PAPER

Scalable Hardware Winner-Take-All Neural Network with DPLL*

Masaki AZUMA[†], *Nonmember and* **Hiroomi HIKAWA**[††a)], *Member*

SUMMARY Neural networks are widely used in various fields due to their superior learning abilities. This paper proposes a hardware winner-take-all neural network (WTANN) that employs a new winner-take-all (WTA) circuit with phase-modulated pulse signals and digital phase-locked loops (DPLLs). The system uses DPLL as a computing element, so all input values are expressed by phases of rectangular signals. The proposed WTA circuit employs a simple winner search circuit. The proposed WTANN architecture is described by very high speed integrated circuit (VHSIC) hardware description language (VHDL), and its feasibility was tested and verified through simulations and experiments. Conventional WTA takes a global winner search approach, in which vector distances are collected from all neurons and compared. In contrast, the WTA in the proposed system is carried out locally by a distributed winner search circuit among neurons. Therefore, no global communication channels with a wide bandwidth between the winner search module and each neuron are required. Furthermore, the proposed WTANN can easily extend the system scale, merely by increasing the number of neurons. The circuit size and speed were then evaluated by applying the VHDL description to a logic synthesis tool and experiments using a field programmable gate array (FPGA). Vector classifications with WTANN using two kinds of data sets, Iris and Wine, were carried out in VHDL simulations. The results revealed that the proposed WTANN achieved valid learning.

key words: *neural network, winner-take-all, supervised learning, digital phase-locked loop, hardware architecture*

1. Introduction

Neural networks are used to predict and recognize climate, images and voices. A winner-take-all neural network (WTANN) is a supervised neural network. The most important feature of WTANN is its ability to determine a winner neuron that has a weight vector nearest to the input vector, which is called a "winner-take-all (WTA)" operation. Hence, in most applications, WTANNs are treated as classifiers. The WTANN has been extensively adopted to solve problems in pattern recognition and signal processings [1], [2]. The MAXNET is a well-known WTANN [3]. The WTA operation is used in other neural networks, such as self-organizing map (SOM) [4].

Neural networks implemented in software are extremely flexible, but performance that can be achieved with software solutions is insufficient if networks are too large. Therefore, hardware implementations of neural networks are preferable to software implementations. Hardware implementations can make use of the parallelism embedded in the neural network algorithm. Field programmable gate arrays (FPGAs) are suitable platforms to implement neural networks due to their shorter development times and lower costs.

Many designers and researchers have been developing VLSI implementations with various techniques, ranging from digital to analog and even optical. One effective approach to implementing neural networks in hardware is a pulse-stream based architecture [5], where pulse density or frequency is used to represent a neuron's signal level. In addition, a stochastic computation is used to implement the required computation [6].

It is generally known that information processing by the brain is performed by exchanging spikes between neurons, which involves rapid electrical changes in the shape of pulses. The Hodgkin-Huxley (HH) [7] model is one of the first detailed neuron models developed. A simple model is the leaky integrate and fire (LIF) model [8], which is computationally cheaper. In recent years, experimental evidence indicates that many biological neural systems use the timing of single spikes to encode and process information. This method, known as "temporal coding," is considered to be the coding mechanism in biological neural systems. In hardware spiking neural networks with temporal coding, coincidence or synchrony detection plays important roles in neural information processing. Many spiking neural networks have been reported [9]–[11].

Hikawa used a digital phase-locked loop (DPLL) as an arithmetic circuit for a SOM since the operation of DPLL is very similar to that of neurons, and the method succeeded in simplifying the circuit [12]. The input value was expressed by timing of the rising edge of the rectangular wave. SOM with phase modulated signals is very similar to the pulse-stream based neural networks because phase modulated rectangular waves can be regarded as temporal coded spikes. The main problem with this architecture is that the WTA was implemented by a global winner search based on a binary-tree search scheme. Thus, the search circuit must be designed in accordance with the number of neurons, so the system was not scalable since the winner search circuit must be re-designed if the number of neurons is increased.

Manuscript received November 7, 2014.
Manuscript revised June 14, 2015.
Manuscript publicized July 21, 2015.

[†] The author is with Aisin Seiki Co., Ltd., Kariya-shi, 448–8650 Japan.

[††] The author is with the Faculty of Science and Engineering, Kansai University, Suita-shi, 564–8680 Japan.

[*] A basic outline of this paper is going to be presented on "A new Winner-Take-All Neural Network Using DPLL and Phase Modulated Signal," IEEE Proc. ISPACS'13, pp.345–350, Nov. 2013.

a) E-mail: hikawa@kansai-u.ac.jp

DOI: 10.1587/transinf.2014EDP7371

In addition, a global communication channel was required because the circuit had to obtain all vector distance information from all neurons.

This paper proposes a new WTA circuit that can find winner neurons efficiently by taking advantage of the nature of phase modulated signals. XNOR is employed as the coincidence detector in the proposed WTA circuit, and a neuron having the nearest internal signal in the phase is detected as a winner neuron. Combined with DPLL, the proposed WTA circuit is employed to form a WTANN and its feasibility is examined. There is no winner search module in the proposed WTANN, since the winner search is carried out locally by all neurons. Therefore, WTANN has a simple structure and it can easily be extended, merely by connecting the neurons. The proposed WTANN was described by VHDL and its operation was tested with VHDL simulations.

