A 12.08-TOPS/W All-Digital Time-Domain CNN Engine Using Bi-Directional Memory Delay Lines for Energy Efficient Edge Computing

Aseem Sayal, *Student Member, IEEE*, S. S. Teja Nibhanupudi, *Student Member, IEEE*, Shirin Fathima, and Jaydeep P. Kulkarni, *Senior Member, IEEE*

Abstract*— In this article, we demonstrate an energy efficient convolutional neural network (CNN) engine by performing multiply-and-accumulate (MAC) operations in the time domain. The multi-bit inputs are compactly represented as a single pulse width encoded input. This translates into reduced switching capacitance (C_{DYN}), compared to baseline digital implementation, and can enable low power neural network computing in an edge device. The time-domain CNN engine employs a novel bi-directional memory delay line (MDL) unit to perform signed accumulation of input and weight products. The proposed MDL design leverages standard digital circuits and does not require any capacitors and complex analog-to-digital converters (ADCs) to realize the convolution operation, thereby enabling easy scaling across the process technology nodes. Four speed-up modes and a configurable MDL length are supported to address throughput versus accuracy trade-off of the time-domain computing approach. Delay calibration units have been accommodated to mitigate the process variation induced delay mismatch among concurrently operating MDL units. The proposed time-domain MDL design implements a LeNet-5 CNN engine in a commercial 40-nm CMOS process achieving an energy efficiency of 12.08 TOPS/W, a throughput of 0.365 GOPS at 537 mV in the 16× speed-up mode. 40-nm CMOS test-chip measurements over 100 MNIST images show 97% classification accuracy. Simulation results over the entire 10 000 MNIST validation dataset images taking into account the circuit non-ideal effects of the MDL-based time-domain approach show a classification accuracy of 98.42%. The test-chip is operational down to the near-threshold voltage (up to 375 mV) while maintaining the classification accuracy over 90% in the 1× speed-up mode. Furthermore, two methods of scaling MDLs to multi-bit weights are proposed. Simulation results for 1000-class AlexNet over 50 000 ImageNet validation dataset images show classification accuracy loss within 1% when compared with software implementation. The proposed MDL based time-domain approach performing 1-bit/8-bit weight and 8-bit input MAC operations when compared with the corresponding baseline digital implementations shows 2.09×-2.32× higher energy efficiency and 2.22×-3.45× smaller area.

***Index Terms*— AlexNet, bi-directional memory delay line (MDL), binary neural network, convolutional neural network (CNN), energy efficient edge computing, ImageNet, LeNet-5, MNIST, multiply-and-accumulate (MAC), time-domain processing.**

I. Introduction

With the rapid progress in machine learning (ML) algorithms coupled with performance improvements in computing resources, there has been an unprecedented increase in the deployment of cognitive devices in various applications such as image classification, speech recognition, object localization, facial recognition, and so on [1]–[3] (see Fig. 1). In particular, convolutional neural networks (CNNs), a class of artificial neural network are extensively used for many such ML applications, due to their state-of-the-art classification accuracy at a much lesser complexity compared to their fully connected network (FCN) counterpart [4], [5]. However, the CNN inference process requires intensive compute and memory resources making it challenging to implement in energy constrained edge devices. For example, the widely known CNNs such as AlexNet [3] and VGG [6] comprise 60 million and 138 million filter weights, respectively, which are used to compute 724 million and ≈15.5 billion multiply-and-accumulate (MAC) operations [4]. Such an enormous number of MAC operations consume a significant fraction of a CNN accelerator's total power budget [7], [8]. Thus, making MAC operation energy efficient is extremely important.

In this article, a time-domain MAC computing approach is proposed by leveraging the concept of a time accumulator. The key attributes of the proposed design include: 1) bi-directional memory delay lines (MDLs) performing time-domain signed MAC operations; 2) multi-precision filter weight support (signed/unsigned 1–8 bits); 3) 16 filters each supporting 2 × 2 sub-sampling (maximum pooling) and averaging; 4) all-digital, technology scalable design without requiring any capacitors, analog-to-digital converters (ADCs), and/or frequency generators/modulators; 5) near threshold voltage operation; and 6) 4 speed-up modes supporting 1–16× throughput improvement by quantizing 4 LSBs of input activation. An energy efficient CNN engine implemented in a commercial 40-nm CMOS process (see Fig. 17) is demonstrated.

