ABSTRACT

Title of dissertation: SUPERCONDUCTING LOGIC CIRCUITS OPERATING WITH RECIPROCAL MAGNETIC FLUX QUANTA

Oliver Timothy Oberg,
Doctor of Philosophy, 2011

Dissertation directed by: Professor Fred Wellstood
Department of Physics

Complimentary Medal-Oxide Semiconductor (CMOS) technology is expected to soon reach its fundamental limits of operation. The fundamental speed limit of about 4 GHz has already effectively been sidestepped by parallelization. This increases raw processing power but does nothing to improve power dissipation or latency. One approach for increasing computing performance involves using superconducting digital logic circuits. In this thesis I describe a new kind of superconducting logic, invented by Quentin Herr at Northrop Grumman, which uses reciprocal pairs of quantized single magnetic flux pulses to encode classical bits. In Reciprocal Quantum Logic (RQL) the data is encoded in integer units of the magnetic flux quantum. RQL gates operate without the bias resistors of previous superconducting logic families and dissipate several orders of magnitude less power.

I demonstrate the basic operation of key RQL gates (AndOr, AnotB, Set/Reset) and show their self-resetting properties. Together, these gates form a universal logic set and provide memory capabilities. Experiments measuring the bit error rate of

the AndOr gate extrapolated a minimum BER of 10^{-480} and a BER of 10^{-44} with 30% margins on flux biasing.

I describe an analytic timing model for RQL gates which demonstrates the self-correcting timing features. From this model I derive equations for the timing behavior and operating limits. Using this timing model I ran simulations to determine correction factions for more accurate predictions at higher frequencies. Using these results, I also develop Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) models to describe the combinational logic of RQL gates.

To test the timing predictions of the timing model, I performed three experiments on $\mathrm{Nb/AlO_x/Nb}$ circuits at 4.2 K. The first measured the time of output. The second measured the operating margins of the circuit. The third measured the maximum frequency of operation for RQL circuits. Together, these three experiments showed quantitative agreement with the model for the timing output, qualitative agreement with the limits of operation, and a projected speed limit of 50 GHz for the Hypres $4.5\,\mathrm{kA/cm^2}$ process.

To power RQL circuits I describe a new design for power splitters and combiners which minimize standing waves. I describe a new kind of Wilkinson power splitter which required numerical optimization but proved to be adequate. I experimentally tested two new designs of the power splitter. Both showed less than 10% variation in standing waves between power splitter and combiner, making it adequate for RQL circuits. I also compared these results with the S-parameters of the power network, which also indicated that the design was adequate for RQL circuits.

Finally, I tested an 8-bit Kogge-Stone architecture carry-look ahead adder designed using VHDL models. The adder contained 815 Josephson junctions and was fully functional at 6.21 GHz with a latency of 1.25 clock cycles. The adder produced the correct logical output, had a measured optimal operating point within 8% of the optimal simulated operating point, and measured power margins of 1 dB. It operated best at the designed clock amplitude of $0.88I_c$ and dissipated 0.570 mW of power.

SUPERCONDUCTING LOGIC CIRCUITS OPERATING WITH RECIPROCAL MAGNETIC FLUX QUANTA

by

Oliver Timothy Oberg

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2011

Advisory Committee:
Professor Frederick Wellstood
Professor Anna Herr
Professor Christopher Lobb
Professor James Anderson
Dr. Benjamin Palmer
Professor Chris Davis

Enjoy the little things in life,

for one day you may look back

and realize they were the big things.

- Robert Brault

Acknowledgments

I have had a harder time properly giving thanks to the many people who made this thesis possible than writing the whole rest of the thesis. Instead, I will try to give acknowledgements as best I can in as short a space as possible, and trust that those whom I may not have explicitly mentioned know they have more gratitude than I can properly express.

First and foremost I'd like to thank Dr. Anna Herr, my research advisor first at UMCP and then at Northrop Grumman. Her guidance and patience are the foundation not only of my thesis but of my academic success so far. She has not only given me fantastic research opportunities beyond any I expected to see in graduate school but also taken a personal interest in my work and education. Even at the busiest times she never failed to help me through a problem and I never had to wait in idle frustration.

I also owe a great amount of appreciation to my academic advisor, Professor Fred Wellstood at UMCP. He was willing to step in for me at the university to take care of all University-related issues, and went above and beyond helping me review and edit this thesis. His input has been both quite helpful and educational.

Dr. Quentin Herr at Northrop Grumman has been as much a mentor to me as Anna or Fred. He has worked with me daily for almost three years and has been instrumental in my understanding of superconductivity and microwave physics. This thesis would not have been possible without his support, patience, and guidance.

There are many other individuals who have not only helped me greatly as

a graduate student but have contributed their ideas and work efforts to the work in this thesis. At Northrop Grumman, John Fusco has been pivotal in setting up my graduate studies here. Stephen Van Campen has likewise been a fantastic supporter in management without whom very little could have been accomplished. Dr. James Baumgartner and Dr. Aaron Pesetski have been fantastically helpful and supportive, always happy to explain concepts, provide feedback, and share a joke. Dr. Ofer Naaman has been part of the same work efforts as I have and has always been willing to help bridge the gaps in my understanding of microwave behavior and superconductivity. Steven Shauck has been a wonderful tutor to me in all things VHDL. He's probably forgotten more about VHDL than I will ever know, but has always been happy to find time in his very busy schedule to teach me about VHDL. Alex Ioanniadis, who has since gone off to graduate school himself, was a fantastic lab partner and experimentalist who taught me much of what I know about running experiments. Donald Miller has been a constant resource of knowledge and wisdom, always ready to help me hash out new and odd ideas and nitpick the details of old ideas.

Many people have contributed directly to the results in this thesis. Quentin Herr and Alex Ioanniadis performed the measurements shown in Figures 2.18 and 2.19. Ofer Naaman performed the numerical optimization of the design shown in Figure 4.10 and made the CAD layout of the same device as seen in Figure 4.17. Pavel Borodulin at Northrop Grumman supplied the analysis of the probe shown in Figure C.2. Steven Shauck supplied the final design of the adder of Chapter 6. Dr. John Pryzbysz at Northrop Grumman supplied the idea of using the spectrum analyzer to measure the side-band power of the CLA.

Although the bulk of my research was done at Northrop Grumman in Baltimore, most of my education was done at College Park, where there are a number of people I have to thank for their friendship, support, and camaraderie. I have to thank Dr. Rupert Lewis — now also at Northrop Grumman — and Dr. Gus Vlahacos for their kindness and support while I was starting my stint as a research assistant. Many thanks to Professor Ellen Williams for years of guidance, listening when others were busy or away.

I can not in good conscience fail to mention the handful of people outside of work, and outside of the university who gave me emotional support and friendship. I am lucky to have friends in almost every of the 50 states and in more countries than I can remember off the top of my head. But a special few never wavered from complete and permanent support in all parts of life, and without them my life couldn't have moved forward let alone would I have been able write this thesis.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ADS	Advanced Design System 2009 software
BCS	Bardeen-Cooper-Schrieffer
BER	Bit Error Rate
CLA	Carry Look Ahead
CMOS	Complimentary metal-oxide-semiconductor
GHz	Gigahertz
HPD	High Precision Devices
IREAP	Institute for Research in Electronics and Applied Physics
JJ	Josephson Junction
JTL	Josephson Transmission Line
NSA	National Security Agency
NRZ	Non-Return to Zero
PCB	printed circuit board
ps	picosecond
RQL	Reciprocal Quantum Logic
RSFQ	Resistive Single Flux Quantum
RSJ	Resistively Shunted Junction
RZ	Return to Zero
SFQ	Single Flux Quantum
SIS	Superconductor-Insulator-Superconductor
SNS	Superconductor-Normal Metal-Superconductor
SQUID	Superconducting QUantum Interference Device
std_ulogic	Synopsys extension to IEEE 1164, a VHDL class
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VLSI	Very Large Scale Integration
θ	Clock phase ($\theta = \omega\, t$ in most cases)
$\nu(\theta)$	Analytic Timing Model equation
ϕ	Phase across Josephson junction

List of Samples

M20LT	Monrovia 20 Logic Test
M20SR	Monrovia 20 Shift Register
M20PS	Monrovia 20 Power Splitter
N23PS	Norwalk 23 Power Splitter
N22TE	Norwalk 22 Timing Experiment
N21CLA	Norwalk 21 Carry Look Ahead Adder

Chapter 1

Introduction to Superconductivity and Josephson Junctions

1.1 Overview

Nearly 200 years passed between Franklin's discovery of electricity and the development of the first electronic computer [1, 2]. Superconductivity was discovered in 1911 by Heike Kamerlingh Onnes [3] and only began to impact computing about 50 years later [4]. In 1962 Brian David Josephson postulated the Josephson effect, which would lead to the invention of the dc SQUID two years later [5]. By the mid-1980s IBM terminated a major effort to build a computer using superconductivity. Shortly thereafter Josephson junctions began being considered for reversible computation [6] and used in Resistive Single Flux Quantum digital circuits [4]. Now, one century after the discovery of superconductivity, contemporary semiconductor-based computation seems to be approaching a fundamental limit [7] and this raises the possibility that a new generation of computers based on superconductivity and Josephson junctions may arise to push technology forward [8].

One technology potentially capable of pushing computation forward is Reciprocal Quantum Logic (RQL), the subject of my research over the past few years. The goal of this research was to demonstrate the feasibility of using RQL for very high speed and very low power computation. This thesis has three main parts. First, I provide a basic overview of superconductivity and Josephson junctions (Chapter 1).

This overview is far from exhaustive but serves to highlight aspects that are most important to the subsequent discussion of RQL. The first part concludes with an introduction to RQL (Chapter 2), which is where my own work begins. Next, in the second part, I describe my research into the behavior of junctions using high-level simulations in Very High Speed Integrated Circuit Hardware Description Language (VHDL). In Chapter 3, I derive an analytic model for the timing behavior of Josephson junctions in RQL circuits. After I verify this model in simulations, I proceed to cast RQL into the industry-standard VHDL. Chapter 4 is a departure from the previous topics and describes the development of a new power network for RQL, but together chapters 2–4 provide the basis for design of functioning circuits. In the third part, I describe my experiments testing the timing behavior (Chapter 5) and a fully functioning 8-bit adder (Chapter 6). Finally, in Chapter 7, I conclude with a brief summary of my main results and make some suggestions for future work.

1.2 Superconductivity

Following the discovery of superconductivity in 1911 many attempts were made to understand the phenomenon. In the Drude model of conduction in normal metals current density is proportional to the average velocity of electrons [9], which accelerate under an electric field over a distance l until colliding with defects and slowing down. A stable current is reached when the average deceleration due to collisions matches the acceleration due to the electric field. In the limit $l \to \infty$ infinite conductivity would result. However, in the 1930s it was found a superconductor is not merely a metal with infinite conductivity. Superconductors exhibit new behavior that ultimately required new physics to be understood.

1.2.1 London Equations

Around the time superconductivity was being discovered, quantum mechanics was being developed. In quantum mechanics the *canonical* momentum of a particle in a magnetic field is given by

$$\vec{p} = m\vec{v} + qe\vec{A}, \tag{1.1}$$

where m is the particle's mass, q is its charge, e is defined as the magnitude of the charge of an electron ($+1.609 \times 10^{-19}$ C), and $\vec{v}$ and $\vec{A}$ are the velocity and magnetic vector potential. If one assumes that in the ground state of a system the (local) average $\langle \vec{p} \rangle = 0$ then the current density J_s can be expressed as

$$\vec{J}_s = n_s e \langle \vec{v}_s \rangle = -\frac{n_s e^2}{m}\vec{A} = -\frac{\vec{A}}{\Lambda}. \tag{1.2}$$

Here the s-subscript refers to *superconducting* currents and electrons and $\Lambda = m/n_s e^2$. We can also define $\Lambda = \lambda^2$; the meaning of λ will become clearer, but for now we note that it has dimensions of length. Taking the time derivative and then the curl of (1.2), one can show that this leads to the London equations [9, 10] for the electric field $\vec{E} = -\partial\vec{A}/\partial t$ and magnetic field $\vec{H} = \nabla \times \vec{A}$

$$\vec{E} = \frac{\partial}{\partial t}(\Lambda \vec{J}_s), \tag{1.3}$$

$$\vec{H} = -\nabla \times (\Lambda \vec{J}_s). \tag{1.4}$$

Finally, using Maxwell's equation $\nabla \times \vec{H} = \vec{J}_s$ on (1.4) gives

$$\nabla^2 \vec{H} = \frac{\vec{H}}{\lambda^2}, \tag{1.5}$$

which implies that λ is the characteristic length scale for the penetration of magnetic field into a bulk superconductor.

Equations (1.3) – (1.5) were first obtained by Fritz and Heinz London in 1935 [10]. The original London equations were phenomenological and Λ was simply a fitting parameter. Two insightful results come from this very cursory derivation. Equation (1.3) implies that the current increases in time for a static electric field. Meanwhile, (1.4) implies that magnetic fields are expelled from the interior of superconductors within a characteristic length λ. Note also that the value of n_s is limited on the upper end by the total number of charges in the metal. It can be seen from energy considerations that (1.3) and (1.5) imply an upper limit on the current density $\vec{J}_s$. In addition, in a wire carrying a current, the magnetic field generated by the current will be constrained to a depth λ in the wire. If the current gets too large, the magnetic forces from the current would drive charge into the interior of the wire, destroying the superconductivity. These rough phenomenological considerations reveal some of the major features of superconductivity. However, the insight they provide of the superconducting state is limited. For a fuller understanding, I turn to the theory developed by Bardeen, Cooper, and Schrieffer in 1957 [11].

I have three goals for this section. First, I show that the superconducting state exists at any temperature below the transition temperature. Second, I show that quasiparticles in a superconductor have an energy that is at least as large as the superconducting energy gap. Finally, the third and most important point is to develop an understanding of the I-V curve of a Josephson Tunnel junction, as this will be the basis for much of the rest of the thesis.

1.2.2 BCS Theory

In superconducting materials and at finite temperatures, ordinary unpaired electrons are present as well as superconducting Cooper pairs [9]. Unpaired electrons yield a normal current component and follow Fermi-Dirac statistics. Two unpaired electrons will generally not have identical energies (with the exception of spin pairs) and consequently their quantum mechanical phase will change at different rates. In contrast, Cooper pairs follow Bose-Einstein statistics and can have identical energies and phase. In conventional BCS superconductors, Cooper pairs are formed by the interaction of electrons mediated though the exchange of phonons. Individual Cooper pairs are much larger than the mean spacing between pairs [9] and the pairs maintain phase coherence amongst each other by the large amount of overlap between their wave functions [9].

1.2.2.1 Cooper Pairs

Since electrons are charged they exert a Coulomb force on the semi-stationary nuclei in a metal. This force can scatter the electron and perturb nuclei from their equilibrium positions. The perturbations of positive nuclei by a scattered electron can attract other electrons, thus resulting in a net attractive potential V between two electrons despite the presence of electron-electron Coulomb repulsion. In a superconductor this attraction leads to electrons pairing up. The general wave function for a Cooper pair is

$$\psi_0(\vec{r}_1, \vec{r}_2) = \sum_{\vec{k}} g_{\vec{k}}\, \mathrm{e}^{i\vec{k}\cdot\vec{r}_1}\, \mathrm{e}^{-i\vec{k}\cdot\vec{r}_2} \chi_1 \chi_2, \tag{1.6}$$

where $\vec{r}_1$ and $\vec{r}_2$ are the positions of the first and second electron, respectively, $g_{\vec{k}}$ is the weighing factor of the orbital wave function, $\vec{k}$ is the wave vector, and χ_1 and χ_2 are spin functions for the first and second electron, respectively. This wave function can be recast into a form that is explicitly anti-symmetric in $\vec{r}_1$ and $\vec{r}_2$ by considering the distance between a pair $\vec{r}_1 - \vec{r}_2$. We can write in general

$$\psi_0(\vec{r}_1 - \vec{r}_2) = \left[\sum_{\vec{k} > \vec{k}_F} g_{\vec{k}} \cos \vec{k} \cdot (\vec{r}_1 - \vec{r}_2) \right] \sigma_{12}^{singlet} \tag{1.7}$$

for the singlet state in conventional BCS theory [11]. The sum in (1.7) is only over wave vectors that have lengths greater than the Fermi wave vector, for reasons which will become apparent shortly. $\sigma_{12}^{singlet}$ is the spin part of the singlet wave function and it is anti-symmetric under exchange of the electrons.

Inserting (1.7) into Schrödinger's equation gives a relationship between the energy E and the interaction potential V [9]:

$$\frac{1}{V} = \sum_{\vec{k} > \vec{k}_F} \frac{1}{2\epsilon_{\vec{k}} - E} \tag{1.8}$$

where $\epsilon_{\vec{k}} = \hbar^2 k^2 / 2m$. Equation (1.8) can be evaluated as an integral from the Fermi energy E_F to a higher energy $E_F + \hbar\omega_c$. One finds

$$\frac{1}{V\,N(0)} = \int_{E_F}^{E_F + \hbar\omega_c} \frac{\mathrm{d}\epsilon}{2\epsilon - E} = \frac{1}{2} \ln \frac{2E_F - E + 2\hbar\omega_c}{2E_F - E}, \tag{1.9}$$

where $N(0)$ is the density of electron states at the Fermi level. In the weak-coupling approximation $V\,N(0) \ll 1$ one finds

$$E \approx 2E_F - 2\hbar\omega_c\,\mathrm{e}^{-2/V\,N(0)}. \tag{1.10}$$

This result shows that the energy of a pair is reduced by the interaction in a non-perturbative manner and bound states (pairs) can exist no matter how small V becomes.

1.2.2.2 Ground State

To get further understanding of the behavior of a superconductor we apply *second quantization* [9]. Let $|F\rangle$ be the state of a metal in which all the electron states below the Fermi surface are occupied. Then the wave function for the state ψ_0 of a superconductor becomes

$$|\psi_0\rangle = \sum_{\vec{k}>\vec{k}_F} g_{\vec{k}}\, c^*_{\vec{k}\uparrow}\, c^*_{-\vec{k}\downarrow}\, |F\rangle \tag{1.11}$$

where $c^*_{\vec{k}\,\sigma}$ and $c_{\vec{k}\,\sigma}$ are the creation and annihilation operators for a pair with momentum $\vec{k}$ and spin σ and the $g_{\vec{k}}$ are weighing factors for the pairs, with the anti-commutation relations $\left\{c_{\vec{k}\,\sigma},\, c^*_{\vec{k}'\,\sigma'}\right\} = \delta_{kk'}\delta_{\sigma\sigma'}$ and $\left\{c_{\vec{k}\,\sigma},\, c_{\vec{k}'\,\sigma'}\right\} = 0$. The number of electrons which wave vector k and spin σ is then given by the operator $n_{k\sigma} = c^*_{k\sigma}c_{k\sigma}$ which gives unity when operating on a filled state and zero when operating on an unoccupied state.

In a macroscopic superconductor at sufficiently low temperature the fluctuations about the ground state will be small and Bardeen, Cooper, and Schrieffer were able to apply a mean-field approach [9]. They wrote the ground state as a product of superposition states with differing momenta:

$$|\psi_G\rangle = \prod_{\vec{k}_1,\ldots,\vec{k}_M} \left(u_{\vec{k}} + v_{\vec{k}}\, c^*_{\vec{k}\uparrow}\, c^*_{-\vec{k}\downarrow}\right) |\psi_0\rangle\,, \tag{1.12}$$

with $\left|v_{\vec{k}}\right|^2 + \left|u_{\vec{k}}\right|^2 = 1$. $\left|v_{\vec{k}}\right|^2$ is the probability of the state $\left(\vec{k}\uparrow, -\vec{k}\downarrow\right)$ being occupied and $\left|u_{\vec{k}}\right|^2$ is the probability of it being empty. We can learn a bit about the ground state ψ_G if we assume $v_{\vec{k}}$ and $u_{\vec{k}}$ differ by a set phase. With this assumption, we can rewrite (1.12) as

$$|\psi_G\rangle = \prod_{\vec{k}_1,\ldots,\vec{k}_M} \left(\left|u_{\vec{k}}\right| + \left|v_{\vec{k}}\right| \mathrm{e}^{i\varphi}\, c^*_{\vec{k}\uparrow}\, c^*_{-\vec{k}\downarrow}\right) |\psi_0\rangle\,. \tag{1.13}$$

The phase φ turns out to be the order parameter of the superconductor, and it obeys an uncertainty relationship with the number of pairs N [9]:

$$\Delta N \Delta\varphi \geq 1. \tag{1.14}$$

1.2.2.3 Pairing Hamiltonian

To arrive at (1.12) for the state ϕ_0, Bardeen, Cooper, and Schrieffer wrote a simplified Hamiltonian $\mathcal{H}$ that included a pairing-interaction term

$$\mathcal{H} = \sum_{\vec{k}\sigma} \epsilon_k\, n_{\vec{k}\sigma} + \sum_{\vec{k}\vec{k}'} V_{\vec{k}\vec{k}'}\, c^*_{\vec{k}\uparrow}\, c^*_{-\vec{k}\downarrow}\, c_{-\vec{k}'\downarrow}\, c_{\vec{k}'\uparrow}. \tag{1.15}$$

The first term is the energy ϵ_k of a Cooper pair with momentum k and spin σ. The second term describes the energy gained by the annihilation of a Cooper pair consisting of electrons with momentum $\vec{k}'$ and the creation of a pair with momentum $\vec{k}$, where $V_{\vec{k}\vec{k}'}$ is the scattering matrix element. Interactions between electrons with different momenta $\vec{k}$ do not play a role in BCS theory but may in other applications. The ground state (1.12) and Hamiltonian (1.15) can then be substituted into Schrödinger's equation. The ground state energy and $g_{\vec{k}}$ can be found by a canonical transformation. Following Tinkham [9] we define $b_{\vec{k}} = \langle c_{-\vec{k}\downarrow}\, c_{\vec{k}\uparrow}\rangle$ and write

$$c_{-\vec{k}\downarrow}\, c_{\vec{k}\uparrow} = b_{\vec{k}} + \left(c_{-\vec{k}\downarrow}\, c_{\vec{k}\uparrow} - b_{\vec{k}}\right). \tag{1.16}$$

The ideas is that the term in parentheses should be small. I can also define $\Delta_{\vec{k}} = -\sum_{\vec{k}'} V_{\vec{k}\vec{k}'} \langle c_{-\vec{k}\downarrow}\, c_{\vec{k}\uparrow}\rangle$ and $\xi_{\vec{k}} = \epsilon_{\vec{k}} - E_F$, neglecting terms that are quadratic in the term in parentheses above. Then the Hamiltonian becomes [9]

$$\mathcal{H} = \sum_{\vec{k}\sigma} \xi_{\vec{k}}\, c^*_{\vec{k}\sigma}\, c_{\vec{k}\sigma} - \sum_{\vec{k}} \left(\Delta_{\vec{k}}\, c^*_{\vec{k}\uparrow}\, c^*_{-\vec{k}\downarrow} + \Delta^*_{\vec{k}}\, c^*_{-\vec{k}\downarrow}\, c^*_{\vec{k}\uparrow} - \Delta_{\vec{k}}\, b^*_{\vec{k}}\right) \tag{1.17}$$

This can be diagonalized if we define new creation operator $\eta^*_{\vec{k}}$ and annihilation operator $\gamma_{\vec{k}}$ from:

$$c_{\vec{k}} = u^*_{\vec{k}}\, \gamma_{\vec{k}} + v_{\vec{k}}\, \eta^*_{\vec{k}} \tag{1.18}$$

$$c^*_{-\vec{k}} = -v^*_{\vec{k}}\, \gamma_{\vec{k}} + u_{\vec{k}}\, \eta^*_{\vec{k}} \tag{1.19}$$

where the $v_{\vec{k}}$ and $u_{\vec{k}}$ satisfy

$$\left|v_{\vec{k}}\right|^2 = \frac{1}{2}\left(1 - \frac{\xi_{\vec{k}}}{E_{\vec{k}}}\right) \tag{1.20}$$

$$\left|u_{\vec{k}}\right|^2 = \frac{1}{2}\left(1 + \frac{\xi_{\vec{k}}}{E_{\vec{k}}}\right) \tag{1.21}$$

Finally (1.15) and (1.17) can be put in the form [9]:

$$\mathcal{H} = \sum_{\vec{k}} \left(\xi_{\vec{k}} - E_{\vec{k}} + \Delta_{\vec{k}}\, b^*_{\vec{k}}\right) + \sum_{\vec{k}} E_{\vec{k}} \left(\gamma^*_{\vec{k}}\gamma_{\vec{k}} + \eta^*_{\vec{k}}\eta_{\vec{k}}\right) \tag{1.22}$$

This Hamiltonian has two terms. The first term is just the condensation energy and is a constant. The second term accounts for the energy due to quasiparticles with energy $E_{\vec{k}} = \left(\xi_{\vec{k}}^2 + \left|\Delta_{\vec{k}}\right|^2\right)^{1/2}$. $\Delta_{\vec{k}}$ is the energy decrease when a Cooper pair forms. A superconducting state will be stable if the energy decrease for a pair forming is greater than that required to leave the Fermi surface. Once a pair is formed, 2Δ is the energy necessary to break a pair into two quasiparticles.

1.2.2.4 Density of States

Quasiparticles behave much like electrons in a normal metal. The density of states $N_s(E)$ of the quasiparticles is related to the density of states of the electrons in the normal metal $N_n(\xi)$ by $N_s(E)\mathrm{d}E = N_n(\xi)\mathrm{d}\xi$. For energies small compared to the Fermi energy the number of normal electron states $N_n(\xi) = N(0)$ can be taken as constant [9]. One then finds:

$$\frac{N_s(E)}{N(0)} = \frac{\mathrm{d}\xi}{\mathrm{d}E} = \begin{cases} E/\sqrt{E^2 - \Delta^2} & \text{if } E > \Delta \\ \\ 0 & \text{if } E < \Delta \end{cases} \tag{1.23}$$

The dependance of N_s on the quasiparticle energy E is directly manifest in the tunneling behavior between superconductors. The quasiparticle current I that flows between two superconductors with a voltage V between them can be written as [9]:

$$
\begin{aligned}
I &\simeq A\int_{-\infty}^{+\infty} N_1(E)N_2(E+eV)(f(E)-f(E+eV))\,\mathrm{d}E \\
&= A\int_{-\infty}^{+\infty} \frac{N_{s1}(E)}{N_1(0)}\frac{N_{s1}(E+eV)}{N_1(0)}(f(E)-f(E+eV))\,\mathrm{d}E \\
&= A\int_{-\infty}^{+\infty} \frac{|E|}{\sqrt{E^2-\Delta_1^2}}\frac{|E+eV|}{\sqrt{(E+eV)^2-\Delta_2^2}}(f(E)-f(E+eV))\,\mathrm{d}E, \qquad (1.24)
\end{aligned}
$$

where $f(E)$ is the Fermi function (probability that a quasiparticle state at energy E is occupied) and A is a constant that depends on the junctions barrier and other details such as temperature.

This integral (1.24) can be evaluated numerically or treated analytically for $T = 0$. Instead, I consider an analysis of the IV curve by Likharev. To proceed, first transform the phase $\varphi(t)$ into its Fourier components $W(\omega)$ by using

$$
\mathrm{e}^{i\varphi/2} = \mathrm{e}^{i\Phi/2}\int_{-\infty}^{+\infty} W(\omega)\mathrm{e}^{i\omega t}\mathrm{d}\omega
$$

where the time derivative of Φ is related to the average voltage $\bar{V}$ by $\dot{\Phi} = (2\,e/\hbar)\bar{V}$. Barone *et al.* then define the supercurrent component $I_S(t)$ and normal current component $I_N(t)$ as [4]

$$
I_S(t) = \mathrm{Im}\left(\int_{-\infty}^{+\infty}\mathrm{d}\omega_1\int_{-\infty}^{+\infty}\mathrm{d}\omega_2\,W(\omega_1)W(\omega_2)I_p\left(\omega_2+\frac{\omega_J}{2}\right)\mathrm{e}^{i(\omega_!+\omega_2)t+i\Phi}\right), \quad (1.25\text{a})
$$

$$
I_N(t) = \mathrm{Im}\left(\int_{-\infty}^{+\infty}\mathrm{d}\omega_1\int_{-\infty}^{+\infty}\mathrm{d}\omega_2\,W(\omega_1)W^*(\omega_2)I_q\left(\omega_2+\frac{\omega_J}{2}\right)\mathrm{e}^{i(\omega_!+\omega_2)t}\right). \quad (1.25\text{b})
$$

In turn, we define I_p and I_q as the Green's functions for the superconducting electrodes. These functions do not depend on the phase dynamics of the junction, only the junction itself. They characterize the junction fully and are given by [4]

$$\begin{aligned} I_p(\omega) &= \frac{1}{G_N(2\pi^2 e)} \int_{-\infty}^{+\infty} \mathrm{d}\omega_1 \int_{-\infty}^{+\infty} \mathrm{d}\omega_2 \left(\tanh \frac{\hbar\omega_1}{2k_B T} + \tanh \frac{\hbar\omega_2}{2k_B T} \right) \qquad (1.26a) \\ &\qquad \times \frac{\mathrm{Im}\,(F_1(\omega_1))\,\mathrm{Im}\,(F_2(\omega_2))}{\omega_1 + \omega_2 - w + j0}, \\ I_q(\omega) &= \frac{1}{G_N(2\pi^2 e)} \int_{-\infty}^{+\infty} \mathrm{d}\omega_1 \int_{-\infty}^{+\infty} \mathrm{d}\omega_2 \left(\tanh \frac{\hbar\omega_1}{2k_B T} + \tanh \frac{\hbar\omega_2}{2k_B T} \right) \qquad (1.26b) \\ &\qquad \times \frac{\mathrm{Im}\,(G_1(\omega_1))\,\mathrm{Im}\,(G_2(\omega_2))}{\omega_1 + \omega_2 - w + j0}, \end{aligned}$$

where the subscripts 1 and 2 refer to the left and right superconducting banks of the junction. The functions F and G can be derived from BCS theory and are given by [4]

$$F(\omega) = \frac{\pi\Delta(T)}{\sqrt{\Delta^2(T) - \hbar^2(\omega + j0)^2}}, \qquad (1.27)$$

$$G(\omega) = \frac{\pi\hbar\omega}{\sqrt{\Delta^2(T) - \hbar^2(\omega + j0)^2}}. \qquad (1.28)$$

The term $\omega + j0$ is real. The latter part of the expression is a remnant from the complex analysis used to derive the expression and included here for clarity. Barone and Paterno provide a detailed analysis including this term [6]. Substituting (1.27) and (1.28) into (1.26a) and (1.26b) one finds relations for the real and imaginary components of I_p and I_q. (See Fig. 1.1.)

$$\operatorname{Re} I_p(\omega) = \frac{\Delta(0)}{e\,R_N} \times \begin{cases} K(x) & \text{if } x < 1 \\ x^{-1}\,K(x^{-1}) & \text{if } x > 1 \end{cases} \tag{1.29a}$$

$$\operatorname{Im} I_p(\omega) = \frac{\Delta(0)}{e\,R_N} \times \begin{cases} 0 & \text{if } x < 1 \\ x^{-1}\,K(x') & \text{if } x > 1 \end{cases} \tag{1.29b}$$

$$\operatorname{Re} I_q(\omega) = \operatorname{sign}(\omega)\frac{\Delta(0)}{e\,R_N} \times \begin{cases} K(x) - 2E(x) & \text{if } x < 1 \\ (2x - x^{-1}\,K(x^{-1}) - 2x\,E(x^{-1}) & \text{if } x > 1 \end{cases} \tag{1.29c}$$

$$\operatorname{Im} I_q(\omega) = \operatorname{sign}(\omega)\frac{\Delta(0)}{e\,R_N} \times \begin{cases} 0 & \text{if } x < 1 \\ 2x\,E(x') - x^{-1}\,K(x') & \text{if } x > 1 \end{cases} \tag{1.29d}$$

where E and K are complete elliptic integrals of the first and second kind, respectively. I also define $x = |\omega|/\omega_g$, the gap frequency $\omega_g = 2\Delta(T)/\hbar$, and $x' = \sqrt{1 - x^{-2}}$. Equations (1.29a)–(1.29d) are true only for $T = 0$; at higher temperatures I_p and I_q must be calculated numerically.

By evaluating these equations at arbitrary temperature T, we can find an expression for $V_c = I_c R_N$ for SIS junctions in terms of the gap energy Δ. Here I_c is the critical current and R_N is the normal resistance of the junction. Substituting (1.27) and (1.28) into (1.26a) and (1.26b) one finds

$$V_c = I_c R_N = \frac{\pi}{2\,e}\Delta(T)\tanh\frac{\Delta(T)}{2k_B T} \tag{1.30}$$

for SIS junctions [4].

Figure 1.2 shows the average current $\bar{I}$ for a constant voltage V found from

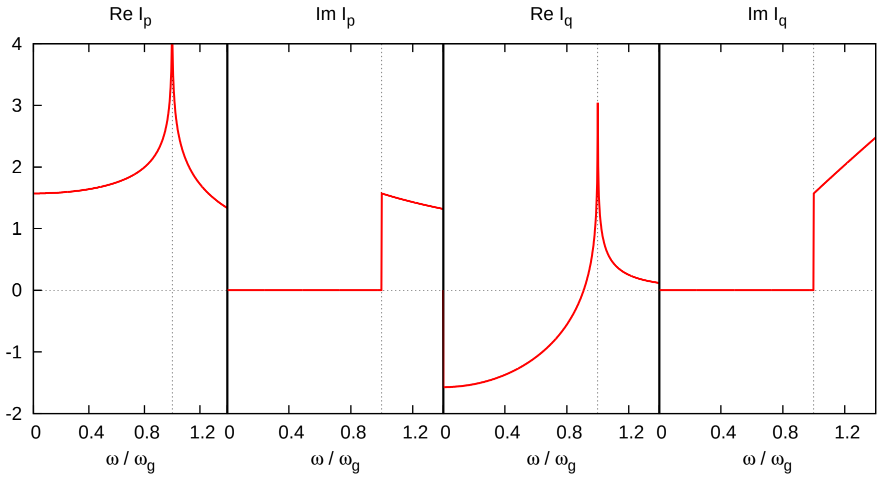


Figure 1.1: Green's Functions I_p and I_q for Josephson Junction at $T = 0$. Real and imaginary components of the Cooper pair and quasiparticle Green's functions from (1.29a) – (1.29d) in superconductor-to-superconductor tunneling. Each components shows a clear transition at the gap frequency ω_c. Below this frequency $\operatorname{Im} I_p = 0$ and $\operatorname{Im} I_q = 0$; no quasiparticles tunnel through the barrier.

(1.29a) – (1.29d). Below V_c only a relatively small number of thermally excited quasiparticles tunnel and the average current is low. Only past a critical voltage $V_c \approx 2\Delta/e$ does current flow. Above V_c the current is similar to that of a non-superconducting tunnel junction. This behavior is important for Josephson junction dynamics, which shall be the topic of the rest of this chapter. However, it is important to note that it does not include supercurrent flow at $V = 0$. $V = 0$ supercurrent flow is considered in the next section.

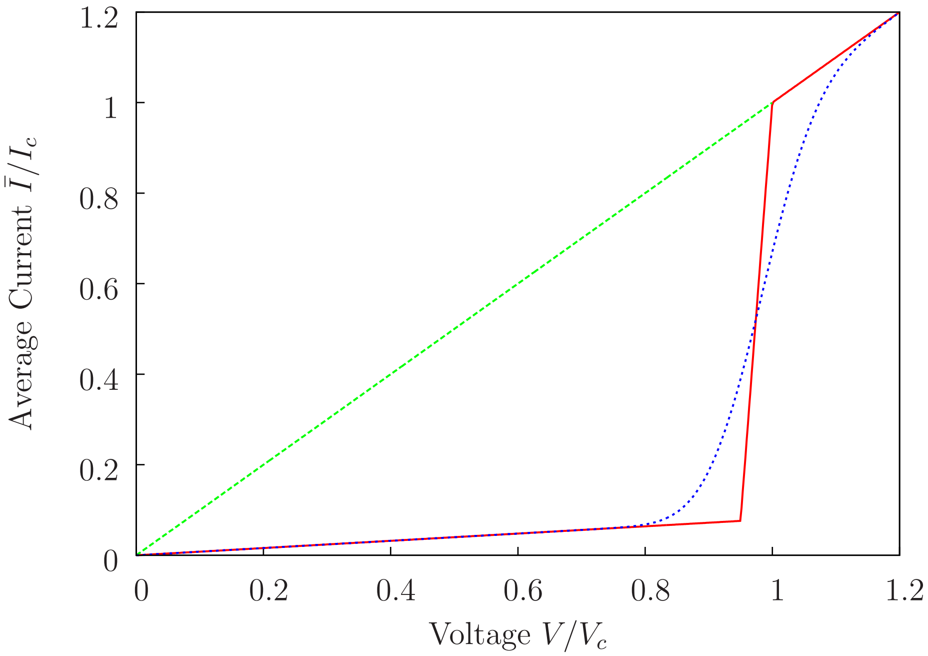


Figure 1.2: Superconductor-Superconductor Tunneling I-V Curve for $V > 0$. The time-averaged current from (1.24) is plotted at zero temperature (red). Blue curve shows the effects of non-zero temperature. Green curve shows linear resistive I-V relationship of the normal state. Below V_c a nearly negligible current of unpaired electrons flows in the superconducting state. Above this critical potential, Cooper pairs break apart and the I-V curve becomes similar to the ohmic resistance curve.

1.3 Josephson Junctions

In this section, I provide a brief review of the behavior of Josephson junctions. In general the equations of motion for junctions include terms to account for thermal noise. For my purposes, these noise considerations are mostly secondary and are deferred to later chapters where I discuss experimental data. A review of the complete range of behaviors and applications for Josephson junctions is well beyond the scope of this thesis. Instead, I highlight some features of junctions that are of particular importance for RQL: the generation of quantized single flux quantum pulses, the finite number of flux states in the single-junction interferometer, and the transport of single flux quantized pulses through transmission lines.

Josephson tunneling was predicted in 1962 by B. D. Josephson [12] and experimentally demonstrated in 1964 [13]. Josephson junctions come in many forms. All junctions have in common two superconducting electrodes which are separated by a region that impedes current. This can be a physical narrowing of the superconductor itself, a normal metal region, or a thin insulator though which the Cooper pairs can tunnel. My focus is exclusively on the Superconductor-Insulator-Superconductor (SIS) type of junctions, though Superconductor-Normal Metal-Superconductor (SNS) types and Superconductor-Ferromagnet-Superconductor (SFS) junctions may play a role in RQL in the future. In particular, the junctions of interest for RQL are SIS junctions with Niobium superconductors and Al_2O_3 tunnel barriers. The SIS type junction can also have a shunt resistor connected across the junction. As we will see, the properties of shunt resistors effect the dynamics of junctions.

1.3.1 Josephson Equations

Given superconductivity and a few assumptions, the Josephson equations can be obtained from the Schrödinger equation. Consider a state for which the magnitude of the order parameter $|\Psi|$ does not change in time. If the phase can change in time, then we can write the order parameter as:

$$\Psi(\vec{r},t) = |\Psi|\, \mathrm{e}^{i\varphi(\vec{r},t)}. \tag{1.31}$$

The Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\,|\Psi\rangle = \mathcal{H}\,|\Psi\rangle \tag{1.32}$$

then becomes:

$$\hbar\,\dot{\varphi} = E, \tag{1.33}$$

where E is the energy of the state $|\Psi\rangle$. I define a phase difference across the junction between superconductor 1 and 2 as $\phi = \varphi_1 - \varphi_2$, where φ_1 is the phase in superconductor 1 and φ_2 is the phase in superconductor 2. The current that flows through the junction from 1 to 2 will depend only on this phase difference, $I_S = I_S(\phi)$. I now make an assumption that the current is zero for directly connected superconductors with no phase difference. (Relaxation of this assumption leads to another theory of junctions not used in this thesis.) Since Ψ is periodic in φ with period 2π, we expect that $I_S(0) = I_S(2\pi n) = I_S(\pi + 2\pi n) = 0$ for any integer n. For SIS junctions these physical requirements lead to the lowest order terms in the *first Josephson relation*,

$$I_s(t) = I_c\sin(\phi(t)), \tag{1.34}$$

where I_c is the critical current of the junction, which is determined by the physics of the materials and the geometry of the junction. It is the maximum supercurrent that can flow through the junction. (Higher order terms such as $\sin 2\phi$, $\sin 4\phi$, *etc.* also satisfy these requirements but in general are not experimentally significant for SIS junctions used in RQL [9].)

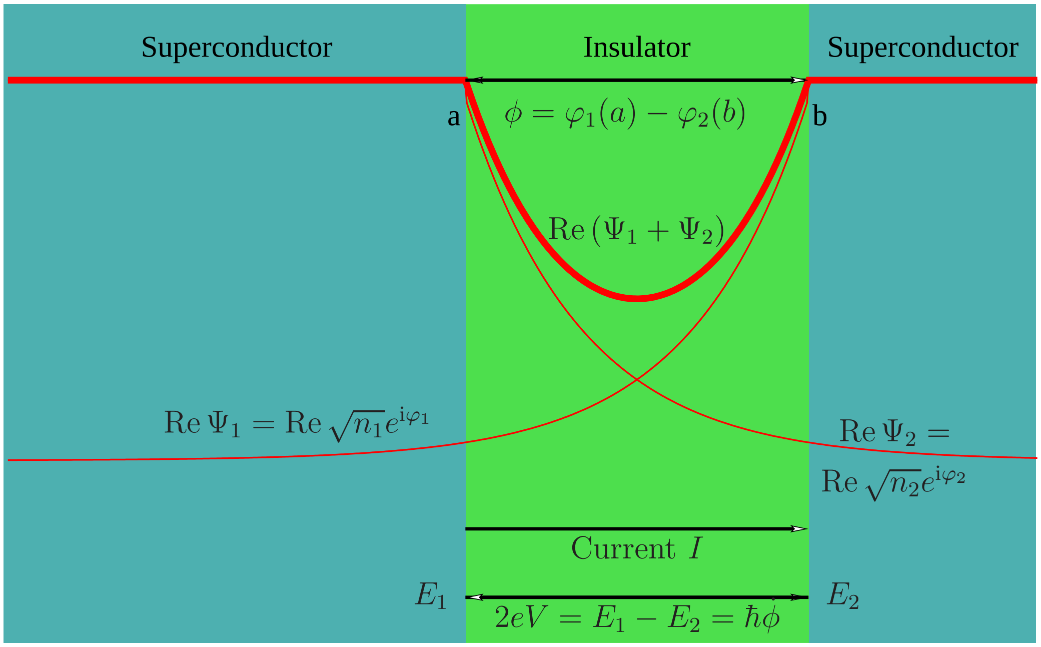

Superconductor
Insulator
Superconductor
a
$\phi = \varphi_1(a) - \varphi_2(b)$
b
$\mathrm{Re}\,(\Psi_1 + \Psi_2)$
$\mathrm{Re}\,\Psi_1 = \mathrm{Re}\,\sqrt{n_1}e^{\mathrm{i}\varphi_1}$
$\mathrm{Re}\,\Psi_2 =$
$\mathrm{Re}\,\sqrt{n_2}e^{\mathrm{i}\varphi_2}$
Current I
E_1
E_2
$2eV = E_1 - E_2 = \hbar\dot{\phi}$

Figure 1.3: Superconductor-Insulator-Superconductor Tunneling. A wave function Ψ tunneling from Superconductor 1 through an insulating barrier to Superconductor 2 has densities and phases n_1 and φ_1 in Superconductor 1 and n_2 and φ_2 in Superconductor 2. A potential difference between E_1 in Superconductor 1 and E_2 in Superconductor 2 develops if the phase difference $\phi = \varphi_1 - \varphi_2$ changes in time.

Finally, Josephson found a relationship between the electrical potential V between two superconducting nodes with phase difference $\phi = \varphi_1 - \varphi_2$ between them and the energy E_1 and E_2 of the two nodes, $2eV = E_1 - E_2 = \hbar\dot{\phi}$. (See Fig. 1.3.) We can write the *second Josephson relation* as

$$V(t) = \frac{\hbar}{2e}\frac{\partial\phi(t)}{\partial t} = \frac{\Phi_0}{2\pi}\frac{\partial\phi(t)}{\partial t} = \frac{\hbar\omega_J}{2\,e}, \tag{1.35}$$

where Φ_0 is the flux quantum $\Phi_0 = 2.062\,\mathrm{mV}\cdot\mathrm{ps}$. The $2eV$ in (1.35) explicitly shows that the voltage is associated with an energy $\hbar\omega_J$ per charge of a Cooper pair $-2e$. Here, ω_J is the angular frequency for the phase difference across the junction.

The quantity Φ_0 is not merely a unit of convenience. The order parameter Φ must be single-valued at every point, meaning that the phase change $\Delta\Phi$ in going one or more times around the loop must take on values that can differ only by $2\pi n$. It can be shown that this limits the total flux through the loop to values of $\Phi = n\Phi_0$ [4]. By integrating (1.35) with respect to time over one cycle of phase change, we get

$$\int V(t)\,\mathrm{d}t = \Phi_0 \int \frac{\mathrm{d}\phi}{2\pi} = \Phi_0. \tag{1.36}$$

Ultimately, it can be shown that this leads to the "area" under a voltage pulse from a junction being quantized exactly to Φ_0 for a single complete switching event. This is a single quantum of magnetic flux, and the voltage pulse associated with it is an SFQ pulse. This result can be related to the flux Φ in a closed loop using $\Phi = \int L\,\mathrm{d}I = \int V\,\mathrm{d}t = \Phi_0$. Any inductive loop with a junction which switches by $2n\pi$ will change the flux by exactly an integer multiple n of Φ_0. This result has profound implications for superconductors in the presence of magnetic fields. For example, from (1.36) we can see that magnetic fields are expelled from the interior of any superconductor, a phenomenon knows as the Meissner effect, and that an inductor loop with a junction in it contains $n\Phi$ magnetic flux.

1.3.2 RSJ Model

In an SIS junction the capacitance occurs because the two superconducting electrodes are separated by an insulating barrier of small thickness, thus taking on the geometry of a capacitor. Figure 1.4 shows the electrical representation of the

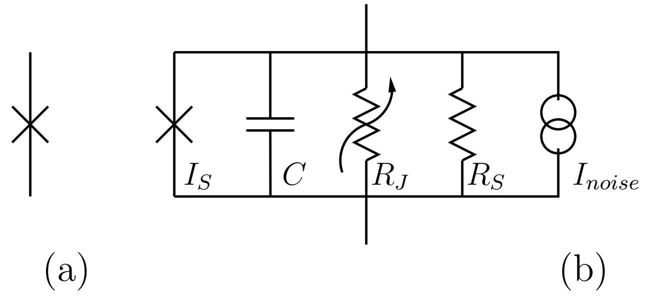


Figure 1.4: Equivalent electrical circuit of a Josephson Junction in the RSJ model. (a) Ideal Josephson junction symbol. (b) Equivalent circuit of real Josephson junctions. The Josephson junction can be treated as a parallel combination of an ideal Josephson junction with the supercurrent I_S, a capacitor C, and a non-linear resistor R_J, a shunt resistor R_S and a noise current I_{noise}.

Josephson junction in the resistively shunted junction (RSJ) model. A junction is represented electrically as an ideal Josephson junction shunted by a resistor (linear or non-linear) and a capacitance. A bias current I can be sent through the junction's components and can be thought of being composed of a displacement current, a normal current, and a supercurrent. The total current through the junction can then be written as

$$\frac{\hbar\, C}{2\, e}\frac{\mathrm{d}^2\phi(t)}{\mathrm{d}t^2}+\frac{\hbar}{2\, e\, R_N}\frac{\mathrm{d}\phi(t)}{\mathrm{d}t}+I_c\sin(\phi(t))=I+I_{noise} \tag{1.37a}$$

This can be put in reduced form:

$$\omega_p^{-2}\,\ddot{\phi}(t)+\omega_c^{-1}\,\dot{\phi}(t)+\sin\phi(t)=i+i_{noise}=\frac{I+I_{noise}}{I_c}, \tag{1.37b}$$

where $\omega_c = (2\pi/\Phi_0)I_cR_N$ is the characteristic frequency of the junction, $\omega_p = 1/\sqrt{L_cC}$ is the plasma frequency (to be defined more rigorously in the next section), and $L_c = \Phi_0/(2\pi\, I_c)$ is the effective Josephson inductance, and $R_N = R_JR_S/(R_J +$

R_S) is the effective normal current resistance across the junction. Equation (1.37b) rearranges the terms in (1.37a) to show the similarity to the damped harmonic oscillator. The time constants of the junction are $\tau_N = R_N C = \omega_c/\omega_p^2$ and $\omega_c = R_N/L_c$. The damping factor of the junction is equivalent to the quality factor Q of the junction, which is related to the Stewart-McCumber parameter β_c by

$$\beta_c = Q^2 = \left(\frac{\omega_c}{\omega_p}\right)^2 = \omega_c R_N C = \frac{2e}{\hbar} I_c R_N^2 C = \frac{2e}{\hbar}(I_c R_N)^2 \frac{c_s}{j_c}, \tag{1.38}$$

where the specific parallel plate capacitance is $c_s = C/A = \epsilon_0 \epsilon_r / d$ and the critical current density is $j_c = I_c/A$. For a given j_c, β_c is a function of c_s and the $I_c R_N$ product.

It is worth pointing out that the equation of motion (1.37a) is identical to that of a damped pendulum, in which the torque is analogous to current, the capacitance is analogous to moment of inertia, the conductance is analogous to the damping coefficient, the critical current is analogous to the maximum gravitational torque, and the junction phase is analogous to angle. This analogy is useful in understanding some of the more complex junction behavior we will shortly discuss.

Figure 1.5 shows phase diagrams in the position (φ) - momentum (p_φ) plane for $I = 0$ of a junction for the underdamped ($Q > 1$), critically damped ($Q = 1$), and overdamped ($Q < 1$) cases. The traces show the trajectories of the junction phase. Two attractors show the equilibrium points. A sufficient increase in momentum will cause the junction to switch to a different equilibrium point, generating an SFQ pulse. The critically damped case shows a return to equilibrium in minimum time. Underdamped junctions show oscillation before reaching equilibrium. Overdamped junctions take on low values of p_ϕ and do not oscillate but do not return to equilibrium as quickly.

In unshunted SIS junctions $I_cR_N = (2\pi/3)\Delta$. Typically $\beta_c \gg 1$, so the junctions are over damped. To decrease β_c an external shunt resistor R_S is added across a junction to reduce β_c, and then R_N can be replaced by $R_N = R_JR_S/(R_J{+}R_S) \approx R_S$ for $R_S \ll R_J$, decreasing the overall shunting resistance of the junction. In most cases $R_S \ll R_N$ and we can substitute the value of R_S for R_N without any further modifications [4]. $V_c = I_cR_N$ then sets the voltage scale for junction behavior, and is typically on the order of a few mV for Nb/Al$_2$O$_3$/Nb junctions.

Thus the behavior of the junction is determined in design by the choice of the shunt resistor R_S, the junction area A, and the critical current density j_c. For $\beta_c > 1$ the junction is underdamped and plasma oscillations with frequency ω_p will occur that will damp out on a time scale $\tau = R_NC$. For $\beta_c < 1$ the junction is overdamped and after a disturbance the phase slowly moves towards equilibrium with a time constant $1/\omega_c$.

The potential energy of the junction can be calculated directly from (1.34) and (1.35). The work W_S done on the junction leads directly to an expression for the potential energy of a junction U_S as follows:

$$\begin{aligned} W_S &= \int_{t_1}^{t_2} I_S(t)V(t)\,\mathrm{d}t = \frac{\Phi_0 I_c}{2e}\int_{\phi_1}^{\phi_2} \sin\phi\,\mathrm{d}\phi \\ &= \frac{\Phi_0 I_c}{2e}(\cos\phi_1 - \cos\phi_2) = U_S(\phi_2) - U_S(\phi_1). \end{aligned} \tag{1.39}$$

From this we can define:

$$U_S(\phi) = -\cos\phi\frac{\Phi_0 I_c}{2e}. \tag{1.40}$$

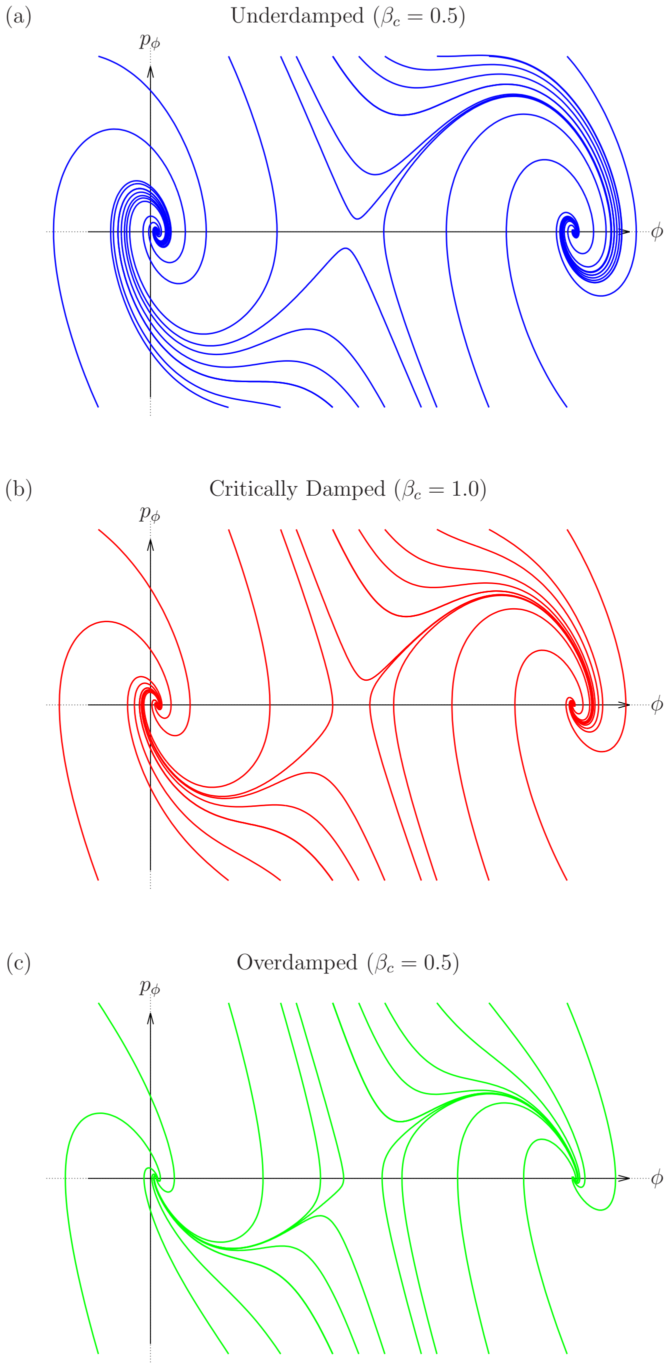


Figure 1.5: Phase Diagram of Josephson Junction. This figure shows plots of trajectories in the phase ϕ and momentum p_ϕ plane. Phase diagrams for (a) underdamped, (b) critically damped, and (c) overdamped junctions. In all three cases the state of the junction moves to $\phi = 2\pi n$, $\dot{\phi} = 0$. Oscillations can be seen in the underdamped case.

This is the Jospehson energy and we can think of it as energy stored in the effective junction inductance. For small oscillations of the phase, for which $\phi \to \phi + \tilde{\phi}$, we can look at the variations in current $\tilde{I}_s$ and voltage $\tilde{V}$. Expanding (1.34) in a Taylor series gives $I_s = I_c(\sin\phi + \cos\phi \times \tilde{\phi} + \ldots)$. Using (1.35) to express the phase as the integral of voltage, we get a relation between current and phase which is that of an effective inductance:

$$L_S \tilde{I}_S = L_S I_c \cos\phi \times \tilde{\phi} = \int \tilde{V} \mathrm{d}t = \frac{\Phi_0}{2\pi}\tilde{\phi}, \tag{1.41a}$$

and we can define the bias-dependent Josephson inductance

$$L_S = \frac{\Phi_0}{2\pi I_c}\frac{1}{\cos\phi} = \frac{\hbar}{2\,e\,I_c}\frac{1}{\cos\phi} = \frac{L_c}{\cos\phi}, \tag{1.41b}$$

with L_c as previously defined. For bias current $I_b < I_c$ the junction can be in the zero-voltage (S) state where the phase is constant at $\phi = \arcsin I_b/I_c + 2\pi n$, where n is an integer. The potential for a biased junction, using (1.40) in the Gibbs free energy relation $G = U - (I_b/I_c)\phi$ [4], is

$$U(\phi) = \frac{\hbar}{2\,e} I_c \left(1 - \cos(\phi) - i\,\phi\right). \tag{1.42}$$

Figure 1.6 shows plots of $U(\phi)$ for the cases of a high critical current compared to I_b, a low critical current, and an overbiased junction with $I_b > I_c$. Three general situations can occur. First, for $I_b < I_c$ a junction can exhibit plasma oscillations about the equilibrium position. Second, if I_b is less than but close to I_c, the junction can tunnel through the barrier to the next minimum (and possibly beyond, depending on β_c) and increase its phase by 2π. Third, if $I_b > I_c$ no minima exist and the junction phase continuously increases with time.

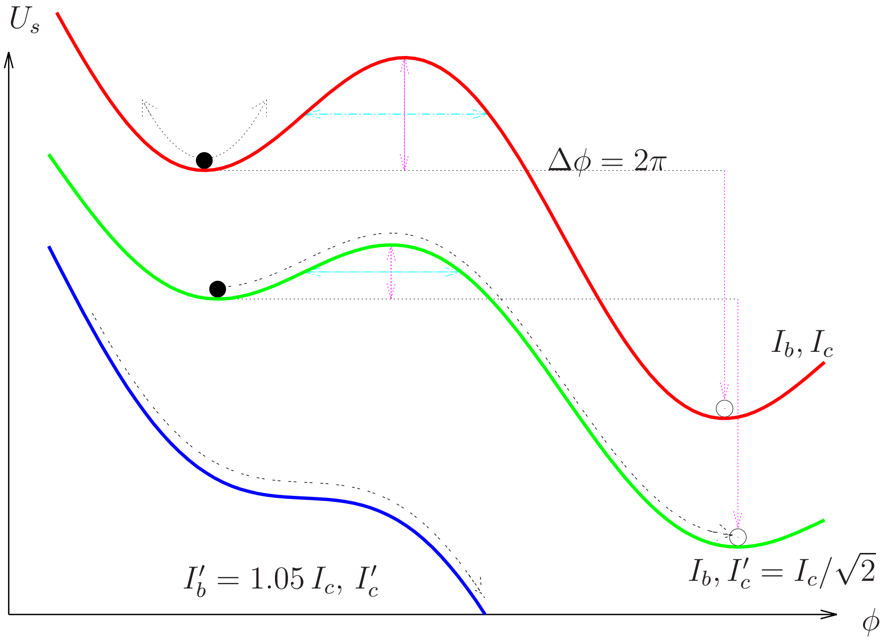


Figure 1.6: Josephson junction potential energy. The potential energy U_s of a junction as a function of phase is shown for high and low critical currents of an underbiased junction ($I_b < I_c$), and for an overbiased junction ($I_b > I_c$). The mechanical analog is a ball rolling down a tilted washboard. The bias current I_b through the junction determines the average downward slope. (top) Plasma oscillations of a junction trapped in a local minima. (middle) Tunneling though the potential barrier of height $\Delta U(I) = 2\Phi_0 I_c(1 - I/I_c)^{3/2}$ to the next local minima [9]. This generates an SFQ pulse and changes the phase by exactly 2π. (bottom) For $I_b > I_c$ no local minima exist and the phase increases without limit.

1.3.3 Behavior of Overdamped Junctions

The relationship $\int V \mathrm{d}t = \Phi_0$ for one cycle of oscillation is fundamental to all Josephson junctions and is a key property of SFQ pulses. The case of an overdamped junction illustrates how SFQ pulses can be generated and shows the utility of the shunt resistor.

The equation of motion for a Josephson junction cannot in general be solved analytically. However in the simple case of an overdamped junction where $\beta_c \to 0$ (which implies $\omega_c \ll \omega_p$) an analytic solution can be found for a constant current. Let $i_b = I_b/I_c$ and let time be normalized to the characteristic time $1/\omega_c$ such that $t' = \omega_c t$. Equation (1.37b) then becomes

$$i_b = \dot{\phi}(t') + \sin\phi(t'), \tag{1.43}$$

which has an analytic solution. We are interested in the SFQ pulse dynamics in the I-V curve characteristics of the junction. For $i_b > 1$ the solution to (1.43) is [4, 6]

$$\phi(t') = 2\arctan\left(\frac{1 + v\,\tan\left(\frac{1}{2}\,v\,t'\right)}{\sqrt{v^2+1}}\right). \tag{1.44}$$

where $v = \sqrt{i_b^2 - 1}$ is the time-averaged normalized voltage. The derivative of (1.44) determines the voltage as a function of time.

$$V(t') = \frac{\Phi_0}{2\pi}\dot{\phi}(t') = \frac{\Phi_0}{2\pi} v^2\,\sqrt{1+v^2}\,\frac{\sec^2\left(\frac{1}{2}\,v\,t'\right)}{1 + v\tan^2\left(\frac{1}{2}\,v\,t'\right)} \tag{1.45}$$

This is a periodic solution with period (in non-normalized time units)

$$\Delta t = \frac{2\pi}{\omega_c\,v}. \tag{1.46}$$

Figure 1.7 shows examples of the voltage behavior of an overdamped junction. Figure 1.7(a) shows the solution of (1.45) for two values of I_b. Figure 1.7(b) shows the numerical solutions to (1.37b) for the same values of I_b but with $\beta_c = 0.25$. Notice in Fig. 1.7 how for low values of v the junction produces individual pulses. The average voltage $\bar{V}$ across the junction can be calculated from (1.45),

$$\langle\dot{\phi}\rangle = \frac{v}{2\pi}\int\limits_{2\pi/v} \dot{\phi}(t)\,\mathrm{d}t = v = \sqrt{i^2-1}, \tag{1.47}$$

$$\bar{V}(I_b) = \frac{\Phi_0\omega_c}{2\pi}\sqrt{(I_b/I_c)^2-1}. \tag{1.48}$$

Figure 1.8 shows I-V curves of over- and underdamped junctions for various values of β_c. For overdamped junctions ($\beta_c < 1$) the behavior is non-hysteretic. For underdamped junctions ($\beta_c > 1$), such as those we will use in RQL, the behavior is hysteretic. Starting from zero bias current, a small increase in bias current results in no steady voltage until the critical current is exceeded. The voltage then jumps to a finite value instead of gradually increasing (solid black arrow). Upon decreasing the current, the momentum of the junction can carry the phase through an infinite number of rotations despite some dissipation so long as the bias current supplies enough tilt to the potential. The voltage quickly decreases with decreasing current (red arrows) and drops to zero once the curve falls below the return current $I_r = 4\,I_c/(\pi\sqrt{\beta_c})$ [9]. The zero-voltage branch and finite voltage branch correspond to different initial conditions for $\dot{\phi}(0)$ when solving (1.37b).

1.4 Superconducting Interferometers

In the previous section, I discussed the behavior of single junctions. Most importantly, I have shown how flux is quantized in loops and that overdamped

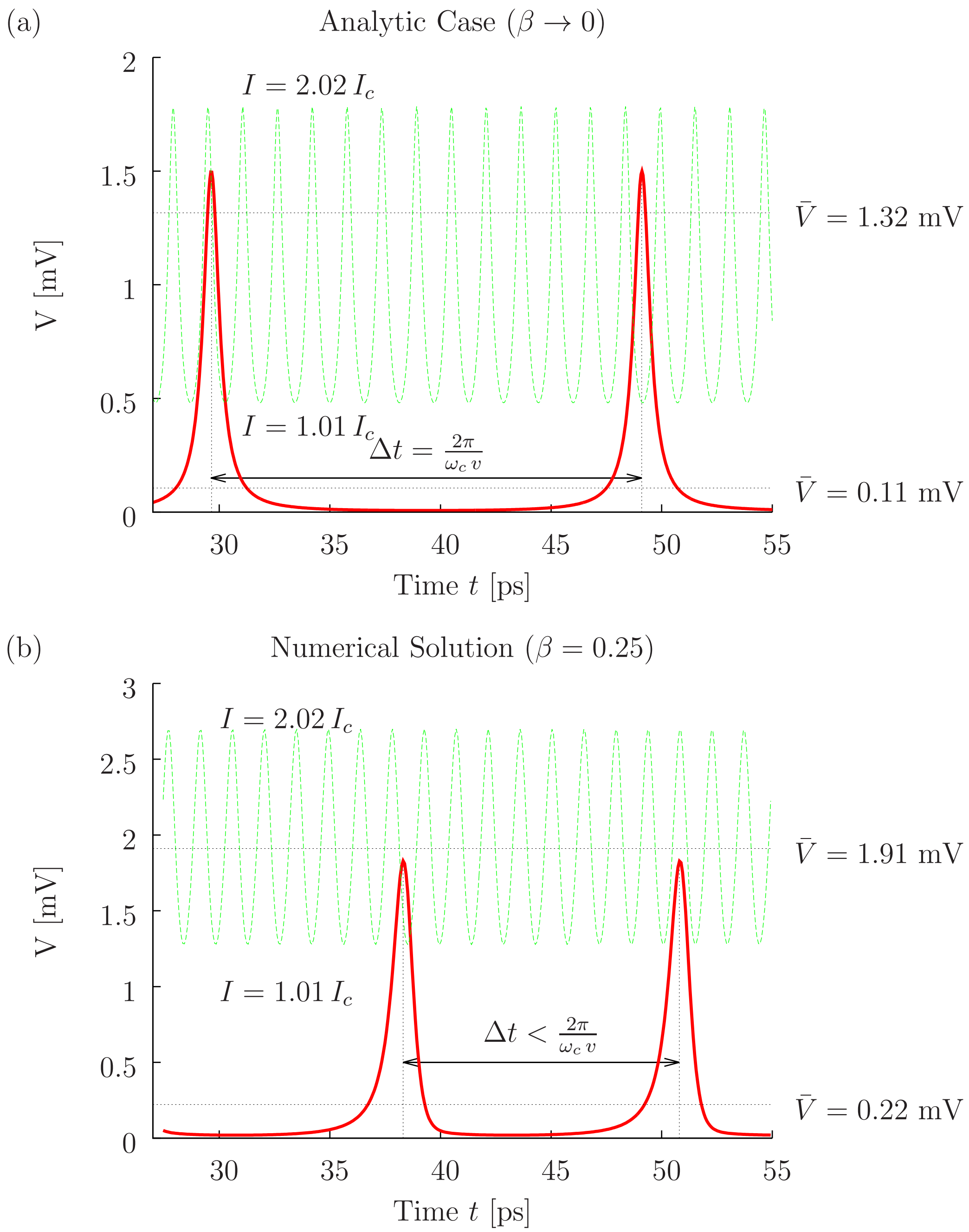


Figure 1.7: Voltage vs time dynamics of overbiased junction for two values of applied bias current. (a) Analytic solution to (1.43). Red curve ($v = 0.142$) for small overbias shows well-separated SFQ pulses. Green curve ($v = 1.76$) for large overbias resembles a high-frequency sinusoidal variation instead of individual pulses. For low overbias the separation between pulses is $\Delta t = 2\pi/\omega_c v$. The average voltage is given by (1.48). (b) Numerical solution to (1.37b) for same values (other than β_c) as in part (a). Pulses become closer together. Average voltages are higher. ($I_c R_N = 0.75\,\text{mV}$)

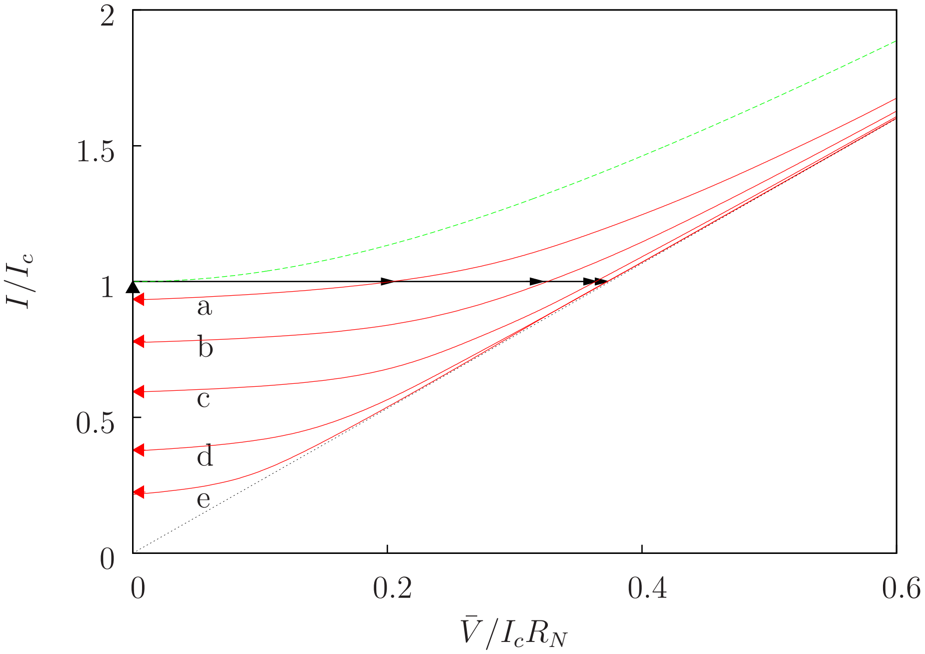


Figure 1.8: I-V curve of current driven junctions. Green curve shows I-V curve for $\beta_c = 0$. For $\beta_c < 1$ there is no hysteresis. For $\beta_c > 1$ the red I-V curves shows hysteretic behavior. Curves a–e have $\beta_c = 1.1, 2, 4, 10, 30$. Junctions remain in the zero voltage state (zero average voltage) until the critical current is reached. The voltage then jumps (horizontal black line) to a finite value (red curves). The voltage does not return to zero until the current has been reduced below the return current value $I_r = 4\,I_c/(\pi\sqrt{\beta})$. (Results calculated numerically from (1.37b).)

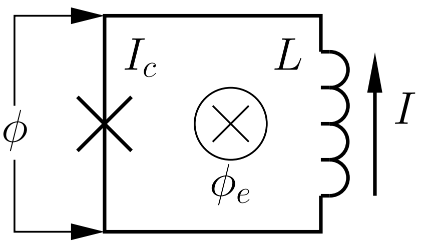


Figure 1.9: Single-junction interferometer equivalent circuit. A junction with critical current I_c is connected at both ends to an inductance L. The phase across the junction is ϕ and the current through the junction is I. An externally applied magnetic field couples flux Φ_{ext} into the loop and this can be thought of an inducing a phase ϕ_e in the loop.

junctions can create individual single-flux-quantum voltage pulses. These two effects lay the foundation for digital logic in superconducting circuits. A digital "one" is stored as a flux Φ_0 in a loop and transmitted as an SFQ voltage pulse. To make further progress requires examining more complicated circuits with an inductor and one or more junctions.

1.4.1 Single Junction Interferometer

In RQL circuits, each Josephson junction is part of one or more superconducting loops. In such circuits the phase difference across the junction is modulated by the magnetic flux applied to the loop. Figure 1.9 shows a single-junction interferometer formed from a superconducting inductor L and a single junction with critical current I_c. The total flux in the loop is related to the current I in the loop [4] and the applied flux Φ_{ext} by $\Phi = LI + \Phi_{ext}$. The flux-phase relation allows us to express the phase across the junction as

$$\phi = \phi_e - \frac{2\pi}{\Phi_0} LI = \phi_e - \lambda i \tag{1.49}$$

where $i = I/I_c$, $\phi_c = 2\pi\Phi_{ext}/\Phi_0$, and the normalized inductance of the loop is $\lambda = L/L_c$, and $L_c = \Phi_0/(2\pi I_c)$. This gives the junction phase ϕ as a function of the applied flux phase ϕ_e and the loop current I.

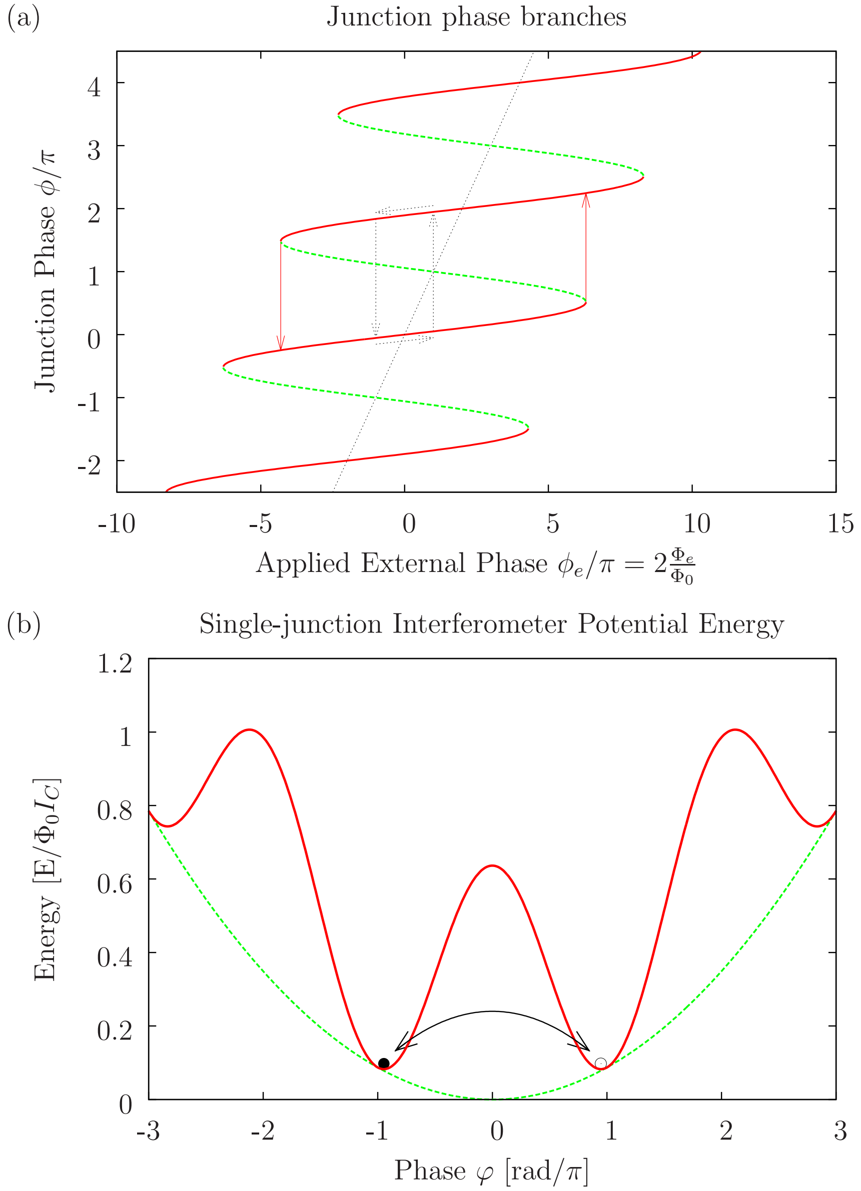
In a stationary state, the current through the junction obeys $i = \sin\phi$, which allows us to rewrite (1.49) as

$$\phi + \lambda \sin\phi = \phi_e. \tag{1.50}$$

The phase of the single junction interferometer ϕ is plotted as a function of ϕ_e in Fig. 1.10(a). For $\lambda < 1$ the junction phase follows the applied phase, $\phi \approx \phi_e$. For $\lambda > 1$ the value of ϕ becomes hysteretic, with only certain values of junction phase allowed. These values correspond to the number of single flux quanta Φ_0 stored in the loop. When the the junction switches, it jumps from one branch to another. Between the branches an SFQ pulse is generated by the changing phase across the junction and the changing current through the inductor. A different way to understand these jumps is to look at the energy of the loop, including both the junction and the inductor. This gives an energy in terms of the phases ϕ and ϕ_e as:

$$U(\phi) = I_c\Phi_0 \left(1 - \cos\phi + \frac{(\phi - \phi_e)^2}{2\lambda}\right). \tag{1.51}$$

This is plotted in Fig. 1.10(b) for the special case of $\phi_e = \pi/2$, which makes the energy symmetric about $\phi = 0$. In switching between the two lowest minima, no energy is dissipated and the switching behavior back and forth is the same.


(a) Junction phase branches

(b) Single-junction Interferometer Potential Energy

Figure 1.10: Single-junction interferometer phase behavior and potential energy. (a) The junction phase ϕ is plotted as a function of the externally applied phase ϕ_e showing both the allowed branches (red solid) and prohibited branches (green dashed). A transition from one branch to the next results in an integer change of the number of flux quanta Φ_0 in the interferometer. (b) Potential energy (red solid curve) of the single-junction interferometer when biased by $\Phi_0/2$ flux in the loop. Solid and empty circles show two meta-stable states at equal energies. Green dashed curve shows quadratic term in (1.49).

This behavior will become very important in the next chapter, in which I describe the nature of a new logic family. We wish for a symmetry between positive flux and negative flux. With this kind of symmetry in a single loop interferometer, no power will be dissipated by switching events.

1.4.2 Josephson Transmission Line

In this section I briefly discuss the Josephson transmission line (JTL). A Josephson transmission line is a series of single junction superconducting interferometers coupled together by inductances L. It is of fundamental importance to RQL because it can carry SFQ pulses from one junction to another. The basic concept of the JTL can be seen in Fig. 1.11. A constant bias current — less than I_c and by convention about $0.7I_c$ — is supplied to each junction. The current of an SFQ pulse causes the underbiased junction to become overbiased and switch through a phase of 2π. This switching generates new SFQ pulses which travels both backwards, canceling out the original SFQ pulse, and forwards, allowing a pulse to propagate forward.

When multiple junctions and inductors L are coupled together, (1.37b) can be generalized and one finds:

$$\begin{aligned}\frac{\Phi_0}{2\pi L}\left(-\phi_{i-1}(t)+2\phi_i(t)-\phi_{i+1}(t)\right)= \\ -\omega_p^{-2}\ddot{\phi}_i(t)-\omega_c^{-1}\dot{\phi}_i(t)-\sin\left(\phi_i(t)\right)+I_b,\end{aligned} \tag{1.52}$$

where i refers to the i^{th} junction in the JTL and I_b is a generic externally applied bias current. The left hand side of the equation describes the currents flowing to and from the i^{th} single-junction interferometer loop. On the right-hand-side of the equation we see the regular terms from (1.37b) and a biasing function of our choosing. On the left-hand-side are the coupling terms between junctions, which are simply the currents flowing through the inductors from the previous $(i-1)$ and next $(i+1)$ junctions. The set of equations for $i = 1, \ldots, N$, plus boundary conditions, forms the equations of motion for the whole transmission line. (This is in fact a discreetized version of the sine-Gordon equation, which implicitly has solutions of traveling SFQ pulses [6].)

The JTL configuration shown in Fig. 1.11 will allow positive pulses to travel rightward and negative pulses to travel leftward. Negative pulses will travel rightward if the direction of the bias current is reversed. We have a choice of I_b. If the bias current is supplied through coupled inductors, the junction forms a single junction interferometer, and the switching can occur between equipotential states. If the bias current is not constant but can vary over time and (discreetly) over space, we can control the flow of pulses. In RQL, one chooses $I_b = A\sin(\omega\, t)$ so that both positive and negative pulses travel rightward during opposite clock phases.

The solution to (1.52) for $I_b = A\sin(\omega t)$ is shown in Fig. 1.12 for four junctions, each on two phases. (See Appendix A.) The propagation of pulses is clear. I also note another important fact, that pulses can be held at a “phase boundary” between different bias conditions. Also it is clear that the junctions can propagate both positive and negative pulses. The detailed behavior of such an arrangement of Josephson junctions is the topic of the rest of this thesis.