First, the learning capabilities and scalability of the proposed WTA method that were demonstrated by multi-dimensional WTANN are described. Vector classifications using general data sets were performed in the second simulation. The WTANN was trained as a vector classifier that identified input vector classes of the Iris and Wine data sets. Then, the system was implemented on an FPGA, and the circuit size and operating speed were evaluated. The function of the proposed WTANN was also verified by experiments.

The remainder of this paper is organized as follows: Section 2 describes the WTANN algorithm. The detail of the proposed WTANN architecture with DPLLs is explained in Sect. 3. Simulation results are presented in Sect. 4. FPGA implementation, FPGA experiments and circuit properties are examined in Sect. 5, followed by conclusions in Sect. 6.

2. Winner-Take-All Neural Network

The general structure of a WTANN is outlined in Fig. 1. A conventional WTANN consists of multiple neurons, a supervised learning unit, and a winner search unit. An N dimensional vector, $\vec{w}_i$, called a weight vector, is assigned to all neurons.

$$\vec{w}_i = \{\mu_{i0}, \mu_{i1} \ldots, \mu_{iN-1}\}. \tag{1}$$

The operation of WTANN can be divided into two phases of learning and recall. Supervised learning is carried out in the learning phase, and weight vectors of all neurons are trained with a set of training vectors. The weights of neurons remain unchanged after the learning phase and are used in the recall phase.

The learning phase starts with an appropriate initialization of the weight vectors and a class number is assigned to each neuron. Training vectors $\vec{x}$ and their class numbers are subsequently presented to the neurons in multiple iterations.

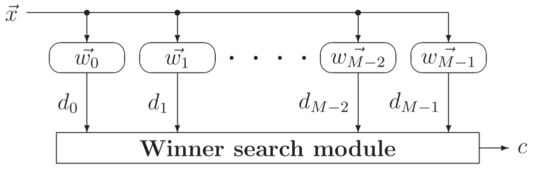


Fig. 1 Winner-take-all neural network (WTANN).

$$\vec{x} = \{\xi_0, \xi_1 \ldots, \xi_{N-1}\}. \tag{2}$$

The supervised learning unit activates neurons that are assigned to a given class, so that the weight vector of the selected neuron is updated to be closer to the training vector.

$$\vec{w_i}(k+1) = \vec{w_i}(k) + \alpha(\vec{x_i} - \vec{w_i}(k)), \tag{3}$$

Here, k is a time index, and α is the learning rate, $(0 \leq \alpha \leq 1)$. The weight vector of each neuron is properly adjusted to the vectors belonging to one of the classes by repeating the above computation.

The distances to all weight vectors are calculated in the recall phase for each input vector. The Euclidean distance is commonly used to measure the vector distance. However, most hardware neural networks employ Manhattan metric d_i instead of the Euclidean distance to reduce the computing cost. The saving in the silicon area by using the Manhattan metric is substantial since no squaring circuits are required.

$$d_i = \sum_{j=0}^{N-1} |\xi_j - \mu_{ij}| \tag{4}$$

The neuron having the smallest distance is determined to be the winner neuron.

$$c = \arg\min_i d_i. \tag{5}$$

The winner neuron search, i.e., the WTA operation, is generally implemented by a global winner search unit, as shown in Fig. 1. The winner search circuit searches for the winner neuron having the smallest vector distance after receiving the vector distances from all neurons. Various WTA architectures that determine the winner neuron were proposed. The MAXNET [3] is a well-known competitive architecture for selecting the maximum or minimum from a set of data. Since all nodes in the MAXNET are mutually connected, its entire configuration must be re-organized if the number of neurons is increased. Another popular WTA is the winner search circuit that is based on binary tree search [13], [14]. The circuit size increases in proportion to the number of neurons, so circuits become complicated due to the nature of binary-tree search. Since the vector distances d_i must be collected from all neurons, the circuit must be designed in accordance with the number of neurons. In addition, circuits must obtain all vector distance information from all neurons, which requires a global communication channel with wide bandwidth. Hence, the structure of the winner search circuit must be modified to accommodate additional neurons. Note that the binary-tree based winner search circuit was used in our previous work [12].

3. Winner-Take-All Neural Network with Digital Phase-Locked Loop

3.1 System Configuration

Figure 2 is a block diagram of the proposed WTANN. Input

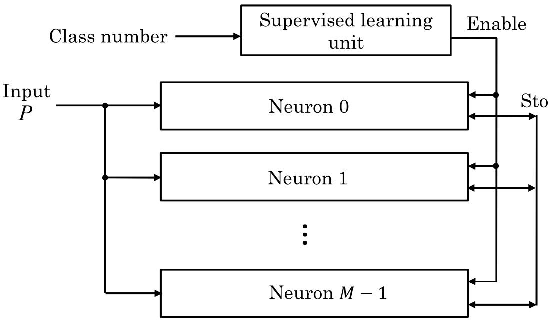


Fig. 2 WTANN with DPLL.

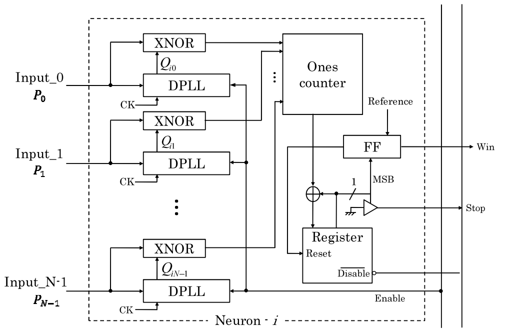


Fig. 3 Multi-dimensional neuron.

to the system is P, whose phase represents input vector elements ξ_0-ξ_{N-1}. The system consists of a supervised learning unit and multiple neurons.