This article is organized as follows. Section II describes the background of this work and discusses the basics of CNNs and motivation for time-domain computation. Section III presents

Manuscript received April 17, 2019; revised July 5, 2019 and August 24, 2019; accepted August 26, 2019. Date of publication October 14, 2019; date of current version December 27, 2019. This article was approved by Guest Editor Mingoo Seok. This work was supported in part by AMD, Austin, TX, USA, and in part by The University of Texas at Austin, USA. *(Corresponding author: Aseem Sayal.)*

A. Sayal, S. S. T. Nibhanupudi, and J. P. Kulkarni are with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA (e-mail: aseem.sayal@utexas.edu; subrahmanya_teja@utexas.edu; jaydeep@austin.utexas.edu).

S. Fathima was with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA. She is now with Apple Inc., Cupertino, CA USA (e-mail: shirin15595@gmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2019.2939888

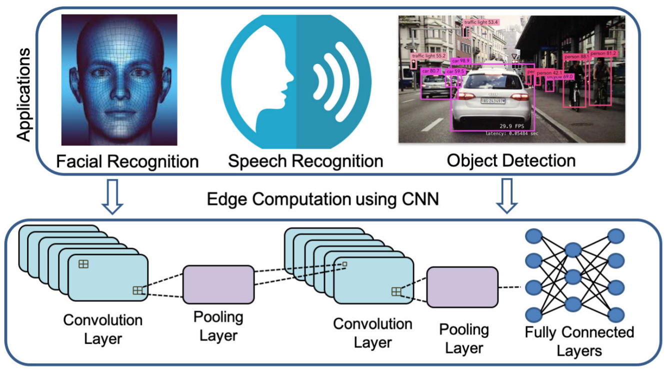


Fig. 1. Edge computation using CNN.

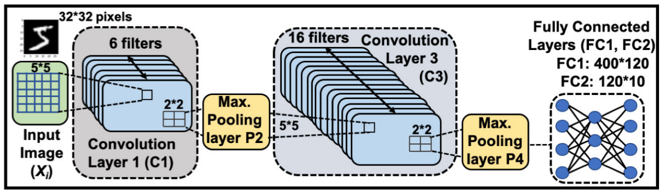


Fig. 2. LeNet-5 CNN [9] architecture.

the concept of proposed time-domain MAC computation using time accumulators, circuit design, architecture, and implementation details. Various design trade-offs along with their mitigation strategies are presented in Section IV. Section V presents the 40-nm CMOS test-chip measurement results. A case-study is also presented in this section comparing the energy efficiency of the proposed time-domain circuit with the baseline digital approach. Section VI concludes this article with the key findings.

II. Background

A. Convolutional Neural Network

A generic CNN consists of convolution layers, pooling layers, activation layers, and FCN layers. The convolution layer is the core building block of a typical CNN. Each convolution layer implements multiple trainable filters which are used to extract feature-like edges, gradients, colors, etc., in an image. Fig. 2 shows the LeNet-5 CNN [9] architecture which can be used to classify the hand-written digits (MNIST dataset [9]). During the training phase, the weights of all the filters for each layer are determined using a training algorithm (such as back-propagation [10], [11]) on the training MNIST dataset. Each filter is convolved across the width and the height of input activation, computing the dot product between the weights of the filter and the input activation, producing a 2-D output activation map for a given filter. As a result, the neural network learns to filter weights that get activated when specific types of features are detected at specific spatial positions in the input activation. During the inference phase, when the test data are presented to the CNN layers, key features (e.g., edges, gradients, color changes, orientation in a handwritten digits MNIST dataset, etc.) in the input activation are extracted at each layer. The major operation in CNNs is the MAC operation given by (1), which is computed by performing a dot product of a weight matrix and an input image matrix [4]. The MAC

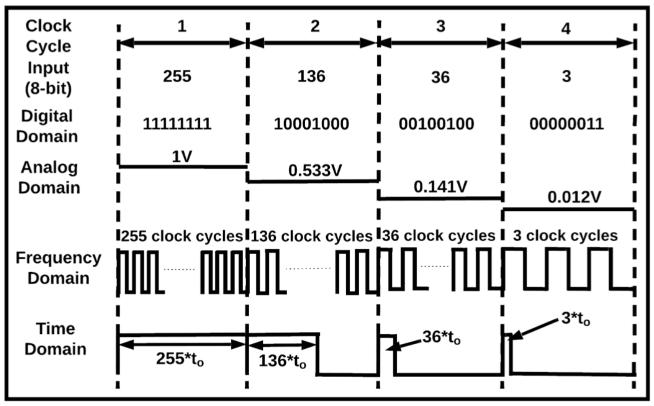


Fig. 3. Representation of 8-bit inputs in different signal domains.