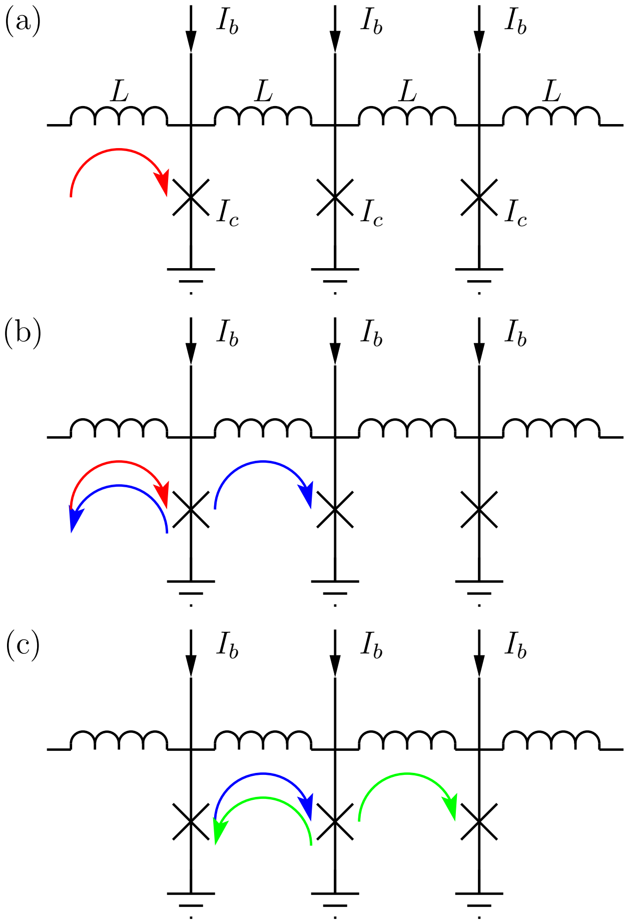


Figure 1.11: Josephson Transmission Line. Junctions with critical current I_c are biased by a current $I_b < I_c$ and coupled to adjacent junctions through inductors L. SFQ current shown by arrows. This configuration allows the propagation of SFQ pulses from one side to the other. (a) though (c) show a time progression of pulse propagation. (a) A current (red) from the SFQ pulse passes through the first junction. The combined SFQ and bias current exceed the critical current. (b) The junction switches and generates an SFQ voltage pulse, which in turn creates exactly one SFQ pulse to the left and right (blue arrows). Red and blue arrow on left cancel out. (c) The newly generated SFQ pulse causes the second junction to switch, repeating the process.

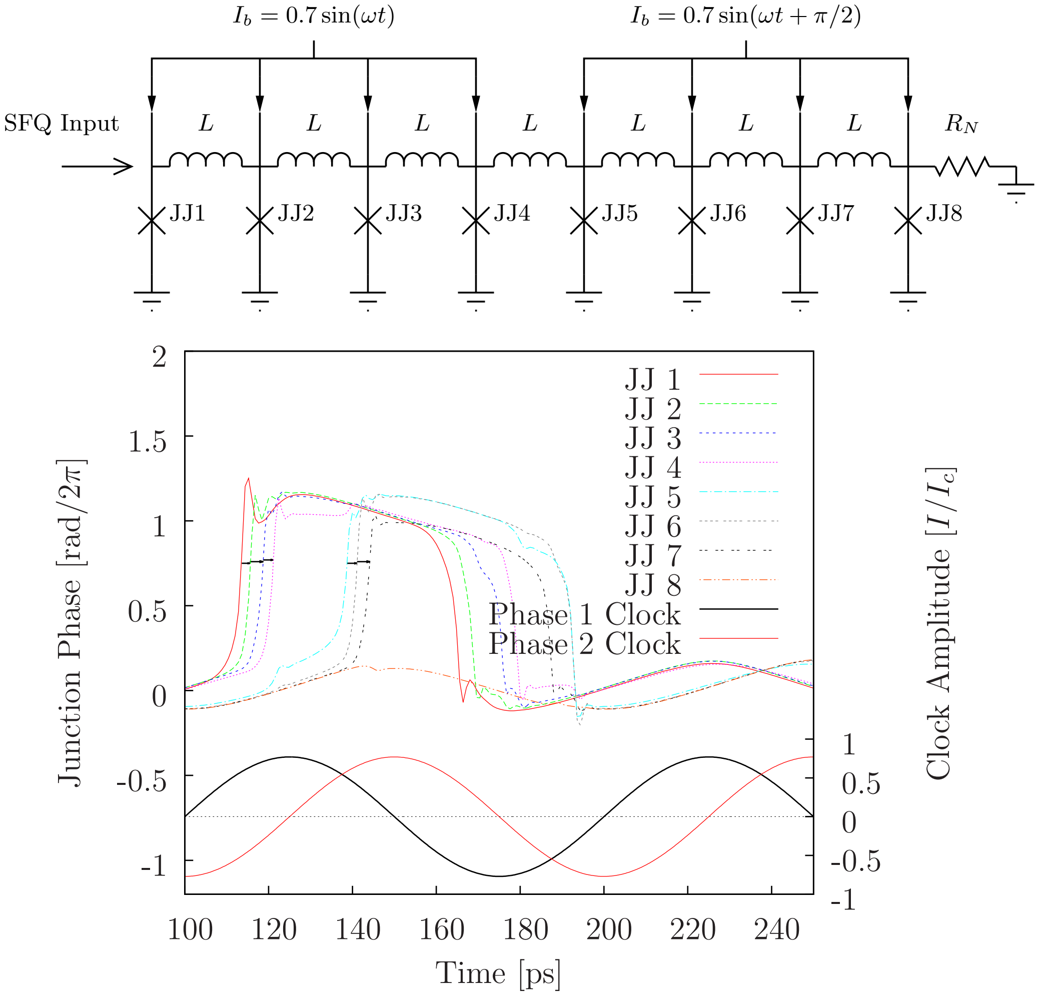


Figure 1.12: Phase behavior of junction in JTL. The solution to (1.52) for the circuit schematic shown on top is shown with phases ϕ_1 to ϕ_8 driven by the bias current shown on bottom. Junctions $1-4$ are driven by black sinusoid, junctions $5-8$ by the red sinusoid, a quarter period later. The junctions can be seen to switch in sequence with the later four junctions switching only when the local bias current is high. (Junction 8 is highly damped to prevent reflections and does not actually switch itself.) The first junction is driven by a positive SFQ pulse followed by a negative SFQ pulse half a period later. (Numerical solution. $I_cR_N = 0.75$, $\beta_c = 1.56$. Further details found in Appendix A.)

1.5 Introduction to Superconducting Digital Logic

Moore's Law predicts that the speed of digital electronics will increase by a factor of two every two years. This has been achieved in practice by making CMOS circuits smaller and smaller. Recently, this progress has slowed and CMOS has been stuck at clock speeds of around 4 GHz for the last few years [14]. Faster circuits have come at the cost of higher power dissipation and heat loading. This is one factor limiting progress of CMOS technologies. Multi-processor schemes have allowed further throughput improvements, however this is expected to reach a limit, too. Parallel processing introduces an overhead which some have predicted will impact performance at about 16 processing units [14]. To break through these limitations, a new class of digital circuits is needed.

Some type of superconducting digital electronics may ultimately fill this need. Superconducting technologies have a number of inherent advantages over CMOS. The flux in a superconducting ring is quantized to exactly $\hbar/2e$ making digital one and digital zero intrinsically defined quantities in the system. In Josephson junctions, creating one such quantized flux corresponds to an energy consumption of typically about $10^{-18}J$, far lower than that for CMOS [14]. Also, the inherent switching speed of junctions is fast; 1 mV applied potential corresponds to a 500 GHz oscillation frequency. Of course, CMOS technology also has many advantages, and developing a new technology that can compete with CMOS is not easy.

The first logic based on the processing of SFQ pulses was Rapid Single Flux Quantum (RSFQ) logic [15]. In RSFQ circuits, digital one and digital zero are encoded as the presence or absence of an SFQ pulse between two clock pulses. Within one clock period, the data is stored as magnetic flux states of superconducting interferometers. Clock pulses are used to read the state of internal memory and reset gates. RSFQ logic has demonstrated fast operating speeds [16] (up to 700 Gbit/s in a static divider), low dynamic power dissipation [17] (a few mW for a whole circuit) [6], chip-to-chip communication of more than 100 Gbit/s [18, 19], and an integration density of tens of thousands of junctions per chip [20], on demonstrated prototypes [21, 22].

However, RSFQ has some issues. For example, pulse encoding used in RSFQ logic imposes some limitations. RSFQ uses a ripple clock distribution where active elements — the Josephson junctions — regenerate the clock pulses. The ripple-clock distribution necessitates active hardware delays between gates and leads to a jitter accumulation [23]. The internal memory of the gates inherently leads to large latency, as the resetting clock signal must propagate through the whole circuit. Another problem is that the DC power scheme uses bias resistors that give at least ten times higher static power dissipation than the switching power [24]. Also, RSFQ circuits are built from finite-state machines and pipelined on the gate level. This allows high throughput at the cost of high latency. Together these properties limit application of RSFQ and make it unsuitable for VLSI applications such as high end computing where operations-per-Joule and latency are prime performance metrics [25].

Many challenges have prevented of superconducting technologies from seeing widespread use in the past. However, recently a number of superconducting digital electronics have found commercial use. Advances in cooling technology make space- and energy-efficient options available for computing [14]. Digital signal processors, adaptive filtering, and direct digitalization has all been performed in a commercial setting using superconducting digital electronics. Despite these advances, a number of issues still remain. In particular, the lack of existing superconducting digital memory prevents its use as a general purpose computer that can compete with silicon-based processors.

Chapter 2

Reciprocal Quantum Logic

2.1 Introduction

Power consumption has increasingly become a limiting factor in high performance digital circuits and systems. According to a U.S. Environmental Protection Agency study [26], the demand of servers and data centers in the U.S. is approaching 12 GW, equivalent to the output of 25 typical 500 MW power plants. Here I describe a new logic family, Reciprocal Quantum Logic, that yields a factor of 300 reduction in power compared to projected nano-scale CMOS, even taking into account the power consumed to maintain a cryogenic operating temperature. In this chapter I discuss the fundamentals of reciprocal quantum logic. I first describe an RQL transmission line and RQL logic gates. I then describe three benchmark experiments that I completed that show the scalability of RQL for very large scale integrated (VLSI) circuits.

In this introduction I describe the encoding of classical digital data using reciprocal SFQ pulses. RQL gates, operate with single magnetic flux quanta (SFQ) generated by overdamped Josephson junctions. This is the same approach used in RSFQ gates. Figure 2.1 illustrates how data is encoded in RQL. A "one" bit is encoded as a pair of positive and negative (reciprocal) SFQ pulses generated in the positive and negative phases of the sinusoidal clock. A "zero" bit corresponds to the absence of positive/negative pulse pairs during a clock cycle. The positive SFQ pulse arrives during the positive part of the clock signal while the negative pulse follows later during the negative part of the clock cycle.

A major difference between RQL and RSFQ is how power is supplied to the gates. RSFQ gates [4] use static dc power applied in parallel through bias resistors, while RQL uses ac power applied in series. Figure 2.1 shows the AC power applied through the inductively coupled bias line; the AC power simultaneously serves as a global clock reference. With no bias resistors there is no static power dissipation. Power induced in L_{AC} is conservative apart from junction switching events.

2.2 Josephson Transmission Line

Figure 2.2 shows the schematic of an RQL Josephson transmission line. The JTL is formed from a series of cells with each cell being an inductive loop formed by junctions JJ1 and JJ2 and inductors L_1 and L_2. The inductances L_1 and L_2 are small ($L_1 I_{c1} \ll \Phi_0$) so an incoming SFQ pulse will induce switching of both junctions in series. Junctions are biased through inductor L_0 which is coupled to the AC line inductance L_c via a mutual inductance M_{0c}. AC current in the clock line induces positive bias current through the junctions in the positive half-period of the clock cycle and negative current during the negative half-period. The flux bias inductor L_b is large ($L_b I_1 > \Phi_0$) and any junction connected directly to a bias inductor (L_0) forms a single junction interferometer with two stable states, similar to the single junction interferometer circuit shown in Fig. 2.1. The circuit in Fig.

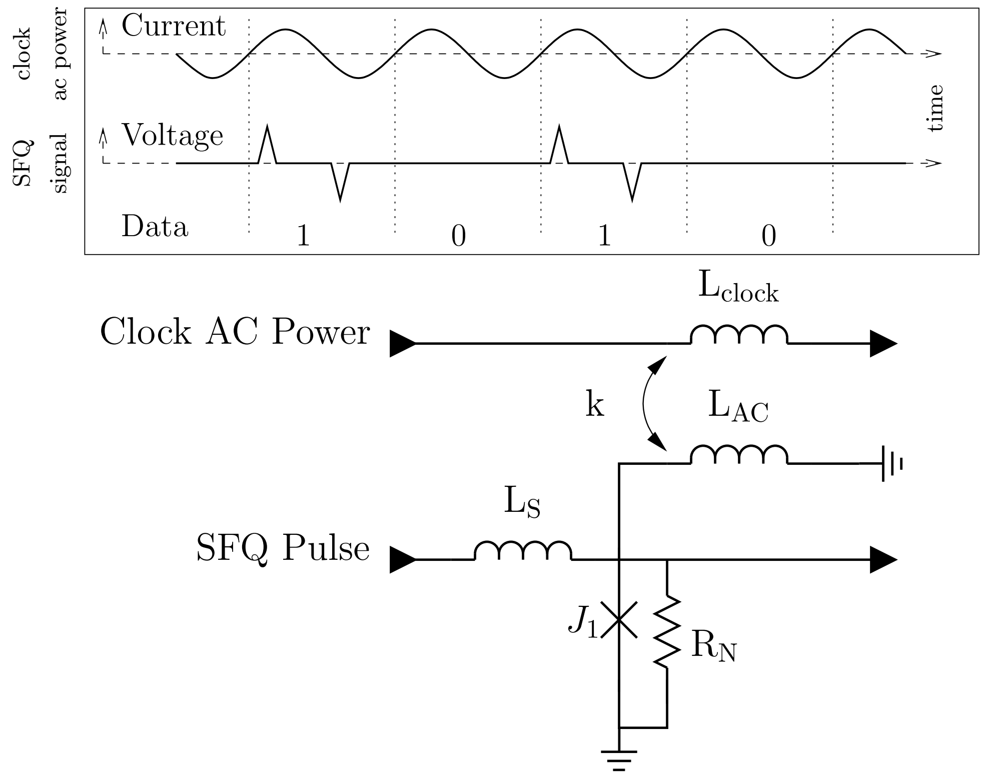


Figure 2.1: Basic RQL active interconnect element showing grounded junction J_1 coupled inductively to clock line. Junctions are coupled to other elements through L_S. The interconnect draws energy from the power line, much like an RSFQ JTL. This element provides isolation, amplification of the current with a characteristic delay. The junction and bias inductor L_{AC} form a single-junction interferometer.

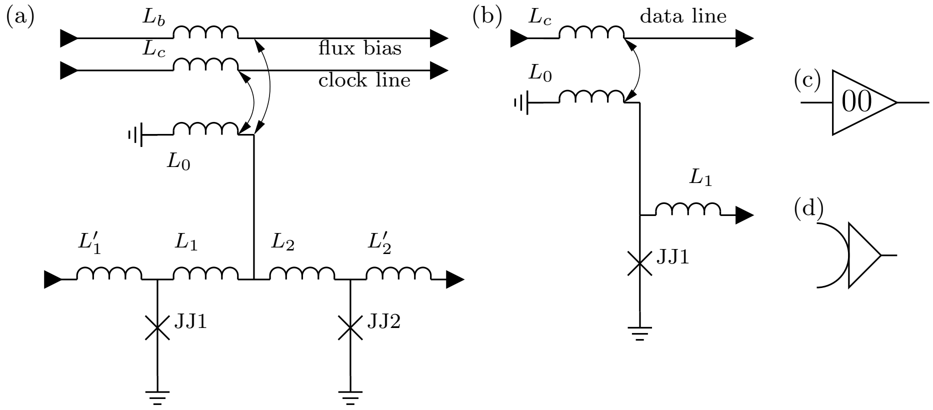


Figure 2.2: Josephson transmission line and SFQ launch circuit diagram. (a) Circuit diagram for RQL JTL. The values of inductors are: $L_0 = 13.4\,\mathrm{pH}$, $L_1 = L_1' = 3.0\,\mathrm{pH}$, $L_2 = L_2' = 2.1\,\mathrm{pH}$, $M_{0b} = 0.5\,\mathrm{pH}$, $M_{0c} = 1.7\,\mathrm{pH}$. The critical currents for the junctions are: $\mathrm{JJ1} = 0.100\,\mathrm{mA}$, $\mathrm{JJ2} = 0.141\,\mathrm{mA}$. (b) Circuit diagram for RQL launch. (c) Block diagram symbol of JTL, with two digits indicating phase and clock line number. (d) Block diagram symbol of launch.

2.2 is electrically equivalent to that shown in Fig. 2.1 for $L_{AC} = L_1 + L_0/2$ and $L_S = L_1'$ for JJ1 and $L_{AC} = L_2 + L_0/2$ and $L_S = L_2'$ for JJ2. A single flux quantum Φ_0 will be stored in the loop formed by the junction J_1 and bias inductor after each increase of the phase by of the junction by 2π. This stored flux is canceled out by the reciprocal pulse. Additional DC flux bias on the clock line induces a flux of $\Phi_0/2$ in the bias inductor. This makes the states of the single-junction interferometer symmetric with $\pm\Phi_0$.

The RQL JTL is known to have wide operating margins, more than 50% on individual critical currents [27]. The critical parameter found in our simulations turns out to be the ratio between bias inductor L_b and transmission inductor $L_1 + L_2$. The bias inductor L_b needs to be as large as possible so the current in the bias inductor does not effect junction switching. However, for practical reasons bias inductors need to be limited in size. Values in Fig. 2.2 correspond to the nominal design I used.

RQL JTLs allow amplification of SFQ pulse energy. This is achieved by stepping up the critical current from one cell to the next. With two sequential steps with amplification by $\sqrt{2}$, the energy of the SFQ pulse is doubled. The nominal design values I used corresponds to an amplification of the SFQ energy by $\sqrt{2}$ per stage. This allows me to use the same JTL cell layout as part of an RQL SFQ splitter. An RQL splitter is formed by attaching the input of two JTL units to the output of a single JTL. This effectively gives the JTL unit a fan-in of one-half and a fan-out of one.

Many cells can be connected in series to form a long JTL. A pulse will propagate through a JTL segment so long as the bias current is sufficient. However, one-phase AC power does not provide directionality for pulses. Using only one phase, during the negative half-cycle, junctions will switch in the opposite order and a positive pulse — which moved forward during the positive half clock cycle — would travel backward during the negative half. To prevent this, RQL uses a four-phase clock; two clock lines with a phase difference of $\pi/2$ provide two phases. By coupling the clock lines to the junctions in a wound or counter-wound fashion one produces a total of four phases differing by 0, $\pi/2$, π, and $3\pi/2$. With four phases, when one phase is nearing the end of the "timing window," the next has already started, allowing a pulse to continue onward. (I will give a precise definition of the timing window in Chapter 3. For now, it can be thought of as the time during which the clock is close to maximum amplitude, approximately the third of the period in which the AC signal has half its maximum amplitude or higher current.) For slower clock speeds, the pulses will wait for the rise of the next clock phase.

Figure 2.3 shows a four-phase RQL JTL. Given the geometry of two clock lines and two winding directions, the natural way to index the clock phases is with a two-bit binary in which the first digit is the winding bit and the second the clock line bit. The first clock line (0 phase) with regular winding is 00 and the third (π phase), for example, is 10.

The four-phase clock provides an implicit pipeline — data processing elements with the output of one element as the input of the next one — without additional devices needed for latches or clock distribution. However, in cases where the phase of the clock on the next element is delayed by $\pi/2$ relative to the current JTL, JTLs on such *phase boundaries* require slightly altered values compared to the inductors shown in Fig. 2.2. Because of the phase difference, the bias current through adjacent junctions will cause current to leak from one JTL to the next. This can be prevented by altering the inductance values so as to redistribute the currents correctly. A pipeline in RQL can have any number of logical elements on a single phase, i.e. the elements are connected in series to a single clock line with a single phase. Short pipelines can have higher operating frequencies with fewer operations per phase. Long pipelines, such as shown in seen in Fig. 2.4, can have more logical operations per phase at the cost of lower operating frequency; the SFQ pulses must reach the next phase before the current becomes too small to bias the junctions. In general, the speed of large circuits is effectively the product of clock frequency and pipeline

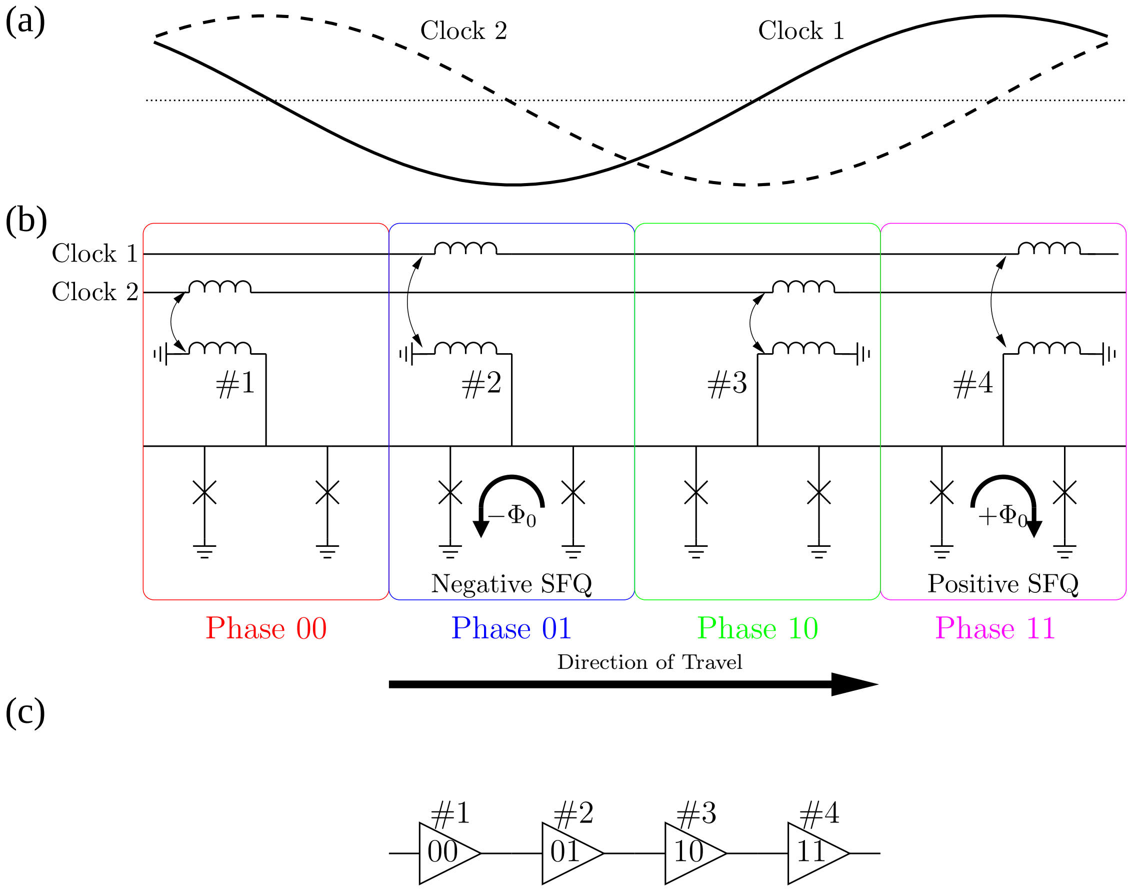


Figure 2.3: Data propagation in an RQL 4-phase clock transmission line. (a) Two clock signals with a quarter-period offset between them. (b) Aligned with the waves in (a), this figure shows four JTL units in series on four different phases labeled 00, 01, 10, and 11. Two clock lines provide four phases by counter-winding, shown by inductors pointing in opposite directions. Pulse directionality is achieved by a four-phase clock. A positive flux propagates bias current is positive, and a negative propagates forward when the bias current is negative. Positive and negative SFQ pulses represented by the current generated by their flux in two pairs of junctions each. Positive pulses drives bias current down both junctions, and the rightmost will switch first. Negative pulses drive bias current upward though junctions; again the right junction of the pair switches first. (Only coupled inductors are shown for clarity.) (c) Block diagram of the circuit shown in (b).

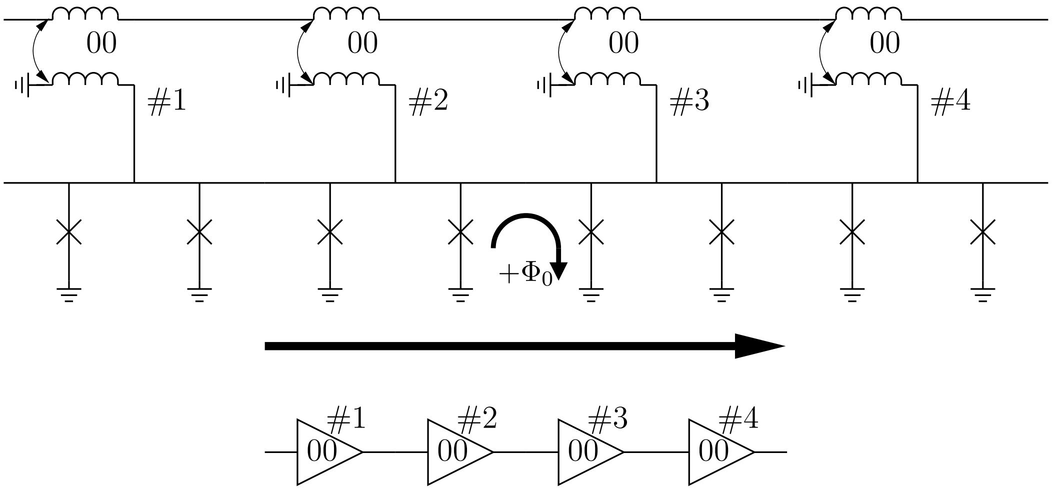


Figure 2.4: Deep Pipeline JTL. Four JTL units are on the same clock phase with a single positive SFQ pulse traveling left-to-right. The reciprocal pulse comes later. Pulses travel through any number of junctions provided that the bias current is correct. Pulses propagate through junctions until a clock phase boundary is reached. This decreases latency as the pulse can travel though more junctions per clock cycle.

depth.

RQL pipelines are robust against timing errors. The data self-synchronizes to the AC clock signal. At a phase boundary — where junctions are coupled to a clock line with different phase — early pulses wait for the rise of the clock signal in the next section. The jitter accumulates only within one pipeline stage and is negligible. Nominally timed pulses have a window during which propagation is possible. Unlike in RSFQ, pulses do not need to wait for a specific clock SFQ pulse to propagate. The hold-and-release operation in RSFQ fixes the clock speed to a specific value. In RQL, the clock speed can be changed freely up to a maximum (which will be derived in Chapter 3). Timing errors can generally be corrected by reducing the clock speed; every pipe can be operated at a lower frequency and the pulses will wait at the pipe phase boundary. The maximum pipeline depth in terms of Jospheson junctions is determined by the operating clock frequency f_{clock} and the delay time per cell. The delay of a pipeline should be less than approximately one-third of a clock cycle. I give a detailed description of RQL timing in Chapter 3.

2.3 Logic Gates

The routing and processing of pulse-based signals is distinct from transistor-based voltage-state logic. In RQL, logic is performed by routing pulses though an inductive network. The Josephson junctions in the JTLs act as signal repeaters. Considering only the positive pulses, the gates are similar to the state machines of RSFQ logic. The trailing negative pulse erases the internal state every clock cycle. Logic gates are unclocked. The timing and bias depend on input and output JTLs connected to the gate. There are three fundamental RQL gates that form a complete set [28] and thus can be used to build any digital circuit: the AndOr gate, the AnotB gate, and the Set-Reset latch.

2.3.1 AND and OR Gates

Figure 2.5(a) shows a schematic of the AndOr gate. The gate has two symmetric inputs and two outputs. The first pulse the gate receives on either input is routed to the OR output; the second to the AND output. The gate contains two junctions that are connected to the inputs through inductive networks formed by two high-efficiency transformers, k_{12} and k_{34}. The high-efficiency transformers form

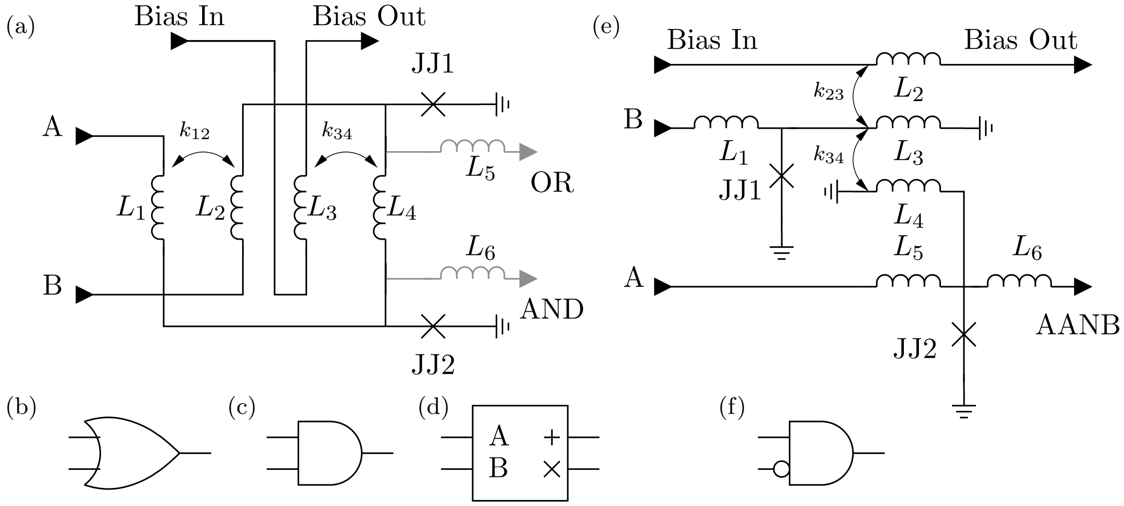


Figure 2.5: RQL Logic Gates. (a) AND and OR logic gate schematic. Inputs A and B are highly coupled through k_{12}. The high common mode inductance drives currents through both junctions upon SFQ input. Flux biasing through k_{34} biases JJ1 to switch first, after which the flux biases JJ2. Low odd-mode inductance between L_1 and L_2 prevents switching junctions from generating backwards-traveling SFQ pulses. (Optional circuit elements shown in grey.) The values of inductors are: $L_1 = L_2 = 26.9\,\text{pH}$, $L_3 = L_4 = 9.8\,\text{pH}$, $L_5 = L_6 = 3.0\,\text{pH}$, $M_{12} = 24.8\,\text{pH}$, $M_{34} = 0.57\,\text{pH}$. The critical currents for the junctions are: JJ1 = JJ2 = 0.141 mA (b–d) Symbols for OR, AND, and the combined AndOr gate. (e) Schematic for the A-and-not-B (AnotB) logic gate. A pulse at B *before* A will reverse-bias JJ2 though the high efficiency k_{34} transformer, inhibiting output. A pulse at A before B will pass though uninhibited. The values of inductors are: $L_1 = L_2 = 3.25\,\text{pH}$, $L_3 = 28.3\,\text{pH}$, $L_4 = 32.3\,\text{pH}$, $L_5 = 4.2\,\text{pH}$ $M_{23} = 0.525\,\text{pH}$, $M_{34} = 15.76\,\text{pH}$. The critical currents for the junctions are: JJ1 = JJ2 = 0.100 mA (f) Symbol for AnotB gate.

a high-inductance differential mode between inputs and a low-inductance common mode between the inputs and the junctions. A pulse at either input will send current through both junctions. The junction JJ1 at the OR-output is preferentially biased by $\Phi_0/2$ flux induced in inductor L_4. This junction will switch when the first input pulse on either input arrives. After switching, the flux state of the gate is reversed and junction JJ2 at the AND output becomes preferentially biased. This means that junction JJ2 will switch if a second positive input pulse arrives. The high differential inductance between inputs prevents propagation of the input pulses from one input to the other. Negative pulses are processed in a similar way, except that junction JJ2 at the AND output will switch first in the case of two input pulses. The first negative pulse will follow the second positive pulse in this case, and the second negative pulse will follow the first positive pulse. This switching does not violate the RQL data encoding, which requires that every positive pulse is followed by a negative pulse approximately half a clock period later, since all positive pulses on the output are followed by reciprocal pulses. The ordering for negative pulses is reversed, though this is only a timing issue and not a logic error.

I note that the AndOr gate does not have an explicit clock bias. The bias current for the junctions is provided from input JTLs. The input JTLs to the gate require special parameters and negligible output inductance. The combination of L_1, L_2, and k_{12} produce a total inductance of 5.1 pH, the same as the JTL would see when connected to another JTL. The AndOr gate parameters are optimized in such a way that the signal is amplified at the input of the gate and there is sufficient bias current to switch the junctions in the gate. The critical margin in the AndOr gate is on the dc bias inductor. The margins on other parameters are more than 50%. The gate has a fan-out of one-half and requires one standard JTL segment at the output to connect to other gates.

The AndOr gate does not have timing restrictions on input signals. If both inputs arrive simultaneously, both junctions switch and produce simultaneous output at the AND and OR outputs. The internal flux state of the gate does not change in this case. The gate can operate either at a phase boundary or inside a single-phase pipeline. Similar to standard JTLs, the input and output JTLs for the AndOr gate have adjusted parameters that compensate for differences in bias current at the phase boundary. The AndOr gate can be used as a stand-alone OR or AND gate. Either (but not both) of the inductors L_5 and L_6 are optional in cases where only one of the two logic functions is desired.

2.3.2 A-and-not-B Gate

Figure 2.5(e) shows a schematic of the AnotB gate. The A-and-not-B (AnotB) gate allows pulses arriving at A to pass through as long as a pulse has not arrived previously at B. The gate consists of two junctions, JJ1 and JJ2, connected through a high-efficiency transformer k_{34}. The high efficiency transformer "negatively" couples the junctions to each other; a positive current through one junction induces a negative current through the other, and *vice versa*. Therefore, when an input pulse arrives at the B input and JJ1 switches, a negative current is induced through JJ2, which inhibits it from switching. In this case an A-input pulse is stored in input inductor L_5 and will annihilate with the reciprocal pulse half a cycle later. In the absence of a B-pulse, junction JJ2 switches with each incoming A-input pulse.

The AnotB gate has a bistable internal flux state corresponding to $\pm\Phi_0/2$. The gate has a DC flux bias line that sets up a positive current through both junctions to ground. When either junction is triggered by an SFQ pulse, the flux state is reversed, which reverses the current through the junctions and inhibits the triggering of the other junction.

The biasing and input/output JTL parameters are the same as for the AndOr gate. However, the AnotB gate has specific timing requirements. The pulse on the B-input has to arrive before or simultaneously with the A-input pulse. The later case can be realized by placing the AnotB gate at a phase boundary. For this reason this gate is often placed at a phase boundary to save on explicit hardware delays.

2.3.3 Set-Reset Gate

The Set-Reset gate shown in Fig. 2.6(a) is the most complicated RQL gate I have worked with. The gate is complicated because the attraction of positive and negative reciprocal pulses makes it difficult to realize internal memory. The Set-Reset gate has an internal state which switches between two bi-stable flux states. A positive pulse is output when the internal state switches to the positive flux state. A negative pulse is output when the state switches to the negative flux state. The state changes only for the first SFQ pulse pair on either input; later pulses do not switch the state or generate output. The waveforms during gate operation are shown

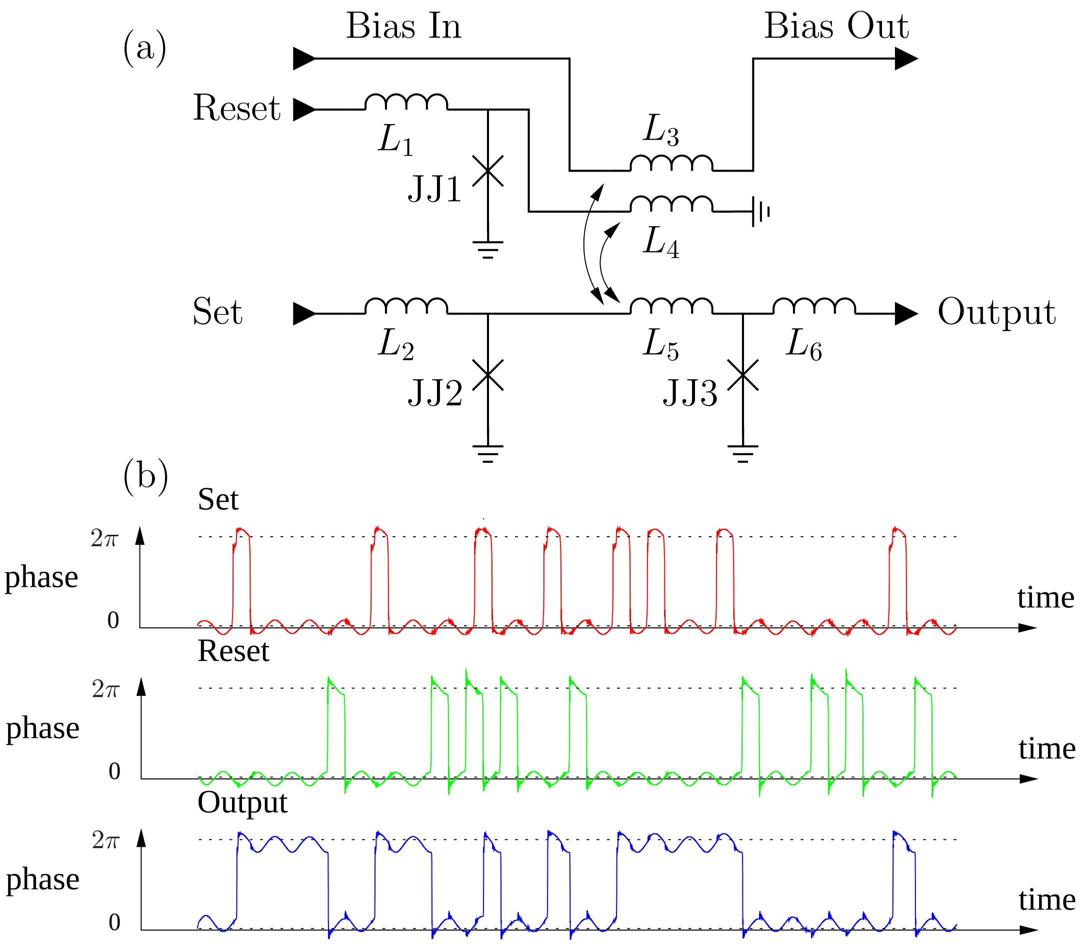


Figure 2.6: Set/Reset (SRS) unit schematic and behavior. (a) Circuit schematic of the Set/Reset gate. The bias induces a flux of $+1/2\,\Phi_0$ though JJ2, L_5, and JJ3 (clockwise current). This puts the unit in the Set state. SFQ pulse pairs that arrive at the Set input when the SRS is in the set state have no effect. A positive pulse at reset will be inverted and output. The trailing negative SFQ pulse changes the internal state to $-1/2\,\Phi_0$. With the SRS in the $-\Phi_0/2$ state, pulses at Reset do nothing while a positive pulse at Set will travel through and the trailing negative pulse will return the internal flux to its original state. The values of inductors are: $L_1 = 3.25\,\mathrm{pH}$, $L_2 = 3.25\,\mathrm{pH}$, $L_3 = 1\,\mathrm{pH}$, $L_4 = 28.3\,\mathrm{pH}$, $L_5 = 32.3\,\mathrm{pH}$, $L_6 = 4.2\,\mathrm{pH}$, $M_{35} = 0.5\,\mathrm{pH}$, $M_{45} = 26.6\,\mathrm{pH}$. The critical currents for the junctions are: JJ1 = JJ2 = JJ3 = 0.118 mA (b) Junction phases across JJ1 (Set), JJ2 (Reset) and JJ3 (Output). Set pulses have no effect unless the most recent pulse was at Reset. Multiple reset pulses have an effect only for the first incoming pulse. Output shows the internal state of the unit.

in Fig. 2.6(b).

The Set-Reset gate consists of three junctions. Junctions JJ2, JJ3, and the inductor L_5 form the Set memory loop. Junction JJ1 and inductor L_4 form the Reset memory loop. Both loops are coupled to each other through the high-efficiency transformer k_{35}. The Set memory loop is initially biased to have $+1/2\,\Phi_0$, so JJ3 is preferentially biased with positive current and JJ2 has a negative current. Junction JJ2 has a small critical current so that it will switch despite the positive flux in the loop. This leaves the internal flux in the Set loop at $+3/2\,\Phi_0$. JJ3 then switches, producing a positive output, and returns the internal flux to $+1/2\,\Phi_0$. The reciprocal pulse switches JJ2 and changes the internal flux from $+1/2\,\Phi_0$ to $-1/2\,\Phi_0$. Any further set pulses will simply change the internal state from $-1/2\,\Phi_0$ to $+1/2\,\Phi_0$ (the leading pulse) and back (the reciprocal pulse) *without* causing any output.

The above process continues until a Reset pulse switches junction JJ1. In this case, $-\Phi_0$ is applied to the Set loop and the flux state in the set loop becomes $-3/2\,\Phi_0$. A negative output follows when JJ3 switches and returns the internal flux state to $-1/2\,\Phi_0$. The following reciprocal Reset pulse switches the internal state to $+1/2\,\Phi_0$. Following Reset pulses will switch the internal state back and forth between $-1/2\,\Phi_0$ and $+1/2\,\Phi_0$, similar to the above behavior of the Set pulses.

For the gate to work properly, the Set and Reset pulses must arrive with a half-phase delay between them. To see why, notice that a positive Reset pulse generates a negative output pulse. In order for this pulse to propagate, it must wait for or be generated during the negative clock cycle on the output. For this reason, the Set-Reset gate must operate on a mixed phase boundary with the Set input on the

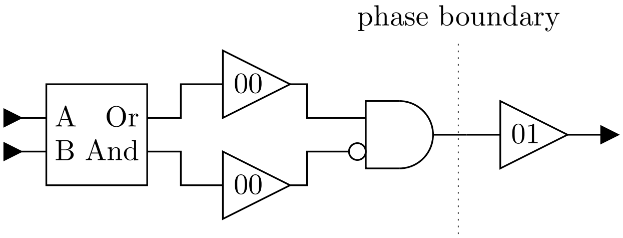


Figure 2.7: RQL Exclusive-OR Gate. An XOR gate can be constructed in RQL by connecting the outputs of an AndOr gate to the inputs of an AnotB gate and placing the AnotB gate output at a phase boundary. A single output from the AndOr gate will propagate through, two outputs will not go through, realizing the logical behavior of the XOR gate.

same phase as the output and the Reset input on a clock phase offset by $\pi/2$. The detailed timing behavior of the Set/Reset gate is discussed in Chapter 3.

2.4 Composite Logic Gates

In this section I briefly describe two logic gates built from more fundamental gate elements.

2.4.1 Exclusive-Or Gate

Figure 2.7 shows the schematic of an exclusive-or gate. An exclusive-or (XOR) gate can be constructed by connecting the outputs of an AndOr gate to the inputs of an AnotB gate, with the OR output going to A and the AND output going to B (see Fig. 2.7). A phase boundary must exist between the input and output. This avoids a “race” condition on the AnotB gate, in which the output of the gate depends on the timing behavior of the JTLs. The gate operates as follows. A single input pulse will produce a single output on the OR-output. The output pulse then travels to the AnotB gate where it must wait until the clock rises on the output JTL (marked 01 in the figure). Provided a second pulse never came, the output of the AnotB gate will not be inhibited. If a second pulse comes, regardless of timing (but on the same clock phase) both pulses will arrive at the AnotB gate and no output will occur. The XOR gate is particularly important for my work because the carry-look ahead adder (Chapter 6) is composed of both AnotB and AndOr gates as in the XOR gate.

2.4.2 Non-Destructive Read-out Gate

The Set-Reset gate is inherently destructive in the sense that any output results in a of the change of the internal flux state. To preserve an internal flux state after multiple read operations we can use a Non-Destructive Read-out (NDRO) Gate. Figure 2.8 shows the schematic of the NDRO gate. In this gate, the Set-Reset gate allows either a positive pulse *or* a negative pulse to be propagated to the B-input of an AnotB gate. Because the reciprocal pulse only follows when the next Reset (or Set) pulse arrives at the Set/Reset gate, the AnotB gate will *remain* in the output-inhibiting flux state. Pulses from the read input will not generate output, nor will the internal flux state change after many read inputs.

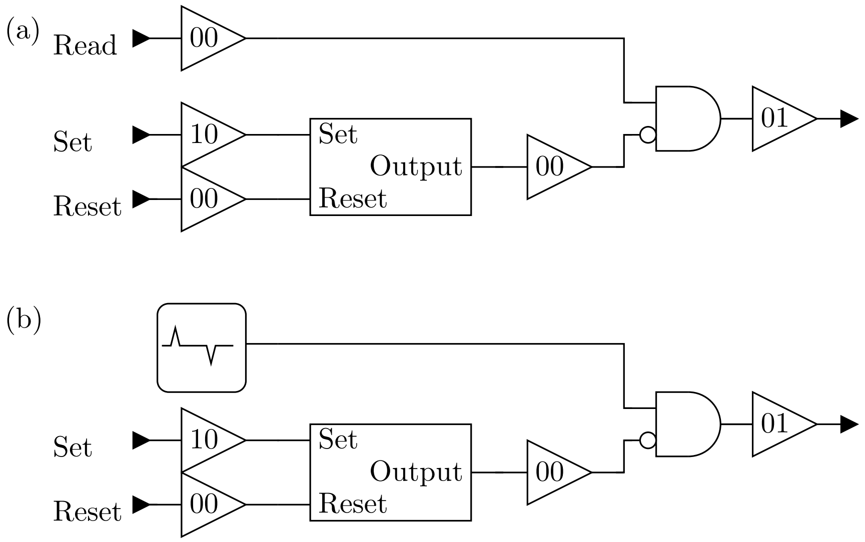


Figure 2.8: Non-Destructive Read-Out Gate. The NDRO gate serves as a memory unit. The Set-Reset gate only outputs a pulse when the internal state changes. By including an AnotB gate the internal state can be determined repeatedly without changing the state. (a) In this configuration a read RQL pulse will be output from the AnotB gate upon input at the Read input, depending on the internal state of the Set-Reset gate. (b) In this configuration a constant series of self-generated SFQ pulse pairs continuously read the state of the Set-Reset gate. This will generate output SFQ pulses every clock cycle until the Set-Reset gate gets a reset signal.

2.5 Fabrication and Equipment

2.5.1 Fabrication

I had test gate circuits fabricated at Hypres using their 4.5 kA/cm^2 process. (Further details of the Nb/AlO$_x$/Nb fabrication process can be found in Appendix E.) These chips were given the name *Monrovia20.* This chip contained experiments for logic and power tests.

Circuits fabricated in Hypres' superconductor fabrication process with 4.5 kA/cm^2 Josephson junction critical current density have a 1.5 μm minimum feature size [29]. (See Appendix E.) The process contains four Nb metallization layers. The second and third layers are used for wiring, Josephson junctions, and gate inductances while the first and fourth metallization layers are used as superconducting ground planes. I designed AC clock lines as microstrips with signal in the first metal layer and ground in the fourth metal layer, connected to the first layer ground through frequently spaced vias. This topology of clock lines gives a high yield because it avoids step coverage problems and film defects in higher metal layers. It also provides a superconducting shield above the signal wire that reduces cross-coupling between adjacent lines. The impedance of the line is limited to 42 Ω because the width is limited to the minimum feature size of the signal layer (2.3 μm) and the SiO$_2$ isolation has a thickness of 850 nm. (A 50 Ω line would be realized with a 2.0 μm wide microstrip in the first layer with a ground plane in the fourth layer.)

The shift register was fabricated in four metal layers by Hypres with 4.5 kA/cm^2 critical current density and 1.5 μm minimum feature size. The junction plasma

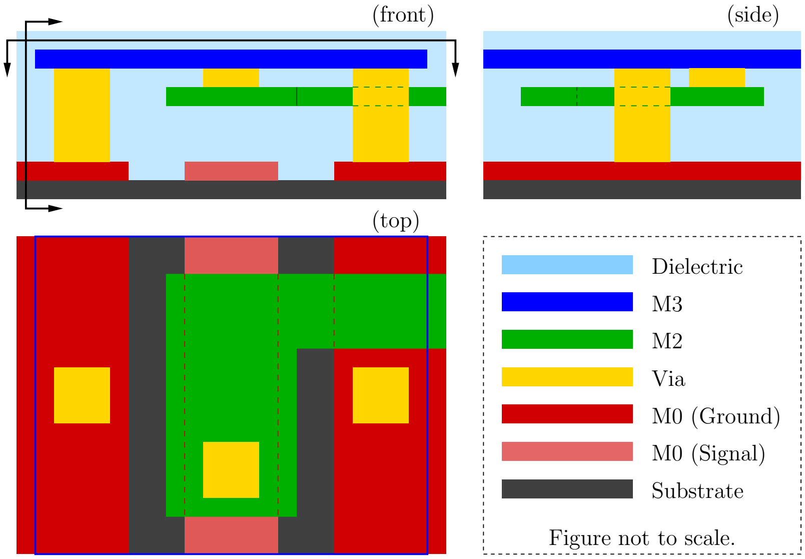


Figure 2.9: RQL Clock Line Transformer Layout. The transformer coupling junctions to clock lines are shown here in three orthogonal views. On the bottom of the chip moats separate the M0 signal line (light red) from the M0 ground plane (red). The M3 layer (dark blue) is connected to the M0 ground plane with vias, creating a grounded skyplane. This creates a microstrip. The transformer is fabricated by depositing a second microstrip in M2 (green) between the M0 signal line and the skyplane. The transformer is grounded on one end by a via connecting to the skyplane. The mutual inductance between M0 and M2 induces currents through the junction (not shown in figure). The M2 transformer extends over the edges of the M0 signal line to ensure that misalignment during fabrication does not affect the mutual inductance between M0 and M2. Cut-away views.

frequency was $\omega_p/2\pi = 250\,\mathrm{GHz}$. This gives a minimum SFQ pulse width of $\tau_p = 1/\omega_p = 3\,\mathrm{ps}$. The clock lines are 2.3 µm wide strips (the minimum width) with 850 µm SiO_2 dielectric thickness to the ground plane[1]. This gives a maximum impedance of 32 Ω. Tapered lines between pads and circuit provided impedance matching. (See Chapter 4 for details on the power delivery network.)

Figure 2.9 shows the design of the transformers that couple the junctions to

[1] The ground plane is the top metal layer in this design

the clock lines. The clock signal is carried in the bottom and first metallization layer (M0). Moats in M0 electrically isolate the signal line from the grounded portions of M0. The fourth and top metallization layer (M3) serves as a local ground plane for signal lines. Vias, between M3 and the grounded portions of M0, ground the M3 skyplane. Bias transformers are formed from the third metallization layer (M2) on top of the clock signal line (M0 Signal). The inductive coupling scales linearly with the length of the transformer and there is small capacitive coupling, typically on the order of 7 fF. The transformer is grounded at one end by a via to M3. The other end (not shown in Fig. 2.9) leads to a grounded Josephson junction, creating an inductive loop with a junction.

Figure 2.10 shows a top view of a similar layout for multiple transformers. The grounds on either side of the three signal lines are connected to the M3 skyplane through vias (shown as gold boxes). In this example the RQL gate is on phase 11. The direction of current is from bottom to top in the clock lines and from top to bottom in the DC bias line. The DC bias current will inductively generate a positive current through the junction, biasing it. Clock I is not utilized here, but supplies current to the next transformer which will use current from Clock I and not Clock Q. During the first half of the clock cycle Clock Q will likewise induce positive current through the junction.

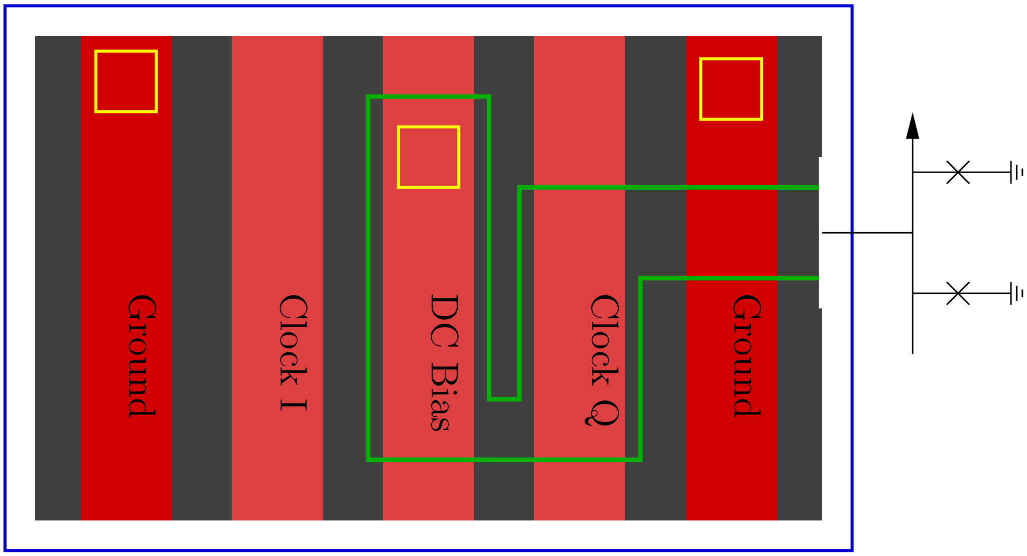


Figure 2.10: RQL Clock Line Transformer with DC Bias. Top-down diagram of clockline transformers to supply flux and current bias to junctions. Two ground planes on the left and right (dark red) connect to the skyplane (dark blue, visible in outline) though vias (yellow squares). Transformer shown in dark green outline. Currents flow upward in Clock I and Q lines, and downward in the DC Bias line. Flux bias pulls current out of junction to bias it at $-\Phi_0/2$. Clock line pulls current out of junction during positive clock phase and pushes current into junction during negative clock phase, making this a phase 11 transformer.

2.5.2 Chip Mounting and Cryogenic Environment

Figure 2.11 shows the overall layout of the cryogenic probe used for measurements on the chip. The probe is inserted into a dewar with a 3 inch wide neck which holds approximately 60 L of liquid helium. (See Fig. 2.11(a).) The liquid helium is at 4.2 K and the chip is completely submerged in liquid. Figure 2.11(b) shows a more detailed schematic view of the probe. Above the dewar, female coaxial connectors attach to stainless steel UT-85 coaxial cables which travel down the neck of the probe to the cold end where they are attached to a printed circuit board (PCB) mount.

The printed circuit board is held by a plastic probe head. The middle of the probe head is open and exposes 48 or 80 gold contact bumps (depending on the probe model). 24 or 40 of the bumps are ground contacts. The remaining 24 or 40 contacts connect to the coaxial cables through the PCB. The chip is placed in contact with these bumps to provide electrical connectivity.

The chip is held in place by a pressure foot. (See Fig. 2.11(b).) Pressure is applied evenly to the chip by means of the pressure foot. The pressure can be adjusted by the pressure screw connecting the pressure foot to the bridge. The bridge is held in place by two screws which are threaded through the bridge and into the probe head. After the chip is securely in place, two μ-metal shields (not shown in Fig. 2.11) are attached to the probe head, one over the other, to exclude magnetic fields from the interior of the probe. Finally, a fiberglass shell is placed around the assembly and held in place with four screws which are threaded through the fiberglass shell and into the probe head.

The above generic description cover both the American Cryoprobe Petersen probe (with 24 signal pads) and the High Precision Devices, Inc., probe (with 40 signal pads). These probes were build-to-order and do not have model numbers.

2.5.3 Experimental Setup

Figure 2.12 shows the overall CAD layout of the *Monrovia 20* chip which I used to test RQL logic gates. This CAD layout was generated by the computer aided

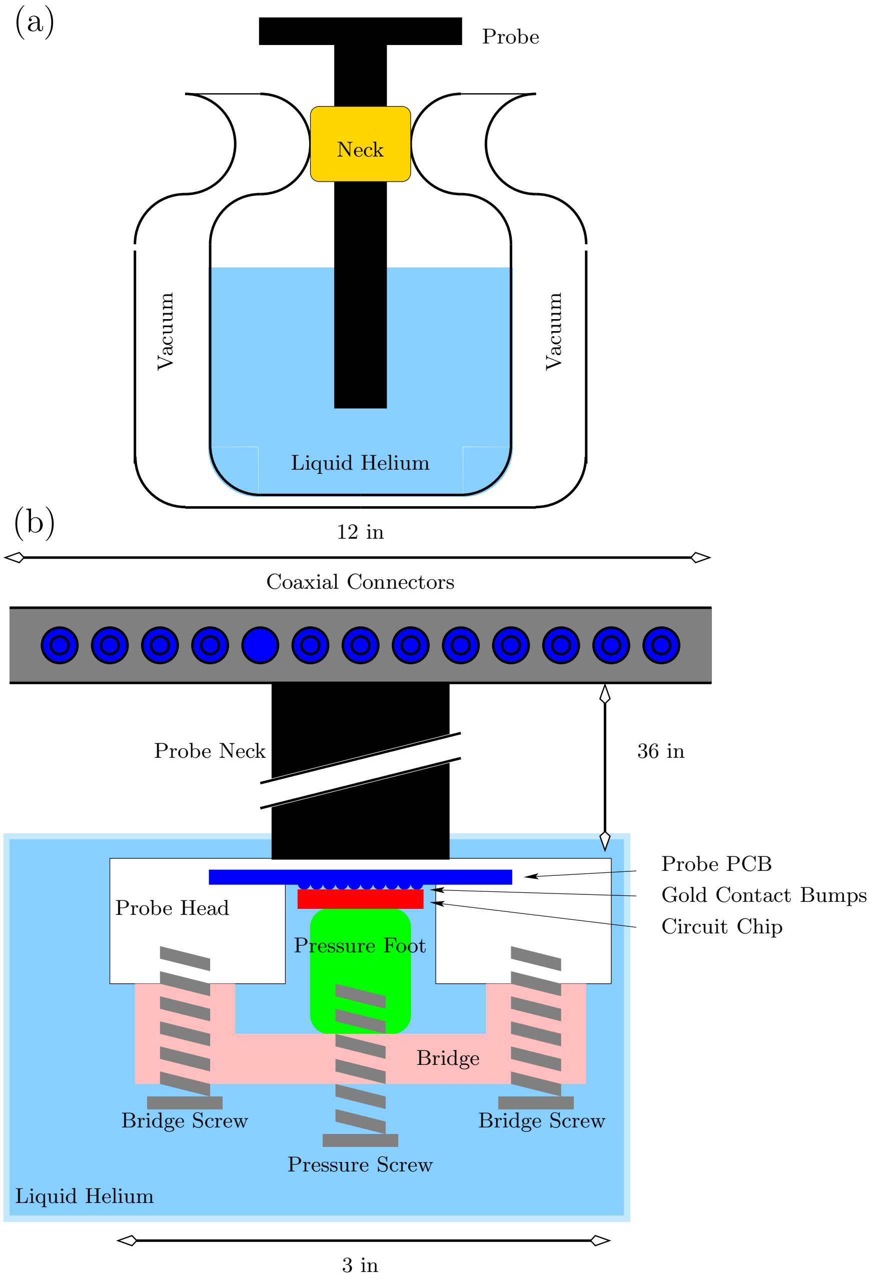


Figure 2.11: Schematic of test probe. (a) The probe used to test chips is cooled by inserting it into a 60 L dewar of liquid helium. (b) Detailed view of the probe. 40 coaxial connectors at the top of the probe lead down the probe neck to the probe's printed circuit board. 80 gold bumps on the PCB serve as pressure contacts to the circuit chip. The chip is held in place by a pressure foot. The pressure of the foot on the chip can be adjusted by the pressure screw. The pressure screw is held in place by the bridge, which is screwed to the probe head by two bridge screws. Figures not to scale.

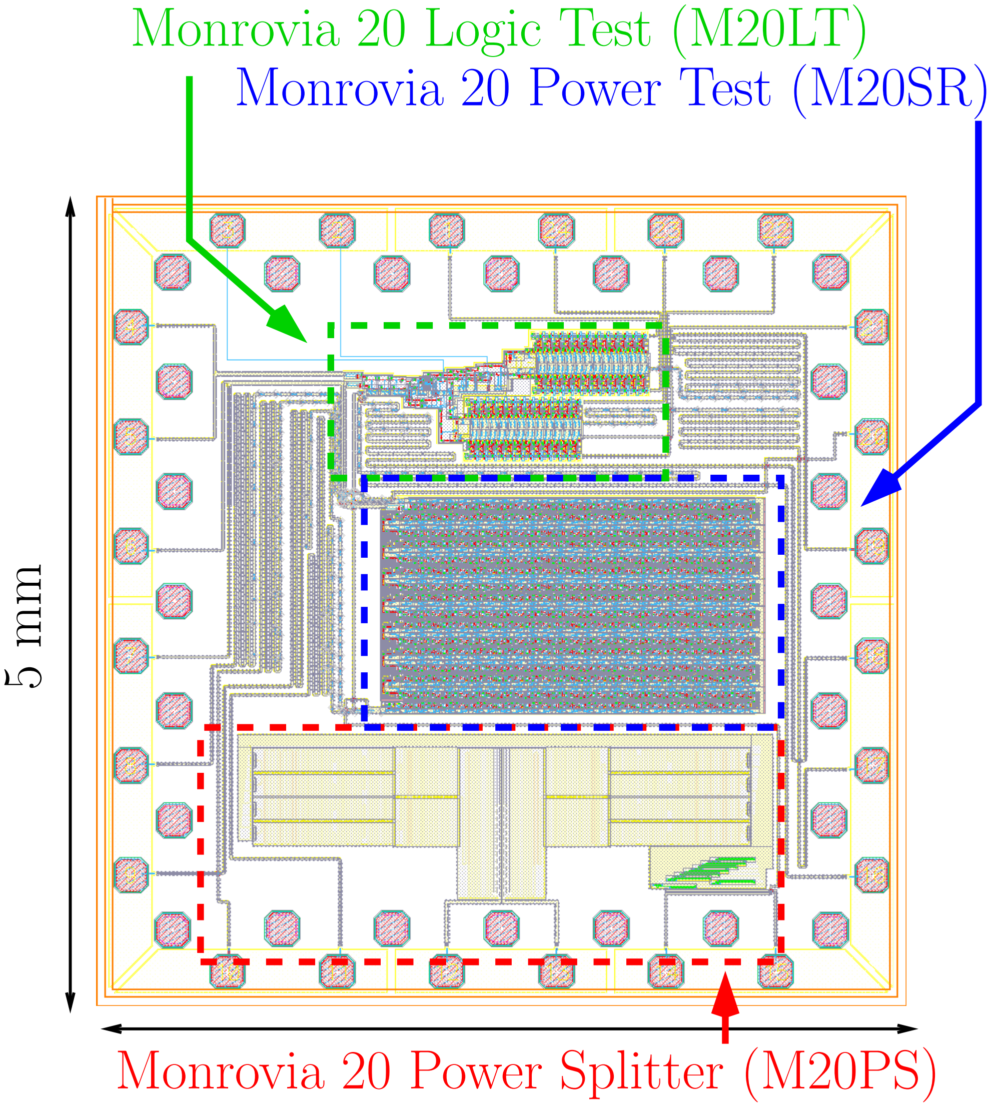

Monrovia 20 Logic Test (M20LT)
Monrovia 20 Power Test (M20SR)
5 mm
Monrovia 20 Power Splitter (M20PS)

Figure 2.12: Layout of Monrovia 20 RQL chip. This chip contains three experimental RQL circuits: the M20LT experiment, which tests the logic behavior of RQL gates and measures the bit error rate of these gates; the M20SR experiment, which measures the effect of switching junctions on phase and amplitude of the clock signal; and M20PS, which is an experiment that tests the behavior of the even mode of a Wilkinson power splitter (see Chapter 4).

design software Cadence.[2] There were three experimental circuits on this chip. The *Monrovia 20 Logic Test* (M20LT) is the subject of the first experiment here and it tested the correct logical operation of RQL gates and the bit error rate (BER) of these gates. The *Monrovia 20 Shift Register* (M20SR) circuit was used to test the phase and amplitude modulation of RQL JTLs. The *Monrovia 20 Power Splitter* (M20PS) was the subject of a third experiment covered in Chapter 4.

Figure 2.13 shows more detailed views of the layout of the logic circuits in M20LT. The input is on the left. The logic circuits can be seen in the middle. The output is on the right, consisting of two large output amplifiers.

Figure 2.14 shows a block diagram for the experimental setup for M20LT and M20SR. When the bit error rate is tested, the oscilloscope is replaced with an Anritsu MP1764C BER detector. The data and clock lines return to room temperature without connecting to the ground on chip. (Each line is inductively coupled to the circuit.) Return lines are marked with (*) on the block diagram. Output data from q0 is generated on chip. One clock generator (#1 in figure) (Agilent Technologies E8275D) generated a synchronization signal for the pattern generator (Anritsu / Hewitt Packard 70843A). The pattern generator operated at a peak-to-peak voltage of 0.25 V–2.0 V. The clock signal to the pattern generator was clocked at twice the speed of the data to provide a return-to-zero (RZ) data input pattern, allowing a maximum data pattern frequency of 6 GHz RZ. The pattern generator also supplied a synchronization signal to the oscilloscope (Tektronix TDS 8000 Digital Sampling Oscilloscope), which always triggered on the beginning of the data input cycle. The

[2] All layout diagrams in this thesis are taken from the Cadence design environment.

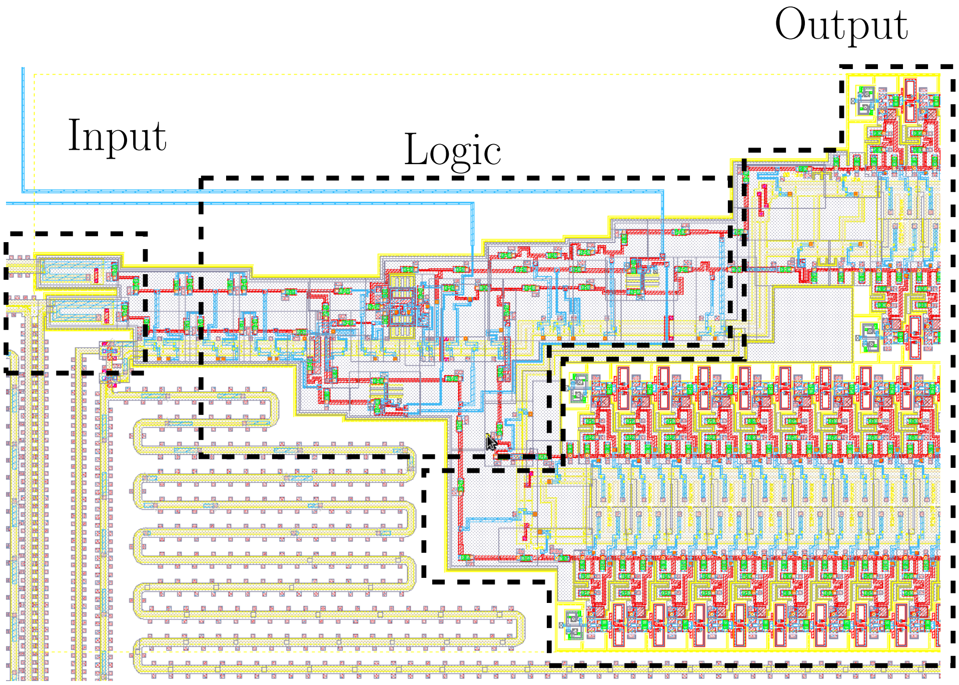


Figure 2.13: Monrovia 20 logic chip. The input consists of two pulse generators triggered off an RZ voltage. The logic section contains four logic gates and 28 JTLs. The output consists of two large amplifiers which produce a return to zero signal at the chip pad.

output of the pattern generator was attenuated by 40 dB to reduce the power going to the on-chip SFQ pulse generator.

A second clock generator (#2) (Agilent Technologies E8275D) was used to feed the on-chip clock lines. The clocks were synchronized through each generators' respective synchronization port. The clock generator had a variable power output and phase for the clock sinusoid. The clock signal was split by a 6 dB splitter before traveling through identical physical delay lines $\tilde{\rho}_1$ and $\tilde{\rho}_2$. Additional hardware delays of unknown electrical length could be added after $\tilde{\rho}_2$. All data input and output lines were connected to the circuit through bias-Ts before entering the probe

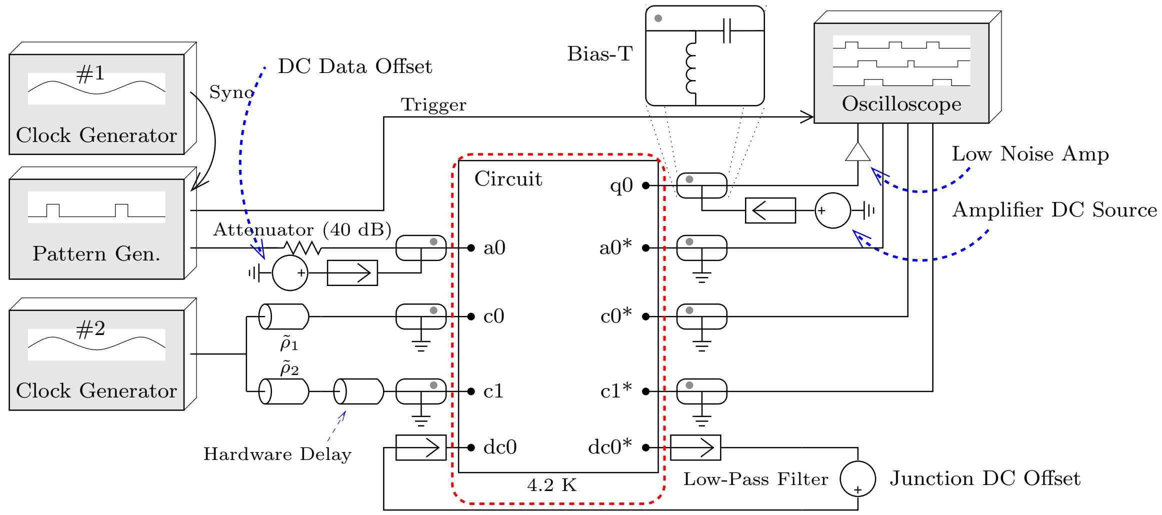


Figure 2.14: Experimental setup for timing experiments. Block diagram of the experimental setup for the timing experiments. a0: data input; a0*: data return; c0, c0*, c1, c1*: clock phases and returns; dc0: DC offset bias; dc0*: offset bias return; q0: experimental output. Low noise bandpass filters have a cutoff frequency of $f_C = 1\,\mathrm{kHz}$. Low noise amplifier is a Miteq LNA with an operation range of 0.518 GHz and a 2.5 dB noise floor.

to reduce noise and isolate the circuit from the measurement equipment. Bias-Ts were also used to isolate the DC offset line, the “Amplifier DC Source” line, and the “DC Data Offset” line (see Fig. 2.14). These were connected to the chip through the bias-Ts and *were* grounded on chip. The output amplifier changed the phase of the final junctions into an RZ signal and required a DC bias. The DC Data Offset supplied an overall DC bias to the data input lines.

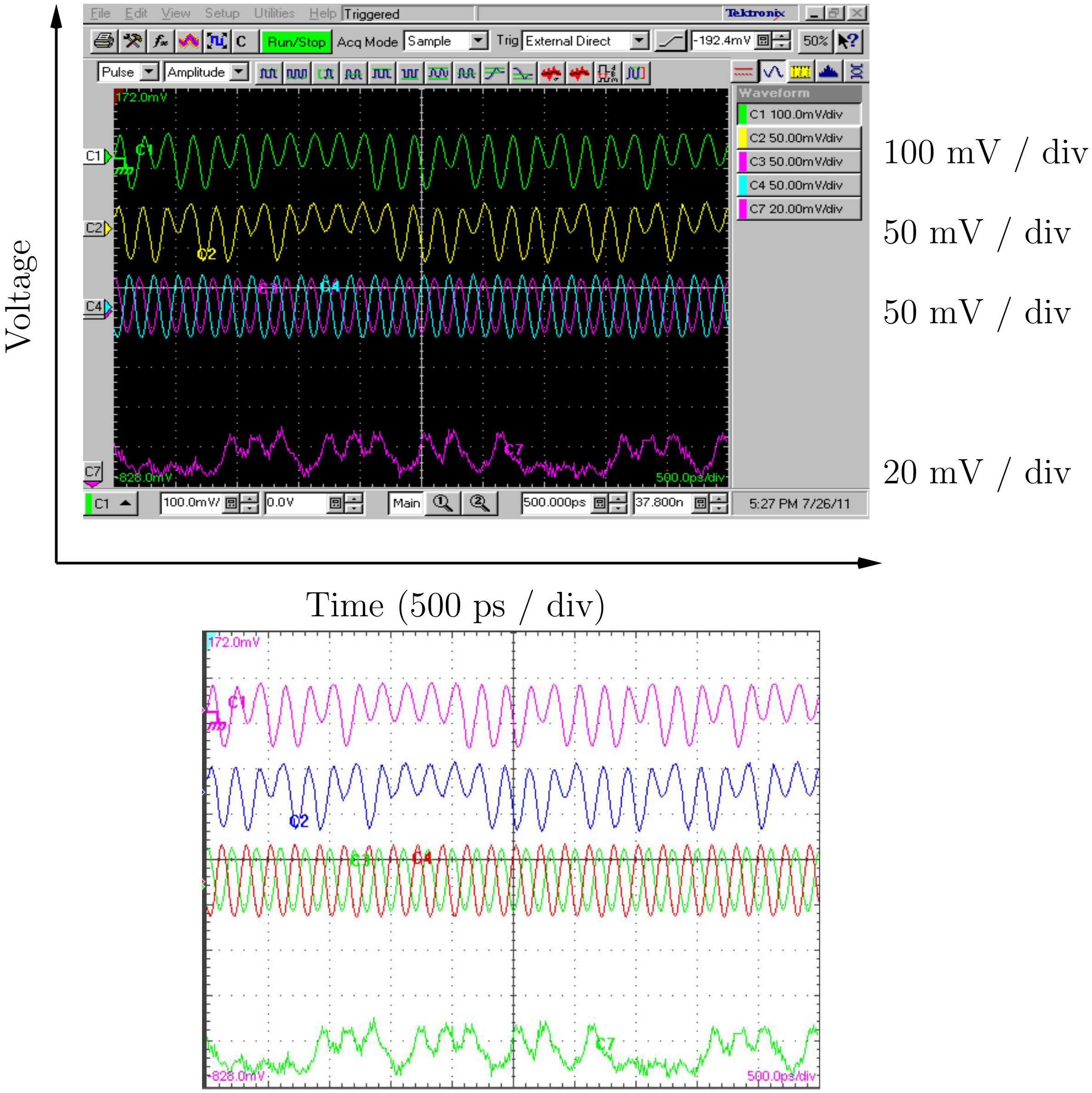
Each circuit under test contained an on-chip output amplifier that generated a DC voltage pulse with approximately 2 mV amplitude. The output voltage pattern is in RZ format, such that the duration of the output pulse is half a clock period. In order to be detectable on the oscilloscope, the output was amplified with a Miteq[3] GaAs low-noise amplifier with a frequency range from 0.5–18 GHz, 20 dB gain, and 2.5 dB noise floor. The output voltage was monitored by an oscilloscope, which could easily detect the 20 mV output switching signal.

Figure 2.15 shows typical data waveforms as recorded on the oscilloscope. Both clock lines were monitored. On the bottom, the output waveform is shown. Typically, there is a certain amount of cross talk between clock line and data output lines due to the coupling between lines on the printed circuit board (PCB) used to make connections to the chip. This crosstalk can be later subtracted from the signal during data analysis. Each output was fed into the oscilloscope, except the DC phase offset.

2.6 Experimental Verification

Routing and processing of pulse-based data in RQL is different from what is used in conventional transistor voltage-based digital logic circuits. While CMOS logic families are sensitive to rise and hold times, pulse-based logic depends on the sequence of arrival of pulses. For example, for the RQL AnotB gate, the B pulse must arrive sufficiently before an A pulse to function properly, even within the same clock phase. As another example, the AndOr gate sends pulses first to the OR output, then (if applicable) to AND. In general, in RQL pulses are transient and are only held at clock phase boundaries. Unlike CMOS transistors, where previous logic

[3] This is a Miteq ASS4-00501800-25-5P-4 with an operation range of 0.5–18 GHz and a 2.5 dB noise floor. All references in this thesis to a Miteq amplifier refer to this model with this performance.



Triggered

Acq Mode Sample

Trig External Direct

-192.4mV

50%

Pulse

Amplitude

Waveform

C1 100.0mV/div

C2 50.00mV/div

C3 50.00mV/div

C4 50.00mV/div

C7 20.00mV/div

C1 100.0mV/ 0.0V

Main

500.000ps 37.800n

5:27 PM 7/26/11

Figure 2.15: Oscilloscope output. Typical output from the Tektronix TDS8000 Digital Sampling Oscilloscope while measuring the RQL circuit. Top two rows are input signals returned from chip. The third row is the two superimposed clock signals returning from chip. The bottom row is output from the RQL chip, where each peak represents a single reciprocal SFQ pulse pair. Measurable voltages ranged from 10 mV / div upward, giving 0.1 mV resolution. Measurable times ranged from 50 ps / division, allowing 1 ps time resolution. Bottom image is a color-inverse of the output, in which the divisions can be seen.

operations have lasting effects on output, correct logical operation in RQL depends on the depth (number of junctions) of each clock phase to ensure synchronized inputs.

The use of pulse-encoding also affects power dissipation and the bit error rate. CMOS operates with constant voltage states, while the voltages in RQL are transient. In both CMOS and RQL, power is expended only during switching, and then only for digital "one." CMOS transistors only expend power when switching, though this has a much different meaning in the technology. RQL junctions switch for every "one" bit, regardless of previous inputs. CMOS only switches for a change from "one" to "zero" (or vice versa). For constant inputs this gives a significantly different number of power dissipating events. For random bits, though, the number of power-dissipating events in CMOS or RQL will be approximately the same.

To find the bit error rate in RQL, I note that the error condition is very well defined. Since SFQ pulses are exactly quantized amounts of flux, there is no threshold voltage as in CMOS. Voltage and current in RQL are defined by the junctions and are not design parameters as in CMOS. In RQL, errors constitute the presence of SFQ pulses in a clock period where there should be none, or the absence of SFQ pulses during a clock period in which they should have been present.

Before designing circuits that used multiple gates, I checked the operation of individual gates. I also did simulations and then indicated a broad range of operating margins. Experimental tests (see Chapter 5) have shown that RQL gates behave much as expected and in some cases actually perform better than expected. In the remainder of this chapter, I describe three experiments to verify the logical

Table 2.1: Universal Logic Test for RQL gates. Two inputs, A and B, lead to four possible output conditions from the logic gates AnotB, OR, XOR, and AND.

Input		Output			
A	B	AnotB	OR	XOR	AND
0	0	0	0	0	0
1	0	1	1	1	0
0	1	0	1	1	0
1	1	0	1	0	1

operation of RQL gates, measure the power dissipation, and determine the bit error rate of the AnotB gate.

2.6.1 Logic Operation Test

I tested the basic logic gates (AndOr and AnotB) with the simple circuit (named M20L) shown in Fig. 2.16(a). For the AndOr gate, two bits of input — A and B — correspond to four possible input conditions. I synthesized the XOR gate by feeding the AndOr gate outputs into the AnotB gate. Because the AnotB gate is on a clock phase boundary, output is inhibited (for any input combination) until the beginning of propagation on the next clock phase; the order of A and B input pulses is unimportant as both will arrive before output occurs. The expected results for each gate are given in Table 2.1.

A block diagram of the test circuit is shown in Fig. 2.16(b) with the SFQ generating junctions shown on the left and amplifiers shown on the right. The SFQ launches convert a return-to-zero (RZ) voltage into SFQ pulses. The output amplifiers are similar to [27] and provide a 2 mV RZ output signal. The logic circuit was designed to operate at 4.2 K with speeds up to 20 GHz. I used an Anritsu pattern generator to supply the input pulses and this limited me to 6 GHz. Figure 2.16(c) shows a representative output of the sampling oscilloscope, which shows the input on the top two lines and the four outputs on the bottom four lines. The room-temperature amplifiers invert the signal, causing positive SFQ pulses to produce downward spikes instead of upward (as on the input). Careful examination of Fig. 2.16(c) reveals the outputs correspond to the expected results (see Table 2.1).

