Implementation of linear differential equations using pulse-coupled oscillators with an ultra-low power neuromoprhic realization

Jafar Shamsi, Wilten Nicola

***Abstract*—Pulse-coupled oscillators (PCOs) are used as models for oscillatory systems in diverse fields such as biology, physics, and engineering. When correctly coupled, PCOs can display sophisticated emergent dynamics for large numbers of oscillators. Here, we propose an algorithm and hardware implementation of PCOs to emulate arbitrary systems of linear differential equations (DEs) with inputs, which are similar to the equations used in feedback control laws or linearizations of nonlinear systems. We show that m populations of oscillators can solve a set of m-dimensional linear DEs with simple coupling schemes, and crucially, without the matrix multiplications required in Euler integration. The emergence of linear dynamical systems in networks of PCOs occurs when the number of oscillators within a population becomes large through an analytically exact mean-field derivation. In addition, a hardware architecture of PCOs for digital implementation is proposed and realized on an ultra-low power FPGA as a proof of concept. These results show that there are simple coupling schemes for networks of pulse-coupled oscillators that collectively compute complex dynamical systems. These PCO networks also have an immediate implementation as low power neuromorphic edge devices.**

***Index Terms*—Pulse coupled oscillators, linear differential equations, Fokker-Planck approximation, Neuromorphic, FPGA.**

I. Introduction

DIFFERENTIAL equations (DEs) act as models for physical, chemical, and biological systems at different scales. From the population of microbes in a petri dish, to the evolution of galactic superclusters, DEs model and simulate systems large and small. DEs are also the backbone of control theory, as simple linear dynamical systems can stabilize complex nonlinear dynamics through feedback control [1]. DEs can also sometimes offer exact solutions through closed-form approaches or real-time solutions using numerical integration methods [2]. However, as the scale of a system increases, the time or space complexity in simulating a system of differential equations often grows super-linearly. For example, in simulating an n-body Newtonian system, n^2 non-linear interactions must be computed at every time step in Euler integration. One method of reducing the simulation time is to configure specialized hardware to solve specific kinds of differential equations by efficiently computing the terms required for simulating a DE.

One type of hardware acceleration is based on the collective behavior of simple distributed elements, pulse-coupled oscillators (PCOs), to approximate the solutions of more complex systems through their interactions with each other. These pulse-coupled oscillators mimic the phenomenological function of neurons [3], or other oscillatory computing units. There are some potential advantages of using PCOs for solving DEs, including parallelism and distributed computing for efficient utilization of computing resources and faster solution times for large-scale problems. On the hardware front, compact and power-efficient oscillators can be implemented using Vanadium Dioxide (VO_2) devices [4], which enable the realization of energy-efficient coupled oscillators [5]–[8]. While emerging devices such as VO_2 are not broadly and commercially available yet, the realization of PCOs based on standard CMOS technologies is a practical solution [9]. Digital implementation of PCOs with a parallel architecture is an alternative paradigm that allows realization of PCOs on off-the-shelf FPGAs [10].

Despite the hardware implementation paradigm, determining the exact coupling between PCOs to collectively simulate an arbitrary dynamical system is a hurdle in an efficient hardware-based simulator. Here, we show that for an m-dimensional linear system of DEs, m populations of PCOs can be used to approximate the solution of the original system. This result is formally derived through the Fokker-Planck equations and mean-field approximation in the limit that the number of oscillators per population becomes large. In this regime, the coupling dynamics between these m populations follow a linear system of DEs. This analytical derivation was verified with simulations, and as a proof of concept, the digital implementation of PCOs that solve linear DEs is considered in this paper. Critically, the PCO implementation does not require matrix multiplication to approximate the solution of a system of linear DEs, thereby allowing for efficient hardware implementation. The linear systems considered here can be used as either ultra low power controllers in wearable devices, or as filters for bio-signals. To verify this result, we developed a digital hardware architecture of PCOs with a pipeline design and resource-sharing method to minimize hardware resource usage while increasing the throughput of the system. Finally, an ultra-low power FPGA board was fabricated, and the design was synthesized and implemented on it. The power consumption of the fabricated device was as little as 6.6 mW in its operational phase, showcasing the applicability of using PCO networks for wearable and neuromorphic edge computing devices.

The remaining sections of the paper are organized as

Jafar Shamsi and Wilten Nicola are part of the Hotchkiss Brain Institute, University of Calgary, Canada, and the of Department of Cell Biology and Anatomy, University of Calgary, Calgary, Alberta, Canada.

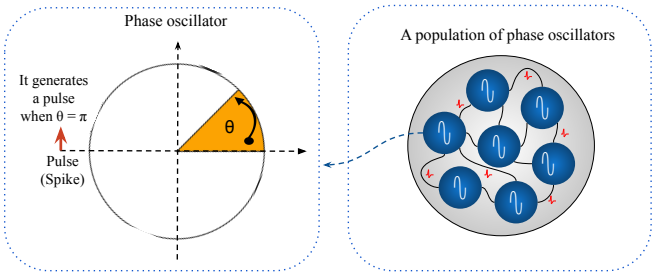


Fig. 1: (Left) A phase oscillator with phase θ. This oscillator is assumed to have a constant phase velocity that induces a counterclockwise oscillation around the ring. The oscillator generates a "spike" when $\theta = \pi$. This spike is typically convolved to create a pulse. (Right) A population of PCOs interacting with each other in a recurrently coupled network. The PCOs can implement dynamical systems collectively through the generation of spikes, and the transmission of pulses (filtered spikes) to each other.

follows: Section II provides the background information about PCOs. The technique for implementing linear DEs is presented in Section III. Results from the simulation and hardware implementation are presented in Section IV, which is followed by a conclusion.

II. Background

Pulse-Coupled Oscillators (PCOs) PCOs are a class of dynamical systems that replicate the phenomenological behavior of oscillatory biological and physical systems such as neurons [11] and fireflies [3] or simple pendulums and other physical systems [12]–[15]. Interactions between the oscillators are based on pulses at discrete time intervals, enabling them to synchronize or transmit information.

Pulse-coupled oscillators (PCOs) are simplified limit cycle oscillators that are connected through weighted connections and communicate using pulses (spikes in the case of neurons). The dynamics of each oscillator are described by

$$\frac{d\theta_i}{dt} = \omega_i \tag{1}$$

where θ_i and ω_i are the phase and frequency (or phase velocity) of the ith oscillator, respectively. As shown in Fig. 1, the oscillator generates a spike when the phase of the oscillator crosses π, although the crossing point that denotes a spike being fired is arbitrary. For this work, we will consider a "spike" as being fired when θ crosses 2π.

