosen to be optimal. We have again realized these devices by fabricated nanowires, that are shunted by an underlying metal layer. This required optimization of fabrication steps as explained in Sec.3.2.4. Experimental characterization of

these showed that the technique indeed provides superior switching characteristics and allows single-fluxon level shuttling (Fig.3-17).

Finally, we have used the simulator and the devices optimized in Chap.3.2.3 in a completely novel application for the superconducting nanoelectronics, mixed-signal accelerators for deep neural network training. For these purposes, we have first analyzed crossbar arrays in training of DNNs, where they provide immense parallelization of the complete training procedure. We have also reviewed existing non-volatile memory technologies that have attempted creating cross-point elements. As explained in Sec.4.1.3, these approaches fail in providing symmetric and fast switching, perfectly non-volatile, multi-level programmable multipliers, whereas the superconducting nanowire-based implementations can provide. We have first designed an alternative method to perform analog multiplication using these devices (Sec.4.2.2). Then, state-programming dynamics of these devices is analyzed using our simulator. We have shown how to obtain perfectly symmetric incremental programming of these devices using SFQ signals (Fig.4-5). Then, we have used the devices we have fabricated before for the characterization of optimized shunted nanowires and demonstrated a nano-scale analog multiplier with 33 programmable levels (Fig.4-7). Finally, we have provided system level analysis and derived the requirements to obtain a full-scale integrated system.

From a software point of view, the simulator can be translated into a more optimal language (such as C++), to provide better performance. Initially, we have selected MATLAB for the convenience of use and powerful built-in functions it provides. However, we are well aware that the performance can be improved significantly at a lower level implementation of the same system. Considering that we are no more using that many MATLAB specific functions, translation of the system is certainly the next step.

Considering that the simulator has been built to provide a platform for hybrid superconducting electronics applications (i.e. JJ and nanowire together), future work should include exploring circuits that involve them together. Each family of devices has particular strengths and weaknesses, where they can be implemented in a comple-

mentary way. For example, the crossbar array requires implementation of Josephson transmission lines (JTLs). The timing constraints for such a system can be analyzed using the simulator constructed in this work.

Furthermore, system level simulations should be done to better understand the dynamics of a complicated circuit. We have realized that when we are optimizing a unit cell, the metric we are optimizing for, is not necessarily the right metric to optimize for. Alternatively, another bottleneck that coexists might disallow benefiting from solving a single-sided problem. Therefore, further phenomena such as cross-talk, coupling, and distortion should be added to count for the system-level effects.

Finally, we want to note once more that superconducting nanoelectronics has an immense potential to solve the existing problems. However, they are very hard to work with as they have intricate and very sensitive operation dynamics. Therefore, devices should be more robust in terms of environmental conditions, fabrication related variations and signaling. The simulator constructed in this work is only a small step towards the right direction and it has already provided immediate returns in real-life problems. Once the void of powerful design and simulation tools is filled in the field, superconducting nanoelectronics can realize near-future technologies with superior speed and power properties.

Bibliography

[1] Stefano Ambrogio, Pritish Narayanan, Hsinyu Tsai, Robert M Shelby, Irem Boybat, Carmelo Nolfo, Severin Sidler, Massimo Giordano, Martina Bodini, Nathan CP Farinha, et al. Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature*, 558(7708):60, 2018.

[2] Anthony J Annunziata, Daniel F Santavicca, Luigi Frunzio, Gianluigi Catelani, Michael J Rooks, Aviad Frydman, and Daniel E Prober. Tunable superconducting nanoinductors. *Nanotechnology*, 21(44):445202, 2010.

[3] MM Bagheri-Mohagheghi, B Pourhassan, M Adelifard, and M Shokooh-Saremi. The feasibility of correlation between superconductivity and magnetic monopole: Establishment of semi-classical electrodynamics projection, tkh topological theory and dynamic vortex models. *arXiv preprint arXiv:1704.02228*, 2017.

[4] Karl K Berggren, Qing-Yuan Zhao, Nathnael Abebe, Minjie Chen, Prasana Ravindran, Adam McCaughan, and Joseph C Bardin. A superconducting nanowire can be modeled by using spice. *Superconductor Science and Technology*, 31(5):055010, 2018.

