Design and implementation of stochastic neurosystem using SFQ logic circuits

Kondo Taizo, Kobori Masayuki, Onomi Takeshi, Nakajima Koji

Journal or Publication Title: IEEE Transactions on Applied Superconductivity
Volume: 15
Number: 2
Page Range: 320-323
Year: 2005
URL: http://hdl.handle.net/10097/48237
doi: 10.1109/TASC.2005.849818
Design and Implementation of Stochastic Neurosystem Using SFQ Logic Circuits

Taizo Kondo, Masayuki Kobori, Takeshi Onomi, and Koji Nakajima

Abstract—We propose a stochastic neurosystem using SFQ logic circuits and design the main components with the following functions: carrying out the multiplication of an input to a neuron on a synaptic weight value, integrating pulses to generate a membrane potential, and generating the output of a neuron. We simulate some circuits by JSIM and confirm their correct operation. We compare two methods of multipliers: using a comparator and using a divider. The multiplication using the divider is effective with respect to integration, and reduces the accumulation time $N_\alpha$ required for higher precision operations. We designed a 4-bit up/down counter assuming the NEC 2.5 kA/cm$^2$ Nb/AlO$_X$/Nb standard process. We show that it is possible to compose the activation function circuit using a comparator.

Index Terms—Neural network, single flux quantum, stochastic logic, up/down counter.

I. INTRODUCTION

APPLICATIONS of a neural network are expected in fields such as the pattern recognition and combinational optimization problems, tasks that conventional computers are not good at. A simple neural network is realizable in software simulation. However, it is not realistic with respect to calculation time and network scale when a neural network is used for applications in real-time processing. Although the number of neurons for the practical application is not obvious, the more the number of neurons increases, the higher the information processing a neural network can carry out. Therefore, it is necessary to integrate a neuron by implementing it as hardware.

A neural network using stochastic logic, which has been proposed by Kondo et al. [1], is suitable for this integration. The stochastic neurochips, including dozens of neurons, have been designed and fabricated using CMOS technology [2]–[4]. In stochastic logic, one can convert an analog value to occurrence probability of a pulse and carry out operations using the probability. A comparator is used for the conversion. The pulse sequence is generated when a value $X$ is more than a random number $R$ (Fig. 1(a)). The relationship between the probability of the pulse sequence generated and $X$ is shown in Fig. 1(b). The pulse sequence has noise, which originates in the coding (coding noise). In some kinds of problems, the performance of the stochastic neural network is improved by the coding noise as compared to the deterministic neural network [1]. Encoding is performed by accumulating the pulse sequence for a certain period of time. This logic has merit in that complex operations are realizable with basic digital logic gates. For example, the multiplication of the pulse sequences is carried out with a single AND gate. Nemoto et al. have reported that the number of transistors for a neuron in stochastic logic is 1/5 of that in deterministic logic [2]. We require $n^2$ synapses in the full-connection neural network including $n$ neurons. The synaptic circuit in stochastic logic requires 1300 Josephson junctions (JJs) at a rough estimate. This number is about half of that in deterministic logic. However, it has demerit as operation becomes slow, because it is necessary to increase accumulation time for more precise operation.

Hence, we propose a stochastic neural network using single-flux-quantum (SFQ) logic circuits [5], [6]. SFQ logic circuits use a single flux quantum as an information carrier and are attractive because of their high speed operation and low power dissipation. Therefore, SFQ logic circuits are the candidates for the main data processor in the next generation. We expect to cancel the demerit of slow operation in stochastic logic with SFQ logic circuits, and SFQ logic circuits have good characteristics for stochastic logic because both are pulse logic.

In this paper, we report the design of necessary circuits in order to compose a stochastic neuron using SFQ logic circuits. We designed the circuit generating membrane potentials assuming the NEC 2.5 kA/cm$^2$ Nb/AlO$_X$/Nb standard process. We investigate the difference between the performance and consumption area of a multiplier circuit using a comparator and that using a divider.

The numerical simulation is carried out by JSIM [7] and SCOPE [8] assuming the NEC 2.5 kA/cm$^2$ Nb/AlO$_X$/Nb standard process.

