PCMOS-based Hardware Implementation of Bayesian Network

Zhang Weijia, Goh Wang Ling, and Yeo Kiat Seng

Abstract – **Bayesian network [1] has received considerable attention in a great variety of research areas such as for artificial intelligence, bioinformatics, medicine, engineering, image processing, and various kinds of decision support systems. But up till now, most of the investigation on Bayesian network has been on its theory, algorithms and software implementations. This paper presents the Bayesian network from a totally new perspective--hardware circuit implementation. By using the new-born technology of Probabilistic CMOS (PCMOS) [2]-[5] and taking advantage of the statistical properties of simple logic gates, the Bayesian network can be constructed using hardware circuits. Such hardware implementation revealed the advantages in aspects of power consumption, delay time and quality of randomness.**

I. INTRODUCTION

A. Bayesian Network

A Bayesian network [1] is a probabilistic graphical model that represents a set of variables and their probabilistic dependencies. From the graphical view, it is a directed acyclic graph whose nodes represent variables and whose arcs encode the dependencies between the variables. Bayesian network is a popular topic among the mathematicians and computer scientists. This is mainly attribute to its conceptual and mathematical nature, not to mention the intense calculation. In this paper, our focus is not on the complicated structures and mathematical characteristics of the Bayesian network. Instead, hardware implementation of a Bayesian network based on the PCMOS technology has been investigated and represented. By implementing the network using hardware circuit, we are able to lower the power consumption, reduce the delay time, and make the system highly randomized.

B. PCMOS

The concept of PCMOS was originated from Professor Krishna V. Palem's theory of probabilistic switching [2]. Rather than being taken as an impediment, the noise is regarded as a resource in the PCMOS technology [3]. As noise is introduced into a digital circuit, the outputs of the circuit become "probabilistic" instead of "deterministic". Hence, the correct outputs (the outputs we desired) will be obtained only with certain probabilities.

One of the applications of the PCMOS technology is in the probabilistic system, such as Encryption system, Automata system and Bayesian network. In a probabilistic system, the PCMOS technology can be used to implement the random number generator that serves to create a highly randomized bit sequence at a very low cost. The core of a PCMOS-based random number generator (also named as probabilistic generating cell in our proposed Bayesian network design) is the PCMOS switch or p-switch for simplicity. It is simply a CMOS switch with a noise source coupled at its input node. Figure 1 shows the realization of a p-switch [3]. The resistor shown in the figure is taken as a source of thermal noise, which is known to follow the Gaussian distribution. An amplifier is added to amplify the noise signal to make it comparable with the supply voltage in today's technology. The sub-threshold amplifier is used because of its ability to save power.

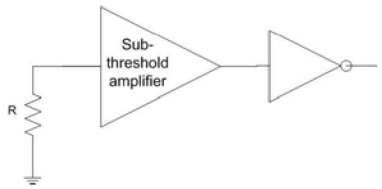

Figure 1. Realization of a p-switch cell

There are several ways to adjust the probability of the output of a p-switch. The probability we are discussing in this paper is defined as the probability of getting a logic "1" at the output of a circuit. Probability can be varied by: (1) adjusting the RMS value of the noise, (2) changing the supply voltage, (3) tuning the PMOS-to-NMOS ratio, and (4) adjusting the DC level of the noise signal. Among these methods, the fourth approach is most effective and convenient, and thus we had adopted it in our design. To adopt such probability adjusting method, a special sub-threshold amplifier that includes a DC-OFFSET pin, which allows us to adjust the DC level of the output of the amplifier, is developed in our proposed design [7].

The relationship between the probability of an ideal p-switch (that has an ideal voltage transfer characteristic) and the DC level of noise can be expressed using the following equation:

$$P = \frac{1}{2} + \frac{1}{2} erf\left(\frac{V_{th} - V_{dc}}{\sigma\sqrt{2}}\right) \quad (1)$$

II. PCMOS-BASED BAYESIAN NETWORK

A. Architecture

Zhang Weijia, Goh Wang Ling and Yeo Kiat Seng are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.
E-mail: zhan0151@ntu.edu.sg

The proposed system of our PCMOS-based Bayesian network consists of two parts: probabilistic generating block and logic network. The probabilistic generating block is for generating random bits with different probabilities, whereas the logic network is used to represent the relationships among the nodes in a Bayesian network, as well as to calculate the unknown probabilities associated with each node. The overall architecture is presented in Figure 2 and detail discussions are given in the following subsections.