I also measured the operating margins on the clock power. I found that the clock power could be varied by ±25% without producing errors, limited by output amplifiers on the low end. At the high end, an excessive clock amplitude causes all junctions to switch and generate output, regardless of input. The total latency through the circuit is one clock cycle. The results shown in Fig. 2.16(c) were performed at a speed of 6 GHz, the upper frequency limit of the pattern generator. Testing at lower frequencies produced the same logical output, though margins on the clock power were larger. At lower clock frequencies the switching time of the junctions becomes a smaller fraction of the clock period. The triangular peaks seen in Fig. 2.16 become rectangular patterns. However, the logical operation is the same.

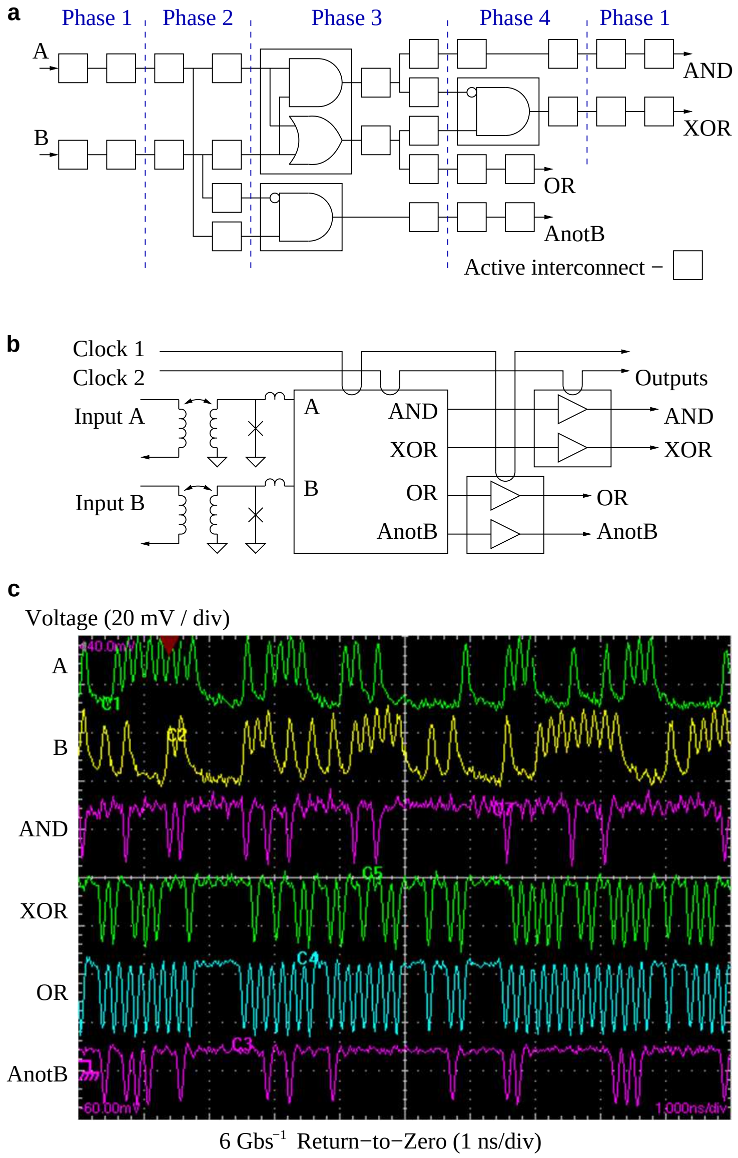


Figure 2.16: Logic Test of Basic RQL Gates. (a) Block diagram of RQL logic test circuit. Two inputs, A and B, enter from the left. Signals are split and sent through five total clock phases, emerging on the left. One AndOr gate and one AnotB gate shown in Phase 3. One AnotB gate shown in Phase 4. Small blocks represent active interconnect JTL units. Four logic operations are synthesized in this circuit. (b) Cold stage wiring diagram of (a). Two clock lines couple to the junctions in the circuit. Inputs on right made via two coupled lines which bias junctions. Clocked output amplifiers (different from JTLs) shown on the right. (c) Oscilloscope output of (b) after room temperature low-noise amplifiers have increased the outputs to the millivolt scale. Amplifiers provide arbitrary amplification and invert signal. Top two traces are inputs A and B; bottom four are logical outputs corresponding to above inputs.

2.6.2 Clock Power Measurement

To better understand the power dissipation in RQL circuits, I tested a 200-bit shift register. This device had 1600 Josephson junctions, and is named the Monrovia 20 Shift Register (M20SR). (The shift register is similar to Fig. 2.3 repeated 100 times is sequence. Figure 2.12 shows a layout of the chip.) In RQL circuits the AC power is delivered on $50\,\Omega$ microstrip clock lines that return to room temperature without termination on chip. The clock lines are inductively coupled to the circuit so the Josephson junctions are effectively biased in series. This allows direct measurement of the amplitude of the clock for both active and inactive circuits. To make power measurements easier, I designed the shift register with many junctions and high coupling between the clock line and junctions. This was necessary because RQL circuits operate with low AC power amplitude and small dissipation in the junctions; the junctions load the clock line only when they switch to the resistive state. With high coupling between the junctions and the clock line, more power will be drawn from the clock line than would regularly be the case. The loss of power will be reflected in a decreased amplitude of the returning clock signal. This setup, although it does not have logic gates, will show the change in amplitude clearly. The clock signal attenuation and phase delay due to the RQL gates scale as the square of the coupling coefficient k_2 and can be minimized by reducing coupling to the clock line and increasing AC clock power. In real RQL circuits, these parameters are chosen to allow at most 10% attenuation and less than 2 ps phase delay in a circuit with 10^6 Josephson junctions [27].

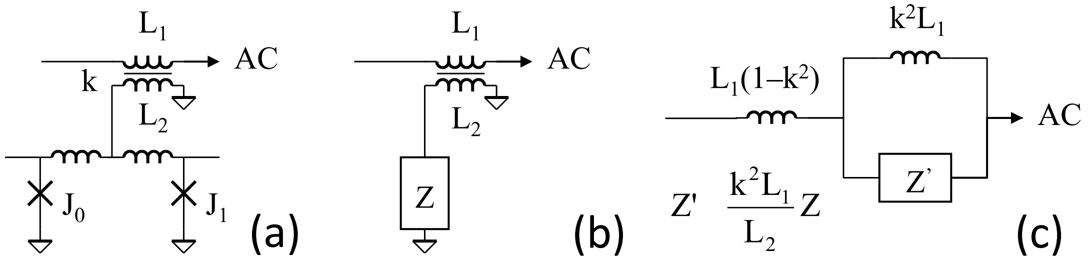


Figure 2.17: Power schematic for RQL delay line: (a) detail schematic; (b) equivalent block diagram, where Z is impedance of RQL gate; (c) equivalent parallel circuit.

AC power losses in Nb microstrips are quite small, on the level of 1% loss per wavelength up to the gap frequency of 700 GHz [30, 31]. In a practical circuit with multiple parallel lines, AC losses can be an order of magnitude less than dynamic power dissipation in the gates. Use of microstrips, as opposed to coplanar wave guides, is essential for integration with digital circuits since such circuits require multiple crossings and couplings to the gates. However, line impedance of microstrips in general favors sub-micron processes, currently only developed by two research groups [32, 33].

Figure 2.17 shows the equivalent circuit for a junction that is switching in an RQL circuit. Switching junctions change the voltage and currents on the clock line and also affect the impedance of the coupled clock lines. In a simple linear model, the junction acts as a perfect superconductor before it switches, and it behaves as a resistor after it switches. Thus, in the case of all digital "ones" we can treat the junctions as resistors, and the clockline time constant (speed) is simply

$$\tau = \sqrt{L_c C_c} = 7.6\,\mathrm{fs}/\mu\mathrm{m} \tag{2.1}$$

where I have used our clockline geometry with $L_C = 0.3\,\mathrm{pH}/\mu\mathrm{m}$ and $C_C = 0.29\,\mathrm{fF}/\mu\mathrm{m}$. For digital "zero" the junction can be treated as an inductance and the clockline time constant becomes $\tau' = \sqrt{L'_C C_C}$ where

$$L'_C = (1-k^2)L_C + k^2 L_C \frac{L_g}{L_g + L_T}. \tag{2.2}$$

Here the magnetic coupling constant is $k = L_{mutual}/\sqrt{L_C L_T}$ and the inductance of the RQL gate attached to the bias inductor is L_g. For JTL elements in a shift register configuration, L_{JJ1} and L_1 are in series, L_{JJ2} and L_2 are in series, and one finds for the parallel combination of these two series inductors

$$L_g = \frac{(L_{JJ1} + L_1)(L_{JJ2} + L_2)}{(L_{JJ1} + L_1) + (L_{JJ2} + L_2)} \tag{2.3}$$

where the junction critical inductances $L_{JJ1} = \Phi_0/2\pi I_{c1}$, and $L_{JJ2} = \Phi_0/2\pi I_{c2}$. The effect of the coupling can be seen in (2.2). The accumulated clock delay though 1600 junctions on the return clock signal on the oscilloscope when data was input and compared to the delay when no data was input. The measured delay was $1.4 \pm 0.2\,\mathrm{ps}$ for the whole chip and was independent of frequency from DC to 6 GHz.

Figure 2.18 shows the results of taking the difference of output clock power of all "ones" to all "zeros" in the shift register. The measured dissipation is 1.35 times higher than that expected from simulations but three times lower than the maximum switching power of $2I_c\Phi_0$ energy dissipation per digital "one" with average $I_c = 170\,\mu A$. The power dissipation in RQL circuits with random data can be approximated as

$$P = \frac{1}{3} I_c \Phi_0 \, N \, f, \tag{2.4}$$

where N is the number of junctions in the circuit, I_c is the weighted average critical current of the junctions, and f is the operational clock frequency. The fraction of 1/3 is due to the behavior of SFQ pulses under AC biasing. Instead of switching at the critical current I_c, the junctions switch earlier in the clock phase at bias current $I_b < I_c$. With increasing phase the junctions switch at larger bias currents, leading to a slight non-linearity in the relationship between power and frequency, as shown in Fig. 2.18.

Current Intel i7 processors demand 8 Amps at 12 V, or a power of about 96 W, for approximately 731 million transistors [34], or approximately 130 μW per transistor at approximately 3 GHz. RQL operates at approximately 0.5 μW per junction at twice the speed, as shown in Fig. 2.18. A direct comparison is not possible because CMOS CPUs by design do not utilize all transistors at once. Some architectures however minimize the execution time of instructions by utilizing as many transistors as possible, leading to a 6% increase in speed at the cost of a 16% increase in power consumption [35]. However, an estimate will show the scale of RQL junction power consumption. RQL logic operations require about four junctions total. Normalized to logic operation count and clock speed, RQL is still more than 100 times more power efficient compared to current CMOS technology.

2.6.3 Bit Error Rate

Another key performance metric in testing RQL gates is the bit error rate. Many mechanisms can lead to failure in a circuit. In many cases, a reduction of clock

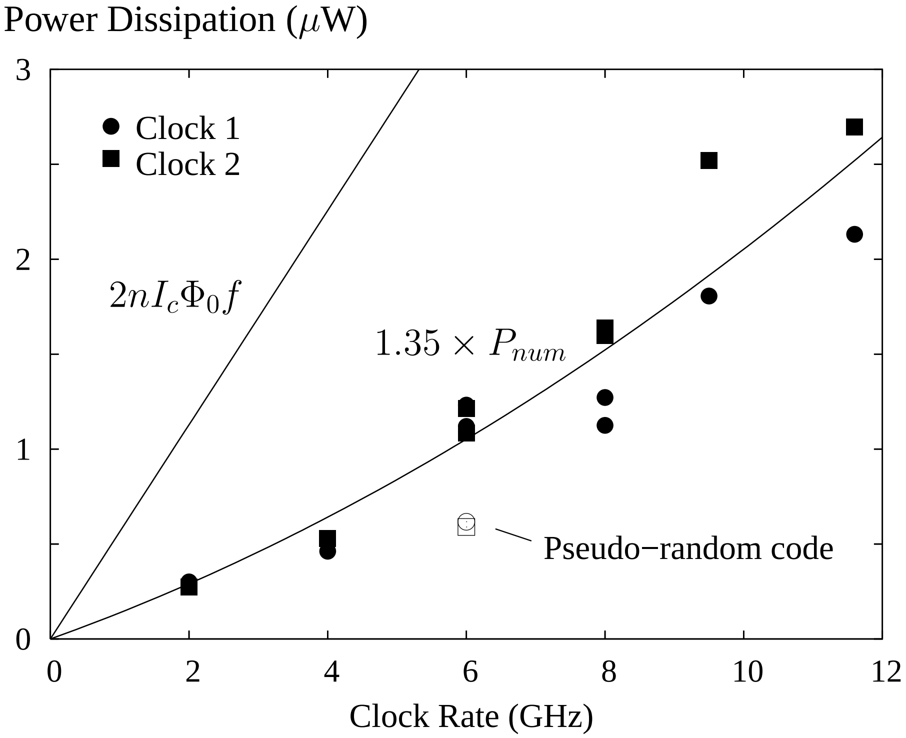


Figure 2.18: Power Dissipation Measurements. Measured dissipation on both clock lines for M20SR. Filled circles and squares show measurements performed by comparing a continuous sequence of 0s versus 1s. Open circle and square show power dissipation for a pseudorandom sequence of bits and 0s, which is approximately at half the value found for the full 1s measurements at 6 GHz. The expected power dissipation based on RSFQ estimates is given by $2nI_c\Phi_0 f$, shown as the straight line. Measured data is fit to a power law and indicates power dissipation approximately 1.35 times higher than in simulation. Slight increase of power rate with frequency in measured data is due to higher average biasing currents during switching for higher frequencies. $I_c = 170\,\mu$A.

frequency will reduce the rate of errors. In other cases, the failure is independent of clock speed. For example, the flux bias creates symmetry between positive and negative SFQ pulses. If the flux bias is too low, pulses fail to propagate and digital "one" reads as "zero." High flux bias has the opposite effect, with "zero" reading as "ones." In general, excessive current causes junctions to switch even in the absence of an SFQ pulse.

The AnotB gate is particularly sensitive because it not clocked and as a result its operating margin depends strongly on the flux bias. I tested an AnotB gate (see Fig. 2.12) by observing the XOR output of circuit M20LT. At a clock speed of 6 GHz I monitored the XOR output while changing the flux bias near failure. A 32-bit input pattern from an Anritsu MP1763C was split and applied to the inputs with a 15-bit relative shift between A and B. The XOR output was compared to the correct pattern with an Anritsu MP1764C error detector. I could operate this setup for no more than 30 hours due to drift of the synchronization signal between the generating and measuring units. This set a lower bound on measuring the bit error rate of about 10^{-15}.

Figure 2.19 shows results I obtained from these measurements. The solid curves in Fig. 2.19 shows fits of the measured error rate to a Gaussian distribution

$$p = \frac{1}{4}\mathrm{erfc}\left(\frac{\pm(I - I_t)/20}{\sqrt{2}\,\sigma_I}\right) \tag{2.5}$$

with the two fit parameters I_t, the current threshold, and σ_I, the root-mean-squared noise current. The factor of 1/4 occurs because only "ones" create readout errors and the factor of 1/20 represents the amount of coupling between the applied flux current

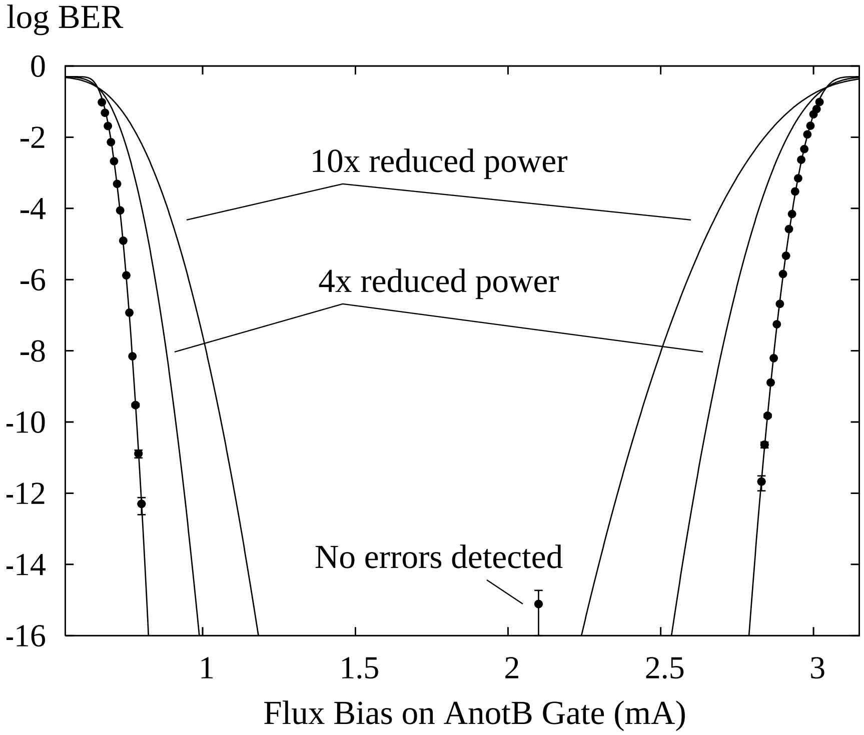


Figure 2.19: Bit Error Rate for AnotB gate from the M20LT circuit at 6 GHz as a function of flux bias I_{flux}. Very broad operating margins even for low BER of 10^{-44}. Error bars on the lowest points correspond to counting statics of 4 errors and 5 errors (left and right). No errors detected for a period of 30 hours given an error floor below 10^{-15}. Data fit extrapolates to a BER of 10^{-480} at optimal bias of 1.82 mA. Curves scaled for decreased size and power.

and the resulting flux induced through the junction. For the low bias error (left curve), I found $I_t = 0.66\,\text{mA}$ and $\sigma_I = 1.02\,\mu\text{A}$; for high bias (right), $I_t = 3.04\,\text{mA}$ and $\sigma_I = 1.56\,\mu\text{A}$. No errors were detected for a period of 30 hours at a bias of about 2.1 mA. This included errors in the chip and the entire measuring apparatus. At the extrapolated optimal bias point of 1.82 mA the expected extrapolated bit error rate, based on these measurements, is 10^{-480}. A bit-error rate of 10^{-44} is considered the norm in CMOS. From the extrapolation, at a bit error rate of 10^{-44} the flux bias margins for a nominal flux bias of 1.75 mA will be 30% on either side. Our extrapolated optimal bit-error rate is phenomenally smaller, and of course it is a very large extrapolation. Nevertheless, this suggests that this RQL circuit is performing well.

2.7 Summary

RQL uses positive and negative pairs of SFQ pulses to encode digital data and performs logic by routing the pulses. This makes logic operations both energy and size efficient, as well as fast. The AndOr and AnotB gates compose a universal set. The NDRO gate provides a form of memory. JTLs provide connections between gates. I found an extrapolated BER of 10^{-480} at 4.2 K on the output of a synthesized XOR gate, which is far below the minimum error rate I could measure, 10^{-15}. Noise current scales as the square root of the Josephson critical current. This gives a negligible BER of 10^{-44} with operating margins of ±30% on flux bias.

Chapter 3

Combinational Gates

3.1 Introduction

The complexity of modern digital circuits necessitates the use of computer aided design. Computer aided design also allows for simpler ways to describe digital circuitry behavior than what would be found from detailed physical simulations, and yet still encompasses the full range of possible behavior. One standard tool in common use is *VHSIC Hardware Description Language*, or VHDL for short. This standard language (one of only two commonly used by the semiconductor industry) is used to design nearly every CMOS digital circuit. By casting RQL into the formalism of VHDL, I can enable the great range of existing design tools and aids developed for CMOS to be applied to superconducting digital logic.

In the previous chapter, I focused on the behavior of individual junctions and gates, and described the qualitative timing requirements of RQL gates. In this chapter I provide a quantitative approach to timing in RQL circuits. First, I derive an analytic expression for the timing of a single junction. This analytic expression allows me to understand the self-correcting timing behavior in RQL and the upper frequency limit of operation. I then fit the analytic equation to simulation data to produce three parameters which describe the timing behavior of a gate at a certain frequency. Finally, I build VHDL models using the timing parameters and combinational logic.

3.2 Junction Switching Time under AC Bias Current

In this section I examine the switching of junctions in RQL circuits and find an analytic equation for the switching time. Using this equation, I show how the non-linear behavior of the junctions leads to stable, jitter-free pulse propagation. I also derive a failure condition to determine the maximum operational speed of RQL circuits. The testing of those models on real RQL circuits is described in Chapter 5.

3.2.1 Analytic Equation for Switching under AC Bias

The switching of junctions in RQL circuits depends on the clock frequency, clock amplitude, and junction I_cR_N product. Here I extend the analysis of characteristic junction switching time for constant biasing to include the case of time-varying bias currents. In particular, I will assume $I_b = AI_c \sin(\omega t)$, where A is the maximum bias current in units of I_c, and find the average switching times for the range of bias conditions. Under the assumption that the junction is overdamped (which the junctions in my RQL circuits with $\beta \approx 1$ approach), the switching time τ of Josephson junctions with shunt resistor R_N can be expressed as[1]:

$$\tau = \frac{L_J}{R_N}. \tag{3.1}$$

[1] Note that in these RQL circuits, the junctions have been shunted by a resistor R_S which is much smaller than the junctions' own resistance R. The junction resistance R_N in the RSJ model is the parallel combination of R_S and R, and because $R_S \ll R$, $R_S \approx R_N$.

For $I_b \ll I_c$ the junction inductance L_J during switching can be approximated as [6]

$$L_J = \frac{\Phi_0}{2\pi\, I_c}\frac{\phi}{I_b/I_c} = \frac{\Phi_0}{2\pi\, I_c}\frac{\arcsin(I_b/I_c)}{I_b/I_c}, \tag{3.2}$$

where $\phi(t)$ is the initial phase across the junction at time t and assuming it has not already switched. Combining (3.1) and (3.2), the switching time becomes

$$\tau = \frac{L_J}{R_N} = \frac{\hbar}{2\,e\,I_c R_N}\frac{\phi}{I_b/I_c} = \frac{\Phi_0}{2\pi}\frac{1}{I_c R_N}\frac{\phi}{I_b/I_c} = \mathrm{t}_0\frac{\phi}{I_b/I_c}, \tag{3.3}$$

where $\mathrm{t}_0 = \Phi_0/(2\pi\, I_c R_N)$ is the characteristic switching time. The quantity t_0 depends on the fabrication process though the $I_c R_N$ product. Table 3.1 compares $I_c R_N$ and t_0 for different fabrication processes.

Equation (3.3) can also be written as

$$\tau\, A \sin(\omega t) = \mathrm{t}_0\, \phi. \tag{3.4}$$

Equation (3.4) implies that the switching time depends on the time t when AC power biases the junction. In order to account for this we find the average static bias switching time between the time at the beginning of dynamic switching t_{in} and the time and the end of dynamic switching t_{out}. This dynamic switching time $\tilde{\tau} = t_{out} - t_{in}$ can be calculated as follows,

$$\frac{1}{\tilde{\tau}}\int_{t_{in}}^{t_{out}} \mathrm{d}t\, \tilde{\tau}\, A \sin(\omega t) = \frac{1}{\tilde{\tau}}\int_{t_{in}}^{t_{out}} \mathrm{d}t\, \mathrm{t}_0\, \phi(t), \tag{3.5a}$$

$$\int_{t_{in}}^{t_{out}} A \sin(\omega t)\mathrm{d}t = \frac{1}{\tilde{\tau}}\int_{t_{in}}^{t_{out}} \mathrm{t}_0\, \phi(t)\mathrm{d}t = \mathrm{t}_0\, \bar{\phi}, \tag{3.5b}$$

where $\bar{\phi}$ is the time-average phase across the junction. The time-averaged value of the phase across the junction is $\bar{\phi} \leq 3$ rad to first order [4]. It is useful to express the time in radians of the AC clock to make the results independent of the clock frequency. Accordingly, I define $\theta = \omega t$ and thus $\mathrm{d}\theta = \omega \mathrm{d}t$. I then solve for the output clock phase $\theta_{out} = \omega t_{out}$ as a function of $\theta_{in} = \omega t_{in}$ by noting that Eq. (3.5b) gives

$$\int_{\theta_{in}}^{\theta_{out}} A \sin(\theta) \frac{\mathrm{d}\theta}{\omega} = 3\,\mathrm{t}_0. \tag{3.6a}$$

I can then write:

$$\int_{\theta_{in}}^{\theta_{out}} \sin\theta \, d\theta = \cos\theta_{in} - \cos\theta_{out} = \frac{3\,\omega\,\mathrm{t}_0}{A}, \tag{3.6b}$$

and thus

$$\theta_{out} = \arccos\left[\cos(\theta_{in}) - \frac{3\,\omega\,\mathrm{t}_0}{A}\right]. \tag{3.6c}$$

For overdamped junctions with $I_b = I_c$ the switching time is approximately $t = 3\,\omega_c^{-1} = 3\,t_0$ [4]. I now define a new quantity

$$\delta = 3\,\omega\,\mathrm{t}_0/A. \tag{3.7}$$

From Eq. (3.6c) we can see that this is the switching time of a DC-biased junction with bias current $I_b = AI_c$, normalized to the clock period $T = 2\pi/\omega$. The quantity δ represents the integral of a normalized voltage over a normalized time period. A constant bias current A would result in a junction switching over a normalized time period of $\theta_{out} - \theta_{in}$. As I will show later in this chapter, it turns out that δ is a good metric of circuit behavior. For example, for $\delta < 1$ SFQ pulses in RQL circuits wait at phase boundaries, whereas for $\delta > 1$ the pulses can be free-running through the circuit.

Table 3.1: Comparison of different J_c, I_cR_N and switching time t_0 fabrication technologies.

	Hypres	*Hypothetical Future Process*	
J_c	4.5	10	kA/cm^2
I_cR_N	0.75	1.00	mV
t_0	0.44	0.33	ps

The amount of change the phase (normalized time) changes from beginning to the end of switching is $\nu = \theta_{out} - \theta_{in}$. Using Eq. (3.6c), the phase delay can be written as a function of input phase θ_{in}

$$\nu(\theta_{in}) = \arccos\left(\cos\theta_{in} - \delta\right) - \theta_{in}, \qquad \theta_{in} > 0. \tag{3.8}$$

Figure 3.1 shows this function for several different clock frequencies. There are a few interesting things that can be understood from this plot. Notice that as the frequency increases, so does the relative phase delay ν for any given input phase, and the curves end at lower and lower input phases. Notice also how the delay increases rapidly for large θ_{in}. The implication is that pulses arriving at sufficiently large θ_{in} will not propagate. Also, note that none of the curves cross, meaning the timing behavior is uniquely determined. Finally, at low clock speeds the switching time is far shorter than the clock period and the change in bias current plays only a small role. That is, at low frequencies $\delta \ll 1$ and the $\cos(\theta)$ term in (3.8) dominates. At higher clock speeds the changing bias current affects the switching time much more, and in (3.8) the δ term becomes important. As $\delta \to 1$ the non-linearity become more pronounced.

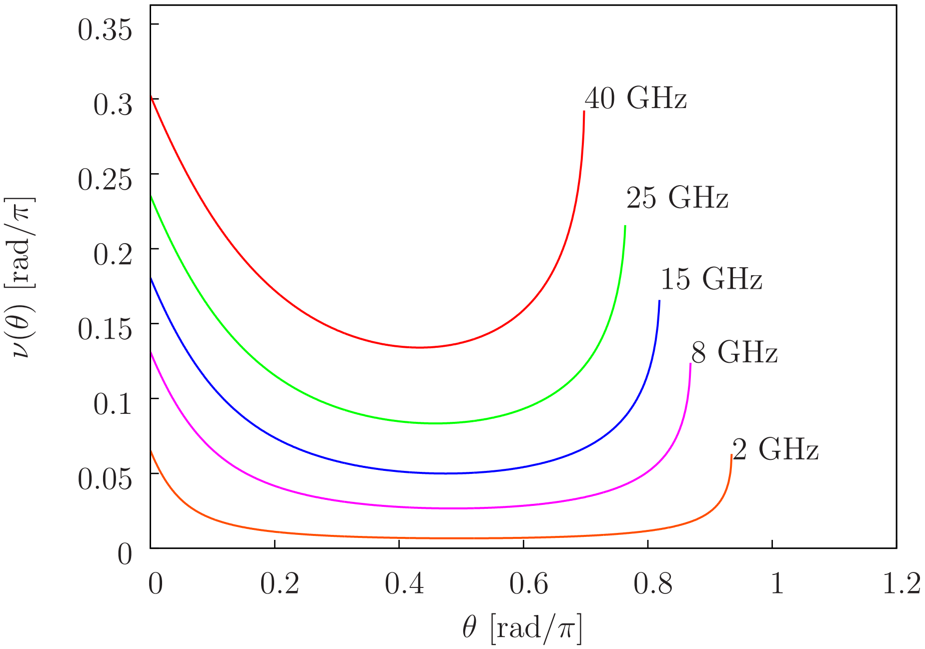


Figure 3.1: Junction phase delay versus starting junction phase θ. $\nu(\theta)$ from Eq. (3.8) plotted for clock frequencies of 2, 8, 15, 25, and 40 GHz. Here $I_c R_N = 0.75\,\text{mV}$ and $A = 0.83$. $\nu(\theta)$ is essentially the switching time $\tilde{\tau}$ normalized to clock frequency, i.e. $\tilde{\tau} = \nu(\theta)/2\pi f$. As clock speeds increase, the delay becomes longer relative to the clock period and the timing window becomes smaller.

With the delay ν known, the switching time for a single junction can be found to be $\tilde{\tau}(t) = \nu(\omega\, t)/\omega$, for a pulse arriving at time t. The switching time for a series of junctions on the same phase is $\sum_i \tilde{\tau}(t_i)$, where $t_{i+1} = t_i + \tilde{\tau}(t_i)$.

3.2.2 RQL Timing Stability

The four phase clock/power used in RQL plays a critical role in pulse propagation. Signal pulses are not free-running in the circuit unlike DC-biased JTLs found in RSFQ [4], but are instead controlled by the phases of the clock. Here I show that the multi-phase clock provides a *self-correcting* timing mechanism for RQL gates. What this means is that small variations in gate delay are corrected by the gate on the next phase. According to Fig. 3.1, pulses that arrive early cause junctions to switch slower and the delay between junctions will be longer. Late pulses see a higher clock bias current and will switch faster. Thus early pulses will arrive later at the next phase, while late pulses will be accelerated. After traveling through several clock phases, pulses achieve an equilibrium speed with zero accumulated jitter.

Figure 3.2 illustrates the behavior of a pulse propagating through a JTL with two junctions on each of four phases. The pulse arrives at JJ1 during the first phase at 6.8 ps, which is late in the clock cycle and about 5 ps after the junction could have switched. Because of this, the switching of JJ2, which is also on the A phase occurs very late in the clock window. As a result, JJ2 has a long switching time, ending at 11.5 ps. JJ3 is the first junction on the second phase. It receives the SFQ pulse early in the B clock phase. The delay of the pulse is shorter and leaves earlier. The delay of the SFQ pulse on the C phase is approximately the same as on the second phase. The SFQ pulse leaves the C phase before it can begin propagating on the D phase. When the D phase reaches sufficient bias current at approximately 21 ps, the first junction (JJ7) on the D phase starts to switch. Because the switching of both junctions JJ7 and JJ8 on the D phase completes within a quarter clock cycle of the beginning of the clock window, all junctions on later phases will start to switch at the earliest possible time. Thus, a pulse which arrived late in phase A is at equilibrium by phase D.

From this example, we can see how the timing stability is enforced by the clock

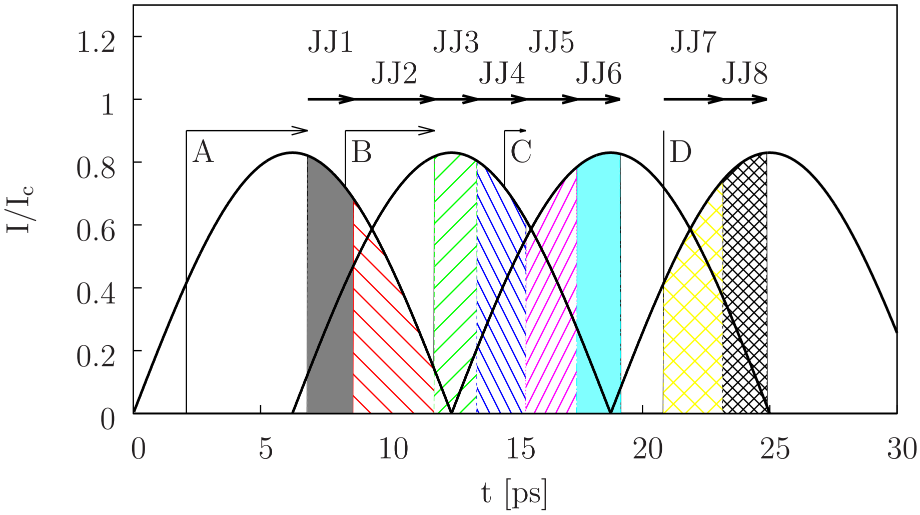


Figure 3.2: Self-correcting timing mechanism of RQL simulated for a clock frequency of 40 GHz and $I_c R_N = 0.75\,\text{mV}$. Four sinusoidal clock phases are shown as a function of time. Eight junctions, two per phase, are represented as different hatched regions beneath their respective curves. The area associated with each junction is equal and defines the beginning and end of the switching process. Each junction must switch sequentially. A–D show the earliest possible switching time for each junction, with the arrows indicating how long after this time switching actually starts. Arrows labeled JJ1–JJ8 show the length of switching time. The delay is about 5 ps for junction A while later delays decrease as the pulse travels through the JTL. Note that switching events near the peak of the bias current occur faster.

phase boundaries. Equation (3.8) gives the delay of an SFQ pulse as a function of the input phase. At a clock boundary, the delay can be expressed in terms of the input phase at one clock phase and the output phase to the next clock phase. Figure 3.3 shows a plot of t_{out} (the actual output time relative to the beginning of the clock cycle, *not* the delay time $\tilde{\tau}$) versus the input phase. Point **a** in the figure is the stable timing point. Pulses arriving earlier are delayed relative to the leading edge of the sinusoid whereas later pulses are accelerated. Notice however, if pulses arrive late enough in the cycle, they will be slowed down. This leads to the stable timing window ending at point **b** in the figure. At this meta-stable timing point any small decrease in speed will cause greater delays and a slight increase in speed will cause smaller delays. Pulses that are so late that they arrive after point **b** will slow down until they reach point **c**, the timing window cut-off point, after which pulses fail to propagate.

Although it is not obvious from Fig. 3.3 which is drawn for $\delta = 1.216$, for slow clock speeds ($\delta < 1$) the stable timing point coincides with the origin and the metastable point does not exist. In this case, all pulses arriving before the cut-off point **c** are sped up. An important advantage of the RQL timing scheme is that timing errors can be corrected by decreasing the clock frequency. This is not always true for RSFQ where fixed hardware delays can cause failure at low or high frequencies [36].

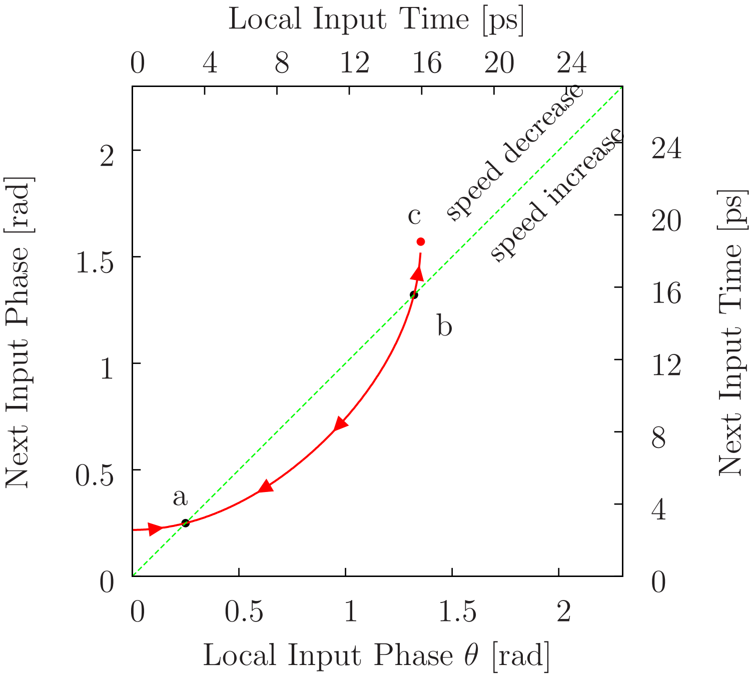


Figure 3.3: Relationship between input time on consecutive phases. In this example for $f = 13.5\,\mathrm{GHz}$, $A = 0.77$, and $N = 8$ the input time at the next phase is shown as a function of input time at the previous phase by the red (solid) line. A green (dotted) line bisects the graph to show regions of speed increase and speed decrease. For points below the green line, pulses arrive earlier at the next phase than they did at the previous phase. Arrows on the red line indicate direction of change. This tends to move input time to point **a**, where both input times are equal. Input before point **b** tends to move the input time to **a**, whereas input times after **b** tend to move the input time *away* from **a**, instead towards **c**, after which pulses cannot propagate.

3.2.3 Frequency Limit

The analytic timing model discussed in the previous two sections allows me to predict the frequency limits of operation for RQL circuits. As can be seen from Fig. 3.4, as the frequency increases the stable timing window gets smaller. Eventually the stable and meta-stable timing points converge, corresponding to the maximum operational frequency. At this limiting frequency, the minimum delay is equal to one quarter the clock period.

Figure 3.4 shows the relationship between the timing window and the stable and meta-stable timing points. In the figure, the timing window is marked by the empty box on the left and solid box on the right. For $\delta \leq 1$ the stable timing window extends from $\theta = 0$ to $\theta = \theta_c$ and no stable or meta-stable timing points exist. All pulses within the timing window will move towards $t = 0$ while pulses outside the timing window will fail to propagate. For $\delta > 1$ the stable timing window extends from θ_s, the input phase corresponding to the stable timing point, to θ_{ms}, the input phase corresponding to the metastable timing point — that is, the clock phase θ of points a and b in Fig. 3.3, respectively. In Fig. 3.4, all stable and metastable timing points correspond to an output phase delay of $\nu = \pi/2$ because this represents a delay of one quarter clock cycle. Pulses arriving earlier than θ_s will be slowed down and move towards $t = \theta_s/\omega$. Pulses arriving later than θ_{ms} will be slowed down as well until they are delayed to the point they can no longer propagate.

Equation (3.8) was derived from the behavior of a single junction. We can extend the equation to cover multiple junctions in a clock phase by changing $\delta \to N\delta$,

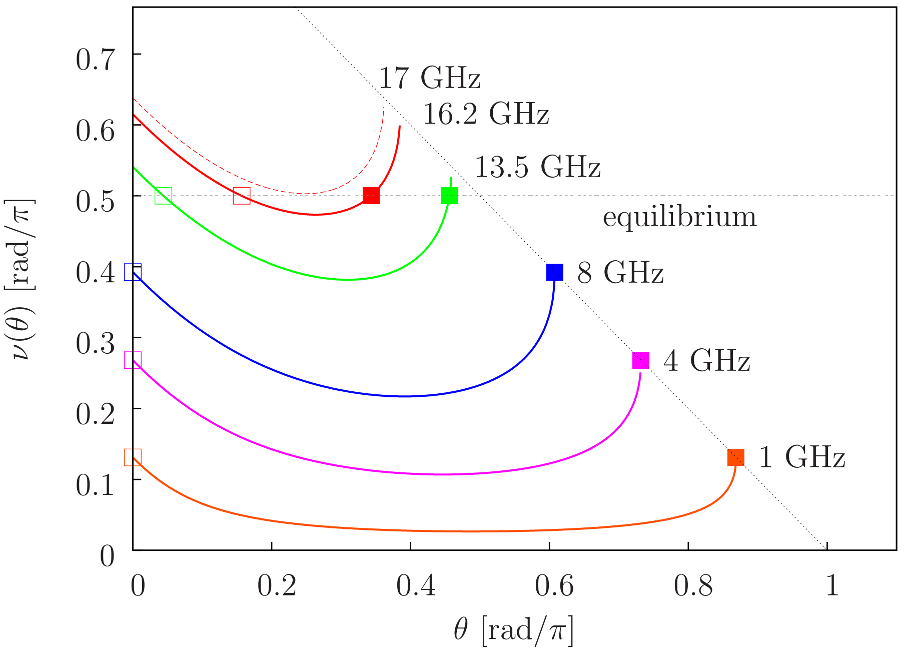


Figure 3.4: Switching delay ν versus input phase for different clock frequencies. Similar to Figure 3.1, this figure shows (3.8) plotted for $I_cR_N = 0.75\,\mathrm{mV}$, $A = 0.83$, and $N = 8$. Open boxes show stable timing points, filled boxed show metastable timing points. Metastable timing points lie along the line $\nu = \pi - \theta$ until they reach a maximum of $\nu = \pi/2$. Both stable and metastable timing points then lie at the intersection of $\nu(\theta)$ and $\nu = \pi/2$. The limiting case for 17 GHz is shown as the top curve.

where N is the number of junctions in a given phase. The fastest propagation is at the minimum of the timing equation. We find the input phase θ_{min} that gives the shortest delay by setting

$$\left.\frac{\mathrm{d}\nu(\theta)}{\mathrm{d}\theta}\right|_{\theta=\theta_{min}} = \frac{\sin\theta_{min}}{\sqrt{1-(\cos\theta_{min}-\delta)^2}} - 1 = 0, \tag{3.9}$$

and thus the input phase at minimum delay is

$$\theta_{min} = \arccos\frac{\delta}{2}. \tag{3.10}$$

Substituting (3.10) into (3.8), I can write for the minimum delay condition:

$$\nu(\theta_{min}) = \arccos\left(\cos\theta_{min} - \delta\right) - \theta_{min} = \frac{\pi}{2}, \tag{3.11a}$$

$$\arccos\left(\frac{\delta}{2} - \delta\right) - \arccos\left(\frac{\delta}{2}\right) = \frac{\pi}{2}. \tag{3.11b}$$

Equation (3.11b) has a solution at $\delta = \sqrt{2}$. Using the definition of δ then gives the following relationship for the maximum expected possible frequency

$$f_{max} = \frac{A}{3\sqrt{2}\pi N t_0}. \tag{3.12}$$

f_{max} can be made large by choosing short SFQ pulses (i.e. small t_0, which can be obtained by choosing large $I_c R_N$) and a small number of junctions per phase N.

I can draw a number of important conclusions about timing in RQL circuits. First, at $f = f_{max}$ the metastable timing point is at $\theta_{min} = \pi/4$. Second, the value of δ is limited to $0 < \delta < 2$. For $\delta < 1$, pulses are clock-limited, propagating through each clock phase and waiting at the phase boundary. For $1 < \delta < \sqrt{2}$ pulses travel ballistically, traveling through several clock phases before reaching equilibrium. For $\delta > \sqrt{2}$ pulses propagate slower than the clock signal, and for $\delta > 2$ no pulses can propagate. Using the values for the Hypres process given in Table 3.1 and choosing $A = 0.83$ and $N = 8$, I estimate the maximum clock frequency for RQL circuits at about 17 GHz (see Fig. 3.4).

3.3 Timing Extraction from Simulation

WRSpice[2] uses the RSJ model of Josephson junctions. The capacitor behaves linearly. The resistor is non-linear and uses a piecewise linear model of the resistance,

[2] http://www.wrcad.com/manual/wrsmanual/wrsmanual.html

using one value of resistance R_s for currents through the junction $I < I_c$, and another value of resistance R_q for currents through the junction $I > I_c$. The junction itself is modeled following the two Josephson equations (1.34) and (1.35), additionally recording the phase of the junction.

The time-domain analysis of the circuit is done in small time steps,[3] identical to the solution method of the original open-source SPICE3 program [37] on which WRSpice is based. At each time step the circuit is represented as a sparse matrix $\mathbf{G}$ relating the voltages at every node to the currents between nodes, where $\mathbf{G}V = I$. Non-linear elements, such as Josephson junctions, are represented in differential or integral form[4]. SPICE solves the ordinary differential equation using the Newton-Raphson iterative method. When convergence is achieved the simulator moves on to the next time step.

WRSpice takes ASCII-format netlist files. To perform my simulations, I first used Cadence Virtuoso schematic editor to layout the circuits shown in Fig. 3.7 using dummy value for the relevant circuit parameters (clock speed, input time). I generated a separate netlist for each gate (two for the AndOr gate, one for each of the two outputs). Then, using a perl computer language script to replace dummy values with real values, I had the simulations done and results analyzed automatically. (See Appendix B for details on the simulations.)

The analytic timing expression (3.8) can be applied to real gates. However, I had to make some assumptions that were not necessarily well justified for RQL

[3] http://bwrc.eecs.berkeley.edu/classes/icbook/spice/

[4] http://www.nutwooduk.co.uk/pdf/Issue82.PDF#page=27

circuits. More accurate results can be obtained by full physical simulation of the junction behavior. The analytic timing model can then be fit to the simulation to obtain a relationship between input phase and output phase. To do this, in this section I first define a delay time in terms of measurable (physical) quantities. I then define a consistent path though a gate to which this delay applies. Finally, I discuss the physical simulation of gates in an appropriate circuit to generate results which I then fit using the analytic model.

The phase ϕ across a junction is the natural coordinate describing the switching of a junction. In the mechanical pendulum analog of a junction, ϕ is the angular position of the pendulum. In the washboard potential, if the particle is moving from one local minimum to the next, then at any time the phase is clearly on one or the other side of the potential barrier. I will assume that $\phi = 0$ initially and that after an SFQ pulse $\phi = 2\pi$. I then define $\phi_c = (1 - e^{-1}) \times 2\pi$ as the transition point. For the reciprocal pulse the transitions point is $\phi'_c = 2\pi - \phi_c$. This choice of ϕ_c is somewhat arbitrary, so I leave ϕ_c as a variable and need to check that my particular choice does not impact critical results. (See Appendix B.)

Figure 3.5 shows the simulated behavior of the phase difference across of two junctions in a JTL that are connected in series and switch on the same clock phase. From this plot, I define the points t_{in} and t_{out} as the time when the JJ1 and JJ2 junction phases respectively cross the value ϕ_c. I then calculate the difference $\Delta t = t_{out} - t_{in}$ as the delay in the two-junction circuit. In normalized units of clock phase, I can define $\theta_{in} = \omega t_{in}$ and $\Delta\theta = \omega\,\Delta t$ for clock frequency ω and clock amplitude $A = I_b/I_c$. I preformed multiple simulations for different values of clock frequency

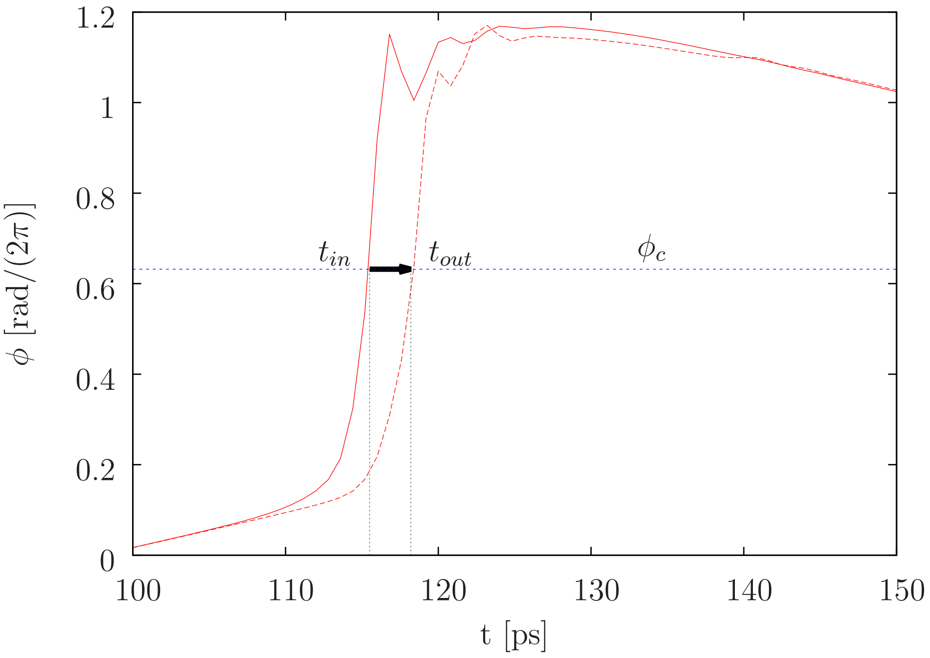


Figure 3.5: Spice simulation of the phases of two sequential junctions in a JTL during switching. The phase ϕ of junction JJ1 (solid curve) and JJ2 (dashed curve) on the same clock phase is plotted as a function of time. The phase of JJ1 crosses ϕ_c at t_{in}, which marks t_{in} and the beginning of the switching time for JJ2. When the phase of JJ2 crosses ϕ_c at t_{out} the switching time ends. The arrow in the figure indicates the length of the switching time.

and clock amplitude to generate a set of timing data for a gate or logic operation.

The gate delay is defined as the time between arrival of the input pulse at the input junction of the gate and the arrival of the output pulse at the input junction to the next gate. Figure 3.6 shows this concept for the AndOr gate. A pulse arrives at junction **b**, causing it to switch. Later, junction **c** at the input of the next gate switches. For consistency I demand that the time of output at one gate is the same as the time of input at the following gate. That is, I define the data path to extend past the conceptual boundary of the logic gate and into the next circuit element

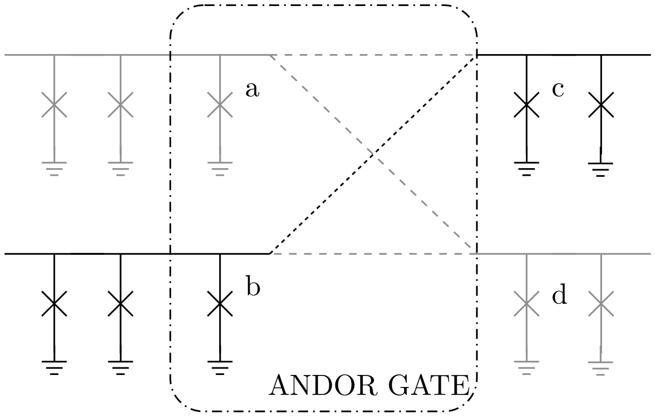


Figure 3.6: Data path through AndOr gate. The AndOr and four JTLs, one each at each input and output, is shown schematically with four junctions of interest labeled a – d. The junctions a and b, enclosed in the dashed box, which are physically part of the AndOr gate. This figure illustrates the data path through the gate for "or." The data path is shown in black; the inactive path is shown in grey. For "or" the data path starts with junction b and ends with junction c, even though c is outside the physical boundaries of the AndOr gate.

(see Fig. 3.6).

3.3.1 Fitting Simulation Results

With the delay and data path defined, I proceeded to simulate the gate behavior in WRSpice. Because the timing parameters should not depend on the test bench (the circuit schematic which is intended to be representative of any generic circuit), I must choose a test bench which generates representative simulation results for junction switching behavior. One of the test circuits I used is shown in Figure 3.7(a). In the circuit, JTL stages are inserted in different numbers for different clock speeds. The number of JTL stages was scaled to the frequency to cause failure (non-switching) of one of the junctions in one of the last units. As the SFQ

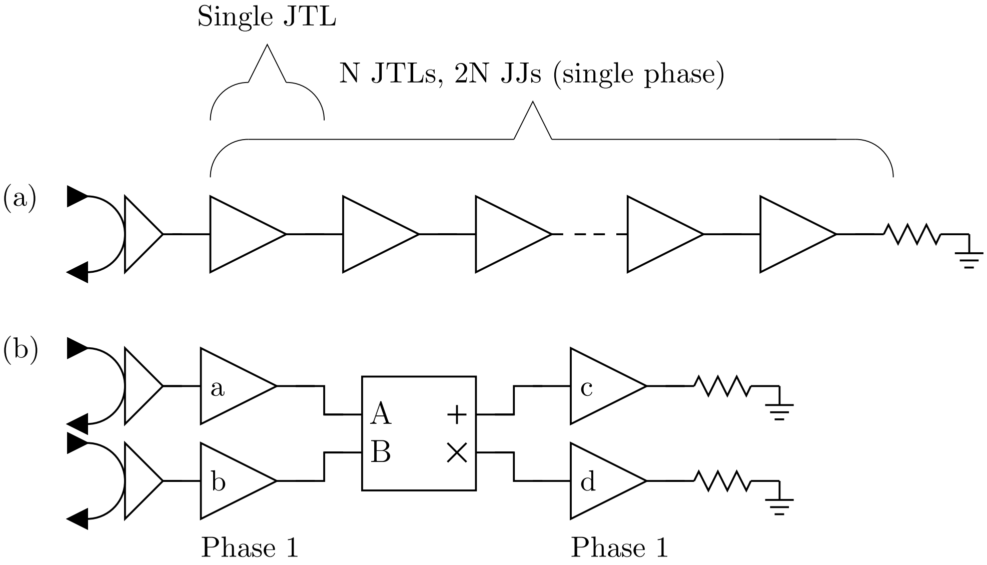


Figure 3.7: Circuits used to extract RQL timing results from spice simulations. Schematics of two circuits used for timing extraction simulations. Input is on the left and signals are terminated in the circuit by resistors to ground on right. (a) A series of JTLs on a single phase lead to a resistor to prevent reflection of the SFQ pulse. The total number N of JTLs in series was scaled to the frequency. High frequencies contained fewer JTLs, while lower frequencies contained as many as forty JTLs on a single phase. (b) Gate timing extraction with AndOr gate as a representative example. Two inputs lead to two JTLs labeled **a** and **b**, which then output to a logic gate. The two gate outputs lead to two further JTLs, **c** and **d**. Output from JTLs **c** and **d** terminate in resistors. Other gates use a different combination of JTLs **a**, **b**, **c**, and **d**.

pulses pass each JTL stage in sequence, it will generate one timing simulation point per stage.

Figure 3.7(b) shows the test circuit for the gates. I chose the AndOr gate as a representative gate because it has two inputs and two outputs. The basic simulation approach is the same for all gates although one must make some modifications for certain gates. For example, for the splitter, input JTL b is omitted. As another example, for the AnotB and Set-Reset gates, output JTL d is omitted. Each simulation provides one timing data point and with multiple simulations, I varied the input phase θ_{in} to give a spread of timing results (phase is a function of time) which I then analyzed. Details of the simulation routine I used are found in Appendix B.

Using a `perl` script I analyzed the simulation output and recorded the time pairs at which successive junctions in a data path crossed the critical phase value ϕ_c. Linear interpolation was used to estimate the time of crossing based on the immediately previous and following points. The delays ν_{fit} for a given input phase θ were then fit to:

$$\nu_{fit}(\theta) = \alpha_1\,\alpha_3\left[\arccos\left(\cos\left(\alpha_2\,\theta\right) - \delta/\alpha_3\right) - \alpha_2\,\theta\right], \tag{3.13}$$

where $\alpha_1, \alpha_2, \alpha_3$ are of fitting parameters and ν is the delay in radians. This is a modified form of our expected timing (3.8) and this choice of fitting parameters removes correlation between them. Ideally, these parameters would all be unity.

Table 3.2 shows a portion of the full timing table I constructed from simulations of the JTL. The full table can be found in Appendix B. The first thing to notice about the Table is that α_1 and α_2 are close to 1, as expected. These two parameters simply scale the input phase and phase delay. α_3 is close to 1 for higher frequencies, but is noticeably different from 1 for low frequencies. In fact, the general trend is that all α parameters get closer to 1 as the frequency is increased. α_3 is a scaling parameter for the curvature and we expect it to be less close to 1 than the other parameters due to the varying bias conditions in real circuits. Because the timing in much less sensitive to the delay at low frequencies, the divergence from 1 for the α parameters at lower frequencies is of lesser consequence. The additional information

Table 3.2: Timing Results for the JTL unit found by fitting simulation results to (3.13). First column is the frequency. α_i parameters are calculated using gnuplot. The end of the timing window θ_c, and first and last timing data points θ_{first} and θ_{last} are also given. Horizontal lines indicate truncated data. The full table is available in Appendix B.

f	α_1	α_2	α_3	θ_c	θ_{first}	θ_{last}
3.5	1.15	1.026	4.611	2.882	0.6598	2.469
4.5	1.084	1.04	15.49	2.916	0.6757	2.507
5.5	1.132	1.042	2.671	2.721	0.8015	2.438
8.5	1.065	0.9487	0.514	2.408	1.064	2.427
9	1.064	0.9584	0.5778	2.411	1.111	2.444
9.5	1.062	0.9614	0.61	2.404	1.159	2.436
10	1.11	0.9596	0.5932	2.334	1.247	2.363
14.5	1.013	0.984	0.9403	2.383	1.58	2.393
15	1.048	0.9892	0.9253	2.318	1.7	2.319
15.5	1.095	0.9883	0.868	2.238	1.818	2.247
17	1.002	0.9941	1.051	2.339	1.839	2.328

in the table, θ_c, θ_{first}, and θ_{last} tell us something about the timing window. As the frequency increases, θ_c has a very definite downward trend. In agreement with earlier predictions, as the frequency increases, the timing window becomes smaller. The range of simulated input phases is given by θ_{first} and θ_{last}. This range decreases as the clock frequency increases, indicating that at higher frequencies, SFQ pulses are more likely to arrive closer to the peak clock amplitude.

It fact, the simulations at low frequencies do not always fit well to (3.13). For example, in Table 3.2 the value of α_3 is often very different from 1. As another example, Fig. 3.9(b) shows the simulation results of the AndOr gate at 1 GHz, and one sees that the points do not neatly fall along a curve of the form of (3.13). To better capture the behavior of gates at speeds where (3.13) does not accurately match the simulation results, I also fit the simulation results between the first recorded input phase θ_{first} and the last recorded input phase θ_{last} to a piecewise polynomial equation

$$p_{fit}(\theta) = \underbrace{((\alpha_{11}\theta + \alpha_{12})\theta + \alpha_{13})}_{\text{polynomial}} \Theta(\theta - w) + \underbrace{((\alpha_{21}\theta + \alpha_{22})\theta + \alpha_{23})}_{\text{polynomial}} \Theta(w - \theta), \quad (3.14)$$

where $\Theta(x)$ is the step function defined by $\Theta(x) = 1$ for $x > 1$ and $\Theta(x) = 0$ for $x < 1$, $w = (\theta_{first} + \theta_{last})/2$ is the phase between the first and last recorded input phases, and the α_{ij} are fitting parameters — unrelated to the α_i parameters in (3.13).

Figure 3.8 shows the simulated delay points for the JTL at a frequency of 16 GHz and clock amplitude $A = 0.83$. The fit to (3.13) is shown by the green dashed curve. We can see that it is similar to the prediction from (3.8) using the nominal circuit parameters (red solid curve) This corresponds to $\alpha_1 = \alpha_2 = \alpha_3 = 1$. More importantly, we see that the points are close to the fit. In Fig 3.8, I also show the polynomial fit as a blue dashed curve. The polynomial fit parameter values are recorded in Table B.2 (see Appendix B, page 229). The polynomial fit is only applicable for $\theta_{first} < \theta < \theta_{last}$. Outside this range the fit to (3.13) gives better results.

If (3.8) is correct, then ideally all the α_i parameters should be close to unity. Clearly, this is not always the case. Also, in certain cases the fit to (3.13) gives large errors, especially at higher frequencies where fewer data points are available and at very low frequencies where the simulation behavior indicates a stepwise timing function. In such cases, the piecewise polynomial fit produces better results.

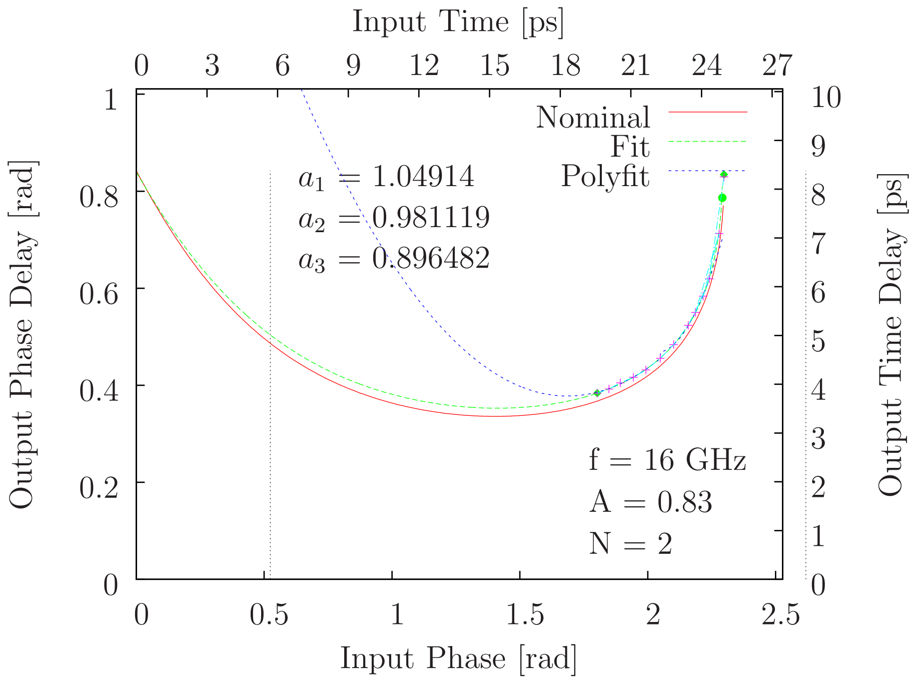


Figure 3.8: Fit of delay equation to simulated switching times for $f = 16\,\text{GHz}$, $A = 0.83$, $N = 2$. Purple data points come from simulation and analysis of the circuit shown in Figure 3.7(a). The nominal behavior (red curve) matches the data closely. The fit to the analytic equation (green dashed curve) is close to the nominal behavior. The fit parameters α_1, α_2, and α_3 are shown on the figure. The fit to the piecewise polynomial function fits well within the region where data is available but diverges strongly from the other fits outside this region.

Logic gates that are not directly biased by the ac clock signal will not necessarily have a timing response that is similar to clock-powered JTLs. In contrast to a JTL, one should expect these logic gates to exhibit behavior that diverges from the analytic timing equation (3.13). Figure 3.9 shows results for the AndOr gate outputting an OR-pulse at $f = 12\,\text{GHz}$ (Fig. 3.9(a)) and the AnotB gate generating output at $f = 1\,\text{GHz}$ (Fig. 3.9(b)). For the AndOr gate the analytic equation (green dashed curve) works surprisingly well even for an unpowered junction. The fitting parameters $\alpha_1, \alpha_2, \alpha_3$ show some divergence from nominal values of one, although for practical purposes the fit is excellent.

Figure 3.9(a) shows an example of poor fitting for the AndOr gate at 1 GHz. The behavior departs strongly from (3.13). For the AnotB gate (see Fig. 3.9(b)) the analytic fit (green dashed) shows significant errors. As can be seen, the timing response is more like a step function than the best fit to the analytic timing equation. Although it is worth noting that the median switching time is close to the value predicted by the nominal case, the data points lie fairly close to the red curve over the region of interest. A piecewise polynomial fit better represents the timing behavior in this case. At low frequencies, early pulses will get stuck at a logic gate, as there is very little current leaking in to the gate from either adjacent JTL unit (see Chapter 2). The pulse must wait until the local bias current is sufficiently high, at which time the switching event follows almost immediately. This behavior can be seen in the figure as a steady increase in the phase delay for shorter input delays. As a practical matter in circuit design, errors for low frequencies are of little importance. For $\delta \ll 1$ the latency is much less than the clock period and pulses will not fail due to timing issues. For frequencies of interest, where $\delta \approx 1$, the fits work well.

Using Table 3.2 I can generate plots of the fitted equation $\nu_{fit}(\theta)$ for different values of frequency. For example, Fig. 3.10 shows timing curves for different clock frequencies that required different α parameters. As in the purely analytic case the trend is toward longer phase delays for higher frequencies and the curves do not cross, as one would expect for real physical processes.

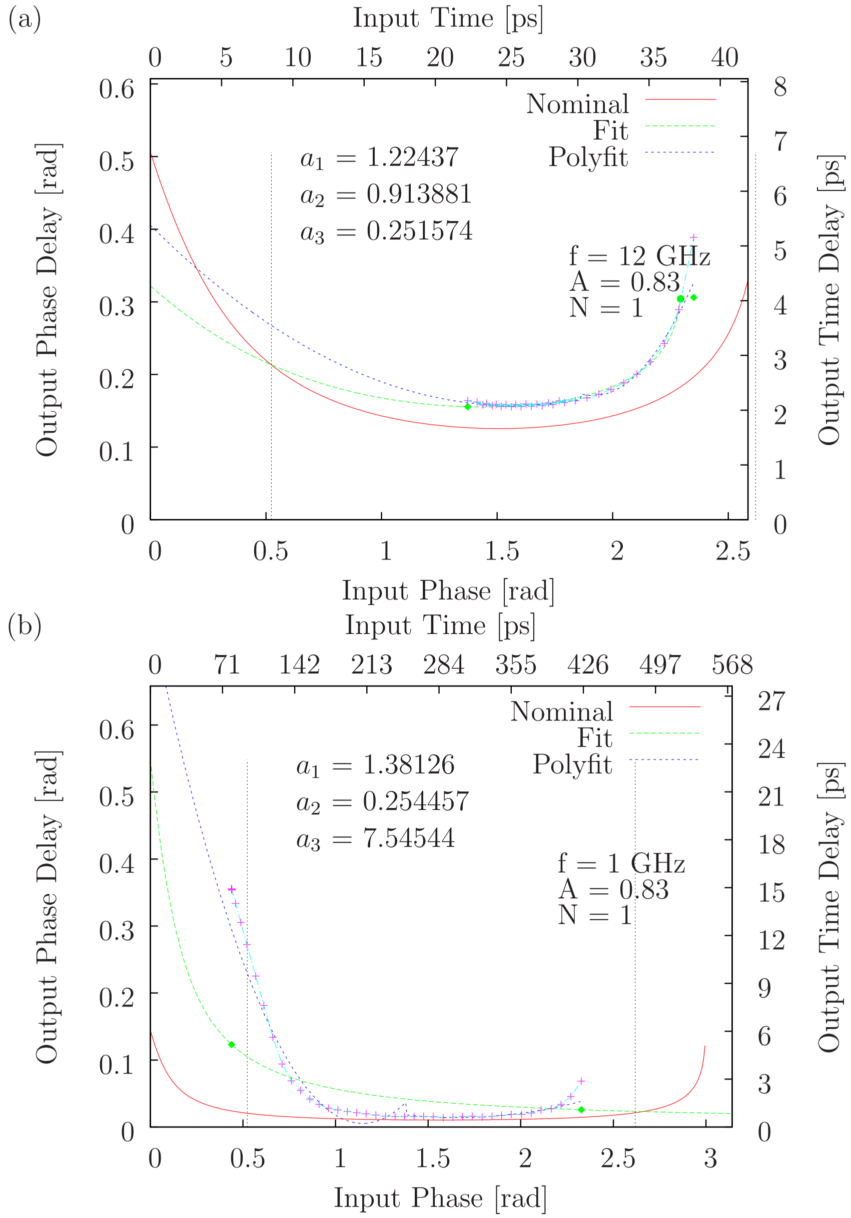


Figure 3.9: Simulated delay versus input phase. (a) Fits of (3.13) and (3.14) to simulated output delays versus input phase θ. AndOr gate at $f = 12\,\text{GHz}$, $A = 0.83$, $N = 1$. The points are noticeably different from the nominal case (red curve) given by (3.13). The value of α_3 is noticeably different from the nominal value of 1. Nevertheless (3.13) and (3.14) still match the simulation points well. (b) Simulated delay versus input phase for the AnotB gate at $f = 1\,\text{GHz}$, $A = 0.83$, $N = 1$. The best fit (green) of (3.13) does not match the simulation points. However, the fit to the polynomial (3.14) (blue) matches the points well.

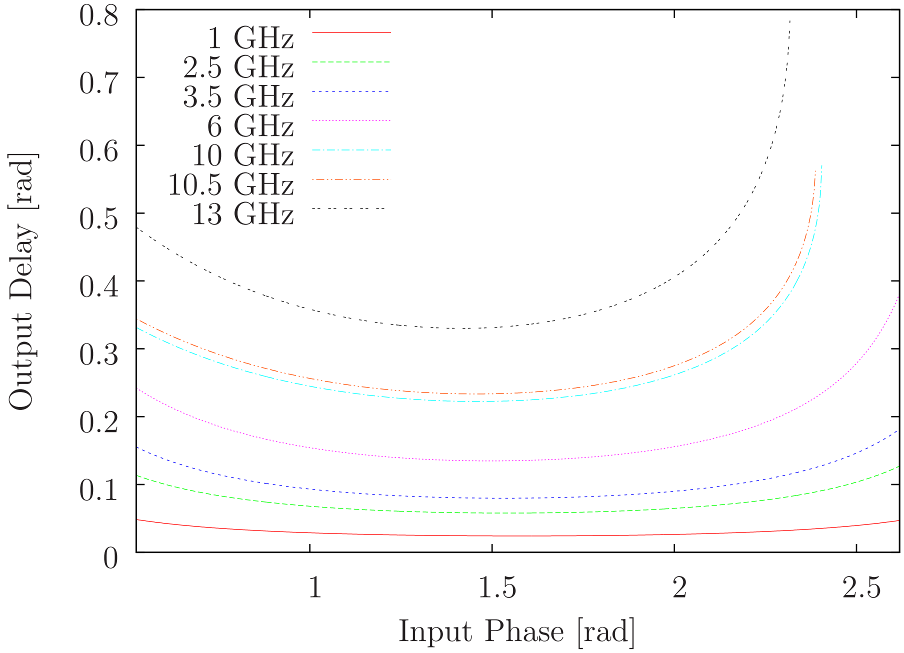


Figure 3.10: Comparison of Extracted Timing Curves. The fitted phase delay (3.13) is plotted using appropriate values for α_1, α_2, and α_3 for seven different frequencies from 1 GHz to 13.5 GHz. These curves have the same properties as the curves shown in Figure 3.1: the curves do not cross, low frequency curves are nearly flat, as the frequency increases, so does the output phase delay, and the endpoints of the curves move inward as the frequency increases.

It would be impractical to tabulate fitting parameters for all frequencies. Instead, I used linear interpolation in the frequency range between simulated frequencies. Figure 3.11 shows that linear interpolation between 10 GHz and 15 GHz fits well to the simulated 13 GHz data. The interpolated curves for both the analytic and piecewise polynomial fits match the data very well. The only drawback of interpolation is that it is limited by the end of the timing curve for the higher frequency.

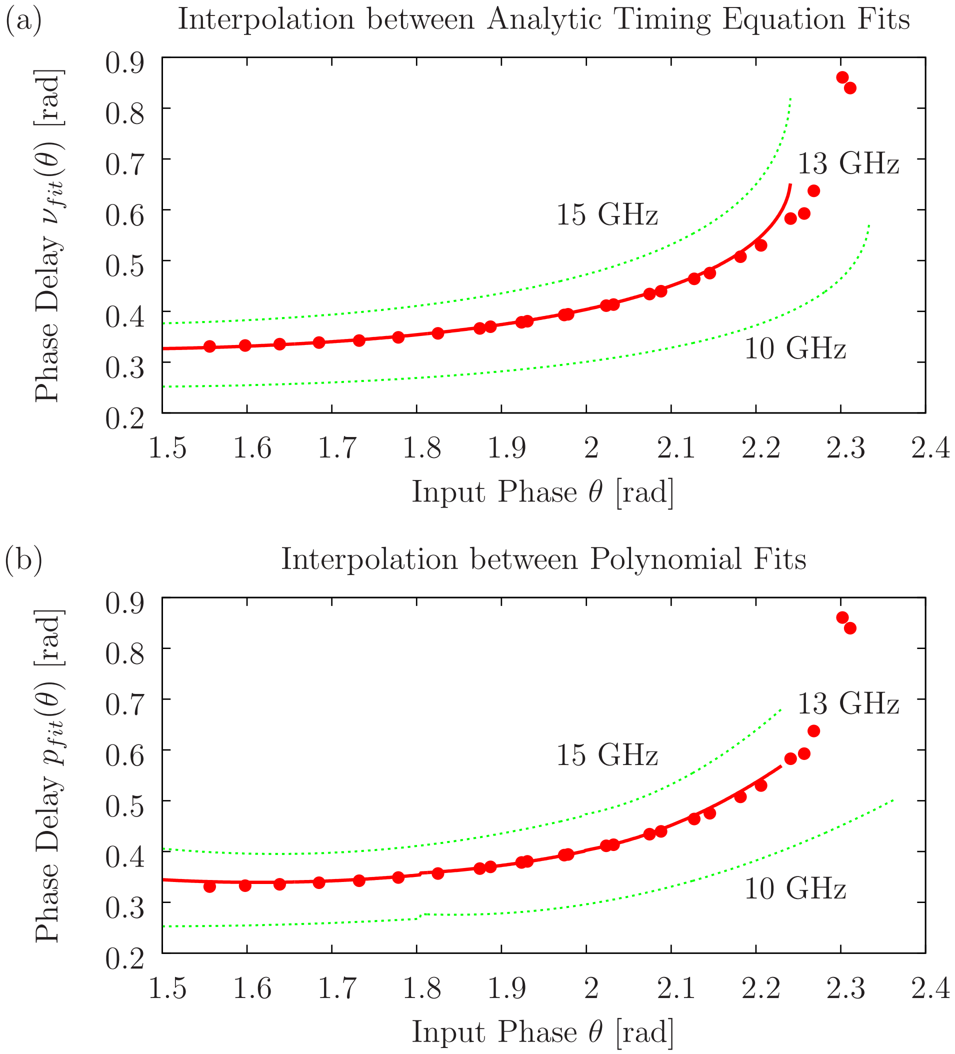


Figure 3.11: Simulated timing data for the JTL at 13 GHz (red solid dots) and plots of (3.13) and (3.14) for the JTL at 10 GHz and 15 GHz (green dashed curves). Both interpolations work well except at the limits of the region of interest, where the circuit is near failure already. The curve passing through the data points (red solid curve) is *not* a fit of (3.13) or (3.14) to the data but a linear interpolation of the functions to 13 GHz. Interpolation works well except at the end of the window higher frequencies.

3.4 VHDL Models for RQL Gates

In this section I explain an RQL gate model that I implemented in VHDL[5], a widely-used standard for timing design in the semiconductor industry. The models describe logic functions, timing behavior, and failure mechanisms. With these VHDL models, I can use semiconductor timing design techniques. VHDL uses multi-valued signals (called a *class* in VHDL) and determines the times at which transitions between the allowed values occur. I used the existing *std_ulogic* class[6], a common CMOS class which I found was appropriate for use with RQL circuits.

3.4.1 Behavior of VHDL Models

The VHDL models of RQL circuits that I built start from a model of the AC clock. The AC sinusoid is partitioned into three equal parts as shown in Figure 3.12. "High" and "Low" are above half maximum or below negative half maximum, respectively, and otherwise the clock is "Off." A positive pulse arriving during High will over-bias the junction and generate a new pulse; likewise for Low and a negative pulse. Insufficient bias during Off means pulses do not propagate and wait for the next High or Low region. This combination of "High," "Low," and "Off" sectioning gives a model for the clock signal identical to the std_ulogic model for CMOS in VHDL.

To simulate the behavior of RQL circuits and gates I developed a VHDL simulation package. The VHDL model contains the pulse-based logic of RQL, calculates

[5] IEEE 1076-2008: VHSIC Hardware Description Language

[6] IEEE 1164-1993: IEEE Standard Multivalue Logic System for VHDL Model Interoperability

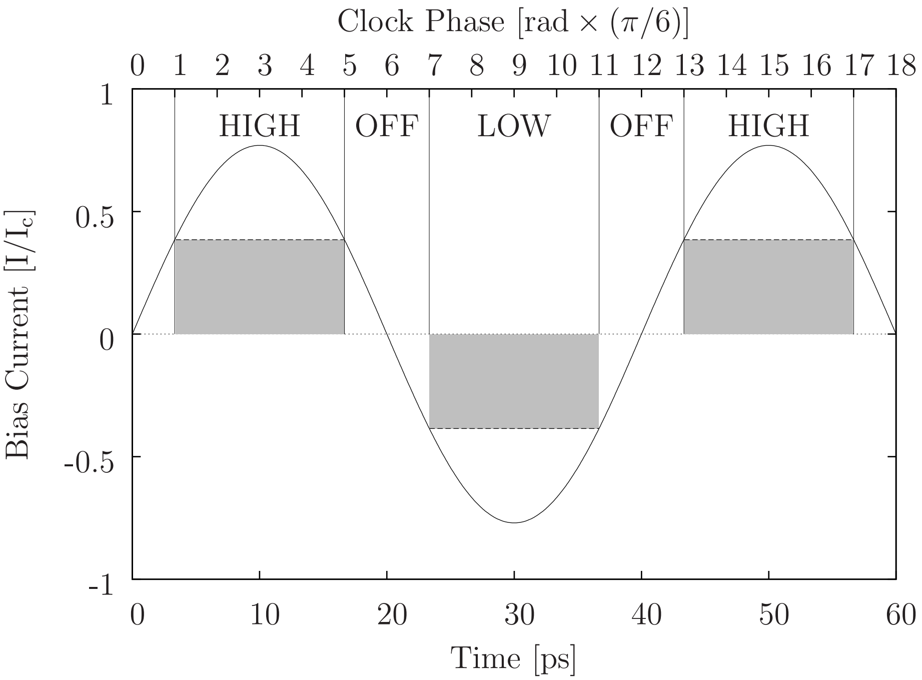


Figure 3.12: Timing model for RQL clock. The AC clock is partitioned into three logical segments. For $|\sin\theta| < 1/2$ the clock is "Off," during which pulse propagation is forbidden because of low bias. Likewise, "High" and "Low" are defined for $\sin\theta > 1/2$ and $\sin\theta < -1/2$, respectively, and only allow positive or negative pulses to propagate.

the delay from the analytic equation, and tracks energy dissipation, switching events, and approximate circuit size. The VHDL model is scalable for different fabrication processes. Table 3.3 shows the basic elements of the VHDL model.

Global parameters include I_cR_N, the junction energy scale set by the fabrication process. Clock frequency ω is the chosen clock frequency for operation, identical for all JTL units in the same circuit. A is the clock amplitude, a quantity important for the timing behavior. β_c is the Stewart-McCumber parameter which can be chosen for a particular design, and effectively changes the I_cR_N product. J_{total} is the total junction count of the chip, which serves as a metric for the area of the circuit.

Table 3.3: Global VHDL Quantities. The interaction between gates and the overall behavior of the circuit is governed by certain global inputs and signals. Global elements are inputs which allow the code to scale to any process, as well as record operation metrics such as power. Signals specify how the output of one gate becomes the input of the next, as well as carry clock phase information.

Global Elements	
$I_c R_N$	Fabrication energy scale
ω	Clock Frequency
A	Clock Amplitude
β_c	Damping scale factor
J_{total}	Total junction count (Area metric)
N_{switch}	Total number of junction switches (Power metric)

Signals	
1	SFQ Unit of Positive Flux (during switching)
0	SFQ Unit of Negative Flux (during switching)
H	SFQ Unit of Positive Flux (residual)
L	SFQ Unit of Negative Flux (residual)

Clock	
H	Clock above $+0.5 I_b/I_c$
W	Clock between $+0.5 I_b/I_c$ and $-0.5 I_b/I_c$
L	Clock below $-0.5 I_b/I_c$

N_{switch} is a running counter in the simulation which keeps track of the total number of junction switching events, which provides a metric for the energy dissipation. For the signals between gates, the values are 1 for a stored positive pulse and H for a transition to the 1 state, and 0 for a stored negative pulse and L for a transition to the 0 state. H and L allow easy visualization of the switching process. The clock signal can take on three values, H, W, and L, corresponding to the segments shown in Figure 3.12.

