Study and Development on High-Performance Superconductive Computing Systems Using Single-Flux-Quantum Circuits

単一磁束量子回路を用いた高性能超伝導演算システムに関する研究

Ph.D Thesis

XIZHU PENG

[March] [2015]

Thesis supervisors:

Prof. Dr. Nobuyuki Yoshikawa

Department of Physics, Electrical and Computer Engineering

Graduate School of Engineering

Yokohama National University

Abstract

Since the world's first general-purpose electronic computer, ENIAC (Electronic Numerical Integrator And Computer) was finished on Valentine's Day of 1946, more than half a century has passed. Computers no4w involved into nearly every field in modern society, and supercomputers are playing an indispensable role in various research fields or subjects. However, nowadays, supercomputer is facing two walls in front of the way to the exa-scale, the extremely large power dissipation, and the development limit of CMOS fabrication process.

In this thesis, I will introduce the progress and study on implementing a high-end, low power dissipation computing system based on superconducting electronics. The major technique used in this research is single-flux-quantum (SFQ) circuit, which is considered as a promising beyond-CMOS electronic device, for its ultra-high operating speed (dozens GHz-clock rate) and low power consumption (10-3 of CMOS). My research is mainly composed of 2 parts: development of high-end microprocessor, and development of reliable cache memory.

I will demonstrate the high speed operation of SFQ floating-point units (FPUs) at dozens GHz clock rate. These FPUs are core components of Large-scale-reconfigurable-data-path (LSRDP), a novel supercomputing structure proposed to solve "memory wall" problem. Their high-speed operations were successfully confirmed by on-chip high-speed tests, and the maximum frequency was evaluated to be 72 GHz and 59 GHz for half- and single-precision FPM and 58 GHz for single-precision FPA. With these FPUs, the data path is estimated to have performance of 8 TFLOPs and power-performance efficiency of 3000 GFLOPs/W.

I will also introduce the design and implementation of a 64-kb Josephson/CMOS hybrid memory using as L2 cache in the superconducting computing system. Hybrid memory is a compromise of the lack of pure Josephson RAM. Stable JLD array used as Josephson-CMOS interface was designed and demonstrated, and I successfully confirmed the fully-function of the hybrid memory with 2 write/read control lines, 11 address selecting lines and 4 data lines.

I believe this research show the possibility that we could implement a superconducting electronics based super-computing system, which has much higher power efficiency and relatively higher computational capability than CMOS system

adopting available technique. I hope this study could help researcher who works on this issue in the future.

Keywords: supercomputer, superconducting electronics, SFQ logic, LSRDP, memory wall, floating-point unit, Josephson/CMOS hybrid memory.

Acknowledgement

This thesis means the ending of my student years, a long but meaningful period. I'd like to express my appreciation to everyone who has supported me during my master and Ph.D studies in Japan.

At first, I'd like to especially appreciate my advisor, Prof. Dr. Nobuyuki Yoshikawa for his graceful guidance in the superconducting field. When I arrived Japan the first time, I was a brat boy just graduated from the university and could hardly speak any Japanese. It is his patience that lead me pass through those difficult days, and used to the Japanese university. He always give me wonderful advising and discuss with me about the progress in the research, and offer me financial support in my life. If I didn't meet him in my master course, I may not continue the Ph. D studies and won't have such precious years.

Also, I'd like to thank Prof. Dr. Yuki Yamanashi for his strict instructions and kindly encouragement. He is a straight scholar with great integrity, and taught me how to be an eligible researcher in and after my school days.

I wish to thank Dr. Hideo Suzuki as well for the help, suggestion and discussion in my studies. He is full of passion to train the next generation and share us with his significant amount of knowledge and experience. I also thank assistant Hisayoshi Kaneda for his help on experiment equipments.

I'd like to appreciate Prof. Coenrad Fourie in Stellenbosch University, South Africa. He provided us the powerful 3D inductance extractor, InductEx and patiently answered my questions. This tool allowed us to calculate complex inductance structure in acceptable time, significantly reduce the time requires in layout design. Also I'd like to thank Dr. Mark Volkmann, once been the Ph. D student of Prof. Coenrad Fourie, now working in D-wave. We became friends at the conference ISEC 2013, and shared our experience and idea in research and other fields. It is a great pleasure to talk with him.

I also wish to thank Dr. Naoki Takeuchi, now in NICT, soon back to YNU, the excellent Ph. D and senior in Yoshikawa laboratory. He kindly helped juniors and can provide valuable suggestions in details. I have learned some ideas from him and felt very useful in my own subjects. I'd like to thank Mr. Yasuhiro Shimamura and Mr. Taichi Kato, my tutors during my master and Ph. D years. They taught me Japanese and check my research documents, pointed out the grammar mistakes and explained them to

me. I also appreciate Mr. Tao Xue, my cousin and also the senior in Yoshikawa lab. He introduced me to Prof. Yoshikawa and helped my life in Japan. For both brother and senior, he did pretty well.

I warmly express my appreciation to other researchers who have ever provided me useful suggestion and discussion: Prof. Dr. Akira Fujimaki and Prof. Dr. Masamitsu Tanaka in Nagoya University, Dr. Mutsuo Hidaka and Dr. Shuichi Nagasawa in AIST, Dr. Hirotaka Terai in NICT, Prof. Hiroaki Myoren in Saitama University, and Dr. Zhen Wang in SIMIT. And I also sincerely appreciate all members who has been and being in Yoshikawa lab, especially Olivia Xu, the promising Ph. D candidate. She has a smart mind without any constraint, and I wish her a happy marriage and bright future.

I sincerely thank all my friends who has been and being in YNU, Ansel Huang, Ben Liu, Luyi Li, Jiajia Song and Yao Xue. With you, these years were not too boring. I wish to express my appreciation deep in my heart to my best friends, Green Guo, Tracy Huang and Crystal Liang. We have known each other for more than 10 years, when I walk in the darkness alone, when my hopes and dreams have shattered, you are the light and shoulders for me.

Finally, I give my deepest gratitude to my parents and family, for their love and support during my whole life. Perhaps I may not be the proud of my mother, but I am her son forever.

Xizhu Peng

27th January, 2014@Yokohama National University

Contents

List of the Figure

1 Introduction

1.1 Background

The world's first general-purpose electronic computer, ENIAC [1](Electronic Numerical Integrator And Computer) was finished on Valentine's Day of 1946, shortly after World War II. The ENIAC was supported by the US army (cost about 50 million US dollars in 1940s) because they planned to use it in ballistic calculation, and designed by the Moore School of Electrical Engineering, University of Pennsylvania. ENIAC contained 17,468 vacuum tubes, weighed 27 tons, and consumed 150 kW of power. Releasing of the ENIAC greatly shocked the scientific and industrial communities since it was the first fast, programmable "computer". Then computers began to be used to assist scientists in numerical analysis and computation.

Based on the development of semiconductor technology, especially complementary metal oxide semiconductor (CMOS) technology, computers developed in an amazing fast speed during the past half-century and now involved into nearly every field in modern society. Unlike the initial military purpose, computers now support the information society and bring great convenience in people's day-life. Moreover, supercomputers provide quite useful assistant to scientists and play an indispensable role in various fields or subjects, such as molecular analysis, whether forecasting and climate prediction. The requirement of high-end supercomputer is always endless. For instance, in order to survey whether an increase in cloud cover from rising temperatures would retard climate change, and whether a hurricane or tropical storm will generate or not, meteorologists is necessary to simulate the dynamic change of the cloud, the temperature and other factors in global order.

Currently, the world's most powerful supercomputer"tianhe-2" has a peak performance of 33.8 peta-FLOPs (Floating Octal Points)[2], while the practical applications require an exa-FLOPs (10^{18}) performance. . Two major obstacles exist on the way to the exa-scale supercomputers.

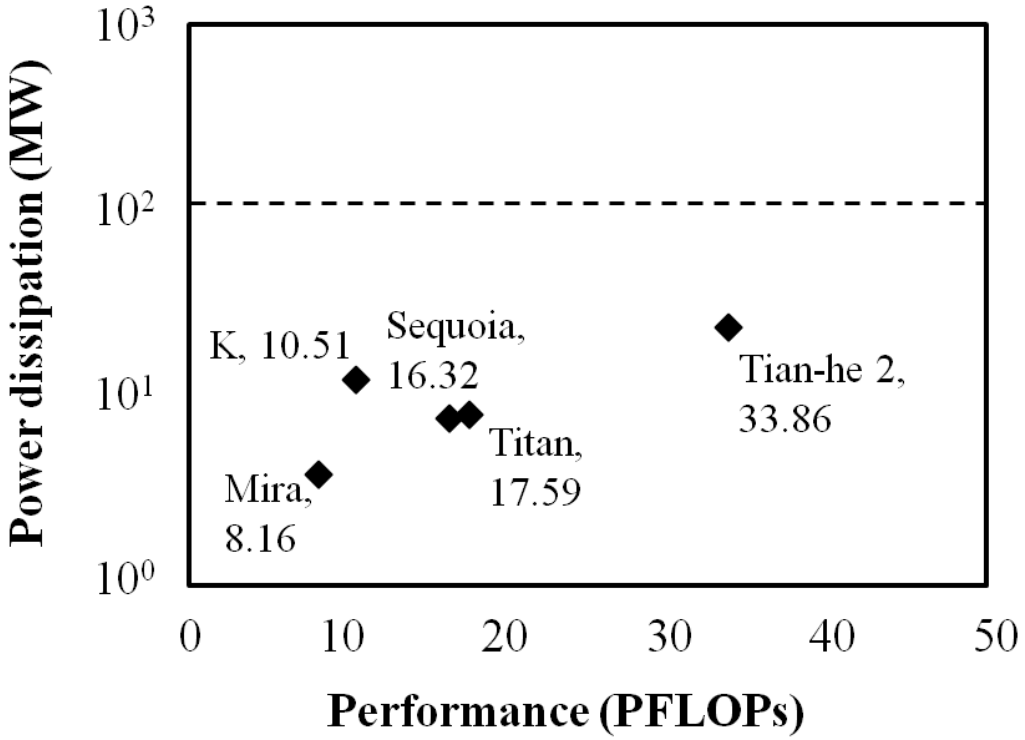


Figure 1-1: The computational performance versus power consumption of the world's top 5 supercomputers.

The first one is the large power dissipation. In personal computers (PCs), the power consumption is far less important that their computational capabilities. However, in supercomputers this is different. **Figure 1-1** shows the performance and power consumption of world's top5 powerful supercomputers [2]. These data clearly show that the power dissipation increases with increasing the computing power. It is estimated to reach around 200 MW eventually for an exa-capability [3]. This is the order of the capacity of a conventional power plant.

Meanwhile, in order to reduce the power consumption, the multi-core structure became majority in computer industry since the year 2002, like IBM's Cell microprocessor, Intel's Core microprocessor and Sun's Rock SPARC microprocessor. This is because the power demanded by a processor is proportional to the cube of its clock frequency. So doubling the frequency of a processor results in an eightfold increase in power by increasing its clock frequency but only twice increase by utilizing

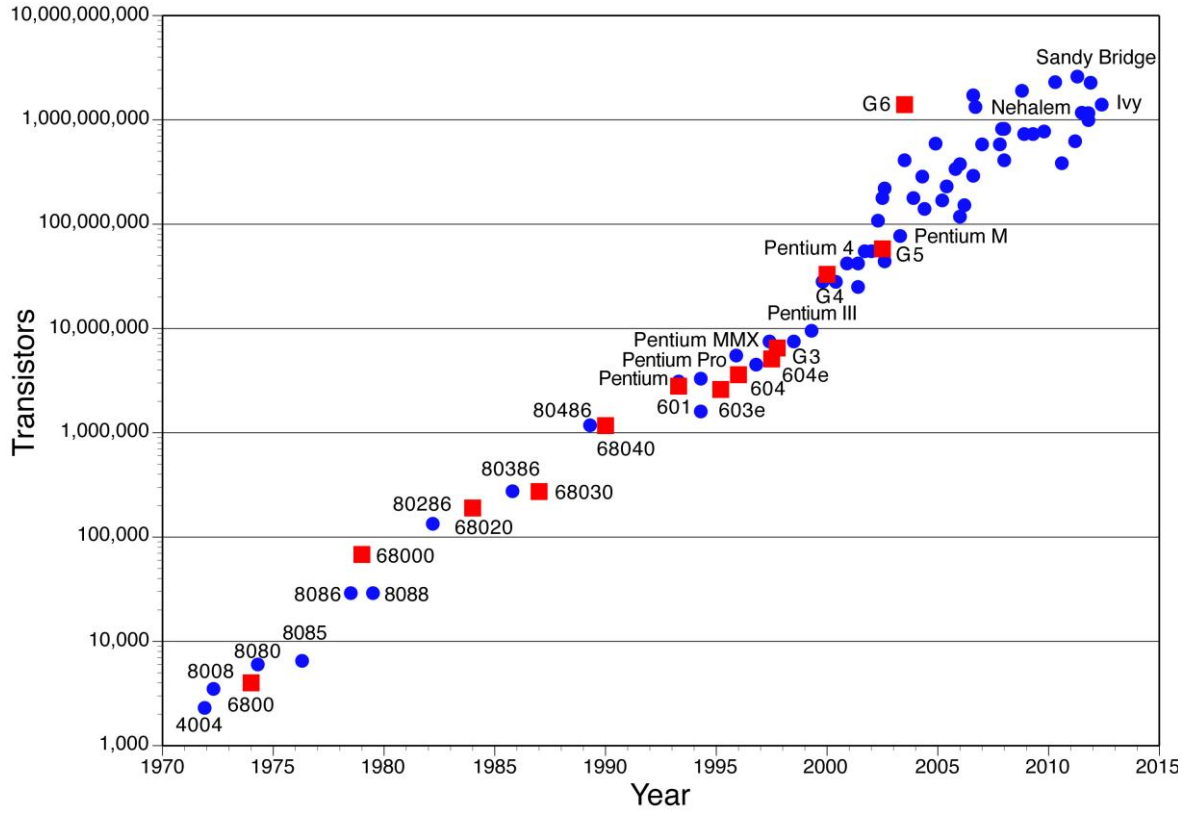


Figure 1-2: The change of transistor counts against dates of released processors.
Source: University of Wisconsin-Madison

duo-core structure. However, multi-core structure leads to an increasing of heat radiation [4], which requires more powerful cooling system for the supercomputer. For example, the cooling consumes 6.4 MW in "Tianhe-2". Moreover, the cooling is not always efficient in various conditions.

Thus, there is little possibility to establish an exa-scale computing system without revolutionary power efficiency improvement. The roadmap of the exa-scale supercomputer is a 20 MW exa-scale system over the technology available today, which means the power efficiency must be more than 50 GFLOP/W[5]. However, the power efficiency of current best supercomputers is less than 2 GFLOP/W, and the problem arises because the currently dominant CMOS digital device consumes too much power. This power is hard to reduce since a minimum switching energy (about $10^6 k_B T$) is required to ensure the reliability, speed, drivability and data communication [6].

Another obstacle comes from the developing limitation of CMOS process. In 1961, Gorden Moore, the co-founder of Intel, made a famous prediction known as Moore's Law. As he predicted, the number of transistors on a single processor die doubles every

18 months. **Figure 1-2** shows the change of transistor counts against dates of released processors.

Moore's Law is based on the rapid development of CMOS process, which allows us to make the size of transistors smaller and smaller, e.g., 180nm, 90 nm, 65 nm, 45nm, 32 nm……Unfortunately, it becomes harder and harder to advance process. Small transistor is too difficult to be manufactured and the cost is very expensive. There is still 1 or 2 decades for the device to be advanced but Moore's law will probably end earlier since the development of speed may be much slower.

In September 5th, 2014, Intel released their "Core M" microprocessor fabricated by the newest 14 nm process. This will prolong the development life of CMOS by several years. However, CMOS process still has a 5 nm limit since the radius of a single atom is 0.2~0.3 nm. And nobody knows whether the 5 nm process could be eventually realized. If the development goes well, this limit will be touched around 2020.

Therefore, in order to continue the development and advance of computer, either for the target of low power consumption to build an exa-scale supercomputer, or for the further development space of integrated circuits (ICs), the next generation of electronics device is necessary. Since late 1960s, people began to research the superconducting logic. In late 1980s, a new superconductor logic family, the rapid single-flux-quantum (RSFQ) logic [7], also called SFQ logic was proposed by the Russian research group. SFQ logic is considered as a promising logic device since its low power dissipation (~$10^4 k_B T$forone switch) and extremely fast operation speed (sub-THz). The two merits provide us the possibility to resolve those obstacles over the exa-scale supercomputer I mentioned above, using SFQ logic. The details of SFQ logic will be introduced in chapter 2.

1.2 Motivation

Since the late 1990s, several intense projects have been started mainly in Japan and US, and vast amount of large-scale SFQ circuits was reported. They include but not limited to the FLUX-1 microprocessor [8, 9], the CORE microprocessor [10-13], and digital RF receiver[14]. Succession of these working shows the possibility to design and implement complex SFQ digital circuits, which could be used as the core processors in superconducting electronics based computing system.

The motivation of this thesis is to implement main components for constructing a

high-end, ultra-low power dissipation computing system based on the SFQ logic. This system will be much more superior to nowadays' semiconductor based supercomputer on the aspect of power efficiency. For the part of processor and data-path, I will discuss about the implementation about floating-point units (FPUs) used in a novel supercomputer data-path structure. Computational capability, integrating density and power efficiency are considered significant in order to compete with CMOS based computer. Additionally, technique and approach on realizing reliable, high-yield large-scale SFQ circuits will be study and discussed.

Memory is the indispensable part of any computer. I also study on the implementation of large capacity cache memory using superconductor/semiconductor hybrid technology for this motivation.

1.3 Outline of the Thesis

My thesis will contains 6 main chapters, their contents are:

- Chapter 1: The introduction of the background of this study
- Chapter 2: Introduce of the operation principle of the SFQ circuit.
- Chapter 3: The design, fabrication and testing progress of this study.
- Chapter 4: In this chapter, I will introduce the implementation of FPUs in details. Algorithm and experimental result is shown in this chapter.
- Chapter 5: I will introduce the implementation of a 64 kb cache memory using superconductor/semiconductor hybrid technology.
- Chapter 6: The summary of this study.

2

Operation Principle of SFQ Circuits

2.1 Introduction

In superconducting loops, magnetic flux is quantized and the minimum magnetic flux unit is a flux quantum, Φ_0 given by$h/2e \approx 2.07 \times 10^{-15}$Wb. Using the presence of Φ_0 in a superconducting loop, we can represent the logic "1" and logic "0". This is where the name "single-flux-quantum" comes from. In this chapter, I will introduce the fundamental operating principle of SFQ circuits, from the basic element Josephson junction (JJ) to logic gate and power consumption of SFQ device.

2.2 Josephson Junction

A Josephson tunnel junction is formed by separating two superconducting electrodes with an insulator like a "sandwich" structure (see **Figure 2-1**). The insulator is thin enough so that electrons can quantum-mechanically tunnel through the barrier. The I/V

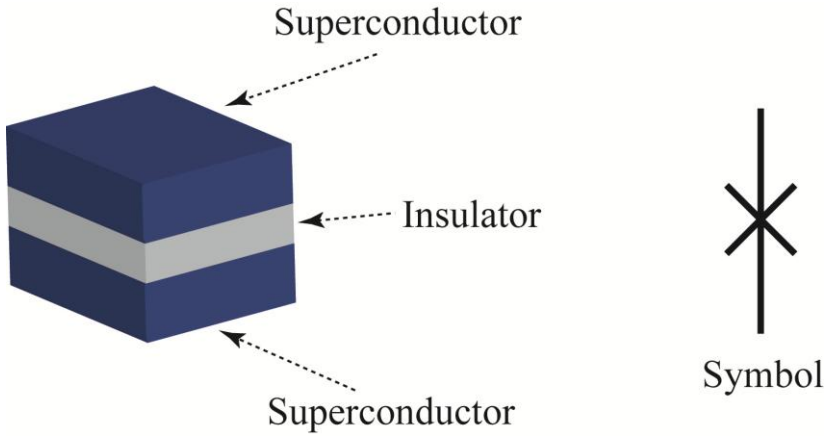


Figure 2-1: Josephson junction and its schematic symbol.

characteristic of Josephson junction can be summarized as Josephson effect.

2.2.1 Josephson Effect

Because of the Quantum tunneling effect, current can flow through the thin insulator in a JJ even without applying any bias voltage. This phenomenon is called **DC Josephson effect**. This current is called Josephson current and given by

$$I = I_c \sin\theta \tag{2.1}$$

Where θ is the phase difference between the superconductors and I_c is the critical current of JJ. If fixed voltage V is applied to the JJ, the phase difference will change alternatively

$$V = \frac{\hbar}{2e} \cdot \frac{d\theta}{dt} = \frac{\Phi_0}{2\pi} \cdot \frac{d\theta}{dt} \tag{2.2}$$

$$\frac{d\theta}{dt} = \frac{2\pi}{\Phi_0} V \tag{2.4}$$

And, using the formula of **DC Josephson effect**, we get

$$I = I_c \sin\left(\frac{2\pi V}{\Phi_0} t + \theta_0\right) \tag{2.5}$$

Which means alternative current flows through the JJ when JJ is supplied by fixed voltage, thus it is called **AC Josephson effect**. Consider a superconductive loop shown in **Figure 2-2**, the magnetic flux in the loop is given by

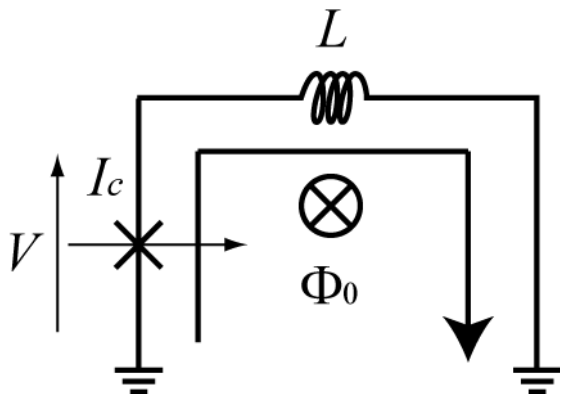


Figure 2-2: Superconductive loop with a JJ

$$\Phi = \int V dt = \int \frac{\Phi_0}{2\pi} \frac{d\theta}{dt} dt$$
$$= \frac{\Phi_0}{2\pi} \int_0^{2\pi} d\theta = \Phi_0 \tag{2.6}$$

Thus we know once the phase difference varies for 2π, a flux quantum Φ_0 will flow into the loop. This is called "switch" of the JJ, the basic function that provide us the cornerstone to manage information and construct digital circuit

When I < I_c the V = 0, this status is called zero voltage stage. When I>Ic the JJ is switched, changed into voltage stage.

2.2.2 McCumber Parameter

A Josephson junction can be analyzed in an equivalent model shown in **Figure 2-3**, where C is the capacitance of JJ, R is sub-gap resistance and L_j is Josephson inductance. Here, L_j is a non-linear parameter given by

$$I = I_c \sin\theta = I_c \sin\left(\int \frac{2\pi}{\Phi_0} V dt\right) = I_c \sin\frac{2\pi}{\Phi_0}\Phi \tag{2.7}$$

$$L_j = \frac{d\Phi}{dI} = \frac{\Phi_0}{2\pi I_c} \frac{1}{\cos\theta}$$
$$= \frac{L_{j0}}{\cos\theta} \quad \left(L_{j0} = \frac{\Phi_0}{2\pi I_c}\right) \tag{2.8}$$

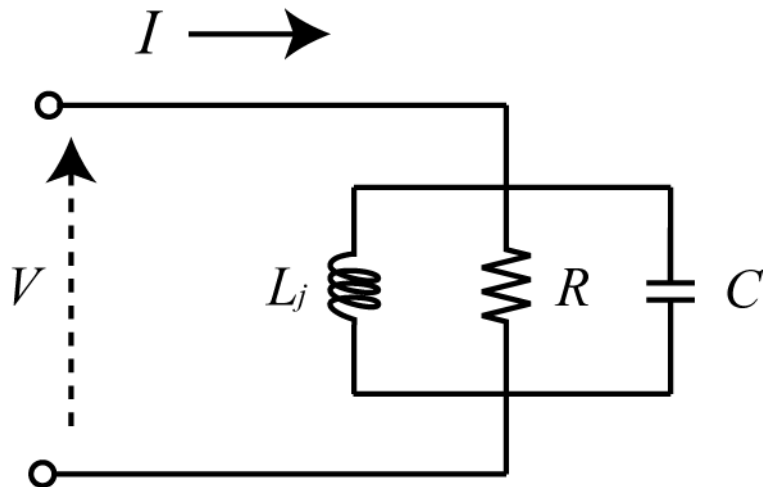


Figure 2-3: Equivalent RLC model of Josephson junction.

Here, L_{j0} is defined as the Josephson inductance at 0 phase difference. The time constant of the RLC parallel resonator is given by

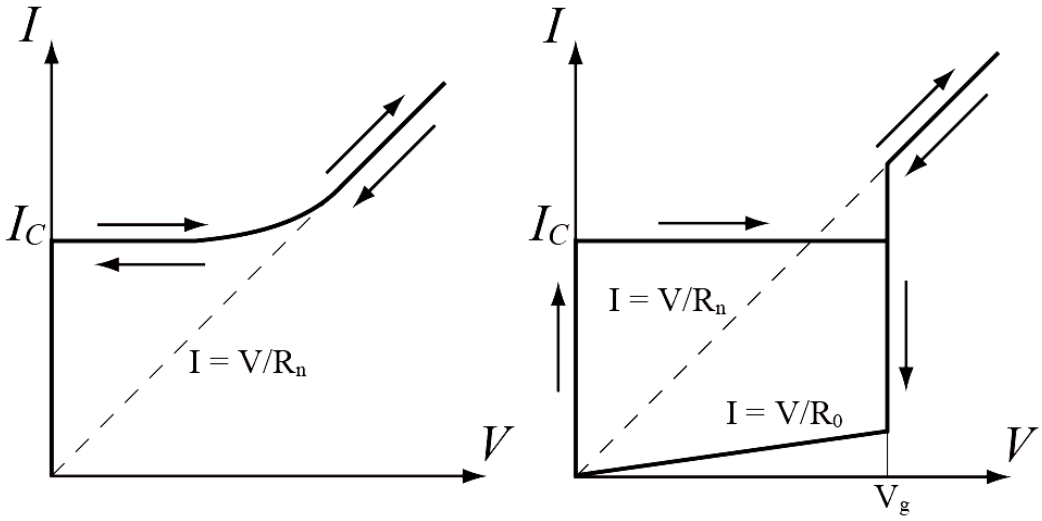
$$\tau_1 = RC, \tau_2 = \frac{L_j}{R} \tag{2.9}$$

And we define the McCumber parameter β_c as

$$\begin{aligned} \beta_c = Q^2 &= \frac{\tau_1}{\tau_2} = \frac{R^2 C}{L_j} \\ &\approx \frac{R^2 C}{L_{j0}} = 2\pi \frac{R^2 C I_c}{\Phi_0} \end{aligned} \tag{2.10}$$

β_c is quite important parameter of JJ since it determines the transient response characteristic of JJ. The switch time of JJ is given by $\tau_w = 2\pi(\tau_1+\tau_2)$ and the minimum is achieved when $\tau_1 = \tau_2$ (Mean Inequality). In this condition, β_c equals to 1 and the resonator corresponds to critical damping.

β_c can be adjusted by applying parallel resistor R_s called shunt resistor. In case of different β_c, I/V characteristic of JJ is different. For β_c around 1, the voltage reduces to 0 very fast once the current drops below critical current I_c. However for β_c much larger than 1, the voltage remains for a short time until the current reduced significantly. This



(a) (b)

Figure 2-4: I/V characteristic of JJ for β_c= 1 and β_c>> 1.
(a) β_c= 1 and (b) β_c>> 1

hysteresis is caused by the over damping of the resonator. Generally, for the high frequency application such as SFQ logic, β_c should be designed near 1. High β_c JJ is employed in various Josephson latching type circuits. I/V characteristic of JJ for β_c= 1 and β_c>> 1 is shown in **Figure 2-4.**

2.3 SFQ Logic Family

The superconducting loop shown in **Figure 2-2** is called SQUID (Superconducting QUantum Interference Device). The SFQ circuit is composed by grouping up several interferometers, and its physical structure is shown in **Figure 2-5**. As I already mentioned, "switch of JJ" corresponds to "phase varing 2π" corresponds to "a Φ_0 is injected into the interferometer", we can use $\Phi_0$as the information carrier and switch as the manipulation of data. In SFQ circuits, presence of $\Phi_0$in the interferometer represents logic "1" and absence of Φ_0 represents logic "0".

Once a JJ is switched, a voltage pulse is generated following

$$\int V dt = \Phi_0 = 2.07 \times 10^{-15} \text{ Wb} \tag{2.11}$$

The Φ_0 carrying information propagates forward to the next interferometer accompanies the voltage pulse, and the voltage pulse (called SFQ pulse hereafter) always corresponds to one Φ_0. It is why the interferometer series is call single-flux-quantum. According to DC Josephson effect, we apply a DC current as bias current to the JJs thus an initial phase difference in JJs is given, in order to ensure the JJ can be switched (phase difference reaches 2π) by the Φ_0. The range of bias current corresponds to the stability of SFQ circuits, which is an extremely significant factor in design.

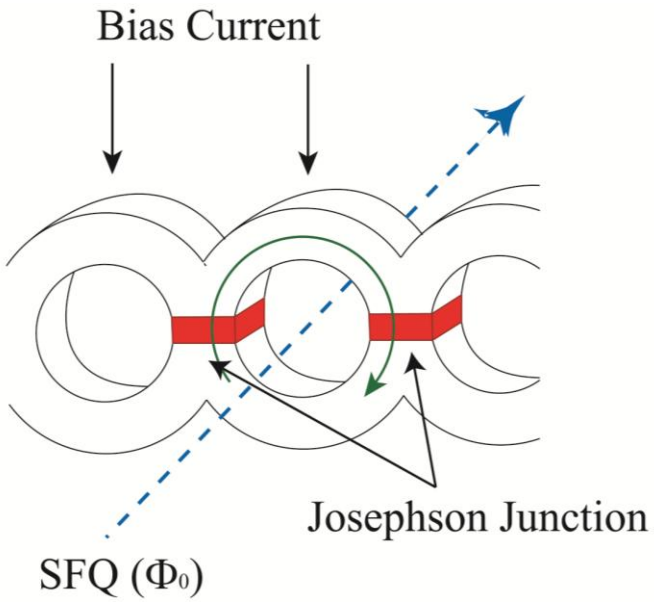


Figure 2-5: Physical structure of SFQ circuit.

Fundamental SFQ Elements

The simplest SFQ element is the Josephson transmission line (JTL), adopted as interconnect and SFQ pulse propagation element. The JTL is simply composed by SQUIDs series as shown in **Figure 2-6**, with small loop inductance L. Operation of JTL could be considered as the following 4 steps:

1). A SFQ pulse arrives and switches J1, a flux quantum $\Phi_0$is generated and transferred in to the loop of J1, L, J2.

2). $I_L=\Phi_0/L$ circles in the loop of J1, L, J2.

3). Since the inductance L is small, I_L will be relatively large. Combines with bias

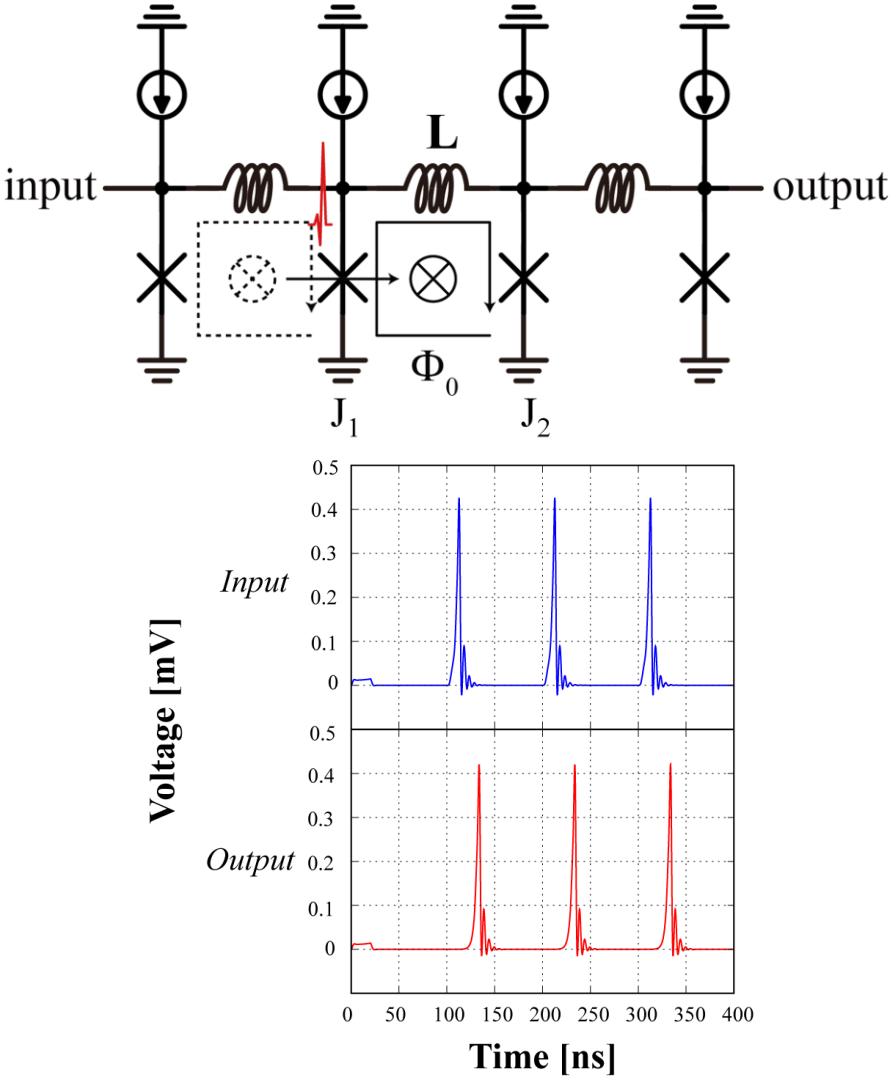


Figure 2-6: Schematic and simulation waveform of JTL

current, it exceeds I_c of J2.

4). J2 is switched. The Φ_0 propagates successfully and this operation repeats in the next stage.

A schematic and simulation waveform of Delay-flip-flop (DFF) is shown in **Figure 2-7**. The structure between din and dout is similar to a JTL, however, the loop inductance L_2 is much larger than that in JTL. The DFF operates in 4 steps:

1). SFQ pulse arrives and switches J1, a flux quantum $\Phi_0$is generated and transferred in to the loop of J1, L_2, J2.