[5] Paul I Bunyk. Automated calculation of mutual inductance matrices of multilayer superconductor intergrated circuits. In *Extended Abstracts of Int. Superconductive Electronics Conf.(ISEC'93)*, volume 62, 1993.

[6] Geoffrey W Burr, Robert M Shelby, Abu Sebastian, Sangbum Kim, Seyoung Kim, Severin Sidler, Kumar Virwani, Masatoshi Ishii, Pritish Narayanan, Alessandro Fumarola, et al. Neuromorphic computing using non-volatile memory. *Advances in Physics: X*, 2(1):89–124, 2017.

[7] Ping Chi, Shuangchen Li, Cong Xu, Tao Zhang, Jishen Zhao, Yongpan Liu, Yu Wang, and Yuan Xie. Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory. In *ACM SIGARCH Computer Architecture News*, volume 44, pages 27–39. IEEE Press, 2016.

[8] Tai-Chang Chiang. Superconductivity in thin films. *Science*, 306(5703):1900–1901, 2004.

[9] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[10] Leon N Cooper. Bound electron pairs in a degenerate fermi gas. *Physical Review*, 104(4):1189, 1956.

[11] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 1–6. ACM, 1987.

[12] Andrew E Dane, Adam N McCaughan, Di Zhu, Qingyuan Zhao, Chung-Soo Kim, Niccolo Calandri, Akshay Agarwal, Francesco Bellei, and Karl K Berggren. Bias sputtered nbn and superconducting nanowire devices. *Applied Physics Letters*, 111(12):122601, 2017.

[13] AP Drozdov, MI Eremets, IA Troyan, Vadim Ksenofontov, and SI Shylin. Conventional superconductivity at 203 kelvin at high pressures in the sulfur hydride system. *Nature*, 525(7567):73, 2015.

[14] Emerson S Fang. A josephson integrated circuit simulator (jsim) for superconductive electronics application. In *Extended Abstracts of 1989 International Superconductivity Electronics Conf.(The Japan Society of Applied Physics, Tokyo, 1989)*, 1989.

[15] Coenrad J Fourie and Mark H Volkmann. Status of superconductor electronic circuit design software. *IEEE Transactions on Applied Superconductivity*, 23(3):1300205–1300205, 2013.

[16] Kris Gaj, Quentin P Herr, Victor Adler, Andy Krasniewski, Eby G Friedman, and Marc J Feldman. Tools for the computer-aided design of multigigahertz superconducting digital circuits. *IEEE transactions on applied superconductivity*, 9(1):18–38, 1999.

[17] VL Ginzburg. Ginzburg, vl, and ld landau, 1950, sov. phys. jetp 20, 1064. *Sov. Phys. JETP*, 20:1064, 1950.

[18] Tayfun Gokmen, Murat Onen, and Wilfried Haensch. Training deep convolutional neural networks with resistive cross-point devices. *Frontiers in neuroscience*, 11:538, 2017.

[19] Tayfun Gokmen, Malte Rasch, and Wilfried Haensch. Training lstm networks with resistive cross-point devices. *arXiv preprint arXiv:1806.00166*, 2018.

[20] Tayfun Gokmen and Yurii Vlasov. Acceleration of deep neural network training with resistive cross-point devices: design considerations. *Frontiers in neuroscience*, 10:333, 2016.

[21] Brian David Josephson. Possible new effects in superconductive tunnelling. *Physics letters*, 1(7):251–253, 1962.

[22] Alan M Kadin. *Introduction to superconducting circuits*. Wiley New York, 1999.

[23] Andrew J Kerman, Joel KW Yang, Richard J Molnar, Eric A Dauler, and Karl K Berggren. Electrothermal feedback in superconducting nanowire single-photon detectors. *Physical review B*, 79(10):100509, 2009.

