One-dimensional discrete-time CNN with multiplexed template-hardware
Manganaro, G.; Pineda de Gyvez, J.

Published in:
IEEE Transactions on Circuits and Systems. I, Fundamental Theory and Applications

DOI:
10.1109/81.847884

Published: 01/01/2000

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 14. Dec. 2018
also by Blauschild in 1981 in a variable-gain amplifier product [9], [10]. Further elaboration of this general technique appeared in a 1981 thesis [11], widely distributed within the analog IC industry. This method is identical in essence to Opris’ although presented normally in the differential form that arises with classic translinear loops (that is, those with equal numbers of clockwise and counterclockwise drops). Opris’ [1, Fig. 2, (5)], in contrast, show a correction for a single transistor’s LCE. The method is capable in principle of canceling the LCE in any translinear circuit, as Opris noted. Moreover, although Opris did not mention this, with larger current-density ratios the method tends to suppress not only ohmic (linear) departures from a logarithmic \( V_{BE} \)-to-\( I_c \) relation, but also nonlinear departures of increasingly higher order [3], [8]. This required current-density ratio can be realized either with emitter area ratios, as Opris showed, or alternatively with current ratios [3], [8].

**REFERENCES**


**One-Dimensional Discrete-Time CNN with Multiplexed Template-Hardware**

Gabriele Manganaro and Jose Pineda de Gyvez

**Abstract**—This paper presents a novel discrete-time and fully programmable cellular neural network (CNN) suitable for processing one-dimensional (1-D) signals. As 1-D signals are typically very long sequences, the system consists of a linear analog shift register for data I/O coupled to a \( A \times B \times \mu \) CNN array. In addition to the 1-D CNN architecture, a unique feature of our implementation is that the number of multipliers needed to implement both CNN templates has been minimized. This is conceivable because the multipliers are multiplexed between the \( A \times B \) and \( D \times \mu \) products during alternating phases of the controlling clock. The CNN system has been implemented in current mode based on the \( S^2 I \) technique using MOSIS Orbit 2 \( \mu \)m CMOS technology. The paper presents a thorough behavioral analysis of the new architecture, circuit-level implementations, and corresponding measured experimental results.

**I. INTRODUCTION**

One-dimensional (1-D) cellular neural networks (CNN’s) have recently received increasing attention. For these, different applications have been reported, ranging from 1-D signal processing [1]–[3], [9]–[11] to instrumentation and control [4]. In particular, a 1-D CNN architecture able to emulate the behavior of FIR filters and to perform the Daubechies wavelet transform has been thoroughly discussed in [1] and [2]. In [9] and [10], interesting 1-D neural architectures for maximum-likelihood sequence detection are studied. Besides, unlike the case of 2-D CNN’s for image processing, the applications reported for 1-D CNN’s require a very reduced number of cells [1]–[4], [9]–[11]. This paper presents the monolithic implementation of a new 1-D discrete-time programmable CNN based on the well-known \( S^2 I \) technique. Moreover, an innovative feature of our approach is an hardware-multiplexing scheme optimizing the number of multipliers required to implement templates \( A \) and \( B \). A similar hardware multiplexing approach has been previously reported in [14]. However, an element of novelty is here introduced. In [14], the hardware multiplexing ultimately halves the processing speed. Here we exploit an intrinsic idle phase of the system, available anyway because a full delay must be realized by cascading two half-delay cells, avoiding processing speed penalties.

**II. ARCHITECTURE AND IMPLEMENTATION OF THE 1-D CNN**

The proposed architecture is shown in Fig. 1. The main blocks are a tapped delay line (sr0–sr7), and a set of locally connected cells. Input data enters the delay line on the left side (through sr0) and the samples are shifted to the right. The cells of the CNN receive their inputs from the delay line and from the outputs of the neighbors. The templates are...
provided externally. The state equation describing the behavior of the cell at position \( c \) is

\[
x_c(n+1) = \sum_{d \in N(c)} A_{c,d} y_d(n) + \sum_{d \in N(c)} B_{c,d} u_d(n)
\]  

(1)

where \( x_c(n+1) \) is the updated state of the cell, \( y_c(n) = f(x_c(n)) \) is its output, \( u_c(n) \) is the output of the delay line \( srC \) (e.g., \( u_1 \) is the output of \( sr1 \)), \( A_{c,d} \) are the feedback templates while \( B_{c,d} \) are the control templates, and \( N(c) = \{ d : |d - c| \leq 1 \} \) is the neighbor set of cell \( c \). Although (1) is undoubtedly a discrete-time CNN, the above CNN equations are slightly different from the classical DT-CNN definition [13]. The CNN in Fig. 1 includes six cells and eight delays. In practice, the first delay stage \( sr0 \), included for clarity, is not strictly necessary and the input (Data In on Fig. 1) can be directly provided in place of it.