2). Unlike JTL, since L_2 is much larger, the circling current I_L is not enough to

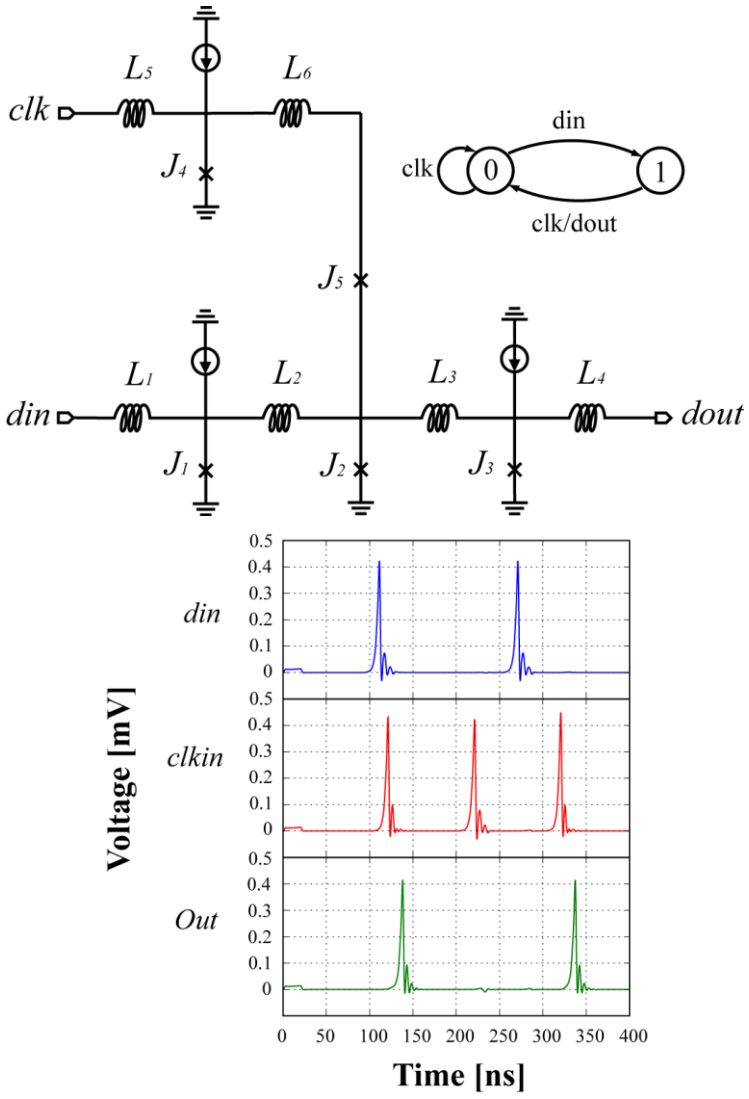


Figure 2-7: Schematic and simulation waveform of DFF

switch J2 even combined with bias current. The $\Phi_0$is stored in the loop.

3). When SFQ pulse arrives and switches J4, an additional circling current is given. The current flows in J2 eventually exceed I_c and switches J2, then $\Phi_0$propagates to J3 and output is obtained.

4). In case of clock arrives alone without din, since I_c of J5 is smaller than J2, J5 will be switch instead of storing Φ_0 in the loop. The J5 is called "escape junction" since it is where the flux quantum escapes from.

The main difference between a JTL and a DFF is the inductance value designed for propagating or storing purpose. Generally, $LI_c = 0.5\ \Phi_0$ is designed for propagating and $LI_c = 1.5\ \Phi_0$ is designed for storing. The JTL and DFF represent 3 fundamental behaviors in SFQ circuits, propagation, store and escape. Complex SFQ logic gates can be designed base on these behaviors by attaching escape JJ, adjusting L and I_c.

2.4 Features of SFQ Circuits

2.4.1 Speed and SFQ Scaling Rule

Maximum clock rate of SFQ circuits is determined by the width of SFQ pulse Δt which is defined as half of the switch time τ_{sw} of JJ[15]. In case of $\beta_c = 1$ ($\tau_1 = \tau_2$), Δt is given by

$$\begin{aligned}\Delta t \equiv \frac{\tau_{sw}}{2} &= \frac{2\pi(\tau_1+\tau_2)}{2} \approx \frac{\Phi_0}{2I_cR} = \sqrt{\frac{\pi\Phi_0 C}{2\beta_c I_c}} \\ &= \sqrt{\frac{\pi\Phi_0 C_0 S}{2\beta_c J_c S}} = \sqrt{\frac{\pi\Phi_0 C_0}{2\beta_c J_c}} \\ &\propto \sqrt{\frac{1}{J_c}}\end{aligned} \tag{2.12}$$

Therefore we know the speed of SFQ circuits is proportional to square root of critical current density J_c. This is the SFQ scaling rule. If we utilize fabricated process with higher critical current density J_c, higher clock rate could be obtained. For example, in AIST standard process with a J_c of 2.5 kA/cm^2, the clock frequency is usually 25-30 GHz. On the other hand in AIST advanced process with a J_c of 10 kA/cm^2, the clock frequency is twice 50-60 GHz.

2.4.2 Power Dissipation of SFQ Circuit

When JJ is switched, it consumes energy of

$$E = IVt = \int_0^\tau I_c V dt = I_c \Phi_0 \tag{2.13}$$

If we assuming a clock rate *f*, then the power dissipation of switching is

$$P_d = I_c \Phi_0 f \tag{2.14}$$

This power dissipation is called dynamic power of SFQ circuits. For a 216 μA JJ used in the JTL cell, assuming the clock rate is 50 GHz, we obtain the dynamic power 22 nW of one JJ, much lower than one transistor of the most advanced CMOS device.

However, in conventional SFQ circuit, static power dissipates as well as dynamic power, since we employed bias resistors to distribute the bias current. For the bias voltage of 2.5 mV, the static power dissipates on one JJ is 375 nW, much higher than that of dynamic power.

This static power seriously deteriorates the power efficiency of SFQ circuits. Until now, several kinds of technique have been proposed and demonstrated to reduce or eliminate this power (i.e.LR biasing [16, 17], ERSFQ[18], eSFQ[19]).

3

Design, Fabricating and Experiment of SFQ Circuits

3.1 Design Guideline of SFQ Circuits

3.1.1 Timing Window

The ultra-high speed feature of SFQ circuits requires very precise timing design. Unlike CMOS logic, instead of using "high" and "low" voltage represent logic bit“1” and “0”, SFQ logic encodes digital bits by the presence of an SFQ pulse in a clock period. Nevertheless, In CMOS logic, the output is determined only by input, which changes directly without an internal state. However in SFQ logic, internal state of an input pulse is changed after it shifts into the input side, where the output is synchronized by the clock pulses. We can treat an SFQ logic gate as a combination of logic gate and DFF. An example of logic in AND gate is shown in **Figure 3-1**.

When a data pulse and clock pulse input into an SFQ logic gate simultaneously, it will probably cause chaos in internal state of the logic gate, result in malfunction of SFQ circuit. Thus, there is an input forbidden period before and after the clock arriving, called “setup time” and “hold time”, respectively. The permission period that data is allowed to arrive called timing window, as shown in **Figure 3-2**.

3.1.2 Clocking Method

The requirement of precise timing design in SFQ logic indicates us the clocking scheme for synchronization must be considered carefully. Generally, there are 3 schemes for clocking.

A. Counter-flow Clocking scheme

In this clocking scheme, clock flows in the opposite direction to that of the data flows, by which conflict in data and clock can be avoided. Also we can get the data with small latency. As a cost, the maximum clock rate is lower than the other 2 schemes, determined by $1/(t_{clk} + t_{data})$.Here t_{clk} is the propagation delay of clock, and t_{data} is the propagation delay of data.

B. Concurrent-flow clocking scheme

In this clocking scheme, clock pulse flows the same direction to that of the data, which implies the clock speed can be very fast and the maximum frequency is determined by $1/(t_{data} - t_{clk})$. However, since conflict of data and clock is inevitable in

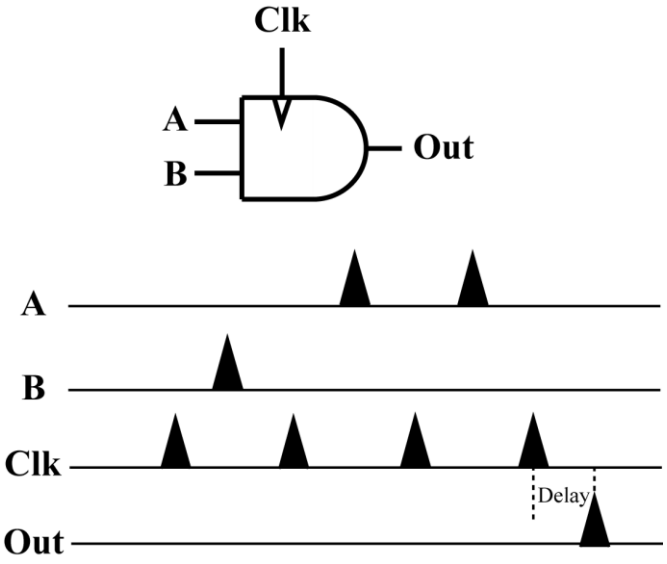


Figure 3-1: Logic characteristic in SFQ AND gate

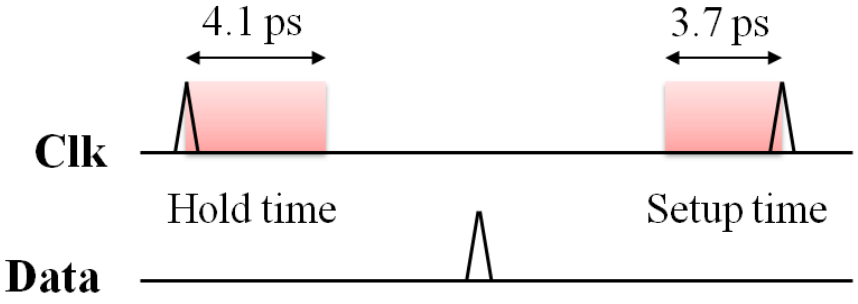
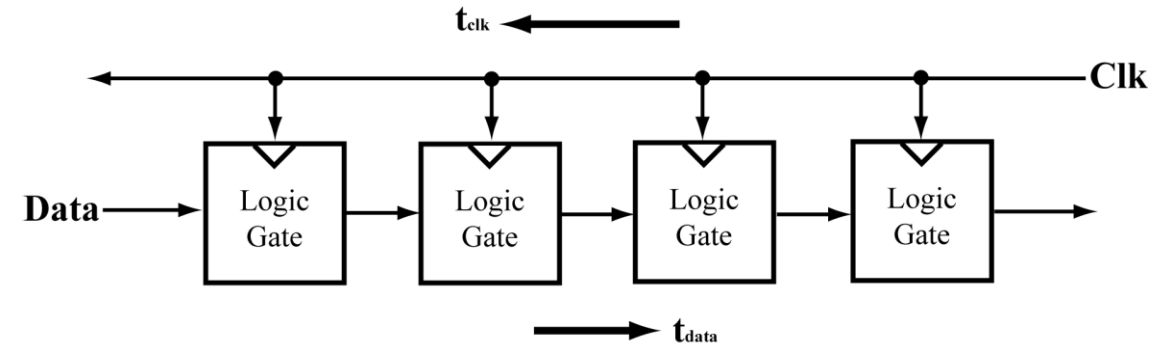


Figure 3-2: Timing window of SFQ exclusive OR gate

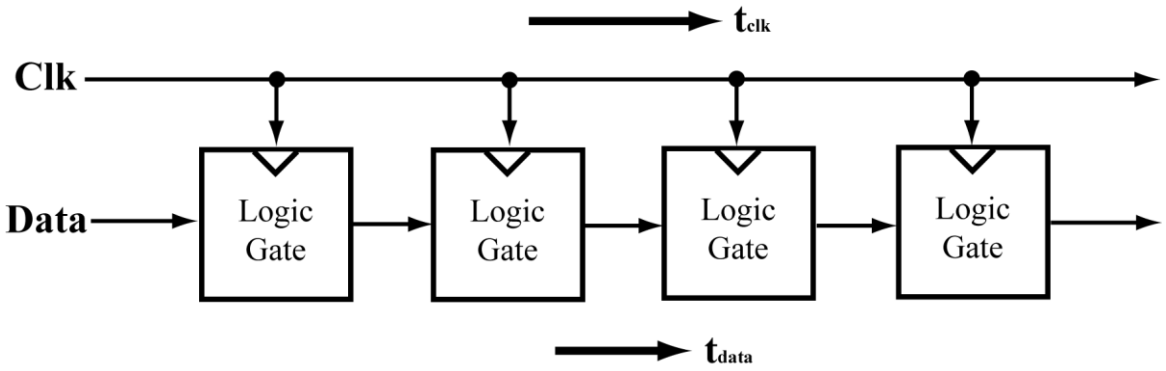
this clock distribution method, one must pay more care in timing design.

C. Clock-flow Clocking scheme

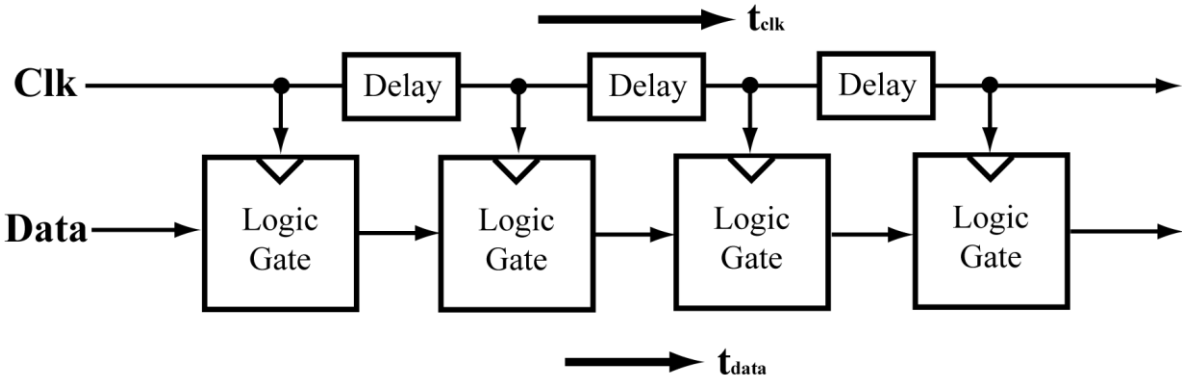
The clock and data propagates in the same direction in this clocking scheme as concurrent-flow scheme. The difference is that the delay elements are inserted in the clock path. Unlike the other two clocking schemes, clock pulses follow data, which allows the data to be output with only one clock similar to CMOS logic. The maximum clock speed is determined by $1/(t_{clk} - t_{data})$as well as concurrent-flow scheme. The imagines of the 3 schemes are shown in **Figure 3-3 (a), (b) and (c)**.



(a)



(b)



(c)

Figure 3-3: Clocking scheme of SFQ logic.

(a) Counter-flow. (b) Concurrent-flow and (c) Clock-flow

3.2 Design Methodology

Since precise timing adjustment is required in SFQ logic design to prevent timing conflict, it is necessary to simulate the logic circuit at cell level. Unfortunately, large-scale SFQ logic circuit usually contains several-dozen thousands of JJ. Analog simulation using simulator such as JSIM (Josephson integrated circuit simulator) [20] or WRspice [21] costs too long time and too much memory and is actually impossible in this case. In order to reduce the simulation time, standard cell-based methodology and top-down design approach[22, 23] is introduced in SFQ logic design.

3.2.1 Standard Cell-based Methodology

In standard cell-based design, the entire logic is divided into basic element cells such as "AND", "OR","DFF" with specifics like timing parameter, logical function, electron

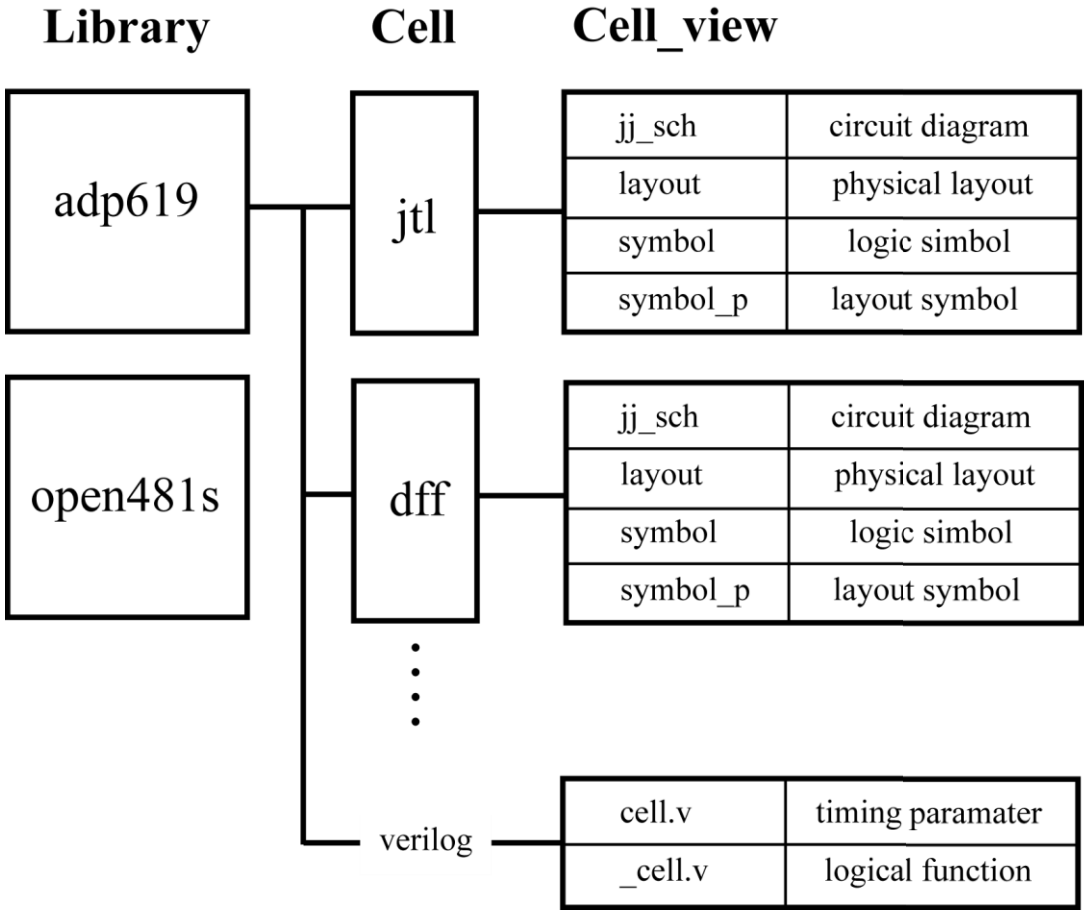
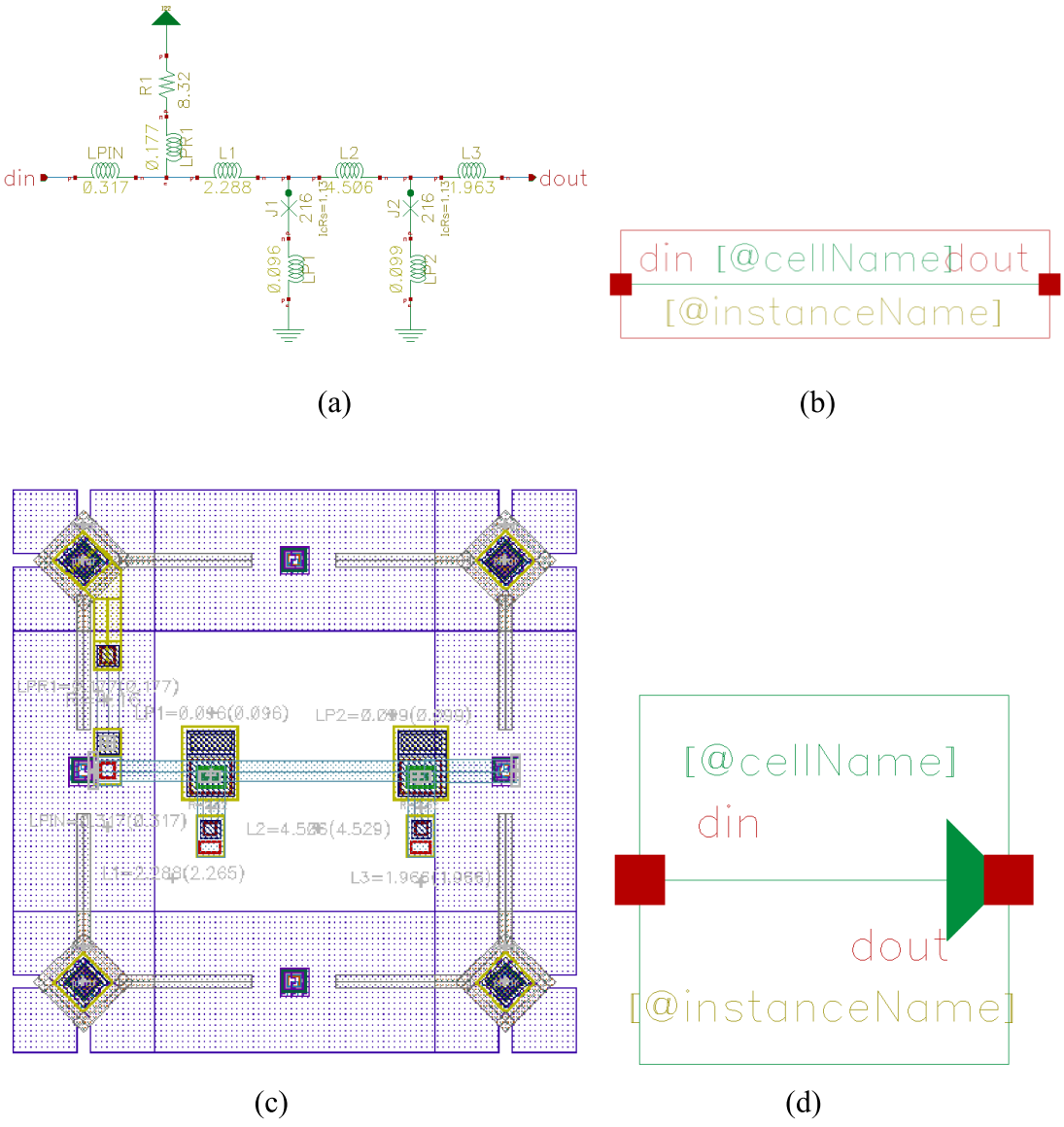


Figure 3-4: Structure of CONNECT cell library

structure, physical layout. This cell group is so called CONNECT cell library [24], developed by the cooperation of Yokohama National University, Nagoya University, ISTEC-SRL and NICT, and contains 390 cells by far. A structure of CONNECT cell library is shown in **Figure 3-4**, and view of a JTL cell in adp619 library is shown in **Figure 3-5**.

By using Cell–based design, high-level designer is avoided to design large-scale circuit



(a) (b)

(c) (d)

Figure 3-5: Cell view of JTL cell in adp619 library.
(a) jj_sch (b) symbol (c) layout and (d) symbol_p

with fundamental element like superconducting interferometer directly, thus they can focus on the logical function design. Moreover, simulation is based on Verilog-HDL [25] only consider the timing parameter and logical function, which lead to a considerable time reduction.

3.2.2 Top-down Design Approach

Top down approach starts with the big picture, and breaks down from there into smaller

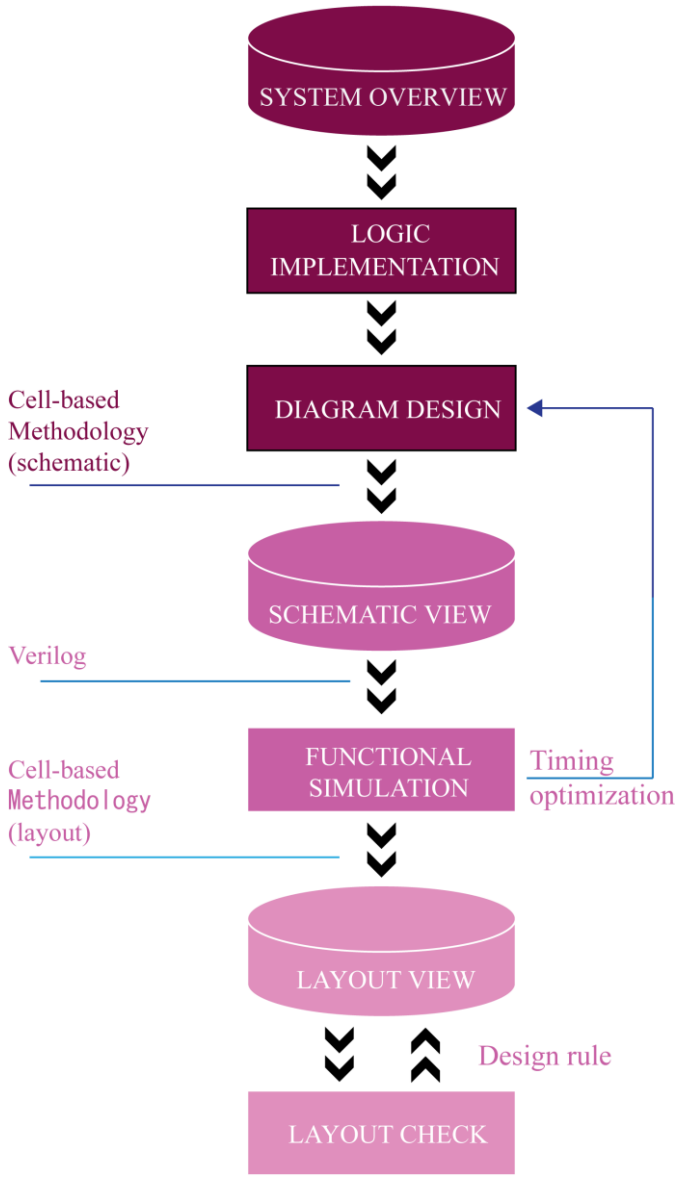


Figure 3-6: Design flow of top-down design approach

segments. In a top-down approach an overview of the system is formulated, specifying but not detailing any first-level subsystems. Each sub-system is then refined in greater detail, sometimes additional subsystem levels is needed until the entire system is reduced to base elements.

Figure 3-6 shows a design flow in top-down approach. At first, we draw the overview picture of the system in terms of function and scale. Then we breakdown this overview to diagram design using the standard cell-based methodology. Functional confirmation is executed by the gate-level simulator (Verilog-HDL), after which we adjust the system timing to optimize the design. A layout view will be generated after the system being optimized finally.

3.3 Fabrication Process

In this research, the circuits were fabricated in the clean room for analog-digital superconductivity (CRAVITY) of AIST (Advanced Industrial Science and Technology) with the standard process 2 (STP2) and advanced process 2 (ADP2). These processes are based on the Nb circuit fabrication process developed at ISTEC.

3.3.1 Standard Process

Figure 3-7 shows the cross-section of a chip fabricated by STP2. Niobium (Nb) is used as superconducting material. Molybdenum (Mo) is used as resistive material. Silicon dioxide (SiO2) is deposited to provide insulation between layers. In this process there are 12 mask layers: GP (Ground Plane), RES (Resister layer), RC (Resister Contact), GC (Ground Contact), JJ (Josephson Junction), JP (Junction Protection pattern), BAS (Base layer), BCC (Base Counter Contact), JCC (Junction Counter Contact), COU (Counter layer), CC (Counter Control Contact) and CTL (Control layer). The critical current density J_c is 2.5 kA/cm^2. The minimum JJ in STP2 is 100 μA.

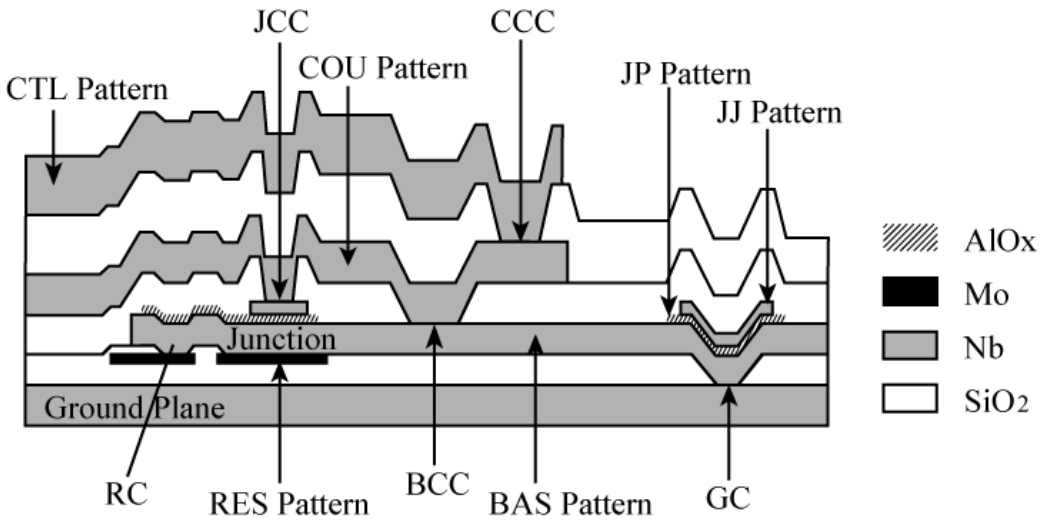


Figure 3-7: Cross-section of a chip fabricated by STP2

3.3.2 Advanced Process

The ADP2 is the next generation of fabricating process. In ADP2, critical current density J_c is 10 kA/cm^2 which allows twice clock rate of STP2. And there are 2 underground layers dedicated for passive transmission line (PTL), thus flexible wiring could be realized. **Figure 3-8** shows the cross-section of ADP2. To reduce the influence of the magnetic field by large bias currents, the active layers are separated from the power layer as much as possible and are shielded by several ground planes. To reduce processing damage, the active layers are laid on the top. The M1-M7 layers are planarized. The minimum JJ in ADP2 is 100 μA as well. **Table 3-1** shows the specifications of JJs in the STP2 and ADP2.

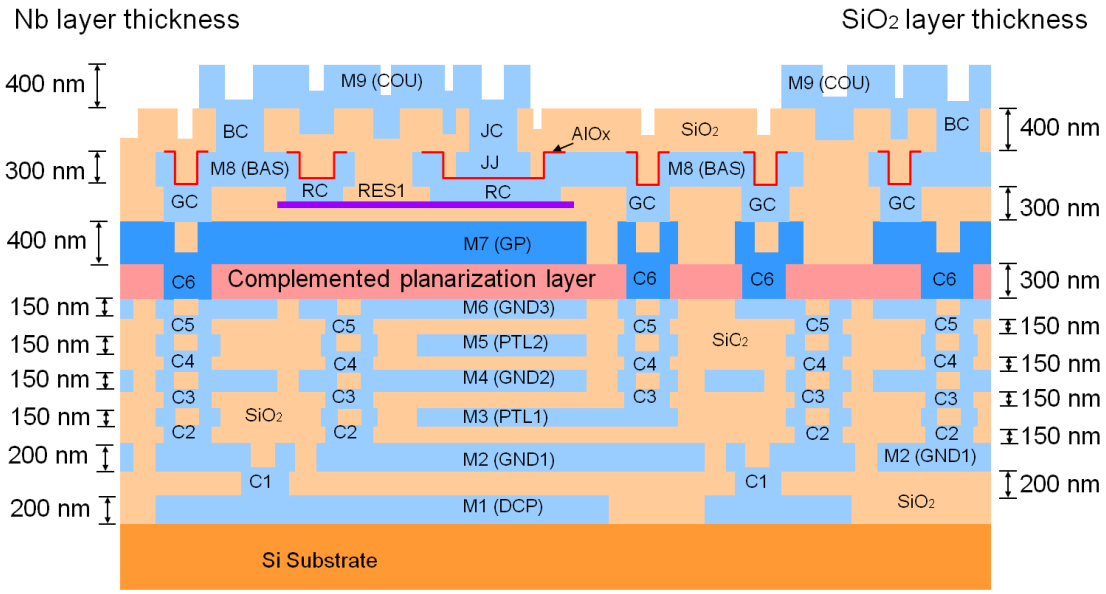


Figure 3-8: Cross-section of a chip fabricated by ADP2

Table 3-1: Specifications of JJs in the STP2 and ADP2

Process	STP2	ADP2
Gap voltage V_g	2.8 mV	2.8 mV
Normal resistance R_n	17 Ω	16 Ω
Sub-gap resistance R_g	200 Ω	100 Ω
Junction capacitance C_s	0.218 pF	0.064 pF

3.4 Experiment

3.4.1 On-chip High Speed Test

The ultra-high clock rate of SFQ circuits lead to the difficulty of high frequency clock supply using common commercial oscillator. As a solution, on-chip high speed test is employed in this research. Beside the testing circuit, a shift register (hereafter SR) for input, a shift register for output and an on-chip clock generator are applied on the same chip. At first, the data is stored into the input SR by external low frequency clock. Then a trigger is send to on-chip clock generator and high frequency clock is then generated. The high frequency clock reads out data from input SR, loading them into testing circuit and execution is performed. Result is stored into the output SR after the processing, and low frequency external clock is applied to read them out and observe them on oscilloscope.

By varying the bias current supplied to the on-chip clock generator, the clock interval is able to be changed thus the circuit could be tested at different frequency. The diagram of the on-chip high speed test system and clock generator is shown in **Figure 3-9**.