Figure 3 outlines the proposed WTANN neuron for the multi-dimensional (N dimensional) vector. Each neuron contains multiple DPLLs, each of which processes one of the vector elements μ_{ij}. Therefore, an N DPLLs are needed for the WTANN neuron to handle N dimensional vector. Note that there is no winner search module in the system, because the winner search is carried out locally by all neurons.

The signals used in the proposed neuron circuit are outlined in Fig. 4. Q_{ij} and P_j are carrier signals whose phases are modulated to convey the vector elements μ_{ij} and ξ_j in Eqs. (1) and (2). Since other signals are used inside the neuron circuit, they are explained in the following subsections.

3.2 Digital Phase-Locked Loop

DPLL is mainly used in data transmission systems [15]. Figure 5 is a block diagram of the DPLL that is used in this paper. A phase detector (PD), a loop filter (LF), and a digital controlled oscillator (DCO) constitute the DPLL. The PD in the DPLL detects the phase difference between the input signal and its internal signal. Then, the LF converts the phase difference signal into a control signal, and the phase difference is decreased by sending the control signal to the DCO. The phase of the local signal eventually synchronizes to that of the incoming signal by repeating the above process.

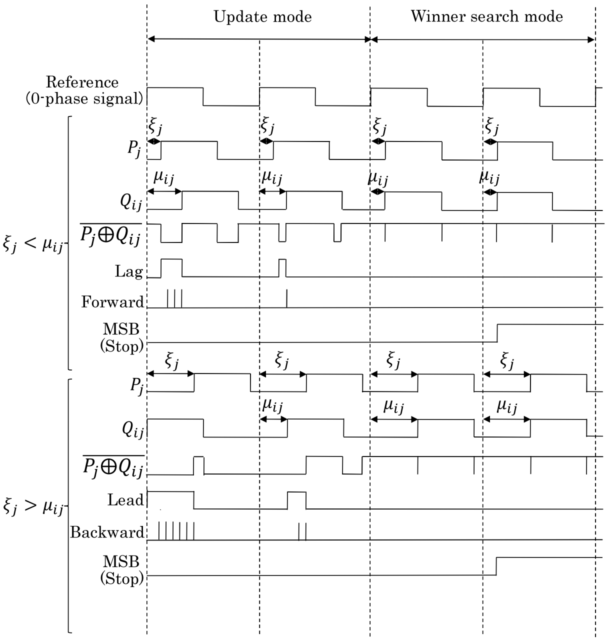


Fig. 4 Signals in proposed WTANN.

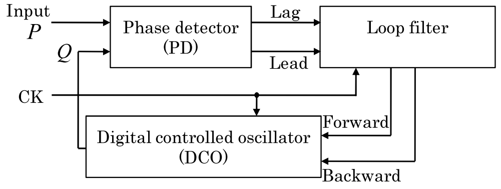


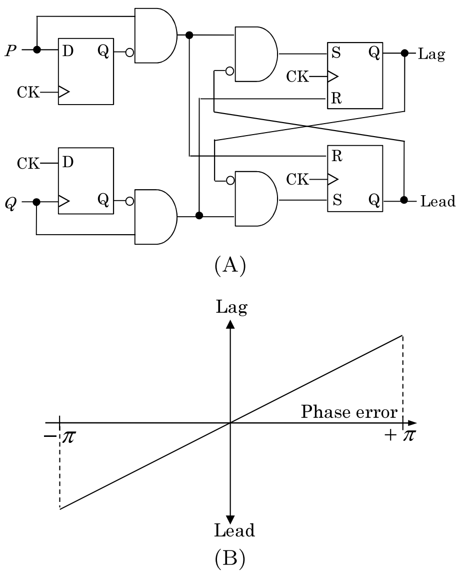
Fig. 5 Digital phase-locked loop (DPLL)

3.2.1 Phase Detector

Figure 6 (A) shows a block diagram of the phase detector. The PD compares the rising edges of the input signal and its local signal; then, one of its Lead or Lag output signals is made to be high level. Figure 4 depicts two situations, where $\xi_j < \mu_{ij}$ and $\xi_j > \mu_{ij}$. In the case of $\xi_j < \mu_{ij}$, the phase of Q_{ij} is behind that of P_j, and the Lag signal is generated from the PD, which is then converted to a Forward signal by the LF. The Lead and Backward signals are generated if $\xi_j < \mu_{ij}$. The output level of the PD is given as the logical-high duration of the output signal. Resulting phase comparison characteristic of the PD is a linear function shown in Fig. 6 (B). The PD output signals are then fed to the LF.