3.4.2 VHDL Combinational Gates

RQL circuits behave as state machines when the input and output are viewed as voltages. SFQ pulses travel from junction to junction and magnetic flux is stored inside the gates. However, from a higher-level view, RQL gates function as combinational logic gates if one considers the input and output of the gates to be *phase*. In the RQL data encoding scheme, the phase of a junction is normally approximately zero, but switches to approximately 2π after a positive SFQ has been generated (see Fig. 2.1). The reciprocal pulse changes the phase from 2π back to zero about half a clock cycle later. Because every positive pulse is followed by a negative one, the history of pulses in RQL is therefore equivalent to the history of phases on a junction.

The behavior of RQL logic gates is combinational. That is, the output of a gate depends on the input phases only, not the history of inputs as in state machines. Figure 3.13 shows two inputs and the four outputs of the three fundamental RQL logic gates described in this Chapter. The two inputs are shown in blue; the outputs are shown in green. Errors are shown in red. For the OR output, the phase is high whenever either the phase of A or B is high. For the AND output, the phase is high only when the phase of both A and B is high. Both these gates behave almost identically to the logic gates found in CMOS.

The phase of the AnotB gate is high when A is high and B is low, but not when A is high and B is also high. For these three outputs, the output phase is always low if both inputs are low. I wrote a special part of the VHDL code for the

AnotB gate, which checks if B transitions from B=0 to B=1 while A=1, and will generate an error in this case. The truth table for the AndOr and AnotB gate is shown in Table 3.4. This error condition is a result of the underlying pulse-based behavior of the junctions. Though it is not part of the combinational logic model, the VHDL models still check for this condition.

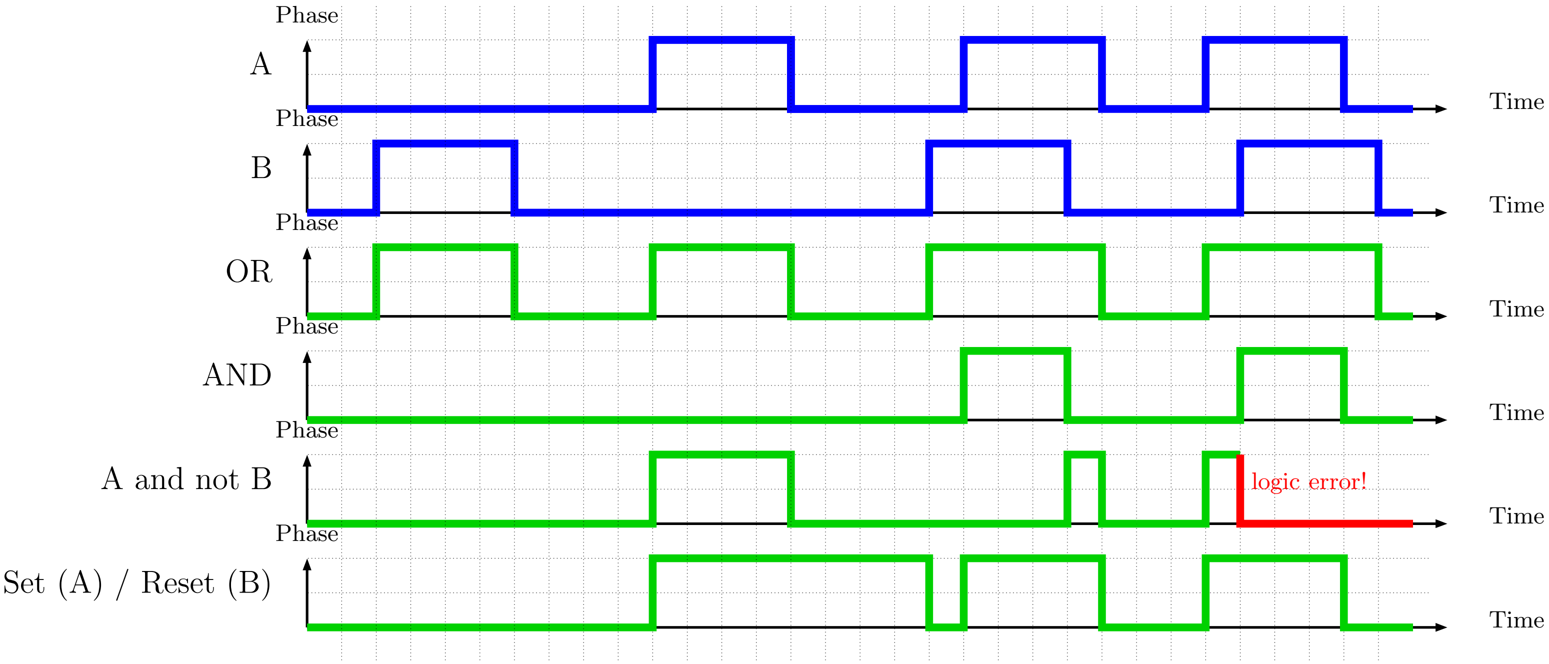


Figure 3.13: Combinational logic of RQL gates. Phases at inputs A and B are shown as functions of time (blue curves). The output phases of the AndOr, AnotB, and Set/Reset gate (where A is used as the Set input and B as the Reset input) are shown by the green curves. This behavior is similar to the behavior of CMOS gates using voltages as inputs. The exception is that because the AnotB gate has a timing requirement on the order of A and B inputs, a transition of B from 0 to 1 is flagged as a logic error in the VHDL code. This is shown by the red section of the AnotB output line. The AndOr and SetReset gates have no inherent timing restrictions and never generate errors.

Table 3.4: Truth table for AndOr and AnotB in VHDL. A and B are input phases to the AndOr and AnotB gates. And, Or, and AnotB in the table refer to the output phases of these gates. 0 is low phase, 1 is high phase. This is analogous to CMOS voltages. Note that this table does not capture one element of the behavior of the RQL AnotB gate; if a transition occurs from A = 1, B = 0 to A = 1, B = 1 for the AnotB gate, the model reports an error.

Input		Output		
A	B	And	Or	AnotB
0	0	0	0	0
1	0	0	1	1
0	1	0	1	0
1	1	1	1	0

The Set/Reset gate has fundamentally different behavior than the AndOr or AnotB gate. It behaves as a memory element. Much like in CMOS, a memory element can be constructed from combinational gates [38] with a bi-stable output. In terms of combinational logic, the Set/Reset gate is described as having three inputs: Set, Reset, and Output. By feeding the output of the gate back into itself, two bi-stable states are are found. Table 3.5 shows the truth table for the Set/Reset gate. The output is stable for 0 output so long as the Set input phase is 0 or the Reset input phase is 1. If the output is 0 and the Set input phase is 1, the output will switch. As can be seen in the table, the state Q=1, S=1, R=0 is a stable state. The output will remain in this state until the the Reset input phase is 1 while the input Set phase is 0. This changes the output to 0. The state Q=0, S=0, R=1 is a stable state again. Because the Set/Reset gate has no timing requirements, it never generates an error in the model.

The RQL logic gates generate output following the timing behavior described

```
case ina is
  when '0' =>
    case inb is
      when '0' =>
        qor <= '0' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
        qand <= '0' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
      when '1' =>
        qor <= '1' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
        qand <= '0' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
      when others => null;
    end case;
  when '1' =>
    case inb is
      when '0' =>
        qor <= '1' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
        qand <= '0' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
      when '1' =>
        qor <= '1' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
        qand <= '1' after Tperiod*nu(phasein, rqlgates.rqlpack.del_lo,
                    ANDORparams.A1_lo, ANDORparams.A2_lo, ANDORparams.A3_lo)/(2.0*pi);
      when others => null;
    end case;
  when others => null;
end case;
```

Figure 3.14: AndOr gate VHDL code. A snippet of VHDL code taken from the AndOr description in VHDL. Note the four cases and the function `nu` calculating the output delay.

in Section 3.3. Figure 3.14 shows the main part of the code for the AndOr VHDL gate. Note the keyword `after` indicating a delay in output.

3.4.3 JTL in Combinational Logic

The Josephson transmission line in RQL can be treated as a combinational logic gate, with two caveats. One, much like the Set/Reset gate, the output must also be considered an input. Unlike similar CMOS elements, this connection between output and input is completely conceptual. Two, unlike logic gates which have two SFQ inputs and where the output is a function of two possible phases on each gate, the JTL has one input which is a phase and one which is a clock signal. The clock signal has three possible values (L, W, and H, see Fig. 3.15), not two. Furthermore,

Table 3.5: Truth table for Set/Reset in VHDL. Q is the current output. Q′ is the resulting output. Set and Reset are the input phases at the respective inputs. 0 indicates a low phase, 1 indicates a high phase. The states Q=0,S=1,R=0 and Q=1,S=0,R=1 are unstable and will produce transitions in output.

Current Output	Input		Resulting Output	notes
Q	Set	Reset	Q′	
0	0	0	0	stable state
		1	0	stable state
	1	0	1	causes a transition
		1	0	stable state
1	0	0	1	stable state
		1	0	causes a transition
	1	0	1	stable state
		1	1	stable state

while the phase of a junction is well-defined, especially in RQL circuits, the clock signal is a sinusoid without the discreet levels given to the phase by the flux quantum. The goal here is to efficiently model the behavior of real RQL gates, I will proceed to describe the JTL as a combinational logic element.

Figure 3.15 shows an example of the combinational behavior of the JTL. The input is shown on top and takes two values as a function of time, 0 and 1, where 0 indicates low phase and 1 indicates high phase. The clock can take on three values, L, W, and H. These correspond to the values described in Table 3.3. They are shown here in cyan (H), gold (W), and magenta (L) for clarity. The output, like the input, takes on the values 0 and 1. The input is blue and the output is green under normal operating conditions, while the output turns to red to indicate an

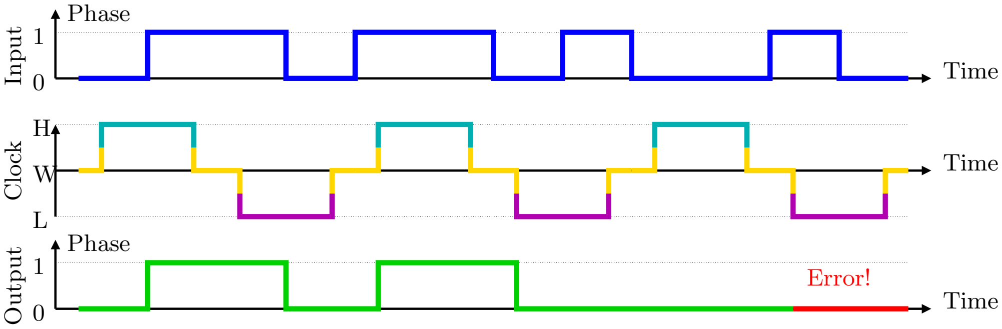


Figure 3.15: Combinational behavior of the JTL in VHDL. Input is shown in blue on top as a function of time, taking on values 0 (low phase) and 1 (high phase). The clock signal takes on three values as a function of time, L, W, and H, corresponding to the definitions given in Table 3.3. For visual aid, the clock signal is color-coded in cyan (H), gold (W), and magenta (L). The output is shown in green on the bottom, and takes the values 0 and 1 like the input. An error is shown on the far right when the clock changes from W to L while the input is high and the output is low.

error has occurred. Although RQL gates can be described by combinational logic, the underlying SFQ pulse-based logic imposes some restrictions on the behavior of the gates, much like in the case of the AnotB gate.

In the example shown in Fig. 3.15, the clock first changes from W to H, then the input changes to 1. This causes the output to change to 1 as well. The clock changes to W and then L. Only when the input changes from 1 to 0 does the output change. So far, the output has simply mirrored the input. Next, the input changes from 0 to 1 while the clock is W. The output changes to 1 only once the clock reaches W. Similarly, the output only changes from 1 to 0 once both input is 0 and the clock is L. When a change in input from 0 to 1 occurs while the clock is L, there is no change in output. Similarly, when the input changes from 1 to 0 while the clock is

Table 3.6: Truth table for a JTL in VHDL. Q is the present output. Q′ is the resulting output. Input (A) is the input phase at the input and Clock (C) is the clock signal. 0 indicates a low phase, 1 indicates a high phase. L indicates low clock, W indicates off clock, and H indicates High clock. The states $Q = 0, A = 1, C = H$ and $Q = 1, A = 0, C = L$ are unstable and will produce transitions in output.

Present Output	Input		Resulting Output	notes
Q	Input	Clock	Q′	
0	0	L	0	stable state
		W	0	stable state
		H	0	stable state
	1	L	0	stable state
		W	0	stable state
		H	1	causes transition
1	0	L	0	causes transition
		W	1	stable state
		H	1	stable state
	1	L	1	stable state
		W	1	stable state
		H	1	stable state

W, there is no change in output. The only error occurs when input is 1, output is 0, and the clock changes from W to L. This is outside the description of the behavioral logic. The VHDL model separately checks for this condition to occur (or similarly, a change in the clock from W to H while input is 0 and output is 1.

The example inputs shown in Fig. 3.15 are not exhaustive. Table 3.6 gives the full truth table for the JTL "gate". Just like the Set/Reset gate, the initial output is considered as an input as well. The JTL output is stable for all but two cases:

when $Q = 0, A = 1, C = H$; or $Q = 1, A = 0, C = L$. While the clock is low, only the condition $Q = 0, A = 1, C = H$ will change the output to $Q = 1$. Once this has occurred, only the condition $Q = 1, A = 0, C = L$ will change the output back. Not captured in the table is the error detection. I wrote a separate code in the VHDL model to check for timing violation conditions, which occur when $Q = 0, A = 1, C = W \rightarrow Q = 0, A = 1, C = L$ or $Q = 1, A = 0, C = W \rightarrow Q = 1, A = 0, C = H$. Finally I note that the figures here show the transitions occurring instantly. In the full VHDL model, these transitions occur only after a delay given by (3.13) and using the α_i parameters discussed in Section 3.3.

3.4.4 Summary of RQL Gates in VHDL

RQL gates with input phases behave very similar to CMOS gates with voltage inputs, as can be seen from Fig. 3.13. This allows them to be used in a similar fashion and their behavior can be analyzed using existing software design tools intended for CMOS. However, RQL gates are still subject to certain constraints and the timing behavior is much different from CMOS. Nevertheless, this behavior can also be captured in VHDL as shown in this Chapter. The AnotB gate in particular has certain timing requirements due to the detailed behavior of the gate. It is the only gate that generates logic errors. Finally, JTL units follow combinational logic rules as well, with the caveat that the clock signal carries three values which do not correspond to any particular physical quantity. The JTL is the only "gate" which generates timing errors.

Chapter 4

Power Network Design

4.1 Introduction

In this Chapter, I discuss the design of the network I developed for powering my RQL circuits. In RQL circuits the power splitter must accomplish several tasks. First, the splitter must step down the impedance from $50\,\Omega$ at the pads to $32\,\Omega$, the impedance of the clocklines coupled to transformers within the circuit. Second, it must not only split the power evenly, but but also recombine the power in the clock lines to be taken off chip. Third, although it must function over a broad frequency range, the amount of space it uses on the chip needs to remain small. Fourth, in order for an RQL circuit to work properly, the distribution of currents between splitters and combiners must remain within 10% of nominal values within the frequency range. Finally, the splitter must also maintain these properties when the electrical length of individual lines between the splitter and combiner are changed by loading or fabrication.

I first describe a general design for an eight-way Wilkinson power splitter and discuss three possible responses: geometric, equal ripple, and maximum flat. I then describe the design of two power splitters that I used for testing RQL circuits. I next compare results from testing both designs. Finally, I describe a second experiment using a maximum flat Wilkinson power splitter to power an RQL circuit containing

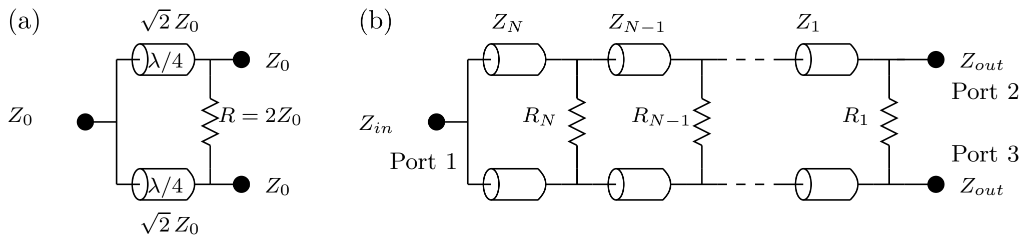


Figure 4.1: Wilkinson Power Splitter. (a) The traditional Wilkinson power splitter: one stage with equal impedances at all ports and quarter-wavelength transmission lines. For the design frequency, evenly splits power without losses between output ports (on right) while completely isolating the output ports from each other. (b) The general Wilkinson power splitter with N stages, 2N transmission lines, and N resistors. Higher bandwidth than one-stage Wilkinson power splitter. Note also that input and output impedances need not be equal and we allow any electrical length ρ so long as all transmission lines are equal in length.

shift registers.

Figure 4.1 shows the general layout of a Wilkinson power splitter using quarter-wave length transmission lines and resistors. Figure 4.1(a) shows the most basic Wilkinson power splitter design with equal impedances on input on left and output on right. The quarter-wave segment is central to the operation of the power splitter. For transmission lines of length l the impedance of the line is

$$Z_{in}(l) = Z_0 \frac{Z_L + \mathrm{i} Z_0 \tan \beta l}{Z_0 + \mathrm{i} Z_L \tan \beta l}, \tag{4.1}$$

where Z_0 is the characteristic impedance of the transmission line, Z_L is the load impedance, and $\beta = 2\pi/\lambda$ is the wavenumber. For quarter-wave segments, this gives $Z_{in} = Z_0^2/Z_L$, which for $Z_L = Z_0$, $Z_{in} = \sqrt{2}Z_0$. Using this value of the impedance for the quarter-wave segments, the impedance $\tilde{R}$ seen at the input can be calculated as the combination of one quarter wave segment in parallel with the resistor and other quarter-wave segment in series, as follows:

$$\frac{1}{\tilde{R}} = \frac{1}{Z_0}\left[\frac{1}{\sqrt{2}} + \frac{1}{2+\sqrt{2}}\right] = \frac{1}{Z_0}\frac{(2+\sqrt{(2)})+\sqrt{2}}{\sqrt{2}(2+\sqrt{2})} = \frac{1}{Z_0}\frac{2+2\sqrt{2}}{2\sqrt{2}+2} = \frac{1}{Z_0}. \qquad (4.2)$$

This gives the effective impedance of the power splitter as Z_0, a perfect impedance match.

Figure 4.1(b) shows a generic layout for a 2-way Wilkinson power splitter with impedance Z_{in} on the input and impedance Z_{out} on the output, with N stages. Note that the numbering of resistors and quarter-wave transmission lines starts at the output. A more thorough description of the Wilkinson power splitter can be found in many textbooks on electrical engineering, such as [39].

4.2 Circuit Design

I designed a Wilkinson power network in two steps. As I discuss below, I first minimized the input port reflections in “even” mode and I then maximized the output port isolation in the “odd” mode. Both steps are important for power networks in digital circuits since reflection at the input port of the power combiner causes standing waves in the power lines and the corresponding nonuniform distribution of the current produces potential spikes at the anti-nodes. Odd mode analysis applies to the situation where the clock lines have different electrical length due to the different topology of the lines or different load by the gates.

To proceed, I considered a generic Wilkinson power splitter design (see Fig. 4.2). The configuration of a Wilkinson power splitter is specified by giving the number of power splitter segments[1] in successive sections. I used a one, two, two, and one or 1221 configuration. In general, a splitter configuration with M segments is designated by $\mathrm{a}_0\mathrm{a}_1 \ldots \mathrm{a}_\mathrm{M}$. The total number of quarter-wavelength stages is $\sum_{m=0}^{M} 2^m a_m$ and the number of resistors is $\sum_{m=1}^{M} 2^{m-1} a_m$. These are important design metrics for system integration. These configuration only describe the *layout* of the power splitter, *not* the values of the impedances Z_m or resistances R_m. In the next section, I introduce three different methodologies for determining the impedances of the stages.

4.2.1 The Even Mode

Figure 4.3 shows the decomposition of a generic Wilkinson power spitter into the "even" and "odd" mode. The full response of the circuit to any input at any port can be found by superposition of the circuits shown in Fig. 4.3(b,c) [39]. In the even mode, two equal voltages +V are applied to the output ports. Because the potential across each resistor connecting the upper and lower segments (see Fig. 4.3(a)) is zero due to symmetry, they can be removed. The resulting circuit is shown in Fig. 4.3(b) and has the layout of a generic quarter-wave impedance matching filter. Because the two halves of the circuit behave identically, they are symmetric or *even*. This is the primary mode of operation of the power splitter I designed; the odd mode shown in Fig. 4.3(c) will be of interest later.

In the even mode the Wilkinson behaves like an impedance matching filter

[1] A note on terminology: In microwave engineering, the term "stage" is common for the quarter-wavelength transmission lines in a WPS. As such, we need a different word to describe groups of these stages into hierarchical units. As both "stage" and "segment" are similar in both sound and meaning, I wish to explicitly note the difference.

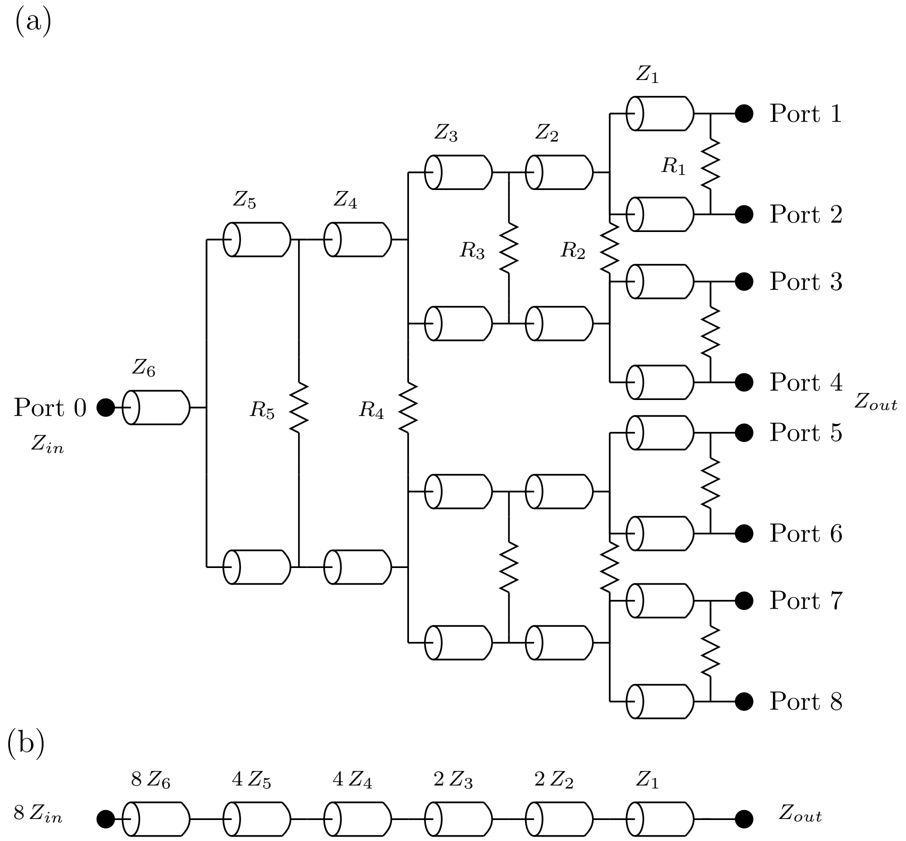


Figure 4.2: Schematic of the six-stage, eight-way Wilkinson power splitter (1221 configuration). Input on the left from a line with impedance Z_{in}. Output on right to a line with impedance Z_{out}. (a) Stage 1 is a single-stage Wilkinson. Stages 2–5 are each part of two two-stage Wilkinson power splitters. Stage 6 is an impedance matching stage. All elements on vertical line share same design values. (b) Even-mode analysis schematic of the schematic in (a). Starting from the output, each Wilkinson stage increases the input impedance by a factor of two. The Eight-way Wilkinson in even mode is equivalent to a six-stage impedance matching filter with impedances $8Z_6$, $4Z_5$, $4Z_4$, $2Z_3$, $2Z_2$, and $1Z_1$.

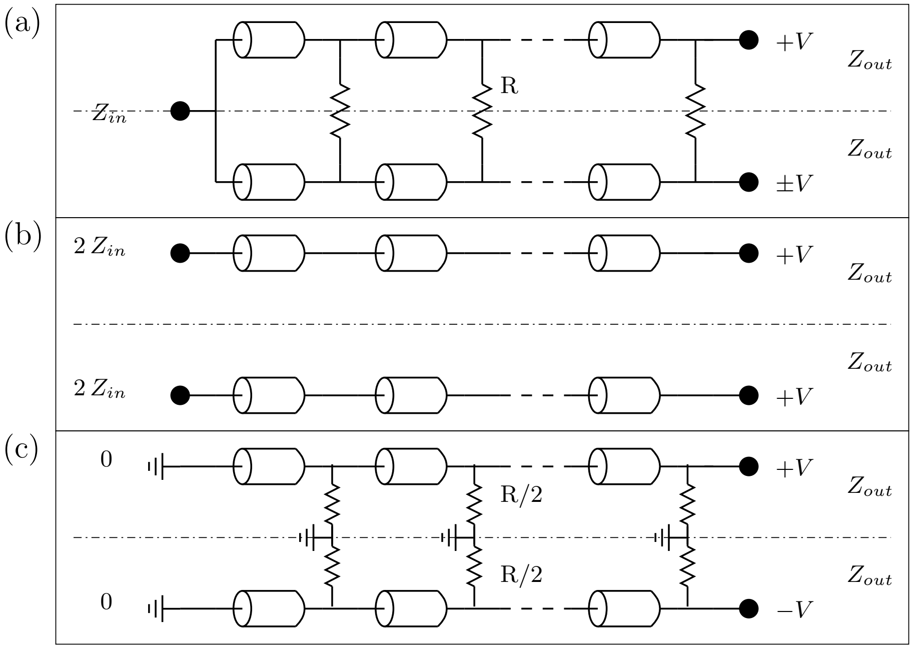


Figure 4.3: Even and Odd mode analysis of the Wilkinson Power Splitter. The Wilkinson power splitter shows a perfect bilateral symmetry, which aids the analysis. Notation mostly removed for clarity. (a) Symmetry line separating the Wilkinson power splitter into two electrically identical halves. We apply a voltage V at port 2 and ±V at port 3. (b) Even mode. +V applied to port 3. Because of symmetry, potential across each resistor is zero and no current flows through them. Input impedance is double, as the two inputs are in parallel. The halves now appear virtually identical to an impedance-matching filter. (c) Odd mode. -V applied to port 3. By symmetry a zero-potential must exist between the halves, effectively grounding the middle of each resistor and electrically separating the two halves. The representative resistor R has value R/2 now for each half in odd mode. No signals propagate through in odd mode.

between impedances Z_{in} and Z_{out}. The Wilkinson power splitter is normally analyzed with equal impedances on both input and output. However, the even- and odd-mode analysis can be generalized to the case of unequal impedances [39]. The response of the equivalent impedance matching filter is determined by the choice of impedances in each stage of the Wilkinson power splitter.

I consider three responses. First, the *geometric response* is of the form $\Gamma(\rho) \sim \cos(N\,\rho) + \cos((N-2)\rho) + \ldots$ and has a constant ratio of impedances, making it easy to design in physical layout. Second, the *maximum flat* response has the form $\Gamma(\rho) \sim (1 - e^{-i\,\rho})^N$ and fulfills the condition $d^n\Gamma/d\rho^n(\pi/2) = 0$ for $n = 0, 1, \ldots, N-1$. Third, the *equal ripple* response has the form $\Gamma(\rho) \sim T_n(\cos\rho)$ where T_n is the Chebyshev polynomial of the first kind. This gives the broadest bandwidth for a given maximum reflection coefficient within the bandwidth. The detailed derivation of filter impedances for the different cases can be found in Appendix C (page 263). I find [39] for

the geometric response:
$$Z_{n+1} = Z_n \sqrt[N+1]{2^M \frac{Z_{in}}{Z_{out}}}, \tag{4.3}$$

for maximum flat response:
$$Z_{n+1} = Z_n \times \exp(2^{-N} \mathrm{K}_n^N \ln(2^M \frac{Z_{in}}{Z_{out}})), \tag{4.4}$$

where binomial coefficient
$$\mathrm{K}_n^N = \frac{N!}{(N-n)!\,n!},$$

and the equal ripple response:
$$Z_{n+1} = Z_n \frac{1+\gamma_n}{1-\gamma_n}, \tag{4.5}$$

where $N = \sum a_m$ is the total depth of the Wilkinson power splitter and γ_j is proportional to the j$^{\text{th}}$ expansion coefficient of the Chebyshev polynomial (see Appendix B).

For $N = 6$ I find the impedance values shown in Table 4.1. Figure 4.4 shows the

Table 4.1: Impedance values for the Wilkinson splitter stages for different configurations with $N = 6$ stages. All values are in Ohms. Values given in order matching Fig. 4.2. Final design values correspond to Fig. 4.10 and are given here for comparison. Fractional bandwidth given for -13 dB. Values for final design are constrained by fabrication limitations, unlike for the three other responses.

	Z_6	Z_5	Z_4	Z_3	Z_2	Z_1	Bandwidth
Maximum Flat	30.88	50.98	31.31	32.70	20.09	33.06	0.9
Equal Ripple	28.19	42.88	28.70	36.47	24.41	37.14	1.2
Geometric	23.78	35.33	26.25	39.01	28.98	43.07	0
Final Design	47.94	37.24	20.72	19.36	21.44	33.36	0.73

even mode reflection parameters[2] for these three configurations as well as the final design which I used. The geometric response has the largest input mismatch and this gives the highest reflection at the design frequency and across the bandwidth. The equal ripple response has better matching and will have lower overall reflections and a higher bandwidth. The maximum flat response has a smaller bandwidth than the equal ripple response, but has the lowest overall reflection. At the target frequency the variation is lowest for the maximum flat response. I discuss my final design in Section 4.2.5.

The 1221 Wilkinson is not the only configuration for a power splitter. I compare two other configurations, the 4440 configuration and 2220 configuration power splitter. Figure 4.5 shows the 4440 Wilkinson splitter. (Resistors have been omitted for clarity.) This kind of design lends itself well to the geometric response, as each segment has identical impedances in each of the quarter-wave stages. Figure 4.6 shows a similar configuration with only two stages per segment.

[2] Unless noted otherwise, all S-parameters are given as $S_{ij} = 10 \log_{10}(V_i/V_j)$.

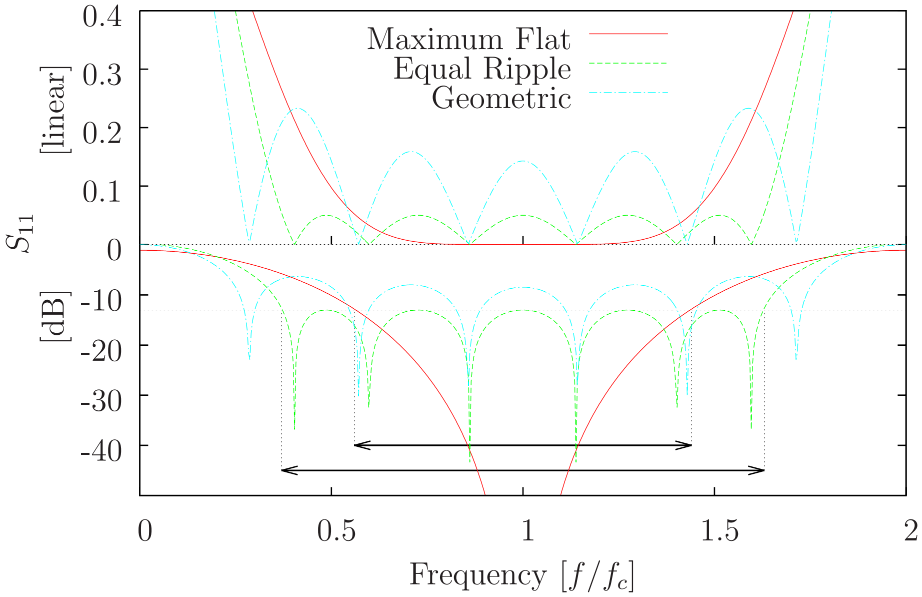


Figure 4.4: Wilkinson 1221 Simulated Reflection Parameters for maximum flat design (red solid), equal ripple design (green dotted), and geometric design (blue dashed) are plotted as a function of normalized frequency in linear and log space. Circuit shown in Fig. 4.2 and circuit parameters correspond to values given in Tables 4.1 and 4.2. Line drawn in at -13 dB (5%) is for reference. Geometric design has high reflections over entire design range. Equal ripple has less than -13 dB reflection over the broadest frequency range (by design), as shown by bottom arrow. Maximum flat response has smaller range (shown by top arrow) for which reflection is below -13 dB but has the smallest variation of reflections within the design range.

Figure 4.7 shows the even mode reflection parameters for the splitter design shown in Fig. 4.5. (Compare with Fig. 4.4.) Notice that the six-stage geometric series response has high reflection parameters. The six-stage device is similar to the 12 stage design, though each of the branches with four quarter-wave segments becomes a branch with two quarter-wave segments, turning it into a 2220 configuration splitter. Comparing the 12-stage geometric series response and the six-stage

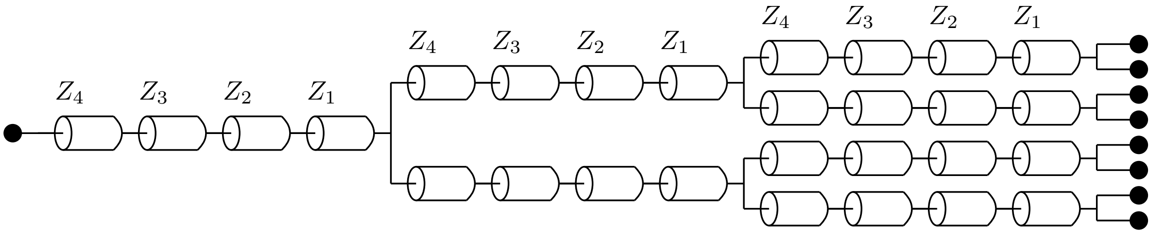


Figure 4.5: Circuit schematic for Wilkinson 4440 configuration for initial prototype power splitter using a geometric response. Resistors *not used* in this design; testing was only done in even mode. Designed for input and output impedance of 32 Ω. Ratio of impedances for each transmission line stage to the next was $2^{-1/4}$, such that $Z_1/\frac{1}{2}Z_4 = Z_4/Z_3$.

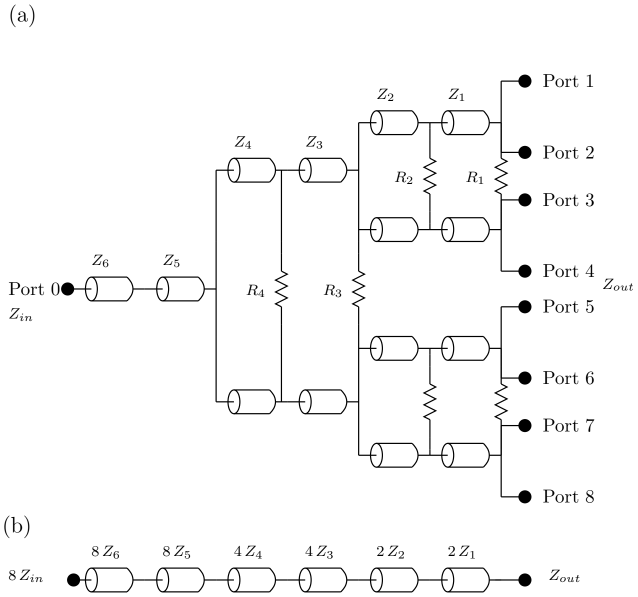


Figure 4.6: Circuit schematic for WPS2220. Similar to the 4440 design.

maximum flat response, the reflection parameters are similar even though the latter has half the number of stages.

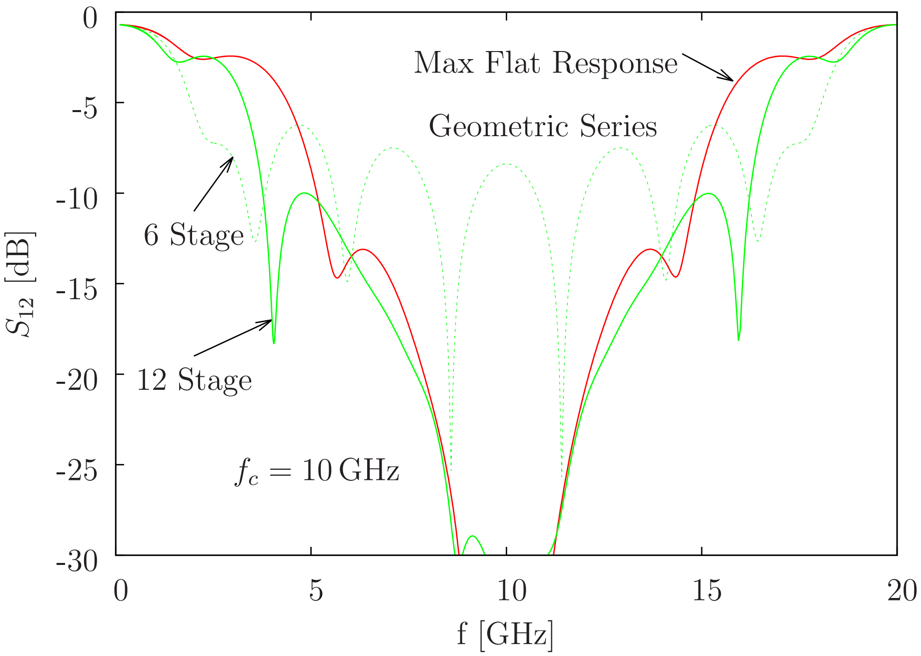


Figure 4.7: Geometric versus max flat power splitter reflections. Simulated reflection parameters of 1:8 Wilkinson power splitter in even mode using actual impedance values. The maximum flat response for a 6-stage deep splitter (red curve, 2220 configuration) is compared to the geometric response for 6 and 12 (4440 configuration) stages (dotted and solid green curves, respectively).

4.2.2 The Odd Mode

The second half of the analysis of a Wilkinson power splitter involves applying a voltage +V to one output port and -V to the other output port, as shown in Fig. 4.3(c). In this case, the circuit has a zero-potential between the top and bottom halves with, making the two halves appear opposite or *odd* to each other. Proper odd mode analysis [39] depends on specific impedance values. We can maintain a mirror symmetry in the odd mode by treating each segment as an individual power splitter with ±V at each input.

Table 4.2: Resistance values in Wilkinson power splitter for different responses. Values in Ohms. Resistors R_2 and R_4 tend to have very small values compared to the rest of the resistors. Final design values given for reference and correspond to Fig. 4.10, which omits resistors R_4 and R_2. Final design is for a 3111 configuration, all others are for a 1221 configuration.

	R_5	R_4	R_3	R_2	R_1
Maximum Flat	49.29	2.05	31.62	2.08	64
Equal Ripple	44.43	2.06	37.39	2.07	64
Geometric	39.77	2.06	43.90	2.05	64
Final Design	23.56	n/a	20.62	n/a	58.62

For $N = 1$ the problem is trivial; for $N = 2$ we follow the method of Cohn[3], which for $N = 2$ and a fractional bandwidth of 1 gives [40]

$$R_2 = \frac{2Z_1Z_2}{\sqrt{(Z_1 + Z_2)Z_2}}, \tag{4.6a}$$

$$R_1 = \frac{2R_2(Z_1 + Z_2)}{R_2(Z_1 + Z_2) - 2Z_2}, \tag{4.6b}$$

where R_2 and R_1 are the resistor values of the resistors in Fig. 4.1(b) and Z_2 and Z_1 are the impedances of the quarter-wave transmission lines shown in Fig. 4.1(b). I can apply (4.6a) and (4.6b) to the results calculated from (4.3), (4.4), and (4.5) for the circuit shown in Fig. 4.2. The results are shown in Table 4.2.

The odd mode analysis normally completes the design of a power splitter. RQL circuits place additional requirements on the design. So far I have only considered a generic 1221 power splitter. I will now also consider several alternative configurations, including a 4440, 2220, and 3111 configuration Wilkinson power splitter.

[3] Cohn's original result is $R_2 = 2Z_1Z_2/\sqrt{(Z_1 + Z_2)(Z_2 - Z_1 \cot^2 \rho_3)}$. In Cohn's method [40] ρ_3 is the fractional bandwidth in units of 2π. For better comparison I simplify the equations for a fractional bandwidth of 1, for which $\cot \rho_3 = 0$. Fine tuning of the circuit occurs later in the design process and this simplification does not impact final results.

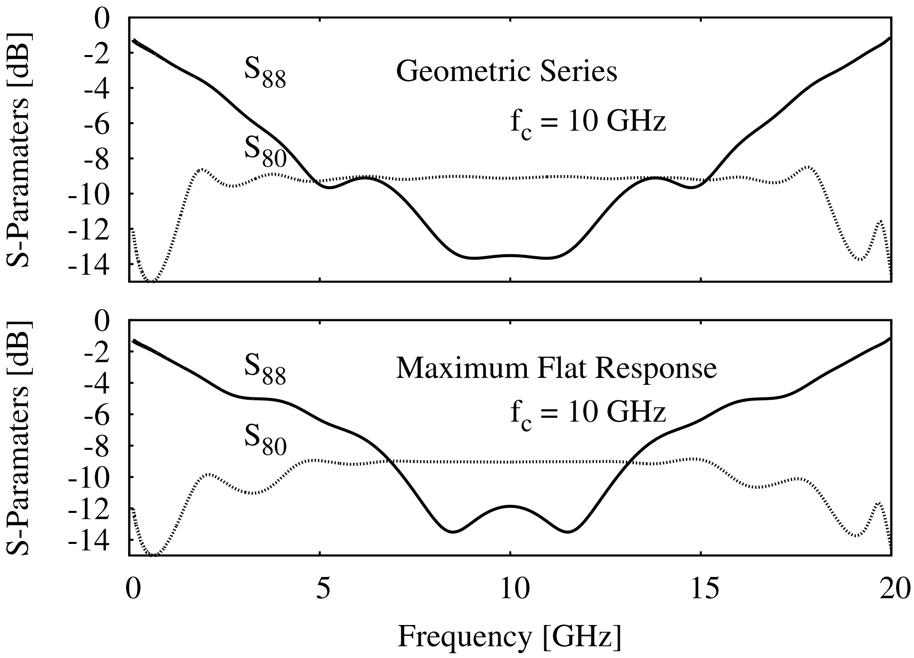


Figure 4.8: Simulated Port Isolation for Geometric 8:1 Wilkinson Power Splitter. Simulated port isolation of 8:1 Wilkinson combiner between Port 0 and Port 8 for geometric series and maximum flat response. Note that the maximum flat responses will be different for the two designs on account of the different designs, 3111 vs 2220.

4.2.3 Isolation

Isolation between the input and output parts of a Wilkinson splitter port is achieved by placing resistors in between divided power branches. No current flows through these resistors in the even mode. Resistor values are chosen to null reflections between ports in the odd mode. The number of resistors and their values are selected to minimize reflections between ports at maximum bandwidth [40].

Figure 4.8 shows S-parameters for refection (S_{88}) and throughput (S_{80}) for the 2220 Wilkinson divider for the worst case, i.e. when power is applied to one of the output ports with the rest of the output ports and the input port matched and

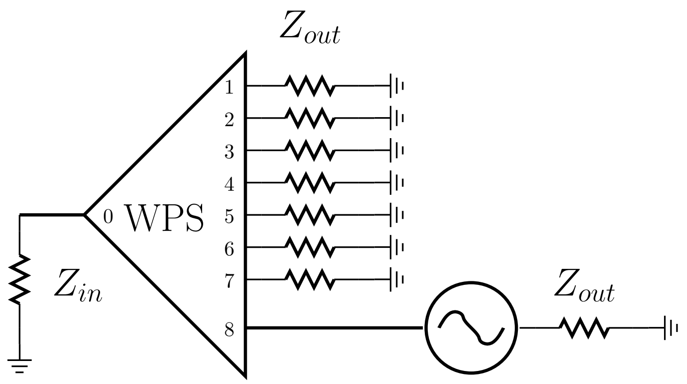


Figure 4.9: Isolation parameter measurement. Power is applied to port 8 of the Wilkinson power splitter while all other ports are terminated in matched loads.

terminated (see Fig. 4.9). One can see that the reflection S_{88} from port 8, where power is applied, is similar for both responses. Figure 4.8 shows the S-parameters for a geometric divider similar to that shown in Fig. 4.5 but with only two quarter-wave segments per stage. The impedance values of this circuit were chosen to give a geometric series response and a maximum flat response, using actual impedance values possible in fabrication instead of the ideal calculated values. The choice of even mode response has little effect on the isolation behavior of the splitter.

As for the even mode analysis, it is worthwhile to consider a new design of the Wilkinson power splitter. Figure 4.10 shows a schematic of the 3111 configuration Wilkinson power splitter. This design has fewer resistors and quarter-wave stages than similar 2220 or 1221 configurations.

Figure 4.11 shows the S-parameters of the 3111 divider shown in Fig. 4.10(a), using actual impedance values. The circuits were simulated using AC analysis in the WRspice simulator [41] for the central frequency of 10 GHz, chosen as a design

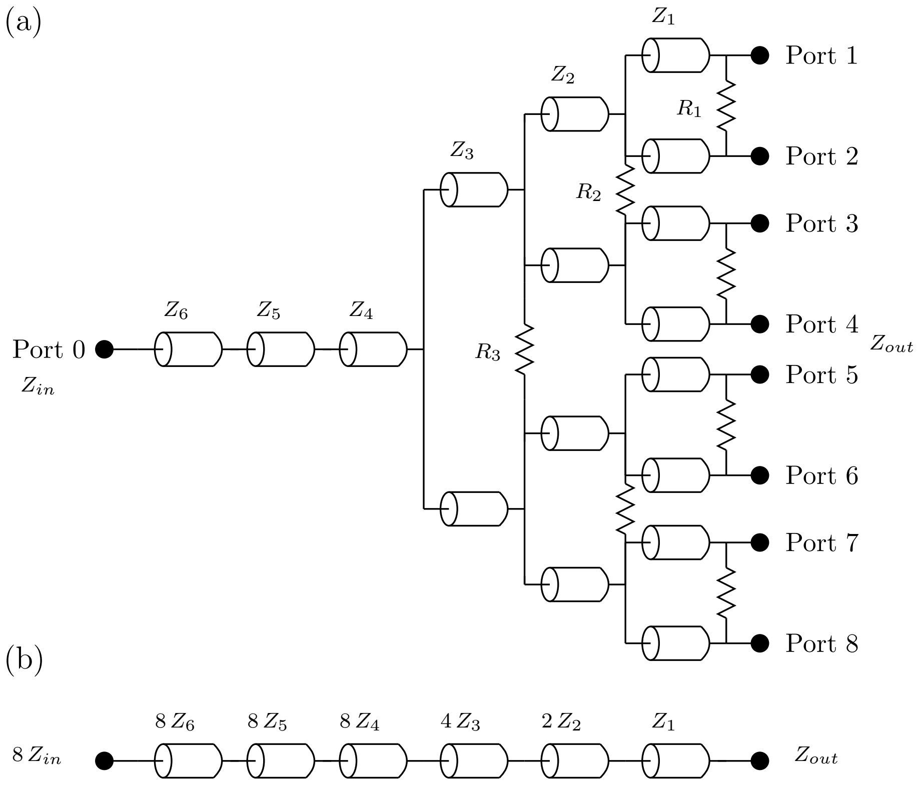


Figure 4.10: Circuit schematic for N23PS. (a) Schematic of final design for N23PS, a six-stage, eight-way Wilkinson power splitter. (3111 configuration.) Similar to Fig. 4.2. Input line on the left has impedance Z_{in}. Output line on right has impedance Z_{out}. Stages 1 – 3 are single-stage Wilkinsons. Stages 4 – 6 are impedance matching stages. All elements on vertical line share same design values. Note that resistor R_3 in this schematic is equivalent to R_4 in Fig. 4.2. Resistors R_3 and R_5 from Fig. 4.2 have been eliminated in the final design. (b) Even-mode analysis schematic of the schematic in (a). Starting from the output, each Wilkinson stage increases the input impedance by a factor of two. The eight-way Wilkinson in even mode is equivalent to a quarter-wave transformer with segment impedances $8Z_6$, $8Z_5$, $8Z_4$, $4Z_3$, $2Z_2$, and $1Z_1$.

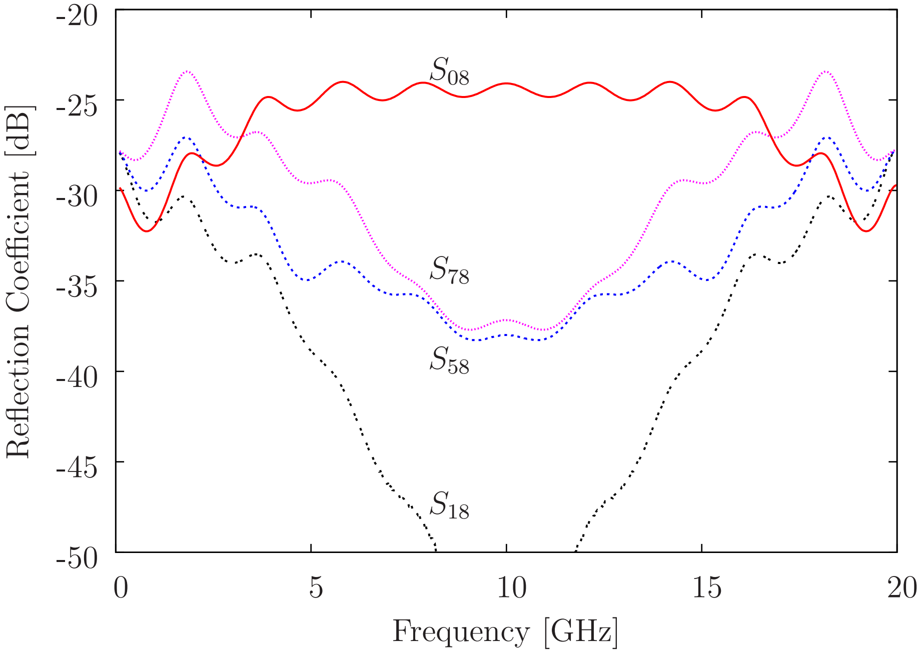


Figure 4.11: Wilkinson 3111 Simulated S-Parameters. Simulated S-Parameters of the Wilkinson 3111 maximum flat response power splitter which has a design frequency of 7.5 GHz. All S-Parameters are given as $10\log(V/V_0)$. S_{80} (red) is the ratio of the voltage at the "input" divided by the voltage applied to output port 8. S_{88} (light blue) is reflection off the output port where the voltage is applied at the output port. S_{87} (green) is throughput to adjacent port. S_{85} (dark blue) and S_{81} (purple) represent ports two and three branches away on the Wilkinson, respectively. Within the design frequency range the throughput to Port 0 is about -5 dB while reflection remains below -15 dB. Throughput to other ports remains under -10 dB.

compromise between available real estate and cryoprobe limitations. The design rules limit the widths of microstrips to certain values, and thus limits the impedances to certain values.

This design was optimized for a 1:8 Wilkinson transformer with a minimum required depth $N = 6$ of $\lambda/4$ segments. Impedances of the $\lambda/4$ segments were calculated by analogy with a quarter-wave transformer of the same depth with all branches taken in parallel, using (4.3) or (4.4) as appropriate. In the geometric series response, the ratio between two adjacent sections was held constant at 1.34. Note that in the geometric series response design, the impedances repeat at each stage, and this makes the design scalable to an arbitrary number of power divisions. On the contrary, parameters in the maximum flat response have to be recalculated for each particular case, as in (4.4) above.

The geometric series and maximum flat response designs involve opposite trade-offs in reflection and bandwidth. As can be seen from S-parameters shown in Fig. 4.7 the geometric series needs double the number of stages that the maximum flat response design required to achieve a comparably small level of reflections. In this case, the reflection S_{88} at the design frequency is less than -30 dB. The geometric response design with $N = 12$ shows similar behavior to the maximum flat response design with $N = 6$.

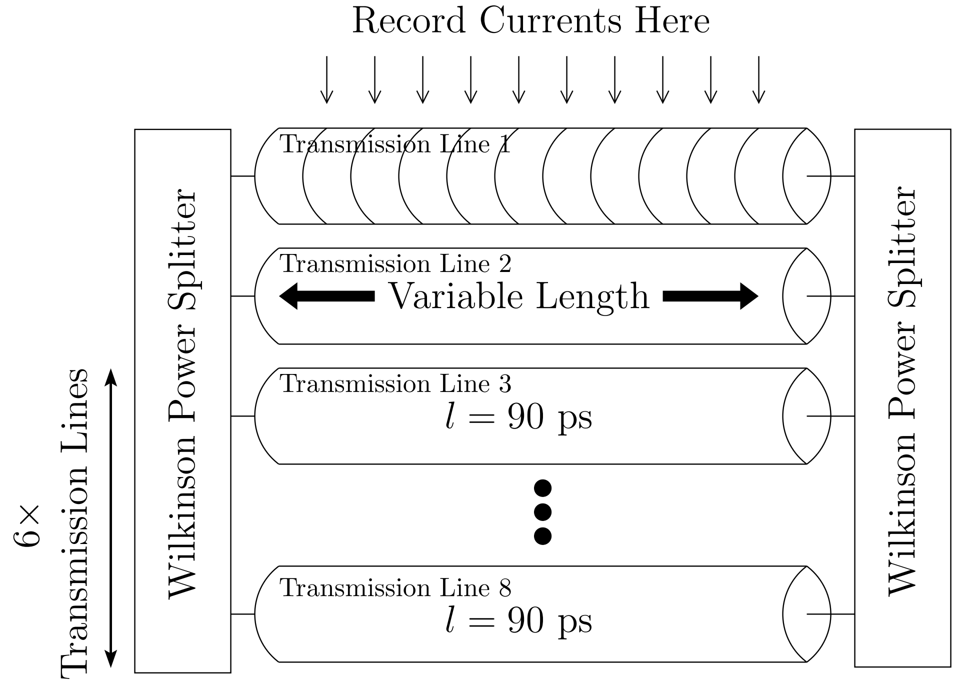


Figure 4.12: Block diagram for measuring standing currents. Two Wilkinson power splitters are connected through eight microwave transmission lines, each with a nominal length of $l = 90\,\mathrm{ps}$. The bottom six transmission lines connect the two power splitters. The second transmission line has a variable length $\tilde{l}$. The top transmission line is simulated as eleven shorter transmission lines in series, each with a length of $l = 8.18\,\mathrm{ps}$. In simulation, the currents between the short transmission lines can be recorded. These currents are plotted in Figs. 4.13 and 4.14.

4.2.4 Current Distribution

The main consideration in choosing one power splitter design over another is the requirement for current uniformity in the clock power lines. I analyzed the current uniformity by embedding relatively long $\tau = 90\,\mathrm{ps}$ ($l = 9$ mm) clock power lines between two dividers (one of which was used as a combiner) and monitoring the current profile at 10 equally spaced points of 9 ps apart. (See Fig. 4.12.)

Figure 4.12 shows a block diagram of the setup used to simulate standing

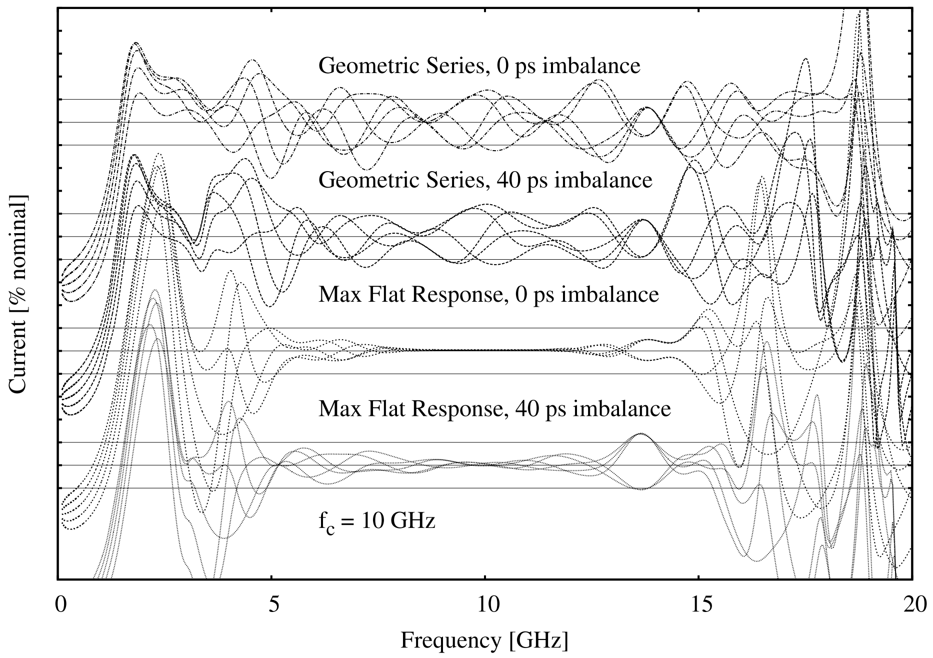


Figure 4.13: Simulated standing wave currents at ten locations inside clock lines between power splitters in the 4440 configuration, as shown in Fig. 4.16. Nominal and 10% of bias current are indicated by straight lines. Graphs offset for clarity. Imbalance refers to extra electrical length of only one line.

waves in the transmission lines. Two Wilkinson power splitters are connected by eight transmission lines in total. Six are regular transmission lines with electrical length $l = 90\,\text{ps}$ on ports 3–8. Port 2 is connected to a transmission line with length $\tilde{l}$, which I vary in simulation to induce odd mode behavior in the Wilkinson power splitters. Port 1 is connected to a series of 11 shorter transmission lines with length $l = 8.18\,\text{ps}$. Though this transmission line is of the same overall length, in the simulation I can record the currents at the nodes between shorter transmission line segments. (See Appendix C, Section C.3 on page 268 for the netlist used to generate this data.)

Figure 4.13 shows the results from standing wave analysis for the case when the electrical length of one power line is either 0 ps or 40 ps longer than the others. In this simulation, I used the 2220 splitter configuration, as shown in Fig. 4.6. For comparison, Fig. 4.14 shows the results from the standing wave analysis for the case when the electrical length of one power line is either balanced or 10% longer than the others for the 3111 design shown in Fig. 4.10 (only for the maximum flat response). The former case corresponds to 44% imbalance in electrical length, which far exceeds the expected worst case in a practical circuit.

As described in Chapter 2 (pg. 75) the delay is less than 2 ps for 10^6 junctions. In that experiment, the coupling between clock lines and junctions was greater by a factor of three than in regular RQL circuits. It would take 10^7 Josephson junctions to accumulate this amount of phase delay due to dynamic switching [27]. Junction switching therefore does not contribute to an imbalance in the power splitters. In Nb microstrips with a propagation speed of about 100 μm/ps, it will take 4 mm to accumulate 40 ps of delay. This is a large distance compared to the scale of circuit elements, and therefore small variations in the clock length through a chip will also not contribute greatly to imbalances in the power splitters.

Figure 4.13 shows that the expected distribution of the bias current is significantly different between the geometric series and maximum flat response for the 2220 splitter. The maximum flat response is designed to have minimum variation within the bandwidth, and this property carries over to the bias current distributions. Figure 4.14 shows that the distribution of the bias current is significantly different between the 2220 layout (see Fig. 4.6) and the 3111 layout (see Fig. 4.10).

The geometric series with 6 stages does not satisfy requirements of $\pm 10\%$ variation in bias current (see Fig. 4.13). In contrast, the maximum flat response for the 3111 design gives an octave of bandwidth (5-15 GHz) with no more than $\pm 10\%$ variance in bias current (see Fig. 4.14). Between 8.5-11.1 GHz the 3111 design gives $\pm 1\%$ variation. This indicates that maximum flat response performs much better for our figure of merit of current uniformity. The 3111 design does even better between about 7–13.5 GHz with $\pm 1\%$ variation. The 3111 design also saves space by matching between $50\,\Omega$ and $32\,\Omega$, which the 2220 design does not.

4.2.5 Final Design

To test the capabilities of Wilkinson power splitters I had two designs fabricated, which I will denote as *Monrovia 20 Power Splitter* (M20PS) and *Norwalk 23 Power Splitter* (N23PS). The first design, M20PS, is a 4440 configuration Wilkinson and has a geometric response and no isolation resistors (see Fig. 4.5). I used this design to test the even mode response of a power network. The second design, N23PS, seen in Fig. 4.10, is a 3111 Wilkinson and has an optimized maximum flat response. I chose the parameters for this design it using (4.4), (4.6a) and (4.6b). In a second iteration of the design of N23PS, the impedances and resistors were simulated and fine tuned to produce better standing wave ratios. In addition to good responses in the even and odd modes, I also checked the behavior of the currents that flow between splitters and combiners.

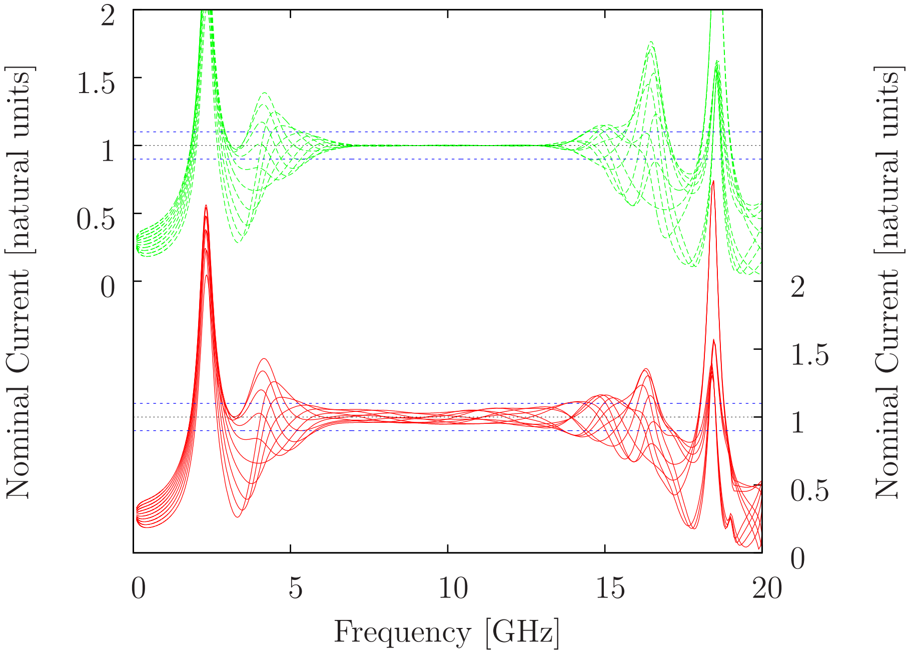


Figure 4.14: Standing Waves in Wilkinson 3111 Power Network. Simulated current amplitudes at ten points along the transmission line connecting Wilkinson 3111 power splitters in Fig. 4.10 as a function of frequency. Top graph (green) shows results with eight transmission lines of equal length between Wilkinsons with $f = 10\,\text{GHz}$ designed center frequency. Bottom graph (red) shows same currents along a regular length of transmission line when another line has a 10% longer electrical length. ±10% lines shown for reference. With the maximum flat design and no length imbalance the currents stay within ±10% between about 5 GHz and 15 GHz, with less than 1% variation between about 7 GHz and 13 GHz. With the 10% length imbalance, the operational range with less that 10% variation in bias current is between about 6 GHz and 14 GHz, with substantial variation throughout. Graphs offset for clarity.

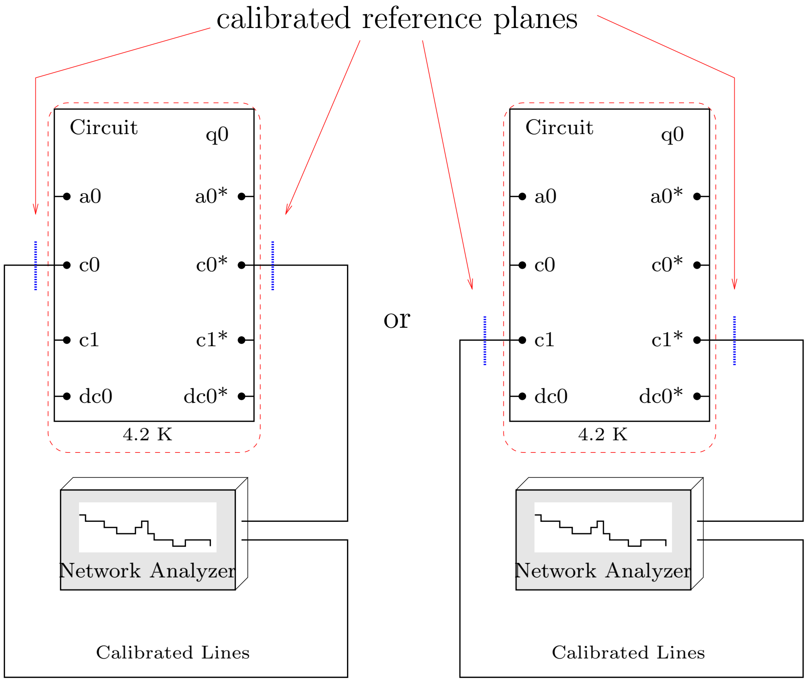


Figure 4.15: Experimental setup for measurement of S-parameters. S-parameter measurements were performed only on inactive circuits and no other equipment was attached. The network analyzer was connected to the clock input and output of the probe. Using either a standards kit or a standards chip, the network analyzer could be calibrated to the top of the probe or to the pads on the chip.

4.3 Standalone Test

To test the above simulation results, I designed and had fabricated two Wilkinson splitters. The first one, N20PS, is shown in Fig. 4.5(a) and was designed with a geometric series in the 4440 configuration. The second one, M23PS, is shown in Fig. 4.10 and was designed with the maximum flat response in the 3111 configuration to measure isolation.

To measure the S-parameters of the M20PS and N23PS circuits, I used the setup shown in Fig. 4.15. The coaxial cables leading to and from the network analyzer were calibrated out of the measurement using a standards kit. Calibration chips were also available, though because the chip had not been characterized, the network analyzer used a generic profile for the standards chip. Ultimately, calibrating to the top of the probe with the calibration kit with a known profile yielded better results.

4.3.1 Experimental Setup

Figure 4.16 shows a microphotograph of the geometric response 1:8 Wilkinson divider/combiner M20PS test circuit. Due to limited space on the chip only the 12-step geometric series response even mode circuit was tested. The circuit was optimized for a center frequency of 17 GHz and 10 GHz bandwidth. The impedance values were adjusted to accommodate the available discrete values of the width of signal wire due to the 0.5 μm lithography step (see Table 4.1). The total occupied area of the circuit is 630 μm (0.08λ) × 1550 μm (0.2λ) with approximate dimensions of one segment of 400 μm (0.05λ) × 320 μm (0.04λ). The dimensions of the circuit compare favorably with previously published lumped-element and distributed designs [42, 43]. The clock signal enters from the left at port 1, is split eight ways, and then combined and taken off chip on the right at port 2 (see Fig. 4.16). Figure 4.16 shows a test circuit schematic with a 18.5 dB resistive tap added to monitor current. The tap is included in the bottom power line at an electrical distance of approximately 0.13λ from the power splitter output port. To balance the circuit, identical taps are added to other lines and resistively terminated on chip (not shown). The

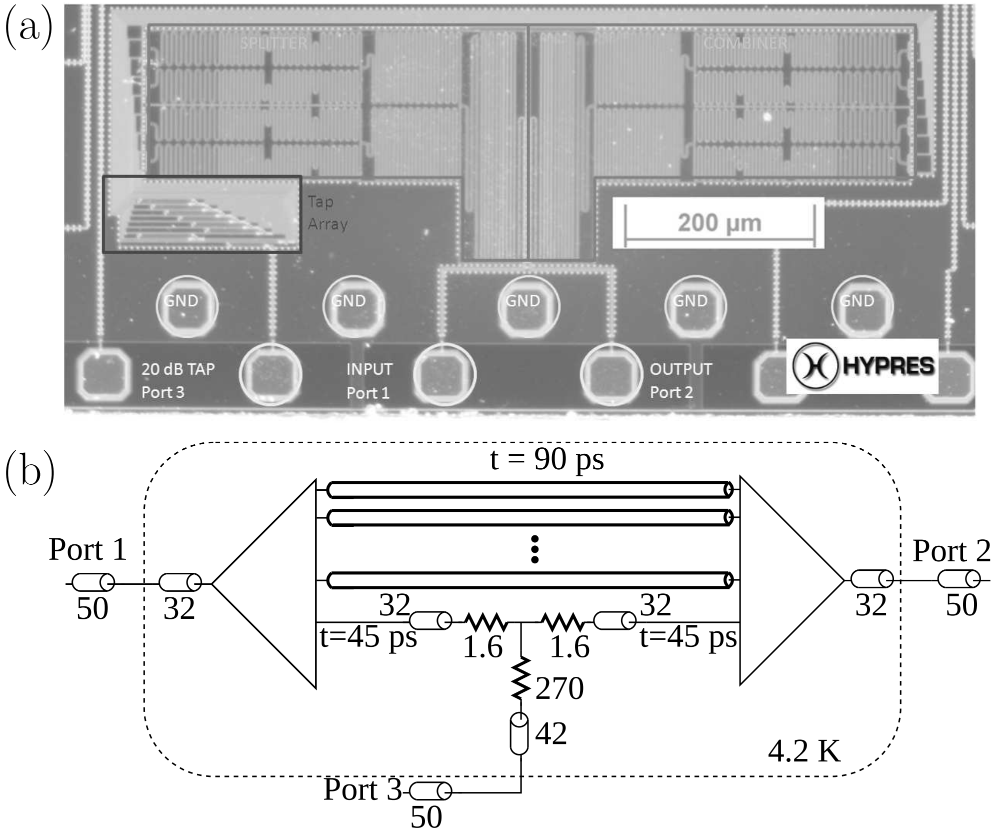


Figure 4.16: M20PS even mode test. (a) Microphotograph of M20PS circuit showing input, output, and 20 dB tap array (left). Fabricated by Hypres using the 4.5 kA/cm^2 process. (b) Circuit schematic of M20PS circuit. All ports were impedance matched. The impedance-matched tap is shown only on one line for clarity. All other lines had a similar tap but grounded directly on chip instead of going to a pad.

circuit has three contact pads to monitor power at each port.

Figure 4.17 shows a microphotograph of the chip containing N23PS, the maximum flat response 3111 Wilkinson divider/combiner test circuit. I used this chip to test the 6-stage maximum flat response in even and odd mode. The circuit was optimized for a center frequency of 7.5 GHz with a 6 GHz bandwidth. The impedance values were adjusted to accommodate the available discrete values of the width of signal wire due to the 0.5 μm lithography step (see Table 4.1). The physical size of the power splitter in N23PS is comparable to that of the geometric response Wilkinson in M20PS. The size of the N23PS relative to the wavelength of the M23PS is smaller by a factor of $7.5/17 = 0.44$, although this comparison does not take into account the different fractional bandwidth. This circuit was designed with RQL shift registers with separate inputs to differentially load the lines and cause an imbalance in the power network. Unfortunately, design errors on the chip prevented all but one of the shift registers from being utilized. Nevertheless, I was still able to obtain S-parameter measurements of both pairs of splitters and combiners on chip.

I tested the Wilkinson divider/combiners using an American Cryoprobe (geometric response) and a High Precision Devices (HPD) probe (maximum flat response) with the microwave test setups shown in Figures 4.16 and 4.17 respectively. The American Cryoprobe probe is designed for a 3 dB at 10 GHz cutoff frequency and has pressure contacts. The HPD probe has a 5 dB cutoff at 26 GHz. For both probes, a calibration of the coaxial lines leading from testing equipment, though the probe, and to the chip was performed on a separate chip for different combinations of the contact pads to accommodate the spread in electrical parameters

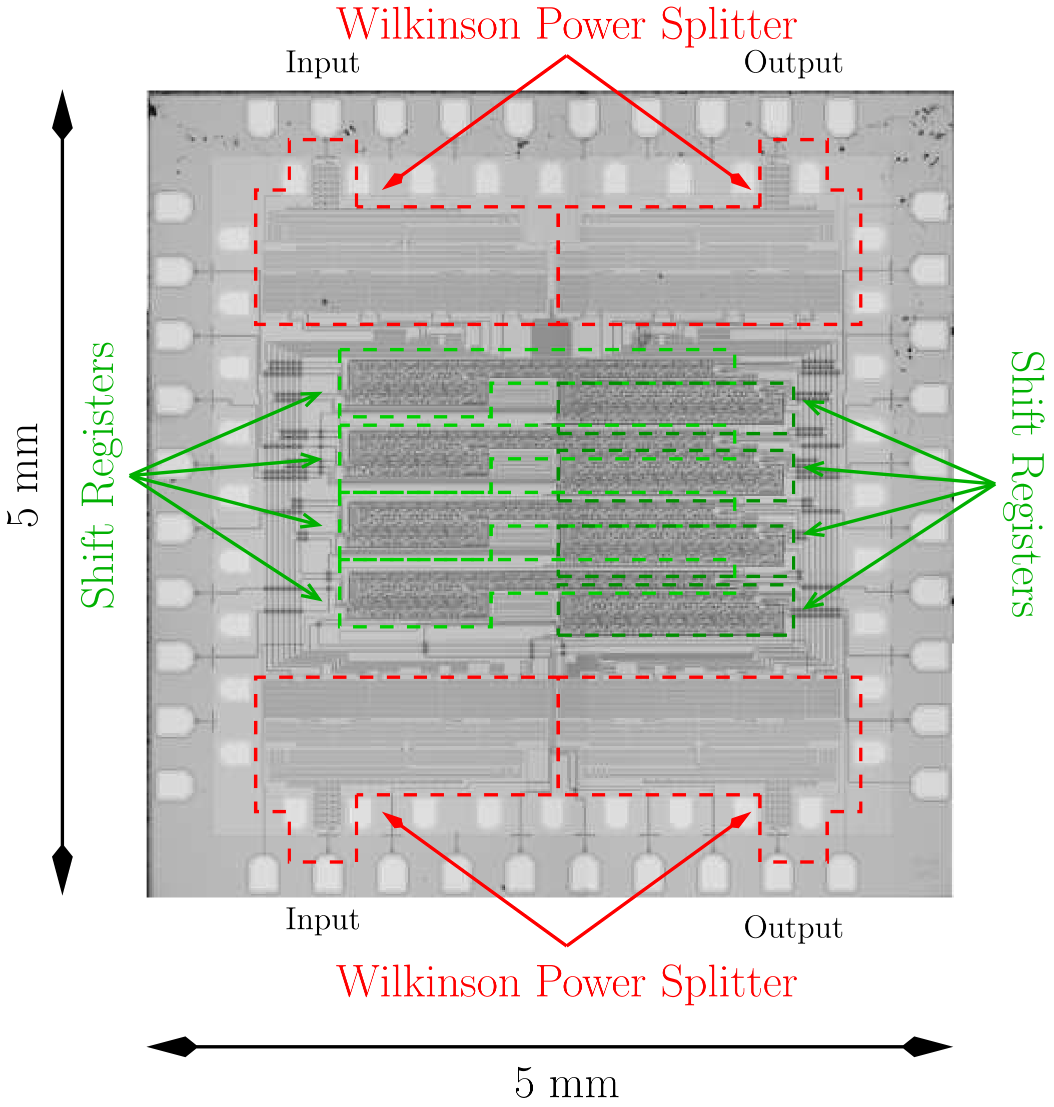


Figure 4.17: Microphotograph of Norwalk 23. The N23PS circuit is shown on this chip. The four Wilkinson power splitters (red boxes) are in the four corners of the chip. The bottom two and top two are connected by eight lines between them. Input and output to each of the splitter/combiners is shown. Eight shift registers are in the middle (green boxes). Data input to the shift registers is on the left (not marked); output is on the right. This chip was used to test the odd mode analysis of a 6-stage maximum flat series response 1:8:1 Wilkinson power splitter/combiner.

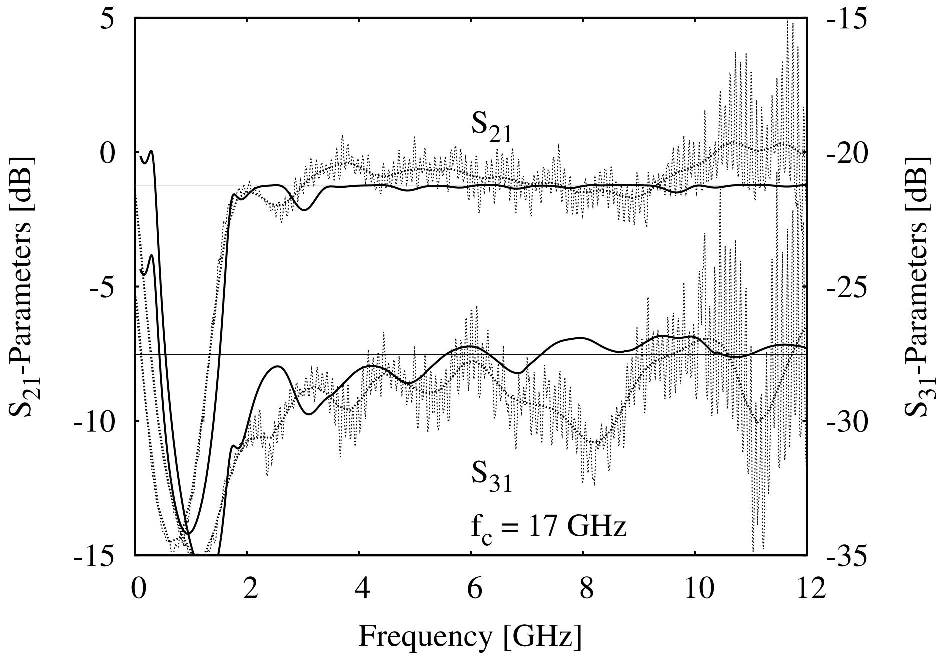


Figure 4.18: Comparison of simulated results (solid)with Measurements (dashed) from M20PS with a 17 GHz center frequency and 10 GHz bandwidth.

between contact pads. These calibrations indicate that this experimental setup and both probes can reliably be used up to 12 GHz. At higher frequencies the response becomes highly non-uniform due at least partially to limitations of the calibration procedure.

4.3.2 Measurements

Figure 4.18 shows a comparison of simulations and measurements on the M20PS circuit shown in Fig. 4.16. Experimental transmission parameters S_{21} and S_{31} match the simulated results to within 3 dB at all frequencies within the probe range. Both curves are flat to within 0.5 dB above 4 GHz. Below 4 GHz the device acts like a current divider and not a power divider. Experimental and simulation results for power at the tap (S_{31}) agree well within 1 dB between 3-7 GHz. Both curves show an approximately 1.8 GHz periodicity which corresponds to pad-to-pad resonance (5.5 mm) due to impedance mismatch at the pressure contact. The dips at 8 GHz and 11 GHz are due to inaccuracy in the probe calibration. From the results, I can conclude that overall response at the tap in the M20PS circuit is flat, which indicates no standing waves in the power line.

Figure 4.19 shows Spice simulation results of the 3111 circuit shown in Fig. 4.17 with both 0 ps and 40 ps imbalance in one of the lines connecting the splitter to the combiner for a center frequency of 7.5 GHz. Outside the bandwidth (about 4.5–10.5 GHz) the simulation with an imbalance plays little role. Within the region of interest, the 0 ps imbalance has very low reflection below -20 dB. Even with a 40 ps imbalance the reflection is between -10 and -20 dB. Thus, in this range which is the desired response of the maximum flat design, the reflection is still very flat.

