Fully-Digital Oscillatory Associative Memories
Enabled by Non-volatile Logic

Vehbi Calayir and Larry Pileggi

Abstract—Due to its brain-like parallel processing, neurocomputing has been regarded as an intriguing alternative to traditional von Neumann architectures for such applications as image processing, pattern recognition, and associative memory. Associative memories based on neurocomputing attempt to mimic the human brain via a parallel network of coupled artificial neurons. Oscillatory neural networks (ONNs) have been proposed for such purposes; however, CMOS-based implementations would be inefficient due to the corresponding circuit complexity of oscillators and phase-locking mechanisms. In addition, programmability of the synaptic weights would require numerous reconfigurable, complex analog circuits that represent an impractical power and area overhead. In this paper we propose a fully-digital ONN architecture that is enabled by non-volatile logic. Using a newly proposed all-magnetic logic family, mLogic, we demonstrate the efficacy of digitizing the oscillators and phase relationships by exploiting the inherent storage. We perform a device-level simulation-based comparison of mLogic and 32nm CMOS for a fully-interconnected 60-neuron system, and show approximately 15x area improvement and 18x power improvement that would be achieved for a large system with 100k neurons.

I. INTRODUCTION

NEUROCOMPUTING is considered an intriguing alternative to computing based on traditional von Neumann architectures due to its brain-inspired massive parallelism. For example, a neurocomputer performs pattern recognition via associative memory instead of having general purpose computation by executing a list of commands. It maps a set of input patterns to a set of output patterns via specific weights between computational units, whereby an output pattern can be retrieved for a given input pattern. Hence, it is a powerful component for applications such as pattern recognition, image processing, and associative memory. These applications would otherwise require numerous memory fetch operations and a processor that is executing a list of commands for optimization.

Neurocomputers constructed as associative memories consist of a network of coupled dynamic units, referred to here as artificial neurons, that process information in parallel. They are analogous to the nervous system in human brain in a way that each brain neuron corresponds to a computational unit in a neurocomputer, and a connection between two artificial neurons represents a synapse connecting two brain neurons. The strength of this synapse is the connection coefficient (i.e., synaptic weight) in a neurocomputer that relates one artificial neuron to another.

Oscillatory neural networks (ONNs) are a promising class of neural networks that can be used as an associative memory due to their unique neurocomputing properties [1]-[3]. But each neuron requires a voltage- or current-controlled oscillator to represent neuronal oscillations, and a mechanism that controls phase relationships among n neurons. For example, the neurocomputing system based on ONNs proposed in [1] relies on voltage-controlled oscillators (VCOs) to represent oscillations, and phase-locked loops (PLLs) to synchronize artificial neurons to each other (see Fig. 1). Although theoretically viable, it is impractical to integrate n VCOs and n PLLs for any sizable associative memory based on power and area requirements.

More importantly, each oscillatory neuron is interconnected to every other neuron, thereby necessitating programmability of n² phase relationships among n neurons. Enabling the programming of these connection coefficients would require n² variable gain amplifiers (VGAs), which further exacerbates circuit complexity and power consumption.

![Fig. 1. Conceptual architecture of PLL NNs for a 2-neuron system. The programmable connection coefficients can be implemented with VGAs.](image)

While CMOS is ineffective for constructing oscillatory associative memories, emerging nanotechnologies could make such implementations feasible. Nanoscale spin-torque oscillators (STOs) [4] and resonant body transistors (RBTs) [5] offer single device oscillators that could be used to model oscillatory neurons. STOs have been demonstrated to couple when implemented on a shared free layer for the magnetic spin [6]; however, efficiently implementing a
programmable phase among all oscillators to represent the stored data is not clearly feasible. Moreover, dc current is required to specify the oscillation frequency, thereby consuming significant stand-by power to store state. Several CMOS amplifying stages are also required to boost the tiny STO output power [4]. RBTs show similar promise as oscillators, but also suffer from similar analog circuit complexity to program phase shifts among oscillators in the array.