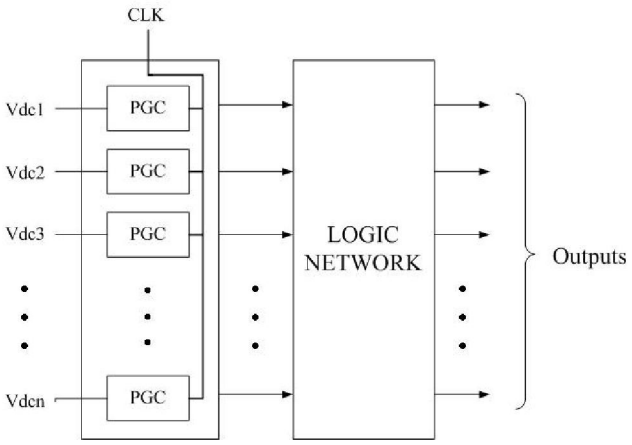

Figure 2. Architecture of a PCMOS-based Bayesian network

B. Probabilistic Generating Block

The probabilistic generating block is made up of a number of probabilistic generating cells (PGCs). Each of this cell can generate a bit of "1" with a specific probability. The architecture of a PGC is given in Figure 3. It has three parts: p-switch, buffer and flip-flop. The p-switch has already been introduced in section I. The buffer is constructed using two inverters. It is used to strengthen the output signal of the switch and to restore the signals whose voltage levels hover around $\frac{V_{dd}}{2}$ to the logical "high" or "low". The flip-flop used to synchronize the PGCs (all the flip-flops in the block are using the same clock) is a master-slave flip-flop formed by two D latches.

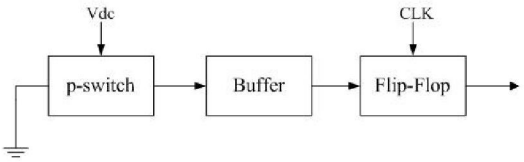

Figure 3. Probabilistic generating cell

C. Logic Network

As we know, the Bayesian network involves an amount of probability addition and multiplication. These arithmetic operations may consume a lot of time and energy when computed in computers. In our proposed design, we made use of two simple logic gates--AND gate and OR gate, together with some inverters, to construct the logic network. By taking advantage of the statistical properties of these two simple logic gates, the probability calculations can be easily accomplished.

To perform the probability multiplication, we can use the AND gate. The output of an AND gate will be "1" if and only if all the inputs are "1". Hence, the probability function of a two-input AND gate can be expressed as:

$$P(Y_{and}) = P(X_{and1}) \times P(X_{and2}) \tag{2}$$

where $P(Y_{and})$ is the probability of the output of the AND gate, $P(X_{and1})$ and $P(X_{and2})$ are the probabilities of its two inputs, respectively.

For the OR gate and probability addition, things are a little more complicated. Since the output of an OR gate can be "1" if only one of its inputs is "1", the equation to represent the relationship between the probability of output of a two-input OR gate and the probabilities of its inputs should therefore be:

$$P(Y_{or}) = P(X_{or1}) + P(X_{or2}) - P(X_{or1}) \times P(X_{or2}) \tag{3}$$

where $P(Y_{or})$ is the probability of the output of the OR gate, $P(X_{or1})$ and $P(X_{or2})$ are the probabilities of its two inputs, respectively.

It is easily noted that $P(Y_{or}) \neq P(X_{or1}) + P(X_{or2})$. That is because $P(X_{or1})$ and $P(X_{or2})$ have an overlap term, $P(X_{or1} = X_{or2} = 1)$ which is equal to $P(X_{or1}) \times P(X_{or2})$. So this term should be deducted when computing the output probability of an OR gate.

However, we are still able to use the OR gate to realize the probability addition in our Bayesian network. This is because, from equation (3), we can see that only when there is a possibility that both inputs are "1" at the same time would there be a difference between the probability of the output bit and the sum of the probabilities of the input bits, and fortunately, this possibility can be totally eliminated in our design. So, under such condition, equation (3) can be modified to:

$$\begin{aligned} &P(Y_{or}) = P(X_{or1}) + P(X_{or2}), \\ &\text{when } P(X_{or1} = X_{or2} = 1) = 0 \end{aligned} \tag{4}$$

In this subsection, we have only discussed the two-input AND/OR gates, because any multi-input AND/OR gates can be implemented by cascading the two-input gates.

III. An Example of Bayesian Network

A. Description of the Network

In this section, a simple illustration on the use of Bayesian network is provided. This is a variation of one example discussed in [1]. The graphical description of this

network is depicted in Figure 4. In this example, we supposed that a flu (node of *Flu*) may cause a fever (node of *Fever*). The symptom can be measured by a thermometer, whose reading *Therm* may be somewhat unreliable. As an imaginary experiment, a person is tested with this thermometer. If the thermometer indicates a high temperature, this person will be given a dose of Aspirin (node of *Take Aspirin*). After some time, it will be checked if this person is still having fever (node of *Feverlater*) and whether there is any side effect (node of *Reaction*). He will then be told of all the test results and thereafter make a decision (node of *Decision*) on whether to buy the drug or not. As the final output is the decision of a person, it can be regarded as a decision network. The conditional probability tables (CPTs) of each node are also provided in the figure. The probabilities given in the figure are based on the assumption that no prior information of the person-in-test had been given. If we can acquire more information, we will only need to change the corresponding probability values in the CPTs. For example, if we constrain the test objects as a group of flu patients, P(Flu=T) should then be changed from 0.05 to 1.