A. Cell Behavior with Multiplexing of Template-Hardware

One CNN is shown in Fig. 2(a). The overall system is based on a two-phase nonoverlapping clock \( \phi_1 \) and \( \phi_2 \), as depicted in Fig. 2(b). These determine the time synchronization represented by the time index of state equation (1). Subphases \( \phi_{1a}, \phi_{1b}, \phi_{2a}, \phi_{2b} \) are primarily used in the shift-register, subphases \( \phi_{1c}, \phi_{2c} \) are required by the multipliers and subphases \( \phi_{1d}, \phi_{2d} \) are used in the delay cells of Fig. 2(a). All these phases will become clearer later.

Each delay cell is composed of two inverting half-delays. On \( \phi_1 \) data enters and leaves the delay line cells, while on \( \phi_2 \) data is internally exchanged between the two half delays. Aside, the result of the multiplication is provided only before the end of a complete clock cycle. Thus, on \( \phi_2 \) the analog shift register is isolated from the rest of the system, then on this phase the rest of the hardware would essentially be idle. A strategy that permits us to exploit the available hardware during the idle phase and implicitly save silicon area has been devised without reducing the system throughput.

Consider Fig. 2(a). The programmable synaptic connections are drawn on the left-hand side of the figure. Those multipliers access the outputs from the delay line \( u_d \) on \( \phi_1 \) with the corresponding weights (namely, the templates \( B_{c,d} \)) provided by off-chip currents. On \( \phi_2 \), the neighboring outputs \( y_d \) are fed to the inputs of the multipliers in place of \( u_d \), while the template \( A_{c,d} \) is fed in place of \( B_{c,d} \). Therefore, taking into account the half-period delay on the cell’s outputs, the multiplexed signal is written in the \( z \) domain as

\[
W(z) = \left( \sum_{d \in N(c)} B_{c,d} U_d(z) \right) \ast \phi_1(z) \\
+ \left( \sum_{d \in N(c)} A_{c,d} Y_d(z) z^{-1/2} \right) \ast \phi_2(z)
\]

(2)

Then the following relationships are obtained:

\[
V_1(z) = \phi_1(z) \ast W(z) z^{-1} = z^{-1} \left( \sum_{d \in N(c)} B_{c,d} U_d(z) \right)
\]

(3)

\[
V_2(z) = \phi_2(z) \ast W(z) z^{-1/2} \\
= z^{-1/2} \left( \sum_{d \in N(c)} A_{c,d} Y_d(z) z^{-1/2} \right) \\
= z^{-1} \left( \sum_{d \in N(c)} A_{c,d} Y_d(z) \right)
\]

(4)

\[
X_c(z) = \phi_1(z) \ast V_1(z) + \phi_1(z) \ast V_2(z) \\
= \phi_1(z) \left[ z^{-1} \left( \sum_{d \in N(c)} B_{c,d} U_d(z) \right) \\
+ z^{-1} \left( \sum_{d \in N(c)} A_{c,d} Y_d(z) \right) \right]
\]

(5)

Observe that (5) results in the CNN state equation (1). Moreover

\[
X_0(z) = \phi_1(z) \ast X_c(z) z^{-1/2} = z^{-1/2} X_c(z)
\]

(6)

\[
Y_a(z) = f(\phi_2(z) \ast X_a(z)) = f(\phi_2(z) \ast X_a(z) \ast z^{-1/2} X_c(z)) \\
= z^{-1/2} f(X_c(z)) = z^{-1/2} Y_c(z).
\]

(7)

Equation (7) points out when the delayed output signal needed at the input of the multipliers is available to obtain the proper time-multiplexing scheme of (2). This also proves the consistency of the above statements. The above approach allow us to 1) exploit the rest of the hardware during the idle phase \( \phi_2 \) of the delay line and 2) use only three multipliers instead of six, saving in area and power. For this, only two half-delay cells have been added.

