Several Arithmetic Operations on Spiking Neural P Systems

Xianwu Peng, Jianxun Liu, Wei Liang
Key Laboratory of Knowledge Processing and Networked Manufacturing, Hunan University of Science and Technology.
Xiangtan Hunan, China
e-mail:xwpeng@hnust.edu.cn

***Abstract*—In this paper,we consider some simple SN P systems that perform the following arithmetic operations:two's complement,addition/subtraction on negative integers and multiplication on two arbitrary positive integers.This paper provides an answer to an open problem about multiplication operation formulated by Miguel A. Gutiérréz-Naranjo.**

Keywords- Spiking neural P systems; Arithmetic operation; Two's Complement; Addition; Multiplication

I. INTRODUCTION

Spiking neural P systems [1] (SN P systems,for short) are distributed and parallel computing models that are a synergy inspired by P systems and spiking neural networks. It has been shown that these systems are computationally complete. In short,an SN P system consists of a set of neurons placed in the nodes of a directed graph and sending spikes along the arcs of the graph (called synapses). The system evolves according to a set of spiking rules and forgetting rules each of which is associated with a neuron that uses the rules for sending or internally consuming spikes. In a standard SN P system there is only one type of objects called spikes which are denoted in what follows by the symbol a.

Arithmetic operations are regarded as basic operations of other complex processes. Recent results show that arithmetic operations can be supported by SN P System [2][5] and SN P system with anti-spikes(a variant of SN P System) [3][4]. Literature [2] firstly presents some SN P systems that perform the following operations on positive integers: addition, subtraction, comparison and multiplication by a fixed factor. All the numbers given as inputs to these systems are expressed in binary form, encoded as a spike train in which at each time instant a spike denotes 1, and no spike denotes 0. The outputs of the computations are also expelled to the environment in the same form. But in these systems,the negative integers are not considered.The another open problem about multiplication operation is how to design an SN P system for the multiplication,where both the numbers to be multiplied are supplied as inputs. SN P system with anti-spikes[4] can encode the binary digits using two types of objects called anti-spikes and spikes and is used to perform arithmetic operations like two's complement, addition and subtraction on binary numbers in [3].

In this paper, we consider SN P system as components of a restricted Arithmetic Logic Unit and perform some arithmetic operations with signed numbers. The presented systems implement some basic arithmetic operations, namely two's complement, addition/subtraction on negative integers (signed integers) and multiplication on two arbitrary positive integers in which we provide an answer to the open problem formulated by Miguel A. Gutiérréz-Naranjo in [2].

II. SPIKING NEURAL P SYSTEM

Formally, an SN P system of degree $m \geq 1$, is a construct $\prod = (O, \sigma_1, \sigma_2, ..., \sigma_m, syn, in, out)$,where:

1) $o = \{a\}$ is singleton alphabet (a is called spike);

2) $\sigma_1, \sigma_2, ..., \sigma_m$ are neurons, of the form $\sigma_i = (n_i, R_i), 1 \leq i \leq m$, where: $n_i \geq 0$ is the initial number of spikes contained by the neuron σ_i; R_i is a finite set of rules of the following two forms:

①firing rules: $E / a^c \rightarrow a; d, c \geq 1, d \geq 0$,where E is a regular expression over O, d is the delay time,that is, the interval between using the rule and releasing the spike. If $d = 0$, then d can be omitted from the firing rules.

If the neuron σ_i contains k spikes, $a^k \in L(E)$ (this means that the regular expression E "covers" exactly the contents of the neuron) and $k \geq c$, then the firing rule $E / a^c \rightarrow a; d$ can be applied in this neuron, and this means that c spikes are consumed (thus only $k - c$ spikes remain in neuron σ_i and a spike produced by neuron σ_i are replicated and they send to all neurons σ_j such that $(i, j) \in syn$ (each σ_j receives a spike) a spike after d time units. If $d = 0$,then the spikes are emitted immediately, if $d = 1$, then the spikes are emitted in the next step, and so on. In the case $d \geq 1$, if the rule is used in step t ,then in steps $t, t+1, t+2, ...t+d-1$ the neuron is closed, and it cannot receive new spikes. In step $t+d$,the neuron spikes and becomes again open, hence can receive spikes from other neurons (which can be used in step $t+d+1$). If a spiking rule is of the form $E / a^r \rightarrow a; t$, with $L(E) = \{a^r\}$, then we will write this rule in the simpler form $a^r \rightarrow a; t$.

