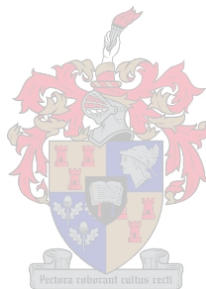


A 10 GHz Oversampling Delta Modulating Analogue-to-Digital Converter Implemented with Hybrid Superconducting Digital Logic

Coenrad Johann Fourie



*Thesis presented in partial fulfilment of the requirements for the degree of **Master of Science in Engineering** at the University of Stellenbosch.*

**Advisors:
Prof. W. J. Perold
Dr J. B. de Swardt**

March 2001

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work, unless stated otherwise, and that I have not previously in its entirety or in part submitted it at any university for a degree.

Date: 12 February 2001

Abstract

Rapid Single Flux Quantum (RSFQ) logic cells are discussed, and new cells developed. The expected yield of every cell is computed through a Monte Carlo analysis, and where necessary these cells are optimized for use in a complex system. A mathematical study of the Josephson junction and SQUIDs (Superconducting Quantum Interference devices) as switching elements precede a discussion on the operation of RSFQ and COSL (Complementary Output Switching Logic.) These logic families are implemented in low temperature niobium technology, and require liquid helium cooling. A 10 GHz oversampling delta modulating analogue-to-digital converter is then designed and constructed using RSFQ and COSL building blocks in a hybrid configuration. The design emphasis is on devising ways to test the operation of RSFQ with limited equipment. Yield analysis procedures on the complex system are discussed, followed by a detailed discussion on the circuit layout and layout problems. Software routines are developed to calculate the required dimensions of layout structures.

Opsomming

Rapid Single Flux Quantum (RSFQ) logiese selle word bespreek, en enkele nuwe selle word ontwikkel. Die verwagte opbrengs, of kans dat 'n sel sal werk, word bereken deur 'n Monte Carlo analise. Waar nodig word selle met behulp van die analise verbeter vir gebruik in 'n komplekse stelsel. 'n Wiskundige studie van die Josephson-vlak en SQUIDs (*Superconducting Quantum Interference devices*) word gevolg deur 'n bespreking oor die werking van RSFQ en COSL (*Complementary Output Switching Logic*.) Hierdie logiese families word geïmplementeer in laetemperatuur niobiumtegnologie, en vereis vloeibare helium-verkoeling. 'n Delta-modulerende analoog-na-digitale omsetter met 'n intree-monstertempo van 10 GHz word ontwerp en vervaardig met 'n hibriede samestelling van RSFQ en COSL boublokke. Die ontwerp fokus op maniere om die werking van RSFQ teen 10 GHz te kan toets met die beperkte toerusting wat beskikbaar is. Opbrengsanalise op die komplekse stelsel word bespreek, gevolg deur 'n volledige bespreking van die stroombaanuitlegprosedure en uitlegprobleme. Roetines word in sagteware ontwikkel om die nodige dimensies van uitlegstrukture te bereken.

Acknowledgements

I am very thankful to Professor W.J. Perold for his endless patience and understanding when several new ideas, and the repeated promises, “I’ll have this done by next week!” did not always deliver the expected results, and also when simulation software got the better of me.

I would like to thank Professor J.G. Lourens for patiently explaining to me the more advanced theory of DSP, and for generating valuable ideas when I hit brick walls.

Thanks should also go to Dr J.B. de Swardt and the other members of the Microwave Data Link research team for listening to my problems over cappuccino. Special thanks must go to Nico Geldenhuys for enthusiastically feeding my transmission line parameters into his copy of Microwave Office 2000 to verify my impedance assumptions.

I would also like to thank F.J. Rabie for handling all my problems with Baumann, for explaining WRSpice to me, and for allowing me the use of his coupled inductor layout schematic in my COSL layouts.

Thanks goes to my Creator for two blessed years and a clear mind granted to me for dedication to this project.

The final acknowledgement is to the National Research Foundation of South Africa for providing funding for my post-graduate studies.

Contents

LIST OF FIGURES	VIII
LIST OF TABLES	XII
NOMENCLATURE	XIII
CHAPTER 1 – INTRODUCTION	1
CHAPTER 2 – QUANTUM THEORY	3
2.1 INTRODUCTION	3
2.2 THE JOSEPHSON JUNCTION	3
2.2.1 <i>Basic Josephson junction</i>	3
2.2.2 <i>Generalized Josephson junction</i>	4
2.3 ONE-JUNCTION SQUID	5
2.4 TWO-JUNCTION SQUID.....	7
2.5 THE JOSEPHSON TRANSMISSION LINE AND RSFQ LOGIC	9
2.6 COSL.....	10
CHAPTER 3 – BUILDING BLOCKS	12
3.1 INTRODUCTION	12
3.2 RSFQ LOGIC.....	12
3.2.1 <i>Basic asynchronous components</i>	12
3.2.1.1 JTL.....	12
3.2.1.2 Pulse splitter	18
3.2.1.3 Pulse merger	19
3.2.2 <i>Registers and latches</i>	21
3.2.2.1 Destructive Readout register (DRO)	21
3.2.2.2 T-flip-flop	23
3.2.2.3 T1-flip-flop	24
3.2.2.4 DRO with 2 readouts	26
3.2.3 <i>Logic gates</i>	28
3.2.3.1 Inverter (NOT-gate).....	28
3.2.3.2 OR-gate.....	29
3.2.3.3 AND-gate.....	31
3.2.3.4 XOR-gate.....	33
3.2.4 <i>Interface devices</i>	35
3.2.4.1 DC-to-SFQ converter	35
3.2.4.2 SFQ-to-DC converter	36
3.3 MISCELLANEOUS CIRCUIT COMPONENTS.....	38
3.3.1 <i>COSL gates</i>	38
3.3.1.1 COSL OR-gate	38
3.3.1.2 COSL NOR-gate.....	39
3.3.1.3 Negative output COSL OR-gate.....	41
3.3.2 <i>Comparator</i>	41
3.3.3 <i>RSFQ DRO to COSL OR-gate interface</i>	44
3.4 SIMULATIONS	46
3.5 MONTE CARLO ANALYSIS AND DEVICE OPTIMIZATION	47

3.5.1	<i>Definition of Monte Carlo analysis and yield prediction</i>	47
3.5.2	<i>Definition of a circuit failure</i>	48
3.5.3	<i>Yield results and optimization procedures</i>	49
3.5.4	<i>Layout extraction technique</i>	52
3.6	DEVICE STATISTICS.....	54
3.7	CONCLUSIONS.....	56
CHAPTER 4 – SYSTEM DESIGN		57
4.1	INTRODUCTION.....	57
4.2	PROBLEMS REGARDING TESTING.....	57
4.3	FREQUENCY MEASURING TECHNIQUES AND FUNCTION SELECTION.....	58
4.4	DSP THEORY AND DELTA MODULATION.....	58
4.5	CONCEPTUAL DESIGN OF ADC WITH AVAILABLE LOGIC CELLS.....	62
4.6	DESIGN OF COMPLEX SUBCIRCUITS.....	66
4.6.1	<i>Full Adder</i>	66
4.6.2	<i>Shift register cell with SFQ-to-DC converter</i>	69
4.6.3	<i>Shift register with inverted feedback</i>	70
4.6.4	<i>Counter with RESET</i>	72
4.7	CLOCKING.....	73
4.8	SYSTEM YIELD OPTIMIZATION (LARGE SCALE MONTE CARLO ANALYSIS).....	74
4.9	OVERALL SYSTEM SPECIFICATIONS.....	75
4.10	TEST BED (FOR FUTURE CIRCUITS).....	77
4.11	CONCLUSIONS.....	77
CHAPTER 5 – SYSTEM TESTING AND SIMULATION RESULTS		79
5.1	SIMULATION RESULTS FOR ADC.....	79
5.1.1	<i>WRSpice simulations</i>	79
5.1.2	<i>Matlab simulations for ADC</i>	80
5.2	PHYSICAL TESTING.....	82
5.2.1	<i>Test-bed setup</i>	82
5.2.2	<i>Testing signals and expected measurements</i>	82
5.3	FAULT ANALYSIS AND RESULT INTERPRETATION.....	85
5.4	CONCLUSIONS.....	86
CHAPTER 6 – CIRCUIT LAYOUT		87
6.1	HYPRES LAYOUT PROCEDURE AND DESIGN RULES.....	87
6.2	LAYOUT DIFFICULTIES AND SOLUTIONS.....	91
6.2.1	<i>Minimum parasitic inductance between JJ and ground</i>	91
6.2.2	<i>DC bias resistor and line layout</i>	93
6.3	FLUX TRAPPING AND MOATS.....	94
6.4	LAYOUT DIAGRAMS.....	94
6.5	CONCLUSIONS.....	105
CHAPTER 7 - CONCLUSIONS AND FUTURE		107
7.1	CONCLUSIONS.....	107
7.2	PROBLEMS AND ALTERNATIVE SOLUTIONS.....	107
7.3	FUTURE AND APPLICATIONS.....	108
REFERENCES		110

APPENDIX A – SUPERCONDUCTING MATHEMATICAL TOOLBOX	114
IMPEDANCE CALCULATOR – THEORY AND SOURCE CODE	114
SOURCE CODE FOR JJ AREA PARAMETERS	117
SOURCE CODE FOR DAMPING RESISTANCE CALCULATOR	128
APPENDIX B - MATLAB SOURCE CODE FOR ADC VERIFICATION.....	129
APPENDIX C - IMPEDANCE OF COPLANAR WAVEGUIDE (CPW).....	134
APPENDIX D – MONTE CARLO SIMULATION FILES	136
WRSPICE SIMULATION FILE FOR RSFQ AND-GATE	136
WRSPICE SIMULATION FILE FOR DRO-COSL INTERFACE	139
HSPICE SIMULATION FILE FOR DC-TO-SFQ CONVERTER BASED ON PARAMETER TOLERANCE	141
HSPICE SIMULATION FILE FOR DC-TO-SFQ CONVERTER BASED ON LAYOUT EXTRACTION.....	143
HSPICE LIBRARY FILES	147
APPENDIX E – LOGIC FUNCTIONS AND TRUTH TABLES	150

List of Figures

Figure 2.1:	Physical structure of (a) weak link and (b) SIS Josephson junction	3
Figure 2.2:	Circuit model for generalized Josephson junction	4
Figure 2.3:	Voltage-current relationship of generalized Josephson junction for (a) $\beta_C \ll 1$, (b) $\beta_C \approx 1$ and (c) $\beta_C \gg 1$	5
Figure 2.4:	Circuit diagram of one-junction SQUID	6
Figure 2.5:	Inductor current against input current for one-junction SQUID	6
Figure 2.6:	Two-junction SQUID	7
Figure 2.7:	Cascade of JTLs	9
Figure 2.8:	Simulated pulse propagation along a cascade of JTLs	9
Figure 2.9:	Schematic of basic COSL gate	11
Figure 3.1:	Circuit diagram for 250 μA Josephson junction	13
Figure 3.2:	Circuit diagram for (a) 250 μA -in 355 μA -out and (b) 355 μA Josephson junction	14
Figure 3.3:	(a) Simulation test setup and (b) simulated response of a cascade of 250 μA JTLs	14
Figure 3.4:	Simulated voltage and current pulses between two 250 μA JTLs	15
Figure 3.5:	Circuit diagram for two 250 μA JTLs connected through a matched TX line	15
Figure 3.6:	(a) Simulation test setup of a cascade of 250 μA JTLs connected through matched TX lines	16
Figure 3.6:	(b) Simulated response of a cascade of 250 μA JTLs connected through matched TX lines	17
Figure 3.7:	Circuit diagram for RSFQ pulse splitter	18
Figure 3.8:	(a) Simulation test setup and (b) simulated response of RSFQ pulse splitter to SFQ input pulse	19
Figure 3.9:	Circuit diagram for RSFQ pulse merger	20
Figure 3.10:	(a) Simulation test setup of RSFQ pulse merger	20
Figure 3.10:	(b) Simulated response of RSFQ pulse merger to SFQ input pulses	21
Figure 3.11:	Circuit diagram for RSFQ DRO	22
Figure 3.12:	(a) Simulation test setup and (b) simulated response of RSFQ DRO-register to SFQ set and reset pulses	22
Figure 3.13:	Circuit diagram for RSFQ T-flip-flop	23
Figure 3.14:	(a) Simulation test setup and (b) simulated response of a cascade of three RSFQ T-flip-flops to SFQ toggle pulses	24
Figure 3.15:	Circuit diagram for RSFQ T1-flip-flop	25
Figure 3.16:	(a) Simulation test setup and (b) simulated response of RSFQ T1-flip-flop to SFQ toggle and destructive read pulses	26
Figure 3.17:	Circuit diagram for RSFQ DRO with 2 readouts	27
Figure 3.18:	(a) Simulation test setup and (b) simulated response of RSFQ DRO with 2 readouts to SFQ clock and input pulses	27
Figure 3.19:	Circuit diagram for RSFQ NOT-gate (inverter)	28
Figure 3.20:	(a) Simulation test setup and (b) simulated response of RSFQ NOT-gate to SFQ clock and input pulses	29

Figure 3.21:	Circuit diagram for RSFQ OR-gate	30
Figure 3.22:	(a) Simulation test setup and (b) simulated response of RSFQ OR-gate to SFQ clock and input pulses	30
Figure 3.23:	Circuit diagram for obsolete first design of RSFQ AND-gate	31
Figure 3.24:	Circuit diagram for RSFQ AND-gate	32
Figure 3.25:	(a) Simulation test setup of RSFQ AND-gate	32
Figure 3.25:	(b) Simulated response of RSFQ AND-gate to SFQ clock and input pulses	33
Figure 3.26:	Circuit diagram for RSFQ XOR-gate	34
Figure 3.27:	(a) Simulation test setup and (b) simulated response of RSFQ XOR-gate to SFQ clock and input pulses	34
Figure 3.28:	Circuit diagram for DC-to-SFQ converter	35
Figure 3.29:	(a) Simulation test setup and (b) simulated response of DC-to-SFQ converter to sinusoidal input voltage	36
Figure 3.30:	Circuit diagram for SFQ-to-DC converter	37
Figure 3.31:	(a) Simulation test setup and (b) simulated response of SFQ-to-DC converter to SFQ input pulses	37
Figure 3.32:	Circuit diagram for COSL OR-gate	38
Figure 3.33:	Simulated response of COSL OR-gate to applied input signal	39
Figure 3.34:	Circuit diagram for COSL NOR-gate	40
Figure 3.35:	Simulated response of COSL NOR-gate to applied input signal	40
Figure 3.36:	Simulated dynamics of negative output COSL OR-gate	41
Figure 3.37:	“Stripped COSL” comparator schematic	43
Figure 3.38:	Circuit diagram for level detector	44
Figure 3.39:	Simulated response of level detector to (a) feedback signals and zero analogue input, and (b) ramp analogue input and no feedback	44
Figure 3.40:	Circuit diagram for DRO-to-COSL OR-gate interface	45
Figure 3.41:	(a) Simulation test setup and (b) simulated response of DRO to COSL OR-gate interface to input signals	45
Figure 3.42:	Test setup for simulating RSFQ devices	50
Figure 3.43:	Results for layout extraction simulation models of (a) meticulous and (b) careless layout of DC-to-SFQ converter	54
Figure 4.1:	Delta modulation system	59
Figure 4.2:	Delta modulator with discrete-time integrator	59
Figure 4.3:	Delta modulation and two types of quantization errors	60
Figure 4.4:	Delta modulator with FIR feedback and parallel digital output	61
Figure 4.5:	Simplified block diagram for 4-bit ADC	62
Figure 4.6:	Block diagram of serial adder	64
Figure 4.7:	Schematic diagram for 15-bit adder	65
Figure 4.8:	Key to schematic diagram symbols	66
Figure 4.9:	Logic diagram of full adder	67
Figure 4.10:	Schematic diagram of RSFQ full adder with (a) three inputs and (b) two inputs and carry feedback	68
Figure 4.11:	Simulated response of (a) three-input full adder and (b) full adder with carry feedback to SFQ input pulses	69
Figure 4.12:	Schematic diagram of dc output cell	70
Figure 4.13:	Simulated response of dc output cell to SFQ input pulses	70
Figure 4.14:	Schematic diagram of shift register cell with inverted feedback and eighth clock readout	71

Figure 4.15:	Simulated response of shift register cell with inverted feedback and eighth clock readout to input signals	71
Figure 4.16:	Schematic diagram of first three feedback stages	72
Figure 4.17:	Schematic diagram showing two cells of four-stage counter	73
Figure 4.18:	Schematic diagram of (a) divide-by-two and (b) divide-by-eight clock scalers	74
Figure 4.19:	Schematic diagram of ADC	76
Figure 5.1:	Simulation results for full electrical circuit model of ADC	80
Figure 5.2:	Simulated results for ADC from mathematical description	81
Figure 5.3:	(a) Reconstructed digital output signal and (b) corresponding power spectrum for ADC simulation	82
Figure 5.4:	(a) Power versus time and (b) power spectrum for output of periodically flipped SFQ-to-DC converter into 50 Ω load	83
Figure 5.5:	Simulated power spectrum for bit 3 of ADC output with a zero input signal	84
Figure 5.6:	Simulated power spectrum for bit 3 of ADC output to analogue input sinusoid of 110 MHz, 3 mV	84
Figure 5.7:	(a) Voltage-time and (b) power spectrum plots for a 110 MHz square wave constructed from a 1.25 GHz SFQ pulse train	85
Figure 5.8:	(a) Reconstructed digital output signal and (b) corresponding power spectrum for ADC simulation with 3 defective feedback cells	86
Figure 6.1:	Hypres niobium process layer description	87
Figure 6.2:	Layout for damped JJ by strict adherence to design rules	91
Figure 6.3:	Step-by-step schematic of minimum inductance layout procedure for grounded JJ with damping resistor	92
Figure 6.4:	Step-by-step schematic of minimum inductance layout procedure for a serial JJ with damping resistor	92
Figure 6.5:	Three dimensional representation of minimum inductance layout technique for grounded JJ with damping resistor	93
Figure 6.6:	Three dimensional representation of minimum inductance layout technique for serial JJ with damping resistor	93
Figure 6.7:	Schematic showing chip layout and pin assignment	95
Figure 6.8:	Layout schematics for (a) smallest 250 μ A JTL, (b) flattened 250 μ A JTL and (c) right-angled 250 μ A JTL	97
Figure 6.9:	Layout schematics for (a) 250 μ A-in 355 μ A-out JTL and (b) 355 μ A JTL	98
Figure 6.10:	Layout schematics for (a) RSFQ pulse splitter and (b) RSFQ pulse merger	98
Figure 6.11:	Layout schematic for RSFQ DRO-register	99
Figure 6.12:	Layout schematic for RSFQ T1 flip-flop	99
Figure 6.13:	Layout schematic for RSFQ NOT-gate	100
Figure 6.14:	Layout schematic for RSFQ OR-gate	100
Figure 6.15:	Layout schematic for RSFQ AND-gate	101
Figure 6.16:	Layout schematic for RSFQ XOR-gate	101
Figure 6.17:	Layout schematic for DC-to-SFQ converter	102
Figure 6.18:	Layout schematic for SFQ-to-DC converter	102
Figure 6.19:	Layout schematic for level detector	102
Figure 6.20:	Layout schematic for shift register cell with feedback output and RSFQ AND-gate readout	103

Figure 6.21:	Layout schematic for 3-input full adder	104
Figure 6.22:	Incomplete layout schematic for ADC	105
Figure A.1:	Superconducting microstrip configuration	114
Figure A.2:	Three dimensional representation of octagonal JJ	117
Figure A.3:	Layer II A area of (a) square, (b) circular and (c) octagonal JJ	118
Figure C.1:	Coplanar waveguide	134

List of Tables

Table 3.1:	Parameter tolerances for niobium IC	49
Table 3.2:	Theoretical yield results for all building blocks	52
Table 3.3:	Physical parameters for all building blocks	55
Table 3.4:	Nominal latency statistics (clock-to-output) for synchronous devices	55
Table 3.5:	Nominal latency statistics (input-to-output) for asynchronous devices	56
Table 4.1:	Truth table for full adder	67
Table 4.2:	Yield results for complex subsystems	75
Table 4.3:	Physical statistics for complex subcircuits	76
Table 6.1:	Tolerance values for 3 μm niobium process from Hypres	87
Table 6.2:	Bias from mask for I1A level in 3 μm niobium process from Hypres (valid for square junctions only)	87
Table 6.3:	Transmission line parameters for M1 layer	88
Table 6.4:	Transmission line parameters for M2 layer	88
Table 6.5:	Transmission line parameters for M3 layer	88
Table 6.6:	Specifications for various CPW structures	90
Table E.1:	Definition of standard logic functions	151
Table E.2:	Switching-algebra theorems	152

Nomenclature

A	Magnetic vector potential.
AC	Alternating Current.
A/D	Analogue-to-Digital.
ADC	Analogue-to-Digital Converter.
AM	Amplitude Modulation.
β_C	Stewart-McCumber parameter.
COSL	Complementary Output Switching Logic.
CPW	Coplanar Waveguide.
dc	Direct Current.
Δ	Step size in delta modulator.
DM	Delta Modulation / Delta Modulator.
DRO	Destructive Readout register.
DSP	Digital Signal Processing.
DUT	Device Under Test.
ϵ_r	Relative dielectric constant.
f	Femto (1×10^{-15} .)
FA	Full Adder.
FEC	Forward Error Correction.
FFT	Fast Fourier Transform.
FIR	Finite Impulse Response.
FM	Frequency Modulation.
G	Giga (1×10^9 .)
<i>i</i>	Current as a variable.
IC	Integrated Circuit.
I_C	Josephson Junction critical current.
J_C	Josephson Junction critical current density.
JJ	Josephson Junction.
JTL	Josephson Transmission Line.
K	Fringe field factor in microstrip theory.
<i>k</i>	Confidence level constant for Monte Carlo analysis.
<i>L</i>	Confidence interval for Monte Carlo analysis.
λ	London penetration depth of a superconductor.
LSB	Least Significant Bit.
μ	Micro (1×10^{-6} .)
μ_0	Permeability of free space ($4\pi \times 10^{-7}$ H/m.)
m	Milli (1×10^{-3} .)
M	Mega (1×10^6 .)
MCM	Multi-Chip Module.
MSB	Most Significant Bit.
n	Nano (1×10^{-9} .)
p	Pico (1×10^{-12} .)
PEC	Perfect Electrical Conductor.
φ	Gauge-invariant phase difference across a Josephson junction.
Φ_0	Magnetic flux quantum (2.07×10^{-15} Wb.)

ρ	Reflection coefficient (ratio of voltage in reflected wave to that in the incident wave).
RF	Radio Frequency.
RSFQ	Rapid Single Flux Quantum.
SDM	Sigma-Delta Modulation.
SFQ	Single Flux Quantum.
SI	<i>Système Internationale</i> – The international system of standard measurements.
SQUID	Superconducting Quantum Interference Device.
τ_J	Time constant for basic Josephson junction.
τ_{RC}	Time constant for resistively loaded capacitor.
t	Time as a variable.
T_C	Critical Temperature of a superconductor.
TX	Transmission.
v	Voltage as a variable.
VLSI	Very Large Scale Integration.
y	True statistical yield.
y'	Observed yield of Monte Carlo analysis.
Z_0	Characteristic Impedance (of transmission line.)

Chapter 1 – Introduction

Superconductivity was discovered in 1911 by Heike Kamerlingh Onnes [1] while he was doing research on the resistivity of metals at cryogenic temperatures. Ever since that very first sample of distilled mercury was measured to have no dc resistance at temperatures below 4.2 K, scientists and engineers have been fascinated by the possibilities opened by this technology.

In 1933, Walther Meissner and Robert Ochsenfeld discovered that a superconducting material expels magnetic flux [2]. The Meissner effect, as this phenomenon is now known, is what really sets superconductors apart from any other metallic conductors or theoretical perfect conductors.

The behaviour of superconductors are characterised by the London equations, developed in 1935 by Fritz and Heinz London. These expressions are phenomenological only, and do not explain superconductivity [1], [3].

The origin of the superconducting state was first explained in 1957 by John Bardeen, Leon Cooper and Robert Schrieffer [4]. In their BCS theory of superconductivity, electrons in the superconducting state move around in groups of two, called a Cooper pair, in a bound state.

By 1972, despite all the progress made on the theoretical explanation of superconductivity, the highest measured critical temperature for a superconductor was still only 20.8 K [5].

At the beginning of 1986, the year that Karl Alex Müller and J. Georg Bednorz discovered superconductivity in an oxide of lanthanum, barium and copper, the highest known critical temperature for any superconductor was 23.3 K for a combination of niobium and germanium (Nb_3Ge) [6]. This limited superconductor applications due to the stringent demand for cryogenic cooling with liquid hydrogen or liquid helium.

The critical temperature (T_C) in the oxide discovered by Müller and Bednorz was a mere 35 K [6], but the discovery was so important that it earned them the 1987 Nobel Prize for physics. It also triggered a surge in research in the field of superconductors, and for a short time the record for highest recorded T_C increased in leaps and bounds. In January 1987, $\text{Yb}_a\text{Cu}_3\text{O}_{7.8}$ (YBCO) was demonstrated to have a T_C of 95 K, high enough to allow the superconducting state to be sustained with liquid nitrogen cooling [7].

Although progress on high- T_C superconductors has slowed down in recent years, and the highest T_C for any known superconductor was but 135 K for the compound mercury barium calcium copper oxide (HBCCO) in 1997 [8], the use of superconductors in equipment that influence our everyday lives is continuing to increase. Examples include MRI (Magnetic Resonance Imaging) for non-invasive body scans [9], and no-loss power cables in utility networks [10].

The most important discovery in the field of superconductivity, from an electronic engineering point of view, must arguably be the Josephson effect. Predicted in 1962 by Brian D. Josephson [11], and verified shortly thereafter [12], this effect led to the development of a whole field of electronics. Superconducting logic families built around the Josephson junction are currently the fastest digital devices ever invented by mankind. With conventional semiconductor logic families fast approaching their physical speed limits, and the interest in quantum electronics rising rapidly, superconducting electronics are set to carry on the digital revolution.

Among the families of superconducting digital logic devices, two stand out very prominently. The first is Complementary Output Switching Logic (COSL) [13], [14]. This is currently the fastest voltage state logic in existence, and has been tested at clock speeds of up to 18 GHz [15]. The second family of great importance is Rapid Single Flux Quantum (RSFQ) [16]. RSFQ is a voltage pulse logic family, and single RSFQ devices have already been tested at clock frequencies of up to 770 GHz [17]. Compared to the current high-end speed limits of around 1 GHz for commercial semiconducting processors, superconducting logic families hold a clear speed advantage.

This thesis discusses the design of a system using RSFQ superconducting logic. Several logic gates and pulse manipulation devices are adapted from the circuit diagrams available in the Stony Brook University cell library [18]. All these devices were tuned for optimal performance depending on where they were used.

The RSFQ AND-gate and OR-gate were not available in public cell libraries and had to be developed and optimized. The optimization of devices using the technique of Monte Carlo analysis is discussed, with particular attention to the case of the RSFQ AND-gate.

The implementation of a four-bit oversampling delta modulating analogue-to-digital converter with RSFQ and COSL devices, as well as the optimization and layout procedures are discussed. This is followed by an overview of DSP theory surrounding A/D conversion and the delta modulator.

Included in the system design is the development of a level detector or comparator, the design of a full adder and the design of an SFQ-to-voltage state converter for interfacing RSFQ outputs with off-chip hot-logic devices.

It is not possible to measure or view the digital outputs of RSFQ logic clocked at microwave frequencies directly with the time domain equipment available at most research institutions today. A way is discussed to allow the verification of circuit operation by measuring the RSFQ outputs in the frequency domain.

The system can also be used as a test bed for future projects on RSFQ logic, since it allows for easy insertion of logic blocks into the established frame of the analogue-to-digital converter. The added logic blocks can therefore also be tested by taking measurements in the frequency domain.

All symbols and abbreviations used in this report are defined in the Nomenclature on page xiii.

Chapter 2 – Quantum Theory

2.1 Introduction

Any treatise on the operation of RSFQ logic must be preceded by an essay on the Josephson junction and the one-junction SQUID, since these are the most rudimentary elements in RSFQ superconducting logic.

The two-junction SQUID is an important part of COSL gates [19], and is therefore included in the discussion.

Finally, the operation of the Josephson Transmission Line (JTL) is explained in some detail, followed by a conceptual discussion on COSL.

2.2 The Josephson Junction

2.2.1 BASIC JOSEPHSON JUNCTION

A Josephson junction can be made in one of two ways. The first is to use a weak link, where a superconducting line is narrowed to create a junction that limits a passing supercurrent to a small cross-section. The current density across the junction can then be made to exceed the critical current density (J_C) of the superconducting material.

The second and most popular technique is to use electron tunneling across a superconducting-insulating-superconducting (SIS) barrier to achieve the Josephson effect [3].

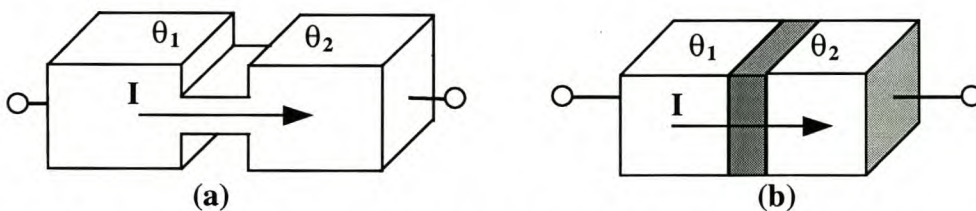


Figure 2.1: Physical structure of (a) weak link and (b) SIS Josephson junction

For a basic lumped Josephson junction, the current i through the junction is given by [3]

$$i = I_C \sin \varphi(t) \quad (2.1)$$

with the gauge-invariant phase difference given by

$$\varphi(t) = \theta_1(t) - \theta_2(t) - \frac{2\pi}{\Phi_0} \int_1^2 \mathbf{A}(\mathbf{r}, t) \cdot d\mathbf{l} \quad (2.2)$$

and the voltage-phase relation by

$$\frac{\partial \varphi(t)}{\partial t} = \frac{2\pi}{\Phi_0} v \quad (2.3)$$

The magnetic flux quantum Φ_0 equals $h/2e$, or 2.068×10^{-15} Wb.

From equations (2.1) and (2.3) it can be seen that a zero voltage across the junction results in a constant current. This is called the DC Josephson effect [1].

Similarly, a constant voltage applied across the junction results in an oscillating current. The oscillation frequency is a constant for a constant voltage, and the relation is $483.598 \text{ MHz}/\mu\text{V}$. This is referred to as the AC Josephson effect [3].

The critical current I_C cannot be exceeded in the basic junction, so that this model only allows supercurrents to flow. In order to allow the junction to also handle normal currents, the basic model has to be extended to a generalized Josephson junction.

2.2.2 GENERALIZED JOSEPHSON JUNCTION

The generalized Josephson junction model provides a complete description of the characteristics of this nonlinear device.

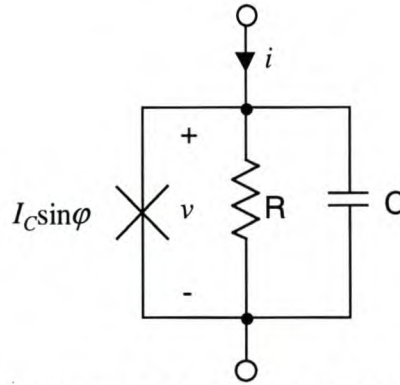


Figure 2.2: Circuit model for generalized Josephson junction

Fig. 2.2 shows the resistively shunted junction (RSJ) model for the generalized Josephson junction. The complete model allows for three parallel conduction channels. The first channel is a basic lumped Josephson junction to handle the supercurrent $i = I_C \sin \phi$. The second channel is purely resistive, and handles the normal current. The nonlinear conductance of this channel is modeled by a constant resistance R in the RSJ model. The third channel represents the capacitance of the device, and allows a displacement current to flow.

If we consider the case in which a current source is connected across the generalized Josephson junction, the total current can be found from Kirchhoff's current law as

$$i = I_C \sin \phi + C \frac{dv}{dt} + \frac{v}{R} \quad (2.4)$$

The voltage across the generalized junction equals the voltage across the basic lumped junction, and can be found by rewriting (2.3) in terms of v as

$$v = \frac{\Phi_0}{2\pi} \frac{\partial \phi(t)}{\partial t} \quad (2.5)$$

We can now substitute equation (2.5) into (2.4) to obtain

$$i = I_C \sin \phi + \frac{1}{R} \frac{\Phi_0}{2\pi} \frac{d\phi}{dt} + C \frac{\Phi_0}{2\pi} \frac{d^2 \phi}{dt^2} \quad (2.6)$$

The current equation (2.6) can be written as a dimensionless equation

$$\frac{i}{I_C} = \sin \phi + \frac{d\phi}{d\tau'} + \beta_C \frac{d^2 \phi}{d(\tau')^2} \quad (2.7)$$

with

$$\tau_J = \frac{\Phi_0}{2 \pi I_C R} \quad (2.8)$$

and

$$\tau_{RC} = RC, \quad (2.9)$$

representing the time constants for the basic Josephson junction and the RC combination respectively.

Also,

$$\beta_C = \frac{\tau_{RC}}{\tau_J} \quad (2.10)$$

and

$$\tau' = \frac{t}{\tau_J} \quad (2.11)$$

The parameter β_C is dimensionless, and is called the Stewart-McCumber parameter. The Stewart-McCumber parameter is a measure of the influence of the capacitance, and should be chosen to be close to 1 in RSFQ circuits [16].

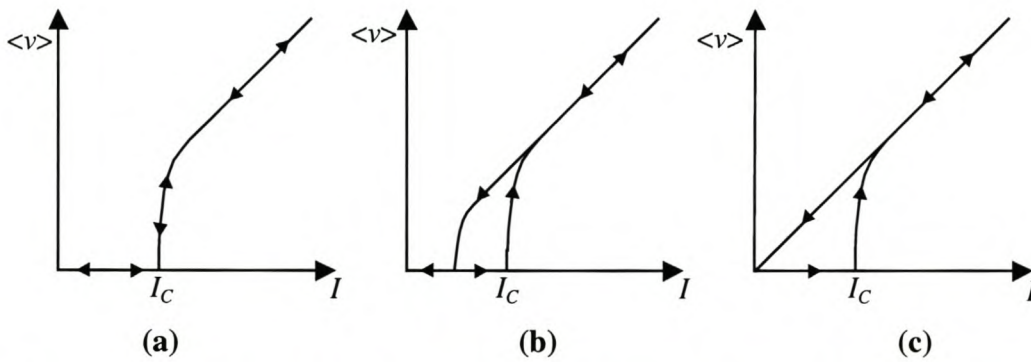


Figure 2.3: Voltage-current relationship of generalized Josephson junction for (a) $\beta_C \ll 1$, (b) $\beta_C \approx 1$ and (c) $\beta_C \gg 1$

The voltage-current relationship of the generalized Josephson junction is shown in Fig. 2.3 for the cases $\beta_C \ll 1$, $\beta_C \approx 1$ and $\beta_C \gg 1$. In Fig. 2.3, I is the dc current through the junction and $\langle v \rangle$ is the average voltage across the junction.

2.3 One-junction SQUID

A one-junction SQUID is formed when a Josephson junction is placed in a superconducting loop. The loop can be modeled by an inductor (see Fig. 2.4.)

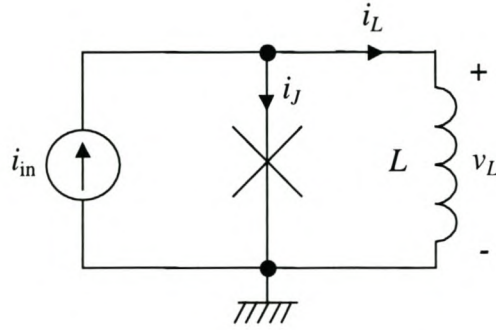


Figure 2.4: Circuit diagram of one-junction SQUID

Referring to Fig. 2.4, we can write i_{in} from Kirchoff's current law as

$$i_{in} = i_J + i_L \tag{2.12}$$

Substitution of i_J in (2.12) with the current-phase relation (2.1) gives

$$i_{in} = I_C \sin\varphi(t) + i_L \tag{2.13}$$

The voltage-current relation of an inductor is given by

$$v = L \frac{di_L}{dt} \tag{2.14}$$

Equating (2.14) to the junction voltage (2.5), and substituting for $\varphi(t)$ in (2.13) leads to

$$i_{in} = I_C \sin\left(\frac{2 \pi L i_L}{\Phi_0}\right) + i_L \tag{2.15}$$

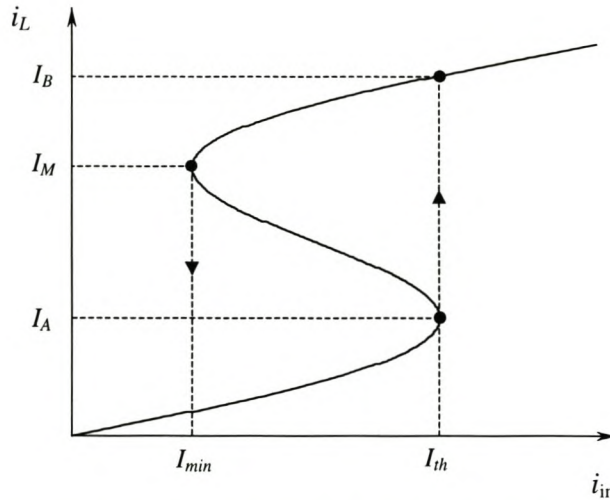


Figure 2.5: Inductor current against input current for one-junction SQUID

We can now define an important new parameter

$$\beta_L = \frac{2 \pi L i_L}{\Phi_0} \tag{2.16}$$

which is often used in connection with SQUID devices [20, p. 259].

If β_L is larger than unity, the current relation (2.15) depicted in Fig. 2.5 displays hysteresis. The one-junction SQUID can therefore be used as a switching element. The current value I_{th} at which the one-junction SQUID switches to the next quantum state can be found by setting

$$\frac{di_{in}}{di_L} = 0 \quad (2.17)$$

Also, from (2.15)

$$\frac{di_{in}}{di_L} = I_C \cos\left(\frac{2\pi L i_L}{\Phi_0}\right) \cdot \frac{2\pi L}{\Phi_0} + 1 \quad (2.18)$$

This yields

$$I_{th} = I_C \sin\left[\cos^{-1}\left(\frac{-1}{\beta_L}\right)\right] + \frac{I_C}{\beta_L} \cos^{-1}\left(\frac{-1}{\beta_L}\right) \quad (2.19)$$

The value I_{min} at which the one-junction SQUID switches back to the original quantum state can be determined by first rewriting (2.15) as [20, p. 259]

$$i_{in} = I_C \sin\left(2n\pi - \frac{2\pi\Phi}{\Phi_0}\right) + i_L \quad (2.20)$$

where Φ is the magnetic flux in the loop.

Since the relation (2.20) contains a sinusoidal term with a constant coefficient, it is fair to assume that I_{min} lies as far below $\Phi_0/2L$ as I_{th} lies above it. This gives

$$I_{min} = \frac{\Phi_0}{L} - I_{th} \quad (2.21)$$

2.4 Two-junction SQUID

A two-junction SQUID, also called a dc SQUID, is formed when two Josephson junctions are connected in a superconducting loop as shown in Fig. 2.6. The path of integration (C) is shown by the dotted line.

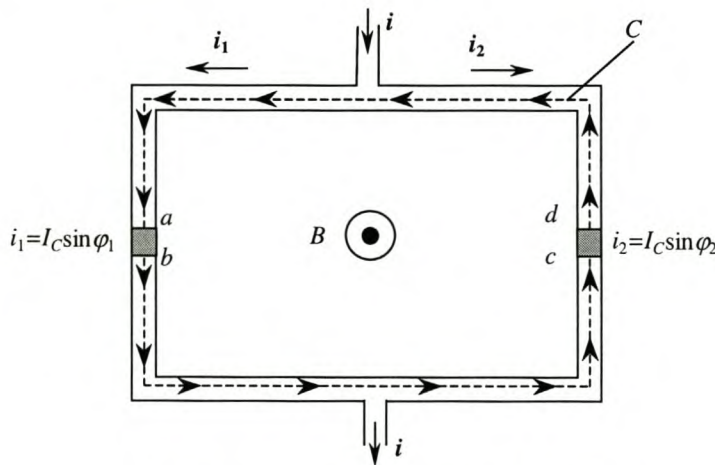


Figure 2.6: Two-junction SQUID

If the junctions have equal values for I_C , the total current entering the loop is

$$i = i_1 + i_2 = I_C \sin\varphi_1 + I_C \sin\varphi_2 \quad (2.22)$$

$$= 2I_C \cos\left(\frac{\varphi_1 - \varphi_2}{2}\right) \sin\left(\frac{\varphi_1 + \varphi_2}{2}\right) \quad (2.23)$$

Through a rather complex mathematical derivation involving the contour integration of supercurrent densities and magnetic vector potentials (substituted for the gradient of phase) [3, pp. 411-413], (2.23) can be written as

$$i = 2I_C \cos\left(\frac{\pi\Phi}{\Phi_0}\right) \sin\left(\varphi_l + \frac{\pi\Phi}{\Phi_0}\right) \quad (2.24)$$

If we now let the loop inductance be L , the total magnetic flux through the loop equals the sum of the externally applied flux Φ_{ext} , and the flux generated by the currents flowing in the loop. The branch currents can also be written as

$$i_1 = \bar{I} + I_{\text{cir}} \quad (2.25)$$

and

$$i_2 = \bar{I} - I_{\text{cir}} \quad (2.26)$$

where

$$\bar{I} = \frac{(i_1 + i_2)}{2} \quad (2.27)$$

is the average current common to both branches. Since \bar{I} represents an equal current flowing through each branch, it generates no flux through the loop. From (2.25), (2.26) and (2.27) it now follows that

$$I_{\text{cir}} = \frac{(i_1 - i_2)}{2} \quad (2.28)$$

The current I_{cir} is the current circulating in the loop. The flux generated through the loop by this flow of current is LI_{cir} . We can now write the total flux as

$$\Phi = \Phi_{\text{ext}} + LI_{\text{cir}} \quad (2.29)$$

which can be shown mathematically to be equivalent to [3]

$$\Phi = \Phi_{\text{ext}} - LI_C \sin\left(\frac{\pi\Phi}{\Phi_0}\right) \cos\left(\varphi_l + \frac{\pi\Phi}{\Phi_0}\right) \quad (2.30)$$

If the loop inductance L is nearly zero, so that $LI_C \ll \Phi_{\text{ext}}$, no flux screening takes place, and the flux in the loop is nearly equal to the external flux.

The derivative of the current in (2.24) with respect to φ_l equals zero where the current has a maximum value. In this special case, the maximum current through the dc SQUID is [3]

$$i_{\text{max}} \approx 2I_C \left| \cos\left(\frac{\pi\Phi_{\text{ext}}}{\Phi_0}\right) \right| \quad (2.31)$$

This current is periodic with the external flux, and can be used to measure the applied flux. In this way, two-junction SQUIDs are used as magnetometers with sensitivities in the range of 1 fT (10^{-15} T) [3], [20], [21].

In COSL circuits, an external flux is applied to the dc SQUID by coupling a control current into the loop through a mutual inductance. A detailed discussion is done by Rabie [19], but is omitted from this project because the COSL circuits are used unaltered.

If $LI_C \gg \Phi_{\text{ext}}$, the circulating current will try to cancel the applied flux. The total flux in the loop now tends to be quantized, and is given by [3]

$$\Phi = \Phi_{\text{ext}} + LI_{\text{cir}} \approx n\Phi_0 \quad (2.32)$$

where n is the integer closest to Φ_{ext}/Φ_0 .

In RSFQ circuits, the dc SQUID is used as a memory cell. With no externally applied flux, (2.32) yields

$$LI_{cir} \approx n\Phi_0 \tag{2.33}$$

For RSFQ memory cells, $LI_C \approx \Phi_0$ [16], [21], and the circulating current I_{cir} can have one of only two stable values, so that $I_{cir} \approx \pm \Phi_0/2L$. It can now be seen that the flux trapped in a two-junction SQUID in an RSFQ memory cell can have two values, or stable states, and that they differ by one fluxon.

2.5 The Josephson Transmission Line and RSFQ Logic

The JTL as described by Likharev and Semenov [16] is the most basic cell in any RSFQ system, and as such is the fundamental building block used to construct all the logic gates and memory cells in RSFQ.

Data is stored in RSFQ cells as magnetic flux quanta (fluxons,) and passed on as small voltage pulses a few picoseconds wide. In simulations for circuits with the parameters for the 1 kA/cm², 3 micron all-niobium process from Hypres [22], these voltage pulses have amplitudes of around 300 μ V to 450 μ V. The width of such a voltage pulse ranges between 5 ps and 10 ps, and depends on the damping of the Josephson junction across which it is generated.

When an input pulse (current or voltage) is applied to a line of cascaded JTLs, as shown in Fig. 2.7 [23], it propagates along the line by switching the Josephson junction in each JTL in turn (Fig. 2.8.)

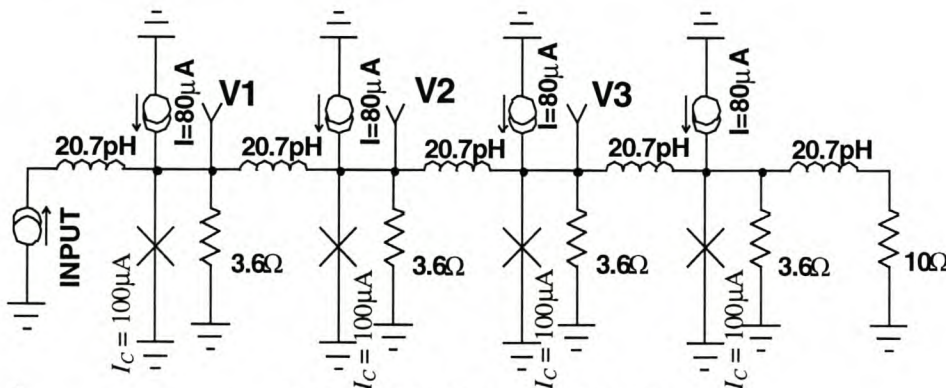


Figure 2.7: Cascade of JTLs

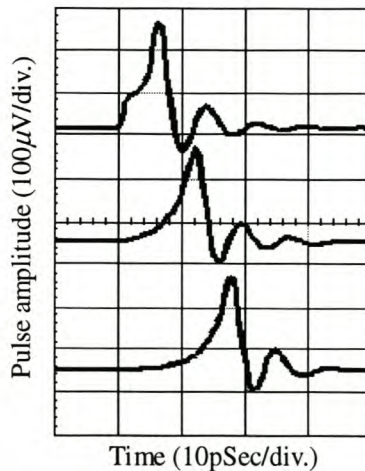


Figure 2.8: Simulated pulse propagation along a cascade of JTLs

The area under the voltage-time trace when an RSFQ cell produces a voltage pulse, integrates to 2.07×10^{-15} Wb (2.07 fWb.) This is the size of the fundamental magnetic flux quantum, or fluxon. Hence the term single flux quantum (SFQ) pulse.

The JTL is a bidirectional device, in the sense that a voltage pulse applied at the output will propagate to the input. One way to block reverse pulses from propagating through the system, is to change the design of the JTL in order to make it a one-directional device [16], [23]. With careful design, the need to insert reverse buffering between logic gates is avoided.

All other RSFQ cells work on the same principle as the JTL, with one-junction SQUIDs switching to generate or pass on voltage pulses. RSFQ is a pulse logic family, and these pulses represent digital data. A pulse propagating through a node in an RSFQ circuit at any time during a clock cycle represents a digital “1.” If no pulses pass through a node during a clock cycle, a digital “0” is implied.

A logic family would be incomplete if there is no way to store data between calculations. In RSFQ, data is stored by “flux trapping.” If the loop inductance between two consecutive Josephson junctions is large enough, and the second junction is not biased with a dc current, a pulse generated by the switching action of the first JJ will fail to trigger the second junction. This effectively traps a fluxon in the loop, and a digital bit is stored. The loop will be reset if another pulse arrives at the second junction via a different path (usually called the reset path.) The reset action also generates an output pulse at the second junction.

The principle of flux trapping is used in several RSFQ gate structures, and always in RSFQ registers. The smallest circuit to use this effect is the Destructive Readout register (DRO,) of which an illustrated discussion can be found in Sec. 3.2.2.1.

The connections between RSFQ components are inductive, and are represented on-chip by superconducting microstrip lines. The inductance per unit length decreases as the line width is increased, so that connections become more bulky when component distances are increased. In order to keep line widths down, the distance between components needs to be kept to a minimum. When SFQ pulses have to be transmitted over long distances (more than about $100 \mu\text{m}$) the transmission line is constructed of a cascade of JTLs.

An alternative proposed by Likharev [16], is to use transmission lines with a specific characteristic impedance to connect JTLs over large distances. The characteristic impedance should then be approximately equal to the damping resistance of the output junction.

2.6 COSL

Since the operation of COSL is well documented [13], [14], [15], [19], this discussion will only be conceptual.

The basic COSL gate is shown in Fig. 2.9. The principle element is the two-junction SQUID. The one-junction SQUID only serves to supply a control current.

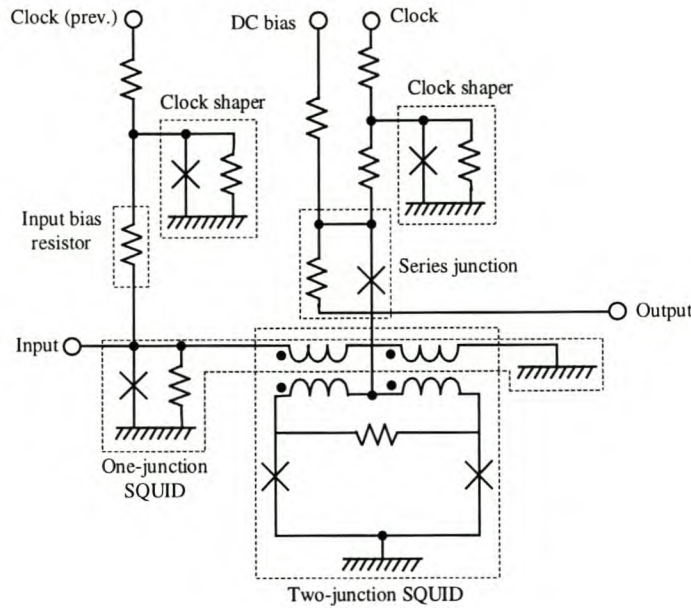


Figure 2.9: Schematic of basic COSL gate

COSL gates are clocked according to a three-phase clocking scheme. The clock signals have an amplitude of 10 mV, and all phases are separated by 120° , or a third of a clock cycle, from one another.

Each gate has two clock inputs. These inputs are connected to two clock phases so that the clock driving the one-junction SQUID leads the clock driving the series junction and the two-junction SQUID by 120° .

Since the input signal has to be in phase with the input clock, and the output signal is generated in phase with the output clock, the latency of the COSL gate is one third of a clock period.

The input of each COSL gate is clocked by the same phase as the output of the previous gate. The only exception is the COSL XOR-gate, for which the input clock lags the previous output by 120° [13].

The clock shapers restrict the clock amplitude to 2.5 mV in the gate. The input bias resistor limits the input clock current, and determines the logic function of the gate. This is demonstrated by comparing the COSL OR-gate and the COSL AND-gate. All COSL gates are driven by $100 \mu\text{A}$ inputs. The COSL OR-gate must therefore switch for an input current of $100 \mu\text{A}$ (see Appendix E for logic definitions.) This requirement is satisfied for a bias resistor is 8.8Ω . With a bias resistor of 14.3Ω , the input clock current is reduced, and the gate will only switch for an input current of $200 \mu\text{A}$. Since this corresponds to 2 logic high inputs, the logic AND function is implemented.

More detailed circuit diagrams for the COSL OR and NOR-gates, as well as simulation results, are presented in paragraph 3.3.1. When these results are viewed, it is important to understand that COSL is a voltage state logic family. This means that a 1 mV signal represents a digital “1,” and a 0 mV signal represents a digital “0.”

Chapter 3 – Building Blocks

3.1 Introduction

The building blocks for RSFQ digital logic circuits consist of some standard logic gates for combinational logic design, registers and latches for sequential logic design, and asynchronous components that merge, split or merely delay SFQ pulses. Included in this collection are converters used for interfacing off-chip room temperature equipment with the cryogenic RSFQ system. The entire collection of RSFQ cells used in this project, is discussed in this chapter.

The analogue-to-digital converter described in Sec. 4.5 requires the use of some devices that are not part of the standard RSFQ collection. A level detector and a few COSL gates are therefore included in the collection of building blocks discussed in this chapter.

Furthermore, simulation models and setups, as well as simulation results and yield analysis procedures are discussed.

The circuit diagrams shown in this chapter were created from the simulation models implemented in WRSpice [24]. It is important to note that the area parameter of a Josephson junction in WRSpice is specified in terms of the unit area size of $100 \mu\text{m}^2$. Since the 1 kA/cm^2 process is used, a unit area represents an I_C of $1000 \mu\text{A}$.

3.2 RSFQ logic

3.2.1 BASIC ASYNCHRONOUS COMPONENTS

3.2.1.1 JTL

The operation of the JTL has already been discussed in Sec. 2.5.

The JTL is used to connect complex cells while performing pulse sharpening [16], [25], and also provides a standard load for testing and optimizing logic circuits [18]. Since the JTL is bidirectional, it can be designed to be completely symmetrical. This ensures that cells connected to the input and output of the JTL will both see an equal load.

The JTL obtained from the Stony Brook University cell library is shown in Fig. 3.1. This JTL is completely symmetrical, with both input and output junctions having an I_C of $250 \mu\text{A}$. In this project, the current rating of a JTL refers to the I_C values of the Josephson junctions, and not to the dc bias current of the device.

The simulated response of a cascade of $250 \mu\text{A}$ JTLs to an SFQ input pulse is shown in Fig. 3.3. Although the pulses appear to be poorly damped, this is not the case, and merely a result of measuring voltage pulses in the centre of connecting inductors. The voltage pulses at either end of a connecting inductor are very well damped, and resemble the pulse across the output resistor in Fig. 3.3. The current pulses are perfectly damped, as can be seen in Fig. 3.4.

It is sometimes preferable to use JTLs with larger critical current values. This is especially useful when the input junctions to larger circuits have I_C values that are more than about 1.5 times larger than that of the driving JTL. The reason for this is

that the input junction of a circuit overloads the driving JTL if the critical current density of the input junction is too large. This not only results in the input junction not switching, but also distorts the output pulse of the driving JTL.

The overloading of junctions can be avoided by using a simple design rule, suggested by Likharev and Semenov [16], that states that I_C for any two interconnected junctions should ideally not differ by more than $\sqrt{2}$.

This design rule applies to the input and output junctions of RSFQ cells connected together when complex circuits are constructed from smaller logic gates or blocks. Within a logic gate or cell, I_C may differ by more than $\sqrt{2}$ from one junction to the next, since some functions require junctions not to switch when, for instance, only one SFQ pulse arrives.

Having discussed the afore-mentioned design rule, we can now design a JTL that can drive RSFQ gates with large I_C values for the input junctions. One such gate is the RSFQ T-flip-flop. The input junction in this register has an I_C of $431 \mu\text{A}$, which is outside the ideal maximum value of about $355 \mu\text{A}$ (roughly $\sqrt{2} \times 250 \mu\text{A}$) for connection to a standard $250 \mu\text{A}$ JTL.

A new JTL is designed to have a current rating of roughly $\sqrt{2} \times 250 \mu\text{A}$, or $355 \mu\text{A}$. The inductance values are obtained from analyzing the RSFQ pulse splitter (Fig. 3.7) found in [18], since it was observed that this device has one junction with $I_C = 355 \mu\text{A}$ driving two junctions, each with $I_C = 250 \mu\text{A}$.

The circuit diagram for the $355 \mu\text{A}$ JTL is shown in Fig. 3.2(b). The standard approach to matching unbalanced inputs and outputs (with regards to critical current) would be to place a $250 \mu\text{A}$ JTL in series with a $355 \mu\text{A}$ JTL between the input and output that have to be connected.

Although the $355 \mu\text{A}$ JTL was used often in system simulations and input-output matching for device optimization, it is more economic to have a single JTL that matches an input and output with different I_C values. Consequently, a $250 \mu\text{A}$ -in $355 \mu\text{A}$ -out JTL was developed. The circuit diagram is shown in Fig. 3.2(a).