II. STOCHASTIC NEURON

We design a stochastic neuron on the basis of the system that has already been proposed [1]–[4]. Fig. 2(a) shows the model of a neuron. The discrete-time dynamics of a neural network are given by

$$u_i(t + 1) = \sum_{j=1}^{N} w_{ij} x_j(t), \quad (1)$$

$$x_i(t) = f(u_i(t)), \quad (2)$$
where $w_{ij}$ is the synaptic weight from the $j$th neuron to the $i$th neuron, $x_j$ is the membrane potential, $u_i$ is the output pulse generated in a neuron, $f(u_i)$ is an activation function, and $N$ is the number of neurons. The activation function is a sigmoid function, or a sign function, etc. (Fig. 2(b) and (c)). In order to realize a neuron which carries out such operations, we require the following circuits: one carrying out the multiplying of the output of a neuron by a synaptic weight value, one accumulating a pulse to generate membrane potential, and one generating the output of a neuron.

### A. Multiplier

We require a circuit for multiplying the synaptic weight $w_{ij}$ on the output $x_{j,pulse}$ of a neuron. Two methods are able to realize this. One method is a multiplier using comparators; the other is one using dividers.

1) Multiplier Using Comparator: Fig. 3 shows multiplication using a comparator. The comparator outputs a pulse when $w_{ij}$ is larger than a random number. In other words, the comparator converts $w_{ij}$ to a pulse sequence generated in a probability proportional to the value. Multiplication is realized by implementing the logical AND operation between the pulse sequence and the output $x_{j,pulse}$ from another neuron. Therefore, the occurrence probability of the output $w_{ij,pulse}x_{j,pulse}$ is given as follows:

$$P(w_{ij,pulse}x_{j,pulse}) = P(w_{ij,pulse}) \cdot P(x_{j,pulse}) = \frac{w_{ij}}{w_{ij,MAX}} P(x_{j,pulse}),$$

where $w_{ij,MAX}$ is the maximum value of the synaptic weight and the random number. This circuit requires a comparator and a random number generator.

a) Random number generator: We use M-code as a random number even though it is only pseudo random number because it can be composed of simple circuits. Some M-code generators were designed and fabricated using SFQ logic circuits [9], [10]. We design a 4-bit M-code generator using XOR, Inverter, and 2 DFFs (D Flip-Flops). This circuit has the feature that no start signal is required and the operation margin is securable (Fig. 4). According to the simulations by JSIM, bias margins are ranged from $-32.5\%$ to $+35.6\%$ at $10.0$ GHz and the maximum operating speed is $18.9$ GHz.

b) Comparator: A multi-bit pipelined comparator can be assembled using 1-bit comparators (Fig. 5(a)) and 4-in-2out comparators (Fig. 5(b)). When a 1-bit comparator compares $a_1$ with $b_1$, it performs the following operations:
- If $a_1$ is more than $b_1$, the output pulse $X$ is generated.
- If $a_1$ is equal to $b_1$, the output pulse $Y$ is generated.
- If $a_1$ is less than $b_1$, no output pulse is generated.

When $A(=a_2,a_1)$ is compared with $B(=b_2,b_1)$, each bit is compared using a 1-bit comparator. Then, by using the outputs of two 1-bit comparators, a 4-in-2out comparator carries out the following operations:
- If $A$ is more than $B$, the output pulse $X$ is generated.
- If $A$ is equal to $B$, the output pulse $Y$ is generated.
- If $A$ is less than $B$, no output pulse is generated.

Fig. 6 shows a 4-bit pipelined comparator. The correct operation is verified by simulation, JSIM. The simulated upper and lower margins are $+25.0\%$ and $-22.2\%$ at $5.0$ GHz, respectively.

2) Multiplier Using Divider: The multiplier (Fig. 7) is also realizable by using dividing circuits including T Flip-Flops and NDROs (Non Destructive Read Out) [6]. The input $x_{j,pulse}$ is divided into 1/2, 1/4, . . . , by TFFs. We can change the ratio of the input $x_{j,pulse}$ and the output $w_{ij,pulse}x_{j,pulse}$ by shifting the internal states of NDROs. When this circuit is used...
as \( n \)-bit multiplier, the occurrence probability of the output \( w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}} \) is obtained as follows:

\[
P(w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}}) = \left( \frac{1}{2} w_{ij,n} + \frac{1}{4} w_{ij,n-1} + \cdots \right) P(x_{j \text{-pulse}}),
\]

where \( w_{ij,n} \) is \( n \)-th bit of \( w_{ij} \). This circuit is a sequential circuit, and has \( 2^n \) internal states.