B. Monolithic Implementation of the 1-D CNN System

The available silicon area is a very stringent limitation. It forces us to discard fully differential circuit architectures, which would require wider areas. This choice implies a number of limitations on both accuracy and speed because the analog blocks are severely subjected to interference coming from the digital blocks. But it allows the use of small blocks. A cascade of \( S/T \) delay cells [5]-[7] composes the delay line. A replica of the current at the output of the cell is needed for the next delay cell and for any other circuit requiring it as input. So, a current mirror with multiple output branches is placed in between any full-delay block. Cascode mirrors have been used. The circuit of a delay line cell is shown in Fig. 3. The impedance drawn between the output and the input node represents an NMOST identical to the ones used for the switches but having its gate connected to the positive power supply. It compensates for the \( \alpha r \) drop due to the output switch [6].

In order to compensate a possible residual output offset current, another delay cell with zero input has been created. This cell’s output current is the residual offset that is copied and subtracted to the output
of all the other delay cells using the terminal Off_c in Fig. 3. Additional off-chip offset compensation has been used. The output of the cell is

\[
y_c(n) = f(x_c(n)) = \frac{1}{2}(|x_c(n) + 1| - |x_c(n) - 1|).
\]  

(8)

Two cascode mirrors synthesize this function. The saturation current in the nonlinearity is \(I_{sat} = 5 \, \mu A\). Four quadrant \(S^2I\) multipliers [7] implement the templates. A four-step algorithm (corresponding to the four phases \(\theta_1-\theta_4\)) is used. The circuit and control signals for the switches are shown in Fig. 5 of [7]. Assuming the same transconductance \(g_m\) and output impedance \(g_{0m}\) for \(M_5-M_{16}\) in Fig. 5 of [7], an approximate relationship for output current \(I_{out}\) provided on the last phase \(\theta_4\) is obtained [7]

\[
i_0 = \frac{\eta^3 \rho}{2I_t} xy \quad \text{with} \quad \eta \equiv \frac{\Psi_{g_m}}{\Psi_{g_m} + 4g_{0m}} \quad \rho \equiv \frac{g_t}{g_t + 4g_{0m}}
\]  

(9)
Fig. 4. (a) Die microphotograph. (b) Experimental results of tapped delay line behavior with an input sinusoid. Axes are 1 ms/div and 2 V/div.

Fig. 5. Simulated waveform for an example of low-pass filtering: upper trace shows the state of an inner cell while the lower trace shows the input to the delay line.

being $g_L$, the conductance of the load. The experimental characteristic is shown in Fig. 7 of [7]. The multiplier’s gain is $\eta^3 p/2I_L = 5000 \text{ A}^{-1}$. The operands must be kept constant at the inputs of the multiplier while the result is released on $\theta_1$. The main phases $\phi_1-\phi_2$ are common to the whole system. While the input delay line use the sampling subphases $\phi_{1a}-\phi_{1b}-\phi_{2a}-\phi_{2b}$, the sampling subphases for the half-delay cells$^1$ are actually $\phi_{1a}-\phi_{1b}-\phi_{2a}-\phi_{2b}$. A whole multiplication cycle is performed on $\phi_1$ and another one on $\phi_2$. This subphase modification does not affect the throughput of the system nor the phase generator complexity. In fact, $\phi_{1a}-\phi_{1b}-\phi_{2a}-\phi_{2b}$ have the same pulse width as the internal subphases of the multipliers [7]. More specifically, these phases correspond to $\theta_{1a}-\theta_{1b}$ on $\phi_1$ and $\theta_{2a}$, respectively, which are the only sampling subphases not used in the multipliers. The phase generation is obtained throughout an eight-stage ring counter and by additional combinatorial and sequential circuits. Finally, note that the value of $x_c$ at the output node is affected by uncompensated offset due to the half-delay cell on the left-hand side of Fig. 2(a). But it can be compensated with off-chip circuitry. Instead, this offset is implicitly compensated at the output of the second half-delay cell before the output cell nonlinearity. Note that the depicted half-delay cells have an ideal transfer function $z^{-1/2}$ instead of $-z^{-1/2}$. Sign inversion is important for the purpose of offset compensation. Straightforward insertion of the necessary inverting blocks (current mirrors) has not been represented to avoid clutter.