Recently, specific memristors have been implemented as electronic synapses, exploiting the non-volatility of such devices [7]-[9]. The voltage-controlled resistances that remain the same even when powered off are adjusted to have programmable synaptic weights. However, neuron circuits are still implemented in CMOS, thus requiring tight CMOS integration with memristors. Also, oscillators and phase-locking mechanisms are still needed for the realization of the ONNs.

In this paper we propose an approach for associative memories whereby the non-volatile storage of state offered by certain nanotechnology devices (e.g., CRRAM [10], nanowires [11], and mCells [12]) is exploited to build a fully-digital ONN with great efficiency. The inherent storage is exploited to digitize the oscillators and phase relationships, thereby enabling efficient control of the reconfigurable connection coefficients and elimination of the need for complex analog circuits (e.g., VGAs and PLLs). In addition, the non-volatile storage enables highly-efficient logic pipelining without requiring D flip-flops [12], which significantly reduces the area and power for large systems.

As a demonstration example we build our proposed ONN architecture using a newly proposed magnetic logic family, mLogic, that is enabled by the mCells [12]. Using circuit-level simulations we demonstrate that only a few hundred mLogic gates with reasonable power consumption are required to model each oscillatory neuron for a fully-interconnected 60-neuron architecture. Moreover, as the number of neurons increases in the system, the mLogic implementation would improve the power and area significantly as compared to a CMOS implementation. For example, for an oscillatory associative memory with 100k nodes the improvements with mLogic reach approximately 15x in device count and 18x in power.

The rest of the paper is organized as follows. Section II describes our proposed ONN architecture, followed by a brief description of the mCells in Section III. Section IV shows our mLogic technology, followed by our implementation comparison with CMOS in Section V. Finally, we summarize with our conclusions in Section VI.

II. PROPOSED ONN ARCHITECTURE FOR ASSOCIATIVE MEMORIES

ONNs are particularly interesting systems since they mimic nature and the human brain. In ONNs, memorized patterns are synchronized oscillatory states in which neurons periodically communicate with each other according to certain relations between their phases. This interaction between neurons is based on phase modulation (PM) encoding. The PM encoding neuron changes its firing pattern (i.e., at which time within the cycle the firing occurs) to represent its state. It is theoretically shown that ONNs can be built using PLLs [1], laser oscillators [13], and microelectromechanical system (MEMS) oscillators [14].

Due to the aforementioned implementation issues associated with ONNs, however, we propose a fully-digital ONN system based on converting the continuous-time ONN dynamics into the discrete-time ONN dynamics as shown below.

A. Derivation of Discrete-Time ONN Dynamics

Referring to the PLL-based NN shown in Fig. 1, its dynamics is given by [1]:

$$\theta_i = \Omega + V(\theta) \sum_{j=1}^{n} s_j V(\theta_j - \pi/2) \quad (1)$$

where $\theta_i$ is the phase of the $i^{th}$ VCO, $\Omega$ is the natural frequency of the system, $s_j$'s are connection coefficients, and $V$ is the VCO output signal. Since $\Omega >> 1$, we can average Eq. (1) over time, which yields:

$$\theta_i = \Omega + \sum_{j=1}^{n} s_j H(\theta_j - \theta_i) \quad (2)$$

where the averaged transfer function $H$ is:

$$H(\theta_j - \theta_i) = \lim_{T \to \infty} \frac{1}{T} \int_{0}^{T} V(\theta_j)V(\theta_j - \pi/2) dt . \quad (3)$$

Various types of transfer functions are given in [1] based on the different PLL output waveforms. Since we discretize the whole system, any of them could be selected; however, the transfer function corresponding to the rectangular waveform is the easiest one to discretize (see Fig. 2). This transfer function has only three stable points at $\theta = 0, \pi$, and $-\pi$. Therefore, synchronization is achieved only if all oscillatory neurons are in-phase or anti-phase with respect to each other.