3.4.2 Measurement Environment

In this study, SFQ circuits are fabricated using Nb, with a critical temperature 9.3 K. In

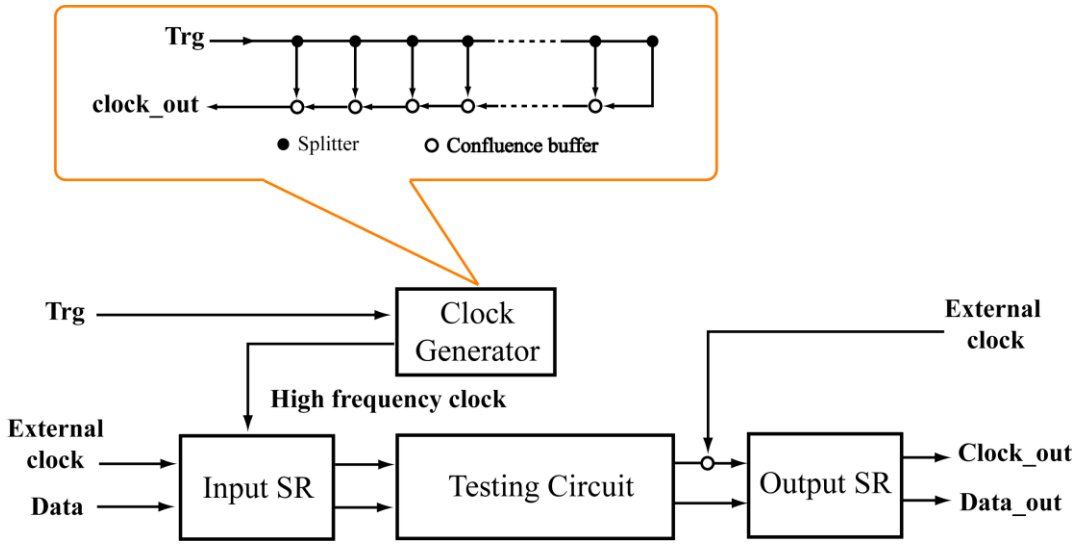
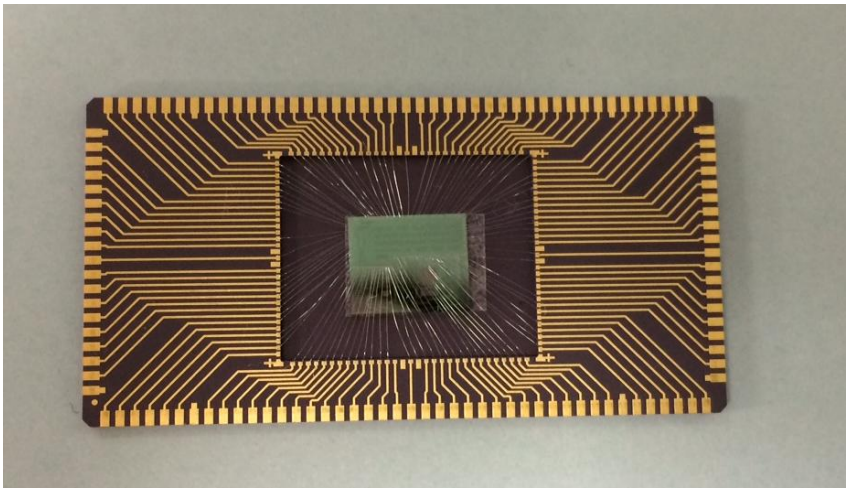
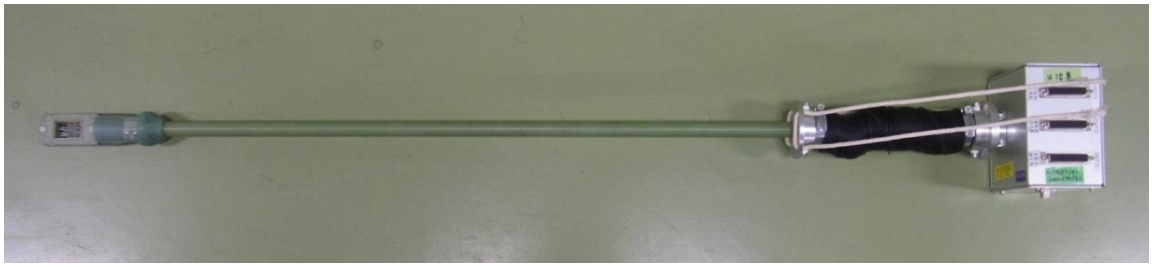


Figure 3-9: On-chip high speed test system.

the measurement, liquid helium is used who can provide cryogenic environment of 4.2 K. The SFQ chip is put on a chip carrier, and is connected using aluminum bonding wires. The chip carrier is mounted on the leading edge of probe. To screen out the external magnetic field, double magnetic shields are used, which are made by Mu-metal. The end of the probe is put in the liquid helium to cool off the chip for measurement. Finally, the probe is connected with data-generators, oscilloscope, differential preamplifier, voltage source. These equipments and systems are shown in **Figure 3-10**, and their information is listed in **Table 3-2**.



(a) SFQ chip and its carrier

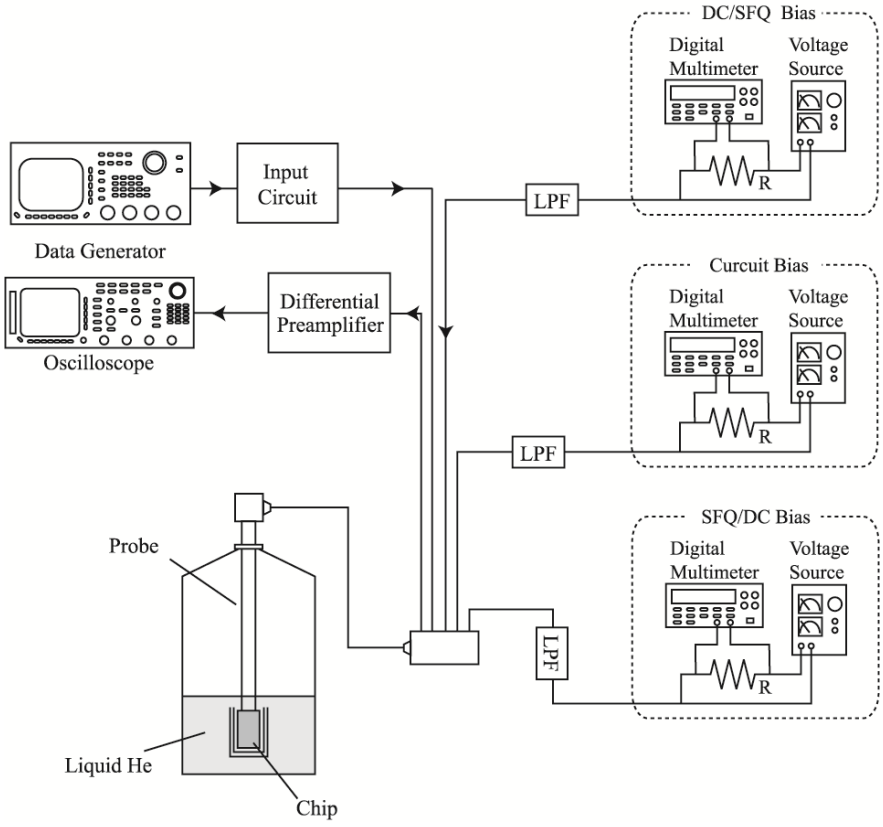


(b) Probe

(c) Magnetic shield and end of the probe

Figure 3-10: Equipment and measurement system.

(d) Liquid helium and probe



(e) Entire measurement system

Figure 3-10: Equipment and measurement system.

Table 3-2: Information of equipments used in experiment.

Products	Maker	Model number
Data generator	Sony Tektronics	DG2020A
Output port	Sony Tektronics	P3420
Attenuator	Tamagawa	VBA-761A
Power source	KIKUSUI	PMR 18-2.5DU
Differential amplifier	Stanford Research Systems	SR560
Oscilloscope	Agilent Technology	DSO5014

4

SFQ Floating-Point Unit

4.1 Introduction

Nowadays, high-end supercomputer with complex numerical analysis and simulation ability is required in various scientific fields. For individual researchers, a desk-side supercomputer will be suitable since it could provide enough computation power in most case, and is very convenient. It will be difficult to achieve this goal by Adopting of semiconductor circuit (i.e. CMOS device) due to the heat radiation, because of the compact structure.

SFQ logic is considered as a solution of this issue, since it has good features of high speed operation and low switch-energy dissipation as we mentioned in chapter 1.

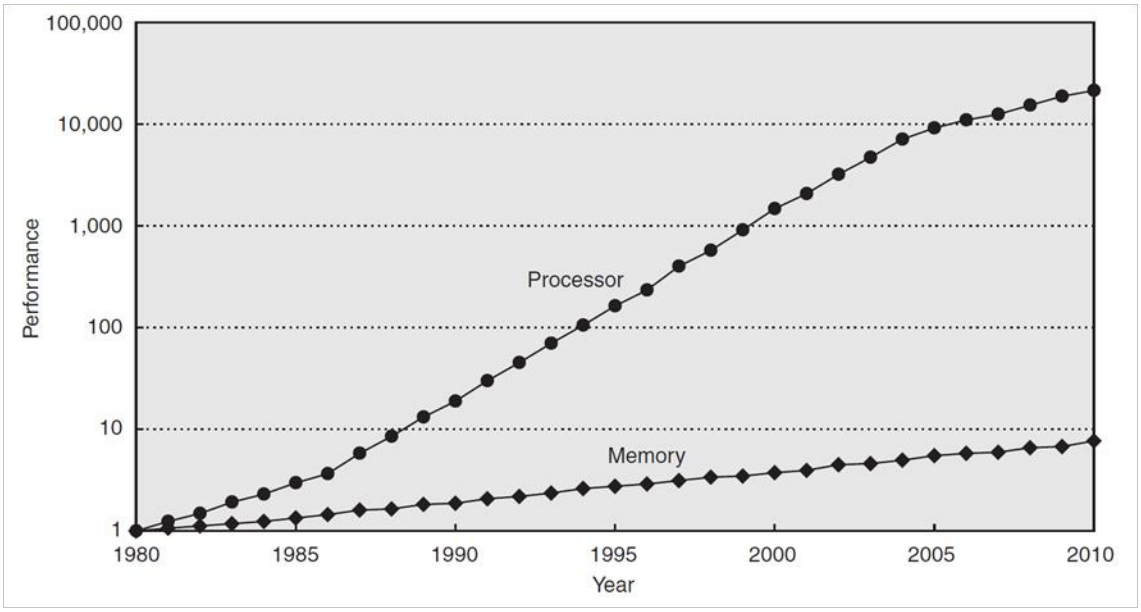


Figure 4-1: Comparison on trend of performance improvement against year between CPU and memory.
These performances are normalized by 1980's one.

However in conventional RISC (Reduced-Instruction-Set-Computer)-based supercomputer, people will face a so-called “memory wall” problem [26, 27], which means the huge gap between the performance of CPUs and memory bandwidth limited the performance of the whole system. If SFQ circuits are employed to implement supercomputer, this imbalance among CPUs and memories will be even more serious since the calculation circle is very fast in SFQ-based microprocessor.

See **Figure 4-1** for a direct imagine. This figure shows the improvement gap among CPUs and memories (DRAM) since the year 1980. Due to the Moore’s Law, every 18 months the integration of CPUs will twice increase, which means the performance of CPUs improves for approximately 60% per year (this speed had been

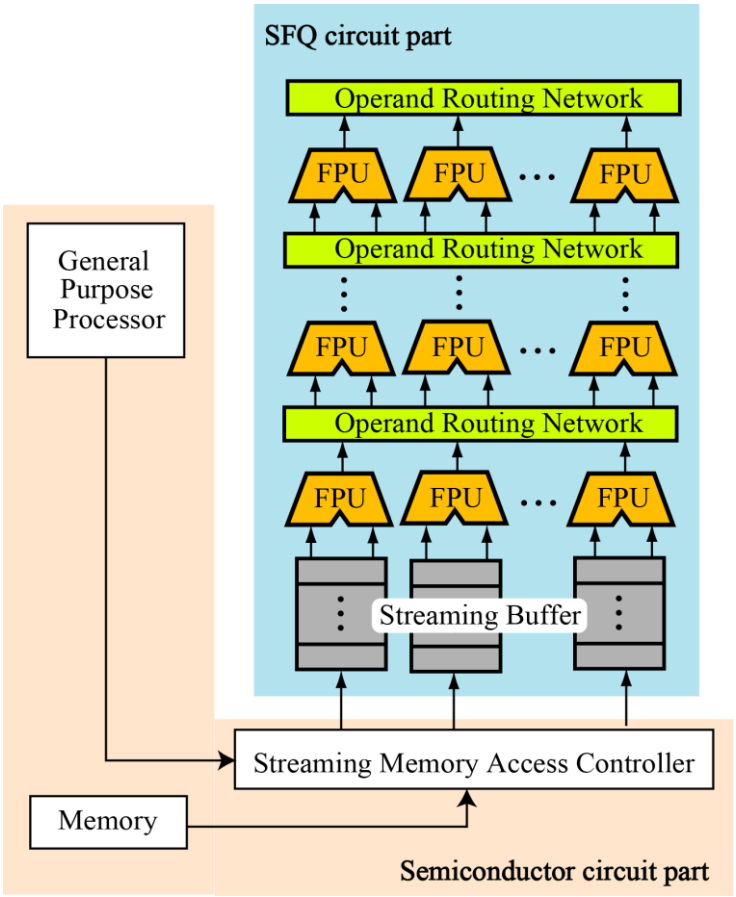


Figure 4-2: Block diagram of the SFQ large-scale reconfigurable data-path (LSRDP). SFQ floating-point units (FPUs) are connected via SFQ operand routing networks (ORNs). The LSRDP is reconfigured to calculate a mathematical problem by setting data routing in the ORNs before the execution. A memory and a general purpose processor are formed using CMOS circuits.

decreased since 2005 because increase of power dissipation limited single-core CPUs, however multi-core has become the future trend). On the other hand, DRAM, which is irreplaceable main memory of computing system, has miserably slow development. Unlike development of CPUs, specific indexes of DRAM improved quite unequally. Although capacity and bandwidth have increased, latency was almost the same during these years (latency of DDR3 is sometimes even longer than that of DDR2). Thus, it is difficult to estimate exactly how much performance-up has been done since the year 1980. The **Figure 4-1** shows a roughly 10% per year's improvement by considering memories' bandwidth.

During the numerical analysis progress in conventional supercomputer, while CPUs are processing a plenty of intermediate data, they have to wait for the response of main memories since their executing time is much shorter than the DARM's store/load time. Even in PC cluster computer or vector type computer which was utilized for these issues, there is a bottleneck in obtaining higher speed. For an extreme instance, 90% of the executing time of molecular orbital calculation was occupied by main memories' store/load instructions. This shows a typical memory wall problem that urgently requires a solution.

4.2 Large-Scale-Reconfigurable-Data-Path (LSRDP)

Since we can't obtain much more powerful DRAM due to the physical limitation, and it is probably that in near future, new device technique that can replace nowadays DRAM would not appear, the available way to conquer memory wall problem is reduction of memory access rate. Based on this idea, a novel data-path architecture was proposed as a solution to memory wall problem by Prof. N. Takagi in 2008. This is called Large-Scale-Reconfigurable-Data-Path (LSRDP) [28]. In contrast to the conventional computing systems, the LSRDP is composed of thousands of floating-point units (FPUs), which are connected to each other via operand routing networks (ORNs) [29] as shown in **Figure 4-2**. Neighboring FPUs can directly exchange their intermediate results, thus memory access rate can be considerably reduced.

Our goal in the LSRDP project is to establish fundamental technologies for realizing a 10-TFLOPs-scale desk-side computing system. In order to achieve this goal, we plan to develop SFQ FPUs with 4-GFLOPs' performance, by which

16.4-TFLOPs performance can be achieved in the whole system [28]. Though advanced CMOS process could accomplish this performance as well, densely integrated processors with high switching activity generate extremely large amount of heat, which make it difficult to integrate the system. Low power dissipation of the SFQ circuits is a significant advantage for realizing the LSRDP system.

In the LSRDP, the SFQ FPUs are those of the main circuit blocks, and is the most complicated circuit block. In this chapter, I will introduce the design, implementation and high-speed test results of the SFQ FPUs. In the end of this chapter, I will give a brief assessment of the FPUs and the whole system in the aspect of throughput and power dissipation.

4.3 Floating Point

Floating point are how real numbers are expressed in computing systems, and they are widely used in many scientific and engineering fields because they are able to represent wider range of numerical value than the fix point. As the cost, floating point is more complicated in representation and hardware implementation. The following formula shows the common floating point from:

$$\mathbf{(-1)}^{S} \times \boldsymbol{F} \times \mathbf{2}^{E}, \tag{4.1}$$

Where S stands for a sign, F is a significand (fraction), and E is an exponent. **Table 4-1** lists the bit lengths of several floating-point number formants, which are defined in the IEEE 754 standard. It should be noted that the exponent is represented as an unsigned number by adding a fixed bias with the value of 2^{n_E-1}-1, where n_E is the bit length of the exponent, to simplify the floating-point calculations, as the following description:

While we use two's complement or any other notation in which negative exponents

Table 4-1

Bit lengths of several floating-point-number formats in the IEEE 754 standard.

Precision	Sign	Exponent	Significand	bias
half	1 bit	5 bit	11 bit	15
single	1 bit	8 bit	24 bit	127
double	1 bit	11 bit	53 bit	1023

It is implied that "1" is contained at the most significant bit of the significand

have a 1 in the most significant bit of the exponent field, a negative exponent will look like a big "positive" number. For example, a single-precision exponent "1"would be represented as

$$\mathbf{(00000001)_2} \tag{4.2}$$

However, a negative exponent "-1"will look like a big number, since it is represented as:

$$\mathbf{(11111111)_2} \tag{4.3}$$

The desirable notation must therefore represent the most negative exponent as $(00\ldots00)_2$and the most positive as $(11\ldots11)_2$. This type of notation could be realized if we add an additional fixed bias with the value of 2^{n_E}-1, i.e. 127 in single-precision. This convention is called **biased notation**, with the bias being the number subtracted from the normal, unsigned representation to determine the real value. In bias notation, "1" and "-1" will be represented as:

$$\mathbf{(1+127)_{10} = (128)_{10} = (10000000)_2} \tag{4.4}$$

$$\mathbf{(-1+127)_{10} = (126)_{10} = (01111110)_2} \tag{4.5}$$

A general floating point unit (FPU) executes several kinds of computing including addition, multiplication, division, square root and others. In the most common FPU structure, each function is implemented in one computing unit using hardware integration algorithm, and multiplexer is employed in order to merge calculating result of these units as output of the FPU.

Another FPU structure is a so-called fused multiply-add (FMA) architecture. It was found in several DSP (digital signal processing) executions (such as FFT, FIR filter), a floating-point multiplication is always followed by an addition. In 1990, IBM released their RISC system/6000 architecture, including a floating-point FMA unit. This unit could calculate $A \times B + C$ in one execution, with less latency and higher precision compare to conventional FPU. However it cost 27% more power, and gives more latency while executing single floating-point instruction.

This research focuses on the conventional type FPU, including two fundamental FPU computing units, floating point multiplier (FPM) and floating point adder (FPA).

In our previous study, half-precision floating point multipliers (FPMs) and adders (FPAs) were designed and fabricated using the ISTEC 2.5 kA/cm^2 standard Nb process. Their successful operations were demonstrated at a maximum frequency of 31.5 GHz for FPMs [30] and 24 GHz for FPAs [31]. In recent years, AIST-ISTEC (AIST: National

Institute of Advanced Industrial Science and Technology. ISTEC: International superconductivity technology center) has developed an advanced 10 kA/cm^{2}Nb process (ADP2) [32] (Chapter 3 shows the process details). Because the operating speed of SFQ circuits increases proportional to the square root of the critical current density, SFQ FPUs at about twice the operating frequency can be achieved using the ADP2. In addition, multilayer passive transmission lines (PTLs) are available in the ADP2, which results in flexible interconnection and the reduction of the circuit size. Therefore, our target is to design and implement bit-serial SFQ FPUs using the ADP2 with a much higher frequency (more than 50 GHz) and much smaller area cost (50% reduction)

The half-precision (16-bit) FPUs were demonstrated firstly, and then we expanded them into single-precision (32-bit) version.

4.4 Hardware Integration Algorithm Design of FPM

Floating point multiplier (FPM) is the computation component in FPU that performs floating point multiplication. At first, I will briefly explain how we do the floating point multiplication in computing system.

4.4.1 Calculating Flow of Floating point Multiplication

Generally, standard floating-point multiplication is performed according to the following calculation flow [33] (**Figure 4-3**):

1). **Exponent addition.** The two exponents are added and then the exponent bias is subtracted. The later step is intrinsic necessary because the exponent is in bias notation format, thus the result will be twice biased if no bias subtraction is done.

i.e.: Ea = 10, Eb = -5, Result = 10-5 = 5

In bias notation without bias subtraction:

(**10**+127) + (**-5**+127) = 259 = (**132**+127), the answer = 132

In bias notation with bias subtraction:

(**10**+127) + (**-5**+127) – 127 = 132 = (**5**+127), the answer =5 equals to the correct

result.

2). **Significand multiplication.** The two fractions are multiplied.

3). **Normalization.** The product is normalized if an overflow appears. If normalization is performed, “1”is added to the exponent result.

4). **Determine the sign of the result**. The sign of the product is plus when the signs of the two multiplicands are same. In other case the sign of the product is minus.

5). **Rounding.** The result is rounded to fit the data format.

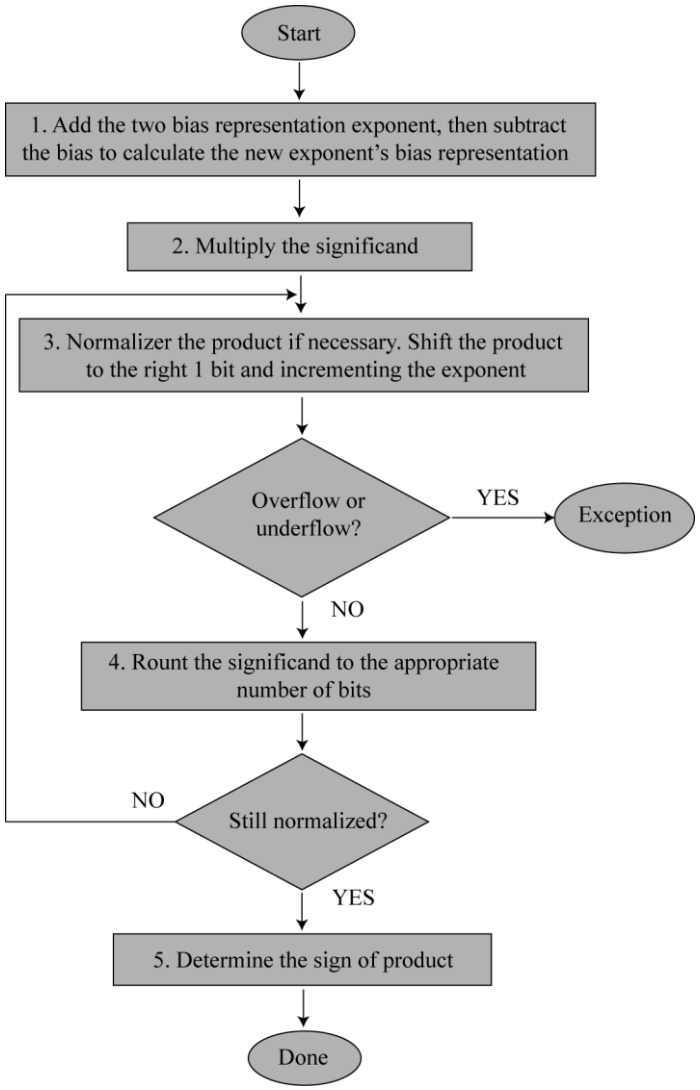


Figure 4-3: calculating flow of floating point multiplication

The fifth step is combined with the third step into one step in our FPM algorithm.

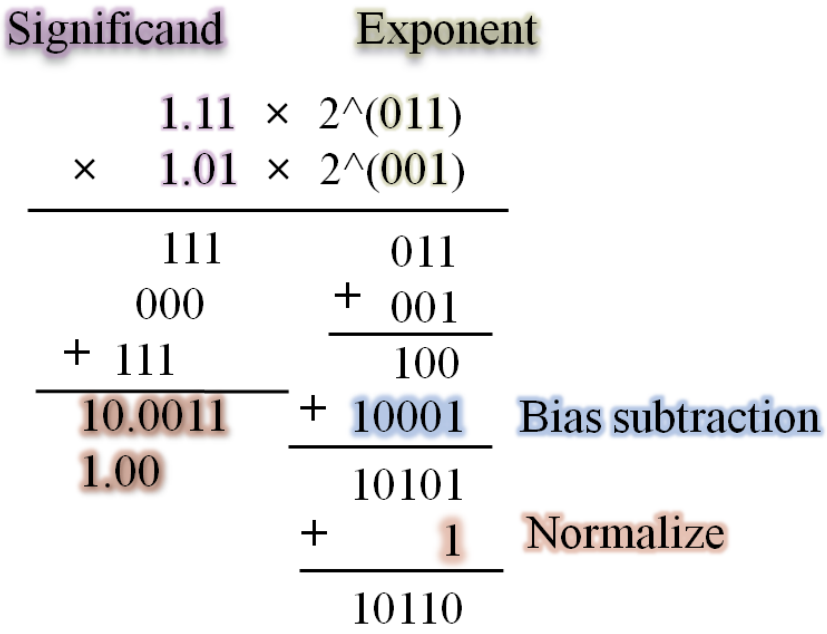


Figure 4-4: Direct imagine of floating point multiplication

In order to reduce the complexity of the circuit, “round toward 0”is adopted by omitting the LSB (least significant bit) and shift the product. **Figure 4-4** shows a more direct imagine of this calculation flow, where we can see how and when the addition of exponent, multiplication of significand and the normalization are done. It should be noted that the bias subtraction is performed via adding two’s complement of the bias to the minuend.

4.4.2 Circuit Block Design

Our FPM design complies with this calculation flow. It based on a bit-serial architecture, which is effective for utilizing the inherently high clock rates of SFQ circuits and minimizing the area costing. The FPM is divided into two parts: a significand part and an exponent part. Floating-point numbers to be calculated are input in a bit-serial data format, where the two bit-serial data correspond to the significand and the exponent with the sign. The block diagram is shown in **figure 4-5**.

According to the FPM algorithm, the significand processing part performs multiplication of the two fractions, where a systolic array structure [34, 35] is employed as the multiplication algorithm. The exponent processing is composed with an adder and a subtractor, where the exponent addition and bias subtraction are performed. Additionally, normalizer is adopted at the end of the algorithm to transform the product into standard data form.

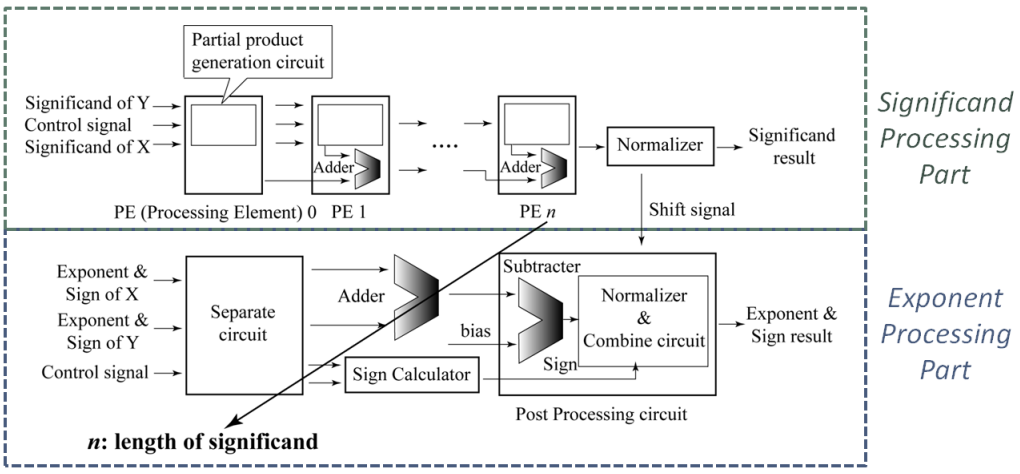


Figure 4-5: Block diagram of SFQ bit-serial FPM

4.4.3 Multiplier

The multiplication of significand is performed based on a systolic multiplier as mentioned in section 4.4.2. A systolic array is a pipeline network arrangement of **processing elements** (PEs). It is a specialized form of parallel computing, where PEs (i.e. processors) compute data and store it independently of each other. It is composed of matrix-like rows of processing elements shown in **Figure 4-6.**These PEs are similar to central processing units (CPUs), with similar structures and are connected to a small number of nearest neighbor PEs in a mesh-like topology. PE receives intermediated data from its neighbors and processing it, then shares the information with its neighbors immediately after processing. The systolic array structure is useful while the dealing progress is homogeneous iteration. One example of systolic array structure based CPU is Intel's "iWarp"[36].

This systolic array multiplier has reduced only one data flow direction. PEs, which perform simple calculations including partial product generation and sum addition, are arranged regularly in series. While the data stream flows through ***n*** PEs, ***n*** times of addition are done and multiplication is completed. It should be noted that this is a pipelining processing, therefore we could obtain very high throughput.

Figure4-7 shows the schematic of the PE, where the clock line is not shown. At

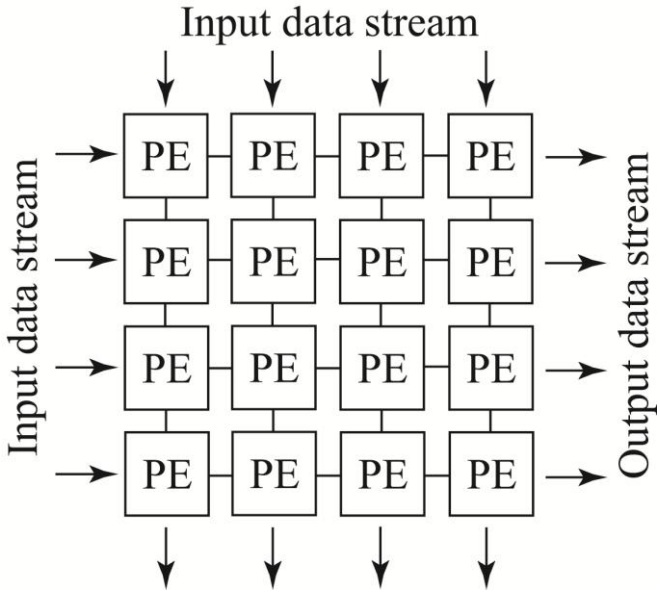


Figure 4-6: Typical systolic array structure

first, data “Y” and “Control signal” arrive at the NDRO cells in the partial product generator circuit (the circuit in the colored frame) with proper timing, where the input signals arrive at the NORO1 with the sequence of “Reset”, “Set” and “Clock”. Then the NORO1 send a “Set” signal for the NORO2, which is already reset by “Control signal”. After that, a partial product is generated by the NDRO2 when data “X” arrive. The partial product is then added to the sum data S by the adder. The two D-flip-flops are used to shift data X and the control signal by 1 bit, to match the timing of data Y and S.

Unlike parallel multiplication such as wave-pipeline carry-save algorithm [37, 38], the scale and the power dissipation of the multiplier increase linearly with increase of

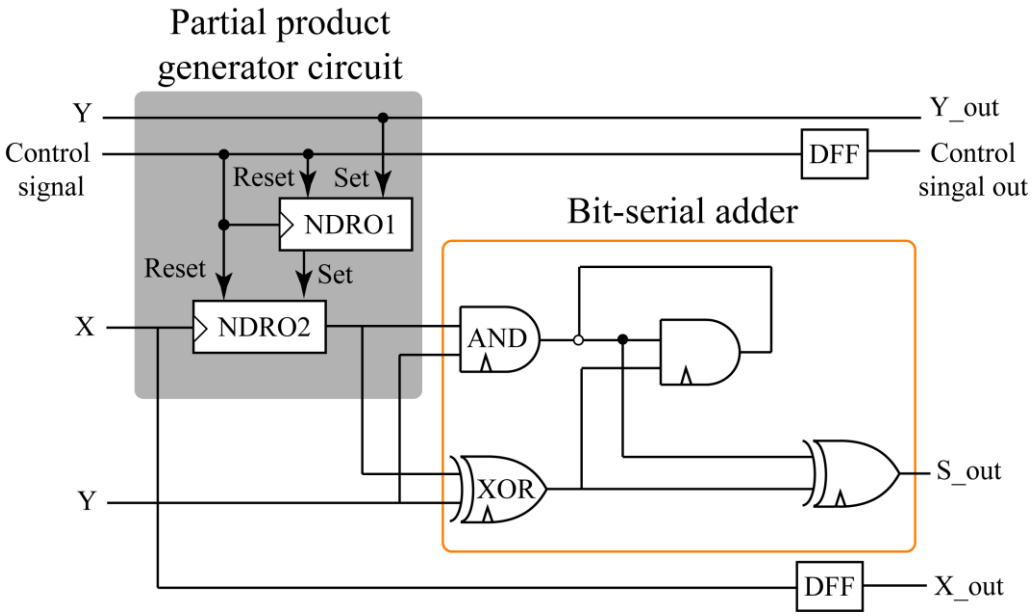


Figure 4-7: Schematic of PE in our multiplier.

Table 4-2

Comparison of *n*-bit systolic array multiplier and wave-pipeline carry-save multiplier

Multiplier	Latency	Scale
Systolic	$\tau_a+\boldsymbol{n}$	$\boldsymbol{n}$
Carry-save	$\tau_a+\log_2(\boldsymbol{n})$	$\boldsymbol{n}^2$

τ_a: Required calculating time for multiplication. In both systolic and carry-save algorithm, this time is $(2\boldsymbol{n}+1)$ clock period.

$\boldsymbol{n}$/$\log_2(\boldsymbol{n})$: The clock delay. It is proportional to $\boldsymbol{n}$ and $\log_2(\boldsymbol{n})$ in systolic and carry-save algorithm, respectively.

bit length in the systolic array structure. Meanwhile, the latency is also proportional with the bit length. However, this structure is scalable and easy to be expanded into larger multipliers. **Table 4-2** shows the comparison of systolic array multiplier and wave-pipeline carry-save multiplier.

4.4.4 Normalizer of Significand

When an overflow occurs in the significand, the result is normalized to fit the appropriate normalized scientific form. **Figure 4-8** shows a block diagram of the normalizer of the significand. The shift register at the bottom stores the calculated product. When an overflow occurs, which corresponds to the most significant bit (MSB) of the significand being equal to "1", the MSB signal disables NDRO2 and enables NDRO1, which sends a set signal to NDRO3. Then the calculated product is shifted left by 1 bit and output from NDRO3. Meanwhile, the set signal is also sent to the exponent part to add "1" to the exponent result. Otherwise, in the case of underflow, the NDRO2 is enabled and the set signal is sent to NDRO4, which allows the data stream output in the original form without any shifting.

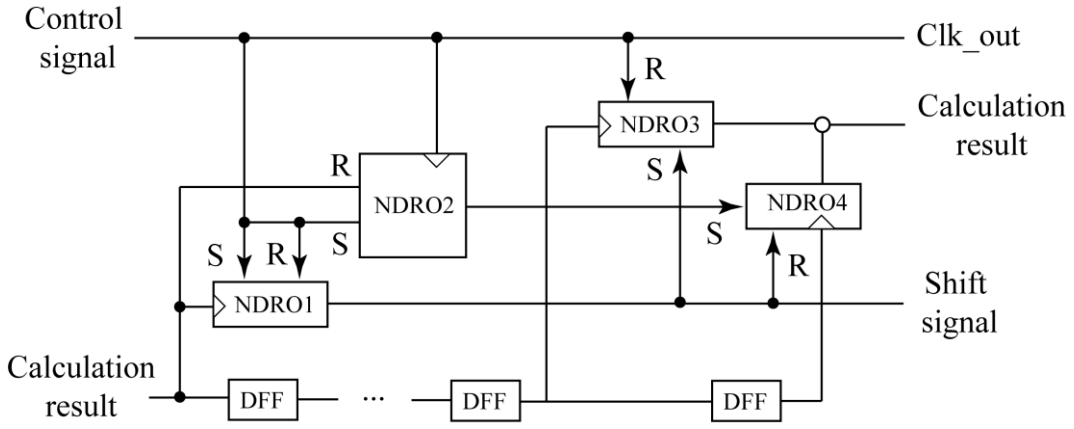


Figure 4-8: Schematic of the significand normalizer.