3.2.2 Loop Filter

The LF is made of two counters, as shown in Fig. 7. One of the counters is enabled while the corresponding input signal (Lag or Lead) is high level, and a phase control signal (Forward or Backward) is generated when the counter overflows. With this operation, the LF can detect the statistical tendency of the occurrence of Lag or Lead, and in this system, the LF is used to provide the learning coefficient α in



(A) (B)

Fig. 6 Phase detector, (A) block diagram, (B) characterristic

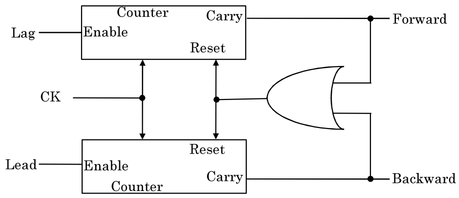


Fig. 7 Loop Filter

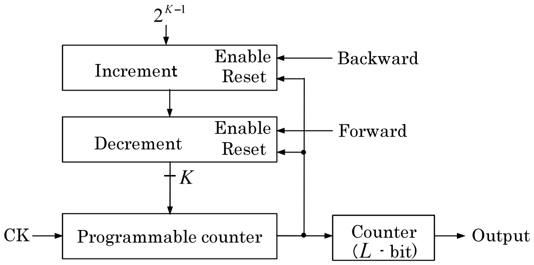


Fig. 8 Digital controlled oscillator (DCO)

(3).

3.2.3 Digital Controlled Oscillator

There is a block diagram of the DCO in Fig. 8. The DCO consists of increment/decrement circuits, a K-bit programmable counter, and an L-bit counter. Typical DCO signals are shown in Fig. 9. A single forward or backward control signal changes the output phase of the DCO by a single clock cycle. Output of the DCO is the phase-modulated carrier signal Q_{ij} in Fig. 4.

The programmable counter divides the input signal (clock signal) by 2^{K-1} in the absence of the control pulse shown in Fig. 9 (A) and then the following L-bit counter divides the signal by 2^L. The cycle of the DCO's output signal is 2^{K+L-1}, and its center frequency is given by

$$f_{DCO} = \frac{f_{CK}}{2^{K+L-1}}. \tag{6}$$

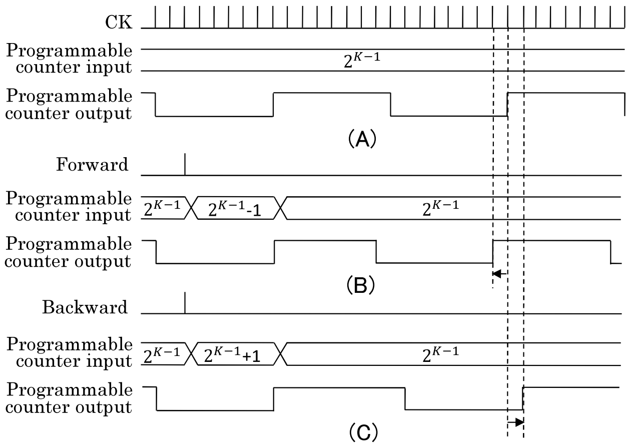

CK
Programmable counter input
2^{K-1}
Programmable counter output
(A)
Forward
Programmable counter input
2^{K-1}
2^{K-1}-1
2^{K-1}
Programmable counter output
(B)
Backward
Programmable counter input
2^{K-1}
2^{K-1}+1
2^{K-1}
Programmable counter output
(C)

Fig. 9 Phase control in DCO.

Figures 9 (B) and (C) indicate phase control by *Forward* and *Backward* signals. The *Forward* signal activates the decrement circuit and the output cycle of the programmable counter is shortened to $2^{K-1} - 1$, resulting in the rising edge of the DCO being moved forward by one clock cycle. In the presence of the *Backward* signal, the cycle period of the programmable counter is altered to $2^{K-1} + 1$, and the edge is moved backward. In this way, each *Forward* or *Backward* pulse can adjust the timing of the output signal's rising edge. Note that the DCO signal cycle is 2^L times longer than the programmable counter output in Fig. 9. A single programmable counter could have been used instead of the two counters, but two counters are used to reduce the circuit sizes of the increment, decrement circuits as well as the programmable counter.

The amount of adjustment is the same as the period of the clock signal. Thus, phase control is quantized as

$$\Delta = \frac{2\pi}{2^{K+L-1}}. \tag{7}$$

As each weight vector element is given by the phase of the local signal, Δ governs the precision of the weight vectors.

3.3 Winner Find Operation

In *update mode* of the learning phase, the supervised learning unit activates the neuron that is assigned to the class to which the training vector belongs. The phase control of the DPLL in this neuron is enabled so that its local signal phase is adjusted to be closer to that of the training vectors.

The mode is switched to *winner search mode* in the recall phase. The number of P_i and Q_{ij} pairs having the same value are detected by the XNOR unit at each clock input, and they are counted by the ones counter and accumulated in the register in Fig. 3. Same as the PD, the output level of the XNOR is the high-level duration of the output signal. As shown in Fig. 4, the closer the two signals P_i and

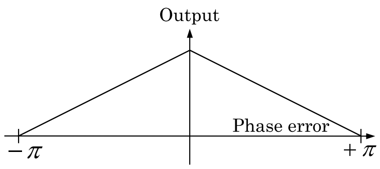


Fig. 10 XNOR output characteristic

Q_{ij} are, the longer the XNOR generates a logical high output; thus, the output characteristic of the XNOR is inversely proportional to the phase difference between two signals, as shown in Fig. 10. If the MSB of the register in neuron i becomes high earlier than other neurons, then neuron i is the winner neuron. The winner neuron activates the *Stop* signal that is broadcast to all neurons and their registers are disabled. Note that *Stop* is a bidirectional signal. The winner neuron is determined in this way by distributed winner search. Thus, no complex winner search methods or wide band-width global communication links are required.