value is averaged out to compute the multiply-accumulate-average (MAV) value [given by (2)], which ensures that the output value does not exceed the range

$$\text{MAC} = \sum_{i=1}^{N}(X_i \times w_i) \tag{1}$$

$$\text{MAV} = \frac{1}{N}\sum_{i=1}^{N}(X_i \times w_i) \tag{2}$$

where N is the number of products per MAC, X_i is the textitith input pixel value, and w_i is the ith weight value.

Typically, an activation layer is used between two consecutive convolutional, pooling, or fully connected layers. The purpose of this layer is to introduce nonlinearity into a neural network. The rectified linear unit (ReLU) is the most commonly used activation layer which returns 0 if it receives any negative input and returns back the same value if it is positive. Pooling (sub-sampling) is performed to reduce the data size feeding into the next layer [12], [13]. Max-pooling is typically used for the sub-sampling operation which chooses the maximum MAC value across a small output window (e.g., 2 $\times$ 2). This can reduce the memory footprint to store intermediate layer outputs by 75%, as the outputs of the pooling layers (P2 or P4) are stored instead of convolution layers (C1 or C3) [12]. Once the data size of input feature maps is reduced after passing through convolution and pooling layers, FCN layers are used. A typical FCN layer comprises a finite number of feature maps, each of size 1 $\times$ 1. Each of these feature maps is connected to all the feature maps of the previous layer. In LeNet-5 CNN, a fully connected layer (FC1) comprises 120 feature maps, each of which is connected to all the 400 nodes of the fourth layer (P4). Similarly, the last layer of LeNet-5 CNN is a fully connected layer (FC2) comprising 10 1 $\times$ 1 feature maps, each of which is connected to all 120 feature maps of FC1.

B. Motivation for Time-Domain Computing

As MAC operations constitute a significant portion of the total CNN power budget, it is worthwhile to consider various methods for compact data representation to improve the energy efficiency (see Fig. 3). The data in the digital domain are represented as a multi-bit digital vector [14], [15]

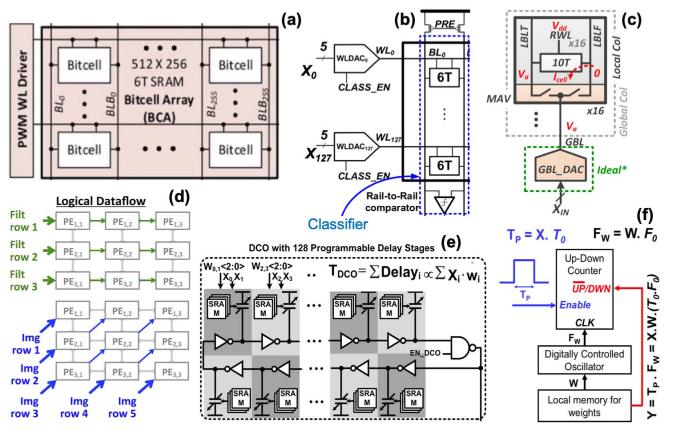


Fig. 4. Prior energy efficient ML accelerators in (a)–(c) analog voltage [16]–[18], (d) digital [4], and (e) and (f) time and frequency domains [19], [20].

[see Fig. 4(d)]. This form of data representation is implemented using higher number of nets, translating into high dynamic switching capacitance (C_{DYN}), and consequently higher power and area. A detailed case-study on the C_{DYN} comparison is presented in Section V-F.

In the analog domain approach, data are represented as a continuous varying voltage signal. Various design techniques using charge manipulation schemes and ADCs have been proposed to realize efficient MAC computations in a CNN accelerator. Analog approaches [16]–[18] compute MAC operation in the analog voltage domain using a static random access memory (SRAM) array, capacitors, and data converters [ADCs and digital-to-analog converters (DACs)]. In these approaches, input pixel data are either encoded as the pulse width modulated (PWM) signal [16] or pulse amplitude modulated (PAM) [17] signal, and applied on the wordline [see Fig. 4(a) and (b)]. The MAC operation is performed by summing up the read current of simultaneously accessed bitcells. The bitline voltage in this case represents the sum of products (SRAM bit weight $\times$ wordline input pixel). The bitline current summation approach is susceptible to: 1) process variations in the nanoscale SRAM bitcell transistors; 2) functional failures due to bit-flips as a result of long duration wordline activations and/or large bitline voltage differential; and 3) the possibility of corrupting a weak bit by shorting it to a strong bit storing opposite value due to concurrent wordline activations. In the design approach discussed in [18], the input pixel digital bit-stream is first converted into an analog voltage using DAC, and then used to precharge the bitline to compute the MAC operation [see Fig. 4(c)] using a ten-transistor SRAM bitcell. This approach mitigates the process variation induced bitcell current variation as well as bit-flip scenarios at the expense of a larger bitcell area. However, the finite voltage headroom and the sensitivity of circuit parameters to slight change in analog domain signals limit the voltage scalability, thereby degrading the MAC accuracy.