Figure 4.20 shows measurements of the N23PS maximum flat response (3111) Wilkinson power splitter/combiners with a center frequency of 7.5 GHz. This data comes from N21CLA, which had an identical design of the power splitter[4] as in N23PS, now was on a different chip. Experimental transmission parameters S_{12} (purple) and S_{21} (blue) match simulations shown in Fig. 4.19. The expected frequency range based on simulation is between 4.5 GHz and 10.5 GHz. As can be seen in Fig. 4.20, the measured bandwidth reaches from about 3.5 GHz to 11 GHz, beyond which the transmission parameters become considerably smaller. The transmission

[4] See Chapter 6

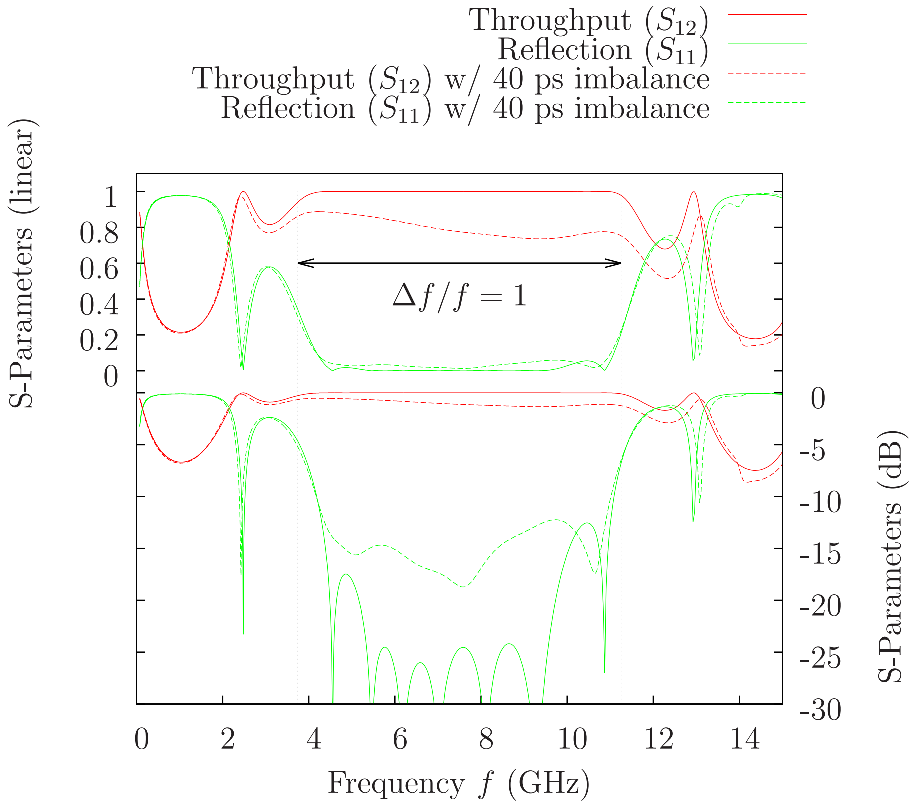


Figure 4.19: Simulated reflection for the N23PS circuit with a configuration as shown in Fig. 4.10. Green (dotted) curve shows nominal case with no imbalance between lines connecting power splitters. Red (solid) curve shows the results on the overall reflection parameter S_{11} at either input or output port of the whole power network when a 10 ps imbalance is in one of the eight lines between power splitters. Close to the center frequency the reflection (solid green) is less than -25 dB, whereas with the imbalance the reflection (dashed green) ranges from -20 dB on the low end to -10 dB lower than 4 GHz or higher than 10 GHz.

parameters of the maximum flat response Wilkinson splitter/combiner compares favorable with the transmission parameters of a simple through line between pads, shown as the dotted black curve in the Fig. 4.20. The inset shows the transmission is fairly flat, usually within 1 dB, between 3 GHz and 12 GHz. As with the 4440 design, below 3 GHz the 3111 power divider acts like a current splitter. The reflection is around -20 dB but reaches closer to -10 dB at some points, notably one peak close to 6 GHz. The flat nature of the responses indicates only minimal standing waves. Finally, I noticed that during layout, part of the 8th segment was inadvertently shortened by $1/20^{th}$ of a wavelength at 7.5 GHz. The results here thus contain an imbalance of at least 2 ps on one of the lines.

4.3.3 Power Splitter Test Conclusions

Among the two designs I tested (M20PS and N23PS) the geometric response was the least desirable because it had the highest reflections within the range of interest. Figure 4.10 shows the schematic of N23PS. The values of R_2 and R_4 in the 1221 design of Fig. 4.2 were uniformly small for all responses. They dissipated relatively little power compared to R_3 and R_5 and I have discarded them in the design of N23PS. This reduces the splitter from a 1221 configuration to a 1111 configuration with only 15 transmission line stages and seven resistors in total. Adding two stages to create a 3111 Wilkinson with $N = 6$ costs little in space, which is why I chose the 3111 configuration for the design of N23PS. Compare this to the 21 stages and 10 resistors in the 4440 configuration. The extra stages in the 3111 design help step

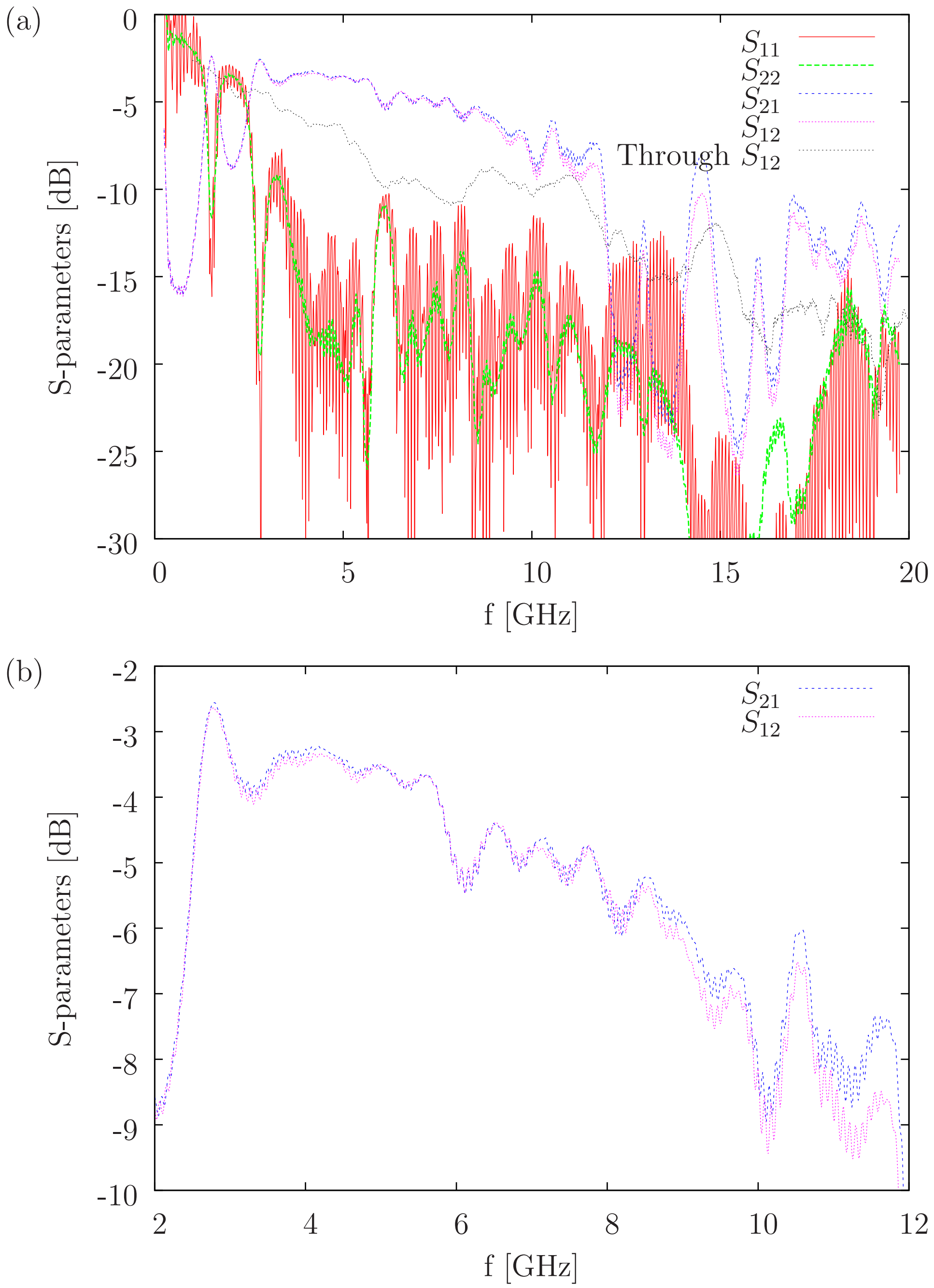


Figure 4.20: Measured S-parameters on N21CLA Wilkinson power splitter. (a) Blue and purple curves show throughput S_{12} and S_{21}. Red and green show reflections S_{11} and S_{22}. The throughput (S_{12}) from calibration measurements of a standards chip is shown as a gray curve for reference. (b) A detailed view of the throughput up to 12 GHz. Between about 3 GHz and 12 GHz the curve is relatively flat. The noise on S_{11} was due to a damaged connector on the network analyzer.

down the impedance from $Z_{in} = 50\,\Omega$ to $Z_{out} = 32\,\Omega$. For the N23PS circuit I found the initial impedance values using (4.4) and used (4.6a) and (4.6b) for the resistor values. I then simulated the Wilkinson 3111 with the schematic shown in Fig. 4.17. (The netlist is given in Appendix C.) Using commercial software (Agilent Advanced Design System 2009) I optimized the impedances and resistors to get the desired frequency responses.

Figure 4.21 shows the results of this optimization on the isolation and crosstalk responses in the N23PS circuit. Using only three resistors between ports, the isolation is substantial. Most power is absorbed, with less than -50 dB reaching the input port and even less reaching any output ports. Figure 4.14 shows the effect of line imbalances on the distribution of current in transmission lines between the Wilkinson power splitters. The effect of imbalances on current distributions is small. When perfectly matched, the transmission lines have almost no variation over 6 GHz range and are still within 10% of nominal down to about 5 GHz. Even a 40 ps imbalance still has all currents within the tolerances down to 5 GHz. On chip, this would amount to a 1 mm or more difference in length on a 10 square millimeter chip, much more than is expected.

Figure 4.19 shows the overall reflection of the whole power network for the optimized 3111 design used in N23PS. A 10 ps imbalance changes the reflection coefficient from -30 dB to about -20 dB. Though this is a jump of 10 dB, the overall reflection still remains low, no higher than other elements in a testing setup. (See Appendix C for details on the S-parameters of the probe itself.) The experimental test of M20PS suffered from pad-to-pad resonances. Figure 4.19 indicates that these

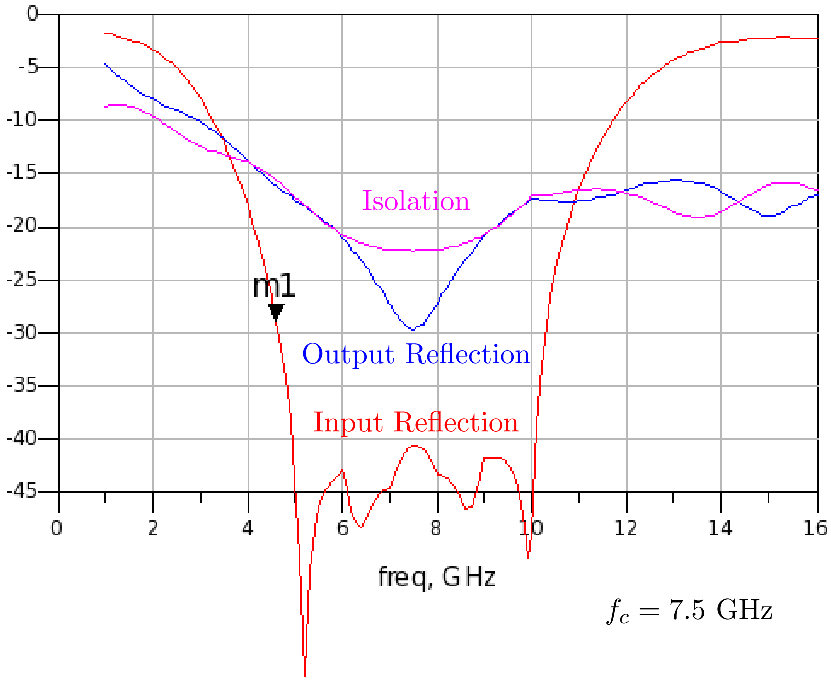


Figure 4.21: S-parameters from ADS for N23PS. Three curves characterizing the final design of N23PS calculated from ADS. Input reflection is S_{00}, the reflections off the input port to the Wilkinson power splitter. Output reflection is S_{88}, the reflection off of signals arriving at one of the outputs. Isolation is S_{78}, the signal at an adjacent port when a signal arrives at one of the outputs.

small variations may be even smaller in the final design.

The tolerance to imbalances in the clock lines of the N23PS power splitter is smaller than for the previously studied Wilkinson 1221 configuration power splitter. However, the design of the N23PS Wilkinson incorporates a 50 Ω input impedances, eliminating the need for a separate high-bandwidth transformer. The maximum flat response design used for N23PS has a bandwidth appropriate to RQL circuit needs. Furthermore, it displayed adequate current uniformity. The design is also space efficient. Looking ahead, I will need four splitters on each chip with more than one clock phase and more than one parallel pipeline. For each chip, efficient size is a design necessity.

4.4 Test with RQL Circuits

After completing the test described in Section 4.3, I still needed to test the Wilkinson power splitter in an RQL circuit. From here forward, the design used was the maximum flat response Wilkinson power splitter in the 3111 configuration and fabricated on the N23PS and N20CLA circuits. This design minimizes the effects of imbalances and achieves a broad frequency operating range.

Despite design and fabrication problems, the chip pictured in Fig. 4.17 still had one functioning shift register powered by the Wilkinson power splitters. Although differential loading of the circuit by selectively turning shift registers on and off was not possible, I was still able to measure the clock power margins on the shift register. The power margins can be found by noting that the RQL shift register will fail if at any point along the clock line powering the shift register the current becomes too high or too low. If too low, the SFQ pulse will fail to propagate. If too high, the junctions will switch without input and generate a continuous series of pulses. Both cases can be readily observed using an oscilloscope. In both cases, failure is defined as the power at which the output becomes probabilistic, the output voltage sometimes registering as high and sometimes as low for a given pulse.

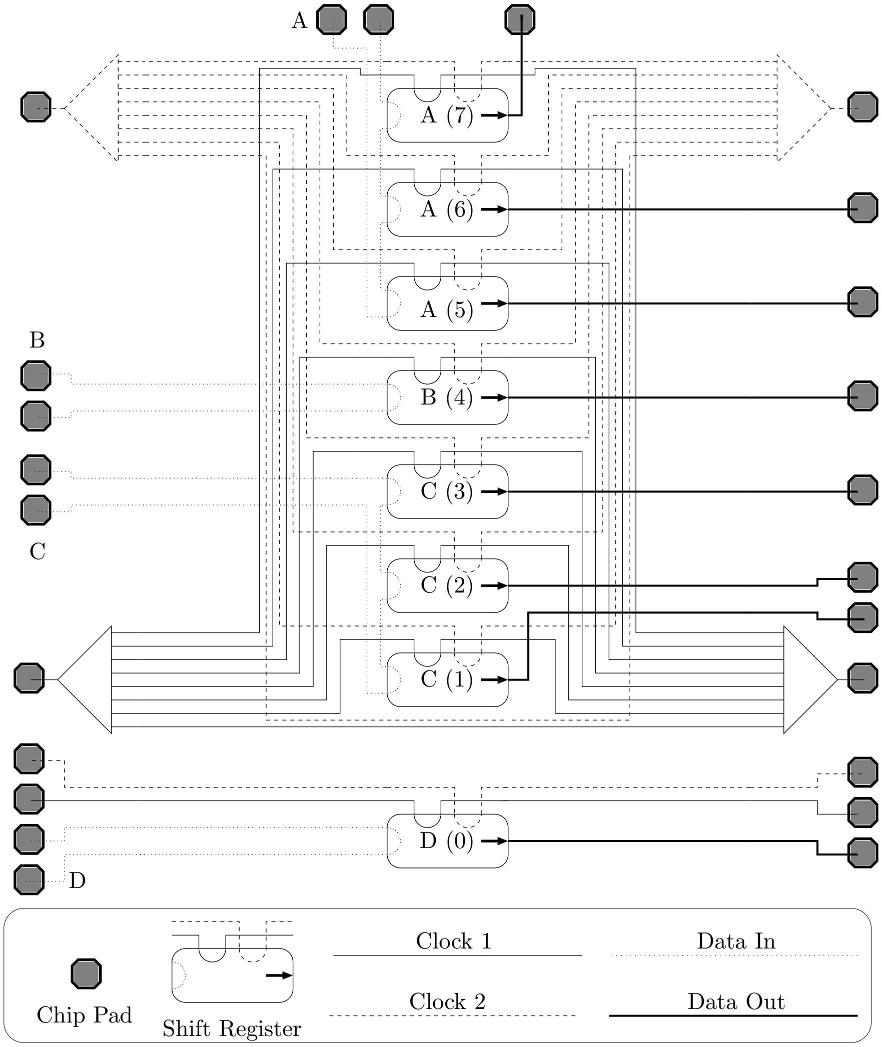


Figure 4.22: Odd mode test block diagram for N23PS. Four Wilkinson power splitters in the corners provide power and a clock signal to seven shift registers in the middle, labeled 1–7 and in groups A–C. The eighth lines from the power splitters goes between splitter and combiner but does not supply power to a shift register. An eight shift register (0) is powered directly by two clock lines and is completely separate from the rest of the circuit. Shift registers 1–3 are in group C, triggered by input on port C Shift register 4 is triggered by an individual input port B in group B. Shift registers 5–7 are in group A and triggered by input from pad A. Each input has a corresponding return pad. Each shift register has its own output pad.

4.4.1 Power Margin Experiment

Figure 4.22 shows a schematic of the circuit I used to find the power margins in the N23PS circuit. Eight shift registers are labeled with a unique number (starting with 0 at the bottom) and a letter corresponding to four pairs of pads A – D. Each pair of pads is connected in series to a corresponding set of shift registers. Thus, shift register 0 (SR0) is part of group D, and so on. Each group of shift registers is triggered by the same input. Four power splitters supply two-phase clock power to the shift registers in A, B, and C, with an eighth line between the splitter and combiner which is unloaded by any digital circuits. Group D serves as a control circuit and has an identical but completely isolated shift register. I used this circuit to check for malfunctions in either the other shift registers or the power network. Each shift register outputs to a different output pad. Not shown are two dc bias lines for D (separate) and A, B, and C (in series). This setup allows certain branches to be loaded while others are left unloaded to determine the effects of RQL loading on the power network. The network analyzer can also be used on the power networks, although not while the RQL circuits are engaged.

The experimental procedure is as follows: First I established the optimal operating point. For a given frequency, the optimal operating point (for flux bias and input phase) will depend on the phase between clocks, the offset between data and the clocks, and the clock power. Because of the self-correcting timing of RQL circuits the data offset has little effect over a broad range of operation. The relative clock phase between the two clock lines, which should be offset by $\pi/2$, is the most sensitive parameter. After I found a general operating point by adjusting the relative clock phase, clock power, and input phase, I turned the clock power down until the circuit was barely functional. I then adjusted the clock phases until correct operation was established. I then turned down the power again, repeating the process until any significant change of the clock phases resulted in failure. At the low-power, optimum clock phase point I turned the data phase up and down, noting failure points, before returning it to the middle of the range. This established the low power margin. From here, I turned the power up until failure. This established the high power margin.

One feature of RQL is that the reciprocal pulses need not follow on the same clock cycle. The Anritsu pattern generator I used is rated only up to 12 GHz for a non-return-to-zero (NRZ) output, essentially limiting me to 6 GHz data input speed. However, by "over-clocking" the clock signal by a factor of 3/2 faster than the return-to-zero (RZ) data rate, the reciprocal pulses can follow one-and-a-half clock cycles later instead of one-half clock cycles later. For example, a data steam triggered off of 6 GHz signal will generate a positive SFQ pulse every 166.6 ps. These pulses can be propagated on an RQL circuit being clocked at 9 GHz. Instead of following half a clock cycle later (as measured by the ac bias applied to all junctions), the reciprocal pulse follows one-and-a-half clock cycles later. Although this does not increase the data speed, it does allow the power splitter to work at higher frequencies where the power network may have better performance.

4.4.2 Data and Analysis

Figure 4.23 shows the results of my power margin measurements on the M20PS circuit. The input power for high and low margins (adjusted for the attenuators and other equipment attached to the probe) is shown by the black points. The region of correct operation is shown as the green shaded area between these data points. On the top of the figure, the power margin in dB is shown by the black curve. Red and blue curves are measurements of the S-parameters of the power network[5] from input chip pad to output chip pad. Above 6.5 GHz data rate the clock rate was increased to $\frac{3}{2}$ the data rate, which is marked by the "overclock" region in the Figure.

Examining 4.23, we see that the S-parameters look similar to those measured previously (compare to Fig. 4.20). I also note that between about 4 GHz and 8.5 GHz the power margins are between 2 dB and 5 dB. The power margins are on average about 2.4 dB in width, or about ±36%. Above 8.5 GHz the data input rate is approaching 12 GHz NRZ, the maximum rated for the equipment. At 1.75 GHz, where the power splitter has a very high local throughput and almost no reflection, the power required takes a sharp drop, as expected. In the region between 4 GHz and 8.5 GHz the throughput (S_{12} and S_{21}) and average power are flat.

Table 4.3 shows the pad-to-pad and splitter-to-combiner lengths for the N23PS circuit in units of the wavelength at the frequencies of interest. The length of each quarter-wave segment in the power splitter is approximately 4421 μm in length, with

[5] Note that these measurements are from N23PS. The data shown in Fig. 4.20 is taken from N21CLA. Because N23PS contained design and fabrication errors I have avoided relying on data from this circuit if at all possible. However, here I am comparing the behavior of the power network to the behavior of the circuit powered by this same power network. A valid comparison can only be made by this direct comparison.

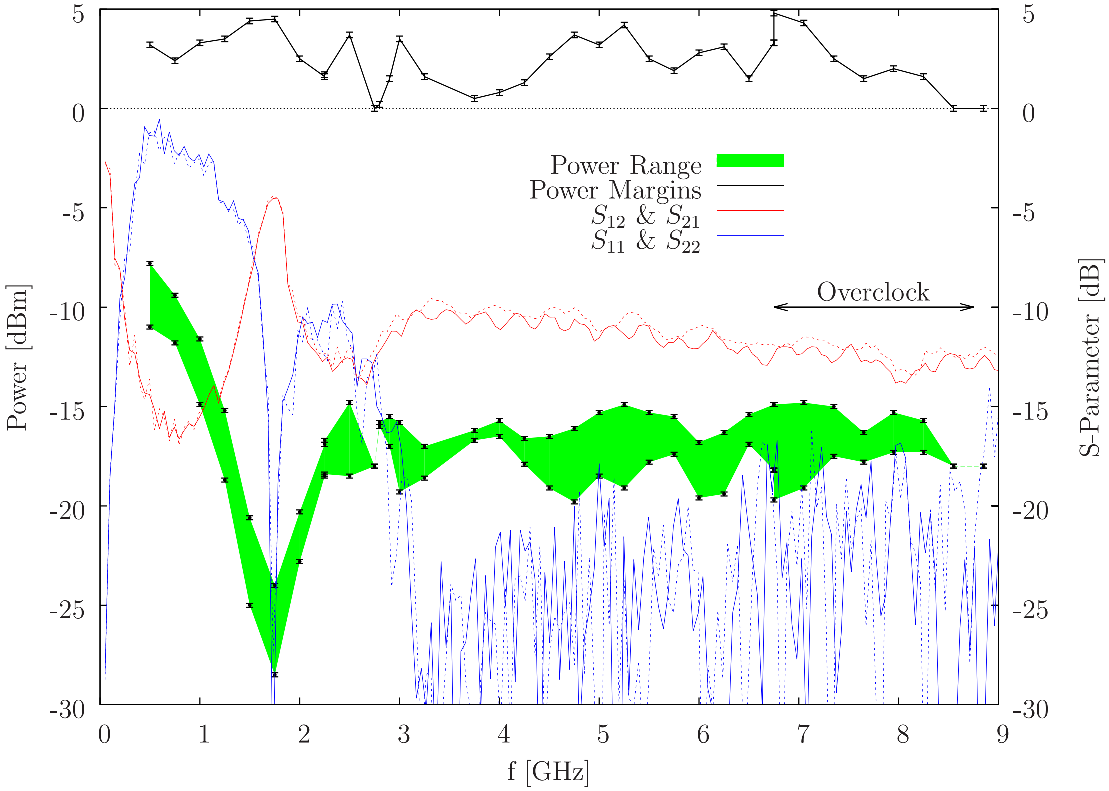


Figure 4.23: Wilkinson-powered RQL circuit measurements of N23PS. Clock amplitude margins (black) overlaid with S-parameters (blue and red) of the on-chip power network. Green region shows operational range of SR1 after adjusting input power for losses in the splitter, attenuator, and on-chip power network. Size of the margins is shown above, ranging from 0 dBm to almost 5 dBm. S-parameters are overlaid with throughput in red (S_{12} and S_{21}) and reflection in blue (S_{11} and S_{22}). Measurements in both directions are shown in solid and dashed lines. The overclock region shows where data was taken with the data rate equal to 3/2 the clock speed. Particularly noticeable is the correspondence between the throughput and the power range over the 3–9 GHz operational region. Frequency is given for the clock speed, not data input speed.

Table 4.3: Chip resonance lengths for frequencies f of interest. Here, $\lambda = c'/f$ is the wavelength at frequency f.

f	Pad-to-Pad distance	Splitter-to-Splitter distance
1.75 GHz	0.84 λ	0.14 λ
2.75 GHz	1.32 λ	0.22 λ
3.5 GHz	1.68 λ	0.28 λ
6.5 GHz	3.13 λ	0.53 λ

slight variation to account for different propagation speeds at different impedances. The length of the clock lines between power splitters is 10 718 μm, with variations of less than 50 μm due to design constraints. Even at 6.5 GHz, the length is still a small fraction of a wavelength (about 6%). At this frequency Fig. 4.23 shows a small, narrow drop in margins (a half-wavelength reflection between splitters could cause standing waves and account for the loss of margins). At 3 GHz the loss of margins is due to the high reflections of the power splitter.

4.5 Conclusions

The Wilkinson power splitter is a common part of many microwave systems. However, the demands of RQL required a new kind of Wilkinson power splitter. Using even and odd mode analysis, I decomposed the design of the 8-way Wilkinson splitter into the design of quarter-wave impedance matching filters for the even mode and two-stage wilkinson power splitters for the odd mode. I considered four different configurations of the Wilkinson power splitter. The 4440, 2220, and 1221 configurations all performed best with the maximum flat response design. The key requirement for RQL circuits is a low VSRW on transmission lines connecting power splitters, for which the maximum flat response is a clear choice. In the final design of the power splitter, I chose a 3111 configuration with 50 Ω input and 32 Ω output. Starting from an initial design using the maximum flat response, I used ADS software to further increase the performance of the power splitter.

In the second half of this chapter I presented experimental measurements in Wilkinson power splitters that shown they can be used for power RQL circuits. In addition to even and odd mode analysis, I used general filter theory to create three designs. This resulted in the 3111 design shown in Fig. 4.10 with the design parameters shown in Tables 4.1 and 4.2. My measurement on this design revealed that the bandwidth was larger than expected and the margins were adequate.

Chapter 5

Experimental Verification of RQL Timing Parameters

5.1 Introduction

In Chapters 1 and 2, I described the foundations of Reciprocal Quantum Logic. In Chapter 3, I described the timing parameters of RQL and corresponding VHDL models that I used to develop larger circuits. This chapter describes my effort to verify the timing behavior of junctions predicted by the analytic timing model (eq. (3.8)) in Chapter 3. With this in mind, I designed and tested three circuits in order to measure: (1) the timing delay of junctions on a single phase as a function of input phase, clock frequency, and clock amplitude; (2) operating margins on the input phase and clock amplitude as a function of frequency; and, (3) operational margins of the clock amplitude and frequency for a long, deep pipeline shift register.

With these three experiments I was able to test the timing behavior predicted by equation (3.8), verify the operational boundaries derived from this equation, the switching time t_0, and test the self-correcting influence of the clock boundaries on input phase delay. These results confirmed the timing behavior of SFQ pulses, and also showed that RQL circuits can operate over greater input phase ranges than expected based on my models.

Figure 5.1 shows a microphotograph of the experimental chip. I designed this chip, designated *Norwalk 22 Timing Experiment*, or N22TE, and had it fabricated

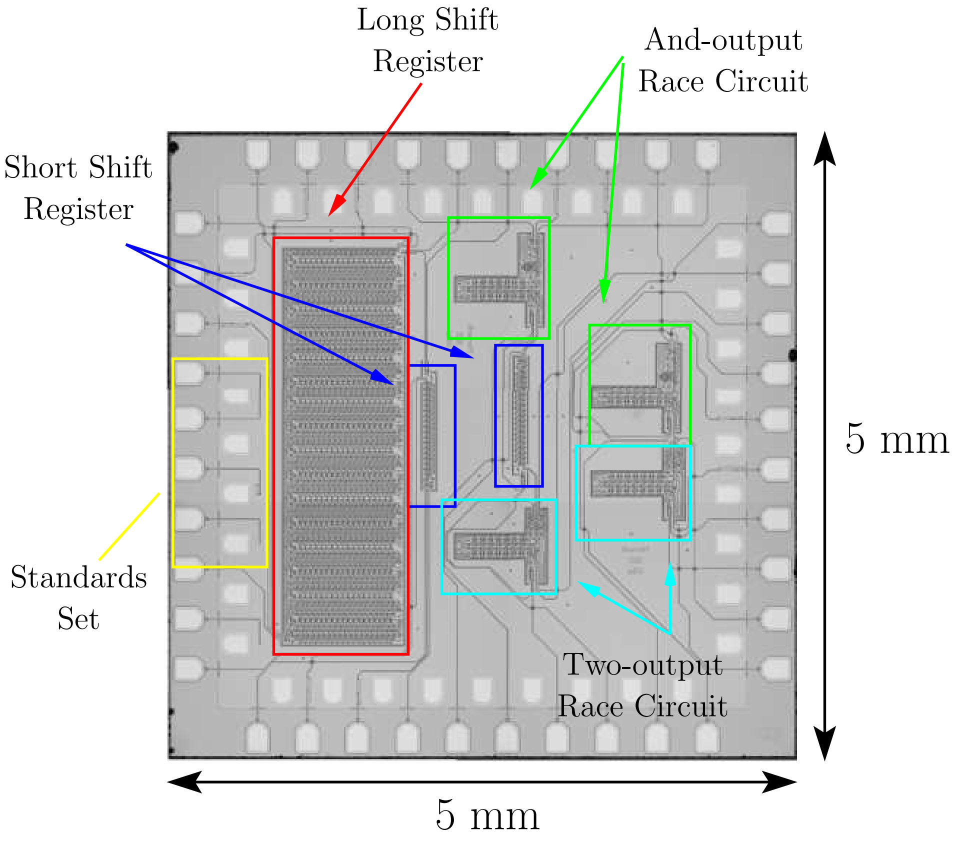


Figure 5.1: Microphotograph of N22TE, fabricated at Hypres. This chip contains three experiments on RQL timing. The chip contains one long shift register (red box), two short shift registers (dark blue boxes), two And-output race circuits (green boxes), two Two-output race circuits, and a set of standards (yellow box). Pads on the far left include a set of standards with a through-line, 50 Ω on-chip termination, an open, and a short. Note that this layout is flipped from the design layout by the fabrication process. The short shift registers were used only to confirm operation of the chip.

at Hypres (see Appendix E for details of the Hypres process). The the circuits I used for the three experiments can be seen on the chip. I duplicated some of the circuits to mitigate the chance of fabrication errors disabling a circuit. Three different circuits were used because no single circuit can test all aspects of the timing model. In the first circuit I measured the output timing as a function of input timing by comparing the timing *difference* between two identical SFQ pulses in a race condition. These two pulses propagate through two parallel JTL lines with different lengths, and each line has a separate output. I observed the output of this *two-output race circuit* experiment directly on a Tektronix TDS 8000 Digital Sampling oscilloscope[1].

In a second experiment, I alternatively compare two propagating pulses fed to the AND gate on chip. While this method lost information on the relative timing, it eliminates the uncertainty introduced by the testing equipment. This *and-output race circuit* experiment provided a binary result of correct or incorrect operation as a function of input time and clock frequency and amplitude, and thus was a good test of the operational margins.

In the third experiment I tested a *long shift register circuit* and examined the behavior of SFQ pulses crossing clock boundaries. In this experiment, I measured the maximum operating frequency of the long shift register with deep pipelines. The measured frequencies correspond to the maximum operating frequency of the circuit with a given pipeline and serve as an experimental measure of the switching time parameter t_0. Also, the timing model predicts a stability limit after which pulses will

[1] All references to an oscilloscope in this Chapter refer to this model.

no longer have self-correcting timing (see pg. 88). I measured the phase boundaries for correct operation and checked these boundaries against the predictions of the analytic model.

5.2 Circuits and Simulation for Experiments 1, 2, 3

Figure 5.2 shows block diagrams and a view of the circuits in the Cadence computer design environment.

5.2.1 Simulation of Experiment 1 — Two-output Race Circuit

In the first experiment, an SFQ pulse generated by a single input splits into two pulses propagating along separate paths with a different number of junctions (see Fig. 5.2(a)). The difference in number of junctions between paths was designed to accumulate enough time delay that could be measured directly on the oscilloscope. Figure 5.3 shows waveforms of pulse propagation between short and long paths simulated at 4 GHz. The short track contains 4 JTLs with 8 junctions and the long path contains 20 JTLs with 40 junctions. The last JTL on both paths is shaded on the figure. It is an amplification JTL with twice the critical current, required for the input to the output amplifier. One can see that the total accumulated delay is around 20 ps. This 20 ps delay is on the limit that can be measured by oscilloscope with sampling measurements at the range of frequencies between 1 and 6 GHz. The number of junctions in the long path was designed for pulse propagation in the range of frequencies up to 5 GHz at maximum clock amplitude. The total

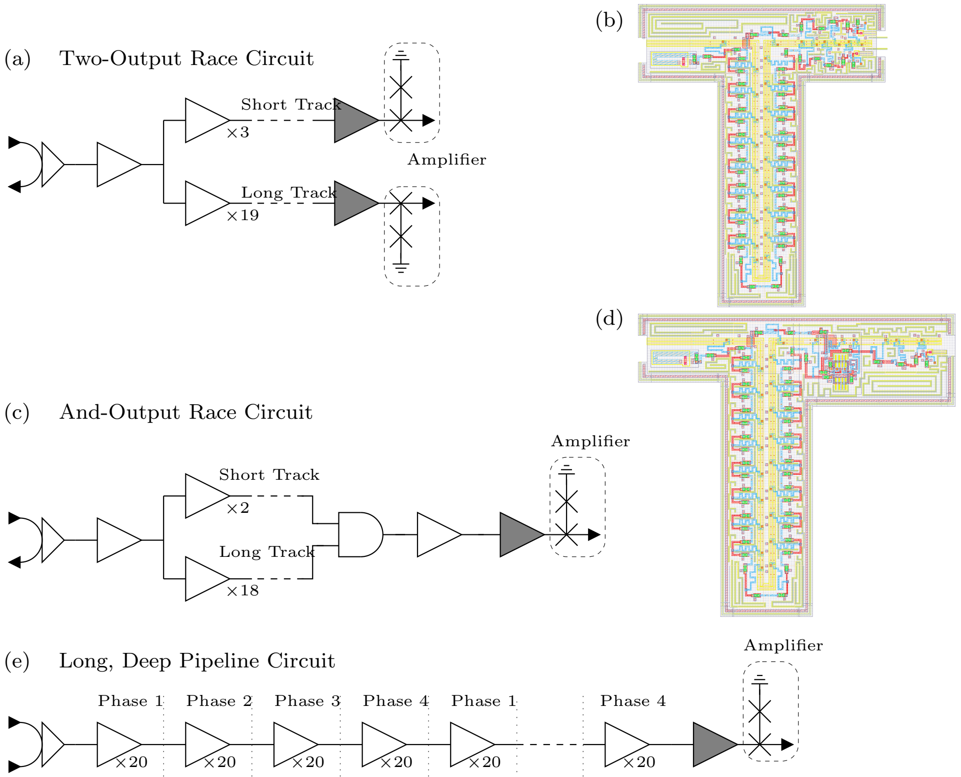


Figure 5.2: Block diagram and layout of N22TE. Block diagrams show the essential features of the three timing experiments. Shaded JTL symbols indicate higher critical currents were used in the junctions, $J_{c1} = 282\,\mu\text{A}$ (instead of the normal $J_{c1} = 141\,\mu\text{A}$) and $J_{c2} = 400\,\mu\text{A}$ (instead of $J_{c2} = 200\,\mu\text{A}$). (See Fig. 2.2 on pg. 44.) (a) & (b) Block diagram and layout of two-output race circuit. A single input generates an SFQ pulse which is split to travel along two paths. The paths have different lengths and each path has its own output. The two outputs can be seen on the right of the layout, which has one track going down the bottom of the "T" shape, and the other along the top. (c) & (d) Block diagram and layout of and-output race circuit. Similar to the previous circuit, one input is split into two tracks. The two tracks feed into an AND gate, which produces output when the circuit is operating. (e) Block diagram of long, deep pipeline shift register. Unlike the previous circuits, no splitting of the SFQ pulse occurs and the circuit uses multiple clock phases.

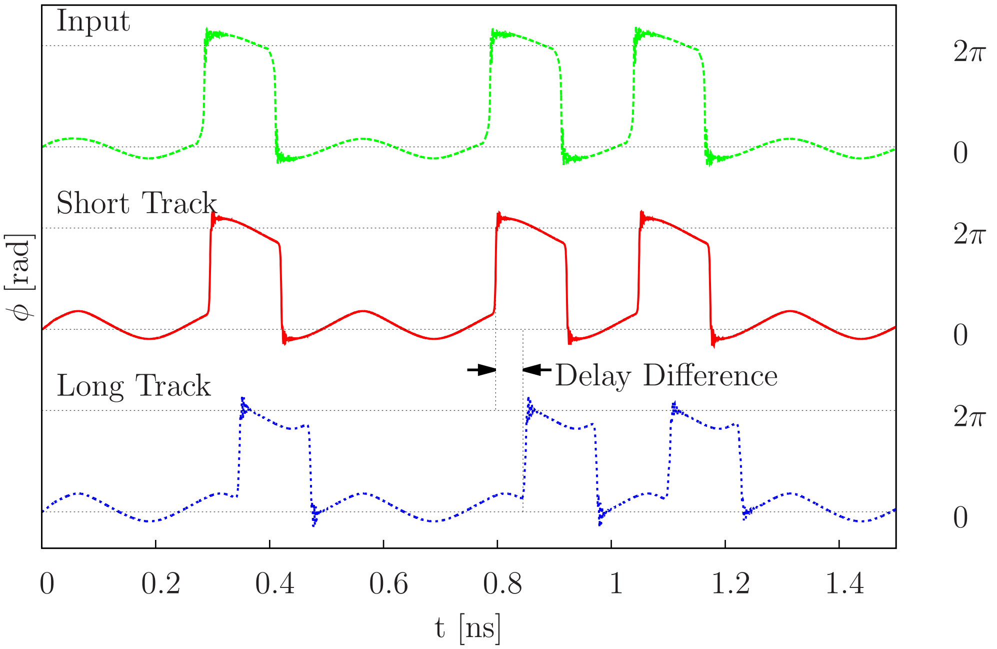


Figure 5.3: Plot of the input and output phases versus time for the two-output race circuit shown in Fig. 5.2(c). This figure shows the input phase of the first and last junctions in the two-output race circuit, as simulated in Spice for Hypres 4.5kA/cm^2 process at 4 GHz. The input shows the phase rise and fall with SFQ pulse pairs. The short track shows an output almost immediately after the input junctions switch. The long track takes noticeably longer. This timing difference between output from the short and long tracks is the delay difference between the two positive SFQ pulses.

clock line path length difference is approximately 1200 µm, the delay in the clock line between the two paths is approximately 12 ps. At 5 GHz the wavelength is approximately 2×10^4 µm. The change in wavelength over the size of the circuit is approximately 6% at the maximum operating frequency. There is always an unknown delay introduced by the output cables. However this difference in delay is constant and can be subtracted out from the measurement data.

Figure 5.4 shows predicted timing behavior in this experiment as a function of input phase as found from (3.8). There are three curves showing output phase delay for the path with 1, 8 and 20 JTLs (or 2, 16, and 40 Josephson junctions, respectively), and also a curve showing the accumulated output delay between long and short paths. One can see that the cutoff point of the input phase is dominated by the longer path and that the time delay difference between long and short paths is fairly constant until the phase approaches the timing limit, at which point it sharply increases. This time delay difference can be measured in a real device as a function of input phase by adjusting the phase of the data relative to the clock.

Figure 5.4 shows the difference in timing behavior for both short and long paths. Three points are worth noting in this graph: (1) the cut-off point is dominated by the long path, for obvious reasons; (2) the timing difference is fairly constant until the phase reaches close to the timing limit, at which point it sharply increases; and (3) the delay flattens out at $\theta = 0$. These qualitative features should be readily observable in the actual data

5.2.2 Simulation of Experiment 2 — And-output Race Circuit

The second experiment was similar in concept to the first experiment in that it also measured relative delay between a short and long path. The same number of junctions is used in the short and long track. However, instead of observing the delay on the oscilloscope it was sampled by an AND gate. This experiment eliminates the uncertainty in measuring picosecond-scale delays on the oscilloscope. The AND gate compares the relative timing of the pulses — modulated by one

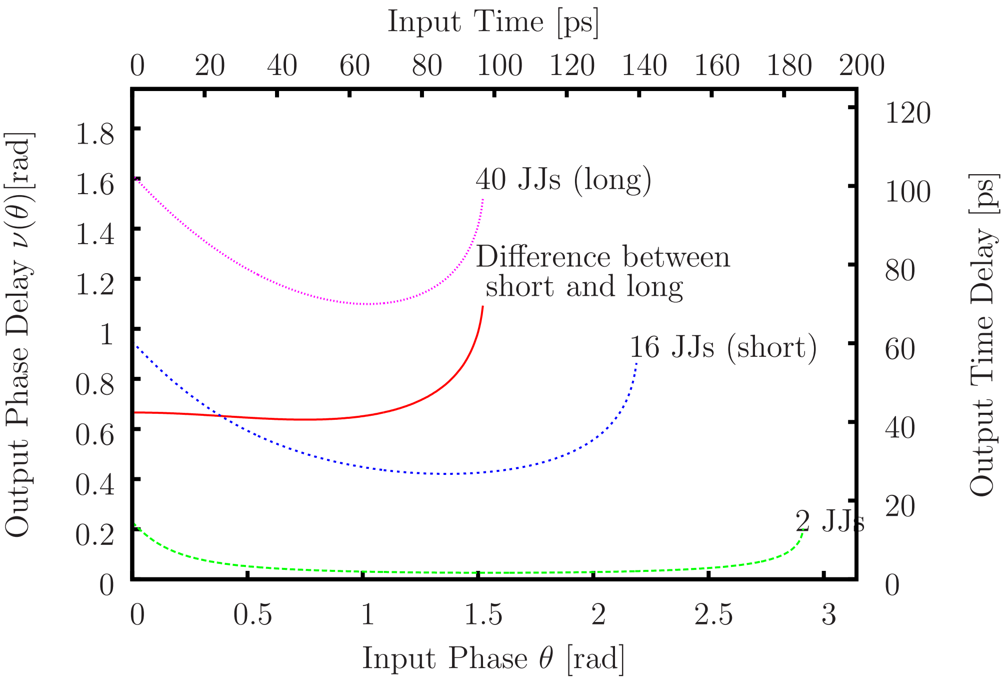


Figure 5.4: Two-output race circuit timing predictions. Output phase delay calculated from (3.8) of the JTL circuit in Experiment 1 for Hypres 4.5 kA/cm^2 process at 2.5 GHz, with two parallel paths — short with 8 junctions total and long with 40 junctions total. Different curves show the phase delay as a function of input phase for 2 JJs, 16 JJs, and 40 JJs on a single phase. Paths with more junctions take longer. The red solid curve shows the expected difference in accumulated delay between long and short path.

clock period — and produces binary data. If two delayed pulses come within the same clock window, then the AND gate output is "one"' and otherwise the output is "zero". By changing the the amplitude of the bias current, I can modulate the AND output and observe the operation of the circuit for different input phases.

Figure 5.5 shows the expected regions with output and without output from the AND gate for this experimental circuit, calculated from (5.3b) and plotted for different frequencies. These calculated curves on the plot are derived from the

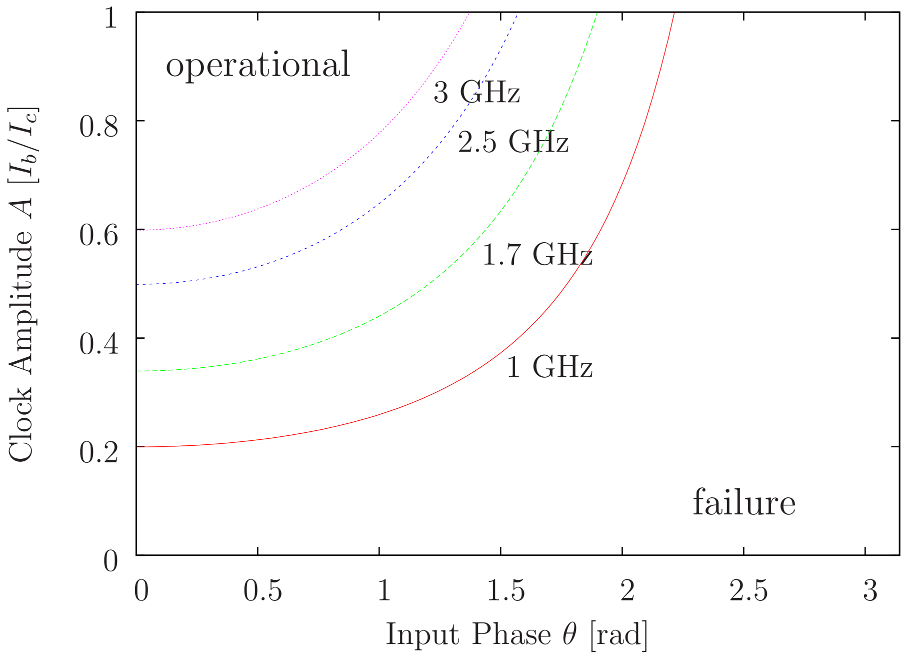


Figure 5.5: Operational space of N22TE. The four curves show the calculated boundary conditions (from (5.3b)) for correct operation for four different frequencies for N = 40. The contours for a given frequency can be mapped out by varying clock amplitude and input phase until a change occurs.

analytical expression (3.8)

$$\nu(\theta) = \arccos\left(\cos\theta - \delta\right) - \theta. \tag{5.1}$$

This is the equation I used in Chapter 3 to derive the boundary conditions for SFQ pulse propagation. Experimentally the most convenient quantity to measure is the limiting clock amplitude $A_{lim}(\theta, f)$ at which operation fails as a function of clock input phase θ and frequency f. From $A_{lim}(\theta, f)$ I can find expressions for the limiting frequency $f_{lim}(A, \theta_{set})$ as a function of clock amplitude A and a set, fixed input phase θ_{set}. I can also find the “late limit” on the clock input phase $\theta_{lim}(f, A)$.

The boundary condition is determined from (3.8) by the criteria

$$\delta - 1 < \cos\theta, \tag{5.2}$$

where $\delta = 6\pi f \mathrm{t}_0/A$. This criteria corresponds to the case where the SFQ pulse arrives too late in the clock cycle to switch the junction. I expect these equations to give a very good approximation to the boundary conditions of SFQ pulse propagation in the limit of frequencies where the pulse timing fitting parameters are close to unity (see Section 3.3, page 95). (In the general case where the pulse timing fitting parameters are not close to unity, the simulation would need to iteratively calculate the next output based on the previous input, using the fitting parameters.) Solving (5.2) for θ, f, and A gives the expected behavior:

$$\theta_{lim}(A, f) = \arccos(2\pi \mathrm{N} t_0 \frac{f}{A} - 1), \tag{5.3a}$$

$$f_{lim}(A, \theta_{set}) = \frac{1 + \cos\theta_{set}}{2\pi \mathrm{N} t_0} A, \tag{5.3b}$$

$$A_{lim}(f, \theta_{set}) = \frac{2\pi \mathrm{N} t_0}{1 + \cos\theta_{set}} f, \tag{5.3c}$$

with θ_{set} now a variable in operational space instead of the input phase. These three equations define the boundaries in operational space between successful pulse propagation and failure. From (5.3a)–(5.3c), one can see that for a given frequency there exists a minimum amplitude and a maximum phase input. As the frequency increases, the operational range decreases. All frequencies have a common upper limit on amplitude at $A = 1$, after which failure occurs due to overdriving of all junctions. This failure mechanism is unrelated to frequency or clock phase.

Coming back to Fig. 5.5, each curve divides the the operating space into regions where pulses propagate — above the curve — and regions where the pulses do not propagate — below the curve. Notice that for a given clock amplitude, the operational space for correct operation grows as frequency is decreased. That is, for slower clock frequencies the pulses may arrive later without preventing operation. For a given clock input phase, the clock amplitude can also decrease as the frequency decreases. Pulses arriving before $\theta = 0$ are not allowed (in this model) and at $A > 1$ a different failure mechanism takes over (all junctions switch regardless of input).

5.2.3 Simulation of Experiment 3 — Long Shift Register

I designed the third circuit to verify the effect of phase boundaries on timing stability and check the maximum operating frequency of an RQL circuit. Instead of trying to measure an RQL circuit in the 20 to 40 GHz range — which is challenging experimentally — I decided to design long, deep pipeline shift registers with 20 JTLs per stage. In this case, the maximum operating frequency from simulation was expected to be 5 GHz at maximum clock amplitude ($A = 1$). The timing behavior of this shift register is equivalent to a short shift register operating at 100 GHz with pipeline depth of one JTL. This follows from the definition of δ. In addition, this experiment allowed me to check the self-correcting behavior of an RQL circuit by measuring the clock power margins as a function of input phase. The detailed circuit schematic of the shift register is shown in Fig. 5.6; there are 1384 junctions and 40 junctions per phase, which gives 20 JTL segments per phase.

The long length ensured that practically any phase delay introduced at the input would be corrected by the time output occurs. The testing equipment (see

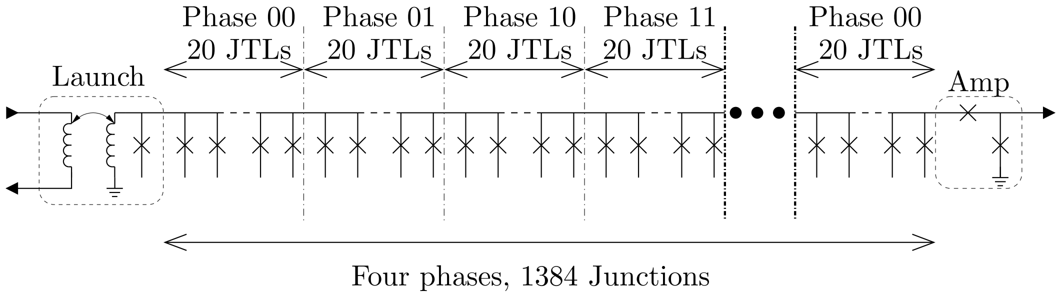


Figure 5.6: Long, deep pipeline shift register. Schematic of the long, deep pipeline shift register used in Experiment 3. (Intermediate stages are omitted for clarity.) Each phase contains 20 JTLs with 40 junctions, a pipeline depth designed to fail at $f = 5\,\text{GHz}$ and $A = 1$. Unlike previous experiments' circuits, this circuit has a four-phase clock with 80 JTLs per cycle, for a total of 17.25 full clock cycles of delay through the circuit. Two extra junctions from a squid and serve as output amplification at the output.

Section 4.4.1 on pg. 157) only worked up to a speed of about 6 GHz, and the circuits were designed to fail below this speed for maximum clock amplitude. Similar to the other two circuits, the final JTL in the shift register had critical currents of $J_{c1} = 282\,\mu\text{A}$ and $J_{c2} = 400\,\mu\text{A}$. This was the only circuit on N22TE that I tested with two clock lines and four clock phases; the previous circuits only had one phase on a single AC clock line and one DC flux offset line. The phase difference between the clock lines was controlled externally (see next Section).

5.3 Experimental Setup

Figure 5.7 shows a block diagram for the experimental setup, which has been reproduced from Fig. 2.14 in Chapter 2. However, the initial settings for the circuits was different. I obtained the optimal operating point under the assumption that

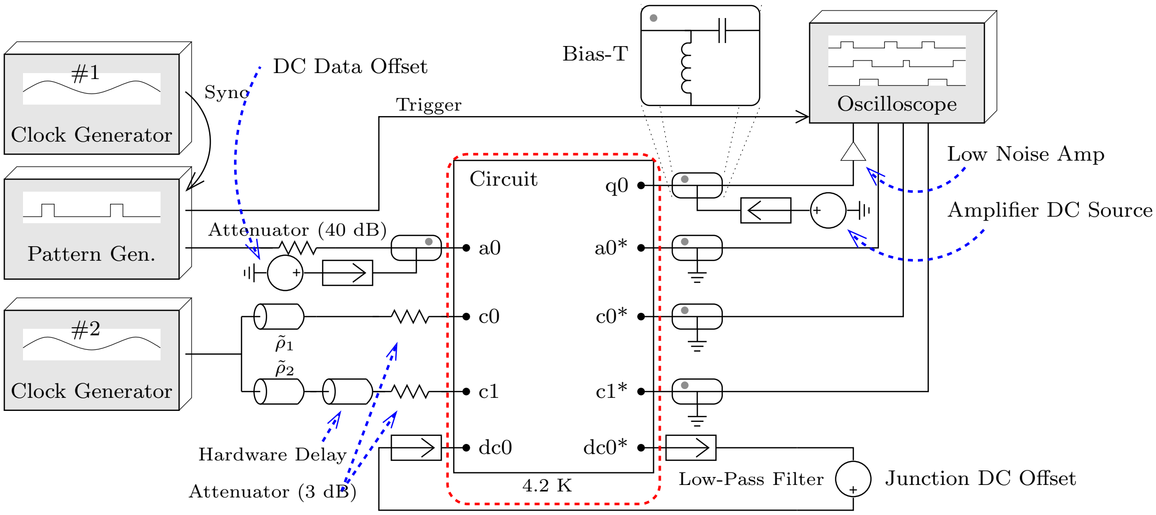


Figure 5.7: Experimental setup for timing experiments. Block diagram of the experimental setup for the timing experiments. a0: data input; a0*: data return; c0, c0*, c1, c1*: clock phases and returns; dc0: DC offset bias; dc0*: offset bias return; q0: experimental output. Similar to Fig. 2.14, although bias-Ts have been replaced with 3 dB attenuators in some cases. Low noise bandpass filters have a cutoff frequency of $f_C = 1\,\text{kHz}$. Low noise amplifier is a Miteq LNA with an operation range of 0.5–18 GHz and a 2.5 dB noise floor.

the margins on clock amplitude, input phase, and relative clock phase are independent. To find the margins on amplitude, I first turned the amplitude down to near failure. I then changed the clock phase up and down to find preliminary margins on the phase. The clock input phase could be searched easily by introducing a small frequency difference between the two clock generators. At the middle of the phase range I then turned the clock amplitude down until I found the lower bound of the clock amplitude. At the same input phase, I next increased the amplitude until failure. The optimum amplitude is assumed to be the geometric mean between clock amplitudes. To establish the optimum relative clock phase, I set the amplitude at its optimal value and then decreased the relative phase until failure, increased it until failure, and chose the mean value as the optimal operating value.

Table 5.1 shows the operating conditions I found for the circuits' dc bias currents it tests on the chip. The nominal operating point was reasonably close to the design values, indicating good correspondence between designed and as-built parameters. To ensure margins on the bias currents were adequate, I checked that each was close to midway between the maximum and minimum limit currents while the circuit was at its optimal operating point.

Each of the three experiments allowed me to test the timing behavior as a function of phase delay between clock and data input. To take data, I initially reduced the clock phase until failure occurred, even with the clock amplitude at the maximum value. Then I increased the clock phase in small increments. At each increment, I measured the high and low values of clock amplitude (in dBm). Failure on the high end was clear to observe — all digital output read "one." At

Table 5.1: Operational bias conditions for N22TE. The nominal values are close to the design values. The minimum and maximum ranges show that the circuit is operating without major fabrication issues. The $a0$ input voltage range is approximate. The DC Data Offset value is dependent on the input data pattern; the value here is for a pseudo-random data pattern.

Input	Nominal	Minimum	Maximum	Design
Amplifier DC Source	175 µA	100 µA	226 µA	168 µA
DC Data Offset	80 µA	27 µA	187 µA	(n/a)
$dc0$	2 mA	0 mA	4 mA	2 mA
$a0$ (with 40 dB attenuator)	$1\,V_{p-p}$	$0.5\,V_{p-p}$	$1.75\,V_{p-p}$	$1\,V_{p-p}$

low amplitudes, the circuits start to produce random errors that result in gradually decreasing voltage levels in sampling measurements on the oscilloscope and flickering measurements for non-sampling measurements. Failure was defined as the clock amplitude at which an increase of +0.1 dBm input to the clock power produced no change in the output signal. Under this condition, the logical operation of the circuit remains the same with a further increase in clock power, i.e. the circuit is operating correctly. When the change in clock power produces a change in the output signal, some part of the circuit is not operating fully correctly.

5.4 Data and Analysis

For the time difference measurements, I define the delay as the difference in output time Δt between the midpoints of two output pulses as measured by the user-adjustable markers on the oscilloscope. I also checked the output for feedthrough or line-to-line coupling by turning off the output bias current. To determine coupling between output and clock lines I turned the bias on and off to observe the change in output voltages.

5.4.1 Experiment 1 Results — Two-output Race Circuit

Figure 5.8 shows the main experimental results from Experiment 1 on the two-output race circuit. The measured timing difference (filled circles) are plotted as a function of the input phase offset. The error bars correspond to the resolution of the oscilloscope. The first thing to notice is that the circuit works over almost the entire clock cycle. This is a surprise because the analytical model given by (3.8) and the VHDL model define operating regions that start from the beginning of the clock window, which is one-third of the clock period or a phase. One implication of this plot is that positive SFQ pulses arriving during negative clock bias wait at the JTL until the bias current reaches the critical value for propagation. In this circuit the delay of the short JTL is approximately constant, whereas the delay of the long JTL is far more dependent on the input clock phase. This suggests that the slight variations in the phase delay below 2.5 rad are due to the long line. The small maximum near 2.5 rad corresponds to $\theta = 0$ input phase relative to the clock. For larger offsets (beyond 2.5 rad) the behavior follows the predictions of (3.8) (see red curves shown in Fig. 5.4). The timing was fairly even until the largest phase offsets, where the timing difference quickly increased before failing.

For the first experiment, I used the oscilloscope to find the output times of two pulses. The delay of the equipment (probe, wires, etc.) was unknown but assumed to be constant. (They need not be the same for both pulses because any constant

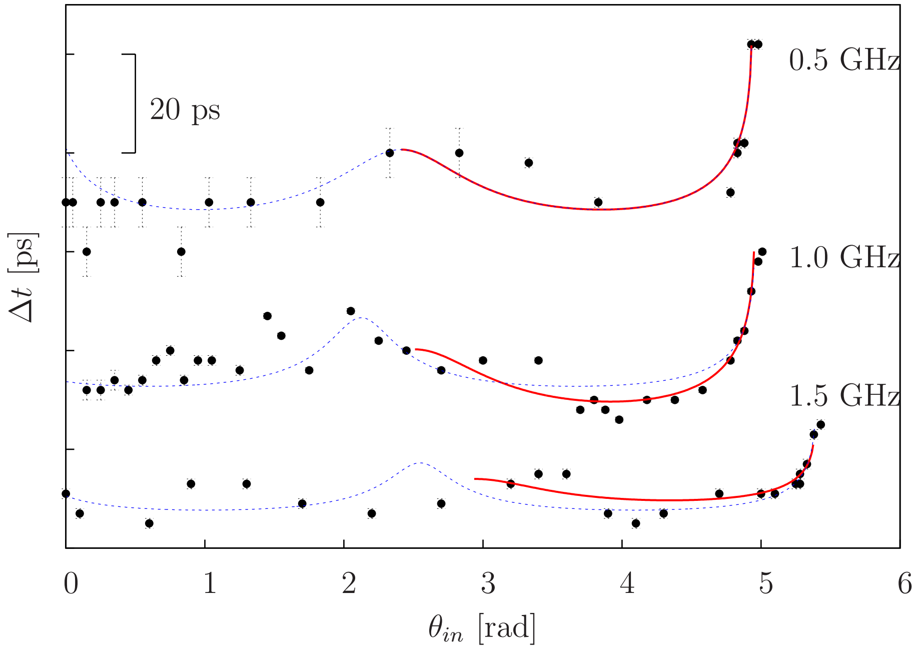


Figure 5.8: Two-output race circuit measured data. Measured timing difference Δt between outputs from short and long JTL paths for several different frequencies plotted as a function of input phase θ_{in}. θ_{in} is relative to the first recorded data point. The blue (dashed) curve is a fit of (5.4a) to all data points. The red (solid) curve is a fit of (5.4a) to the points at large phase offset where (3.8) is valid. The range of phase delays that result in SFQ propagation were considerably larger than predicted by (5.3c). The sharp upturn of the phase delay close to failure was a prediction of (5.3c) that was verified here. Figures are offset for clarity. The delay includes a contribution from the delay of the coaxial lines in the probe, making only a relative measurement of the timing difference possible.

term will produce an offset but otherwise not affect the results.) The expected relationship for the timing output of a pulse is given by (5.1). Since I was looking for the timing difference, the results were expected to be of the form

$$\begin{aligned}\Delta\nu(\theta) &= \nu_{long}(\theta) - \nu_{short}(\theta) \\ &= \arccos(\cos\theta - N_{long}\delta) - \arccos(\cos\theta - N_{short}\delta). \end{aligned} \tag{5.4a}$$

In practice, I tried fitting to a function of the form

$$\begin{aligned}\Delta\nu(\theta) = \gamma_4\gamma_1[&\arccos(\cos(\theta+\gamma_2) - N_{long}\,\delta/\gamma_1) - \\ &\arccos(\cos(\theta+\gamma_2) - N_{short}\,\delta/\gamma_1)] + \gamma_3, \end{aligned} \tag{5.4b}$$

where $\delta = 3\omega t_0/A$ (as defined in Chapter 3), N_{short} and N_{long} are the number of junctions in the short and long paths, respectively, and γ_1, γ_2, γ_3, and γ_4 are fitting parameters. Ideally I expected $\gamma_1 = \gamma_4 = 1$; γ_2 and γ_3 merely allow for the unknown offset due to the unknown length of the electrical lines. The parameters of particular interest are γ_1 and γ_4, which are related to the curvature and stretching of the curve. These factors are similar to those found in Chapter 3. Although I will shortly show two ways in which the clock amplitude could be determined, for this experiment the clock amplitude was still an unknown but it was not critical to the measurements at hand. Likewise, as I discovered in Chapter 3, the effective value of δ could change in real circuits due to leakage currents from adjacent JTLs and gates. The parameters γ_2 and γ_3 have no restrictions and account for the systematic uncertainty in the absolute phase and delay, respectively.

I plotted two different fits in Fig. 5.8 corresponding to (5.4b) with different fitting ranges. The red (solid) curve is analogous to the timing difference curve

Table 5.2: Fitting parameters of two-output circuit data. γ_1 and γ_4 were expected to be close to unity, but are still within the ranges of analogous parameters found by fitting simulations in Chapter 3. Compare to Table B.1. γ_2 and γ_3 merely account for systematic constant phase offset uncertainty in the experimental setup and carry no particular significance.

		0.5 GHz	1.0 GHz	1.5 GHz
Red Fit	γ_1	1.0635	0.6677	1.5924
	γ_2	-2.5064	-2.4101	-2.9371
	γ_3	37.4684	67.2539	49.6685
	γ_4	0.5821	0.8502	0.2378
Blue Fit	γ_1	5.2273	0.6677	8.4889
	γ_2	-2.1259	-2.4101	-2.5474
	γ_3	48.1487	67.2539	49.6719
	γ_4	5.3251	0.8502	4.078

shown in Fig. 5.4, which starts at $\theta = 0$. The blue (dashed) curve fits the same function to the entire data range. As one can see in Fig. 5.8 there is a very good fit to the experimental data. The fitting parameters are given in Table 5.2. Both fits are similar to each other and closely follow the measured data. In both cases the abrupt increase in timing difference at large phase delays was captured by the function. This qualitatively confirms that the timing behavior is following (5.4b) but not necessarily (5.4a). The values of the parameters γ_1 and γ_4 for the red fitting are closer to unity than those for the blue fit. This is not unexpected as the model was derived only for $\theta > 0$, whereas the blue curve is fit to data for which $\theta < 0$ as well.

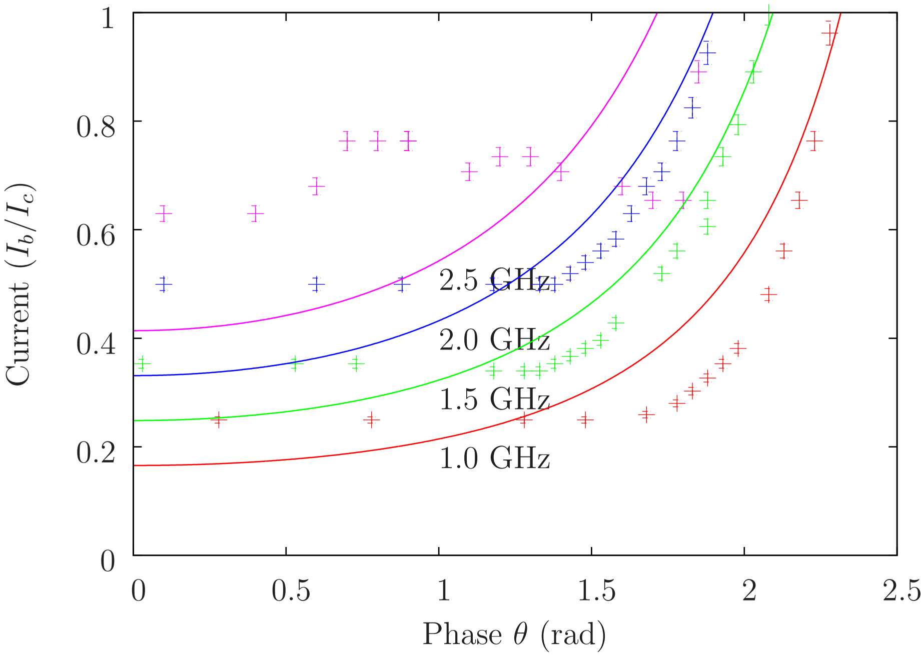


Figure 5.9: And-output race circuit data. This figure shows the data from Experiment 2 and predictions for the boundary based on (5.3c). Four frequencies between 1.0 GHz and 2.5 GHz are shown. Data between 1.0 GHz and 2.0 GHz follows the correct qualitative behavior. The data matches very closely f or the current limits and somewhat closely for the current limits.

5.4.2 Experiment 2 Results — And-output Race Circuit

Figure 5.9 shows the main results of the And-output race circuit experiment. The purpose of this experiment was to measure the operational margins of an RQL circuit as a function of clock frequency, amplitude, and input phase. In Fig. 5.9 I have plotted input phase θ versus the lower bound of bias current I_b. The points are measured data, and the curves are predictions based on (3.12) (see pg. 172). This plot can be compared to Fig. 5.5. The data in Fig. 5.9 is shown for four frequencies (1.0 GHz, 1.5 GHz, 2.0 GHz, 2.5 GHz). The limiting factor on measurements was the step size of the power of the clock signal generator. With a step size of 0.1 dBm, the error on bias current depends on the bias current $A = I_b/I_c$ and goes as $\Delta A = A \log 10/200$. The main observation is that for the 1.0, 1.5, and 2.0 GHz data, the phase limit is found near the expected value (as the solid curves) from (5.3b). However, the measured lower current bound occurs at a higher value than expected. As a result, there is less dependence of the current on clock phase at the lower measured limits of clock amplitude. Apart from from this, the observed behavior qualitatively matches the predictions shown in Fig. 5.5.

The cutoff input phase for the three lower frequencies 1.0, 1.5, and 2.0 GHz matches data to within 5% when one compares the value of the predictions at $I_b/I_c = 1$ and the data point with the highest phase θ. However, the cutoff lower clock amplitude is notably above the values predicted. The data for these three frequencies shows a specific cutoff for the lower clock amplitude: about 0.25 for 1 GHz instead of 0.175, about 0.35 for 1.5 GHz instead of 0.25, and about 0.5 for 2.0 GHz instead of 0.35. Instead of a gradual decrease in minimum current, it seems a different effect may take over the lower clock amplitude limit.

In the model, at the lowest clock bias values the junctions just barely manage to switch in time. The model considers only single junctions in isolation. In real circuits, junctions are coupled to each other through inductors. In particular, at phase boundaries, junctions on one side of the phase boundary will influence junctions on the other side (this can be seen in Fig. 1.12). Although the distribution of current is set by the choice of inductance values in JTL units on phase boundaries, this additional coupling is still present, adding an additional resistance to switching of the last junction on a phase. The result is that additional "torque" needs to be applied to the junction in order for it to switch. However, for later input phases, the bias current on the next phase will already be higher, thus making it easier for junctions on both sides of the phase boundary to switch. Hence, the effect goes away after a certain input phase.

For this experiment, I had to determine the bias $A = I_b/I_c$ from the measured clock output. The mean power in dBm for a given measured peak-to-peak voltage V_{p-p} can be found from

$$P_{dBm} = 10 \log_{10} \left(\frac{V_{p-p}^2}{8 \times 50\,\Omega} \frac{1000}{\mathrm{W}} \right), \tag{5.5}$$

where the factor of 8 is necessary to convert the peak-to-peak voltage into a root-mean-squared value, $50\,\Omega$ is the internal impedance of the oscilloscope, and the factor $1000/W$ defines the dBm unit. The test setup (see Fig. 5.7) was symmetric between input and output with the exception of the power splitter on the clock output (which had a loss of approximately 6 dB) and any attenuators placed on the probe. If we account for all power attenuators on the clock lines with a factor $P_{attenuator}$, the power delivered to the circuit on-chip P_{chip} is related to the input power P_{in} and output power P_{out} at the oscilloscope by

$$P_{chip} = \frac{1}{2}(P_{in} - P_{attenuator} + P_{out}), \tag{5.6}$$

where all values are in dBm. (The factor of 1/2 is due to the square root in the geometric mean when going from linear to log scale.) We can then express the current I_b induced in a the bias inductor as

$$I_b = \frac{M}{L}\sqrt{\frac{2P_{chip}}{Z_{clock}}} = \frac{M}{L}\sqrt{\frac{2}{1000 \times Z_{clock}} 10^{(P_{in} - P_{attenuator} + P_{out})/20}}, \tag{5.7}$$

Table 5.3: Summary of measurements of P_{in} and V_{p-p} in N22TE used for calibration of the junction bias current I_b. $P_{attenuator}$ is the amount of attenuation found between the clock signal generator and the chip that is not found between the chip and the oscilloscope. P_{out} is calculated from V_{p-p} by (5.5). P_{chip} is calculated by (5.6), and given in both dBm and mW. I_b (total) is calculated using (5.7). I_b (JJ1) and I_b (JJ2) are calculated as fractions of I_b (total) based on the distribution of supercurrents through two parallel inductances (see Fig. 2.2). The attenuator shown in Fig. 5.7 was measured to have an attenuation of 2.83 dB. Calibration of the circuit to a known current of $I_b = I_c$ gives an additional factor of 1.75 (or 2.43 dB) to current due to attenuation.

f (GHz)	1.0	1.5	2.0	2.5	2.5	GHz
P_{in}	3.4	5.1	5.3	5.5	5.7	dBm
V_{p-p}	81.1	84.9	82.2	74.6	76.2	mV
$P_{attenuator}$	5.26	5.26	5.26	5.26	5.26	dBm
P_{out}	-17.840	-17.442	-17.723	-18.566	-18.382	dBm
P_{chip}	-9.850	-8.801	-8.842	-9.163	-8.971	dBm
P_{chip}	0.104	0.132	0.131	0.121	0.127	mW
I_b (total)	0.221	0.250	0.248	0.239	0.245	mA
I_b (JJ1)	91.072	102.761	102.285	98.570	100.775	µA
I_b (JJ2)	130.103	146.802	146.122	140.815	143.965	µA

where M is the mutual inductance of the RQL clock-line transformer, L is the bias inductor, and Z_{clock} is the clock line impedance (by design about $32\,\Omega$). Equation (5.7) can be checked at a known reference point where $I_b = I_c$ and the actual attenuation can be found.

Equation (5.7) was fundamental to the second experiment because I needed to find the input phase at known clock amplitudes. The highest working clock amplitude was very consistent and represented a failure mechanism independent of the input phase. This power set the baseline for the critical current though the junction. Using this as a reference point, I then found the current through the junction from

$$I_b = I_c \left(10^{\Delta P/10}\right)^{1/2} \tag{5.8}$$

with $\Delta P = P_{high} - P_{low}$, the difference in measured values for the high and low clock power values, respectively. This method avoided the effects of uncertainties in measurements of V_{p-p}, as well as the effect of losses in the probe, losses in devices, and the uncertainties in values of mutual and linear inductances. Furthermore, (5.8) and (5.7) give independent estimate for the critical current, and these values matched within the uncertainty of the attenuation of the clock splitter, which is about 12%. Table 5.3 shows results from this analysis of the data.

The main result is that calibration suggests a loss of 2.43 dB in power before reaching the chip. The measured attenuation of the 3 dB attenuator was actually 2.83 dB, which is within the tolerances of the device. The average currents I_b (JJ1) and I_b (JJ2) should be as close to their design values of 100 μA and 141 μA as possible. This occurs for $P_{attenuator} = 5.26$, leaving a remaining 2.43 in unaccounted-for attenuation. This extra attenuation is likely found in the variable delays $\tilde{\rho}_1$ and $\tilde{\rho}_2$.

In conclusion, the results from the and-output race circuit experiment qualitatively matched the predictions of the analytic timing model. In principle, there were clear cutoff values for both the clock current amplitude and the input phase and the margins shrank as the frequency was increased. The cutoff values approximately matched with the predictions from (5.1). However, whereas the predictions were for gradual transitions from one limit to the other, the measured data had a

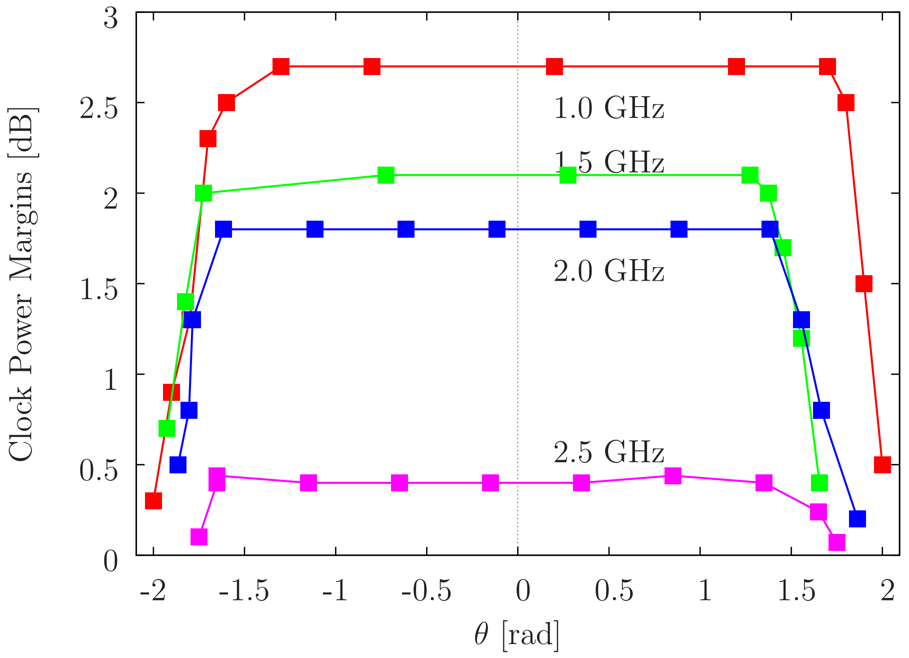


Figure 5.10: Multi-phase shift register clock amplitude margins plotted versus input phase for four frequencies. The data has been centered at zero phase. The sharp drop in margins at either end confirmed the expected behavior of a long, deep pipeline shift register. For $|\theta| < 1.5$ rad the clock margins generally show less than 0.2 dBm variation. Maximum clock amplitude was 4.5 dBm (measured) for all frequencies.

more abrupt and earlier cutoff. Also, at a clock frequency of 2.5 GHz, the data did not fit the predictions. Nevertheless, there was still an increasing minimum clock amplitude with increasing frequency.

5.4.3 Experiment 3 Results — Long, Deep Pipeline Shift Register

Figure 5.10 shows my main results from tests of the long, deep pipeline multi-phase shift register. In this figure I plot the clock margins versus the input phase. The power margins show a small variation over a range of ±1.5 radians. The rapid decline in clock margins at either extreme indicates rapid failure of the shift register. These wide margins confirm the main results of Experiment 1, in which I found the pulse propagation window extended well beyond that assumed by the model. I note also that as the clock frequency increased the nominal clock power margins decrease.

The primary goal of this experiment was to measure the junctions switching time parameter t_0. At failure, $\delta = \sqrt{2}$ and using the definition of δ we can express the baseline switching time as

$$\mathrm{t}_0 = \frac{A}{3\sqrt{2}\pi f N}, \tag{5.9}$$

where $N = 40$ is the number of junctions in each clock phase of the long, deep pipeline shift register in N22TE and A is the experimentally determined clock current amplitude in units of I_c. Table 5.4 shows the results of my analysis of the data, using the nominal clock power measurements shown in Fig. 5.10 and taking into account the calibration from Experiment 2. The average value for t_0 is 0.44 ps, with 20% variation between measurements at different frequencies. This compares well with the value of 0.44 ps expected from the I_cR_N product 0.75 mV and application of (3.7). Equivalently, the equivalent I_cR_N value from $\mathrm{t}_0 = 0.44$ ps is 0.77 mV, close to the 0.75 mV assumed in design.

A secondary goal in testing the multi-phase shift register was to observe if there was any shift in the output timing as a function of input phase. No such shift was observed. Only at the very ends of each range was there a noticeable change in the power margins. Thus, these results showed that long pipelines with "slow" clocks could adequately propagate SFQ pulses. Similarly, from the 2.5 GHz

Table 5.4: Analysis of the long, deep shift register from N22TE. Estimated parameters for results of the junction switching time of the long, deep pipeline shift register experiment. f is the clock frequency; P_{in} is the displayed value of power on the clock generator; P_{out} is the measured output from the chip; P_{chip} is the *calculated* power delivered to the chip within the $\|\theta\| < 1.5$ rad limit; I_b/I_c is the minimum bias current calculated from P_{chip} using (5.8); t_0 is calculated using (5.9); and I_cR_N is calculated using (3.7). The average value of t_0 is 0.440 ps. Average value of I_cR_N is 0.770 mV. Both values have an uncertainty of about 20%. The value of I_cR_N used for design was assumed to be 0.75 mV.

f	1.0	1.5	2.0	2.5	GHz
P_{in}	4.500	4.500	4.500	4.500	dBm
P_{out}	1.800	2.400	2.700	4.100	dBm
P_{chip}	2.700	2.100	1.800	0.400	dB
I_b/I_c	0.307	0.352	0.378	0.521	
t_0	0.576	0.441	0.354	0.391	ps
I_cR_N	0.570	0.744	0.926	0.839	mV

maximum operating frequency measured for 40 junctions per phase, I can conclude that an RQL circuit with 2 junctions per phase will operate at 50 GHz.

