Continuous-Valued Probabilistic Behavior in a VLSI Generative Model

Hsin Chen, *Member, IEEE*, Patrice C. D. Fleury, and Alan F. Murray, *Senior Member, IEEE*

***Abstract*—This paper presents the VLSI implementation of the continuous restricted Boltzmann machine (CRBM), a probabilistic generative model that is able to model continuous-valued data with a simple and hardware-amenable training algorithm. The full CRBM system consists of stochastic neurons whose continuous-valued probabilistic behavior is mediated by injected noise. Integrating on-chip training circuits, the full CRBM system provides a platform for exploring computation with continuous-valued probabilistic behavior in VLSI. The VLSI CRBM's ability both to model and to regenerate continuous-valued data distributions is examined and limitations on its performance are highlighted and discussed.**

***Index Terms*—Boltzmann machine, continuous-valued probabilistic VLSI, noise, on-chip training, probabilistic generative model, stochastic computation.**

I. Introduction

As INTEREST in implantable instruments [1]–[4] and bioelectrical systems [5]–[7] grows, exposing electronic circuits and sensors to noisy environments becomes unavoidable. Although many signal-processing techniques are available for dealing with noisy and drifting signals in a digital computer, transmitting *all* measured raw data out of an implantable device is power-consuming. An optimized "intelligent" embedded system would extract useful information from signals at sensory or bioelectrical interfaces, to reduce noise and interference and save power by transmitting only useful information. In addition, although implantable instruments can be miniaturised by reducing transistor sizes, intrinsic electronic noise is expected to grow dramatically as transistor size shrinks toward the deep-submicron scale [8]–[10]. In the longer term, the ability to perform useful computation in the presence of noise will be essential to an intelligent embedded system.

Probabilistic models use stochastic computation to represent the natural variability of real data, and thus point toward a potential approach to an intelligent embedded system. The probabilistic input-output relationship embedded in each computing node (neuron) of a probabilistic model further makes VLSI implementation of probabilistic models attractive. Computing elements with probabilistic input-output relationship could ameliorate the effects of noise in VLSI, by effectively making use of the noise as an essential element of computation. Signals derived from the real world are generally continuous-valued, so probabilistic models that are able to model continuous-valued data are of great interest. Bayesian rules [11]–[14] and Kernel Machines [15], [16] both of which rely upon probability, have been demonstrated in VLSI. However, the VLSI implementation of these models require precise calculations of conditional probabilities or vector products. As precise calculations are vulnerable to VLSI process variations and noise, a VLSI implementation with explicit probabilistic behavior would be preferable for an intelligent embedded system. The probabilistic VLSI models reported in [17]–[21] support this suggestion by exploiting the *stochastic-arithmetic* algorithm [22]–[24], which encodes and transmits continuous values as stochastic pulse streams, such that reliable computation and long transmitting distance of continuous values are achieved. The stochastic-pulse scheme, however, is based upon a fully digital VLSI implementation, in order to simplify circuit design and to increase immunity to noise caused by distributed digital pulses. As fully digital implementation is power- and area-consuming, VLSI models based on the stochastic-pulse scheme are unlikely to be best suited to an intelligent embedded system.

This paper presents the VLSI implementation of the continuous restricted Boltzmann machine (CRBM), a probabilistic generative model capable of modeling continuous-valued data with a simple and hardware-amenable training algorithm [25], [26]. The CRBM consists of continuous stochastic neurons whose stochasticity arises from the addition of Gaussian noise to neural input activities. Noise has been used to introduce uncertainty in many applications [27], [28], and [29] has already demonstrated isolated CRBM neurons in VLSI. The Boltzmann machine in VLSI reported in [27] and [30] utilizes noise to introduce the binary stochasticity required by the Boltzmann Machine [31]. CRBM neurons in VLSI, however, preserve the continuous-valued nature of noise at the outputs of neurons to enhance computation. The "Contrastive-Divergence" [32] training algorithm of the CRBM has also been demonstrated in VLSI in [29]. This paper describes a complete (small) CRBM system implemented in VLSI, with on-chip adaptability. The continuous-valued nature of the CRBM means that even a small CRBM system has the potential to model nontrivial data distributions. The full CRBM system provides a platform for exploring both the utility and on-chip adaptability of noise-induced, continuous-valued probabilistic behavior in VLSI. As a generative model, the training procedure for the CRBM aims toward an ability to "regenerate" data with the same distribution as that of its training data. Therefore, on-chip adaptability is included, to test the CRBM system's ability to adapt its parameters in real time. We explore the VLSI CRBM system's ability

Manuscript received April 12, 2005; revised September 9, 2005.

H. Chen is with the Department of Electrical Engineering, National Tsing-Hua University, Hsin-Chu 30055, Taiwan (e-mail: hchen@ee.nthu.edu.tw).

P. C. D. Fleury and A. F. Murray are with The School of Engineering and Electronics, Edinburgh University, Edinburgh EH9 3JL, U.K.

Digital Object Identifier 10.1109/TNN.2006.873278

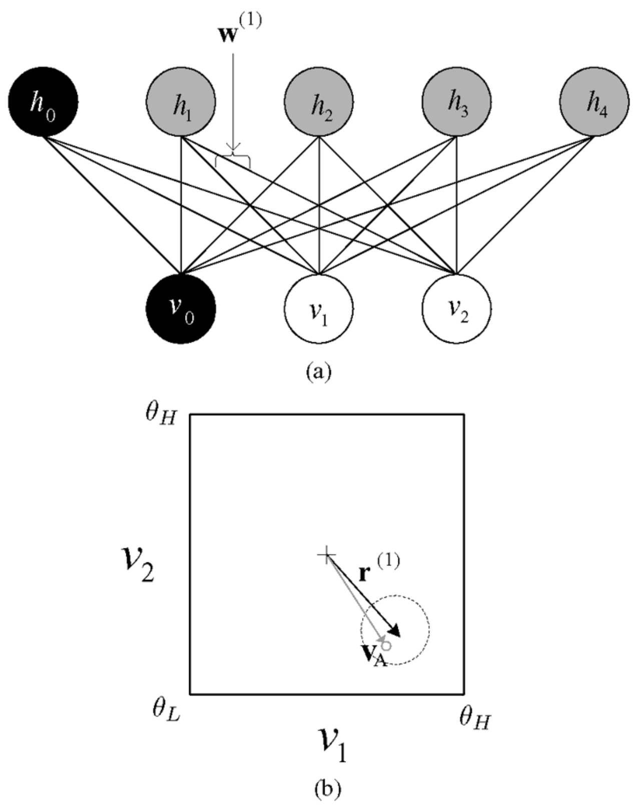


Fig. 1. (a) CRBM with two visible and four hidden neurons. v_0 and h_0 are bias units. (b) Projection of the weight vector of hidden neuron h1, $\mathbf{w}^{(1)}$, in the visible-state space, where $\mathbf{r}^{(1)} = \varphi(\mathbf{w}^{(1)})$. Activation of h1 encourages the CRBM to regenerate visible states around the region circled by the dotted line.

to regenerate several low-dimensional, continuous-valued data distributions. The results of these experiments aim to demonstrate the feasibility of computation with continuous-valued probabilistic behavior in VLSI, and the potential of a CRBM system as an intelligent embedded system.

II. The CRBM

The CRBM consists of continuous stochastic neurons with restricted inter neural connections, as shown in Fig. 1. Each circle represents one neuron and each line a bidirectional and symmetric $(w_{ij} = w_{ji})$ connection between neurons. Let s_i denote the state of a neuron [either v_i or h_j in Fig. 1(a)], and w_{ij} the connection between neuron i and neuron j. The stochastic behavior of a neuron i is described by the following equations:

$$s_i = \varphi_i\left(\sum_j w_{ij}s_j + \sigma \cdot N_i(0,1)\right) \tag{1}$$

with

$$\varphi_i(x_i) = \theta_L + (\theta_H - \theta_L) \cdot \frac{1}{1+\exp(-a_i x_i)} \tag{2}$$

where $N_i(0,1)$ represents a unit-valued, zero-mean Gaussian noise, σ is a noise-scaling constant, and $\varphi_i(\cdot)$ a sigmoid function with asymptotes at θ_H and θ_L. The presence of Gaussian noise renders s_i both probabilistic and continuous-valued. Given inputs from other neurons, $\{s_j\}$, a value of s_i can be "Gibbs sampled" from (1). Parameter a_i controls the slope of the sigmoidal nonlinearity and, thus, the effective influence of noise on the probability density of s_i [25], [26]. a_i is adapted along with the CRBM's weights. This introduces a representational flexibility that endows the CRBM with the ability to model different forms of data variability in a single model [26]. Finally, the black circles in Fig. 1 represent bias units whose outputs are permanently "on" (i.e., $v_0 = h_0 = \theta_H$).

Neurons in a CRBM model are divided into two layers, one *visible* and one *hidden*, as shown in Fig. 1(a). Given the states of neurons in one layer, the states of neurons in the other layer are conditionally independent and can be sampled simultaneously. As a *generative* model, the CRBM aims to "regenerate" training data distributions in its visible neurons. Hidden neurons then function as individual "experts" that cooperate to "explain" the data modeled by the CRBM. For example, hidden neuron h_1 in Fig. 1(a) is an expert characterized by its connections to all visible neurons, $\mathbf{w}^{(1)}$. Turning h_1 $(h_1 \simeq \theta_H)$ on causes the CRBM to tend to regenerate visible states whose state vectors $\mathbf{v} = \{v_i\}$ align with $\mathbf{w}^{(1)}$, as illustrated by Fig. 1(b). Conversely, when a visible state aligning with $\mathbf{w}^{(1)}$[e.g., $\mathbf{v}_A$ in Fig. 1(b)] is applied, h_1 has a high probability of activation. This feature is especially useful for tasks like pattern recognition, in which responses of hidden neurons to a testing datum indicate the category of the datum, as demonstrated in [26] and [33].

Training rules for the CRBM are derived by minimizing the contrastive divergence (MCD) [32] between training data and one-step Gibbs-sampled data [26], as illustrated in Fig. 2. Each training datum $\mathbf{d}_i$ is first presented to visible neurons as initial visible state, $\{v_i\} = \mathbf{d}_i$. Hidden neurons are then "Gibbs sampled" to obtain corresponding hidden state, $\mathbf{h} = \{h_j\}$. Repeating this process once more to sample visible and hidden neurons then produces one-step Gibbs-sampled states $\hat{\mathbf{v}} = \{\hat{v_i}\}$ and $\hat{\mathbf{h}} = \{\hat{h_j}\}$. Let η_w and η_a denote the training rates for parameters $\{w_{ij}\}$ and $\{a_i\}$, respectively. Parameters $\{w_{ij}\}$ and $\{a_i\}$ can be updated according to [26]

$$\Delta\widehat{w_{ij}} = \eta_w(\langle v_i h_j\rangle - \langle \hat{v_i}\hat{h_j}\rangle) \tag{3}$$

$$\Delta\hat{a_i} = \frac{\eta_a}{a_i^2}\left(\left\langle s_i^2\right\rangle - \left\langle \hat{s_i}^2\right\rangle\right) \tag{4}$$

where s_i represents both v_i for visible neurons and h_j for hidden neurons, and $\langle\cdot\rangle$ refers to the expectation over all training data. Although the simplicity of the rules in (3) and (4) makes them intrinsically hardware-amenable, they can be further simplified as follows [29]:

$$\Delta\widehat{w_{ij}} = \eta_w \mathrm{sign}(\langle v_i h_j\rangle_4 - \langle \hat{v_i}\hat{h_j}\rangle_4) \tag{5}$$

$$\Delta\hat{a_i} = \eta_a \mathrm{sign}\left(\left\langle s_i^2\right\rangle_4 - \left\langle \hat{s_i}^2\right\rangle_4\right) \tag{6}$$

where $\langle\cdot\rangle_4$ denotes averaging over only *four* training data, and the simplified rules take only the signs of contrastive divergences to update parameters in the correct direction, with fixed stepsizes of η_w and η_a. It has been shown in [29], [34] that this increases training time, but has little or no effect on modeling ability. The simulations in [29], for example, showed that the required training time increased from 4000 to 30 000 epochs when the CRBM was trained to model a simple but nontrivial dataset with (5) and (6). Section III will show that each training epoch

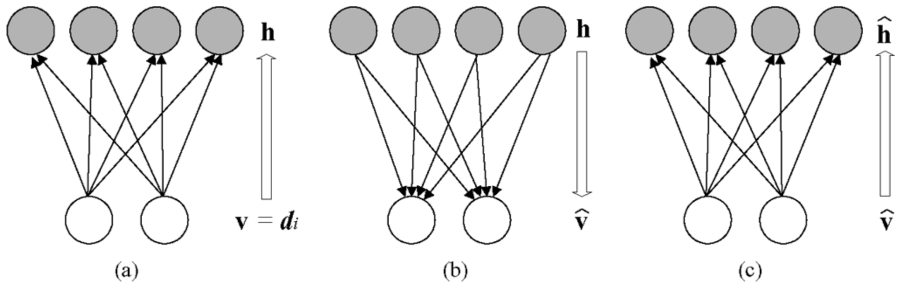


Fig. 2. Training the CRBM by one-step Gibbs sampling. (a) Set initial visible state as a training datum $\mathbf{d}_i$, sample $\mathbf{h}$. (b) Given $\mathbf{h}$, sample a one-step reconstructed visible state $\hat{\mathbf{v}}$. (c) Given $\hat{\mathbf{v}}$, sample a one-step hidden state $\hat{\mathbf{h}}$.