$^1$The half-delay cell storing $x_c$, shown in the right hand side of Fig. 2(a), would not necessarily need this modification. However, using $\phi_{2a}-\phi_{2b}$, the feedthrough offset of the first half-cell can be canceled and the layout is simplified.
IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-I: FUNDAMENTAL THEORY AND APPLICATIONS, VOL. 47, NO. 5, MAY 2000

III. EXPERIMENTAL RESULTS

A chip prototype, shown in Fig. 4(a), has been designed and fabricated in MOSIS Orbit n-well 2 µm technology. The total area of the chip is 1.8 × 1.8 mm. The input currents have been provided by means of off-chip V–I converters. The power supply is ±3 V for a power consumption of 30 mW to 80 mW depending on the input signals and templates.

Let us consider the delay line. A sinusoid is fed at the input of the delay line. The outputs of the on-chip linear I–V converters corresponding to the delay stages Sr2 and Sr6 are shown in Fig. 4(b). A four-unit delay from one wave to the other is seen. In Fig. 4(b) it seems that attenuation is present. In reality what is shown in the photo is the output of on-chip I–V converters, which have not been designed for an accurate transresistance gain but only to verify the delay line functionality. In fact, from these pins we have noticed slight random variations on the amplitudes. But the real amplitude of the signal tapped out from the last stage is negligibly attenuated (the total current gain between the input of the delay line and the output of its last stage is 0.997 A/A) with respect to the input signal. The average current gain of a single element of the delay line (delay cell plus corrective current mirror) is estimated to be 0.99957 A/A.

Halving the number of multiplier, whose outputs are tied to a common node, relaxes the requirements on their output impedance. However, short-term analog storage will inevitably degrade the accuracy of the processing. The degradation is equivalent to the one introduced by the delay line. The choice of using single-ended analog circuits implies severe interference coming from the digital circuits. This pick-up noise degrades the SNR of the analog blocks even at relatively low frequency. This seems enough to be adequate enough for many of the applications in the cited bibliography such as voice processing [1] or low-speed measurements [4]. From signal to noise and distortion ratio (SNDR) measurements performed on a prototype of the S2I multiplier (see also [7]) and, more importantly, on the state of the CNN we estimate an ENOB (effective number of bits) of 8–9b at a sampling frequency of around 15 KHz. In this case the spurious-free dynamic range is around 55 dB. These measurements have been performed by means of a spectrum analyzer [12]. Due to the single-ended nature of the circuits, the second harmonic distortion is the dominant factor, while the dominant source of noise comes from interference with the digital modules. The system works well for telephony speech frequencies, e.g., a bandwidth of 3.4 KHz.

In the last experiment a square wave is low-pass filtered. This requires both $A_{cc}$ and $B_{cc}$ activated

$$A = [0 \quad 1/2 \quad 0], \quad B = [1/2 \quad 1 \quad 1/2]. \quad (10)$$

The simulated (behavioral simulation) input of the delay line and the state of an inner (namely with complete neighbor set) cell are shown in Fig. 5. The corresponding waveforms are depicted in Fig. 6(a) for cell 2. A detail of the state is reported in Fig. 6(b) both with a signal related to the node voltage at the summing node of the corresponding cell. This is a nonlinear image (biased and distorted image of $W(z)$), which is a current) of the signal $W(z)$ represented in (2).

IV. CONCLUSION

The VLSI implementation of a discrete-time 1-D CNN has been discussed. One of the peculiarities of the proposed architecture is a hardware-multiplexing strategy. This allows to efficiently use the hardware halving the number of multipliers and storing the intermediate results into temporary memories. To implement the proposed architecture for video signal processing applications the basic $S^2I$ memory cell discussed in this paper needs to be substituted by more advanced building blocks [8], [5] at the expenses of increased power and area. Nonetheless, the multiplexing approach introduced is invariably applicable to these more advanced architectures. A CMOS N-well MOSIS Orbit 2 µ chip has been fabricated. Experimental results have been reported.

REFERENCES


Limit Cycles Elimination in Delta-Operator Systems

Kamen R. Ralev and Peter H. Bauer

Abstract—The existence of nonzero equilibria in δ-operator fixed point and block floating point (BFP) systems is investigated and methods for avoiding such equilibria are proposed. In the fixed point case these methods work by mapping the region in which nonzero equilibria may appear to zero. This is possible if the region is small. It is also shown that nonzero equilibria and limit cycles of any period can always be avoided by using BFP arithmetic with a sufficiently large mantissa wordlength.