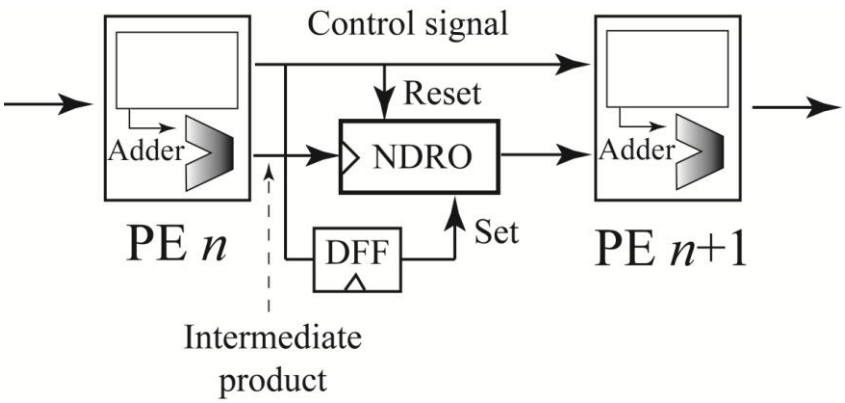


Figure 4-9: Schematic of the post-handling circuit.

4.4.5 Enhancement of Throughput of Multiplier

In a systolic-array multiplier, the intermediate product will be increased by 1 bit when the data pass through one processing element. Thus an ***n***-bit multiplication generates a final result with a length of 2***n***bits, and 2***n*** clock cycles are required to complete the calculation. This doubling of length is a significant drawback in terms of the throughput.

In order to enhance the throughput, a post-handling circuit between every pair of PEs was added, which cuts off the least significant bit (LSB)of the intermediate product. Its schematic is shown in **Figure 4-9**. A "reset" signal arrives at the NDRO cell immediately before the arrival of the LSB of the intermediate product, to disable the NDRO.

By using the post-handle circuit, the final result of the product is compressed into ***n***+1 bits and ***n***+1clock cycles are necessary to complete the calculation. Therefore, the throughput is improved from *f/2**n*** to *f/(**n***+1*)*. More details of the throughput assessment will be given in later sections.

4.4.6 Exponent Processing Part

In floating-point multiplication, the exponent result is calculated by adding the two exponents of the two floating-point numbers. Because the exponent bias is added in the bias notation, the exponent bias with the value (i.e. in half-precision floating point, 2^4-1 = 15) has to be subtracted after the addition of two exponents. In the design of half-precision FPM, it was realized by adding $(10001)_2$, the two's complement of "15",

to the result. **Figure 4-10** shows a schematic of the half-precision exponent processing part, which is composed by 4 main logic parts.

A. Sign separate& determination circuit

Since the sign of multiplicand is combined with the exponent in our FPM, it must be separated from the exponent before the processing. In original form, sign is the LSB (last significant bit) of exponent, thus "and" gate is used to extract it from the exponent. While the two signs are same, the sign of the multiplication result would be positive and in the case of the two signs are opposite, the sign result would be negative. Following this rule, an exclusive "or" (XOR) gate is adopted to determine the sign result.

B. The bit-serial adder

A bit-serial adder similar to the one in PE of the multiplier is employed to add the two exponents. Data streams are inputted and outputted in bit-serial format.

C. The serial-parallel converter

For the design's convenience, a parallel ripple-carry adder [39] is used to subtract the bias value from the addition result from the former bit-serial adder. Thus before the

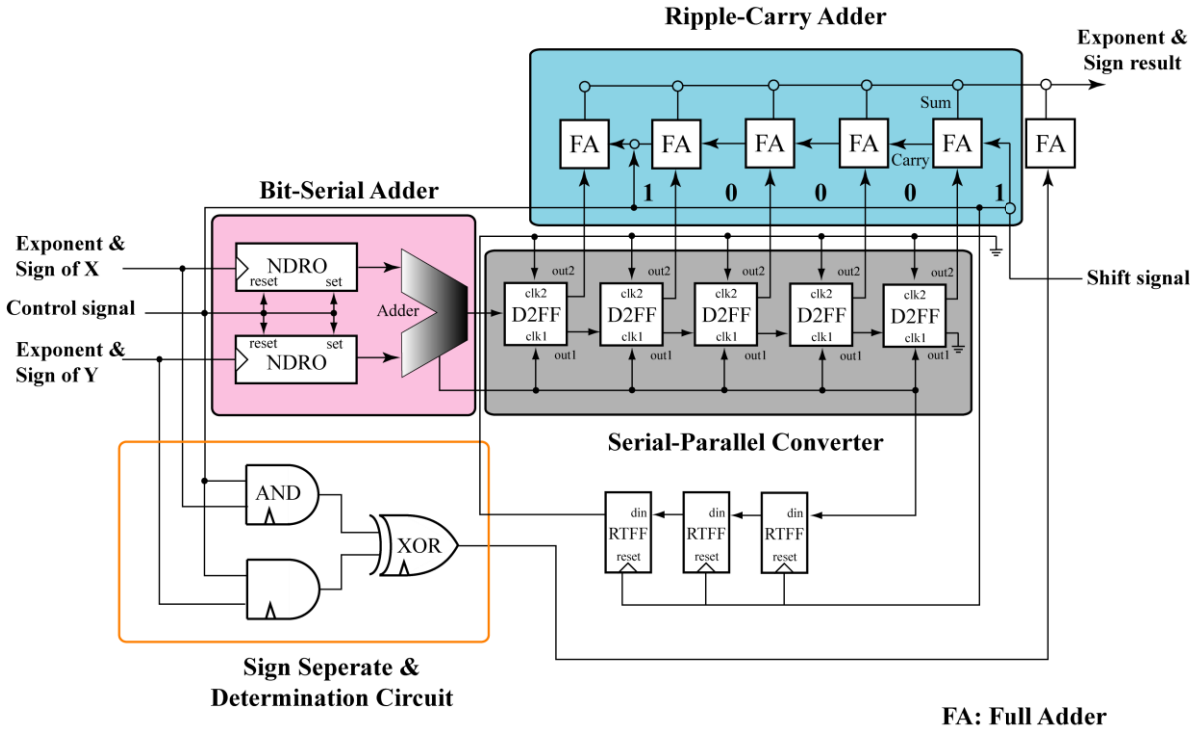


Figure 4-10: Schematic of the exponent processing part

subtraction is performed, it is necessary to convert the bit-serial data format into the bit-parallel data format. The serial-parallel converter (SPC) is designed based on the D2FF cells. It stores bit-serial data similar to a shift register, and when the clock signal arrive the “clk2” terminal of D2FFs, it would output the stored data in the bit-parallel format. 3 TFFs compose a 3-bit counter, which makes sure the clock signal arrive “clk2” after the data has been stored into the SPC completely.

D. Ripple-carry adder

A 5-bit ripple-carry adder is employed to add “10001”, the two’s complement of bias value “15”$(01111)_2$ and the shift signal from the significand normalizer to the addition result. The data”10001” is generated using the control signal as shown in **Figure 4-10**. The full adders are implemented by simply using “T1” cell. An additional full adder (T1 cell) is placed as the LSB of the subtraction result, in order to combine the sign result and the subtraction result. The calculation result is outputted in bit-serial data format as well.

4.5 Implementation and On-chip High-speed Test of FPM

Based the hardware integration algorithm I introduced previously, I designed the half- (16-bit) and single-precision (32-bit) FPM using the AIST advanced Nb process (ADP2) with a critical density of 10 kA/cm^2 and the CONNECT cell library for the ADP2[40]. Logic simulations were performed to optimize the internal timing, where picosecond-level timing adjustment is required because the target clock frequency is 50 GHz. Using the verilog-HDL environment, the behavior of circuit and the timing details of critical path and cell are viewed.

4.5.1 Implementation and On-chip High-speed Test of Half-precision FPM

The simulation result shows the DC bias margin of my half-precision FPM is from -20% to +25% at the target clock frequency of 50 GHz. The margin shrinks as the frequency increases, and the highest frequency is estimated to be 85 GHz. The overall circuit contains 11066 Josephson junctions which includes on-chip high-speed test circuits, such as in/output shift registers and an on-chip clock generator. Size and power consumption of the FPM are 6.66 mm × 1.92 mm and 2.83 mW, respectively. A microphotograph of the FPM is shown in **Figure 4-11**. I measured the DC bias margins of the FPM for several data patterns by low-speed tests and on-chip high-speed tests.

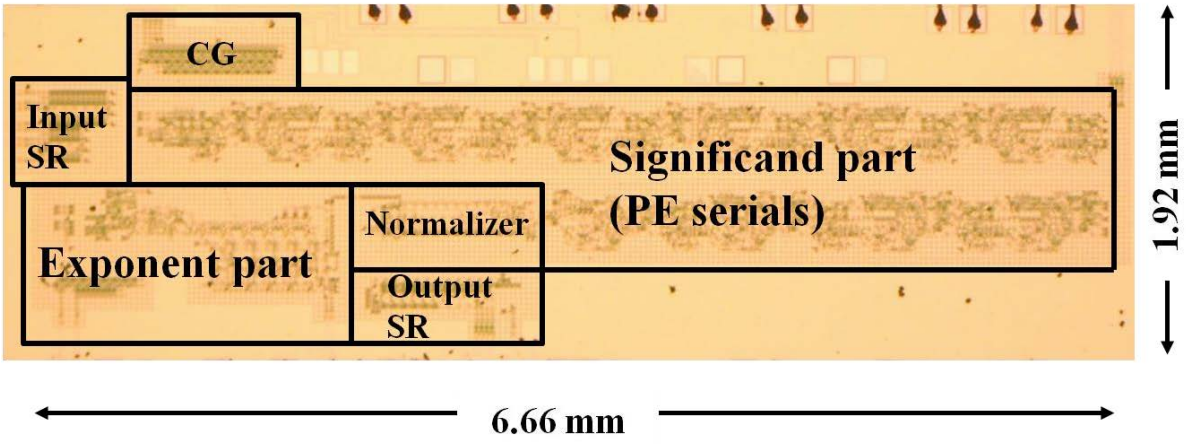


Figure 4-11: Microphotograph of half-precision FPM
SR = shift register, CG = clock generator

Figure 4-12 shows an example of on-chip high-speed test results. In this test, the input data *X* and *Y* are -$(11110111100)_2$×exp $(00110)_2$ and -$(11111111111)_2$×exp $(00011)_2$, respectively. It should notice that all bits of *Y* are set to "1" to check whether all PEs are working correctly. This pattern also checks the normalization of the result. The correct answer +$(11110111011)_2$×exp $(11011)_2$ can be seen in **Figure 4-12**, where the inputs of the significand and exponent are denoted as (S_X, S_Y) and (E_X, E_Y), respectively.

DC bias margins of each circuit block at low speed and at 50 GHz are shown in **Figure 4-13(a)** and **Figure 4-13(b)**, respectively. And frequency versus measured and simulated DC bias margins of the FPM is shown in **Figure 4-14**. One can see that the narrowest DC bias margin at 50 GHz is ±3.6%. The maximum operation frequency is measured to be 72 GHz for the significand part and 93.4 GHz for the exponent part. It should note that the DC bias margin of the circuit block, which contains PE6-8 in the significand part, shrinks significantly when the clock frequency is increased from low speed to 50 GHz. One possible reason is the local parameter variation, which results in a timing variation in each PE.

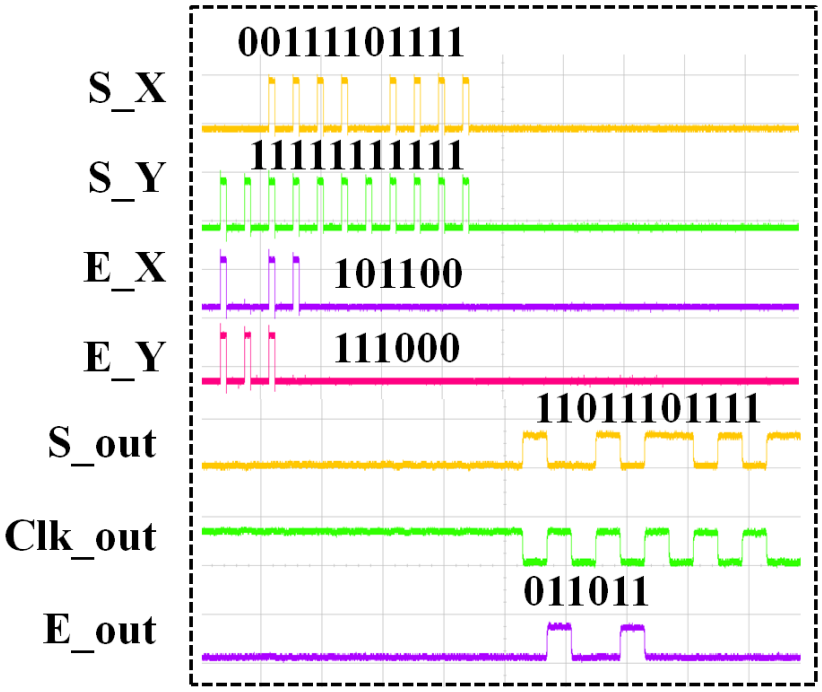
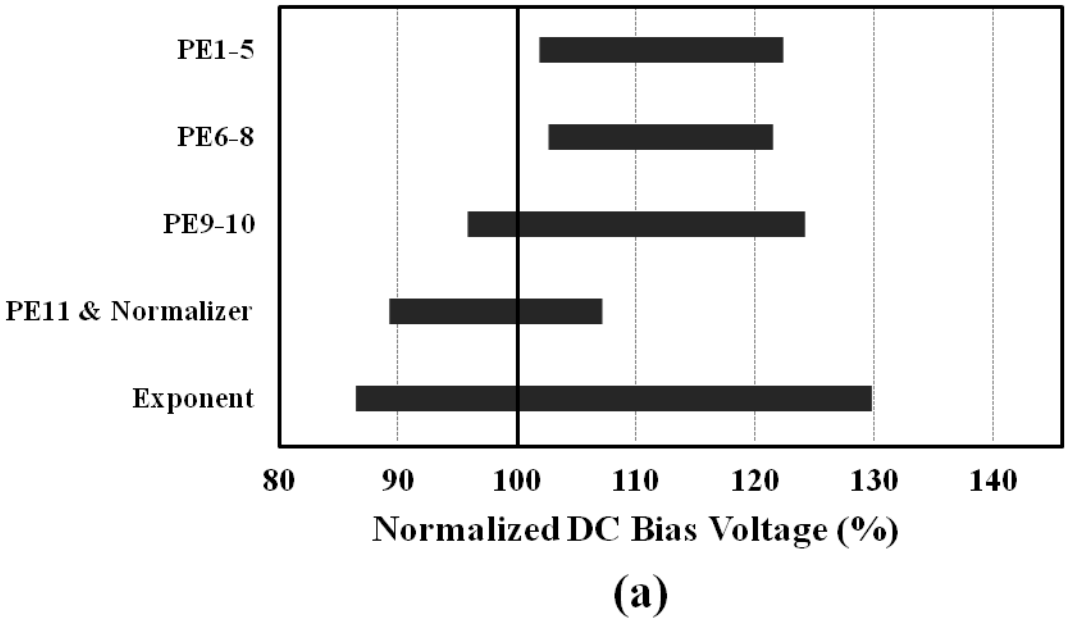


Figure 4-12: Example test results of the half-precision FPM in the on-chip high-speed test at 50 GHz.



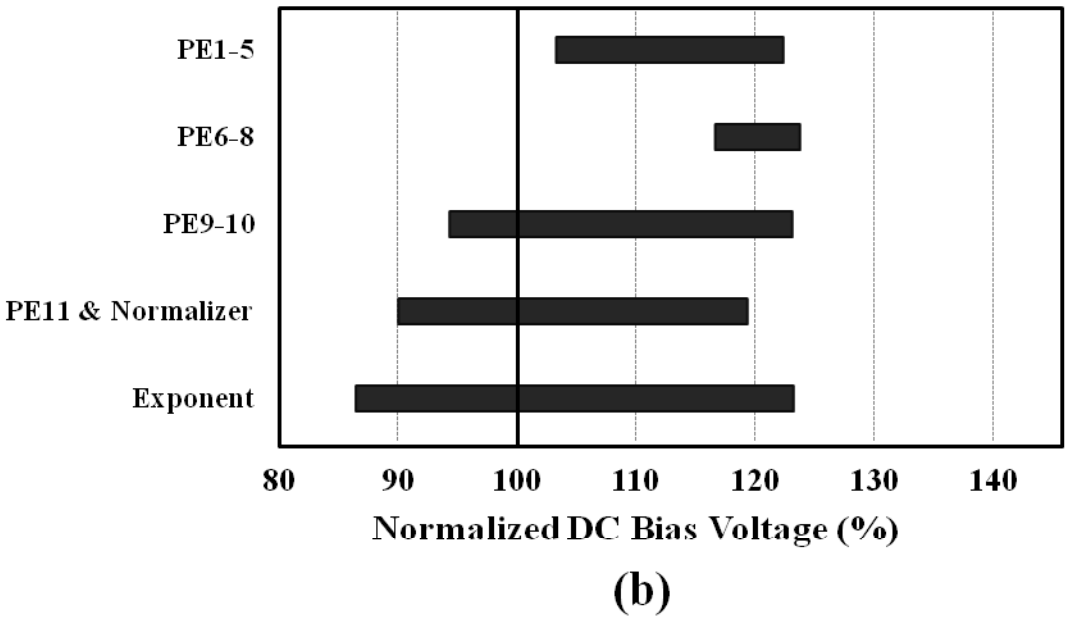


Figure 4-13: DC bias margins of each circuit block of half-precision FPM.
(a) Low speed and (b) 50 GHz

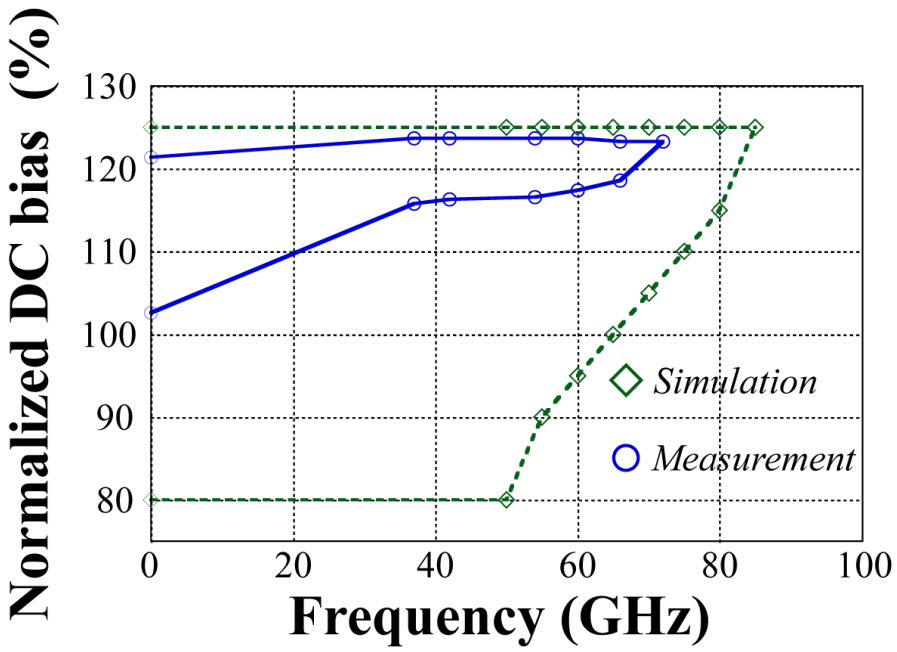


Figure 4-14: Frequency versus DC bias margins of the tested half-precision FPM. Solid line shows the measurement result and dotted line shows the Verilog

4.5.2 Implementation and On-chip High-speed Test of Single-precision FPM

The single-precision FPM (hereafter SFPM) bases on the same algorithm of half-precision FPM, thus it is not intrinsically difficult to extend the half-precision FPM into single-precision version in hardware design. However, I still faced the following possible obstacles:

1). **Ground current**. The returning bias current through the ground plane always induces a serious problem in SFQ circuits, especially in large-scale circuits. Since the intrinsic operation principle of SFQ circuits is based on the dynamics of SFQ, magnetic fields caused by large ground current seriously affect the circuit operations. Single-precision FPM is expected to have around 20000 junctions and require the bias current of more than 3 A. The management of the large ground current would be a critical challenge. This issue will be discussed more in detail in the next chapter.

2). **Parameter variations and circuit yields**. The increase of the circuit scale brings about increased possibility of local parameter variation as

well. As we mentioned before, the local parameter variation is the main cause of performance deterioration in our half-precision FPM, because we have to consider large margins in designing the timing of data and clock lines. Moreover, serious parameter variations would cause complete malfunction of the circuits, resulting in decreasing of the circuit yields.

Based on the same design goal, I obtained the DC bias margin of single-precision FPM from -20% to +25% at the target clock frequency of 50 GHz by verilog simulation. Similar to the half-precision FPM, the margin shrinks as the frequency increases, and the highest frequency is estimated to be 80 GHz. The scale of the single-precision FPM is approximately twice of the half-precision one: The overall circuit contains 21917 Josephson junctions Size and power consumption of the SFPM are 7.56 mm ×3.24 mm and 5.76 mW, respectively. A microphotograph of the SFPM is shown in **Figure 4-15**.

Similar to the half-precision FPM testing, the significand of the input Y is fixed to $(11111111111111111111111)_2$ as well in order to check whether each PE operates correctly while we measured single-precision FPM. **Figure 4-16** shows an example of on-chip high-speed test results. In this test, the input data X and Y are "$(11111011110011110111100)_2 \times \exp (00001101)_2$" and "$-(11111111111111111111111)_2 \times \exp (00000101)_2$," respectively. Additionally, this test pattern also checks the normalization of the result. The correct answer can be seen in

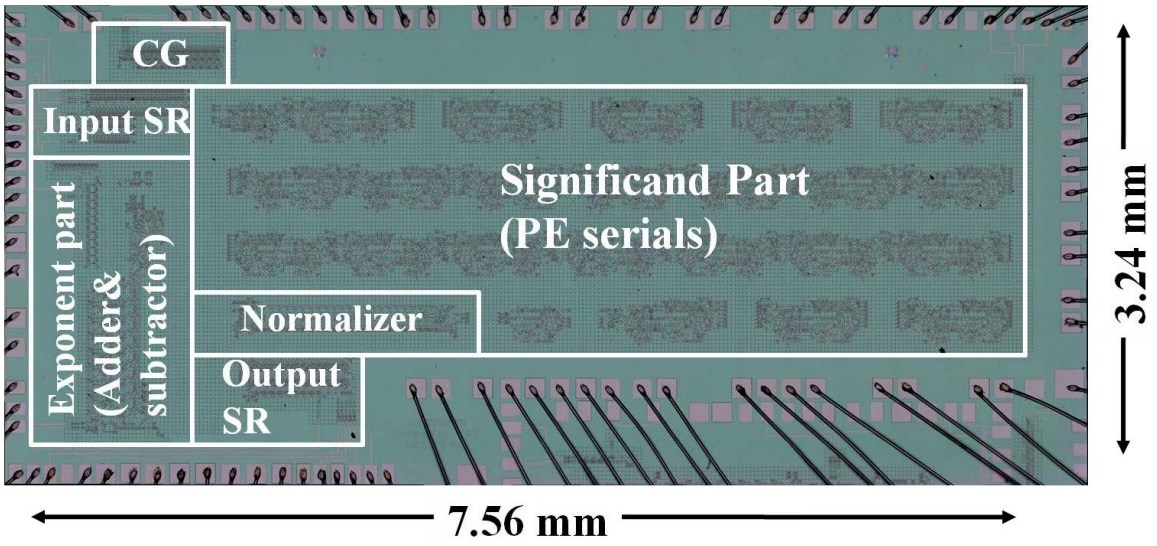


Figure 4-15: Microphotograph of single-precision FPM
SR = shift register, CG = clock generator

Figure 4-16.

Frequency versus measured and simulated DC bias margins of the SFPM is shown in **Figure 4-17**, and DC bias margins of each circuit block at low speed and at 50 GHz are shown in **Figure 4-18(a)** and **Figure 4-18(b)**. The DC bias margin at 50 GHz is 103.35% - 111.73%. This margin is limited by the circuit block PE20-21, and the maximum operation frequency of the FPM is 59 GHz for the significand part and 69 GHz for the exponent part.

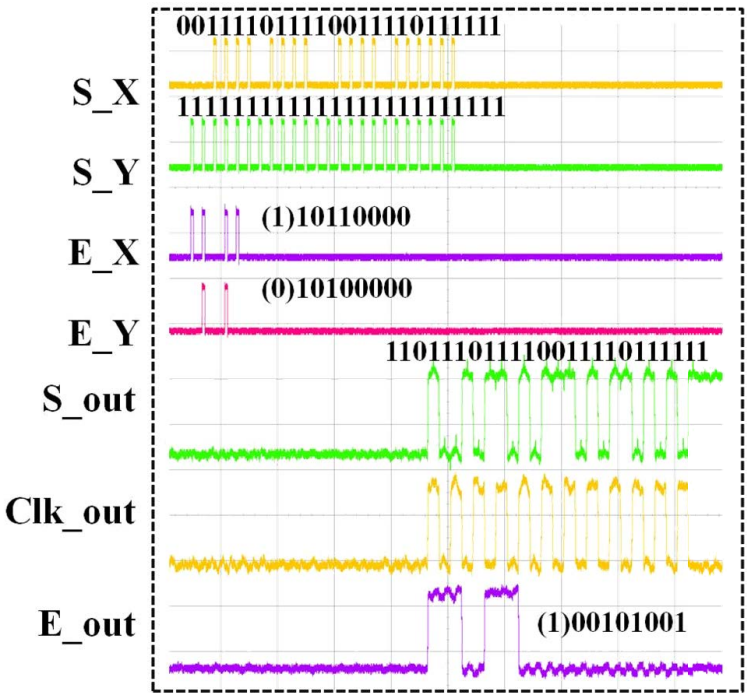


Figure 4-16: An example of waveforms in the on-chip high-speed test of the single-precision FPM.

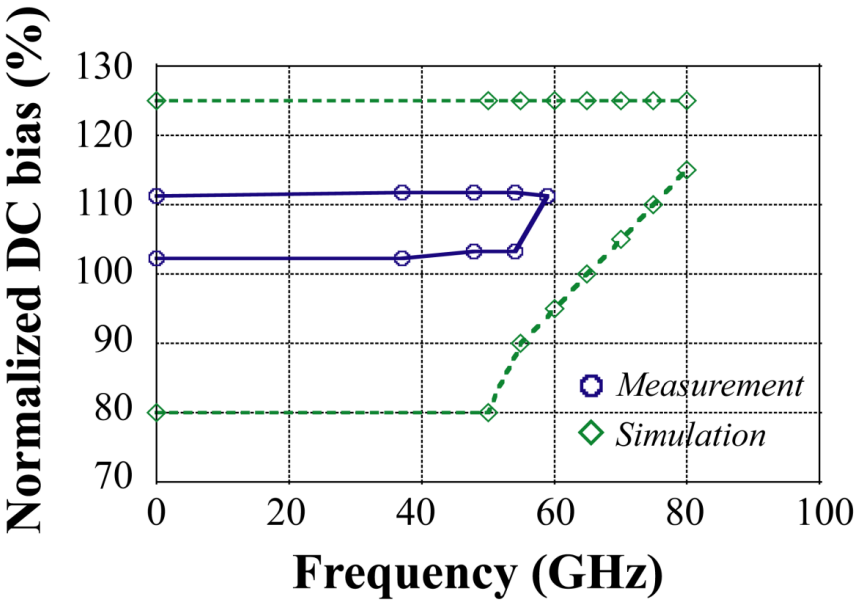


Figure 4-17: Frequency versus DC bias margins of the tested single-precision FPM.

Solid line shows the measurement result and dotted line shows the Verilog.

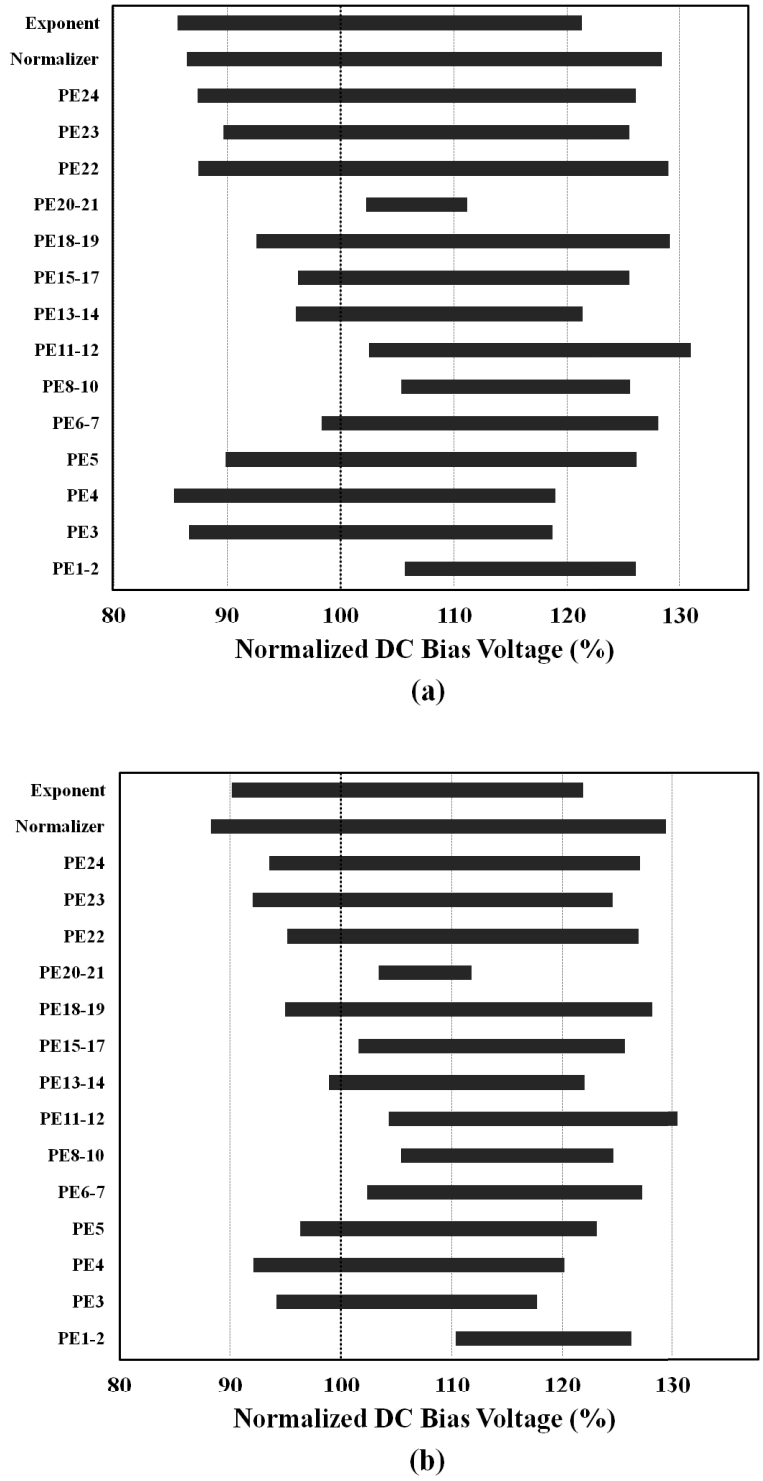


Figure 4-18: DC bias margins of each circuit block of single-precision FPM.
(a) Low speed and (b) 50 GHz

4.6 Hardware integration Algorithm Design of FPA

Floating-point adder (FPA) is relatively more complicated than the FPM in bit-serial data format structure, since the floating-point addition requires more steps and couldn't be hardware realized by regular homogenous structures. Previously H. Park and T. Kato have demonstrated successful on-chip high speed operation of the half-precision FPA at 24 GHz and 55.1 GHz, respectively [31, 41]. These circuits were designed and fabricated using AIST standard (STP2) and advanced (ADP2) process. The single-precision FPA reported this time used the same top algorithm as the above two, with several components replaced by new design.

4.6.1 Calculating Flow of Floating point Addition

Generally, standard floating-point addition is performed according to the following 6-step calculation flow [33] (**Figure 4-19**), and **Figure 4-20** shows a more direct imagine:

1). **Exponent subtraction.** The difference of two exponents is calculated.
2). **Significand alignment.** The significand of the smaller number is shifted to the right by the difference of the two exponents. Since we can only represent limited data bits, less important bits of the smaller number are omitted.
3). **Add (or subtract) the two significands.** This calculation is signed addition (or subtraction).
4). **Determine sign of the result.** The sign of the result depends on the signs of the numbers to be calculated, the operation, and the relative magnitudes of the numbers.
5). **Normalization.** If the significand of the result is not in normalized format, the significand is shifted to match the normalized form, and the exponent is adjusted according to the shift value.
6). **Rounding.** The result is rounded to fit the data format.

However, for the convenience of design and enhance the throughput, we have modified some steps to suit the bit-serial data format.

The third step "addition" and the forth step "determine sign" are combined into one step. The sequence is opposite: the sign of the result is determined before the calculation.

Thus, regardless of the values of inputs A and B, the result of addition/subtraction is always positive. For instance, when A = 1.1×2^3 and B = -1.0×2^4, A + B is expected to has minus sign. In this case the adder/subtractor calculates $\overline{A} + B + 1$ = (B - A) and combines it with a sign "-".

Meanwhile, similar to the FPM, the fifth "normalization" step and the sixth "rounding" step are also combined into one step.