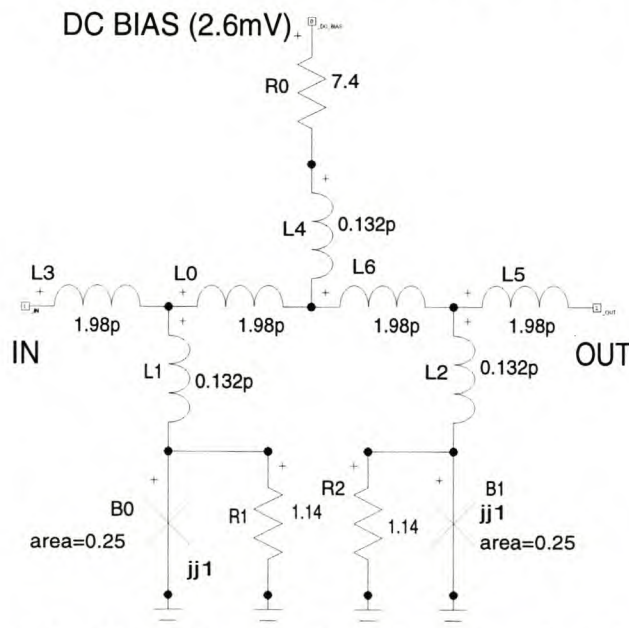


Figure 3.1: Circuit diagram for $250 \mu\text{A}$ Josephson junction

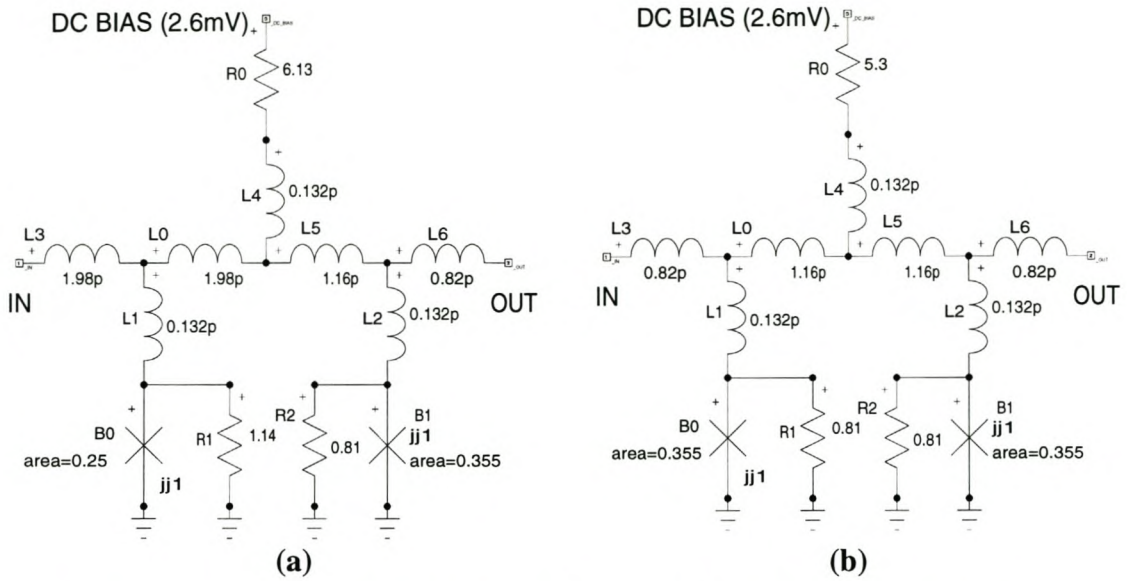


Figure 3.2: Circuit diagram for (a) 250 μA -in 355 μA -out and (b) 355 μA Josephson junction

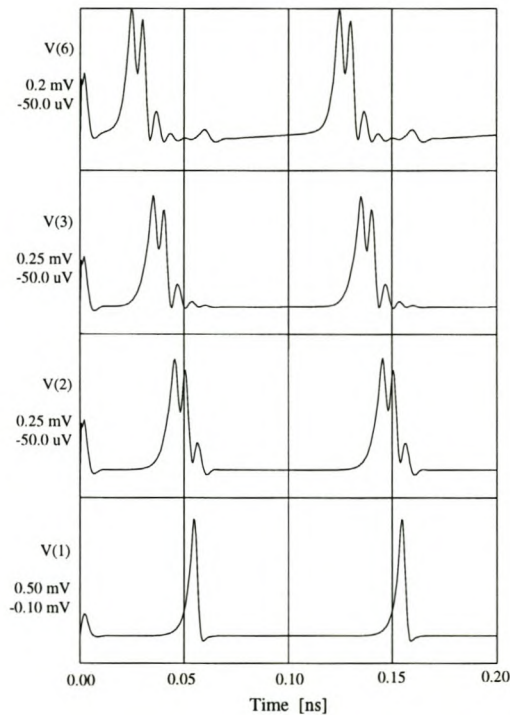
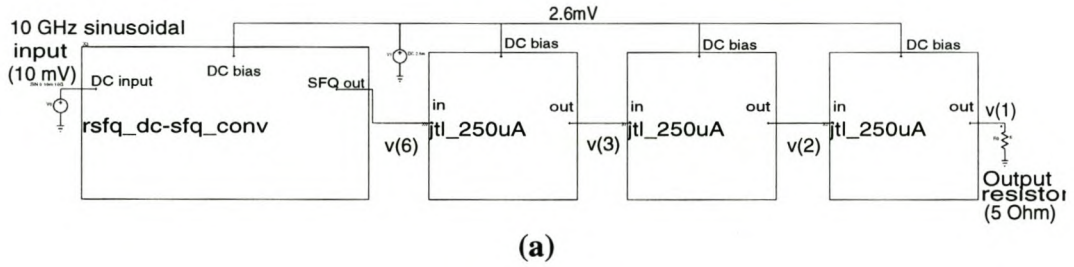


Figure 3.3: (a) Simulation test setup and (b) simulated response of a cascade of 250 μA JTLs

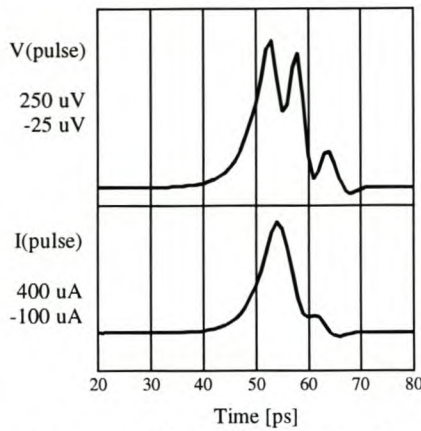


Figure 3.4: Simulated voltage and current pulses between two 250 μA JTLs

Finally, it is important to note that the 250 μA -in 355 μA -out JTL can also be used to match a large output junction to a small input junction by merely interchanging the input and output pins, since the device is bidirectional. Technically, it should then be called a 355 μA -in 250 μA -out JTL.

As has been discussed in Sec. 2.5, the inductive connections between RSFQ cells can be replaced by transmission lines matched at both ends to the damping resistance of the terminating junctions.

Since the average value for the damping resistances of the Stony Brook JTL circuit [18] is in the order of 1 Ohm (see Fig. 3.1,) these transmission lines have to be very wide. However, since the TX line is in parallel with the damping resistor of an output or input junction, the equivalent impedance seen by the junction will be the required value if the values of the damping resistor and characteristic impedance of the TX line are double that.

The circuit diagram in Fig. 3.5 shows two JTLs connected through a matched TX line. Since the TX line replaces the inductive connections, the inductors are stripped away. The damping resistance R_2 in both JTLs is increased to 2.28 Ω .

The damping resistances, in parallel with the matched TX line, provide an effective damping resistance of 1.14 Ω for junction B_1 in each JTL.

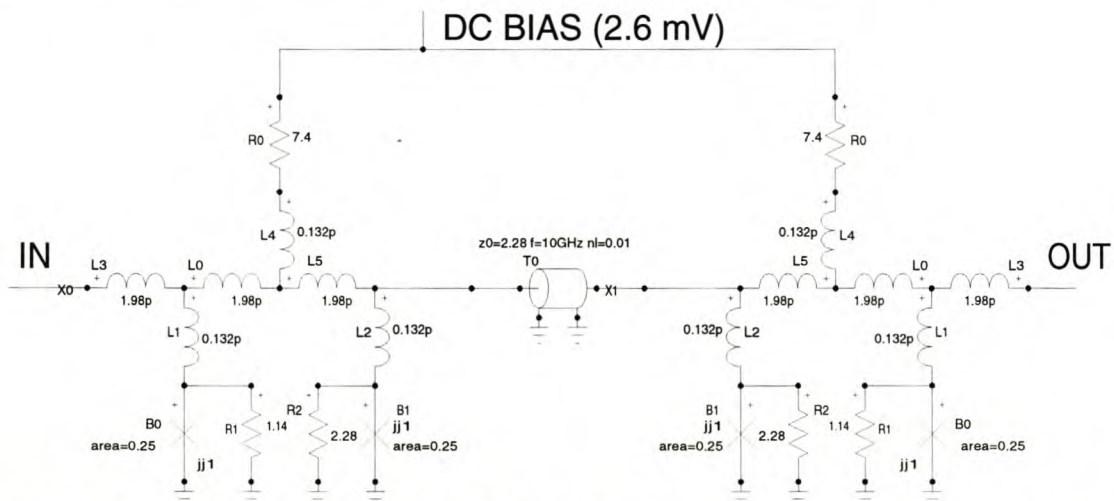


Figure 3.5: Circuit diagram for two 250 μA JTLs connected through a matched TX line

Fig. 3.6(a) shows the test setup for a cascade of JTLs connected through matched TX lines. The three TX lines, in order from left to right, have normalised lengths (to the wavelength at 10 GHz) of 0.01, 1.5 and 0.005 respectively. These values correspond to on-chip line lengths of $99 \mu\text{m}$, 14.9 mm and $49.5 \mu\text{m}$ respectively for layer M1 TX lines in the tri-layer niobium process from Hypres [22].

The short TX lines were used to test the validity of the TX line approach for standard inter-component distances, while the 14.9 mm line was chosen to test this approach for extremely long distances.

The distances obtainable with TX lines far outperform the maximum distance of about $100 \mu\text{m}$ obtainable by using inductive connections, although the minimum on-chip line width for such TX lines is about $19.5 \mu\text{m}$. Although this line width is larger than that of inductive connections, it dispenses with the necessity for rather bulky JTLs after every $100 \mu\text{m}$ of a connecting line. Furthermore, the transmission delay of 26.7 ps for a 4 mm TX line (the width of the active chip area) also beats that of a cascade of inductively connected JTLs over the same distance by more than 180 ps .

One limiting factor in the length of transmission lines is dispersion [26, pp. 451-454], the degrading effect that different phase velocities for each harmonic has on a pulse. Since the effects of dispersion in a thin-film superconductor can be quite large [20], this has to be factored into any design of long TX lines, especially when inter-chip lines are designed to carry SFQ pulses.

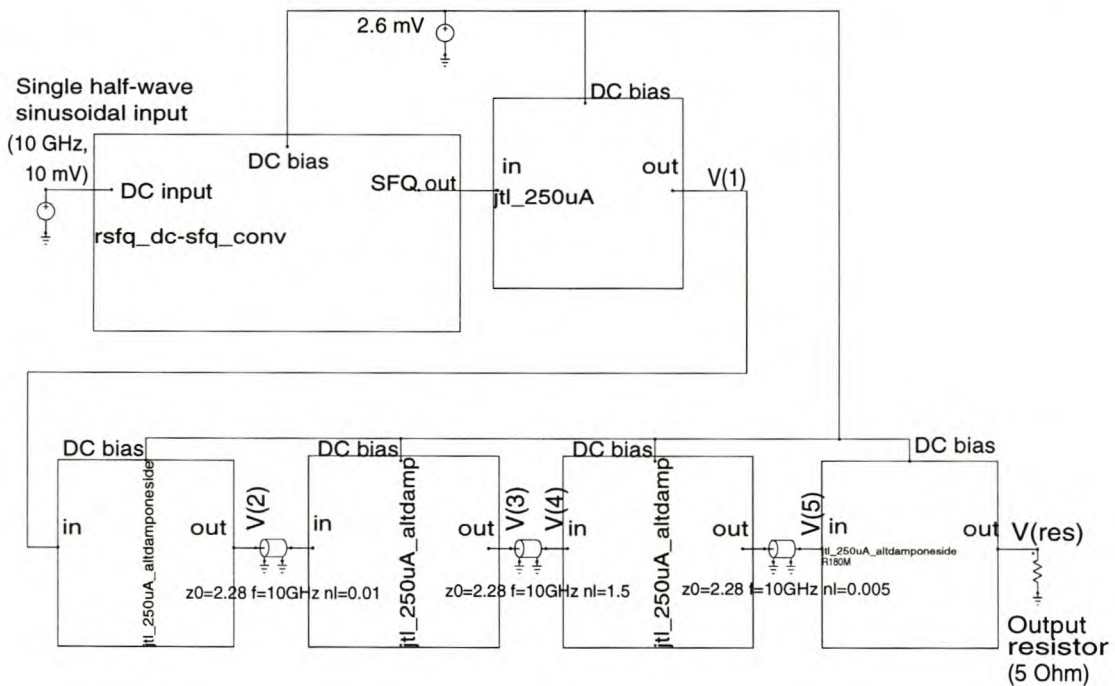


Figure 3.6: (a) Simulation test setup of a cascade of $250 \mu\text{A}$ JTLs connected through matched TX lines

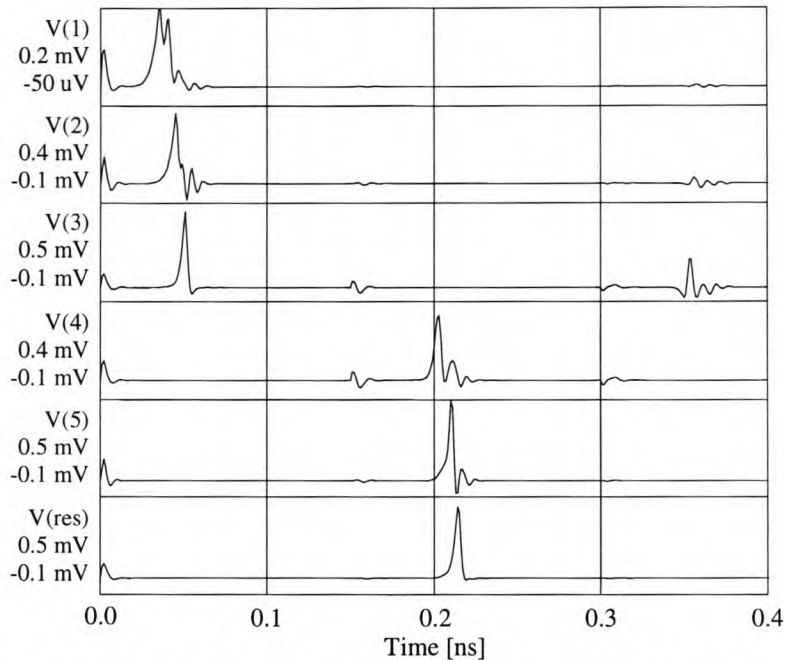


Figure 3.6: (b) Simulated response of a cascade of 250 μA JTLs connected through matched TX lines

The pulse transition time for a 14.9 mm length of TX line etched in the tri-layer niobium process from Hypres [22] is 150 ps. This is clearly visible in the simulation results presented in Fig. 3.6(b).

Also visible in Fig. 3.6(b) is the reflected pulse arriving back at V(3) 150 ps after the forward pulse through the TX line arrives at V(4). This is a feature of all transmission lines of which the connections are not exactly matched. Whenever line lengths are longer than about a tenth of a wavelength at the operating frequency, impedance matching becomes extremely important.

Matched TX line interconnections were not used in the layout of the ADC designed in this project, and consequently no Monte Carlo simulations were done on such structures. This technique deserves serious consideration in future projects though, and if Monte Carlo simulations predict theoretical yields comparable to that of inductively coupled devices, impedance matched TX lines can be used to connect any RSFQ cells spaced more than approximately 100 μm apart.

A final mention of impedance matched TX lines must be made for the case where the input and output junctions require different damping resistances. In this case the characteristic impedance of the TX line needs to have a different value at each end. This requires additional matching along the TX line.

The traditional approach to matching a TX line to different values of Z_0 at each end, is to use lumped element matching networks at each end, or to use stub matching in microstrip lines [27]. Although both techniques can deliver a wide bandwidth, the lumped element technique cannot be used on-chip, while stub lines occupy a lot of space.

A superior alternative (due to size and realizability) to both the above-mentioned techniques is the use of tapering in the microstrip line itself [27], [28]. Tapers have the advantage of being easy to implement in circuit layouts, and allows engineering trade-offs between reflection coefficient, bandwidth, taper length and complexity [29].

The SFQ pulses in RSFQ logic are very short in time, and are therefore wideband signals. In any of the above-mentioned techniques, a low reflection coefficient has to be traded with a wide bandwidth in order to keep down unwanted reflections at higher order harmonics.

In conclusion, cascaded JTLs are used to connect RSFQ cells over large distances. It is possible to connect JTLs over great distance by using impedance matched TX lines. These lines can even replace JTLs, and be used between other RSFQ cells, although engineering factors such as reflection coefficient, bandwidth and dispersion need to be taken into account.

3.2.1.2 Pulse splitter

A limiting factor in RSFQ design is the fact that RSFQ gates have a fan-out of one. If an output pulse is to be fed to multiple input ports, it has to be passed through a pulse multiplication unit first. Such a unit is available in RSFQ as a pulse splitter. The circuit for this splitter is shown in Fig. 3.7. The device creates two simultaneous output pulses with the application of a single input pulse.

The pulse splitter has an input junction (B_1) with $I_C = 355 \mu\text{A}$. This is $\sqrt{2}$ times larger than the value of I_C for a standard $250 \mu\text{A}$ JTL; the RSFQ component it is most often connected to.

The input junction then drives two output junctions (B_0 and B_2), both with $I_C = 251 \mu\text{A}$. All three JJs are biased by the single resistor R_2 . This resistor and inductors L_6 and L_8 are selected to insure that all three junctions are biased to 70 % of their respective values for I_C .

The I_C of B_1 is large enough so that, when B_1 is switched, the currents through both the output junctions will exceed their critical values. This induces the switching action in both output junctions, and a single SFQ input pulse is converted to two simultaneous output pulses.

Inductors L_0 to L_4 model the parasitic inductances inherent to the physical layout.

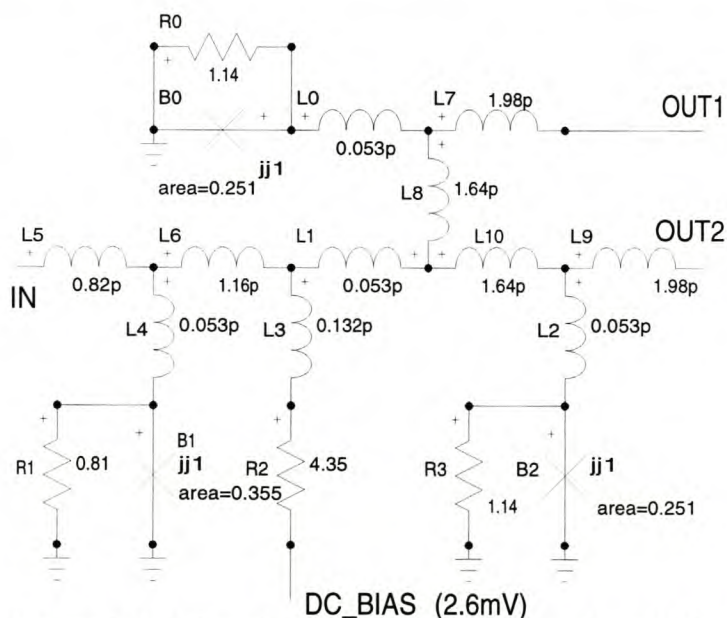
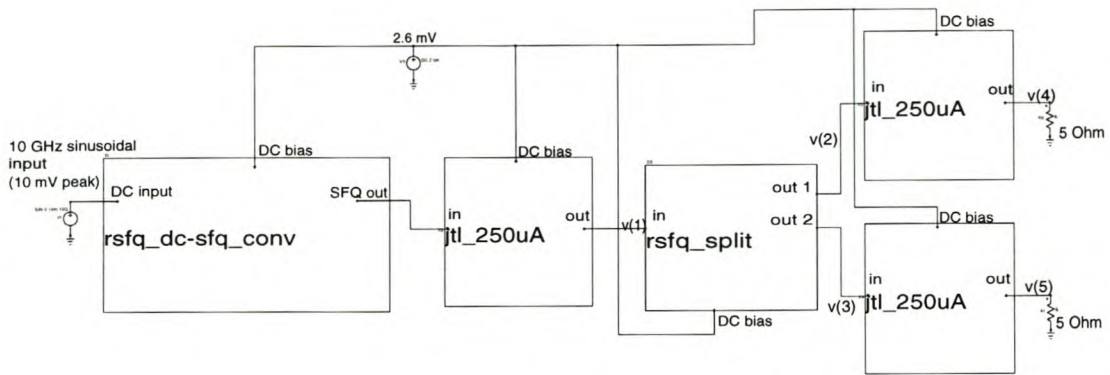
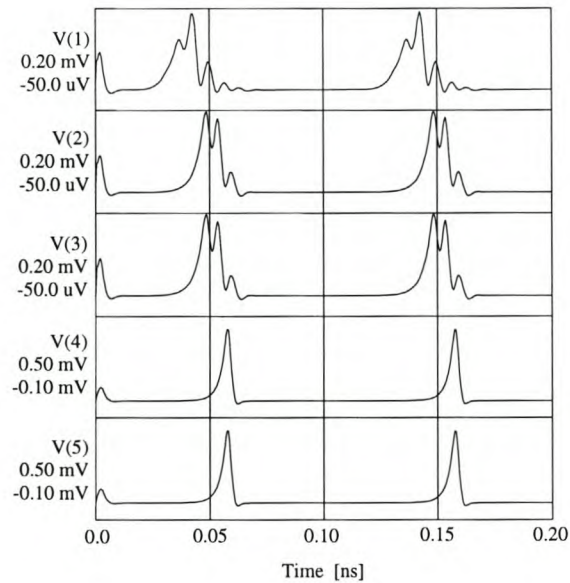


Figure 3.7: Circuit diagram for RSFQ pulse splitter



(a)



(b)

Figure 3.8: (a) Simulation test setup and (b) simulated response of RFSQ pulse splitter to SFQ input pulse

3.2.1.3 Pulse merger

The RFSQ pulse merger combines two input paths into a single output. An SFQ pulse is produced at the output if either one or both of the inputs are pulsed.

A discussion of the operation of the pulse merger also provides useful insight into the use of auxiliary junctions to provide buffering.

If an input SFQ pulse arrives at the biased junction B_1 (see Fig. 3.9,) the junction switches. The switching action leaves the phase over auxiliary junction B_0 virtually undisturbed, and the pulse propagates to the output junction B_4 , causing it to switch and produce an output pulse.

The switching action of B_1 also sends current through auxiliary junction B_2 . Since this junction has a slightly lower I_C than B_3 , B_2 switches first. This absorbs the pulse, and prevents the input pulse at B_1 from propagating to the second input at B_3 .

Due to symmetry, the behaviour of the pulse merger for an input pulse at junction B_3 is exactly the same as for an input pulse at B_1 .

When both inputs are pulsed simultaneously, only one output pulse appears at B_4 .

The pulse merger also has inherent reverse buffering capability. If an input pulse is applied to the output (at B₄), the pulse propagating in the reverse direction through the pulse merger will switch both auxiliary junctions B₀ and B₂. This absorbs the reverse pulse, leaving the inputs undisturbed.

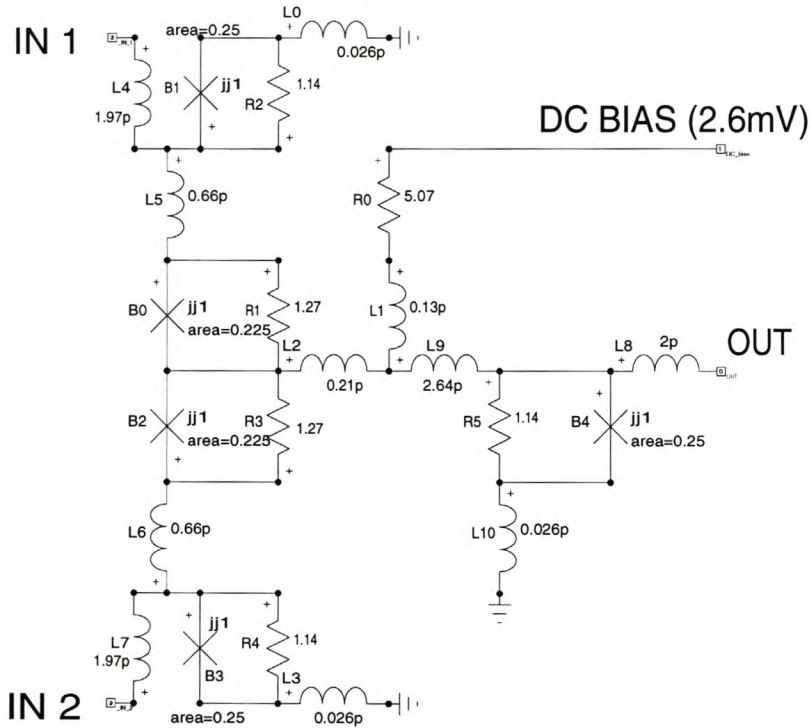


Figure 3.9: Circuit diagram for RSFQ pulse merger

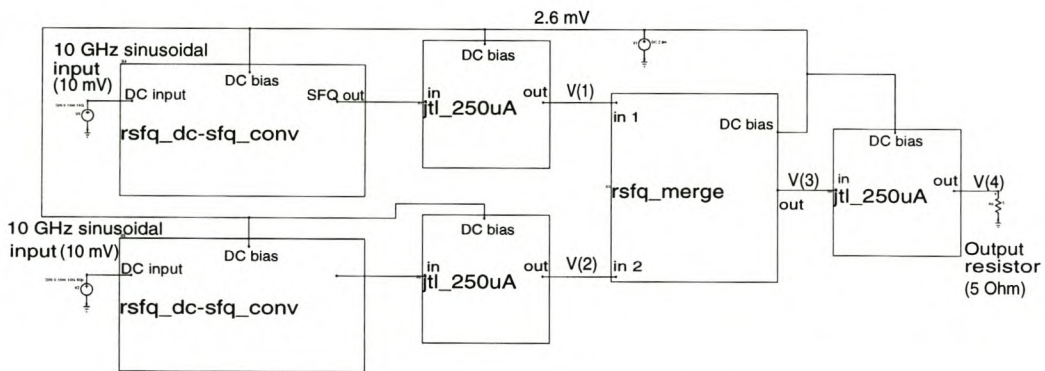


Figure 3.10: (a) Simulation test setup of RFSQ pulse merger

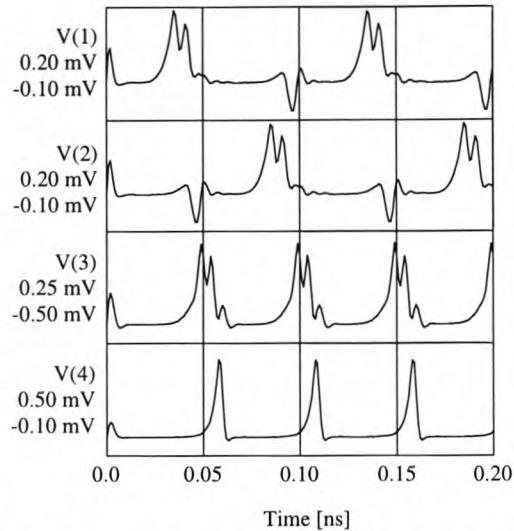


Figure 3.10: (b) Simulated response of RFSQ pulse merger to SFQ input pulses

3.2.2 REGISTERS AND LATCHES

3.2.2.1 Destructive Readout register (DRO)

The Destructive Readout register (DRO) is the simplest memory cell in RSFQ. One logic bit can be stored in the DRO by flux trapping. The circuit diagram for the DRO is shown in Fig. 3.11.

The DRO is built around a two-junction SQUID (B_0, L_0, B_1 .) There is only one bias current, and during start-up or power-on this current is forced by the size of inductor L_0 to flow through junction B_0 .

In the initial, or unset condition, only junction B_0 is biased through resistor R_2 . If a reset pulse is applied when the DRO is in the initial condition, the auxiliary junction B_3 switches, since B_1 has no current bias. The result is that the reset pulse falls out over B_3 , and no output pulse appears at F.

When a set pulse is applied, the biased junction B_0 switches before auxiliary junction B_2 can do so. The current in the two-junction SQUID now reverses direction, and flows through B_1 . Since B_1 has no other bias current, it has insufficient current to switch. The DRO is now in a second stable state, with all the bias current flowing through R_2, L_0 and B_1 . This is called the “set” state, and a digital bit is stored through the trapping of a magnetic fluxon in the two-junction SQUID.

If another set pulse arrives while the DRO is already set, it finds junction B_0 without any bias current, and switches auxiliary junction B_2 instead.

The stored bit can be read out by applying a pulse at the reset input. The reset pulse adds to the bias current through B_1 , causing it to switch. This results in an output pulse being passed on to the output F, and also resets the bias current to flow through B_0 .

The value of inductor L_2 was lowered to 0.58 pH from the value of 1.58 pH used in the Stony Brook cell library [18], after Monte Carlo simulations on the RSFQ AND-gate delivered better yield results with a lower inductance value.

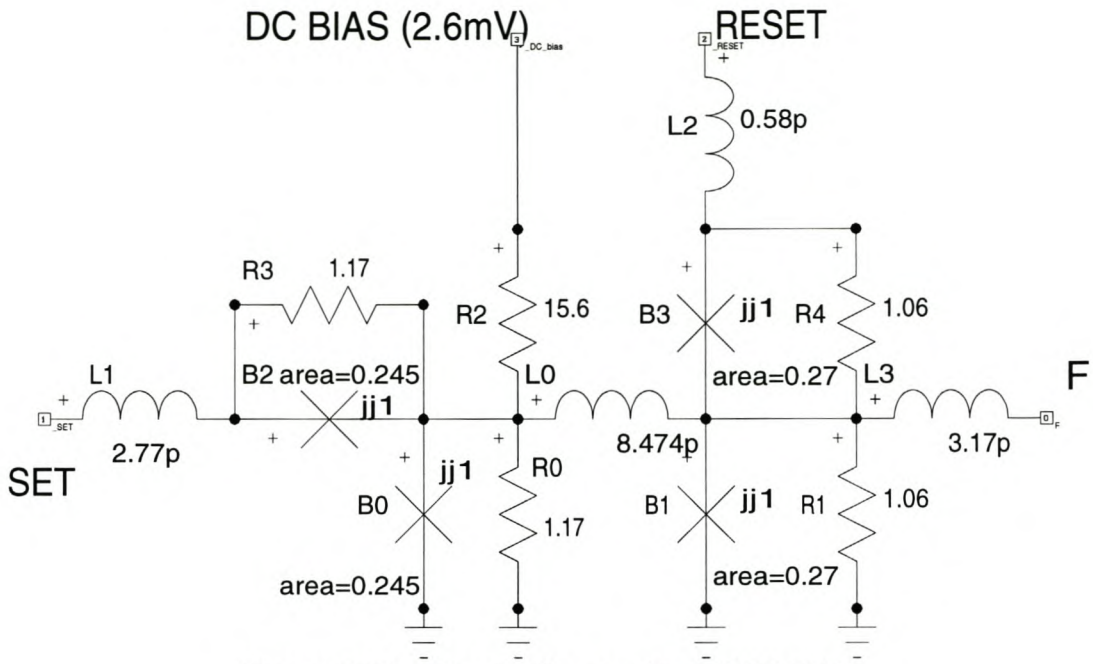
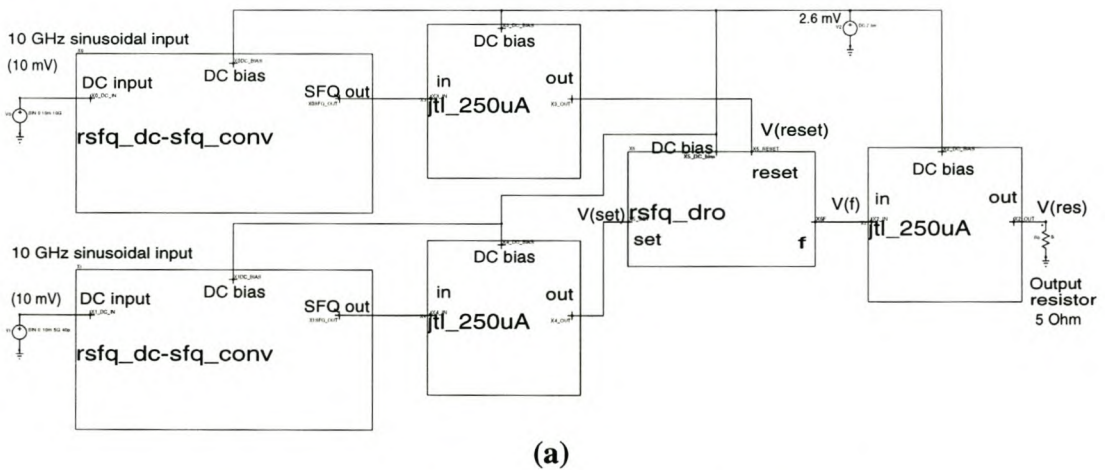
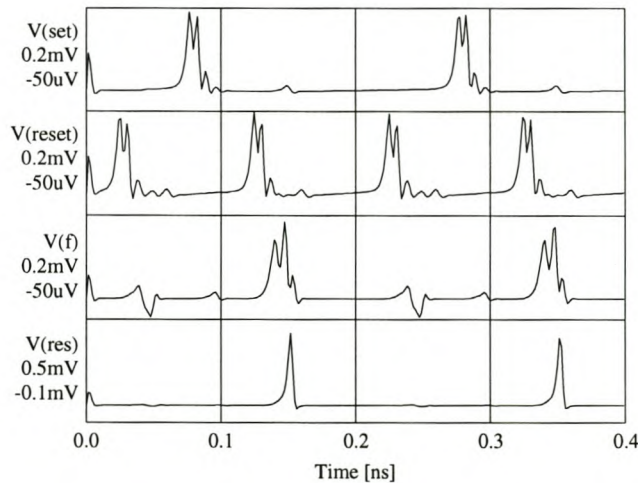


Figure 3.11: Circuit diagram for RFSQ DRO



(a)



(b)

Figure 3.12: (a) Simulation test setup and (b) simulated response of RFSQ DRO-register to SFQ set and reset pulses

3.2.2.2 T-flip-flop

The T-flip-flop was also obtained from the Stony Brook cell library [18]. The bipolar dc supply voltages supposedly increase the parameter margins.

The device operates like a divide-by-two circuit. For every second input pulse, one output pulse is generated.

The circuit diagram is shown in Fig. 3.13. As with all the other RSFQ registers, the T-flip-flop is built around a two-junction SQUID. This two-junction SQUID is formed by L_{12} , B_4 , L_4 , L_3 , L_{10} , B_2 and L_9 . In the initial condition the bias current flows through junction B_4 .

When a toggle pulse is applied at IN, it switches B_1 , B_4 and the auxiliary junction B_0 . The switching of B_4 reverses the current in the two-junction SQUID, and a fluxon is trapped in the loop.

A second toggle pulse applied at IN switches B_1 , finds B_2 biased, and consequently switches B_2 and auxiliary junction B_3 . When B_2 switches, the fluxon in the loop is released, the bias current is restored to flow through B_4 , and an output pulse is produced at OUT.

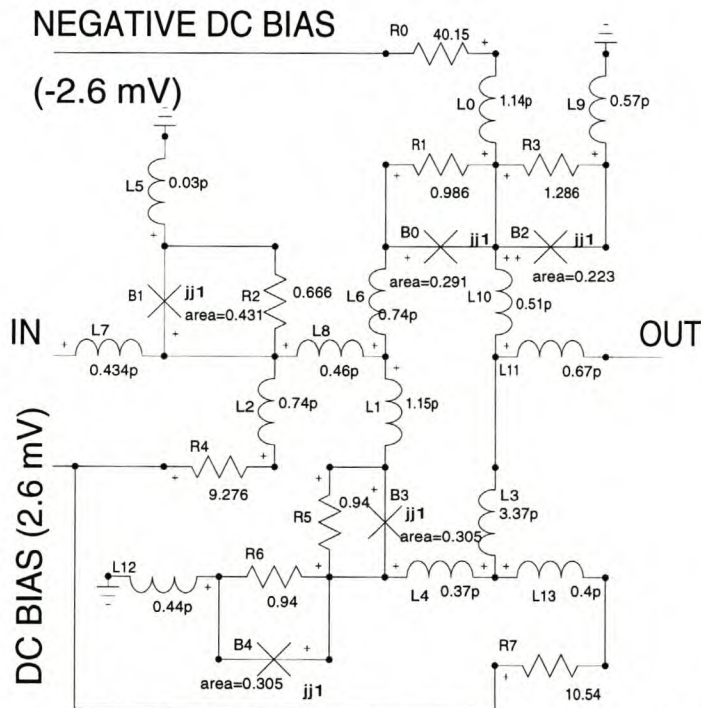


Figure 3.13: Circuit diagram for RSFQ T-flip-flop

The T-flip-flop is most often used as a clock division circuit. In the ADC discussed in Chapter 4, cascades of T-flip-flops divide the primary clock signal by a factor of 8 or 16. Since the three-register divide-by-eight cell is used more often, the simulations in Fig. 3.14 was done for this configuration.

Although the T-flip-flop functions well when connected to standard JTL input and output loads, Monte Carlo simulations predicted poor theoretical yield figures. The problem was traced to the input junction B_1 , which is too large to be driven by a $250 \mu\text{A}$ JTL. The failure rate can be reduced significantly by using $355 \mu\text{A}$ JTLs to drive the input junctions. Similarly, $355 \mu\text{A}$ JTLs used to load the outputs also increase circuit stability.

As a result of the high input and output current requirements of the RSFQ T-flip-flop, the test setup in Fig. 3.14(a) shows a cascade of three T-flip-flops connected through 355 μA JTLs. The first T-flip-flop is driven by a 250 μA -in 355 μA -out JTL that matches the RSFQ DC-to-SFQ converter to the T-flip-flop. The last T-flip-flop is followed by a 355 μA -in 250 μA -out JTL (or a 250 μA -in 355 μA -out JTL flipped around the vertical axis to interchange IN and OUT.) This last JTL matches the last T-flip-flop to the standard JTL dynamic load.

When the RSFQ T-flip-flop is used, either as a single register or in a cascade of flip-flops, the JTL matching rules discussed above and showed in Fig. 3.14(a) has to be observed in order to guarantee a high theoretical circuit yield.

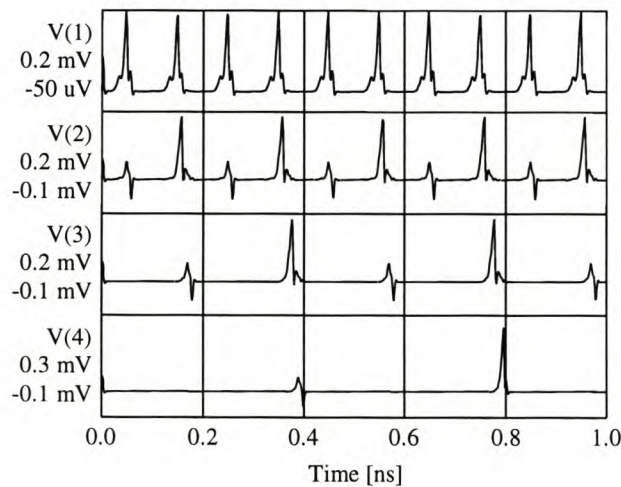
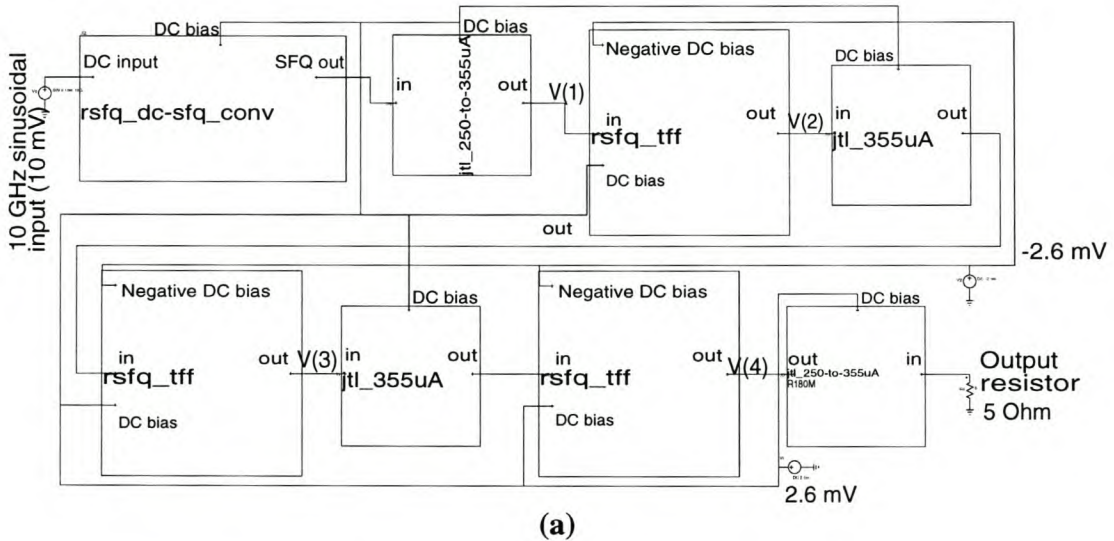


Figure 3.14: (a) Simulation test setup and (b) simulated response of a cascade of three RFSQ T-flip-flops to SFQ toggle pulses

3.2.2.3 *T1-flip-flop*

The T1-flip-flop, shown in Fig. 3.15, was obtained from the Stony Brook cell library [18]. Damping resistances had to be calculated.

The T1-flip-flop has an asynchronous output at C, and a destructive read input (DR) with a readout at S.

The register is built around a two-junction loop comprised of (including parasitics) L_1 , B_1 , L_8 , L_9 , B_6 , L_{10} , B_4 and L_0 . The initial state has current flowing through L_8 , B_1 and L_1 .

If a toggle pulse is applied at A, it switches junction B_1 and B_3 , to provide an output pulse at C. Auxiliary junction B_0 also switches. The switching action of B_1 reverses the current through the two-junction SQUID, and a fluxon is stored.

With the T1-flip-flop in the set state, another input pulse at A switches junction B_4 to reset the current in the loop. Auxiliary junctions B_2 and B_5 also switch, and prevent the input pulse from reaching the output stages of the circuit.

If the circuit is in the unset condition, a destructive read pulse at DR switches junction B_7 , and disappears. In the set state, a pulse at DR switches junction B_8 , and produces an output pulse at S. It also switches B_6 , releasing the fluxon in the loop, and resetting the T1-flip-flop to the initial state.

It is important to note that a $250 \mu\text{A}$ -in $355 \mu\text{A}$ -out JTL is used to match a toggle pulse to the input junction at A. This was done after Monte Carlo simulations predicted circuit failures with input A driven by a standard $250 \mu\text{A}$ JTL. The failures were traced to the input A, which with parameter deviations often demands more current to switch the state than a $250 \mu\text{A}$ JTL can supply.

Wherever this register is used in a circuit layout, the input A must be fed through a JTL with an output junction of $355 \mu\text{A}$ or larger, or a logic cell with a similar output current capability.

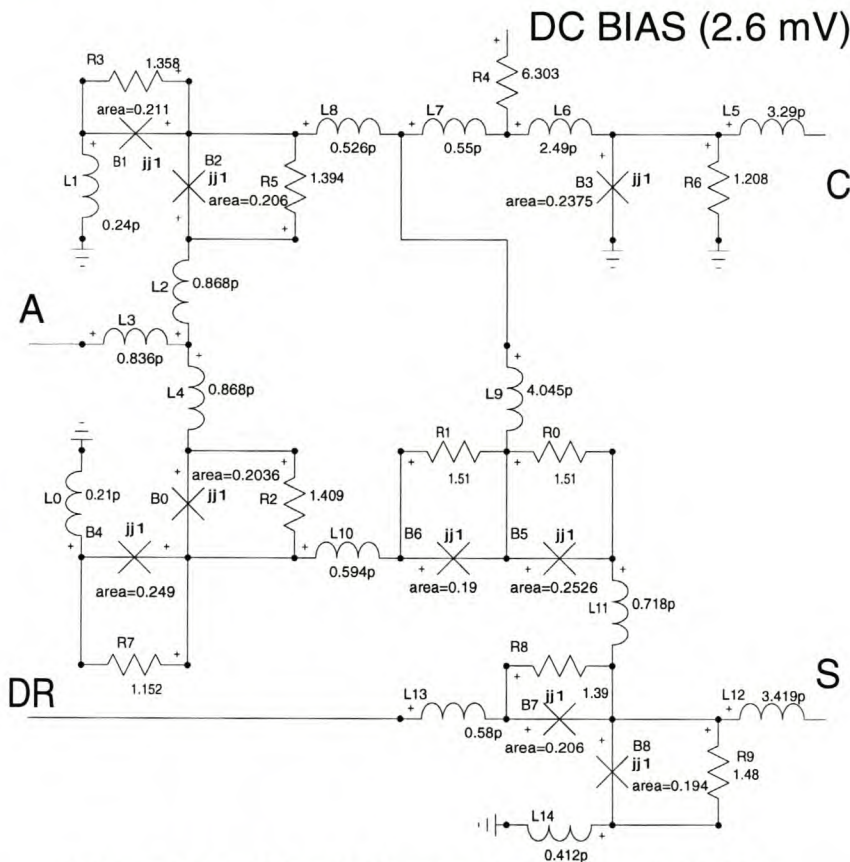


Figure 3.15: Circuit diagram for RSFQ T1-flip-flop

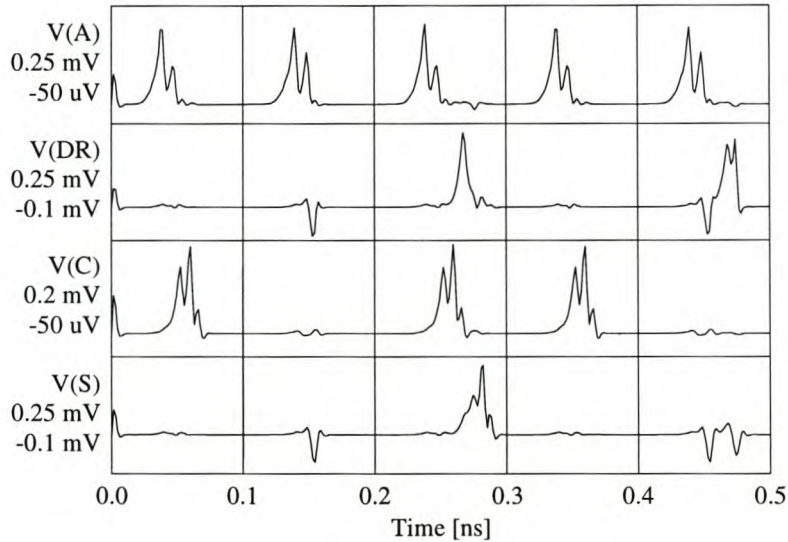
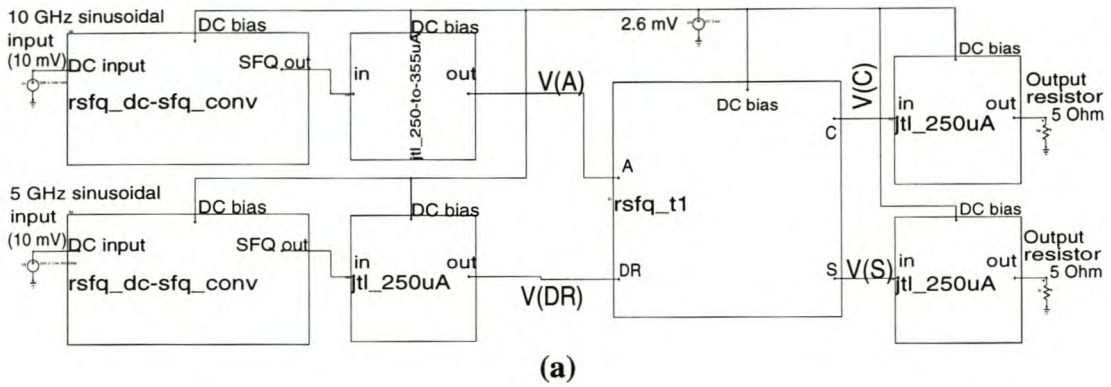


Figure 3.16: (a) Simulation test setup and (b) simulated response of RFSQ T1-flip-flop to SFQ toggle and destructive read pulses

3.2.2.4 DRO with 2 readouts

The DRO with two readouts was also obtained from the Stony Brook cell library [18], but is unique in the sense that bias currents are injected into the cell at all the outputs and inputs.

This register is basically a DRO with two symmetric reset inputs. The device is set by applying an input pulse to D (Fig. 3.17,) after which it can be reset by applying a pulse at either C_1 or C_2 . A reset pulse applied at C_1 while the device is set, results in an output at Q_1 . Due to symmetry, the same holds true for $C_2 - Q_2$. After the register has been reset, input pulses at C_1 or C_2 produce no outputs.

An initial Monte Carlo analysis on the Stony Brook cell predicted unacceptable yield results. In an unsuccessful attempt to increase the theoretical yield and reduce the complexity and power consumption of the circuit, the bias currents and junctions at the inputs and outputs were removed.

An acceptable circuit was eventually obtained by restoring the junctions and bias currents, and reducing L_{14} from 8.73 pH for the Stony Brook model to 4.23 pH. Inductor L_{15} was not part of the Stony Brook model, but was added to model parasitic inductance in the physical layout.

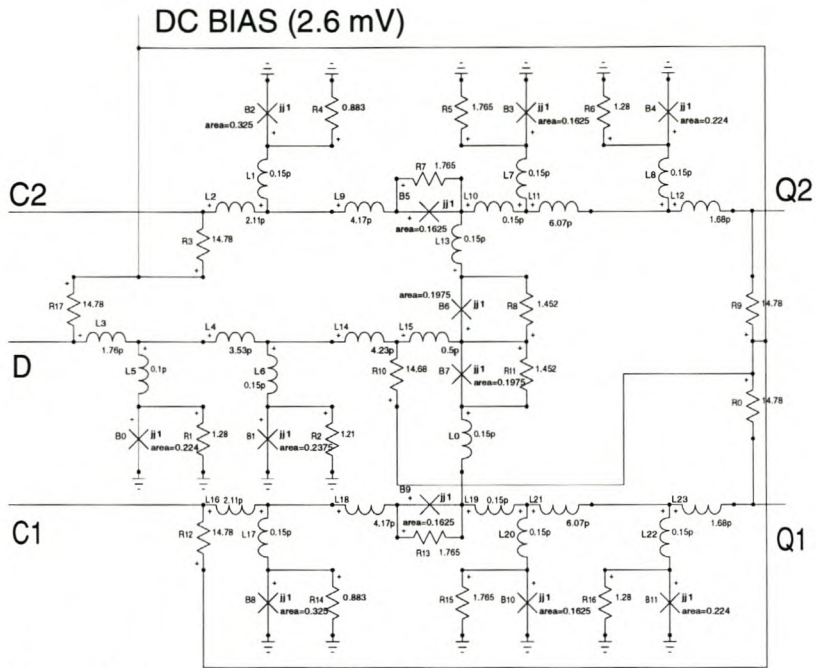
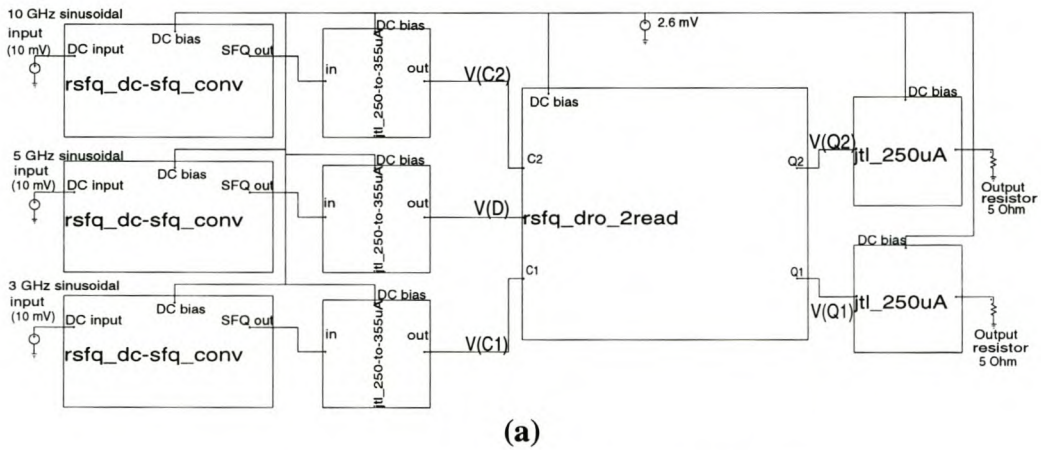
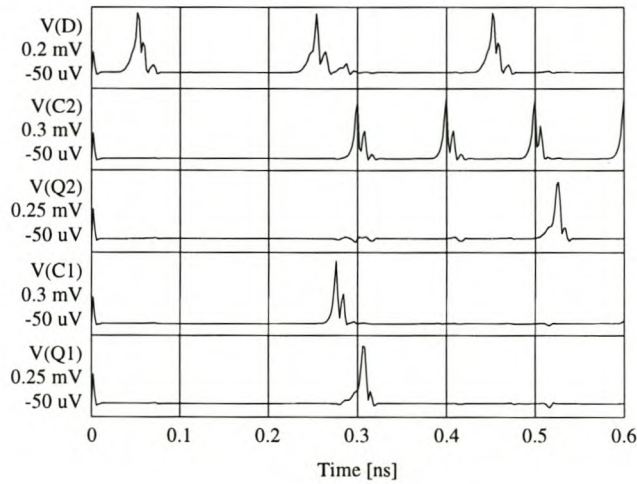


Figure 3.17: Circuit diagram for RFSQ DRO with 2 readouts



(a)



(b)

Figure 3.18: (a) Simulation test setup and (b) simulated response of RFSQ DRO with 2 readouts to SFQ clock and input pulses

From the simulation results in Fig. 3.18(b) it can be seen that only the first reset pulse applied after the register was set, produces an output pulse.

3.2.3 LOGIC GATES

3.2.3.1 Inverter (NOT-gate)

The inverter shown in Fig. 3.19 was obtained from the Stony Brook cell library [18], although the damping resistances still had to be calculated.

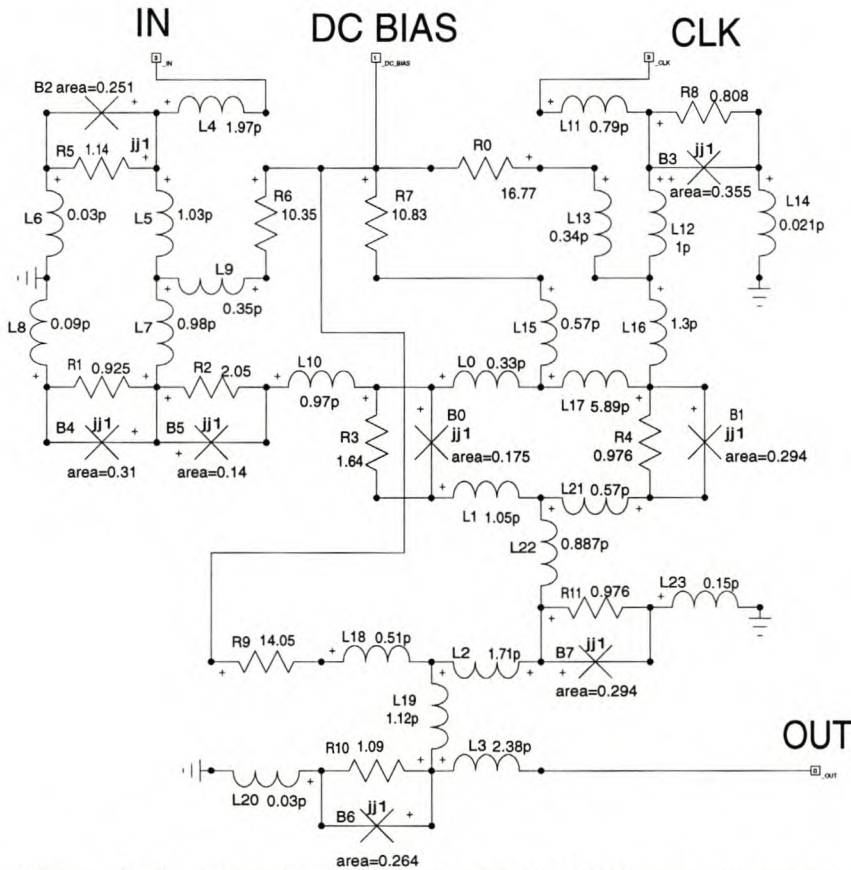


Figure 3.19: Circuit diagram for RSFQ NOT-gate (inverter)

The inverter, like the DRO, is built around a two-junction SQUID (B_0 , L_1 , L_{21} , B_1 , L_{17} , L_0 .) The initial condition of the inverter is determined by the size of L_{17} , which is large enough to force the bias current through R_7 to flow counter-clockwise through the two-junction SQUID.

When an SFQ pulse is applied at IN, it propagates through junctions B_2 and B_4 , and then through B_5 to switch the biased junction B_0 . The switching action reverses the current flow through the two-junction SQUID, so that B_1 is now biased.

A clock pulse applied at CLK now switches B_3 , and finds B_1 biased, so that B_1 also switches. This resets the two-junction SQUID to the initial state, and no output pulse is generated. One half of the inverting function (see Table E.1) is thus implemented, as an IN pulse followed by a CLK pulse results in no output.

If no pulse is applied at IN during a clock cycle, the two-junction SQUID is still in the initial state when a pulse arrives at CLK. This pulse then propagates

through B_3 , finds B_1 unbiased, and rather switches B_7 and B_8 in turn to produce an output pulse at OUT. The inverting function is thus completed.

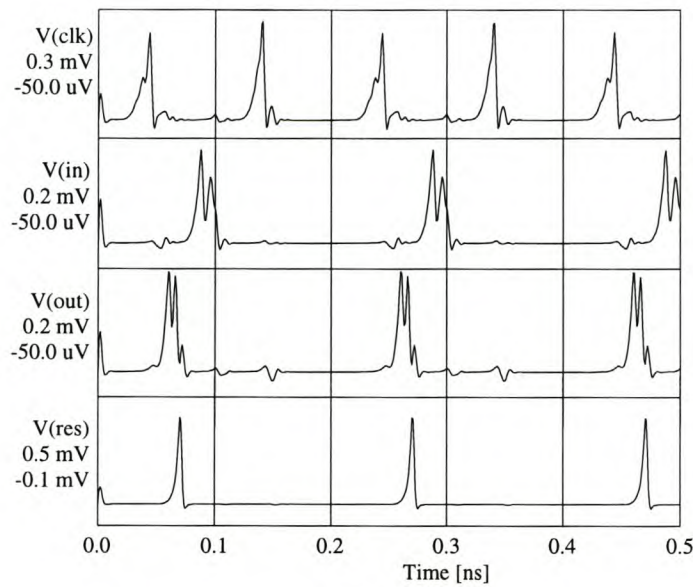
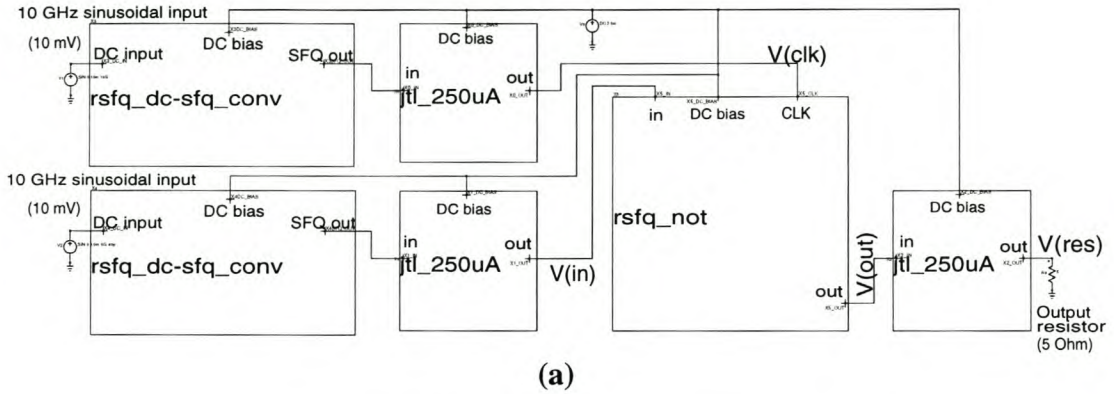


Figure 3.20: (a) Simulation test setup and (b) simulated response of RFSQ NOT-gate to SFQ clock and input pulses

3.2.3.2 OR-gate

The RSFQ OR-gate is not available in the Stony Brook cell library [18], probably because it can be completely constructed from other cells. An RSFQ OR-gate was needed for the construction of a logic full adder (Sec. 4.6.1,) and one had to be assembled.