4. Simulation

The proposed WTANN was described through VHDL and tested by VHDL simulations. Two kinds of VHDL simulations were carried out to test and verify the operation of WTA circuit and the learning characteristic of the WTANN.

4.1 Simulation Setup

Figure 11 is a block diagram of the system used to simulate the classifications, where M is the number of classes that were classified. The system consisted of the DPLL WTANN, M ROMs, M phase modulators, and Signal generator to provide input signals and the timing signal generator. The ROM stored $K + L - 1 + T$-bit sample data, T-bits of which indicated the class number. $M \times N$ phase measurement circuits were also included to measure the weight vectors assigned to the neurons. These were not necessary for WTANN operation, but were used to monitor the signal phases. The timing generator produced various timing signals to control the system.

4.2 Learning Capability and Scalability

Artificially generated test data sets were used to verify learning capabilities and scalability. The data sets were made of 8 or 16 classes each of which was made of 16 three-dimensional random vectors. Therefore, the DPLL WTANNs consisted of 8 or 16 neurons each of which included three DPLLs to process three dimensional vectors. The bit length of each counter in the DCO was set to, $K = 10$ and $L = 3$, thus each weight vector element had 13-bit accuracy, and 15-bit register was used in the neurons. The purpose of this paper is to provide a scalable WTA architecture in which the number of neurons can be easily increased. In terms of the WTA mechanism, an additional neuron can be included by simply connecting its input and stop signal ports to the input and stop signal lines in the WTANN, while other neurons are left intact. Using this feature, the experimental WTANN was easily scaled up so that the number of neurons was increased from 8 to 16 by simply connecting 8 additional neurons to the WTANN as described above.

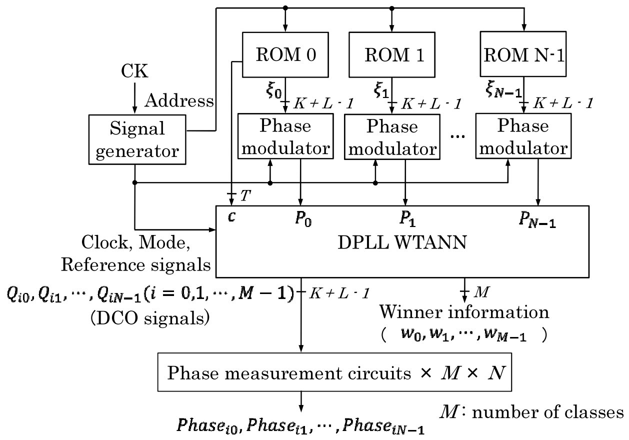
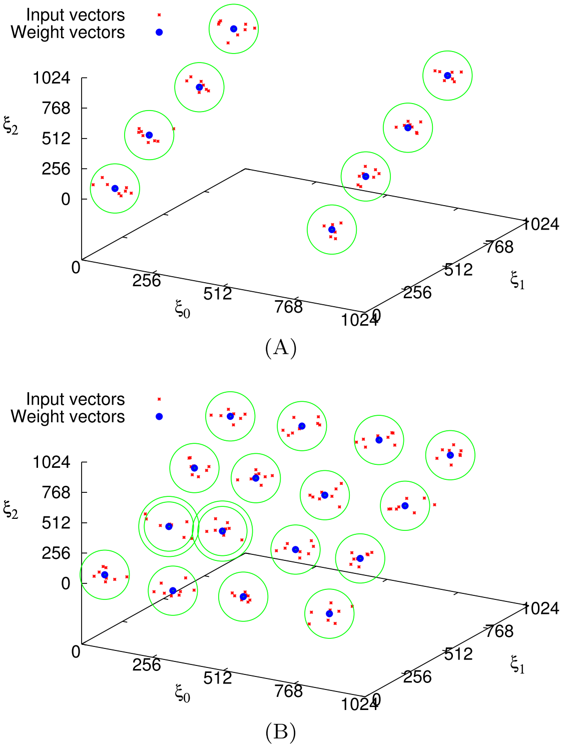


Fig. 11 Configuration for WTANN simulation



(A)

(B)

Fig. 12 Training and weight vector after training. (A) 8 classes and (B) 16 classes.

Each neuron performed supervised learning, using training signals with class information. Each test class data consists of 16 samples, from which 8 samples were randomly selected for learning and remaining samples were used for testing the system. The learning rate is set to $\alpha = 1/64$, and the number of training iteration was set to 48. Example of the training data and trained weight vectors are shown in Figs. 12 (A) and (B). The weight vectors