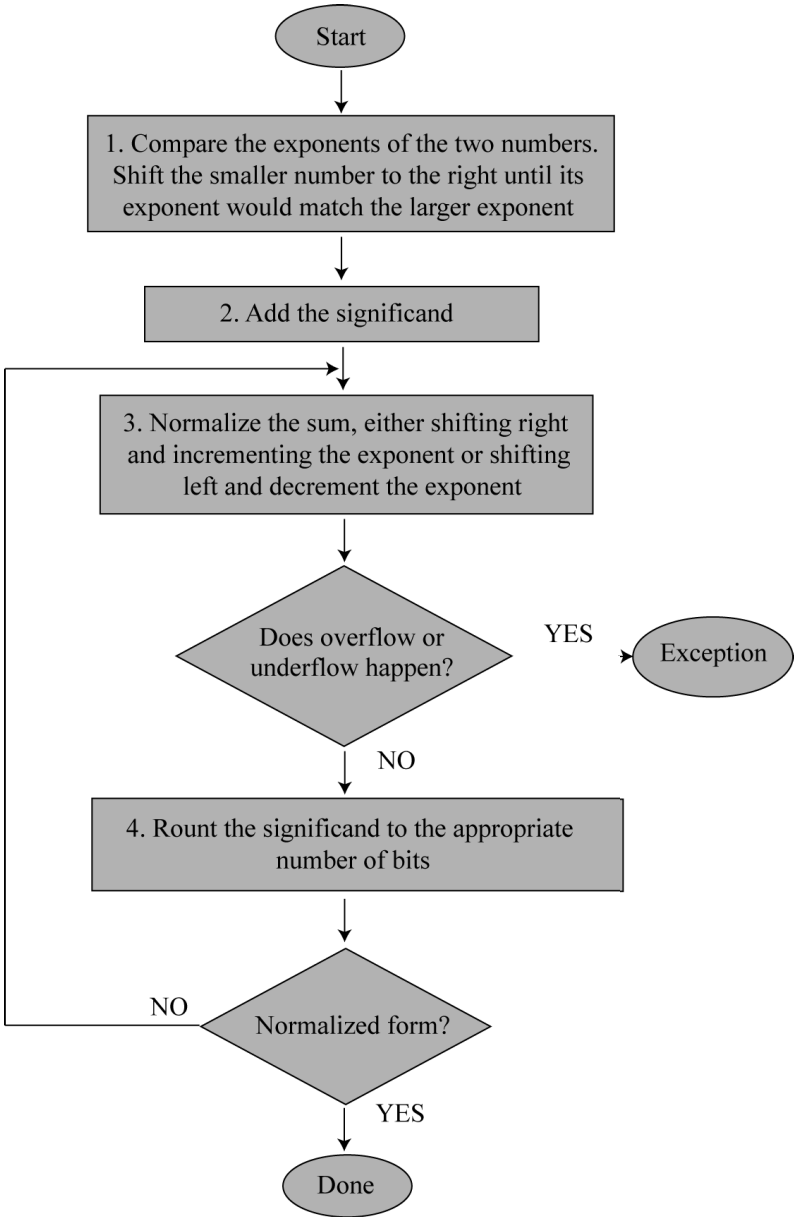


Figure 4-19: calculating flow of floating point addition

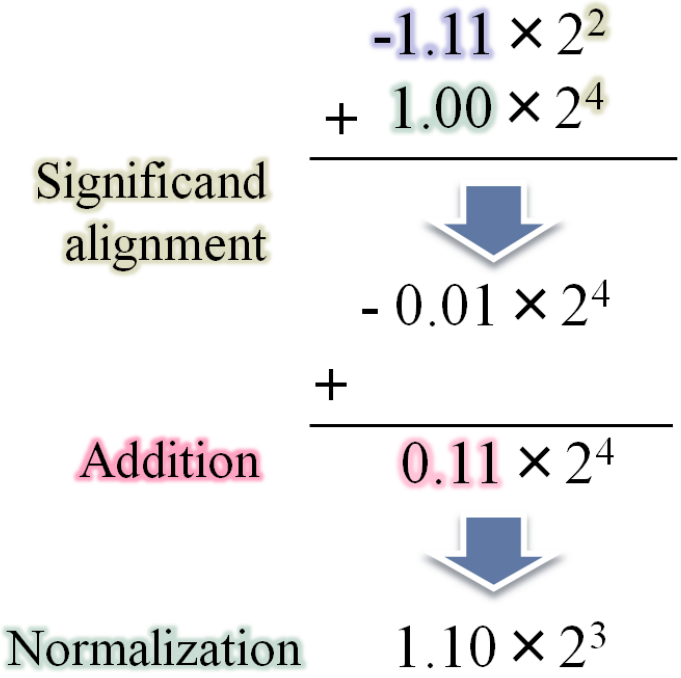


Figure 4-20: Direct imagine of floating-point addition

4.6.2 Circuit Block Design

Figure 4-21 shows the block diagram of the SFQ FPA. Similar to the SFQ FPM, the FPA mainly consists of two data paths: one for significand and the other for the sign and exponent. In our design, the calculation is executed in the following three pipeline stages: 1) alignment and rounding of the significand and determination of the operation of the adder/subtractor, 2) addition (or subtraction), and 3) normalization.

The main circuit components include a controller, which controls the operation of whole circuit components, two shifters, which shift the significand, an adder/subtractor, which performs the addition or subtraction of two significands, and two normalizers, which normalize the significand and the exponent. Several other components assists these main part to complete their function, which contains exponent subtractor, significand comparator, and novel serial-parallel converter [42].

In processing, the difference of two exponents is calculated in order to align and round the significands at the beginning of the calculation. Then the controller decodes the result and sends the shift value to two shifters. Left shifting is performed for the significand of the input number with smaller exponent. The controller simultaneously determines the operation of the adder/subtractor by comparing the sign and the values of

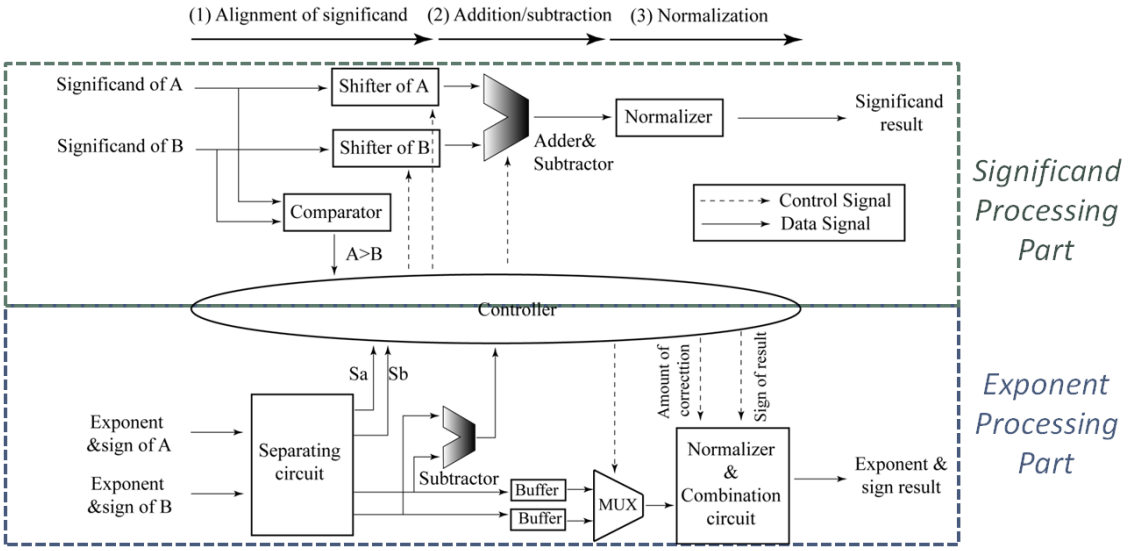


Figure 4-21: Block diagram of SFQ bit-serial FPA

the exponent and the significand of two inputs A and B.

Again, bit-serial architecture is employed in this design.

4.6.3 Controller

The controller is used to determine the operation of the FPA. It has the following 2 functions:

1). Determining the adder/subtractor's operation and,

2). Decoding the shifting value and sending it to the significand shifter.

A. Determination of the adder/subtractor's operation

As I mentioned before, the calculation result of the adder/subtractor is always positive because the sign of result is predicted by examining the sign of the numbers and the relative magnitudes of the exponents and significands. The adder/subtractor performs addition when the two significands have equal signs, and performs subtraction while the signs are opposite. The sign of result is corresponding to the larger exponent number, unless the exponents are equal.In this case, the magnitudes of the significands are compared and set the sign corresponds to the larger one.

For instance, $(A - B)$ is calculated as $A + \overline{B} + 1$ if the prediction of sign is positive, however $\overline{A} + B + 1$ (B - A) is calculated if the prediction of sign is negative.

Notice that either $A + \overline{B} + 1$ or $\overline{A} + B + 1$ is a positive number. The sign will be combined with the result of exponent like I did in the FPM. I list the behavior of adder/subtractor in all of the conditions in Table 4-3.

This circuit part can be realized by using a combination of NDRO and NDROC cells. Figure 4-22 shows the schematic of this circuit part, where Fa, Fb represent the significands (Fractions) of A and B, respectively. Signal "Ea<Eb" comes from the MSB of the result of exponent subtractor, "Fa< Fb" comes from the significand comparator, "Ea = Eb" comes from the shift value send to the significand shifter, generated by another part of the controller I will explain the next, and "Sa ≠ Sb" comes from a exclusive "OR" gate calculates the signs of the inputs data.

Table 4-3

Determination of operation of the adder/subtractor

Sa, Sb	Magnitude of exponent	Magnitude of significand	Sign of result	Operation of the adder/subtractor
+, +	—	—	+	$A+B$
-, -	—	—	-	$A+B$
+, -	Ea>Eb	—	+	$A+\bar{B}+1$
	Ea = Eb	Fa≥ Fb	+	
		Fa< Fb	-	$\bar{A}+B+1$
	Ea<Eb	—	-	
-, +	Ea>Eb	—	-	$A+\bar{B}+1$
	Ea = Eb	Fa≥ Fb	-	
		Fa< Fb	+	$\bar{A}+B+1$
	Ea<Eb	—	+	

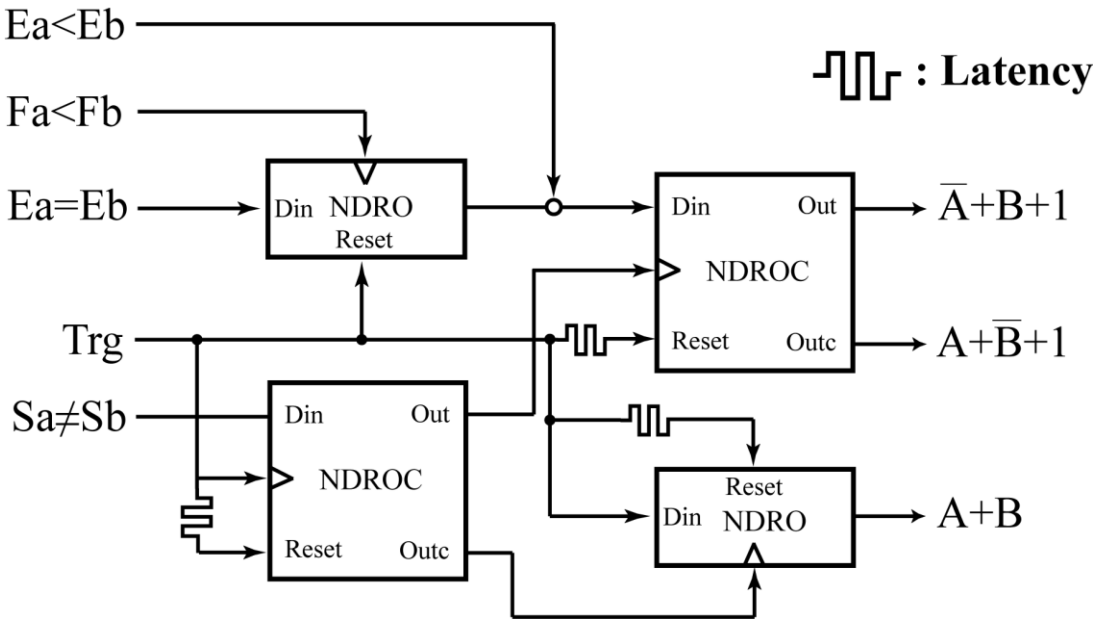


Figure 4-22: Schematic of the determination circuit part

B. ***Decoding the shift value***

This circuit part is employed as a decoder which decoding the result from the exponent subtractor and send shifting instruction to the significand shifter. We designed the decoder based on a binary select tree composed by NDROCs. 2 mirrored decoders construct a symmetric complimentary structure which can send shifting instructions to selected shifter for Fa or shifter for Fb, respectively. **Figure 4-23** shows the schematic of the half-precision decoder corresponding to the shifter for Fb. At first, the bit-serial format subtraction result is transformed into bit-parallel format via a serial-parallel converter (SPC), and then sent to the decoder to set the original status of NDROCs.

It should be noted that although the subtraction result is in the range of (-15 ~ +15), only result in range of (-10 ~ +10) will be handled. This is because the bit length of

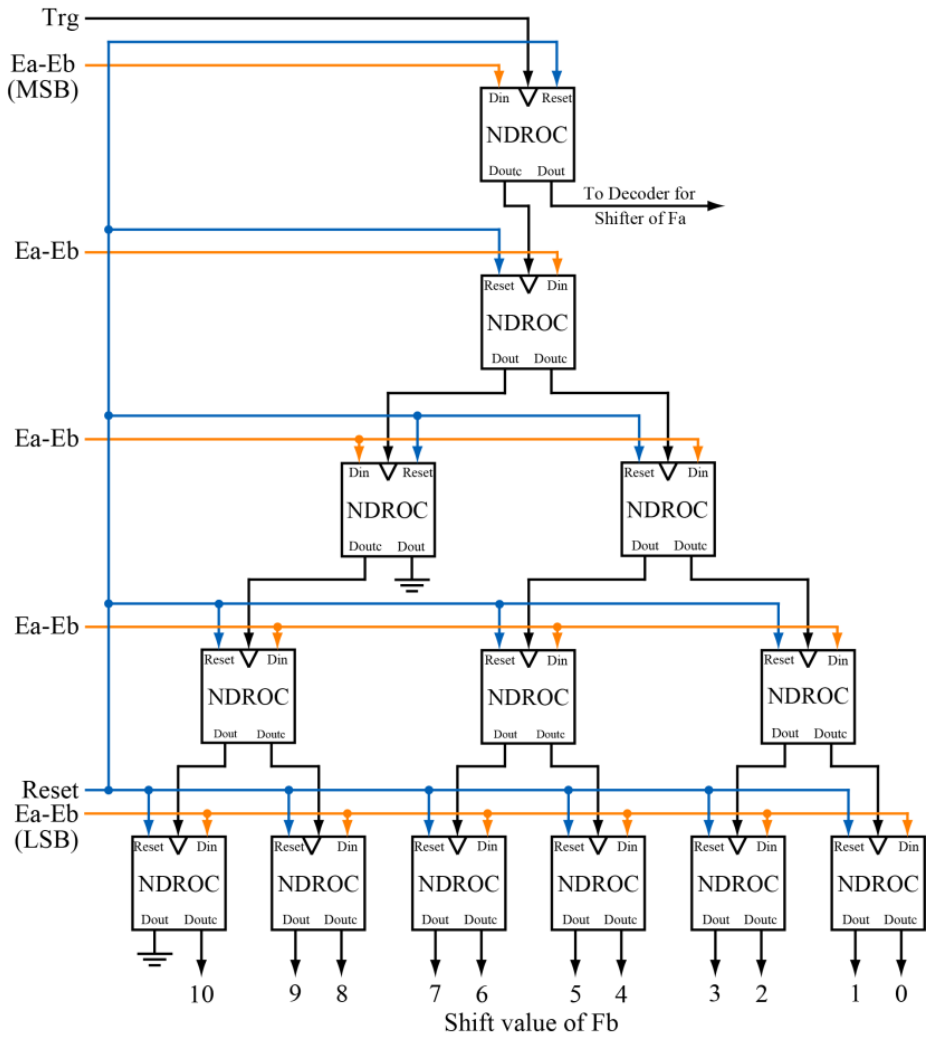


Figure 4-23: Schematic of the half-precision decoder for the shifter of Fb

significand in half-precision floating point is 11. The others will be handled as exceptions: shifting instruction will not be send to the shifterand the output of shifter will be “0”, then the result of the adder/subtractor will be the original larger input data.

4.6.4 Significand Shifter

The significand shifter is used to right shift the smaller number thus it could match the larger exponent. In this shift register based shifter, the data is moved into the next pipeline stage. Therefore, a “left shift *m* bits” operation will look like a “right shift *n* - *m* bits” operation (*n* is the bit length of the siginificand). The shifted significand is then sent to the adder/subtractor.

Figure 4-24 shows the schematic of a 4-bit version significand shifter. At first, clocks of the first pipeline stage store the data into the bottom shift register. Then the shifting instruction from the decoder enables the corresponding NDRO. After this step, clocks of the next pipeline bring the stored data output from the enabled NORO, with

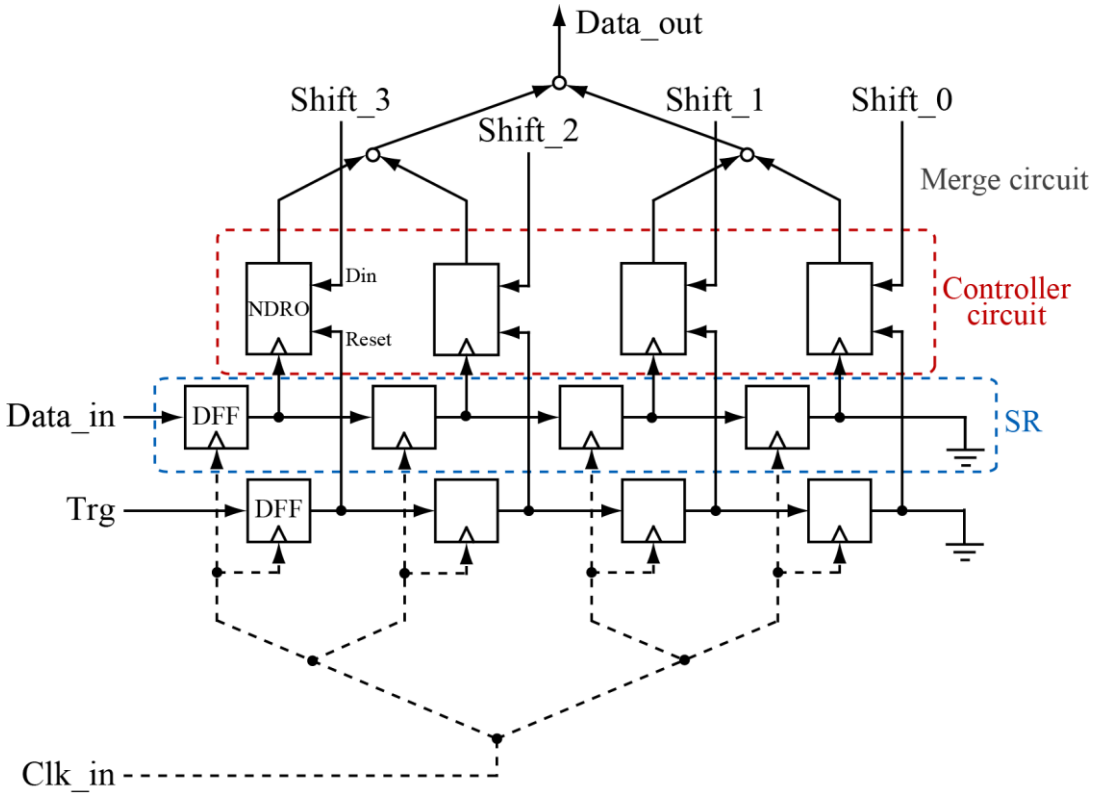


Figure 4-24: Schematic of the significand shifter (4 bits)

shifting operation simultaneously. The less significant bits of the shifted data are omitted.

4.6.5 The Adder/subtractor

As shown in **Figure 4-25**, the adder/subtractor is composed of a bit-serial adder and an operation controller. Unlike the bit-serial adder we used in processing element (PE) of FPM, the one in the adder/subtractor realizes carry propagation and reset by utilizing a NDRO [43]. While the carry is "1" the NDRO is enabled and it is disabled while the carry is "0". Additionally, the NDRO can be set to "1" before the calculation then the adder could perform "A+B+1". It is important when we are doing subtraction.

The operation controller controls the data inversion thus $A + \overline{B}$ and $\overline{A} + B$ can be performed. This function is implemented by enabling and disabling the NDROs. Once the NDRO is enabled, it output "1,1,1…..1" and the exclusive "OR" (XOR) inverts the data (Fa, Fb).

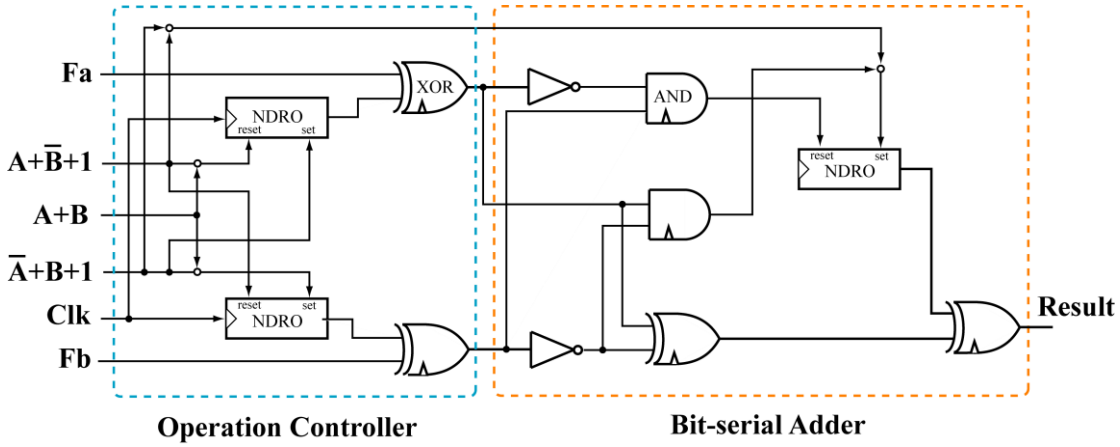


Figure 4-25: Schematic of the adder/subtractor

4.6.6 The Normalizer

The normalizers are applied to convert the calculating result into standard format, and they are more complicated than that in FPM. Both significand processing part and exponent processing part have their isolated normalizer, and the algorithm is different.

A. The significand normalizer

The significand normalizer is similar to the shifter but has expanded function. The shifter is applied to right shift the data, however the normalizer must be able to left shift the data in case of overflow. **Figure 4-26** shows a block diagram of the normalizer of a 4-bit version. This normalizer can perform shifting operation ranging from 4-bit left shift to one-bit right shift.

The normalizer is composed of a shifter in the top and a D2FF based serial-parallel converter (SPC) in the bottom. The shifter has the same algorithm with our significand shifter, realizes shifting operation by changing the pipeline stage. SPC is adopted in order to generate shifting instruction. For example, the MSB = 1 corresponds to left

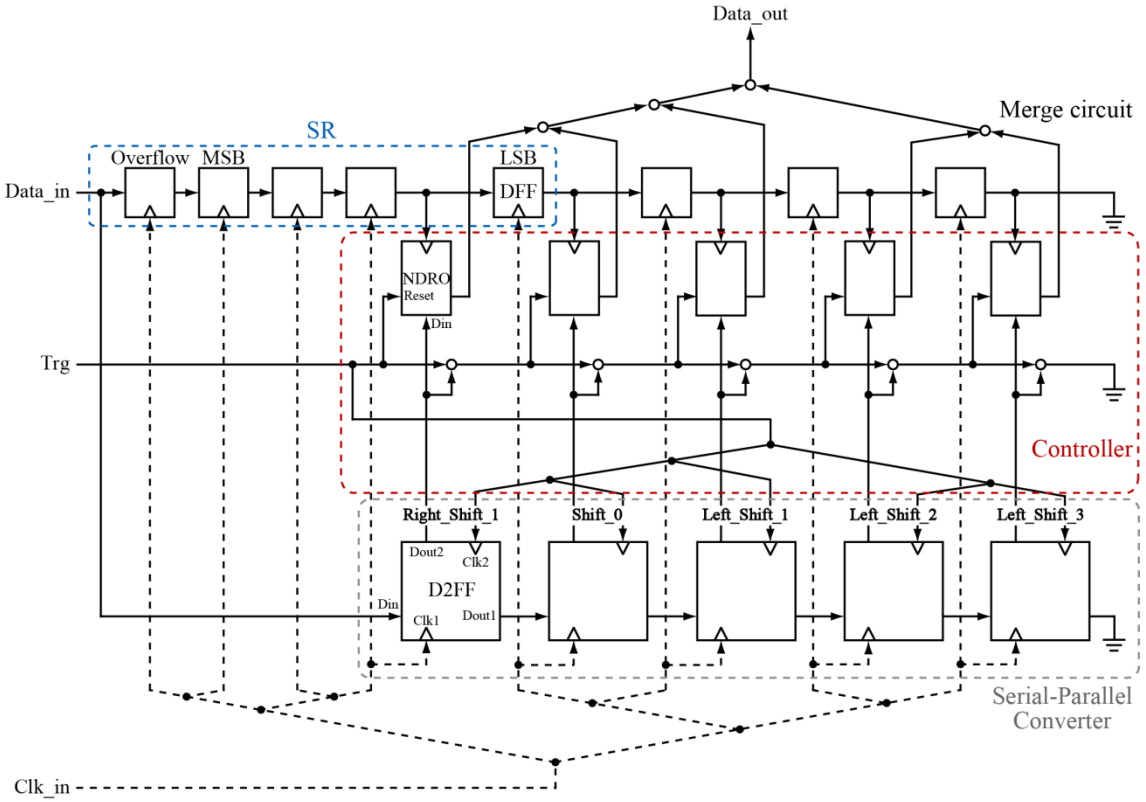


Figure 4-26: Schematic of a 4-bit version significand normalizer.

shift 1 bit (overflow), second significant bit =1 corresponds to no shifting and so on. It should be noticed that a higher "1" bit has higher priority than all the lower "1" bits and would reset those corresponding NDRO. For example, when the data is $(00.111)_2$, the third "1" bit will enable the third NDRO and disabled the fourth and fifth enabled NDRO, then the data is output from the third NDRO and right shifted for 1 bit.

B. The exponent normalizer

If the significand result is right/left shifted for *n* bits, *n* must be subtracted/added from/to the exponent result. **Figure 4-27** shows a block diagram of the normalizer for the exponent. The normalizer is composed of a D2FF, a NOT gate, a binary counter composed of T1s (resettable toggle flip-flops), and an adder/subtractor similar to the one I explained previously.

At first, the result of the addition (or subtraction) is sent to the binary counter from the LSB to the MSB after being inverted by the NOT gate to count the number of "0" bits in the input data. Because the last inputs of "1" in the data resets the counter, the

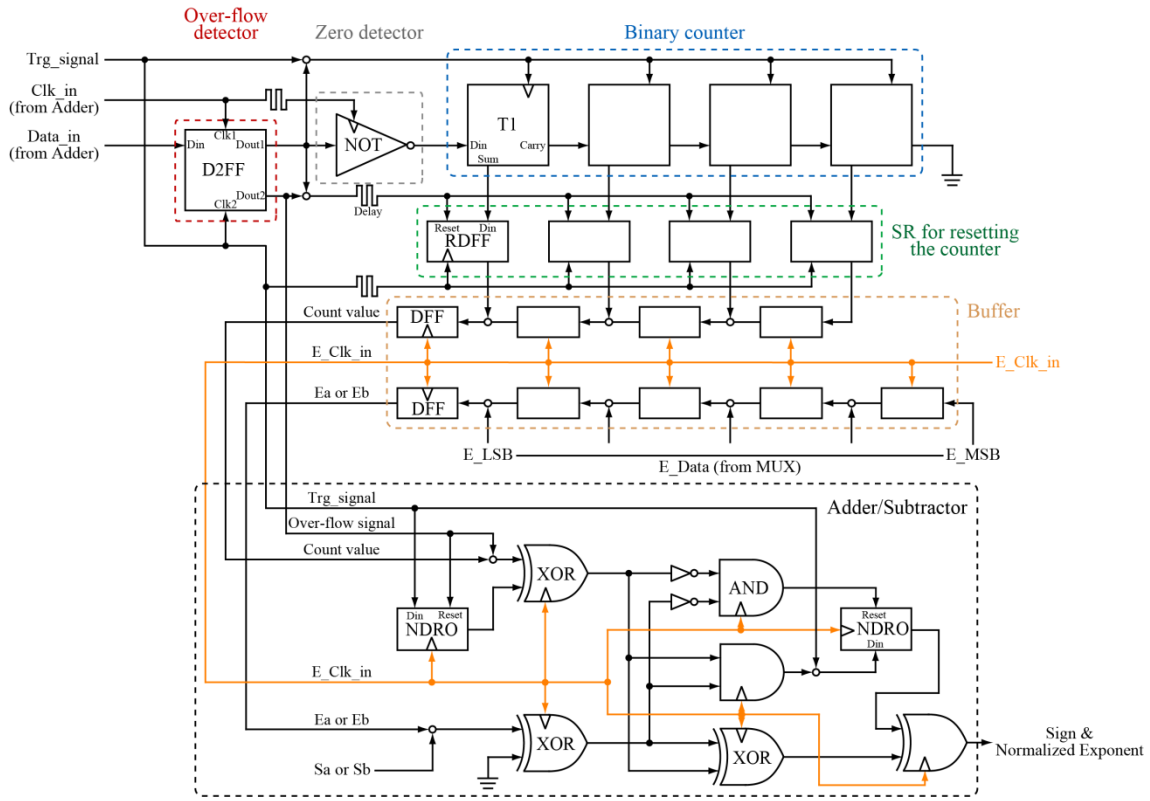


Figure 4-27: Schematic of the exponent normalizer.

resultant value in the counter corresponds to the number of the last inputs of "0". When a "Trig" pulse is applied to the normalizer, the counted results are read out by the RDFFs, and sent to the DFFs. This data is then send to the adder/subtractor and subtracted from the larger exponent selected by a multiplexer.

The D2FF is used to detect the overflow in the result. In this case the adder/subtractor performs addition function and "1" is added to the larger exponent.

C. Asynchronous exponent normalizer

When we were measuring the half-precision FPA, it was found that the DC bias margin of the exponent normalizer is very narrow [41]. The reason was considered as timing conflict in RDFFs (resettable SR) at high clock rate. Since it requires "write_in" and "reset" to be completed in one clock cycle, the timing restriction is more serious than other cells. Therefore, I designed an asynchronous type exponent normalizer for the single-precision FPA which has wide timing range.

The asynchronous normalizer is composed by a two's complement encoder which realizes subtraction and an 8-bit T1 based ripple-carry adder (the same structure of that in FPM's exponent part). This ripple-carry adder is also used as a binary counter. Since we already have shifting instruction generated by the significand normalizer, it can be

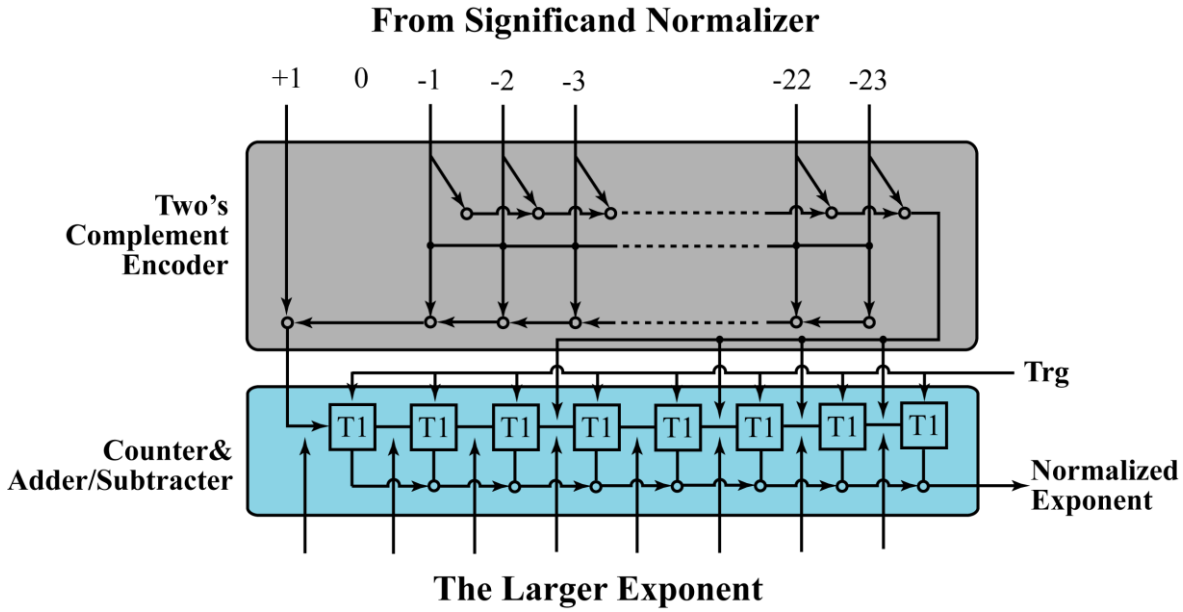


Figure 4-28: Schematic of the asynchronous type exponent normalizer.

used and encoded into the subtraction/addition value. The encoder is based on a ladder type clock generator, and by setting the first, second, third and fifth T1 in the adder. The shifting instruction is converted into numerical value stored in the 8-bit adder in its two's complement format. For example, when the significand is right shifted for 1 bit, we should add $(00000001)_2$'s two's complement $(11111111)_2$to the larger exponent. The clock generator will generate 23(00010111) pulses and send them into the counter (ripple-carry adder). Simultaneously the counter is set into (11101000), thus we obtain $(11111111)_2$. The ripple-carry adder then adds this value to the larger exponent and the normalization is completed.

This normalizer occupies approximately 50% more area than the previous one, however the timing restriction is much less. It would be valuable when we implement double-precision (64 bits) FPA.

4.6.7 Other Assistant Component

Those 4 components I explained above composed the main structure of FPA. There are several simple assistant components applied to help those 4 complete their function.

A. Exponent subtractor

This subtractor is used to calculate the difference between the two exponents, thus the controller can determine how many bits the significand should be shifted.

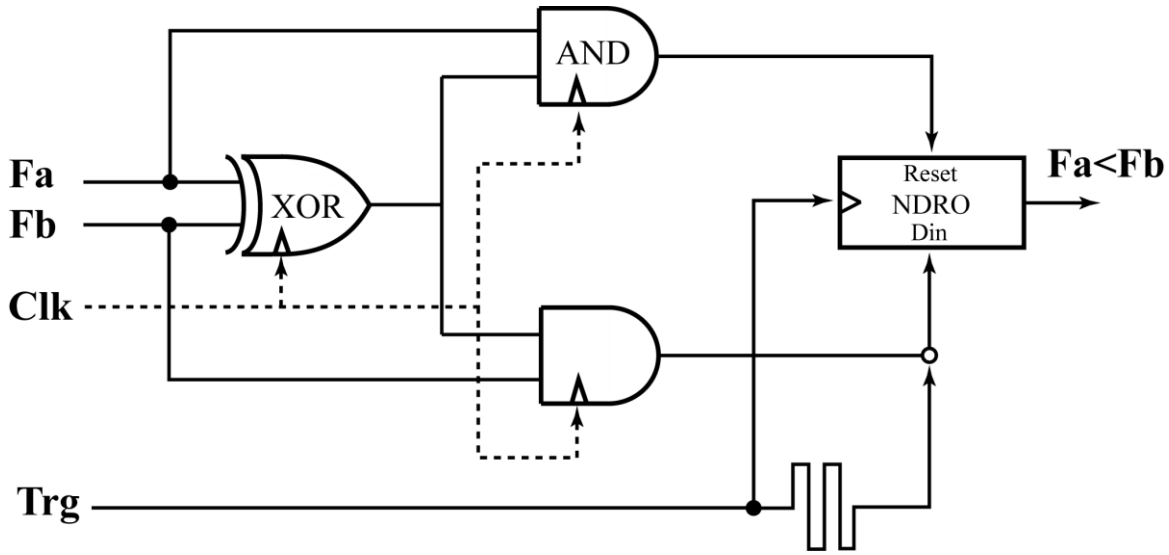


Figure 4-29: Schematic of the significand comparator

Additionally, the MSB of the subtraction result is used as “Ea<Eb” for the controller to determine the operation of the adder/subtractor. The structure can be considered as a operation fixed adder/subtractor.