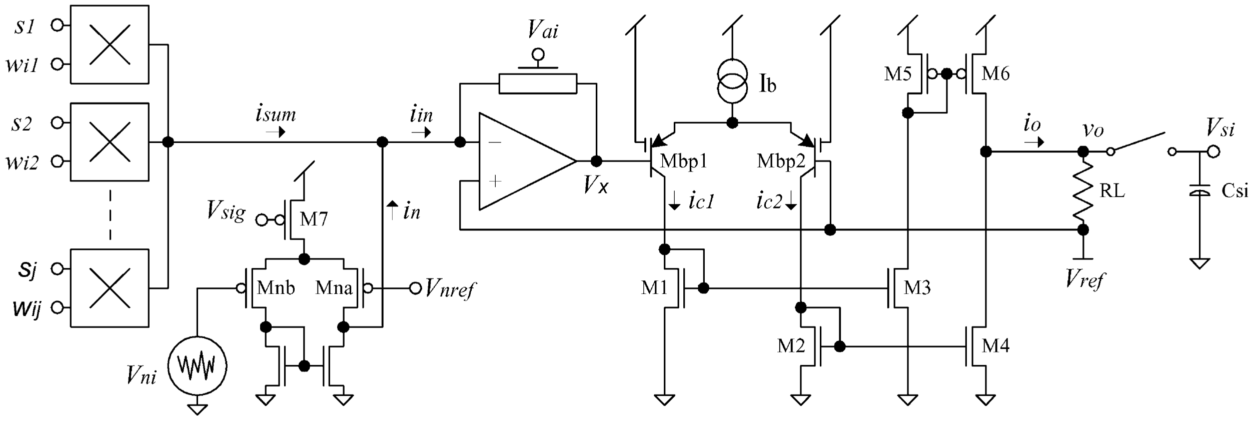


Fig. 3. Continuous stochastic neuron in CMOS VLSI.

requires 24 clock cycles. As the CRBM system is designed to operate at a frequency of 100 kHz, 30 000 epochs correspond to a computing time of merely 7.2 s. The slower convergence time is thus a small price to pay for simpler circuits and an economic chip size.

The performance of a trained CRBM model can be evaluated by generating an "approximate equilibrium reconstruction" through multistep Gibbs sampling [26]. In a multistep Gibbs-sampling process, visible states are initialized to random values. Hidden neurons and visible neurons are then sampled alternatively, as in Fig. 2, for many times. Let the sampled visible states at Nth iteration be called *N-step reconstructed visible states*. If N is large, the distribution of N-step reconstructed visible states approximate the equilibrium distribution of visible states, and the similarity between the N-step reconstructed data and the training data indicates how well the CRBM models the training data.

III. A Full CRBM System

A CRBM model with two visible and four hidden neurons [Fig. 1(a)] has been fabricated on the AMS 0.6-μm 2P3M CMOS process. The following subsections describe the VLSI implementation of the full CRBM system, including continuous stochastic neurons, on-chip MCD training circuits, and the architecture of the full system.

A. Continuous Stochastic Neurons in VLSI

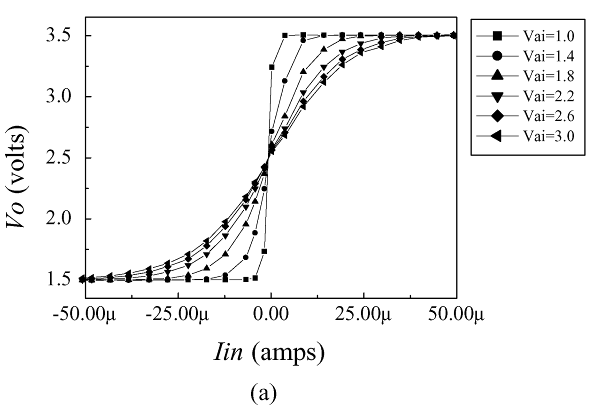
Fig. 3 shows the circuit diagram of a continuous stochastic neuron for the CRBM. The four-quadrant multipliers output a total current proportional to $i_{\text{sum}} = \sum_j w_{ij}s_j$ [29], while the differential pair, Mna and Mnb, transforms noise voltage V_{ni} into a noise current $i_n = g_m(V_{ni} - V_{nr})$. V_{sig} controls the transconductance g_m through $M7$ and thus scales the noise current (σ in (1)). As a result, i_{in} represents the presigmoid input $(\sum_j w_{ij}s_j + \sigma \cdot N_i(0,1))$ in (1). After conversion of i_{in} into V_x, the sigmoid-function circuit basing on transistors(Mbp1-2) in lateral-bipolar operation [35] produces an output current of [29]

$$i_o = i_{c1} - i_{c2} = I_b \cdot \phi\left(\frac{i_{in} \cdot R(V_{ai})}{V_t}\right) \tag{7}$$

where $\phi(\cdot)$ denotes the sigmoid function $\varphi_i(\cdot)$ with $\theta_H = -\theta_L = 1$, and $V_t = kT/q$ is the thermal voltage. Finally, the resistor R_L converts i_o into a voltage v_o, representing s_i in (1), and the switch is used to Gibbs sample a continuous-valued output state V_{si}.

Equation (7) implies that V_{ai} controls the feedback resistance of the *I-V* converter and thus adapts the slope of the sigmoidal nonlinearity [a_i in (1)]. The measured dc characteristics of the sigmoid function, corresponding to various V_{ai}, are shown in Fig. 4(a). A mapping between V_{ai} and the arithmetical value for $\{a_i\}$ in simulation, as shown in Fig. 4(b), is also derived to facilitate cross verification between a simulated CRBM model and a CRBM system. The derivation method is described in the Appendix.

Fig. 5(a) shows the measured output of a continuous-stochastic neuron with inputs $\{s_j\}$ connected to a triangular wave sweeping between 1.5 to 3.5 V, $\{w_{ij}\} = 4$ V, and $V_{ai} = 1.8$ V. These values of $\{s_j\}$ and $\{w_{ij}\}$ correspond to $\{s_j\} = \pm 1$ and $\{w_{ij}\} = 1.5$ numerically (Table I), forcing the neuron's



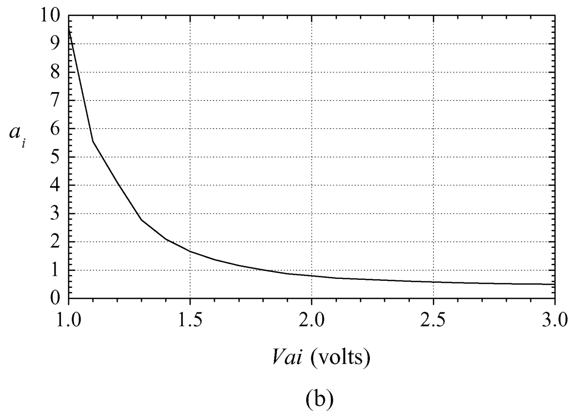
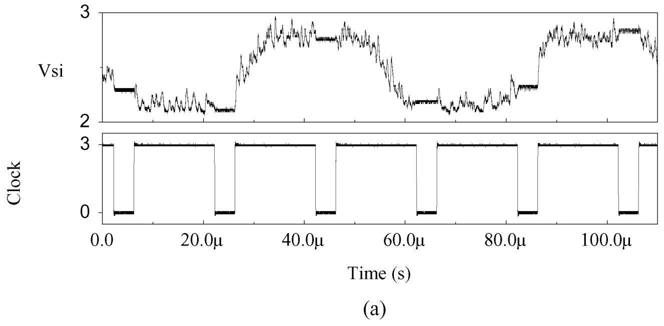


Fig. 4. (a) Measured dc characteristics of a sigmoid-function circuit with variable nonlinearity. (b) Mapping for parameter $\{a_i\}$ between software simulation and hardware implementation.



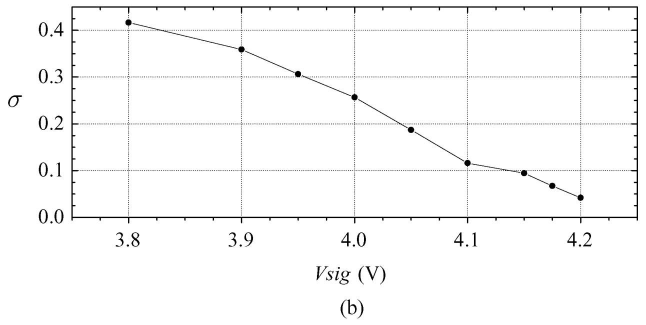


Fig. 5. (a) Measured output of a continuous stochastic neuron (upper trace) and the switching signal (lower trace) that samples $V_{\rm si}$. (b) Mapping for σ between software simulation and hardware implementation.

TABLE I
MAPPING OF PARAMETER VALUES BETWEEN SOFTWARE SIMULATION AND HARDWARE IMPLEMENTATION

	Matlab	VLSI (V)	Mapping ratio
s_i	$[-1.0, 1.0]$	$[1.5, 3.5]$	1:1
w_{ij}	$[-2.5, 2.5]$	$[0.0, 5.0]$	1:1
a_i	$[0.5, 9.0]$	$[1.0, 3.0]$	Fig.4(b)

output to sweep much of the range of the sigmoid curve, while the noise injection clearly gives the curve a (continuous-valued) probabilistic output that can be sampled. This neuron state V_{si} is sampled periodically by the clock signal in Fig. 5(a) and held with negligible clockfeedthrough (i.e., when the switch in Fig. 3 is opened). This pleasing result demonstrates that the incorporation of noise is an effective and inexpensive way of introducing continuous-valued probabilistic behavior to analog VLSI. The near-linear mapping between $V_{\rm sig}$ and the value of σ in (1) is shown in Fig. 5(b), and the derivation of the mapping is described in the Appendix.

B. Minimizing-Contrastive-Divergence (MCD) Training

The similarity between (5) and (6) indicates that parameters $\{w_{ij}\}$ and $\{a_i\}$ can share the same MCD training circuit. Fig. 6(a) and (b) shows the block diagram of the MCD training circuit, along with its digital control signals [29], [36]. To ensure normal operation of voltage-controlled resistors in Fig. 3 [37], the training circuit for $\{a_i\}$ differs from that for $\{w_{ij}\}$ in the inclusion of a voltage-limiting circuit that constrains V_{ai} to the range $[1, 3]$ (V).

In learning mode $(\mathrm{LER}/\overline{\mathrm{REF}} = 1)$, the first two clocks, CKsi and CKsj, sample the initial states s_i and s_j, and the multiplier outputs a current I^+ proportional to $s_i \cdot s_j$. After CK+ samples and holds I^+, CKsip and CKsjp sample the one-step Gibbs-sampled states $\widehat{s_i}$ and $\widehat{s_j}$ to produce another current I^-. CKq then triggers the accumulator to sample and hold the output of the current subtractor $I_{\rm sub}$ representing the contrastive divergence $\langle v_i h_j\rangle - \langle\widehat{v_i}\widehat{h_j}\rangle$ between the initial datum and its one-step Gibbs sample. With the above sequence repeated for *four* cycles, four $I_{\rm sub}$ are accumulated and averaged to derive $I_{\rm ave}$, representing $\langle v_i h_j\rangle_4 - \langle\widehat{v_i}\widehat{h_j}\rangle_4$ in (5) or $\langle s_i^2\rangle_4 - \langle\widehat{s_i}^2\rangle_4$ in (6). Finally, the sign circuit compares $I_{\rm ave}$ to $I_{\rm ref}$ to determine the update direction DIR, and the learning circuit, triggered by CKup, updates the parameter once. In "refresh" mode $(\mathrm{LER}/\overline{\mathrm{REF}} = 0)$, the signal REFR rather than DIR determines the update direction, causing the learning circuit to maintain the parameter at its currently correct value.