②forgetting rules: $a^s \rightarrow \lambda$, for some $s \geq 1$, with the restriction that $a^s \notin L(E)$ for any firing rule $E / a^c \rightarrow a; d$ in neuron σ_i (i.e., a firing rule and a forgetting rule cannot be enabled simultaneously in a neuron).If neuron σ_i contains exactly s spikes, then the forget ting rule $a^s \rightarrow \lambda$ can be applied (no firing rule can be enabled in this step). Applying the forgetting rule as above means s spikes are consumed, but no new spike is produced.

3) $syn \subseteq \{1,2,...,m\} \times \{1,2,...,m\}$ with $(i,i) \notin syn$ for $1 \le i \le m$ (synapses among cells);

4) $in, out \in \{1,2,...,m\}$ indicates that the input and output neurons of $\prod$.

In each time unit, if a neuron σ_i can use one of its rules, then a rule from R_i must be used. Since two firing rules, $E_1 / a^r \to a$ and $E_1 / a^k \to a$ with $L(E_1) \cap L(E_2) \neq \varphi$, it is possible that two or more rules can be applied in a neuron. In such a case, only one of them is chosen nondeterministically.Thus, the rules are used in the sequential manner in each neuron, but neurons function in parallel with each other.

Using the rules, we pass from one configuration of the system to another configuration, such a step is called a transition. For two configurations C and C' of $\prod$ we denote by $C \Rightarrow C'$, if there is a direct transition from C to C' in $\prod$.

A computation of $\prod$ is a finite or infinite sequences of transitions starting from the initial configuration, and every configuration appearing in such a sequence is called reachable. Note that the transition of C is non-deterministic in the sense that there may be different set of rules applicable to C. A computation halts if it reaches a configuration where no rule can be used.

III. TWO'S COMPLEMENT

In order to perform arithmetic operations with signed numbers on SN P systems,the first work is encoding to the operands as appropriate sequences of spikes.In this paper, the numbers given as inputs to SN P systems are expressed in binary form.The positive integers are encoded as a spike train in which at each time instant the presence of a spike denotes 1, and the absence of a spike denotes 0 and the negative integers are encoded the complemental code of the signed binary numbers.

The two's complement is used to represent a negative of a binary number. It also gives us a way to add and subtract positive and negative binary numbers. A simple way to find the two's complement of a number is to start from the least significant bit keeping every 0 as it is until you reach the first 1 and then complement all the rest of the bits after the first 1.The SN P system that performs m-bits two's complement is depicted by Figure 1.

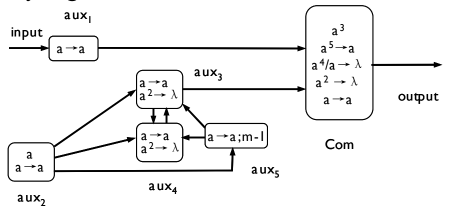

Figure 1. SN P system computing m-bits two's complement

Neuron aux_1 is the input neuron. Neuron Com is the output neuron, which sends output to the environment. A Speacal function moudule including neuron aux_2,aux_3,aux_4 and aux_5 can provide m sequential spikes to Neuron Com from the 2nd step to the (m+1)th step. Simultaneously,the m-bits true code is inputted into the input neuron bit by bit from the 1st step to the (m)th step and the neuron Com can receive the first bit at the (m+1)th step.In the function moudule,neuron aux_5 acts as a counter which will fire at the 1st step but send a spike at (m)th step by the delaying rule $a \to a; m-1$,so the function moudule will halt after the (m+1)th step. The output neuron Com initially has 3 spikes and as long as it receives a spike(actual input to the input neuron is 0 and the spike comes from neuron aux_3) after the (m+1)th step, it uses the rule $a^4 / a \to \lambda$ and no spike send to the environment, which is same as the input. But if the neuron Com receives 2 spikes(that means we got the first 1), it will use the first rule $a^5 \to a$ and sends a spike to the environment(that is first 1 is unchanged). After firing the rule, the neuron Com has no spike and then simply complements the input it receives by using the third and fourth rule and sends it to the environment. That means after the first 1 code, the output will be the complement of input. We can easily observe that the system correctly calculates the two's complement and emits its first output bit at t=3 as there is not any intermediate neurons.

As an example, let us consider a 5-bits binary number 01100 (12 in decimal). The way the SN P system computes the two's complement is represented in TABLE Ⅰ. It depictes the number of spikes present in some key neurons and the output produced by the neuron Com to the environment. The input and output sequences are written in bold.