In the frequency domain approach, data are represented as signals with varying frequency using ring oscillators or RC loaded circuits. In [19] and [20], MAC operations are computed in the frequency domain using a digital controlled oscillator (DCO) with either resistor or capacitor loading. Various nodes of a ring oscillator are loaded with different capacitor banks to alter the RC time constant of the oscillator [see Fig. 4(e) and (f)]. The capacitor value is controlled by the SRAM bitcells storing the weight. This approach makes the design sensitive to parasitic diffusion capacitance of the DCO. The added R and C components need to be larger than the diffusion capacitance to linearly modulate the ring oscillator frequency in response to the weight change. In addition, adding a capacitor at every node of a continuously running ring oscillator increases the total C_{DYN}, and consequently the power dissipation. Furthermore, implementing such binary weighted capacitor banks can occupy significant area, thereby degrading the area efficiency. In [20], the MAC operation is performed in the frequency domain by counting the number of binary weighted frequency pulses (representing multi-bit weights) in a given PWM input signal. Thus, large range frequency generators/modulators limit the performance scalability of such frequency domain approaches. Moreover, larger magnitude inputs would result in higher toggle activity incurring higher switching power.

In the time-domain computing approach, data can be represented as PWM inputs or pulse position modulated inputs or a combination of thereof. In [21], the time-domain analog-to-digital mixed-signal processing (TDAMS) approach is proposed to implement time-domain binary neural networks. TDAMS uses the time difference in the rising edges to generate variable delays in order to realize the MAC operation of 1-bit input activation and weight value. This approach further employs fully spatially unrolled architecture where every memory has its dedicated processing element (PE) to increase its energy efficiency. A time-mode adder circuit is proposed in [22] to realize a trans-linear principle in time. The exponential relationship between the voltage and time in an RC circuit is leveraged to implement time adders.

In the proposed work, a multi-bit digital bit-stream is encoded as a single PWM signal with its pulse width representing the magnitude of the input data. This results in multi-bit data to single data-signal compaction resulting in reduced C_{DYN} compared to the conventional digital data representation. Furthermore, the time-domain computing approach can leverage standard digital gates with full rail-to-rail voltage swing outputs. This mitigates the reduced voltage headroom challenges encountered in an analog voltage domain approach enabling an ultra-low voltage operation. Additionally, the time-domain approach does not require multiple clock sources unlike frequency domain approaches which result in reduced C_{DYN}. Unlike the frequency domain, the toggle activity does not depend on the input magnitude in the proposed time-domain approach. Thus, although the time-domain approach results in slower throughput due to its sequential operation, it can be a promising approach for realizing MAC computations, especially in energy constrained edge devices.

III. Proposed Time-Domain CNN Engine Design

A. Concept of Time-Domain MAC Computation

The concept of time domain MAC computation using PWM inputs is illustrated in Fig. 5. In this example, the MAC

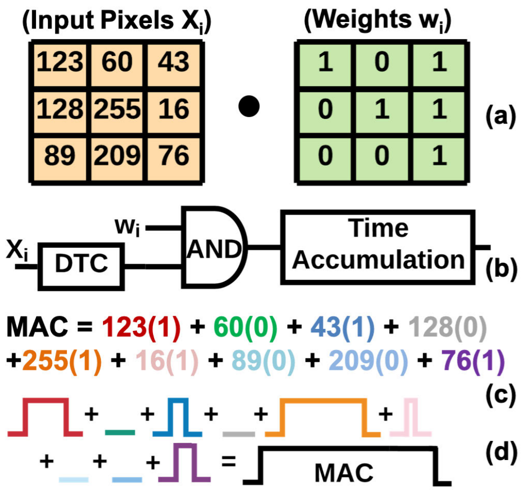


Fig. 5. Concept of time-domain MAC operation. (a) Dot product operation of the input pixel matrix and the weight matrix. (b) Time-domain MAC circuit concept. (c) MAC operation in the digital domain. (d) MAC operation in the time domain.