A population of pulse-coupled oscillators is formed by connecting phase oscillators through neuronal-like synaptic connections, where every oscillator generates a pulse when a spike is fired (Fig. 1). A single, self-coupled population of PCOs is described by

$$\frac{d\theta_i}{dt} = \omega_i + g_i(t), \tag{2}$$

$$\frac{dg_i(t)}{dt} = -g_i(t) + \frac{1}{N}\sum_{j=1}^{N} w_{ij} \sum_{t_{jl}<t} \delta(t - t_{jl}) \tag{3}$$

where $g_i(t)$ is the synaptic dynamics that translates input pulses from other oscillators (δ) into a temporal frequency change of oscillator i. The function $\delta(t - t_{jl})$ is the delta function which increments g_i by w_{ij} when oscillator j crosses 2π. This occurs at the time point t_{jl} which corresponds to the lth spike fired by the jth neuron. The parameter w_{ij} is the synaptic weight from oscillator j to oscillator i and effectively weights the pulse size. If $w_{ij} > 0$, then the connection between oscillator j and i is excitatory. If $w_{ij} < 0$, then this connection is inhibitory. The parameter N acts as the number of PCOs in the population.

When we consider (3), an important insight emerges: we never have to multiply the weight matrix w_{ij} by any quantity. The variable $g_i(t)$ is instead incremented (increased or decreased) by the weight w_{ij} when oscillator j crosses a threshold or fires a "spike". We will exploit this fact to approximate solutions to linear differential equation

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{c}(t) \tag{4}$$

without taking the matrix multiplication $\boldsymbol{A}\boldsymbol{x}$ in every time step. This multiplication is required by Euler integration for example as

$$\boldsymbol{x}(t + \Delta) \approx \boldsymbol{x}(t) + \Delta(\boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{c}(t)) \tag{5}$$

and is also required by higher-order integration schemes. For large systems, these matrix multiplications can be very costly as if $\boldsymbol{x}$ is an $m \times 1$ variable, the matrix multiplication $\boldsymbol{A}x$ requires somewhere between $\mathcal{O}(m^2)$ and $\mathcal{O}(m^3)$ operations at each time step, depending on the specific numerical algorithm used for a general A, although this can be reduced if the matrix A is sparse. We also remark here that the system in equation (4) can also be used as a linear controller for nonlinear systems, when $\boldsymbol{A}$ and $\boldsymbol{B}$ are chosen appropriately [1], [16].

III. Methods

In the limit that $N \to \infty$ for system (3), one can show that the dynamics of PCOs are those of a linear system of ordinary differential equations, under reasonable assumptions on the weights, w_{ij}. This will be the basis of our construction of networks of PCOs that solve linear differential equations.

A. A One-Dimensional Linear DE Emerges from a Population of PCOs

Suppose that there is one population of N pulse-coupled oscillators described by (3) where the frequency and synaptic weights of the oscillators are identical to ω and W, respectively. That is, all the oscillators are coupled with the same weight, $W_{ij} = W$, $\forall i, j$ and all the oscillators have the same intrinsic frequency, $\omega_i = \omega$, $\forall i$. This results in

$$\frac{d\theta_i}{dt} = \omega + g(t) \tag{6}$$

$$\frac{dg(t)}{dt} = -g(t) + \frac{W}{N}\sum_{j=1}^{N} \sum_{t_{jl}<t} \delta(t - t_{jl}) \tag{7}$$

where neuron j emits spikes at t_{jl}. In the limit that this system is large, we no longer need to consider the phase $\theta_i(t)$ of any particular oscillator, and can instead consider the density function of phases $\rho(\theta,t)$, and the Fokker-Planck system that describes the systems evolution [17]–[20] and has been extensively applied to PCOs in the form of Integrate-and-fire neurons [21]–[24] (see Appendix 1):

$$\frac{\partial\rho}{\partial t} = -(\omega+g(t))\frac{\partial\rho(\theta,t)}{\partial\theta} \quad (8)$$

$$= -\frac{\partial J(\theta,t)}{\partial\theta} \quad (9)$$

$$\frac{dg}{dt} = -g + WJ(2\pi,t) \quad (10)$$

where $J(\theta,t) = (\omega+g(t))\rho(\theta,t)$ is typically referred to as the flux [20]–[24]. Due to the oscillatory nature of the PCOs, the Fokker-Planck system also has the following boundary conditions:

$$J(2\pi,t) = J(0,t). \quad (11)$$

This boundary condition is applied when the oscillators are in the explicitly oscillatory regime $\omega + g(t) > 0$, $\forall t > 0$. Fortunately, for the PCOs considered here, this system has an exact solution in the form of

$$\rho(\theta,t) = \rho_0\left(\text{mod}\left[\theta - \omega t - \int_0^t g(t')\,dt', 2\pi\right]\right) \quad (12)$$

where $\rho_0(\theta) = \rho(\theta,0)$ is the initial oscillator density when $\omega + g(t) > 0$, $\forall t$.

The dynamics of $g(t)$ are determined both by the initial density function $\rho_0(\theta)$ and the coupling parameter W:

$$\begin{aligned}\frac{dg(t)}{dt} &= -g(t) + WJ(2\pi,t)\\ &= -g(t) + \\ &W(\omega+g(t))\rho_0\left(\text{mod}\left[2\pi - \omega t - \int_0^t g(t')\,dt', 2\pi\right]\right)\end{aligned} \quad (13)$$

where mod is the modulus function. Considering (13), if we initialize the system with the uniform distribution of phases $\rho_0(\theta) = \frac{1}{2\pi}$, we have the following, analytically exact mean-field system:

$$\frac{dg(t)}{dt} = -g(t) + W\left(\frac{\omega+g(t)}{2\pi}\right) \quad (14)$$

$$= \left(\frac{W}{2\pi} - 1\right)g(t) + \frac{W\omega}{2\pi}. \quad (15)$$

The $\frac{\omega}{2\pi}$ term is also easily eliminated with the substitution $x = g - g_0$, which results in the final linear dynamical system equation as follows:

$$\frac{dx}{dt} = Ax \quad (16)$$

where

$$A = 2\pi\,(W+1) \quad (17)$$

$$g_0 = -\left(\frac{W}{2\pi} - 1\right)^{-1}\frac{W\omega}{2\pi}. \quad (18)$$

To summarize, for a simple network of PCOs, the initial distribution $\rho_0(\theta)$ for a population is critical in influencing the dynamics in perpetuity. By using a uniform initial distribution of phases, the resulting large-system dynamics of $g(t)$ are linear and pre-configurable by choosing the correct weights and driving frequencies. The parameter A can be configured with

$$W = \frac{A}{2\pi} - 1 \quad (19)$$

The equations are easily extended when there are m populations of oscillators and they receive an input $c(t)$:

$$\frac{d\theta_i^k}{dt} = \omega + \sum_{k'=1}^{m} g^{kk'}(t) + \sum_{q=1}^{d}\tilde{W}_{kq}c_q(t) \quad (20)$$

$$= \omega + \tilde{g}^k(t) + \sum_{q=1}^{d}\tilde{W}_q c(t) \quad (21)$$

$$\frac{dg^{kk'}(t)}{dt} = -g^{kk'}(t) + \sum_{j=1}^{N}\sum_{t_{jl}<t}\frac{W_{kk'}}{N}\delta(t - t_{jl}^{k'}) \quad (22)$$