The stepsize of each update is controlled by the voltage V_{mu}, which sets the pulsewidth controlling the period of charging or discharging C_w when CKup is high [29], [36]. V_{mu}, therefore, controls the training rates η_w and η_a in (5) and (6). As shown in [36], the pulsewidth can vary from 5 ns to many seconds,

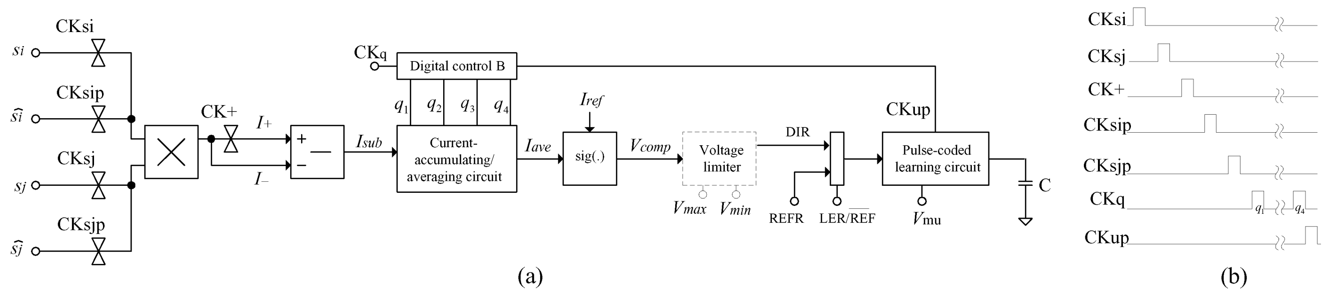


Fig. 6. (a) MCD training circuit (schematic) for both $\{w_{ij}\}$ and $\{a_i\}$. (b) Digital control signals.

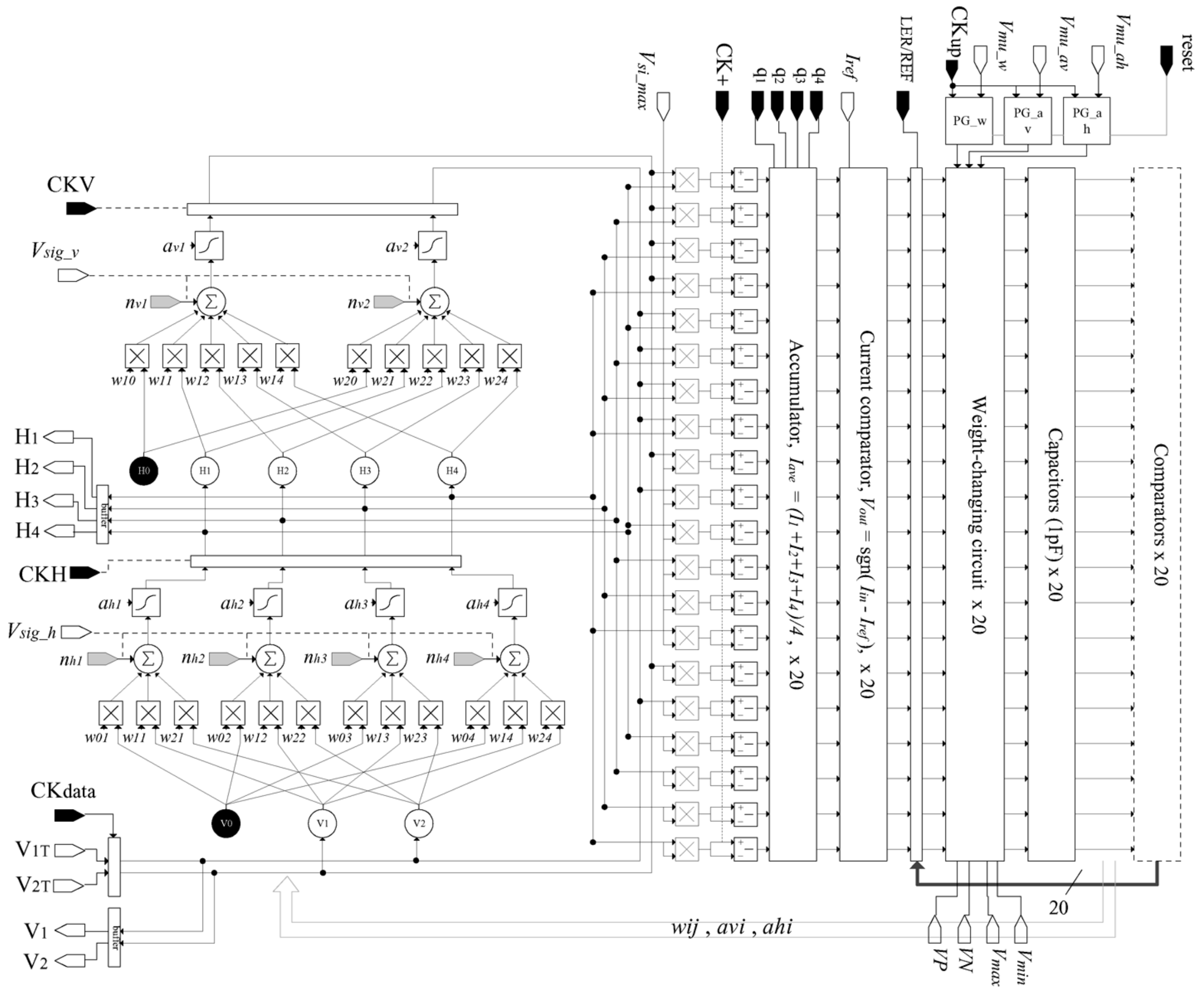


Fig. 7. Modules of the on-chip CRBM.

corresponding to several orders of magnitude for training rates (stepsizes in each update).

C. System Architecture

Combining the CRBM neurons in VLSI with the MCD training circuits, a full CRBM system with two visible and four hidden neurons was implemented. Fig. 7 shows the modular diagram of a full CRBM system. The black pins represent digital-control signals and the grey pins indicate noise inputs into the neurons. The white pins represent both the analog signals that connect the CRBM to its external, real-valued environment and the reference voltages and currents for the system. The 20 capacitors store the $14\{w_{ij}\}$ between neurons, and the $\{a_i\}$ of the six neurons. All of them are adapted by the 20 learning circuits simultaneously. The mapping for all parameters between software simulation and hardware implementation are summarized in Table I.

The digital-control signals in Fig. 6(b) are applicable to the full CRBM system. In learning mode $(\mathrm{LER}/\overline{\mathrm{REF}} = 1)$, simply set $\mathrm{CKdata} = \mathrm{CKsi}, \mathrm{CKV} = \mathrm{CKsip}$, and $\mathrm{CKH} = \mathrm{CKsj} + \mathrm{CKsjp}$. The CRBM system will sample an external training datum on clock CKdata, and will execute

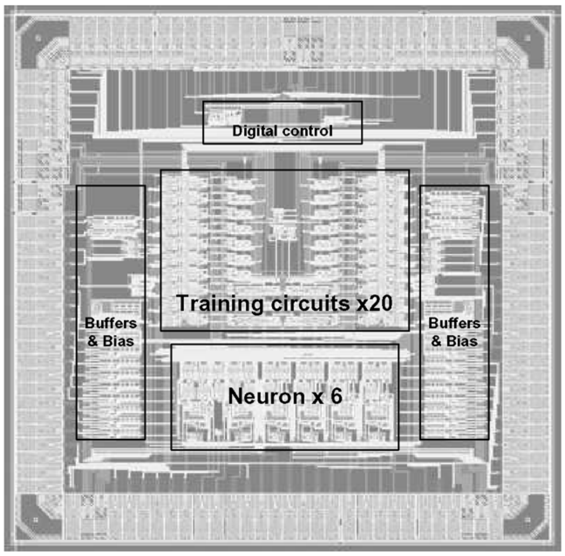


Fig. 8. Full CRBM system on silicon.

TABLE II
POWER CONSUMPTION AND AREA OF A FULL CRBM SYSTEM

	Power supply current (mA)	Circuit Area ($\mu m \times \mu m$)
Visible neuron	0.941	367×523
Hidden neuron	0.876	257×523
MCD training for $\{w_{ij}\}$	0.109	1128×145
MCD training for $\{a_i\}$	0.255	898×145
Analogue buffer	0.624	327×87
Full CRBM system	19.644	3197×2928

*Power supply currents are measured at $V_{dd} = 5V$.

one step of Gibbs sampling on clocks CKV and CKH. The following clock sequence (CK+, q1–q4, and CKup) then drives the learning circuits to update parameters once. In regenerating mode $(\mathrm{LER}/\overline{\mathrm{REF}} = 0)$, simply sets $\mathrm{CKV} = \mathrm{CKsi} + \mathrm{CKsip}$ and $\mathrm{CKH} = \mathrm{CKsj} + \mathrm{CKsjp}$. The CRBM will then execute multiple steps of Gibbs sampling, while the parameters of the CRBM are refreshed to reference values set by external voltage comparators. Note that the digital-control circuits for training a CRBM system, although not shown in Fig. 7, were also implemented on the same chip. Fig. 8 shows the chip layout of the full CRBM system described, with power consumption and layout areas summarized in Table II. Although the full system consumes a total current of 19.644 mA, most current consumption is attributable to the analog buffers that read out parameter values, as shown in Fig. 8. Excluding the 20 analog buffers, the CRBM system alone draws only 7.164 mA.

IV. MODELING CONTINUOUS-VALUED DATA DISTRIBUTIONS

To probe a CRBM system's ability to adapt its noise-induced probabilistic behavior, a CRBM system was trained on a variety of continuous-valued data distributions. Let $+\eta_\lambda$ and $-\eta_\lambda$ denote the decreasing and increasing stepsizes, respectively, for

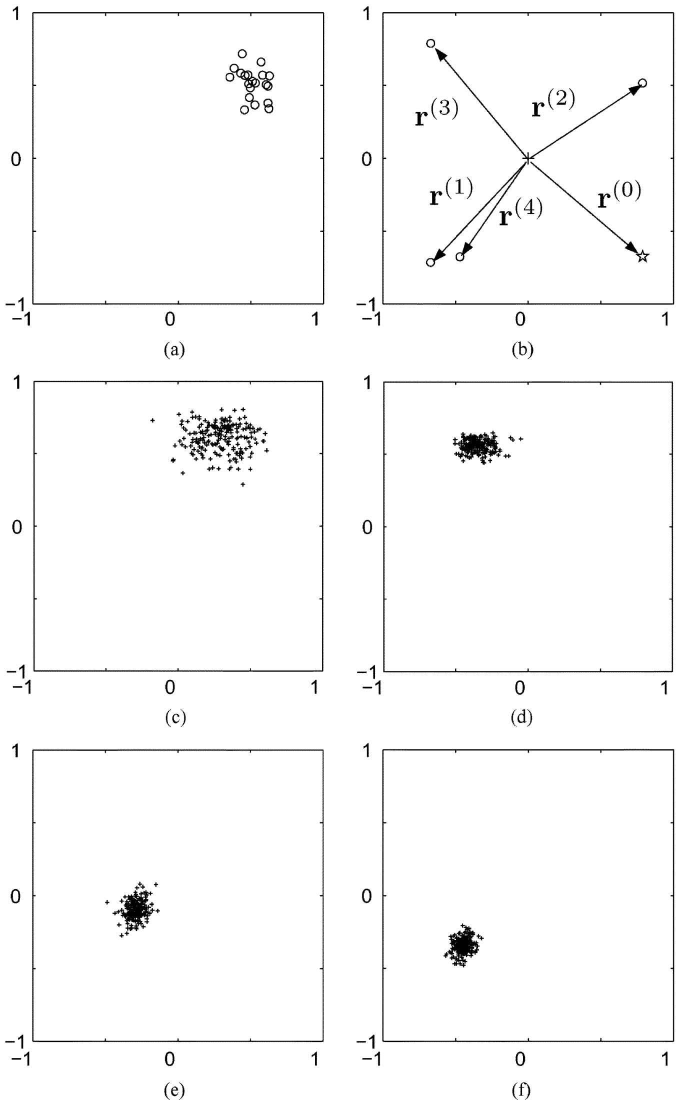


Fig. 9. (a) Twenty training data points sampled from a single Gaussian distribution. (b) Projections of the trained weight vectors of hidden neurons in data space, where $\mathbf{r}^{(i)} = \phi(\mathbf{w}^{(i)})$ and $\mathbf{w}^{(i)}$ represents the weight vector of hidden neuron i. (c)–(f) Twenty-step reconstructions generated by a CRBM model with parameters values learnt by a CRBM system after (c) 50; (d) 120; (e) 550; (f) 1000 training epochs.

a parameter λ. The training stepsizes for $\{w_{ij}\}$ and $\{a_i\}$ are $+\eta_{wij} = -\eta_{ahi} = +\eta_{avi} = 15$, $-\eta_{wij} = +\eta_{ahi} = 3$, and $-\eta_{avi} = -45$ mV. The noise scaling constants, $V_{\mathrm{sig\_v}}$ and $V_{\mathrm{sig\_h}}$, were set to be 4.05 V, corresponding to $\sigma = 0.2$ in simulation [Fig. 5(b)]. The following subsections present and discuss the results of training a CRBM system on continuous-valued data distributions.