Index Terms—(Block) floating point arithmetic, delta operator systems, limit cycles, quantization effects.

I. INTRODUCTION

The importance of δ-operator formulated systems is twofold. First, they allow unified treatment of continuous and discrete systems [1]. Second, they offer lower roundoff noise [2] and lower coefficient sensitivity [3] compared to shift operator implementations if the poles of the shift operator system are clustered around one, which corresponds to a fast sampling of the continuous-time system.

Limit cycles sometimes occur in δ-operator implementation of linear systems. Later, an example of an eight-bit fixed point filter implementation shows that with excellent roundoff properties (shown in [2]) but sustains limit cycles occupying the last five bits. In this paper we propose methods for elimination of limit cycles in δ-operator systems. For a fixed point implementation, a bound on the region in which nonzero equilibria may appear is derived. If this region is small, it may be mapped to zero in order to eliminate such equilibria, as shown later. Otherwise, one may apply block floating point (BFP) arithmetic. It is shown later that if the mantissa word length is sufficiently large, all nonzero equilibria, as well as the limit cycles of any period, are eliminated. At the expense of a small increase in complexity, this approach also yields a higher dynamic range and improved signal-to-noise ratio [4].

The problem of limit cycle elimination in regular (shift-operator) fixed point systems is well studied. However, the elimination of limit cycles in δ-operator systems has not been addressed in the literature before. In [5] a sufficient condition for absence of nonzero equilibria in fixed point δ-operator systems is proposed, which is improved upon here. The treatment of block floating point δ-operator systems is new. Thorough treatment of the δ-operator systems appears in [1] and [6] and the references therein, however, the problem of limit cycles is not addressed.

II. PRELIMINARIES

Let $\mathbb{R}$ be the set of reals, $\mathbb{Z}$ the set of integers, $\mathbb{R}^n$ the set of $n$-dimensional real vectors, and $f$ a mapping from $\mathbb{R}$ to $\mathbb{R}^n$. Define the shift operator $q$ by $qf(t) := f(t + \Delta)$ where $\Delta$ is a positive real constant. The δ-operator is defined as

$$\delta f(t) := \frac{qf(t) - f(t)}{\Delta}.$$  

Let $f(k) = f(k\Delta)$ for all $k \in \mathbb{Z}$. Using the δ operator, a linear system sampled with period $\Delta$ is described by

$$\begin{align*}
\delta x(k) &= A_{\delta} x(k) + B_{\delta} u(k) \\
y(k) &= C_{\delta} x(k) + D_{\delta} u(k)
\end{align*}$$

where $x(k) \in \mathbb{R}^n$ state vector; $u(k) \in \mathbb{R}^n$ input; $y(k) \in \mathbb{R}^m$ output; $A_{\delta}, B_{\delta}, C_{\delta}, D_{\delta}$ matrices of appropriate size.

We shall also frequently refer to the corresponding $g$-operator system

$$\begin{align*}
gx(k) &= x(k + 1) = A_{g} x(k) + B_{g} u(k), \\
gy(k) &= C_{g} x(k) + D_{g} u(k).
\end{align*}$$

One can write (1) using the definition of the δ operator as

$$x(k + 1) = x(k) + \Lambda(A_{\delta} x(k) + B_{\delta} u(k)).$$

The corresponding $q$ and δ-operator formulations are therefore related by

$$\begin{align*}
A_{\delta} &= \frac{A_{g} - I_{n}}{\Delta}, \\
B_{\delta} &= \frac{1}{\Delta} B_{g}, \\
C_{\delta} &= C_{g}, \\
D_{\delta} &= D_{g}
\end{align*}$$

where $I_{n}$ is the identity matrix of appropriate size.

As pointed out in [7], $\Delta$ depends on the time units selected. Therefore, we are free to choose it as a negative power of two so that the entries of $A_{\delta}$ are in the proper dynamic range, i.e., $\Delta = 2^{-d}$ for some $d \in \mathbb{Z}$. Thus, some operations in fixed and block floating point implementations can be eliminated.

Consider the zero-input system obtained from (3)

$$x(k + 1) = x(k) + \Delta A_{\delta} x(k).$$

If $x(k + M) \neq x(k)$ for some integers $k, M$, the response $\{x(k)\}_{k=0}^{M}$ is called a limit cycle with period $M$. If $x_0 = x(k + 1) = \cdots = x(k + M)$, then