where $g^{kk'}(t)$ describes the history of pulses generated by population k', weighed and transmitted to population k by $W_{kk'}$. The term $\tilde{g}^k(t) = \sum_{k'=1}^{m} g^{kk'}(t)$. Note that in equation (22), as the delta function is used, there is no multiplication of the weight matrix $W_{kk'}$ with $g^{kk'}$. Instead, $g^{kk'}(t)$ is incremented by $W_{kk'}$ every time an oscillator in population k' fires a "spike". In the mean-field limit, we have the following equation for $\tilde{g}^k(t)$

$$\begin{aligned}\frac{d\tilde{g}^k(t)}{dt} &= -\tilde{g^k} + \sum_{k'=1}^{m} W_{kk'}J_{k'}(2\pi,t)\\ &= -\tilde{g}^k + \\ &\frac{1}{2\pi}\sum_{k'=1}^{m} W_{kk'}(\omega + \tilde{g}^{k'} + \sum_{q=1}^{d}\tilde{W}_{k'q}c_q(t))\end{aligned} \quad (23)$$

which yields

$$\frac{d\tilde{\boldsymbol{g}}}{dt} = \left(\frac{\boldsymbol{W}}{2\pi} - \boldsymbol{I}_m\right)\tilde{\boldsymbol{g}}(t) + \boldsymbol{W}\frac{\boldsymbol{\omega}}{2\pi} + \boldsymbol{W}\tilde{\boldsymbol{W}}\boldsymbol{c}(t) \quad (24)$$

which can be rewritten into an m-dimensional LDEs:

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{c}(t) \quad (25)$$

with the following:

$$\boldsymbol{A} = \frac{\boldsymbol{W}}{2\pi} - \boldsymbol{I}_m \rightarrow \boldsymbol{W} = 2\pi(\boldsymbol{A} + \boldsymbol{I}_m), \quad (26)$$

$$\boldsymbol{B} = \boldsymbol{W}\tilde{\boldsymbol{W}} \rightarrow \tilde{\boldsymbol{W}} = \boldsymbol{W}^{-1}\boldsymbol{B}. \quad (27)$$

Note that as before, the $\boldsymbol{\omega}/2\pi$ term is removable with the substitution

$$\boldsymbol{x} = \tilde{\boldsymbol{g}} - \boldsymbol{g}_0 \quad (28)$$

where

$$\boldsymbol{g}_0 = -\boldsymbol{A}^{-1}\boldsymbol{W}\frac{\boldsymbol{\omega}}{\boldsymbol{2\pi}} \quad (29)$$

where $\boldsymbol{\omega}$ is now a $m \times 1$ vector consisting of $\omega_k = \omega$, $k = 1, 2, \ldots m$. While computing $\boldsymbol{g}_0$ does require matrix multiplication (and inversion), this operation only needs to be

performed once with $\boldsymbol{g}_0$ being stored after. This is in contrast to the $\boldsymbol{A}\boldsymbol{x}(t)$ multiplication required at every time step for approximating (25) with Euler integration.

Collectively, these analytical results demonstrate that in the large PCO limit, there exist configurations of networks of these PCOs that exactly implement linear differential equations. Importantly, when implemented as a finite oscillator set, these systems do not require matrix multiplication to implement (aside from the single instance for $\boldsymbol{g}_0$) and only require increasing or decreasing the mean-field variables $\boldsymbol{g}$ by the weights in the network. Next, we consider the hardware implementation of these systems.

B. Hardware Architecture

Pulse-coupled oscillators can be implemented using both analog [9] or digital circuits [10]. In this section, a digital hardware architecture of PCOs is introduced which can be used to implement the coupling scheme in the preceding section.

First, we used the Euler method to discretize the PCO differential equations:

$$\theta_i^k[n+1] = \theta_i^k[n] + \Delta\left(\omega + \tilde{g}^k[n] + \sum_{q=1}^{d} \tilde{W}_{kq} c_q[n]\right) \quad (30)$$

$$g^{kk'}[n+1] = g^{kk'}[n] - \Delta g^{kk'}[n] + \frac{W_{kk'}}{N} \sum_{j=1}^{N} \sum_{n_{jl}<n} \delta[n - n_{jl}^{k'}]) \quad (31)$$

where n and Δ are the time and time-step duration. The term $\tilde{g}^k[n] = \sum_{k'=1}^{m} g^{kk'}[n]$.

Fig. 2 shows a hardware architecture of PCOs with m populations. As a typical digital system, it includes a control unit, a data-path unit, a memory unit, and interface signals. The memory unit stores the weights and the control unit controls the flow of the computation of the data-path unit.

1) Memory unit: A memory is used to store the input weights ($\tilde{W}_{kq}$) and the coupling weights ($W_{kk'}$) within a population ($k = k'$) and between the populations ($k \neq k'$). All weights are organized in one memory address and they are always available without addressing. Therefore, the weights can be used in computations in parallel causing an increase of the processing speed. The memory is divided into m sections each belonging to a population. For instance, the weights for the first population are stored from the first bit to bit $m.l_w$, where m and l_w are the number of populations and the bit precision for weights, respectively (the first l_w bits are used for $W_{1,1}$, the second l_w bits are used for $W_{1,2}$, and so on).

The input weights are stored at the end of each section. For instance, if the input is a one dimensional signal, the last l_w bits of each section are used to store the corresponding weight. In total the memory capacity is $(m^2 + d).l_w$ bits, where m and d are the number of populations and the input dimension, respectively. Therefore, the space complexity of the memory with respect to the dimension of DEs is $\mathcal{O}(n^2)$. Since the weights are organized in one memory address, the time complexity of the reading the weights is $\mathcal{O}(1)$. In this architecture, one-dimensional input is considered ($q = 1$) requiring a memory with the capacity of $(m^2 + 1).l_w$ bits.

2) Data-path unit: The data-path unit implements the computations of the PCOs. For each population, several computational blocks are used including m multiplexers, an adder tree, an accumulator, a coupling block, and an oscillator block.

There are m multiplexers corresponding to m weights per population where their selector are connected to the output of the oscillator blocks. The multiplexers pass either a weight value or a zero value to the adder tree depending on the spikes generated by oscillators. If a spike is generated the value of the selector becomes 1 and the multiplexer passes the corresponding weight to the adder tree. Otherwise, it sends value 0 to the adder tree. The output of the multiplexer computes $W_{kk'}/N\delta[n-n_{jl}^{k'}]$ in (31). The total multiplexers of this architecture is m^2 and the space complexity of the multiplexers with respect to the dimension of DEs is $\mathcal{O}(n^2)$. As a combinational circuit, the time complexity of the calculations by the multiplexers is $\mathcal{O}(1)$.

An adder tree is a parallel computing architecture that efficiently computes the sum of multiple numbers by recursively adding pairs of numbers until a final sum is obtained. For each population, there is one adder tree with a pipeline design and an accumulator to calculate $\sum_{j=1}^{N}\sum_{n_{jl}<n} W_{kk'}/N\delta[n-n_{jl}^{k'}]$ in (31). Given m inputs, the number of stages and the total number of adders are $[log_2(m)]$ and $([log_2(m)] + 1)$, respectively. Therefore, the space complexity of adders with respect to the dimension of DEs is $\mathcal{O}(log(n))$. Although the number of pipeline stages in the adders increases by increasing the dimension of DEs, the time complexity is $\mathcal{O}(1)$ due to the pipeline design.