The definition of the logic OR-function (Table E.1) requires that an output pulse F be generated if either or both of the inputs A and B received logic “1” inputs during the clock period.

When it is clocked, the RSFQ DRO will produce an output pulse if an input pulse arrived at any time during the clock cycle. The auxiliary junction B_2 in Fig. 3.11 throws out any pulse arriving after the first in a given clock period, so that the DRO will still produce an output pulse even if two or more input pulses are applied.

The RSFQ DRO therefore functions like an OR-gate, except that it has only one input. This is rectified by adding an RSFQ pulse merger to the input of the RSFQ DRO. The resulting circuit for the RSFQ OR-gate is shown in Fig. 3.21.

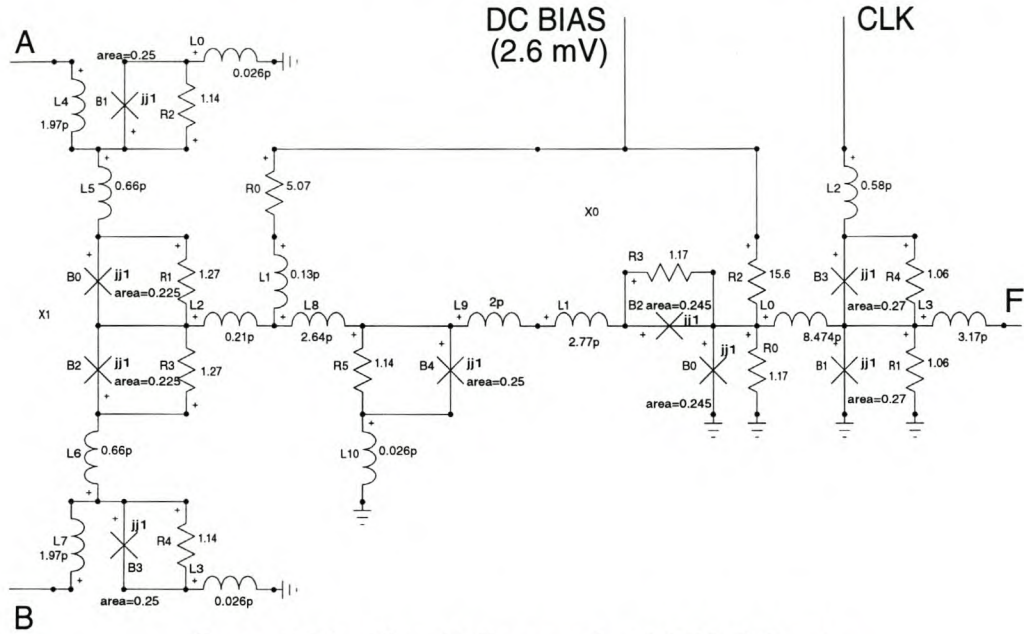
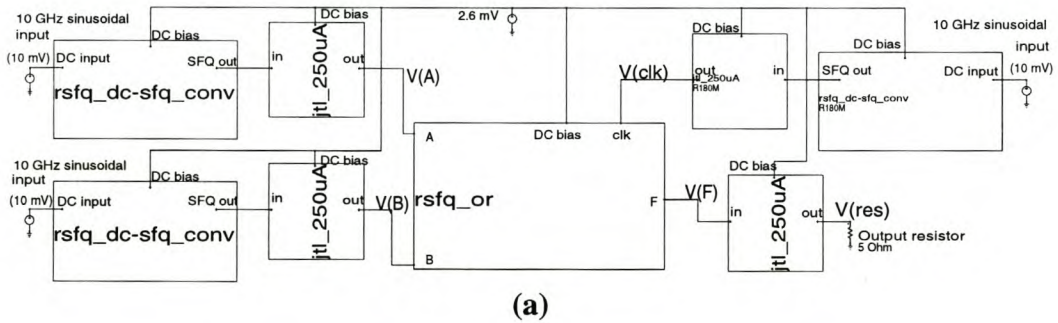
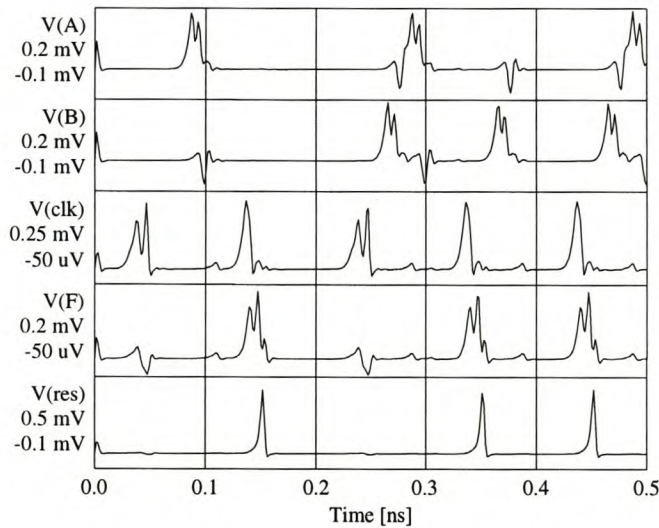


Figure 3.21: Circuit diagram for RSFQ OR-gate



(a)



(b)

Figure 3.22: (a) Simulation test setup and (b) simulated response of RFSQ OR-gate to SFQ clock and input pulses

A final improvement was made after Monte Carlo simulations showed inductors L_5 and L_{31} to be too large. This is discussed in more detail in Sec. 3.5.3.

The RSFQ AND-gate still functions exactly the same as the one proposed by Likharev and Semenov. The inputs IN A and IN B are fed into two flux loops formed by $B_0 - L_1 - B_1$ and $B_{16} - L_{34} - B_{17}$ respectively.

A clock pulse at CLK is branched by the pulse splitter formed by B_9, B_5 and B_{10} , and the two resulting pulses reset the two flux loops. If a flux loop was in the set state, the output pulse propagates into a pulse merger, switching a one-junction JTL (B_4 or B_{13}) along the way. The auxiliary junctions B_6 and B_7 absorb reverse pulses.

Only if both the flux loops are in the set condition when a clock pulse is applied, and both junctions B_4 and B_{13} are switched as a result of the flux loops resetting, will enough current be diverted through junction B_8 in order for it to switch.

Junction B_8 has $I_C = 600 \mu A$. The output JTL with junctions B_{11} and B_{12} merely serves to interface B_8 to the standard JTL input junction I_C of $250 \mu A$.

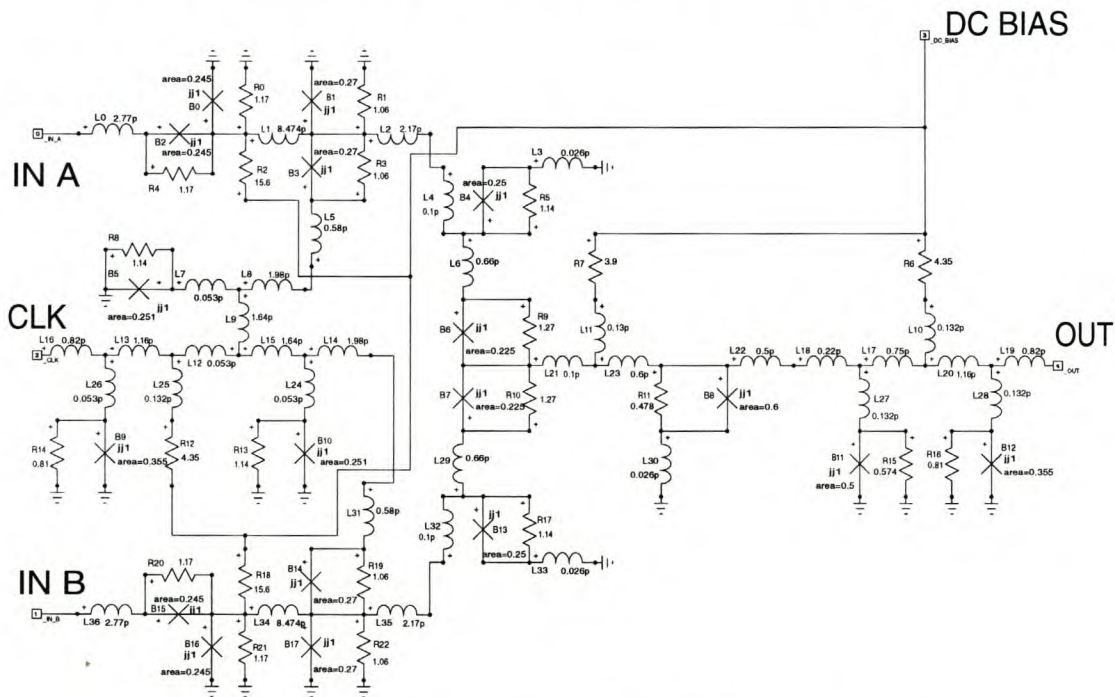


Figure 3.24: Circuit diagram for RSFQ AND-gate

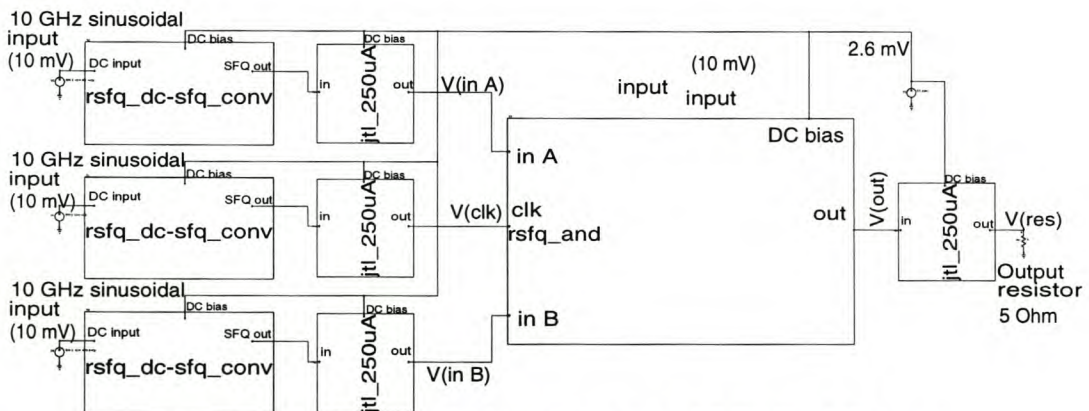


Figure 3.25: (a) Simulation test setup of RFSQ AND-gate

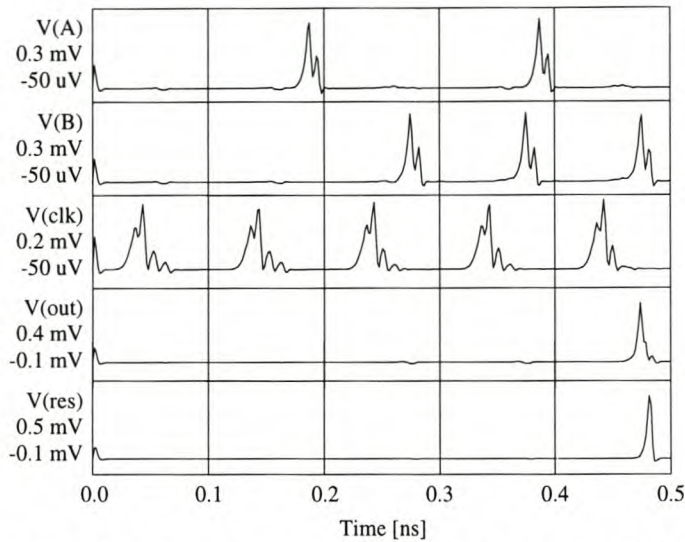


Figure 3.25: (b) Simulated response of RFSQ AND-gate to SFQ clock and input pulses

3.2.3.4 XOR-gate

The RSFQ XOR-gate was also obtained from the Stony Brook cell library [18], although the damping resistances had to be calculated. This is one of the more complex gates, and the circuit diagram is shown in Fig. 3.26.

The gate is constructed around two quantizing loops. The first loop is formed by B_5 , B_1 , L_0 , L_1 , B_2 and B_3 , and the second by B_8 , B_0 , L_2 , L_1 , B_2 and B_3 . Both loops share L_1 , B_2 and B_3 .

The bias currents are injected through R_4 and R_6 , and because of inductors L_0 and L_2 , are constrained to flowing through the junction pairs $B_1 B_5$ and $B_0 B_8$ respectively in the initial state.

With both loops in the initial or unset state, an input pulse at CLK switches junction B_6 . No output pulse is produced, in accordance with the definition of the XOR logic function shown in Table E.1.

If an input pulse is applied at A, it switches the biased junction B_5 , reversing the current through the loop so that it now flows through junctions B_2 and B_3 . An input pulse at CLK now switches junction B_3 . This yields an output pulse at F, and also switches junction B_2 to reset the loop current to the initial state. Since the quantizing loops are symmetric, the same explanation holds true for an input pulse at B.

If input pulses are applied at both A and B, both quantizing loops switch to divert current through B_2 and B_3 . The combined current is large enough to cause junction B_2 to switch asynchronously. This resets both quantizing loops to the initial state, and when a pulse arrives at CLK, it finds the gate in exactly the same condition as when there were no inputs at A or B. The result is that no output pulse is produced.

Auxiliary junctions B_4 and B_7 serve as buffers to throw out any extra pulses that might arrive at A or B after a first pulse. Junctions B_0 and B_1 are also auxiliary junctions, and absorb the pulses generated by the reset of the quantizing loops. These junctions prevent parasitic pulses from propagating in the reverse direction out of inputs A and B.

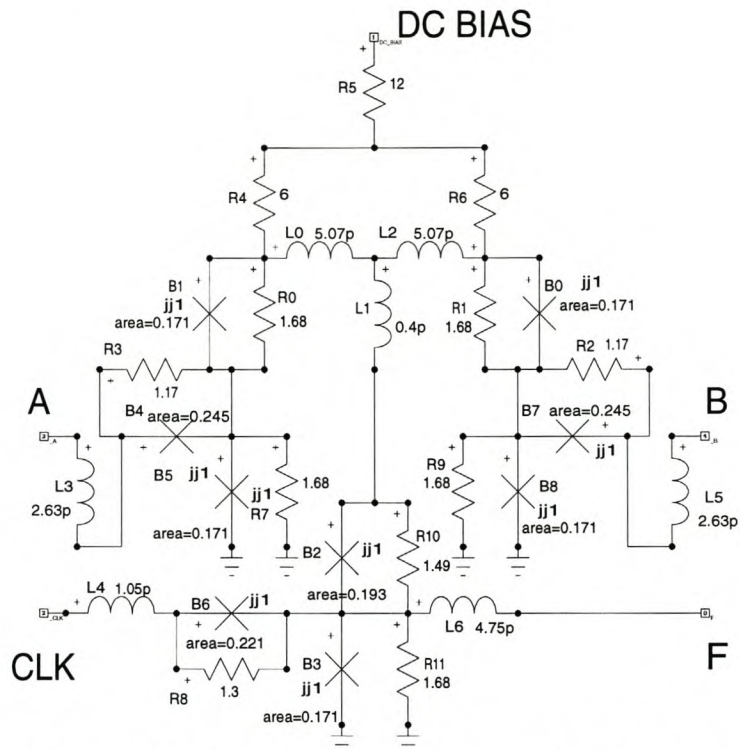
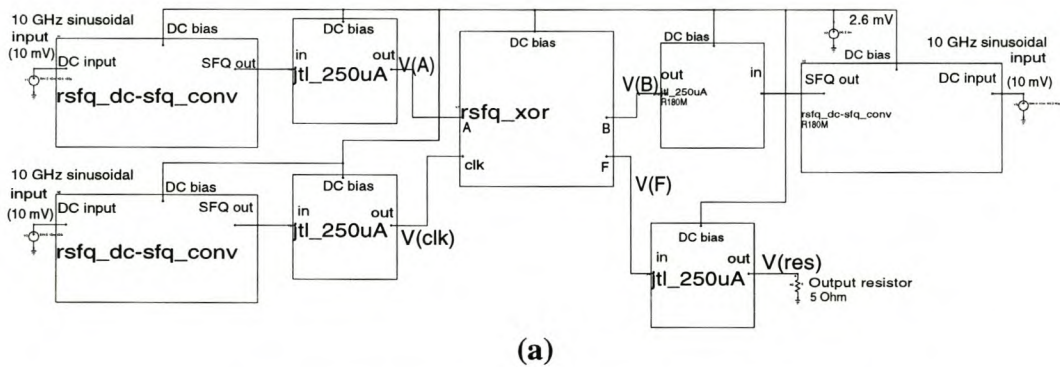
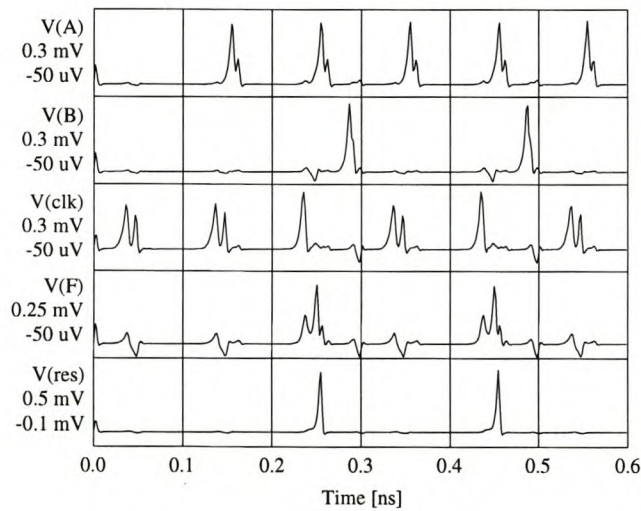


Figure 3.26: Circuit diagram for RSFQ XOR-gate



(a)



(b)

Figure 3.27: (a) Simulation test setup and (b) simulated response of RFSQ XOR-gate to SFQ clock and input pulses

3.2.4 INTERFACE DEVICES

3.2.4.1 DC-to-SFQ converter

The DC-to-SFQ converter is one of the most important devices in the RSFQ cell collection, since this component is used to generate SFQ clock pulses for nearly all the synchronous components in the ADC design.

Several configurations were considered as an alternative to the circuit in the Stony Brook cell library [18], as it was initially considered to be unreliable and difficult to clock. The best alternative is an adaptation of the COSL OR-gate (see Sec. 3.3.1.1.) With the inductively coupled second stage stripped away, the COSL OR-gate can be made to fire SFQ pulses when a sinusoidal clock signal is supplied. The Stony Brook circuit was eventually selected because it turned out to be faster, more reliable after trimming, and less prone to pulse staggering when the circuit parameters are varied.

The stripped COSL OR-gate did find application though, as it was used to create the comparator discussed in Sec. 3.3.2.

The DC-to-SFQ converter is shown in Fig. 3.28. Once again the damping resistances had to be calculated.

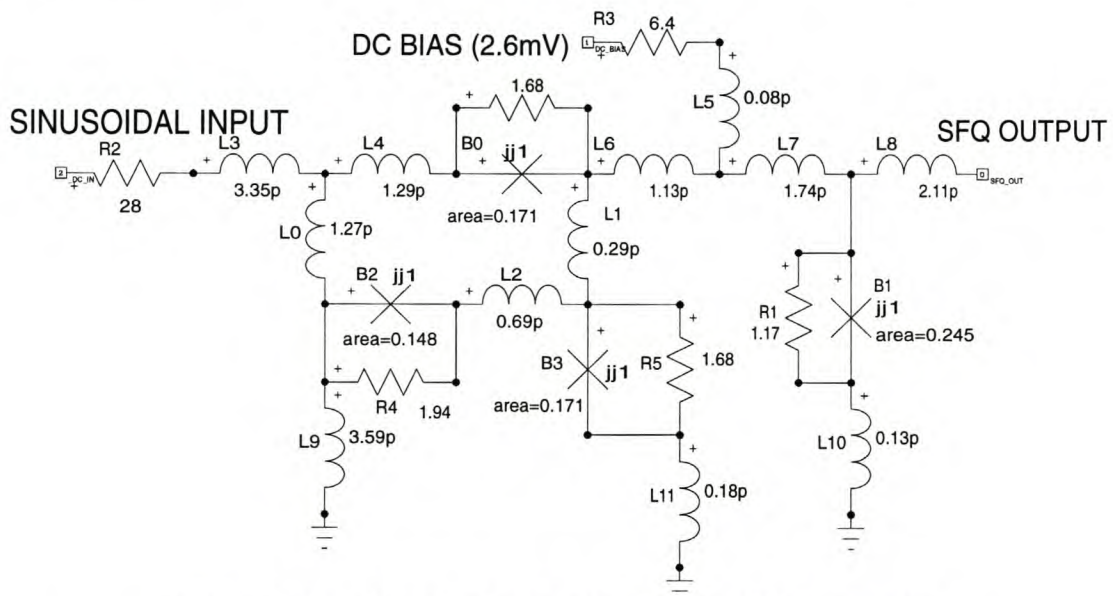


Figure 3.28: Circuit diagram for DC-to-SFQ converter

The DC-to-SFQ converter is built around the two-junction SQUID formed by B_3 , B_2 , B_0 , L_9 , L_4 and L_2 . Junctions B_2 and B_0 act as a single junction. When the input current is raised above a certain value, junction B_3 switches. The resulting pulse propagates to the SFQ output after being sharpened by optional junction B_1 . A fluxon is also captured in the two-junction SQUID. This lowers the current through junction B_3 , and allows the input signal to be noisy or inexact within 0.5 mA [18] without causing extra pulses to be generated.

When the input current falls below a certain value, B_0 and B_2 flip, releasing the stored fluxon and resetting the device.

The dc bias resistor (R_3) was lowered from the 7 Ω used in the Stony Brook model to 6.4 Ω , in order to increase the theoretical yield.

Another component value that needs explanation, is the input resistor R_2 . This resistor is needed to limit the current injected into the DC-to-SFQ converter by an applied clock signal. A value of 28.57Ω was obtained by mathematically transforming the test setup used at Stony Brook to a resistor fed by a 10 mV peak sinusoidal signal. The final value was adjusted to 28Ω after several Monte Carlo simulations with slightly different values for R_2 were analyzed.

The input signal amplitude was selected to be 10 mV, so that the three-phase COSL clock signals could be used to also generate the SFQ clocks. During Monte Carlo simulations, however, it became evident that the global variations on resistance could alter the value of R_2 enough to cause circuit failures. The solution was to use voltage trimming, as discussed in Sec. 3.5.3 and in more detail in Sec. 6.4.

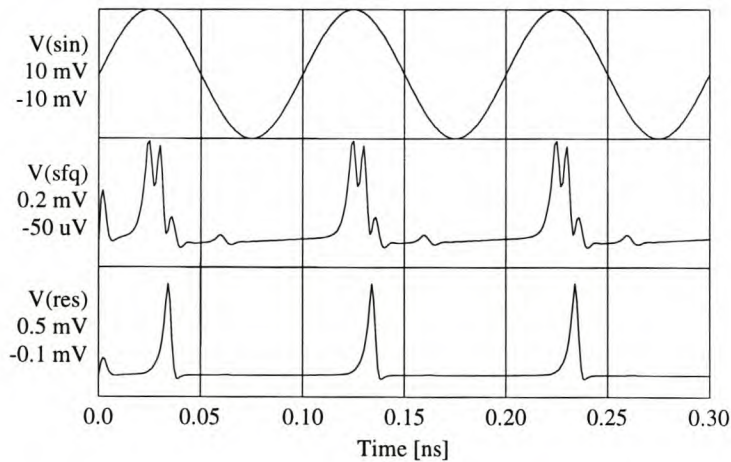
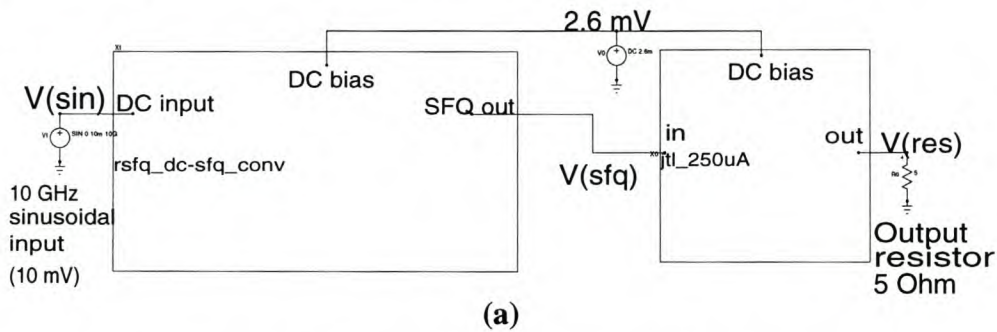


Figure 3.29: (a) Simulation test setup and (b) simulated response of DC-to-SFQ converter to sinusoidal input voltage

3.2.4.2 SFQ-to-DC converter

The SFQ-to-DC converter was obtained from the Stony Brook cell library [18]. Even though the circuit diagram in Fig. 3.30 may look complicated, the device is merely a T-flip-flop that supplies a control current to a dc SQUID. Unlike COSL gates, the dc SQUID is connected directly to the T-flip-flop, and not coupled inductively.

The T-flip-flop is formed by junctions $B_0, B_2, B_3, B_4, B_5, B_8$ and B_9 . The dc SQUID reads the state of the T-flip-flop. When no fluxon is stored in the T-flip-flop, the dc SQUID is unbiased and in the superconducting state. An input pulse at the SFQ input sets the T-flip-flop. A fluxon is stored in the T-flip-flop, and the circulating current causes the dc SQUID to be biased into the normal state. When B_1

is in the normal state, an oscillating voltage is produced that can be measured at the output of L_{16} (DC OUT.) A second input pulse at the SFQ input resets the T-flip-flop, and the dc SQUID returns to the superconducting state.

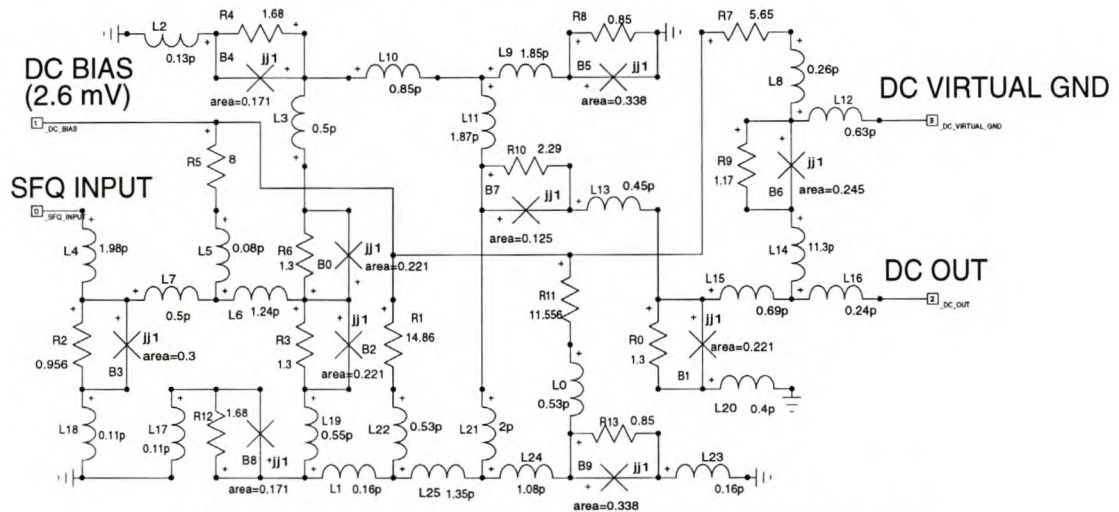
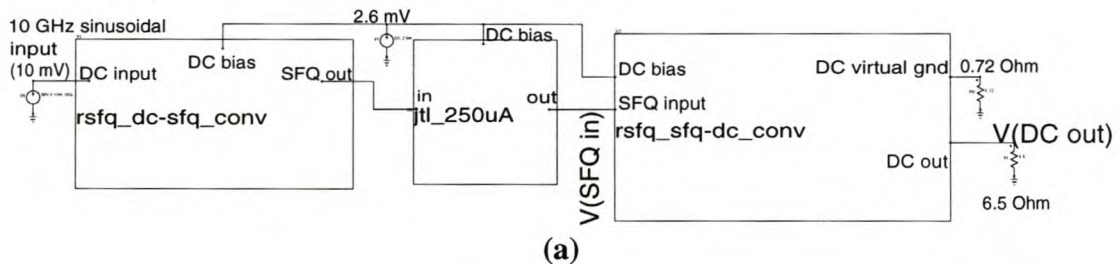
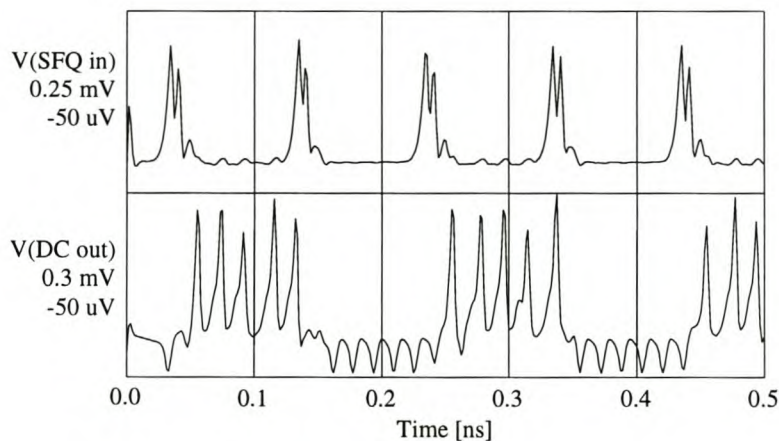


Figure 3.30: Circuit diagram for SFQ-to-DC converter

The average value of the oscillating output for the normal state of the dc SQUID is between $100 \mu\text{V}$ and $130 \mu\text{V}$ higher than the average output when the dc SQUID is in the superconducting state, and the difference can easily be measured with high frequency measurement equipment.



(a)



(b)

Figure 3.31: (a) Simulation test setup and (b) simulated response of SFQ-to-DC converter to SFQ input pulses

3.3 Miscellaneous circuit components

3.3.1 COSL GATES

3.3.1.1 COSL OR-gate

The COSL OR-gate was designed and optimized by W.J. Perold *et al.* [13]. COSL (Complementary Output Switching Logic) is a voltage state logic, and operates with a three-phase clock. Both of these characteristics proved essential for the design of an analogue-to-digital converter (Chapter 4,) and COSL-gates were consequently incorporated into the RSFQ design. Hence the inclusion of some of these gates into the list of building blocks.

The 50 Ω resistor (R_9) in Fig. 3.32 connects the OR-gate input to an off-chip trim voltage. This voltage is an adjustable dc bias that makes it possible to trim a non-functional circuit in order to compensate for global parameter offsets [13]. One trimming input serves all COSL OR-gates on a chip, since the global offsets affect all the gates equally.

The COSL OR-gate is optimized for nominal input and output currents of 200 μA into a 5 Ω load. The nominal output voltage is therefore 1 mV.

Since input currents of 100 μA as well as 200 μA will switch the gate and produce a 1 mV output, the logic OR function is achieved.

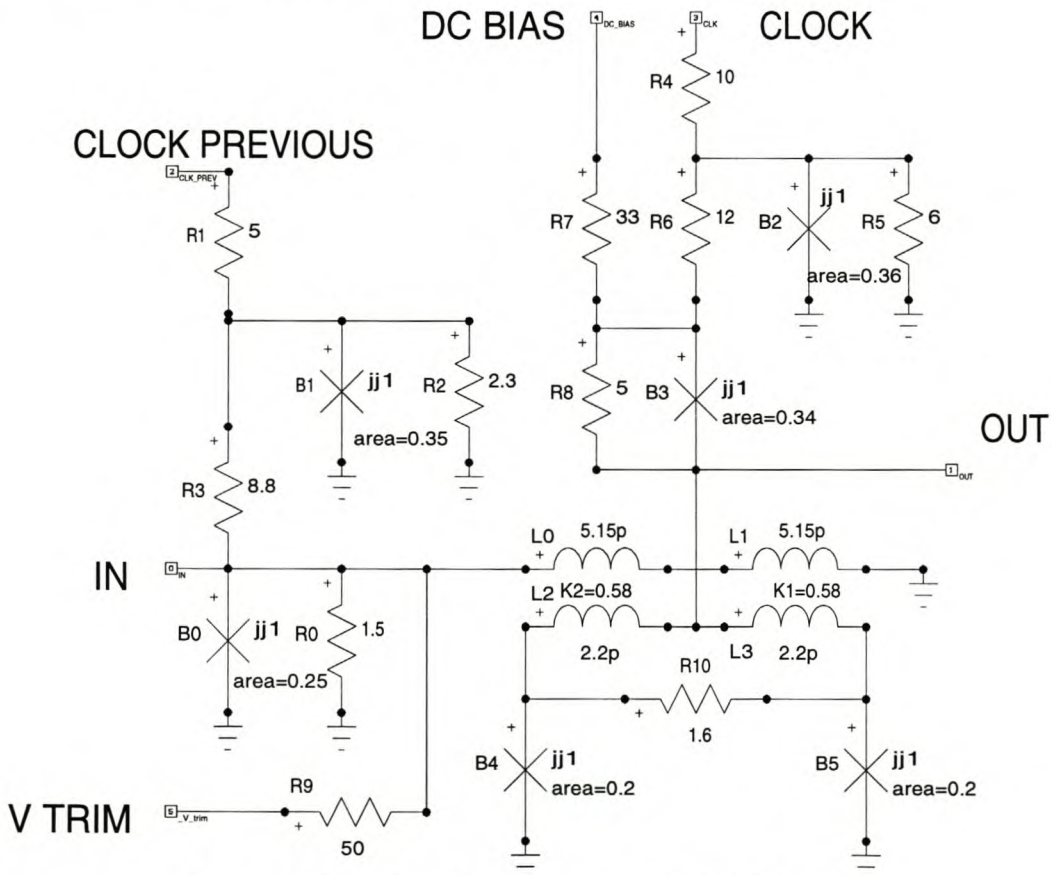


Figure 3.32: Circuit diagram for COSL OR-gate

The simulation results shown in Fig. 3.33 were generated for a COSL OR-gate of which the output was connected to a $5\ \Omega$ resistive load. The input signals A and B were both applied at IN through separate series resistances of $10\ \Omega$ each. CLOCK lags CLOCK PREVIOUS by $33.33\ \text{ps}$, or 120° of a wavelength.

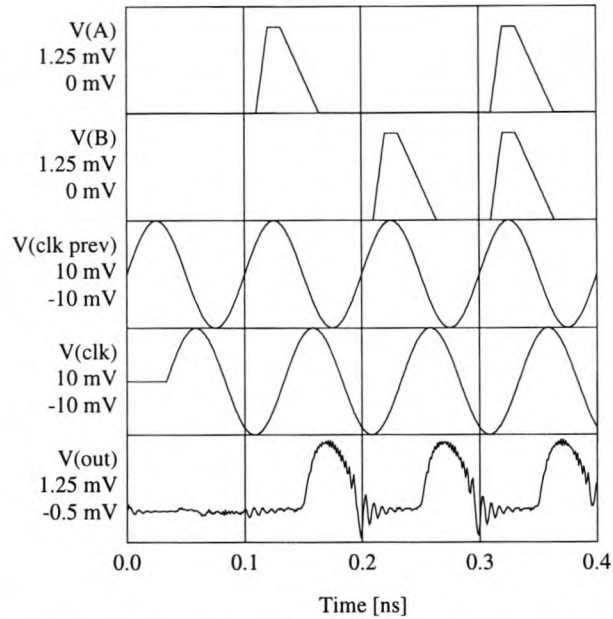


Figure 3.33: Simulated response of COSL OR-gate to applied input signal

When the clock signal applied at CLOCK goes high, either the single junction B_3 or the dc SQUID formed by B_4 and B_5 switch. With no control current to induce current in the dc SQUID, the single junction B_3 switches. However, if a control current is inducing current in the dc SQUID, the SQUID switches to the normal state, and an output voltage is generated.

3.3.1.2 COSL NOR-gate

The COSL NOR-gate [13] is shown in Fig. 3.34. This gate is exactly the same as the OR-gate as far as the clocking scheme and the input and output loading requirements are concerned.

The difference between the COSL NOR-gate and the COSL OR-gate lies in the position of the output node and the single junction (B_5 in Fig. 3.34, and B_3 in Fig. 3.32.) When the sinusoidal clock signal at CLOCK goes high, either the single junction B_5 or the dc SQUID (B_4 and B_3) switches. Just as is the case for the COSL OR-gate, the dc SQUID switches if a control current (caused by one or more high inputs) is present. However, since the output is now measured over the single junction B_5 , it will only be high when this junction switches. This happens when all the inputs are low, and no control current is flowing in L_1 and L_0 . The result is that the gate in Fig. 3.34 implements the inverse function of the COSL OR-gate, and is therefore a COSL NOR-gate.

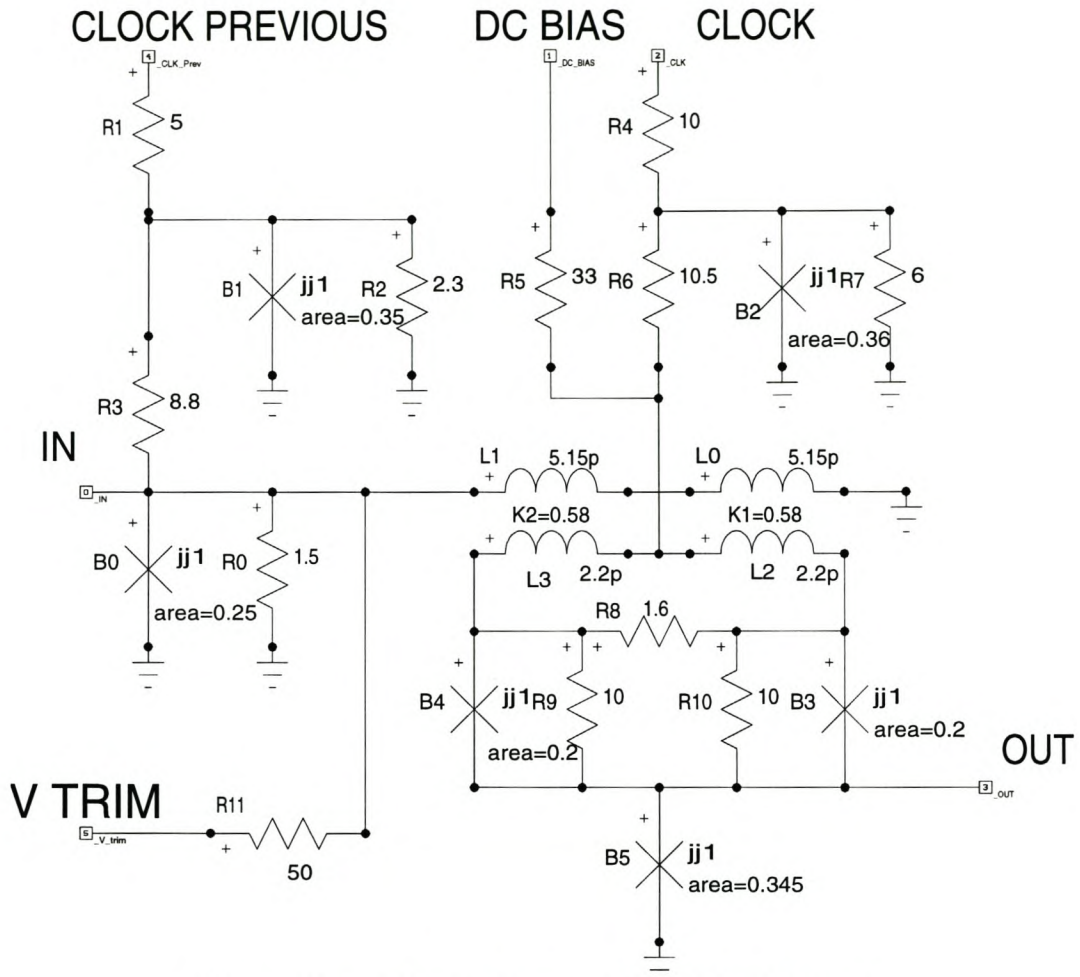


Figure 3.34: Circuit diagram for COSL NOR-gate

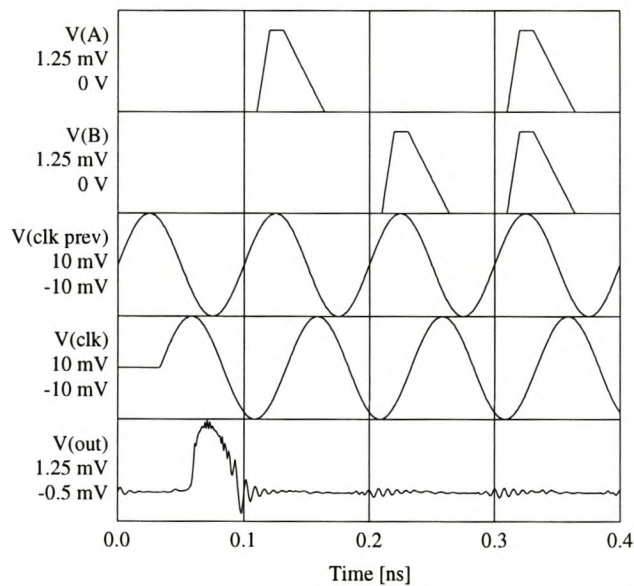


Figure 3.35: Simulated response of COSL NOR-gate to applied input signal

3.3.1.3 Negative output COSL OR-gate

The initial design of the ADC required a negative dc feedback voltage (see Sec. 4.5.) The least complicated solution would be to use negative output COSL OR-gates [19].

The negative output COSL OR-gate is architecturally exactly similar to the standard COSL OR-gate – the difference between the gates only lies in the dc bias and clock sequence.

With a dc bias of -5 mV and an output clock phase that leads the input clock phase by a third of a clock cycle, the negative output COSL OR-gate switches when the output clock current is negative. This produces an output voltage of -1 mV into a 5Ω load.

Since the negative output COSL OR-gate is exactly similar the standard COSL OR-gate in Fig. 3.X1, no circuit diagram is shown here. The test setup is the same as for a COSL OR-gate, with a 5Ω resistor loading the output, and series resistors of 10Ω limiting the current from each input signal.

The dynamic response of the negative output COSL OR-gate is shown in Fig. 3.36. Refer to Fig. 3.32 for signal designations.

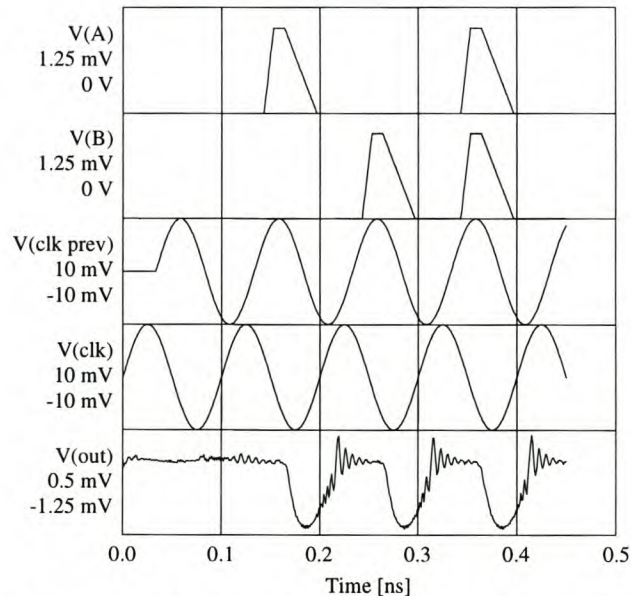


Figure 3.36: Simulated dynamics of negative output COSL OR-gate

Although negative output COSL OR-gates were used in earlier feedback systems, it had to be replaced by standard COSL OR-gates due to compatibility problems with the level detector. This is discussed in Sec. 3.3.2.

Finally, since the circuit diagram for the standard and negative output COSL OR-gates are exactly the same, the Monte Carlo analysis results and a circuit layout diagrams are not repeated.

3.3.2 COMPARATOR

The design of an analogue-to-digital converter (see Sec. 4.5) requires the availability of a level detector, or comparator.

The level detector is required to:

- reset fast enough to sample unambiguously at the next clock,
- switch only when a clock pulse is applied, and not between clocks, and
- produce an SFQ output pulse.

All of the above requirements can be met if the comparator is itself an RSFQ device. Research on RSFQ comparators has already been done, and one of the most promising comparators [30] was used in early versions of the system design.

This “SQUID wheel/decision-making pair” comparator was eventually abandoned because it did not simulate well at the time, and because of problems with the feedback path.

The first designs for the ADC featured analogue integration in the current feedback path. This would be achieved by using a lowpass filter to approximate the integrator. Unfortunately, the filter components turned out to have values that would be difficult to achieve on the physical chip, and the current flowing into the feedback path put an excessive load on the comparator.

Analogue integration was eventually replaced with digital integration, and although this solved some of the problems experienced with the SQUID wheel comparator, this component was discarded along with analogue integration.

A new comparator had to be designed, and the most promising candidate was the COSL OR-gate, since it was clear by this stage that a hybrid logic structure would be used to implement the digital feedback system of the ADC.

The COSL OR-gate triggers if an input current of $100\ \mu\text{A}$ is applied when the input clock is high. This allows current feedback to be used, since the currents can be added into the COSL OR-gate input through resistive networks. This is made possible by one of the characteristics of superconductivity: the input junction of the COSL OR-gate is a perfect dc short-circuit except during the time that the junction switches. Consequently, all feedback resistor calculations are simplified, since the COSL OR-gate input potential equals ground.

The COSL OR-gate fulfills all the requirements for a comparator, except that the output is a voltage state signal, and not an SFQ pulse as desired. This is rectified by inserting a DC-to-SFQ converter after the COSL OR-gate.

At this stage it is important to mention that several design choices had to be made as the project progressed. Although certain subsystems had to be exchanged for better alternatives when changes to the ADC structure rendered them obsolete, later changes could sometimes reverse the process.

The comparator is the subsystem subjected to the largest number of forced design changes. One of these design changes required a faster comparator, since the cascaded structure of the COSL OR-gate and DC-to-SFQ converter took roughly 50 ps to produce an output pulse.

Due to timing restrictions imposed by the first two fast feedback cells discussed in Sec. 4.6.3, the comparator needed to be faster than 50 ps. A new design was thus necessitated, but instead of starting from scratch, the existing COSL OR-gate structure was used as a starting point.

With the DC-to-SFQ converter removed from the comparator configuration, the device was speeded by between 10 ps and 15 ps.

A close study of the operation of a COSL gate (see Fig. 2.9 and Fig. 3.32) shows that the combination of the output SQUID and the series junction generates the voltage state logic output. As part of the operation, a current can be induced in the output SQUID by inductive coupling to the one-junction SQUID at the input. The one-junction SQUID works like an RSFQ JTL, and this observation led to the design

of a new comparator. If the output stage of a standard COSL OR-gate is removed, the stripped down remainder produces SFQ pulses.

The COSL OR-gate input SQUID is not matched to standard RSFQ logic inputs, hence the addition of a second one-junction SQUID, or pulse shaper, at the output. The “stripped COSL” comparator is shown in Fig. 3.37.

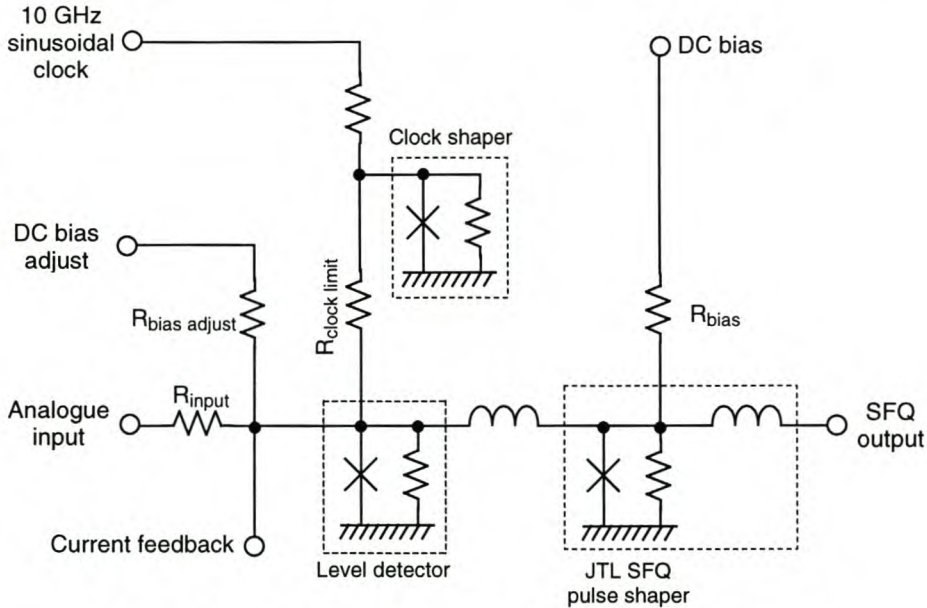


Figure 3.37: “Stripped COSL” comparator schematic

In the comparator circuit shown in Figure 3.38, the value of R_1 was adjusted until the device switched when a $100 \mu\text{A}$ input current was applied at the input, and the clock signal was high. This value was chosen to be comparable to the switching currents in standard COSL gates. The value for R_1 was ultimately fixed at 13.5Ω .

A discussion on bit size and feedback resistor sizes requires familiarity with the specifications and design of the ADC, and is therefore relegated to Sec. 4.5.

The “stripped COSL” comparator is not the only circuit that would function correctly as a level detector. After the final design iteration on the FIR filter and feedback path of the ADC (Chapter 4,) it turned out that the COSL OR-gate in series with the DC-to-SFQ converter could still be used as a level detector, although two special “COSL-only” feedback paths would be needed for the first two feedback paths. However, the “stripped COSL” comparator worked so well that it was decided not to alter the design.

Unlike the other RSFQ components, the level detector is biased by a 5 mV dc voltage. This is because the higher bias voltage narrows the time window in which a pulse is generated when the input current varies, and because 5 mV is available on-chip for biasing the COSL components.

In the simulation results Fig. 3.39(a), six other feedback signals (not shown) are high between 300 ps and 400 ps, and also between 400 ps and 500 ps.

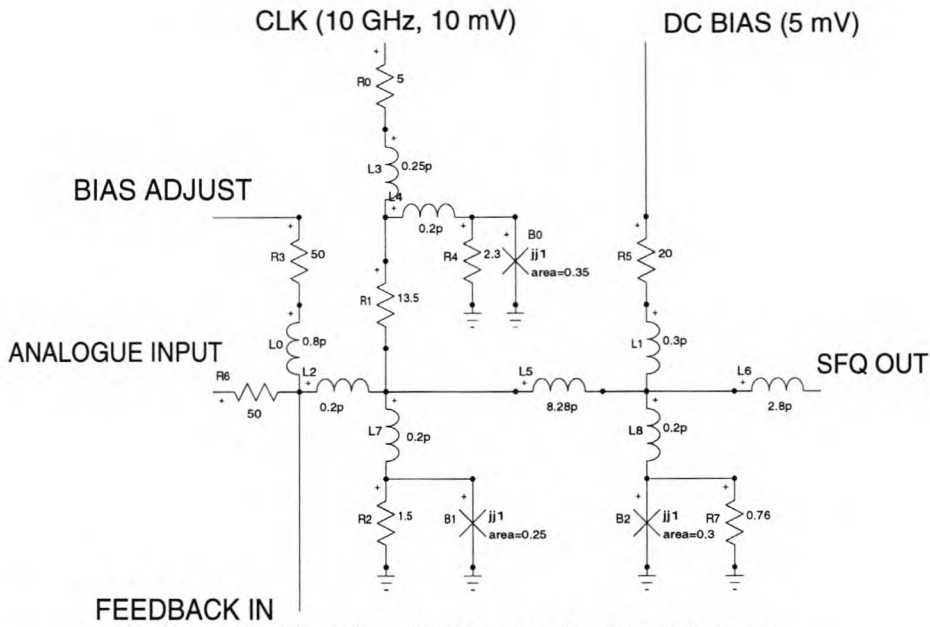


Figure 3.38: Circuit diagram for level detector

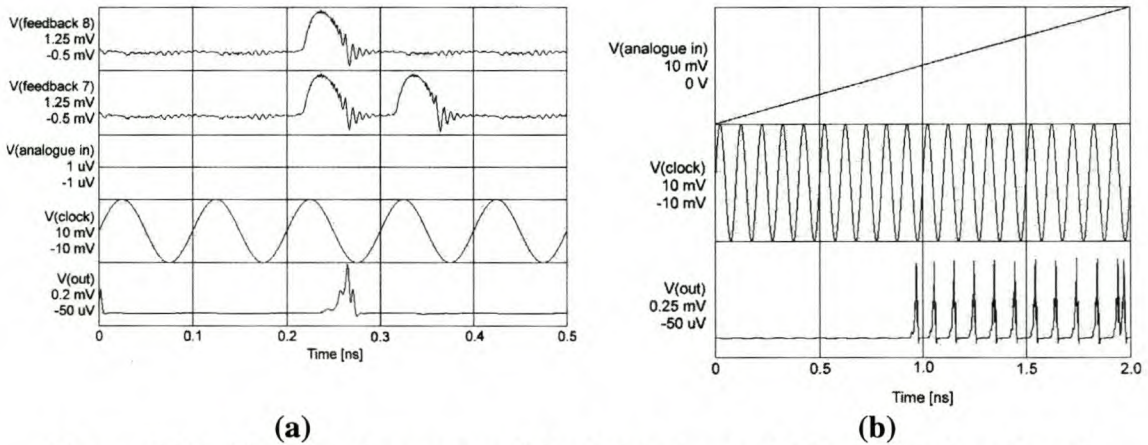


Figure 3.39: Simulated response of level detector to (a) feedback signals and zero analogue input, and (b) ramp analogue input and no feedback

3.3.3 RSFQ DRO TO COSL OR-GATE INTERFACE

A component was needed to provide an interface between RSFQ and COSL systems. The interface was constructed by connecting an RSFQ DRO to a COSL OR-gate through a series Josephson junction (B_7 in Fig. 3.40.)

The interface works on the principle that an SFQ pulse can be applied to the DRO at any time during a clock cycle, and that a clock pulse can reset this DRO to inject the SFQ pulse into the COSL OR-gate. If the SFQ clock pulse is generated by the same sinusoidal clock phase (through a DC-to-SFQ converter) that clocks the input to the COSL OR-gate (CLOCK PREVIOUS,) the SFQ pulse leaving the DRO will add to the clock current in the COSL OR-gate, and cause the latter to switch.

Series junction B_7 prevents reverse pulse propagation. The other elements are all part of either the DRO or COSL OR-gate, although certain values were adjusted during optimization routines.

The simulation test setup is shown in Fig. 3.41(a), and the simulation results in Fig. 3.41(b).