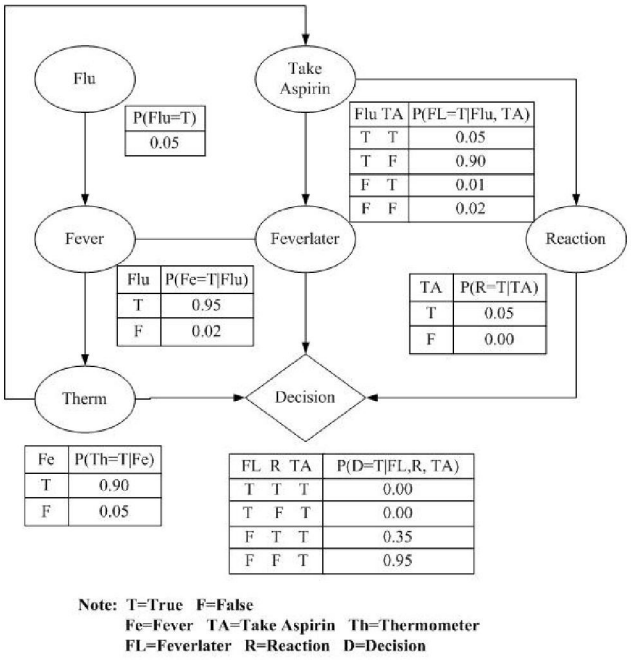

Figure 4. An example of Bayesian network

B. Implementation of the Network

Since there are 15 different probabilities involved in the network, fifteen PGCs are required in the probabilistic generating block. According to the probabilities required, and using equation (1), the DC voltage levels of the noise sources to be introduced into these PGCs can be determined and summarized in Table I. The logic network of the system is depicted in Figure 5. The circuit is simply constructed with AND gates, OR gates and invertors based on the equations discussed in section II. In this circuit, IN1~IN15 denote the 15 inputs coming from the probabilistic generating block, and OUT1~OUT5 represent the outputs of the five nodes shown in the Bayesian network of Figure 4. These nodes are: *Fever*, Take Aspirin, *Feverlater*, *Reaction*, and *Decision*.

TABLE I
DC LEVELS OF THE NOISE SOURCES (V)

Vdc1	**Vdc2**	**Vdc3**	**Vdc4**	**Vdc5**
1.40	0.41	1.52	0.52	1.40
Vdc6	**Vdc7**	**Vdc8**	**Vdc9**	**Vdc10**
1.40	0.52	1.60	1.52	1.40
Vdc11	**Vdc12**	**Vdc13**	**Vdc14**	**Vdc15**
1.80	1.80	1.28	1.02	0.41

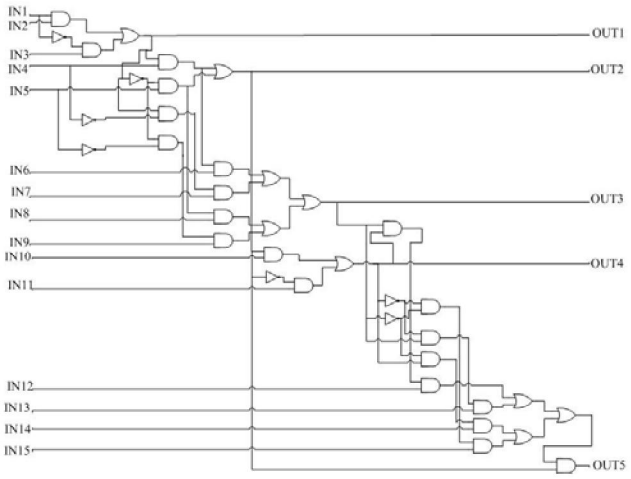

Figure 5. Schematic circuit of logic network

IV. SIMULATIONS AND RESULTS

A. Simulation Condition and Noise Model

The whole system of the proposed Bayesian network has been simulated in HSPICE. All the circuits are realized using Chartered 0.18 um CMOS process. In our simulations, the noise is modeled as a PWL (piecewise linear) voltage source whose data points are random numbers following Gaussian distribution and are generated by Matlab.

B. Simulation Results

To validate our analysis on the relationship between the probability of the output of a p-switch and the DC level of the noise source that is coupled to the input node, we first performed simulation on a single p-switch. Figure 6 provides both the simulation and the analytical results. It is noted that the two sets of results match very well.