The coupling block and oscillator block implement the rest part of (31) and (30), respectively. Fig. 2 (Right) shows the implementation of the coupling dynamics and oscillators. Since a large number of oscillators are required to approximate DEs, it is crucial to use techniques to reduce the hardware resource usage. In this regards, resource sharing was used in which one coupling block and oscillator block were used per population [25], [26]. The cost of resource sharing is reducing the throughput by the order of N - the number of oscillators in a population.

To compensate for the throughput, a 3-stage and a 6-stage pipeline design were used to implement the coupling dynamics and oscillators, respectively (Fig. 2 (Right)). The pipeline architecture breaks the combination circuits into smaller parts to increase the throughput of the system at the cost of a small increase of the total circuit latency and consumption of a few registers [26], [27]. At each clock cycle the calculations of one coupling dynamic per population is computed and is fed into the oscillator block, as well as stored in the Random Access Memory (RAM) to be used in the next time step. It is worth noting that considering (30), at each time step the same value of the coupling dynamic ($\tilde{g}^k[n]$) is used for all oscillators in a population, thus only one memory location is required to store the state variable of coupling dynamic for the next time step. However, a RAM with N memory locations is used for the sake of the generality of the design for the future applications,

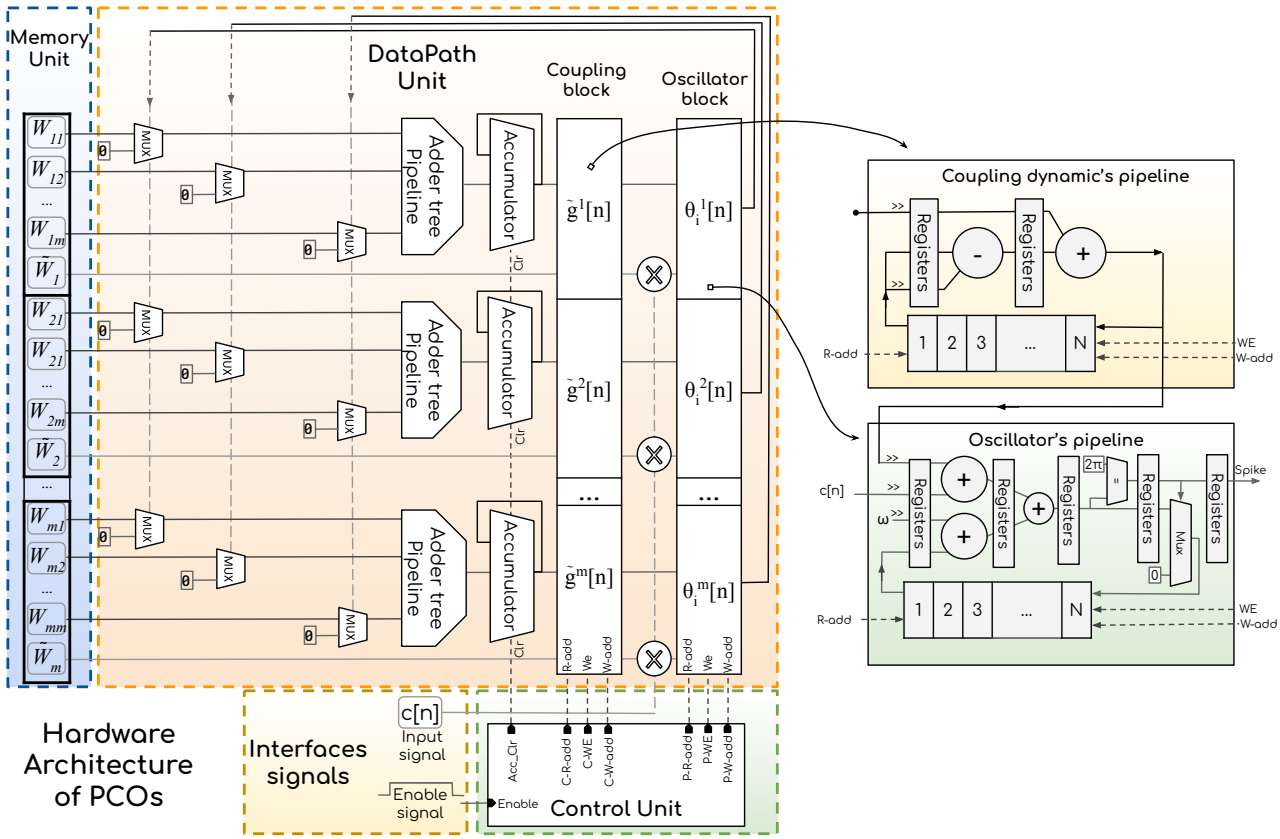


Fig. 2: (Left) Hardware architecture of a PCO network with m populations. The populations are implemented in parallel. (Right) The dynamics of the coupling dynamics and oscillators in each population are calculated sequentially using a pipeline design with a resource-sharing method. Each block includes a computational block and memory. The coupling and population blocks perform the computations and the results are stored in RAMs for the next time step.

where each oscillator has its own coupling dynamic. Moreover, it allows for the design of a modular control unit which is discussed in section III-B3.

By using the pipeline design and resource sharing method, the space complexity and time complexity of the coupling and oscillator blocks with respect to the dimension of DEs are $\mathcal{O}(n)$ and $\mathcal{O}(1)$, respectively.

3) Control unit: The control unit is used to control the flow of the computations in the data-path unit, specifically, controlling the read/write operation of the RAMs and clearing the accumulator output. One possible control unit implementation is shown in Fig. 3. The control unit is based on a modular block, counter module (CM), comprising a counter and two comparators. The counter is enabled through a pulse with the duration of at least one clock cycle at the ON pin and stays enabled until it receives a pulse at the OFF pin in which it is disabled and the counter value resets to zero. In this block, the counter is always disabled when the counter value becomes $N-1$. The EN pin is 1 when the counter is enabled.

The read/write operation of the RAMs in the coupling and oscillator blocks is performed sequentially. Therefore, the output of one CM block (CM1) is used to generate the read address (C-R-add) of the RAM in the coupling block and it is also used as a reference block to trigger the other CM blocks. Theoretically, the write operation of the RAM in the coupling block should be concurrent with the read operation of the oscillator block, since they have the same input; Thus the corresponding blocks (CM2 and CM3) should use the same value (A=B) as the comparison value. However, depending on the read/write operation of the RAM block these values can be different; Specifically, in FPGAs, it usually takes two clock cycles after providing the address for the RAM memory to make the content of the memory available, thus the read operation of the RAM in the oscillator block should start two clock cycles before the write operation in the coupling block. The next step is the write operation of the RAM in the oscillator block, followed by the clear operation of the accumulator. The clear operation of the accumulator is necessary to start computing a new value for the next time step. Values A, B, C, and D are related to the number of pipeline stages in the coupling block (C_{stage}), oscillator block (P_{stage}), and adder tree Block (A_{stage}): $A = C_{stage} - 1, B = C_{stage} + 1, C = C_{stage} + P_{stage} + 1, D = C_{stage} + P_{stage} + A_{stage} + 1$.