Table 1 Recognition rate V.S. the number of class

The number of class	8	16
Recognition rate	100%	99.8%

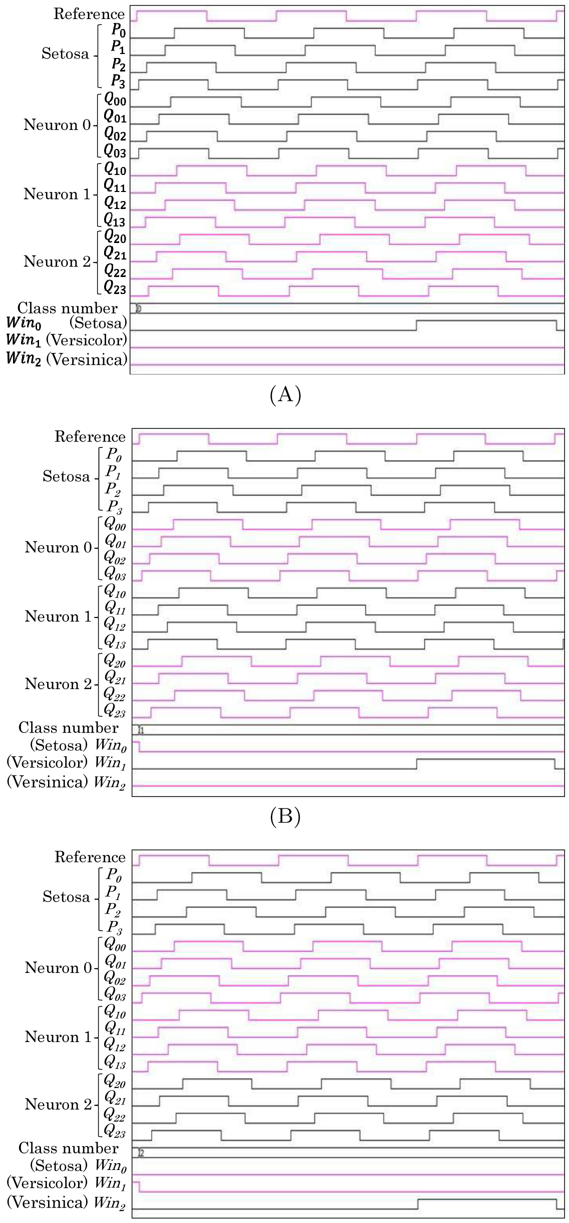
learned by training vectors are used for recognition of testing data. Table 1 shows the average recognition rate given by 100 recognition tests. It also shows the DPLL WTANNs are capable of classifying the test data set accurately. However, in the 16-class case, the recognition rate was not 100%. Some of the clusters marked by double-circles in Fig. 12 (B) were closely located to each other. The weight vectors assigned to these clusters were not always precisely trained to be at the center of the clusters because the training results depended on the training vectors. Therefore, vectors near the cluster border were not correctly classified.

4.3 Vector Classifications

The second simulation tested WTANN with Iris data and Wine data sets in the Machine Learning Repository (MLR) [16] of the University California, Irvine (UCI). The Iris data set consisted of three classes (Iris Setosa, Iris Versicolor, and Iris Versinica) with 50 instances of each. The instances were sepal length, sepal width, petal length, and petal width. Thus, there were four dimensions of vectors representing the Iris data. The Wine data set contained the results from chemical analysis of wines grown in the same region in Italy but were derived from three different cultivars. The Wine data set contained 178 instances belonging to three classes. The number of instances of each class was 59, 71, and 48. Analysis determined the quantities of 13 constituents found in each of the three types of wines. The attributes were alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, the OD280/OD315 of diluted wines, and proline. Thus, the vector representing the Wine data had 13 dimensions.

The DPLL WTANN in the simulated classification of Iris data, consisted of three neurons, each of which included four DPLLs to process four dimensional vectors. The bit length of each counter in the DCO was set to, $K = 9$ and $L = 3$, therefore, each weight vector element had 12-bit accuracy, and 15-bit register was used in the neurons. The DPLL WTANN in the simulated classification of Wine data consisted of three neurons each of which included 13 DPLLs so that 13 dimensional vectors were processed. The bit length of each counter in the DCO was set to, $K = 10$ and $L = 3$, thus each weight vector element had 13-bit accuracy, and 16-bit register was used in the neurons.

There are examples of the signals found in the simulations for Iris data in Figs. 13 (A), (B), and (C). The signals in the system correspond to Iris Setosa, Iris Versicolor, and Iris Versinica. $P_0 - P_3$ are the input signals of WTANN. Q_{0j}, Q_{1j}, and Q_{2j} are the signals from neuron-i, and $Win_0 - Win_2$ indicate the winner neuron. These figures show that neuron 0 is assigned to Iris Setosa, neuron 1 is assigned to Iris Versinica, and neuron 2 is assigned to Iris Versicolor. The black lines in the figures indicate input signals and signals in the winner neurons. Note that two sets of signals are very close in their phases.



(A) (B) (C)

Fig. 13 Timing chart for recognizing simulation, where input vector are (A) Iris Setosa, (B) Iris Versicolor and (C) Iris Versinica

Each set of Iris class data consisted of 50 samples, from which 25 samples were randomly selected for learning and the remaining samples were used for testing the system. The numbers of samples belonging to the three Wine classes were different, therefore, 50% of the samples were randomly selected for training and the remaining 50% were used for the recognition test. The learning rate was $\alpha = 1/64$, and the number of training iterations was set to 32.