A. Modeling a Single-Gaussian Distribution

A VLSI CRBM system was trained to model the single-Gaussian distribution shown in Fig. 9(a) for 1000 epochs(steps). Fig. 10 shows the adaptation of on-chip parameter values during training, and Fig. 9(b) shows the projections of all weight vectors learnt after 1000 epochs into the data space. To visualise the distributions modeled by the CRBM system, parameter values at training epochs 50, 120, 550,

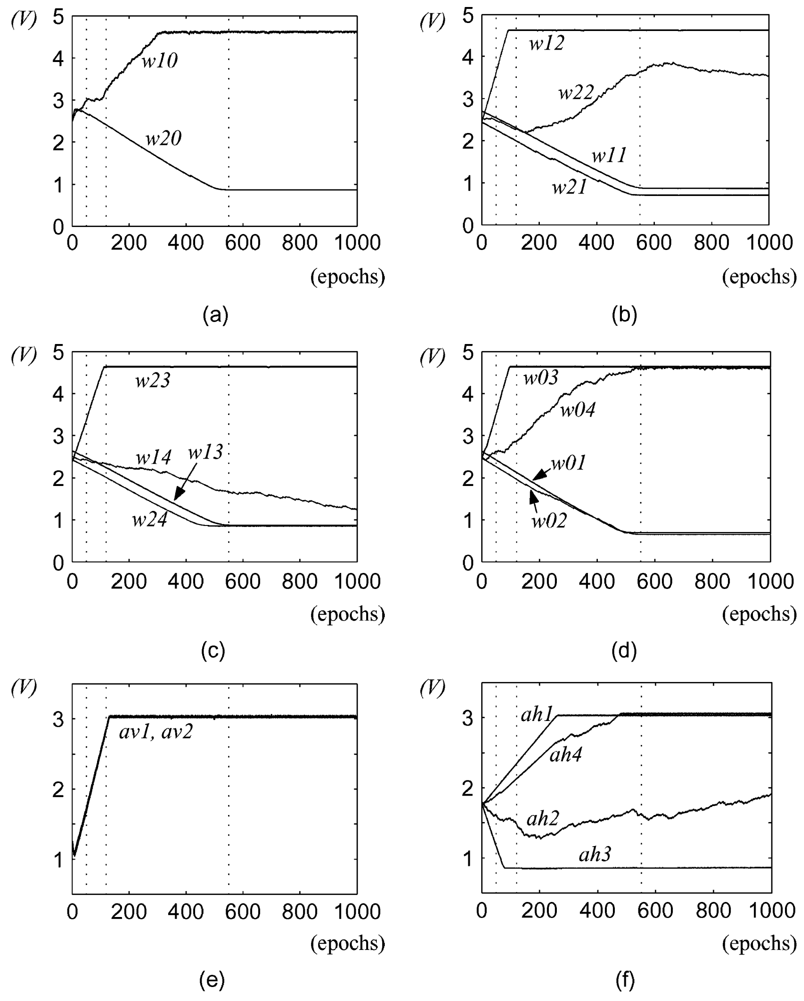


Fig. 10. Measured traces of $\{a_i\}$ and $\{w_{ij}\}$ when a CRBM system was trained with the data in Fig. 9(a) for 1000 epochs (a) $\mathbf{w}^{(0)}$; (b) $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$; (c) $\mathbf{w}^{(3)}$ and $\mathbf{w}^{(4)}$ (d) $\{w_{0j}\}$; (e) $\{a_{vi}\}$; (f) $\{a_{hi}\}$. The dotted lines highlight the parameter values used to generate the 20-step reconstructions in Fig. 9(c)–(f).

and 1000 were substituted into a CRBM model in Matlab to generate the 20-step reconstructions shown in Fig. 9(c)–(f).

The reconstruction in Fig. 9(c) indicates that the CRBM system modeled a single-Gaussian distribution roughly in the first 50 epochs. However, as the system continued to shape the variance in accordance with the training data, the center(mean) of the distribution gradually drifted away from its initial position [Fig. 9(d) and (e)]. The CRBM system finally modeled a single-Gaussian distribution at epoch 1000, as shown by Fig. 9(f), whose center differs significantly from that of training data.

The nonideal training result is caused by the saturation of several parameter values, as revealed by Fig. 10. At epoch 120, parameters w_{23}, a_{h3}, and w_{03} reach their limits. The weight vector of hidden neuron $h_3, \mathbf{w}^{(3)} = \{w_{23}, w_{13}\}$, thus, point toward the upper-left corner of the data space [Fig. 9(b)]. As large values of a_{h3} and w_{03} further enhance the activation of h_3, the reconstruction drifts toward the left of the data space, as shown in Fig. 9(d). After 550 epochs, weight vectors $\mathbf{w}^{(1)} = \{w_{11}, w_{21}\}$ and $\mathbf{w}^{(4)} = \{w_{14}, w_{24}\}$ also reached their limits. Both vectors point toward the bottom-left corner of the state space [Fig. 9(b)], and thus encourage the reconstruction to drift toward the bottom-left corner.

Giving a clearer insight into the training result, Fig. 9(b) shows that $\mathbf{w}^{(2)}$ models the location of the training cluster, but $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(4)}$ points at an opposite direction against $\mathbf{w}^{(2)}$, causing reconstruction to appear at the bottom-left region. Nevertheless, Fig. 10(f) shows that the CRBM training process has tended to enhance the impact of $\mathbf{w}^{(2)}$ by increasing a_{h2}, and reducing the impact of $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(4)}$ by reducing $a_{h1}a_{h4}$, respectively. Note that high voltages of $\{a_i\}$ in a CRBM system correspond to small values of $\{a_i\}$ in a CRBM model [Fig. 4(b)].

B. Modeling Data With a Symmetric Distribution

A CRBM system was trained to model two clusters of Gaussian-distributed data, as shown in Fig. 11(a), for 1000 epochs. The traces of parameter values recorded during training are shown in Fig. 12, and Fig. 11(b) shows the projections of all weight vectors learnt after 1000 epochs into the data space. The 20-step reconstructions with parameter values learnt after 50, 200, 500, and 1000 epochs are shown in Fig. 11(c)–(f).

Fig. 11(c) shows that the CRBM system generates a crude approximation to the training data shortly after the onset of training. As $\{a_{vi}\}$ reached their limits at epoch 200, the variance of the reconstruction decreases to as small as a single cluster in

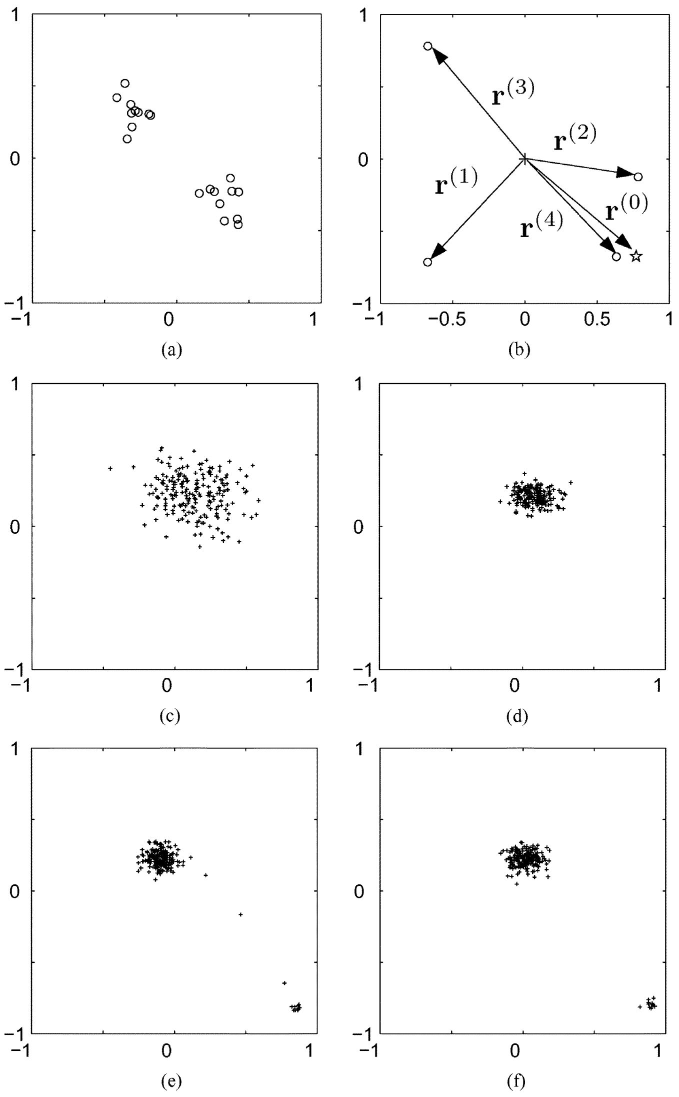


Fig. 11. (a) Twenty training data points sampled from two circular Gaussians. (b) Projections of the trained weight vectors of hidden neurons in data space, where $\mathbf{r}^{(i)} = \phi(\mathbf{w}^{(i)})$. (c)–(f) Twenty-step reconstructions generated by a CRBM model with parameters values learnt after (c) 50; (d) 200; (e) 500; (f) 1000 training epochs.

the training data [Fig. 11(d)]. Although parameter a_{h3} reached its limit again at epoch 200, parameters w_{23} and w_{03} did not approach their limits as rapidly as they did in the previous experiment. This allowed the reconstruction in Fig. 11(d) to maintain its center at about the same position as in Fig. 11(c). At epoch 500, the bias vectors $\mathbf{w}^{(0)} = \{w_{10}, w_{20}\}$ approached their limits and pointed toward the bottom-right corner of the data space [Fig. 11(b)]. The reconstruction in Fig. 11(e) therefore began to develop data points between the center and bottom-right corner of the data space. Finally, as w_{14} increased to a positive value after 1000 epochs, the weight vector $\mathbf{w}^{(4)}$ approximately aligns with $\mathbf{w}^{(0)}$, encouraging data points to be reconstructed at the bottom-right corner, as shown in Fig. 11(f). This result indicates that the CRBM system finally modeled a two-cluster distribution, while the saturation of several parameters prevented the CRBM system from modeling the correct positions of the clusters. It is particularly clear that the variance of the bottom-right cluster is "squashed" by the asymptote of the sigmoid function.

Comparing the projection of weight vectors in Fig. 11(b) to that in Fig. 9(b) reveals that $\mathbf{w}^{(2)}$ and $\mathbf{w}^{(4)}$ differ significantly in the two experiments. When modeling the single-Gaussian distribution, the CRBM system adapted $\mathbf{w}^{(2)}$ and $\mathbf{w}^{(4)}$ to discourage reconstruction at the bottom-right corner, while $\mathbf{w}^{(2)}$ and $\mathbf{w}^{(4)}$ were adapted to encourage reconstruction at the bottom-right corner for modeling the two-cluster distribution. This evident difference indicates that the CRBM system has adapted its limited number of parameters to minimizing the contrastive divergence between training data and modeled distribution. In other words, this demonstrates that continuous-valued probabilistic behavior in VLSI can be adapted toward modeling continuous-valued data distributions. Limitations, attributable to parameter saturation and not present in CRBM simulations [26], are apparent. These will be explored in Section V

V. Nonideal Training in a CRBM System

A simplified training task was used to investigate the cause of these nonideal training effects in the VLSI CRBM system. Furthermore, the nonideal training effects will be modeled analytically and tested by Matlab simulation.

A. *The Cause of Nonideal Training*

To investigate why parameters tend to approach their minimum or maximum limits, the training task was simplified to a single training datum, $(V_{1T}, V_{2T}) = (3, 3)$ (V). The trace of w_{20} and the output of visible neuron V2 were then monitored during a training process. Ideally, the updating direction for w_{20} follows:

$$\Delta w_{20} \propto \mathrm{sgn}(\langle v_2 \rangle_4 - \langle \hat{v}_2 \rangle_4) \tag{8}$$

where v_2 and $\hat{v}_2$ denote the initial and the one-step sampled outputs of V2, respectively, [note that $h_0 = 1$ and is, thus, not shown in (8)]. As the single training datum sets $v_2 = V_{2T} = 3$ V continuously, (8) indicates that w_{20} should be incremented as long as $\hat{v}_2 < 3$ V.

Fig. 13 shows the measured w_{20}, V2, and related digital control signals. CKdata and CKV activate alternately to sample initial datum V2 to v_2 and then one-step sample $\hat{v}_2$. Before training, $(\mathrm{LER}/\overline{\mathrm{REF}} = 0)$, $\hat{v}_2$ is clearly smaller than v_2. As soon as $\mathrm{LER}/\overline{\mathrm{REF}}$ goes high, w_{20} increments for several steps as predicted by (8), and the difference between v_2 and $\hat{v}_2$ decreases gradually. However, w_{20} starts to decrement after 15 ms and the difference between v_2 and $\hat{v}_2$ subsequently stops decreasing. This leads to the suspicion that the training circuit can not determine a correct updating direction when the contrastive divergence, $v_2 - \hat{v}_2$ in this experiment, is smaller than a certain threshold value (e.g., 200 mV).