3) Comparison of Two Circuits: The comparison of estimated number of JJs is shown in Fig. 8(a). The consumption area of a circuit is proportional to the number of JJs. Thus the multiplier using dividers has an advantage for integration over that using comparators. However, the multiplier using comparators has a constant throughput regardless of the bit length since it is a pipelined circuit.

Fig. 8(b) shows the comparison of average root-mean-square (RMS) errors of the outputs \( w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}} \) of both methods.

The average RMS error is obtained by numerical simulation on the following conditions. First, the error is defined as the difference between deterministic outputs. Second, each trial is carried out 10,000 times in each accumulation time \( N_\alpha \). Third, \( w_{ij} \) and \( x_{j \text{-pulse}} \) are given at random in each trial. The output \( w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}} \) is more precise with increasing \( N_\alpha \). Infinite \( N_\alpha \) means deterministic. Therefore, the average RMS error is almost inversely proportional to \( N_\alpha \) in both cases. The multiplier using the comparator has two coding noises, one from the input \( x_{j \text{-pulse}} \), and one from the pulse sequence \( w_{ij \text{-pulse}} \).

The average RMS error of the multiplier using the divider is less than that using the comparator. 8 bits in general is required as the average RMS error of the multiplier using the divider is less than that using the comparator. In contrast, the multiplier using the divider has coding noise only in \( x_{j \text{-pulse}} \). Since the coding noise influences the output a little, the average RMS error of the multiplier using the divider is less than that using the comparator. 8 bits in general is required as operation accuracy for dealing with practical applications. In the case of the multiplier using the comparator, \( N_\alpha \) is few tens thousands; in case of that using the divider, \( N_\alpha \) is approximately half. Therefore, the performance of a neural network using dividers may improve with respect to the operation speed because the required \( N_\alpha \) is reduced for higher precision operation.

B. Up/Down Counter

The membrane potential \( u_{ij} \) is generated by accumulation of the pulse sequence \( w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}} \) from other neurons. An up/down counter, which satisfies this function, is designed using an adder cell [11], [12]. An adder cell is a serial input adder (Fig. 9) achieved by improving an ICF gate, which is the fundamental device of the Phase-Mode logic proposed by Nakajima et al. [5]. The up/down counter is easily assembled by connecting adder cells serially (Fig. 10(a)). This circuit has two inputs (Up, Down), several outputs (\( O_1, O_2, \ldots, O_h \)), and a reset input (Re).

Incrementing the value of the up/down counter occurs by sending a signal to the Up input. The signal to the Up input propagates only to the input of the adder cell which is the least significant bit (LSB) of the up/down counter. Thus, the value of the up/down counter is set to \(+1\).

On the other hand, decrementing the internal state of the up/down counter occurs by sending a signal to the Down input. The signal to the Down input propagates to the inputs of every adder cell. In other words, the up/down counter receives “111, . . . , 1” input. “111, . . . , 1” is expressed as \(-1\) by two’s complement. Hence, the value of up/down counter is set to \(-1\).

The value of the up/down counter can be read out by sending a signal to the Re of every adder cell. The outputs of the multiplier \( w_{ij \text{-pulse} \cdot x_{j \text{-pulse}}} \) are brought to Up or Down according to
The bit length of the comparator must correspond to the maximum value of the membrane potential. Here, the bit length of a random number is shorter than that of the membrane potential and sets up the threshold value. Fig. 11(b) shows the activation function, which is a monotonic increasing function. It is obtained when the bit length of the membrane potential and the random number are 9 bits and 7 bits, respectively. The noise decreases, and the stochastic output $x_{j,\text{pulse}}$ is close to the deterministic with increasing $N_a$.

### III. Conclusion

We propose the stochastic neuron using SFQ logic circuits and design the main components required for that. We simulate some circuits by JSIM and confirm their correct operation. We compare the comparator multiplier with the divider one. The divider multiplier has an advantage with respect to integration, and reduces the accumulation time $N_a$ required for higher precision operations. We designed the 4-bit up/down counter assuming the NEC 2.5 kA/cm$^2$ Nb/AlO$_x$/Nb standard process. The activation function circuit can be assembled by using a comparator.

### ACKNOWLEDGMENT

The authors gratefully acknowledge the Superconductive Circuits Fabrication Team of SRL-ISTEC and NEC Corp. for the fabrication of Nb chips.

### REFERENCES