The total number of clock that are required for the calculations in one time step depends on the control unit. In this architecture, the number of clocks per time step is determined by

$$N_{clk} = N + C_{stage} + P_{stage} + A_{stage}, \tag{32}$$

which is independent of the dimension (m) of the differential

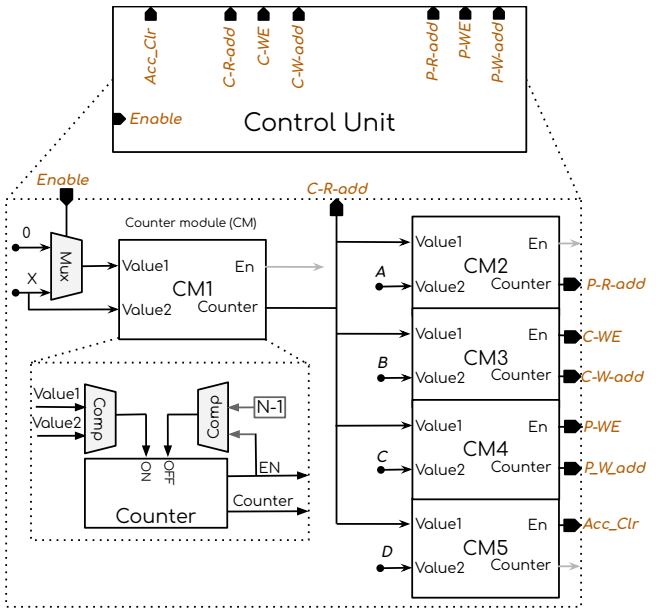


Fig. 3: Modular architecture of the control unit based on a counter module (CM). The counter module comprises two comparator and a counter.

equations showing that the time complexity is $\mathcal{O}(1)$.

The time and space complexity of the proposed architecture is summarized in Table I. Considering the hardware architecture, the maximum space complexity belongs to the weight memory, which is $\mathcal{O}(n^2)$, while the time complexity of the architecture is $\mathcal{O}(1)$.

IV. Results

Here, we compare the results of the analytical derivations, the simulated PCO system and the results of the hardware implementation on the FPGA.

A. Simulation results

An m-dimensional linear differential equation can be implemented using m populations of PCOs. Here one-, two- and 20-dimensional DEs are simulated and the results are compared with the simulated PCOs. To implement a one-dimensional linear differential equation, consider one population of PCOs with an input $c(t)$

$$\frac{d\theta_i}{dt} = \omega + g(t) + \tilde{W}c(t) \tag{33}$$

$$\frac{dg}{dt} = -g + \frac{W}{N}\sum_{t_{jk}<t}\delta(t-t_{jk}). \tag{34}$$

The term $\tilde{W}$ can be interpreted as an input weight for the input $c(t)$ while W is an all-to-all coupling strength. Then, we have the following linear mean-field system:

$$\frac{dg}{dt} = -g\left(1-\frac{W}{2\pi}\right) + \frac{W\omega}{2\pi} + \frac{W\tilde{W}c(t)}{2\pi}. \tag{35}$$

Suppose we want this system to mimic:

$$\frac{dx}{dt} = -Ax + c(t) \tag{36}$$

Then, we require the following:

$$W = 2\pi(1-A), \quad \tilde{W} = \frac{2\pi}{W} \tag{37}$$

As this yields

$$\frac{dg}{dt} = -Ag + c(t) + \frac{W\omega}{2\pi} \tag{38}$$

The substitution $x(t) = g(t) - \frac{W\omega}{2\pi A}$ yields:

$$\frac{dx}{dt} = \frac{dg}{dt} = -Ag(t) + c(t) + \frac{W\omega}{2\pi} \tag{39}$$

$$= -A\left(x(t) + \frac{W\omega}{2\pi A}\right) + c(t) + \frac{W\omega}{2\pi} \tag{40}$$

$$= -Ax(t) + c(t) \tag{41}$$

Note that the substitution $x(t) = (g(t) - \frac{W\omega}{2\pi A})$ implies that $x(0) = g(0) - \frac{\omega}{2\pi A}$. Thus, with an all-to-all coupling strength of $W = 2\pi(1-A)$ and an input coupling strength of $B = 2\pi/W$, the network collectively implements the dynamics of $x' = -Ax + Bc(t)$. An example with $N = 512$ oscillators with a random input $c(t)$ (Fig. 4a), and $\omega = 250$, with $A = 1.06$ and $B = 0.72$ is shown in Fig. 4b, with the simulated PCO network correctly tracking the target dynamics. Random input $c(t)$ was generated by

$$c(t) = \frac{1}{a}\left(r(t) - \mu_{r(t)}\right)/\sigma_{r(t)} \tag{42}$$

where a is an integer value and

$$r(t) = \sum_{k=1}^{a} cos(\alpha_k \pi t). \tag{43}$$

The values $\mu_{r(t)}$ and $\sigma_{r(t)}$ are the mean value and the standard deviation of the signal r(t). In the simulations, $a = 5$ and α_k was generated randomly with a normal distribution.

To implement a two-dimensional linear DE, two populations of PCOs are used. Consider the following pair of pulse-coupled oscillators with an input $c(t)$

$$\frac{d\theta_i^1}{dt} = \omega + \sum_{k'=1}^{2} g^{1k'}(t) + \tilde{W}c(t) \tag{44}$$

$$\frac{d\theta_i^2}{dt} = \omega + \sum_{k'=1}^{2} g^{2k'}(t) \tag{45}$$

$$\frac{dg^{kk'}}{dt} = -g^{kk'} + \sum_{j=1}^{N}\sum_{t_{jk}<t}\frac{W_{kk'}}{N}\delta(t-t_{jl}) \tag{46}$$

All the oscillators in each population have the same ω. The term $\tilde{W}$ can be interpreted as an input weight for the input $c(t)$ while $\boldsymbol{W}$ is an all-to-all coupling strength between oscillators in each population. The mean-field dynamics for this network are:

$$\frac{d\tilde{\boldsymbol{g}}}{dt} = -\tilde{\boldsymbol{g}} + \frac{\boldsymbol{W}}{2\pi}\left(\tilde{\boldsymbol{g}} + \begin{pmatrix}\tilde{W}\\0\end{pmatrix}c(t) + \begin{pmatrix}\omega\\\omega\end{pmatrix}\right) \tag{47}$$

Suppose we want this system to mimic the two-dimensional system

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x} + \begin{pmatrix}\tilde{W}\\0\end{pmatrix}c(t) \tag{48}$$

TABLE I: The space and time complexity of the proposed hardware architecture.