To verify this suspicion, all training circuits were tested individually to detect various levels of contrastive divergence, $\Delta_D = (\langle s_i \cdot s_j \rangle_4 - \langle \hat{s}_i \cdot \hat{s}_j \rangle_4)$. Similar to the experiment above, the results reveal that, if Δ_D is smaller than a threshold value, say Δ_T, each training circuit updates its parameter in a fixed direction, either upwards or downwards. In addition, the fixed direction and the threshold value vary between training circuits.

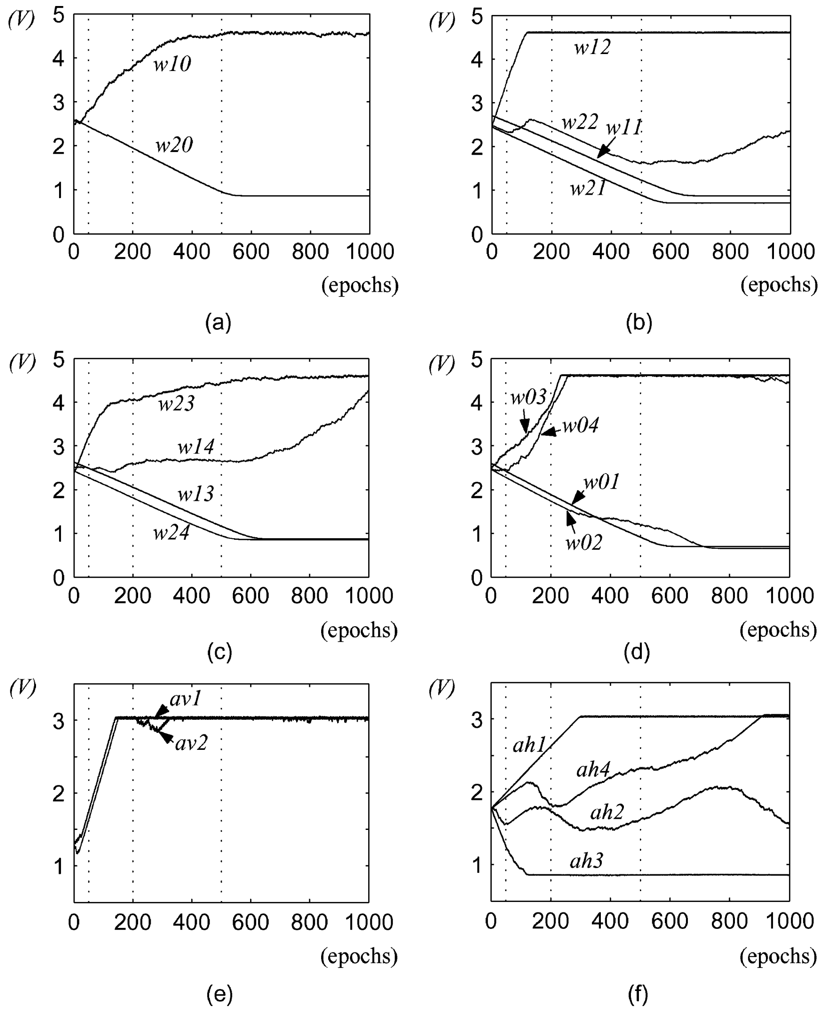


Fig. 12. Measured traces of $\{a_i\}$ and $\{w_{ij}\}$ when a CRBM system was trained with the data in Fig. 11(a) for 1000 epochs (a) $\mathbf{w}^{(0)}$; (b) $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ (c) $\mathbf{w}^{(3)}$; and $\mathbf{w}^{(4)}$ (d) $\{w_{0j}\}$; (e) $\{a_{vi}\}$; (f) $\{a_{hi}\}$. The dotted lines highlight the parameter values used to generate the 20-step reconstructions in Fig. 11(c)–(f).

The training circuits thus actually performs MCD training rules as

$$\begin{aligned}\Delta\lambda &= \eta_\lambda \cdot \mathrm{sgn}[(\langle s_i \cdot s_j\rangle_4 - \langle \hat{s}_i \cdot \hat{s}_j\rangle_4) + \Delta_T] \\ &= \eta_\lambda \cdot \mathrm{sgn}[\Delta_D + \Delta_T] \end{aligned} \tag{9}$$

where λ represents any parameter of a CRBM system, and Δ_T denotes an "offset" existing in a training circuit. If $\Delta_T > |\Delta_D| > 0$, (9) indicates that the training circuit always increases λ, regardless of $\Delta_D < 0$. Conversely, if $\Delta_T < -|\Delta_D| < 0$, (9) indicates that the training circuit always decreases λ, regardless of $\Delta_D > 0$. Let $\Delta_{TW}, \Delta_{TAV}, \Delta_{TAH}$ denote the measured offsets for $\{w_{ij}\}$, the $\{a_i\}$ for visible neurons, and the $\{a_i\}$ for hidden neurons, respectively. The measured offsets are

$$\begin{aligned}\Delta_{\mathrm{TW}} &= \begin{bmatrix} \times & -0.26 & -0.22 & +0.02 & -0.06 \\ +0.1 & -0.4 & +0.16 & -0.52 & -0.08 \\ -0.24 & -0.4 & -0.06 & +0.08 & -0.3 \end{bmatrix} (\mathrm{V}) \\ \Delta_{\mathrm{TAV}} &= [+0.14 \quad +0.18]\,(\mathrm{V}) \\ \Delta_{\mathrm{TAH}} &= [+0.14 \quad 0 \quad -0.12 \quad +0.16]\,(\mathrm{V}). \end{aligned} \tag{10}$$

Equation (10) indicates that the training circuit for w_{20} has an offset of -240 mV. A contrastive divergence smaller than 200 mV is thus not large enough to overcome this offset for w_{20} to increment, as shown in Fig. 13. Moreover, the measured offsets in (10) agree with the measured traces of parameters in Section IV. Parameters with training offsets smaller than 100 mV ($w_{03}, w_{04}, w_{22}, w_{23}$, and a_{h2}) had more dramatic changes during training, while others tended to increase or decrease monotonically. This suggests that offsets in training circuits are the primary nonideal effect that inhibits a CRBM system from modeling data faithfully.

B. Simulating Nonideal Training in A CRBM System

To confirm the suggestion above, a CRBM system was simulated in Matlab and was trained on the single-Gaussian data in Fig. 9(a), with the training rule in (9) and the offsets given in (10). Fig. 14 shows the simulated traces of parameters during 1000 training epochs. Fig. 14 is strikingly similar to Fig. 10[1] which represents the response of the VLSI CRBM, complete with training offsets. This agreement supports the suggestion

[1]Note that a higher value of $\{a_i\}$ in simulation corresponds to a lower voltage of $\{a_i\}$ in hardware, as indicated by Fig. 4(b).

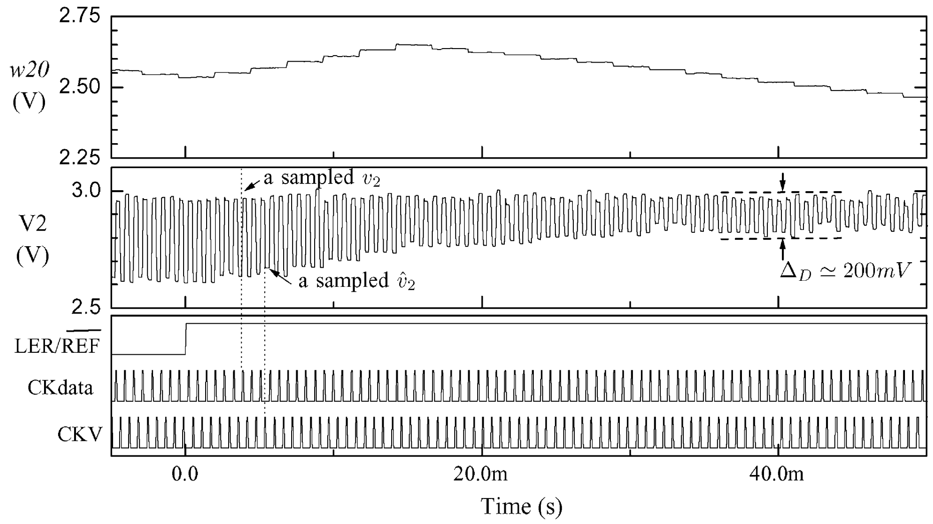


Fig. 13. Measured w_{20} and $V2$ when only one training datum, $(V_{1T}, V_{2T}) = (3, 3)$ (V), is presented to visible neurons.

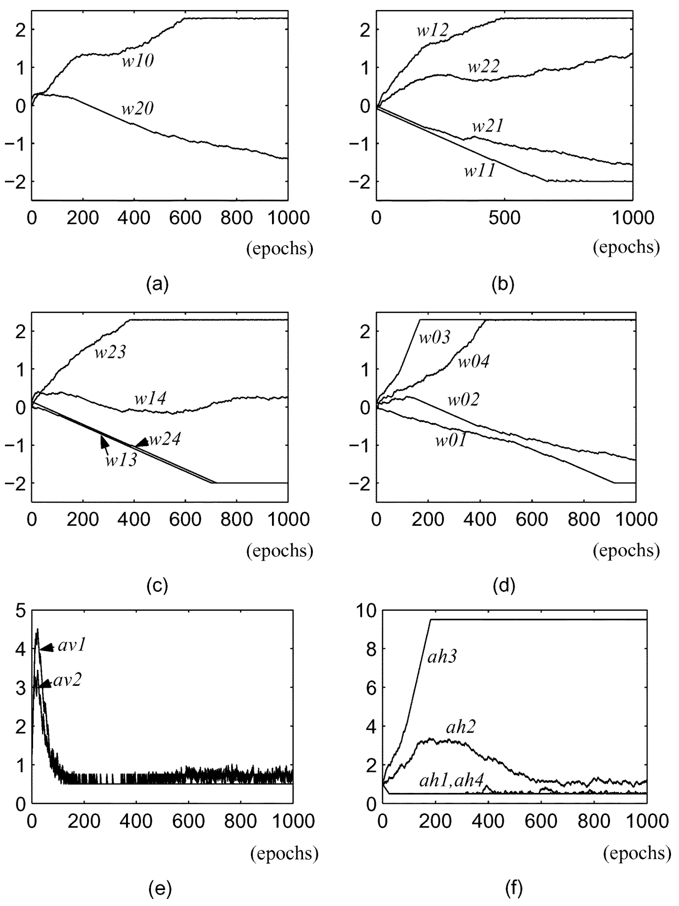


Fig. 14. Simulated traces of $\{a_i\}$ and $\{w_{ij}\}$ when a CRBM model was trained with the data in Fig. 9(a) for 1000 epochs: (a) $\mathbf{w}^{(0)}$; (b) $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ (c) $\mathbf{w}^{(3)}$; and $\mathbf{w}^{(4)}$ (d) $\{w_{0j}\}$; (e) $\{a_{vi}\}$; (f) $\{a_{hi}\}$.

that the CRBM system's inability to model training data faithfully can be attributed to training offsets. Furthermore, the experiment demonstrates that a good mapping (i.e., Table I) between software simulation and hardware implementation of a CRBM system has been established.

The near-perfect mapping between software and hardware facilitates the simulation of a CRBM system's tolerance to training offsets. To examine the maximum tolerable offset, it is assumed that $|\Delta_T|$ for all parameters are equivalent, whereas the signs of Δ_T remain in accord with (10). A CRBM system's performance on modeling the same single-cluster data [Fig. 9(a)], with various values of $|\Delta_T|$, was then simulated. Fig. 15 shows the 20-step reconstructions generated by a trained CRBM model when $|\Delta_T|$ equals (a) 0.05; (b) 0.01. As these values of $|\Delta_T|$ are all smaller than the measured values in (10), Fig. 15 shows that the CRBM modeled one cluster of data points corresponding with the training data. However, the CRBM also generated extra data points when $|\Delta_T|$ equals 0.05 [Fig. 15(a)]. This undesirable effect finally disappears when $|\Delta_T|$ equals 0.01 [Fig. 15(b)]. Therefore, the modeled CRBM system can tolerate a maximum offset of only 0.01, even when modeling a dataset as simple as a single cluster of data points.

C. Discussion

Fig. 6 suggests that the training offsets can be attributed to nonlinearity, mismatches, and clockfeedthrough errors in component circuits. Among these factors, mismatches and clockfeedthrough errors in the accumulator section should be the dominant ones. Although the accumulators cannot be tested individually in our fabricated CRBM system, they are identical to those used in [36], which known to introduce offsets with an average value of 7%. For s_i with a voltage range of 2 V (Table I), the offset is estimated to be 140 mV. This estimate is consistent with the measured values in (10), supporting the suggestion that the training offsets primarily arise in the accumulator section.