TABLE I. NUMBER OF SPIKES PRESENT IN KEY NEURONS OF AN SN P SYSTEM DURING THE COMPUTATION OF TWO'S COMPLEMENT OF 01100

Time step	**Neuron aux_1**	**Neuron aux_5**	**Neuron aux_3**	**Neuron Com**	**Output**
1	**0(0)**	1	1	3	-
2	**0(0)**	1	1	4	-
3	**a(1)**	1	1	4	**0(0)**
4	**a(1)**	1	1	5	**0(0)**
5	**0(0)**	-	1	2	**a(1)**
6	-	-	-	1	**0(0)**
7	-	-	-	0	**a(1)**

IV. ADDITION AND SUBTRACTION OF SN P SYSTEMS ON SIGNED INTEGERS

The SN P system performing addition on signed integers is shown in Figure 2.

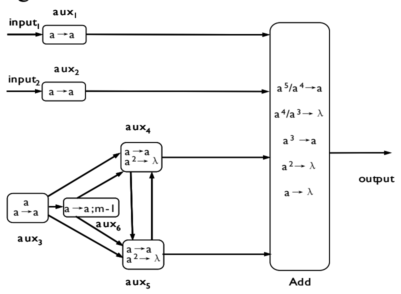

Figure 2. SN P System simulating addition on signed integers

The system has two input neurons, the first m-bits signed integers is provided through input neuron aux_1 and the second one is inputted into neuron aux_2. The presence of a spike in the neuron Add indicates a carry of the previous addition. The function moudule including neuron aux_3,aux_4,aux_5 and aux_6 provides both-routes m sequential spikes to Neuron Add from the 2nd step to the (m+1)th step.The function of this moudule is the same as above besides the number of output-routes.Integrating all of input combination, we can have 3 cases:

① If both the inputs are 1(spike), then the output neuron Add receives 4 spikes from the neuron aux_1, aux_2, aux_4 and aux_5 respectively. If the neuron Add is already having a spike(carry), then the number of spikes becomes 5 and triggers the rule $a^5/a^4 \to a$ otherwise it has 4 spikes and fires using the rule $a^4/a^3 \to \lambda$ leaving one spike in the neuron Add in either case. The presence of a spike in the neuron indicates a carry. This encodes the two operations 1+1=0 with carry 1 and (1)+1+1=1 with carry 1.

② If one of the input bit is zero, for example if the input $_1$ is 0 and input $_2$ is 1 then the neuron Add receives 3 spikes from the neuron aux_2, aux_4 and aux_5 respectively.The output neuron Add has either 3 or 4(in case carry) spikes and fires using $a^3 \to a$ or $a^4/a^3 \to \lambda$ respectively. These rules encode the two operations 0+1=1 and (1)+0+1=0 with carry 1 respectively.

③ If both the input neurons do not receive a spike(both the inputs is 0), then the neuron Add receives 2 spikes from the neuron aux_4 and aux_5 respectively and it will have either 2 or 3(again in case of carry of the previous operation) spikes and fires using $a^2 \to \lambda$ or $a^3 \to a$. These two rules do not leave any carry encoding the operations 0+0=0 and (1)+0+0=1 respectively.

The last rule in the output neuron $a \to \lambda$ allows the last overflow bit to be discarded.The procedure confirms the correctness of the system for performing the addition of two equal-bits binary signed integers.

For example, let us consider the addition of -1 and -5 on 4-bits signed integers. Number -1 is represented as 1111 and -5 is represented as 1011 in two's complement form respectively. The two binary sequences will form the inputs for the SN P system. The number of spikes present in each neuron in every step and the output produced by the system is depicted in TABLE Ⅱ. The input and output sequences are written in bold.The output result is 1010 in two's complement form,namely -6 in decimal code.This confirms the correctness of the system.

TABLE II. NUMBER OF SPIKES PRESENT IN EACH NEURON OF ADDITION SN P SYSTEM DURING THE ADDITION OF 1111+1011 = 1010

Time step	Neuron aux_1	Neuron aux_2	Neuron aux_4	Neuron aux_5	Neuron Add	Output
1	**a(1)**	**a(1)**	1	1	-	-
2	**a(1)**	**a(1)**	1	1	4	-
3	**a(1)**	**0(0)**	1	1	5	**0(0)**
4	**a(1)**	**a(1)**	1	1	4	**a(1)**
5	-	-	-	-	5	**0(0)**
6	-	-	-	-	1	**a(1)**

Two's complement subtraction is the binary addition of the minuend to the two's complement of the subtrahend (adding a negative number is the same as subtracting a positive one). That means $a-b$ becomes $a+(-b)$. The SN P system for addition can to perform subtraction.