[24] Kuk-Hwan Kim, Siddharth Gaba, Dana Wheeler, Jose M Cruz-Albrecht, Tahir Hussain, Narayan Srinivasa, and Wei Lu. A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications. *Nano letters*, 12(1):389–395, 2011.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[26] Jeffrey C Lagarias, James A Reeds, Margaret H Wright, and Paul E Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, 9(1):112–147, 1998.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[28] Fritz London and Heinz London. The electromagnetic equations of the supraconductor. *Proc. R. Soc. Lond. A*, 149(866):71–88, 1935.

[29] Adam N McCaughan, Nathnael S Abebe, Qing-Yuan Zhao, and Karl K Berggren. Using geometry to sense current. *Nano letters*, 16(12):7626–7631, 2016.

[30] Adam N McCaughan and Karl K Berggren. A superconducting-nanowire three-terminal electrothermal device. *Nano letters*, 14(10):5748–5753, 2014.

[31] Maryam M Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1):1, 2015.

[32] Anant V Narlikar. *The Oxford Handbook of Small Superconductors*. Oxford University Press, 2017.

[33] Terry P Orlando and Kevin A Delin. *Foundations of applied superconductivity*, volume 8. Addison-Wesley Reading, MA, 1991.

[34] SV Polonsky, VK Semenov, and PN Shevchenko. Pscan: personal superconductor circuit analyser. *Superconductor Science and Technology*, 4(11):667, 1991.

[35] Jan Pool, Retief Gerber, and Dirk Bull. Niocad—brief history and current activities. In *IEEE/CSC ESAS Eur. Supercond. News Forum*, volume 14, 2010.

[36] Mirko Prezioso, Farnood Merrikh-Bayat, BD Hoskins, GC Adam, Konstantin K Likharev, and Dmitri B Strukov. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature*, 521(7550):61, 2015.

[37] Steven T Ruggiero. *Superconducting devices*. Elsevier, 2013.

[38] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[39] Karl Steinbuch. Die lernmatrix. *Biological Cybernetics*, 1(1):36–45, 1961.

[40] I Synopsys. Hspice user's manual: Simulation and analysis. 2010.

[41] Naoki Takeuchi, Dan Ozawa, Yuki Yamanashi, and Nobuyuki Yoshikawa. An adiabatic quantum flux parametron as an ultra-low-power logic device. *Superconductor Science and Technology*, 26(3):035010, 2013.

[42] Michael Tinkham. *Introduction to superconductivity*. Courier Corporation, 2004.

[43] Emily Toomey, Murat Onen, Marco Colangelo, Brenden A Butters, Adam N McCaughan, and Karl K Berggren. Bridging the gap between nanowires and josephson junctions: a superconducting device based on controlled fluxon transfer across nanowires. *arXiv preprint arXiv:1810.09542*, 2018.

[44] Emily Toomey, Qing-Yuan Zhao, Adam N McCaughan, and Karl K Berggren. Frequency pulling and mixing of relaxation oscillations in superconducting nanowires. *Physical Review Applied*, 9(6):064021, 2018.

[45] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

[46] Ming-Jye Wang. Sinis josephson junction fabrication results. http://www.asiaa.sinica.edu.tw/ mingjye/dir-3/result-f.htm.

[47] SR Whiteley. Josephson junctions in spice3. *IEEE Transactions on Magnetics*, 27(2):2902–2905, 1991.

[48] Qing-Yuan Zhao, Adam N McCaughan, Andrew E Dane, Karl K Berggren, and Thomas Ortlepp. A nanocryotron comparator can connect single-flux-quantum circuits to conventional electronics. *Superconductor Science and Technology*, 30(4):044002, 2017.

[49] Qing-Yuan Zhao, Emily A Toomey, Brenden A Butters, Adam N McCaughan, Andrew E Dane, Sae-Woo Nam, and Karl K Berggren. A compact superconducting nanowire memory element operated by nanowire cryotrons. *Superconductor Science and Technology*, 31(3):035009, 2018.

[50] Di Zhu, Qing-Yuan Zhao, Hyeongrak Choi, Tsung-Ju Lu, Andrew E Dane, Dirk Englund, and Karl K Berggren. A scalable multi-photon coincidence detector based on superconducting nanowires. *Nature nanotechnology*, page 1, 2018.