Next we define excess phase as:

$$\phi_i = \theta_j - \Omega t . \quad (4)$$

![Fig. 2. Transfer function for fully-digital ONN.](image)
Plugging (4) into (2) results in:

$$\phi_j = \sum_{j=1}^{n} s_j H(\phi_j - \phi) .$$

(5)

Discretizing the continuous-time ONN system in (5) gives the discrete-time ONN dynamic equation as:

$$\phi[k+1] = \phi[k] + \sum_{j=1}^{n} s_j H(\phi_j[k] - \phi[k])$$

(6)

where $k$ is the iteration step. The training algorithm is applied to specify the connection coefficients that represent the patterns that are stored in this system. Solutions to this difference equation correspond to the stored patterns.

B. Implementation of Fully-Digital ONN

The proposed fully-digital ONN architecture shown in Fig. 3 is the representation of the discrete-time ONN dynamic system in (6). One adder without a reset signal is needed to accumulate excess phase. The transfer function block in Fig. 3 is the realization of $H$ function drawn in Fig. 2. $u_1$ and $u_2$ are the initial inputs for the first and second neurons, respectively.

Fig. 3. High-level circuit representation of the discrete-time ONN. The signal-selecting blocks are the multiplexers.

Most importantly, efficient implementation of this discretized ONN system is enabled by the inherent storage of the non-volatile logic family that allows us to digitally represent oscillations in the phase domain instead of converting the phase-domain computations into the voltage domain. With local storage of state we can easily track the accumulation of excess phase, thus transforming the oscillator’s function into the digital domain. Moreover, phase relationships among $n$ neurons are discretized via the non-volatile storage by digitally representing the transfer function shown in Fig. 2.

The non-volatile logic with inherent storage of state offers additional advantages: i) easy programmability without the need for extra memory; and ii) high throughput via a fully-pipelined architecture without the need for D flip-flops [12]. With inherent pipelining and simplified programmability, only a few hundred non-volatile logic cells are required to model each oscillatory neuron. With improvements in training algorithms for ONNs, even fewer logic gates would be needed. For example, the training algorithm proposed in [15] that can be potentially applied to any NNs used for associative memories is claimed to render implementation of sparsely-interconnected NNs possible. If it can be successfully applied to oscillatory associative memories as well, it will enable sparsely-interconnected ONNs, thus further simplifying the proposed ONN implementation.

C. Behavioral Simulation

To validate the functionality of our discretized ONN system we generate a high-level behavioral model with Matlab based on our proposed architecture that is shown in Fig. 3. This model is formed by direct implementation of each individual block (i.e., multiplier, multiplexer, accumulator, subtractor, and transfer function) in this proposed architecture. Using the pattern recognition example from [1], the 60-pixel bit patterns shown in Fig. 4 are stored in the system via connection coefficients. We use Hebbian Rule to calculate synaptic weights as described in [1].

In our model we represent inputs and weights as 5-bit signed binary numbers. Therefore, 31 possible phase levels can be defined in this system. Moreover, number of step size to recognize the pattern would alter with different choices of representing data and how higher order bits generated during neural computation are handled.

Fig. 5 shows our first example with an input pattern that is generated as close to that used in [1] as possible. Eight steps are required to fully recognize the pattern. In this example, the oscillatory associative memory gets locked to one of the stored patterns as designated in the example in [1].

Fig. 4. 60-pixel memorized bit patterns. They are stored in the neural network via connection coefficients.

Fig. 5. Pattern recognition process (Example 1). The initial input pattern is the distorted version of the bit pattern “1”.

Fig. 6 demonstrates our second example using a different input pattern. Seven steps are required to fully retrieve one of the stored patterns. These examples verify that our proposed ONN system functions correctly, and does not lose any significant data due to discretization of system dynamics.

III. mCell: Non-Volatile Logic Device Example

Our device-level design demonstration is based on the use of a non-volatile device, the mCell, that has been recently
proposed [12]. It is based on modification of an existing spin-transfer torque magnetic tunnel junction (STT-MTJ)-based MRAM device [16] that incorporates an electrically isolating, but magnetically coupling layer between the STT switching layer and coupled free layer as shown in Fig. 7.

Fig. 6. Pattern recognition process (Example 2). The initial input pattern is the distorted version of the bit pattern “0”.