Nonideal hardware effects on a CRBM system were not analyzed before fabrication because a CRBM system should be able to use probabilistic behavior to enhance its robustness against noise or computational errors. This idea can be illustrated by noting that $\{s_i\}$ in (9) are inherently *noisy*, as indicated by (1). As long as the variance of noise is greater than the offset, the noise inherited in $\{s_i\}$ can shield the effects of the offset. However, experiment results indicated that nonzero offsets dominate instead, preventing training circuits from minimizing contrastive divergence when $|\Delta_D| < |\Delta_T|$. In other words, the

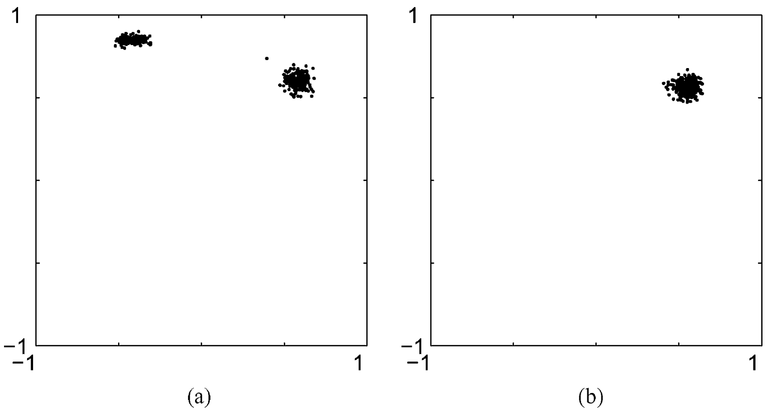


Fig. 15. Twenty-step reconstructions generated by a modeled CRBM system after 4000 training epochs with the offset $|\Delta_T|$ in (9) equaling to (a) 0.05; (b) 0.01.

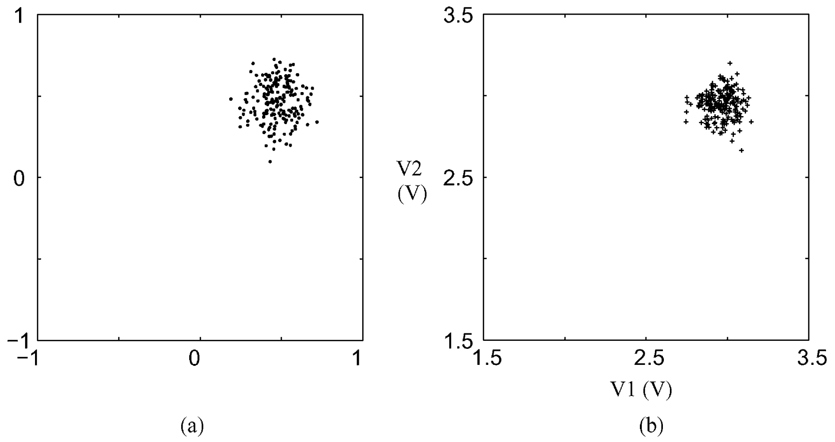


Fig. 16. (a) Twenty-step reconstruction generated by a CRBM model after being trained on the data in Fig. 9(a) for 3000 epochs. (b) Twenty-step reconstruction generated by a CRBM system with parameters refreshed to the levels in (11).

minimum divergence achievable for a CRBM system is equivalent to the offsets $|\Delta_T|$. Therefore, training circuits with reduced offsets will be essential if a CRBM system is to be trained optimally.

An improved training circuit with an offset smaller than 1% is quite a challenge, but not infeasible. One potential approach is redesigning accumulators and subtractors with the dynamic current mirrors proposed in [38], [39]. The dynamic current mirror avoids device mismatches by using the same transistor both to sample and to output a current. With clockfeedthroughs well compensated, the error of a dynamic current mirror can be as small as 500 ppm [39], 20 times smaller than the tolerable offset in a CRBM system. As dynamic current mirrors have an architecture similar to conventional current mirrors, training circuits based on dynamic current mirrors are expected to achieve satisfactory precision without consuming extra power and area. Finally, the saturation of parameters observed in Section IV should not be attributed to the limited voltage ranges set for parameters, as will be seen in the experiments in next section.

VI. Regenerating Continuous-Valued Data Distributions

Although nonnegligible training offsets prevent optimum modeling with the reported circuits for on-chip training, simulation of the VLSI CRBM system allows us to explore optimal training of a CRBM system in Matlab, and then to "download" the parameter values resulting from that training process on to a VLSI CRBM system for regenerating modeled distributions. This allows us to side-step the offsets introduced by the current training circuits and anticipate the performance of a VLSI CRBM with improved accumulators. Such experiments allow the exploration of noise-induced, continuous-valued probabilistic behavior in a VLSI CRBM system. The results are presented and discussed in the following section.

A. Regenerating a Single Cluster of Data Points

Taking into account all hardware simplifications and limitations, while removing training offsets, a CRBM model was trained on the single-Gaussian data in Fig. 9(a) for 3000 epochs. The trained CRBM model generated the 20-step reconstruction shown in Fig. 16(a). According to Table I and Fig. 4(b), the parameter values learned by the CRBM model are translated into parameter voltages for a CRBM system as follows:

$$\begin{aligned}\{w_{ij}\} &= \begin{bmatrix} \times & 3.452 & 2.612 & 3.064 & 0.961 \\ 2.916 & 2.569 & 2.738 & 2.849 & 2.219 \\ 2.484 & 3.033 & 2.313 & 2.441 & 1.959 \end{bmatrix} (\mathrm{V}) \\ \{a_{vi}\} &= [1.737 \quad 1.520]\,(\mathrm{V}) \\ \{a_{hi}\} &= [2.275 \quad 1.326 \quad 1.456 \quad 1.241]\,(V). \qquad (11)\end{aligned}$$

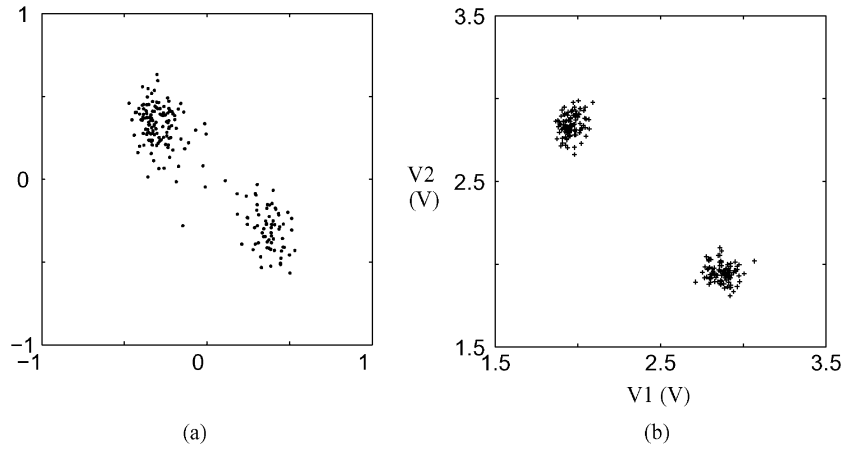


Fig. 17. (a) Twenty-step reconstruction generated by a CRBM model after being trained on the data in Fig. 11(a) for 30 000 epochs. (b) Twenty-step reconstruction generated by a CRBM system with parameters refreshed to the levels in (12).

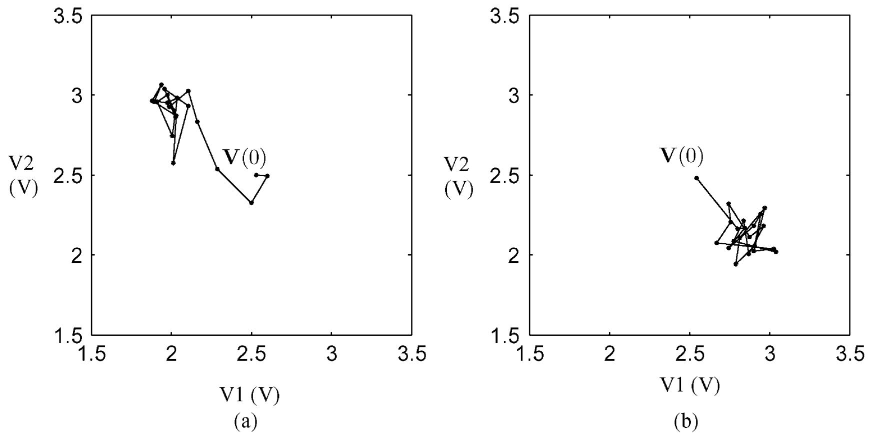


Fig. 18. Two distinct measured activities of visible neurons during 20 steps of Gibbs sampling in a CRBM system. (a) Trace composed of sampled visible states moving from initial state, $\mathbf{V}(0) = (2.5, 2.5)$ (V), toward the upper-left cluster. (b) Trace composed of sampled visible states moving from initial state, $\mathbf{V}(0) = (2.5, 2.5)$ (V), toward the bottom-right cluster.

With $\{w_{ij}\}$ and $\{a_i\}$ refreshed to the voltage levels in (11), a CRBM system generated the 20-step reconstruction shown in Fig. 16(b). The match between Fig. 16(a) and (b) demonstrates that the CRBM system is able to reconstruct data in single-Gaussian distribution, by the use of its noise-induced, continuous-valued probabilistic behavior. It is notable that all parameter voltages in (11) lie comfortably in the range specified in Table I. The saturation of parameter values therefore must not be attributed to limited voltage ranges available for parameters. The CRBM system could have modeled training data faithfully using its on-chip training circuits, if training offsets were negligible.

B. Regenerating Data With a Symmetric Distribution

To demonstrate the probabilistic behavior of a CRBM system more clearly, a CRBM system was set to regenerate the two-cluster data in Fig. 11(a), with parameter voltages learnt by a CRBM model. Fig. 17(a) and (b) shows the 20-step reconstructions generated by the trained CRBM model and by a CRBM system, respectively. The corresponding parameter voltages are

$$\{w_{ij}\} = \begin{bmatrix} \times & 3.203 & 2.598 & 2.400 & 1.319 \\ 2.862 & 2.280 & 2.180 & 4.061 & 2.589 \\ 2.616 & 2.365 & 2.471 & 1.811 & 2.565 \end{bmatrix} \text{(V)}$$
$$\{a_{vi}\} = [2.458 \quad 1.651]\,\text{(V)}$$
$$\{a_{hi}\} = [1.503 \quad 2.516 \quad 1.286 \quad 1.551]\,\text{(V)}. \tag{12}$$

Once again that the CRBM system is able to model two-cluster data with some accuracy, if training offsets are removed. Furthermore, the voltage ranges in (12) indicate that parameters have not saturated artificially owing to the constraints of hardware.

Moreover, this experiment facilitates the exploration of the "continuous-valued, probabilistic dynamics" of a CRBM system. Fig. 18 shows two distinct and different measured activities of the visible neurons in a CRBM system during 20-step Gibbs sampling. In each measurement, visible neurons were initialized to $\mathbf{V}(0) = (2.5, 2.5)$ (V), and then sampled for 20 steps. Each measurement thus contains 20 consecutively sampled visible states. The states of the visible neurons reflect

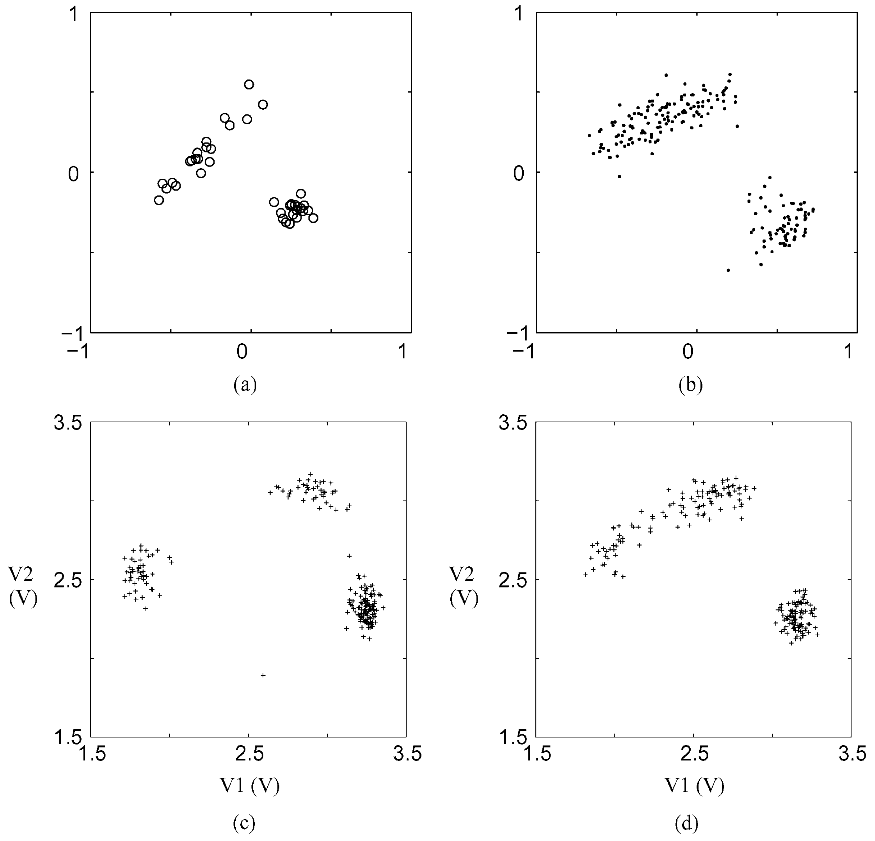


Fig. 19. (a) Training data sampled from one elliptic and one circular Gaussian. (b) Twenty-step reconstruction generated by a CRBM model after being trained on the data in (a) for 30 000 epochs. (c) Twenty-step reconstruction generated by a CRBM system with parameters refreshed to the levels in (13). (d) Twenty-step reconstruction generated by a CRBM system after a_{v1}, a_{h1}, and w_{10} are adjusted to 2.1, 2.274, and 2.45 V, respectively.

the changes during the 20 steps of sampling and Fig. 18 plots two such distinct traces of sampled visible states. It is clear that, although the visible neurons have the same initial states, the iteratively sampled visible states move either toward the top-left cluster or toward the bottom-right cluster in a series of steps. The visible states reach one of the clusters within the first few steps, and subsequently "move around" the region corresponding to this cluster. The mechanism behind formation of the two-cluster reconstruction in the generative model that is the CRBM [see Fig. 17(b)] thus, becomes clear. Furthermore, the "continuous-valued probabilistic dynamics" of a CRBM system are demonstrated clearly. The noise injected into the CRBM neurons prevents visible states from stabilising in a particular state, while the parameters of the CRBM system encode the regions with higher probabilities, corresponding to two attractors in this experiment, which encourage visible states to move toward and then around these regions.