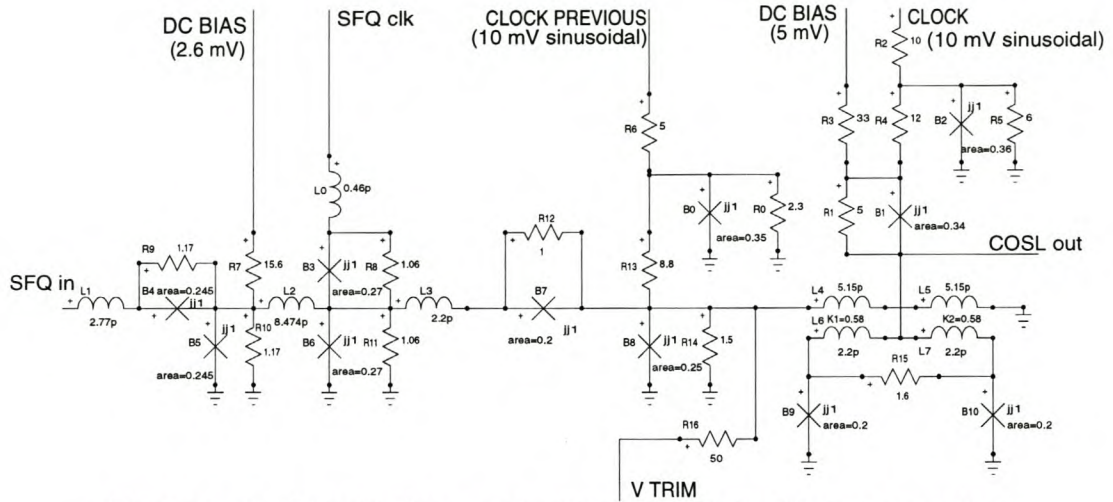
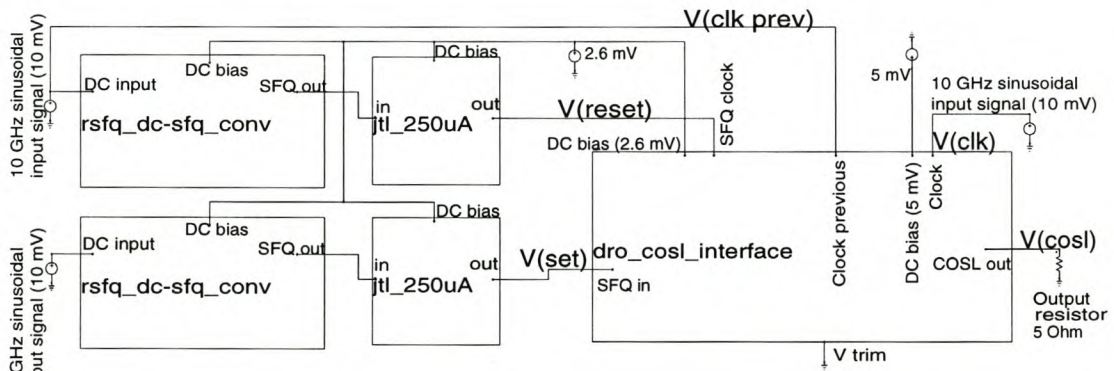
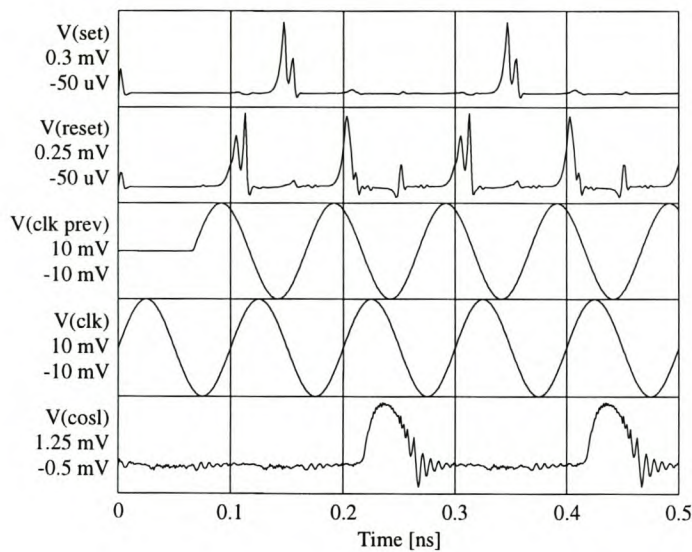


Figure 3.40: Circuit diagram for DRO to COSL OR-gate interface



(a)



(b)

Figure 3.41: (a) Simulation test setup and (b) simulated response of DRO to COSL OR-gate interface to input signals

3.4 Simulations

Although all simulated results in this chapter, with the exception of Fig. 3.4, were generated by WRSpice [24], several circuit simulation packages were used at various stages of the project. Some Monte Carlo simulations were done in HSpice [31], while Intusoft's IsSpice4 (part of the ICAP/4 package) [32] was used as a slower, less user friendly alternative to WRSpice when the latter was unavailable or inaccessible.

The simulation diagram of each RSFQ logic cell in Sec. 3.4 is preceded by a schematic diagram showing the simulation test setup. The rationale behind the use of standardized test setups with dynamic input and output loads in the form of JTLs is discussed in detail in Sec. 3.5.3.

In order to simulate SFQ input pulses, DC-to-SFQ converters were used at the input stages of all RSFQ simulation setups. The DC-to-SFQ converters were then fed with sinusoidal input signals to produce SFQ pulses, and these pulses were applied to the device under test (DUT.) A schematic diagram showing the standard simulation setup is shown in Fig. 3.42.

As has been mentioned in Sec. 3.1, the circuit diagrams shown in this chapter were generated from the simulation models used in WRSpice. The numerous inductors of less than 1 pH each present in all the RSFQ circuit diagrams, represent the parasitic inductances caused by circuit layout.

These parasitic inductances had to be included in the simulation models, as they are only in the region of one order of magnitude or less smaller than the inductors used to create the one-junction and two-junction SQUIDs at the heart of the RSFQ circuits. Large parasitic inductances have a rather severe degrading effect on circuit operation and yield, so that layout had to be done carefully to keep these parasitics down.

Since the physical layouts were done after simulations were used to design or verify each RSFQ circuit, the parasitic inductances were initially taken to be equal to those of the Stony Brook cell library layouts [18]. After the layouts shown in Sec. 6.4 were completed, the parasitic inductances were calculated from the layout dimensions. Where different values than those of the Stony Brook cell library were obtained, the new values were used in Monte Carlo simulations in order to ascertain the effect of layout parasitics on circuit yield.

The COSL gates do not have parasitic inductances built into the circuit diagrams or simulation models, as these parasitics are not critical, and do not degrade the performance of COSL components in the same way that it does that of RSFQ circuits.

After a logic component was found to function properly when simulated in a standard environment, and with the proper input signals, it was tested for reliability. In these simulations, the values of all elements in the circuit were varied randomly within certain boundaries, after which the simulation was performed. The elements were then assigned new values, and the simulation repeated. The cycle was repeated hundreds or thousands of times, and the simulation results were then used to evaluate the reliability of the circuit. This technique is known as a Monte Carlo analysis, and is the subject of the following section.

3.5 Monte Carlo analysis and device optimization

Goldfinger's flat, hard stare didn't flicker. ... He said, 'Mr. Bond, they have a saying in Chicago: "Once is happenstance. Twice is coincidence. The third time it's enemy action."'

Ian Fleming's "Goldfinger"

If something can go wrong, it will.

Murphy's Law

3.5.1 DEFINITION OF MONTE CARLO ANALYSIS AND YIELD PREDICTION

Due to component and layer variations in the manufacturing of integrated circuits, the physical cells will have tolerances on all parameters.

It is necessary to verify that each logic cell, as well as the entire circuit, will have a high yield. This is done by performing a Monte Carlo analysis on each logic cell [13], as well as on more complex structures (Sec. 4.8.)

A traditional technique used to verify reliability in simulated gates is called margin analysis, and relies on the variation of one or two parameters for consecutive runs. The functionality of a gate is then observed, and the sensitivity of the gate to changes in the parameters is obtained [16].

It is clear that margin analysis does not take into account the fact that all parameters may vary, and that any parameter can have a random deviation independent of any other parameter deviation in a specific simulation run.

During the last decade, desktop computing power has increased so significantly that it is now possible to do repeated simulations even on advanced or complex circuits in short timespans. This allows design engineers to perform a Monte Carlo analysis on every logic cell or gate.

In a Monte Carlo analysis, every parameter in a circuit simulation model can be assigned a value that is statistically varied around the nominal value, independent of any other parameter in the model.

There are two ways in which a parameter value can vary. The chip-to-chip variations in microchip parameters that influence all components equally, like layer thickness, are modeled by global variables. Each of these variables are assigned a Gaussian distribution, and multiplied into all component values that are functions of the specific global variable. Local (on-chip) variations like different line widths or other etching differences are modeled by creating an independent Gaussian distribution for every component in the circuit model. Both effects are taken into account for every component by varying the nominal value with both the global and local distributions.

A Monte Carlo yield analysis provides a much more accurate prediction than margin analysis of the theoretical yield of any circuit, as all parameters can be varied according to the physical parameter variations of the construction process.

It can be shown [13], [33] that the true statistical yield y of any gate lies within the interval delimited by the observed yield y' and the confidence interval L . The governing equations for the statistical yield and confidence interval are

$$y = y' \pm L \quad (4.1)$$

and

$$L = k \sqrt{\frac{y'(1-y')}{N}} \quad (4.2)$$

where k varies from 2 to 2.6 for confidence levels of 95 % and 99 % respectively, and N is the number of Monte Carlo cycles.

As can be expected, the confidence interval can be decreased by increasing the number of Monte Carlo cycles.

3.5.2 DEFINITION OF A CIRCUIT FAILURE

Having discussed the yield analysis procedure, we can now define a circuit failure. The traditional definition for a voltage state logic output classifies a failure as the appearance of a high output when none is desired, or the absence of a high output (or the appearance of a low output) when a high is expected.

Although RSFQ is a pulse logic family, the definition still applies. The absence of an output pulse when one should be generated constitutes a failure, and so too does the appearance of such a pulse when none should be generated.

The input pulse to an RSFQ cell has to arrive within a certain time window during the clock cycle. This is necessary to ensure that the cell has enough time to complete the required switching actions before the arrival of the next clock pulse. If a pulse arrives outside this window, even though it may still be in the correct clock period, the circuit operation may be compromised. Consequently, the definition of a failure has to be extended to include the safe window.

There are other mechanisms of failure that are also unique to pulse voltage logic. As an example, consider the DC-to-SFQ converter.

The DC-to-SFQ converter is used to generate single flux quantum pulses from periodic input signals. The primary use of these converters is to generate clock pulses to drive RSFQ circuitry. The definition of failure as applied to this situation thus implies that a circuit fails if any amount of SFQ pulses other than one is generated during a full cycle of the periodic input signal.

It must immediately be clear that the multiple pulse failure, usually the result of an overdriven circuit, is unknown to semiconducting voltage state logic, where a high output is held constant for an entire clock period.

The DC-to-SFQ converter was optimized for a 10 GHz implementation (see Chapter 4.) The clock period of the RSFQ logic system is therefore 100 ps. The SFQ pulse can appear at any time during this clock cycle, but this may jeopardize the logic operation of a system if two consecutive gates are clocked by different DC-to-SFQ converters, and the respective clock pulses are so far apart that it causes data to “race” through the two gates in one clock cycle. The definition of a failure was therefore adapted to include the time requirements of the circuit. If a 10 GHz sinusoidal input is applied to the DC-to-SFQ converter so that the phase angle equals zero degrees at time zero, then a circuit fails if the SFQ pulse arrives outside the 20 ps (or one fifth of the clock period) between 10 ps and 30 ps.

The time requirement was chosen arbitrarily after looking at the results of repeated Monte Carlo analyses on a single DC-to-SFQ converter and on more complex systems.

Finally, a few general comments about failure definition can be listed.

- Some gates, like the RSFQ OR-gate, are immune to multiple input pulses. In complex systems, the failure definitions for such gates can be eased.
- The safe window is dependent on the length of the clock period. All cells must therefore be analyzed at the same clock frequency. Since the ADC in

Chapter 4 runs at 10 GHz, all logic cells had to be analyzed at this frequency.

- The safe window for pulse transmission between adjacent logic cells depends not only on the length of the clock period, but also on the setup time of the receiving cell and the transmission delay of the interconnecting line. In a typical 10 GHz system, the safe windows deemed acceptable for inter-gate pulse transmission was in the region of 50 ps to 80 ps wide. This is much more relaxed than the 20 ps restriction imposed on the clock generators.

3.5.3 YIELD RESULTS AND OPTIMIZATION PROCEDURES

For a Monte Carlo analysis, each parameter in the simulated circuit is given a tolerance on the chip-to-chip, as well as on the component-to-component deviation. The values of these deviations are estimated from the design rules as specified by Hypres [22]. The physical tolerance values for the 3 μm niobium process from Hypres are given in Table 6.1 and Table 6.2.

Every circuit was analyzed with two different sets of parameter tolerances. The first set exceeds the absolute worst case possible for each component, such as the tolerance for the smallest width for a transmission line or resistor. This set was only used as a coarse measurement of circuit reliability during optimization, since less simulation runs were needed to determine the effect on reliability after each parameter tweak. The second or standard set was estimated for a more careful layout with larger component dimensions [34], and was used to obtain the final yield figures for each circuit. These tolerance values are shown in Table 3.1.

Table 3.1: Parameter tolerances for niobium IC

Parameter	Worst case tolerance values		Standard tolerance values	
	Global 3σ variation [%]	Local 3σ variation [%]	Global 3σ variation [%]	Local 3σ variation [%]
Resistor	20	15	15	10
Inductor	10	20	10	15
Josephson junction critical current	15	15	10	10

The standard procedure for yield calculation is to connect the device to be analyzed to a virtual test bed. This test bed consists of DC-to-SFQ converters to generate input pulses, and JTLs to connect the DC-to-SFQ converters to the device inputs. All outputs of the DUT (Device Under Test) are also connected to JTLs to emulate dynamic load conditions. A schematic illustration of the standard test setup is shown in Fig. 3.42.

The advantage of a standard test bed is that all devices can be optimized for the same load conditions on both inputs and outputs. This increases the chance that two different RSFQ devices will function correctly when they are connected to each other. In the event that two devices that were optimized for standard JTL loads fail to function correctly when they are connected, a JTL can be inserted between them. This provides both devices with the load that they were optimized for (input or output,) and has been used with great effect in this project.

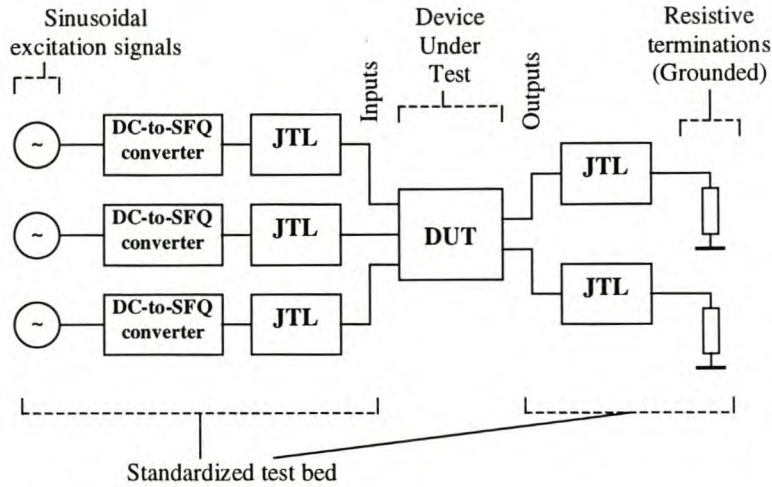


Figure 3.42: Test setup for simulating RSFQ devices

Another advantage of JTL load matching in the Monte Carlo test bed is that the JTL is by far the most frequently used device in RSFQ systems where large inter-gate distances are involved. Due to inductance problems during layout, as discussed in Sec. 6.2.1, most gates are connected through transmission lines constructed from JTLs. It must be evident that JTL load matching makes good sense, since this is the load that any device is most likely to see.

Although the test bed architecture is standard for all the components analyzed on it, it is also programmed with Gaussian parameter deviations on all constituent components. This means that even the standard JTL loads connected to the inputs and outputs of any DUT are subject to parameter spreads. It is believed that this technique gives a more accurate yield prediction than that of a test bed consisting of nominal components.

Another technique used during Monte Carlo simulations is voltage trimming. The justification for voltage trimming in the Monte Carlo simulations is that it is done in physical testing. Since the global variation on resistors affect all bias resistors equally, this effect can be nullified by adjusting the bias voltages. These voltages are set by adjusting current limiting resistors off-chip during testing. This means that varying these voltages until the circuit works is a matter of tuning a variable resistor for each dc supply.

In simulations on RSFQ circuits, trimming is modeled by multiplying all the dc bias voltages by the global resistance offset value. The inclusion of trimming in Monte Carlo simulations resulted in a discernible rise in circuit yields.

The trim model can be made more accurate by including the global junction current density offset in the dc bias value, although this was not used for determining the theoretical yield results in Table 3.2.

Trimming is also used in COSL circuits [13], where the rise in circuit yield after the inclusion of trimming is significant. The important difference between the two techniques is that for COSL, the dc bias voltages are left constant, while a dc voltage can be applied over a $50\ \Omega$ resistor. This voltage causes a controlled bias current to be injected into the input junction of the COSL gate. Where COSL gates were used in this project, the WRSpice trimming model of F.J. Rabie [19] was used.

During this project, trimming was also used in all simulations containing DC-to-SFQ converters, where the sinusoidal input voltage (usually 10 GHz) was

amplitude-adjusted to counter the effect of global variations in resistivity. Again this is easily reproducible during physical testing.

Although all the RSFQ building blocks were put through Monte Carlo analyses, and parameter tweaking applied where necessary to optimize the yield, only the procedure for the AND-gate will be discussed in some detail. This procedure represents the standard approach to device optimization that was used throughout the project.

The final realization of the RSFQ AND-gate (Fig. 3.24) started with a low theoretical yield of about 80 %, although this was much higher than that of the previous attempts. It was felt that parameter tweaking could push the yield figures above 95 %, which would make the circuit acceptable.

The optimization was started by targeting specific Josephson junctions one at a time. These junctions would be edged nearer to either “starvation” or “overdriving” by adjusting the value of I_C , the size of the closest bias resistor, or both. More visible improvements resulted from the varying of inductor values. These values would be lowered in order to prevent chance trapping of flux, or increased to lower the proportion of bias current flowing through certain junctions.

Although several adjustments were made, the most prominent changes were the lowering of L_2 and L_{35} from 3.17 pH to 2.17 pH, L_4 and L_{32} from 2 pH to a mere 0.1 pH, and the increase of L_{23} from 0.4 pH to 0.6 pH. The value of I_C for junction B_8 was also lowered from 700 μA to 600 μA . These adjustments increased the theoretical yield of the RSFQ AND-gate to about 90 %.

The next step in optimization would be to use matching JTLs at the input and output connections, so that they have similar current capabilities as the input and output junctions. However, the input junction of the AND-gate was already matched to the standard 250 μA JTL, and so was the output through the matching JTL (B_{11} and B_{12}) built into the gate. Since external matching would not increase the theoretical yield, it was still stuck at around 90 %.

The final, and most productive parameter tweak resulting from the Monte Carlo analysis procedure was the lowering of the values of inductors L_5 and L_{31} in Fig. 3.24 from 1.58 pH to 0.58 pH. This was the result of assiduous analysis and post-processing of several Monte Carlo runs on the semi-optimized AND-gate and a few shift register configurations. All the circuits with high failure rates contained DROs, and a close study of the voltages at all the nodes in these circuits revealed serious voltage pulse deformations at the reset inputs of the DROs.

The voltage pulses were stretched in time, and deformed to have two peaks several tens of picoseconds apart. A study of the currents showed that the reset inputs started to behave as two-junction SQUIDs when parameter spreads were done during Monte Carlo simulations. The reset pulse would then be stored in the loop formed by the input inductor and the nearest junctions until it could be released by other external events.

The obvious solution was to lower the DRO reset input inductance in steps until the best yield results were obtained in larger circuits. The best inductance value was eventually fixed at 0.58 pH, 1 pH lower than the value in the Stony Brook cell library.

This single parameter tweak led to tremendous yield increases in all the circuits where DROs are used. The most inspiring was that of the RSFQ AND-gate, which was boosted from a mere 90 % to 99.63 %.

The COSL OR- and NOR-gates were not put through Monte Carlo analyses, since these results (included in Table 3.2) are already available [13], [19]. The COSL

OR- and NOR-gates were, however, included in Monte Carlo analysis models for the pulse-to-logic state converters used in feedback paths in the analogue-to-digital converter (see Sec. 4.5 and Sec. 4.6.3.) The values in Table 3.2 are for 10 GHz operation, with trimming. The untrimmed yields are $94^{+6}_9\%$ for the COSL OR-gate and $86^{+13}_{-13}\%$ for the COSL NOR-gate [13].

The Monte Carlo yields and confidence levels are shown in Table 3.2. Since the worst case tolerance values were only used to force excessive numbers of circuit failures during optimization, only the more accurate yield results for the standard tolerance values are given.

Table 3.2: Theoretical yield results for all building blocks

Device	Monte Carlo yield and confidence level for standard tolerance values (99 % confidence level) [%]
COSL NOR-gate	≈ 100
COSL OR-gate	≈ 100
DC-to-SFQ converter	100 *
RSFQ AND-gate	$99.63^{+0.37}_{-0.48}$
RSFQ DRO-register	100
RSFQ JTL (all versions)	100 *
RSFQ NOT-gate	100
RSFQ OR-gate	100
RSFQ T1-flip-flop	100
RSFQ T-flip-flop	96.97 ± 1.35 §
RSFQ XOR-gate	99.17 ± 0.71
SFQ-to-DC converter	99.36 ± 0.63
Comparator	91.37 ± 2.21
RSFQ DRO to COSL OR-gate interface	93.3 ± 1.97 †
RSFQ DRO2-register	92.93 ± 2.02 ‡

* DC-to-SFQ converter and JTL verified with 3721 Monte Carlo runs each, as opposed to 1089 runs for all other components.

§ When the T-flip-flop is used in a cascade of three as in a divide-by-eight clock reduction circuit, the combination has a theoretical yield of $89.4 \pm 2.42\%$.

† This climbs to $95.7 \pm 1.6\%$ if the dc bias voltage of 2.6 mV is also trimmed to compensate for the global junction current density offset.

‡ Yield rises to $96.8 \pm 1.4\%$ when voltage trimming also compensates for global junction current density offset.

The yield percentage deemed acceptable varies from cell to cell, and depends on the number of times a specific cell is used in the ADC.

3.5.4 LAYOUT EXTRACTION TECHNIQUE

A Monte Carlo analysis using the parameter tolerance technique, where global and local parameter spreads are used to emulate the manufacturing inaccuracies or uncertainties in physical circuits, delivers a very good yield prediction. However,

since this technique merely specifies general parameter tolerance values, it does not take into account the way in which a circuit is laid out.

A more exact yield prediction can be obtained by extracting the layout diagram to a circuit simulation model, and calculating the parameter distribution of every component from the layout dimensions [35]

With this technique, the superconducting lines are not represented by lumped element inductors as was done in the WRS_{Spice} simulations, but by transmission lines. The main reason for this approach is that the transmission line parameters [26, pp. 213-254], [27, pp. 130-164] can be calculated for each geometry generated by the random parameter spreads.

Apart from the advantage that component values are now calculated from the physical dimensions, which is a much more accurate technique than merely varying the lumped element values, this technique also has other less obvious advantages. The first is that the ground plane is now taken into account during simulations, as the distributed capacitance between line and ground is inherent to the TX line model [35]. The second is that TX line components allow the Monte Carlo simulations to take into account the effect of finite pulse propagation times, and more importantly, pulse reflections. The adverse effects of energy “sloshing around” in line structures due to impedance mismatches, are wholly ignored in lumped element simulations.

In defense of the lumped element techniques used in this project, it has to be said that the average component dimensions are much smaller than a wavelength at 10 GHz. Over short sub-wavelength distances, the reflections and re-reflections merely add to the main pulse, and smear it slightly in time. This is even more true of the shorter component distances inside gates and registers.

Unchecked or unduly large reflections adding in the reverse direction could, however, escape backwards to the input of a gate and wreak havoc by forcing unexpected junction switchings. These reflections are taken into account when simulations are done with transmission lines, and the theoretical yield results predicted by such simulations must be more reliable than that of lumped element simulations.

In order to test the performance of the layout extraction technique, the DC-to-SFQ converter was simulated with both the lumped element and parameter extraction techniques.

Simulations in H_{Spice} on the circuit with specified global and local tolerances, and using lumped elements, showed a yield of 99 %. The one percent failure was caused by the appearance of an output pulse outside the 20 ps “window.” Note that WRS_{Spice} simulations showed the circuit to have a yield of 100 %.

The rest of the pulses were all located so that the 100 μ V levels (on rise and fall) were between 30.5 ps and 47.5 ps, with the window defined as the time between 30 ps and 50 ps. (Damped or “sharpened” RSFQ pulses are typically between 400 μ V and 450 μ V in height.) The clock signal was started at 20 ps, hence the 20 ps offset of the safe window discussed here from the one mentioned in Sec. 3.5.2.

Simulations on the layout based circuit showed a yield of 98 %, with the 100 μ V levels of all pulses located between 32 ps and 52 ps. The failures were caused by pulses that were too late to fall inside the 20 ps “window.” The simulation results for 100 Monte Carlo runs on the layout based circuit are shown in Fig. 3.43(a), and the safe window is demarcated by the two dashed vertical lines at 30 ps and 50 ps. The parameter extraction simulation was generated from the meticulously laid out circuit shown in Fig. 6.17.

In order to determine the effect of careless circuit layout on circuit performance, a layout extraction simulation was performed on a slightly altered version of the DC-to-SFQ converter layout [35]. For this “carelessly laid out circuit,” certain dimensions such as resistor and inductor length or width would be varied randomly by 0.5, 1 or 1.5 μm . The simulation results for 100 Monte Carlo runs in this model are shown in Fig. 3.43(b).

From the simulation results in Fig 3.43(b) it is clear that careless layout has a degrading effect on circuit performance. The observed yield for the careless layout circuit was a mere 85 %. Of the 15 observed failures, 14 were due to pulses falling outside the arbitrarily defined 20 ps safe window (the area between the dashed vertical lines at 30 ps and 50 ps,) and 1 was accounted for by the complete absence of an output pulse. This shows that careless layout techniques can cause physical circuits to behave much worse than their lumped element circuit models.

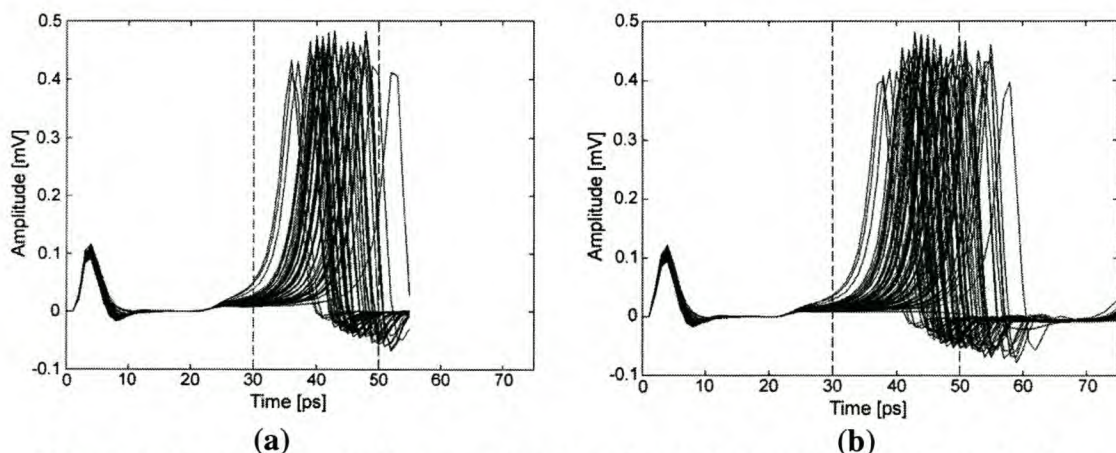


Figure 3.43: Results for layout extraction simulation models of (a) meticulous and (b) careless layout of DC-to-SFQ converter

For the DC-to-SFQ converter used in this example, the simulations show that the meticulous circuit layout is very accurate, and that the predefined parameter tolerance simulations agree well with parameter extraction simulations. The parameter extraction simulations show, however, that the circuit layout creates a circuit that is slightly slower than that predicted by parameter tolerance simulations.

As a measure of the reliability of the DC-to-SFQ converter, and the quality of the layout, these simulations delivered very satisfying results.

The layout based simulations provide a much better estimate of circuit yield, and one of the reasons for applying this technique to RSFQ was to show that these simulations may be necessary to check circuit layout even after parameter tolerance simulations have been done.

3.6 Device statistics

The physical parameters for all the building blocks, RSFQ or COSL, used in this project are given in Table 3.3. From this table, the effect of adding a logic device to a system can be determined in terms of power consumption and layout size.

The layout dimensions were taken from that of the smallest layout done for each component, although it was sometimes necessary to do layouts that consume more space.

Table 3.3: Physical parameters for all building blocks

Device	Static power consumption [μ W]	Minimum layout dimensions [μ m]	Minimum layout area [μ m ²]	Junction count	DC bias [mV]
COSL NOR-gate	0.758	101 x 169	17 069	6	5
COSL OR-gate	0.758	99 x 157	15 543	6	5
DC-to-SFQ converter	1.06	57 x 111	6 327	4	2.6
RSFQ AND-gate	5.71	144 x 193	27 792	18	2.6
RSFQ DRO-register	0.433	53 x 63	3 339	4	2.6
RSFQ JTL	0.914	32 x 39	1 248	2	2.6
RSFQ NOT-gate	2.16	86 x 115	9 890	8	2.6
RSFQ OR-gate	1.77	66 x 121	7 986	9	2.6
RSFQ T1-flip-flop	1.08	88 x 96	8 448	9	2.6
RSFQ T-flip-flop	1.54	N/A	N/A	5	\pm 2.6
RSFQ XOR-gate	0.451	86 x 96	8 256	9	2.6
SFQ-to-DC converter	3.08	109 x 161	17 549	10	2.6
RSFQ Pulse Merger	1.33	50 x 51	2 550	5	2.6
RSFQ Pulse Splitter	1.55	51 x 55	2 805	3	2.6
Comparator	1.25	96 x 322	30 912	3	5
DRO to COSL OR-gate interface	1.19	142 x 200	28 400	11	2.6 5
RSFQ DRO2-register	2.75	N/A	N/A	12	2.6

The nominal latency statistics for all RSFQ components are given in Tables 3.4 and 3.5. As these latencies may vary substantially due to manufacturing offsets, ample time needs to be left for pulse propagation when designing complex logic circuits. It is important to keep in mind that manufacturing offsets can also speed up the clock-to-output propagation time, so that a “guard band” has to be left on both sides of the nominal latency value.

Table 3.4: Nominal latency statistics (clock-to-output) for synchronous devices

Device	Latency (clock-to-output) for synchronous (clocked) devices [ps]
RSFQ AND-gate	35
RSFQ DRO-register	5.5
RSFQ NOT-gate	21
RSFQ OR-gate	5.5
RSFQ T1-flip-flop	15.5 (A to C) 9.5 (DR to S)
RSFQ XOR-gate	\approx 12
Comparator	31 - 54 *
RSFQ DRO2-register	26 - 30 §

* Faster for higher input signal.

§ Depends on the number of set (D) pulses applied before a C1 or C2 read occurs. Due to symmetry, C1-Q1 and C2-Q2 have equal latencies.

For consistency, all latency values were measured as the difference between the times that the input and output pulses rise through $100 \mu\text{V}$. Where the input signal is sinusoidal, the starting time is defined as the moment when the input signal rises above 0 V.

Table 3.5: Nominal latency statistics (input-to-output) for asynchronous devices

Device	Latency (clock-to-output) for asynchronous devices [ps]
DC-to-SFQ converter	22
RSFQ JTL	10
RSFQ T-flip-flop	9.5
SFQ-to-DC converter	N/A

3.7 Conclusions

The optimization of all logic gates, registers and cells discussed in Chapter 3 were subject to the same engineering parameters. An optimal device would be one with:

- The minimum number of circuit elements.
- The lowest dc bias currents and power dissipation.
- The lowest latency.
- The highest theoretical yield.
- The smallest physical layout area.

A smaller number of circuit elements and a smaller layout area allows the engineer to fit more logic gates onto a microchip.

Lower dc bias currents reduce the strain on off-chip dc voltage sources. Power dissipation is a square function of bias current, and a reduction in dc bias current and power dissipation also reduces the rate at which expensive liquid helium boils off from the cryogenic bath.

A decrease in latency allows clock periods to be shortened, and consequently raises the maximum clock frequency at which a specific device can operate.

Although some of these parameters can be optimized in unison, many work against each other. As an illustration, consider the reduction of the number of circuit elements. This is usually achieved by removing buffering JTLs from the input and output stages of an RSFQ device. The reduction in the number of elements leads to a smaller layout area, and also reduces the power dissipation. On the other hand, the removal of the buffering or matching elements reduces the circuit yield. Finally, latency can either increase or decrease. Although a reduction in the number of elements between an input or output and the core of a device reduces the pulse propagation time, the consequent lack of buffering and possible mismatch also increase the switching delay of the device.

Similarly, a reduction in dc bias current leads to increased latency. However, circuit yield can either increase or decrease, since it is dependent on the upper and lower current margins of every biased Josephson junction in the circuit.

As can be concluded from the above discussion, the optimization parameters result in trade-offs that have to be balanced by the design engineer. Since the importance of the different optimization parameters may vary in a complex system, the realization of a logic component could differ from one subsystem to the next.

Chapter 4 – System Design

4.1 Introduction

This chapter discusses the core of the project, namely the design of an RSFQ system that has a microwave frequency application, and can be tested with available measurement equipment.

From the above-mentioned system specifications, it is clear that a microwave application needs to be found that will utilize digital logic. Furthermore it is necessary that the RSFQ circuit, operating under cryogenic conditions, can be interfaced and measured by room temperature equipment, and that the measured outputs fall within the resolution limits of the measurement equipment.

The first stage in system design must therefore be an analysis of the demands imposed by the RSFQ system on the measurement equipment. This is discussed in Sec. 4.2.

The rest of this chapter is devoted to the selection of a system function, a discussion on digital signal processing theory surrounding the selected function, and the design of the actual system.

4.2 Problems regarding testing

The extremely high operating speeds of RSFQ logic places it well outside the measuring capabilities of standard commercial time domain equipment. The selection of a circuit function to implement with RSFQ is therefore dependent on the technique and equipment that is to be used for device testing.

The global clock frequency for the RSFQ system was chosen as 10 GHz. This selection will be discussed in more detail in Sec. 4.7. RSFQ pulses in 3 μm niobium technology, and with 1 kA/cm^2 critical current density, are typically between 7 and 10 ps wide. These pulses have amplitudes of between 200 μV and 500 μV (see Fig. 3.3(b) for a clear example.) In order to view output pulses in the time domain, oscilloscopes with a sampling frequency of 100 GHz, and amplitude resolution in the order of 100 μV are required.

The best available time domain equipment at the time when this project was undertaken, were oscilloscopes with a time resolution of 2.5 nanoseconds. These oscilloscopes could therefore not resolve an input signal of more than 400 MHz. It is clear that RSFQ logic operating at 10 GHz cannot be measured with time domain equipment of which the maximum input frequency is 400 MHz, even if the SFQ output pulses are converted to voltage state logic levels.

There were, however, frequency domain instruments available that could perform measurements on signals and circuits well into the microwave band. These instruments would eventually be used to verify the microwave signals processed by the RSFQ system, irrespective of what the function of the system would be.

Since frequency domain testing would be used to analyse the performance of analogue circuits like mixers and filters by evaluating the output signal, the question arose: Why try to measure RSFQ pulses when one can look at the frequency characteristics of a processed output signal instead?

4.3 Frequency measuring techniques and function selection

The decision to test the RSFQ circuit in the frequency domain simplified the problem of system function selection. All that was necessary was to find a system that would provide periodic microwave outputs while utilizing digital logic to perform operations on the microwave signals.

As long as the output remained a periodic microwave signal, or at least nearly periodic with relatively small deviations in the harmonic frequencies, the ability to verify circuit operation from the measured output would still hold. It would even be possible to use microwave frequency mixers to mix down periodic signals that fall outside the frequency range of the spectrum analyzers.

The main concern now was to obtain a signal processing function that utilizes digital logic, and would have increased performance for higher logic clock speeds.

Several standard functions in microwave communication systems were considered. These included digital filters, FEC encoders and decoders, spread spectrum modulators and direct digital demodulators for FM. All of these functions had one thing in common: the need for a high speed digital data input stream. The best way to generate such a data stream would be to convert it from an analogue RF or microwave signal, and this introduced the next obvious candidate for a suitable system function: an analogue-to-digital converter (ADC.)

Several architectures are available to create high-performance ADCs, including flash ADCs, half-flash, subranging or cascaded flash ADCs, successive approximation ADCs and voltage-to-frequency ADCs [36], [37]. Many of these have already been done in superconducting logic [38], [39]. Even though there are exciting possibilities in all of these analogue-to-digital conversion techniques, most of them would not need, or perform better with, high speed RSFQ logic. However, one class of ADC very popular in audio applications, the delta modulators [40], turned out to be extremely well suited to implementation in RSFQ digital logic.

Since RSFQ logic is so extremely fast, an ADC constructed with RSFQ components can be used to convert RF or microwave signals directly to digital data. Although a secondary system can process the data before reconvert the output to RF or microwave signals through a digital-to-analogue converter, this is not a necessity for frequency domain measurements. As will be explained in Chapter 5, the digital output of the ADC can itself be measured in the frequency domain, and analyzed to verify the operation of the ADC.

The delta modulating ADC was therefore chosen as the circuit function for the RSFQ system.

4.4 DSP theory and delta modulation

The advent of high speed digital computers resulted in the ever increasing need to convert analogue signals into digital words for further processing.

Conventional converters require the use of high-accuracy analogue components to provide high resolution sampling, but offer the advantage of low sampling frequencies (in the order of the Nyquist rate of the signal.)

Oversampling converters, on the other hand, allows one to use simple analogue components, while resolution in time is traded for resolution in amplitude [41]. With these converters, the number of bits in the quantizer can be reduced to one, leading to differential quantization [42, p. 758].

Delta modulation (DM) is the simplest form of differential predictive quantization. Delta modulation uses a one-bit (two-level) quantizer and a first-order predictor (see Fig. 4.1.)

The quantizer compares the analogue input signal to the previous value of the staircase approximation of the input signal (Fig. 4.3.) At every sampling instant, the quantized signal is therefore updated by one step of size Δ in the direction of the input signal (see Fig. 4.3.) A few aspects of delta modulation should now be clear:

- In the DM output bitstream, every bit represents the sign of the next step in the staircase approximation of the input signal.
- The staircase approximation cannot change by more than one step size per sampling period. This is the cause of slope overload quantization noise.
- The staircase approximation cannot remain the same from one sampling period to the next, leading to granular noise.
- The DM bitstream cannot be chopped into multi-bit words for output as a digital representation of the value of the input signal. First, the DM bitstream has to be coded to give a digital output that equals the sum of all the signed bits in the DM signal since the system was switched on. The mathematical function is called integration.
- The staircase approximation used for feedback equals the value represented by the multi-bit digital word found by integration of the DM bitstream. There will thus be an integrator in the feedback path too.

The multi-bit digital output is found right after the integrator in Fig. 4.1. The low-pass filter is a decoder, and it reconverts the digitized signal to an analogue output.

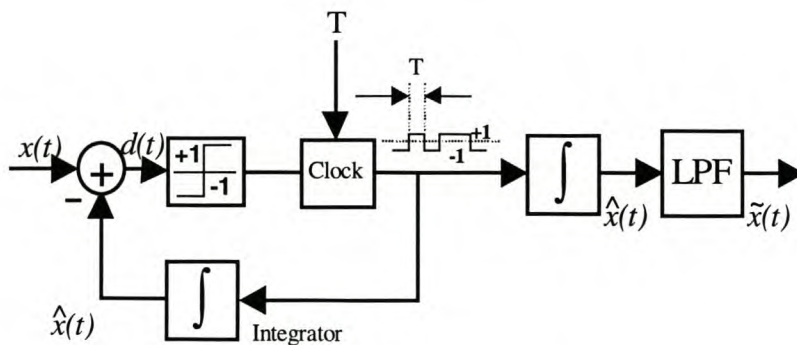


Figure 4.1: Delta modulation system

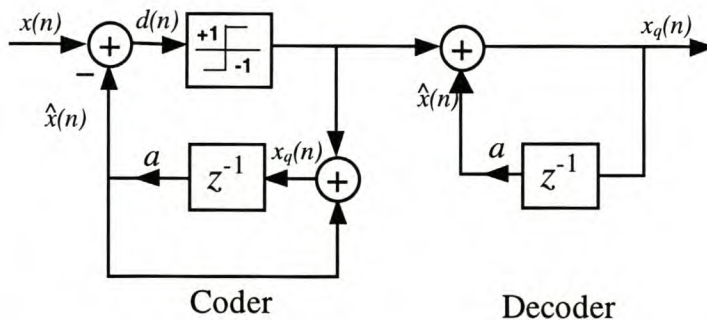


Figure 4.2: Delta modulator with discrete-time integrator

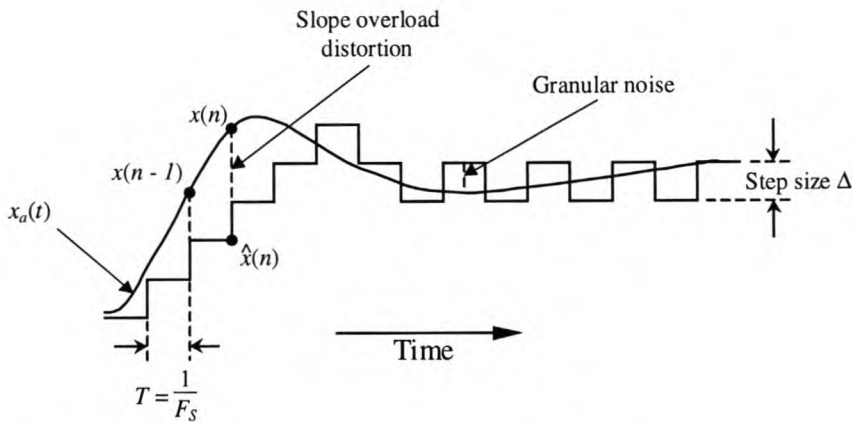


Figure 4.3: Delta modulation and two types of quantization errors

The distortion introduced by the quantization errors can be minimized by

- restricting the frequency of the analogue input signal to avoid slopes that are steep enough to cause slope overload distortion,
- decreasing the step size, and
- using a slightly different configuration known as a sigma-delta modulation (SDM) [42].

A sigma-delta modulator can be obtained by replacing both integrators in Fig. 4.1 by a single integrator just before the level detector. However, SDM requires signal processing capabilities that cannot be implemented in the restricted layout area of the RSFQ system.

Due to restrictions in the technology used to fabricate niobium-die integrated circuits [22], the use of analogue integration in the feedback path is not feasible, although it was considered initially.

Low-pass filters constructed from inductors and resistors could be used to approximate the perfect integrators, but the inductance values would be restrictive (by way of being too large,) and the feedback currents would need to be amplified. Both restrictions make analogue integration impractical, and the idea was discarded.

The integrators in Fig. 4.1 therefore have to be implemented in digital logic. Fig. 4.2 shows all the integrators replaced by their discrete-time equivalents.

The value of a in Fig. 4.2 is 1 for a perfect integrator. If a is smaller than 1, the integrators are leaky.

As has been mentioned before, RSFQ is a voltage pulse logic family. Since voltage pulses cannot be used to do level addition, the coder in Fig. 4.2 has to be ruled out.

The most practical solution is to replace the discrete-time integrator with a finite impulse response (FIR) filter (Fig. 4.4,) though this imperfect integrator introduces errors into the sampled data. Through careful design of the second stage of the converter, where the delta modulated digital bitstream is converted to a four-bit word, this error can be rectified.

The ADC is required to have a parallel multi-bit digital output in order to avoid off-chip post-processing of the digital bitstream at 10 GHz. The output word can have any number of bits corresponding to a power of two, because the only difference

between the different circuits would be the length of the shift registers (Fig. 4.4.) the number of digital full adders in the multi-bit coder and the minimum delay between successive full-adding operations. The calculation delay in serial full adders is a linear function of the number of bits in the input words.

The size of the output word was chosen as four bits, because an eight-bit system would not fit onto a niobium-die with Hypres's 3 micron etching resolution [22].

The delta modulators in Fig. 4.1 and 4.2 do not convert the modulated bitstream to multi-bit words. Consequently, a way had to be devised to convert the single-bit data stream to a four-bit word on-chip. A schematic diagram of such a system is shown in Fig. 4.4.

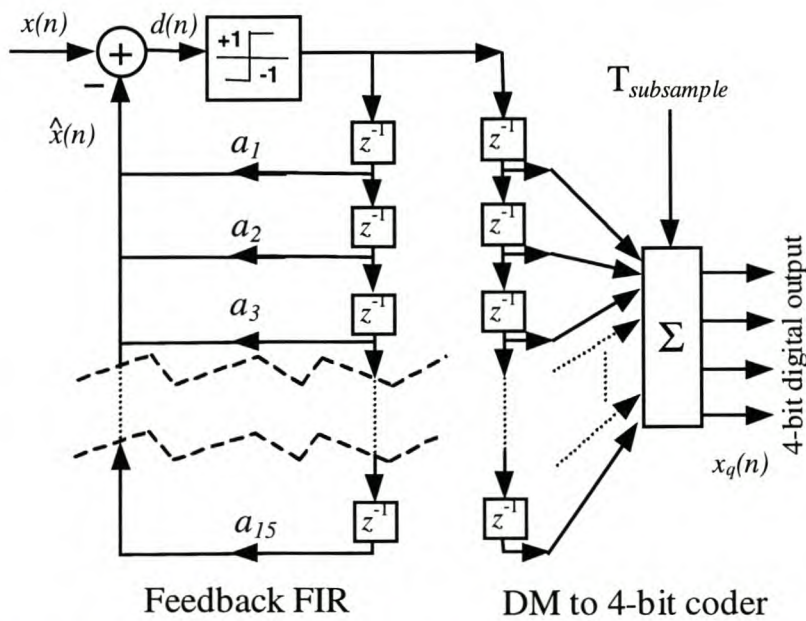


Figure 4.4: Delta modulator with FIR feedback and parallel digital output

The bitstream is clocked into a serial shift register at the primary frequency of 10 GHz. This shift register is limited to 15 cells to prevent overflow of the four-bit word during addition.

The configuration used to design minimum-gate RSFQ full adders (see Sec. 4.6.1) allows full-addition to be performed in two 10 GHz clock cycles. These serial full adders thus require 8 clock cycles to perform a complete addition operation on two four-bit words. Consequently, the maximum rate at which the contents of the shift register can be added is eighth times slower than the primary clock frequency.

The resulting subsampling frequency is 1.25 GHz, and this is the rate at which digital data appears at the parallel output.

Subsampling (or downsampling [41]) also rejects out-of-band noise by performing antialiasing filtering on the delta modulated data [40].

Although the best noise filtering characteristics can be achieved by subsampling at 625 MHz (one sixteenth of the primary clock frequency) [41], the space limitation on the integrated circuit necessitated a design that is best suited to a one eighth subsampling rate.

With the number of delay elements in the four-bit coder fixed at 15, the number of elements in the feedback FIR filter can be selected.

The shift register and full adders in the ΔM to four-bit coder (Fig. 4.4) implement an imperfect integrator that only “remembers” the last 15 bits. The error that this integrator introduces into the transfer function of the complete system can be nullified if the feedback FIR filter has the same finite memory length as the output integrator [40], hence the selection of 15 as the number of elements in the FIR filter.

Since both the feedback path and the four-bit coder utilize a fifteen-element FIR filter, the two FIR filters can be combined into a single serial shift register (Fig. 4.5) to reduce circuit complexity.

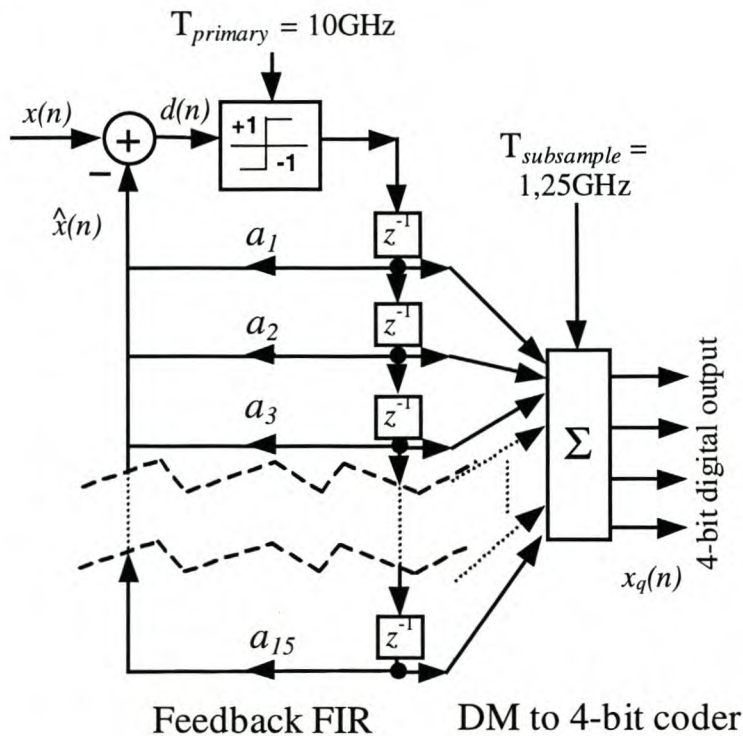


Figure 4.5: Simplified block diagram for 4-bit ADC

4.5 Conceptual design of ADC with available logic cells

With the ADC defined by the schematic diagram of the delta modulator in Fig. 4.5, system design could commence.

The discussion in this section is focused mainly on the design philosophy, and the overhead design and interaction of the subsystems. This is intended to justify the selection of subsystems, and explain the function and position of each in the ADC. A detailed discussion on the design, and schematic circuit diagrams for each of the larger subsystems, are available in Sec. 4.6.

The FIR filter is implemented as a cascade of RSFQ DROs, all clocked by phase 1 of the three-phase clock signal. The first DRO is connected to the output of the level detector, and every DRO output is branched through pulse splitters to connect it to the feedback and readout circuitry.

The delta modulating ADC requires negative feedback, so that the value of the previous output can be subtracted from the input signal before sampling.

In the first design, negative output COSL OR-gates were connected to the output of each DRO in the fifteen-cell serial shift register. These negative output

COS OR-gates were then connected resistively to the input of the level detector. If a high bit moved through a DRO, the corresponding negative output COSL OR-gate would apply a voltage of -1 mV to the resistive feedback path. This would drain current equivalent to one bit in the output from the input to the level detector.

Although open-loop simulations were successful, the closed-loop system failed. This happened because the negative feedback currents would only flow for about a quarter of a clock period, after which the current drain would cease, and the residual input clock current would cause erroneous switching of the level detector.

The solution to this problem is to use standard COSL OR-gates to provide the feedback voltages. The design change merely comes down to a shift in the feedback levels. Instead of the feedback voltage for a subtracted bit being -1 mV, it now is 0 V. Similarly, the feedback voltage for no bit changes from 0 V to $+1$ mV. Of course this also necessitates an upward shift in the level above which the level detector samples a high bit when no feedback is applied (see Fig. 3.39(b).)

Since the ADC is a 4-bit system, it can convert an analogue signal to 2^4 , or 16 quantized levels. These 16 levels are represented by the numbers 0 to 15 in the output words, and the relative difference between the highest and lowest output words is 15 quantized steps. In order that the input signal can be sampled for nearly equal negative and positive amplitudes, the zero input level has to be digitized as 7 or 8. This level is determined by the size of the clock current limiting resistor in the level detector (R_1 in Fig. 3.38,) and the value of the dc bias adjust voltage applied to the level detector. The former value is fixed on-chip, while the latter is adjustable.

The characteristics of the delta modulator cause the output to flip between two contiguous values for a constant input signal, and therefore the zero input level was eventually taken as somewhere between 7 and 8, but closer to 8.

For the calculation of bit size, consider the case where the analogue input amplitude is a positive maximum. Since this results in no feedback bits, the level detector should only switch if the input amplitude remains at a maximum. The level detector was designed to switch for an input current of about $100 \mu\text{A}$, and since the maximum positive output is somewhere between 7 and 8 bits higher than the zero output, one bit is represented by about $13 \mu\text{A}$.

The feedback paths are each required to feed one or zero bits back into the level detector. Therefore, if the COSL OR-gate output in a feedback cell is high, $13 \mu\text{A}$ should be injected into the level detector. The COSL OR-gates are matched to 5Ω outputs, and generate output voltages of about 1 mV. By connecting the output of a feedback COSL OR-gate through a 5.35Ω resistor to ground, and a 76.8Ω resistor to the feedback input pin in the level detector, all the design requirements are met.

The shift register was also designed to have a non-destructive readout that would not interfere with the serial shifting operation. RSFQ AND-gates are used to implement this function, with the AND-gate inputs connected to the output of a shift register cell (DRO) and a global read-activate input.

The 15 bits generated by the shift register readout has must added together to create a 4-bit word. For the first design iteration, one circuit was designed to calculate the four parallel bits in the 4-bit output word simultaneously. Logic functions were created from truth tables for all the possible input combinations, but even after minimizing these functions required too many logic gates.

The next step in the evolution of the adding circuitry design was to use 5 standard three-input full adders to sum the 15 single-bit inputs to 5 two-bit words. These words would then be added by a succession of multi-bit input-word full adders. The minimized basic design (excluding pulse splitters, JTLs and DC-to-SFQ

converters) of one full adder with dual two-bit inputs and a three-bit output required 19 basic logic gates, compared to the 5 gates for the basic three-input single-bit full adder (Fig. 4.9.) The total number of gates for this design escalated beyond reasonable limits, and the design was abandoned.

The shift register is read after every 8 clock cycles, and initially all designs for the adding structure were focused on calculating the 4-bit output within 8 clock cycles. However, since data ripples through the adders at one gate per clock cycle, a calculation does not need to be finished by the time a new set of data is loaded into the adder. With this new design philosophy, an efficient full adding circuit was developed.

The serial full adder [43, p. 630] used in the final design is shown in Fig. 4.6. With these full adders, two binary words of any length can be added together by feeding both into the adder at a rate of one bit per clock cycle, and starting with the LSB of both words. The sum is shifted out serially, starting at the LSB, while the carry output is fed back to the input through a one-clock delay mechanism.

The delay is only needed for typical semiconducting voltage state logic systems where the absence thereof will cause C_{out} to propagate to C_{in} during the same clock cycle in which it is calculated. In RSFQ logic, data can only pass through one gate per applied clock pulse, thereby rendering the delay element redundant.

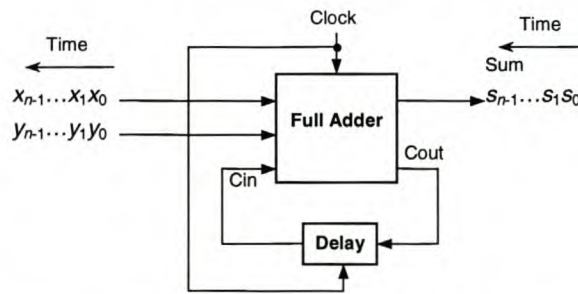


Figure 4.6: Block diagram of serial adder

The full circuit block diagram for calculating the four-bit output word is shown in Fig. 4.7. The system uses 5 three-input full adders to convert the 15 single-bit inputs to 5 two-bit words. The S-output (sum) of each full adder is clocked out at the second clock after the inputs were loaded, while the C-output takes three clock cycles to compute (see Sec. 4.6.1.) The C-output of each three-input full adder is connected to the next adder stage through one more delay than the corresponding S-output. Along with the internal delay of one clock cycle between S and C, this adds up to two delays, giving the S-bit time to pass through the next addition stage before the C-bit is fed in.

The first bit of the four-bit output word is read out by the eighth clock after the 15 data bits enter the system. This clock also reads in a new set of data from the shift register, and calculation of the next output word is started while the previous calculation is still in progress. Since the output bits emerge from the S-output of the final full adder, they are all spaced two clock cycles apart, with the spacer bits always represented by the absence of an SFQ pulse. The data and spacer bits for one four-bit word therefore occupy eight clock periods, and consecutive output words follow one another with only one spacer bit between the MSB of the previous word, and the LSB of the next.

testing device would be about 12 dB more (see Fig. 5.6) than that delivered by an SFQ pulse train.

The SFQ-to-DC converters not only deliver more power into the measuring equipment, but also produce less frequency clutter than SFQ pulse trains. The primary reason for this is that the oscillation in the pseudo dc output has a frequency of about 64 GHz, around which the first mixing products appear when the output is modulated by a low frequency square wave. The mixing products are therefore far removed from the spectrum below 1 GHz where the ADC outputs are measured.

The full adders and dc output generators can be replaced by a 4-stage counter with built-in SFQ-to-DC conversion, as discussed in Sec. 4.6.4, but this option was not implemented.

4.6 Design of complex subcircuits

In the following subsections, the logic blocks are represented by the symbols defined in Fig. 4.8 in order to reduce clutter in the schematic diagrams.

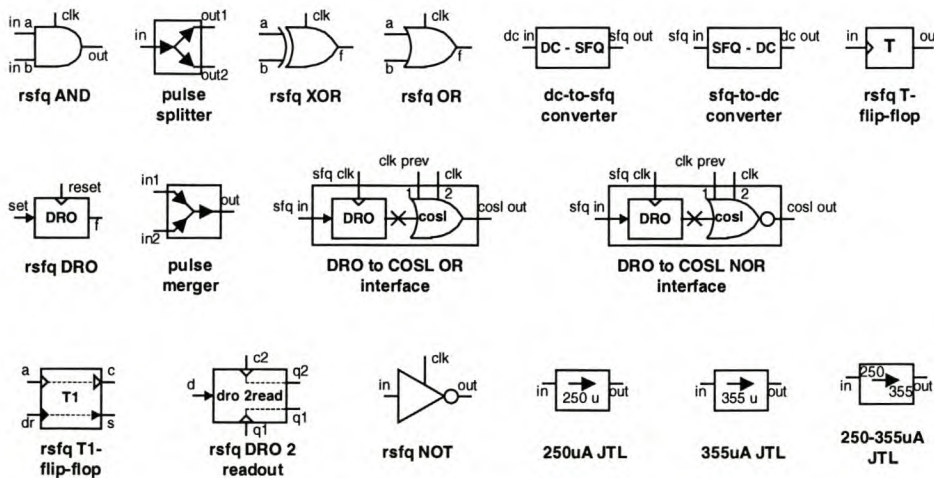


Figure 4.8: Key to schematic diagram symbols

4.6.1 FULL ADDER

By definition, a full adder adds 2 single-bit inputs to a carry input to produce a one-bit sum and one-bit carry as output [44, p. 163], [43, p. 350].

With the two inputs defined as X and Y, the carry input as Z, and the sum and carry outputs as S and C respectively, the logic truth table is as given in Table 4.1.

An RSFQ cell is proposed in [16] that would perform the full adding function. The proposed full adding cell has few elements, and would be able to complete the operation in a single clock cycle. Unfortunately this cell is unstable and suffers from extremely narrow parameter margins [16]. Previous attempts at implementing a functional version of this cell have failed [23, p. 37].

The superior alternative is to design a full adder that uses the basic high-yield logic cells already available in RSFQ.