Tables 2 and 3 summarize the recognition rates for

Table 2 Iris data set recognition

System	Class	Recognition rate
DPLL WTANN (proposed method)	Setosa	100%
	Versicolor	86.2%
	Versinica	93.4%
	Average	93.4%
Numerical computation with binary-tree search	Setosa	99.8%
	Versicolor	90.1%
	Versinica	89.2%
	Average	93.0%
k-NN [17]	Average	96.7%
LDA [17]	Average	98.0%

Table 3 Wine data set recognition

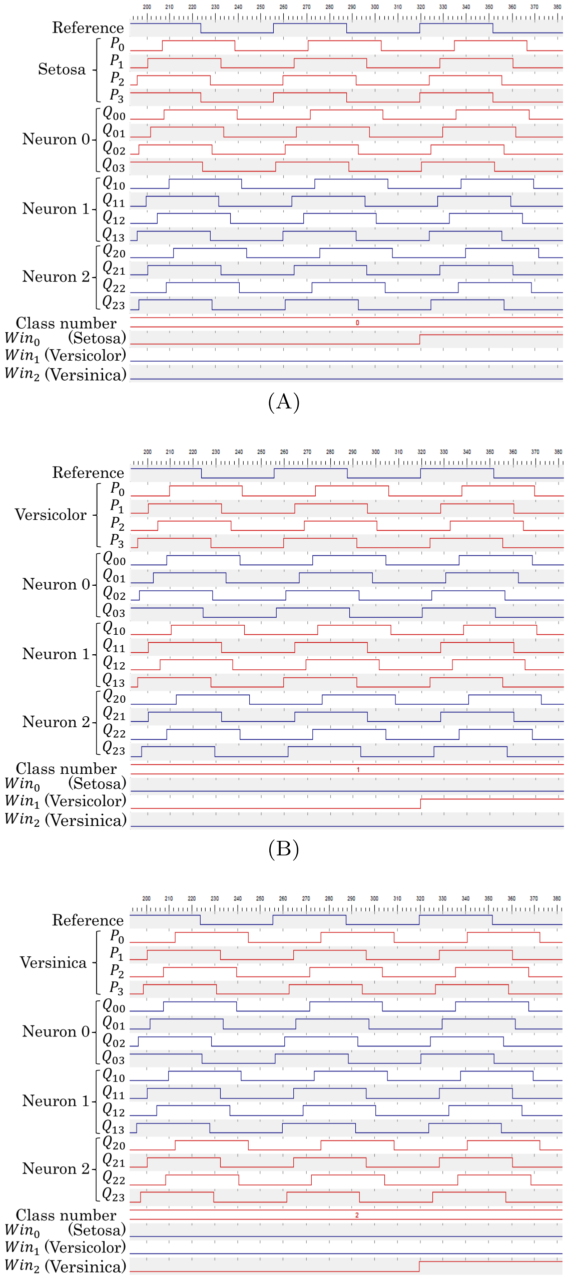
System	Class	Recognition rate
DPLL WTANN (proposed method)	1	95.7%
	2	96.6%
	3	98.6%
	Average	97.0%
Numerical computation with binary-tree search	1	97.2%
	2	97.2%
	3	99.0%
	Average	97.8%
k-NN [17]	Average	97.8%
LDA [17]	Average	98.9%

the system given by 100 recognition test. The tables also include the recognition accuracy of conventional WTANN based on binary-tree search, k-NN, and the linear discriminant analysis (LDA) classification algorithm [17]. Table 2 indicates that the system is capable of classifying Iris Setosa very accurately, but the recognition rates for Versicolor and Versinica are worse than those for Setosa. Table 3 indicates that the system is also capable of classifying the Wine data set very accurately.

5. Implementation and Experiments

The circuit size and speed were evaluated by using a logic synthesis tool. Xilinx ISE 14.6 was used for the logic synthesis, and the DPLL WTANN for the Iris data and Wine data were implemented on a Spartan-6 FPGA, XC6SLX16, which includes 2,278 slices. Moreover, conventional WTANN based on binary-tree search was implemented on the same FPGA.

The WTANN used for Iris data recognition consisted of three neurons, each of which contained four DPLLs. This WTANN consumed 583 slices in the FPGA. The highest operable clock frequency was 293.5 MHz. The 293.5 MHz clock and DCO configured with $K = 9$ and $L = 3$ generated a 143.3 kHz signal. Therefore, each input vector was recognized at a frequency of 143.3 kHz or every 6.98μs. There are examples of signals provided by the implemented circuit for Iris data in Figs. 14 (A), (B), and (C). These figures indicate that neuron 0 is assigned to Iris Setosa, neuron 1 is assigned to Iris Versinica, and neuron 2 is assigned to Iris Versicolor. The red lines in the figures indicate input signals and signals in the winner neurons. Note that two sets of signals are very close in their phases.



(A)

(B)

(C)

Fig. 14 Experimental results, where input vector are (A) Iris Setosa, (B) Iris Versicolor, and (C) Iris Versinica.

The WTANN for Wine data recognition consisted of three neurons, and each neuron contained 13 DPLLs. This WTANN consumed 1,111 slices of FPGA. The highest operable clock frequency was 283.2 MHz. The 283.2 MHz clock and DCO configured with $K = 10$ and $L = 3$ generated a 69.1 kHz signal. Therefore, each input vector was recognized at a frequency of 69.1 kHz or every 14.46μs.

Circuit size and speed are summarized in Table 4. The proposed WTAN has no advantage in terms of circuit size and speed. Improving the circuit size and speed of the

Table 4 Circuit size and speed

System	Circuit structure	Circuit size	Recognition speed
DPLL WTANN (proposed method)	4 dimensions, 3 classes (Iris)	367 slices	143.3×10^3 samples/sec.
	13 dimensions, 3 classes (Wine)	1111 slices	69.1×10^3 samples/sec.
	4 dimensions, 16 classes	1814 slices	66.6×10^3 samples/sec.
Numerical computation with binary-tree search	4 dimensions, 3 classes (Iris)	222 slices	74.2×10^6 samples/sec.
	13 dimensions, 3 classes (Wine)	837 slices	69.6×10^6 samples/sec.
	4 dimensions, 16 classes	1213 slices	39.3×10^6 samples/sec.