To verify the functionality of this network, we performed the simulation for a sufficient duration to let it generate a 10000-bits sequence at each node and then "count" how many "1" in each sequence so that we can determine the approximate probability of the output at each node. The simulation results together with the analytical results are all tabulated in Table II. In this table, we created two sections, one for the analytical result and the other for the simulation result. The column of "flu unknown" means that we know nothing about the health of the person-in-test

and the column of "flu known" means that we already knew he/she is a patient suffering from flu.

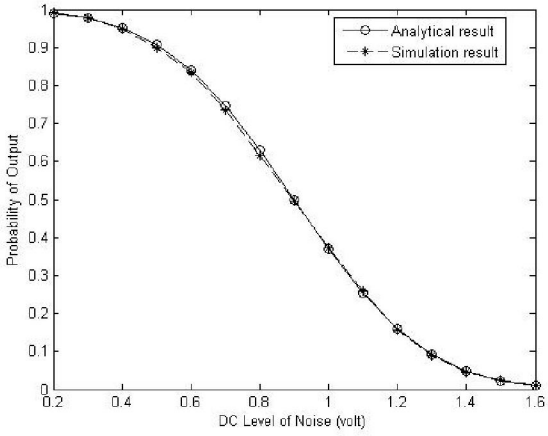

Figure 6. Relationship between probability and DC level of noise

TABLE II
PROBABILITIES OF OUTPUT BITS

Node	Analytical Result	
	Flu Unknown	Flu Known
Fever	0.066	0.950
Take Aspirin	0.11	0.857
Feverlater	0.027	0.130
Reaction	0.005	0.043
Decision	0.10	0.69
Node	Simulation Result	
	Flu Unknown	Flu Known
Fever	0.058	0.945
Take Aspirin	0.1	0.846
Feverlater	0.03	0.130
Reaction	0.005	0.036
Decision	0.09	0.70

C. Comparison Between Hardware and Software Implementation

To prove that the PCMOS-based hardware implementation of the Bayesian network is superior than its software counterpart, we had also written a C program to simulate the network. This program was run on a CPU with Intel Pentium M processor that is of 1.73 GHz. The comparisons in aspects of energy consumption (in generating one single bit of output), performance and quality of randomness between our design and the software implementation are provided in Table III. It is evident that our PCMOS-based hardware design outperformed the software implementation in all aspects.

TABLE III
COMPARISON BETWEEN PROPOSED HARDWARE DESIGN AND SOFTWARE IMPLEMENTATION

	Software	PCMOS
Energy (joule)	2.2e-6	1.5e-10
Delay (second)	2.3e-7	9.5e-10
Quality of Randomness	LOW	HIGH

V. CONCLUSION

In this paper, we have illustrated our hardware implementation of a Bayesian network by using the PCMOS technology and simple logic gates. Simulation results have been demonstrated together with the analytical results, to verify the functionality of the proposed system. The performance of this system has also been analyzed. Comparison between our proposed design and its software counterpart proved the advantages of the PCMOS-based hardware design.

ACKNOWLEDGEMENT

Zhang Weijia is grateful to the Nanyang Technological University (NTU) of Singapore for providing the graduate scholarship.

REFERENCES

[1] Kevin B.Korb, Ann E.Nicholson, *Bayesian Artificial Intelligence,* CHAPMAN & HALL/CRC, 2004.

[2] Krishna V. Palem, "Energy Aware Computing through Probabilistic Switching: A Study of Limits," *IEEE Transactions on Computers,* Vol.54, No.9, Sept. 2005.

[3] S.Cheemalavagu, P.Korkmaz, K.V.Palem, B.E.S.Akgul and L.N.Chakrapani, "A Probabilistic CMOS Switch and its Realization by Exploiting Noise".

[4] S.Cheemalavagu, P.Korkmaz, and K.V.Palem, "Ultra Low Energy Computing via Probabilistic Algorithms and Devices: CMOS Device Primitives and the Energy-probability Relationship," *Proc. of The 2004 International Conference on Solid State Devices and Materials,* pages 402–403, Tokyo, Japan, Sept. 2004.

[5] P.Korkmaz, B.E.S.Akgul, K.V.Palem and L.N.Chakrapani, "Advocating Noise as an Agent for Ultra-low Energy Computing: Probabilistic Complementary Metal-Oxide-Semiconductor Devices and Their Characteristics," *Japanese Journal of Applied Physics,* Vol.45, No.4B, 2006.

[6] Mohamed Abbas, Makoto Ikeda, Kunihiro Asada, "Noise Effects on Performance of Low Power Design Schemes in Deep Submicron Regime," *Proceedings of the 19th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2004.

[7] Huang Yan Yuan Daniel, *Report on Industrial Research Attachment,* Nanyang Technological University, Singapore, 2006.