Table 4.1: Truth table for full adder

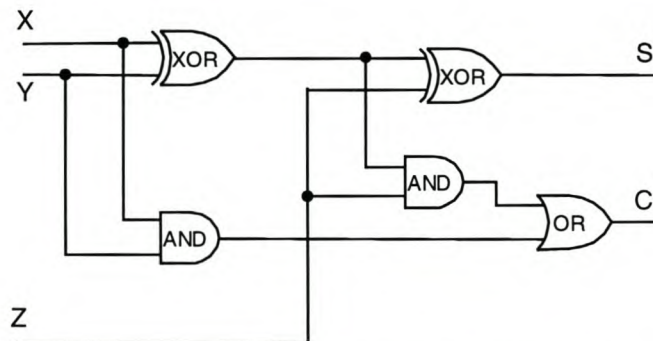
Inputs			Outputs	
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

The logic functions for S and C can be minimized to [44, p.167] (see Appendix E)

$$S = z \oplus (x \oplus y) \quad (4.1)$$

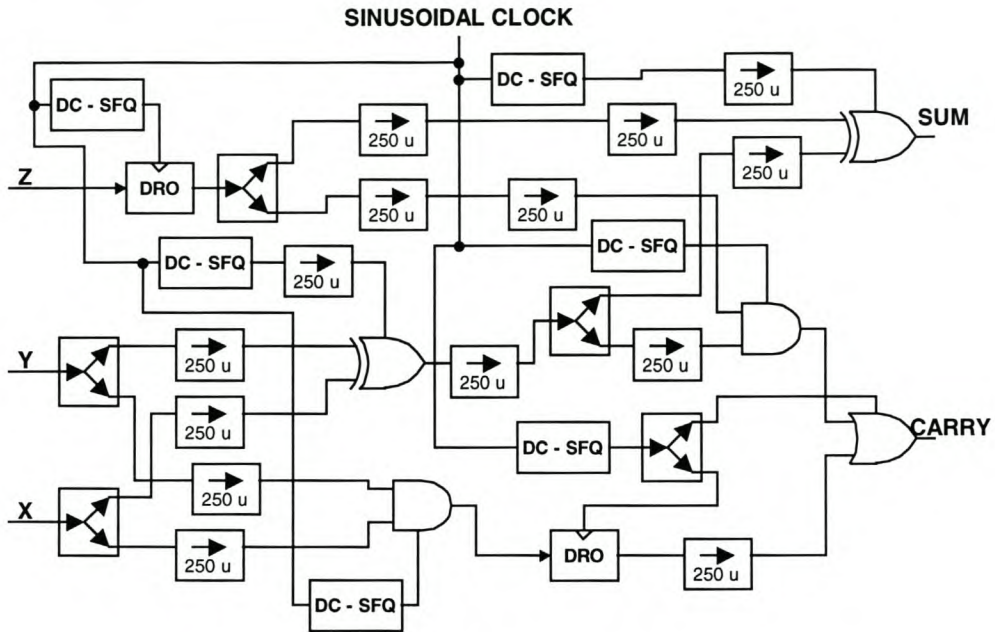
$$C = z \cdot (x \oplus y) + x \cdot y \quad (4.2)$$

The schematic diagram for a full adder based on these logic functions is shown in Fig. 4.9.

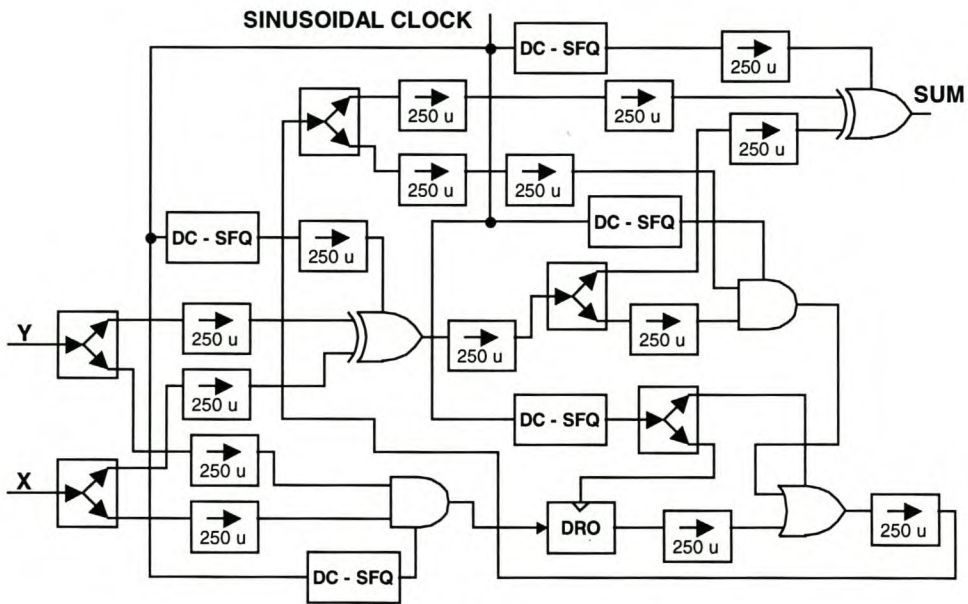
**Figure 4.9: Logic diagram of full adder**

The full adder shown in Fig. 4.9 was adapted to RSFQ logic, and the schematic diagram for the resulting circuit is shown in Fig. 4.10(a). With all the gates in the RSFQ full adder clocked by the same clock phase, the sum output takes two clock cycles to calculate, and the carry output three.

In the initial design, different clock phases were used to try and speed the full adder. This led to timing violations caused by the setup time of the RSFQ XOR-gate, and the fastest result was one and a third of a clock cycle for the sum output. The carry output was still slower, and it was decided to use one clock phase for the entire full adder. In the ADC, the full adders are clocked by phase 3 of the three-phase clock signal.



(a)



(b)

Figure 4.10: Schematic diagram of RSFQ full adder with (a) three inputs and (b) two inputs and carry feedback

The Z-input should arrive at the second XOR and AND gates during the clock cycle following the one during which inputs are applied to X, Y and Z, in order to allow for the first XOR and AND gates to be clocked out. That is the reason why the Z-input is stored in a DRO for one clock cycle. In the full adder with carry feedback (Fig. 4.10(b),) the carry output is connected directly to Z. Since the carry output takes one cycle longer to calculate than the sum output, and inputs at X and Y are applied every second clock cycle in serial full adders of the ADC, the DRO at the Z-input has to be omitted from the carry feedback circuit.

The full adder circuits shown in Fig. 4.10 were constructed by recursion. A model for the three-input full adder, containing all the essential logic gates and pulse splitters, was simulated to work, after which a circuit layout was done (Fig. 6.19.) During the layout procedure, it became evident that extra JTLs were necessary to allow SFQ pulses to be transported over the required inter-gate distances. These JTLs were added to the layout schematic, and also to the simulation model in order to ensure that the full adder would still function correctly. Although JTLs are reliable components that do not deteriorate pulses, they add delays which might lead to timing violations.

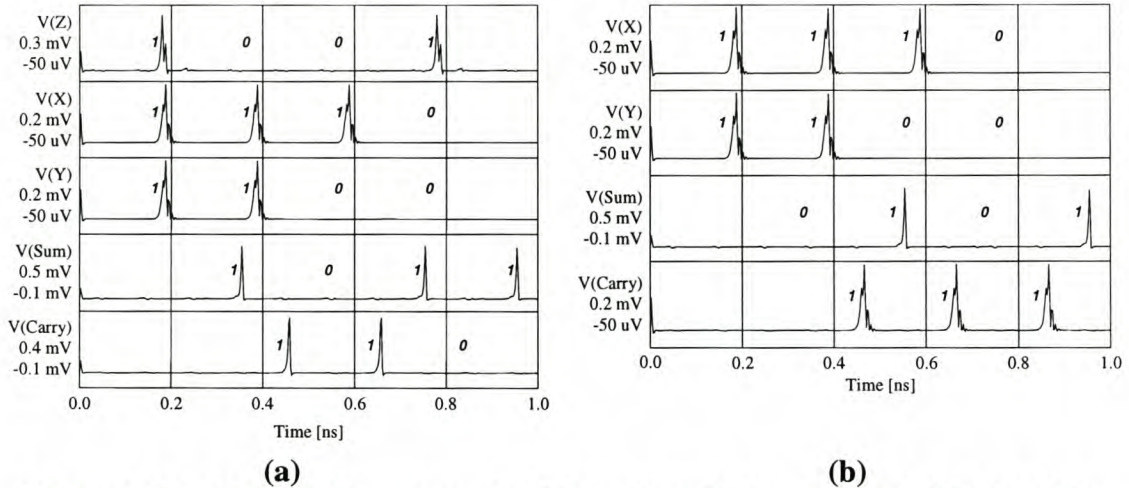


Figure 4.11: Simulated response of (a) three-input full adder and (b) full adder with carry feedback to SFQ input pulses

4.6.2 SHIFT REGISTER CELL WITH SFQ-TO-DC CONVERTER

As has been discussed in Sec. 4.5, the dc output cells are cascaded to create a shift register, and each cell has an SFQ-to-DC converter output for interfacing with off-chip signal measuring equipment.

The SFQ-to-DC converter does not have inputs that provide the ability to select “high” or “low” outputs, but merely a “toggle state” input. Since a logic “1” output should always be indicated by the same state of the SFQ-to-DC converter, the design of self-resetting circuitry around each SFQ-to-DC converter is necessary.

The four-bit result of each full adding operation consists of four SFQ pulses and their spacers (empty clock cycles resulting from the two-clock full adding operation,) and they arrive at the first dc output cell at 10 GHz (10×10^9 bits per second.) These pulses shift through the four cells in the serial array of dc output cells at 5 GHz, and the spacer bits disappear due to the telescoping effect. After every eighth clock cycle, all the dc output cells are triggered simultaneously to commence the next cycle in their 1.25 GHz data output routine.

During this output cycle, the dc output cell has to be reset to the original, or zero state, and then set to the high state if an SFQ pulse is shifting through the cell. As can be seen from the schematic diagram in Fig. 4.12, the set pulse applied to the SFQ-to-DC converter is also stored in a DRO. With the arrival of the next pulse at the eighth clock input, the state of the DRO is read into the SFQ-to-DC converter, and the latter cell is toggled only if it was set by the previous clock pulse.

In the simulation results shown in Fig. 4.13, the reset action can clearly be seen at about 1.7 ns in $V(\text{dc out})$, after which the output is set for the following high bit. Another reset occurs at 2.5 ns, followed by a low output bit.

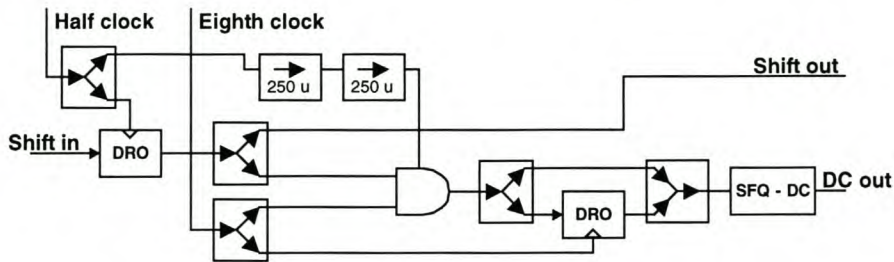


Figure 4.12: Schematic diagram of dc output cell

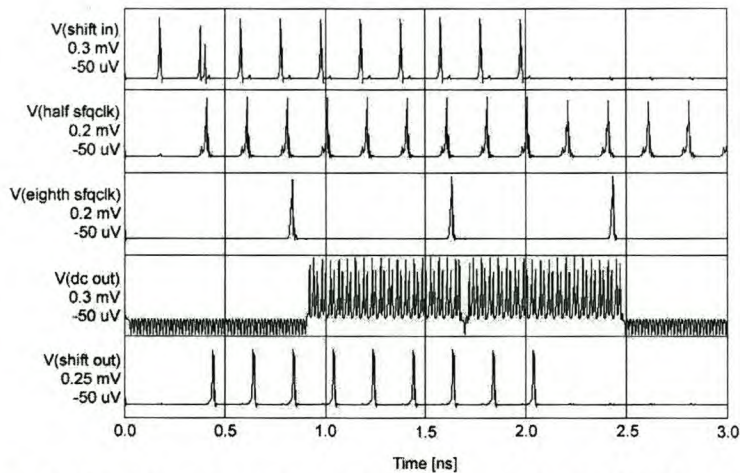


Figure 4.13: Simulated response of dc output cell to SFQ input pulses

4.6.3 SHIFT REGISTER WITH INVERTED FEEDBACK

The feedback system discussed in Sec. 4.5 requires that a COSL OR-gate has a high output when a low bit is passed through the corresponding shift register cell.

The shift register is designed as a cascade of cells. The schematic diagram for one cell is shown in Fig. 4.14. The delta modulated bitstream shifts into the DRO at a rate of one bit per cycle, and is clocked out by clock phase 1. Pulse splitters then send the bit to the inverter, the RSFQ AND-gate and the shift output to the next cell. Two thirds of a clock cycle later, clock phase 3 goes high and clocks out the RSFQ AND-gate and the inverter. The AND-gate produces a readout pulse only if it also receives an input pulse from the eighth clock input during the clock cycle, thereby implementing the times 8 subsampling function.

When clock phase 3 goes high for the next clock cycle, the DRO to COSL interface is clocked out to produce the appropriate feedback voltage.

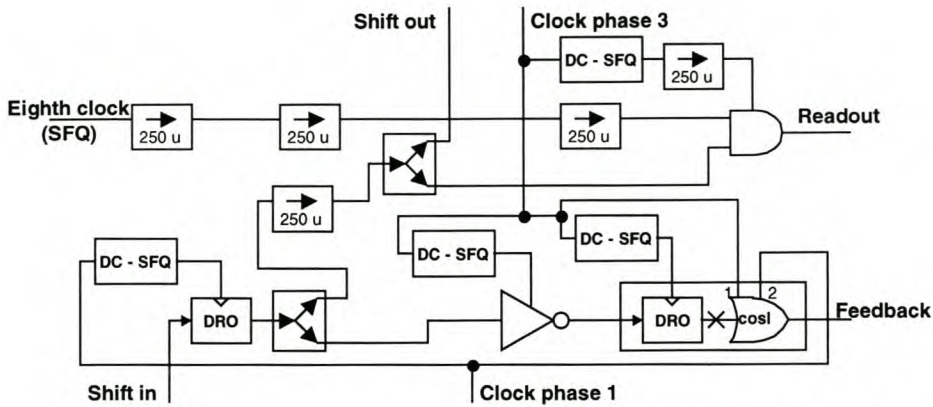


Figure 4.14: Schematic diagram of shift register cell with inverted feedback and eighth clock readout

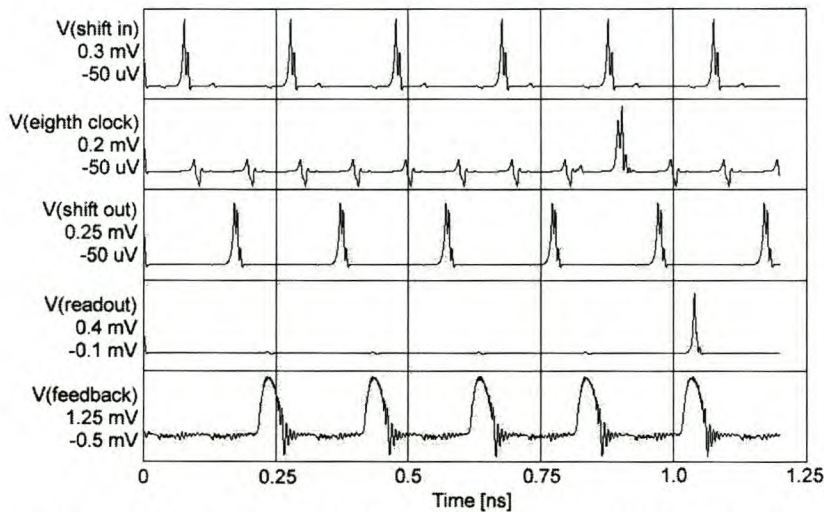


Figure 4.15: Simulated response of shift register cell with inverted feedback and eighth clock readout to input signals

The shift register cell shown in Fig. 4.14 produces a feedback voltage at the end of the second clock cycle after a bit is shifted in. For the delta modulating ADC to function correctly, the first cell in the shift register should produce a feedback voltage at the end of the cycle during which it was loaded. This requires the design of two special “fast” feedback stages at the beginning of the shift register (Fig. 4.16.)

These “fast” stages utilize COSL NOR-gates to handle inversion and SFQ-to-COSL conversion simultaneously. The COSL NOR-gates were not used for all the cells, because that would replace the RSFQ inverters. Not only was it preferable to use RSFQ logic wherever possible, but the inclusion of RSFQ inverters would also verify their correct operation if the physical ADC was tested to function correctly.

The two COSL NOR-gate feedback cells are not part of the 15-element shift register that implements the FIR filter, but they do implement two of the 15 feedback paths. The final two cells in the 15-element shift register therefore only have RSFQ AND-gate readouts, and no COSL OR-gate feedback outputs.

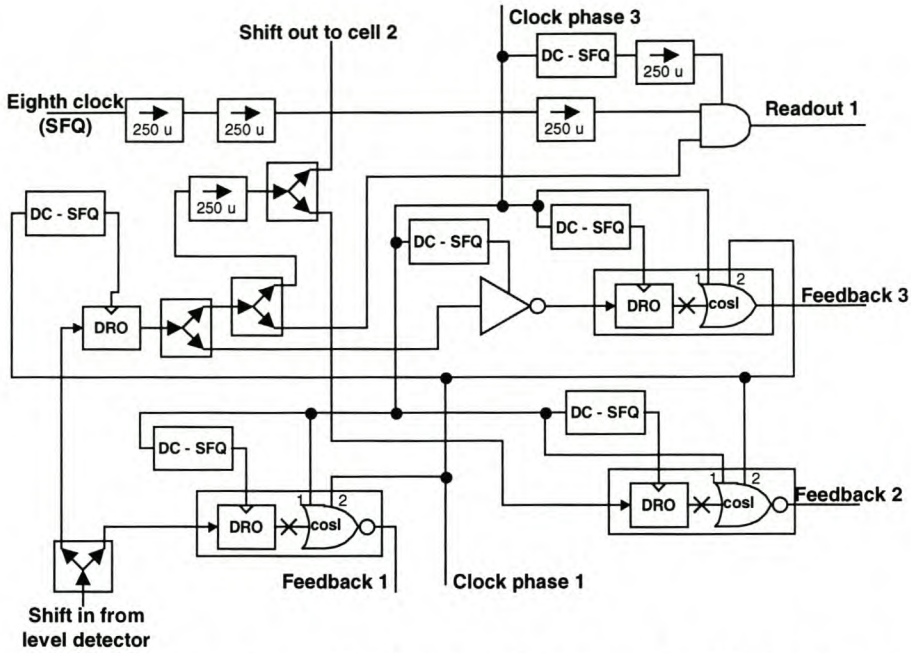


Figure 4.16: Schematic diagram of first three feedback stages

4.6.4 COUNTER WITH RESET

The use of full adders is an expensive way to sum the 15 shift register outputs to a four-bit word. The main incentive for using full adders is the chance to test a wide selection of RSFQ logic cells in a single system.

A low-cost (in terms of size and power dissipation) alternative to the 9 full adders is to use a four-stage counter to count the number of high bits passing through a certain cell in the shift register.

Such a counter would consist of four cascaded T-flip-flops, with the input to the first T-flip-flop being fed from a branched output of one of the cells in the shift register. The output of any cell in the shift register can be used. However, it is better to connect the counter to a cell that lies close to the comparator, since the chance that any cell between the level detector and the counter input may be “dead” increases with the number of cells in the chain.

One disadvantage to the counter is that it has to be clocked 16 times before a read pulse is applied, since a read action resets the counter to zero by destroying the memory. The subsampling factor thus has to be equal to 16.

The main reason for the design of the counter was that, at the time, the RSFQ AND-gate used in the full adder still suffered from a low theoretical yield. The yield for the DRO with two readouts was not only higher, but it would also be used only thrice, compared to 18 times for the RSFQ AND-gate. The yield situation was eventually reversed after the RSFQ AND-gate was optimized (Sec. 3.5.3,) and the counter was abandoned in favour of the serial full adding subcircuit. No layout was therefore done for the counter, and simulated results are not shown.

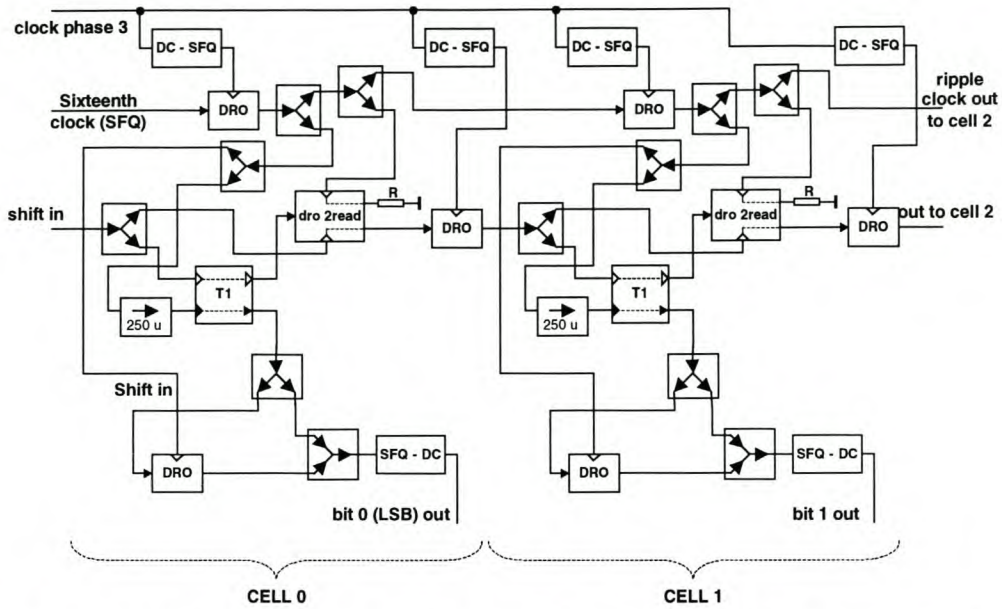


Figure 4.17: Schematic diagram showing two cells of four-stage counter

4.7 Clocking

One of the first design choices that had to be made, was the selection of the primary clock frequency.

Certain RSFQ cells can be clocked, even in 3 μm all-niobium technology, at frequencies approaching 50 GHz or more. The selection of a primary clock frequency, however, is dependent on the slowest elements in the system, as well as on other design issues such as synchronization and signal path lengths.

The highest frequency at which COSL gates have been experimentally verified to work is 18 GHz [15], although they were optimized for operation at 5-10 GHz [13], [14]. This set the upper clock frequency for the hybrid system at 10 GHz, and since all the RSFQ components can operate at higher clock frequencies, 10 GHz was chosen as the primary clock frequency.

Although the COSL gates can operate at a higher clock frequency, say 15 GHz, this frequency resulted in timing problems for the feedback paths, and was rejected.

Even though the primary clock frequency is only 10 GHz, RSFQ blocks can be clocked faster. This is made possible by the three-phase clocking scheme used by COSL gates, and by the use of individual DC-to-SFQ converters to clock each RSFQ logic gate. If successive RSFQ components are clocked by the same clock phase (through their respective DC-to-SFQ converters,) the frequency at which data ripples through the gates is 10 GHz. However, by clocking successive RSFQ gates with clock signals that are 240° (lagging) out of phase, the effective frequency of the RSFQ structure is raised to 15 GHz. This becomes 30 GHz if each consecutive RSFQ component is clocked by a signal of which the phase lags that of the previous component by 120° .

Although this technique of clock speeding was not used extensively in this project, it did find application in the feedback and readout paths of the FIR shift register.

The final design choice regarding clocking was whether a single RSFQ clock pulse should be generated and distributed through JTLs and pulse splitters to every clocked RSFQ component, or if sinusoidal clocks should be routed to every component before being converted to SFQ pulses.

The latter technique was chosen because it allowed superior clock synchronization ability, was easier to lay out on-chip, would save layout space, and would allow the use of the clock speeding technique discussed above.

Finally, the scaled-down SFQ clocks are generated from any of the primary clock phases by circuits containing cascaded T-flip-flops. The schematic diagrams of these clock division circuits are shown in Fig. 4.18.

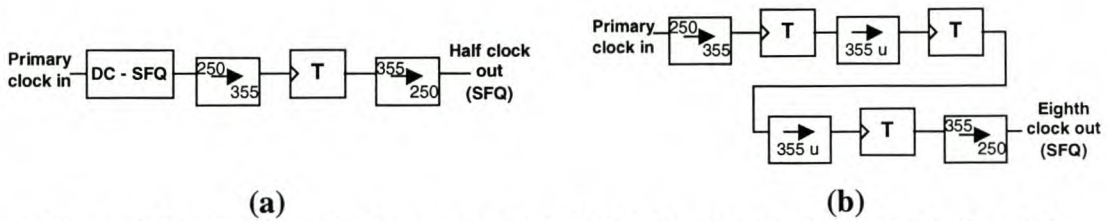


Figure 4.18: Schematic diagram of (a) divide-by-two and (b) divide-by-eight clock scalars

4.8 System yield optimization (Large scale Monte Carlo analysis)

Except for the larger scale of the simulation models, the yield optimization techniques used on the large subsystems were exactly the same as those for the individual logic components discussed in Chapter 3.

With the exception of the counter, each of the subsystems discussed in Sec. 4.6 was subjected to a Monte Carlo analysis. Just as for the individual logic components, individual virtual test beds were constructed for each subsystem, and voltage trimming was used. The test beds mimicked the exact dynamic loads connected to the inputs and outputs of each subsystem in the final ADC, as well as the timing of input signals and clock pulses.

The full adder circuits shown in Fig. 4.10(a) and (b) were initially constructed without the JTL between the output of the first XOR-gate and the input to the pulse splitter. Initial Monte Carlo simulations on these configurations showed a theoretical yield of 100 % over 121 runs, and circuit layout was done with confidence.

The Monte Carlo simulation models did not provide for all eight possible combinations of the inputs though, and when an analysis was done for the full range of inputs, the theoretical yield plummeted to a disappointing 68.6 ± 11 % for the three-input full adder, and 65.3 ± 11.25 % for the full adder with carry feedback.

The problem was identified after voltage trace analysis showed that the first XOR-gate occasionally misfires when a single input pulse is followed by the clock pulse. A study of the XOR-gate (Fig. 3.26) shows that the last junction before the output F has $I_C = 171 \mu\text{A}$. The first junction after the input for the RSFQ pulse splitter (Fig. 3.7) has $I_C = 355 \mu\text{A}$. The ratio of I_C for the two junctions is thus more than 2, and they are clearly incompatible.

A 250 μA JTL was inserted to provide load matching for the XOR-gate, and the theoretical yield for the three-input full adder immediately increased to $99.17^{+0.87}_{-2.14}$ %, while that of the full adder with carry feedback rose to approximately 100 %.

The dc output cell shown in Fig. 4.12 was constructed from optimized RSFQ logic blocks. For Monte Carlo simulations, this circuit was connected to a test setup that included the exact clock division circuitry as used in the ADC, and an input signal generated by an RSFQ XOR-gate exactly as for the sum output of a full adder. This test setup ensured that the input pulses of the Monte Carlo simulation model would be the same in size and time as those driving the first dc output cell in the full ADC.

A Monte Carlo analysis on this model predicted a theoretical yield of only 75.5 ± 11.1 % for the SFQ shift output of the dc output cell. Voltage traces were subsequently analyzed, and it was found that the shift input to the dc output cell, even though it was shaped by a JTL after being generated by the RSFQ XOR-gate in the test bed, would occasionally trap a fluxon. The input inductance of the DRO connected to the shift input in the dc output cell was consequently lowered from 2.77 pH to 1.77 pH in order to avoid inadvertent flux trapping. This adjustment raised the theoretical yield of the SFQ shift output of the dc output cell to approximately 100 %. The yield for the cell's dc output is slightly lower (see Table 4.2.)

Table 4.2: Yield results for complex subsystems

Subsystem	Monte Carlo yield and confidence level for standard tolerance values (99 % confidence level) [%]
Shift register cell with COSL feedback and RSFQ AND-gate readout	$\approx 100^*$ (shift out) $99.17^{+0.87}_{-2.14}$ (readout) 92.56 ± 6.20 (feedback)
Full adder with 3 inputs	$99.17^{+0.87}_{-2.14}$
Full adder with internal carry feedback	$\approx 100^*$
Shift register cell with SFQ-to-DC converter	$99.17^{+0.87}_{-2.14}$ (dc output) $\approx 100^*$ (SFQ shift out)
Counter cell with RESET	N/A

* No observed failures in 121 runs.

4.9 Overall system specifications

A simplified schematic diagram for the full ADC circuit is shown in Fig. 4.19. In order to reduce the complexity of the diagram, no clocking structures are shown.

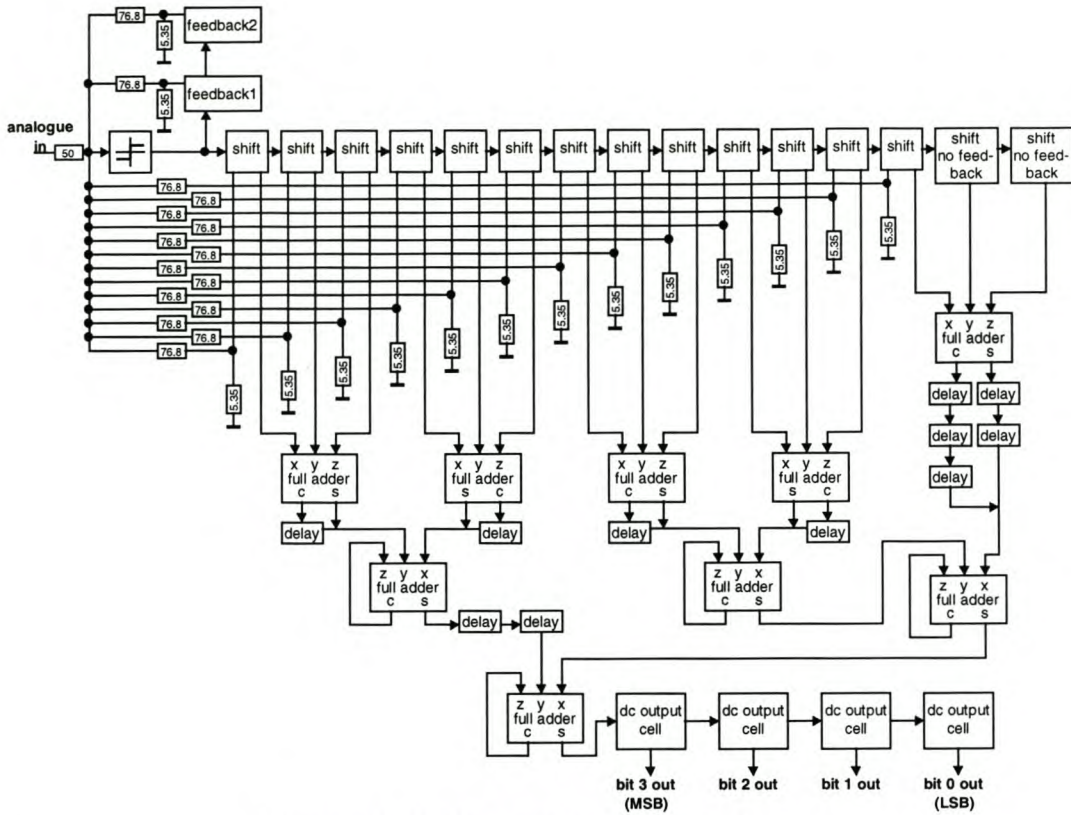


Figure 4.19: Schematic diagram of ADC

The physical statistics for all the large subcircuit cells are listed in Table 4.3. At the time of writing, the layout had not yet progressed to a point where the dc output cell could be completed, hence the absence of layout dimensions for this cell. Layout for the counter was not done, since it was not used in the final design of the ADC.

Table 4.3: Physical statistics for complex subcircuits

Subsystem	Static power consumption [μW]	Minimum layout dimensions [μm]	Minimum layout area [μm^2]	Junction count
Shift register cell with COSL feedback and RSFQ AND-gate readout	21.40	295 x 579	170 805	73
Full adder with 3 inputs	41.86	534 x 617	329 478	138
Full adder with internal carry feedback	41.28	~ 540 x 620	~ 335 000	132
Shift register cell with SFQ-to-DC converter	19.01	N/A	N/A	57
Counter cell with RESET	20.32	N/A	N/A	73

The full ADC circuit has about 2830 JJs, and a static (unlocked) power dissipation of about $870 \mu W$. The 10 mV sinusoidal clock (all three phases) dissipates another $250 \mu W$ into 140 DC-to-SFQ converters, and $205 \mu W$ into 15 COSL OR- and NOR-gates. The total power dissipation under full operation is thus 1.325 mW.

4.10 Test Bed (for future circuits)

Apart from the fact that the ADC provides a useful platform for testing the response and verifying the functionality of RSFQ logic cells, it can also be used as a test bed for other RSFQ cells and circuits.

The ADC is composed of a level detector, a shift register, RSFQ inversion logic, a COSL feedback system, and RSFQ full adding circuitry. The idea behind the test bed is that the RSFQ subsystems can be replaced by other RSFQ cells that would implement similar functions. In this way, alternative RSFQ logic blocks can be tested for functionality by comparing the measured results for the ADC with substitute cells with that of the standard ADC.

The RSFQ full adders can also be replaced by RSFQ systems that compute the same digital result, or by others that implement processing functions like filtering or demodulation.

The ability to use the ADC as a test bed for future RSFQ logic blocks is one of the most important achievements of this project. After a chip created from the ADC layout is tested to function correctly, the simulation model and layout masks of the ADC can be used as a shell to integrate new RSFQ logic blocks into. All of these systems can now be tested by simply inserting them into the structure of the ADC, measuring the results of the manufactured system, and comparing it to the results expected (from simulations) for that system. This technique will cut down on development time, especially as far as layout is concerned. It will also provide an easy way to test the new logic blocks without having to resort to expensive and complex time domain techniques.

4.11 Conclusions

A system function was needed that would allow the use of RSFQ logic components at a clock frequency of 10 GHz, while still providing output signals that could be measured with standard laboratory equipment.

The selection fell on an analogue-to-digital converter. Although extremely fast multi-bit ADCs can be created by using a flash-architecture [36], or similar fast configuration, the oversampling delta modulator was chosen because it can be constructed with the available digital logic cells.

A system was designed and constructed with the available RSFQ logic blocks, and COSL gates were used for level feedback because of their voltage state characteristics.

Optimization procedures were then applied to each of the subsystems. From the Monte Carlo analysis results for the full adder, it became clear that load matching is extremely important. Since almost all the logic blocks were optimized for connectivity to 250 μ A JTLs, care has to be taken that equivalent loads are used when these logic blocks are added to complex circuits.

It is important to check that blocks connected without using JTLs are also compatible with one another. If the output junction of a component has an I_C of $\sqrt{2}$ times smaller than that of the standard load, while the input junction of another component has an I_C of $\sqrt{2}$ larger than that of the standard load, a factor two current mismatch will occur when these two components are connected directly. In such instances it is important to provide load matching by inserting a JTL.

The input impedance of the ADC is a purely resistive 50 Ohm load. This allows easy connection to a standard 50 Ohm RF system.

The comparator and feedback current amplitude was designed as 13 μA . This produces an input sensitivity of 650 μV (-50.7 dBm) per bit, and results in a peak-to-peak input range of just under 10 mV (-27 dBm.)

Due to the inherently low noise characteristics of cryogenic systems, this sensitivity can be reduced even further. This sensitivity, coupled with an input bandwidth of 100 MHz or more and a matched 50 Ohm input, allows the digitization of RF signals without significant pre-amplification or down-mixing.

Although the quantization noise for a four-bit ADC is quite large, it can be reduced by increasing the sampling frequency, thus spreading the quantization noise over a larger frequency band [41]. This noise reduction equals 3 dB for a doubling of the sampling frequency.

Alternatively, the bandwidth of the analogue signal can be decreased, as lower frequencies will experience less additive quantization noise.

Chapter 5 – System Testing and Simulation Results

5.1 Simulation results for ADC

5.1.1 WRSPICE SIMULATIONS

For the WRSpice simulations [24], the ADC was constructed as a complete simulation model. Every logic cell used in the final design was included in the simulation model, which consisted of between 11 000 and 12 000 lumped elements. Roughly 2 800 of these were Josephson junctions.

The simulated results shown in Fig. 5.1 were generated by feeding a 110 MHz sinusoidal voltage signal with an amplitude of 3 mV peak (-37.5 dBm) into the analogue input of the ADC. All the clock signal inputs were fed by 10 GHz sinusoidal voltages, each with an amplitude of 10 mV peak, and each differing by one third of a wavelength, or 120 degrees of phase. The digital outputs were measured over the resistive terminations of the SFQ-to-DC converter outputs.

The analogue input trace was measured at the 50 Ω analogue input to the ADC, and the comparator trace was measured at the output of the level detector, just before the SFQ signal enters the fifteen-element shift register.

The serial output was measured between the last full adder and the first dc output generator, while the four output bits were measured over resistive terminations at the dc output of each cell in the dc output shift register. Bit 3 is the MSB, and bit 0 the LSB.

For clarity, bit values have been superimposed on the simulation results in Fig. 5.1. It is also clear from the simulation results that the calculation delay between input and output is about 2 - 2.5 ps.

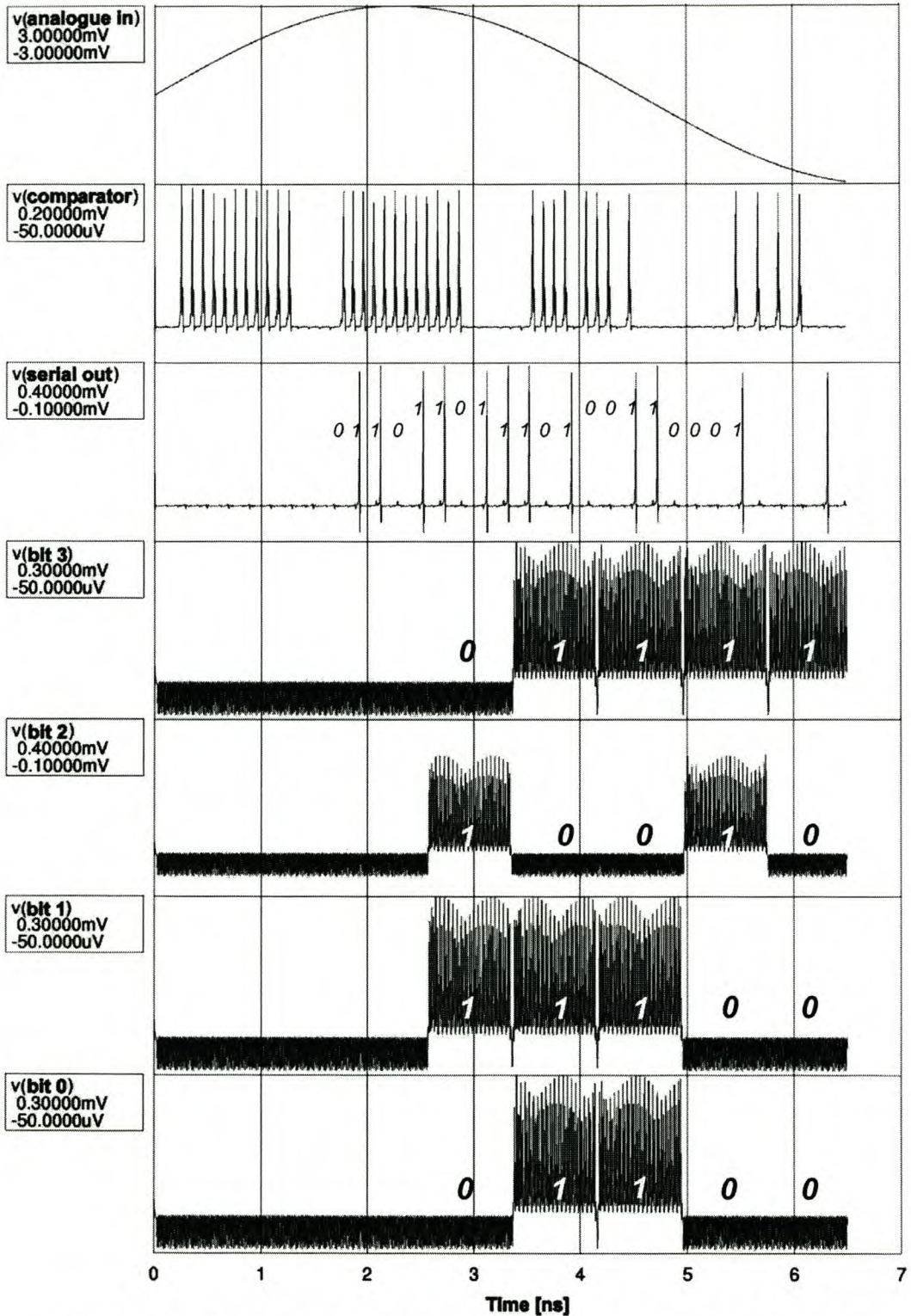


Figure 5.1: Simulation results for full electrical circuit model of ADC

5.1.2 MATLAB SIMULATIONS FOR ADC

The complexity of the full electrical simulations, the long simulation run-times and the enormous size of the generated output files necessitated the use of a reduced-

complexity simulation to check the signal processing validity and quantization noise characteristics of the ADC.

With the ADC circuit operation verified by WRSpice, the characteristics were programmed into a Matlab [45] simulation file (available in Appendix B.) The reduced-complexity simulation was run for longer simulation time spans than could be achieved by WRSpice. Not only were the Matlab simulations used to verify the validity of the analogue-to-digital conversion, but also to generate large sets of data on the ADC outputs.

The generation of thousands of data points was essential for the use of FFT algorithms to determine the spectral characteristics of the digitized signals. This could not be done from the full eleven thousand element circuit analysis, as WRSpice ran out of computing resources after generating about 6.5 ns worth of data.

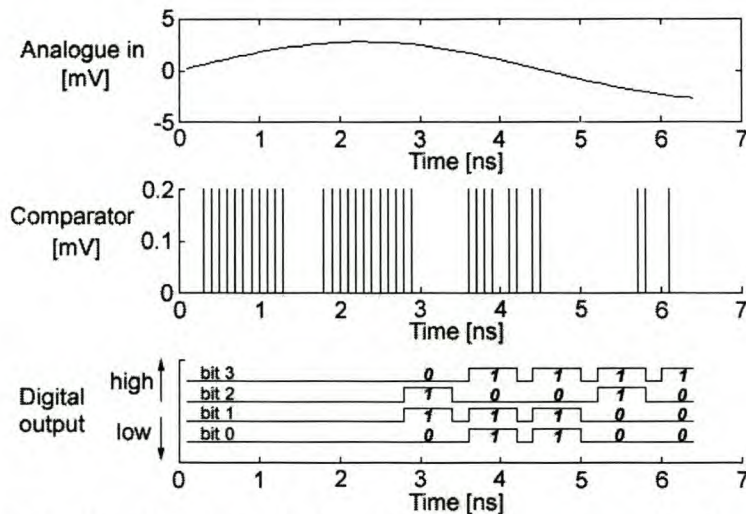


Figure 5.2: Simulated results for ADC from mathematical description

Figure 5.2 shows the Matlab simulation results for the same input signal as that in the WRSpice simulation shown in Fig. 5.1. The digital output bits of the Matlab simulation agrees exactly with that of the WRSpice simulation, because it was tuned to do so. However, the serial outputs of the two simulations start drifting apart after 4 ns because of minute differences between the two models.

The digital output for the first 100 ns retrieved from the Matlab simulation model, is shown in Fig. 5.3(a) as a reconstructed time domain signal. Although the 110 MHz sinusoidal signal is clearly visible in the digitized output, it is evident from this graph that the 4-bit ADC adds relatively large quantization noise. The power spectrum of the digitized output (Fig. 5.3(b)) shows that the 110 MHz signal stands out above the closest noise component by about 23 dB. Since the output for this signal swings through about 60 % of the available output levels, the signal to quantization noise level will increase by just more than 4 dB for a full scale output.

However, the results in Fig. 5.3 cannot be measured from the physical ADC unless equipment is available to record and reconstruct the digitized output at 1.25×10^9 samples per second. A more practical alternative is to connect the output for one of the bits in the digitized word to a spectrum analyzer, determine the spectral characteristics, and compare that to the results predicted by simulation. This technique is discussed in Sec. 5.2.2.

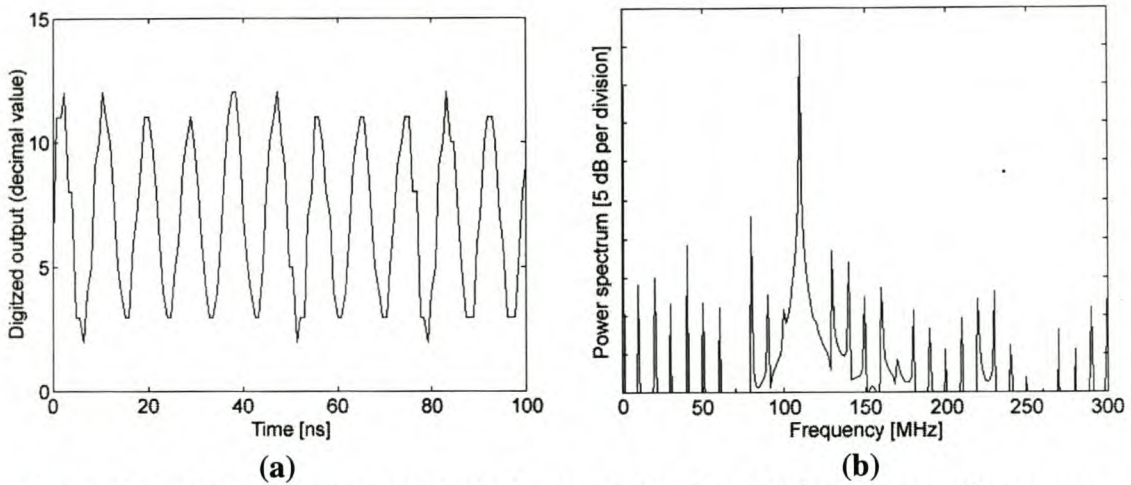


Figure 5.3: (a) Reconstructed digital output signal and (b) corresponding power spectrum for ADC simulation

5.2 Physical testing

5.2.1 TEST-BED SETUP

For physical testing, the ADC is mounted on a specialized probe, surrounded by a magnetic shield, and then immersed in a dewar filled with liquid helium.

All off-chip equipment are connected to the ADC through the probe. The dc bias voltages are generated by good voltage regulators, and applied to the ADC through resistive voltage dividers. Fine tuning of one of these resistors allows the corresponding bias voltage to be trimmed. A single microwave oscillator generates a 10 GHz sinusoidal signal (phase 1 of the three-phase clock,) and two 120° (at 10 GHz) phase shifters are used to derive the remaining phases. These clock signals are also applied through resistive voltage dividers in order to allow amplitude tuning.

The ADC outputs are measured with a spectrum analyzer. The HP 8562A available at the Department of Electrical and Electronic Engineering at the University of Stellenbosch has an upper frequency limit of 22 GHz, and is therefore ideally suited for the intended measurements. All off-chip transmission lines are impedance matched to 50 Ω , so that virtually no signal loss occurs as a result of reflections.

Although the ADC will still be manufactured, there was insufficient time to complete the layout before the completion of this thesis. Hence the absence of physical test results. However, expected measurements are still showed in the next section.

5.2.2 TESTING SIGNALS AND EXPECTED MEASUREMENTS

The physical ADC will be tested by connecting each single-bit output channel in the parallel 4-bit output word to a spectrum analyzer, and evaluating the spectral characteristics. It is therefore important to know what the power spectrum for each bit should look like.

The full electrical simulations use small time step sizes to take into account the high frequency components in the output signal. With these simulations, a time span that is sufficiently long enough to include the required low frequency signal components produce too many data points for a Matlab FFT-routine.

The solution to the above-mentioned problem is to approximate the bit outputs by average low and high voltages interrupted by low voltage bit reset periods. An example of such a trace is shown in the bottom graph in Fig. 5.2.

These approximations use ten data points for each bit in the output bitstream: nine for the bit value, and one low bit for the reset period. The value for a low bit is 0 V, and that of a high bit is $123 \mu\text{V}$. The latter value was calculated with an IsSpice4 [32] simulation.

The validity of the approximation technique is justified by the results in Fig. 5.4. The graph in Fig. 5.4(a) shows power against time for the output of an SFQ-to-DC converter toggled by a 10 GHz SFQ pulse train. The output signal is a high frequency carrier modulated by a 5 GHz square wave signal.

The power spectrum for the output signal is shown in Fig. 5.4(b). It is clear that the frequency band between 0 and 20 GHz contains only the 5 GHz modulating signal and its harmonics. The AM carrier and its sidelobes are centered around 64 GHz, and are too far removed from the baseband signal to have any significant influence. The 64 MHz carrier can therefore be omitted from Matlab simulations.

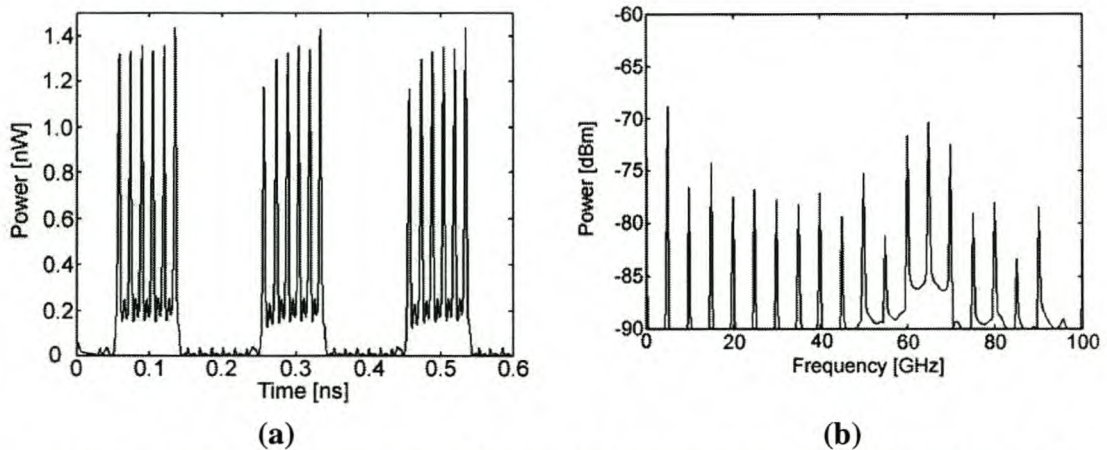


Figure 5.4: (a) Power versus time and (b) power spectrum for output of periodically flipped SFQ-to-DC converter into 50Ω load

The simulation results for the Matlab output approximation model are shown in Fig. 5.5 and Fig. 5.6. The power spectrum in Fig. 5.5 was generated from output bit 3 (MSB) for a zero input signal. The -77.5 dBm component at 1.25 GHz is the fundamental frequency of the reset period. The appearance of this component in a measured output will indicate that the SFQ-to-DC converters in the output are toggled, and that the reset circuitry is functional.

The digitized output for a zero input signal flips between 7_{10} (0111_2) and 8_{10} (1000_2), and occasionally 6_{10} (0110_2). The values for bits 2 and 3 are therefore exactly the same in time, and also exactly opposite to those for bit 3. Consequently, the power spectrum for each of these three bits are exactly equal. In the real system, this property of the ADC for a zero input signal can be used to trim the dc offset adjust voltage for the level detector.

The ADC output for a zero input flips, with slight irregularity, at nearly every output cycle. A large frequency component can therefore be observed around 625 MHz in Fig. 5.5.

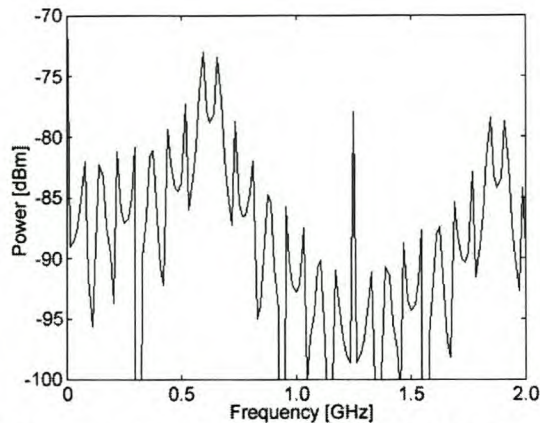


Figure 5.5: Simulated power spectrum for bit 3 of ADC output with a zero input signal

For every output word with the value 8 or higher, bit 3 will be a digital high. Every output value of 7 or lower causes bit 3 to be a digital low. A periodic input signal, such as a sinusoid, will result in bit 3 being high for one half of the period of the input signal, and low for the other half. The power spectrum of bit 3 will therefore have a fundamental harmonic at the exact frequency of the input signal. This is the most important test for the physical system, and if the test results also have the 110 MHz frequency component visible in Fig. 5.6 for a 110 MHz sinusoidal input, the ADC operation will have been verified.

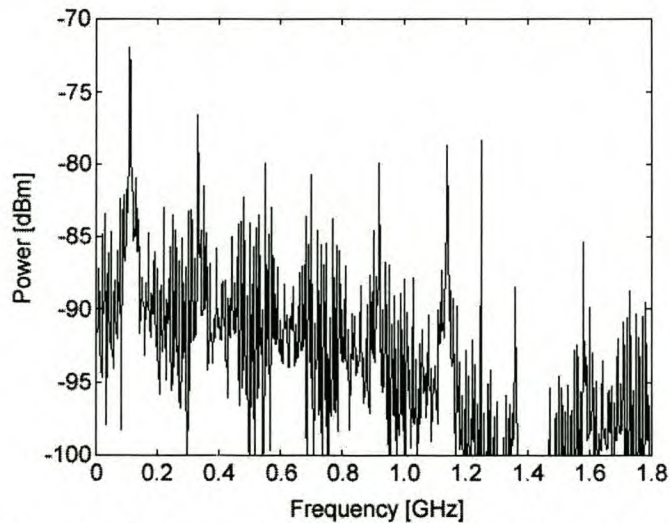


Figure 5.6: Simulated power spectrum for bit 3 of ADC output to analogue input sinusoid of 110 MHz, 3 mV

Finally, the advantage of SFQ-to-DC converters at the ADC outputs is demonstrated by the simulation results shown in Fig. 5.7. For these results, the ADC was simulated without an SFQ-to-DC converter at the MSB output, and with the same 110 MHz input signal. The output is now effectively a 1.25 GHz SFQ pulse train carrier, modulated by a 110 MHz square wave signal.

The power spectrum in Fig. 5.7(b) shows the 110 MHz component of the SFQ output to be more than 12 dB down on that of the converted output shown in Fig. 5.6.

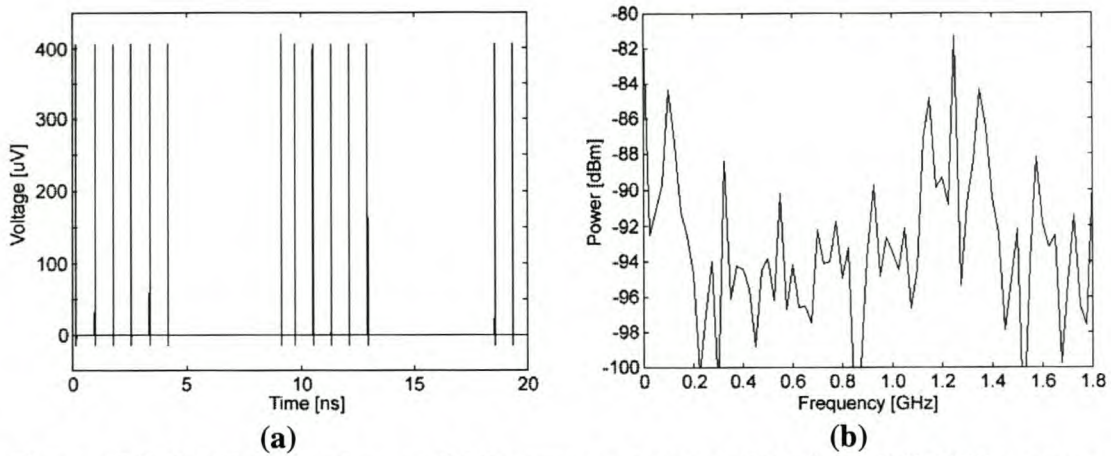


Figure 5.7: (a) Voltage-time and (b) power spectrum plots for a 110 MHz square wave constructed from a 1.25 GHz SFQ pulse train

5.3 Fault analysis and result interpretation

The aim of this project is to test RSFQ circuits, and verify the correct operation thereof. Although the layout was not manufactured by the time that this dissertation was compiled, the chip will still be made.

As was shown by Monte Carlo simulations, most logic cells have a finite chance of failure. In a large system such as the ADC, with hundreds of logic cells, it is not inconceivable that there might be one or more defunct cells in the manufactured circuit. Except for a few bottlenecks, such as the comparator, the clock division circuitry and the last full adder, most cell malfunctions will not cause the entire ADC to be “dead.”

It is expected that a few cell failures will cause the digitized output to differ from the expected output for a certain analogue input signal, but that these differences will either be too small to detect, or else can be traced back to the malfunctioning cells. In both instances, it will still be possible to verify the correct operation of the remaining RSFQ logic cells.

In order to recognize the effect of specific device failures in a faulty output, these failures have to be simulated.

As can be seen from Table 4.2, the chance that any subsystem will fail is extremely low, especially since most subsystems are only used a few times. From the cumulative binomial distribution function [46, pp. 52-53]

$$F_X(x) = \sum_{k=0}^N \binom{N}{k} p^k (1-p)^{N-k} u(x-k) \quad (5.1)$$

the probability that all 15 of the AND-gate readouts in the FIR filter will function correctly, is calculated as 88.24 %. The probability that all 9 full adders will function correctly is in excess of 92.8 %, and that for the 4-stage dc output shift register is 96.72 %.

However, the probability that all 15 feedback outputs of the FIR filter will function correctly is only 31.4 %. The chance of 1 or fewer cells being defective is 69.2 %, and this rises to 90.4 % for 2 or fewer, and 97.85 % for 3 or fewer defective cells.

Calculations on the probability of feedback errors show that 4 or more defective cells are extremely unlikely, and Matlab simulations were only done for systems with 1, 2 or 3 defective feedback cells. These simulations did not produce

clearly discernible differences in the power spectrum for bit 3 with that for the case of no failures (Fig 5.6.)

A time domain reconstruction of the digitized outputs for a circuit with 3 dead feedback cells is shown in Fig. 5.8(a), and the corresponding power spectrum in Fig. 5.8(b). These results reveal that the output dc level is shifted a little downwards, and that the reconstructed sinusoid is slightly deformed, but that it is still clearly recognizable as a 110 MHz sinusoidal signal.