B. Comparator

The comparator is applied to compare the magnitude of the significands and generate “Fa< Fb” for the controller. We applied a bit-serial structure shown in **Figure 4-29.**

While two numbers have different digital (bit), the one with a higher “1” bit is the larger number. The XOR gate examines whether there is a different bit, and the 2 ANDs check the “1” bit is from Fa or Fb. After enabled and disabled repeatly, the final status of the NDRO represents which significand has the higher “1” bit.

C. Multiplexer

This component is used to select the larger exponent. It is composed by two D2FF based SR used to store the two exponents. The controller uses “Ea<Eb” to generate “Ea_Trg” or “Eb_Trg”, and send them to the multiplexer to select the larger exponent.

4.7 Implementation and On-chip High-speed Test of FPA

Based on the hardware algorithm introduced above, we designed and measured the single-precision FPA. The target clock rate is 50 GHz as well as the FPM with DC bias margin from -20% to +25%. The highest frequency is estimated to be 75 GHz. The Scale of the single-precision FPA is a little smaller than the FPM: The overall circuit contains 19850 Josephson junctions which includes on-chip high-speed test circuits, such as in/output shift registers and an on-chip clock generator. Size and power consumption of the FPM are 4.66 mm ×5.88 mm and 4.92 mW, respectively. A microphotograph of the FPA is shown in **Figure 4-30**.We should notice although the FPA cost less cells and junctions, it occupies more area. This is because the PTL wire we used in shifter and normalizer cost large space.

Figure 4-31 shows two of several test data patterns we used to test the FPA. The inputs of the significand and exponent are denoted as (S_X, S_Y) and (E_X, E_Y), respectively. Similarly, the outputs of the significand and exponent are denoted as S_out and E_out, respectively. In the first data pattern, the input data X and Y are "-$(11111111111111111111111)_2$ × exp $(10000000)_2$" and "-$(11111111111111111111111)_2$ × exp $(1111111)_2$," respectively. The calculation result of the significand is expected to be overflow and should be left shifted by 1 bit. On the other hand, in the second data pattern, X is "-$(11111111111111111111101)_2$×exp $(1111111)_2$" and Y is "$(10000000000000000000000)_2$ × exp $(10000000)_2$." The significand result for this input pattern is expected to be right shifted by 23 bits. In **Figure 4-31**, one can see that the correct answers "-$(10111111111111111111111)_2$ × exp $(10000001)_2$" and "$(10000000000000000000000)_2$ × exp $(1101001)_2$" are obtained.

DC bias margins of each circuit block at low speed and at 50 GHz are shown in **Figure 4-32(a)** and **Figure 4-32(b)**, and frequency versus measured and simulated DC bias margins of the FPA are shown in **Figure 4-33**. The measured DC bias margin at the target frequency of 50 GHz is 97.39% - 111.23%. The shifter limited the margin of the whole circuit, and the maximum operation frequency is 58 GHz, similar to that of single-precision FPM.

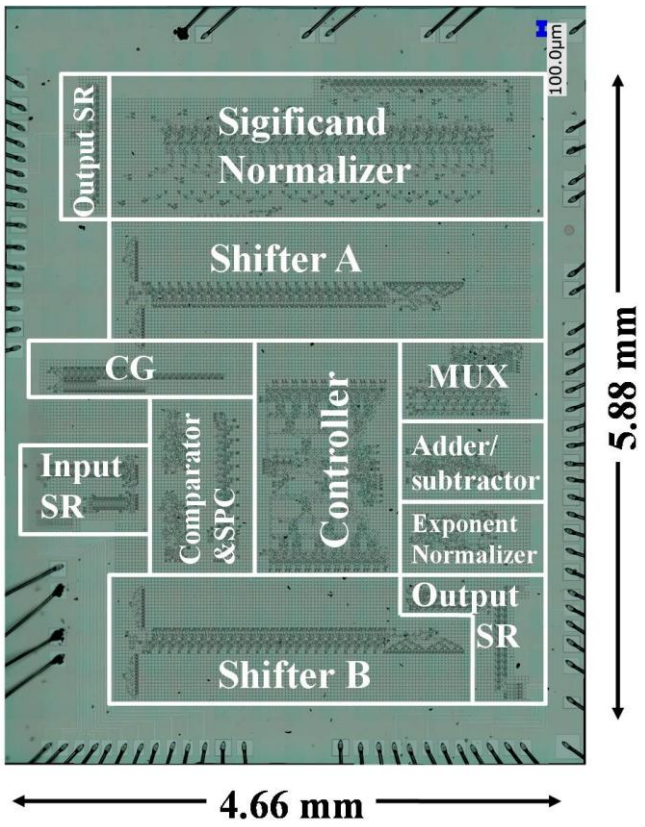


Figure 4-30: Microphotograph of the single-precision FPA

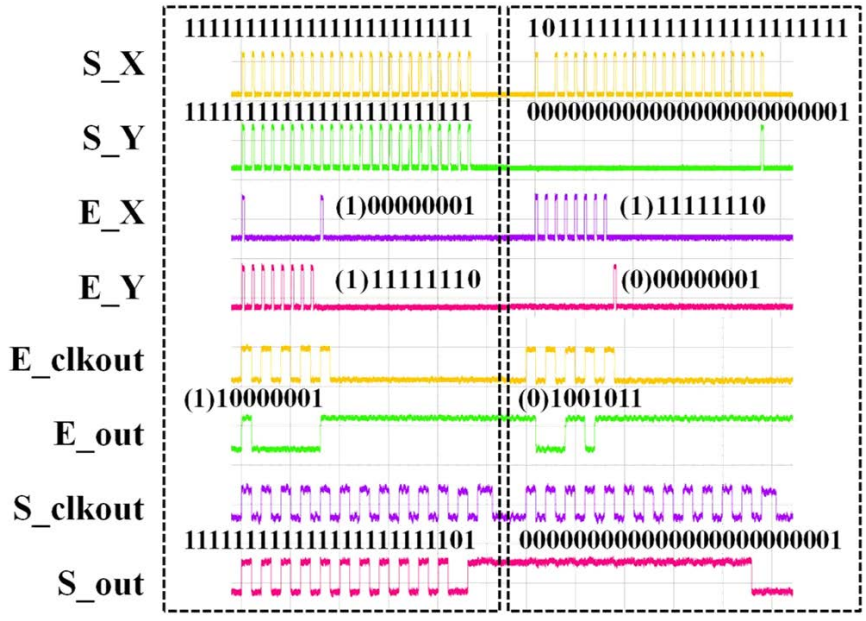
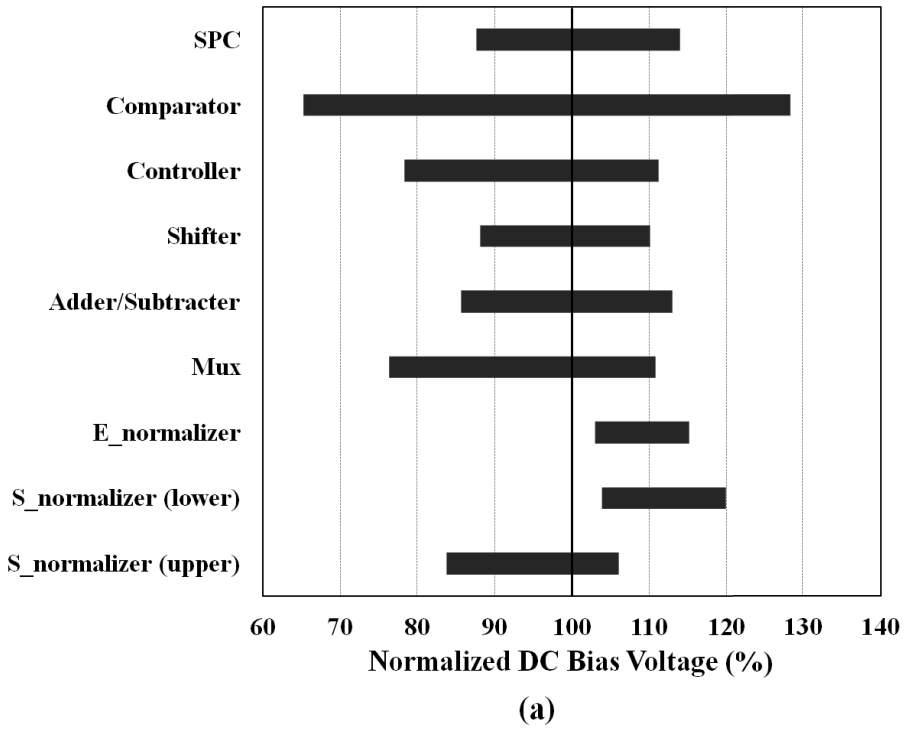


Figure 4-31: On-chip high-speed test waveform of the single-precision FPA.



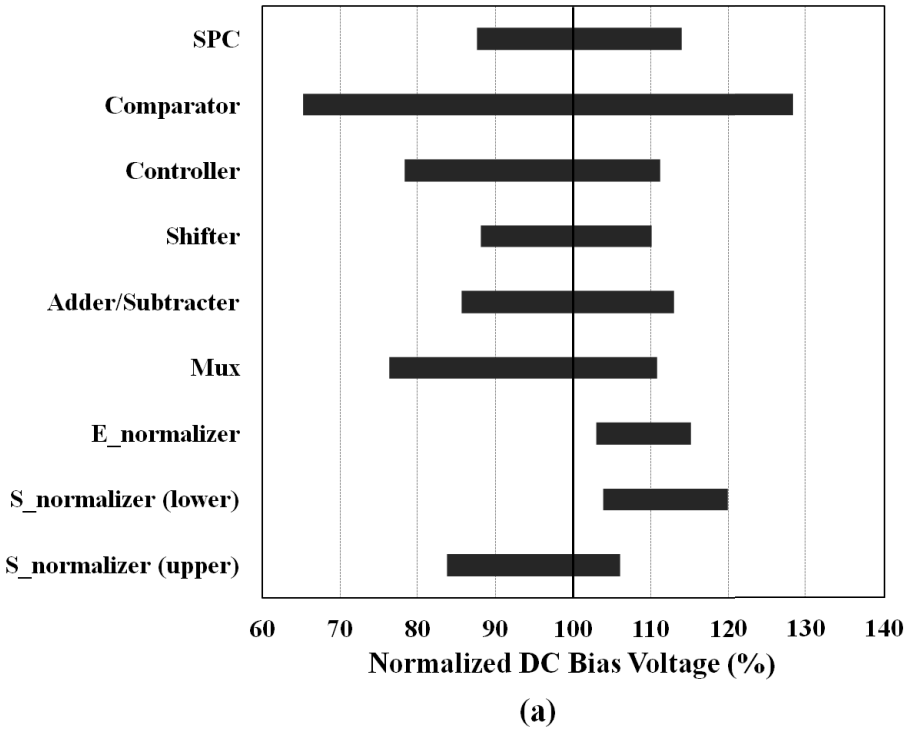


Figure 4-32: DC bias margins of each circuit block of single-precision FPA.
(a) Low speed and (b) 50 GHz

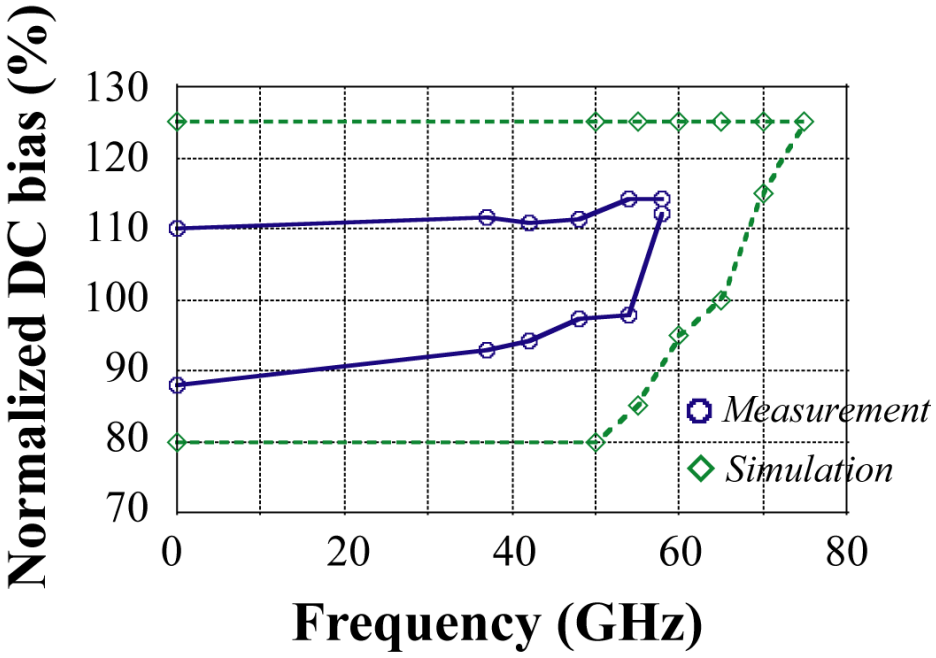


Figure 4-33: Frequency versus DC bias margins of the tested single-precision FPA.

Solid line shows the measurement result and dotted line shows the Verilog

4.8 Performance Assessment

Table 4-4 shows the performance details of half- and single-precision FPMs and FPAs. Throughput is calculated by f/n_c, where f is the clock frequency and n_c denotes the necessary clock number for calculating one floating-point number. n_c is determined by the length of the longest intermediate data, and given by $n_s + 2$ and $n_s + 1$ in the FPA and the FPM, respectively, where n_s is the length of the significand. Additionally, the latency of both FPA and FPM is $2n_s + 1 \times \tau_c$, where τ_c is the clock period.

The throughput of our single-precision FPUs is approximately 2.3 GFLOPS, and we can achieve the total throughput of nearly 10 TFLOPs by using these FPUs in the LSRDP system. Recently developed 20 kA/cm2 Nb process would probably increases the performance by the factor of 1.4-1.5.

Low power consumption is the attractive merit of SFQ circuits comparing with CMOS circuits. If we adopt the LR-biasing technique [16] to reduce the static power dissipation of the FPUs, the power-performance efficiency could be nearly 9000 GFLOPs/W. This is approximately 2250 times larger than the Intel Xeon Phi™ coprocessor [44] , the CMOS processor for the components of the world fastest supercomputers at present. Additionally, power-performance efficiency of the whole LSRDP would achieve about 3000 GFLOPs/W, 1560 times better in contrast of "Tian-he 2"'s 1.92 GFLOPs/W.

Table 4-4: The performance details of FPUs and LSRDP

FPU	Maximum Frequency (GHz)	Throughput (GFLOPS)	Power dissipation* (mW)	Efficiency* (GFLOPS/W)
Half-precision FPA	55.1	4.23	0.13 (2.90)	32500
Half-precision FPM	72.0	6.00	0.13 (2.83)	46000
Single-precision FPA	58.0	2.23	0.25 (4.92)	8920
Single-precision FPM	59.0	2.36	0.26 (5.76)	9076
The entire system	50.0	8.192 [TFLOPs]	3.2 [W]	~3000

*The LR-biasing technique was assumed in the estimation. The number in the blanket is the measured power dissipation using the conventional biasing scheme.

4.9 Summary

We designed, implemented and demonstrated SFQ bit-serial FPAs and FPMs. The FPUs were fabricated using the AIST advanced 10 kA/cm^2 Nb process. Their high-speed operations were successfully confirmed by on-chip high-speed tests, and the maximum frequency was evaluated to be 72 GHz and 59 GHz for half- and single-precision FPM and 58 GHz for single-precision FPA.

The throughput of our single-precision FPUs is approximately 2.3 GFLOPS, by which we can achieve the total throughput of nearly 10 TFLOPs in the LSRDP system, with power-performance efficiency of 3000 GFLOPs/W.

On the other hand, the results of this study indicate that the effect of large bias current is not a major obstacle for the implementation of SFQ circuits with the scale of 20000 junctions if proper current management is fulfilled. However, though the effect of large bias current is not serious for this scale, the way to supply the bias current somewhat affects the bias margin of each circuit block.

5

Josephson/CMOS Hybrid Memory

5.1 Introduction

In the previous chapter, I have introduced relatively large-scale SFQ processors which enable dozens GHz operation and ultra-low power dissipation. These processors (FPUs) are sufficient to perform fundamental calculation in computer and able to compose complicated superconducting computing system. However, with only processor, the system is not a computer: the memory is another required component.

Unfortunately, due to the low integration density and small driving ability of SFQ circuits, it is difficult to realize large-scale, high-performance superconducting memories based on pure Josephson circuits. The size of largest superconducting RAMs which have been successfully demonstrated up to now is only 4k bits [45]. On the other hand, CMOS device has high integration density and mature memory technique has already been developed. Therefore we propose an idea to bypass the above demerits: using Josephson/CMOS hybrid technique.

A Josephson/CMOS hybrid memory [46, 47] is a promising memory technology because the advantages of both Josephson and CMOS devices can be utilized; high-speed and high-sensitive properties in Josephson devices as well as high-density and large driving ability in CMOS devices. In the Josephson/CMOS hybrid memory, the clock rate is a bit lower and the power dissipation is slightly higher than that of superconducting RAMs. However the capacity is much larger. Considering all these factors, the Josephson/CMOS hybrid memory is suitable to be adopted as the L2 cache in the future superconducting computing system. **Figure 5-1** shows the hierarchy of our prospective superconducting memory system, with the capacity growing larger and speed becoming slower from top to bottom. For each core of the CPU, it requires L1 cache of 32- or 64-kb and L2 cache of 256- or 512-kb (or a several Mb shared L2 cache for multi-core structure, i.e. Intel's Xeon) to achieve more than 90% memory hit rate and acceptable latency (proportional to capacity). In the prospective of superconducting

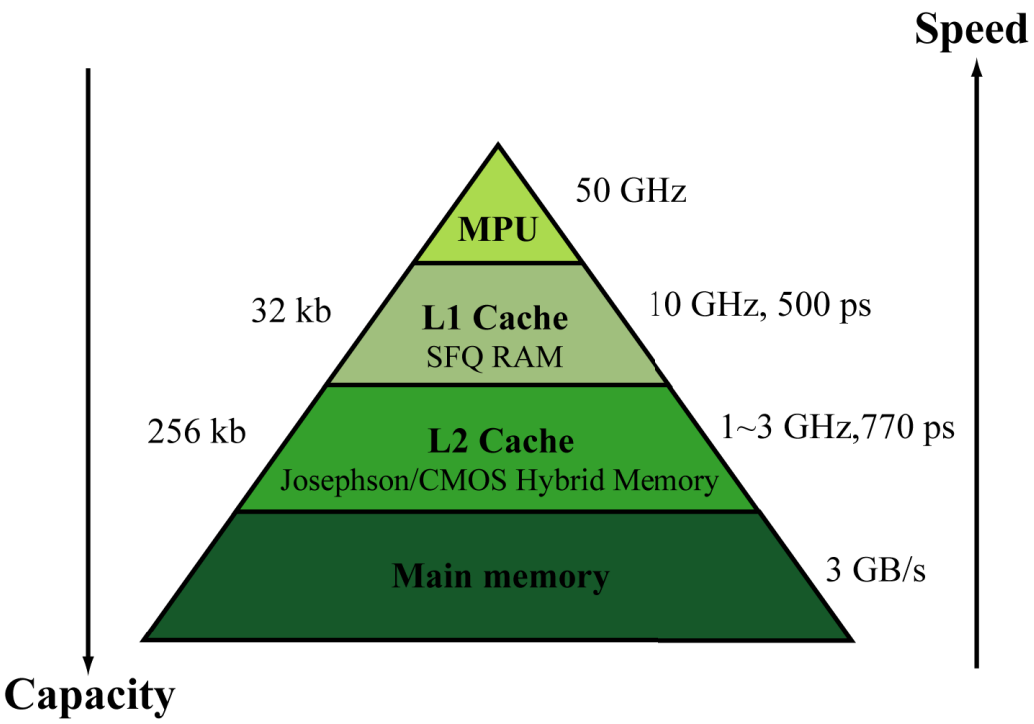


Figure 5-1: Hierarchy of superconducting memory system

computing system, there will be a 32-kb L1 cache with 500 ps latency and 256-kb L2 cache with 770 ps latency in one core. Additionally, with microprocessor using CORE 1γ [12] structure and 4 FPUs I introduced in chapter 4, performance of one core is 2000 MIPS and 9.2 GFLOPS, and it requires a maximum memory bandwidth of 3 GB/s.

In this chapter, I will introduce how we implement a reliable fully functional 64-kb Josephson/CMOS hybrid memory. This work includes the design of Josephson interface circuit, the enhancement of stability of the multi-channel operation and low-power dissipation approach. I will also show the experiment results of 2 versions of hybrid memory. By simply adding 2 address control interface lines, the capacity could be expanded into 256-kb, which matches the requirement of a single core of CPU.

5.2 Architecture of Josephson/CMOS Hybrid Memory

Figure 5-2 shows a block diagram of the Josephson/CMOS hybrid memory. The main parts of the system were composed of a Josephson/CMOS interface, CMOS decoders, a CMOS static random access memory (SRAM) cell array and Josephson current sensors. The CMOS memory cell array consists of eight blocks of 256 x 32-bit cell array, where 32-bit data corresponding to single-precision data are written and read out. However, in this study, we reduced the data width to 8-bit due to limited pad numbers.

Since there is huge difference in amplitude and waveform between an SFQ pulse and a CMOS signal, two stages of amplifiers are employed as an extremely significant interface between SFQ circuits and CMOS memories. The first stage amplifier is an AC-biased Josephson latching driver (JLD) [48], which amplifies a few hundred μV of SFQ pulses to 40 mV-level signals. This level is then amplified into CMOS voltage by the CMOS differential amplifier [49], the second stage of the interface circuit. Two CMOS decoders then send these signals into correct address, perform write/read

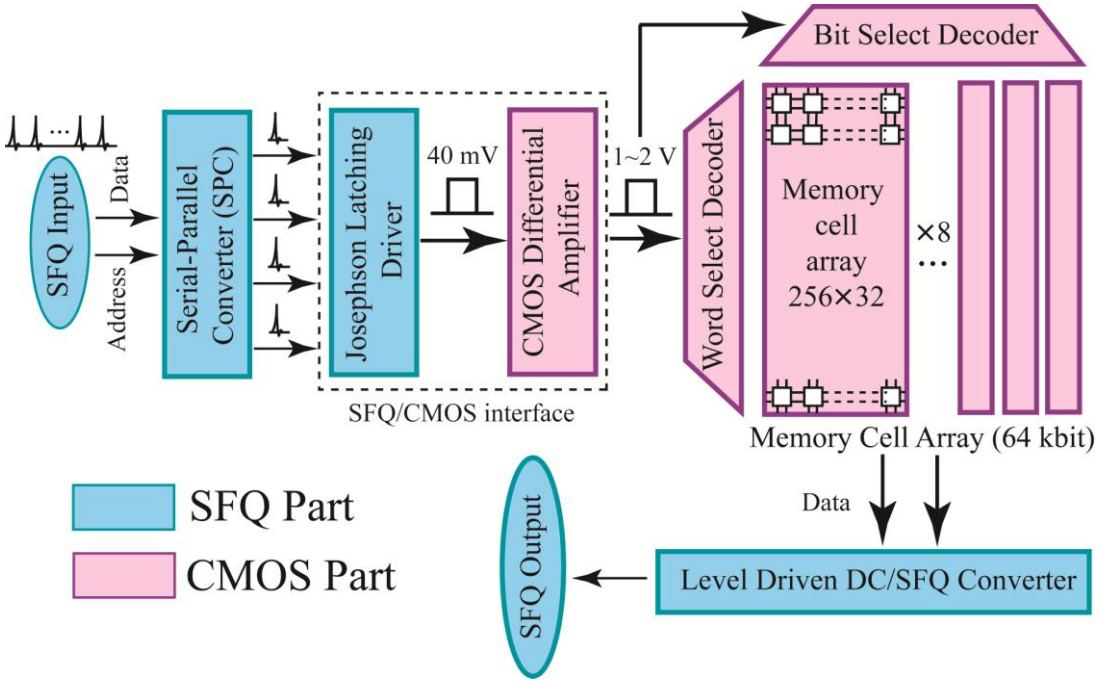


Figure 5-2: Architecture of Josephson/CMOS hybrid memory

execution in the CMOS SRAM. The output of CMOS SRAM is detected by the Josephson current sensor. In **Figure 5-2**, the blue blocks represent Josephson circuits and pink blocks represent CMOS circuits.

The CMOS Circuits in Hybrid Memory

Self-biased CMOS differential amplifiers are used as second-stage amplifiers to amplify a 40 mV signal to a CMOS voltage level. The CMOS differential amplifier has high immunity to noises compared with a single-end amplifier. It was shown that the self-biased CMOS differential amplifier is suitable for the interface of the hybrid memory in terms of speed, power consumption and robustness.

Eight-transistor (8T) SRAM cells to build the CMOS static RAM [50]. The cell offers high-speed readout ability with good read/write stability. It provides about readout current of 200 μA to drive the following Josephson current sensors. We use 3-bit block lines to access eight blocks of memory cell arrays, and 8-bit address lines to access 256 word lines. The access time measured at 4.2 K ranges from 1.32 ns to 1.47 ns, corresponding to address “0” to address “255”.

The schematic of the CMOS differential amplifier and 8T SRAM cell are shown in **Figure 5-3** and **Figure 5-4.**

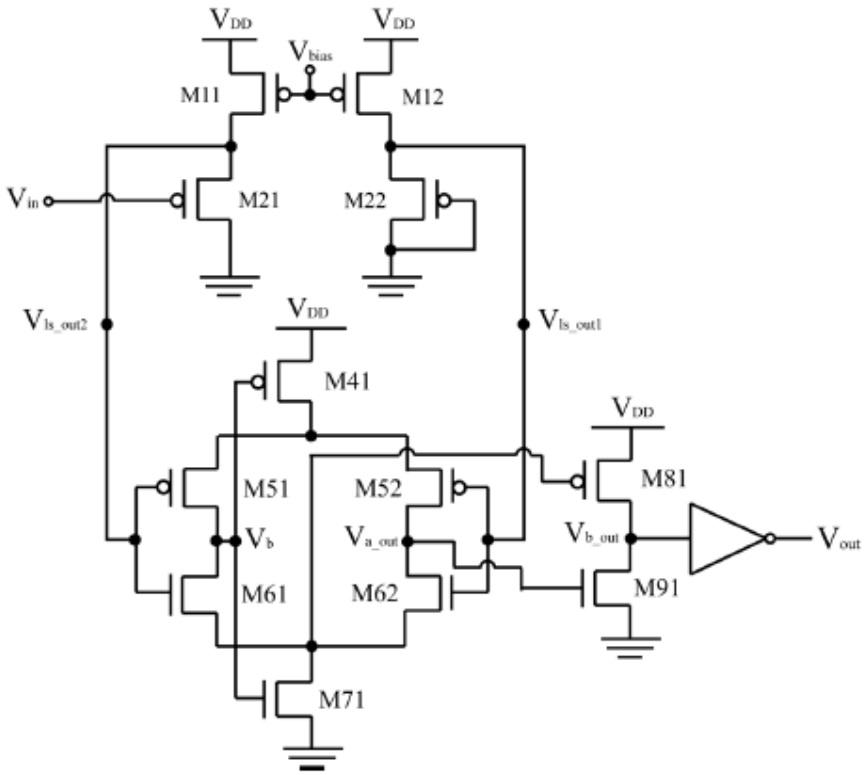


Figure 5-3: PMOS-input source-follower self-bias differential amplifier.
t_p = 367 ps, power dissipation = 733 μW, using Rohm 180nm CMOS process

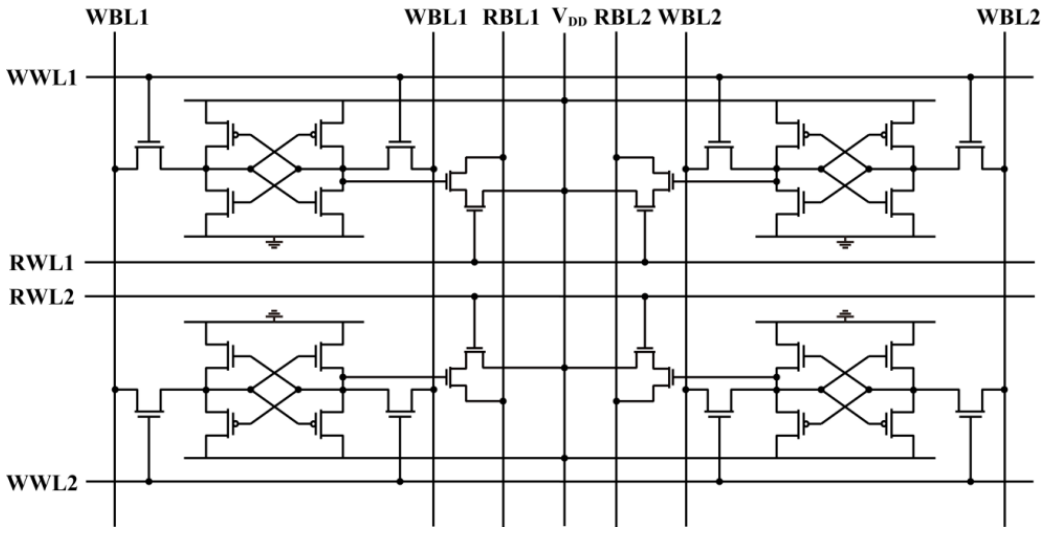


Figure 5-4: Schematic of 8T SRAM cell.

5.3 Josephson Latching Driver

The first stage of the interface circuit is Josephson latching driver (JLD), also called Suzuki stack. It is an AC biased high-speed, high gain Josephson amplifier. **Fig 5-5** shows the schematic of the JLD, which is composed of two parallel 17/16-junction stacks of under damped Josephson junctions. When an SFQ pulse is input to the terminal "din" in high-level period of the AC bias, the left and right stacks switch alternatively, which generates 40 mV-level voltage amplitude at the output terminal. When the AC bias becomes low-level, the two stacks stop switching because of the lack of bias current, and the output is reset to "0". I designed the JLD considering the following 2 factors.

5.3.1 AC Bias Margin

This is the most significant factor in design. In order to obtain wide bias margin, several modifications were adopted. I removed the ground plane under the stacks to reduce the parasitic capacitance since the parasitic capacitance has increased the bit error rate (BER) [51]. The bias resistor R_b has been increased and achieved impedance matching

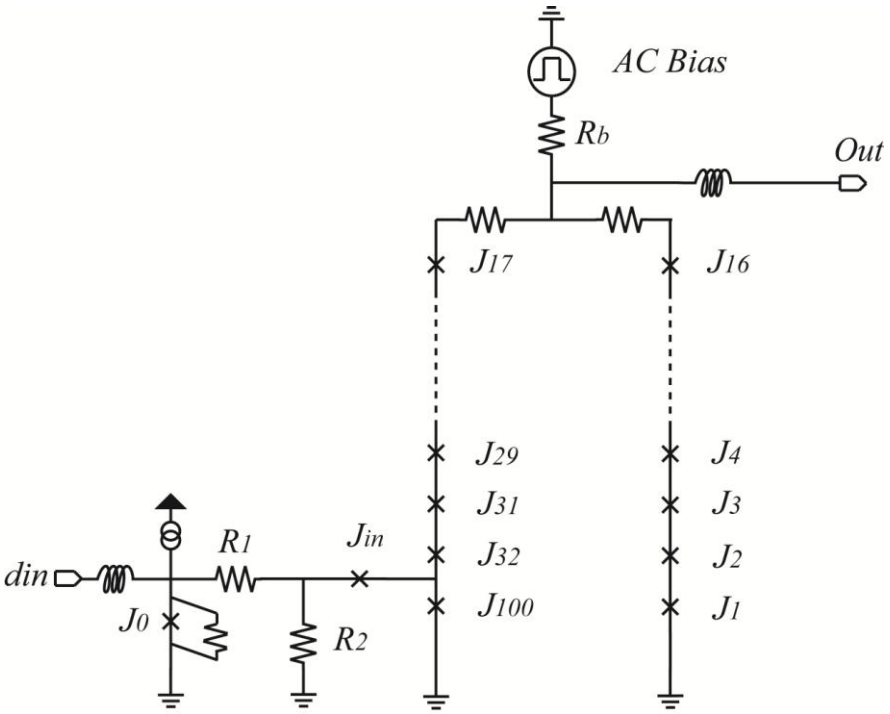


Figure 5-5: Schematic of the Josephson latching driver
Critical current of the JJ in the two stacks is 400 μA

in order to obtain high stability of the output voltage level [52]. And a pre-amplifier is added in front of the stack thus the input sensitivity could be highly increased.

In conventional design, the SFQ pulse is considered to be input into the stack directly. However the drive ability of single SFQ pulse is relatively small since its amplitude is only several hundred μV. Thus the JLD is not able to be switched unless supplied with very high bias voltage. Adding a pre-amplifier can increase the sensitivity considerably hence the JLD can work at low bias voltage.

I have attempted 3 kinds of pre-amplifier, including internal 4JL gate [53], external 4JL gate and multi-flux quantum (MFQ) driver [54]. Both external 4JL gate type and MFQ driver type has margin more than ±20% and surpass the internal 4JL gate type, but the bias margin of MFQ driver is much narrower than the JLD, thus I adopted external 4JL gate type as the final design. A schematic of the modified JLD is shown in **Figure 5-6**.

5.3.2 Parameter of Output RLC Load

It is important to design the parameter of output RLC load since it is related to the output stability and latency. The output terminal is considered connected directly to the CMOS amplifier via bonding wire and the schematic is shown in **Figure 5-7.** A careless design lead to a typical unstable output wave form is shown in **Figure 5-8(a)**.

Generally, the capacitance should be reduced as possible since it caused extremely large latency by RC time constant, thus I removed the ground plane under the output strip line and the pad. The resistance and inductance could release the effect of capacitance but on the other hand they also increase the latency, thus there is a tradeoff between the two targets, low latency and high stability. Designing the RLC parameter to achieve critical damping is a good choice [55]. Thus we have:

$$L_o = \frac{1}{4} R_o^2 C_{pad} \tag{5.1}$$

Simulation in JSIM shows there should be an at least 120 ohm output resistor. However considering the length of the bonding wire is uncertain, I added a larger 150 ohm one. An output waveform in well designed output load is shown in **Figure 5-8(b)**.