The mCell is a four terminal device comprised of a write-path (w+, w-) and an electrically-isolated read-path (R, R’) (see Fig. 7). The magnetic moments of the top magnetic electrodes are in the same direction and permanently fixed, acting as magnetic reference layer. The STT switching layer is composed of a soft magnetic metal connecting the bottom magnetic electrodes. For these electrodes, the magnetic moments at the opposite ends are oriented in the opposite direction by a pinning mechanism. A domain wall (i.e., a transition region of rotating magnetic moments) forms due to this opposite magnetization in the bottom electrodes.

Fig. 7. Schematic symbol and 2D cross section view of the mCell. The direction of the write-path current sets the position of the domain wall and thus, the read-path resistance of the mCell.

The domain wall in the STT switching layer can be moved by sending a small current pulse through write-path, utilizing an STT effect. This also programs the magnetization of coupled free layer via magnetic coupling between that layer and the STT switching layer. The domain wall has only two stable locations as shown in Fig. 7. As such, the magnetization of coupled free layer becomes only parallel or antiparallel to the magnetic moments of the top electrodes. When these two are in parallel, the read-path resistance is low and vice versa. This makes the mCell work in two stable resistance states that are determined by both direction and pulse duration of the write-path current. These resistances are set by a non-volatile magnetic polarization and therefore, remain the same even when powered off. With today’s technology, the read-path resistance changes by only a factor of 2-3x, and is limited to change less than 10x in theory [12].

IV. IMPLEMENTING LOGIC FUNCTIONS FOR PROPOSED ARCHITECTURE

Although building non-volatile logic using the mCells is challenging due to the read-path resistance change ratio as small as 2-3x with today’s material (e.g., 2.5kΩ/1.25kΩ), a complete mLogic family has been recently proposed based on novel current-steering logic [12]. Pulsed supply voltages that synchronize and power the logic circuit provide current-based logic programming signals via resistor dividers. Hence, both input and output signals of mLogic gates are of currents, not voltages. The corresponding input and output logic levels are then determined by the direction of these currents (e.g., a positive current corresponds to logic ‘1’ while a negative current corresponds to logic ‘0’).

mLogic gate examples of NAND and NOR are illustrated in Fig. 8. A unique property of this logic technology is that inversion is free, since the read-path resistance of the mCell device is determined by the direction of its write-path current. For example, referring to Fig. 8, the A and B signals can be connected to the right terminal of the mCell devices in the pull-up networks instead of connecting their inversions to the left terminal of those devices. Similarly, any other mLogic gate can be implemented without needing complementary inputs.

Moreover, the complementary mLogic gates can be obtained by simply interchanging the pull-up and pull-down networks, since the output logic level is determined by the resistance ratio of these networks. In this way, for instance, NAND and NOR gates in Fig. 8 can be converted to AND and OR gates, respectively, without requiring an inverter. It is also important to note that inverters/buffers may still be used for signal buffering and restoration, or as delay elements in this logic family.

The vertical terminals of the mCells (i.e., read-path) are characterized by the two series MTJ resistances that are formed between the top electrodes and free layer via a tunnel barrier as shown in Fig. 7. The horizontal path (i.e., write-path) for the programming current has small resistance as low as 120Ω [12]. This low input resistance represents the fanouts that are connected in series to form the current path that programs the states of the mCells as shown in Fig. 9. As such, each fanout mCell receives the same programming current, thereby preventing current shunting through
unbalanced loads.

Even with only 2-3x read-path resistance change, with proper sizing and voltage levels the programming current signals can be properly steered to drive the logic states of the fanouts [12]. For example, referring to Fig. 9, consider that the non-overlapping power clocks are positive (pClk1,+) and negative (pClk1,−) with respect to the ground shown at the end of the fanout chain. If the read-path resistance of the 

\( mCells \) with the A input to the upward arrow is high resistance, and that to the downward arrow is low resistance, then the logic signal F would be a current flowing from right to left in that figure. The direction of this current would program the states of the mCells with the F input signal.