5.5 Conclusions

In this Chapter I described three experiments to test different aspects of the analytic timing model. With Experiment 1, I examined the applicability of the model to actual junction switching events. With Experiment 2, I showed that the experimental limits of RQL circuits mainly follow the pattern predicted by the analytic model. A significant difference was that the data showed an earlier clock amplitude cutoff, implying that margins decrease in fabricated circuits. In Chapter 6, I examine further the issue of suppressed operating margins . Finally, in Experiment 3 I showed that the long, deep pipeline behavior of RQL circuits matches the timing behavior predicted by the analytic model fairly well. I note that an alternate analysis in Appendix D shows that a suppressed I_cR_N, or an increased β_c, explains the upper experimental limit of 2.5 GHz instead of the designed 5 GHz.

Chapter 6

Carry-Look Ahead Adder Experiment

6.1 Introduction

In Chapter 5, I described results from three experiments which showed the applicability of the timing model to simple test RQL circuits. What remains is to design and test more complex RQL circuits that could be used for practical applications. In this Chapter, I first describe the design of a digital adder. This circuit is powered by a Wilkinson power splitter, contains non-local interconnects, and has a fan-out of four in the chosen architecture. I then describe experiments I performed to verify operation of the device. Figure 6.1 shows a microphotograph of the completed circuit, which I designate *Norwalk 21 Carry Look Ahead*, or N21CLA.

6.2 Circuit Design

A Carry-Look Ahead (CLA) adder adds numbers with a minimum latency. I chose to build and test a CLA for several reasons. When built in RQL, an 8-bit CLA contains 815 junctions with non-local interconnects between physically separated logic gates. This means it is a fairly complex circuit that will provide a fairly hard operational test for RQL. Also, the effective fanout was four as in a CMOS design. This was achieved through the use of amplification junctions. These features would

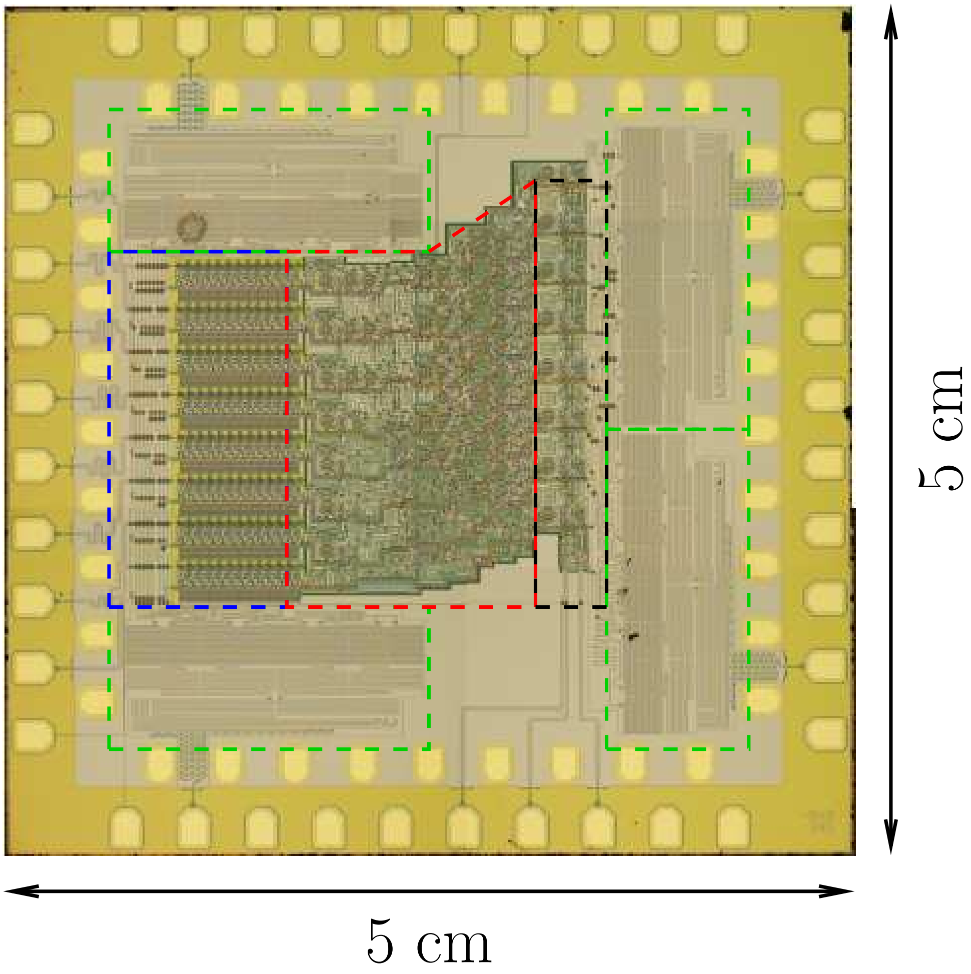


Figure 6.1: Photo of N21CLA. Four power splitters (green boxes) were in the four corners of the chip. Data was input from the bottom and moved into a shift register (black box) on the middle-right. The CLA (red box) pipeline moved data from right to left. The density of circuit elements decreased towards the right. Eight high-fidelity output amplifiers (blue box) on the left amplified the output signal to measurable levels with a very low bit-error rate. Output was on the left side. (Chips were mirror imaged from design by the fabrication process.)

be essential in any real-world application. Many different CLA architectures exist in CMOS design. I chose to use the Kogge-Stone architecture [44]. Implementing this architecture in RQL required me to use the VHDL models of Chapter 3. Needless to say, correct operation of the CLA depends on getting all the details right. For example, correct synchronization of clock signals is necessary and thus provided a secondary test of the Wilkinson power splitter.

6.2.1 Propagate/Generate Logic

As noted above, I chose to build a Carry-Look Ahead (CLA) adder with a Kogge-Stone architecture for our digital adder demonstration. The CLA is based on the AndOr operation, making it particularly well suited to a demonstration of RQL. The CLA reduces latency (and size) by pre-calculating the *propagate* and *generate* signals of each bit being added. The propagate signal P is the OR or XOR operation between the bits, while the generate signal G is the AND operation, as follows:

$$P_i = A_i \oplus B_i, \tag{6.1}$$

$$G_i = A_i \cdot B_i. \tag{6.2}$$

Both these operations can be performed in one quarter of a clock cycle by the RQL AndOr gate, including use of JTLs to increase the fanout to four. The carry bit C_{i+1} from any bit-level i is given by

$$C_{i+1} = G_i + (P_i \cdot C_i), \tag{6.3}$$

where A_i and B_i are the i-th bits of the two binary numbers being added.

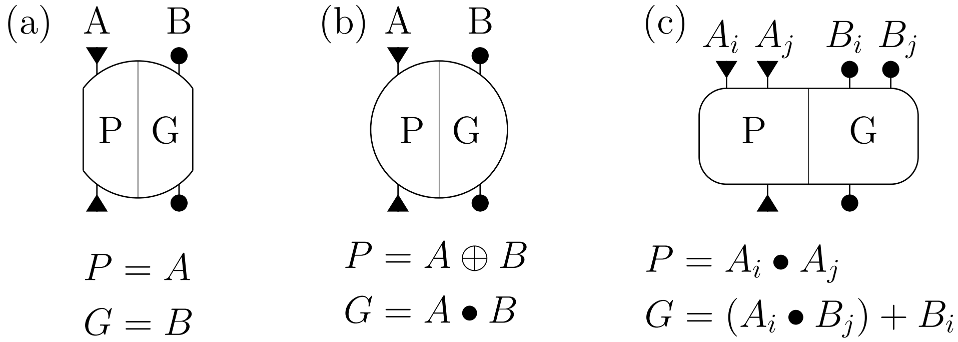


Figure 6.2: Carry-Look Ahead elements. The Kogge-Stone Adder architecture uses these three logical elements. Logical operations are shown below symbols. Note that each unit contains both a pipe for propagate (P) and generate (G) signals. (a) Simple delay unit. Corresponds to JTL units in RQL. (b) PG generation unit. This unit is the first step in any CLA and it is used only once at the beginning of each pipeline, changing separate 8-bit signals into PG signals. (c) Carry-Look Ahead gate. Takes four inputs, two each for P- and G-signals, and outputs a P- and G-signal.

Figure 6.2 shows the elements of the CLA adder. Figure 6.2(a) shows a delay unit, which is simply a JTL. Figure 6.2(b) shows the PG calculation unit that is necessary at the beginning of every pipeline and corresponds to the AndOr gate. Finally, the actual summing is done by the unit in 6.2(c), which takes two inputs each for the propagate and generate signals and generates the output according to (6.1) and (6.3).

In the Carry-Look Ahead scheme the carry bit is pre-calculated ahead of the actual sum. Whether or not a given carry will propagate or generate is pre-computed instead of waiting for the carry signal from lesser bits, as in the ripple-carry adder architecture. By recursively substituting (6.3) into itself for $i \rightarrow i + 1$, one obtains the following results for the carries going into the first 3 bits. For example, for a four-bit adder,

$$C_1 = G_0 + (P_0 \cdot C_0), \tag{6.4}$$

$$C_2 = G_1 + (P_1 \cdot C_1) = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0, \tag{6.5}$$

$$C_3 = G_2 + (P_2 \cdot C_2) = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0. \tag{6.6}$$

By definition $C_0 = 0$, but (6.4)–(6.6) hold even if we allow $C_0 \neq 0$. The importance of (6.4)–(6.6) is that it shows that the final carry bit C_3 can be calculated without waiting for the calculation of the previous carry bit, as in the ripple-carry adder. In the CLA scheme addition can be broken into groups of n bits. The group-carry and group-propagate bits can be passed to the next block of n bits, in which case $C_0 \neq 0$. This branching structure gives the CLA a latency of $O(\log n)$ [44], where n is the number of bits added.

6.2.2 Kogge-Stone Architecture

The specific architecture of the CLA can be optimized for latency, size (transistor count in CMOS or junction count in RQL), or congestion [38], which is the number of crossings of data pathways [45]. The Kogge-Stone Architecture [44] has more crossings than other designs and requires more chip area, but has the fastest performance [45].

Figure 6.3 shows the generic architecture of an 8-bit Kogge-Stone Adder. The top row of operations is the pre-calculation of propagate and generate bits. The following three rows contain calculation units and delay units, each with identical latency. As the depth increases the number of crossings increases. Between the first

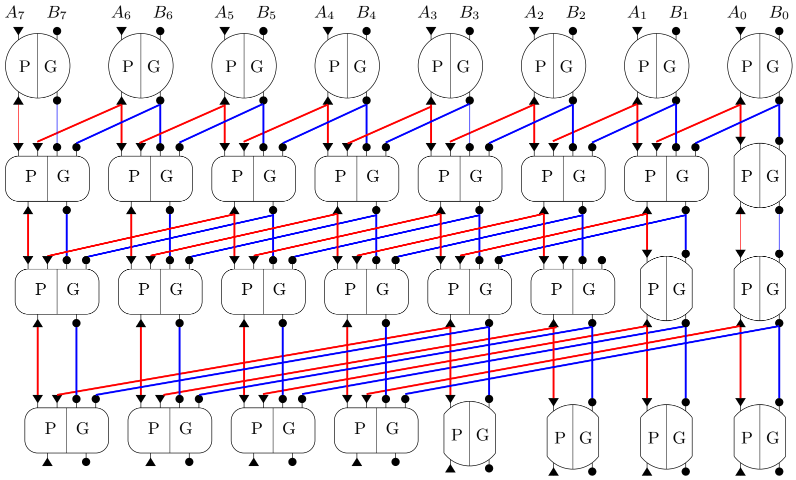


Figure 6.3: Generic Kogge-Stone CLA Architecture. Data flows from top to bottom. Most significant bit on left, least significant bit on right. Red lines indicate P-signal pipelines. Blue lines indicate G-signal pipelines. There are eight pipes and four stages. Latency of the Kogge-Stone architecture CLA is $O(\log n)$, where n is the number of bits added, here $n = 8$. The longest interconnect can be seen between step 3 and 4, where many signal lines jump by 4 pipelines.

and second rows there is one crossing bit path. Between the second and third rows there are three crossings, and between the third and fourth rows there are seven crossings. For two four-bit numbers, only the left-most four columns are needed, and the first three rows. Doubling the number of input bits requires adding only one additional row of logic gates, while doubling the number of pipes. In this design the branching structure leading to the $O(\log n)$ latency can be seen. However, the number of additional rows decreases as the number of input bits increases. Other than the crossings, columns 0–3 and 4–7 are nearly identical, with only a delay being swapped for a regular adding unit.

I made a number of refinements to the generic Kogge-Stone architecture in our circuit. I designed and synthesized an adder using the RQL gate library of Chapter 3, optimized for latency. The fan-in and fan-out of each stage of the design can be changed to increase power consumption but decrease latency. I also had a choice of the amount of congestion due to crossings. My design included non-local interconnects. Because the design did not include active transmission lines (a device which can transmit an SFQ pulse between junctions where the inductance between junctions $L > \Phi_0/I_c$), some of the longer interconnects necessitated multiple delay units (JTLs) between the logic gates, which increased the latency. Additionally, RQL circuits have specific fan-in and fan-out requirements for each gate (see Section 2.2 on pg. 42). This leads to an optimization problem between size, congestion, and power. Remarkably, the design was done by use of CMOS design tools with the necessary VHDL additions.

The overall design of the CLA is shown in Fig. 6.4. Five sequential clock phases are shown in different colors and the connections are shown between logic elements. The branching structure of the Carry-Look Ahead design is shown by the decreasing number of data paths and the greater distances crossed by the data paths on the right in the figure compared to the left. The single colored boxes are logic elements while the long, graduated-color boxes represent delays. In this design, the critical path contains a non-local interconnect between the third and fourth phases, the path connecting bit pipeline 3 to pipeline 7. This interconnect spans not only four bits but crosses nine other data paths. In this design, this connection is made by three JTL elements (six junctions) and is the longest delay in the circuit. We expect the timing constraints to be tightest on this particular pipeline. To alleviate this

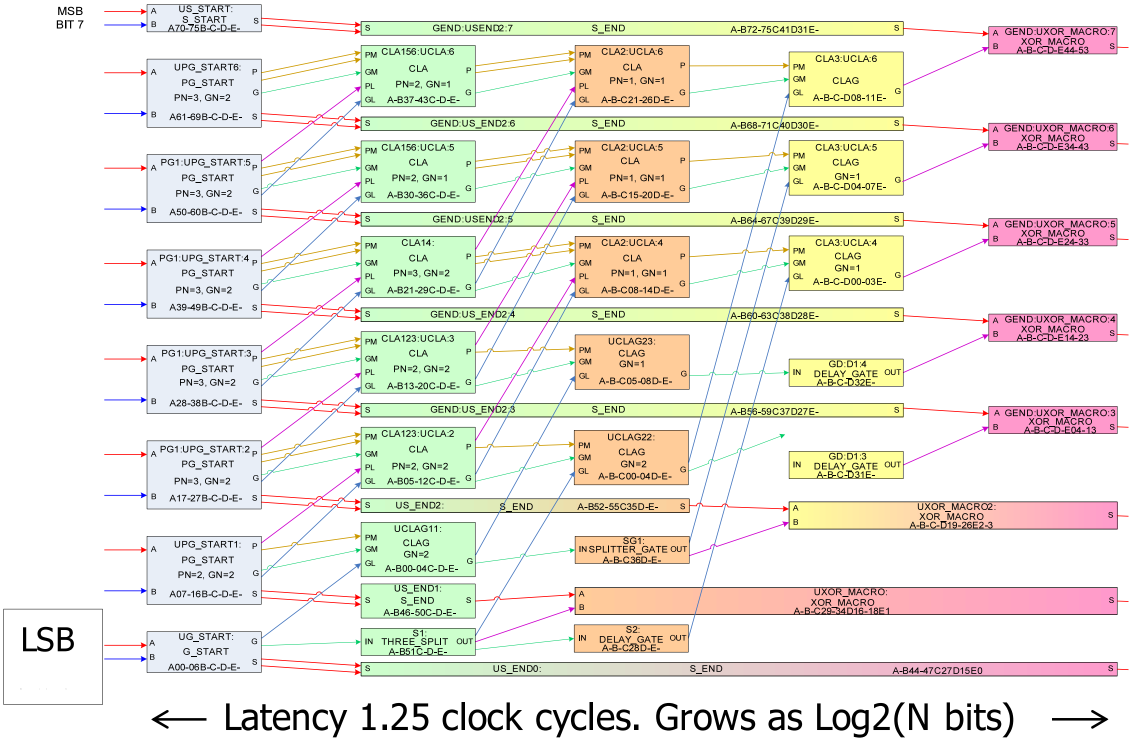


Figure 6.4: Final Carry-Look Ahead Adder design. This figure shows the synthesized layout of the CLA in schematic form, with sixteen bit pipelines and five stages. Data flows left to right, with the most-significant bit on top and least significant bit on bottom. The five clock phases are shown in color: 1 in blue, 2 in green, 3 in orange, 4 in yellow, and 5 in pink. Solid color elements are logic elements. Graduated-color elements are delays. Lines between elements are data connections. The pipeline with greatest timing constraints can be seen between phases 3 and 4, the top of three lines. This pipeline crosses nine other data paths.

issue, we added a fifth phase to the design, which increased latency but increased timing margins.

6.3 Experimental Setup

The setup of this experiment is shown in a block diagram in Fig. 6.5. This Figure is similar to Figs. 2.14 and 5.7 with only a few notable differences. There are eight outputs, labeled q0 to q7, from least-significant to most-significant bit. Instead of bias-Ts on the clock inputs and outputs, I used 3 dB attenuators in an effort to reduce reflections and standing waves on the chip. The oscilloscope has six inputs. With eight data outputs, I decided to monitor one clock output, the data return from the input, and four of the eight CLA outputs. The other clock output was terminated with a matched load, as were all CLA outputs not fed into the oscilloscope. Because of the size of the bias-Ts necessary to apply DC current to the output amplifiers, no two adjacent outputs could be observed at once. The Figure shows an example where q1, q3, q5, and q7 are monitored. Observing q0, q2, q4, and q6 was simply a matter of unplugging the bias-Ts from one port and plugging them into the others. No additional cool-down was necessary, though the DC current sources were turned off during the switch. The operating point of the N21CLA circuit was the same found for the N22TE circuit, shown in Table 5.1.

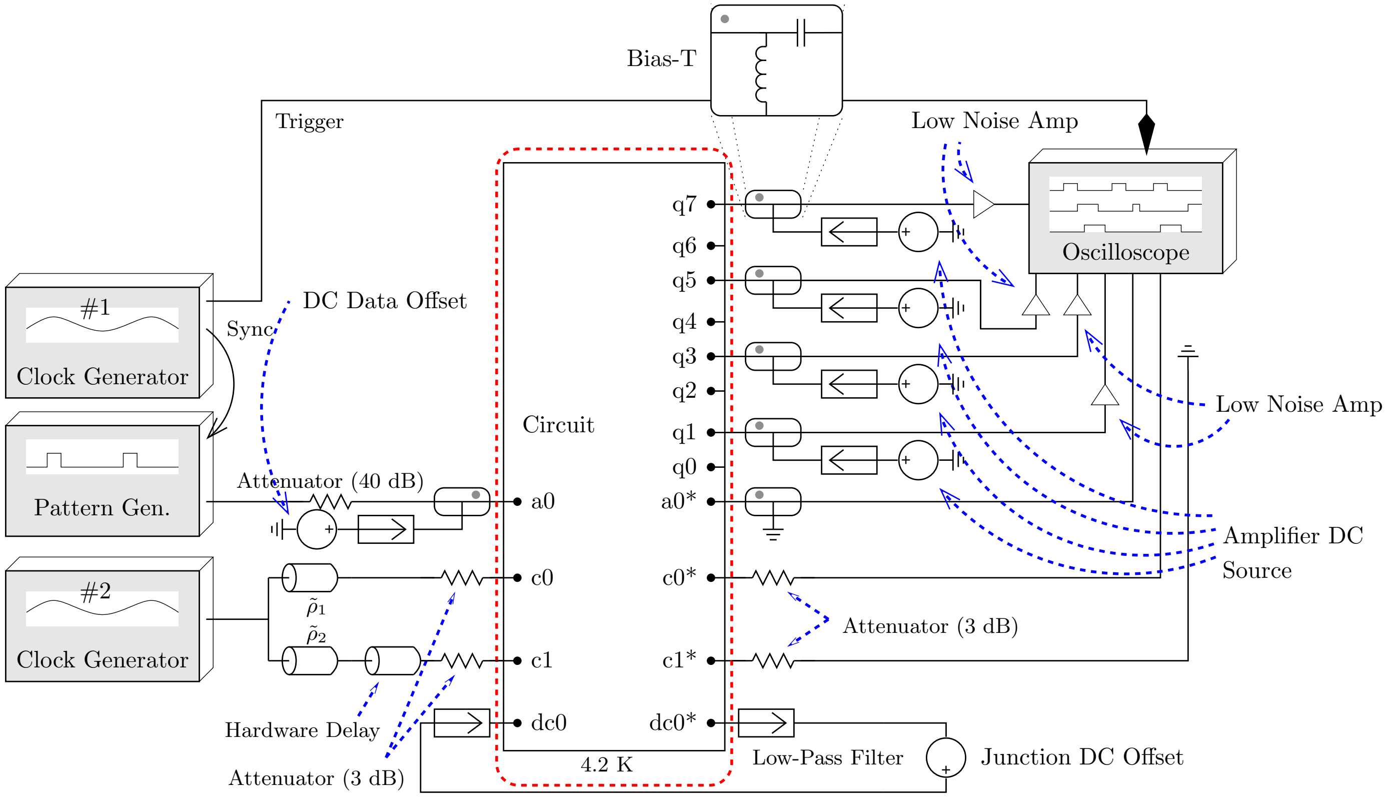


Figure 6.5: Block diagram of experimental setup for N21CLA. Similar to Fig. 5.7, the same elements present in the experiments from Chapters 2 and 5 are again found here. Block diagram of the experimental setup for the timing experiments. a0: data input; a0*: data return; c0, c0*, c1, c1*: clock phases and returns; dc0: DC offset bias; dc0*: offset bias return; q0 – q7: experimental output. Due to space limitations, only four outputs could be observed at once, and only on non-adjacent ports. Ports not used were terminated with matched impedances (not shown). Low noise bandpass filters have a cutoff frequency of $f_C = 1\,\mathrm{kHz}$. Low noise amplifier is a Miteq LNA with an operation range of 0.518 GHz and a 2.5 dB noise floor.

There were four major circuit components on the CLA chip. The Wilkinson power splitters have been discussed previously in Chapter 4. A stack of 12 SQUIDs provided amplification of each output signal from the CLA core [46]. Figure 6.6 shows a simplified version of the shift register that was used to send inputs bits to the CLA. A 16-bit serial shift register provided input to the CLA. With each clock cycle the data progressed one bit in the shift register and was also inputted to the CLA. Bits A_0–A_7 received bits 0–7 from the shift register. Bits B_0–B_7 received bits 15–8 from the shift register. Note the reverse ordering of the shift register bits for input B. In Fig. 6.6 I use blue arrows for the A input and red arrows for the B input. Any 16-bit pattern could be sent in serial form down the input to the shift register and then applied to the CLA I tested two specific patters. (See Table 6.1). A full test of the CLA would measure the error rate for many different input patterns. Overall, the clocks and single data input were arranged in the same fashion described for the shift register in Chapter 5. (See Fig. 6.5.)

6.4 Experimental Results

I first established the optimum operating point for clock power, relative clock phase, and data input phase using a procedure that was similar to the procedure in Chapter 4 for Experiments 1–3. I then tested the digital output of the circuit. Correct operation was verified by comparing the measured output to the expected output. Additionally, I measured the power margins for the CLA gates and compared them with simulation results. Finally, I measured the power dissipation of

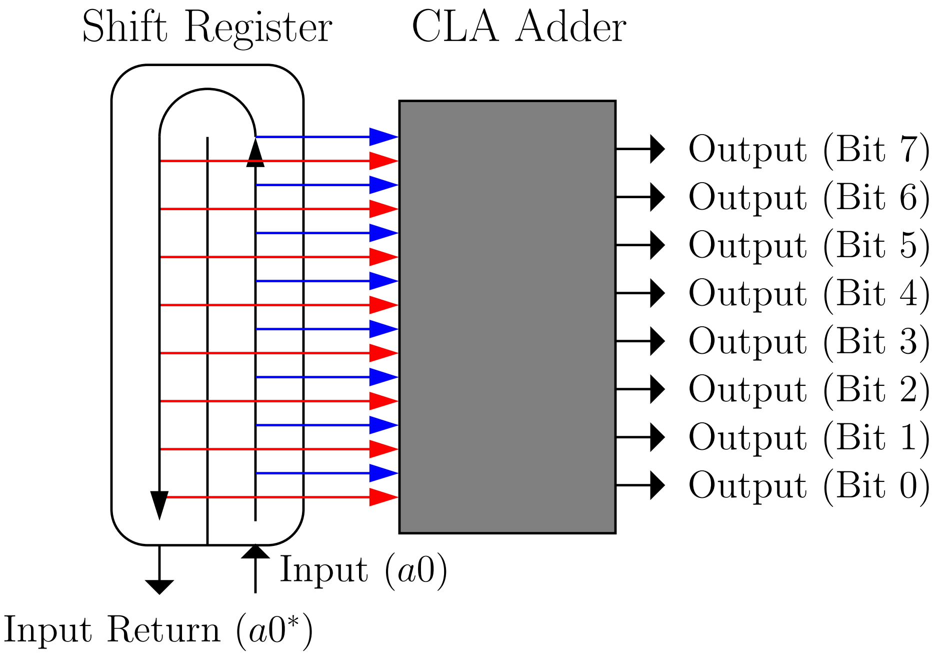


Figure 6.6: Shift register input for CLA. The sixteen input bits to the CLA are delivered by an on-chip shift register. Blue arrows represent bits of A, red arrows represent bits of B. The input to the register is on the bottom right and the output on the bottom left. This 16-bit register passes the first eight bits to the CLA in reverse order, with the first bit being the least significant bit, and passes the last eight bits to the CLA in reverse order, with the 8th bit being most significant.

the CLA core and compared it with the expected dissipation.

6.4.1 Logic Test

The input a0 to the shift register was supplied with two arbitrary patterns of 16 bits each, at a repetition rate of $1/(16f)$. Table 6.1 shows the two 16-bit input sequences I used and the expected output from the CLA. The ID number in the left column is for reference. The columns labeled q7 to q0 are the output bits, from most-significant to least-significant. The far right column lists the corresponding

Table 6.1: Expected CLA output pattern for two cyclic input sequences. This table shows the expected output of each bit of the CLA in sequence in columns. Two short, arbitrary inputs sequences were chosen. Rows constitute simultaneous output values. Base-10 decimal numbers are given in the right column for reference. ID number on left is for reference only.

	1111111111111100									1110110111111100								
ID	q7	q6	q5	q4	q3	q2	q1	q0	D_{10}	q7	q6	q5	q4	q3	q2	q1	q0	D_{10}
0	1	1	1	1	1	0	1	1	251	1	0	1	1	0	0	1	1	179
1	1	1	1	1	1	0	0	0	248	1	1	0	1	0	1	0	0	212
2	1	1	1	1	0	0	1	0	242	1	1	1	0	0	0	0	0	224
3	1	1	1	0	0	1	1	0	230	1	1	0	1	1	1	0	1	221
4	1	1	0	0	1	1	1	0	206	1	1	0	0	1	0	0	1	201
5	1	0	0	1	1	1	1	0	158	1	0	0	1	1	0	1	0	154
6	0	0	1	1	1	1	1	0	62	0	0	1	1	1	0	0	1	57
7	1	1	1	1	1	1	1	0	254	1	1	1	1	0	1	0	1	245
8	0	0	1	1	1	1	1	0	62	0	0	1	0	1	1	0	0	44
9	1	0	0	1	1	1	1	0	158	0	1	1	1	1	0	1	0	122
10	1	1	0	0	1	1	1	0	206	1	0	0	0	0	1	1	0	134
11	1	1	1	0	0	1	1	0	230	0	1	0	1	0	1	1	0	86
12	1	1	1	1	0	0	1	0	242	0	1	0	1	0	0	1	0	82
13	1	1	1	1	1	0	0	0	248	0	1	1	1	1	0	0	0	120
14	1	1	1	1	1	0	1	1	251	0	1	0	1	1	0	1	1	91
15	1	1	1	1	1	1	0	0	252	0	1	1	0	1	1	0	0	108

base-10 number associated with each 8-bit output. Each output can be found by the following procedure. Shift the input pattern to the left by the value in the ID column. The first number becomes the last with every shift. Starting from the left, the first eight numbers are binary digits B_0 through B_7, in that order. The next eight numbers are binary digits A_7 through A_0, in that order. Note that the most significant bit of both numbers appears in the middle of the pattern. Add the decimal numbers $A = \sum_{i=0}^{7} A_i 2^i$ and $B = \sum_{i=0}^{7} B_i 2^i$ to get $D = A + B$, and then take the modulo 256 to get $D_{10} = D \mod 256$, which is shown in the table. Then q7 through q0 are the digits in the binary representation of D_{10}.

This procedure will give 16 different outputs, for each of the 16 cyclic permutations of the input pattern. For example, for ID 12 from the second input pattern, $A = 11011111|_{binary} = 223|_{decimal}$ and $B = 01110011|_{binary} = 115|_{decimal}$. As shown in the table, $(A + B) \mod 256 = 82|_{decimal} = 01010010|_{binary}$.

Figure 6.7(a) shows the measured output of the CLA while operating at a clock speed of 6.21 GHz as measured by a sampling digital oscilloscope when the first 16 bit input pattern was fed to the shift register. Figure 6.7(b) shows the measured output of the CLA while operating at a clock speed of 6.21 GHz as measured by a sampling digital oscilloscope when the second 16 bit input pattern was fed to the shift register. I tested and found correct operation at 4 GHz as well. Because the Wilkinson power splitter operates better closer to the design frequency of 7.5 GHz, I will describe the results for the highest operational frequency. Additionally, I found no operating points at frequencies between 4 GHz and 6.21 GHz. An on-chip 50 MHz resonance between pads prevented operation at all but a few frequencies. No

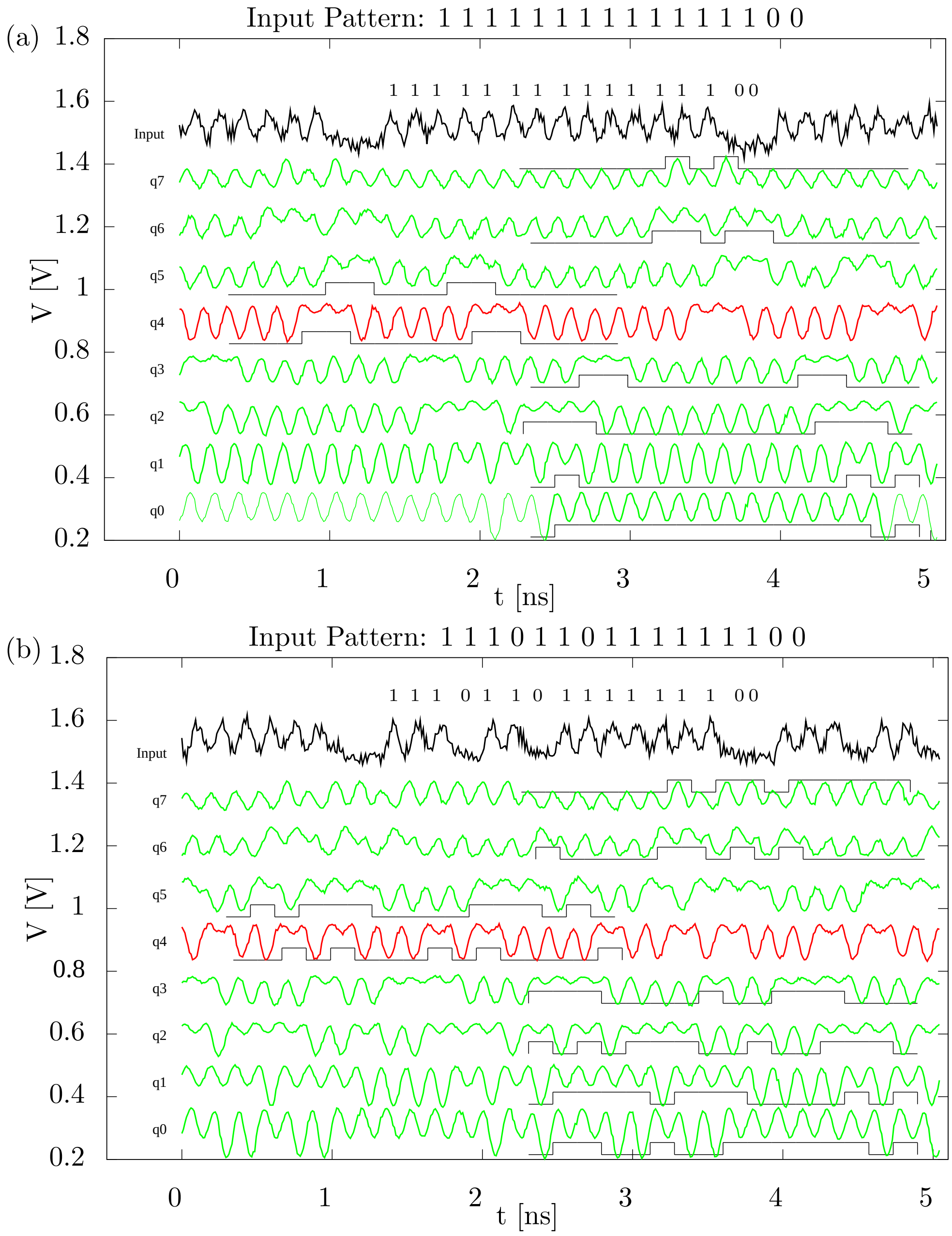


Figure 6.7: Measured CLA Output. Two examples of CLA outputs for different inputs. Output voltage levels as measured on oscilloscope from the CLA chip with wafer coordinates +9+B. Top row shows the voltage level of the return from the shift register on chip. Next rows show output from CLA outputs q7 through q0. Digital one corresponds to a low voltage, digital zero corresponds to a high voltage. Black curves show expected digital output. Data for q4 is shown in red because its correct operation was recorded during a separate cool down of the chip.

smoothing or averaging was applied. The top row is the returned input signal. The eight lower rows are, from top to bottom, the outputs from the most-significant to least-significant bits.

As in my previous experiments, outputs had inverting amplifiers which made digital “zero” appear as sustained high output voltages, and digital “one” as low-dipping voltages. The input return signal was nosier than the CLA output bit it is not flipped. An input “one” was 1 V with a 36 dB attenuator applied to the chip and returned without amplification. The eight output bits were amplified on-chip by the SQUID amplifiers and then the Miteq amplifiers at room temperature. Digital “zero” and “one” appear as peaks of different height — instead of peaks and flats — due to crosstalk between data and clock lines carrying a sinusoidal signal. Also, on this chip, the clock power was eight times greater than I used in my previous experiments, since each of the eight pipelines required a fully-powered clock line, and there are eight clock lines per Wilkinson power splitter. Also, crosstalk was more pronounced for outputs with pads physically closer to the clock signal lines. Nevertheless, the output was clearly discernible. I made note that the difference in length of coaxial cables in the probe and from probe to equipment was small, such that the CLA outputs were not skewed in time relative to each other. An additional elbow connector used for q4 and q5 introduced the delay seen in Fig. 6.7.

The measured signals (solid curves) shown in Fig. 6.7 correspond to the expected output (black curves) shown in Table 6.1. The pattern was cyclic and I found the beginning of the pattern (corresponding to ID 0) by inspection.

6.4.2 Operating Margins

Figure 6.8(a) shows the expected power margins based on a spice simulation I did of the entire CLA (see Appendix F) and Fig. 6.8(b) shows the measured power margins of the CLA. In the figure, the optimal operating point from measurements is marked for 6.21 GHz and a clock amplitude of $0.88 I_c$. This is about 8% above the 6.21 GHz design value marked in Fig. 6.8(a).

The clock power supplied to the probe was measured using a Agilent E4416A EMP-P Series Power Meter, and checked against the oscilloscope readings. The output power was measured using an oscilloscope or Hewitt Packard 70620B Spectrum Analyzer. The power delivered on-chip was then calculated as the geometric mean of the applied and measured power, or the average value in dBm. Figure 6.8(b) shows the margins and the optimal operating point for power at 6.21 GHz. The optimal power is indicated by the × and is centered between the high and low power margins. The nominal operating power of -2.4 dBm was found to correspond to the design power of -2.4 dBm to within 8%. The correct logical operation of the CLA, along with verification of the designed optimum operating point, also confirms that the Wilkinson power splitter is operating correctly.

A significant discrepancy with these results is that the predicted margins of 5.5 dB at 6.21 GHz, were much greater than the 1 dB margins I measured. Nevertheless, the measured optimal operation point of -2.4 dBm clock power was the same as the design value. The narrowing of the margins is due to microwave effects such as resonances within the power network (between pads, between splitter and combiner,

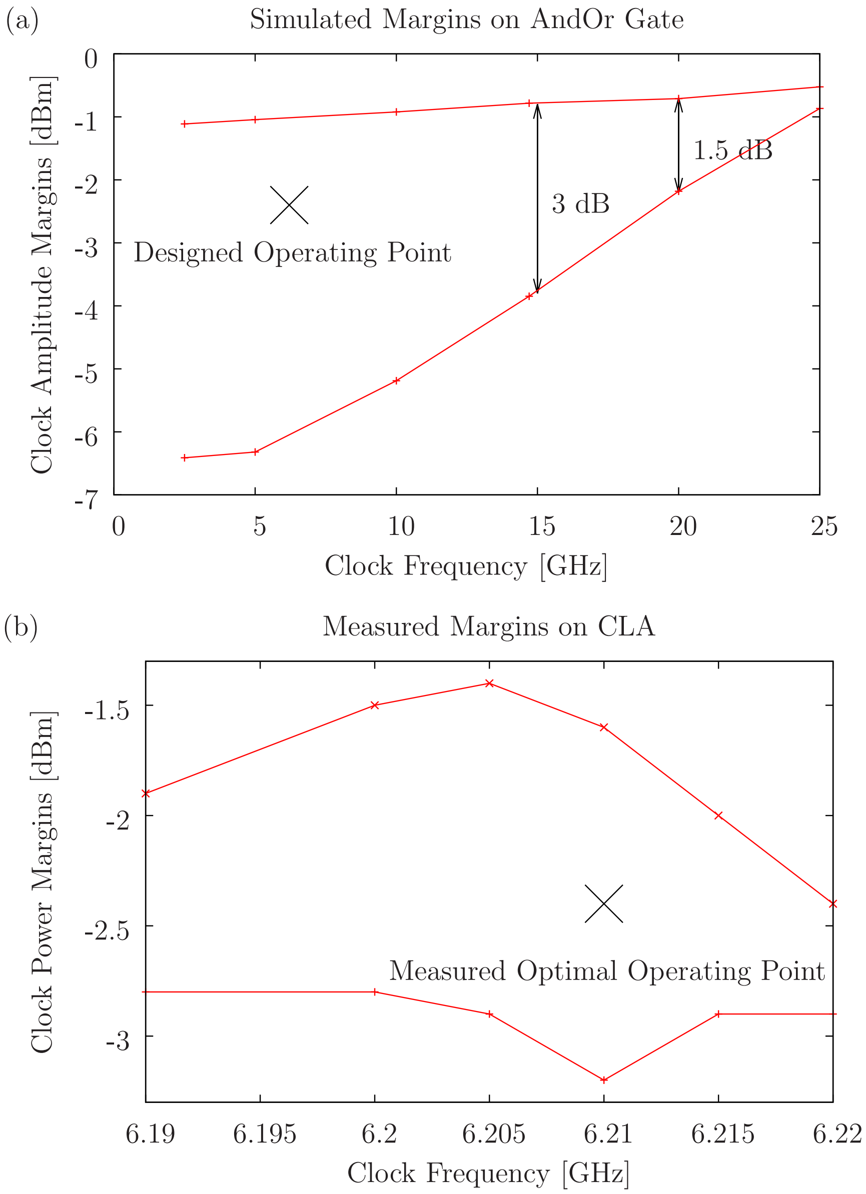


Figure 6.8: Power margins for CLA. (a) Simulation of the CLA (both powered and unpowered gate designs) shows the upper and lower power margins of the gate. The designed operating point of the AndOr gate is marked at approximately -2.4 dBm. (b) Measured power margins on the CLA in the immediate vicinity of 6.21 GHz. The optimal operating point is the geometric mean of the power to the chip at high failure and low failure. The optimal operating point is marked here at approximately -2.4 dBm.

etc.) in the power network, not operational problems in the digital parts of the circuit, since any errors in the digital part of the circuit would be immediately apparent in the incorrect digital output seen on the oscilloscope. By measuring the power margins over a range much greater than the 50 MHz shown in the Figure one should be able to determine if microwave resonances are present. I tried to find operational frequencies at 6.0 GHz, 6.1 GHz, and 6.3 GHz, none of which yielded positive results. In fact, at other clock speeds I found even smaller margins for clock power. I also found that the operating margins of clock power of the CLA fluctuated rapidly as a function of frequency and correlated with neighboring measurements only within small 50 MHz steps. This behavior is not at all like what I expected, as shown in Fig. 6.8, 50 MHz corresponds to an on-chip pad-to-pad resonance which I noted in Chapter 5. This made finding other operational frequencies difficult, but with further testing I found another good point at 4.0 GHz.

6.4.3 Power Dissipation Test

All power for the CLA came from two clock lines. The first clock line powered one phase of the CLA and the output amplifiers. The second clock powered only the other phase of the CLA. I measured the power drawn from both clocks. In Chapter 2, I showed that the amount of power drawn by a single junction in RQL was very small. For the approximately 815 junctions in the CLA, a direct measurement of the power loss on the clock lines would require measuring 1 μW or less out of 1 mW. This was not possible with available equipment. Instead, I used an alternative method of measuring power dissipation that involves measuring the clock sidebands which appear due to phase and amplitude modulation.

Figure 6.9 shows the measurement of the clock signal using a spectrum analyzer. The main peak at 6.21 GHz stands well above the noise floor. Sidebands at 6.210 258 GHz and 6.209 742 GHz can clearly be seen in both measurements. The sidebands are at ±258.75 kHz with respect to the main band, because they were generated by a cyclic 24,000-bit input sequence containing 12,000 successive "zeros" and either 12,000 successive random bits or "ones." The key thing to notice is that the sideband peaks in Fig. 6.9(a) are higher than the side band peaks in Fig. 6.9(b). This is because power was drawn from Clock 0 by the output amplifiers than from Clock 1.

The calculation of the power drawn by the CLA at the optimal operation point follows in three parts. First, using the results of Chapter 2, Section 2.6.2, I determine the relative effect of junction switching on phase and amplitude modulation. In Chapter 2, I argued that the expected delay in the clock signal due to junction switching was 1.4 ps, independent of clock frequency. Since the time period corresponding to 6.21 GHz is 161 ps, the phase modulation of the signal from one junction switch is expected to be

$$\Delta\theta = 2\pi \frac{1.4\,\mathrm{ps}}{161\,\mathrm{ps}} = 0.0546\,\mathrm{rad}. \tag{6.7}$$

Because this is the *total* variation, the expected amplitude of variation of the phase is half the value given in (6.7) is

$$g_p = \Delta\theta/2 \approx 0.0273\,\mathrm{rad}. \tag{6.8}$$

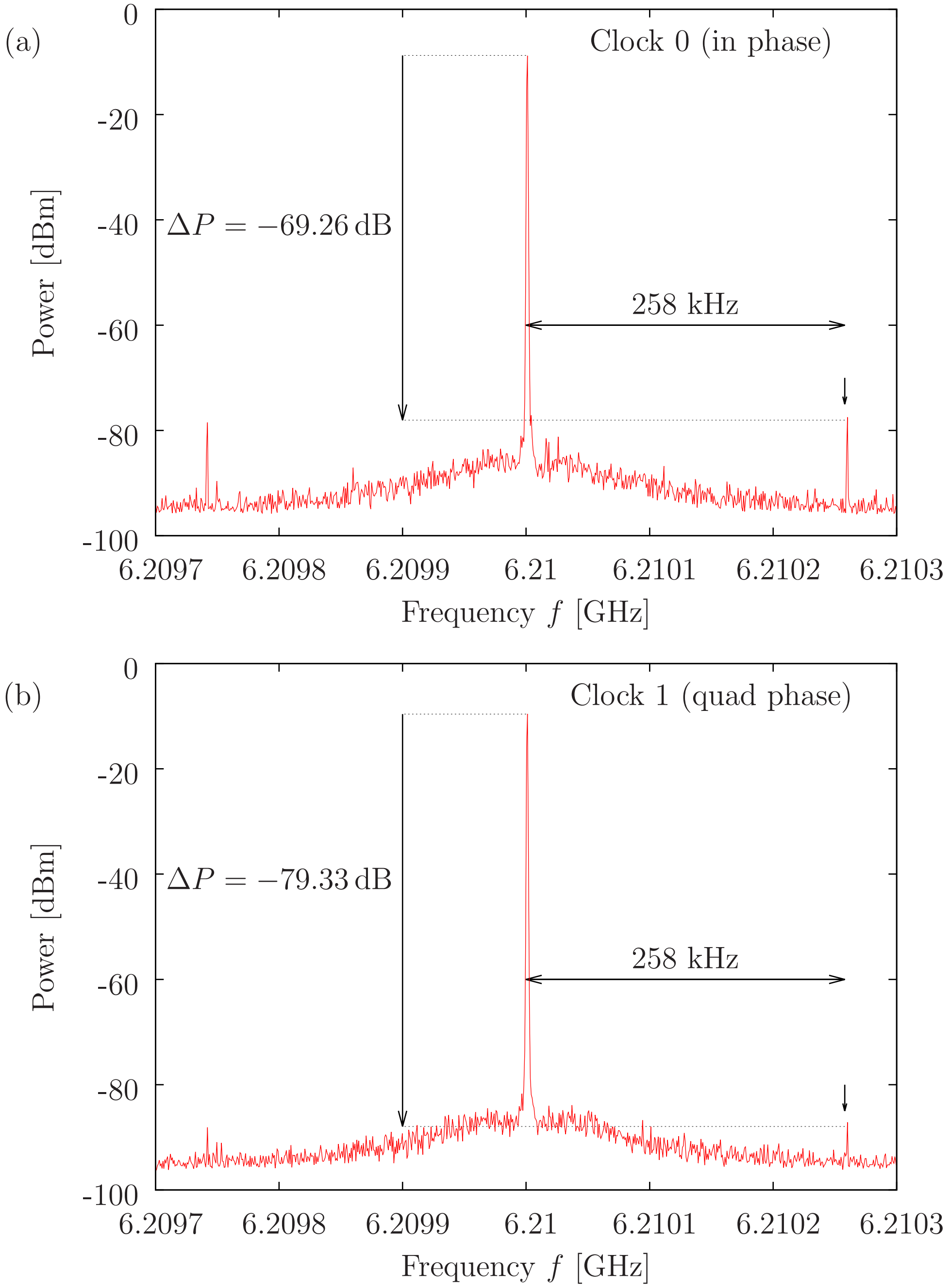


Figure 6.9: Measured power spectrum of Carry-Look Ahead Adder. This output from the Agilent spectrum analyzer displays three peaks. The center peak is at the carrier frequency (6.21 GHz) and the two sidebands are at the modulation frequency ±258 kHz from the center peak. Data input was a sequence of 48,000 non-return-to-zero bits, composed of 12,000 return-to-zero random bits followed by 12,000 return-to-zero "zero" bits. (a) shows clock phase 0, (b) shows clock phase 1.

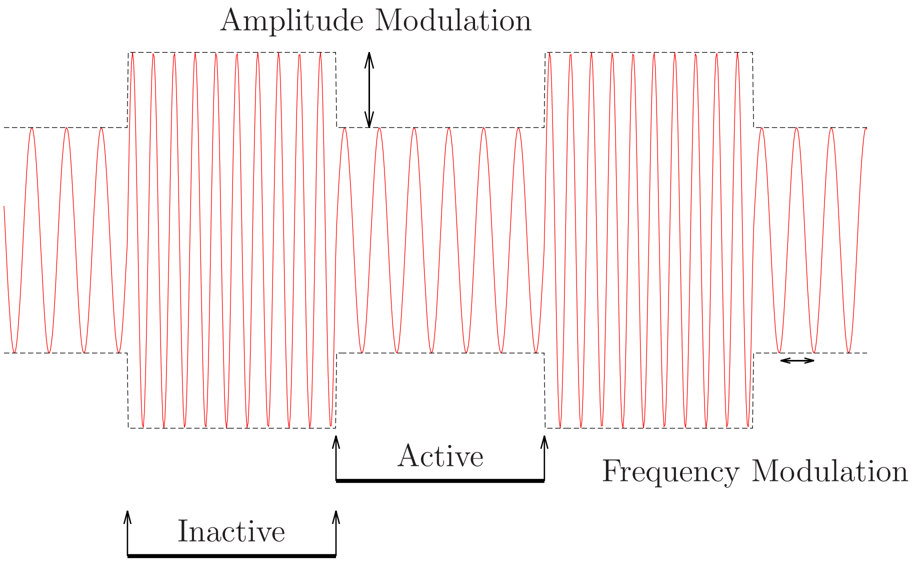


Figure 6.10: Modulation of Clock Signal by RQL Gate Operation. This figure shows a clock signal modulated by an input signal consisting of a sequence of random bits and all zeros. While the CLA is active the impedance of the clock lines changes, which results in both amplitude and frequency modulation. This modulation of the clock signal is a measure of the power drawn by the CLA while operational.

The power dissipation of the circuit is 0.6 μW (see Chapter 2, Section 2.6.2) for a total applied power of 12.5 μW [8]. Thus, the variation in power has an amplitude of

$$g_a = 1 - \frac{12.5\,\mu\text{W} - 0.6\,\mu\text{W}}{12.5\,\mu\text{W}} = 0.0243, \tag{6.9}$$

which is close to the value of the phase modulation factor g_p. Thus, the two effects are different but we expect them to be of approximately equal strength.

Figure 6.10 shows the behavior of a clock signal while the CLA was at times active and at times inactive. While active, the loading of the circuit by the switching junctions drew power (decreasing amplitude), the effective inductance of the transformers changed, and this changed the propagation time (modulating phase). This modulation can be expressed as

$$V(t) = V_0(1 + g_a \sin \omega_a t) \sin[\omega_0 t(1 + g_p \sin(\omega_p t)], \tag{6.10}$$

where V_0 is the nominal output voltage, ω_0 is the nominal clock frequency, ω_a is the amplitude modulation frequency, ω_p is the *phase* modulation frequency, g_a is the amplitude modulation factor, and g_p is the *phase* modulation factor. For our test $\omega_a \ll \omega_0$, $\omega_p \ll \omega_0$, $g_p \ll 1$, and $g_a \ll 1$.

A Fourier decomposition of (6.10) gives the main peak power P_0 at $\omega_0/2\pi$ and a sideband power ΔP below P_0 at $(\omega_0 \pm n\omega_a)/2\pi$, where in is an integer. As $|n|$ grows larger, the sidebands grow smaller. I am only interested in the first sideband; sidebands with $|n| > 1$ are below the noise floor of the spectrum analyzer. A Taylor series expansion of (6.10) will have three terms of interest. A constant value representing the main peak power at 6.21 GHz, and two terms at $\omega_0 + \omega_a$: one representing the effect of amplitude modulation and one representing the effect of phase modulation. The relative weights of each are 0.53 for g_a and 0.47 for g_p, which we can see by comparing (6.8) and (6.7).

To calculate the effect of amplitude modulation, I temporarily set $g_p = 0$ in (6.10). The relative strength of the effect of phase modulation on the clock is known from (6.7) and will be accounted for later. The voltage of the clock line as measured at the output is different while operational (on) and non-operational (off), and can be expressed as

$$V_{on} = (1+x)V_0, \tag{6.11a}$$

$$V_{off} = (1-x)V_0, \tag{6.11b}$$

where $x = (V_{on} - V_{off})/2V_0$ is the ratio of the variation with respect to the average voltage V_0 for both periods with and without data together. Since x is small, the power dissipated while the chip is on and off is,

$$P_{on} = \frac{1}{R}(1+x)^2 V_0^2 = P_0(1+2x), \tag{6.12a}$$

$$P_{off} = \frac{1}{R}(1-x)^2 V_0^2 = P_0(1-2x), \tag{6.12b}$$

where P_0 is the nominal power in the clock line. Here I have approximated $(1+x)^2 \approx 1+2x$. This gives the differential power to the chip as

$$P_{chip} = P_{on} - P_{off} = 4x\, P_0, \tag{6.13}$$

or as a ratio,

$$\frac{P_{chip}}{P_0} = 4x. \tag{6.14}$$

I now come to the key idea of the sideband measurement. The power measurement gives us the ratio of power between the main peak and the *single side band*. To use (6.14) to calculate the power dissipation of the CLA, I can recast it in terms of the measured sideband power difference shown in Fig. 6.9. In both linear and logarithmic forms, I can write the power dissipation ratio as

$$\left.\frac{P_{chip}}{P_0}\right|_{linear} = 2 \times 4 \times \sqrt{10^{\Delta P/10} \times 0.53} \times \frac{\pi}{4}, \tag{6.15a}$$

$$\left.\frac{P_{chip}}{P_0}\right|_{dB} = 9\,\mathrm{dB} + \frac{1}{2}\Delta P - 1.38\,\mathrm{dB} - 1\,\mathrm{dB}, \tag{6.15b}$$

where $\Delta P = 10 \log_{10} x$, or the ratio of P_{chip} to P_0 as given in dB, which can be seen in Fig. 6.9. The terms in (6.15b) require some explanation. In (6.15a), there is a factor of two for the two sidebands in Fig. 6.10 at $\omega_0 + \omega_a$ and $\omega_0 - \omega_a$. There is a factor of four from the derivation of (6.14). The factor of 0.53 comes from (6.7) because 53% of the variation is due to amplitude modulation and 47% due to phase modulation, whereas x is the power ratio taking both effects into account. Finally, the factor of $\pi/4$ is a correction factor due to the data being a square wave and not a sinusoid, as assumed in (6.10). This last correction is calculated as the difference in amplitude between the pure sine wave and the fundamental harmonic of a square wave. Equation (6.15b) follows directly from (6.15a), and is more useful here because the power measurements are made in dBm.

For eight $32\,\Omega$ clock lines with 2 mA AC current amplitude passing through each of the eight clock lines fed from the same input pad, the estimated power to the chip is

$$P_0 = 8 \times \left(\frac{2\,\mathrm{mA}}{\sqrt{2}}\right)^2 32\,\Omega = 512\,\mu\mathrm{W}. \qquad \text{(estimate)} \qquad (6.16)$$

Table 6.2 shows the results of the calculation of P_{chip}. ΔP is measured from Fig. 6.9. P_{chip}/P_0 is given by (6.15b). P_{in} and P_{out}, the clock line input and output power, and the value of the attenuator on the probe are measured quantities used to calculate P_0. P_{chip} is calculated from P_{chip}/P_0 from (6.15b) and P_0. Because Clock 0 powers only half the CLA and Clock 1 powers both half the CLA and the output amplifiers, I estimated the CLA power as twice that being drawn from Clock 0. In the end, this gives the estimated CLA power as 570 nW. This does not take into account power dissipated by the junctions in the shift register, nor the slightly higher number of junctions powered by Clock 1 than Clock 0. The expected power for 815 junctions with a weighted average critical current of 162 μA (using (2.4)) is $P = 1/3\, I_c \Phi_0 N f = 563\,\mathrm{nW}$ for the CLA at 6.21 GHz. The measurement and prediction match within 2%, which is remarkable given the many assumptions and corrections than needed to be made. Finally, I note that this corresponds to only 700 pW per junction, roughly a factor of 200 smaller than CMOS transistors.

6.5 Conclusions

In this Chapter I described an experiment on an RQL CLA that demonstrates that RQL circuits operate at very low power. At the time of this writing, common processors typically have 750 million transistors and use over 100 W of power, or about 130 nW/transistor. The CLA, with 570 nW power dissipation for 815 junctions, uses about 700 pW/junction, i.e. almost 200 times less power per junction than CMOS uses per transistor. An implication of this experiment is that RQL is one of the lowest-power digital technologies now known. Additionally, my experiments showed that RQL can be scaled to large, non-trivial circuits and also showed that RQL was compatible with existing CMOS design and analysis methods.

Table 6.2: Power Measurement Calculations. ΔP is measured from Fig. 6.9. P_{chip}/P_0 is calculated from (6.15b). P_{in} and P_{out} were measured separately using a power meter. P_0 is calculated from P_{in}, P_{out}, and the attenuator value (also measured separately). P_{chip} is calculated from P_{chip}/P_0 from (6.15b) and P_0. The sum of powers is calculated by adding the power dissipation from both clocks. CLA and Amp power are calculated assuming CLA drain from Clock 1 is the same as Clock 0. The predicted value of power for the CLA was 563 nW.

	Clock 0	Clock 1	
ΔP	-79.33	-69.26	dBm
P_{chip}/P_0	-33.045	-28.01	dB
P_{chip}/P_0	0.000496	0.001581	
P_{in}	1.77	1.85	dBm
P_{out}	-9.41	-8.66	dBm
Attenuator	2.83	2.83	dB
P_0	-2.405	-1.99	dBm
	0.57478	0.63241	mW
P_{chip}	0.00029	0.00100	mW
	285	1000	nW
	-35.45	-30	dBm
Sum of Both		0.0013	mW
		1285	nW
		-28.9	dBm
CLA Only		0.000570	mW
		570.2	nW
		-32.4	dBm
Amplifier Only		0.000715	mW
		714.9	nW
		-31.5	dBm

Chapter 7

Summary and Conclusions

7.1 Summary

In the opening to the first chapter, I mentioned that RQL is a superconducting technology that may eventually replace CMOS technology. It is a classical digital technology based on encoding classical digital data as decidedly quantum flux units. Demonstrated RSFQ technologies support the use of SFQ pulses to encode digital data. However, although RSFQ has many advantages, it is not applicable for many applications, nor for general purpose processing. An alternative encoding as pulse pairs gave birth to RQL (see Chapter 2).

Theoretical predictions for RQL estimated a power consumption of approximately $\Phi_0 I_c$ per junction switching event, which I confirmed in Chapter 2. This, together with the demonstration of functioning logic gates, opened up the possibility of large scale integrated RQL digital circuits. RQL logic gates have three or fewer junctions and are powered through an inductively coupled clock line, removing the power consumption from dc-biasing resistors. The gates behave as combinatorial gates on a higher level, but as state machines on a lower, pulse-based level.

In Chapter 3, I examined the detailed behavior of some RQL circuits. The analytic timing model provides a detailed description of the timing behavior of RQL junction switching events. Starting from this model, I also derived the self-correcting timing behavior of RQL circuits. Finally, the model also provides estimates on the limits of operation for the number of junctions per phase, the clock frequency, clock amplitude, and input time. More than just a useful behavioral model, this analytic timing model was the basis for a VHDL behavioral model. Together with simulation results, the VHDL model described RQL circuits in an industry-standard language, opening up the possibility of applying the large library of existing CMOS design tools to RQL.

Chapter 4 was focused on developing a suitable power network to supply bias current to the junctions in an RQL circuit. This power network needed to meet a number of goals. My primary metric was the current amplitude distribution across the clock lines. Too much or too little current at any junction would cause improper operation of the whole chip. Too much variation in the current along the clock line leads to bad timing properties or a complete failure of the circuit. Because the clock power has to be pulled back off the chip, using a limited number of pads, I designed a power splitter/combiner with minimal current variation on the chip, minimal reflection from the chip, and maximum isolation of different clock lines on the chip. This was accomplished with a modified set of cascaded Wilkinson power splitters. The design was collapsed into only six stages with a maximum flat response, and then this design was optimized using numerical simulations. The test of two power networks, a 12-stage geometric design to test even mode operation and a 6-stage maximum flat design to test the odd mode, showed acceptable margins for the RQL circuit under test.

In Chapter 5, I described three experiments to test the analytic timing model.

I directly measured the output timing difference between two pulses on Josephson transmission lines of different lengths to compare results with (3.8). I measured the clock power margins as a function of input phase, testing the predictions of the limits of operation from the analytic timing model. Finally, I measured the clock power margins of a very long, deep pipeline shift register to directly get an estimate for the value of t_0. The final experiment on a long, deep pipeline shift register gave a value of the switching time parameter t_0 within 2% of the predicted value of 0.47 ps, but with 20% spread in the measured values. Together, these results demonstrated that the analytic timing model provides reasonably accurate descriptions of RQL junction switching behavior.

Finally, Chapter 6 put all previous results together and demonstrated the operation of a practical, integrated RQL circuit. I designed an eight-bit adder using the Kogge-Stone architecture. It was fully operational at 6.21 GHz with power margins of 1.5 dB. The device was also operational at about 4 GHz. I found a small operational regions in clock frequency due to on-chip resonances. The power consumption was predicted to be 563 nW and measured to be 570 nW. For comparison, a CMOS transistor would be expected to require roughly two orders of magnitude more power.

7.2 Conclusions and Future Work

RQL began as an alternative to RSFQ. Many of the problems with RSFQ appear to have been mitigated or absolved by RQL. This thesis has provided some key groundwork for future VHSIC applications of RQL. The timing model has been shown to be appropriate. The power supply, a limiting factor of modern CMOS design, is in RQL a matter of designing an appropriate Wilkinson power splitter. Unlike CMOS, only the power actually used in switching is dissipated on the chip. The VHDL models of RQL allow the combinational RQL gates to be used in much the same way as CMOS gates. Finally, the CLA experiment is proof that RQL can perform digital processing tasks.

Needless to say, despite the progress a number of hurdles remain. The cryo-packaging of superconducting digital logic of any sort remains expensive and complicated. Though recent advances in this technology may allow superconducting digital logic to be used in more mainstream applications, it is still a large barrier to the vast markets served by CMOS technology. Also, the 6.21 GHz clock frequency demonstrated here is not markedly faster than the current limits of CMOS, around 4 GHz. Although advanced materials promise critical current densities of $10\,\mathrm{kA/cm^2}$ — a fourfold increase over the switching speeds displayed here — these processes are still at an experimental stage. To accommodate the non-local interconnects between logic elements in the CLA, I had to add an additional clock phase — and thus latency — to cover the distance. A passive transmission line will solve this issue but remains untested. In addition, while the RQL gates described here constitute a universal set, many additional digital elements would be needed to produce the versatility of CMOS designs. Amongst others, the return-to-zero input requirement of RQL clashes directly with the non-return to zero patterns in CMOS. To interface or be compatible to CMOS, RQL must adopt a method of NRZ input.

Perhaps the most glaring omission for a fully-functional CMOS replacement using RQL is memory. While the Set-Reset gate provides a limited memory functionality, contemporary users require vast arrays of memory. The Set-Reset gate cannot be efficiently scaled up to meet the needs of real data storage. A few possible solutions may exist. For example, in Chapter 1, I considered only junctions for which the phase difference between nodes of a junction with zero current was zero. Other kinds of junctions, π-Josephson junctions instead have a phase difference of π when no static current flows through the junction. Such exotic junctions can exhibit hysteretic I-V curves [47], and they can be used to store nonvolatile data. Implementing such a memory structure in RQL is the topic of ongoing research [48, 49, 50].

As a practical matter, the number of junctions on a single chip is limited. While junction counts into the millions have been reported [51], modern processors may require hundreds of millions of junctions. Scaling of the chip size to accommodate hundreds of millions of junctions will be very challenging because of fabrication errors. Instead, individual chips which each are part of a larger circuit may need to be fabricated and tested separately, only later to be included as part of a multi-chip-module.

Finally, all aspect of RQL will benefit from advances in fabrication technologies. Circuit density is limited by the number of metal layers. The four used here lead to sometimes convoluted designs when multiple crossings are needed. A six-layer process may solve some of these issues. Congestion could further be relieved by adding more layers, such as the 10-layer process developed by Tanaka *et al.* [52]. In the horizontal instead of vertical direction, smaller lithography sizes will also allow improvement to circuit density.

7.3 Final Words

Originally conceived as a classical logic family capable of interfacing neatly with quantum computers and quantum bits, RQL's strengths make it a viable classical computer technology in its own right. Many of RQLs advantages are found in the inherently quantum mechanical behavior of Josephson junctions. Superconductivity gives rise to the quantization of flux, tunneling gives birth to the behavior of Josephson junctions, and the Schrödinger equation gives equations of motion which allow traveling wave solutions. In RQL, these traveling waves are not the solitons found on individual junctions, but chains of junctions coupled together through inductors. Unique to RQL, the data is encoded as pairs of pulses, switching junctions back and forth as it travels through the circuit. In this sense, it has aspects of inherently quantum behavior, but is not quantum computing as it is currently understood.

RQL is still a technology in its infancy. Many aspects of my work on RQL have not been mentioned here. Development of the design environment and VHDL models are ongoing processes. Development of RQL logic gates is an ongoing process. Although the combinational behavior of RQL logic gates makes it similar to CMOS in many ways, CMOS has decades of research and development behind it. If RQL proves to be even partially as technologically successful as CMOS, there will undoubtedly be many discoveries in RQL in the future.

Although not yet realized, the idea of quantum computation is old enough to have been studied to a considerable extent in theory. The implications range from simple computational advantages [53] to world view-changing. As for RQL, it is in a somewhat unique place. The underlying behavior of the junctions is manifestly quantum, but the output decidedly classical, so RQL in some ways bridges the behavior between quantum and classical realms. The equations of motion for a junction are those of a damped pendulum, and yet the behavior of Josephson junctions is fantastically rich and complex. Perhaps someday RQL will help bring this complex quantum behavior into everyday use.

Appendix A

Numerical Solution of the Sine-Gordon Equation

Figure 1.12 on page 36 was generated by solving the sine-Gordon equation numerically for an AC bias current, four junctions per phase, and two phases. The final junction is overdamped to prevent reflections, and does not switch itself.

```
f=10;(* GHz *)
bf=Sqrt[2]; (* Beta Factor for Ic stepup *)
\[Beta]1=1.1; (* Damping Factor *)
\[Beta]2=\[Beta]1 bf; (* Alt Damping Factor *)
IcRN = 0.75;(* mV *)
L=9.9; (* pH *)
Amp = 0.77;(* Clock Amplitude *)
A2 = 0.8; (* SFQ Amplitude *)
Phi0 = 2.07; (* mV ps *)
DC=Phi0/(2L IcRN); (* DC Bias *)
t0=Phi0/(2 IcRN); (* Baseline switching time *)
\[Omega]=2\[Pi] f / 1000;
InterL=6.2;
endp=8\[Pi]/\[Omega]; (* End of simulation time *)
delta = \[Omega] t0 / Amp;(* calculated value *)
wc=(2\[Pi])/Phi0 IcRN; (* Calculated Value *)
wp1 = wc/Sqrt[\[Beta]1]; (* Calculated Value *)
wp2 = wc/Sqrt[\[Beta]2]; (* Calculated Value *)
R1=Phi0/(2\[Pi]) wc; (* Calculated Value *)
R2 = R1/bf;(* Calculated Value *)
\[Tau] = 2.3\[Pi]/\[Omega](* Pulse input time *)
x=\[Pi]/2;(* next clock phase *)
(* pDrive[t_,t1_]:=1/2 2\[Pi](-2+Erf[t1/(Sqrt[2] t0)]+Erf[(\[Pi]-t
\[Omega]+t1 \[Omega])/(Sqrt[2] t0 \[Omega])]+Erfc[(-t+t1)/(Sqrt[2]
t0)]+Erfc[(\[Pi]+t1 \[Omega])/(Sqrt[2] t0 \[Omega])]);*)
pD[t_,t1_]:=\[Pi] (1- Erf[Sqrt[2] ( IcRN (-t+t1))/(3 Phi0)]);
pDrive[t_,t1_]:=pD[t,t1]-pD[t,t1+\[Pi]/\[Omega]];
eqn1=Phi0/(2\[Pi] L) (p1[t]-p2[t])==-(1/wp1^2)p1''[t]-1/wc
p1'[t]-Sin[p1[t]]+Amp Sin[\[Omega] t] +0.5DC+A2 Sin[pDrive[t,\[Tau]]];
eqn2=InterL Phi0/(2\[Pi] L)
(-p1[t]+p2[t]/bf+p2[t]-p3[t]/bf)==-(1/wp2^2)p2''[t]-1/wc
```

```
p2'[t]-Sin[p2[t]]+Amp Sin[\[Omega] t] + DC;
eqn3= InterL Phi0/(2\[Pi] L)
(-p2[t]/bf+p3[t]+p3[t]/bf-p4[t])==-(1/wp1^2)p3''[t]-1/wc
p3'[t]-Sin[p3[t]]+Amp Sin[\[Omega] t] + DC;
eqn4=InterL Phi0/(2\[Pi] L)
(-p3[t]+p4[t]/bf+p4[t]-p5[t]/bf)==-(1/wp2^2)p4''[t]-1/wc
p4'[t]-Sin[p4[t]]+Amp Sin[\[Omega] t] + DC;
eqn5=InterL Phi0/(2\[Pi] L)
(-p4[t]/bf+p5[t]+p5[t]/bf-p6[t])==-(1/wp1^2)p5''[t]-1/wc
p5'[t]-Sin[p5[t]]+Amp Sin[\[Omega] t-x] + DC;
eqn6= InterL Phi0/(2\[Pi] L)
(-p5[t]+p6[t]/bf+p6[t]-p7[t]/bf)==-(1/wp2^2)p6''[t]-1/wc
p6'[t]-Sin[p6[t]]+Amp Sin[\[Omega] t-x] + DC;
eqn7= InterL Phi0/(2\[Pi] L)
(-p6[t]/bf+p7[t]+p7[t]/bf-p8[t])==-(1/wp1^2)p7''[t]-1/wc
p7'[t]-Sin[p7[t]]+Amp Sin[\[Omega] t-x] + DC;
eqn8=Phi0/(2\[Pi] L) (p7[t]-p8[t])==-(1/wp2^2)p8''[t]-1/wc
p8'[t]-Sin[p8[t]]+Amp Sin[\[Omega] t-x] +DC;
s=NDSolve[{eqn1, eqn2, eqn3, eqn4, eqn5, eqn6, eqn7, eqn8, p1[0]==0,
p1'[0]==0, p2[0]==0, p2'[0]==0, p3[0]==0, p3'[0]==0, p4[0]==0,
p4'[0]==0, p5[0]==0, p5'[0]==0, p6[0]==0, p6'[0]==0, p7[0]==0,
p7'[0]==0, p8[0]==0, p8'[0]==0}, {p1, p2, p3, p4, p5, p6, p7, p8},
{t,0, endp}, MaxSteps-> 100000];
Plot[{Amp Sin[\[Omega] t],Amp Sin[\[Omega] t-x],1/(2\[Pi])
Evaluate[p1[t]/.s], 1/(2\[Pi]) Evaluate[p2[t]/.s], 1/(2\[Pi])
Evaluate[p3[t]/.s], 1/(2\[Pi]) Evaluate[p4[t]/.s], 1/(2\[Pi])
Evaluate[p5[t]/.s], 1/(2\[Pi]) Evaluate[p6[t]/.s], 1/(2\[Pi])
Evaluate[p7[t]/.s], 1/(2\[Pi]) Evaluate[p8[t]/.s], }, {t,\[Tau]-0.2
(2\[Pi])/\[Omega],\[Tau]+1.2 (2\[Pi])/\[Omega]}, PlotRange-> All]
Plot[{Amp Sin[\[Omega] t],Amp Sin[\[Omega] t-x],1/(2\[Pi])
Evaluate[p1[t]/.s], 1/(2\[Pi]) Evaluate[p2[t]/.s], 1/(2\[Pi])
Evaluate[p3[t]/.s], 1/(2\[Pi]), 1/(2\[Pi]) Evaluate[p5[t]/.s],
1/(2\[Pi]) Evaluate[p6[t]/.s], 1/(2\[Pi]) Evaluate[p7[t]/.s],
1/(2\[Pi]) Evaluate[p8[t]/.s], }, {t,0,endp}, PlotRange-> All]
```

Appendix B

Parameters for fits

B.1 Timing Extraction Results for the JTL

Tables B.1, B.2, and B.3 below show the results from simulation and analysis discussed in Chapter 3, including the analytic function fit to the data for the JTL, the piecewise polynomial function fit to the data for the JTL, and the analytic function fit to the data for the AndOr gate OR operation.

Table B.1: Extracted JTL Timing Parameters. Analytic timing data function fit to data for JTL, as described in Section 3.3. A block diagram of the circuit simulated is shown in Fig. 3.7. f is the clock frequency of the circuit. α_1, α_2, and α_3 are fitting parameters defined in (3.13). In this simulation, $A = 0.83$ and $\phi_c = 4.29$. The netlist for the circuit is given in B.3.4 on page 253.

f [GHz]	α_1	α_2	α_3
1	1.158	0.9926	5.928
2	1.121	1.013	5.592
3	1.1	1.022	4.867
4	1.088	1.029	4.01
4.5	1.084	1.04	15.49
5	1.081	1.034	3.085
5.5	1.078	1.034	2.453
6	1.075	1.032	2.246
6.5	1.076	0.9538	0.4551
7	1.072	0.9484	0.454
7.5	1.07	0.9463	0.4658
8	1.068	0.9526	0.5127
8.5	1.065	0.9487	0.514
9	1.064	0.9584	0.5778
9.5	1.062	0.9614	0.61
10	1.06	0.9698	0.6737
10.5	1.059	0.9645	0.6624
11	1.058	0.9584	0.6456
11.5	1.057	0.9716	0.7328
12	1.056	0.9692	0.7353
13	1.055	0.9799	0.8295
14	1.05	0.9919	0.9262
15	1.048	0.9892	0.9253
16	1.049	0.9811	0.8965
17	1.042	0.9963	1.008

Table B.2: Extraction of JTL Timing Parameters (polynomial fit). Analytic timing data function fit to data for JTL, as described in Section 3.3. A block diagram of the circuit simulated is shown in Fig. 3.7. f is the clock frequency of the circuit. $\alpha_1 1$, $\alpha_1 2$, $\alpha_1 3$, $\alpha_2 1$, $\alpha_2 2$, and $\alpha_2 3$, are the parameters defined in (3.14). In this simulation, $A = 0.83$ and $\phi_c = 4.29$. The netlist for the circuit is given in B.3.4 on page 253.

f [GHz]	α_{11}	α_{12}	α_{13}	α_{21}	α_{22}	α_{23}	w
1	0.04943	-0.1293	0.1068	0.03458	-0.1199	0.1261	1.531
2	0.06989	-0.1908	0.175	0.06889	-0.2382	0.2505	1.547
2.5	0.07512	-0.2095	0.202	0.08034	-0.2757	0.2924	1.542
3	0.08075	-0.2284	0.2287	0.09868	-0.3386	0.3575	1.558
3.5	0.08178	-0.2359	0.2486	0.09908	-0.3334	0.3589	1.554
4	0.08486	-0.2484	0.2712	0.1362	-0.4688	0.4936	1.583
4.5	0.09129	-0.268	0.2973	0.1539	-0.5278	0.5535	1.591
5	0.0895	-0.2679	0.3119	0.1859	-0.6462	0.6747	1.62
5.5	0.09162	-0.2762	0.3308	0.2094	-0.7294	0.7603	1.633
6	0.09336	-0.2834	0.3487	0.2163	-0.7487	0.7847	1.645
6.5	0.09548	-0.2912	0.3668	0.3	-1.069	1.102	1.682
7	0.09773	-0.2987	0.3843	0.2998	-1.06	1.099	1.699
7.5	0.1039	-0.3172	0.4092	0.3305	-1.17	1.209	1.713
8	0.1068	-0.3256	0.4267	0.3791	-1.355	1.397	1.729
8.5	0.1121	-0.3414	0.4497	0.4216	-1.514	1.56	1.745
9	0.1202	-0.3656	0.4789	0.523	-1.91	1.957	1.778
9.5	0.1281	-0.3889	0.5073	0.5681	-2.086	2.144	1.798
10	0.1387	-0.4205	0.5422	0.6981	-2.601	2.665	1.822
10.5	0.1369	-0.413	0.5463	0.614	-2.246	2.308	1.826
11	0.1414	-0.4224	0.5612	0.6775	-2.489	2.554	1.84
11.5	0.1719	-0.5186	0.6482	0.9123	-3.445	3.539	1.871
12	0.1905	-0.5773	0.7056	1.017	-3.872	3.989	1.894
13	0.2589	-0.8039	0.9165	1.681	-6.631	6.883	1.935
14	0.3398	-1.078	1.172	1.526	-5.903	6.081	1.973
15	0.3885	-1.24	1.332	2.43	-9.689	10.08	2.01
16	0.5837	-1.968	2.036	3.486	-14.16	14.86	2.05
17	0.8756	-3.074	3.111	3.93	-15.95	16.68	2.097

Table B.3: Extraction of AndOr OR output timing parameters. Analytic timing data function fit to data for JTL, as described in Section 3.3. A block diagram of the circuit simulated is shown in Fig. 3.7. f is the clock frequency of the circuit. α_1, α_2, and α_3, are the parameters defined in (3.13). In this simulation, $A = 0.83$ and $\phi_c = 4.29$.

f [GHz]	α_1	α_2	α_3
1	1.532	0.3301	5.706
2	0.8508	0.3227	13.53
3	1.408	0.7678	7.683
3.5	1.393	0.8385	6.265
4	1.404	0.9214	5.169
4.5	1.385	0.9689	4.473
5	1.365	1.004	3.904
5.5	1.343	0.9888	3.7
6	1.329	1.009	3.281
6.5	1.315	1.025	2.882
7	1.302	1.038	2.441
7.5	1.297	1.018	2.49
8	1.287	1.029	2.139
8.5	1.281	1.033	1.531
9	1.26	0.899	0.1862
9.5	1.252	0.9183	0.2147
10	1.243	0.9177	0.2196
10.5	1.238	0.865	0.1799
11	1.233	0.8826	0.2019
11.5	1.228	0.8991	0.2259
12	1.224	0.9139	0.2516
13	1.214	0.9414	0.3097
14	1.205	0.9037	0.2685
15	1.181	0.9523	0.3581
16	1.167	0.9738	0.4234

B.2 Comparison of Threshold Values in Timing Extraction

Throughout this thesis, the timing parameters α_i have been used assuming $\phi_c = 2\pi \times 0.632$. This is a convenient value for damped harmonic motion, but still an arbitrary choice. To be sure that the results do not depend strongly on my choice of ϕ_c, I compare the results from both $\phi_c = 2\pi \times 0.632$ and $\phi_c = 2\pi \times 0.75$. As can be seen in Fig. B.1, the agreement between analyses is good. First, α_1 and α_2 lie close to 1. Though α_3 has a clear frequency dependence, as frequency increases α_3 goes to one. As for the comparison between ϕ_c values, both results match closely in all but a few cases. Particularly at low frequencies, some fitting parameters are much different than 1, and in these cases the particular data points are discarded and linear interpolation is used for that frequency.