The delta modulating ADC is therefore rather immune against 3 or fewer dead feedback cells, although the quantization noise will increase slightly with every dead cell. Hence it is unlikely that the effect of a few defective cells will be detectable in the ADC outputs.

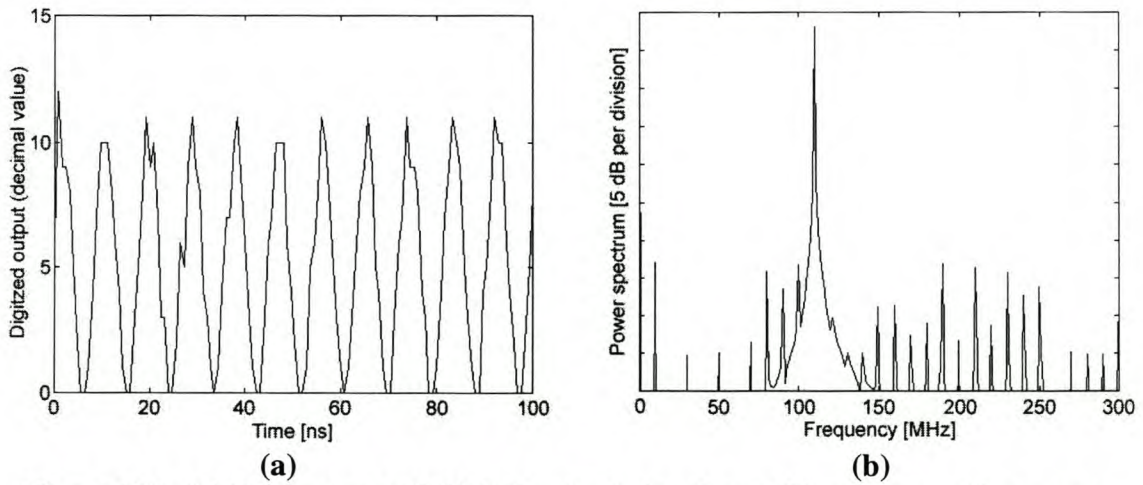


Figure 5.8: (a) Reconstructed digital output signal and (b) corresponding power spectrum for ADC simulation with 3 defective feedback cells

5.4 Conclusions

Simulation results showed that the ADC can digitize an applied analogue signal with an acceptable signal-to-quantization noise figure, and that the ADC operation can be verified by indirect measurements in the frequency domain.

The full electrical simulation also showed that the circuit will function correctly, provided that physical layout is done carefully in order to keep the values of on-chip elements as close as possible to their simulation model equivalents.

Chapter 6 – Circuit layout

6.1 Hypres layout procedure and design rules

Layout was done according to the design rules specified by Hypres [22] for the tri-layer all-niobium process. The entire circuit was laid out for the 1kA process.

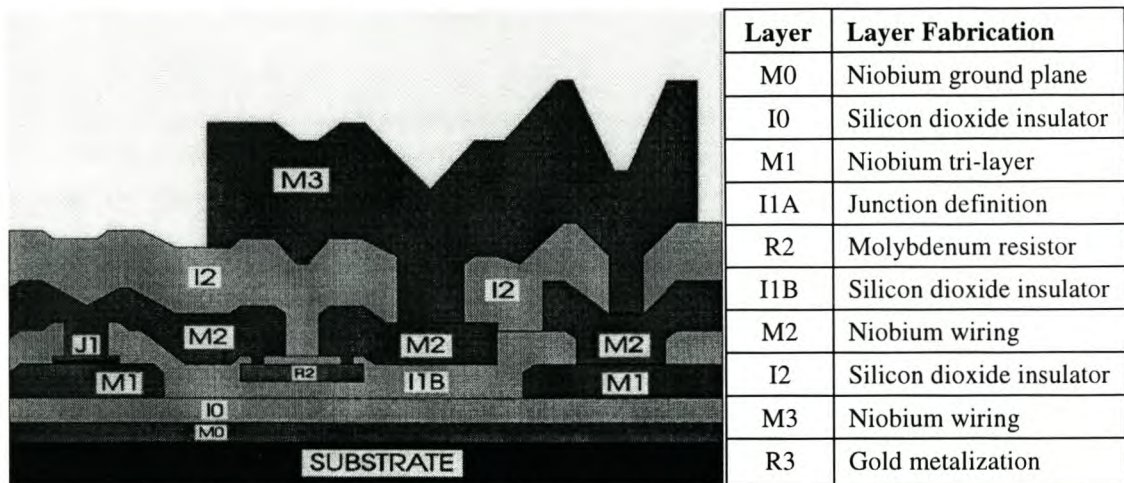


Figure 6.1: Hypres niobium process layer description

Table 6.1: Tolerance values for 3 μm niobium process from Hypres

Layer	Bias (Wafer – Mask)	Target thickness (\AA)
M ₀	$-0.50 \mu\text{m} \pm 0.25 \mu\text{m}$	1000 ± 100
I ₀	$0.0 \mu\text{m} \pm 0.25 \mu\text{m}$	1500 ± 150
M1	$-0.90 \mu\text{m} \pm 0.25 \mu\text{m}$	2000 ± 200
I1A	See Table 6.2. All values $\pm 0.25 \mu\text{m}$	–
SiO ₂ Deposition	–	1000 ± 200
R2	$0.0 \mu\text{m} \pm 0.50 \mu\text{m}$	1200 ± 200
SiO ₂ Deposition	–	1000 ± 200
I1B	$0.0 \mu\text{m} \pm 0.25 \mu\text{m}$	–
M2	$-0.50 \mu\text{m} \pm 0.25 \mu\text{m}$	3000 ± 300
I2	$0.0 \mu\text{m} \pm 0.25 \mu\text{m}$	5000 ± 500
M3	$-0.75 \mu\text{m} \pm 0.25 \mu\text{m}$	6000 ± 500
R3	$+1.0 \mu\text{m} \pm 1.0 \mu\text{m}$	12700 ± 600

Table 6.2: Bias from mask for I1A level in 3 μm niobium process from Hypres (valid for square junctions only)

Mask (μm)	3.00	3.25	3.50	3.75	4.00	4.25	4.50	4.75	5.00	5.25	5.50	5.75	6.00**
Wafer (μm)	2.60	2.90	3.20	3.50	3.80	4.10	4.35	4.65	4.90	5.15	5.45	5.75	6.00**

** All bias values for junctions greater than this are 0 μm

Particular attention was given to inductance values in the RSFQ layout, while transmission line impedance was the main concern during the layout of COSL gates and clock lines.

Where possible, inductors were laid out to be long and wide in order to reduce the effect of manufacturing tolerances on the inductance value. Square bends were almost always used when inductive connections had to change direction. A square bend was modeled as a straight line section with a length of $1/\sqrt{2}$ the side length of the square.

All clock lines were made to be nearly equal in electrical length. This was necessary to prevent clock skew due to the difference in propagation time for different lengths of transmission line. The speed with which electrical signals propagate along superconducting transmission lines is a function of the physical parameters of the line.

The impedance and propagation delay values for superconducting microstrip lines in the tri-layer niobium process from Hypres are shown in Tables 6.3 to 6.5. Only the parameters for the four impedance values used most often are listed as an example. All values were calculated with SLINE [47].

The uncertainty of each impedance value was calculated from the maximum (3σ) manufacturing offsets for the on-chip TX lines.

Table 6.3: Transmission line parameters for M1 layer

Required Impedance [Ω]	Layout width (mask definition) [μm]	Physical width (on-chip) [μm]	Propagation delay [psec/ μm]	Actual impedance and uncertainty [Ω]
28*	–	–	–	–
14	3.5	2.6 ± 0.25	1.0135×10^{-2}	13.705 ^{+8.74 %} -7.42 %
10	4.5	3.6 ± 0.25	1.0132×10^{-2}	10.385 ^{+6.43 %} -5.69 %
5	9	8.1 ± 0.25	1.0126×10^{-2}	4.995 ^{+2.96 %} -2.79 %

* Required line width smaller than minimum line width specified by Hypres.

Table 6.4: Transmission line parameters for M2 layer

Required Impedance [Ω]	Layout width (mask definition) [μm]	Physical width (on-chip) [μm]	Propagation delay [psec/ μm]	Actual impedance and uncertainty [Ω]
28	2.5	2.0 ± 0.25	8.2801×10^{-3}	26.477 ^{+9.40 %} -7.84 %
14	5.0	4.5 ± 0.25	8.2770×10^{-3}	14.440 ^{+4.69 %} -4.28 %
10	7.5	7.0 ± 0.25	8.2759×10^{-3}	9.997 ^{+3.16 %} -2.97 %
5	15.5	15.0 ± 0.25	8.2747×10^{-3}	5.071 ^{+1.56 %} -1.51 %

Table 6.5: Transmission line parameters for M3 layer

Required Impedance [Ω]	Layout width (mask definition) [μm]	Physical width (on-chip) [μm]	Propagation delay [psec/ μm]	Actual impedance and uncertainty [Ω]
28	4.5	3.75 ± 0.25	7.3281×10^{-3}	27.912 ^{+4.43 %} -4.05 %
14	10.5	9.75 ± 0.25	7.3280×10^{-3}	14.071 ^{+2.08 %} -2.00 %
10	15.5	14.75 ± 0.25	7.3280×10^{-3}	10.019 ^{+1.45 %} -1.41 %
5	33	32.25 ± 0.25	7.3280×10^{-3}	5.029 ^{+0.71 %} -0.70 %

Not only clock lines, but also all other signal lines feeding COSL inputs were made equal in electrical length. The most obvious example is the feedback lines connecting the COSL NOR- and OR-gates to the level detector. All the feedback lines were designed to be 1071.5 μm in length, as that is the length of the second longest feedback line. The longest feedback line exceeds this value by 50 μm , but because this results in only half a picosecond path difference, the second longest line length was deemed sufficient for temporal matching purposes.

All off-chip signal generation and measurement equipment are standard commercial microwave equipment. These equipment are all impedance matched to 50 Ω at the outputs and inputs. In order to minimize signal reflections and the subsequent power loss, all inputs and outputs connecting the ADC to off-chip probes and equipment also have to be matched to 50 Ω .

Although microstrip transmission lines are used for inter-gate connections, these lines become impractical when characteristic impedance values of more than 30 Ω are required. The alternative is to use a coplanar waveguide (CPW,) of which the characteristic impedance [28] can be made to be much higher than that of microstrip lines.

Very good analytical solutions [48] and computer programs [47] exist to solve inductance and characteristic impedance (Z_0) values for superconducting microstrip lines, yet no literature could be found at the time this project was undertaken to cover superconducting CPW.

Since the use of characteristic impedance calculations for CPW would simply be to determine how close the characteristic impedance of a given structure is to 50 Ω , and not to match lines exactly, certain approximations were made. Firstly it is necessary to show that these approximations will not result in serious signal losses, or alternatively that a slight mismatch between transmission lines will have a relatively small, if not negligible effect on signal amplitude.

The equation for the reflection coefficient ρ of a mismatched transmission line junction is given by [26]

$$\rho = \frac{Z_L - Z_0}{Z_L + Z_0} \quad (6.1)$$

From equation (6.1) it can easily be seen that a 10% mismatch in the characteristic impedance values of two TX lines (for instance 45 Ω and 50 Ω) will result in 5.3 % of the signal amplitude (voltage) being reflected. This leaves 94.7% of the signal amplitude (or 89.8% of the signal power) to pass into the second transmission line.

Inaccuracies in the chip manufacturing process will inevitably result in slight TX line mismatches. Since no accurate model for a superconducting CPW was available, it was decided to approximate the characteristic impedance of superconducting CPWs by using models for CPWs with a perfect electrical conductor (PEC.) Not only are very good analytical equations available to obtain Z_0 for a PEC CPW with an infinite dielectric half-plane [28] (see Appendix C,) but there are also extremely accurate numerical analyzing programs like Microwave Office [49] to solve for Z_0 . Microwave Office 2000 can obtain Z_0 for almost any dielectric depth or any non-superconducting metallic conductor.

The validity of this technique was verified by comparing the Z_0 for a superconducting microstrip TX line, calculated with SLINE [47], with that of a similar PEC microstrip TX line obtained with Microwave Office 2000. The superconducting microstrip line has a characteristic impedance of 27.3 Ω , compared to 27.5 for the PEC microstrip line. The difference is less than 1%, and from this

result it is confidently predicted that the PEC approximation for calculating Z_0 for a superconducting CPW will be adequate, especially when this difference is compared to the uncertainty for on-chip superconducting transmission lines shown in Tables 6.3 to 6.5.

Due to the size of CPW transmission lines, and the long slots etched into the ground layer, these transmission lines are kept as short as possible. One way to achieve this is to route inputs and outputs via CPW to an on-chip matching network, where the impedance is transformed to a level that suits microstrip lines. The other solution is simply to place input and output cells as close as possible to the connection pins. The latter technique was used for the ADC layout.

Several configurations of CPW are possible in the tri-layer niobium process, especially since the separation between consecutive metalization layers is much smaller than the gap and conductor widths. The use of different ground and conductor layers allows the gap width to be smaller than the minimum spacing allowed between structures on one layer. A few of these configurations are listed in Table 6.6 (see Fig. C.1 for the definition of the dimensions,) along with the Z_0 values obtained from Microwave Office 2000 [49], and the analytical estimate from equation C.1. The analytical estimates are about 10 % lower than the values calculated by Microwave Office 2000, but the former technique does not take into account the very small thickness of the transmission lines.

The uncertainty value in Table 6.6 were calculated for the maximum (3σ) on-chip manufacturing offsets in the CPW line and gap widths.

Table 6.6: Specifications for various CPW structures

Outer conductor	Inner conductor	2W + S (mask) [μm]	S (mask) [μm]	W (on-chip) [μm]	S (on-chip) [μm]	Z_0 (Software analysis) [Ω]	Z_0 (Analytical estimate) [Ω]
M0	M0	49	45	2.5	44.5	49.934 ^{+1.72 %} _{-1.80 %}	43.8
M0	M1	22	21	1.2	20.1	50.817 ^{+3.50 %} _{-3.83 %}	44.4
M0	M1	28.5	26.5	1.7	25.6	50.674 ^{+2.71 %} _{-2.88 %}	45.5
M1	M1	45	41	2.9	40.1	49.977 ^{+1.76 %} _{-1.82 %}	46.3
M1	M1	51	46	3.4	45.1	50.071 ^{+1.54 %} _{-1.59 %}	46.8

When the dimensions of a Josephson junction are calculated, care has to be taken to include the bias values (mask-to-wafer offset) for the $1\text{kA}/\text{cm}^2$ process into the calculations. Manual calculations are laborious and error-prone, and it was decided to incorporate the process into a mathematical toolbox for superconducting logic design (see Appendix A.)

The mathematical toolbox supports the calculation of rectangular and circular junctions with strict adherence to the Hypres design rules. However, it was found that eight-sided junctions can generally be made to be more accurate than the rectangular or circular junctions, while still keeping to the design rules. The eight-sided junction calculation was added to the mathematical toolbox, and used almost exclusively in the layouts for this project.

Although the eight-sided junction is very seldom a perfect octagon, the calculation routine refers to the octagonal junction.

6.2 Layout difficulties and solutions

6.2.1 MINIMUM PARASITIC INDUCTANCE BETWEEN JJ AND GROUND

One of the largest contributing factors to performance degradation between simulation models and real RSFQ circuits is the parasitic inductance values involved in damping a Josephson junction.

The problem exists for the case where a Josephson junction is connected in parallel with a resistor to ground, and also when the junction is connected in series with other circuit elements, but in parallel with a damping resistor. Fig. 6.2 shows a typical layout for the first example, where both the Josephson junction and the resistor are connected in parallel to ground. This layout is done with strict adherence to the Hypres design rules.

Although the layout in Fig. 6.2 appears to be extremely compact, a careful study of the line lengths between the junction and the resistor, and the resistor and ground reveals excessive parasitic inductance values. With the line widths used, the parasitic series inductance of the damping resistor path is in the order of 1.5 pH. This is very large, as the inter-gate inductances of RSFQ cells are about 1 pH to 4 pH.

In order to reduce connection lengths and parasitic inductances, the layouts for grounded and serial JJs were done as shown step-by-step in Fig. 6.3 and Fig. 6.4. Three-dimensional representations are shown in Fig. 6.5 and Fig. 6.6 to clarify the layout schematics. The series inductance of the damping resistor path in Fig. 6.3 is expected to be in the order of 0.45 pH.

Although the techniques used for the minimum parasitic inductance layouts are in contravention of the design rules specified by Hypres [22], it can be manufactured, and received a nod of approval from Hypres personnel [50].

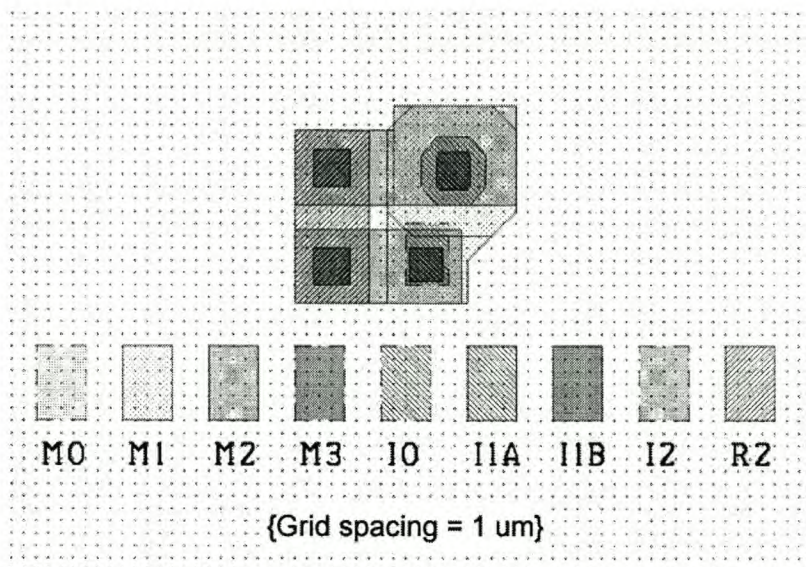


Figure 6.2: Layout for damped JJ by strict adherence to design rules

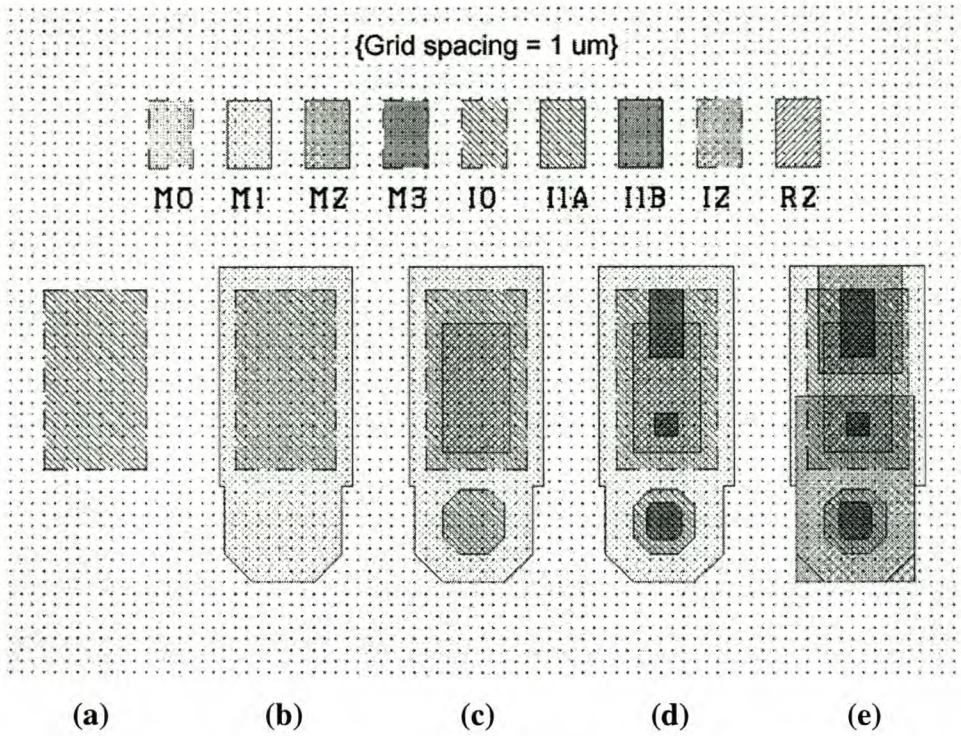


Figure 6.3: Step-by-step schematic of minimum inductance layout procedure for grounded JJ with damping resistor

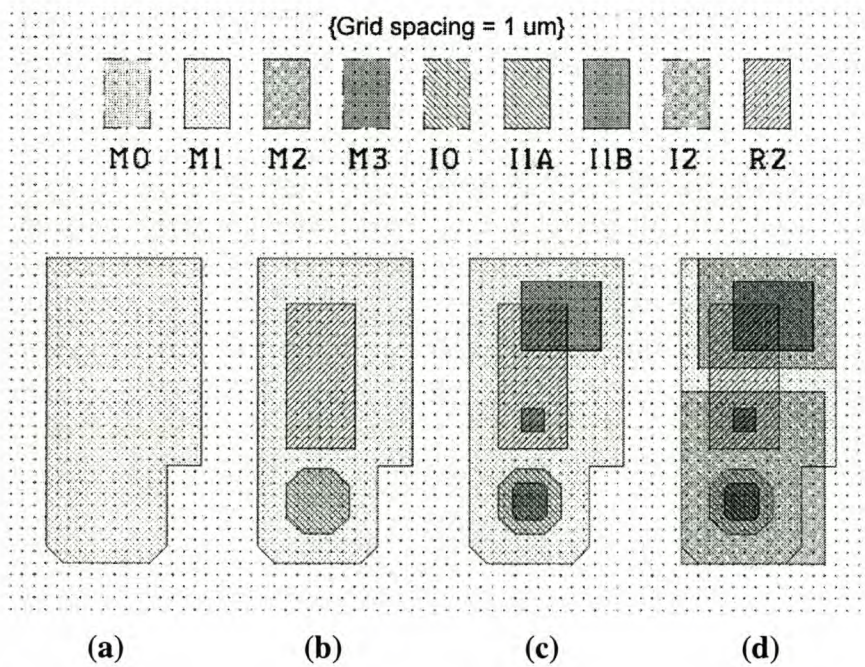


Figure 6.4: Step-by-step schematic of minimum inductance layout procedure for a serial JJ with damping resistor

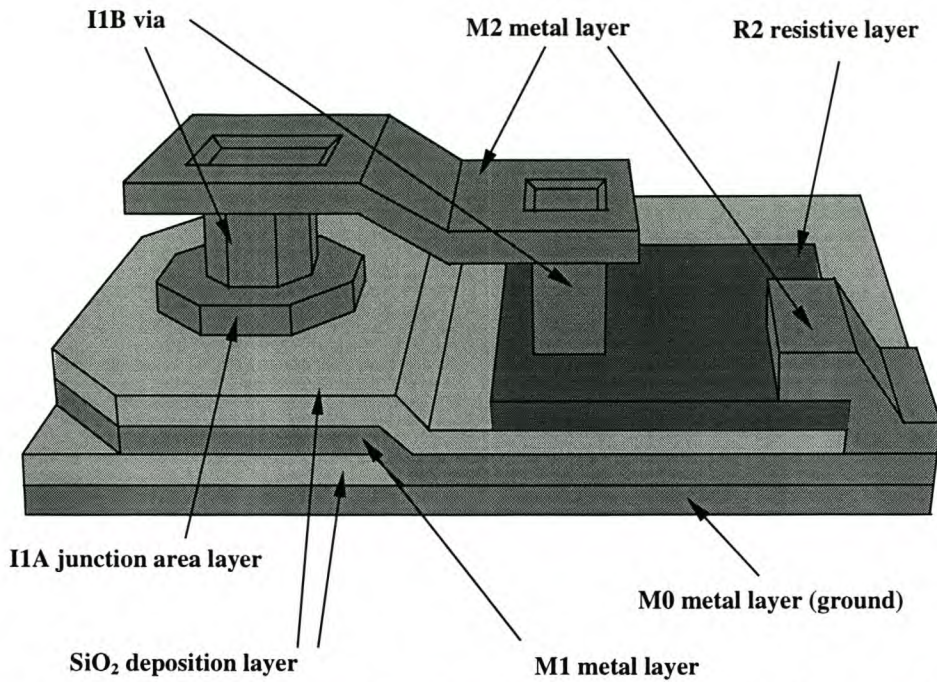


Figure 6.5: Three dimensional representation of minimum inductance layout technique for grounded JJ with damping resistor

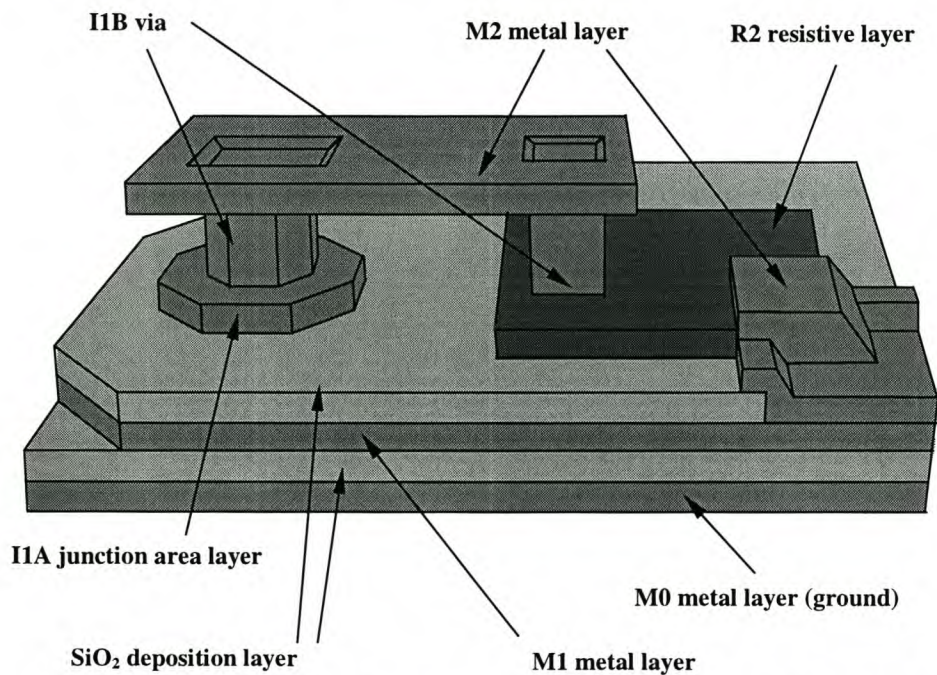


Figure 6.6: Three dimensional representation of minimum inductance layout technique for serial JJ with damping resistor

6.2.2 DC BIAS RESISTOR AND LINE LAYOUT

Another design problem frequently encountered, is how to fit dc bias resistors into a device to ensure the smallest possible layout area. Since some of these bias resistors are quite large, a lot of right angle bends are necessary. These bends increase the uncertainty in the resistance value, and are therefore undesirable.

Superconducting lines, just as any other conducting lines, have zero inductive reactance at dc. Consequently no inductance problems result from the length of a dc line. This is the reason why no matching is required for dc lines, and why these lines are always the last to be inserted into any layout. Monte Carlo analyses also showed that very large inductance values can be inserted into dc lines without having any influence on circuit operation.

These results show that a dc bias resistor does not need to be as close as possible to the RSFQ component it is feeding. An extension of this thought is that the resistor does not have to be in one piece either! During layout, resistors can thus be broken into two or more sections to fit into small open spaces, and all these sections connected by any length of superconducting line. This technique was used often during the layout of logic gates for the ADC.

6.3 Flux trapping and moats

A lot of circuit failures in superconducting electronics can be attributed to flux trapping [51]. This happens when the circuit is cooled from room temperature to a subcritical operating temperature. The magnetic flux lines “frozen” into the circuit are trapped for the duration of the superconducting state, and disrupt Josephson junction switching operations.

The Josephson junction switching is inhibited if a magnetic flux line passes through the junction. This problem can be avoided by providing alternative paths for the flux lines to penetrate through the die.

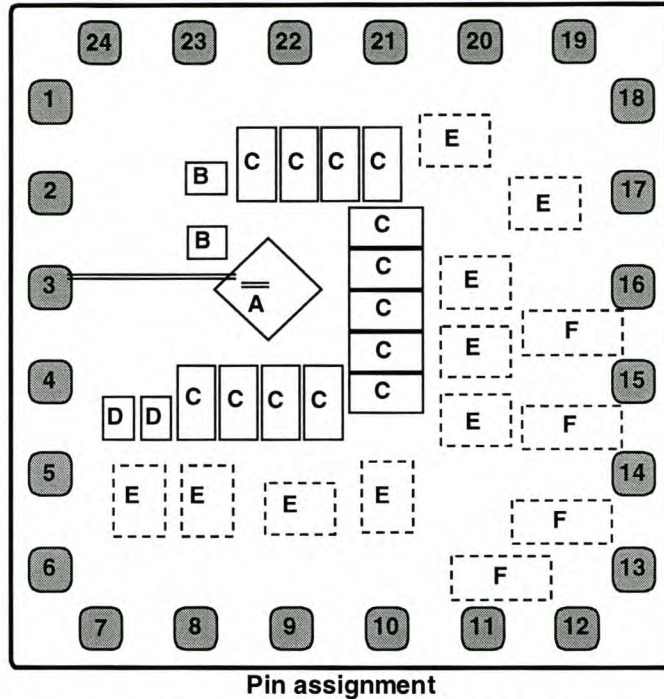
The best way of creating alternative paths is to etch moats around JJs into the M0 ground plane [51], [52]. Flux lines are then trapped in these moats when the circuit is cooled below T_C , and the Josephson junctions are protected.

The moats in individual gate layout diagrams (paragraph 6.4) may sometimes appear small and insignificant. This is merely to prevent the moats from inhibiting the interconnection of gate structures. After a larger structure is created from smaller gates, and all the interconnections sorted out, the moats are filled in wherever space is available.

6.4 Layout diagrams

The chip template used for circuit layout in the 3 μm niobium process from Hypres provides 24 pins. The designer is free to assign any function to these pins. The pin assignment for the ADC layout was done only when layout was nearing completion, and it was fairly certain which pins would be nearest to each input or output section of the circuit. A diagram showing the pin assignment and the most important circuit blocks of the ADC, is shown in Fig. 6.7.

The niobium die shown in Fig. 6.7 is exactly 5000 μm by 5000 μm in size [22] and thus has a surface area of 0.25 cm^2 .



- | | |
|---------------------------|---|
| 1 CLK phase 3 (RSFQ) | 13 Digital Out (bit 0) |
| 2 CLK phase 3 (COSL) | 14 Digital Out (bit 1) |
| 3 Analogue IN | 15 Digital Out (bit 2) |
| 4 Level trim (comparator) | 16 Digital Out (bit 3) |
| 5 +5mV DC (COSL) | 17 Test Out (1/8 th Prescaler) |
| 6 COSL Voltage trim | 18 Test In (1/8 th Prescaler) |
| 7 CLK phase 1 (COSL) | 19 Counter Out (bit 3) |
| 8 CLK phase 1 (RSFQ) | 20 Counter Out (bit 2) |
| 9 -2.6mV DC (RSFQ) | 21 Counter Out (bit 1) |
| 10 GROUND | 22 Counter Out (bit 0) |
| 11 CLK phase 2 (COSL) | 23 GROUND |
| 12 CLK phase 2 (RSFQ) | 24 +2.6mV DC (RSFQ) |

ADC Schematic layout: LEGEND

- A Level detector (comparator) and feedback resistors
- B Serial shift registers with COSL NOR-gate feedback
- C Serial shift registers with COSL OR-gate feedback and RSFQ AND-gate parallel readout
- D Serial shift registers with RSFQ AND-gate parallel readout
- E Full adders (approximate positions when wired in)
- F DC output cells (yet to be incorporated)

Figure 6.7: Schematic showing chip layout and pin assignment

Both the RSFQ DC-to-SFQ converters and the COSL devices operate on the same three-phase clock. The reason for splitting each clock phase into two inputs lies in the way in which the circuits are trimmed to compensate for global manufacturing offsets.

The COSL gates are trimmed by a dc bias voltage applied to their input JJs. The RSFQ devices, on the other hand, are trimmed by changing the value of the 2.6 mV dc bias. However, Monte Carlo simulations have shown that the yield of the DC-to-SFQ converters can be increased by also trimming the amplitude of the sinusoidal global clock signal. COSL gates are very sensitive to a variation in the amplitude of the 10 mV sinusoidal clock signal, and since they share the clock signal with the DC-

to-SFQ converters, a way had to be found to trim the RSFQ clock amplitudes independently.

The solution is to keep the COSL clock line and the DC-to-SFQ clock line separated in the chip layout, and then to connect both input pins to the same clock phase generator through different variable resistors. In this way, the COSL clock amplitude can be adjusted to 10 mV, while the RSFQ clock amplitude can be varied around 10 mV.

Should the DC-to-SFQ converters turn out to be reliable enough without the need for clock amplitude trimming, the test setup can be simplified by shorting out the RSFQ and COSL clock lines of equal phase, and using only one connection per phase to the external clock generator. Hence the placing of RSFQ and COSL clock inputs of equal phase next to one another on the microchip.

Although the second clock phases of both the RSFQ and COSL three-phase clocks are never used, connections are included in the pin assignment. This is necessary to allow the three phases of each clock to balance one another on-chip, and eliminate problems such as ground bounce.

The layout schematics for most of the RSFQ gates are shown without further explanation in Fig. 6.8 to 6.19. Several layouts may exist for each gate, depending on where it was used, where the input and output connections had to be, and what area it needed to occupy. All of these layouts are not shown. However, three layout schematics for the 250 μ A JTL are shown in Fig. 6.8 in order to demonstrate the different realizations possible for a single circuit.

Fig. 6.20 shows the completed layout for a cell of the FIR filter shift register, while Fig. 6.21 shows a completed full adder with 3 inputs. The final ADC layout up to where work has progressed is shown in Fig. 6.22. This diagram is inserted to show the relative size of the subsystem blocks to the active layout area. In the ADC layout, the entire FIR filter is already completed, as well as the layouts of the feedback paths and level detector. The readout clock distribution circuitry to the shift register cells is also completed. The 9 full adder subsystem blocks are already placed in their approximate final positions.

In order to complete the ADC layout, the full adders still need to be connected, the dc output cells have to be laid out and added to the ADC, and the clock division circuits have to be added. The final layout task will be to connect the three-phase clock lines, and then the dc bias lines.

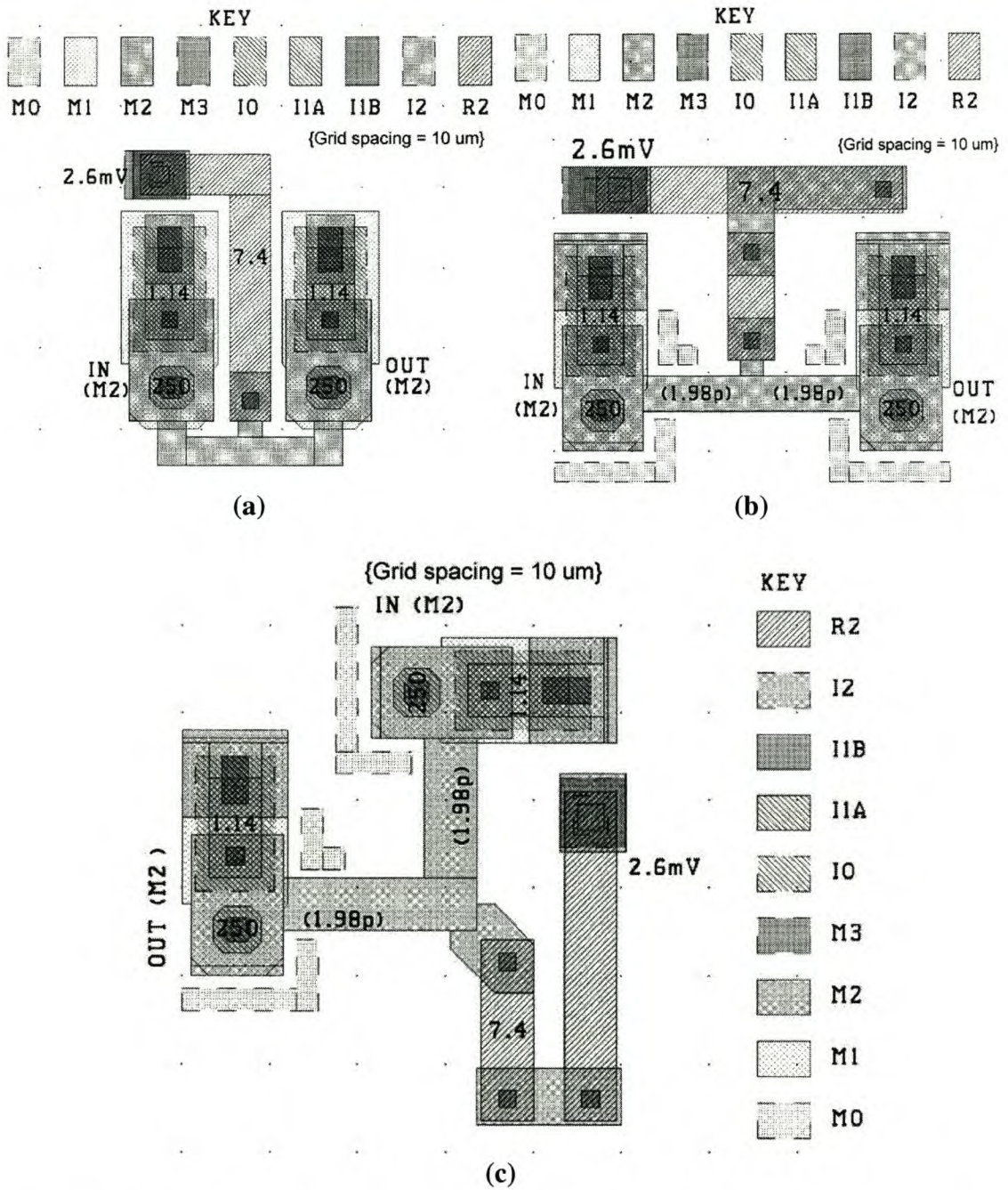


Figure 6.8: Layout schematics for (a) smallest 250 μA JTL, (b) flattened 250 μA JTL and (c) right-angled 250 μA JTL

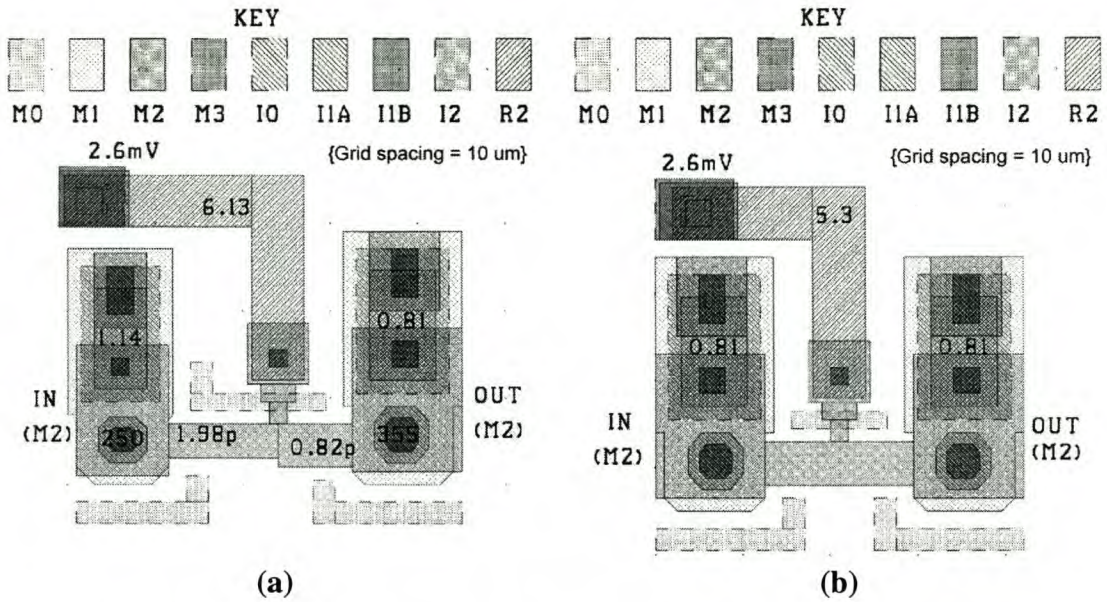


Figure 6.9: Layout schematics for (a) 250 μA -in 355 μA -out JTL and (b) 355 μA JTL

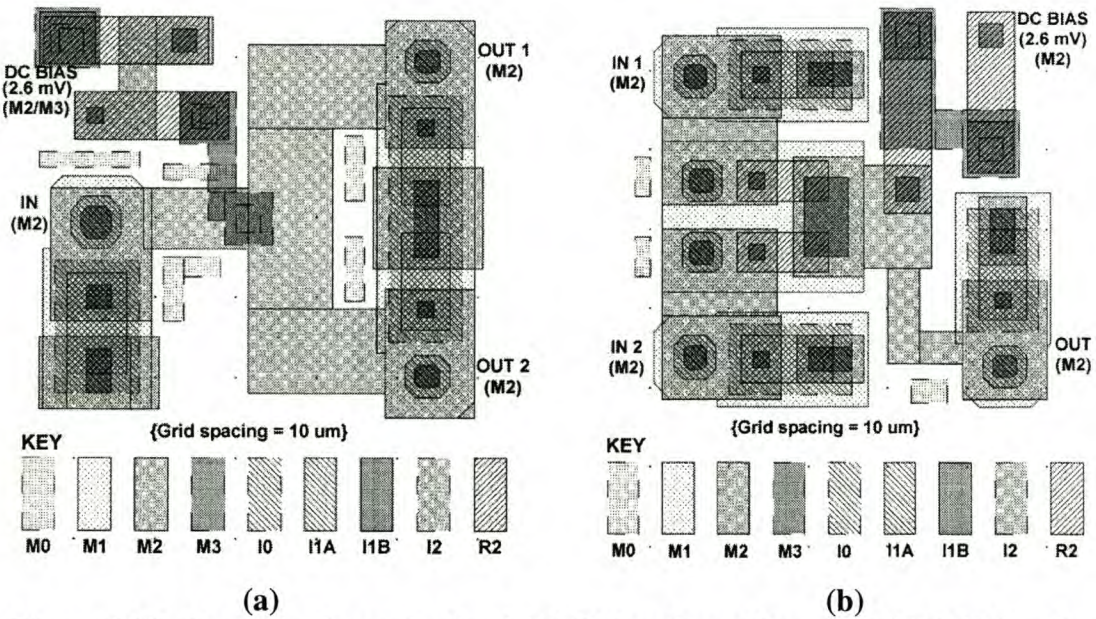


Figure 6.10: Layout schematics for (a) RSFQ pulse splitter and (b) RSFQ pulse merger

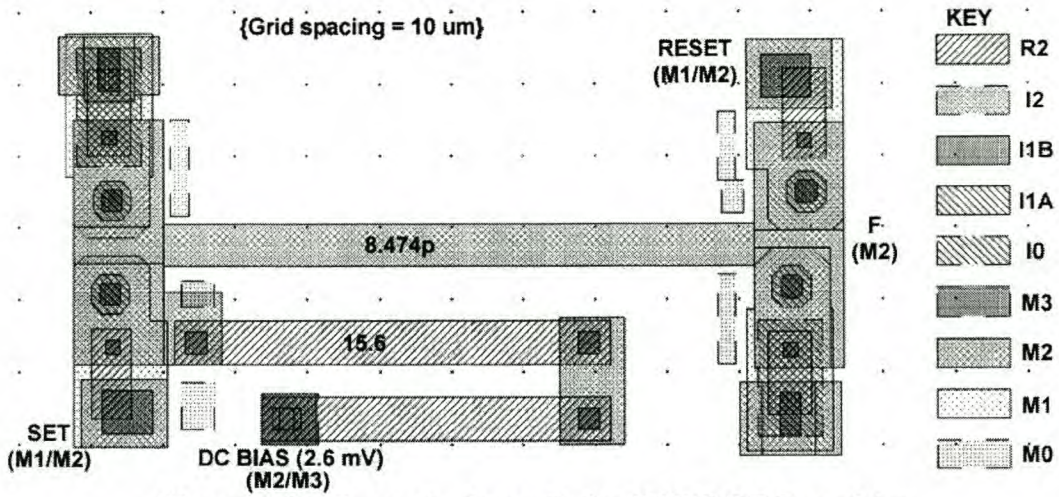


Figure 6.11: Layout schematic for RSFQ DRO-register

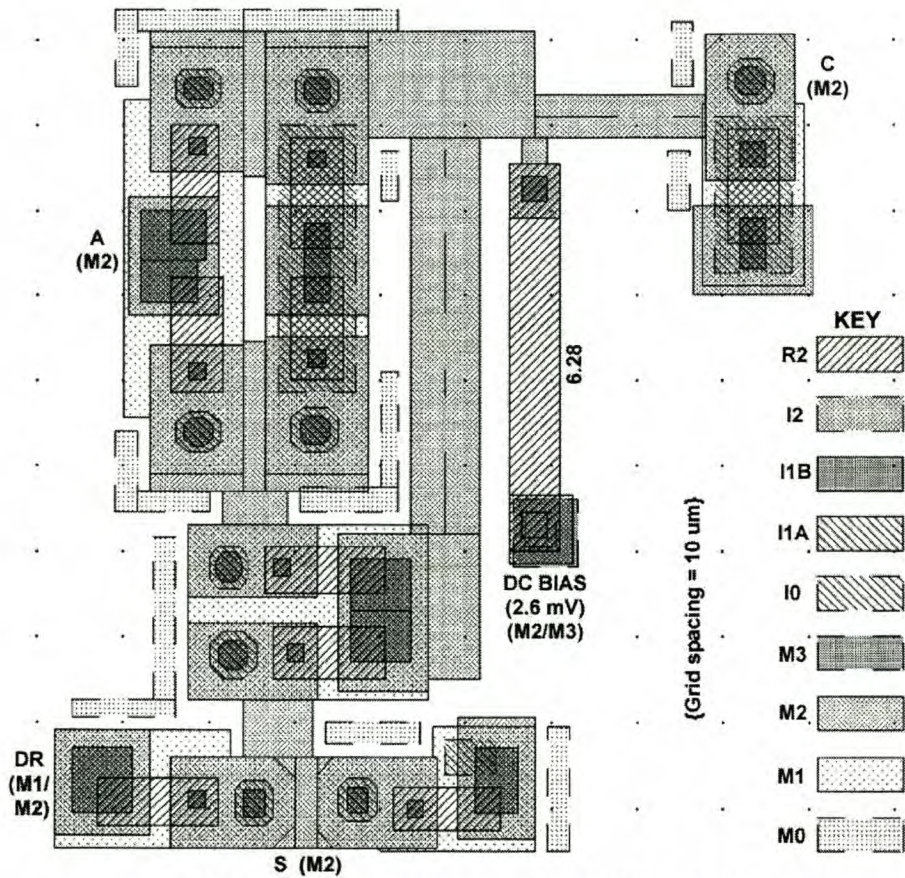


Figure 6.12: Layout schematic for RSFQ T1 flip-flop

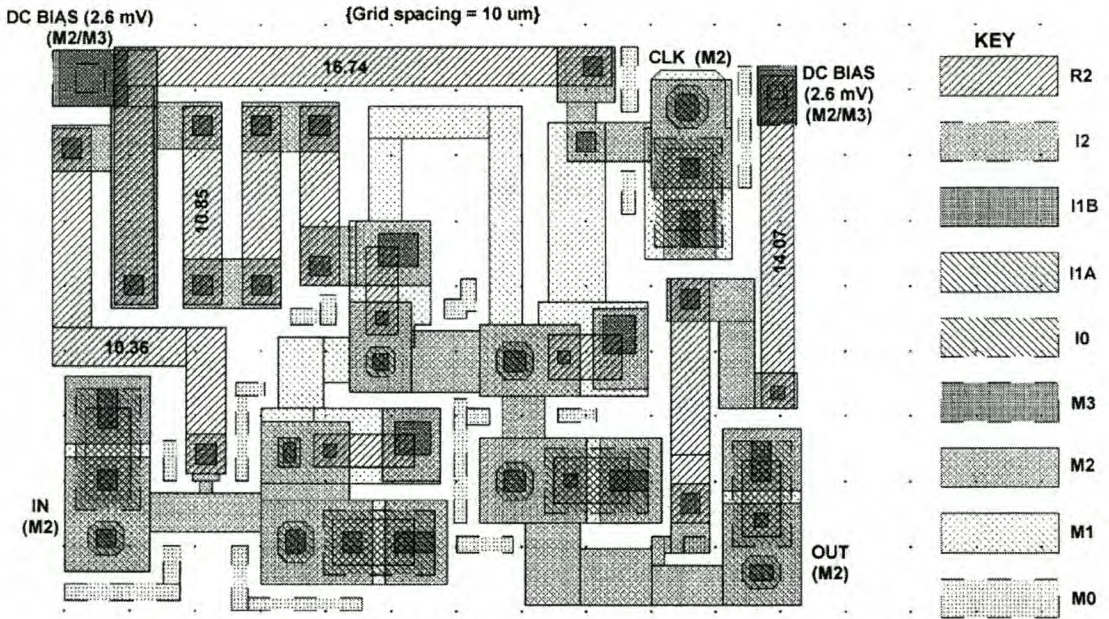


Figure 6.13: Layout schematic for RSFQ NOT-gate

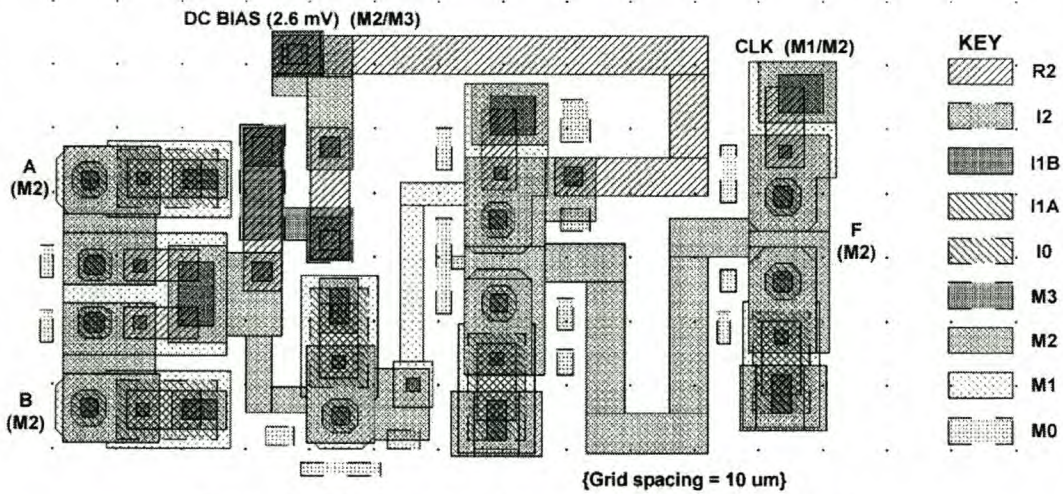


Figure 6.14: Layout schematic for RSFQ OR-gate

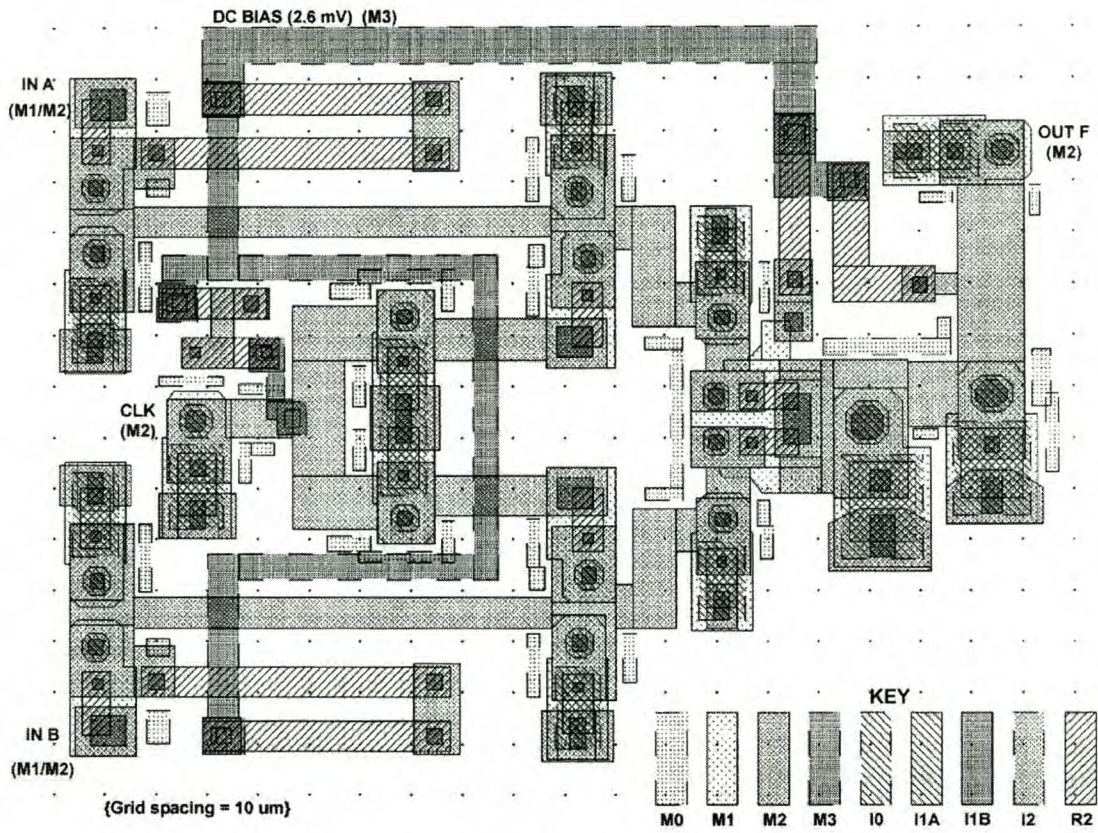


Figure 6.15: Layout schematic for RSFQ AND-gate

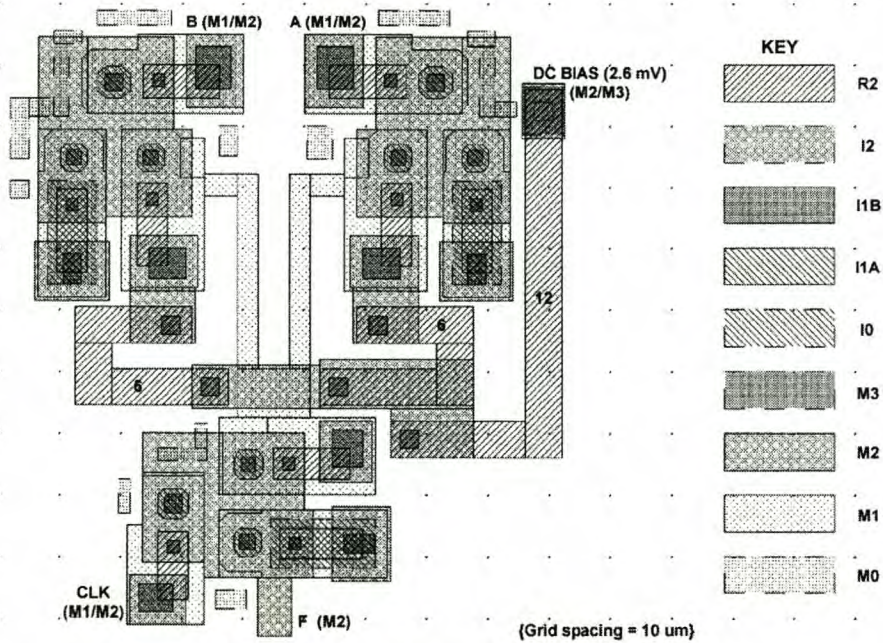


Figure 6.16: Layout schematic for RSFQ XOR-gate

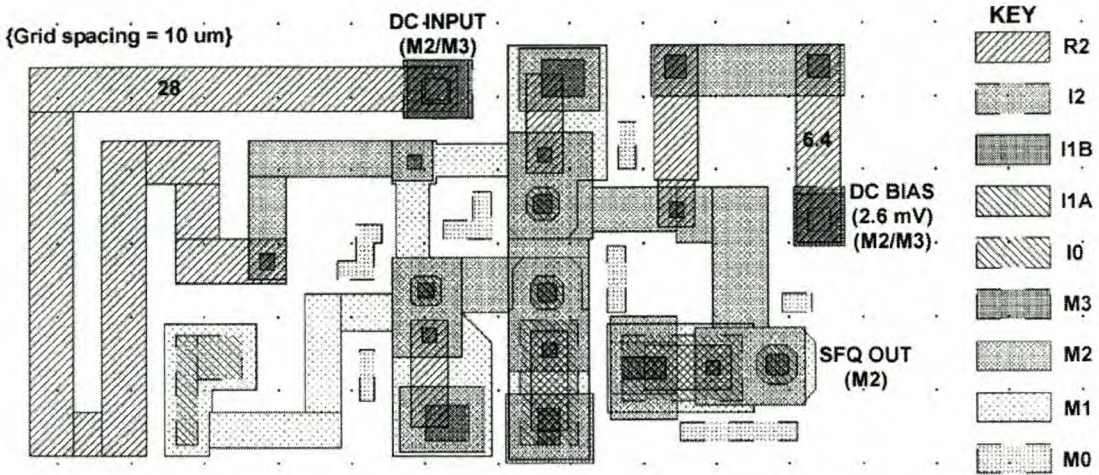


Figure 6.17: Layout schematic for DC-to-SFQ converter

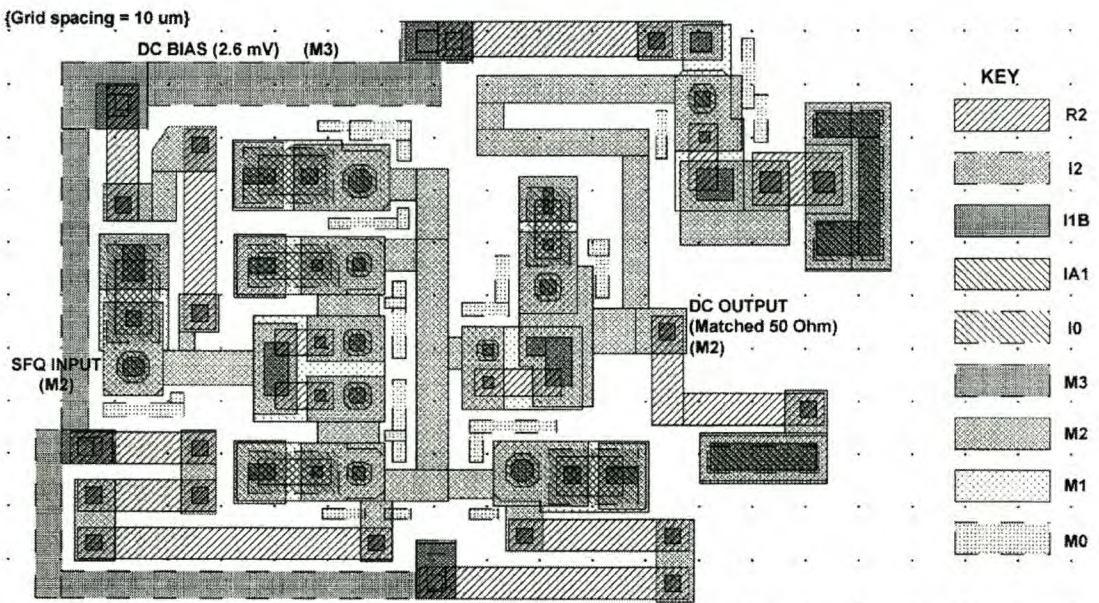


Figure 6.18: Layout schematic for SFQ-to-DC converter

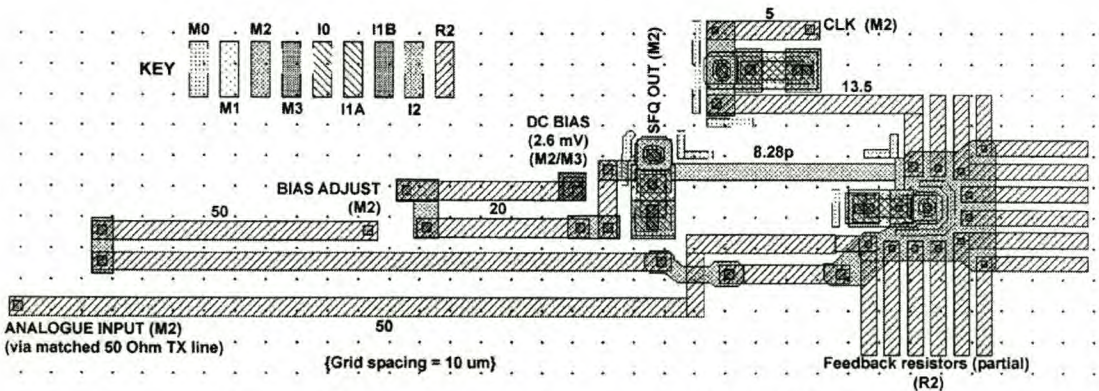


Figure 6.19: Layout schematic for level detector

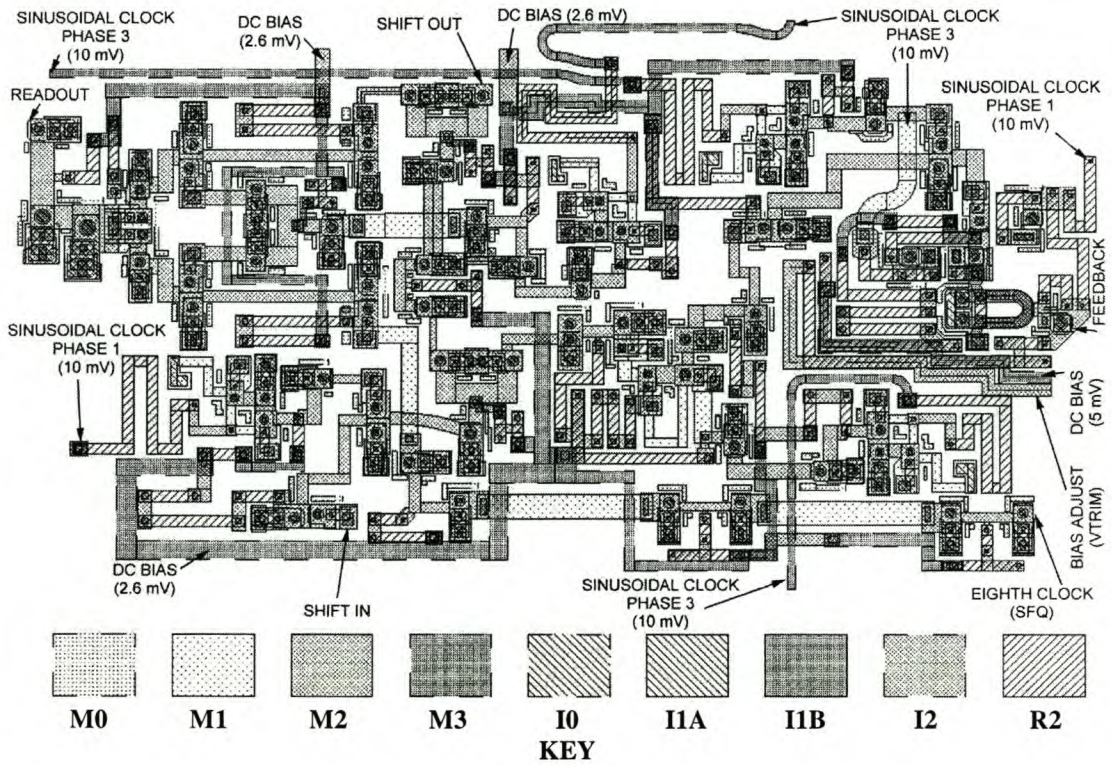


Figure 6.20: Layout schematic for shift register cell with feedback output and RSFQ AND-gate readout