V. MULTIPLICATION ON TWO POSITIVE INTEGERS

The spiking neural P system that performs the multiplication of two arbitrary positive integers is shown in Figure 3. In the SN P system, both the two operands are supplied as inputs.If we consider to perform the multiplication of $Z = X \times Y$, the multiplicand X is m-bits and the multiplicator Y is n-bits binary true code ,then the multiplicand X should be inputted to $input_1$ and the multiplicator Y should be inputted to $input_2$ respectively as Figure 3.In order to reduce the numbers of computing neurons,we should choose the smaller operand as the multiplicator,namely $n \le m$.

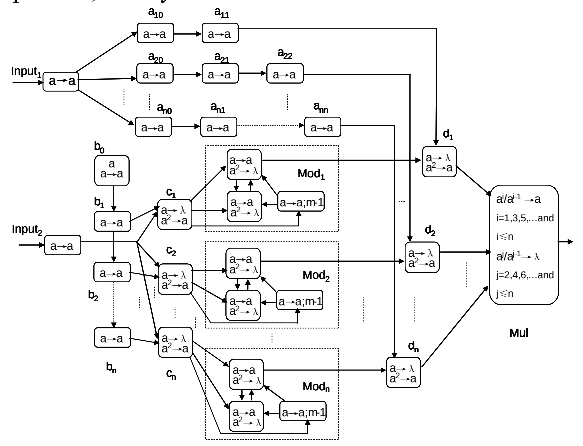

Figure 3. SN P System simulating multiplication with two arbitrary positive integers

Given: $X = \sum_{i=0}^{m-1} x_i 2^i$, $Y = \sum_{j=0}^{n-1} y_i 2^i$,then $Z = X \times Y = (\sum_{i=0}^{m-1} x_i 2^i) \times (\sum_{j=0}^{n-1} y_i 2^i)$

$$= \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} x_i y_{ji} 2^{i+j} = \sum_{i=0}^{m-1} y_0 x_i 2^{i+j} + \sum_{i=0}^{m-1} y_1 x_i 2^{i+j} + \ldots + \sum_{i=0}^{m-1} y_{n-1} x_i 2^{i+j}$$

According to this expression above, it becomes obvious that computing the multiplication $Z = X \times Y$ can translate to computing the addition of n sub-expressions(n-additions),namely $Z = X \times Y$ can be calculated as the addition of as many copies of X as the number of digits Y_j equal to 1 that appear in the binary representation of Y.Such copies have to be padded with j zeros on the right (that is, they have to be multiplied by 2^j), to take into account the correct weight of Y_j.

In Fig.3., the m-bits multiplicand X is inputted to input1 and the n-bits multiplicator Y is inputted to input2 bit by bit respectively.the functional neurons of the first sub-expression $\sum_{i=0}^{m-1} y_0 x_i 2^{i+j}$ consists of the neuron aux10,aux11 and d1.Accordingly, the functional neurons of the (k)th sub-expression $\sum_{i=0}^{m-1} y_k x_i 2^{i+j}$ consists of the neuron aux(k+1)0,aux(k+1)1,…, and d(k+1) (k+1) ($0 \le k \le n-1$).If the number of digits Y_j ($0 \le j \le n-1$)equal to 1 that appear in the binary representation of Y ,then the neuron cj+1 receive 2 spikes at the (2+j)th step.The 2 spikes will trigger the rule $a^2 \to a$ and send 3 spikes to the neighbor moudulej+1,which will work at the next steps.The function of the moudulej+1 is same as above,namely providing m sequential spikes to the neuron dj+1 from the (4+j)th step to the (m+4+j)th step,then the neuron d j+1 will have 1 or 2 spikes from the (4+j)th step to the (m+4+j)th step. If the number of digits Y_j equal to 0 that appear in the binary representation of Y ,then the neuron dj+1 will receive 0 or 1 spike from the (4+j)th step to the (m+4+j)th step.The rules consists of a family of $a^i / a^{i-1} \to a$ (if i is odd and i⩽n) and $a^j / a^{j-1} \to \lambda$ (if j is even and j⩽n) in the output neuron Mul.

For example, let us consider $Y = 26$, whose binary representation is 11010_2.Such a representation has three digits equal to 1, at the positions 2, 4 and 5. In the system illustrated in Figure 3, the neuron d_2, the neuron $d_4$and the neuron d_5 will have 1 or 2 spikes from the (5)th step to the (m+5)th step, the (7)th step to the (m+7)th step and the (8)th step to the (m+8)th step respectively. The rules will include $a \to a$, $a^2/a \to \lambda$, $a^3/a^2 \to a$, $a^4/a^3 \to \lambda$, $a^5/a^4 \to a$ and $a^6/a^5 \to \lambda$ in the neuron Mul.Because only the neuron d_2, the neuron $d_4$and the neuron d_5 can send spikes to the neuron Mul,so the spike number of the neuron Mul will be 1 or 2 or 3,namely the three rules include $a^4/a^3 \to \lambda$, $a^5/a^4 \to a$ and $a^6/a^5 \to \lambda$ will be not work.