C. Regenerating Data With a Nonsymmetric Distribution

A CRBM system was further tested to regenerate a more complicated distribution, as shown in Fig. 19(a), comprising data points sampled from one elliptic Gaussian and one circular Gaussian. This training dataset has been used to demonstrate the ability of the CRBM to model nonsymmetric data distributions [26], as well as to simulate a CRBM system's modeling ability under hardware simplifications and limitations [29]. A VLSI CRBM model was trained on this dataset for 30 000 epochs with $\eta_w = \eta_{ah} = 0.003$ and $\eta_{av} = 0.03$. The CRBM model generated the 20-step reconstruction as shown in Fig. 19(b), and the learnt parameter values correspond to parameter voltages of

$$\begin{aligned}\{w_{ij}\} &= \begin{bmatrix} \times & 2.662 & 2.702 & 1.927 & 2.425 \\ 2.571 & 4.544 & 1.661 & 2.446 & 2.748 \\ 2.186 & 3.732 & 3.875 & 2.451 & 2.593 \end{bmatrix} (\mathrm{V}) \\ \{a_{vi}\} &= [2.011 \quad 2.151]\,(\mathrm{V}) \\ \{a_{hi}\} &= [1.8287 \quad 1.2083 \quad 2.0883 \quad 2.4835]\,(\mathrm{V}). \qquad (13)\end{aligned}$$

With $\{w_{ij}\}$ and $\{a_i\}$ refreshed to the level in (13), a CRBM system generated the 20-step reconstruction as shown in Fig. 19(c). The cluster at the bottom-right corner in Fig. 19(c) corresponds to the circular cluster in training data. However, the reconstruction in Fig. 19(c) models the elliptic cluster in the training data rather roughly as two separate clusters. As separation of clusters are largely mediated by large values of $\{a_i\}$ [26], a better reconstruction of the training data, shown in Fig. 19(d), was obtained by adjusting a_{v1} to 2.1 V, a_{h1} to 2.274 V, and w_{10} to 2.45 V. This implies that reconstructing a complicated distribution requires some "tuning" of parameter values. This requirement is, however, able to be removed when the CRBM system is trained on-chip, without offsets, to achieve a minimum-divergence solution. On-chip training circuits, with substantially reduced offsets, are therefore necessary for a CRBM system.

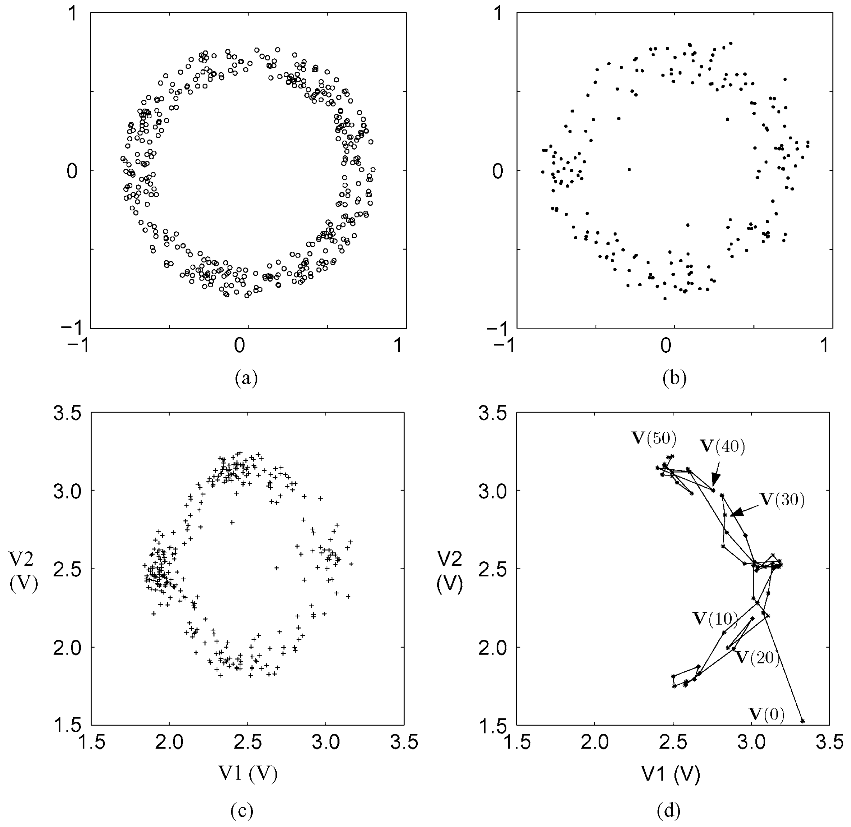


Fig. 20. (a) Training data with a doughnut-shaped distribution. (b) Twenty-step reconstruction generated by a CRBM model after being trained on the data in (a) for 20 000 epochs. (c) Twenty-step reconstruction generated by a CRBM system with the parameter values learnt by the CRBM model (a_{v1} and a_{v2} were adjusted to 1.41 and 1.1 V, respectively). (d) Trace of sampled visible states when the CRBM system was Gibbs sampled for 50 steps.

D. Regenerating Data With a Doughnut-Shaped Distribution

To highlight the distinctive continuous-valued probabilistic behavior a CRBM system possesses, a CRBM system was used to regenerate data with a doughnut-shaped distribution, as shown in Fig. 20(a). To model this training dataset well, the CRBM must be able to capture correlations between probabilities in the two dimensions.

A CRBM model was trained on the data for 20 000 epochs with $\eta_w = 0.000\,75, \eta_{av} = \eta_{\mathrm{ah}} = 0.0075$. The trained CRBM model generated the 20-step reconstruction shown in Fig. 20(b). The reconstruction forms a circular band with larger variance than the training data and a slightly "uneven" distribution of reconstructed data. The CRBM's four hidden neurons provide only a limited number of parameters to represent the circular band perfectly. The corresponding parameter voltages learnt by the CRBM model are

$$\begin{aligned}\{w_{ij}\} &= \begin{bmatrix} \times & 2.503 & 2.548 & 2.451 & 2.498 \\ 2.497 & 2.964 & 2.514 & 2.526 & 2.025 \\ 2.498 & 2.973 & 2.503 & 2.434 & 2.975 \end{bmatrix} (\mathrm{V}) \\ \{a_{vi}\} &= [1.287 \quad 1.294]\,(\mathrm{V}) \\ \{a_{hi}\} &= [1.154 \quad 1.835 \quad 1.898 \quad 1.173]\,(\mathrm{V}). \end{aligned} \tag{14}$$

By downloading the learnt parameters into a CRBM system with slight adjustments on a_{v1} and a_{h1}, the CRBM system regenerated the 20-step reconstruction shown in Fig. 20(c). The reconstruction not only retains a slight uneven distribution similar to that in Fig. 20(b), but also represents the circular band roughly. The roughness is due to the fact that the multipliers in neurons do not calculate $\sum w_{ij}s_j$ as linearly as in simulation. Multi-step sampling amplifies the nonlinearity, and thus results in a distorted circular shape. This reinforces, once again, the need for on-chip training circuits to model a complex distribution optimally, adapting with the multipliers' nonlinearity (for example) in place. Nevertheless, this experiment demonstrates that the CRBM system is able to model a complex distribution with only four hidden neurons, a capability that is a result of the continuous-valued probabilistic behavior for which the CRBM system was developed.

To further illustrate the richly probabilistic, continuous-valued behavior of a CRBM system, Fig. 20(d) shows the trace of sampled visible states during 50 steps of Gibbs sampling. The visible state was initialized to $\mathbf{V}(0)$, a point at the bottom-right corner of the data space. The visible state then moved into the circular band of the doughnut-shaped distribution in the first few steps, as shown by Fig. 20(d), and visited different states in the right half of the circular band with nearly equal probability. This result demonstrates the "dynamics" of a CRBM system as a more sophisticated, stochastic moving of its visible states. The high-probability regions defined by the CRBM system form a circular *low-energy* valley. Fig. 20(d) shows that the visible state travels along the right half of the valley, but is "bounced" back at the top and bottom ends. This indicates that the CRBM system creates small energy barriers in these places. If the visible state does not get enough "noise" (energy) it cannot pass the barriers to visit the left half of the circular band. The slight uneven distribution in Fig. 20(c) thus becomes clear.

VII. Conclusion

This paper set out to explore the both abilities and limitations of a mixed-mode implementation of a probabilistic architecture (the CRBM) developed previously by the authors [25], [26]. The CRBM was developed to use continuous-valued stochastic computation to model real-valued data and complex distributions where a binary representation is inappropriate. A mixed-mode VLSI CRBM system with two visible and four hidden neurons has been fabricated and tested. The training experiments indicate that continuous-valued probabilistic behavior in VLSI can be achieved and trained, based on the algorithm of the CRBM, toward modeling a range of continuous-valued data distributions. This is all clear from the significantly different results between training experiments, even though nonideal training offsets in current circuitry prevent the CRBM system from modeling data optimally.

Moreover, the utility of continuous-valued probabilistic behavior in VLSI has been demonstrated as a CRBM system's ability to regenerate a variety of simple and more complex *continuous-valued* data distributions, with as few as four hidden neurons. The measurements of the probabilistic dynamics of visible neurons further highlights the distinctive and rich probabilistic behavior a CRBM system possesses. These promising results indicate that computation with continuous-valued probabilistic behavior in VLSI is feasible, and that a CRBM system is potential for an intelligent embedded system, provided improved training circuits can be achieved.

For high-dimensional data and real applications, it is important that a CRBM system is developed to be scalable and programmable. The nonideal training results observed in this paper further indicate that probabilistic behavior appears not to always guarantee the robustness of a CRBM system, as was initially assumed. It is thus essential to analyze the robustness of a large-scale CRBM system quantitatively, and this will lead to the interesting examination on the suggestion that VLSI systems with probabilistic behavior are robust against computational errors and noise. While Gaussian noise has been used in this paper, it will be also interesting to explore the CRBM system's performance in the presence of other types of noise. Furthermore, as continuous-valued, probabilistic dynamics have been proved to be implementable in VLSI, the similarity between the CRBM and the diffusion network [40], [26] further encourages the exploitation of continuous-valued, probabilistic "dynamics" in VLSI.

Appendix
Derivation of the Mapping for Parameters in A CRBM System

A. Noise-Control Parameter $\{a_i\}$

The mapping for parameter $\{a_i\}$ of the CRBM system, shown in Fig. 4(b) is derived by the following steps.

1) Select $V_{ai} = 1.8$ V to correspond to $a_i = 1$ in simulation.
2) Choose a reference current input I_r. Let V_r denote the output voltage of the sigmoid-function circuit when $I_{\text{in}} = I_r$ and $V_{ai} = 1.8$ V.
3) With the same input current I_r but various levels of V_{ai}, measure the output voltages of the sigmoid-function circuits $V_o(V_{ai})$.
4) The mapping between V_{ai} and a_i is then derived according to

$$a_i(V_{ai}) = \frac{\phi^{-1}(V_o(V_{ai}))}{\phi^{-1}(V_r)} \tag{15}$$

where $\phi^{-1}(\cdot)$ is the inverse of the sigmoid function $\phi(\cdot)$ with $\theta_H = -\theta_L = 1$. The result in Fig. 4(b) is an average over measurements from three fabricated chips.