	Weight Memory	Multiplexers	Adder-Tree	Copuling block	Population block	Overall
Space complexity	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$	$\mathcal{O}(log(n))$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Time complexity	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

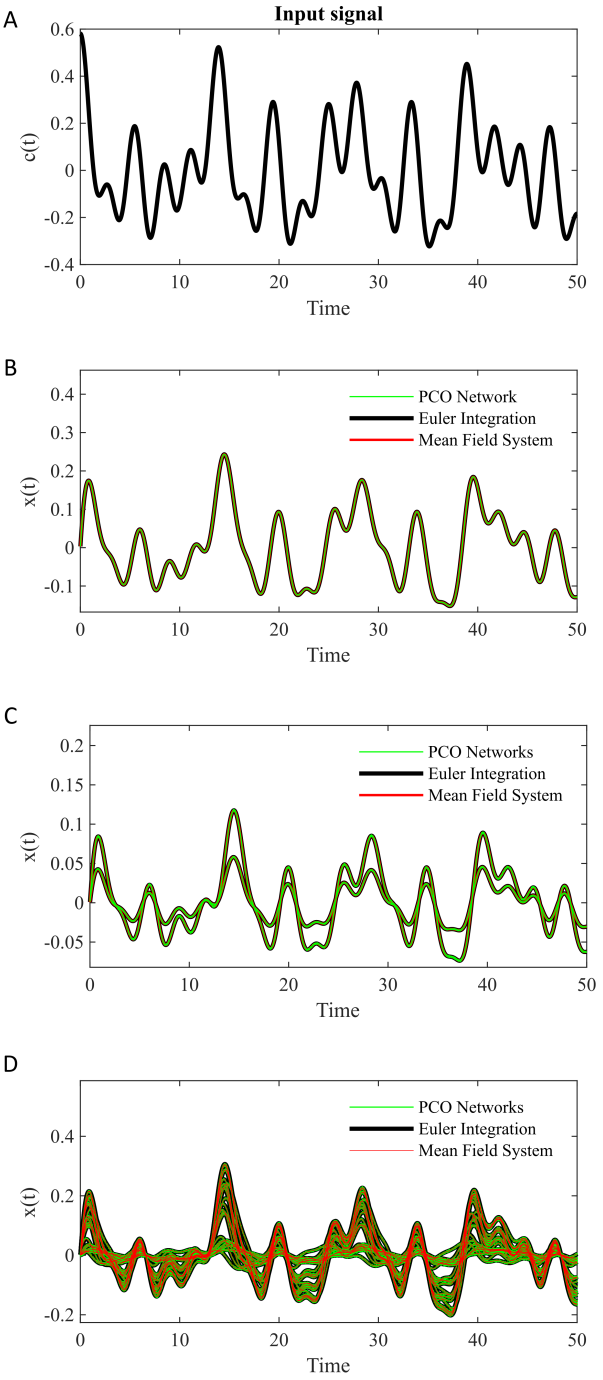


Fig. 4: Simulated PCO networks mimic linear dynamical systems. (a) A random input signal is applied to a single PCO network in (b), a two coupled PCO networks in (c), and a 20 coupled PCO networks in (d). The linear dynamics of the PCO coupled networks in (b)-(d) (green) are analytically determined (red, mean-field), and match the Euler integration solution for the corresponding system $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{B}c(t)$

Then, we require the following:

$$\boldsymbol{W} = 2\pi(\boldsymbol{A} + \boldsymbol{I}_2) \quad (49)$$

$$\tilde{W} = 2\pi \quad (50)$$

along with the substitution $\boldsymbol{x}(t) = \boldsymbol{g}(t) - \boldsymbol{g}_0$ where

$$\boldsymbol{g}_0 = -\boldsymbol{A}^{-1}\boldsymbol{W}\begin{pmatrix}\frac{\omega}{2\pi}\\ \frac{\omega}{2\pi}\end{pmatrix} \quad (51)$$

Note that the substitution $\boldsymbol{x}(t) = \boldsymbol{g}(t) - \boldsymbol{g}_0$ implies that

$$\boldsymbol{x}(0) = \boldsymbol{g}(0) + \boldsymbol{A}^{-1}\boldsymbol{W}\begin{pmatrix}\frac{\omega}{2\pi}\\ \frac{\omega}{2\pi}\end{pmatrix} \quad (52)$$

An example with $N = 512$ oscillators (in each population) with a random input $c(t)$, and $\omega = 250$, with $A = \begin{pmatrix}-1.06 & -0.08\\ -0.11 & -1.1\end{pmatrix}$ is shown in Fig. 4c.

To show the scalability of the method, we also considered a 20-dimensional system of linear differential equations (Fig. 4d) when $\omega = 250$ and matrix A was generated randomly by

$$A = \boldsymbol{I}_m + \alpha \times R_{m,m} \quad (53)$$

where $\boldsymbol{I}_m$ and $R_{m,m}$ are the identity and the random matrices, respectively. The elements of $R_{m,m}$ were randomly drawn from a normal distribution. Parameter α is 0.1 in this simulation. This system also receives an input $\mathbf{B}c(t)$, where $c(t)$ is the input signal and $\mathbf{B}$ is randomly generated with a normal distribution.

B. Hardware implementation results

The proposed hardware architecture with three populations was implemented by VHDL (VHSIC Hardware Description Language) and synthesized for an ultra low-power FPGA (iCE40UP5K) from Lattice Semiconductor. In the following subsections, the details of the implementations are provided.

1) FPGA emulation results: The implemented architecture on the FPGA was capable of emulating up to three populations of PCOs with 512 oscillators in each population. The frequency of oscillators was $\omega = 15$. The emulation results of a one-, two- and three-dimensional differential equations are shown in Fig. 5a-b, Fig. 5c, and Fig. 5d, respectively. The input signal is a random signal generated by (42) and matrix A in the DEs was generated randomly using (53) where $\alpha = 0.5$.

2) Resource usage: The resource usage of the synthesized hardware for the FPGA (iCE40UP5K) is provided in Table II. The usage number of 4-input Look-Up-Tables (4-LUTs) and 4 kbit Block-RAMs (BRAMs) were 1221 (23 %) and 26 (93 %), respectively. Eight BRAMs were used to store the weights with 10-bit precision in a fixed-point format. Nine BRAMs for the coupling and oscillator blocks were used to store the oscillators' and couplings' state variables with the 24-bit precision. All other variables and values are stored in 24-bit registers in a fixed-point format. In [10], a PCO of