We conclude this section with an example of multiplication.We will take $Y = 26$ as the multiplicator and input 11010 to input_2 bit by bit. Simultaneously , We take $X = 29$ as the multiplicand and input 11101 to input_1 bit by bit. TABLE Ⅲ reports the number of spikes contained in some key neurons during the computation, as well as the number of spikes sent to the environment. According to this computation, the output of the multiplication is 1011110010_2, which is the binary representation of 754.

TABLE III. NUMBER OF SPIKES PRESENT IN KEY NEURONS OF MULTIPLICATION SN P SYSTEM DURING THE MULTIPLICATION OF 11101x11010 = 1011110010

step	**In_1**	**In_2**	**aux_{11}**	**aux_{22}**	**aux_{33}**	**aux_{44}**	**aux_{55}**	**d1**	**d2**	**d3**	**d4**	**d5**	**Mul**	**Out**
1	**1**	**0**	-	-	-	-	-	-	-	-	-	-	-	-
2	**0**	**1**	-	-	-	-	-	-	-	-	-	-	-	-
3	**1**	**0**	1	-	-	-	-	-	-	-	-	-	-	-
4	**1**	**1**	0	1	-	-	-	1	-	-	-	-	-	-
5	**1**	**1**	1	0	1	-	-	0	2	-	-	-	0	-
6	-	-	1	1	0	1	-	1	1	1	-	-	1	**0**
7	-	-	1	1	1	0	1	1	2	0	2	-	0	**1**
8	-	-	-	1	1	1	0	1	2	1	1	2	2	**0**
9	-	-	-	-	1	1	1	-	2	1	2	1	3	**0**
10	-	-	-	-	-	1	1	-	-	1	2	2	3	**1**
11	-	-	-	-	-	-	1	-	-	-	2	2	3	**1**
12	-	-	-	-	-	-	-	-	-	-	-	2	3	**1**
13	-	-	-	-	-	-	-	-	-	-	-	-	2	**1**
14	-	-	-	-	-	-	-	-	-	-	-	-	1	**0**
15	-	-	-	-	-	-	-	-	-	-	-	-	0	**1**

VI. CONCLUSION

In this paper we have presented some simple SN P systems that perform the following arithmetic operations: two's complement , addition/subtraction on negative integers and multiplication on two arbitrary positive integers. All the numbers given as inputs to these systems are expressed in binary form, encoded as a spike train in which at each time instant the presence of a spike denotes 1, and the absence of a spike denotes 0. The outputs of the computations are also expelled to the environment in the same form.Especially,the multiplication operation with SN P system provides an answer to an open problem formulated by by Miguel A. Gutiérréz-Naranjo in [2].

The motivation for this work lies in implement a CPU using only spiking neural P systems. To this aim, the first work is to design the Arithmetic Logic Unit of the CPU and hence to perform a compact arithmetical and logical operations on spiking neural P systems.

ACKNOWLEDGMENT

Xianwu Peng, Born in 1970, Ph.D. candidate. His main reasearch interests include membrane computing, distributed computing. The authors would like to thank the anonymous referees for their valuable comments. The research of the authors is supported by the Scientific Research Fund of Hunan Provincial Education Department Nos.12c0120 and the National Natural Science Fund of China under Grant Nos.61272063,61202462.

REFERENCES

[1] M. Ionescu, Gh. Păun, T. Yokomori, Spiking Neural P Systems, Fund. Infor. 71,279-308 (2006).

[2] Gutiérréz-Naranjo, M. A., Leporati, A., First Steps Towards a CPU Made of Spiking Neural P Systems, Int. J. of Computers, Communications and Control 4 244-252, (2009).

[3] Venkata Padmavati Metta, Kamala Krithivasan, Deepak Garg, Some Characteristics of Spiking Neural P Systems with Anti-Spikes, Proceedings of the Eleventh International Conference on Membrane Computing (CMC11), 291-304,(2010).

[4] L.Pan, Gh. Păun, Spiking Neural P Systems with Anti-Spikes, Int. J. of Computers, Communications and Control 4,273-282, (2009).

[5] M. Ionescu, D. Sburlan, Some Applications of Spiking Neural P systems, J. of Computing and Informatics 27 515-528 (2008).