WTAN is left for future research.

6. Conclusion

We proposed a hardware WTANN that used DPLL and a new WTA circuit. The vector information was provided by a phase modulated signal, and the system used DPLL as a computing element since its operation was very similar to WTANN neurons. Information given to the system was expressed by the phase of a rectangular wave.

The most important function of the system is WTA operation, which was simplified by the proposed winner search circuit. The winner search circuit in the proposed system was distributed among neurons. Since winner search was carried out locally by all neurons, no global communication channel was needed. Therefore, the number of neurons could easily be increased without redesigning the winner search circuit.

The system used DPLL as a computing element. The WTANN was designed with VHDL, and its feasibility was tested and verified through VHDL simulations. The vector classification simulations and experiments demonstrated the proposed WTANN had valid learning characteristic and classification capabilities with the new WTA circuit. Two kinds of data sets, i.e., Iris data and Wine data sets, were used in these tests.

We proved that our design can be synthesized through logic synthesis. The hardware cost was evaluated with a logic synthesis tool. Experiments with the system implemented on an FPGA verified the efficiency of its physical design and operational speed, but we found that the proposed WTANN architecture needs further improvement to decrease its circuit size while increasing the operating speed.

References

[1] J.P.F. Sum, C.-S. Leung, P.K.S. Tam, G.H. Young, W.K. Kan, and L.-W. Chan, "Analysis for a Class of Winner-Take-All Model," IEEE Trans. Neural Netw., vol.10, no.1, pp.64–71, 1999.
[2] Y.C. Chang and C.J. Kuo, "Automatic edge detection," Proc. IEEE Int'l Conf. on ISPACS, pp.173–176, Nov. 2002.
[3] R.P. Lippmann, "An Introduction Computing with Neural Nets," IEEE Trans. Acoust., Speech, Signal Process. Mag., vol.4, no.2, pp.4–22, April 1987.
[4] T. Kohonen, Self-Organizing Maps. 3rd, ser. Springer Series in Information Sciences 30, Springer-Verlag, New York, 2001.
[5] H. Hikawa, "Frequency-Based Multilayer Neural Network with On-Chip Learning and Enhanced Neuron Charcteristics," IEEE Trans. Neural Netw., vol.10, no.3, pp.545–553, May 1999.
[6] L.M. Reyneri, "A performance analysis of pulse stream neural and fuzzy computing systems," IEEE Trans. on CAS, vol.42, no.10, pp.642–660, Oct. 1995.
[7] A.L. Hodgkin and A.F. Huxley, "A quantitative description of ion currents and its applications to conduction and excitation in nerve," Journal of Physiology, vol.117, no.4, pp.500–544, 1952.
[8] W. Gerstner and W.M. Kistler, Spiking Neuron Models: Single Neurons, Populations, Plasticity, Cambridge University Press, 2002.
[9] A. Gupta and L.N. Long, "Hebbian Learning with Winner Take All for Spiking Neural Networks," Proc. of International Joint Conference on Neural Networks, pp.1054–1060, June 14-19, 2009.
[10] D.T. Pham, M.S. Packianather, and E.Y.A. Charles, "A self-organising spiking neural network trained using delay adaptation," Proc. IEEE International Symposium on Industrial Electronics, 2007 (ISIE 2007), pp.3441–3446, 2007.
[11] B. Ruf and M. Schmitt, "Self-Organization of Spiking Neurons Using Action Potential Timing," Proc. IEEE Trans. on Neural Netowrks, vol.9, no.3, pp.575–578, May 1998.
[12] H. Hikawa, "FPGA implementation of self-organizing map with digital phase locked loops," Neural Networks, vol.18, no.5-6, pp.514–522, 2005.
[13] B. Mailachalam and T. Srikanthan, "Area-time issues in the VLSI implementation of self organizing map neural networks," Microprocessors and Microsystems, vol.26, no.9-10, pp.399–406, Dec. 2002.
[14] D.C. Hendry, "Comparator trees for winner-take-all circuits," Neurocomputing, vol.62, pp.389–403, 2004.
[15] H. Yamamoto and S. Mori, "Performance of a Binary Quantized All Digital Phase-Locked Loop with a New Class of Sequential Fitler," IEEE Trans. on Commun., vol.26, no.1, pp.35–45, Jan. 1978.
[16] University of California at lrvine web site, http://www.ics.uci.edu/mlearn/MLRepository.html
[17] T. Yang and V. Kecman, "Adaptive local hyperplane classification," Neurocomputing, vol.71, no.13-15, pp.3001–3004, 2008.

Masaki Azuma received the B.E. and M.E. degrees in electrical engineering from Kansai University, Japan, in 2013 and 2015, respectively. He joined Aisin Seiki Co., Ltd. in 2015.

Hiroomi Hikawa received the B.E., M.E., and Ph.D degrees in electrical engineering from Keio University, Japan, in 1984, 1986, and 1989, respectively. He then served as a post-doctoral researcher at University of South Florida. From 1992 to 2008, he worked at the Computer Science and Intelligent Systems Department of Oita University. In 2008, he joined Department of Electrical and Electronic Engineering of Kansai University where he is a Professor. His research interests includes architecture for signal processing and neural networks.