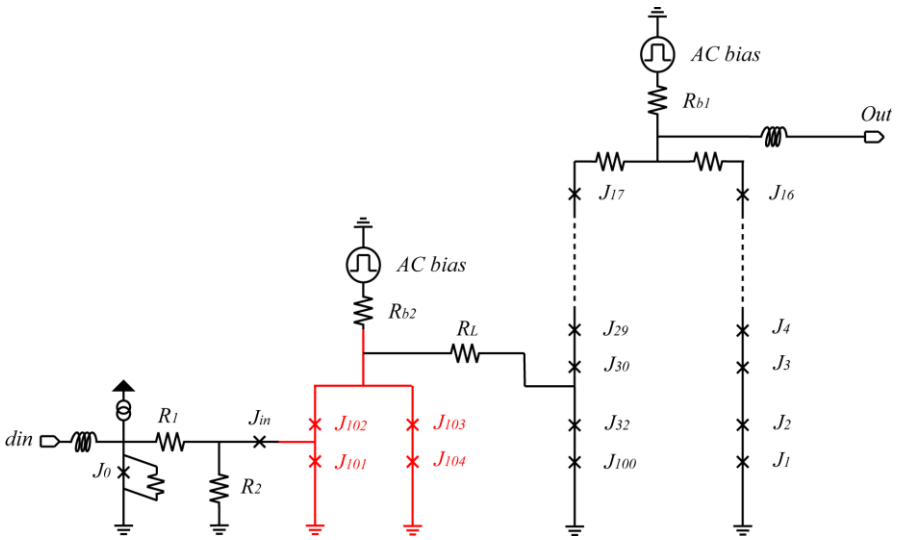


Figure 5-6: Schematic of the external 4JL gate type JLD

$J_{101} = J_{102} = 100\ \mu A$, $J_{103} = J_{104} = 300\ \mu A$

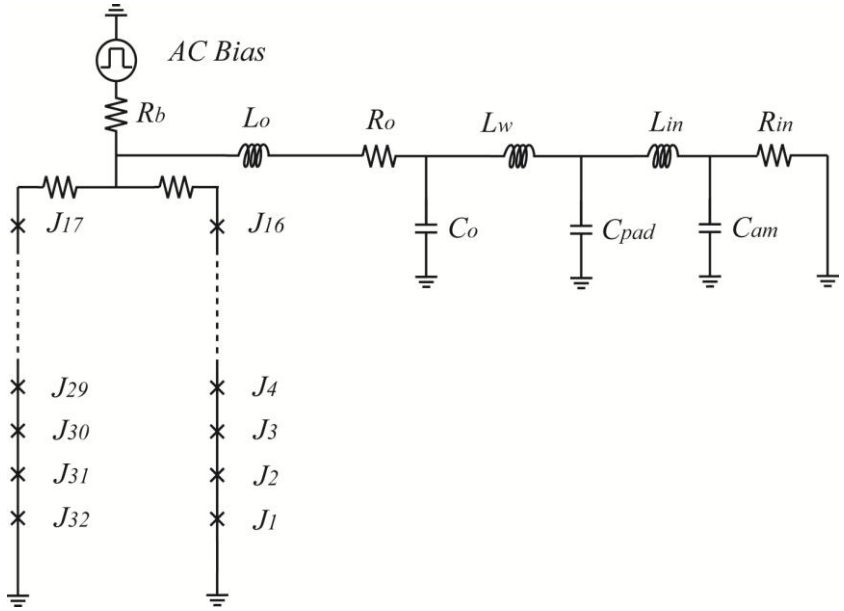
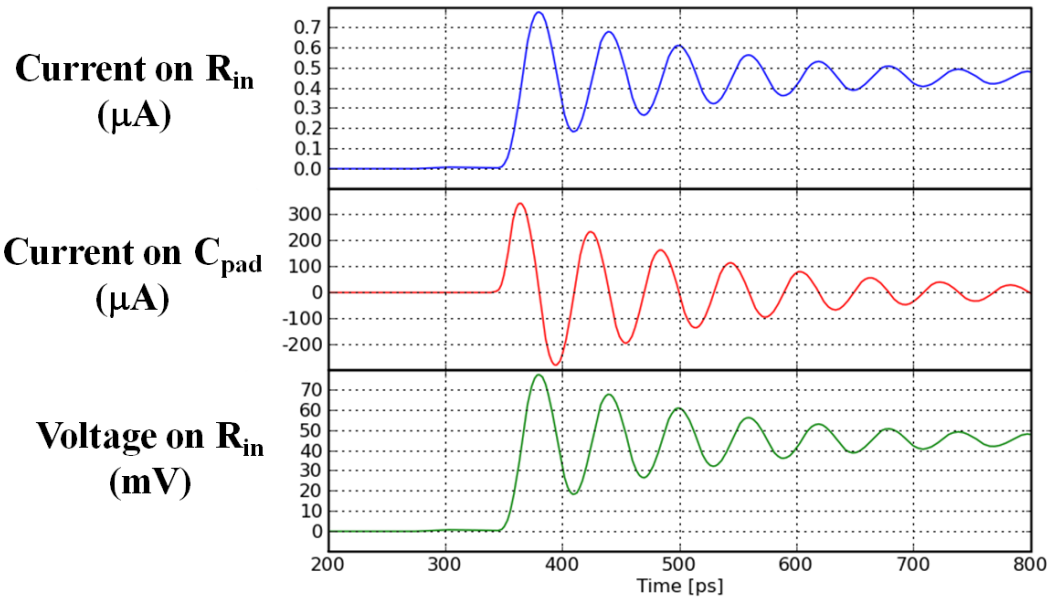
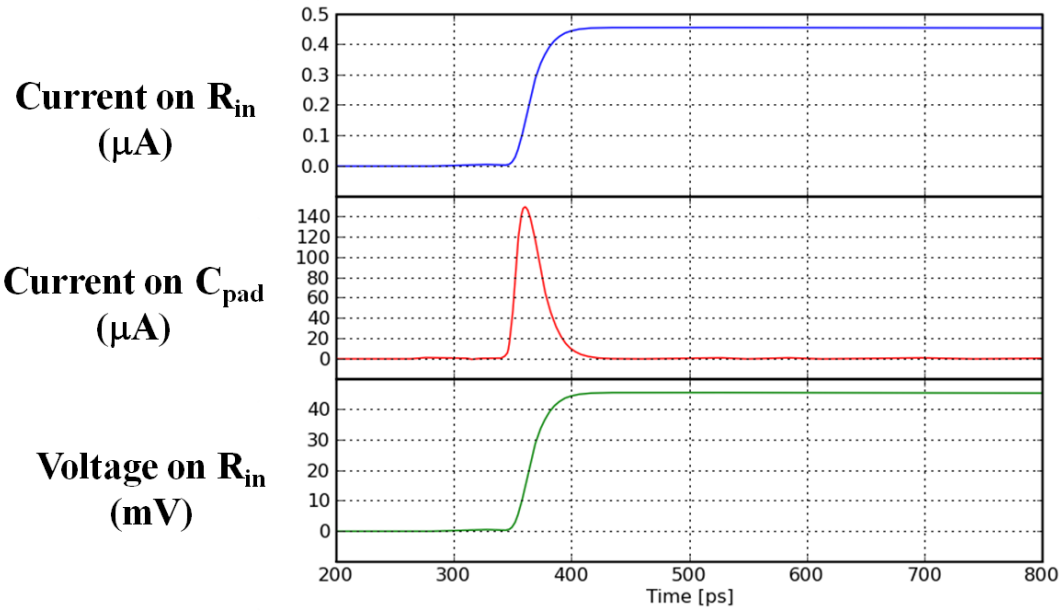


Figure 5-7: Schematic of the output load when consider to be connected to CMOS amplifier.

C_o is the parasitic capacitance of output strip line of JLD, C_{pad} is the parasitic capacitance of output pad on CMOS chip, C_{am} is the input capacitance of CMOS amplifier. L_o is the inductance of output strip line of JLD, the inductance of bonding wire and the inductance of input strip line of CMOS amplifier, respectively. R_o is the output resistance of JLD and R_{in} is the input resistance of CMOS amplifier, it is hundreds kilo-ohm order..



(a)



(b)

Figure 5-8: Output waveform of the JLD using Jsim simulator.
(a)unstable waveform and (b)stable waveform

5.4 Enhancement of Stability in JLD Array

In previous study, although single JLD was well designed and had good characteristic in measurement, they became unstable and had narrow margins when we composed several JLDs into a JLD array. Investigation showed the reason is the effect of the AC bias [56]. Since we used totally 21 channels of JLD to control fully function of the hybrid memory, the supplied AC current became extremely large which flows back through the ground plane, and brings out intensive variations in the magnetic field of the circuits. This caused a serious problem because some SFQ cells, such as DC/SFQ converters, are very sensitive to the magnetic fields. Thus, solution toward reduction of effect of the AC bias is strongly required.

My idea is to separate the ground plane of the JLD array and the SFQ circuit part including DC/SFQ converters. Here I used locally isolated ground (LIG) [57] structure to construct a isolate ground plane for the SFQ circuit and eliminate the effect of AC bias. **Figure 5-9** shows a microphotograph of a 10-channel JLD array with a SPC converter fabricated using the AIST standard Nb process (STP2) with the critical current density of 2.5 kA/cm^2. The black rectangle around the SFQ circuit is LIG structure.

I tested the JLD array at low speed and AC bias margin of each channel was measured. Stable output waveform was obtained and the margin is wide enough (±20%). **Figure**

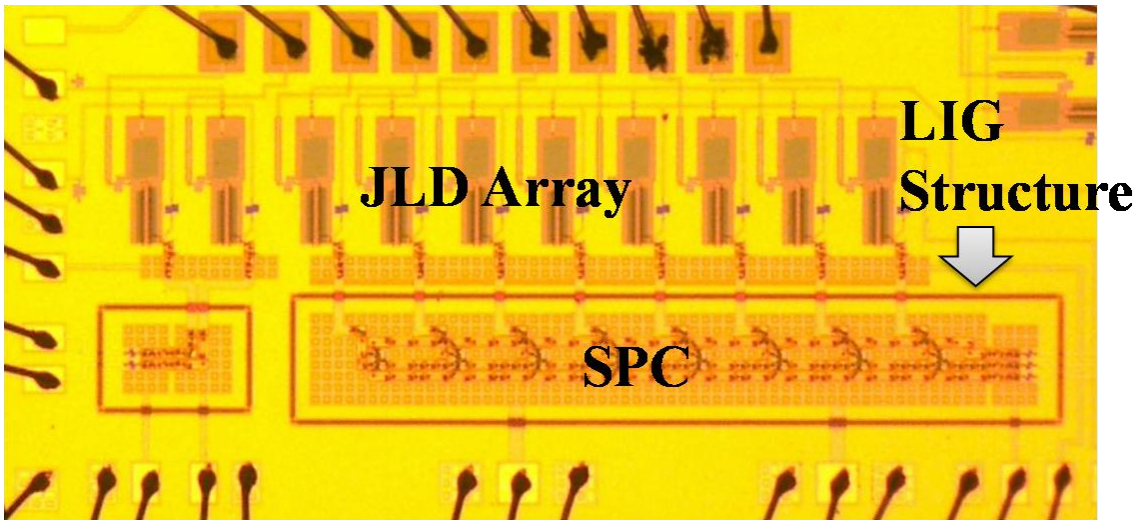


Figure 5-9: Microphotograph of a 10-channel JLD array with a SPC converter. Including 2 write/read lines and 8 data lines, the black rectangle around the SFQ circuit is LIG structure.

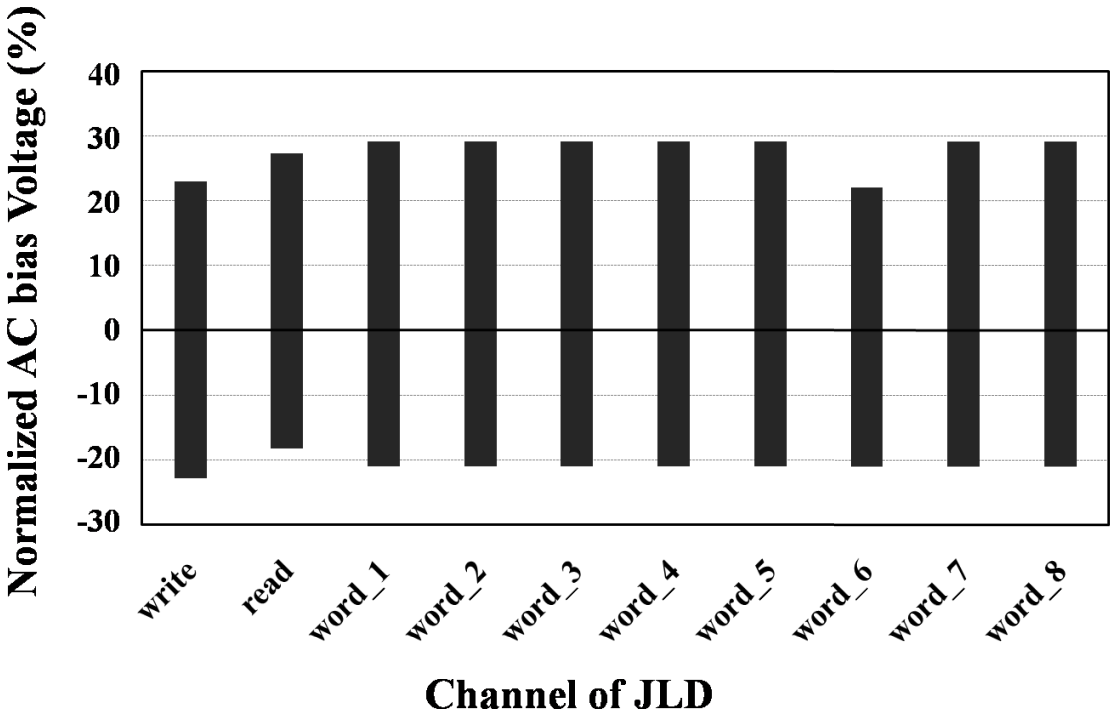


Figure 5-10: The experiment AC bias margin of each channel of JLD.

5-10 shows the experimental AC bias margin of each channel. The results shows that the LIG structure is a useful approach to reduce the effect of AC bias, and this JLD array is a reliable Josephson interface circuit could be adopted in the hybrid memory.

5.5 Implementation and Experiment of Hybrid Memory (Version 1)

We implemented our Josephson/CMOS memory using the JLD array and the CMOS chip. The CMOS chip, including CMOS differential amplifiers and a 64-kb CMOS static RAM was fabricated by the Rohm 180 nm CMOS process. The CMOS chip of the die size of 2.5 mm is placed on the Josephson chip of the die size of 5.0 mm with a piggyback configuration. A photograph of the 64-kb hybrid memory is shown in **Figure 5-11**.

Unfortunately the LIG structure in JLD array occupied too much space in our Josephson chip, I could not bonding all the channels between Josephson chip and CMOS chip because the pad location limitation. Thus, in this experiment, I only tested the data write in function and the data read out function. **Figure 5-12** shows an example of testing waveform of the hybrid memory at 4.2 K. In the figure, raising edges of "Write", "Read" and "Data" signals correspond to the input of an SFQ pulse to the hybrid memory. In this test, the address $(00000000)_2$ is fixed at first, then the data "$(11111111)_2$" is written by enabling the "Write" signal, which corresponds to the "write 1" mode. Output current appears when the "Read" is enabled, which corresponds to the "Read 1" mode. In the next test sequence, the data "$(00000000)_2$" is written into the address $(00000000)_2$ and the stored data "0" is read out, which corresponds to the "Read

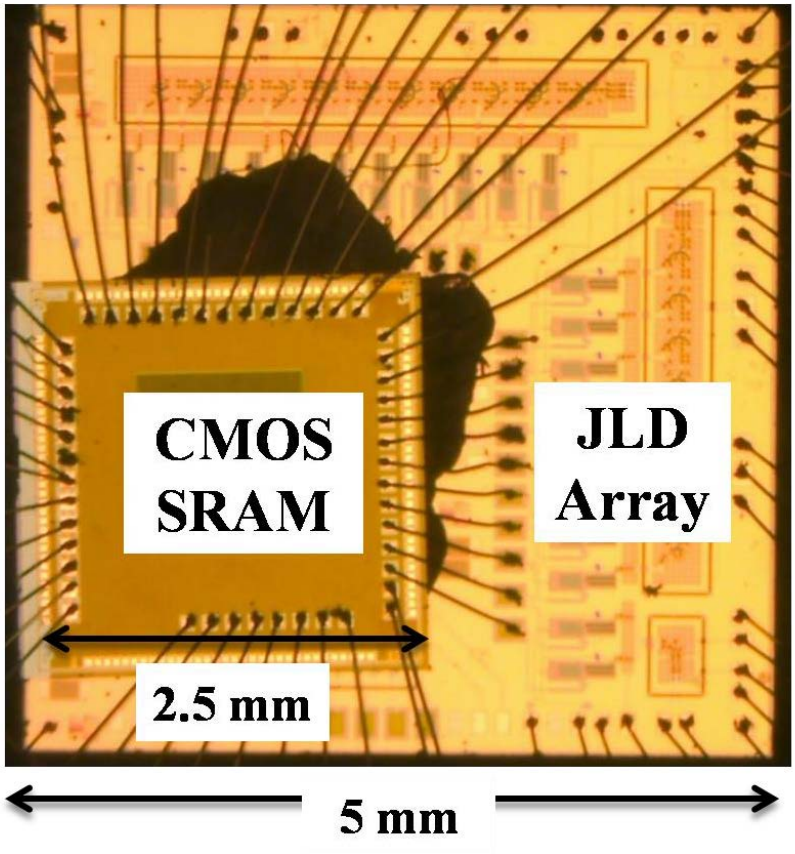


Figure 5-11: Microphotograph of the 64-kb hybrid memory.
2 write/read lines and 8 data lines are bonding. The output is readout directly from the CMOS RAM.

0” mode. These results demonstrate the correct write and read operations of the Josephson CMOS hybrid memory. The address selecting function is also confirmed in another test.

The access time was measured and evaluated to be 1.69 ns, which is somewhat larger than the simulation result of 1.42 ns. This difference is thought to be arising from the long bonding wire in our hybrid memory system, which caused a larger L_w value.

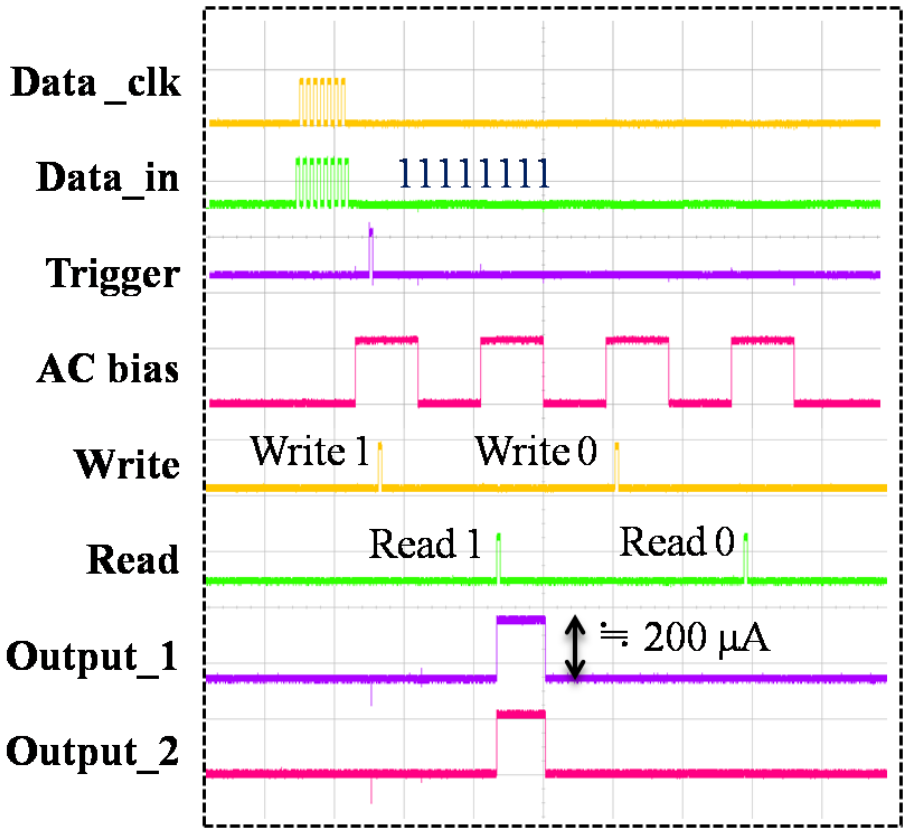


Figure 5-11: An example of testing waveform of the hybrid memory at 4.2 K
8-bit data is input but only 2-bit output is shown since the limitation of oscilloscope channel.

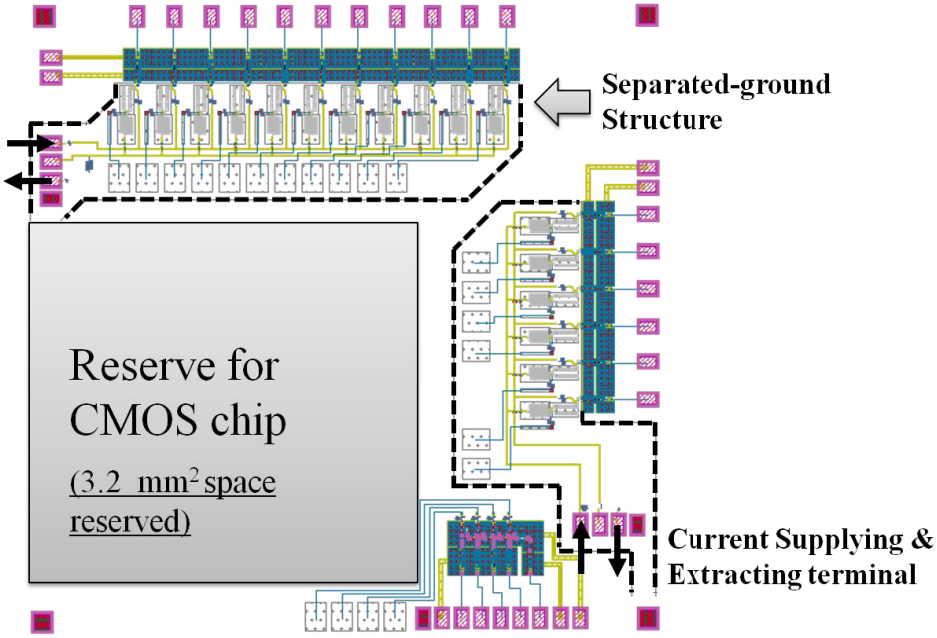
5.6 Reduction of the Area of JLD Array

Although the fully function of the Josephson/hybrid memory has been confirmed, however it is difficult to implement the hybrid memory with all the 21 control lines since the area of JLD array is very large. The next work on the hybrid memory is reducing the area of JLD array, with eliminating the effect of AC bias simultaneously.

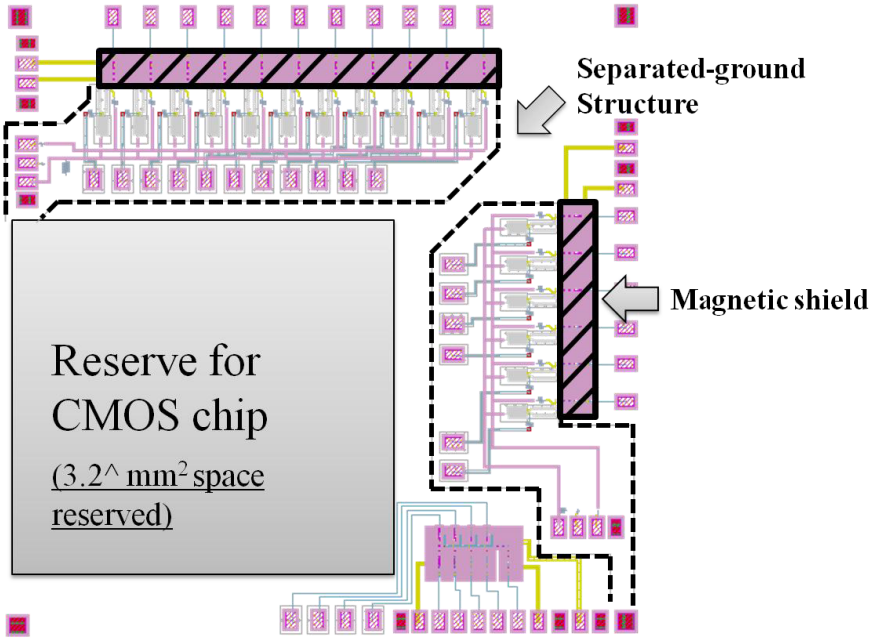
The area increase is majorly caused by the LIG structure. Thus we should find some other method to prevent the effect of AC bias. I proposed two approaches towards this purpose. In the first approach, a separated-ground structure is employed in order to control the ground current flow. However I did not make a completely isolated ground due to the purpose of the area reduction. Differential bias current is supplied to make sure the AC current does not affect the SFQ circuit seriously. Totally 17 channels of JLDs were designed, including all 11 word address lines, 4 word data lines and 2 write/read control lines. The data lines are less than the previous design because the area restriction. The layout is shown in **Figure 5-12(a)**.

In the second approach, the uncompleted separated-ground structure is also employed. Additionally, I covered the SFQ circuit part with "sky plane" (upper superconductive layer) as the magnetic shield to prevent the external magnetic field [58]. Similar to approach 1, there are totally 17 channels of JLDs in this array. The layout is shown in **Figure 5-12(b)**.

Both the two types of JLD array are stable and have very wide bias margin in the measurement. **Figure 5-13 (a)** and **(b)** show the bias margin of each channel of JLD in the array. We can see each channel has an AC bias margin more than ±20%, and there were less scatters between these margins. Both of the two types are suitable for the implementation of hybrid memory, however, the approach 1 requires differential AC bias which is relatively difficult at high frequency. Thus I adopted the magnetic shield type JLD array.



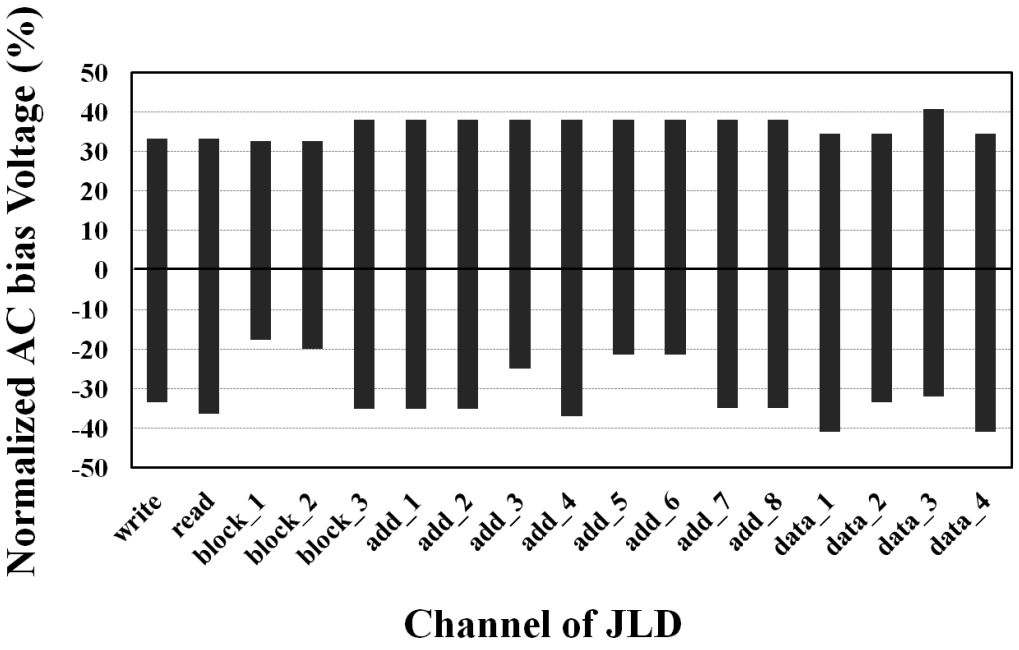
(a)



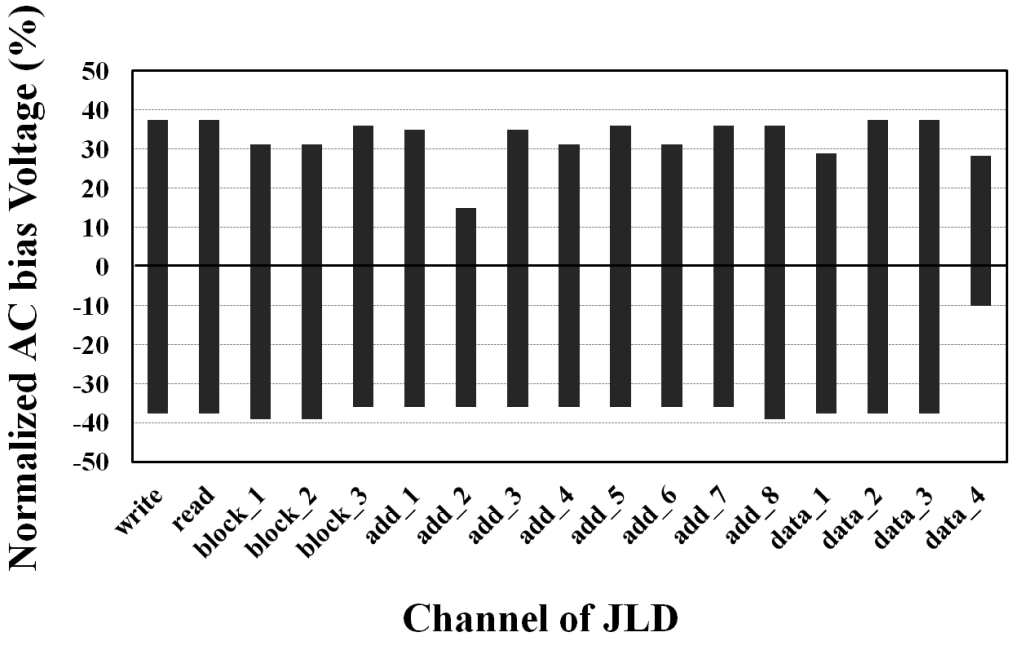
(b)

Figure 5-12: Layout of the two approached on the JLD array

(a) Differential bias type (approach 1) and (b) Magnetic shield type (approach 2)



(a)



(b)

Figure 5-13: AC bias margin of the two types of JLD array

(b) Differential bias type (approach 1) and (b) Magnetic shield type (approach 2)

5.7 Implementation and Experiment of Hybrid Memory (Version 2)

We implemented our 64-kb Josephson/CMOS memory (version 2) using the magnetic shield type JLD array and the CMOS chip. Similar to the previous one, the CMOS chip is placed on the Josephson chip with a piggyback configuration. A microphotograph of the hybrid memory is shown in **Figure 5-14**, and all the 17 interface lines are bonded.

Figure 5-15 shows an example of testing waveform of the hybrid memory at 4.2 K which demonstrates the *data write in*, *data read out* and *address selecting* function. Similar to the previous testing, raising edges of "Write", "Read", "Data" and "Address" signals correspond to the input of an SFQ pulse to the hybrid memory. In this test, the address $(00000000001)_2$ is selected at first, then the data "1" is written by enabling the "Write" signal, which corresponds to the "write 1" mode. Output current appears when the "Read" is enabled, which corresponds to the "Read 1" mode. In the next test sequence, the data "0" is written into the address (00000000001)2 and the stored data "0" is read out, which corresponds to the "Read 0" mode. In the third test sequence, the data "1" is written into the address $(00000000000)_2$, and the data is read out from the address $(00000000001)_2$, in which the data "0" appears.

The write and read operations of the memory for all the memory addresses (2^{11}) were confirmed in this test. It should be mentioned there was a defect that the Josephson current sensors did not work correctly, thus **Figure 5-15** shows output directly from the CMOS RAM (approximately 200 μA current). The reason that caused the malfunction of the current sensors is still considered as the effect of AC bias, since it operated normally in another separated experiment. Fortunately, it is not required that the current sensors and the JLD array have the same ground potential, which allow us to separated their ground plane completely in order to prevent the AC bias flowing to the LDDS.

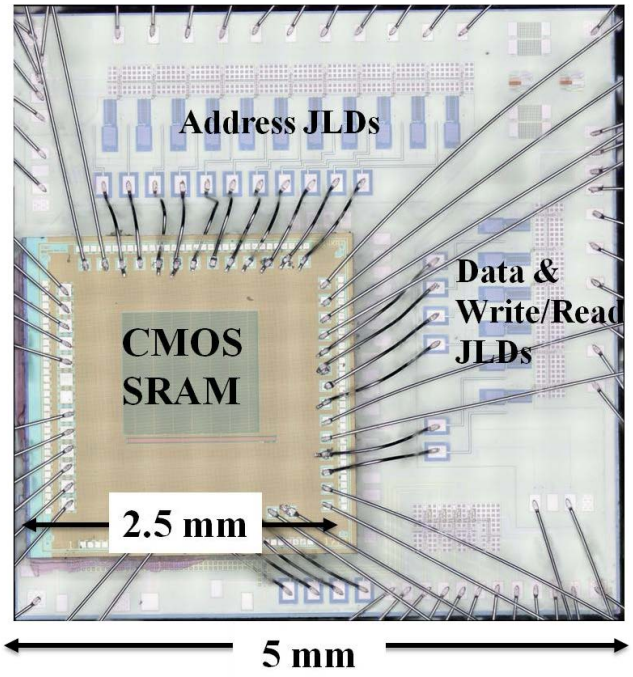


Figure 5-14: Microphotograph of the 64-kb hybrid memory version 2. All the 17-channel JLD was bonded.

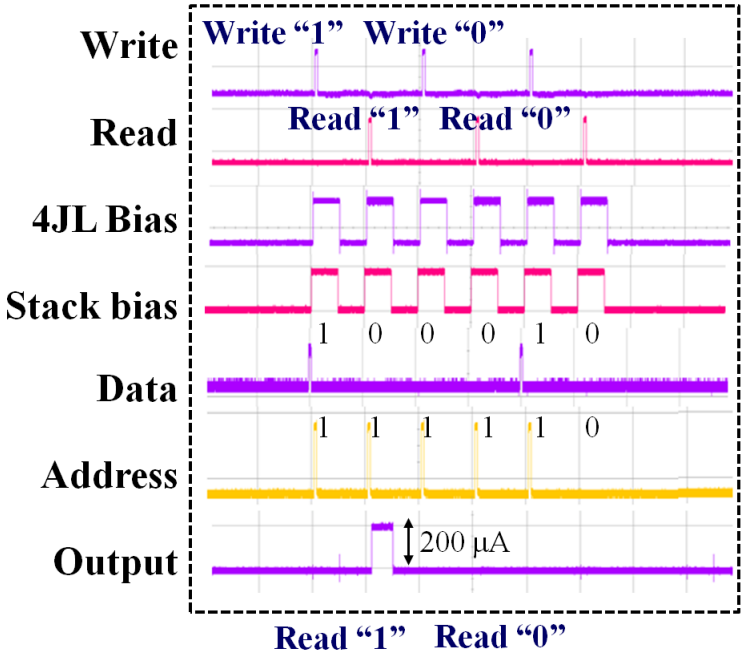


Figure 5-15: An example of testing waveform of the hybrid memory version 2 at 4.2 K.