B. Noise-Scaling Parameter σ

With constant V_{ni} but various levels of V_σ in Fig. 3, the peak-to-peak values of a neuron's output were measured when all $\{s_j\} = 2.5$ V (i.e., no deterministic input). Let $\triangle_n(V_\sigma)$ denote a measured peak-to-peak value for a particular V_σ. As 99.7% of samples from a Gaussian distribution lie in three standard deviations, the corresponding value of σ is estimated according to

$$\sigma(V_\sigma) = \frac{\phi^{-1}(\triangle_n(V_\sigma)/2)}{3} \tag{16}$$

where $\phi^{-1}(\cdot)$ is the inverse of the sigmoid function $\phi(\cdot)$. The derived mapping between V_σ and σ is shown in Fig. 5(b).

References

[1] T. B. Tang, E. Johannessen, L. Wang, A. Astaras, M. Ahmadian, A. F. Murray, J. M. Cooper, S. P. Beaumont, B. W. Flynn, and D. R. S. Cumming, "Toward a miniature wireless integrated multisensor microsystem for industrial and biomedical applications," *IEEE Sens. J. Special Issue on Integr. Multisensor Syst. Signal Process.*, vol. 2, no. 6, pp. 628–635, 2002.

[2] E. A. Johannessen, L. Wang, L. Cui, T. B. Tang, M. Ahmadian, A. Astaras, S. W. Reid, P. Yam, A. F. Murray, B. W. Flynn, S. P. Beaumont, D. R. S. Cumming, and J. M. Cooper, "Implementation of distributed sensors in a microsystems format," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 3, pp. 525–535, 2004.

[3] K. Najafi and K. D. Wise, "An implantable multielectrode array with on-chip signal processing," *IEEE J. Solid-State Circuits*, vol. 21, no. 6, pp. 1035–1044, 1986.

[4] T. W. Berger, M. Baudry, R. D. Brinton, J.-S. Liaw, V. Z. Marmarelis, A. Y. Park, B. J. Sheu, and A. R. Tanguay, "Brain-implantable biomimetic electronics as the next era in neural prosthetics," *Proc. IEEE*, vol. 89, no. 7, pp. 993–1012, 2001.

[5] K. D. Wise, D. J. Anderson, J. F. Hetke, D. R. Kipke, and K. Najafi, "Wireless implantable microsystems: High-density electronic interfaces to the nervous system," *Proc. IEEE*, vol. 92, no. 1, pp. 76–97, 2004.

[6] M. Sun, M. Mickle, L. Wei, Q. Liu, and R. Sclabassi, "Data communication between brain implants and computer," *IEEE Trans. Neural Syst. Rehab. Eng.*, vol. 11, no. 2, pp. 189–192, 2003.

[7] M. A. L. Nicolelis, "Actions from thoughts," *Nature*, vol. 409, pp. 403–407, Jan. 2001.

[8] M. Mihaila, "Low-frequency noise in nanomaterials and nanodevices," in *Proc. Int. Semicond. Conf. (CAS 2001)*, vol. 1, Oct. 2001, pp. 31–36.

[9] Z. Celik-Butler, "Low-frequency noise in deep-submicron metal-oxide-semiconductor field-effect transistors," in *Inst. Elect. Eng. Proc. Circuits, Devices, Syst.*, vol. 149, Feb. 2002, pp. 23–31.

[10] G. Ghibaudo and T. Boutchacha, "Electrical noise and its fluctuations in advanced cmos devices," *Microelectron. Reliab.*, vol. 42, pp. 573–582, 2002.

[11] D. F. Specht, "Probabilistic neural networks," *Neural Netw.*, vol. 3, p. 109, 118, 1990.

[12] I. C. Jou, R. Y. Liu, and C. Y. Wu, "CMOS implementation of neural networks for speech recognition," in *IEEE Asia-Pacific Conf. Circuits Syst.*, 1994, pp. 513–518.

[13] N. Aibe, M. Yasunaga, I. Yoshihara, and J. H. Kim, "A probabilistic neural network hardware system using a learning-parameter parallel architecture," in *Proc. 2002 Int. Joint Conf. Neural Netw.*, vol. 3, 2002, pp. 2270–2275.

[14] D. Hsu, S. Bridges, M. Figueroa, and C. Diorio, "Adaptive quantization and density estimation in silicon," in *Advances in Neural Info. Process. Syst. (NIPS02)*, vol. 15, Vancouver, BC, Canada, Dec. 2003.

[15] R. Genov and G. Cauwenberghs, "Stochastic mixed-signal VLSI architecture for high-dimensional kernel machines(nips0l)," *Advances in Neural Info. Process. Syst. (NIPS0l)*, vol. 14, pp. 1099–1105, 2002.

[16] ——, "Kerneltron: Support vector machine in silicon," *IEEE Trans. Neural Netw.*, vol. 14, no. 8, pp. 1426–1434, 2003.

[17] J. M. Quero, J. G. Ortega, C. L. Janer, and L. G. Franquelo, "VLSI implementation of a fully parallel stochastic neural network," in *IEEE Int. Conf. Neural Netw.*, vol. 4, 1994, pp. 2040–2045.

[18] D. E. V. D. Bout and T. K. Millerlll, "A digital architecture employing stochasticism for the simulation of hopfield neural nets," *IEEE Trans. Circuits Syst.*, vol. 36, no. 5, pp. 732–738, 1989.

[19] A. Torralba, F. Colodro, E. Ibanez, and L. G. Franquelo, "Two digital circuits for a fully parallel stochastic neural network," *IEEE Trans. Neural Netw.*, vol. 6, no. 5, pp. 1264–1268, 1995.

[20] T. G. Clarkson, D. Gorse, J. G. Taylor, and C. K. Ng, "Learning probabilistic ram nets using VLSI structures," *IEEE Trans. Comput.*, vol. 41, no. 12, pp. 1552–1561, 1992.

[21] T. G. Clarkson, C. K. Ng, and Y. Guan, "The pram: An adaptive VLSI chip," *IEEE Trans. Neural Netw.*, vol. 4, no. 3, pp. 408–412, 1993.

[22] B. R. Gaines, "Stochastic computing systems," in *Advances in Information System Science*, B. R. Gaines, Ed. New York: Plenum, 1969, vol. 2, pp. 37–172.

[23] B. D. Brown and H. C. Card, "Stochastic neural compuation I: Computational elements," *IEEE Trans. Comput.*, vol. 50, no. 9, pp. 891–905, 2001.

[24] ——, "Stochastic neural computation II: Soft competitive learning," *IEEE Trans. Comput.*, vol. 50, no. 9, pp. 906–920, 2001.

[25] H. Chen and A. F. Murray, "A contiuous restricted Boltzmann machine with a hardware-amenable learning algorithm," in *Proc. 12th Int. Conf. Artificial Neural Netw. (ICANN2002)*, Madrid, Spain, Aug. 2002, pp. 358–363.

[26] ——, "A continuous restricted boltzmann machine with an implementable training algorithm," in *Inst. Elect. Eng. Proc. Vision, Image and Signal Process.*, vol. 150, 2003, pp. 153–158.

[27] J. Alspector, J. W. Gannett, S. Haber, M. B. Parker, and R. Chu, "A VLSI-efficient technique for generating multiple uncorrelated noise sources and its application to stochastic neural networks," *IEEE Trans. Circuits Syst.*, vol. 38, no. 1, pp. 109–123, 1991.

[28] G. Cauwenberghs, "Delta-sigma cellular automata for analog VLSI random vector generation," *IEEE Trans. Circuits Syst.*, vol. 46, no. 3, pp. 240–250, Mar. 1999.

[29] H. Chen, P. Fleury, and A. F. Murray, "Minimizing contrastive divergence in noisy, mixed-mode VLSI neurons," in *Advances in Neural Info. Process. Syst. (NIPS03)*, vol. 16, 2004. in press.

[30] J. Alspector, A. Jayakumar, and S. Luma, "Experimental evaluation of learning in a neural microsystem," in *Advances in Neural Info. Process. Syst. (NIPS91)*, vol. 4, Dec. 1992, pp. 871–878.

[31] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in boltzmann machine," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D. Rumelhart and J. McClelland, Eds. Cambridge, MA: MIT, 1986, pp. 283–317. PDP Res. Group.

[32] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[33] T. Tang, H. Chen, and A. Murray, "Adaptive stochastic classifier for noisy pH-ISFET measurements," in *Proc. 13th Int. Conf. Artificial Neural Netw. (ICANN2003)*, Istanbul, Turkey, Jun. 2003, pp. 638–645.

[34] A. F. Murray, "Novelty detection using products of simple experts—A potential architecture for embedded systems," *Neural Netw.*, vol. 14, no. 9, pp. 1257–1264, 2001.

[35] J. Lansner and T. Lehmann, "An analog cmos chip set for neural networks with arbitrary topologies," *IEEE Trans. Neural Netw.*, vol. 4, no. 3, pp. 441–444, 1993.

[36] P. Fleury, H. Chen, and A. F. Murray, "On-chip contrastive divergence learning in analogue VLSI," in *Proc. Int. Joint Conf. Neural Netw.*, Budapest, Hungary, Jul. 2004.

[37] M. Banu and Y. Tsividis, "Floating voltage-controlled resistors in cmos technology," *Electron. Lett.*, vol. 18, pp. 678–679, 1982.

[38] G. Wegmann and E. A. Vitozz, "Very accurate dynamic current mirrors," *Electron. Lett.*, vol. 25, no. 10, pp. 644–646, 1989.

[39] ——, "Basic principles of accurate dynamic current mirrors," in *Inst. Elect. Eng. Proc. Circuits, Devices Syst.*, vol. 137, 1990, pp. 95–100.

[40] J. R. Movellan, P. Mineiro, and R. J. Williams, "A Monte-Carlo EM approach for partially observable diffusion processes: Theory and applications to neural networks," *Neural Comput.*, vol. 14, no. 7, pp. 1507–1544, 2002.

Hsin Chen (M'04) was born in 1974 in Taiwan. He received the B.S. and M.S. degrees in electrical engineering in 1996 and 1998, respectively, from National Tsing-Hwa University (NTHU), Taiwan. After obligating two years' military service in the Taiwan Air Force, he received the Ph.D. degree in electronics in 2004 from Edinburgh University, Edinburgh, U.K.

Since then, he was appointed Assistant Professor with the Department of Electrical Engineering, NTHU. His electrical engineering expertise lies in analog- and mixed-mode VLSI design. As a researcher, he has devoted his greatest enthusiasm to neural hardware, looking to explore bioinspired, particularly brain-inspired, computation in VLSI. The neural hardware involves VLSI implementation of probabilistic neural computation, neuromorphic VLSI that mimics the morphology of biological systems, and neuroengineered VLSI that could communicate with biological nervous systems.

Patrice C. D. Fleury received the Bac.F2 degree in 1994 and the D.U.T 'Genie Electronique et Informatique Industrielle' from the University of Lannion, France, in 1997. He received the B.Eng. (Hons.) degree in electronic engineering and the M.Sc. degree in electronics and computer-based systems from the University of Huddersfield, West Yorkshire, U.K.

He is currently working toward the Ph.D. degree in electrical engineering at the University of Edinburgh, Edinburgh, U.K. Since 2002, he has also been working as a Research Associate with the Institute for Integrated Micro and Nano Systems (IMNS), Edinburgh University. His research interests include neuroengineering, interfacing artificial devices with biological systems, analog VLSI and mixed-signal design, and probabilistic neural computation.

Alan F. Murray (SM'93) was born in Edinburgh in 1953. He received the B.Sc. Hons. degree in physics, in 1975, and the Ph.D. degree in solid-state physics, in 1978, both from the University of Edinburgh, Edinburgh, U.K.

He worked for three years as a Research Physicist (two years in Canada and the third year in Edinburgh), and for three years as an Integrated Circuit Design Engineer with Wolfson Microelectronics. In 1984, he was appointed a lecturer in electrical engineering at Edinburgh University, became a Reader in 1991, and a Professor of neural electronics in 1994. He is interested in all aspects of neural computation and hardware issues and applications have been his primary research interest since 1985. In 1986, he developed the "pulse stream" method for neural integration. His interests have since widened to include all aspects of neural computation, particularly hardware-compatible learning schemes, probabilistic neural computation, and neural forms that utilize the temporal—and noisy characteristics of analog VLSI—as well as applications of hardware neural networks. He is also developing a new interest in the interface between silicon and neurobiology, along with colleagues in biomedical sciences and at Glasgow University, U.K. He has more than 200 publications, including an undergraduate textbook and research texts on neural VLSI, applications of neural networks, and noise in neural training (with P. Edwards).

Dr. Murray is a Fellow of the Institute of Electrical Engineers (U.K.), a member of INNS, and a fellow of the Royal Society of Edinburgh.