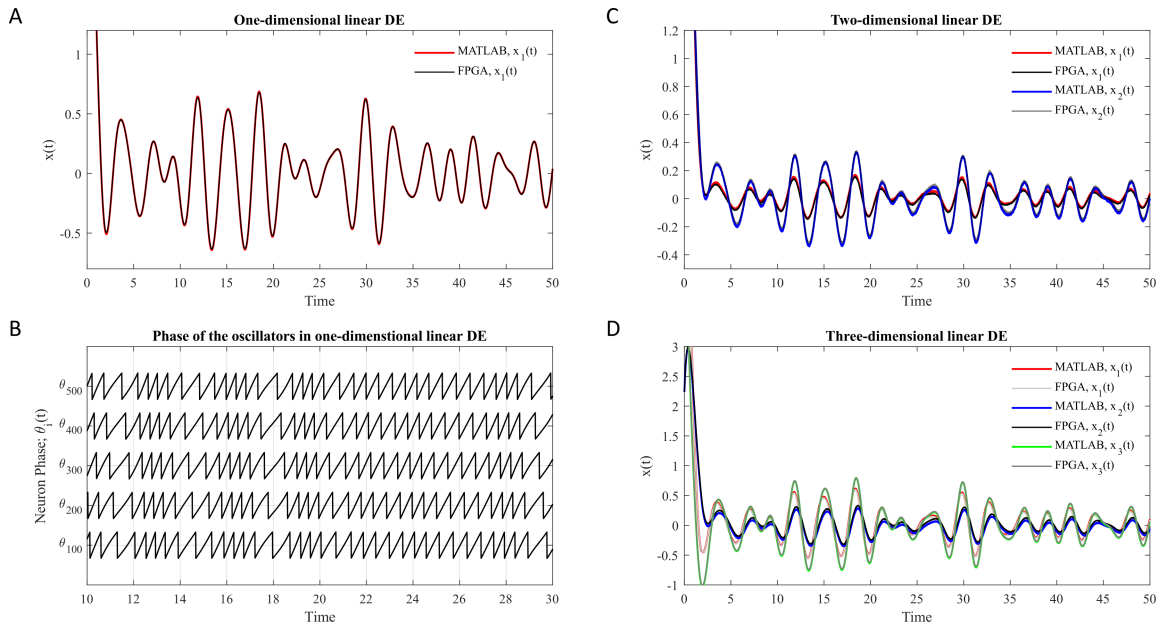


Fig. 5: The hardware emulation results of linear differential equations using a PCO network with $m = 1$ (a-b),$m = 2$ (c),$m = 3$ (d), for $N = 512$ and $\omega = 15$ in each PCO network. A random input signal is applied to the PCOs. (a) The simulated PCO network (red) versus the results of the FPGA emulation (black). (b) The oscillators' phases for $i = 100, 200, 300, 400$ and $i = 500$. (c) The simulated PCO networks (red/blue) versus the emulated FPGA networks. (d) The simulated PCO networks (red/blue/green) versus the emulated FPGA networks (greys/black for clarity).

TABLE II: Synthesis resource usage of iCE40UP5K for three populations of PCOs.

	LUT4	Registers	DSP multiplier	BRAMs
MAC operation	129	256	3	0
Coupling block	51	270	0	9
Population block	96	469	0	9
Memory	0	120	0	8
Control unit	79	82	0	0
Total	355	1197	3	26

TABLE III: Resource usage comparison

	# Oscillators	LUT	Registers	FPGA
Mesh topology [10]	100	2283	1695	XC6VLX240T
This work	1500	355	1197	iCE40UP5K

with mesh architecture $N \times N$ has been implemented for reservoir computing (RC). For each oscillator, a dedicated hardware block was used leading to a linear growth in the space complexity. A resource usage comparison is provided in Table III showing the efficiency of our implementation due to the resource sharing method. Specifically, while the number of oscillators in our design is 15 times more, the number of registers and LUTs are 30% and 85% less then the design in [10].

3) Power Consumption: To measure the power consumption, we designed and fabricated a compact board, including the iCE40UP5K FPGA and power supply components (Fig. 6a). The iCE40UP5K FPGA requires three input voltages which were provided using two switching voltage regulators (1.2 v and 3.3 v), one diode for generating 2.5 v from 3.3 v. To minimize the power consumption, the external flash memory was not used, but the FPGA was directly configured. It is worth mentioning that this FPGA can store the configuration in the One Time Programmable on-chip Non-Volatile Configuration Memory (NVCM) without requiring an extra flash memory.

Since the power consumption of a digital system is proportional to the frequency ($P \propto C.f$), the power consumption was measured for different frequencies. The total power consumption of the board was 6.6 mW, 12.6 mW, and 24.6 mW corresponding to frequencies 16 Mhz, 24 Mhz, and 48 Mhz, respectively (Fig. 6b). We note that due to the ultra-low-power consumption of the FPGA board, our implementation here allows it to be used in wearable neuromorphic devices or neuromorphic edge computing.

4) Processing time: The processing time which is required to solve the differential equation can be calculated by

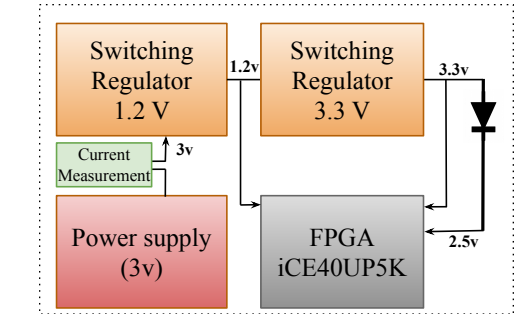
$$T_{proc} = N_{clk} \times N_{ts} \times 1/Frq, \tag{54}$$

where N_{clk}, N_{ts}, and Frq are the number of the clocks per time step, the number of time steps, and the frequency of the FPGA, respectively.

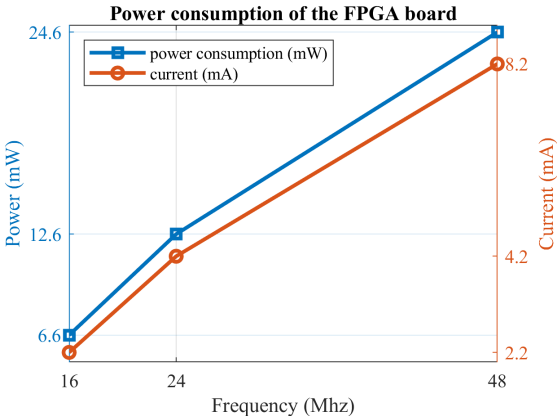
In this implementation, 523 clocks were required for calculations per time step (see (32), $N_{clk} = 512 + 3 + 6 + 2$). The input signal with the duration of 50 time unit was sampled with the period of $\Delta = 0.0078$, thus there were 6410 samples, as well as time steps ($50/0.0078 = 6410$). The processing time of the FPGA with $Frq = 24Mhz$ was 139.2 ms for solving one-, two- and three-dimensional DEs.

V. Conclusion

PCOs provide a convenient alternative to approximating solutions of linear differential equations with inputs. They



(a)



(b)

Fig. 6: The FPGA power consumption. (a) The fabricated board including the voltage regulators and the FPGA (iCE40UP5K). (b) The measurement of the current and power consumption of the FPGA board during its fully operational phase.

can be implemented in simple configurations that are mathematically guaranteed to converge to a specified linear dynamical system as the number of oscillators becomes large. These systems are efficient as they drastically reduce the number of matrix multiplications required to implement an approximation to a linear dynamical system. Furthermore, we have introduced a hardware architecture designed specifically for the implementation of PCOs. This work, encompassing analytical derivations, simulation results, and the hardware implementation of m-dimensional linear differential equations, substantiates the viability and effectiveness of our proposed methodology, and the potential application to edge devices as linear feedback controllers, or linear filters for wearable, edge computing devices.