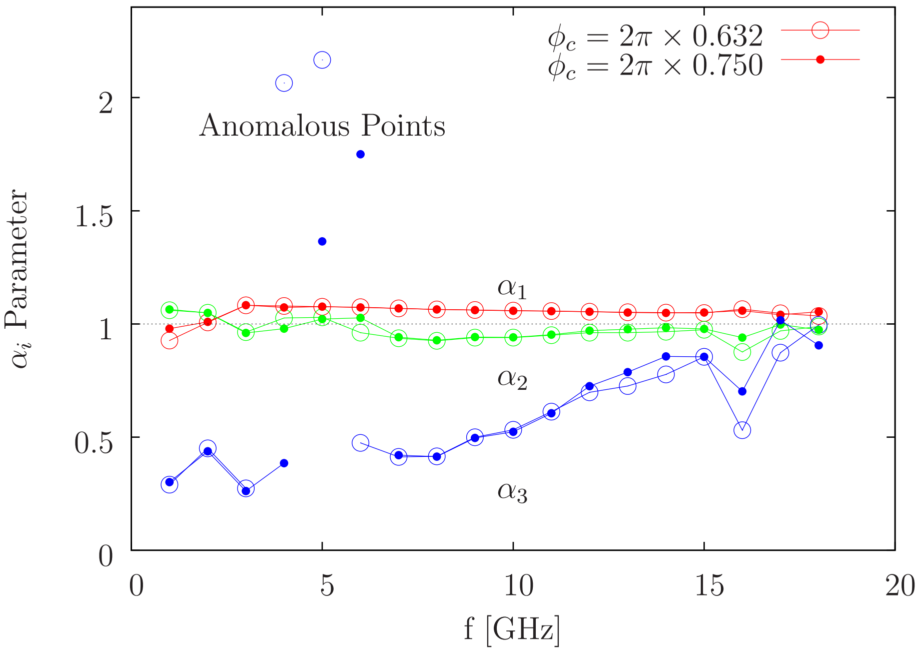


Figure B.1: Comparison of Threshold Values. The resulting α_i parameters are plotted as functions of frequency for $\phi_c = 2\pi \times 0.632$ and $\phi_c = 2\pi \times 0.75$. α_1 and α_2 agree for all cases. α_3 values differ in the region marked "Anomalous Points", though these points are not used in timing calculations due to their anomalous values. Otherwise, the values agree and the choice of ϕ_c is of little importance.

B.3 Simulation File for Timing Extraction

The following perl script performs the extraction of timing data from simulations described in Chapter 3. Inline notes in the code explain the steps performed. A number of smaller subroutines are called which do not reflect significant steps in the overall task.

Each run of the timing extraction started with definition of several constants, such as I_cR_N product, number of junctions in the data path, and the clock amplitude. The threshold value ϕ_c of the phase for crossing was also defined at this time. Then, for each frequency and amplitude, the simulation was run for different values of the input phase. The simulation was run by substituting dummy values in a template with the actual values desired.

The results of the simulation are a file containing time-phase ($t - \phi$) data pairs, which are analyzed with a separate script. One input to this script is the threshold value ϕ_c. Simulations do not depend on this value, however the analysis does. The same simulation results can be analyzed with different values of ϕ_c. Each run performs this same analysis for the desired values of ϕ_c. This analysis runs through the time series data and monitors when each phase crosses the threshold value. Linear interpolation between the points immediately proceeding and following the crossing gives a more accurate value of the time at which the threshold is crossed. Thus finding the timing of the switching of the first junction, the timing of the switching of the second junction can be found by the same way and the difference in time calculated. The timing of the switching of the first junction also gives the phase input time. This data pair of input phase time and delay in switching is recorded in a data file, to be analyzed separately once all simulations have been performed and analyzed.

After all input phases have been simulated for a given clock frequency, clock amplitude, and threshold value, a `gnuplot` script is called to fit all input phase-timing delay values to (3.13). The results of this fit are written to a file; each line of which contains the clock frequency, clock amplitude, threshold value, α_1, α_2, and α_3. (The gnuplot script also fits the data to (3.14), and creates a similar file containing the clock frequency, clock amplitude, threshold value, α_{11}, α_{12}, α_{13}, α_{21}, α_{22}, and

α_{23}.) These files are shown in Tabels B.1, B.2, B.3.

B.3.1 Main Script

```
#!/usr/bin/perl

## This script is the grand-daddy script which runs the timing
## extraction for a whole circuit.  It needs slight modification in a
## few places to account for changes in number of junctions track and
## so forth, but for the most part is automated.  Only this script
## needs to be called to do a timing extraction; individual other
## scripts are called as needed.

## Verbose and trial are good for testing the script before commiting
## an hour or two of computer time to simulations.

## Set the basic parameters here.
$zname = "JTL"; ## Name of the gate being simualted.  Make sure it
## matches the gate in print.tpl.
$zjj = 2; ## Junction count per gate
$zicrn = 0.75; ## IcRN value (0.75 for Hypres)
$ztol = 20; ## Fit parameter tolerance
$znom = 0.83; ## Actual clock amplitude

$verbo = 0; ## 0: runthrough; 1: verbose
$trial = 0; ## 1: trial; 0: full parameter extraction

## Set up a "table" of values to simulate for.
## NO TRAILING ZEROES!!!  @freqs = ("1", "2", "3", "4", "5", "6", "7",
"8", "9", "10", "11", "12", "13", "14", "15", "16", "17", "18", "19",
"20", "21", "2.5", "3.5", "4.5", "5.5", "6.5", "7.5", "8.5", "9.5",
"10.5", "11.5"); @inph = ("0", "1", "2", "3", "4"); @amps = ("0.9",
"1", "1.1"); ## this value is the modifier of the nominal value, not
the actual clock amplitude @amps = ("1"); ## single-amplitude

## This part changes the values if you just want a short runthough to
## test.
if ($trial == 1) {
    @freqs = ("1", "4", "7", "14", "20");
    @amps = ("1"); ## this value is the modifier of the nominal value,
   ## not the actual clock amplitude
    @inph = ("0", "1", "3");
}
```

```
## The threshold (not to be confused with tolerance) is a value used
# to indicate the timing of switching.  Adding additional numbers
# doesn't require additional simulations, but does increase the number
# of output files.  The difference between any reasonable values is
# small.  uncomment the first line here to compare, but for practical
# purposes, only the second line is needed.  @thrs = ("0.6321",
# "0.75");
@thrs = ("0.6321");

## calculate the number of simulations needed.
$runcounter = 1;
$size_freqs = @freqs;
$size_amps = @amps;
$size_inph = @inph;
$size_total = $size_freqs * $size_amps * $size_inph;
$idx = 0;

## clear out old data
my $status = system("\\rm -r datafiles/*");
my $status = system("\\rm -r points/*");
my $status = system("\\rm -r figs/*");
my $status = system("\\rm -r results/*");
my $status = system("\\rm -r latex/*");
my $status = system("\\rm -r compare*.plt");

## run through ALL combinations of frequency and amplitude.
foreach $zcl (@freqs) {
    foreach $zam (@amps) {

$idx = $idx + 1;

## Run through each input phase to generate different sets of
## timing points on the same circuit for the same clock freq
## and amp.
$phasecounter = 2;
foreach $zpn (@inph) {

    ## verbose display
    my $status = system("clear");
    print "\n ** NEW RUN ** \n";
    print "** $zname **\n";
    print "Frequency\t $zcl GHz\n";
    print "Clock Amp\t $zam Nominal\n";
    print "Input Time\t $zpn pi/6\n";
    print "Phase Count\t $phasecounter\n";
    print "Run $runcounter out of $size_total\n\n";
    $runcounter = $runcounter + 1;

    ## create a new script to simulate current run parameters
    open (fTempl, ’scan.tpl’);
    open (fOutpt, ’>scan.pl’);
    while (<fTempl>) {
chomp;
$outdoc = $_;
$outdoc =~ s/ZCL/$zcl/g;
$outdoc =~ s/ZAM/$zam/g;
$outdoc =~ s/ZPN/$zpn/g;
print fOutpt "$outdoc\n";
    }
    close(fTempl);
    close(fOutpt);

    ## run simulation (once)
      my $status = system("perl scan.pl");

    ## The following if segment is depreciated and does not run.
       if ($phasecounter <= 1) {
  foreach $zth (@thrs) {
      open (fTempl, ’extract.tpl’);
      open (fOutpt, ’>extract.pl’);
      while (<fTempl>) {
  chomp;
  $outdoc = $_;
  $outdoc =~ s/ZCL/$zcl/g;
  $outdoc =~ s/ZAM/$zam/g;
  $outdoc =~ s/ZTH/$zth/g;
  $outdoc =~ s/ZSTPT/$phasecounter/g;
  print fOutpt "$outdoc\n";
      }
      close(fTempl);
      close(fOutpt);
      print ("\n\nNew Threshold: $zth\n");
      ## This section needs not be modified ## my
      $status = system("perl extract.pl
      datafiles/start_${zname}_cl${zcl}_am${zam}_15_16.dat
      $zth $zcl"); } } $phasecounter = $phasecounter +1;
      } $phasecounter = 2;
## go through each threshold value and extract switching times
foreach $zth (@thrs) {
    ## create a new script to simulate current run parameters
    open (fTempl, ’extract.tpl’);
    open (fOutpt, ’>extract.pl’);
    while (<fTempl>) {
chomp;
$outdoc = $_;
$outdoc =˜ s/ZCL/$zcl/g;
$outdoc =˜ s/ZAM/$zam/g;
$outdoc =˜ s/ZTH/$zth/g;
$outdoc =˜ s/ZSTPT/$phasecounter/g;
print fOutpt "$outdoc\n";
    }
    close(fTempl);
    close(fOutpt);
    print ("\n\nNew Threshold: $zth\n");
    ## This section needs to be modified to match the files
    ## found in "print.tpl" ##
    my $status = system("perl extract.pl
    datafiles/${zname}_cl${zcl}_am${zam}_02_04.dat $zth
    $zcl"); my $status = system("perl extract.pl
    datafiles/${zname}_cl${zcl}_am${zam}_04_06.dat $zth
    $zcl"); my $status = system("perl extract.pl
    datafiles/${zname}_cl${zcl}_am${zam}_06_08.dat $zth
    $zcl");
}

## create a new gnuplot fitting script foreach $zth (@thrs) {
open (fTempl, ’gnufit.tpl’);
open (fOutpt, ’>gnufit.plt’);
while (<fTempl>) {
    chomp; $outdoc = $_;
    $outdoc =˜ s/ZCL/$zcl/g;
    $outdoc =˜ s/ZAM/$zam/g;
    $outdoc =˜ s/ZNOM/$znom/g;
    $outdoc =˜ s/ZTH/$zth/g;
    $outdoc =˜ s/ZTOL/$ztol/g;
    $outdoc =˜ s/ZICRN/$zicrn/g;
    $outdoc =˜ s/ZJJ/$zjj/g;
    $outdoc =˜ s/ZNAME/$zname/g;
    $outdoc =˜ s/ZIDX/$idx/g;
    print fOutpt "$outdoc\n";
}
close(fTempl);
close(fOutpt);
print ("\n Now Fitting Data to Parameters\n");
```

```
## sort data before analyzing
my $status = system("sort -n -o sorted.dat
points/points_cl${zcl}_am${zam}_th${zth}.dat");
## determine first and last recorded data points
my $status = system ("firstlast.sh");
## perform the fit
my $status = system ("gnuplot gnufit.plt");
}
if ($verbo == 1) { use strict;
   use warnings;
   print "\nPlease press enter key to continue.";
   <STDIN>; } } }

## output results
foreach $zth (@thrs) {
    $fname = "compare_${zname}_v${zicrn}_th${zth}_x${ztol}.plt";
    print "$fname \n";
    open (fTempl, ’compare.tpl’);
    open (fOutpt, ’>’, $fname);
    while (<fTempl>) {
chomp;
$outdoc = $_;
$outdoc =~ s/ZNAME/$zname/g;
$outdoc =~ s/ZICRN/$zicrn/g;
$outdoc =~ s/ZTH/$zth/g;
$outdoc =~ s/ZTOL/$ztol/g;
print fOutpt "$outdoc\n";
    }
    close(fTempl);
    close(fOutpt);
}

if ($trial == 1) {
    my $status = system ("evince figs/*");
}

my $status = system("\\rm -r datafiles/*");
```

B.3.2 Timing Extraction Script

This script has a simple purpose. Analyze the output of the simulation to determine when the output phase has crossed the given threshold value ϕ_c, determine the time difference between the two outputs, and calculate the corresponding input phase θ_{in} and phase delay $\Delta\theta$.

This script reads in the data in the format produced by spice, discarding simulation information provided in the file. The file is a series of time values and the phase values of the two junctions under consideration. The script checks the phase value of the first junction every time step, making note when it has crossed the threshold value. When the threshold has been crossed, the time of crossing is interpolated from the value immediately proceeding the crossing, and the value immediately following it. The script flags the first junction as having crossed the threshold, and waits for the second junction to cross the threshold.

When the second junction crosses the threshold, the time is estimated in the same was as for the first junction. From these two time values t_{in} and t_{out}, and given the clock frequency f, the script calculates the input phase θ_{in} and phase delay $\Delta\theta$, recording these values to a file. JTL simulations produce several data points with each execution of this script, gates produce only one.

```
#!/usr/bin/perl

## This is an extraction tool and generally does NOT need to be
## modified in any way.
$pi = 3.14159265358979323;

## read in data
(scalar(@ARGV) == 3) || die "SYNOPSIS: <filename> <threshold> <freq>\n";
$thresh = $ARGV[1];
```

```
$frq = $ARGV[2];
$fname = $ARGV[0];

$stpt = ZSTPT;

## open input file
open (dataF, $fname) || die "ERROR: Can’t Open File $fname";

## skip junk lines
## The number of junk lines is set by the output format of WRspice.
$line=<DATA>;
$line=<DATA>;
$line=<DATA>;
$line=<DATA>;
$line=<DATA>;
$line=<DATA>;

## open output file (in append mode)
open (resultF, ’>>points/points_clZCL_amZAM_thZTH.dat’);
# open (resultG, ’>>points/metric_clZCL_amZAM_thZTH.dat’); printf
# resultF ("\# inPhase, DeltaPhase, OutPhase, frequency, threshold,
# flagtype, index\n");

## the flags keep track of phase 0 and 1 if they are in the flipped or
## unflipped state
$flag0 = 0;
$flag1 = 0;
$idx = 1;
$flag1st = 0;

while($line=<dataF>) {
    ($index, $time, $p0, $p1)=split(" ", $line);

    ## look for upward threshold on p0
    if ($p0 >= $thresh && $flag0 == 0) {
$Dpost = $p0 - $thresh;
$Dprior = $thresh - $p0old;
$up0 = ($time* $Dprior + $timeold* $Dpost)/($Dpost + $Dprior);
$flag0 = 1;
    }

    ## look for upward threshold on p1
    if ($p1 >= $thresh && $flag1 == 0) {
$Dpost = $p1 - $thresh;
$Dprior = $thresh - $p1old;
```

```
$up1 = ($time* $Dprior + $timeold* $Dpost)/($Dpost + $Dprior);
## calculate normalized phase
$DeltaT = $up1 - $up0;
$DeltaP = $DeltaT * 2 * $pi * $frq * 1e9;
$InputP = $up0 * 2 * $pi * $frq * 1e9;
$InputP = $InputP / (2*$pi);
$InputP = $InputP - int($InputP);
$InputP = $InputP * 2 * $pi;
$OutptP = $up1 * 2 * $pi * $frq * 1e9;
$OutptP = $OutptP / (2*$pi);
$OutptP = $OutptP - int($OutptP);
$OutptP = $OutptP * 2 * $pi;
## print results to file file format: inPhase, DeltaPhase,
## OutPhase, frequency, threshold, flagtype, index
if ($DeltaP
>= 0 && $DeltaP <= $pi/2.0 && $InputP < $pi) { if ( $stpt == 0
&& $flag1st == 0) { printf resultG ("earlyp = %f \# ZPN + \n",
$InputP); $flag1st = 1; } if ( $stpt == 1 && $flag1st == 0) {
printf resultG ("fitstart = %f \# ZSTPT +\n", $InputP);
$flag1st = 1; } if ( $flag1st == 0) { printf resultF ("%f \t
%f \t %f \t %f \t %f \t %f \t %f \n", $InputP, $DeltaP,
$OutptP, $frq, $thresh, $flag1, $idx); } } print "$InputP \t
$DeltaP \t $flag1 \t $idx \n"; $idx = $idx + 1; $flag1 = 1; }
    ## look for downward threshold on p0
    if ($p0 <= (1-$thresh) && $flag0 == 1) {
$Dpost = (1-$thresh) - $p0;
$Dprior = $p0old - (1-$thresh);
$down0 = ($time* $Dprior + $timeold* $Dpost)/($Dpost + $Dprior);
$flag0 = 0;
    }

    ## look for downward threshold on p1
    if ($p1 <= (1-$thresh) && $flag1 == 1) {
$Dpost = (1-$thresh) - $p1;
$Dprior = $p1old - (1-$thresh);
$down1 = ($time* $Dprior + $timeold* $Dpost)/($Dpost + $Dprior);
## calculate normalized phase
$DeltaT = $down1 - $down0;
$DeltaP = $DeltaT * 2 * $pi * $frq * 1e9;
$InputP = ($down0 * 2 * $pi * $frq * 1e9);
$InputP = $InputP / (2*$pi);
$InputP = $InputP - int($InputP);
$InputP = $InputP * 2 * $pi - $pi;
$OutptP = ($down1 * 2 * $pi * $frq * 1e9);
$OutptP = $OutptP / (2*$pi);
```

```
$OutptP = $OutptP - int($OutptP);
$OutptP = $OutptP * 2 * $pi - $pi;
## print results to file file format: inPhase, DeltaPhase,
## OutPhase, frequency, threshold, flagtype
if ($DeltaP >= 0
&& $DeltaP <= $pi/2.0 && $InputP < $pi) { if ( $flag1st == 0)
{ printf resultF ("%f \t %f \t %f \t %f \t %f \t %f \t %f \n",
$InputP, $DeltaP, $OutptP, $frq, $thresh, $flag1, $idx); } }
print "$InputP \t $DeltaP \t $flag1 \t $idx \n"; $idx = $idx +
1; $flag1 = 0; }

    $p0old = $p0;
    $p1old = $p1;
    $timeold = $time;
}
close(resultF);
# close(resultG);
```

B.3.3 Gnuplot Fitting of Data

This gnuplot script fits the data to both (3.13) and (3.14). It also produces various graphs for analysis of the simulations.

```
## GNUplot fitting instructions.  Does not need to be modified.  ##

## This gnuplot script does the bulk of the fitting work need to
#  extract the fitting parameters.  It is a general script which takes
#  a standard input generated elsewhere in the timing extraction
#  process.  Thus, it generally shouldn't need to be modified at all.
#  A number of variables appear here in ALL CAPS, which means they
#  will be replaced by the motherscript (generate_pulses.pl) before
#  the run-copy is executed.  I have added a lot of commentary here to
#  explain what is actually going on.

## Double hashes indicate comments and should not be changed.  Single
# hashes are used to comment out code that could, in principle, be
# run.  This line is the exception.

## IcRN is a fixed value for a given process.
IcRN = ZICRN # fixed for each process, in mV

## These three are variables that change with each simulation, though
```

```
# is generally constant for any given gate.  f = ZCL # in GHz A =
ZNOM*(0.72*ZAM+0.28) ## linear fit for nominal, high, low values N =
ZJJ

## Initialize the fitting parameters to unity.  This is not just a
#  mathematical convenience; in theory, these should stay at unity.  a1 =
1.0 a2 = 1.0 a3 = 1.0


## Do some math to calculate more easy-to-use parameters.
w = 2*pi*f ## angular frequency
ep=1/100.0 ## fake derivative constant
t = 2.07 / (2*IcRN) # calculate t0 from IcRN, the minimum switching time
d = N*w*t/A; d = d/1000 # fixes GHz x ps scale factor
z = acos(d-1) # failure point
zE= acos(d+1)
# early window point (this is pretty much depreciated and not used at
#  all, but is here for legacy reasons)
z1= 1/a2*acos(d/a3-1)
# at this point z1 = z, though in principle it could change.  Again,
#  something of a holdover.

## A custom digit cutoff function needed for output.  I don’t want too
## many digits clogging up my results.  Results are normalized to
## values of order unity, so past the fourth digit or so no real
## information is lost.  rdx = 4 ;round(x) = (x != 0) ?
## 10**(floor(log10(x))-(rdx-1))*floor(0.5+x/(10**(floor(log10(x))-(rdx-1))))
## : 0

## Make things pretty for gnuplot.
reset
unset label
unset arrow
set print ’-’
set term x11
set samples 600

## An interactive output.  Not important for big runs, but useful for
## debugging.

pr "\nParameters: "
pr "t0 = ", t, " ps", "\t", "A  = ", A
pr "f  = ", f, " GHz", "\t", "N  = ", N
pr "a1 = ", a1, "\t", "a2 = ", a2, "\t", "a3 = ", a3
pr " "
pr "d  = ", round(d)
pr "z  = ", round(z), " rad \t(Failure Point)"
pr "z  = ", round(zE), " rad \t(Early Failure Point)"

## Define the Taylor Series Expansion of ArcCos(x) and Cos(x) about 0
## and pi/2, respectively.

p_acos(x) = \
pi/2 - x - \
1/6.0 * x**3 - \
3/40.0 * x**5 - \
5/112.0 * x**7 - \
35/1152.0 * x **9 - \
63/2816.0 * x**11

# p_cos(x) = 1 - 1/2.0 * x**2 + 1/24.0 * x**4 - 1/720.0 * x**6 +
# 1/40320.0 * x**8 - 1/3628800.0 * x**10 legacy fit about zero.
# Improved by the following:

p_cos(x) = \
-(x-pi/2) + \
1.0/6 * (x-pi/2)**3 -\
1.0/120 * (x-pi/2)**5 + \
1.0/5040 * (x-pi/2)**7 - \
1.0/362880 * (x-pi/2)**9

## Define the analytic timing equation, both in trig and taylor.
f(x) = acos(cos(x)-d)-x
p_f(x) = p_acos(p_cos(x)-d)-x

##  Define three fitting values used for the fitting process.
c1=1.0
c2=1.0
c3=1.0

## Define the fit version of the timing equation.  So far, it is
## identical to f(x) and p_f(x).

g(x) = c1*(acos(cos(c2*x)-d*c3)-(c2*x))
p_g(x) = c1*(p_acos(p_cos(c2*x)-d*c3)-(c2*x))

## Define some limits.  This will plot a graph only in the range of
## the unfitted curve.

x_lim = 1.2 * z
```

```
y_lim = 1.2 * f(0)
# if (y_lim < f(0)) y_lim = 1.2*f(0) # legacy, delete

## Do some good gnuplot stuff.  I'm really just making the graphs look
## nice.  This section isn't vital for fitting.

set xrange [0:x_lim]
set xlabel "Input Phase [rad]"
set xtics nomirror
set yrange [0:y_lim]
set ylabel "Output Phase Delay [rad]"
set ytics nomirror

## Here, I'm calculated the appropriate time (in ps) for the
## normalized time (i.e. input phase).

set x2range [0:x_lim/w*1000]
set x2label "Input Time [ps]"
set x2tics 0,floor(x_lim/w*1000/8.0)
set y2range [0:y_lim/w*1000]
set y2label "Output Time Delay [ps]"
# set y2tics 0,floor(y_lim/w*1000/8.0)
stack = (floor(y_lim/w*1000/8.0) > 1 ? floor(y_lim/w*1000/8.0) : 1)
set y2tics 0, stack

## This section uses Newton's method to calculate the point at which
## the meta-stable point is found.

x0 = 0.9*z ## start point
h(x) = f(x)-pi/2 ## zero'd function
h1(x) = (h(x+ep)-h(x-ep))/(2*ep) ## fake derivative

x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)
x0 = x0 - h(x0)/h1(x0); x0 = real(x0)

## For N delta < 1 the metastable point coincides with the end of the
## timing window.  So just leave it there.
```

```
if (f(z)-pi/2 < 0) x0 = z; pr "x0 = z"
if (imag(x0)!=0) print "Stability Limit Not Found"
pr "x0 = ", round(x0), " rad \t(Stability Limit)"


## Now we’re going to do the same with the stable timing point.
y0 = 0.0
h(x) = f(x)-pi/2
h1(x) = (h(x+ep)-h(x-ep))/(2*ep)

y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)
y0 = y0 - h(y0)/h1(y0); y0 = real(y0)

if (imag(y0)!=0) print "Stability Point Not Found"
if (real(y0)< 0) print "Stability Point Probably at 0"; y0 = 0.0
pr "y0 = ", round(y0), " rad \t(Stability Point)"
pr " " ## End of message outputs

## Draw in some dots where the stability points are found.
set label at 0.9*z,f(0.9*z) point lt 0 pt 7 ps 1
set label at 0.9*z1,g(0.9*z1) point lt 0 pt 7 ps 1
set label at x0,f(x0) point lt 1 pt 7 ps 2
set label at y0,f(y0) point lt 1 pt 7 ps 2

## This part outputs the timing parameters individually in a file.
## It’s a legacy part of the script and not essential.  This is
## different from the fitting parameters, which have not been
## calculated yet and are output separately.

set print "output_params.txt"
pr d, z, x0, y0
set print ’-’

## END GENERAL INPUT ## This ends the setup for the fitting.  We now
## begin the actually fitting process.

## Make pretty gnuplot graphs
unset label
```

```
## Legacy stuff, delete
# fitstart = 0
# earlyp = 0
# load "points/metric_clZCL_amZAM_thZTH.dat"

## This file contains the range of fitting values.  We need to know
## over which range to fit.

load "firstlast.dat"

## Initialized fitting parameters.
c1 = 1.0
c2 = 1.0
c3 = 1.0

## fit to analytic equation, excuding the very last few points.  We
## only want the middle 90%.

fit \
[firstpoint+0.05*(lastpoint-firstpoint):lastpoint-0.05*(lastpoint-firstpoint)]\
g(x) ’sorted.dat’ using 1:2 via c1, c2, c3

## Save the fit values for later use.
b1 = c1
b2 = c2
b3 = c3

## Gnuplot doesn’t work well with interdependant fitting variables.
## The Out-values here correspond with theory.

out1 = c1*c3
out2 = c2
out3 = 1.0/c3

## calculate the new cutoff point based on the fit data.
z1 = 1/c2*acos(d*c3-1)

## display the endpoint on the graph with an explicit dot
set label at z1,g(0.999*z1) point lt 2 pt 7 ps 1

## Introduce a piecewise 2nd-order polynomial with break at midway
p1(x) = (k11*x+k12)*x+k13
p2(x) = (k21*x+k22)*x+k23
## note that midway is read from the file "firstlast.dat"
```

```
w1 = midway

## Join the two polynomials in a piecewise fashion.
z2(x) = x < w1 ? p1(x) : (x < lastpoint ? p2(x) : 1/0)

## Do the fit for the middle 90%.
fit \
firstpoint+0.05*(lastpoint-firstpoint):lastpoint-0.05*\
lastpoint-firstpoint)] z2(x) "sorted.dat" using 1:2 via \
11, k12, k13, k21, k22, k23

## Legacy
# out1 = c1
# out2 = c2
# out3 = c3
# z1E= z1
# z1 = 1/out2*acos(d/out3-1)

## The tolerace is a factor chosen to weed out bad fits.  In
## principle, any fit works.  However, for practical reasons we may
## wish to ignore fits with excessivly large or small values, as these
## tend to be hard to work with in VHDL.
tolerance = ZTOL

## gobot is a flag to check for values within the tolerance.  if any
## value is too big or too small, the value is NOT written to the
## results file (though it is written to the gnu-file or refernce
## later).

gobot = 1

if (out1 > tolerance) gobot = 0
if (out2 > tolerance) gobot = 0
if (out3 > tolerance) gobot = 0

if (out1 < 1.0/tolerance) gobot = 0
if (out2 < 1.0/tolerance) gobot = 0
if (out3 < 1.0/tolerance) gobot = 0

if (gobot == 0) pr "Warning: Tolerances Exceeded"
pr ""
pr "params : \t a1 \t a2 \t a3"
pr "nu_fit : \t", out1, "\t", out2, "\t", out3
pr "polyfit: \t", b1, "\t", b2, "\t", b3
pr ""
```

```
## The fit is done and checked.  We now output the values to a file.
set print "results/TX_ZNAME_vZICRN_thZTH_xZTOL.dat" append
if (gobot == 1) pr f, A, out1, out2, out3, z1, firstpoint, lastpoint

## Also output a more human-readable version.
set print "results/TX_ZNAME_vZICRN_thZTH_xZTOL_human.dat" append
if (gobot == 1) pr f, " & ", A, " & ", round(out1), " & ", round(out2), \
 & ", round(out3), " & ", round(z1), " & ", round(firstpoint), " & ", \
ound(lastpoint), " \\\\"

## Also output the polyfit values to a SEPARATE file.
set print "results/TX_ZNAME_vZICRN_thZTH_xZTOL_poly.dat" append
pr f, A, k11, k12, k13, k21, k22, k23, w1, firstpoint, lastpoint

## Human-readable polytext
set print "results/TX_ZNAME_vZICRN_thZTH_xZTOL_polyhuman.dat" append
pr f, " & ", round(A), " & ", round(k11), " & ", round(k12), " & ", \
ound(k13), " & ", round(k21), " & ", round(k22), " & ", round(k23), " & ", \
ound(w1), " & ", round(firstpoint), " & ", round(lastpoint), " \\\\"

idx = ZIDX

## We also want the values easily accessible in gnuplot.  The
## following outputs the same data in a gunplot-readable file.

set print "results/gnu_ZNAME_vZICRN_thZTH_xZTOL.dat" append
pr \
"f", idx, " = ", f, \
"; A", idx, " = ", A, \
"; outA", idx, " = ", round(out1), \
"; outB", idx, " = ", round(out2), \
"; outC", idx, " = ", round(out3), \
"; z", idx, " = ", round(z1), \
"; fp", idx, " = ", round(firstpoint), \
"; lp", idx, " = ", round(lastpoint), "\n", \
"nu_", idx, "(x) = ", c1, \
*(acos(cos(", c2, "*x)-", d*c3, ")-(", c2, "*x))\n", \
"k11_", idx, " = ", round(k11), \
"; k12_", idx, " = ", round(k12), \
"; k13_", idx, " = ", round(k13), \
"; k21_", idx, " = ", round(k21), \
"; k22_", idx, " = ", round(k22), \
"; k23_", idx, " = ", round(k23), \
"; w1_", idx, " = ", round(w1), "\n", \
"poly_", idx, "(x) = x < ", round(w1), " ? (", k11, "*x+", k12, ")*x+", k13, " :
"maxidx = ", idx, "\n"

## check the limits of the functions to make sure we get a good view
## of the whole thing

ylim = (g(0) > f(0)) ? g(0) : f(0)
set yrange [0:1.2*ylim]

x_lim = (lastpoint > z1) ? lastpoint : z1
x_lim = 1.1 * x_lim
x_lim = x_lim > pi ? pi : x_lim

## display the relevant data on the graph
set label sprintf("a_1 = %g", out1) at x_lim/4.0, 1.0*ylim
set label sprintf("a_2 = %g", out2) at x_lim/4.0, 0.9*ylim
set label sprintf("a_3 = %g", out3) at x_lim/4.0, 0.8*ylim

## calculate a few values for putting the values on the figure
x_loc = 0.7*x_lim
y_loc = ylim*0.8
delta = ylim*0.07

## output the circuit run parameters for reference.
set label sprintf("f = %g GHz", f) at x_loc, y_loc-delta
# set label sprintf("$\\mathrm{t_0}$ = %g ps", t) at x_loc, y_loc-delta
set label sprintf("A = %g", A) at x_loc, y_loc-2*delta
set label sprintf("N = %g", N) at x_loc, y_loc-3*delta
set label sprintf("z = %g rad", round(z1)) at x_loc, y_loc-4*delta
set label at firstpoint,g(firstpoint) point lt 2 pt 13 ps 1
set label at lastpoint,g(lastpoint) point lt 2 pt 13 ps 1

# set arrow from firstpoint,0 to firstpoint,g(firstpoint) if
# (lastpoint<z1) set arrow from lastpoint,0 to lastpoint,g(lastpoint);
# else set label "boundary" at z1,g(0.99*z1)

## Add an arrow for pi/6
set arrow from pi/6,0 to pi/6, ylim lt 0 nohead
set arrow from 5*pi/6,0 to 5*pi/6, ylim lt 0 nohead

## Output an EPS file for viewing later.
set term postscript eps enhanced color
set output "figs/TX_ZNAME_vZICRN_clZCL_amZAM_thZTH_xZTOL.eps"
plot [0:x_lim]\
f(x) title "Nominal", \
g(x) title "Fit", \
z2(x) title "Polyfit", \
"points/points_clZCL_amZAM_thZTH.dat" using 1:2 with points notitle, \
"points/points_clZCL_amZAM_thZTH.dat" using 1:2 with lines notitle

## Output a latex file for use later.
set term epslatex color
set output "latex/TX_ZNAME_vZICRN_clZCL_amZAM_thZTH_xZTOL.tex"
replot

set term x11
```

B.3.4 Circuit Netlist

This netlist corresponds to the block diagram of the circuit shown in Fig. 3.7(a).

```
netlist
*
*
* HNL Generated netlist of FT_JTL

.global 0
* MODEL Declarations
* Found stopping cell - V
* Found stopping cell - resistor
* Found stopping cell - rsj
.subckt rsj a b jjmod=jj110D ic=0.25 icrn=0.7 rsh=2.8 lprsh=1.5p
b0 a b phi jjmod area=ic
r_sh a a1 rsh
lp_rsh a1 b lprsh
.ends rsj
* Found stopping cell - lp
* Found stopping cell - muind
* Found stopping cell - lm_open
* Found stopping cell - inductor

.subckt jtl_141_200 a d0 d1 c0 c1 q
c0 c0 0 $&jtl_141_200_c0_cap%f
LPbias net016 net019 $&jtl_141_200_LPbias_ind%f
K1 L1 L0 $&jtl_141_200_K1_mut%
K1d L1d L0 $&jtl_141_200_K1d_mut%
```

```
Lg0 net013 0 $&jtl_141_200_Lg0_ind%f
Lg1 net022 0 $&jtl_141_200_Lg1_ind%f
Xb1 net022 net014 rsj jjmod=Hyp5a ic=$&jtl_141_200_b1_ic
rsh=$&jtl_141_200_b1_rsh lprsh=$&jtl_141_200_b1_lprsh%p
Xb0 net013 net021 rsj jjmod=Hyp5a ic=$&jtl_141_200_b0_ic
rsh=$&jtl_141_200_b0_rsh lprsh=$&jtl_141_200_b0_lprsh%p
L3 a net021 $&jtl_141_200_L3_ind%p
L1 c0 c1 $&jtl_141_200_L1_ind%p
L4 net021 net019 $&jtl_141_200_L4_ind%p
L0 net016 0 $&jtl_141_200_L0_ind%p
L5 net019 net014 $&jtl_141_200_L5_ind%p
L6 net014 q $&jtl_141_200_L6_ind%p
L1d d0 d1 $&jtl_141_200_L1d_ind%p
.ends jtl_141_200

.subckt clock_biases c0 c1 d0 d0p d1p c0p c1p d1
L3 c1p c1 $&clock_biases_L3_ind%f
L2 d1p d1 $&clock_biases_L2_ind%f
L1 c0 c0p $&clock_biases_L1_ind%f
L0 d0 d0p $&clock_biases_L0_ind%f
.ends clock_biases

.subckt xjtl_10jj_1p a clk0in clk0out clk1in clk1out clk2in clk2out q
Lg1 clk2in clk2out $&xjtl_10jj_1p_Lg1_ind%p
XI8 net149 net2380 net2381 net2382 net2383 q jtl_141_200
XI6 net152 net2430 net2431 net2432 net2433 net149 jtl_141_200
XI4 net155 net2480 net2481 net2482 net2483 net152 jtl_141_200
XI2 net158 net2530 net2531 net2532 net2533 net155 jtl_141_200
XI0 a net2580 net2581 net2582 net2583 net158 jtl_141_200
XI9 net240 clk1out clk0in net2380 net2381 net2382 net2383 net242 clock_biases
XI7 net245 net240 net242 net2430 net2431 net2432 net2433 net247 clock_biases
XI5 net250 net245 net247 net2480 net2481 net2482 net2483 net252 clock_biases
XI3 net255 net250 net252 net2530 net2531 net2532 net2533 net257 clock_biases
XI1 clk1in net255 net257 net2580 net2581 net2582 net2583 clk0out clock_biases
.ends xjtl_10jj_1p

.subckt q_in a0 a1 q
Xb0 net020 net013 rsj jjmod=Hyp5a
ic=$&q_in_b0_ic rsh=$&q_in_b0_rsh lprsh=$&q_in_b0_lprsh%p
Lb0 net020 0 $&q_in_Lb0_ind%f
L1 net013 q $&q_in_L1_ind%f
K2 L2 L0 $&q_in_K2_mut%
L0 0 net013 $&q_in_L0_ind%p
L2 a1 a0 $&q_in_L2_ind%p
.ends q_in
```

```
.subckt launch1jj clk0in clk1in clk1out clk2in clk2out datin0 datin1 q
XI1 datin0 datin1 q q_in
R0 0 clk0in $&launch1jj_R0_res%
L0 clk2out clk2in $&launch1jj_L0_ind%p
L1 clk1out clk1in $&launch1jj_L1_ind%p
.ends launch1jj

.subckt terminator a clk0out clk1in clk2in
R2 0 clk2in $&terminator_R2_res%
R1 0 clk1in $&terminator_R1_res%
R0 0 a $&terminator_R0_res%
V0 clk0out 0 pwl(0 0 20ps $&Voff)
.ends terminator

XI1 net19 net21 net5 net6 net22 net4 net20 net23 xjtl_10jj_1p
XI0 net5 clk1in net6 clk2in net4 datin0 net25 net19 launch1jj
XI4 net23 net21 net22 net20 terminator
R0 0 net25 $&R0_res%
.include /home/cds5/spice/model.lib
Vdatin0 datin0 0 pwl(0 0 $&ramp $&lo \
 $&ta1 $&lo $&tb1 $&hi $&tc1 $&hi $&td1 $&lo \
 $&ta2 $&lo $&tb2 $&hi $&tc2 $&hi $&td2 $&lo \
 $&ta3 $&lo $&tb3 $&hi $&tc3 $&hi $&td3 $&lo \
 $&ta4 $&lo $&tb4 $&hi $&tc4 $&hi $&td4 $&lo \
 $&ta5 $&lo $&tb5 $&hi $&tc5 $&hi $&td5 $&lo \
 $&ta6 $&lo $&tb6 $&hi $&tc6 $&hi $&td6 $&lo \
 $&ta7 $&lo $&tb7 $&hi $&tc7 $&hi $&td7 $&lo \
 $&ta8 $&lo $&tb8 $&hi $&tc8 $&hi $&td8 $&lo \
 $&ta9 $&lo $&tb9 $&hi $&tc9 $&hi $&td9 $&lo \
 $&ta0 $&lo $&tb0 $&hi $&tc0 $&hi $&td0 $&lo \
)
Vclk2in clk2in 0 sin(0 $&Vamp $&clk $&cphs)
Vclk1in clk1in 0 sin(0 $&Vamp $&clk 0)
.tran $&res $&tend

.end

netlist
.control

* Reduced set of global parameters
Lp = L
Lprsh = L
Resi = R
```

```
Resh = R

* Delete Voltage Bus and Insert this before Individual Variables
* Clock, phase, and Clock Amp Factor are already loaded This is just a
* generice block of text used inside netlist.param.  The conversion
* script wsp2mlt2r makes certain assumptions which are not valid in
* this case.  Furthermore, this block of text introduces the timing
* quantities needed to run the simulation, such as pulse inputs times.


* This is for the output amplifier, not really used in the CIM
* extraction process

Bout=0.805

tau = 1/(clk)
rep = 3*tau
tend = 16*tau
res = tend/10000
ramp=0.25*tau
ofst = tau*(1*Pin)
ofs2 = ofst+tau/2
cphs = tau/4
wdh = 10ps
stp = tau/120.0
Fix = 0/12.0

* Voltage bus
Vamp = 110.0mV
* one half flux quantum offset
Phi0=2.0679;   Voff=50*R*Phi0/L/1000
Vbus=100; Vamp=Vbus*B*jc*R/1000
Vamp = Vamp * Vmod

* Output
Vout=12/1000*Bout
* Input
amp=9*jc
off=3.15/L

hi=(off+amp)/1000
lo=(off-amp)/1000

* Generate timing points
```

```
ta1 = tau*(1+Pin)
tb1 = ta1 + wdh
tc1 = ta1 + 2*cphs
td1 = tc1 + wdh

ta2 = tau*(3+Pin)+1*stp
tb2 = ta2 + wdh
tc2 = ta2 + 2*cphs
td2 = tc2 + wdh

ta3 = tau*(4+Pin)+2*stp
tb3 = ta3 + wdh
tc3 = ta3 + 2*cphs
td3 = tc3 + wdh

ta4 = tau*(6+Pin)+3*stp
tb4 = ta4 + wdh
tc4 = ta4 + 2*cphs
td4 = tc4 + wdh

ta5 = tau*(7+Pin)+4*stp
tb5 = ta5 + wdh
tc5 = ta5 + 2*cphs
td5 = tc5 + wdh

ta6 = tau*(8+Pin)+5*stp
tb6 = ta6 + wdh
tc6 = ta6 + 2*cphs
td6 = tc6 + wdh

ta7 = tau*(10+Pin)+6*stp
tb7 = ta7 + wdh
tc7 = ta7 + 2*cphs
td7 = tc7 + wdh

ta8 = tau*(12+Pin)+7*stp
tb8 = ta8 + wdh
tc8 = ta8 + 2*cphs
td8 = tc8 + wdh

ta9 = tau*(13+Pin)+8*stp
tb9 = ta9 + wdh
tc9 = ta9 + 2*cphs
td9 = tc9 + wdh
```

```
ta0 = tau*(15+Pin)+9*stp
tb0 = ta0 + wdh
tc0 = ta0 + 2*cphs
td0 = tc0 + wdh

tFa1 = tau*(1+Fix)
tFb1 = tFa1 + wdh
tFc1 = tFa1 + 2*cphs
tFd1 = tFc1 + wdh

tFa2 = tau*(3+Fix)
tFb2 = tFa2 + wdh
tFc2 = tFa2 + 2*cphs
tFd2 = tFc2 + wdh

tFa3 = tau*(4+Fix)
tFb3 = tFa3 + wdh
tFc3 = tFa3 + 2*cphs
tFd3 = tFc3 + wdh

tFa4 = tau*(6+Fix)
tFb4 = tFa4 + wdh
tFc4 = tFa4 + 2*cphs
tFd4 = tFc4 + wdh

tFa5 = tau*(7+Fix)
tFb5 = tFa5 + wdh
tFc5 = tFa5 + 2*cphs
tFd5 = tFc5 + wdh

tFa6 = tau*(8+Fix)
tFb6 = tFa6 + wdh
tFc6 = tFa6 + 2*cphs
tFd6 = tFc6 + wdh

tFa7 = tau*(10+Fix)
tFb7 = tFa7 + wdh
tFc7 = tFa7 + 2*cphs
tFd7 = tFc7 + wdh

tFa8 = tau*(12+Fix)
tFb8 = tFa8 + wdh
tFc8 = tFa8 + 2*cphs
tFd8 = tFc8 + wdh
```

```
tFa9 = tau*(13+Fix)
tFb9 = tFa9 + wdh
tFc9 = tFa9 + 2*cphs
tFd9 = tFc9 + wdh

tFa0 = tau*(15+Fix)
tFb0 = tFa0 + wdh
tFc0 = tFa0 + 2*cphs
tFd0 = tFc0 + wdh


tGa1 = tau*(1+Fix)
tGb1 = tGa1 + wdh
tGc1 = tGa1 + 2*cphs
tGd1 = tGc1 + wdh

tGa2 = tau*(3+Fix)
tGb2 = tGa2 + wdh
tGc2 = tGa2 + 2*cphs
tGd2 = tGc2 + wdh

tGa3 = tau*(4+Fix)
tGb3 = tGa3 + wdh
tGc3 = tGa3 + 2*cphs
tGd3 = tGc3 + wdh

tGa4 = tau*(6+Fix)
tGb4 = tGa4 + wdh
tGc4 = tGa4 + 2*cphs
tGd4 = tGc4 + wdh

tGa5 = tau*(7+Fix)
tGb5 = tGa5 + wdh
tGc5 = tGa5 + 2*cphs
tGd5 = tGc5 + wdh

tGa6 = tau*(8+Fix)
tGb6 = tGa6 + wdh
tGc6 = tGa6 + 2*cphs
tGd6 = tGc6 + wdh

tGa7 = tau*(10+Fix)
tGb7 = tGa7 + wdh
tGc7 = tGa7 + 2*cphs
tGd7 = tGc7 + wdh
```

```
tGa8 = tau*(12+Fix)
tGb8 = tGa8 + wdh
tGc8 = tGa8 + 2*cphs
tGd8 = tGc8 + wdh

tGa9 = tau*(13+Fix)
tGb9 = tGa9 + wdh
tGc9 = tGa9 + 2*cphs
tGd9 = tGc9 + wdh

tGa0 = tau*(15+Fix)
tGb0 = tGa0 + wdh
tGc0 = tGa0 + 2*cphs
tGd0 = tGc0 + wdh


* Individual Variables *
jtl_141_200_c0_cap = 0.4
jtl_141_200_LPbias_ind = 1*L
jtl_141_200_K1_mut = 0.447989
jtl_141_200_K1d_mut = 0.131762
jtl_141_200_Lg0_ind = 300*L
jtl_141_200_Lg1_ind = 300*L
jtl_141_200_b1_ic = 0.200
jtl_141_200_b1_icrn = 0.75 * icrn
jtl_141_200_b1_rsh = jtl_141_200_b1_icrn * Resi / jtl_141_200_b1_ic
jtl_141_200_b1_lprsh = jtl_141_200_b1_rsh*Lprsh
jtl_141_200_b0_ic = 0.141
jtl_141_200_b0_icrn = 0.75 * icrn
jtl_141_200_b0_rsh = jtl_141_200_b0_icrn * Resi / jtl_141_200_b0_ic
jtl_141_200_b0_lprsh = jtl_141_200_b0_rsh*Lprsh
jtl_141_200_L3_ind = 3.0*L
jtl_141_200_L1_ind = 1*L
jtl_141_200_L4_ind = 3.0*L
jtl_141_200_L0_ind = 14.4*L
jtl_141_200_L5_ind = 2.1*L
jtl_141_200_L6_ind = 2.1*L
jtl_141_200_L1d_ind = 1*L
clock_biases_L3_ind = 1*L
clock_biases_L2_ind = 1*L
clock_biases_L1_ind = 1*L
clock_biases_L0_ind = 1*L
xjtl_10jj_1p_Lg1_ind = 1*L
q_in_b0_ic = 0.141
```

```
q_in_b0_icrn = 0.75 * icrn
q_in_b0_rsh = q_in_b0_icrn * Resi / q_in_b0_ic
q_in_b0_lprsh = q_in_b0_rsh*Lprsh
q_in_Lb0_ind = 170*L
q_in_L1_ind = 200*L
q_in_K2_mut = 0.484934
q_in_L0_ind = 12*L
q_in_L2_ind = 93*L
launch1jj_R0_res = 50*Resh
launch1jj_L0_ind = 1*L
launch1jj_L1_ind = 1*L
terminator_R2_res = 50*Resh
terminator_R1_res = 50*Resh
terminator_R0_res = 5.3*Resh
R0_res = 50*Resh
.endc

.control
set noaskquit
set scaletype=single

* This is a generic run script.  All variables are contained in ohter
*  files, such as input.txt and netlist.param.  The most important part
*  of this file is to run the simulation and output the important
*  quantities.  The real critical part is inside "print.txt", which
*  specifices which junctions are recorded.  iplot is just a useful
*  tool to see that the simulation is going as expected.

*Global Parameters
L = 1.0
R = 1.0
jc = 1.0
icrn = 1.0
B = 1.0

* Individual Variables

input.txt
netlist.param
netlist.cir

* choose a smart pair to observe while the simulation runs, or comment
*  out to avoid graphs

iplot  v(I1:I2:b0:phi)*(-1/(2*pi)) v(I1:I4:b0:phi)*(-1/(2*pi))
```

```
1/(2*Vamp)*v(clk1in)
run
* plot  (-1/(2*pi))*v(I1:I2:b0:phi) (-1/(2*pi))*v(I1:I4:b0:phi)
        1/(2*Vamp)*v(clk1in)

set nopage
print.txt

exit

.endc
```

Appendix C

Wilkinson Power Splitter Response Parameters

C.1 Derivation of Impedance Values

Derivation of quarter-wave impedance values for impedance transformers is done in Pozar [39]. Here I follow the major steps.

C.1.1 Geometric response

By definition in the geometric response, the ratio of adjacent impedances is constant. This implicitly defines all reflection coefficients to be identical. For N stages between Z_0 and Z_L there are $N+1$ identical reflection coefficients. Thus the ratio between impedances is constant at $\sqrt[N+1]{Z_L/Z_0}$. Equation (4.3) follows directly.

C.1.2 Maximum Flat Response

"Maximum flat" in the name refers to the value of derivatives at the center frequency. The variation is to be a minimum, for which $\frac{d^n\Gamma}{d\rho^n}(\pi/2) = 0$ for $n = 0, 1, \ldots, N-1$. The function $\Gamma(\rho) = A(1 - e^{-i\,\rho})^N$ has this property. This must be matched to the form of the multi-segment quarter-wave filter $\sum_{n=0}^{N} \Gamma_n e^{-2ni\rho}$. If we expand the above equation, we get

$$\begin{aligned}\Gamma(\rho) = \Gamma_0 + \Gamma_1 e^{-2i\rho} + \Gamma_2 e^{-4i\rho} + \ldots &= \sum_{n=0}^{N} \Gamma_n e^{-2ni\rho} \\ &= A(1 - e^{-i\,\rho})^N = A \sum_{n=0}^{N} C_n^N e^{-2ni\rho} \end{aligned} \qquad \text{(C.1)}$$

We see that we can match terms of the form

$$\Gamma_n = AC_n^N. \qquad \text{(C.2)}$$

Boundary conditions define $A = 2^{-N} \frac{Z_L - Z_0}{Z_L + Z_0}$. Using the relation $\frac{1}{2} \ln x \approx \frac{x-1}{x+1}$ we write

$$\Gamma_n = \frac{Z_{n+1} - Z_n}{Z_{n+1} + Z_n} \approx \frac{1}{2} \ln \frac{Z_{n+1}}{Z_n} \approx AC_n^N \approx 2^{-(N+1)} C_n^N \ln \frac{Z_L}{Z_0}, \qquad \text{(C.3)}$$

and (4.4) follows. This also ensures that $Z_{N+1} = Z_{out}$.

C.1.3 Equal Ripple Response

This response is also known as the Chebyshev response. Chebyshev polynomials $T_n(x)$ have the property that $|T_n(x)| \leq 1$ for $-1 < x < 1$. If the frequency response is mapped to this region of the polynomial the reflection coefficient will be less than a given value over that region. The n^{th} Chebyshev polynomial is

$$T_n(x) = \cos(n \arccos x)$$

for $|x| < 1$. The cosine is the natural coordinate for the Chebyshev polynomial and we map $x \to \cos\rho / \cos\rho_m = S \cos\rho$ where ρ_m is the bandwidth boundary and $S = 1/\cos\rho_m$. Now $|T_n(S\cos\rho)| < 1$ for $\rho_m < \rho < \pi - \rho_m$. Much as in (C.1) we now expand in terms of the cosine and match parameters.

$$\begin{aligned}\Gamma(\rho) &= 2e^{-iN\rho}[\Gamma_0 \cos(N\rho) + \Gamma_1 \cos((N-2)\rho) + \dots \\ &= Ae^{-iN\rho} T_N(S\cos\rho). \end{aligned} \tag{C.4}$$

By definition I set $|\Gamma(\rho_m)| = A$ where A is chosen to give the maximum reflection amplitude (within the bandwidth). This boundary condition gives

$$S = \cosh\left[\frac{1}{N}\text{arccosh}\left|\frac{\ln Z_L/Z_0}{2A}\right|\right]$$

and all that remains is to expand the Chebyshev polynomial. This is non-trivial and I consider only the example at hand, $N = 6$ and $A = 0.05$. Then:

$$\begin{aligned}\Gamma(\rho) &= Ae^{-iN\rho} T_N(S\cos\rho) \\ &= Ae^{-iN\rho}(1 + 18S^2\cos^2\rho - 48S^4\cos^4\rho + 32S^6\cos^6\rho)\end{aligned} \tag{C.5}$$

I use the relation $\cos^n\rho = \frac{1}{2^n}C^n_{n/2} + \frac{2}{2^n}\sum_{k=0}^{\frac{n}{2}-1} C^n_k \cos((n-2k)\rho)$ to reduce the powers of the cosines and get:

$$\begin{aligned}\Gamma(\rho) = Ae^{-iN\rho}\times(&-1 + 9\,S^2 - 18\,S^4 + 10\,S^6 + \\ &3\,S^2(3 - 8\,S^2 + 5\,S^4)\cos 2\rho + \\ &6\,S^4(-1 + \,S^2)\cos 4\rho + \\ &S^6\cos 6\rho)\end{aligned} \tag{C.6}$$

We get (4.5) if we identify

$$\gamma_0 = \frac{A}{2}(\,S^6)$$
$$\gamma_1 = \frac{A}{2}(6\,S^4(-1+\,S^2))$$
$$\gamma_2 = \frac{A}{2}(3\,S^2(3-8\,S^2+5\,S^4))$$
$$\gamma_3 = 2\times\frac{A}{2}(-1+9\,S^2-18\,S^4+10\,S^6)$$
$$\gamma_4 = \gamma_2$$
$$\gamma_5 = \gamma_1$$
$$\gamma_6 = \gamma_0$$

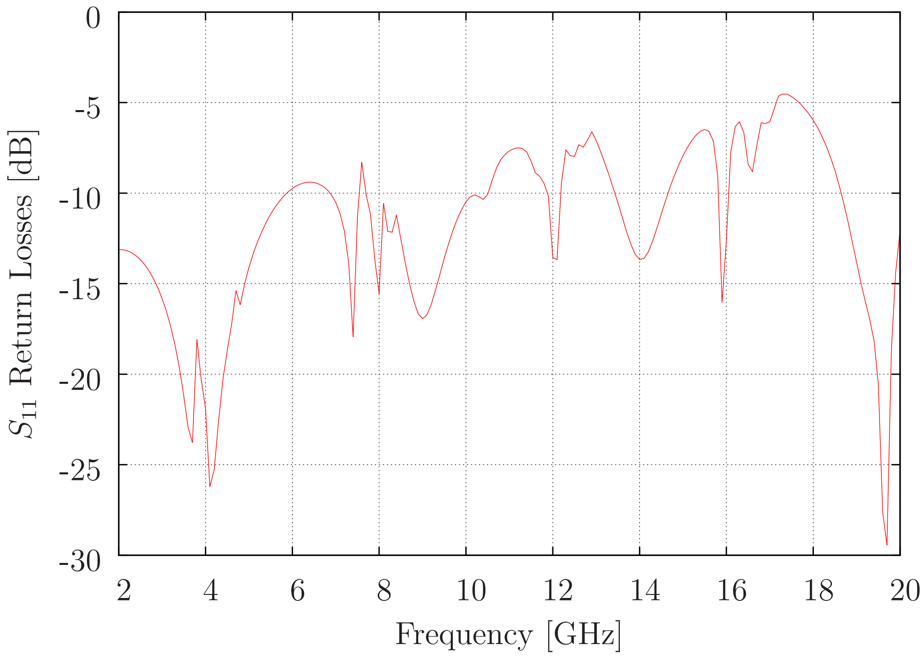


Figure C.1: Simulated Probe PCB Losses. Simulated S_{11} parameters of the printed circuit board on the Hypres 40-pin probe used for measurements.

C.2 HPI Probe Internal Reflections

Figure C.2 shows the simulated S_{11} parameter of one of the printed circuit board's connectors inside the probe. Above 6 GHz the reflection is above -10 dB with a fair amount of structure. Though this does not correlate with any particular features of measured S-parameters, it does demonstrate that the design and fabrication of the Wilkinson power splitter is not the only significant factor in the performance of RQL circuits.

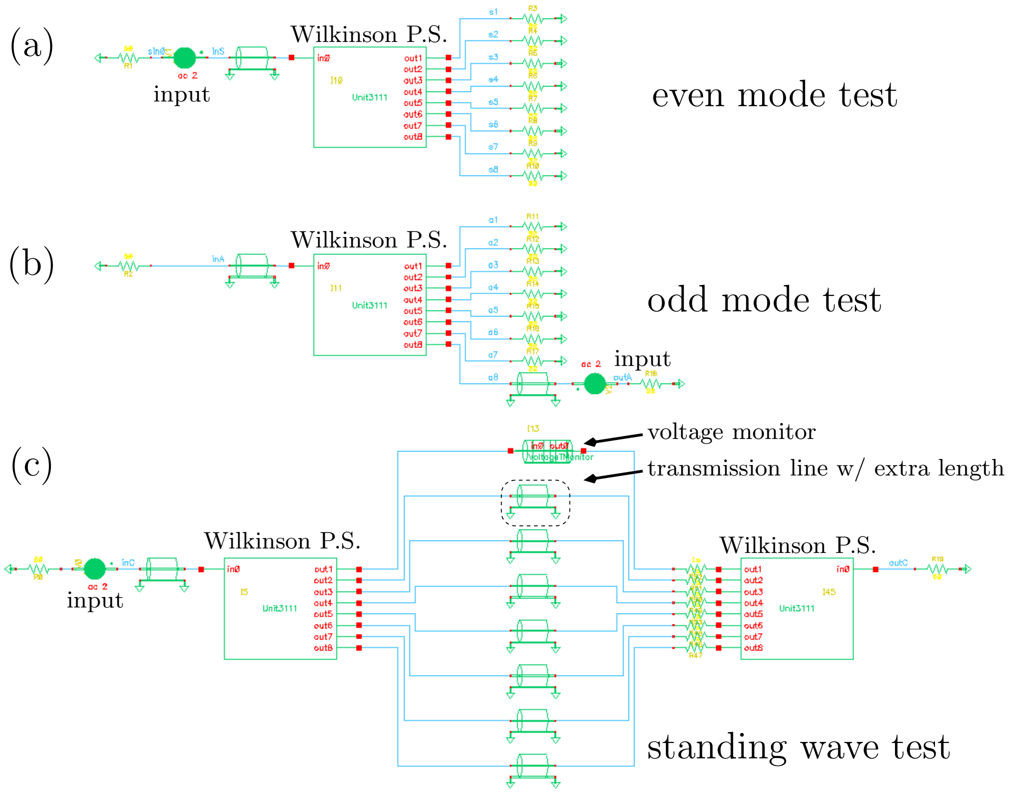


Figure C.2: S-Parameter Test Circuits.

C.3 Netlist for simulation of S-Parameters

Figure C.2 shows a block diagram of the circuits used to obtain the S-parameters shown in Figs. 4.7, 4.8, and 4.11, and the standing wave current shown in Figs. 4.13 and 4.14.

```
* HNL Generated netlist of FullCompare

.global 0
* MODEL Declarations
* Found stopping cell - V
* Found stopping cell - resistor
* Found stopping cell - tline
* Found stopping cell - lp
* End MODEL Declarations
```

```
.subckt voltagetmonitor in0 out0
L0 vl1 net028 1p
L4 vl5 net5 1p
L1 vl2 net032 1p
L2 vl3 net036 1p
L3 vl4 net015 1p
L5 vl6 net048 1p
L6 vl7 net052 1p
L7 vl8 net056 1p
L8 vl9 net07 1p
L9 net04 out0 1p
T8 net056 0 vl9 0 Z0=32 td=11p
T9 net07 0 net04 0 Z0=32 td=11p
T4 net015 0 vl5 0 Z0=32 td=11p
T5 net5 0 vl6 0 Z0=32 td=11p
T7 net052 0 vl8 0 Z0=32 td=11p
T6 net048 0 vl7 0 Z0=32 td=11p
T2 net032 0 vl3 0 Z0=32 td=11p
T3 net036 0 vl4 0 Z0=32 td=11p
T1 net028 0 vl2 0 Z0=32 td=11p
T0 in0 0 vl1 0 Z0=32 td=11p
.ends voltagetmonitor

.subckt unit3111 in0 out1 out2 out3 out4 out5 out6 out7 out8
T9 net77 0 out1 0 Z0=33.36 td=25p
T11 net57 0 out3 0 Z0=33.36 td=25p
T13 net73 0 out5 0 Z0=33.36 td=25p
T15 net53 0 out7 0 Z0=33.36 td=25p
T16 net53 0 out8 0 Z0=33.36 td=25p
T14 net73 0 out6 0 Z0=33.36 td=25p
T12 net57 0 out4 0 Z0=33.36 td=25p
T10 net77 0 out2 0 Z0=33.36 td=25p
T4 net69 0 net45 0 Z0=19.36 td=25p
T3 net69 0 net49 0 Z0=19.36 td=25p
T8 net45 0 net53 0 Z0=21.44 td=25p
T6 net49 0 net57 0 Z0=21.44 td=25p
T0 in0 0 net61 0 Z0=47.94 td=25p
T1 net61 0 net65 0 Z0=37.24 td=25p
T2 net65 0 net69 0 Z0=20.72 td=25p
T7 net45 0 net73 0 Z0=21.44 td=25p
T5 net49 0 net77 0 Z0=21.44 td=25p
R24 0 out8 5G
R17 0 out1 5G
R18 0 out2 5G
R19 0 out3 5G
```

```
R21 0 out5 5G
R20 0 out4 5G
R22 0 out6 5G
R23 0 out7 5G
R13 out2 out1 58.62
R14 out4 out3 58.62
R16 out8 out7 58.62
R15 out6 out5 58.62
R6 net45 net49 23.56
R11 net57 net77 20.62
R12 net53 net73 20.62
R0 0 in0 5G
R1 0 net61 5G
R2 0 net65 5G
R3 0 net69 5G
R4 0 net49 5G
R8 0 net57 5G
R7 0 net77 5G
R10 0 net53 5G
R9 0 net73 5G
R5 0 net45 5G
.ends unit3111

XI13 net0108 net073 voltagetmonitor
T7 net0104 0 net072 0 Z0=32 td=90p
T8 net064 0 net070 0 Z0=32 td=90p
T10 net068 0 net080 0 Z0=32 td=90p
T9 net0102 0 net065 0 Z0=32 td=90p
T5 net076 0 net088 0 Z0=32 td=90p
T6 net0105 0 net092 0 Z0=32 td=90p
T4 net084 0 net096 0 Z0=32 td=90p
T3 a8 0 net055 0 Z0=50 td=250p
T0 inc 0 net059 0 Z0=50 td=250p
T2 ina 0 net095 0 Z0=50 td=250p
T1 ins 0 net043 0 Z0=50 td=250p
XI45 outc net0208 net087 net079 net083 net063 net067 net075 net071 unit3111
XI5 net059 net0108 net084 net076 net0105 net0104 net064 net0102 net068 unit3111
XI11 net095 a1 a2 a3 a4 a5 a6 a7 a8 unit3111
XI10 net043 s1 s2 s3 s4 s5 s6 s7 s8 unit3111
V2 net055 outa ac 2
V0 inc net0136 ac 2
V1 ins sin0 ac 2
R19 0 outc 50
R53 net063 net072 1n
R52 net0208 net073 1n
```

```
R51 net087 net096 1n
R50 net079 net088 1n
R49 net083 net092 1n
R48 net075 net065 1n
R47 net071 net080 1n
R46 net067 net070 1n
R18 0 outa 32
R0 0 net0136 50
R2 0 ina 50
R11 0 a1 32
R12 0 a2 32
R14 0 a4 32
R13 0 a3 32
R17 0 a7 32
R16 0 a6 32
R15 0 a5 32
R7 0 s5 32
R8 0 s6 32
R10 0 s8 32
R9 0 s7 32
R5 0 s3 32
R6 0 s4 32
R4 0 s2 32
R3 0 s1 32
R1 0 sin0 50
.ac dec 800 100Meg 20G

.end
```

Appendix D

Fitting Functions for Race Circuit Experiments

D.1 Two-Output Fitting Code

The following `gnuplot` code was used to perform the fit of the Experiment 1 data to the predictions. Comments in the code explain the details.

```
## This file performs the plotting of data from Experiment 2
## First do some graphical stuff
reset
set sample 300
## define a convenient rounding function
round(x) = \
(floor(x*10**(4-floor(log10(x))-1)))*1.0/10**(4-floor(log10(x))-1)

## correction term for asymmetric attenuation in experiment
zz = 3.35
## adjusting for zero phase
zot = 5.1
ll = 0

## adjusting for zero phase refence
offset1 = zot/2
offset2 = zot/2
offset3 = zot/2
offset4 = zot/2

## Correction factors from Chapter 3
a1(x) = -0.029*x+1.1214
a2(x) = 0.0147*x+0.9798
a3(x) = -0.5305*x+6.52333

## graphical options
unset key
set samples 600

## Define physical constants
```

```
Phi0 = 2.07/1000
ICRN = 0.75
nn = 40
t0 = 3*nn*Phi0/(2*pi*ICRN)

## define frequencies and delta values
f1 = 1.0
f2 = 1.5
f3 = 2.0
f4 = 2.5
dltA1 = 2*pi*f1*t0
dltA2 = 2*pi*f2*t0
dltA3 = 2*pi*f3*t0
dltA4 = 2*pi*f4*t0

# upside down predictions
amp1(x) = dltA1/(1+cos(a2(f1)*x))
amp2(x) = dltA2/(1+cos(a2(f2)*x))
amp3(x) = dltA3/(1+cos(a2(f3)*x))
amp4(x) = dltA4/(1+cos(a2(f4)*x))

## fit data to curves
fit [ll:pi] amp1(x+zx1)+zy1 \
"dat_andout_10.csv" u ($1-offset1):(1.0*10**(-($3-$2)/20.0*zz))\
via zx1, zy1
fit [ll:pi] amp2(x+zx2)+zy2 \
"dat_andout_15.csv" u ($1-offset2):(1.0*10**(-($3-$2)/20.0*zz))\
via zx2, zy2
fit [ll:pi] amp3(x+zx3)+zy3 \
"dat_andout_20.csv" u ($1-offset3):(1.0*10**(-($3-$2)/20.0*zz))\
via zx3, zy3
fit [ll:pi] amp4(x+zx4)+zy4 \
"dat_andout_25b.csv" u ($1-offset4):(1.0*10**(-($3-$2)/20.0*zz))\
via zx4, zy4

## record results
set print "and_out.tex"
print " & $\\theta_0$ & $A_0$ \\\\ \\hline"
print f1,  " \& ",  round(zx1),   " \& ",  round(zy1),  "\\\\"
print f2,  " \& ",  round(zx2),   " \& ",  round(zy2),  "\\\\"
print f3,  " \& ",  round(zx3),   " \& ",  round(zy3),  "\\\\"
print f4,  " \& ",  round(zx4),   " \& ",  round(zy4),  "\\\\"

## graphical options
set xrange [ll:2.5]
set yrange [0:1]
set xlabel "Phase (rad)"
set ylabel "Current ($I_b / I_c$)"
PNTP = 1
LNTP = 2
set label "1.0 GHz" at 1, amp1(1)-0.025
set label "1.5 GHz" at 1, amp2(1)-0.025
set label "2.0 GHz" at 1, amp3(1)-0.025
set label "2.5 GHz" at 1, amp4(1)-0.025

plot \
"dat_andout_10.csv" u \
($1-offset1):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 1, \
"dat_andout_15.csv" u \
($1-offset2):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 2, \
"dat_andout_20.csv" u \
($1-offset3):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 3, \
"dat_andout_25b.csv" u \
($1-offset4):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 4, \
amp1(x) w l lt 1 lw 2 lc 1, \
amp2(x) w l lt 1 lw 2 lc 2, \
amp3(x) w l lt 1 lw 2 lc 3, \
amp4(x) w l lt 1 lw 2 lc 4, \
amp1(x+zx1)+zy1 w l lt LNTP lc 1, \
amp2(x+zx2)+zy2 w l lt LNTP lc 2, \
amp3(x+zx3)+zy3 w l lt LNTP lc 3, \
amp4(x+zx4)+zy4 w l lt LNTP lc 4

plot \
"dat_andout_10.csv" u \
($1-offset1):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 1, \
"dat_andout_15.csv" u \
($1-offset2):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 2, \
"dat_andout_20.csv" u \
($1-offset3):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 3, \
"dat_andout_25b.csv" u \
($1-offset4):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 4, \
amp1(x) w l lt 1 lw 2 lc 1, \
amp2(x) w l lt 1 lw 2 lc 2, \
amp3(x) w l lt 1 lw 2 lc 3, \
amp4(x) w l lt 1 lw 2 lc 4
set term epslatex color
set output "fig_andout.tex"
replot
unset label
set label "1.0 GHz" at 1, amp1(1+zx1)-0.025+zy1
set label "1.5 GHz" at 1, amp2(1+zx2)-0.025+zy2
set label "2.0 GHz" at 1, amp3(1+zx3)-0.025+zy3
set label "2.5 GHz" at 1, amp4(1+zx4)-0.025+zy4
set output "fig_andout_detail.tex"

plot \
"dat_andout_10.csv" u \
($1-offset1):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 1, \
"dat_andout_15.csv" u \
($1-offset2):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 2, \
"dat_andout_20.csv" u \
($1-offset3):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 3, \
"dat_andout_25b.csv" u \
($1-offset4):((1.0*10**(-($3-$2)/20.0*zz))):\
((1.0*10**(-($3-$2)/20.0*zz)))*log(10)/100 \
w yerrorbars pt PNTP ps 2 lt 2 lc 4, \
amp1(x+zx1)+zy1 w l lt LNTP lc 1, \
amp2(x+zx2)+zy2 w l lt LNTP lc 2, \
amp3(x+zx3)+zy3 w l lt LNTP lc 3, \
amp4(x+zx4)+zy4 w l lt LNTP lc 4

set term x11
```

```
system "epstopdf fig_andout.eps"
system "epstopdf fig_andout_detail.eps"
```

D.2 Fit to Experiment 2 Data

Unlike the results of Experiment 1, in Chapter 5, Experiment 2 was not fit to the expected results. Figure D.1 shows an alternative to Fig. 5.9. Figure 5.9 plotted the measured minimum bias current as a function of input phase and showed the predictions based on (5.3c). Here, I use the same data but allow the curve to be fit to the data. The dashed lines are fits made to the points of the equation

$$A_{lim} = \frac{2\pi f N \mathrm{t}_0}{1+\cos(\theta+\tilde{\theta})} + \tilde{A}, \tag{D.1}$$

where $\tilde{\theta}$ and $\tilde{A}$ are translation parameters which do not alter the shape of the function. The parameters $\tilde{\theta}$ and $\tilde{A}$ allowed for shifting in amplitude, and phase. The values are shown in Table D.1. Phase was again unknown but quantifiable. Amplitude was known to within errors, assuming full symmetry of the circuit from the clock splitter onward. With this fit the same general behavior is persevered. There is a tradeoff in comparison with the unfitted predictions. Though the lower current amplitude limit is now closer to expectations, the phase limit predictions no longer match as closely. This alternative analysis indicates that while the intermediate behavior may not follow predictions closely, limits on phase and clock amplitude are reliable.

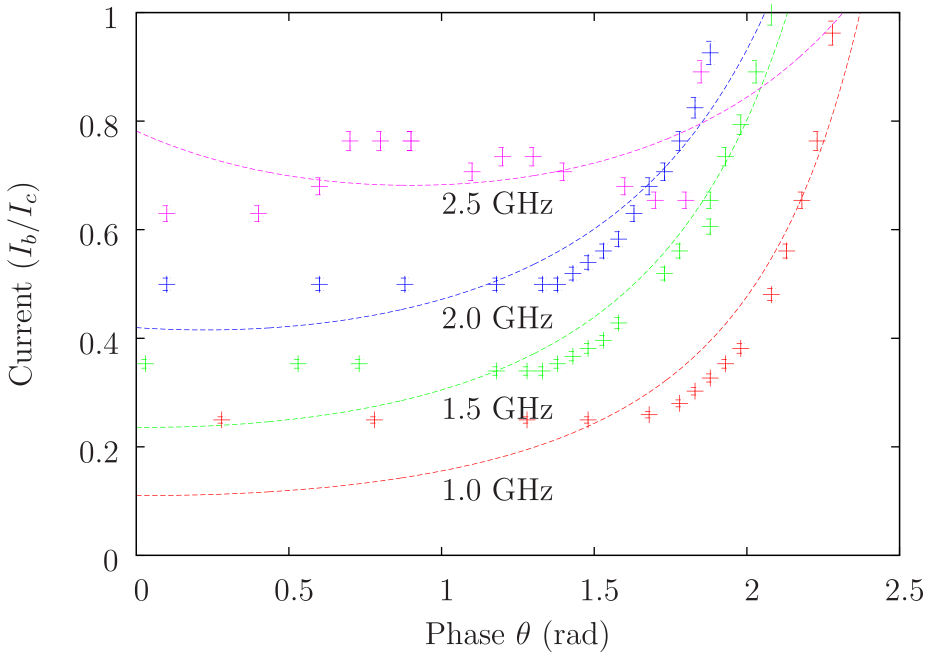


Figure D.1: Curve fitting to Experiment 2. Minimum clock amplitude I_b plotted as a function of input phase θ. The original analysis of Experiment 2 shows the same data without fitted curves. This figure shows the results of fitting the predicted curves to the data by translation along the axes.

Table D.1: Fitting parameters of and-output circuit data. $\tilde{\theta}$ and $\tilde{A}$ are as defined in (D.1); they are a shift in the reference point of the start of the clock phase and the critical current I_c, respectively. An alternate fitting of the data from Experiment 2 to the data shifts the curves by only small amounts, except for the highest frequency, 2.5 GHz. The data recorded for 2.5 GHz does not follow the predicted qualitative.

f [GHz]	$\tilde{\theta}$	$\tilde{A}$
1.0	-0.03085	-0.05538
1.5	-0.03194	-0.01255
2.0	-0.2251	0.0841
2.5	-0.8992	0.2677

D.3 And-Output Fitting Code for gnuplot

```
## This code generates the plots displaying information
## from the and-output race circuit experiment.
## First take care of some graphical stuff.
reset
set key bottom left
set samples 600
unset key
set format y ""
set arrow from 0.5, 100 to 0.4,100 lt -1 nohead
set arrow from 0.5, 120 to 0.4,120 lt -1 nohead
set arrow from 0.5, 100 to 0.5,120 lt -1 nohead
set label at 0.6, 110 "20 ps"
set xlabel "$\\theta$ [rad]"
set ylabel "$\\Delta t$ [ps]"

## This defines a convenient rounding function.  It is cosmetic only.
round(x) = 1e-4*floor(x*1e4)

## Here I define some important physical quantities.
Phi0 = 2.07 # Magnetic flux quantum
d0 = pi*1*Phi0/0.75/0.7/1000 # values of delta
d8 = 8*d0
d20= 20*d0

## Start the fitting parameters for the RED curve
## at the values for the BLUE curve.
## 0.5 GHz
g1 = -2.1259
g2 = 48.1487
g3 = 0.1913
g4 = 1.0187

## 1.0 GHz
k1 = -2.4101
k2 = 67.2539
k3 = 1.4976
k4 = 1.2733

## 1.5 GHz
l1 = -2.5474
l2 = 49.672
l3 = 0.1178
l4 = 0.4804

## record original values for comparison
Ag1 = g1
```

```
Ag2 = g2
Ag3 = g3
Ag4 = g4

Ak1 = k1
Ak2 = k2
Ak3 = k3
Ak4 = k4

Al1 = l1
Al2 = l2
Al3 = l3
Al4 = l4

## Here I define the delay functions (RED curve)
n0(x) = g4*(acos(cos(x) - d0) - x)/(2*pi/1000)
n8(x) = g4*(acos(cos(x) - g3*d8) - x)/(2*pi/1000)
n20(x)= g4*(acos(cos(x) - g3*d20) - x)/(2*pi/1000)

m0(x) = k4*(acos(cos(x) - d0*0.5) - x)/(2*pi/500)
m8(x) = k4*(acos(cos(x) - k3*d8*0.5) - x)/(2*pi/500)
m20(x)= k4*(acos(cos(x) - k3*d20*0.5) - x)/(2*pi/500)

h0(x) = l4*(acos(cos(x) - d0*1.5) - x)/(2*pi/1500)
h8(x) = l4*(acos(cos(x) - l3*d8*1.5) - x)/(2*pi/1500)
h20(x)= l4*(acos(cos(x) - l3*d20*1.5) - x)/(2*pi/1500)

## Here I define the delay functions (BLUE curve)
An0(x) = Ag4*(acos(cos(x) - d0) - x)/(2*pi/1000)
An8(x) = Ag4*(acos(cos(x) - Ag3*d8) - x)/(2*pi/1000)
An20(x)= Ag4*(acos(cos(x) - Ag3*d20) - x)/(2*pi/1000)

Am0(x) = Ak4*(acos(cos(x) - d0*0.5) - x)/(2*pi/500)
Am8(x) = Ak4*(acos(cos(x) - Ak3*d8*0.5) - x)/(2*pi/500)
Am20(x)= Ak4*(acos(cos(x) - Ak3*d20*0.5) - x)/(2*pi/500)

Ah0(x) = Al4*(acos(cos(x) - d0*1.5) - x)/(2*pi/1500)
Ah8(x) = Al4*(acos(cos(x) - Al3*d8*1.5) - x)/(2*pi/1500)
Ah20(x)= Al4*(acos(cos(x) - Al3*d20*1.5) - x)/(2*pi/1500)

## This fit finds the middle starting port for the red curve fit.
fit n20(x+g1)-n8(x+g1)+g2 "dat_twoout.dat" u 1:2:3 via g1
fit m20(x+k1)-m8(x+k1)+k2 "dat_twoout2.dat" u 1:2:3 via k1
fit h20(x+l1)-h8(x+l1)+l2 "dat_twoout3.dat" u 1:2:3 via l1
```

```
## Now I perform the actual fitting of the data.
fit [-g1:6] n20(x+g1)-n8(x+g1)+g2 "dat_twoout.dat" \
u 1:2:3 via g4, g2, g1, g3
fit [-k1:6] m20(x+k1)-m8(x+k1)+k2 "dat_twoout2.dat" \
u 1:2:3 via k4, k2, k1, k3
fit [-l1:6] h20(x+l1)-h8(x+l1)+l2 "dat_twoout3.dat" \
u 1:2:3 via l4, l2, l1, l3

## define a new function to display the fit only over the fitting region.
## This is entirely cosmetic
trun1(x) = x > -g1 ? (n20(x+g1)-n8(x+g1))+g2 : 1/0
trun2(x) = x > -k1 ? (m20(x+k1)-m8(x+k1))+k2 : 1/0
trun3(x) = x > -l1 ? (h20(x+l1)-h8(x+l1))+l2-25 : 1/0

## cosmetic appearance
set yrange [20:130]
set label at 5.2, 120 "0.5 GHz"
set label at 5.2, 80  "1.0 GHz"
set label at 5.2, 52  "1.5 GHz"
# set border 3
set xtics nomirror
set ytics nomirror

## plot the results
plot \
"dat_twoout.dat" u 1:2:($3/2) w yerrorbars ls 0 pt 7, \
"dat_twoout2.dat" u 1:($2+10):($3/2) w yerrorbars ls 0 pt 7, \
"dat_twoout3.dat" u 1:($2-25):($3/2) w yerrorbars ls 0 pt 7, \
trun1(x) ls 1 lw 3, \
trun2(x)+10 ls 1 lw 3, \
trun3(x) ls 1 lw 3, \
(An20(x+Ag1)-An8(x+Ag1))+Ag2 ls 3 lw 1, \
(Am20(x+Ak1)-Am8(x+Ak1))+Ak2+10 ls 3 lw 1, \
(Ah20(x+Al1)-Ah8(x+Al1))+Al2-25 ls 3 lw 1

## Output the data in a useful fashion.
pr ''
pr "g1 = ", round(g1), "   g2 = ", round(g2), "   \
g3 = ", round(g3), "   g4 = ", round(g4)
pr "k1 = ", round(k1), "   k2 = ", round(k2), "   \
k3 = ", round(k3), "   k4 = ", round(k4)
pr "l1 = ", round(l1), "   l2 = ", round(l2), "   \
l3 = ", round(l3), "   l4 = ", round(l4)
pr ''
```

```
set print "two_out.tex"

pr " &  & 0.5 GHz & 1.0 GHz & 1.5 GHz \\\\ \\hline"
pr "\\multirow{4}{*}{Red Fit} & $\\gamma_1$ & ", \
round(1.0/g3), " & ", round(1.0/k3), " & ", \
round(1.0/l3), " \\\\"
pr " & $\\gamma_2$ & ", round(g1), " &", round(k1), \
" & ", round(l1), " \\\\"
pr " & $\\gamma_3$ & ", round(g2), " & ", round(k2), \
" &  ", round(l2), " \\\\"
pr " & $\\gamma_4$ & ", round(g4/g3), " & ", \
round(k4/k3), " & ", round(l4/l3), " \\\\ \\hline"


pr "\\multirow{4}{*}{Blue Fit} & $\\gamma_1$ & ", \
round(1.0/Ag3), " & ", round(1.0/Ak3), " & ", \
round(1.0/Al3), " \\\\"
pr " & $\\gamma_2$ & ", round(Ag1), " &", round(Ak1), \
" & ", round(Al1), " \\\\"
pr " & $\\gamma_3$ & ", round(Ag2), " & ", round(Ak2), \
" &  ", round(Al2), " \\\\"
pr " & $\\gamma_4$ & ", round(Ag4/Ag3), " & ", \
round(Ak4/Ak3), " & ", round(Al4/Al3), " \\\\ \\hline"

# set xrange [0:2*pi]

set term epslatex color
set output "fig_twoexp.tex"
replot
set term x11

system "epstopdf fig_twoexp.eps"

set term epslatex color standalone
set output "fig_twoexp_sa.tex"
replot
set term x11

set print ’-’
pr "latex fig_twoexp_sa.tex; dvips fig_twoexp_sa; \
epstopdf fig_twoexp_sa.ps"
```

D.4 Calculation of Depressed I_cR_N Product

Experiment 2 in Chapter 5 indicated that at a known calibration point, $I_b = I_c$, the measurement of the bias current through the junctions could confirm the power delivered to the chip. Results matched expectations, given that attenuators in the test setup are accurate only to approximately 10%. This reference was subsequently used in calculations for Experiment 3. When adjusted using this calibration, the measured I_cR_N product of the experimental chip was 0.770 mV.