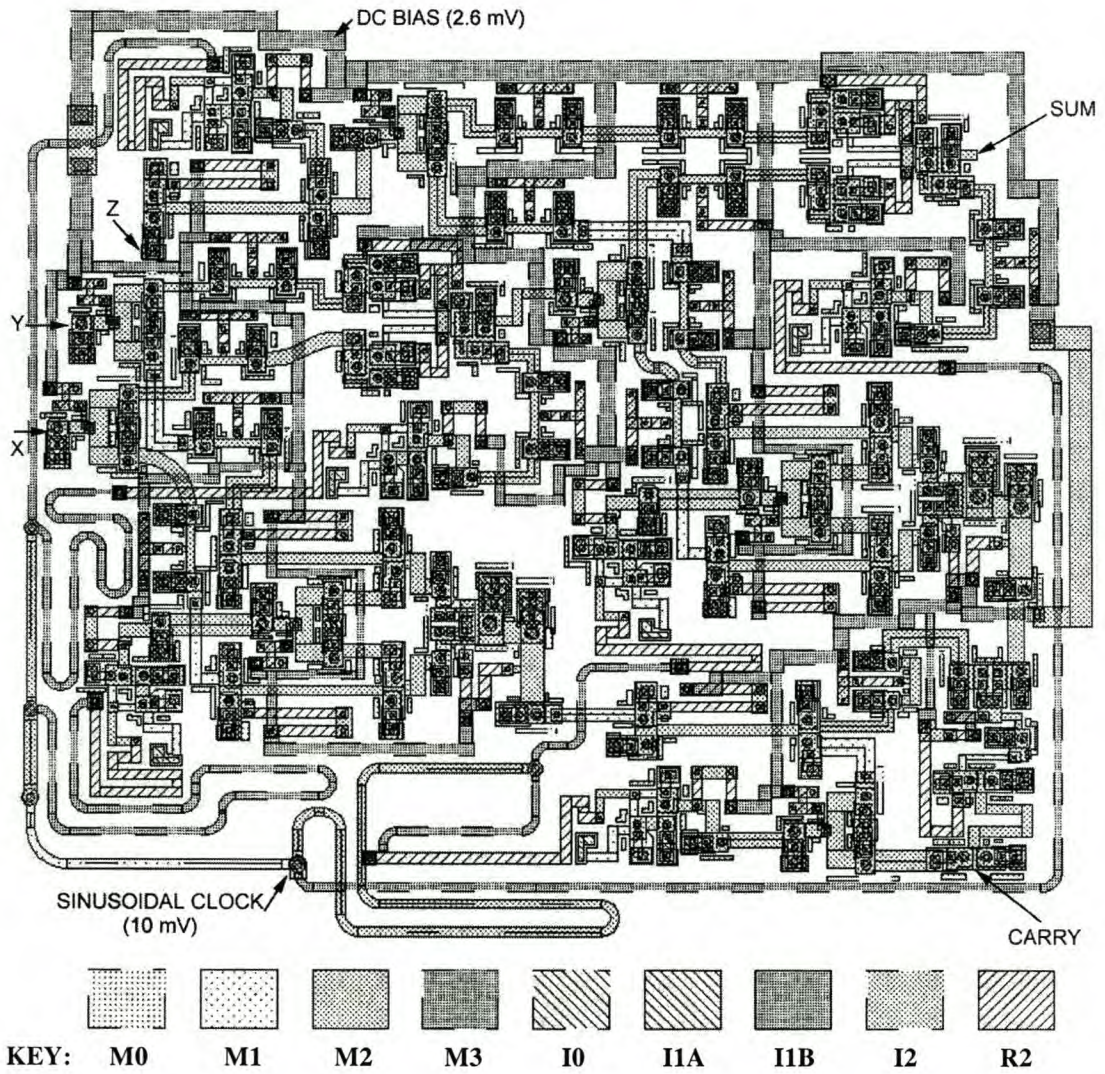


Figure 6.21: Layout schematic for 3-input full adder

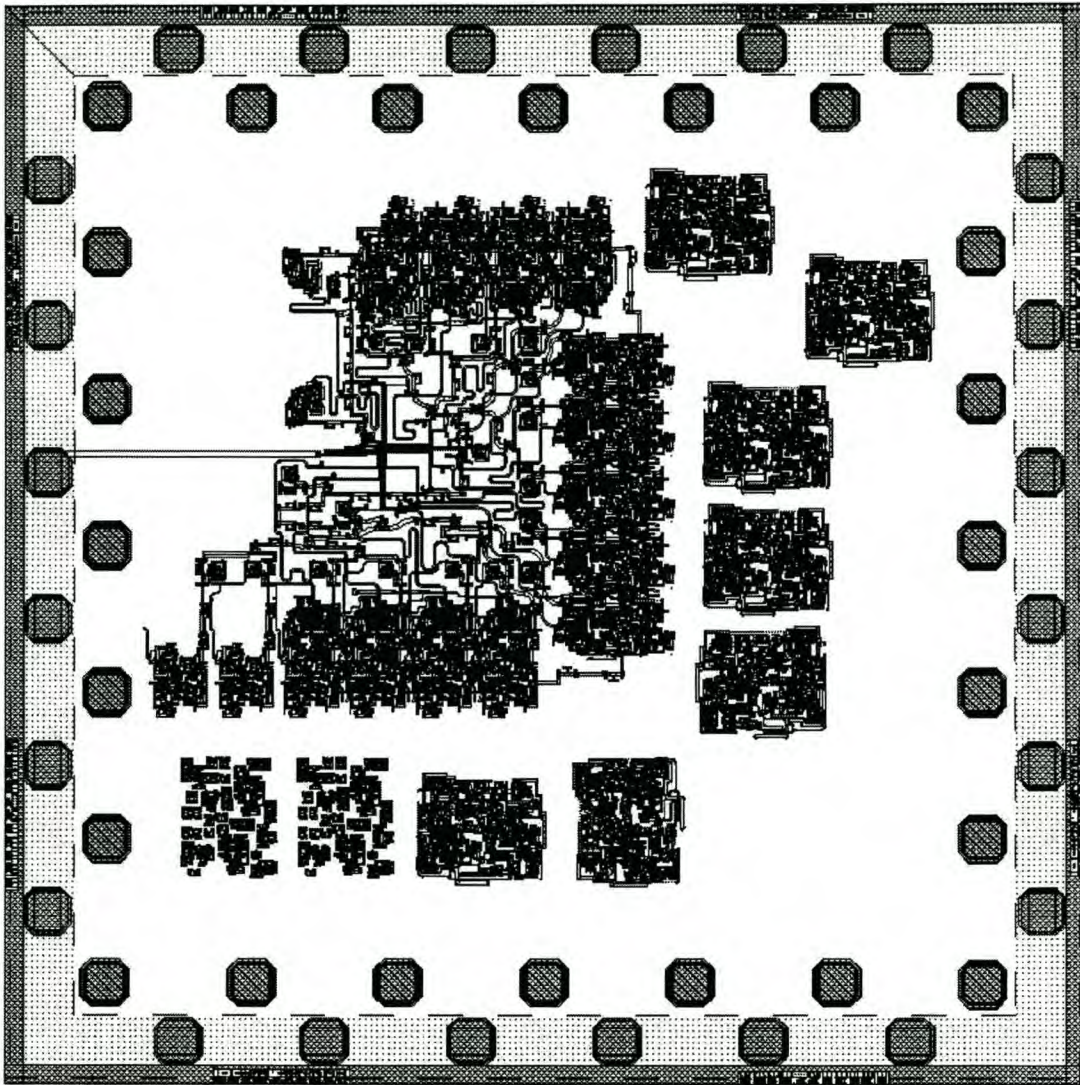


Figure 6.22: Incomplete layout schematic for ADC

6.5 Conclusions

The inductance values of every connection line in every RSFQ cell were calculated with SLINE [47]. The transmission line model [48] used by SLINE is not well suited to the small length-to-width ratios, right-angle bends and line crossings of some of the transmission lines.

The speed of RSFQ circuits, and the fact that all the switching elements are connected inductively, makes precise control over on-chip inductance values very important [53]. The best way to achieve good control over inductance values is to use layout extraction to determine the inductance of line structures after the layout is done. Through recursion, the actual inductance values can then be brought very close to the design values. Work on such a technique has already been done by Guan *et al.* [53], although they used interpolation to estimate the inductance from lookup tables generated for certain standard structures.

A more accurate technique, albeit more time consuming, would be to simulate these structures with numerical methods using magnetoquasistatic approximations. This can be done for smaller subcircuits, and it may be well worth the effort by

reducing the chance that an expensive chip could malfunction. Such a technique will also reveal the influence of line crossings by providing mutual inductance values.

Although all RSFQ interconnections in the ADC layout were implemented with inductively connected cascaded JTLs, future projects will most probably use matched TX line structures. Not only will matched TX lines reduce the layout area wasted on interconnections, but it will also lower the power consumption of the system by about $1 \mu\text{W}$ for every JTL replaced by a stretch of matched transmission line.

The most important advantage of impedance matched TX lines would be the ability to interconnect different superconducting chips in a multi-chip module (MCM,) without the need to convert SFQ pulses to voltage state signals and back.

Such inter-chip connections would, however, require careful attention to the matching of the chip input and output pins, and the off-chip TX lines.

Chapter 7 - Conclusions and Future

7.1 Conclusions

The primary objective of the project was to design, simulate and construct a complex RSFQ circuit that could find application in RF or microwave systems. This was achieved by designing the ADC, and simulating the correct functionality thereof.

Various simulations on the analogue-to-digital converter showed that the RSFQ system can work, and a reasonable theoretical yield was obtained from simulations. It was also established from Monte Carlo simulations that RSFQ has very good parameter tolerance immunity.

The results discussed in Chapter 5 show that even limited failures can be tolerated in signal processing systems like the ADC. This makes RSFQ logic extremely suitable to high speed digital communication applications, where error correction techniques are commonly used to fix bit errors.

The secondary objective was to contrive a way of testing RSFQ circuits with limited measuring equipment, and this was achieved by tailoring the system so that time domain output signals could be observed and validated by performing frequency domain measurements.

This project also achieved other results. Firstly, the design of the ADC created an RSFQ test bed, in which subsystems of the ADC can be interchanged with other RSFQ systems in order to test new gates or logic structures. This is especially important, since the ADC was designed to deliver output signals that can be observed with frequency domain measurements, whereas the SFQ outputs of other RSFQ circuits would be very nearly impossible to measure, even with very expensive time domain measurement equipment.

Another interesting achievement was the use of COSL logic gates alongside RSFQ logic to create a hybrid circuit. This proves the usefulness of COSL gates in other superconducting digital systems, and shows that the use of these gates need not only be confined to all-COSL microchips or ADCs.

Finally it has to be mentioned that the operating power dissipation of the ADC, calculated as 1.325 mW, proves that such superconducting chips can run quite efficiently in liquid helium. The heat of vaporization for liquid helium is 2.09×10^4 J/kg, and it has a density of 0.125 g/cm^3 [1]. This means that, in a perfectly adiabatic operating environment (one in which no exchange of heat with the surrounding environment takes place,) the ADC will take nearly 550 hours to boil off one litre of liquid helium. Of course, even good dewars will allow some thermal energy to leak into the system and speed up the change of phase in the helium, but the system can still run a long time on a modest amount of liquid helium.

7.2 Problems and alternative solutions

Advanced circuits utilizing hundreds of gates become impractical to clock using sinusoidal input signals, as the equivalent impedance seen by the global clock signal becomes too low. Etching difficulties arise due to the low impedance of the lines.

The etching demands can be eased by increasing the input resistance of each DC-to-SFQ converter. A twofold increase in the value of each input resistor will

cause the impedance of the connection to the off-chip oscillator to double as well. This will in turn almost halve the width of the transmission lines distributing the signal through the chip. The shortcomings of this technique are the increase in physical resistor size at the input to each DC-to-SFQ converter, and the need to increase the amplitude of the global clock signal by the same factor as the increase in resistance.

Alternatively, the impedance of clock lines can be raised without changing the values of input resistors. The resulting mismatch will cause signal reflections, and the amplitude of the global clock signal will have to be increased to ensure that the DC-to-SFQ converters function correctly. This technique is not a good engineering practice, and unless better solutions are ruled out by space limitation, does not merit serious consideration.

The best solution to impedance problems in large scale clocking structures is to do away with sinusoidal clocking altogether. A single DC-to-SFQ converter can generate a master SFQ clock pulse that can in turn be divided into numerous pulses through banks of pulse splitters. This technique has some disadvantages:

- Every pulse splitter can have a different pulse propagation delay due to manufacturing offsets. These delays may add to total pulse propagation delays that differ more than the clock period down different pulse routing paths. This could cause data that pass through different logic paths to slip clock cycles, and cause incorrect computations. It may be impossible to ensure that the entire circuit is clocked in phase.
- The length of clock lines may cause inductance problems when connecting divergers. Line widths can be kept small by decreasing the distance between divergers, but this will result in large tracts of layout space being consumed by clocking structures.

Even though circuit clocking by the distribution of SFQ pulses creates serious design problems, ways can be found to work around these problems.

The first design change may be to abandon in-phase clocking as a prerequisite for correct circuit operation. One way of doing this is to use handshaking [16]. Another is to propagate a clock pulse along the same path as a logical operation, and to use the propagation delay as a way of regulating clock periods. This technique of asynchronous clock distribution could actually speed up certain operations when faster logic gates are given shorter clock periods. Careful design will still be needed when results from different logic paths need to be integrated in a single operation.

The other way to cut down on the area occupied by clock distribution circuitry, is to use matched impedance transmission lines to replace cascades of JTLs.

7.3 Future and applications

It has been apparent for the past few years that, although semiconductor technology is still advancing at a rapid pace, it will soon run into physical limits.

Superconducting electronic technology on the other hand, can still advance quite far from where it is today. Results achieved by Chen *et al.* [17] on circuits containing self-damped Josephson junctions are particularly promising. The clock frequencies of VLSI systems using RSFQ logic are expected to reach 100 GHz when the etching resolution scales down to 0.4 μm [54].

It seems that, when the progress on the physical technology of semiconductors slows down, the near future, technology-wise, will belong to superconducting or optical electronics. However, on-chip optical logic structures are large [55], and

research in semiconductor lasers seems to be preoccupied with commercially successful applications such as high density data storage devices and fibre-optic communications. However, optical fibre is considered an attractive option for introducing signals into RSFQ chips, since glass fibre is a poor conductor of heat [54].

With the advent of large superconducting processors, our ideas of what computer architecture should look like, might have to change.

The classic Von Neumann architecture at the core of modern computers, where an arithmetic logic unit, instruction registers and memory banks are interconnected to form a processor, is most likely to become obsolete in ultra-fast digital computers. The reason for this is that faster arithmetic logic units would waste most of their time waiting for signals to propagate from the instruction registers, and to and from the memory cells.

In order to better utilize the extremely high operating speeds of future superconducting logic circuits, it might be advantageous to switch to neural architectures when designing processors. In such neural systems, small logic blocks would have to be able to adopt a function such as that of memory cell, accumulator, instruction register or mathematical function as it is needed. This would require the system to plot a solution course through its hardware blocks for every task it encounters. Neural computing should enhance the ability of processors to work in parallel, and provide the ability for the system itself to overcome defects in the circuitry. Not only would such neural processors require vast amounts of logic blocks, but also some very ingenious software routines. The advent of such computers is, however, still a long way off.

Apart from the obvious advantages of RSFQ superconducting computers in the continuing quest for increased computing power and speed, specific applications such as the ADC will also find immediate application. RSFQ is particularly well suited to digital communication systems, where high bit speeds are required, and bit errors can be corrected or sometimes tolerated.

The ADC discussed in this project was designed for the purpose of testing RSFQ logic, and not for maximum sampling speed. If an RSFQ-based ADC is designed for maximum sampling frequency, this frequency can approach several hundreds of GHz. The speed advantage over semiconducting equivalents is therefore substantial.

Faster ADCs will not only find application in telecommunication systems, but are also a prerequisite for increased performance in future radars [56].

The requirement for cryogenic operating conditions are not restrictive, since modern radars, military or civilian, are large and expensive. The cost of developing phased array antennas, or the sheer size of mechanical turning gear easily dwarf the cost and size of a cooler unit.

In certain applications, such as infrared radio astronomy, the sensor arrays require cryogenic cooling. The addition of RSFQ electronics to such a system is thus rather straightforward.

In conclusion: the future of high speed digital electronics, from niche applications in military systems, to more general use in commercial systems, is almost certain to include the use of RSFQ superconducting electronics.

References

- [1] R.A. Serway, *Physics for Scientists and Engineers*, Third Edition, Saunder College Publishing, 1992
- [2] W. Meissner, R. Ochsenfeld, "Ein neuer Effekt bei Eintritt der Supraleitfähigkeit," *Die Naturwissenschaften*, Vol. 21, pp. 787-788, 3 November 1933
- [3] T.P. Orlando, K.A. Delin, *Foundations of Applied Superconductivity*, Addison-Wesley Publishing Company, 1991
- [4] J. Bardeen, L.N. Cooper, J.R. Schrieffer, "Theory of Superconductivity," *Physical Review*, Vol. 108, no. 5, pp. 1175-1204, December 1957
- [5] I.M. Firth, *Superconductivity*, Mills & Boon Limited, 1972
- [6] J.G. Bednorz, K.A. Müller, "Possible High T_C Superconductivity in the Ba - La - Cu - O System," *Zeitschrift für Physik B - Condensed Matter*, Vol. 64, pp. 189-193, 1986
- [7] M.K. Wu, J.R. Ashburn, C.J. Torng, P.H. Hor, R.L. Meng, L. Gao, Z.J. Huang, Y.Q. Wang, C.W. Chu, "Superconductivity at 93 K in a New Mixed-Phase Y-Ba-Cu-O Compound System at Ambient Pressure," *Physical Review Letters*, Vol. 58, no.9, pp. 908-911, 2 March 1987
- [8] L. Solymar, D. Walsh, *Electrical Properties of Materials*, Sixth Edition, Oxford University Press, 1998
- [9] H. Nowotny, U. Felt, *After the Breakthrough – The emergence of high-temperature superconductivity as a research field*, Cambridge University Press, 1997
- [10] L. Garwin, "Superconductors make their public debut in US utility network," *Nature*, Vol. 395, p. 733, 22 October 1998
- [11] B.D. Josephson, "Possible new effects in superconducting tunneling," *Physical Review Letters*, Vol. 1, no. 7, pp. 251-253, July 1962
- [12] P.W. Anderson, J.M Rowell, "Probable observation of the Josephson superconducting tunneling effect," *Physical Review Letters*, Vol. 10, pp. 230-232, March 1963
- [13] W.J. Perold, M. Jeffrey, Z. Wang, T. Van Duzer, "Complementary Output Switching Logic - A New Superconducting Voltage-State Logic Family," *IEEE Transactions on Applied Superconductivity*, Vol. 6, no. 3, pp. 125-131, 3 September 1996

- [14] M. Jeffrey, W. Perold, T. Van Duzer, "Superconducting complementary output switching logic operating at 5-10 Gb/s," *Applied Physics Letters*, Vol. 69, no. 18, pp. 2746-2748, 28 October 1996
- [15] W.J. Perold, "Complementary Output Switching Logic: A Superconducting Voltage-State Logic Family Operating at Microwave Frequencies," in *Proceedings of the 1998 South African Symposium on Communications and Signal Processing: COMSIG '98*, pp.435-440, 1998
- [16] K.K. Likharev, V.K. Semenov, "RSFQ Logic/Memory Family: A New Josephson-Junction Technology for Sub-Terahertz-Clock-Frequency Digital Systems," *IEEE Transactions on Applied Superconductivity*, Vol. 1, no. 1, pp. 3-28, March 1991
- [17] W. Chen, A.V. Rylyakov, V. Patel, J.E. Lukens, K.K. Likharev, "Rapid Single Flux Quantum T-Flip Flop Operating up to 770 GHz," *IEEE Transactions on Applied Superconductivity*, Vol. 9, no. 2, pp. 3212-3215, June 1999
- [18] Stony Brook University Homepage:
<http://pavel.physics.sunysb.edu/RSFQ/RSFQ.html>
- [19] F.J. Rabie, "Superconducting COSL building blocks for ultra-high speed logic circuits," MscEng Thesis, University of Stellenbosch, 1999
- [20] T. Van Duzer, C.W. Turner, *Principles of Superconductive Devices and Circuits*, Second Edition, Prentice-Hall, Inc., 1999
- [21] K.K. Likharev, *Dynamics of Josephson Junctions and Circuits*, Gordon and Breach Science Publishers, pp. 206-220, 1986
- [22] Hypres, 175 Clearbrook Road, Elmsford, New York 10523, *Niobium Design Rules*, 1997.
Homepage: <http://www.hypres.com>
- [23] C.J. Fourie, "Ultra-hoëfrekwensie RSFQ (Rapid Single-Flux-Quantum) supergeleidende digitale skakellogika," BEng thesis, University of Stellenbosch, 1998
- [24] Whiteley Research Inc., 456 Flora Vista Avenue, Sunnyvale, CA 94086, *WRSpice Circuit Simulation System*, Homepage: <http://www.srware.com>
- [25] C.J. Fourie, W.J. Perold, "Ultra High-Speed Superconducting RSFQ (Rapid Single-Flux- Quantum) Digital Switching Logic," *IEEE Africon*, Vol. 2, pp. 1183-1188, 1999
- [26] S. Ramo, J.R. Whinnery, T. Van Duzer, *Fields And Waves In Communication Electronics*, Third Edition, John Wiley & Sons, Inc., pp. 214-221, 1994
- [27] R.E. Collin, *Foundations for Microwave Engineering*, Second Edition, McGraw-Hill, Inc., 1992

- [28] K.C. Gupta, R. Garg, I. Bahl, P. Bhartia, *Microstrip Lines and Slotlines*, Second Edition, Artech House, pp. 375-399, 1996
- [29] C.A.W. Vale, University of Stellenbosch, private communication
- [30] P.D. Bradley, S.V. Rylov, "A Comparison of Two Types of Single Flux Quantum Comparators for a Flash ADC with 10 GHz Input Bandwidth," *IEEE Transactions on Applied Superconductivity*, Vol. 7, no. 2, pp. 2677-2680, June 1997
- [31] Avant! Corporation, 46871 Bayside Parkway, Fremont, CA 94538, *HSpice*. Homepage: <http://www.avanticorp.com>
- [32] Intusoft, P.O. Box 710, San Pedro, CA 90733-0710, *ICAP/4*. Homepage: <http://www.intusoft.com>
- [33] R. Spence, R.S. Soin, *Tolerance Design of Electronic Circuits*, New York: Addison-Wesley, pp. 56-87, 1988
- [34] W.J. Perold, University of Stellenbosch, private communication
- [35] W.J. Perold, C.J. Fourie, "Modeling superconducting components based on the fabrication process and layout dimensions," *IEEE Transactions on Applied Superconductivity*, (accepted for publication)
- [36] S. Soclof, *Design and Applications of Analog Integrated Circuits*, Prentice-Hall International, Inc., pp. 746-792, 1991
- [37] P. Horowitz, W. Hill, *The Art of Electronics*, Second Edition, Cambridge University Press, pp. 621-625, 1989
- [38] D. Gupta, D.V. Gaidarenko, S.V. Rylov, "A 16-bit Serial Analog-to-Digital Converter Module with Optical Output," *IEEE Transactions on Applied Superconductivity*, Vol. 9, no. 2, pp. 3030-3033, June 1999
- [39] R.D. Sandell, B.J. Dalrymple, A.D. Smith, "An SFQ Digital to Analog Converter," *IEEE Transactions on Applied Superconductivity*, Vol. 7, no. 2, pp. 2468-2470, June 1997
- [40] J.G. Lourens, University of Stellenbosch, private communication
- [41] J.C. Candy, G.C. Temes, *Oversampling Delta-Sigma Data Converters, Theory, Design and Simulation*, IEEE Press, p.1, 1992
- [42] J.G. Proakis, D.G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*, Third Edition, Prentice-Hall International, pp. 756-762, 1996
- [43] J.F. Wakerley, *Digital Design Principles and Practices*, Second Edition, Prentice-Hall, Inc., 1994

- [44] M.C. Pistorius *et al.*, *Rekenaarwetenskap*, Department of Computer Science and Information Systems, University of South Africa, pp. 163-168, 1992
- [45] The MathWorks, Inc., 24 Prime Park Way, Natick, Massachusetts 01760, *Matlab Version 4 for Windows*, Homepage: <http://www.mathworks.com>
- [46] P.Z. Peebles, *Probability, Random Variables, and Random Signal Principles*, Third Edition, McGraw-Hill, Inc., 1993
- [47] S.R. Whiteley, *SLINE Version 1.0*, June 1996. Available via the Whiteley Research homepage: <http://www.srware.com>
- [48] W.H. Chang, "The inductance of a superconducting strip transmission line," *Journal of Applied Physics*, Vol. 50, no. 12, pp. 8129-8134, December 1979
- [49] Applied Wave Research Inc., *Microwave Office 2000 Version 3.22*, Homepage: <http://www.mwoffice.com>
- [50] M. Radparvar, Hypres Inc., private communication
- [51] M. Jeffrey, T. Van Duzer, "Magnetic imaging of moat-guarded superconducting electronic circuits," *Applied Physics Letters*, Vol. 67, no. 12, pp. 1769-1771, 18 September 1995
- [52] S. Bermon, T. Gheewala, "Moat-guarded Josephson junctions," *IEEE Transactions on Magnetics*, Vol. MAG-19, no. 3, pp. 1160-1164, May 1983
- [53] B. Guan, M.J. Wengler, P. Rott, M.J. Feldman, "Inductance Estimation for Complicated Superconducting Thin Film Structures with a Finite Segment Method," *IEEE Transactions on Applied Superconductivity*, Vol. 7, no. 2, pp. 2776-2780, June 1997
- [54] D. K. Brock, E. K. Track, J. M. Rowell, "Superconductor ICs: the 100-GHz second generation," *IEEE Spectrum*, Vol. 37, no. 12, pp. 40-46, December 2000
- [55] J. M. Rowell, "Photonic Materials," *Scientific American*, pp. 125-134, October 1986
- [56] S. Kingsley, S. Quegan, *Understanding Radar Systems*, Scitech Publishing, pp. 319-320, 1999
- [57] Borland International, Inc., 100 Borland Way, P.O. Box 660001, Scotts Valley, CA 95067-0001, *Delphi Version 3.0 for Windows 95 & Windows NT*
- [58] W. Hilberg, "From Approximations to Exact Relations for Characteristic Impedances," *IEEE Trans.*, Vol. MTT-17, pp. 259-265, 1969

Appendix A – Superconducting Mathematical Toolbox

Impedance calculator – theory and source code

The impedance calculator is a procedure in the Superconducting Mathematical Toolbox developed for this project. The Toolbox was programmed in Borland Delphi 3.0 [57], and the inductance calculation formulae were obtained from [48].

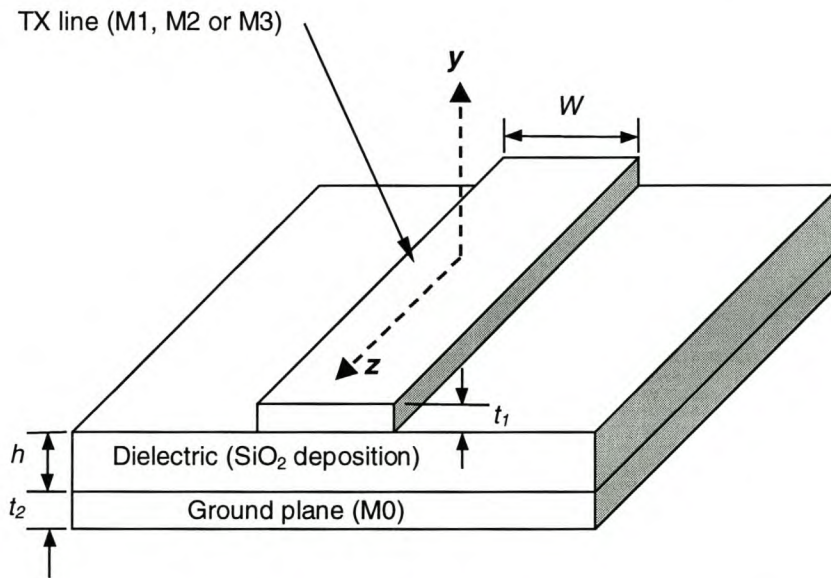


Figure A.1: Superconducting microstrip configuration

The expressions for the inductance of a superconducting microstrip transmission line as they are derived by Chang [48] are listed below to clarify the variables in the source code.

The inductance of the superconducting microstrip transmission line depicted in Fig. A.1 is

$$L = \frac{\mu_0}{WK(W, h, t_1)} \left\{ h + \lambda_1 \left[\coth\left(\frac{t_1}{\lambda_1}\right) + \frac{2p^{1/2}}{r_b} \operatorname{csch}\left(\frac{t_1}{\lambda_1}\right) \right] + \lambda_2 \coth\left(\frac{t_2}{\lambda_2}\right) \right\} \quad (\text{A.1})$$

where h , t_1 , t_2 and W are the physical dimensions as defined in Fig. A.1, in SI units, and μ_0 is the permeability of free space.

The fringe field factor $K(W, h, t_1)$ is defined by

$$K(W, h, t_1) = \frac{h}{W} \frac{2}{\pi} \ln \frac{2r_b}{r_a} \quad (\text{A.2})$$

The other variables are defined as follows:

$$a = \frac{r_b}{r_a} + \left[\left(\frac{r_b}{r_a} \right)^2 - 1 \right]^{1/2} \quad (\text{A.3})$$

$$\ln r_a = -1 - \frac{\pi W}{2h} - \frac{p+1}{p^{1/2}} \tanh^{-1}(p^{-1/2}) - \ln\left(\frac{p-1}{4p}\right) \quad (\text{A.4})$$

$$r_b = r_{bo} \text{ for } W/h \geq 5, \quad (\text{A.5})$$

$$r_b = r_{bo} - [(r_{bo} - 1)(r_{bo} - p)]^{1/2} + (p + 1) \times \tanh^{-1}\left(\frac{r_{bo} - p}{r_{bo} - 1}\right)^{1/2} - 2p^{1/2} \tanh^{-1}\left(\frac{r_{bo} - p}{p(r_{bo} - 1)}\right)^{1/2} + \frac{\pi W}{2h} p^{1/2} \quad \text{for } 5 > \frac{W}{h} \geq 1 \quad (\text{A.6})$$

$$r_{bo} = \eta + \frac{p+1}{2} \ln \Delta \quad (\text{A.7})$$

$$\Delta = \text{larger value, } \eta \text{ or } p \quad (\text{A.8})$$

$$\eta = p^{1/2} \left\{ \frac{\pi W}{2h} + \frac{p+1}{2p^{1/2}} \left[1 + \ln\left(\frac{4}{p-1}\right) \right] - 2 \tanh^{-1}(p^{1/2}) \right\} \quad (\text{A.9})$$

$$p = 2\beta^2 - 1 + [(2\beta^2 - 1)^2 - 1]^{1/2} \quad (\text{A.10})$$

$$\beta = 1 + \frac{t_1}{h} \quad (\text{A.11})$$

The Delphi 3.0 procedure for inductance calculation is listed below. The main program is not included. The program was developed for a Microsoft Windows environment, hence the use of object oriented programming techniques.

```
procedure TForm2.CalcIndClick(Sender: TObject);

var
par_K, par_b1, par_b2, par_w, par_d, par_mu, par_lam1, par_lam2 :
    double;
TF11, TF12, coth1, coth2 : double;
    par_beta, par_p, re_eta, im_eta, re_delta, im_delta, re_rbo,
        im_rbo, re_rb, im_rb, par_ra : double;
    re_atanhqrtp, im_atanhqrtp, abs_eta, abs_delta, ang_delta, re_K,
        im_K : double;

begin      {Calculate Inductance of Microstrip Line}
    par_K := 1;
    par_b1 := StrToFloat(param_b1.Text)*1e-9;      {Read line thickness
        from manual input}
    par_b2 := StrToFloat(param_b2.Text)*1e-9;      {Read ground plane
        thickness from manual input}
```

```

par_w := StrToFloat(param_w.Text)*1e-6;           {Read line width from
manual input}
par_d := StrToFloat(param_d.Text)*1e-9;           {Read dielectric
thickness from manual input}
par_lam1 := StrToFloat(param_lam1.Text)*1e-9; {Read lambda1
(penetration depth of line) from manual input}
par_lam2 := StrToFloat(param_lam2.Text)*1e-9; {Read lambda2
(penetration depth of ground plane) form manual input}
par_beta := 1 + (par_b1/par_d);                    {Calculate Beta}
par_p := 2*par_beta*par_beta - 1 + sqrt(sqr((2*par_beta*par_beta)-
1) - 1); {Calculate p}
re_atanhsqrtp := 0.5*ln(abs((1+sqrt(par_p))/(1-sqrt(par_p))));
{Real part of complex arctanh argument}
im_atanhsqrtp := 0.5*pi;
{Imaginary part of complex arctanh argument}
re_eta :=
sqrt(par_p)*(((pi*par_w)/(2*par_d))+((par_p+1)/(2*sqrt(par_p)))
*(1+ln(4/(par_p-1)))-2*re_atanhsqrtp); {Real part of eta}
im_eta := sqrt(par_p)*(-2*im_atanhsqrtp);
{Imaginary part of eta}
abs_eta := sqrt(sqr(re_eta)+sqr(im_eta));
if abs_eta > par_p then
  begin                                           {Assign to delta the
bigger value of p and eta}
    re_delta := re_eta;
    im_delta := im_eta;
    abs_delta := abs_eta;
    ang_delta := arctan(im_delta/re_delta);
  end
else
  begin
    re_delta := par_p;           {p always real}
    im_delta := 0;
    abs_delta := abs(par_p);
    ang_delta := 0;
  end;
re_rbo := re_eta + 0.5*(par_p+1)*ln(abs_delta); {might cause
problems if Delphi can't compute negative angles...}
im_rbo := im_eta + 0.5*(par_p+1)*ang_delta;
{rb = rbo if W/h > 5}
if (par_W/par_d) >= 5 then
  begin
    re_rb := re_rbo;
    im_rb := im_rbo;
  end
else
  begin
    {Not implemented: In the Hypres layout process, W/d will
only be less than 5 for a 2 micron M2 line and an M3 line
of 5 micron or less. (Neither were ever used for
inductive connections in the ADC layout)}
    This calculation will however appear in future versions.}
  end;
par_ra := exp(-1-((pi*par_W)/(2*par_d))-
((par_p+1)/sqrt(par_p))*arctanh(1/sqrt(par_p))-ln((par_p-
1)/(4*par_p)));
re_K :=
((2*par_d)/(par_W*pi))*ln((2*sqrt(sqr(re_rb)+sqr(im_rb)))/par_r
a);
im_K := ((2*par_d)/(par_W*pi))*arctan(im_rb/re_rb);
par_K := sqrt(sqr(re_K)+sqr(im_K));

```

```

{ par_eta :=
  sqrt(par_p)*(((pi*par_w)/(2*par_d))+((par_p+1)/(2*sqrt(par_p)))
  *(1+ln(4/(par_p-1)))-2*arctanh(sqrt(par_p))); }
if RadioButton2.Checked then
  begin
    par_K := StrToFloat(param_K.Text);          {Read fringe factor
    from manual input}
    label_K.Caption := 'Fringe factor (Manual)';
  end;
if RadioButton3.Checked then
  begin
    label_K.Caption := 'Fringe factor (Calculated)'; {Change
    caption in parameter-input box...}
    param_K.Text := FloatToStr(par_K);
  end;
  coth1 := cosh(par_b1/par_lam1)/sinh(par_b1/par_lam1);
  coth2 := cosh(par_b2/par_lam2)/sinh(par_b2/par_lam2);
  TF11 := 1 + (par_lam1/par_d)*coth1 + (par_lam2/par_d)*coth2;
  TF12 := ((4e-7*pi*par_d)/(par_K*par_w))*TF11*1e6;
  IndUnit.Text := FloatToStr(TF12); { Displays answer IndUnit }
end;

```

Source code for JJ area parameters

The Delphi source code listed in unit `cjf_jmsk` below, contains routines to calculate the most accurate rectangular, circular or eight-sided geometry that conforms to the Hypres design rules [22]. The definition of the dimensions are given graphically in Fig. A.3.

For the rectangular and eight-sided junctions, the calculation routines are programmed to minimize the circumference of the geometry, as well as the error between the specified I_C and that of the layout.

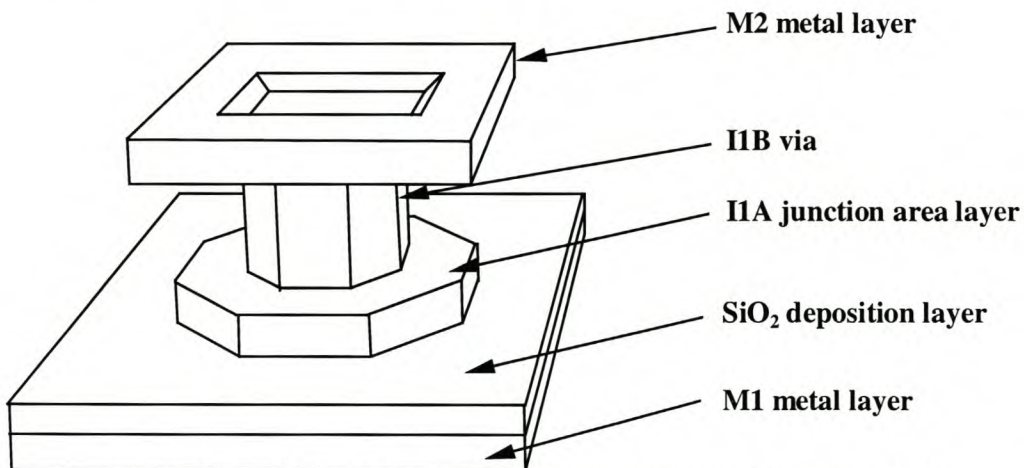


Figure A.2: Three dimensional representation of octagonal JJ

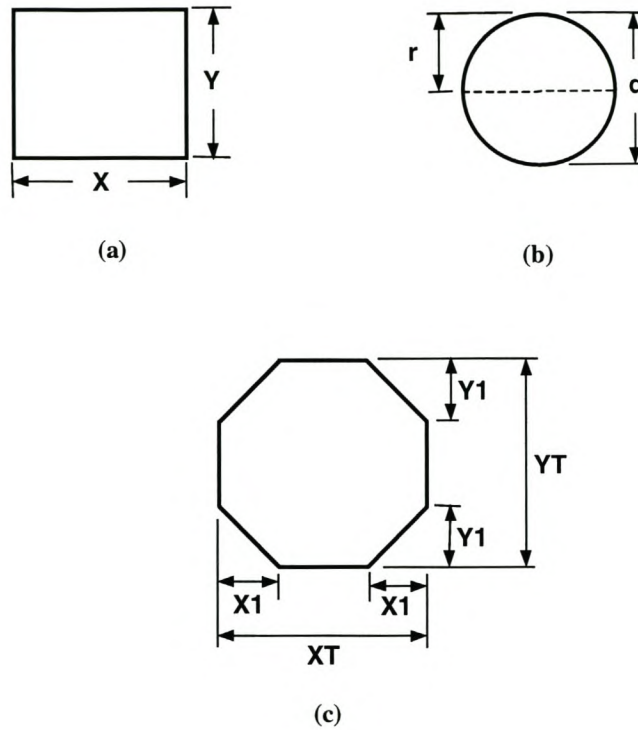


Figure A.3: Layer I1A area of (a) square, (b) circular and (c) octagonal JJ

```

{*****}
{ UNIT for Superconducting Mathematical Toolbox }
{ FreeWare                                     }
{ Programmed by Coenrad J Fourie              }
{ Dept. E&E Eng., University of Stellenbosch  }
{ May 2000                                     }
{*****}
unit cjf_jmsk;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs,
  StdCtrls, ExtCtrls, WinTypes, Menus, cjf_err1;

type
  TForm4 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    jj_critcurrent: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    jj_curdens: TComboBox;
  end;

```

```

Label3: TLabel;
Panel5: TPanel;
Image1: TImage;
Panel6: TPanel;
Panel7: TPanel;
Image2: TImage;
Image3: TImage;
Edit2: TEdit;
Label4: TLabel;
sirk_d: TLabel;
sirk_r: TLabel;
Label5: TLabel;
Label6: TLabel;
Panel8: TPanel;
Label7: TLabel;
Label8: TLabel;
opt_depth: TComboBox;
Label9: TLabel;
Label10: TLabel;
sirk_a: TLabel;
sirk_reldif: TLabel;
vier_xout: TLabel;
vier_yout: TLabel;
vier_a: TLabel;
vier_reldif: TLabel;
Label11: TLabel;
x1_dev: TComboBox;
okt_XTout: TLabel;
okt_YTout: TLabel;
okt_X1out: TLabel;
okt_Y1out: TLabel;
okt_a: TLabel;
okt_reldif: TLabel;
best5: TButton;
procedure Button2Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure best5Click(Sender: TObject);
procedure FormActivate(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  procedure ShowForm4(FormClass: TFormClass);
  procedure Calculate_Dimensions;
end;

type
  dimensie_array = array[1..100] of extended;
  best_rec = record
    xt,yt,x1,y1,area,rd : extended;
  end;
  best_array = array[1..5] of best_rec;

var
  Form4: TForm4;
  wafer, mask : dimensie_array;
  herhaal : extended;           {stepsize of mask grid}
  besttrans : best_array;

implementation

```

```

uses cjf_best;

{$R *.DFM}

procedure TForm4.ShowForm4(FormClass: TFormClass); {Procedure by
  David Berneda, Delphi 3.0 Demo Library}
begin
  With FormClass.Create(Self) do
    try
      {Protect Resource-using code / Beskerm
      hulpbronroepende kode}
      ShowModal; {Open window / Maak venster oop}
    finally
      {Free resources, even on exception / Bevry
      hulpbronne, selfs na "exception"}
      Free;
    end;
  end;

procedure TForm4.Button2Click(Sender: TObject);
begin
  Close;
end;

procedure TForm4.Calculate_Dimensions;

var
  jj_area, jj_critical_current, jj_current_density : extended;
  vierx, viery, vierarea, sirkarea, sirkdiam, sirkfout1, sirkfout2 :
    extended;
  vierfout1, vierfout2, vierfout_carry : extended;
  oktfout_carry, oktXT, oktYT, oktX1, oktY1, oktarea, oktfout1,
    oktfout2 : extended;
  gevind : boolean;
  indeksx, indeksy, teller1, teller2, diepte, telonder, telbo :
    integer;
  telbox1, telonderx1, x1diepte, teller_yt, teller_x1 : integer;
  telonderyt, telboyt, teller3, teller4, telondery1, telboyl :
    integer;
  indeksxt, indeksyt, indeksx1, indeksy1, caltel : integer;
  x1, oktX1step, oktY1step : extended;
  indeks : longint;

  {----- subprocedure of
  TForm4.Calculate_Dimensions}
procedure swopbeste(sxt, syt, sx1, sy1 : integer);

var
  wyser, nommer, stel : integer;
  sfout, oppv, sX1step, sY1step : extended;

begin
  sX1step := herhaal*wafer[sxt]/mask[sxt]; {calculate WAFER-step size
  for X1}
  sY1step := herhaal*wafer[syt]/mask[syt]; {calculate WAFER-step size
  for Y1}
  oppv := wafer[sxt]*wafer[syt]-2*sX1step*sx1*sY1step*sy1;
  sfout := (oppv/jj_area-1)*100;
  nommer := 10;
  for stel := 5 downto 1 do
    if abs(sfout) < abs(besttrans[stel].rd) then
      begin
        nommer := stel;

```

```

    end;
  if nommer < 5 then
    for wyser := 5 downto (nommer+1) do
      begin
        besttrans[wyser] := besttrans[wyser-1];
      end;
    if nommer < 6 then
      begin
        besttrans[nommer].xt := mask[sxt];
        besttrans[nommer].yt := mask[syt];
        besttrans[nommer].x1 := herhaal*sx1;
        besttrans[nommer].y1 := herhaal*sy1;
        besttrans[nommer].area := oppv;
        besttrans[nommer].rd := sfout;
      end;
    end;
  end;
  {-+----- end of
    subprocedure}

begin
  for caltel := 1 to 5 do
    begin
      {initialization}
      besttrans[caltel].xt := 0;
      besttrans[caltel].yt := 0;
      besttrans[caltel].x1 := 0;
      besttrans[caltel].y1 := 0;
      besttrans[caltel].area := 0;
      besttrans[caltel].rd := 100;
    end;
    jj_critical_current := StrToFloat(jj_critcurrent.Text);
    jj_current_density := StrToFloat(jj_curdens.Text);
    jj_area := jj_critical_current*100/jj_current_density;
    Edit2.Text := FloatToStr(jj_area);
    diepte := StrToInt(opt_depth.Text);    {Read step deviation from
      smallest circumference area}
    xldiepte := StrToInt(x1_dev.Text);
    vierx := sqrt(jj_area);                {Calculate parameters for
      Rectangular Junction}
    gevind := false;
    teller1 := 2;
    while not gevind do                    {Find best length for X}
      begin
        if wafer[teller1] > vierx then
          begin
            indeks := teller1;
            gevind := true;
          end;
        inc(teller1);
      end;
    if abs(wafer[indeks]-vierx) > abs(wafer[indeks-1]-vierx) then
      indeks := indeks-1;                  {Select X value closest to
      SRT(jj_area)}
    telonder := indeks - diepte;           {Fix lower limit for
      optimization}
    if telonder < 1 then
      telonder := 1;                       {Can't go beneath 1}
    telbo := indeks + diepte;              {Fix upper limit for
      optimization}
    if telbo > 100 then
      telbo := 100;                        {Limited to 100}
    vierfout_carry := jj_area;             {Initialize value...}

```

```

for teller1 := telonder to telbo do
  begin                                     {Step through allowed values
    for X}
    viery := jj_area/wafer[teller1]; {Calculate precise value of Y
    required}
    gevind := false;
    teller2 := 2;
    while not gevind do                   {Find best length for Y from
    wafer-array}
      begin
        if wafer[teller2] > viery then
          begin
            indeks := teller2;
            gevind := true;
          end;
          inc(teller2);
        end;
        vierfout1 := jj_area-(wafer[teller1]*wafer[indeks-1]);
        {Calculate error-values}
        vierfout2 := jj_area-(wafer[teller1]*wafer[indeks]);
        if abs(vierfout2) > abs(vierfout1) then {Compare
        error-values}
          begin
            if ((teller1 = telonder) or (abs(vierfout1) <
            vierfout_carry))
              then {First optimization-cycle: no previous value..., OR
              error < stored value...}
                begin
                  vierfout_carry := abs(vierfout1);
                  indeksx := teller1;
                  indeksy := indeks-1;
                end;
              end
            else
              begin
                if ((teller1 = telonder) or (abs(vierfout2) <
                vierfout_carry))
                  then {First optimization-cycle: no previous value...,
                  OR error < stored value...}
                    begin
                      vierfout_carry := abs(vierfout2);
                      indeksx := teller1;
                      indeksy := indeks;
                    end;
                  end;
                end;
              end;
            end;
            {Output calculated parameters to screen}
            vier_xout.Caption := 'X = ' + FloatToStr(mask[indeksx])+ 'um';
            vier_yout.Caption := 'Y = ' + FloatToStr(mask[indeksy])+ 'um';
            vierarea := wafer[indeksx]*wafer[indeksy];
            vier_a.Caption := 'Wafer Area = ' + FloatToStrF(vierarea,
            ffNumber,5,3)+' square um';
            vier_reldif.Caption := 'Relative difference from required Area = '
            + FloatToStrF((vierarea/jj_area-1)*100,ffNumber,5,3)+'%';

            {Calculate parameters for "squashed" octagonal junction}
            oktXT := sqrt(jj_area/(1-(2/sqr(2+sqrt(2))))); {Optimal value for
            XT for smallest circumference}
            gevind := false;
            teller1 := 2;
            while not gevind do           {Find best length for XT}

```



```

begin
  if wafer[teller1] > oktXT then
    begin
      indeks := teller1;
      gevind := true;
    end;
    inc(teller1);
  end;
  if abs(wafer[indeks]-oktXT) > abs(wafer[indeks-1]-oktXT) then
    indeks := indeks-1;           {Select XT value closest to
    least-circumference (perfect oct.) value}
  telonder := indeks - diepte;    {Fix lower limit for
  optimization}
  if telonder < 1 then
    telonder := 1;                {Can't go beneath 1}
  telbo := indeks + diepte;       {Fix upper limit for
  optimization}
  if telbo > 100 then
    telbo := 100;                {Limited to 100}
  oktfout_carry := jj_area;      {Initialize value...}
  indeksxt := 1;
  indeksyt := 1;
  indeksx1 := 1;
  indeksy1 := 1;
  for teller1 := telonder to telbo do
    begin                          {Step through allowed values
    for XT}
      oktX1 := mask[teller1]/(2+sqrt(2)); {Calculate optimum value
      for X1 from XT's MASK-value!!}
      indeks := trunc(oktX1 / herhaal);
      (* gevind := false;
      teller2 := 2;
      while not gevind do          {Find closest length for X1
      from wafer-array}
        begin
          if wafer[teller2] > oktX1 then
            begin
              indeks := teller2-1;    {X1 not allowed to exceed
              XT/(2+sqrt2) }
              gevind := true;
            end;
            inc(teller2);
          end; *)
      telbox1 := indeks;           {fix upper limit on x1}
      telonderx1 := telbox1-x1diepte; {fix bottom limit on x1}
      oktX1step := herhaal*wafer[teller1]/mask[teller1]; {calculate
      WAFER-step size for X1}
      if telonderx1 < 1 then
        telonderx1 := 1;
      for teller_x1 := telonderx1 to telbox1 do
        begin                          {Step through allowed values
        for X1}
          x1 := oktX1step*teller_x1;
          oktYT := jj_area/(wafer[teller1]-2*x1+2*x1*(1-
          1*(x1/wafer[teller1])));
          gevind := false;
          teller3 := 2;
          while not gevind do        {Find closest length for
          YT from wafer-array}
            begin
              if wafer[teller3] > oktYT then

```

```

        begin
            indeks := teller3;          {X1 not allowed to exceed
            XT/(2+sqrt2) }
            gevind := true;
        end;
        inc(teller3);
    end;
    telonderyt := indeks-1;           {fix lower limit on YT}
    telboyt := indeks;               {fix upper limit on YT}
    for teller_yt := telonderyt to telboyt do
        begin                          {Step through allowed values
vir YT}
            oktY1 := (jj_area-wafer[teller_yt]*(wafer[teller1]-
2*x1)
                -2*x1*wafer[teller_yt])/(-2*x1); }
            oktY1 := mask[teller_yt]/(2+sqrt(2));

            (*
                gevind := false;
                teller4 := 2;
                while not gevind do
                    begin                {Find closest values for
Y1}
                        if wafer[teller4] > oktY1 then
                            begin
                                indeks := teller4;
                                gevind := true;
                            end;
                            inc(teller4);
                        end; *)
            indeks := trunc(oktY1 / herhaal);
            telondery1 := indeks-1;
            telboyl := indeks;
            if telondery1 < 1 then
                telondery1 := 1;
            oktY1step := herhaal*wafer[teller_yt]/mask[teller_yt];
{calculate WAFER-step size for Y1}
            oktfout1 := jj_area-(wafer[teller_yt]*(wafer[teller1]-
2*x1)+
                2*x1*(wafer[teller_yt]-telondery1*oktY1step));
{Calculate error-values}
            Swopbeste(teller1,teller_yt,teller_x1,telondery1);
            oktfout2 := jj_area-(wafer[teller_yt]*(wafer[teller1]-
2*x1)+
                2*x1*(wafer[teller_yt]-telboyl*oktY1step));
{Calculate error-values}
            Swopbeste(teller1,teller_yt,teller_x1,telboyl);
            if abs(oktfout2) > abs(oktfout1) then
{Compare error-values}
                begin
                    if abs(oktfout1) < oktfout_carry then
                        begin
                            oktfout_carry := abs(oktfout1);
                            indeksxt := teller1;
                            indeksx1 := teller_x1;
                            indeksyt := teller_yt;
                            indeksy1 := telondery1;
                        end;
                    end
                else
                    begin
                        if abs(oktfout2) < oktfout_carry then

```

```

        begin
            oktfout_carry := abs(oktfout2);
            indeksxt := teller1;
            indeksx1 := teller_x1;
            indeksyt := teller_yt;
            indeksy1 := telboy1;
        end;
    end;
end;
end;
end;
    {Output calculated parameters to screen}
oktX1step := herhaal*wafer[indeksxt]/mask[indeksxt]; {calculate
    WAFER-step size for X1}
oktY1step := herhaal*wafer[indeksyt]/mask[indeksyt]; {calculate
    WAFER-step size for Y1}
okt_XTOut.Caption := 'XT = ' + FloatToStr(mask[indeksxt])+ 'um';
okt_YTOut.Caption := 'YT = ' + FloatToStr(mask[indeksyt])+ 'um';
okt_Xlout.Caption := 'x1 = ' +
    FloatToStrF(indeksx1*herhaal, ffNumber, 5, 3)+ 'um';
okt_Ylout.Caption := 'y1 = ' +
    FloatToStrF(indeksy1*herhaal, ffNumber, 5, 3)+ 'um';
oktarea := wafer[indeksyt]*(wafer[indeksxt]-2*indeksx1*oktX1step)+
    2*indeksx1*oktX1step*(wafer[indeksyt]-
    indeksy1*oktY1step);
okt_a.Caption := 'Wafer Area = ' + FloatToStrF(oktarea,
    ffNumber, 5, 3)+ ' square um';
okt_reldif.Caption := 'Relative difference from required Area = '
    + FloatToStrF((oktarea/jj_area-1)*100, ffNumber, 5, 3)+'%';

sirkdiam := 2*sqrt(jj_area/(2*pi));    {Calculate circle diameter}
gevind := false;
teller1 := 2;
while not gevind do
    begin
        if wafer[teller1] > sirkdiam then
            begin
                indeks := teller1;
                gevind := true;
            end;
            inc(teller1);
        end;
sirkfout1 := jj_area - 2*pi*sqr(0.5*wafer[indeks-1]);    {Calculate
    error-values}
sirkfout2 := jj_area - 2*pi*sqr(0.5*wafer[indeks]);
if abs(sirkfout2) > abs(sirkfout1) then    {Compare error-
    values}
    begin
        {Output calculated parameters to screen}
        sirk_d.Caption := 'Diameter (d) = ' + FloatToStr(mask[indeks-
            1])+ 'um';
        sirk_r.Caption := 'Radius (r) = ' + FloatToStr(mask[indeks-
            1]/2)+ 'um';
        sirkarea := 2*pi*sqr(0.5*wafer[indeks-1]);
        sirk_a.Caption := 'Wafer Area = ' + FloatToStrF(sirkarea,
            ffNumber, 5, 3)+ ' square um';
        sirk_reldif.Caption := 'Relative difference from required Area = '
            + FloatToStrF((sirkarea/jj_area-1)*100, ffNumber, 5, 3)+'%';
    end
else
    begin
        {Output calculated parameters to screen}

```

```

    sirk_d.Caption := 'Diameter (d) = ' +
FloatToStr(mask[indeks])+ 'um';
    sirk_r.Caption := 'Radius (r) = ' +
FloatToStr(mask[indeks]/2)+'um';
    sirkarea := 2*pi*sqr(0.5*wafer[indeks]);
    sirk_a.Caption := 'Wafer Area = ' +
FloatToStrF(sirkarea,ffNumber,5,3)+' square um';
    sirk_reldif.Caption := 'Relative difference from required
Area = '
    + FloatToStrF((sirkarea/jj_area-1)*100,ffNumber,5,3)+'%';
end;
end;

procedure TForm4.Button1Click(Sender: TObject);
{Calculate optimum dimensions for specified junction area}
var
    teks_string : string;
    data_string : string;
    getalle, klaar : boolean;
    karak, karak2 : char;
    teller1, lynteller : integer;
    Offset_Param : Textfile;

begin
    herhaal := 0.25; {default value in case of fileread-failure}
    assignfile(Offset_Param, 'JC1000.jpr');
    {$I-}
    reset(Offset_Param);
    if IOResult <> 0 then
        begin
            ShowForm4(TForm5); {File I/O error : call warning window}
            exit; {TForm5 declared in UNIT cjf_err1}
        end;
    {$I+}
    {Now proceed to load mask-wafer offsets from file...}
    getalle := false;
    while not (eof(Offset_Param) or getalle) do
        begin
            read(Offset_Param, karak);
            if karak = '%' then
                readln(Offset_Param, teks_string) {commentary line read}
            else
                begin {OK, this line is data}
                    readln(Offset_Param, teks_string);
                    data_string := karak + teks_string; {concatenate char and
teks_string}
                    getalle := true; {jump to next subroutine}
                end;
            end;
        end;
    klaar := false;
    lynteller := 1;
    while not (eof(Offset_Param) or klaar) do
        begin {read data from file}
            karak := data_string[1];
            teks_string := '';
            teller1 := 1;
            while karak <> ',' do {mask / wafer offsets separated by
','}
                begin
                    teks_string := teks_string + karak;
                    inc(teller1);

```

```

    karak := data_string[teller1];
  end;
mask[lynteller] := StrToFloat(teks_string);
teks_string := '';
while not (karak in ['0'..'9', '.']) do
  begin
    inc(teller1);
    karak := data_string[teller1];
  end;
while karak in ['0'..'9', '.'] do
  begin
    teks_string := teks_string + karak;
    inc(teller1);
    karak := data_string[teller1];
  end;
wafer[lynteller] := StrToFloat(teks_string);
teks_string := '';
read(Offset_Param, karak);
if karak = '>' then
  begin
    {Read repeat-size...}
    readln(Offset_Param, teks_string);
    data_string := karak + teks_string; {concatenate char and
teks_string}
    teller1 := 1;
    karak2 := data_string[1]; {we already know that
data_string[0] = '>'}
    while karak2 <> '=' do
      begin
        inc(teller1);
        karak2 := data_string[teller1];
      end;
    teks_string := '';
    while not (karak2 in ['0'..'9', '.']) do
      begin
        inc(teller1);
        karak2 := data_string[teller1];
      end;
    while karak2 in ['0'..'9', '.'] do
      begin
        inc(teller1);
        teks_string := teks_string + karak2;
        karak2 := data_string[teller1];
      end;
    herhaal := StrToFloat(teks_string);
    klaar := true;
  end
else
  begin {OK, this line is still data}
    readln(Offset_Param, teks_string);
    data_string := karak + teks_string; {concatenate char and
teks_string}
  end;
inc(lynteller);
end;
closefile(Offset_Param);
{Pad rest of mask and wafer with appropriate values}
teller1 := lynteller - 1;
while lynteller < 101 do
  begin
    mask[lynteller] := wafer[teller1] + herhaal*(lynteller-
teller1);
  end;
end;

```

```

        wafer[lynteller] := mask[lynteller];
        inc(lynteller);
    end;
    Calculate_Dimensions;
end;

procedure TForm4.best5Click(Sender: TObject);
begin
    ShowForm4(TForm6);    {TForm6 declared in UNIT cjf_best}
end;

procedure TForm4.FormActivate(Sender: TObject);

var
    acttel : integer;

begin
    for acttel := 1 to 5 do
        begin                {initialization}
            besttrans[acttel].xt := 0;
            besttrans[acttel].yt := 0;
            besttrans[acttel].xl := 0;
            besttrans[acttel].yl := 0;
            besttrans[acttel].area := 0;
            besttrans[acttel].rd := 100;
        end;
    end;

end.