Fig. 9. Inverter driving 3 two-input NAND gates. The fanout mCells are connected in series as highlighted by the red path. Two non-overlapping power clocks are applied to the driven and driving mLogic gates.

Moreover, efficient logic pipelining (see Fig. 10) is enabled by the inherent storage of the mCell state and the non-overlapping power clocks [12]. In each pipeline stage the mLogic gates marked by P1, and those marked by P2 are clocked with the non-overlapping pClk1,+/− and pClk2,+/− shown in Fig. 9, respectively. This results in further power reduction for our proposed associative memory because only some of the mLogic gates will dissipate power at each time interval.

Fig. 10. Pipelined mLogic gate stages. Each pipeline stage is divided into two sub-stages. The mLogic gates in the first stages and second stages are clocked with two non-overlapping power clocks for efficient power savings.

It is possible to integrate CMOS transistors with the mCells to form a hybrid non-volatile logic family, but this eliminates some of the aforementioned energy-saving benefits. Moreover, there are integration issues that make this costly and impractical, since tightly integrating the mCells and mLogic on top of CMOS would consume routing resources that are needed for all of the artificial synapses in the oscillatory associative memory architecture. Instead, we envision a fully-functional, mLogic-based, oscillatory associative memory chip that can be layered or stacked (using through-silicon vias) with CMOS only for power, clocking and input/output (I/O).

V. CIRCUIT-LEVEL COMPARISON

We constructed our proposed ONN architecture (see Fig. 3) with CMOS and mLogic based on circuit-simulation modeling for both. We assumed a fully-interconnected 60-neuron system to comply with the behavioral simulation presented in Section II.C. Similarly, 5-bit signed binary numbers were chosen for inputs and weights.

For the CMOS implementation we first generated a VHDL (VLSI hardware description language) based model for an oscillatory neuron and then compiled it using a logic synthesis tool (Synopsys Design Compiler) with 32nm CMOS logic library that completed the circuit optimization based on timing targets we provided. We verified the implementation of the same ONN architecture based on mLogic with +/-12.5mV supplies for buffers and multiplexers, and +/-50mV supplies for other logic gates using a circuit simulation with a physics-based device model for the mCells [12].

In both implementations each neuron consists of two major blocks: i) a computation block (CB) that performs neural computation; and ii) an interconnection block (IB) that stores the weights and inputs them to the computation block in an orderly manner. With our 60-neuron system example for each neuron the CMOS implementation with 0.7V requires 2168 transistors and 87.4µW power for CB, and 17208 transistors and 581µW power for IB. Nevertheless, for the same system the mLogic implementation per neuron requires 2238 mCell devices and 573µW power for CB, and 1200 mCell devices and 33µW power for IB. Hence, for a 60-neuron system the mLogic implementation represents a 5.6x improvement in the number of devices required while consuming a comparable power as CMOS implementation. With the increase in the number of neurons, however, interconnections would dominate due to the fully-interconnected architecture. Hence, for large-scale oscillatory associative memories the improvement with mLogic would reach approximately 15x for the number of devices required and 18x for power as shown in Fig. 11. Moreover, these comparisons do not consider the potential benefits of layout and stacking of this all-magnetic logic technology.

It is important to note that both the simulated mCell and a 32nm CMOS transistor occupy roughly the same footprint area. Thus, the device count comparison here represents the area comparison for both implemented architectures. Since the area comparison is also affected by how well each technology scales, a comparison based on device count better demonstrates the corresponding circuit complexities for both technologies.
VI. CONCLUSION

In this paper we propose an oscillatory associative memory architecture based on digitized oscillators and phase relationships among oscillatory neurons that can be efficiently implemented using non-volatile logic. The proposed architecture offers efficient programmability and high throughput via fully-pipelined logic stages that exploit the inherent storage. We specifically demonstrate the use of mLogic to construct this neurocomputing system and compare it with a 32nm CMOS implementation. The comparison results demonstrate that this new all-magnetic logic technology could enable the practical implementation of low-voltage, low-power, and nanoscale oscillatory associative memories.

REFERENCES