5.8 Low Power Dissipation Approach of Hybrid Memory

Our estimation shows that in the Josephson/CMOS hybrid memory, most of the power is consumed in the CMOS differential amplifiers and decoders (**Figure 5-16**). In the hybrid memory system, the 8T memory cells provide about 200 μA output current, which is able to be detected by the Josephson current sensor. If we decrease the size of transistor in the memory cell, smaller stray capacitance can be obtained and 40% power reduction in maximum is realized. In our new memory cell design, its output current was decreased to about 120 μA to reduce the power dissipation of the CMOS circuits, therefore higher sensitivity in the Josephson current sensor is required.

The Josephson current sensor is also known as "level-driven DC/SFQ converter (LDDS)". It operates as an NRZ/RZ converter to mediate the logic difference between CMOS and SFQ circuits. **Figure 5-17** shows its schematic and input/output characteristics in simulator JSIM. Considering the input current as a bias source to the SQUID loop, the LDDS can be divided into a JTL and a SQUID detector. Once an SFQ pulse arrive the "clk" terminal, the SQUID biased by input current detect it and provide an SFQ output. To increase the detecting sensitivity of the SQUID, I increased the McComber parameter βc of J_1 and bias Ib_1, decreased the critical current of J_1 and adjusted the loop-inductance L_3 to the proper value. In this study, we optimized those

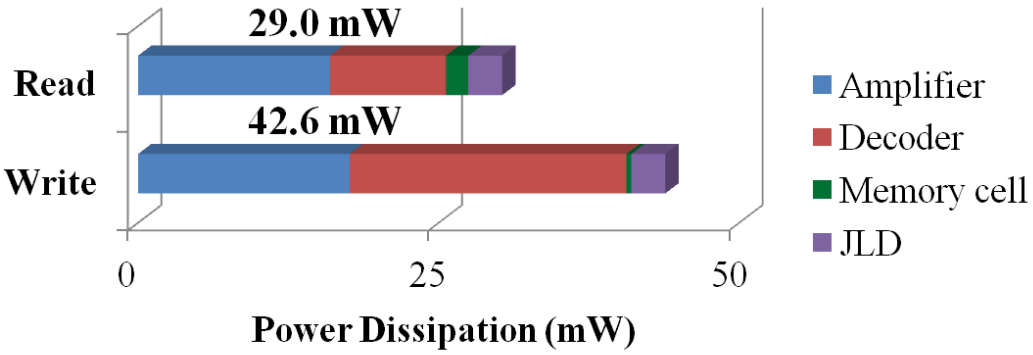


Figure 5-16: Power dissipation of each hybrid memory part.

parameters under the condition of the input current of 120 μA. The bias margin of the modified LDDS is shown in **Figure 5-18**, and the parameters before and after modification are shown in **Table 5-1**. I measured and confirmed the correct operation of the high sensitive LDDS. **Figure 5-19** shows a waveform example of a 4-bit LDDS array used in the hybrid memory. One can see that the output "1" appears corresponding to the high-level of input and the clock arriving. The bias margin is shown in **Figure 5-20**, though it has shifted to the right, it still has a range of about -32% ~ 60% while

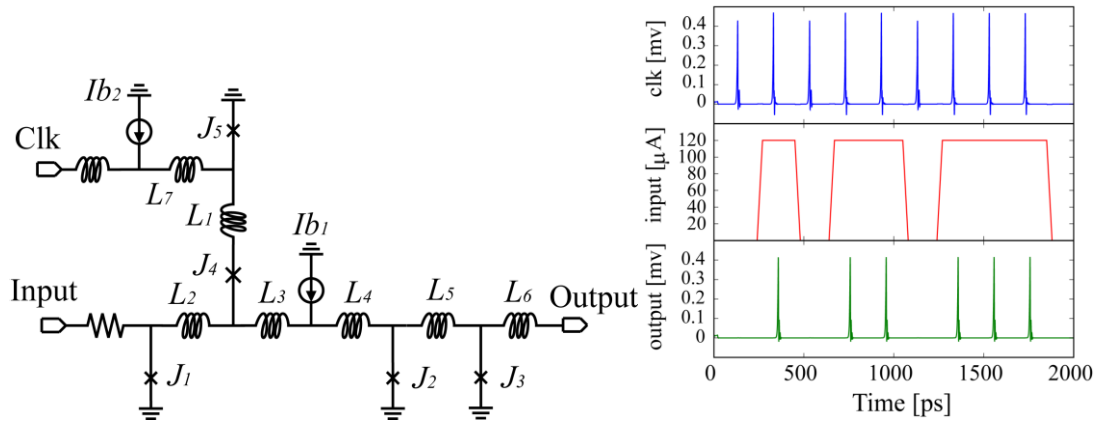


Figure 5-16:Schematic and I/O characteristic of LDDS

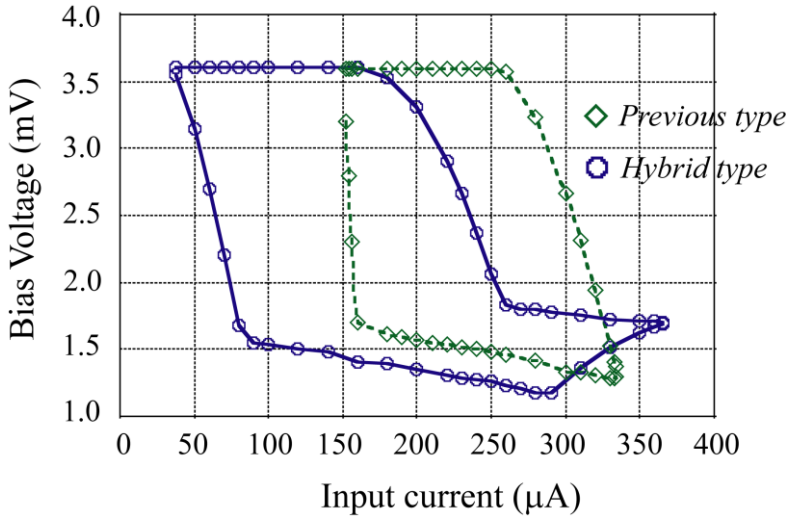


Figure 5-17: DC bias margin of designed (blue line) and previous LDDS (green line)

the input current is 120 μA. This right shifting is considered to be caused by parameter variation in fabrication: the critical current is 20% larger than design.

Table 5-1: The parameters before and after modification of the LDDS

	Before	After
βc of J1	0.89	2
J1	252 μA	184μA
Ib1	116.6 μA	134.4 μA
L3	6.42 pH	6.38 pH

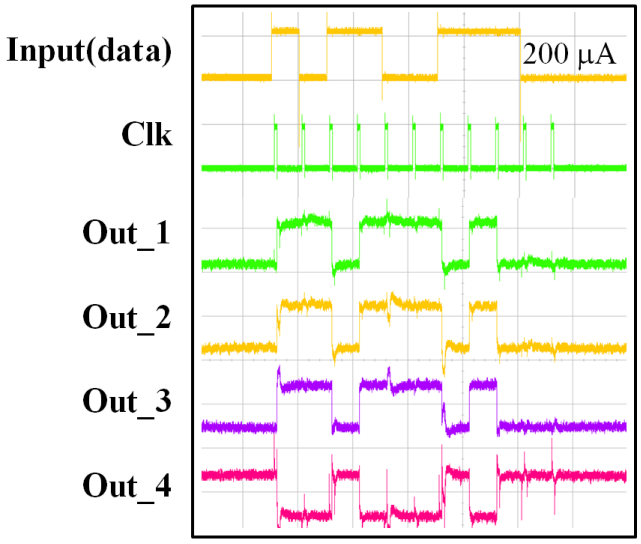


Figure 5-18: Waveform example of a 4-bit LDDS array

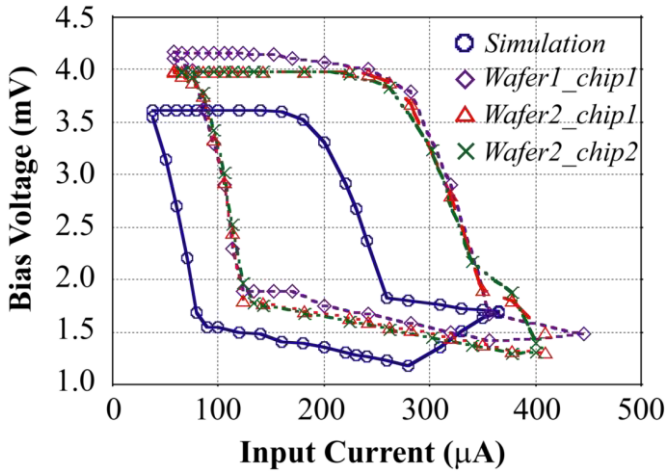


Figure 5-19: Experimental bias margin of the 4-bit LDDS array

5.9 Summary

In this chapter I introduced the design and implementation of a 64-kb Josephson/CMOS hybrid memory using as L2 cache in the superconducting computing system. Stable JLD array was designed and demonstrated, and I successfully confirmed the full-function of the hybrid memory with 2 write/read control lines, 11 address selecting lines and 4 data lines.

I also designed a high sensitive Josephson current sensor for the low power dissipation approach of hybrid memory. 40% power consumption could be reduced by adopting this approach.

6
Summary

In this thesis, the progress and study on implementing a high-end, low power dissipation computing system based on superconducting electronics was introduced. The major technique used in this research is single-flux-quantum (SFQ) circuit, which is considered as a promising beyond-CMOS electronic device, for its ultra-high operating speed (dozens GHz-clock rate) and low power consumption (10^{-3} of CMOS). My research is mainly composed of 2 parts: development of high-end microprocessor, and development of reliable memory.

In the first chapter, the background of my research was briefly presented. Nowadays CMOS based supercomputer is facing 2 large obstacles which prevent it from going to exa-scale. One is the extremely large power dissipation, and another is the development limit of CMOS fabricating process. Superconducting electronics will probably break these walls, as the electronic device of the next generation.

In the second chapter, I introduced the fundamental operating principle of SFQ circuits, from the basic element Josephson junction (JJ) to the logic gate. The power consumption and scaling rule of SFQ device are given in the ending.

In the third chapter, I introduced the methodology we used in designing, fabricating and measuring SFQ logic. For the convenience and high efficiency, cell-based design methodology and top-down approach are adopted. The experiment is done in liquid helium, which provide 4.2 K cryogenic environment.

In the fourth chapter, I introduced the design, implementation and test of SFQ floating-point units (FPUs) used in a novel supercomputing structure, LSRDP. Their high-speed operations were successfully confirmed by on-chip high-speed tests, and the maximum frequency was evaluated to be 72GHz and 59 GHz for half- and single-precision FPM and 58 GHz for single-precision FPA. With these FPUs, we can construct the supercomputing data path with performance of 8 TFLOPs and power efficiency of 3000 GFLOPs/W.

In the fifth chapter, I introduced the design and implementation of a 64-kb Josephson/CMOS hybrid memory used as L2 cache in the superconducting computing system. Stable JLD array was designed and demonstrated, and I successfully confirmed the fully-function of the hybrid memory with 2 write/read control lines, 11 address selecting lines and 4 data lines. I also designed a high sensitive Josephson current sensor for the low power dissipation approach of hybrid memory, and 40% power could be reduced by adopting this approach.

This research shows the possibility that implements a superconducting electronics based super-computing system. The system could have much higher power efficiency than CMOS system over available technique.

To obtain higher computation capability, one can use future Nb fabricating process which has a critical current density of 20 kA/cm^2. This approach could improve the performance by a factor of 1.4 ~1.5 since we can achieve clock rate by a factor of that. Bit-slice structure may be adopted as well, which could also increase the performance by costing larger area.

The static power dissipation in conventional SFQ circuits currently became a critical problem, because it has deteriorated the power efficiency seriously. It is necessary and valuable to apply techniques which could reduce or remove this consumption. This will be the next target in SFQ logic field.

Large capacity, enough reliable memory is still the urgent requirement for SFQ based computing system. My effort eliminated the effect of AC biasing for a 64-kb hybrid memory, however accompany the increasing of capacity, this probably remains as an intrinsic problem. We can solve this problem by using DC biased Josephson amplifier and separating the DC bias part and AC bias part of the circuit completely in different chips. Moreover, pure Josephson RAM, the L1 cache with higher clock frequency than hybrid memory is demanded otherwise the performance of our system will be limited by the memory bandwidth.

Reference

[1] A. W. Burks, and A. R. Burks, "First general-purpose electronic computer," *Annals of the History of Computing,* vol. 3, no. 4, pp. 310-389, 1981.

[2] [Online]http://www.top500.org/.

[3] R. Service, "Computer science. What it'll take to go exascale," *Science (New York, NY),* vol. 335, no. 6067, pp. 394, 2012.

[4] P. Ball, "Computer engineering: Feeling the heat," *Nature,* vol. 492, no. 7428, pp. 174-176, 2012.

[5] A. Geist, "Paving the roadmap to exascale," *SciDAC Review,* vol. 16, 2010.

[6] S. Mukhopadhyay, A. Raychowdhury, and K. Roy, "Switching Energy in CMOS Logic: How far are we from physical limit?," 2006.

[7] K. K. Likharev, and V. K. Semenov, "RSFQ logic/memory family: A new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Trans. Appl. Supercond.,* vol. 1, no. 1, pp. 3-28, 1991.

[8] P. Bunyk, M. Leung, J. Spargo, and M. Dorojevets, "FLUX-1 RSFQ microprocessor: Physical design and test results," *IEEE Trans. Appl. Supercond.,* vol. 13, no. 2, pp. 433-436, 2003.

[9] M. Dorojevets, and P. Bunyk, "Architectural and implementation challenges in designing high-performance RSFQ processors: A FLUX-1 microprocessor and beyond," *IEEE Trans. Appl. Supercond.,* vol. 13, no. 2, pp. 446-449, 2003.

[10] M. Tanaka, T. Kondo, N. Nakajima, T. Kawamoto, Y. Yamanashi, Y. Kamiya, A. Akimoto, A. Fujimaki, H. Hayakawa, and N. Yoshikawa, "Demonstration of a single-flux-quantum microprocessor using passive transmission lines," *IEEE Trans. Appl. Supercond.,* vol. 15, no. 2, pp. 400-404, 2005.

[11] Y. Yamanashi, M. Tanaka, A. Akimoto, H. Park, Y. Kamiya, N. Irie, N. Yoshikawa, A. Fujimaki, H. Terai, and Y. Hashimoto, "Design and Implementation of a Pipelined Bit-Serial SFQ Microprocessor, CORE 1β," *IEEE Trans. Appl. Supercond.,* vol. 17, no. 2, pp. 474-477, 2007.

[12] M. Tanaka, Y. Yamanashi, N. Irie, H. Park, S. Iwasaki, K. Takagi, K. Taketomi, A. Fujimaki, N. Yoshikawa, and H. Terai, "Design and implementation of a pipelined 8 bit-serial single-flux-quantum microprocessor with cache memories," *Supercond. Sci. Tech.,* vol. 20, no. 11, pp. S305, 2007.

[13] A. Fujimaki, M. Tanaka, T. Yamada, Y. Yamanashi, P. Heejoung, and N.

Yoshikawa, “Bit-serial single flux quantum microprocessor CORE,” *IEICE Trans. Electron.,* vol. 91, no. 3, pp. 342-349, 2008.

[14] O. A. Mukhanov, D. Kirichenko, I. V. Vernik, T. V. Filippov, A. Kirichenko, R. Webber, V. Dotsenko, A. Talalaevskii, J. C. Tang, and S. Anubhav, “Superconductor digital-RF receiver systems,” *IEICE Trans. Electron.,* vol. 91, no. 3, pp. 306-317, 2008.

[15] A. M. Kadin, C. A. Mancini, M. J. Feldman, and D. K. Brock, “Can RSFQ logic circuits be scaled to deep submicron junctions?,” *IEEE Trans. Appl. Supercond.,* vol. 11, no. 1, pp. 1050-1055, 2001.

[16] N. Yoshikawa, and Y. Kato, “Reduction of power consumption of RSFQ circuits by inductance-load biasing,” *Supercond. Sci. Tech.,* vol. 12, no. 11, pp. 918, 1999.

[17] Y. Yamanashi, T. Nishigai, and N. Yoshikawa, “Study of LR-loading technique for low-power single flux quantum circuits,” *IEEE Trans. Appl. Supercond.,* vol. 17, no. 2, pp. 150-153, 2007.

[18] D. Kirichenko, S. Sarwana, and A. Kirichenko, “Zero static power dissipation biasing of RSFQ circuits,” *IEEE Trans. Appl. Supercond.,* vol. 21, no. 3, pp. 776-779, 2011.

[19] M. H. Volkmann, A. Sahu, C. J. Fourie, and O. A. Mukhanov, “Implementation of energy efficient single flux quantum digital circuits with sub-aJ/bit operation,” *Supercond. Sci. Tech.,* vol. 26, no. 1, pp. 015002, 2013.

[20] E. S. Fang, and T. Van Duzer, “A Josephson integrated circuit simulator (JSIM) for superconductive electronics application,” *Ext. Abstr. 2nd ISEC, Tokyo, Japan*, pp. 407-410, 1989.

[21] S. Whiteley, “Josephson junctions in SPICE3,” *IEEE Trans. Magn.,* vol. 27, no. 2, pp. 2902-2905, 1991.

[22] N. Yoshikawa, and J. Koshiyama, “A cell-based design approach for RSFQ circuits using a binary decision diagram,” *Supercond. Sci. Tech.,* vol. 12, no. 11, pp. 782, 1999.

[23] N. Yoshikawa, and J. Koshiyama, “Top-down RSFQ logic design based on a binary decision diagram,” *IEEE Trans. Appl. Supercond.,* vol. 11, no. 1, pp. 1098-1101, 2001.

[24] S. Yorozu, Y. Kameda, H. Terai, A. Fujimaki, T. Yamada, and S. Tahara, “A single flux quantum standard logic cell library,” *Physica C,* vol. 378, pp. 1471-1474, 2002.

[25] N. Yoshikawa, S. Mori, and J. Koshiyama, “Timing design of RSFQ logic

circuits using Verilog HDL (In Japanese)," *IEICE Trans. C,* vol. 83, pp. 643-650, 2000.

[26] W. A. Wulf, and S. A. McKee, "Hitting the memory wall: implications of the obvious," *ACM SIGARCH computer architecture news,* vol. 23, no. 1, pp. 20-24, 1995.

[27] A. Kagi, J. R. Goodman, and D. Burger, "Memory bandwidth limitations of future microprocessors," *Proc. 23rd Annual International Symposium on Computer Architecture*, pp. 78-89, 1996.

[28] N. Takagi, K. Murakami, A. Fujimaki, N. Yoshikawa, K. Inoue, and H. Honda, "Proposal of a desk-side supercomputer with reconfigurable data-paths using rapid single-flux-quantum circuits," *IEICE Trans. Electron.,* vol. E91-C, no. 3, pp. 350-355, Mar, 2008.

[29] I. Kataeva, H. Akaike, A. Fujimaki, N. Yoshikawa, N. Takagi, K. Inoue, H. Honda, and K. Murakami, "An operand routing network for an SFQ reconfigurable data-paths processor," *IEEE Trans. Appl. Supercond.,* vol. 19, no. 3, pp. 665-669, 2009.

[30] H. Hara, K. Obata, H. Park, Y. Yamanashi, K. Taketomi, N. Yoshikawa, M. Tanaka, A. Fujimaki, N. Takagi, and K. Takagi, "Design, implementation and on-chip high-speed test of SFQ half-precision floating-point multiplier," *IEEE Trans. Appl. Supercond.,* vol. 19, no. 3, pp. 657-660, 2009.

[31] H. Park, Y. Yamanashi, K. Taketomi, N. Yoshikawa, M. Tanaka, K. Obata, Y. Ito, A. Fujimaki, N. Takagi, and K. Takagi, "Design and implementation and on-chip high-speed test of SFQ half-precision floating-point adders," *IEEE Trans. Appl. Supercond.,* vol. 19, no. 3, pp. 634-639, 2009.

[32] S. Nagasawa, K. Hinode, T. Satoh, H. Akaike, Y. Kitagawa, and M. Hidaka, "Development of advanced Nb process for SFQ circuits," *Physica C,* vol. 412, pp. 1429-1436, Oct, 2004.

[33] D. A. Patterson, and J. L. Hennessy, *Computer organization and design: the hardware/software interface, 4th edition*: San Matteo, CA: Morgan Kaufmann, 2009.

[34] H. T. Kung, "Why Systolic Architectures," *Computer,* vol. 15, no. 1, pp. 37-46, 1982.

[35] K. Obata, M. Tanaka, Y. Tashiro, Y. Kamiya, N. Irie, K. Takagi, N. Takagi, A. Fujimaki, N. Yoshikawa, and H. Terai, "Single-flux-quantum integer multiplier with systolic array structure," *Physica C: Superconductivity,* vol. 445, pp. 1014-1019, 2006.

[36] S. Borkar, R. Cohn, G. Cox, T. Gross, H. T. Kung, M. Lam, M. Levine, B. Moore, W. Moore, C. Peterson, J. Susman, J. Sutton, J. Urbanski, and J. Webb, "Supporting systolic and memory communication in iWarp," *SIGARCH Comput. Archit. News,* vol. 18, no. 2SI, pp. 70-81, 1990.

[37] J. Earle, "Latched carry-save adder," *IBM Technical Disclosure Bull,* vol. 7, pp. 909-910, 1965.

[38] M. Dorojevets, A. K. Kasperek, N. Yoshikawa, and A. Fujimaki, "20-GHz 8 x 8-bit Parallel Carry-Save Pipelined RSFQ Multiplier," *IEEE Trans. Appl. Supercond.,* vol. 23, no. 3, pp. 1300104, 2013.

[39] M. M. Morris, "Digital Logic and Computer Design," Prentice-Hall of India, 1997.

[40] Y. Yamanashi, T. Kainuma, N. Yoshikawa, I. Kataeva, H. Akaike, A. Fujimaki, M. Tanaka, N. Takagi, S. Nagasawa, and M. Hidaka, "100 GHz Demonstrations Based on the Single-Flux-Quantum Cell Library for the 10 kA/cm^2 Nb Multi-Layer Process," *IEICE Trans. Electron.,* vol. 93, no. 4, pp. 440-444, 2010.

[41] T. Kato, "Research of an SFQ floating-point adders using Nb multi-layer process with a high critical current density," Master thesis, Department of electrical and computer engineering, Yokohama national university, 2013.

[42] H. Park, Y. Yamanashi, K. Taketomi, N. Yoshikawa, A. Fujimaki, and N. Takagi, "Novel serial–parallel converter using SFQ logic circuits," *Physica C,* vol. 468, no. 15, pp. 1977-1982, 2008.

[43] M. Tanaka, H. Akaike, A. Fujimaki, Y. Yamanashi, N. Yoshikawa, S. Nagasawa, K. Takagi, and N. Takagi, "100-GHz Single-Flux-Quantum Bit-Serial Adder Based on 10-Niobium Process," *IEEE Trans. Appl. Supercond.,* vol. 21, no. 3, pp. 792-796, 2011.

[44] *Intel® Xeon Phi™ Coprocessor Datasheet*.

[45] S. Nagasawa, Y. Hashimoto, H. Numata, and S. Tahara, "A 380 ps, 9.5 mW Josephson 4-Kbit RAM operated at a high bit yield," *IEEE Trans. Appl. Supercond.,* vol. 5, no. 2, pp. 2447-2452, 1995.

[46] U. Ghoshal, H. Kroger, and T. Van Duzer, "Superconductor-semiconductor memories," *IEEE Trans. Appl. Supercond.,* vol. 3, no. 1, pp. 2315-2318, 1993.

[47] T. Van Duzer, Y. Feng, X. Meng, S. R. Whiteley, and N. Yoshikawa, "Hybrid Josephson-CMOS memory: a solution for the Josephson memory problem," *Supercond. Sci. Tech.,* vol. 15, no. 12, pp. 1669, 2002.

[48] H. Suzuki, A. Inoue, T. Imamura, and S. Hasuo, "A Josephson driver to interface Josephson junctions to semiconductor transistors," in Dig. Tech., Int. Elec. Dev.

Meeting, 1988, pp. 290-293.

[49] H. Jin, K. Kuwabara, Y. Yamanashi, and N. Yoshikaw, “Investigation of robust CMOS amplifiers for Josephson-CMOS hybrid memories,” *Physics Procedia,* vol. 36, pp. 229-234, 2012.

[50] K. Kuwabara, H. Jin, Y. Yamanashi, and N. Yoshikawa, “Design and implementation of 64-kb CMOS static RAMs for Josephson-CMOS hybrid memories,” *IEEE Trans. Appl. Supercond.,* vol. 23, no. 3, pp. 1700704-1700704, 2013.

[51] T. Xue, “The study of the stabilization of Josephson latching drivers,” Master thesis, Department of electrical and computer engineering, Yokohama national university, 2010.

[52] X. Peng, “The research of the stabilization and high-speed operation of Josephson latching driver for hybrid memory,” Master thesis, Department of electrical and computer engineering, Yokohama national university, 2012.

[53] H. Nakagawa, E. Sogawa, S. Kosaka, S. Takada, and H. Hayakawa, “Operating characteristics of Josephson four-junction logic (4JL) gate,” *Jpn. J. Appl. Phys.,* vol. 21, no. 4A, pp. L198, 1982.

[54] D. Ozawa, Y. Yamanashi, and N. Yoshikawa, “Design and Evaluation of Multi-Flux-Quantum Drivers Using Under-Damped Josephson Junctions,” *IEEE Trans. Appl. Supercond.,* vol. 21, no. 3, pp. 835-838, 2011.

[55] T. Ortlepp, L. Zheng, S. Whiteley, and T. Van Duzer, “Design guidelines for Suzuki stacks as reliable high-speed Josephson voltage drivers,” *Supercond. Sci. Tech.,* vol. 26, no. 3, pp. 035007, 2013.

[56] X. Peng, Y. Yamanashi, and N. Yoshikawa, “Design and Measurement of Multi-Channel Josephson Latching Driver used for SFQ/CMOS Hybrid Memory Interface (in Japanese),” in IEICE General Conference, Gifu, 2013, pp. C-8-7.

[57] H. Suzuki, M. Oikawa, K. Nishii, and M. Hidaka, “Investigation of a 5-bit Flash-Type SFQ A/D Converter Using 10 Niobium Process and Locally Isolated Grounds,” *IEEE Trans. Appl. Supercond.,* vol. 23, no. 3, pp. 1400805-1400805, 2013.

[58] Y. Yamanashi, and N. Yoshikawa, “Design and Evaluation of Magnetic Field Tolerant Single Flux Quantum Circuits for Superconductive Sensing Systems,” *IEICE Trans. Electron.,* vol. 97, no. 3, pp. 178-181, 2014.

Accomplishments

Journal Articles

[1] X. Peng, Q. Xu, T. Kato, Y. Yamanashi, N. Yoshikawa, A. Fujimaki, N. Takagi, K. Takagi, and M. Hidaka, "High-speed demonstration of bit-serial floating-point adders and multipliers using single-flux-quantum circuits," *IEEE Trans. Appl. Supercond.,* vol. 25, no. 3, pp. 1-6, 2014.

[2] Q. Xu, X. Peng, T. Ortlepp, Y. Yamanashi and N. Yoshikawa, "Demonstration of bit-serial SFQ based computing for integer iteration hardware algorithm", *IEEE Trans. Appl. Supercond.,* vol. 25, no. 3, pp. 1-4, 2014

[3] Coenrad J. Fourie, X. Peng, R. Numaguchi and N. Yoshikawa, "Inductance and coupling of stacked vias in a multilayer superconductive IC process", *IEEE Trans. Appl. Supercond.,* vol. 25, no. 3, pp. 1-4, 2014

[4] X. Peng, Y. Yamanashi, N. Yoshikawa, A. Fujimaki, N. Takagi, K. Takagi, and M. Hidaka, "Design and High-Speed Demonstration of Single-Flux-Quantum Bit-Serial Floating-Point Multipliers Using a 10kA/cm 2 Nb Process," *IEICE Trans. Electron.,* vol. 97, no. 3, pp. 188-193, 2014.

Conference Paper

[1] X. Peng, Y. Sasaki, H. Jin, K. Kuwabara, Y. Yamanashi and N. Yoshikawa, "Demonstration of Fully Functional 64-kb Josephson/CMOS Hybrid Memory", Ext. Abs., The IEEE 14th International Superconductive Electronics Conference (ISEC2013), pp1-3, Jun., 2013.

International Conference

[1] Q. Xu, X. Peng, Y. Yamanashi, T. Ortlepp and N. Yoshikawa, "High-speed Demonstration of an SFQ-based Computing System for Solving 3n+1 Problem", The 27th International Symposium on Superconductivity (ISS 2014), Tokyo, Japan, Nov. 2014

[2] X. Peng, Q.Xu, Y. Yamanashi, N. Yoshikawa, A. Fujimaki, N. Takagi, K. Takagi, and M. Hidaka, "High-speed demonstration of bit-serial floating-point adders

and multipliers using single-flux-quantum (SFQ) circuits", 2014 Applied Superconductivity Conference (ASC 2014), Charlotte, USA, Aug., 2014 (invited)

[3] Yuta Sasaki, X. Peng, Y. Yamanashi, and N. Yoshikawa, "Improvement of 64-kb Josephson-CMOS hybrid memories toward their complete operation" 2014 Applied Superconductivity Conference (ASC 2014), Charlotte, USA, Aug., 2014

[4] X. Peng, Y. Yamanashi, N. Yoshikawa, "High-speed demonstration of single-precision bit-serial floating-point multipliers using single-flux-quantum circuits", Superconducting SFQ VLSI Workshop (SSV), Nagoya, Japan, Mar. 2014.

[5] X. Peng, Y. Sasaki, Y. Yamanashi, N. Yoshikawa, "Improvement of Interface Circuit for Josephson/CMOS Hybrid Memories toward Ground-Current Reduction and Low Power Dissipation", Superconducting SFQ VLSI Workshop (SSV), Tsukuba, Japan, Nov. 2013.

[6] T. Kato, X. Peng, Y. Yamanashi, N. Yoshikawa, "High-Speed Demonstration of an SFQ Bit-Serial Floating-Point Adder Using 10 kA/cm2 Nb Process", Superconducting SFQ VLSI Workshop (SSV), Tsukuba, Japan, Nov. 2013.

[7] D. Si, N. Takeuchi, K. Inoue, X. Peng, Y. Yamanashi, and N. Yoshikawa, "Yield analysis of Large-scale Adiabatic Quantum Flux Parametron Logic: the Effect of the Deviation of the Inductance Parameter,", Superconducting SFQ VLSI Workshop (SSV), Tsukuba, Japan, Nov. 2013.

[8] X. Peng, H. Suzuki, Y. Yamanashi and N. Yoshikawa, "A Method to Provide Bias Current for Large-Scale SFQ Circuits using Locally Isolated Ground Planes", The 11th European Conference on Applied Superconductivity Conference(EUCAS2013), Genoa, Italy, Sep. 2013

[9] X. Peng, Y. Yamanashi, N. Yoshikawa, A. Fujimaki, N. Takagi, K. Takagi, and M. Hidaka, "Design and High-Speed Demonstration of SFQ Bit-Serial Floating-Point Multipliers Using ISTEC 10 kA/cm2 Nb Process", The 14th International Superconductive Electronics Conference (ISEC2013), A-3, Cambridge, USA, Jun., 2013 (invited)

[10] X. Peng, Y. Sasaki, H. Jin, K. Kuwabara, Y. Yamanashi and N. Yoshikawa, "Demonstration of Fully Functional 64-kb Josephson/CMOS Hybrid Memory", The 14th International Superconductive Electronics Conference (ISEC2013), , H-1, Cambridge, USA, Jun., 2013

[11] X. Peng, Y. Simamura, Y. Yamanashi, N. Yoshikawa, A. Fujimaki, N. Takagi, K. Takagiand S. Nagasawa, "Demonstration of SFQ Half-Precision Floating-Point Multiplier using 10 kA/cm^{2}Nb Process", Superconducting SFQ VLSI Workshop (SSV), Nagoya, Japan, Nov. 2012.

[12] X. Peng, Y. Yamanashi and N. Yoshikawa, "Optimization of the structure and

circuit parameters of Josephson latching drivers for Josephson/CMOS hybrid memories", 2012 Applied Superconductivity Conference (ASC 2012), Portland, USA, Oct., 2012

Domestic Conference

[1] 彭析竹, 徐秋韻, 山梨裕希, 吉川信行, "SFQ 浮動小数点加算器の性能向上に向けた新たなコンポーネント回路の設計", 2014 年電子情報通信学会ソサイエティ大会, 徳島大学, 徳島, 2014 年 9 月

[2] Q. Xu, X. Peng, Y. Yamanashi, N. Yoshikawa, and T. Ortlepp, "Improvement of Computational Efficiency Collatz Conjecture Processors" 2014 年電子情報通信学会ソサイエティ大会, 徳島大学, 2014 年 9 月

[3] 佐々木悠太, 彭析竹, 山梨裕希, 吉川信行, "完全動作実証に向けた 64-kb Josephson-CMOS ハイブリッドメモリの改良" 電子情報通信学会超伝導エレクトロニクス研究会, 東京, 2014 年 7 月.

[4] 彭析竹, 山梨裕希, 吉川信行, "10 kA/cm^2 Nb プロセスを用いた単一磁束量子単精度浮動小数点乗算器の高速動作実証", 2014 年度春季低温工学・超電導学会, タワーホール船堀, 東京, 2014 年 5 月

[5] 彭析竹, 佐々木悠太, 山梨裕希, 吉川信行, "SFQ/CMOS Hybrid Memory の完全動作に向けた研究", 2014 年電子情報通信学会総合大会, 新潟大学, 新潟, 2014 年 3 月

[6] 加藤泰一, 彭析竹, 山梨裕希, 吉川信行, "単一磁束量子回路を用いた単精度浮動小数点加算器の設計と測定" 2013 年度秋季低温工学・超電導学会, ウィンクあいち, 愛知, 2013 年 12 月.

[7] 佐々木悠太, 彭析竹, 西村考正, 山梨裕希, 吉川信行, "64-kb SFQ/CMOS ハイブリッドメモリの測定と評価" 2013 年電子情報通信学会ソサイエティ大会, 福岡工業大学, 福岡, 2013 年 9 月.

[8] 彭析竹, 山梨裕希, 吉川信行, "SFQ/CMOS Hybrid memory インタフェース用多チャンネル Josephson Latching Driver の動作検証",2014 年電子情報通信学会総合大会, 岐阜大学, 岐阜, 2013 年 3 月.

[9] 彭析竹, 島村泰浩, 山梨裕希, 吉川信行, 藤巻朗, 高木一義, 高木直史, 永沢秀一, "ISTEC 10 kA/cm2 Nb プロセスを用いた単一磁束量子半精度浮動小数点乗算器の動作実証",2012 年電子情報通信学会ソサイエティ大会, 富山大学, 富山, 2012 年 9 月.

[10] 彭析竹, 山梨裕希, 吉川信行, “Josephson latching driver における接合バラツキの影響と入力部の検討”, 第 59 回応用物理学関係連合講演会, 早稲田大学, 東京, 2012 年 3 月.