```

Source code for damping resistance calculator

```

procedure TForm3.Button1Click(Sender: TObject);
    {Calculates Damping Resistance for One-Junction SQUID}
var
    par_cap, par_jjsize, par_zeta, par_indsquid, par_res : double;
begin
    par_cap := StrToFloat(param_ucap.Text)*1e-15;
    par_jjsize := StrToFloat(param_CC.Text)/StrToFloat(param_UC.Text);
    par_zeta := StrToFloat(param_zeta.Text);
    par_indsquid := StrToFloat(param_indsquid.Text)*1e-12;
    par_res :=
        (1/(2*par_zeta))*sqrt(par_indsquid/(par_jjsize*par_cap));
    param_dampres.Text := FloatToStr(par_res);
end;

```

Appendix B - Matlab source code for ADC verification

A Matlab [45] simulation file (anad.m) was written to emulate the ADC from the point of view of DSP functionality. The Matlab simulation does not take into account any superconducting phenomena or electrical characteristics of the ADC; it simply emulates the way in which the comparator decides between a high and low output, the way bits are fed back to the comparator, and the full adding structure.

The power of the simplified Matlab simulation lies in the speed with which the accuracy of the ADC output can be verified over a large number of output words. The full WRSpice simulation on the entire circuit takes two hours to produce one output word on the available workstation, making it impractical to verify the DSP validity of the ADC. With the Matlab simulation, thousands of output words can be generated almost instantly, and signal processing operations like Fourier transforms can easily be performed on the generated data.

The validity of the Matlab simulation was verified by comparing the first few four-bit words generated by Matlab with that obtained with the full circuit simulation in WRSpice. After a little fine-tuning in the level on which the comparator in the Matlab simulation triggers, the results for both simulations were nearly exactly the same (compare Fig. 5.1 and Fig. 5.2.)

anad.m

```
% Coenrad Johann Fourie
%
% Simulation of oversampling delta sigma modulator for analogue-to-
% digital conversion in RSFQ logic

% frequency = RSFQ clock frequency
frequency =10e9;
tmin=1/frequency;

% analogfrequency = Input frequency
analogfrequency =110e6;

% input signal : x = sin(w*t + phi)
w=2*pi* analogfrequency;

% amplitude is the input signal amplitude
amplitude = 4.65;

% skew is the leveldetector-border between last high and last low...
skew = 0.4;

% number of 4-bit samples
aantal=7;

% sommeerder keeps last level
sommeerder = 8;

figure(1);
```

```

subplot(3,1,2);
hold on;

%Skuiflengte is the number of elements per summing cycle
skuiflengte=8;

DCoffset=7.956; %...random

DRObank=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];

gt = 1; %graph counter

%Feedback (=1 on negative feedback)
fb = 0;

%first 8 cycles are odd...
fb = 2;
d = amplitude*sin(w*tmin);
detect=sign(d+fb-DCoffset-skew*(1-DRObank(1)));
dhou(1) = d*600e-6;
tyd(1) = tmin; %tyd = the time variable
if detect == 0
    c = rand(1);
    if c < 0.5
        detect = 0;
    else
        detect = 1;
    end;
end;
if detect == -1;
    detect = 0;
end;
for drotel=1:14
    DRObank(16-drotel)=DRObank(15-drotel);
end;
DRObank(1) = detect;
fb = 2 - detect;
plot([tmin tmin tmin tmin*2], [0 detect 0 0], 'w');

% cycle 2
d = amplitude*sin(w*2*tmin);
detect=sign(d+fb-DCoffset-skew*(1-DRObank(1)));
dhou(2) = d*600e-6;
tyd(2) = tmin*2; %tyd = the time variable
if detect == 0
    c = rand(1);
    if c < 0.5
        detect = 0;
    else
        detect = 1;
    end;
end;
if detect == -1;
    detect = 0;
end;
for drotel=1:14
    DRObank(16-drotel)=DRObank(15-drotel);
end;
DRObank(1) = detect;
fb=15;
for drotel=1:15

```



```

    fb = fb - DRObank(drotel);
end;
plot([tmin*2 tmin*2 tmin*2 tmin*3], [0 detect 0 0], 'w');

% cycles 3-8
for b = 3 : 8
    d=amplitude*sin((w*b)*tmin);
    detect=sign(d+fb-DCoffset-skew*(1-DRObank(1)));
    tyd(b)=b*tmin;           %tyd = the time variable
    if detect == 0
        c = rand(1);
        if c < 0.5
            detect = 0;
        else
            detect = 1;
        end;
    end;
    if detect == -1;
        detect = 0;
    end;
    for drotel=1:14
        DRObank(16-drotel)=DRObank(15-drotel);
    end;
    DRObank(1) = detect;
    fb = 15;
    for drotel=1:15
        fb = fb - DRObank(drotel);
    end;
    plot([tmin*b tmin*b tmin*b tmin*(b+1)], [0 detect 0 0], 'w');
    dhou(b) = d*600e-6;
end;
sommeerder = 0;
for drotel=1:15
    sommeerder = sommeerder + DRObank(drotel);
end;
woord(1)=sommeerder;

laaste=0;
for a = 1 : (aantal)           % make number of 4-bit words
    c=0;                         % counter
    for b=1:8                   % 8 samples = 4-bit word
        d=amplitude*sin(w*(a*skui lengte+b)*tmin);
        detect=sign(d+fb-DCoffset-skew*(1-DRObank(1)));
        if detect == 0
            c = rand(1);
            if c < 0.5
                detect = 0;
            else
                detect = 1;
            end;
        end;
    end;
    if detect == -1;
        detect = 0;
    end;
    for drotel=1:14
        DRObank(16-drotel)=DRObank(15-drotel);
    end;
    DRObank(1) = detect;
    fb = 15;
    for drotel=1:15
        fb = fb - DRObank(drotel);
    end;
end;

```

```

    end;
    c=(a)*skuiflengte+b;
    plot([tmin*c tmin*c tmin*c tmin*(c+1)], [0 detect 0 0], 'w');
    dhou(a*skuiflengte+b) = d*600e-6;
    tyd(a*skuiflengte+b) = tmin*(a*skuiflengte+b);
    end;
    sommeerder = 0;
    for drotel=1:15
        sommeerder = sommeerder + DRObank(drotel);
    end;
    woord(a+1) = sommeerder;

end;

hold on;
subplot(3,1,1);
plot(tyd,dhou,'w');
subplot(3,1,3);
hold on;
for tel2 = 1 : 27
    plot([tyd(tel2) tyd(tel2+1)], [4 4], 'w');
    plot([tyd(tel2) tyd(tel2+1)], [3 3], 'w');
    plot([tyd(tel2) tyd(tel2+1)], [2 2], 'w');
    plot([tyd(tel2) tyd(tel2+1)], [1 1], 'w');
end;
for tel2 = 1:5 %extract bits
    wrd = woord(tel2);
    bis3(tel2) = 0;
    bis2(tel2) = 0;
    bis1(tel2) = 0;
    bis0(tel2) = 0;
    if (wrd / 2) > 3.99999
        bis3(tel2) = 1;
    end;
    wrd = wrd - bis3(tel2)*8;
    toets = wrd/2;
    if toets > 1.99999
        bis2(tel2) = 1;
    end;
    wrd = wrd - bis2(tel2)*4;
    toets = wrd/2;
    if toets > 0.999999
        bis1(tel2) = 1;
    end;
    wrd = wrd - bis1(tel2)*2;
    if wrd > 0.001
        bis0(tel2) = 1;
    end;
end;
for tel2 = 1:4
    plot([tyd(tel2*8+20) tyd(tel2*8+20) tyd(tel2*8+26) tyd(tel2*8+26)
tyd(tel2*8+28)], [4 (4+0.65*bis3(tel2)) (4+0.65*bis3(tel2)) 4 4], 'w');
    plot([tyd(tel2*8+20) tyd(tel2*8+20) tyd(tel2*8+26) tyd(tel2*8+26)
tyd(tel2*8+28)], [3 (3+0.65*bis2(tel2)) (3+0.65*bis2(tel2)) 3 3], 'w');
    plot([tyd(tel2*8+20) tyd(tel2*8+20) tyd(tel2*8+26) tyd(tel2*8+26)
tyd(tel2*8+28)], [2 (2+0.65*bis1(tel2)) (2+0.65*bis1(tel2)) 2 2], 'w');
    plot([tyd(tel2*8+20) tyd(tel2*8+20) tyd(tel2*8+26) tyd(tel2*8+26)
tyd(tel2*8+28)], [1 (1+0.65*bis0(tel2)) (1+0.65*bis0(tel2)) 1 1], 'w');
end;
plot([tyd(5*8+20) tyd(5*8+20) tyd(5*8+24)], [4 (4+0.65*bis3(5))
(4+0.65*bis3(5))], 'w');

```

```
plot([tyd(5*8+20)    tyd(5*8+20)    tyd(5*8+24)], [3    (3+0.65*bis2(5))  
(3+0.65*bis2(5))], 'w');  
plot([tyd(5*8+20)    tyd(5*8+20)    tyd(5*8+24)], [2    (2+0.65*bis1(5))  
(2+0.65*bis1(5))], 'w');  
plot([tyd(5*8+20)    tyd(5*8+20)    tyd(5*8+24)], [1    (1+0.65*bis0(5))  
(1+0.65*bis0(5))], 'w');  
hold off;
```

Appendix C - Impedance of coplanar waveguide (CPW)

Several different forms of waveguide exist for stripline. Of these, the coplanar waveguide (CPW) is the most widely used [26].

In a CPW, the signal is applied between a centre conductor and two ground strips. Fig. C.1 shows a cross-section of a CPW. The conducting strip has width S and thickness t . The gaps have width W , and the dielectric has height h .

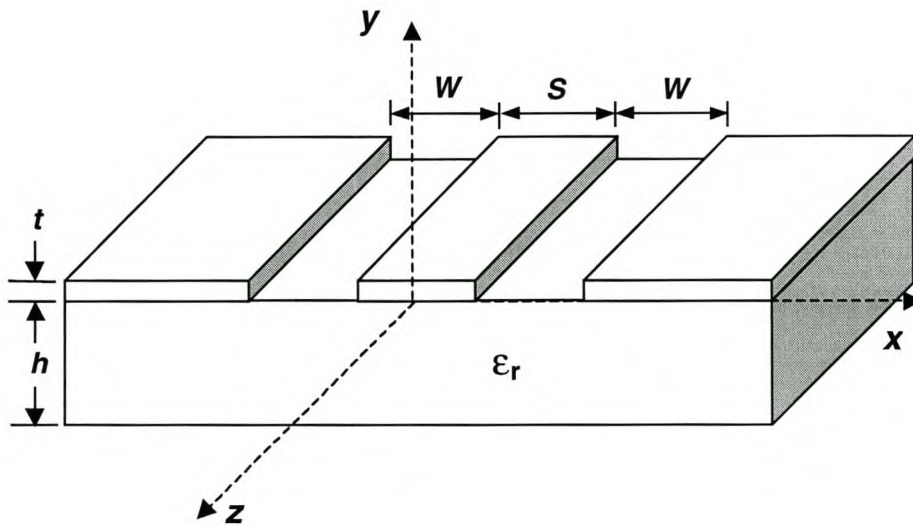


Figure C.1: Coplanar waveguide

The equations describing the impedance of a coplanar waveguide with an infinite dielectric in one half-plane, and air in the other, are derived in [28]. These equations are listed here without going through the derivation. The characteristic impedance of a conventional CPW is

$$Z_{0cp} = \frac{30\pi K'(k_1)}{\sqrt{\epsilon_{re}} K(k_1)} \quad (C.1)$$

where ϵ_{re} is the effective dielectric constant of a CPW and is defined by

$$\epsilon_{re} = \frac{\epsilon_r + 1}{2} \quad (C.2)$$

The expression of the ratio K/K' is [58]

$$\frac{K(k)}{K'(k)} = \frac{\pi}{\ln [2(1 + \sqrt{k'})/(1 - \sqrt{k'})]} \quad \text{for } 0 \leq k \leq 0.707 \quad (C.3)$$

$$\frac{K(k)}{K'(k)} = \frac{1}{\pi} \ln [2(1 + \sqrt{k})/(1 - \sqrt{k})] \quad \text{for } 0.707 \leq k \leq 1 \quad (\text{C.4})$$

Also

$$K'(k_1) = K(k_1') \quad \text{with } k_1' = \sqrt{1 - k_1^2} \quad (\text{C.5})$$

and

$$k_1 = a/b = S/(S + 2W) \quad (\text{C.6})$$

These equations are also accurate to within several percent if the substrate thickness h is finite, but greater than the total width of the gap strips [26].

Appendix D – Monte Carlo simulation files

WRSpice simulation file for RSFQ AND-gate

```

* Generated by Xic from cell rsfq_en_nuut
.monte
#           run Monte Carlo checks
.exec
mplot -on
#           plot Monte Carlo results real-time
checkSTP1=16
checkSTP2=16
#           (2*16+1)*(2*16+1)=1089 MC runs
let Itol = gauss(0.1/3,1)
let Rtol = gauss(0.2/3,1)
let Ltol = gauss(0.05/3,1)
#
.endc
.control
if (t1*70e-12) > 0.5f
    let checkFAIL=1
end
if (t2*70e-12) > 0.5f
    let checkFAIL=1
end
if (t3*70e-12) > 0.5f
    let checkFAIL=1
end
if (t4*70e-12) < 1.3f
    let checkFAIL=1
end
if (t4*70e-12) > 2.5f
    let checkFAIL=1
end
.endc
.tran 1p 410p UIC
.param Avar = Itol*gauss(0.15/3,1)
.param Rvar = Rtol*gauss(0.15/3,1)
.param Lvar = Ltol*gauss(0.2/3,1)
.measure tran t1 from=30p to=100p avg v(51)
.measure tran t2 from=130p to=200p avg v(51)
.measure tran t3 from=230p to=300p avg v(51)
.measure tran t4 from=330p to=400p avg v(51)
B0 33 0 53 jj1 area=$&(0.27*Avar)
B1 31 0 54 jj1 area=$&(0.245*Avar)
B10 38 0 63 jj1 area=$&(0.355*Avar)
B11 10 0 64 jj1 area=$&(0.355*Avar)
B12 9 0 65 jj1 area=$&(0.5*Avar)
B13 7 5 66 jj1 area=$&(0.25*Avar)
B14 6 1 67 jj1 area=$&(0.27*Avar)
B15 3 2 68 jj1 area=$&(0.245*Avar)
B16 1 0 69 jj1 area=$&(0.27*Avar)

```

B17 2 0 70 jj1 area=\$&(0.245*Avar)
B2 32 31 55 jj1 area=\$&(0.245*Avar)
B3 28 33 56 jj1 area=\$&(0.27*Avar)
B4 27 29 57 jj1 area=\$&(0.25*Avar)
B5 19 0 58 jj1 area=\$&(0.251*Avar)
B6 17 14 59 jj1 area=\$&(0.225*Avar)
B7 15 12 60 jj1 area=\$&(0.6*Avar)
B8 11 14 61 jj1 area=\$&(0.225*Avar)
B9 13 0 62 jj1 area=\$&(0.251*Avar)
L0 34 32 \$&(2.77p*Lvar)
L1 18 28 \$&(0.18p*Lvar)
L10 19 40 \$&(0.053p*Lvar)
L11 40 41 \$&(1.64p*Lvar)
L12 43 42 \$&(0.13p*Lvar)
L13 45 44 \$&(0.132p*Lvar)
L14 41 46 \$&(1.64p*Lvar)
L15 46 16 \$&(1.98p*Lvar)
L16 39 47 \$&(1.16p*Lvar)
L17 41 47 \$&(0.053p*Lvar)
L18 39 38 \$&(0.053p*Lvar)
L19 47 48 \$&(0.132p*Lvar)
L2 27 17 \$&(0.66p*Lvar)
L20 46 13 \$&(0.053p*Lvar)
L21 42 15 \$&(0.6p*Lvar)
L22 15 49 \$&(0.5p*Lvar)
L23 14 42 \$&(0.1p*Lvar)
L24 44 50 \$&(1.16p*Lvar)
L25 50 51 \$&(0.82p*Lvar)
L26 49 52 \$&(0.22p*Lvar)
L27 52 44 \$&(0.75p*Lvar)
L28 50 10 \$&(0.132p*Lvar)
L29 52 9 \$&(0.132p*Lvar)
L3 24 39 \$&(0.82p*Lvar)
L30 12 0 \$&(0.026p*Lvar)
L31 7 11 \$&(0.66p*Lvar)
L32 16 6 \$&(0.18p*Lvar)
L33 4 7 \$&(0.1p*Lvar)
L34 5 0 \$&(0.026p*Lvar)
L35 1 4 \$&(2.17p*Lvar)
L36 2 1 \$&(8.474p*Lvar)
L4 21 3 \$&(2.77p*Lvar)
L5 33 30 \$&(2.17p*Lvar)
L6 31 33 \$&(8.474p*Lvar)
L7 29 0 \$&(0.026p*Lvar)
L8 30 27 \$&(0.1p*Lvar)
L9 40 18 \$&(1.98p*Lvar)
R0 8 43 \$&(3.9*Rvar)
R1 8 45 \$&(4.35*Rvar)
R10 38 0 \$&(0.81*Rvar)
R11 23 20 28
R12 0 19 \$&(1.14*Rvar)
R13 17 14 \$&(1.27*Rvar)
R14 51 0 5
R15 15 12 \$&(0.478*Rvar)
R16 11 14 \$&(1.27*Rvar)
R17 13 0 \$&(1.14*Rvar)
R18 48 8 \$&(4.35*Rvar)
R19 10 0 \$&(0.81*Rvar)
R2 37 36 28
R20 9 0 \$&(0.574*Rvar)
R21 7 5 \$&(1.14*Rvar)

```

R22 6 1 $$(1.06*Rvar)
R23 8 2 $$(15.6*Rvar)
R24 3 2 $$(1.17*Rvar)
R25 1 0 $$(1.06*Rvar)
R26 2 0 $$(1.17*Rvar)
R3 33 0 $$(1.06*Rvar)
R4 31 0 $$(1.17*Rvar)
R5 28 33 $$(1.06*Rvar)
R6 8 31 $$(15.6*Rvar)
R7 32 31 $$(1.17*Rvar)
R8 27 29 $$(1.14*Rvar)
R9 26 25 28
V0 37 0 SIN 0 10m 10G 135p
V1 26 0 SIN 0 10m 10G
V2 8 0 DC $$(2.6m*Rtol)
V3 23 0 SIN 0 10m 5G 30p
X0 8 35 34 suny_jt1
X1 8 22 21 suny_jt1
X2 36 35 8 rsfq_dc-sfq_conv
X3 25 24 8 rsfq_dc-sfq_conv
X4 20 22 8 rsfq_dc-sfq_conv
.subckt rsfq_dc-sfq_conv 16 18 10
B0 8 7 11 jj1 area=0.171
B1 6 2 12 jj1 area=0.245
B2 5 4 13 jj1 area=0.148
B3 3 1 14 jj1 area=0.171
L0 15 5 1.27p
L1 7 3 0.29p
L10 2 0 0.13p
L11 1 0 0.18p
L2 4 3 0.69p
L3 16 15 3.35p
L4 15 8 1.29p
L5 9 17 0.08p
L6 6 18 2.11p
L7 17 6 1.74p
L8 7 17 1.13p
L9 5 0 3.59p
R0 8 7 1.68
R1 6 2 1.17
R2 10 9 6.4
R3 5 4 1.94
R4 3 1 1.68
.ends rsfq_dc-sfq_conv
.subckt suny_jt1 4 8 9
B0 2 0 10 jj1 area=$$(0.25*Avar)
B1 1 0 11 jj1 area=$$(0.25*Avar)
L0 6 5 $$(1.98p*Lvar)
L1 6 2 $$(0.132p*Lvar)
L2 7 1 $$(0.132p*Lvar)
L3 8 6 $$(1.98p*Lvar)
L4 3 5 $$(0.132p*Lvar)
L5 7 9 $$(1.98p*Lvar)
L6 5 7 $$(1.98p*Lvar)
R0 4 3 $$(7.4*Rvar)
R1 2 0 $$(1.14*Rvar)
R2 1 0 $$(1.14*Rvar)
.ends suny_jt1
.model jj1 jj(rtype=1, cct=1, icon=10m, vg=2.8m, delv=0.08m,
+ icrit=1m, r0=30, rn=1.64706, cap=3.8065p)
*Nb 1000 A/cm2 area = 100 square microns (generated by JJMODEL)

```


WRSpice simulation file for DRO-COSL interface

```

* Generated by Xic from cell mc_dro_cosl2
.monte
.exec
mplot -on
checkSTP1=16
checkSTP2=16
#
let Jtol = gauss(0.1/3,1)
let Rtol = gauss(0.15/3,1)
let Ltol = gauss(0.1/3,1)
#
let Ic = 250u
let B = 2*pi
let Arg = -1/B
let Acos = -j(ln(Arg+j(sqrt(1-Arg^2))))
let Ith_nom = -(Ic*sin(Acos)+Ic/B*Acos)
let Rbias_nom = 6.5
let Iin_nom = -2.5m/Rbias_nom
let Ratio = Ith_nom/Iin_nom
let L1_nom = 10.3p
let L2_nom = 4.4p
let k = 0.58
let Phi0 = 2.06783461f
let Lone = L1_nom*Ltol
let Ltwo = L2_nom*Ltol
let M = k*sqrt(Lone*Ltwo)*Ltol
let Lj = Phi0/2/pi/0.2/Jtol/1m
let Leff = Lone - M*M/(Ltwo+2*Lj)
let Ic = 0.25*Jtol*1m
let B = 2*pi*Leff*Ic/Phi0
let Arg = -1/B
let Acos = -j(ln(Arg+j(sqrt(1-Arg^2))))
let Ith = -(Ic*sin(Acos)+Ic/B*Acos)
let Rbias = Rbias_nom*Rtol
let Iin = -2.5m/Rbias
let Vtrim = 50*Rtol*(Ith/Ratio-Iin)
.endc
.control
if t1 > 0.4m or t1 > 0.4m or t3 < 0.6m or t4 > 0.4m or t5 < 0.6m
    let checkFAIL=1
end
.endc
.tran 1p 450p UIC
.param Avar = Jtol*gauss(0.1/3,1)
.param Rvar = Rtol*gauss(0.1/3,1)
.param Lvar = Ltol*gauss(0.15/3,1)
.measure tran t1 from=20p to=40p avg v(12)
.measure tran t2 from=120p to=140p avg v(12)
.measure tran t3 from=220p to=240p avg v(12)
.measure tran t4 from=320p to=340p avg v(12)
.measure tran t5 from=420p to=440p avg v(12)
B0 13 0 34 jj1 area=$&(0.2*Avar)
B1 11 0 35 jj1 area=$&(0.2*Avar)
B10 1 2 44 jj1 area=$&(0.2*Avar)
B11 4 0 45 jj1 area=$&(0.245*Avar)
B12 3 0 46 jj1 area=$&(0.27*Avar)
B13 28 29 47 jj1 area=$&(0.171*Avar)

```

```
B14 27 21 48 jj1 area=$&(amp;0.245*Avar)
B15 22 23 49 jj1 area=$&(amp;0.148*Avar)
B16 24 20 50 jj1 area=$&(amp;0.171*Avar)
B2 9 4 36 jj1 area=$&(amp;0.245*Avar)
B3 17 0 37 jj1 area=$&(amp;0.36*Avar)
B4 15 0 38 jj1 area=$&(amp;0.35*Avar)
B5 16 12 39 jj1 area=$&(amp;0.34*Avar)
B6 2 0 40 jj1 area=$&(amp;0.25*Avar)
B7 6 0 41 jj1 area=$&(amp;0.25*Avar)
B8 5 0 42 jj1 area=$&(amp;0.25*Avar)
B9 31 3 43 jj1 area=$&(amp;0.27*Avar)
K1 L2 L4 0.58
K2 L1 L3 0.58
L0 25 31 $&(amp;0.158p*Lvar)
L1 2 14 $&(amp;5.15p*Lvar)
L10 7 9 $&(amp;2.77p*Lvar)
L11 55 6 $&(amp;0.132p*Lvar)
L12 56 5 $&(amp;0.132p*Lvar)
L13 3 1 $&(amp;2.17p*Lvar)
L14 4 3 $&(amp;8.474p*Lvar)
L15 57 22 $&(amp;1.27p*Lvar)
L16 29 24 $&(amp;0.29p*Lvar)
L17 23 24 $&(amp;0.69p*Lvar)
L18 26 57 $&(amp;3.35p*Lvar)
L19 57 28 $&(amp;1.29p*Lvar)
L2 14 0 $&(amp;5.15p*Lvar)
L20 53 58 $&(amp;0.08p*Lvar)
L21 27 25 $&(amp;2.11p*Lvar)
L22 58 27 $&(amp;1.74p*Lvar)
L23 29 58 $&(amp;1.13p*Lvar)
L24 22 0 $&(amp;3.59p*Lvar)
L25 21 0 $&(amp;0.13p*Lvar)
L26 20 0 $&(amp;0.18p*Lvar)
L3 13 12 $&(amp;2.2p*Lvar)
L4 12 11 $&(amp;2.2p*Lvar)
L5 51 54 $&(amp;0.132p*Lvar)
L6 55 54 $&(amp;1.98p*Lvar)
L7 33 55 $&(amp;1.98p*Lvar)
L8 56 7 $&(amp;1.98p*Lvar)
L9 54 56 $&(amp;1.98p*Lvar)
R0 9 4 $&(amp;1.17*Rvar)
R1 32 26 $&(amp;28*Rvar)
R10 12 0 5
R11 2 0 $&(amp;1.5*Rvar)
R12 13 11 $&(amp;1.6*Rvar)
R13 30 51 $&(amp;7.4*Rvar)
R14 52 8 28.001
R15 6 0 $&(amp;1.14*Rvar)
R16 5 0 $&(amp;1.14*Rvar)
R17 30 4 $&(amp;15.6*Rvar)
R18 31 3 $&(amp;1.06*Rvar)
R19 1 2 $&(amp;1*Rvar)
R2 18 17 $&(amp;10*Rvar)
R20 3 0 $&(amp;1.06*Rvar)
R21 4 0 $&(amp;1.17*Rvar)
R22 28 29 $&(amp;1.68*Rvar)
R23 27 21 $&(amp;1.17*Rvar)
R24 30 53 $&(amp;6.4*Rvar)
R25 22 23 $&(amp;1.94*Rvar)
R26 24 20 $&(amp;1.68*Rvar)
R27 59 2 $&(amp;50*Rvar)
```

```

R3 17 0 $$(6*Rvar)
R4 17 16 $$(12*Rvar)
R5 19 16 $$(33*Rvar)
R6 60 15 $$(5*Rvar)
R7 15 0 $$(2.3*Rvar)
R8 16 12 $$(5*Rvar)
R9 15 2 $$(8.8*Rvar)
V0 30 0 DC $$(2.6m*Rtol)
V1 32 0 SIN 0 $$(10m*Rtol) 10G 66.6667p
V2 19 0 DC 5m
V3 18 0 SIN 0 10m 10G
V4 10 0 DC 2.6001m
V5 52 0 SIN 0 10m 5G 100p
V6 59 0 $$(Vtrim)
V7 60 0 SIN 0 10m 10G 66.6667p
X0 8 33 10 rsfq_dc-sfq_conv
.subckt rsfq_dc-sfq_conv 16 18 10
B0 8 7 11 jj1 area=0.171
B1 6 2 12 jj1 area=0.245
B2 5 4 13 jj1 area=0.148
B3 3 1 14 jj1 area=0.171
L0 15 5 1.27p
L1 7 3 0.29p
L10 2 0 0.13p
L11 1 0 0.18p
L2 4 3 0.69p
L3 16 15 3.35p
L4 15 8 1.29p
L5 9 17 0.08p
L6 6 18 2.11p
L7 17 6 1.74p
L8 7 17 1.13p
L9 5 0 3.59p
R0 8 7 1.68
R1 6 2 1.17
R2 10 9 6.4
R3 5 4 1.94
R4 3 1 1.68
.ends rsfq_dc-sfq_conv
.model jj1 jj(rtype=1, cct=1, icon=10m, vg=2.8m, delv=0.08m,
+ icrit=1m, r0=30, rn=1.64706, cap=3.8065p)
*Nb 1000 A/cm2 area = 100 square microns (generated by JJMODEL)

```

HS spice simulation file for DC-to-SFQ converter based on parameter tolerance

```

RSFQ DC-to-SFQ CONVERTER - CJ FOURIE/WJ PEROLD (lumped parameter
implementation)

.lib 'g:\avanti\2000.2\rsfq\jjmodel.lib' jjmodel

.options post=2 probe interp measout delmax=0.5p

.param rtol = gauss(1, 0.15, 3)
.param rdev = gauss(1, 0.10, 3)
.param jt看ol = gauss(1,0.10, 3) $ Jc replaces global area variation

```

```

.param atol = gauss(1,0, 3)    $ No global area variation
.param adev = gauss(1, 0.10, 3)
.param ltol = gauss(1, 0.15, 3)
.param ldev = gauss(1, 0.10, 3)

.param rglob = 'rtol'
.param jglob = 'jtol'
.param aglob = 'atol'
.param lglob = 'ltol'

.param l0_1_27p = '1.27p * lglob'
.param l1_0_29p = '0.29p * lglob'
.param l2_0_69p = '0.69p * lglob'
.param l3_3_35p = '3.35p * lglob'
.param l4_1_29p = '1.29p * lglob'
.param l6_2_11p = '2.11p * lglob'
.param l7_1_74p = '1.74p * lglob'
.param l8_1_13p = '1.13p * lglob'
.param l9_3_59p = '3.59p * lglob'
.param l10_0_13p = '0.13p * lglob'
.param l11_0_18p = '0.18p * lglob'
.param r0_1_68 = '1.68 * rglob'
.param r1_1_17 = '1.17 * rglob'
.param r2_6_4 = '6.4 * rglob'
.param r3_1_94 = '1.94 * rglob'
.param r4_1_68 = '1.68 * rglob'
.param r5_28 = '28 * rglob'

.param a0_171g = '1.71 * aglob'
.param a1_245g = '2.45 * aglob'
.param a2_148g = '1.48 * aglob'
.param a3_171g = '1.71 * aglob'

*Main circuit

L0 3 4 'l0_1_27p * ldev'
L1 10 6 'l1_0_29p * ldev'
L2 5 6 'l2_0_69p * ldev'
L3 2 3 'l3_3_35p * ldev'
L4 3 9 'l4_1_29p * ldev'
L6 12 8 'l6_2_11p * ldev'
L7 11 12 'l7_1_74p * ldev'
L8 10 11 'l8_1_13p * ldev'
L9 4 0 'l9_3_59p * ldev'
L10 14 0 'l10_0_13p * ldev'
L11 7 0 'l11_0_18p * ldev'

Rout 8 0 5
R0 9 10 'r0_1_68 * rdev'
R1 12 14 'r1_1_17 * rdev'
R2 13 11 'r2_6_4 * rdev'
R3 4 5 'r3_1_94 * rdev'
R4 9 10 'r4_1_68 * rdev'
R5 1 2 'r5_28 * rdev'

X0jsq 9 10 jj area='a0_171g * adev' jc='jglob'
X1jsq 12 14 jj area='a1_245g * adev' jc='jglob'
X2jsq 4 5 jj area='a2_148g * adev' jc='jglob'
X3jsq 6 7 jj area='a3_171g * adev' jc='jglob'

*Clocks and bias

```

```
Vclk1 1 0 sin(0 10m 10g 20p)
Vdcbias 13 0 2.6m

*Simulations

.tran 1p 55p 0 0.5p
+ uic sweep monte=100
.probe vout=v(8)
.MEASURE TRAN voutavg AVG V(8) FROM=30P TO 50P
.end
```

HSpice simulation file for DC-to-SFQ converter based on layout extraction

```
RSFQ DC-to-SFQ CONVERTER - CJ FOURIE/WJ PEROLD

.lib 'd:\avanti\2000.2\rsfq\jjmodel.lib' jjmodel
.lib 'd:\avanti\2000.2\rsfq\hypres_libw.lib' hypres_lib

.options acct list post probe delmax=0.5p method=gear

*.op debug all

*JJ statistics

.param jctol = gauss(1,0.15, 3)      $ Jc global variation of JJ
.param jcglob = 'jctol'             $ Make Jc variation offset for
    all JJ's on chip
.param lwjjloc = agauss(0,0.25u,3)  $ L and W local variation of JJ
    in m

*Definition of JJ's

.param l171 = 4.25u                 $ Length in m
.param w171 = 4.25u                 $ Width in m

.param l245 = 5u                    $ Length in m
.param w245 = 5u                    $ Width in m

.param l148 = 3.5u                  $ Length in m
.param w148 = 4.75u                 $ Width in m

.param l171b = 4.25u                $ Length in m
.param w171b = 4.25u                $ Width in m

*Resistor statistics

.param rtol = gauss(1,0.2,3)        $ Global variation of R (sheet
    resistance)
.param rglob = 'rtol'               $ Make R variation offset for all
    R's on chip
.param rloc = agauss(0,0.5u,3)      $ L and W local variation of R in
    m

*Definition of resistors
```

```

.param rl1_68a = 7u           $ Length in m
.param rw1_68a = 5u           $ Length in m
.param rc1_68a = 0.26         $ Contact resistance

.param rl1_17 = 5.5u          $ Length in m
.param rw1_17 = 6u           $ Length in m
.param rc1_17 = 0.26         $ Contact resistance

.param rl1_94 = 8.5u          $ Length in m
.param rw1_94 = 5u           $ Length in m
.param rc1_94 = 0.26         $ Contact resistance

.param rl6_4_1 = 3u           $ Length in m
.param rw6_4_1 = 5u           $ Length in m
.param rc6_4_1 = 0.26         $ Contact resistance

.param rl6_4_2 = 32u          $ Length in m
.param rw6_4_2 = 6u           $ Length in m
.param rc6_4_2 = 0.2         $ Contact resistance

.param rl1_68b = 7u           $ Length in m
.param rw1_68b = 5u           $ Length in m
.param rc1_68b = 0.26         $ Contact resistance

.param rl28 = 167u            $ Length in m
.param rw28 = 6u              $ Length in m
.param rc28 = 0.2             $ Contact resistance

* M1 statistics

.param Mitolh = agauss(0,0.015u,3) $ H global variation of M1 in m
.param Miglobh = 'Mitolh'         $ Make H variation offset for all
                                   M1 layers on chip
.param Mitolt = agauss(0,0.02u,3) $ T global variation of M1 in m
.param Miglobt = 'Mitolt'         $ Make T variation offset for all
                                   M1 layers on chip

.param Milloc = agauss(0,0.25u,3)  $ L and W local variation of M1
                                   in m

* Definition of M1 transmission lines

.param lm1_10_1=10u
.param wm1_10_1=4.5u

.param lm1_10_2=2u
.param wm1_10_2=6u

.param lm1_10_3=2.5u
.param wm1_10_3=9.5u

.param lm1_14_1=2.5u
.param wm1_14_1=26u

.param lm1_14_2=2u
.param wm1_14_2=6u

.param lm1_14_3=10u
.param wm1_14_3=4.5u

```

```
.param lm1_19_1=2.5u
.param wm1_19_1=27u

.param lm1_19_2=2u
.param wm1_19_2=18u

.param lm1_19_3=37.5u
.param wm1_19_3=5u

.param lm1_110_1=2u
.param wm1_110_1=10.5u

.param lm1_111_1=1.5u
.param wm1_111_1=10u

*M2 statistics

.param M2tolh = agauss(0,0.035u,3)  $ H global variation of M2 in m
.param M2globh = 'M2tolh'          $ Make H variation offset for all
                                   M2 layers on chip
.param M2tolT = agauss(0,0.03u,3)  $ T global variation of M2 in m
.param M2globT = 'M2tolT'          $ Make T variation offset for all
                                   M2 layers on chip

.param M2loc = agauss(0,0.25u,3)    $ L and W local variation of M2
                                   in m

* Definition of M2 transmission lines

.param lm2_11_1=7.5u
.param wm2_11_1=11u

.param lm2_12_1=3.5u
.param wm2_12_1=13.5u

.param lm2_12_2=6.5u
.param wm2_12_2=7.5u

.param lm2_12_3=4u
.param wm2_12_3=15.5u

.param lm2_13_1=30.5u
.param wm2_13_1=5u

.param lm2_16_1=35u
.param wm2_16_1=10.5u

.param lm2_17_1=22u
.param wm2_17_1=7.5u

.param lm2_17_2=3.5u
.param wm2_17_2=15u

.param lm2_18_1=5.5u
.param wm2_18_1=32.5u

.param lm2_18_2=11.5u
.param wm2_18_2=6u

*Main circuit
```

**Bias voltages*

Vdc2_6m 13 0 pwl (0 0 1p 0 2p 2.6m)
Vclk1 1 0 sin 0 10m 10g 20p

** DC-SFQ converter circuit*

```
xr28 2 1 rsub lr='rl28' wr='rw28' rcontact='rc28'
xr1_94 4 5 rsub lr='rl1_94' wr='rw1_94' rcontact='rc1_94'
xr1_68a 9 10 rsub lr='rl1_68a' wr='rw1_68a' rcontact='rc1_68a'
xr1_68b 6 7 rsub lr='rl1_68b' wr='rw1_68b' rcontact='rc1_68b'
xr6_4_1 8 11 rsub lr='rl6_4_1' wr='rw6_4_1' rcontact='rc6_4_1'
xr6_4_2 13 8 rsub lr='rl6_4_2' wr='rw6_4_2' rcontact='rc6_4_2'
xr1_17 12 14 rsub lr='rl1_17' wr='rw1_17' rcontact='rc1_17'
Rout 15 0 5

Xjj148 4 5 jj_lws6 ljj='l148' wjj='w148'
Xjj171a 9 10 jj_lws6 ljj='l171' wjj='w171'
Xjj171b 6 7 jj_lws6 ljj='l171b' wjj='w171b'
Xjj245 12 14 jj_lws6 ljj='l245' wjj='w245'

xm1_10_1 3 0 16 0 M1_mstrip length='lm1_10_1' width='wm1_10_1'
xm1_10_2 16 0 17 0 M1_mstrip length='lm1_10_2' width='wm1_10_2'
xm1_10_3 17 0 4 0 M1_mstrip length='lm1_10_3' width='wm1_10_3'
*lm1_10 3 4 1.27p

xm2_11 10 0 6 0 M2_mstrip length='lm2_11_1' width='wm2_11_1'
*lm2_11 10 6 0.29p

xm2_12_1 5 0 18 0 M2_mstrip length='lm2_12_1' width='wm2_12_1'
xm2_12_2 18 0 19 0 M2_mstrip length='lm2_12_2' width='wm2_12_2'
xm2_12_3 19 0 6 0 M2_mstrip length='lm2_12_3' width='wm2_12_3'
*lm2_12 5 6 0.69p

xm2_13 2 0 3 0 M2_mstrip length='lm2_13_1' width='wm2_13_1'
*lm2_13 2 3 3.35p

xm1_14_1 3 0 20 0 M1_mstrip length='lm1_14_1' width='wm1_14_1'
xm1_14_2 20 0 21 0 M1_mstrip length='lm1_14_2' width='wm1_14_2'
xm1_14_3 21 0 9 0 M1_mstrip length='lm1_14_3' width='wm1_14_3'
*lm1_14 3 9 1.29p

xm2_16 12 0 15 0 M2_mstrip length='lm2_16_1' width='wm2_16_1'
*lm2_16 12 15 2.11p

xm2_17_1 11 0 22 0 M2_mstrip length='lm2_17_1' width='wm2_17_1'
xm2_17_2 22 0 12 0 M2_mstrip length='lm2_17_2' width='wm2_17_2'
*lm2_17 11 12 1.74p

xm2_18_1 10 0 23 0 M2_mstrip length='lm2_18_1' width='wm2_18_1'
xm2_18_2 23 0 11 0 M2_mstrip length='lm2_18_2' width='wm2_18_2'
*lm2_18 10 11 1.13p

xm1_19_1 4 0 24 0 M1_mstrip length='lm1_19_1' width='wm1_19_1'
xm1_19_2 24 0 25 0 M1_mstrip length='lm1_19_2' width='wm1_19_2'
xm1_19_3 25 0 0 0 M1_mstrip length='lm1_19_3' width='wm1_19_3'
*lm1_19 4 0 3.59p

xm1_110_1 14 0 0 0 M1_mstrip length='lm1_110_1' width='wm1_110_1'
*lm1_110 14 0 0.13p
```



```

xm1_l11_1 7 0 0 0 M1_mstrip length='lm1_l11_1' width='wm1_l11_1'
*lm1_l11 7 0 0.18p

```

**Simulations*

```

.tran 1p 55p 0 0.5p sweep monte=50
.probe vout=v(15)
.MEASURE TRAN outpls AVG V(15) FROM=30P TO 50P
.end

```

HSPICE library files

The HSPICE library models “jjmodel.lib” and “hypres_libw.lib” were written by W.J. Perold, and are included here only as a complete reference, for they are used extensively in the simulation files for the layout extraction techniques.

.lib jjmodel

** 1 kA/cm² HYPRES process*

** (Area=1 ----> Ic=100u Area=10 um²)*

```

.subckt jj 2 4 area=1 ij=100u rn=26 rg=300 cj=0.4p jc=1
+ vg=2.6m dlv=0.3m phi=0
*rg 1 2 1p
c1 2 4 c= 'cj*area' ctype=1
gr 2 4 vcr pwl(1) 2 4
+ '-1 * vg', 'rn/area/jc'
+ '-1 * (vg - dlv)', 'rg/area/jc'
+ 'vg - dlv', 'rg/area/jc'
+ 'vg', 'rn/area/jc'
gjos 2 4 cur='ij*jc*area*sin(v(3,4)*10k)'
gphi 4 3 cur='v(2,4)'
rphi 4 3 1000g
cphi 3 4 3.291090p ic='phi/10k'
ephi 5 4 vol='v(3,4)*10k'
.ends jj

```

.endl jjmodel

.lib hypres_lib

**Subcircuit descriptions*

**Coupled line model (2 lines)*

```

.subckt coupled_line 1 2 3 4 5 6 cx1=100p cx2=50p cx12=200p lx11=70n
lx22=200n lx12=60n lenx=1u

.param l11dev='LCLglob*LCLloc'
.param l22dev='LCLglob*LCLloc'
.param l12dev='LCLglob*LCLloc'
.param cr1dev='CCLglob*CCLloc'

```

```

.param cr2dev='CCLglob*CCLloc'
.param c12dev='CCLglob*CCLloc'
wel 1 2 3 4 5 6 n=2 l='lenx' rlgcmodel=w4
.model w4 w modeltype=rlgc n=2
+ Lo=
+ 'lx11*l11dev'
+ 'lx12*l12dev' 'lx22*l22dev'
+ Co=
+ 'cx1*cr1dev'
+ '-cx12*c12dev' 'cx12*c12dev+cx2*cr2dev'
.ends coupled_line

*HYPRES M1 layer

.subckt M1_mstrip 1 2 3 4 width=2u length=10u height=0.15u
thick=0.2u M1wlumps=1
+ t0=0.1u lamda=0.09u eps_r=3.9
.param ldevM1='M1loc'
.param wdevM1='M1loc'
.param lamda1='lamda/tanh(t0/lamda)'
.param lamda2='lamda/tanh((thick+M1globt)/lamda)'
.param Kfac='1+4*(height+M1globh)/(width-0.9u+wdevM1)'
.param Lx='1.2566u*((height+M1globh)+lamda1+lamda2)/Kfac/(width-
0.9u+wdevM1)'
.param
Z0x='190.77*(height+M1globh)*sqrt(1+(lamda1+lamda2)/(height+M1globh))
/Kfac/(width-0.9u+wdevM1)'
.param Cx='Lx/Z0x/Z0x'
*elx 5 0 vol='Lx*1e6'
*ezo 7 0 vol='Z0x'
*ecx 6 0 vol='Lx*1e6/Z0x/Z0x'
wel 1 2 3 4 n=1 l='length-0.9u+ldevM1' rlgcmodel=w1
.model w1 w modeltype=rlgc n=1
+ Lo=
+ 'lx'
+ Co=
+ 'cx'
.ends M1_mstrip

*HYPRES M2 layer

.subckt M2_mstrip 1 2 3 4 width=2u length=10u height=0.35u
thick=0.3u M2wlumps=20
+ t0=0.1u lamda=0.09u eps_r=3.9
.param ldevM2='M2loc'
.param wdevM2='M2loc'
.param lamda1='lamda/tanh(t0/lamda)'
.param lamda2='lamda/tanh((thick+M2globt)/lamda)'
.param Kfac='1+3.52*(height+M2globh)/(width-0.5u+wdevM2)'
.param Lx='1.2566u*((height+M2globh)+lamda1+lamda2)/Kfac/(width-
0.5u+wdevM2)'
.param
Z0x='190.77*(height+M2globh)*sqrt(1+(lamda1+lamda2)/(height+M2globh))
/Kfac/(width-0.5u+wdevM2)'
.param Cx='Lx/Z0x/Z0x'
*elx 5 0 vol='Lx*1e6'
*ezo 7 0 vol='Z0x'
*ecx 6 0 vol='Lx*1e6/Z0x/Z0x'
wel 1 2 3 4 n=1 l='length-0.5u+ldevM2' rlgcmodel=w2
.model w2 w modeltype=rlgc n=1
+ Lo=

```

```

+ 'lx'
+ Co=
+ 'cx'
.ends M2_mstrip

*HYPRES M3 layer

.subckt M3_mstrip 1 2 3 4 width=2u length=10u height=0.85u
thick=0.6u
+ t0=0.1u lamda=0.09u eps_r=3.9
.param ldevM3='M3loc'
.param wdevM3='M3loc'
.param lamda1='lamda/tanh(t0/lamda)'
.param lamda2='lamda/tanh((thick+M3globt)/lamda)'
.param Kfac='1+3.96*(height+M3globh)/(width-0.75u+wdevM3)'
.param Lx='1.2566u*((height+M3globh)+lamda1+lamda2)/Kfac/(width-
0.75u+wdevM3)'
.param
Z0x='190.77*(height+M3globh)*sqrt(1+(lamda1+lamda2)/(height+M3globh))
/Kfac/(width-0.75u+wdevM3)'
.param Cx='Lx/Z0x/Z0x'
*elx 5 0 vol='Lx*1e6'
*ezo 7 0 vol='Z0x'
*ecx 6 0 vol='Lx*1e6/Z0x/Z0x'
wel 1 2 3 4 n=1 l='length-0.75u+ldevM3' rlgcmodel=w3
.model w3 w modeltype=rlgc n=1
+ Lo=
+ 'lx'
+ Co=
+ 'cx'
.ends M3_mstrip

*R2 layer

.subckt rsub 1 2 lr=1u wr=1u rcontact=0.3
.param ldev='rloc'
.param wdev='rloc'
rx 1 2 'rglob*(lr+ldev)/(wr+wdev)+2*rcontact'
.ends rsub

*JJ's with widths/lengths<=6u

.subckt jj_lws6 1 2 ljj=3u wjj=3u
.param lwdev='lwjjloc'
.param l1='(1.12*ljj-0.72u)+lwdev'
.param w1='(1.12*wjj-0.72u)+lwdev'
xjj 1 2 jj jc='jcglob' area='l1*w1*1e12/10'
.ends jj_lws6

*JJ's with widths/lengths>6u

.subckt jj_lwg6 1 2 ljj=3u wjj=3u
.param lwdev='lwjjloc'
.param l1='ljj+lwdev'
.param w1='wjj+lwdev'
xjj 1 2 jj jc='jcglob' area='l1*w1*1e12/10'
.ends jj_lwg6

.endl hypres_lib

```

Appendix E – Logic functions and truth tables

As a complete reference, the standard logic operators are listed in Table E.1, along with the corresponding symbols, functions and truth tables.

From Table 4.1 the canonical sum, or sum of minterms for S can be written as

$$S = x' \cdot y' \cdot z + x' \cdot y \cdot z' + x \cdot y' \cdot z' + x \cdot y \cdot z \quad (\text{E.1})$$

Using the theorems in Table E.2 [44], equation (E.1) can be simplified as follows

$$S = z \cdot (x \cdot y + x' \cdot y') + z' \cdot (x \cdot y' + x' \cdot y) \quad (\text{Distributivity}) \quad (\text{E.2})$$

$$= z \cdot (x' \cdot y + x \cdot y')' + z' \cdot (x \cdot y' + x' \cdot y) \quad (\text{De Morgan}) \quad (\text{E.3})$$

$$= z \oplus (x \oplus y) \quad (\text{E.4})$$

Similarly, the canonical sum of C can be written and simplified as follows

$$C = x' \cdot y \cdot z + x \cdot y' \cdot z + x \cdot y \cdot z' + x \cdot y \cdot z \quad (\text{E.5})$$

$$= z \cdot (x' \cdot y + x \cdot y') + x \cdot y \cdot (z + z') \quad (\text{Distributivity}) \quad (\text{E.6})$$

$$= z \cdot (x \oplus y) + x \cdot y \quad (\text{Complements}) \quad (\text{E.7})$$

Table E.1: Definition of standard logic functions

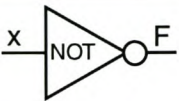
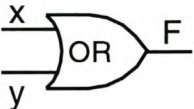
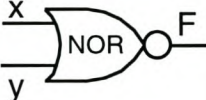
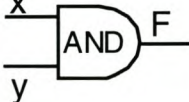
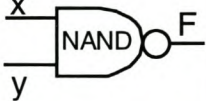

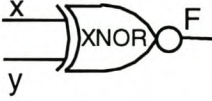
Logic operation (gate)	Graphic symbol	Function	Truth table		
NOT		$F = x'$	X	F	
			0	1	
			1	0	
OR		$F = x + y$	X	Y	F
			0	0	0
			0	1	1
			1	0	1
			1	1	1
NOR		$F = (x + y)'$	X	Y	F
			0	0	1
			0	1	0
			1	0	0
			1	1	0
AND		$F = x \cdot y$	X	Y	F
			0	0	0
			0	1	0
			1	0	0
			1	1	1
NAND		$F = (x \cdot y)'$	X	Y	F
			0	0	1
			0	1	1
			1	0	1
			1	1	0
XOR		$F = x \cdot y' + x' \cdot y$ $= x \oplus y$	X	Y	F
			0	0	0
			0	1	1
			1	0	1
			1	1	0
XNOR		$F = x \cdot y + x' \cdot y'$ $= (x \oplus y)'$	X	Y	F
			0	0	1
			0	1	0
			1	0	0
			1	1	1

Table E.2: Switching-algebra theorems

i	Associativity	$x + (y + z) = (x + y) + z$ $x \cdot (y \cdot z) = (x \cdot y) \cdot z$
ii	Commutativity	$x + y = y + x$ $x \cdot y = y \cdot x$
iii	Distributivity	$x + (y \cdot z) = (x + y) \cdot (x + z)$ $x \cdot (y + z) = x \cdot y + x \cdot z$
iv	Identities	$x + 0 = x$ $x \cdot 1 = x$
v	Complements	$x + x' = 1$ $x \cdot x' = 0$
vi	Idempotency	$x + x = x$ $x \cdot x = x$
vii	Covering	$x + (x \cdot y) = x$ $x \cdot (x + y) = x$
viii	Null elements	$x + 1 = 1$ $x \cdot 0 = 0$
ix	De Morgan	$(x + y)' = x' \cdot y'$ $(x \cdot y)' = x' + y'$
x	Involution	$(x')' = x$