Appendix A
Fokker-Planck approximation

The Fokker-Planck equation is a partial differential equation that describes the evolution of probability densities in stochastic processes. We used the Fokker-Planck equation for a population of uncoupled oscillators where each oscillator receives an identical input $g(t)$ and the number of oscillators approaches infinity $N \to \infty$:

$$\frac{d\theta_i}{dt} = \omega_i + g(t), \quad i = 1, 2, \dots N. \tag{55}$$

We assumed that the function $g(t)$ is integrable all $t > 0$, and that it possesses the anti-derivative $G(t)$:

$$G(t) = \int_0^t g(t')\, dt'. \tag{56}$$

Further, we assume that the oscillators are otherwise homogeneous ($\omega_i = \omega$). In the limit of large numbers of oscillators ($N \to \infty$), the system converts into a Fokker-Planck equation:

$$\frac{\partial \rho(\theta,t)}{\partial t} = -\frac{\partial J(\theta,t)}{\partial \theta} = -\frac{\partial(\omega + g(t))\rho(\theta,t)}{\partial \theta}, \tag{57}$$

which has the following boundary conditions and initial value problem.

$$J(2\pi,t) = J(0,t), \quad \int_0^{2\pi} \rho(\theta,t)\, d\theta = 1, \quad \rho(\theta,0) = \rho_0(\theta). \tag{58}$$

The Fokker-Planck equation in (57) has a convenient analytical solution in the form

$$\rho(\theta,t) \quad = \quad \rho_0(\mathrm{mod}\,[\theta - G(t) - \omega t, 2\pi]) \tag{59}$$

where the *modulo* function enforces the flux boundary conditions.

References

[1] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback control theory*. Courier Corporation, 2013.

[2] K. Atkinson, W. Han, and D. E. Stewart, *Numerical solution of ordinary differential equations*. John Wiley & Sons, 2011.

[3] J. Buck, "Synchronous rhythmic flashing of fireflies. ii." *The Quarterly Review of Biology*, vol. 63, pp. 265–289, 9 1988.

[4] A. Velichko, M. Belyaev, V. Putrolaynen, A. Pergament, and V. Perminov, "Switching dynamics of single and coupled vo 2 -based oscillators as elements of neural networks," *International Journal of Modern Physics B*, vol. 31, p. 1650261, 1 2017. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/S0217979216502611

[5] A. Parihar, N. Shukla, S. Datta, and A. Raychowdhury, "Synchronization of pairwise-coupled, identical, relaxation oscillators based on metal-insulator phase transition devices: A model study," *Journal of Applied Physics*, vol. 117, p. 054902, 2 2015. [Online]. Available: http://aip.scitation.org/doi/10.1063/1.4906783

[6] E. Corti, B. Gotsmann, K. Moselund, A. M. Ionescu, J. Robertson, and S. Karg, "Scaled resistively-coupled vo2 oscillators for neuromorphic computing," *Solid-State Electronics*, vol. 168, p. 107729, 6 2020. [Online]. Available: https://doi.org/10.1016/j.sse.2019.107729 https://linkinghub.elsevier.com/retrieve/pii/S0038110119307324

[7] J. Shamsi, M. J. Avedillo, B. Linares-Barranco, and T. Serrano-Gotarredona, "Hardware implementation of differential oscillatory neural networks using vo 2-based oscillators and memristor-bridge circuits," *Frontiers in Neuroscience*, vol. 15, pp. 1–14, 7 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnins.2021.674567/full

[8] ——, "Effect of device mismatches in differential oscillatory neural networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 2, pp. 872–883, 2023.

[9] K. Matsuzaka, T. Tohara, K. Nakada, and T. Morie, "Analog cmos circuit implementation of a pulse-coupled phase oscillator system and observation of synchronization phenomena," *Nonlinear Theory and Its Applications, IEICE*, vol. 3, no. 2, pp. 180–190, 2012.

[10] D. Pramanta and H. Tamukoh, "Design and implementation of pulse-coupled phase oscillators on a field-programmable gate array for reservoir computing," in *Neural Information Processing*, H. Yang, K. Pasupa, A. C.-S. Leung, J. T. Kwok, J. H. Chan, and I. King, Eds. Cham: Springer International Publishing, 2020, pp. 333–341.

[11] T. Tateno and H. P. Robinson, "Phase resetting curves and oscillatory stability in interneurons of rat somatosensory cortex," *Biophysical Journal*, vol. 92, pp. 683–695, 1 2007.

[12] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks*, vol. 3, pp. 281–323, 5 2005.

[13] J. S. Alshudukhi, Z. G. Al-Mekhlafi, M. T. Alshammari, and B. A. Mohammed, "Desynchronization traveling wave pulse-coupled-oscillator algorithm using a self-organizing scheme for energy-efficient wireless sensor networks," *IEEE Access*, vol. 8, pp. 196 223–196 234, 2020.

[14] T. Anglea and Y. Wang, "Decentralized heading control with rate constraints using pulse-coupled oscillators," *IEEE Transactions on Control of Network Systems*, vol. 7, pp. 1090–1102, 9 2020.

[15] H. Gao and Y. Wang, "A pulse-based integrated communication and control design for decentralized collective motion coordination," *IEEE Transactions on Automatic Control*, vol. 63, pp. 1858–1864, 6 2018.

[16] A. Isidori, *Nonlinear control systems: an introduction*. Springer, 1985.

[17] H. Risken and H. Risken, *Fokker-planck equation*. Springer, 1996.

[18] R. Jordan, D. Kinderlehrer, and F. Otto, "The variational formulation of the fokker–planck equation," *SIAM journal on mathematical analysis*, vol. 29, no. 1, pp. 1–17, 1998.

[19] N. G. Van Kampen, *Stochastic processes in physics and chemistry*. Elsevier, 1992, vol. 1.

[20] S. Vellmer and B. Lindner, "Fokker–planck approach to neural networks and to decision problems," *The European Physical Journal Special Topics*, vol. 230, no. 14, pp. 2929–2949, 2021.

[21] D. Q. Nykamp and D. Tranchina, "A population density approach that facilitates large-scale modeling of neural networks: Analysis and an application to orientation tuning," *Journal of computational neuroscience*, vol. 8, pp. 19–50, 2000.

[22] L. F. Abbott and C. Van Vreeswijk, "Asynchronous states in networks of pulse-coupled oscillators," *Physical Review E*, vol. 48, no. 2, p. 1483, 1993.

[23] N. Brunel, "Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons," *Journal of computational neuroscience*, vol. 8, pp. 183–208, 2000.

[24] W. Nicola and S. A. Campbell, "Bifurcations of large networks of two-dimensional integrate and fire neurons," *Journal of computational neuroscience*, vol. 35, pp. 87–108, 2013.

[25] H. Soleimani, A. Ahmadi, and M. Bavandpour, "Biologically inspired spiking neurons: Piecewise linear models and digital implementation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 12, pp. 2991–3004, 2012.

[26] E. Jokar, H. Abolfathi, A. Ahmadi, and M. Ahmadi, "An efficient uniform-segmented neuron model for large-scale neuromorphic circuit design: Simulation and fpga synthesis results," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 6, pp. 2336–2349, 2019.

[27] W. Guo, H. E. Yantır, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Toward the optimal design and fpga implementation of spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3988–4002, 2021.