Alternatively, I can try ignoring this correction and proceed with the calculations. The results are shown in Table D.2. In this case, the average value of the timing parameter t_0 is 0.77 ps and the average value of I_cR_N is 0.44 mV. Just as importantly, as frequency increases the lower bias current limit increases to close to $A = 1$, which one would expect close to the highest operating frequency. Assuming a depression of the I_cR_N product by almost one-half, the maximum frequency of the circuits in the experiment would decrease by the same factor. This fits observations that the maximum operating frequency of the experiments was 2.5 GHz, despite a design for a maximum frequency of 5 GHz.

While the I_cR_N product is generally constant for a given process, it can effectively be changed by a different β_c value. The β_c value is determined both by design and fabrication. The value of the shunt resistor R_N can be chosen to essentially scale the I_cR_N product and change the switching time of junctions. However, misalignment during the fabrication process can change the resistor values from the design, thus slowing down the junctions. A test of the chip fabrication properties requires specialized circuits, and the results can depend greatly on the location of the chip on the wafer. This alternative explanation is given to account for different possible reasons for the low maximum frequency instead of only a shift in current values from those measured before calibration.

Table D.2: Alternative switching time calculation results from the long, deep pipeline shift register experiment. (See Chapter 5, particularly Table 5.4, pg. 189.) Average timing parameter t_0 is 0.77 ps. Average value of I_cR_N is 0.44 mV. Both averages have a spread of about 20%. The value of I_cR_N used for design was assumed to be 0.75 mV. The correction factor from Experiment 2 was *not* applied in this case. The lower I_cR_N value explains the lower maximum operating frequency of the circuits.

f	1.0	1.5	2.0	2.5	GHz
P_{in}	4.500	4.500	4.500	4.500	dBm
P_{out}	1.800	2.400	2.700	4.100	dBm
P_{chip}	2.700	2.100	1.800	0.400	dBm
I_b/I_c	0.537	0.617	0.661	0.912	
t_0	1.007	0.771	0.620	0.684	ps
I_cR_N	0.325	0.425	0.529	0.479	mV

Appendix E

Hypres Fabrication Summary

The experiments described in this thesis were performed on chips manufactured by Hypres, Inc. This appendix briefly summarizes the design rules for this process. The Hypres niobium integrated circuit fabrication design rules are available on the Hypres website [29]. The designs described here follow the 24th revision, dated January 11, 2008.

Niobium is the superconducting material in the Hypres integrated circuits. The junctions are Niobium/Aluminum-Oxide/Niobium SIS Josephson junctions fabricated using an *in-situ* trilayer over the entire wafer. Junction areas are created through photolithography and etching of the trilayer. The photolithography does not employ any size reduction. All my integrated circuits used the $4.5\,\mathrm{kA/cm^2}$ critical current density process. There are four superconducting niobium metal layers. Josephson junctions are connected between the second and third metallization layers. Junctions are shunted by normal metal in a separate molybdenum normal metal layer. The molybdenum layer has a sheet resistance of $2.1 \pm 0.3\,\Omega$ per square. The metal layers are insulated with silicon dioxide. Josephson junctions are additionally insulated by anodization of the base electrode of the trilayer. The specific capacitance of junctions is approximately $59\,\mathrm{fF/\mu m^2}$ for the $4.5\mathrm{kA/cm^2}$ process.

Fabrication is done on 6-inch (150 mm) diameter oxidized silicon wafers. The

Hypres design rules specify a number of constraints on design of integrated circuits and the accuracy of circuit elements in fabrication. These can be found in the published design rules. The physical design of integrated circuits is shown in Table E.1. This includes 11 process layers, a minimum feature size of 1 μm, and current density tolerance and resistor tolerances of ±5% on chip and ±15% between runs.

The maximum microstrip impedance using this process is $42\,\Omega$, using M0 as the signal layer, M3 as the ground plane. In this case, the width of the microstrip is 2.5 μm and the spacing between the M0 signal and M0 ground is 2.5 μm. Due to fabrication bias of -0.2 μm, the actual fabricated width of the microstrip is about 2.3 μm wide.

Table E.1: Hypres fabrication design specifications. Taken from Hypres Design Rules. Some thicknesses not given or not applicable (na/ng)

Layer Name	Thickness	Description
(none)	*na/ng*	Niobium deposition
M0	100 ± 10 µm	M0 paterning (holes in niobium ground plane)
(none)	*na/ng*	SiO_2 deposition
I0	150 ± 15 µm	Contact (via) between M1 and ground plane
(none)	*na/ng*	Niobium / Aluminum Oxide / Niobium trilayer deposition
I1C	50 ± 5 µm	Counter-electrode (junction area) definition
(none)	*na/ng*	Base electrode anodization
AI	40 ± 5 µm	Anodization layer patterning
M1	135 ± 10 µm	Trilayer base electrode patterning
(none)	100 ± 10 µm	SiO_2 deposition
(none)	*na/ng*	Resistive layer deposition
R2	*na/ng*	Resistor patterning
(none)	100 ± 10 µm	SiO_2 deposition
I1B	*na/ng*	Contact (via) between M2 and (I1A, R2, or M1)
(none)	*na/ng*	Nb deposition
M2	300 ± 20 µm	M2 layer patterning
(none)	500 ± 40 µm	SiO2 deposition
I2	*na/ng*	Contact (via) between M2 and M3
(none)	*na/ng*	Nb deposition
M3	600 ± 50 µm	M3 layer patterning
(none)	*na/ng*	Ti/Pd/Au contact metallization deposition
R3	350 ± 60 µm	Contact pad patterning

Appendix F

Spice Netlist of CLA

```
* HNL Generated netlist of AN_chip21

.global 0
* MODEL Declarations
* Found stopping cell - lm_open
* Found stopping cell - junction
* Found stopping cell - muind
* Found stopping cell - rsj
.subckt rsj a b jjmod=jj110D ic=0.25 icrn=0.7 rsh=2.8 lprsh=1.5p
b0 a b phi jjmod area=ic
r_sh a a1 rsh
lp_rsh a1 b lprsh
.ends rsj
* Found stopping cell - lp
* Found stopping cell - inductor
* Found stopping cell - resistor
* End MODEL Declarations

.subckt on_input_wireup ci1 ci2 ci3 ci4 ci5 ci6 ci7 ci8 cq1 cq2 cq3
cq4 cq5 cq6 cq7 cq8 wi1 wi2 wi3 wi4 wi5 wi6 wi7 wi8
wq1 wq2 wq3 wq4 wq5 wq6 wq7 wq8

L15 wq1 cq1 1p
L14 wq2 cq2 1p
L13 wq3 cq3 1p
L12 wq4 cq4 1p
L11 wq5 cq5 1p
L10 wq6 cq6 1p
L9 wq7 cq7 1p
L8 wq8 cq8 1p
L7 wi8 ci8 1p
L6 wi7 ci7 1p
L5 wi6 ci6 1p
L4 wi5 ci5 1p
L3 wi4 ci4 1p
L2 wi3 ci3 1p
L1 wi2 ci2 1p
```

```
L0 wi1 ci1 1p
.ends on_input_wireup

.subckt on_output_wireup ci1 ci2 ci3 ci4 ci5 ci6 ci7 ci8 cq1 cq2 cq3
cq4 cq5 cq6 cq7 cq8 wi1 wi2 wi3 wi4 wi5 wi6 wi7 wi8 wq1 wq2 wq3 wq4
wq5 wq6 wq7 wq8

L15 wq1 cq1 1p
L14 wq2 cq2 1p
L13 wq3 cq3 1p
L12 wq4 cq4 1p
L11 wq5 cq5 1p
L10 wq6 cq6 1p
L9 wq7 cq7 1p
L8 wq8 cq8 1p
L7 wi8 ci8 1p
L6 wi7 ci7 1p
L5 wi6 ci6 1p
L4 wi5 ci5 1p
L3 wi4 ci4 1p
L2 wi3 ci3 1p
L1 wi2 ci2 1p
L0 wi1 ci1 1p
.ends on_output_wireup

.subckt on_50_ohm_wps_40pin p1 p2 p3 p4 p5 p6 p7 p8 uwin
R6 net49 net11 23.56
R5 net47 net13 20.62
R4 net43 net15 20.62
R3 p8 p7 58.62
R2 p6 p5 58.62
R1 p4 p3 58.62
R0 p2 p1 58.62
L16 net47 p8 1p
L15 net47 p7 1p
L14 net13 p6 1p
L13 net13 p5 1p
L12 net43 p4 1p
L11 net43 p3 1p
L10 net15 p2 1p
L9 net15 p1 1p
L8 net11 net15 1p
L7 net11 net43 1p
L6 net49 net13 1p
L5 net49 net47 1p
```

```
L4 net53 net49 1p
L3 net53 net11 1p
L0 uwin net53 1p
.ends on_50_ohm_wps_40pin

.subckt n_bias_out bias c0 c1 d0 d1
c0 c0 0 5.28f
K1 L1 L0 0.936529
K1d L1d L0 0.218218
L0 0 bias 1p
L1 c1 c0 13.18p
L1d d1 d0 5.25p
.ends n_bias_out

.subckt m_out_rql a b c c00 c01 c10 c11 dc0 dc1
L0 net72 net33 12.6p
L6 net41 net72 750f
L9 net57 net59 1.5p
Lf c10 c00 1f
L10 net59 net38 9.2p
L5 net61 b 962f
L4 net61 net59 962f
L3 net68 net59 750f
L2 a net68 750f
L8 net57 net70 2.56p
L7 net70 net72 1.05p
Lp0 net39 0 200f
Lc net41 c 2f
Lp4 net43 0 200f
Lp2 net45 0 200f
Lp3 net47 0 200f
Lp1 net49 0 200f
Xb2 net45 net57 rsj jjmod=Hyp5a ic=0.200 icrn=0.7 rsh=3.5 lprsh = 1.75p
Xb0 net39 net41 rsj jjmod=Hyp5a ic=0.400 icrn=0.35 rsh=875.000m lprsh = 437.5f
Xb4 net43 net61 rsj jjmod=Hyp5a ic=0.312 icrn=0.7 rsh=2.24359 lprsh = 1.121795p
Xb3 net47 net68 rsj jjmod=Hyp5a ic=0.400 icrn=0.7 rsh=1.75 lprsh = 875.000f
Xb1 net49 net70 rsj jjmod=Hyp5a ic=0.282 icrn=0.7 rsh=2.48227 lprsh = 1.241135p
XI14 net33 c11 net34 dc0 net37 n_bias_out
XI13 net38 net34 c01 net37 dc1 n_bias_out
.ends m_out_rql

.subckt m_out_squid a b c
c0 g 0 44f
c1 h 0 16f
c2 f 0 16f
```

```
c3 c f 5f
c4 d h 5f
c5 e bb 10f
K0 L0 L1 0.64
K1 L2 L3 0.64
Xb1 f g rsj jjmod=Hyp5a ic=0.168 icrn=0.42 rsh=2.5 lprsh = 1.25p
b0 h g phib0 Hyp5a area=0.168
L6 b bb 120p
L5 g a 120p
L7 c d 200f
L2 bb h 3.92p
L3 d e 5.17p
L1 e 0 5.17p
L0 f bb 3.92p
.ends m_out_squid

.subckt m_out12 a c00 c01 c10 c11 dc0 dc1 q
R1 net23 0 50
R0 0 net28 50
XI11 net77 net23 net33 net021 net022 c10 c11 dc0 net019 m_out_rql
XI10 net84 net77 net36 net030 net031 net021 net022 net019 net028 m_out_rql
XI9 net91 net84 net39 net033 net040 net030 net031 net028 net037 m_out_rql
XI8 net98 net91 net42 net042 net049 net033 net040 net037 net046 m_out_rql
XI7 net105 net98 net97 net051 net050 net042 net049 net046 net045 m_out_rql
XI6 net112 net105 net104 net066 net067 net051 net050 net045 net064 m_out_rql
XI5 net119 net112 net51 net069 net076 net066 net067 net064 net073 m_out_rql
XI4 net126 net119 net118 net078 net085 net069 net076 net073 net082 m_out_rql
XI3 net133 net126 net125 net087 net094 net078 net085 net082 net091 m_out_rql
XI2 net140 net133 net132 net096 net0103 net087 net094 net091 net0100 m_out_rql
XI1 net147 net140 net139 net0111 net0112 net096 net0103 net0100 net0109
                  m_out_rql
XI0 a net147 net146 c00 c01 net0111 net0112 net0109 dc1 m_out_rql
XI23 q net37 net33 m_out_squid
XI22 net37 net40 net36 m_out_squid
XI21 net40 net43 net39 m_out_squid
XI20 net43 net44 net42 m_out_squid
XI19 net44 net49 net97 m_out_squid
XI18 net49 net52 net104 m_out_squid
XI17 net52 net55 net51 m_out_squid
XI16 net55 net58 net118 m_out_squid
XI15 net58 net61 net125 m_out_squid
XI14 net61 net64 net132 m_out_squid
XI13 net64 net67 net139 m_out_squid
XI12 net67 net28 net146 m_out_squid
.ends m_out12
```

```
.subckt n_bias bias c0 c1 d0 d1
c0 c0 0 2.64f
L0 0 bias 1p
L1 c1 c0 6.59p
L1d d1 d0 5.25p
K1 L1 L0 0.662226
K1d L1d L0 0.218218
.ends n_bias

.subckt q_out400e_v a bias q
Xb1 net010 net050 rsj jjmod=Hyp5a ic=0.400 icrn=0.75 rsh=1.875 lprsh = 937.5f
Xb2 net035 net026 rsj jjmod=Hyp5a ic=0.200 icrn=0.42 rsh=2.1 lprsh = 1.05p
LPbias1 net011 0 1f
b3 net023 net030 phib3 Hyp5a area=0.141
Lg1 net050 0 200f
Lg2 net026 0 600f
Lg3 net023 net035 100f
Lg5 net035 q 100f
Lp net010 net012 55f
L7 net030 net011 3.5p
Lbias bias net010 9.3p
L5 net012 a 1.05p
L6 net012 net030 2.5p
.ends q_out400e_v

.subckt q_out282e a bias q
Lg0 net013 0 200f
Xb0 net08 net013 rsj jjmod=Hyp5a ic=0.282 icrn=0.75 rsh=2.659574
                        lprsh = 1.329787p
L3 a net08 1.5p
L4 net08 q 1.5p
Lbias bias net08 13.7p
.ends q_out282e

.subckt q_in a0 a1 q
Xb0 net020 net013 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                             lprsh = 2.659575p
Lb0 net020 0 170f
L1 net013 q 200f
K2 L2 L0 0.484934
L0 0 net013 12p
L2 a1 a0 93p
.ends q_in
```

```
.subckt q_c0_io a a0 a1 c01 c11 dc0 dc1 q qv
XI4 net14 c01 net11 net13 dc1 n_bias
XI3 net19 net11 c11 dc0 net13 n_bias
XI2 net20 net14 qv q_out400e_v
XI1 a net19 net20 q_out282e
XI0 a0 a1 q q_in
.ends q_c0_io

.subckt a_jtl_chop a bias q
Lg0 net013 0 220f
Lg1 net05 0 150f
L6 net014 q 1f
Xb1 net014 net05 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net021 net013 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                             lprsh = 2.659575p
L3 a net021 3.0p
L4 net021 net019 3.0p
Lbias bias net019 13.4p
L5 net019 net014 2.1p
.ends a_jtl_chop

.subckt n_bias_ihm bias c0 c1 d0 d1
c0 c0 0 1.79f
L0 0 bias 1p
L1 c1 c0 4.47p
L1d d1 d0 3.67p
K1 L1 L0 0.402036
K1d L1d L0 0.260998
.ends n_bias_ihm

.subckt a_anotb a b d0 d1 q
cb b 0 58.71f
ca a 0 91.30f
Lp0 net19 net023 96f
Lg1 net018 0 172f
Lg0 net027 0 106f
Xb0 net023 net027 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                             lprsh = 2.659575p
Xb1 net019 net018 rsj jjmod=Hyp5a ic=0.100 icrn=0.75 rsh=7.5 lprsh = 3.75p
K3 L10 L11 0.289
k05 L0 L5 0.0672226
k65 L5 L6 0.815404
L11 0 net031 1p
L10 d1 net044 1p
L3 b net019 2.1p
```

```
L7 net19 q 3.0p
L4 a net19 2.1p
L0 d0 net044 1p
L5 net019 0 18.355p
L6 net031 net023 19.913p
.ends a_anotb

.subckt a_jtle_chop a bias q
Lg0 net08 0 230f
Lg1 net010 0 150f
L6 net014 q 1f
Xb1 net014 net010 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net013 net08 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                         lprsh = 2.659575p
L3 a net013 3.0p
L4 net013 net019 2.67p
Lbias bias net019 11.7p
L5 net019 net014 2.43p
.ends a_jtle_chop

.subckt a_or a b bias qo
L5 net10 net029 20p
L6 net10 net62 20p
L3 net050 net44 20.5p
L4 net58 net44 20.5p
L8 net51 qo 3.0p
L7 net19 net20 23.5p
L9 net059 net034 15.2p
Lbias bias net085 17.4p
Xb0 net51 net29 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                   lprsh = 3.177966p
Xb1 net059 net41 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                         lprsh = 3.177966p
Lp1 net20 net059 671f
Lp2 net19 net44 200f
Lp0 net19 net085 277f
Lb0 net29 0 245f
Lb1 net41 0 62f
Lp5 a net050 387f
Lp4 net51 net085 11f
Lp6 b net58 442f
Lp3 net20 net10 630f
k35 L3 L5 0.78
k46 L4 L6 0.78
LPag net034 0 1f
```

```
LPqq net050 net62 1f
LPq net58 net029 1f
.ends a_or

.subckt a_and a b bias qa
LPqq net030 net62 1f
LPq net026 net028 1f
L4 net026 net034 20.5p
L6 net58 net62 20p
L5 net58 net028 20p
L3 net030 net034 20.5p
L7 net19 net20 23.9p
L9 net018 qa 3.0p
Lbias bias net036 17.4p
Xb0 net036 net019 rsj jjmod=Hyp5a ic=0.128 icrn=0.375 rsh=2.929688
                              lprsh = 1.464844p
Xb1 net018 net049 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                             lprsh = 3.177966p
Lp1 net20 net018 760f
Lp2 net19 net034 500f
Lp0 net19 net036 470f
Lg0 net019 0 170f
Lg1 net049 0 90f
Lp5 a net030 620f
Lp6 b net026 620f
Lp3 net20 net58 550f
k35 L3 L5 0.78
k46 L4 L6 0.78
.ends a_and

.subckt a_jtl_e a bias q
Lg0 net08 0 230f
Lg1 net010 0 230f
Xb1 net014 net010 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net022 net08 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                         lprsh = 2.659575p
L3 a net022 3.0p
L4 net022 net019 3.3p
Lbias bias net019 11.2p
L5 net019 net014 1.8p
L6 net014 q 2.1p
.ends a_jtl_e

.subckt a_c3_b3 a b bias15 bias16 c00 c01 c10 c11 dc0 dc1 dc2 dc3 g gl gm pm1 q
L0 net070 c11 1p
```

```
XI47 bias16 net075 net070 net0153 net085 n_bias
XI46 bias15 c01 net075 net0148 net056 n_bias
XI5 net31 net54 net55 a_jtl_chop
XI40 net33 net0150 net0145 net085 net0148 n_bias_ihm
XI43 net25 net0100 net0120 net0103 net0123 n_bias_ihm
XI48 net0101 net097 c00 net059 dc1 n_bias
XI44 net36 c10 net0115 dc0 net0118 n_bias
XI22 net0149 net0145 net097 net056 net059 n_bias
XI38 net0134 net090 net0150 net0133 net0153 n_bias
XI26 net54 net0120 net090 net0123 net0133 n_bias
XI27 net047 net0115 net0100 net0118 net0103 n_bias
Xanotb a b dc2 dc3 net013 a_anotb
XI2 gm net0134 net22 a_jtle_chop
XI3 gl net0149 net34 a_jtle_chop
XI4 pm1 net0101 net32 a_jtle_chop
Xor net22 net55 net25 net49 a_or
Xand1 net32 net34 net33 net31 a_and
XI1 net49 net36 g a_jtl_e
XI6 net013 net047 q a_jtl_e
.ends a_c3_b3

.subckt a_jtle_ a bias q
Lg0 net013 0 230f
Lg1 net05 0 230f
Xb1 net023 net05 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net021 net013 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                            lprsh = 2.659575p
L3 a net021 3.0p
L4 net021 net037 2.67p
Lbias bias net037 11.7p
L5 net037 net023 2.43p
L6 net023 q 2.1p
.ends a_jtle_

.subckt a_and011 a b bias qa
LPqq net053 net62 1f
LPq net034 net029 1f
L4 net034 net051 20.5p
L6 net58 net62 20p
L5 net58 net029 20p
L3 net053 net051 20.5p
L7 net19 net20 12.5p
L9 net018 qa 3.0p
Lbias bias net020 13.0p
Xb0 net020 net019 rsj jjmod=Hyp5a ic=0.118 icrn=0.375 rsh=3.177966
```

```
                            lprsh = 1.588983p
Xb1 net018 net063 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                            lprsh = 3.177966p
Lp1 net20 net018 616f
Lp2 net19 net051 76f
Lp0 net19 net020 400f
Lg0 net019 0 275f
Lg1 net063 0 96f
Lp5 a net053 450f
Lp6 b net034 450f
Lp3 net20 net58 503f
k35 L3 L5 0.78
k46 L4 L6 0.78
.ends a_and011

.subckt a_jtl a bias q
Lg0 net013 0 230f
Lg1 net010 0 230f
Xb1 net022 net010 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net021 net013 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                            lprsh = 2.659575p
L3 a net021 3.0p
L4 net021 net019 3.0p
Lbias bias net019 13.4p
L5 net019 net022 2.1p
L6 net022 q 2.1p
.ends a_jtl

.subckt a_c2_b3 a a_ b b_ bias1a bias2a bias2i bias3 bias7 c00 c01 c10
c11 dc0 dc1 g1 g1a g2 g2a g2i g3 g7 g15 g16 gl gm p1 pl pm1 pm2
L0 c00 c10 1p
XI2 net222 g2i net253 a_jtl_chop
XI40 bias2a net168 net163 net165 net170 n_bias_ihm
XI41 bias1a c01 net168 net170 dc1 n_bias_ihm
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI42 net234 net198 net133 net135 net0149 n_bias
XI43 bias7 net133 c11 dc0 net135 n_bias
XI36 net231 net183 net198 net0149 net185 n_bias
XI30 net249 net193 net183 net185 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI39 bias2i net163 net158 net160 net165 n_bias
XI38 bias3 net158 net173 net175 net160 n_bias
XI35 net181 net173 net178 net180 net175 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
```

```
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
Xand2 pm2 pl g2a net227 a_and011
XI4 net244 net231 p1 a_jtl_e
XI8 net250 net234 g2 a_jtl_e
XI7 net250 g7 g1 a_jtl_e
XI3 net227 g3 net244 a_jtl
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 g15 b a_jtl
XI16 net262 g16 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b3

.subckt a_jtl_chop_e a bias q
Lg0 net010 0 230f
Lg1 net015 0 158f
L6 net022 q 1f
Xb1 net022 net015 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net013 net010 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                             lprsh = 2.659575p
L3 a net013 3.0p
L4 net013 net019 3.3p
Lbias bias net019 11.2p
L5 net019 net022 1.8p
.ends a_jtl_chop_e

.subckt m_outjtl282_400 a bias q
Lg0 net010 0 200f
Lg1 net04 0 200f
Xb1 net023 net04 rsj jjmod=Hyp5a ic=0.400 icrn=0.75 rsh=1.875 lprsh = 937.5f
Xb0 net013 net010 rsj jjmod=Hyp5a ic=0.282 icrn=0.75 rsh=2.659574
                             lprsh = 1.329787p
L3 a net013 1.5p
L4 net013 net027 1.5p
Lbias bias net027 6.2p
L5 net023 net027 1.05p
L6 net023 q 1.05p
.ends m_outjtl282_400

.subckt a_andor011 a b bias qa qo
```

```
LPqq net060 net022 1f
LPq net62 net031 1f
L5 net58 net031 20p
L6 net58 net022 20p
L3 net060 net035 20.5p
L4 net62 net035 20.5p
L8 net055 qo 3.0p
L7 net19 net20 20.0p
L9 net066 qa 3.0p
Lbias bias net51 11.8p
Xb0 net055 net29 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                        lprsh = 3.177966p
Xb1 net066 net41 rsj jjmod=Hyp5a ic=0.118 icrn=0.75 rsh=6.355932
                        lprsh = 3.177966p
Lp1 net20 net066 760f
Lp2 net19 net035 500f
Lp0 net19 net51 470f
Lg0 net29 0 140f
Lg1 net41 0 90f
Lp5 a net060 620f
Lp4 net055 net51 1f
Lp6 b net62 620f
Lp3 net20 net58 550f
k35 L3 L5 0.78
k46 L4 L6 0.78
.ends a_andor011

.subckt a_c5_b5 a b c00 c01 c10 c11 dc0 dc1 dc2 dc3 q
XI31 b net072 net0110 a_jtl_chop_e
XI32 a net067 net0107 a_jtl_chop_e
XI51 net067 net068 net063 net065 net070 n_bias
XI50 net072 c00 net068 net070 net0123 n_bias
XI22 net059 net051 q m_outjtl282_400
XI47 net056 net078 net053 net081 net096 n_bias
XI48 net051 c11 net078 dc0 net081 n_bias
XI21 net062 net056 net059 a_jtle_
XI11 net49 net36 net062 a_jtl_e
XI38 net0134 net0145 net053 net0133 net0153 n_bias
XI26 net54 c01 net092 net0123 dc1 n_bias
XI40 net33 net092 net0145 net0153 net065 n_bias_ihm
Xandor net0110 net0107 net33 net31 net087 a_andor011
XI44 net36 c10 net063 net096 net0133 n_bias
Xanotb net22 net55 dc2 dc3 net49 a_anotb
XI1 net087 net0134 net22 a_jtl
XI2 net31 net54 net55 a_jtl
```

```
.ends a_c5_b5

.subckt a_jtle_e a bias q
Lg0 net09 0 250f
Lg1 net011 0 250f
Xb1 net014 net011 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net013 net09 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
                        lprsh = 2.659575p
L3 a net013 3.0p
Lbias bias net022 9.9p
L4 net013 net022 3.0p
L5 net022 net014 2.1p
L6 net014 q 2.1p
.ends a_jtle_e

.subckt a_c5_b0 b c00 c01 c10 c11 dc0 dc1 dc2 dc3 q
L1 dc2 dc3 1p
L0 net042 c11 1p
XI31 b net038 net033 a_jtl_e
XI50 net038 c00 net035 net037 dc1 n_bias
XI21 net060 net051 net041 a_jtle_
XI48 net046 net042 net047 dc0 net050 n_bias
XI47 net051 net047 net0150 net050 net069 n_bias
XI22 net041 net046 q m_outjtl282_400
XI1 net033 net0134 net49 a_jtle_e
XI11 net49 net36 net060 a_jtle_e
XI38 net0134 c01 net0150 net0123 net037 n_bias
XI44 net36 c10 net035 net069 net0123 n_bias
.ends a_c5_b0

.subckt a_c4_b5 _gl6 _gl7 a a_ c00 c01 c10 c11 dc0 dc1 g gl gl6_ gl7_ gm pm
XI5 net281 net220 net280 a_jtl_chop
XI23 net190 net186 net181 net189 net184 n_bias
XI24 net185 net181 net236 net184 net239 n_bias
XI28 net180 net176 c01 net179 dc1 n_bias
XI43 net195 net217 c10 dc0 net219 n_bias_ihm
XI40 net283 net227 net197 net199 net229 n_bias_ihm
XI22 net205 c00 net202 net204 net189 n_bias
XI55 net210 net222 net207 net209 net224 n_bias
XI26 net220 net207 net217 net219 net209 n_bias
XI38 net225 net197 net222 net224 net199 n_bias
XI16 net293 net202 net227 net229 net244 n_bias
XI18 net245 c11 net246 net244 net249 n_bias
XI25 net276 net236 net176 net239 net179 n_bias
XI19 net273 net246 net186 net249 net204 n_bias
```

```
XI13 gl net245 net253 a_jtl_e
XI10 a_ net185 net259 a_jtle_e
XI12 pm net180 net256 a_jtle_e
XI11 gm net190 net262 a_jtle_e
XI1 net259 net210 a a_jtle_
XI14 gl6_ net273 _gl6 a_jtl
XI15 gl7_ net276 _gl7 a_jtl
Xand1 net291 net294 net283 net281 a_and
Xor net297 net280 net195 g a_or
XI4 net256 net205 net291 a_jtle_chop
XI3 net253 net293 net294 a_jtle_chop
XI2 net262 net225 net297 a_jtle_chop
.ends a_c4_b5

.subckt a_c4_b6 _gl7 a a_ c00 c01 c10 c11 dc0 dc1 g gl gl7_ gm pm
XI5 net281 net220 net280 a_jtl_chop
XI23 net190 net186 net181 net189 net184 n_bias
XI24 net185 net181 net236 net184 net239 n_bias
XI28 net180 net236 c01 net239 dc1 n_bias
XI43 net195 net217 c10 dc0 net219 n_bias_ihm
XI40 net283 net227 net197 net199 net229 n_bias_ihm
XI22 net205 c00 net202 net204 net189 n_bias
XI55 net210 net222 net207 net209 net224 n_bias
XI26 net220 net207 net217 net219 net209 n_bias
XI38 net225 net197 net222 net224 net199 n_bias
XI16 net293 net202 net227 net229 net244 n_bias
XI18 net245 c11 net246 net244 net249 n_bias
XI19 net273 net246 net186 net249 net204 n_bias
XI13 gl net245 net253 a_jtl_e
XI10 a_ net185 net259 a_jtle_e
XI12 pm net180 net256 a_jtle_e
XI11 gm net190 net262 a_jtle_e
XI1 net259 net210 a a_jtle_
XI14 gl7_ net273 _gl7 a_jtl
Xand1 net291 net294 net283 net281 a_and
Xor net297 net280 net195 g a_or
XI4 net256 net205 net291 a_jtle_chop
XI3 net253 net293 net294 a_jtle_chop
XI2 net262 net225 net297 a_jtle_chop
.ends a_c4_b6

.subckt a_c4_b7 a a_ c00 c01 c10 c11 dc0 dc1 g gl gm pm
XI5 net281 net220 net280 a_jtl_chop
XI23 net190 net186 net181 net189 net184 n_bias
XI24 net185 net181 net236 net184 net239 n_bias
```

```
XI28 net180 net236 c01 net239 dc1 n_bias
XI43 net195 net217 c10 dc0 net219 n_bias_ihm
XI40 net283 net227 net197 net199 net229 n_bias_ihm
XI22 net205 c00 net202 net249 net189 n_bias
XI55 net210 net222 net207 net209 net224 n_bias
XI26 net220 net207 net217 net219 net209 n_bias
XI38 net225 net197 net222 net224 net199 n_bias
XI16 net293 net202 net227 net229 net244 n_bias
XI18 net245 c11 net186 net244 net249 n_bias
XI13 gl net245 net253 a_jtl_e
XI10 a_ net185 net259 a_jtle_e
XI12 pm net180 net256 a_jtle_e
XI11 gm net190 net262 a_jtle_e
XI1 net259 net210 a a_jtle_
Xand1 net291 net294 net283 net281 a_and
Xor net297 net280 net195 g a_or
XI4 net256 net205 net291 a_jtle_chop
XI3 net253 net293 net294 a_jtle_chop
XI2 net262 net225 net297 a_jtle_chop
.ends a_c4_b7

.subckt a_c4_b4 _gl5 _gl6 _gl7 a a_ bias1 bias10 c00 c01 c10 c11 dc0 dc1 g g_ gl5_
L0 net241 c11 1p
XI31 bias1 c00 net066 net068 net0108 n_bias
XI23 net190 net186 net181 net204 net184 n_bias
XI24 net185 net181 net236 net184 net239 n_bias
XI28 net180 net176 c01 net179 dc1 n_bias
XI55 net210 net222 c10 dc0 net224 n_bias
XI38 net225 net066 net222 net224 net068 n_bias
XI25 net276 net236 net176 net239 net179 n_bias
XI19 net273 net0100 net186 net249 net204 n_bias
XI30 bias10 net241 net0100 net0108 net249 n_bias
XI10 a_ net185 net259 a_jtle_e
XI11 g_ net190 net262 a_jtle_e
XI2 net262 net225 g a_jtle_
XI16 gl7_ net180 _gl7 a_jtle_
XI1 net259 net210 a a_jtle_
XI14 gl5_ net273 _gl5 a_jtl
XI15 gl6_ net276 _gl6 a_jtl
.ends a_c4_b4

.subckt a_c4_b3 _gl5 _gl6 a a_ bias1 bias10 c00 c01 c10 c11 dc0 dc1 g
                             g1 g10 g_ gl5_ gl6_
L0 net186 c11 1p
XI31 bias1 c00 net052 net054 net077 n_bias
```

```
XI15 gl5_ net276 _gl5 a_jtl
XI23 net190 net059 net236 net249 net239 n_bias
XI28 net180 net176 c01 net179 dc1 n_bias
XI38 net225 net052 c10 dc0 net054 n_bias
XI30 bias10 net186 net059 net077 net249 n_bias
XI25 net276 net236 net176 net239 net179 n_bias
XI10 g_ g10 net259 a_jtle_e
XI11 a_ net190 net262 a_jtle_e
XI2 net262 net225 g a_jtle_
XI16 gl6_ net180 _gl6 a_jtle_
XI1 net259 g1 a a_jtle_
.ends a_c4_b3

.subckt a_c4_b2 _gl5 a a_ bias1 bias10 c00 c01 c10 c11 dc0 dc1 g g1
                          g10 g_ gl5_ L0 net186 c11 1p
XI30 bias10 net186 net059 net052 net249 n_bias
XI31 bias1 c00 net055 net057 net052 n_bias
XI23 net190 net059 net236 net249 net239 n_bias
XI28 net180 net236 c01 net239 dc1 n_bias
XI38 net225 net055 c10 dc0 net057 n_bias
XI10 g_ g10 net259 a_jtle_e
XI11 a_ net190 net262 a_jtle_e
XI2 net262 net225 g a_jtle_
XI1 net259 g1 a a_jtle_
XI16 gl5_ net180 _gl5 a_jtl
.ends a_c4_b2

.subckt a_c4_b1 a a_ c00 c01 c10 c11 dc0 dc1 g g1 g10 g_
L0 net186 c11 1p
XI23 net190 net186 c01 net249 dc1 n_bias
XI38 net225 c00 c10 dc0 net249 n_bias
XI10 g_ g10 net259 a_jtle_e
XI11 a_ net190 net262 a_jtle_e
XI2 net262 net225 g a_jtle_
XI1 net259 g1 a a_jtle_
.ends a_c4_b1

.subckt a_c4_b0 a a_ c00 c01 c10 c11 dc0 dc1
XI24 net185 c11 c01 net224 dc1 n_bias
XI55 net210 c00 c10 dc0 net224 n_bias
XI10 a_ net185 net259 a_jtle_e
XI1 net259 net210 a a_jtle_
.ends a_c4_b0

.subckt a_c3_b0 a b c00 c01 c10 c11 dc0 dc1 dc2 dc3 q
```

```
L0 c01 c11 1p
XI27 net047 c10 c00 dc0 dc1 n_bias
Xanotb a b dc2 dc3 net013 a_anotb
XI1 net013 net047 q a_jtl_e
.ends a_c3_b0

.subckt a_c3_b2 a b bias16 c00 c01 c10 c11 dc0 dc1 dc2 dc3 g g_ q
L0 net045 c11 1p
XI46 bias16 c01 net045 net034 dc1 n_bias
XI2 g_ net0134 net49 a_jtle_
XI44 net36 c10 net0115 dc0 net0118 n_bias
XI38 net0134 net0100 c00 net0123 net034 n_bias
XI27 net047 net0115 net0100 net0118 net0123 n_bias
Xanotb a b dc2 dc3 net013 a_anotb
XI1 net013 net047 q a_jtl_e
XI6 net49 net36 g a_jtl_e
.ends a_c3_b2

.subckt a_c3_b4 a b bias8 bias15 bias16 c00 c01 c10 c11 dc0 dc1 dc2
                   dc3 g g4 gl gm pm1 q
L0 net080 c11 1p
XI47 bias16 net075 net080 net0148 net059 n_bias
XI48 bias15 c01 net075 net068 net062 n_bias
XI5 net31 net54 net55 a_jtl_chop
XI40 net33 net0150 net0145 net0153 net0148 n_bias_ihm
XI43 net25 net0100 net0120 net0103 net0123 n_bias_ihm
XI45 bias8 net072 c00 net062 dc1 n_bias
XI44 net36 c10 net0115 dc0 net0118 n_bias
XI22 net0149 net0145 net072 net059 net068 n_bias
XI38 net0134 net090 net0150 net0133 net0153 n_bias
XI26 net54 net0120 net090 net0123 net0133 n_bias
XI27 net047 net0115 net0100 net0118 net0103 n_bias
Xanotb a b dc2 dc3 net013 a_anotb
XI2 gm net0134 net22 a_jtle_chop
XI3 gl net0149 net34 a_jtle_chop
XI4 pm1 g4 net32 a_jtle_chop
Xor net22 net55 net25 net49 a_or
Xand1 net32 net34 net33 net31 a_and
XI1 net013 net047 q a_jtl_e
XI6 net49 net36 g a_jtl_e
.ends a_c3_b4

.subckt a_c3_b7 a b c00 c01 c10 c11 dc0 dc1 dc2 dc3 g g4 g8 g9 gl gm p
                    pl pm1 pm2 q L0 c01 c11 1p
XI40 net33 net0150 net0145 net0153 net0148 n_bias_ihm
```

```
XI43 net25 net0100 net0120 net0103 net0123 n_bias_ihm
XI39 net27 net0125 net0110 net0138 net0133 n_bias_ihm
XI42 net016 net072 c00 net068 dc1 n_bias
XI33 net094 net090 net0125 net093 net0138 n_bias
XI44 net36 c10 net0115 dc0 net0118 n_bias
XI22 net0149 net0145 net072 net0148 net068 n_bias
XI38 net0134 net0110 net0150 net0133 net0153 n_bias
XI26 net54 net0120 net090 net0123 net093 n_bias
XI27 net047 net0115 net0100 net0118 net0103 n_bias
Xanotb a b dc2 dc3 net013 a_anotb
XI8 pl g8 net018 a_jtle_
XI9 net018 g9 net30 a_jtl_chop
XI5 net31 net54 net55 a_jtl_chop
XI2 gm net0134 net22 a_jtle_chop
XI3 gl net0149 net34 a_jtle_chop
XI4 pm1 g4 net32 a_jtle_chop
XI7 pm2 net016 net29 a_jtle_chop
Xor net22 net55 net25 net49 a_or
Xand1 net32 net34 net33 net31 a_and
Xand2 net29 net30 net27 net28 a_and
XI1 net013 net047 q a_jtl_e
XI10 net28 net094 p a_jtl_e
XI6 net49 net36 g a_jtl_e
.ends a_c3_b7

.subckt a_c3_b5 a b bias8 c00 c01 c10 c11 dc0 dc1 dc2 dc3 g g4 g8 g9
                                gl gm p pl pm1 pm2 q
XI40 net33 net0150 net0145 net0153 net0148 n_bias_ihm
XI43 net25 net0100 net0120 net0103 net0123 n_bias_ihm
XI39 net27 net0125 net0110 net0138 net0133 n_bias_ihm
XI42 net016 net072 net073 net068 net074 n_bias
XI33 net094 net090 net0125 net093 net0138 n_bias
XI45 bias8 net073 c00 net074 dc1 n_bias
XI44 net36 c10 net0115 dc0 net0118 n_bias
XI22 net0149 net0145 net072 net0148 net068 n_bias
XI38 net0134 net0110 net0150 net0133 net0153 n_bias
XI26 net54 net0120 net090 net0123 net093 n_bias
XI27 net047 net0115 net0100 net0118 net0103 n_bias
L0 c01 c11 1p
Xanotb a b dc2 dc3 net013 a_anotb
XI8 pl g8 net018 a_jtle_
XI9 net018 g9 net30 a_jtl_chop
XI5 net31 net54 net55 a_jtl_chop
XI2 gm net0134 net22 a_jtle_chop
XI3 gl net0149 net34 a_jtle_chop
```

```
XI4 pm1 g4 net32 a_jtle_chop
XI7 pm2 net016 net29 a_jtle_chop
Xor net22 net55 net25 net49 a_or
Xand1 net32 net34 net33 net31 a_and
Xand2 net29 net30 net27 net28 a_and
XI1 net013 net047 q a_jtl_e
XI10 net28 net094 p a_jtl_e
XI6 net49 net36 g a_jtl_e
.ends a_c3_b5

.subckt a_jtle_chop_e a bias q
Xb1 net014 net09 rsj jjmod=Hyp5a ic=0.200 icrn=0.75 rsh=3.75 lprsh = 1.875p
Xb0 net021 net013 rsj jjmod=Hyp5a ic=0.141 icrn=0.75 rsh=5.319149
lprsh = 2.659575p
Lg0 net013 0 250f
Lg1 net09 0 250f
L6 net014 q 1f
L3 a net021 3.0p
Lbias bias net023 9.9p
L4 net021 net023 3.0p
L5 net023 net014 2.1p
.ends a_jtle_chop_e

.subckt q_c0_b7 _b a a_ b c00 c01 c10 c11 dc0 dc1
XI13 net89 net0114 a a_jtle_chop_e
XI12 net73 net0119 b a_jtle_chop_e
XI7 net70 net0104 _b a_jtle_e
XI6 net73 net0109 net70 a_jtle_e
XI5 net76 net0129 net73 a_jtle_e
XI4 net078 net0139 net76 a_jtle_e
XI3 net86 net099 net078 a_jtle_e
XI2 net89 net0124 net86 a_jtle_e
XI1 net92 net088 net89 a_jtle_e
XI0 a_ net091 net92 a_jtle_e
XI8 net091 net097 net34 net079 net0100 n_bias
XI10 net099 net60 net65 net67 net62 n_bias
XI17 net0104 net39 net60 net62 net22 n_bias
XI18 net0109 net19 net50 net22 net42 n_bias
XI15 net0114 net097 net096 net0100 net37 n_bias
XI14 net0119 c11 net34 dc0 net47 n_bias
XI11 net0124 net096 net19 net37 net67 n_bias
XI19 net0129 net39 c00 net42 net52 n_bias
XI9 net088 c10 net65 net47 net079 n_bias
XI16 net0139 c01 net50 net52 dc1 n_bias
.ends q_c0_b7
```

```
.subckt q_c0_b6 _a _b a a_ b b_ c00 c01 c10 c11 dc0 dc1
XI13 net89 net0114 a a_jtle_chop_e
XI12 net73 net0119 b a_jtle_chop_e
XI7 net70 net0104 _b a_jtle_e
XI6 net73 net0109 net70 a_jtle_e
XI5 net76 net0129 net73 a_jtle_e
XI4 b_ net0139 net76 a_jtle_e
XI3 net86 net099 _a a_jtle_e
XI2 net89 net0124 net86 a_jtle_e
XI1 net92 net088 net89 a_jtle_e
XI0 a_ net091 net92 a_jtle_e
XI8 net091 net097 net34 net079 net0100 n_bias
XI10 net099 net60 net65 net67 net62 n_bias
XI17 net0104 net39 net60 net62 net22 n_bias
XI18 net0109 net19 net50 net22 net42 n_bias
XI15 net0114 net097 net096 net0100 net37 n_bias
XI14 net0119 c11 net34 dc0 net47 n_bias
XI11 net0124 net096 net19 net37 net67 n_bias
XI19 net0129 net39 c00 net42 net52 n_bias
XI9 net088 c10 net65 net47 net079 n_bias
XI16 net0139 c01 net50 net52 dc1 n_bias
.ends q_c0_b6

.subckt a_c2_b5 a a_ b b_ bias1a bias2a bias2i bias3 bias4 bias7 bias9
             c00 c01 c10 c11 dc0 dc1 g1 g1a
g2 g2a g2i g3 g7 gl gm p1 p3 pl pm1 pm2
XI2 net222 g2i net253 a_jtl_chop
XI44 bias4 c10 net0129 net0123 net140 n_bias
XI47 bias9 net0129 c00 net145 net200 n_bias
XI40 bias2a net168 net163 net165 net170 n_bias_ihm
XI41 bias1a c01 net168 net170 dc1 n_bias_ihm
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI37 net240 net138 c11 dc0 net0123 n_bias
XI42 net234 net198 net133 net135 net0186 n_bias
XI43 bias7 net133 net138 net140 net135 n_bias
XI36 net231 net183 net143 net200 net185 n_bias
XI30 net249 net193 net148 net150 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI39 bias2i net163 net158 net160 net165 n_bias
XI38 bias3 net158 net173 net175 net160 n_bias
XI35 net181 net173 net178 net180 net175 n_bias
XI23 net186 net148 net183 net185 net150 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
```

```
XI24 net201 net143 net198 net0186 net145 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
Xand2 pm2 pl g2a net227 a_and011
XI4 net244 net231 p1 a_jtl_e
XI8 net250 net234 g2 a_jtl_e
XI7 net250 g7 g1 a_jtl_e
XI6 net244 net240 p3 a_jtl_e
XI3 net227 g3 net244 a_jtl
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 net186 b a_jtl
XI16 net262 net201 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b5

.subckt a_c2_b4 a a_ b b_ bias1a bias2a bias2i bias3 bias4 bias7 bias9
            c00 c01 c10 c11 dc0 dc1 g1 g1a g2 g2a
 g2i g3 g7 gl gm p1 pl pm1 pm2
XI44 bias4 c10 net0125 dc0 net0156 n_bias
XI47 bias9 net0125 c00 net145 net0149 n_bias
XI2 net222 g2i net253 a_jtl_chop
XI40 bias2a net168 net163 net165 net170 n_bias_ihm
XI41 bias1a c01 net168 net170 dc1 n_bias_ihm
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI42 net234 net198 net133 net135 net200 n_bias
XI43 bias7 net133 c11 net0156 net135 n_bias
XI36 net231 net183 net143 net0149 net185 n_bias
XI30 net249 net193 net148 net150 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI39 bias2i net163 net158 net160 net165 n_bias
XI38 bias3 net158 net173 net175 net160 n_bias
XI35 net181 net173 net178 net180 net175 n_bias
XI23 net186 net148 net183 net185 net150 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
XI24 net201 net143 net198 net200 net145 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
```

```
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
Xand2 pm2 pl g2a net227 a_and011
XI4 net244 net231 p1 a_jtl_e
XI8 net250 net234 g2 a_jtl_e
XI7 net250 g7 g1 a_jtl_e
XI3 net227 g3 net244 a_jtl
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 net186 b a_jtl
XI16 net262 net201 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b4

.subckt a_c2_b6 a a_ b b_ bias1a bias2a bias2i bias3 bias4 bias7 bias9
             c00 c01 c10 c11 dc0 dc1 g1a
g2 g2a g2i g3 gl gm p1 p3 pl pm1 pm2
XI47 bias9 net0114 c00 net145 net0160 n_bias
XI44 bias4 c10 net0114 net0122 net140 n_bias
XI2 net222 g2i net253 a_jtl_chop
XI40 bias2a net168 net163 net165 net170 n_bias_ihm
XI41 bias1a c01 net168 net170 dc1 n_bias_ihm
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI37 net240 net138 c11 dc0 net0122 n_bias
XI42 net234 net198 net133 net135 net200 n_bias
XI43 bias7 net133 net138 net140 net135 n_bias
XI36 net231 net183 net143 net0160 net185 n_bias
XI30 net249 net193 net148 net150 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI39 bias2i net163 net158 net160 net165 n_bias
XI38 bias3 net158 net173 net175 net160 n_bias
XI35 net181 net173 net178 net180 net175 n_bias
XI23 net186 net148 net183 net185 net150 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
XI24 net201 net143 net198 net200 net145 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
Xand2 pm2 pl g2a net227 a_and011
XI4 net244 net231 p1 a_jtl_e
XI8 net250 net234 g2 a_jtl_e
```

```
XI6 net244 net240 p3 a_jtl_e
XI3 net227 g3 net244 a_jtl
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 net186 b a_jtl
XI16 net262 net201 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b6

.subckt a_c2_b7 a a_ b b_ bias4 c00 c01 c10 c11 dc0 dc1 g1a g2 g2a g2i
                         g3 gl gm p1 p3 pl pm1 pm2
XI44 bias4 c10 c00 net135 net0114 n_bias
XI2 net222 g2i net253 a_jtl_chop
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI37 net240 net138 c11 dc0 net135 n_bias
XI42 net234 net198 net138 net0114 net200 n_bias
XI36 net231 net183 net143 net145 net185 n_bias
XI30 net249 net193 net148 net150 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI35 net181 c01 net178 net180 dc1 n_bias
XI23 net186 net148 net183 net185 net150 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
XI24 net201 net143 net198 net200 net145 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
Xand2 pm2 pl g2a net227 a_and011
XI4 net244 net231 p1 a_jtl_e
XI8 net250 net234 g2 a_jtl_e
XI6 net244 net240 p3 a_jtl_e
XI3 net227 g3 net244 a_jtl
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 net186 b a_jtl
XI16 net262 net201 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b7

.subckt a_c2_b2 a a_ b b_ bias1a bias2a bias2i bias3 bias7 bias15 c00
            c01 c10 c11 dc0 dc1 g1 g1a g2 g2i g7
 g15 g16 gl gm pm1
```

```
L0 c00 c10 1p
XI2 net222 g2i net253 a_jtl_chop
XI40 bias2a net168 net163 net165 net170 n_bias_ihm
XI41 bias1a c01 net168 net170 dc1 n_bias_ihm
XI33 net126 net153 net123 net125 net155 n_bias_ihm
XI42 net234 net198 net133 net135 net150 n_bias
XI43 bias7 net133 net0125 net0127 net135 n_bias
XI44 bias15 net0125 c11 dc0 net0127 n_bias
XI30 net249 net193 net198 net150 net195 n_bias
XI31 net216 net188 net153 net155 net190 n_bias
XI39 bias2i net163 net158 net160 net165 n_bias
XI38 bias3 net158 net173 net175 net160 n_bias
XI35 net181 net173 net178 net180 net175 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI1 gm net216 net217 a_jtle_chop
Xor net217 net253 net126 net218 a_or
Xand1 pm1 gl g1a net222 a_and011
XI8 net250 net234 g2 a_jtl_e
XI7 net250 g7 g1 a_jtl_e
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 g15 b a_jtl
XI16 net262 g16 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b2

.subckt a_c2_b1 a a_ b b_ bias1a bias2i bias15 c00 c01 c10 c11 dc0 dc1
                            g1 g2 g7 g15 g16 g_
L0 c00 c10 1p
XI41 bias1a c01 net168 net165 dc1 n_bias_ihm
XI42 net0136 net0115 net086 net088 net0117 n_bias
XI44 bias15 net086 c11 dc0 net088 n_bias
XI30 net249 net193 net0115 net0117 net195 n_bias
XI31 net216 net188 net123 net125 net190 n_bias
XI39 bias2i net168 net158 net175 net165 n_bias
XI35 net181 net158 net178 net180 net175 n_bias
XI34 net191 net178 net188 net190 net180 n_bias
XI17 net196 net203 net193 net195 net205 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
```

```
XI1 g_ net216 net218 a_jtle_chop
XI8 net250 net0136 g2 a_jtl_e
XI7 net250 g7 g1 a_jtl_e
XI13 net214 net206 net247 a_jtl
XI5 net218 net249 net250 a_jtl
XI15 net247 g15 b a_jtl
XI16 net262 g16 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b1

.subckt a_c2_b0 a a_ b b_ c00 c01 c10 c11 dc0 dc1 g15 g16
L0 c00 c10 1p
XI35 net181 c01 net178 net180 dc1 n_bias
XI34 net191 net178 net123 net125 net180 n_bias
XI17 net196 net203 c11 dc0 net205 n_bias
XI28 net206 net123 net203 net205 net125 n_bias
XI12 a_ net191 net211 a_jtle_
XI11 b_ net181 net214 a_jtle_
XI13 net214 net206 net247 a_jtl
XI15 net247 g15 b a_jtl
XI16 net262 g16 a a_jtl
XI14 net211 net196 net262 a_jtl
.ends a_c2_b0

.subckt a_c1_b5 a5 b5 bias5 bias7 bias10 bias11 c00 c01 c10 c11 dc0
                dc1 g1_chop g2 g3 g5 g7 g10
                g11 p1_chop p2_chop p3_chop p4
XI29 bias11 net0124 net92 net94 net0126 n_bias
XI30 bias7 net0129 net97 net99 net0131 n_bias
XI31 bias5 net0134 net102 net104 net0136 n_bias
L0 c01 c11 1p
XI14 net85 c00 net82 net84 dc1 n_bias_ihm
XI28 bias10 net0109 net112 net114 net0111 n_bias
XI27 net138 net92 c10 dc0 net94 n_bias
XI26 net164 net97 net0124 net0126 net99 n_bias
XI25 net135 net102 net0129 net0131 net104 n_bias
XI24 net144 net122 net0134 net0136 net124 n_bias
XI17 net110 net112 net107 net109 net114 n_bias
XI16 net156 net127 net0109 net0111 net129 n_bias
XI22 net141 net107 net117 net119 net109 n_bias
XI23 net153 net117 net122 net124 net119 n_bias
XI15 net130 net82 net127 net129 net84 n_bias
XI10 net145 g10 p4 a_jtl_e
XI13 net154 net135 g3 a_jtl_e
XI12 net154 net138 g2 a_jtl_e
```

```
XI3 net157 net141 net142 a_jtl
XI4 net157 net144 net145 a_jtl
XI2 net158 net130 net148 a_jtl
XI5 net148 g5 net151 a_jtl
XI6 net148 net153 net154 a_jtl
XI1 net162 net156 net157 a_jtl
Xandor a5 b5 net85 net158 net162 a_andor011
XI9 net145 net164 p3_chop a_jtl_chop_e
XI8 net142 net110 p2_chop a_jtl_chop_e
XI11 net151 g11 g1_chop a_jtl_chop_e
XI7 net142 g7 p1_chop a_jtl_chop_e
.ends a_c1_b5

.subckt a_c1_b6 a5 b5 bias5 bias7 bias10 bias11 c00 c01 c10 c11 dc0
                 dc1 g2 g3 g10 p2_chop p3_chop p4
L0 c01 c11 1p
XI29 bias11 net0124 net92 net94 net0126 n_bias
XI30 bias7 net0129 net97 net99 net0131 n_bias
XI31 bias5 net0134 net102 net104 net0136 n_bias
XI14 net85 c00 net82 net84 dc1 n_bias_ihm
XI28 bias10 net059 net112 net0111 net0133 n_bias
XI27 net138 net92 c10 dc0 net94 n_bias
XI26 net164 net97 net0124 net0126 net99 n_bias
XI25 net135 net102 net0129 net0131 net104 n_bias
XI24 net144 net122 net0134 net0136 net124 n_bias
XI17 net110 net112 net107 net109 net0111 n_bias
XI16 net156 net127 net059 net0133 net129 n_bias
XI22 net141 net107 net117 net119 net109 n_bias
XI23 net153 net117 net122 net124 net119 n_bias
XI15 net130 net82 net127 net129 net84 n_bias
XI10 net145 g10 p4 a_jtl_e
XI13 net154 net135 g3 a_jtl_e
XI12 net154 net138 g2 a_jtl_e
XI3 net157 net141 net142 a_jtl
XI4 net157 net144 net145 a_jtl
XI2 net158 net130 net148 a_jtl
XI6 net148 net153 net154 a_jtl
XI1 net162 net156 net157 a_jtl
Xandor a5 b5 net85 net158 net162 a_andor011
XI9 net145 net164 p3_chop a_jtl_chop_e
XI8 net142 net110 p2_chop a_jtl_chop_e
.ends a_c1_b6

.subckt a_c1_b7 a5 b5 c00 c01 c10 c11 dc0 dc1 g3 g10 p4
L0 c01 c11 1p
```

```
XI14 net85 c00 net82 net84 dc1 n_bias_ihm
XI25 net135 net0134 c10 dc0 net104 n_bias
XI24 net144 net122 net0134 net104 net124 n_bias
XI16 net156 net127 net117 net119 net129 n_bias
XI23 net153 net117 net122 net124 net119 n_bias
XI15 net130 net82 net127 net129 net84 n_bias
XI10 net145 g10 p4 a_jtl_e
XI13 net154 net135 g3 a_jtl_e
XI4 net157 net144 net145 a_jtl
XI2 net158 net130 net148 a_jtl
XI6 net148 net153 net154 a_jtl
XI1 net162 net156 net157 a_jtl
Xandor a5 b5 net85 net158 net162 a_andor011
.ends a_c1_b7

.subckt a_c1_b1 a5 b5 bias5 bias10 bias11 c00 c01 c10 c11 dc0 dc1
            g1_chop g2 g3 g5 g7 g10 g11 p1_chop p2_chop p4
L0 c01 c11 1p
XI29 bias11 net97 net92 net94 net0126 n_bias
XI31 bias5 net0134 net102 net104 net0136 n_bias
XI14 net85 c00 net82 net84 dc1 n_bias_ihm
XI28 bias10 net0109 net112 net114 net0111 n_bias
XI27 net138 net92 c10 dc0 net94 n_bias
XI25 net135 net102 net97 net0126 net104 n_bias
XI24 net144 net122 net0134 net0136 net124 n_bias
XI17 net110 net112 net107 net109 net114 n_bias
XI16 net156 net127 net0109 net0111 net129 n_bias
XI22 net141 net107 net117 net119 net109 n_bias
XI23 net153 net117 net122 net124 net119 n_bias
XI15 net130 net82 net127 net129 net84 n_bias
XI10 net145 g10 p4 a_jtl_e
XI13 net154 net135 g3 a_jtl_e
XI12 net154 net138 g2 a_jtl_e
XI3 net157 net141 net142 a_jtl
XI4 net157 net144 net145 a_jtl
XI2 net158 net130 net148 a_jtl
XI5 net148 g5 net151 a_jtl
XI6 net148 net153 net154 a_jtl
XI1 net162 net156 net157 a_jtl
Xandor a5 b5 net85 net158 net162 a_andor011
XI8 net142 net110 p2_chop a_jtl_chop_e
XI11 net151 g11 g1_chop a_jtl_chop_e
XI7 net142 g7 p1_chop a_jtl_chop_e
.ends a_c1_b1
```

```
.subckt a_c1_b0 a5 b5 bias10 c00 c01 c10 c11 dc0 dc1 g1_chop g2 g3 g5 g11 p4
L0 c01 c11 1p
XI14 net85 c00 net82 net84 dc1 n_bias_ihm
XI28 bias10 net0109 net089 net091 net0111 n_bias
XI27 net138 net97 c10 dc0 net0126 n_bias
XI25 net135 net096 net0134 net104 net098 n_bias
XI24 net144 net089 net112 net124 net091 n_bias
XI17 net110 net112 net096 net098 net124 n_bias
XI16 net156 net127 net0109 net0111 net129 n_bias
XI23 net153 net0134 net97 net0126 net104 n_bias
XI15 net130 net82 net127 net129 net84 n_bias
XI10 net145 net110 p4 a_jtl_e
XI13 net154 net135 g3 a_jtl_e
XI12 net154 net138 g2 a_jtl_e
XI4 net157 net144 net145 a_jtl
XI2 net158 net130 net148 a_jtl
XI5 net148 g5 net151 a_jtl
XI6 net148 net153 net154 a_jtl
XI1 net162 net156 net157 a_jtl
Xandor a5 b5 net85 net158 net162 a_andor011
XI11 net151 g11 g1_chop a_jtl_chop_e
.ends a_c1_b0

.subckt a_add a0 a1 c00_0 c00_1 c00_2 c00_3 c00_4 c00_5 c00_6 c00_7
           c01_0 c01_1 c01_2 c01_3 c01_4 c01_5 c01_6 c01_7
           c10_0 c10_1 c10_2 c10_3 c10_4 c10_5 c10_6 c10_7
    c11_0 c11_1 c11_2 c11_3 c11_4 c11_5 c11_6 c11_7
    dc0_0 dc0_1 dc0_2 dc0_3 dc0_4 dc0_5 dc0_6
    + dc0_7 dc1_0 dc1_1 dc1_2 dc1_3 dc1_4 dc1_5 dc1_6
           dc1_7 dc2 dc3 q0 q1 q2 q3 q4 q5 q6 q7 qv
XI49 net0408 net0412 net0409 c10_0 c11_0 dc0_0 net0411 q0 m_out12
XI50 net0416 net0420 net0417 c10_1 c11_1 dc0_1 net0419 q1 m_out12
XI51 net0424 net0428 net0425 c10_2 c11_2 dc0_2 net0427 q2 m_out12
XI52 net0432 net0436 net0433 c10_3 c11_3 dc0_3 net0435 q3 m_out12
XI53 net0440 net0444 net0441 c10_4 c11_4 dc0_4 net0443 q4 m_out12
XI54 net0565 net0564 net0563 c10_5 c11_5 dc0_5 net0559 q5 m_out12
XI55 net0456 net0460 net0457 c10_6 c11_6 dc0_6 net0459 q6 m_out12
XI56 net0464 net0468 net0465 c10_7 c11_7 dc0_7 net0467 q7 m_out12
XI48 net90 a0 a1 c01_0 net86 net94 dc1_0 net83 qv q_c0_io
XI28 net141 net140 net0265 net0850 net134 net030 net035 net034 net029
           net124 net053 net025 net039 net041 net032
    net040 net038 a_c3_b3
XI14 net141 net127 net140 net311 net165 net177 net178 net179 net240
           net316 net315 net134 net030 net124 net132 net128
           net239 net032 net248 net249 net250 net166 net0295
    net0294 net410 net129 net040 net145 net286 net285 a_c2_b3
XI42 net0617 net0615 net0612 net0611 net0460 net0457 net0459 net0605
           net0248 net0292 net0456 a_c5_b5
XI45 net0653 net0658 net0656 net0655 net0436 net0433 net0435 net0647
           net0259 net0303 net0432 a_c5_b5
XI47 net0677 net0681 net0680 net0679 net0420 net0417 net0419 net0673
           net0319 net0314 net0416 a_c5_b5
XI43 net0603 net0600 net0596 net0595 net0564 net0563 net0559 net0589
                 net0304 net0248 net0565 a_c5_b5
XI41 net0629 net0627 net0626 net0625 net0468 net0465 net0467 net0619
           net0292 dc3 net0464 a_c5_b5
XI44 net0637 net0644 net0640 net0639 net0444 net0441 net0443 net0631
           net0303 net0304 net0440 a_c5_b5
XI46 net0667 net0671 net0670 net0669 net0428 net0425 net0427 net0661
           net0314 net0259 net0424 a_c5_b5
XI40 net0690 net0689 net0688 net0412 net0409 net0411 net0683 net091
           net0319 net0408 a_c5_b0
XI39 net0616 net0613 net0600 net0114 net0111 net0110 net0596 net0595
           net0589 net0105 net0603 net0635 net0645 net0642
           net0115 net0121 a_c4_b5
XI38 net0628 net0615 net0135 net0132 net0131 net0612 net0611 net0605
           net0126 net0617 net0616 net0613 net0136 net0142 a_c4_b6
XI37 net0627 net072 net069 net068 net0626 net0625 net0619 net063
           net0629 net0628 net073 net078 a_c4_b7
XI36 net0635 net0645 net0642 net0644 net055 net0477 net0478 net052
           net051 net0640 net0639 net0631 net046 net0637
           net056 net0651 net0659 net039 a_c4_b4
XI35 net0651 net0659 net0658 net038 net0506 net0507 net035 net034
           net0656 net0655 net0647 net029 net0653 net0477
           net0478 net039 net0665 net0702 a_c4_b3
XI34 net0665 net0671 net026 net0520 net0521 net023 net022 net0670
           net0669 net0661 net018 net0667 net0506 net0507
    net0702 net0715 a_c4_b2
XI33 net0681 net0715 net010 net09 net0680 net0679 net0673 net05
           net0677 net0520 net0521 net013 a_c4_b1
XI32 net0690 net093 net090 net089 net0689 net0688 net0683 net084 a_c4_b0
XI31 net278 net082 net086 net273 net090 net089 net084 net267 net092
           net091 net093 a_c3_b0
XI29 net245 net244 net0874 net238 net019 net023 net022 net018 net229
           net025 net012 net0702 net235 net026 a_c3_b2
XI30 net264 net263 net0895 net07 net258 net010 net09 net05 net252
           net012 net092 net0715 net0861 net013 a_c3_b2
XI27 net045 net168 net0122 net0294 net0295 net048 net047 net052 net051
           net046 net050 net0112 net053 net056 net0240 net233
           net049 net159 net055 a_c3_b4
```

```
XI24 net062 net061 net217 net216 net069 net068 net063 net067 dc2
           net070 net073 net081 net079 net0807 net100 net213
           net078 net103 net074 net076 net072 a_c3_b7
XI26 net111 net110 net0120 net106 net105 net0111 net0110 net0105 net95
           net0113 net0112 net0115 net0783 net0122 net0755
           net128 net0108 net0121 net040 net103 net99 net0114 a_c3_b5
XI25 net0125 net0124 net079 net0128 net189 net0132 net0131 net0126
           net180 net070 net0113 net0136 net0805 net0120
           net0144 net156 net0129 net0142 net159 net0137 net184 net0135 a_c3_b5
XI23 net16 net393 net13 net392 c00_7 c01_7 net391 net390 net396 dc1_7 q_c0_b7
XI22 net13 net28 net375 net25 net374 net16 c00_6 c01_6 net373 net372
           net381 dc1_6 q_c0_b6
XI21 net25 net40 net352 net37 net351 net28 c00_5 c01_5 net350 net349
           net360 dc1_5 q_c0_b6
XI20 net37 net52 net329 net49 net328 net40 c00_4 c01_4 net327 net326
           net337 dc1_4 q_c0_b6
XI19 net49 net64 net55 net61 net48 net52 c00_3 c01_3 net51 net53 net57
           dc1_3 q_c0_b6
XI18 net61 net76 net67 net73 net60 net64 c00_2 c01_2 net63 net65 net69
           dc1_2 q_c0_b6
XI17 net73 net88 net79 net85 net72 net76 c00_1 c01_1 net75 net77 net81
           dc1_1 q_c0_b6
XI16 net85 net90 net91 net83 net84 net88 c00_0 net86 net87 net89
           net430 net94 q_c0_b6
XI15 net111 net353 net110 net109 net123 net116 net117 net118 net0783
           net108 net0144 net97 net361 net106 net105 net95
           net348 net100 net119 net0108 net120 net121 net122
           net107 net113 net101 net103 net99 net115 net332
           net331 a_c2_b5
XI13 net045 net330 net168 net334 net119 net120 net121 net122 net0240
           net166 net0755 net154 net338 net048 net047 net050
           net160 net156 net165 net049 net177 net178 net179
           net108 net290 net312 net159 net287 net309 net308 a_c2_b4
XI12 net0125 net376 net0124 net194 net218 net226 net227 net228 net0805
           net107 net0807 net182 net382 net0128 net189 net180
           net188 net123 net0129 net116 net117 net118 net336
           net185 net0137 net184 net333 net355 net199 a_c2_b6
XI11 net062 net394 net061 net219 net081 net209 net208 net217 net216
           net067 net215 net218 net213 net226 net227 net228
           net359 net212 net074 net076 net356 net378 net224 a_c2_b7
XI10 net245 net284 net244 net243 net239 net248 net249 net250 net261
           net0848 net231 net230 net238 net019 net229 net236
           net233 net260 net235 net265 net240 net0265 net0850
           net429 net234 net406 a_c2_b2
XI9 net264 net255 net263 net262 net260 net265 net0876 net413 net253
```

```
           net07 net258 net252 net400 net041 net0861 net261
           net0848 net0874 net266 a_c2_b1
XI8 net278 net426 net082 net427 net269 net431 net086 net273 net267
           net421 net0876 net0895 a_c2_b0
XI2 net67 net60 net300 net301 net318 net299 net63 net65 net231 net230
           net236 net69 net290 net129 net243 net323 net324
           net295 net322 net287 net286 net285 net284 a_c1_b5
XI3 net55 net48 net323 net324 net341 net322 net51 net53 net316 net315
           net132 net57 net113 net312 net311 net346 net347
           net318 net345 net115 net309 net308 net127 a_c1_b5
XI4 net329 net328 net346 net347 net344 net345 net327 net326 net154
           net338 net160 net337 net336 net101 net334 net343
           net342 net341 net340 net333 net332 net331 net330 a_c1_b5
XI5 net352 net351 net343 net342 net384 net340 net350 net349 net97
           net361 net348 net360 net359 net185 net109 net387
           net388 net344 net386 net356 net355 net199 net353 a_c1_b5
XI6 net375 net374 net387 net388 net385 net386 net373 net372 net182
           net382 net188 net381 net212 net194 net384 net378
           net224 net376 a_c1_b6
XI7 net393 net392 net391 net390 net209 net208 net215 net396 net219
           net385 net394 a_c1_b7
XI1 net79 net72 net434 net295 net433 net75 net77 net413 net253 net400
           net81 net410 net234 net262 net300 net301 net435
           net299 net145 net406 net255 a_c1_b1
XI0 net91 net84 net435 net87 net89 net269 net431 net421 net430 net429
           net266 net427 net434 net433 net426 a_c1_b0
.ends a_add

R7 dc0_7 w10 270.00m
R6 dc0_6 w10 270.00m
R5 dc0_5 w10 270.00m
R4 dc0_4 w10 270.00m
R3 dc0_3 w10 270.00m
R2 dc0_2 w10 270.00m
R1 dc0_1 w10 270.00m
R0 dc0_0 w10 270.00m
L2 net0154 net0152 1p
L5 net0148 net0146 1p
L6 net0146 net0144 1p
L7 net0144 net0142 1p
L3 net0152 net0150 1p
L0 s1 net0156 1p
L1 net0156 net0154 1p
L4 net0150 net0148 1p
L8 net0172 net0198 1p
```

```
XI6 net0188 net58 net59 net0182 net0342 net0343 net0344 net0174 net49
            net0185 net0332 net0333 net53 net0335 net0336
  net0337 net14 net15 net16 net0170 net0169 net19
  net20 net0166 net5 net6 net7 net0161 net9 net0159
  net0158 net0157 on_input_wireup
XI5 c10_7 c10_6 c10_5 c10_4 c10_3 c10_2 c10_1 c10_0 c11_7 c11_6 c11_5
          c11_4 c11_3 c11_2 c11_1 c11_0 net0172 net0199 net37
  net36 net0202 net0195 net33 net0204 net30 net0190
  net28 net27 net0193 net25 net24 net23 on_output_wireup
XI4 net5 net6 net7 net0161 net9 net0159 net0158 net0157 e9 on_50_ohm_wps_40pin
XI3 net14 net15 net16 net0170 net0169 net19 net20 net0166 e2 on_50_ohm_wps_40pin
XI2 net23 net24 net25 net0193 net27 net28 net0190 net30 s2 on_50_ohm_wps_40pin
XI1 net0198 net0199 net37 net36 net0202 net0195 net33 net0204 n2
                        on_50_ohm_wps_40pin
XI0 s7 s6 net0174 net0344 net0343 net0342 net0182 net59 net58 net0188
             net0337 net0336 net0335 net53 net0333 net0332
  net0185 net49 c10_0 c10_1 c10_2 c10_3 c10_4 c10_5
  c10_6 c10_7 c11_0 c11_1 c11_2 c11_3 c11_4 c11_5
  c11_6 c11_7 dc0_0 dc0_1 dc0_2 dc0_3
+ dc0_4 dc0_5 dc0_6 dc0_7 net0156 net0154 net0152 net0150 net0148
            net0146 net0144 net0142 n7 n6 w9 w8 w7 w6 w5 w4 w3
  w2 s8 a_add

.end
```

Bibliography

[1] Wikipedia, Benjamin franklin — wikipedia, the free encyclopedia, 2011, [Online; accessed 14-October-2011].

[2] Wikipedia, Eniac — wikipedia, the free encyclopedia, 2011, [Online; accessed 14-October-2011].

[3] H. Kamerlingh Onnes, Koninklijke Nederlandse Akademie van Weteschappen Proceedings Series B Physical Sciences **13**, 1274 (1910).

[4] K. Likharev, *Dynamics of Josephson Junctions and Circuits*, Gordon and Breach Science Publishers, New York, 1986.

[5] R. C. Jaklevic, J. Lambe, J. E. Mercereau, and A. H. Silver, Phys. Rev. **140**, A1628 (1965).

[6] A. Barone and G. Paternò, *Physics and applications of the Josephson effect*, UMI books on demand, Wiley, 1982.

[7] Y. Taur, IBM Journal of Research and Development **46**, 213 (2002).

[8] O. T. Oberg, Q. P. Herr, A. G. Ioannidis, and A. Y. Herr, IEEE Transactions on Applied Superconductivity **21**, 571 (2011).

[9] M. Tinkham, *Introduction to superconductivity*, Dover books on physics and chemistry, Dover Publications, second edition, 2004.

[10] F. London and H. London, Proceedings of the Royal Society of London. Series A-Mathematical and Physical Sciences **149**, 71 (1935).

[11] J. Bardeen, L. Cooper, and J. Schrieffer, Physical Review **108**, 1175 (1957).

[12] B. Josephson, Physics Letters **1**, 251 (1962).

[13] P. W. Anderson and J. M. Rowell, Phys. Rev. Lett. **10**, 230 (1963).

[14] S. Anders et al., Physica C: Superconductivity **470**, 2079 (2010), European Roadmap on Superconductor Electronics - Status and Perspectives.

[15] K. Likharev and V. Semenov, IEEE Transactions on Applied Superconductivity **1**, 3 (1991).

[16] W. Chen, A. V. Rylyakov, V. Patel, J. E. Lukens, and K. K. Likharev, Applied Physics Letters **73**, 2817 (1998).

[17] S. Yorozu, Y. Kameda, Y. Hashimoto, and S. Tahara, IEEE Transactions on Applied Superconductivity **13**, 450 (2003).

[18] Q. P. Herr, A. D. Smith, and M. S. Wire, Applied Physics Letters **80**, 3210 (2002).

[19] Y. Hashimoto, S. Yorozu, T. Satoh, and T. Miyazaki, Applied Physics Letters **87**, 022502 (2005).

[20] A. Inamdar et al., IEEE Transactions on Applied Superconductivity **19**, 670 (2009).

[21] I. V. Vernik et al., Superconductor Science and Technology **20**, S323 (2007).

[22] Y. Hashimoto, S. Yorozu, and Y. Kameda, IEICE Trans. on Electronics **91**, 325 (2008).

[23] S. Intiso et al., IEEE Transactions on Applied Superconductivity **15**, 328 (2005).

[24] Applied Superconductivity Conference, *Reciprocal Quantum Logic*, 2008.

[25] N. S. Agency, NSA superconducting technology assessment, 2005, `http://www.nitrd.gov/pubs/nsa/sta.eps`.

[26] Public Law 109-431, Report to congress on server and data center energy efficiency, 2007, `http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf`.

[27] Q. P. Herr, A. Y. Herr, O. T. Oberg, and A. G. Ioannidis, Journal of Applied Physics **109**, 103903 (2011).

[28] S. Shiva, *Introduction to Logic Design*, M. Dekker, New York, 1998.

[29] HYPRES Design Rules, Available: `http://www.hypres.com`.

[30] A. Vayonakis, C. Luo, H. Leduc, R. Schoelkopf, and J. Zmuidzinas, The millimeter-wave properties of superconducting microstrip lines, in *AIP Conference Proceedings*, pages 539–542, Citeseer, 2002.

[31] M. Bin, M. Gaidis, J. Zmuidzinas, T. Phillips, and H. LeDuc, Applied physics letters **68**, 1714 (1996).

[32] M. Johnson et al., Superconductor Science and Technology **23**, 065004 (2010).

[33] T. Satoh, K. Hinode, S. Nagasawa, Y. Kitagawa, and M. Hidaka, IEEE Transactions on Applied Superconductivity **17**, 169 (2007).

[34] Intel, Intel core i7 processor — integration overview (lga1366-land package), `http://www.intel.com/support/processors/corei7/sb/CS-030866.htm`.

[35] J. Charles, P. Jassi, N. Ananth, A. Sadat, and A. Fedorova, Evaluation of the intel® core$^{\text{TM}}$ i7 turbo boost feature, in *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*, pages 188–197, IEEE, 2009.

[36] P. Bunyk and P. Litskevitch, IEEE Transactions on Applied Superconductivity **9**, 3714 (1999).

[37] L. W. Nagel and D. Pederson, Spice (simulation program with integrated circuit emphasis), Technical Report UCB/ERL M382, EECS Department, University of California, Berkeley, 1973.

[38] P. Horowitz and W. Hill, *The art of electronics*, volume 2, Cambridge university press Cambridge, 1989.

[39] D. Pozar, *Microwave engineering*, Wiley-India, 2009.

[40] S. Cohn, IEEE Trans. on Microw. Theory Tech. **MTT-16**, 110 (1968).

[41] S. R. Whiteley, IEEE Trans. Magn. **27**, 2902 (1991).

[42] M. Elsbury et al., IEEE Trans. on Microwave Theory and Techn. **57**, 2055 (2009).

[43] M. Elsbury, P. Dresselhaus, S. Benz, and Z. Popovic, Integrated broadband lumped-element symmetrical-hybrid n-way power dividers, in *Microwave Symposium Digest, 2009. MTT'09. IEEE MTT-S International*, pages 997–1000, IEEE.

[44] P. Kogge and H. Stone, IEEE Transactions on Computers **100**, 786 (1973).

[45] S. Knowles, A family of adders, in *14th IEEE Symposium on Computer Arithmetic, 1999. Proceedings.*, pages 30–34, IEEE, 1999.

[46] A. Silver et al., Superconductor Science and Technology **16**, 1368 (2003).

[47] J. M. Martinis and R. L. Kautz, Physical Review Letters **63**, 1507 (1989).

[48] C. Bell et al., Applied physics letters **84**, 1153 (2004).

[49] T. Khaire, M. Khasawneh, W. Pratt Jr, and N. Birge, Physical review letters **104**, 137002 (2010).

[50] H. Hilgenkamp, Superconductor Science and Technology **21**, 034011 (2008).

[51] M. Dorojevets, On the road towards superconductor computers: Twenty years later, Technical report, DTIC Document, 2004.

[52] M. Tanaka et al., IEEE Transactions on Applied Superconductivity **21**, 1 (2011).

[53] E. Farhi et al., A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem, 2001.