operation is performed by adding the products of a 3 × 3 weight matrix and an input pixel matrix [see Fig. 5(a) and (c)]. Binary weights [23]–[25] are used to simplify the discussion. However, the time-domain approach is scalable to multi-bit weight values. An input pixel value is encoded into a PWM signal using a digital-to-time converter (DTC). Then, the input PWM signal is multiplied by the corresponding weight bit using an AND gate. This time-encoded product signal is then passed onto the time accumulator circuit [see Fig. 5(b)] to compute the MAC operation in the time domain. Sequentially, all input pixel values are encoded as PWM signals, and its product with respective weight bit is loaded on the time accumulator circuit. Thus, all time-encoded products are applied onto the time accumulator circuit, and the width of these timing pulses gets accumulated to realize the MAC operation in the time domain [see Fig. 5(d)]. The time accumulator circuit will be discussed in detail in Section III-D4.

B. Architecture Overview

The proposed time-domain CNN architecture performing the MAC operation of 8-bit inputs with 1-bit weights is shown in Fig. 6. First, 16 PWM signals (generated using the pulse generator module) are selected based on the 8-bit input image pixel value. The pulse selector acts as a DTC, converting the digital image pixel input into a time-encoded PWM signal. The higher the pixel value, the larger the pulse width of the time-encoded input. The generated PWM signal is passed through a pulse gating logic to avoid repetitive products of same input being added to the MAC output. This ensures that the time-encoded input pixel signal is utilized only once in the time-domain MAC computation. This gated PWM signal is then multiplied by the binary weight (0 or 1 or −1) using an AND gate. Multiplication by weight bit “0” results in no toggling at the AND output, while multiplying by weight bit “±1” results in PWM output at the AND gate which is the same as the input PWM signal. The weight bits are stored in the weight register, which is implemented as a cyclic shift register. The parallel-in

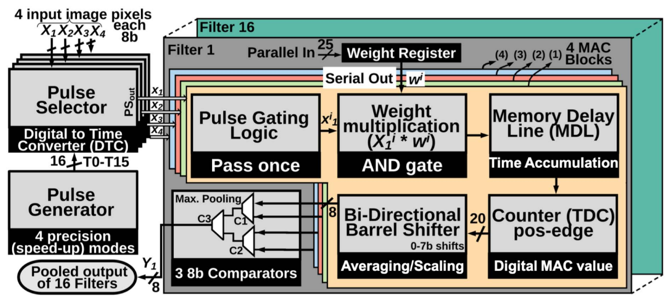


Fig. 6. Architecture of the proposed time-domain-based CNN engine.

and serial-out shift register is used to perform multiplication of the input pixel value and weight. The time-encoded product signal is then added using the proposed bi-directional MDL to perform the signed addition of the products for each MAC operation. The operation of the proposed MDL performing time-domain accumulation is discussed in detail in the next section. A 20-bit positive edge triggered up-down counter is followed by the MDL to convert the time-encoded MAC signal into digital value. It acts as a time-to-digital converter (TDC) for performing post-processing in the digital domain. This signal is right-/left-shifted using a 20-bit bi-directional barrel shifter to perform scaling and averaging after a MAC operation to compute the MAV output. The shift operation correctly scales the MAC output value before feeding it as an input to the next convolution layer. Once the shifter output value from four MDLs for each filter is obtained, the max-pooling operation (2 × 2 window) [12], [13] is performed using three 8-bit comparators. The pooled output from each filter is stored off-chip and reused as the input to the next convolution layer.

C. Data Flow Overview

Sixteen filters are implemented on-chip, where each filter block comprises 4 MAC blocks to perform signed time accumulation and 2 × 2 max-pooling. Each filter comprises a 25-bit weight register to store weights. In the C3 layer, 5 × 5 1-bit weights of each channel are stored in the weight register, and this process is repeated six times to perform the MAC operation. Four input pixel values encoded in the time domain are applied to these 16 filters using four pulse selector modules and a shared pulse generator module. The input activation and weight values (for all the channels) are stored off-chip and applied to the test-chip using a scan chain. The output activations (computed MAV values after pooling) from the scan out port of the scan chain are stored off-chip using the National Instruments PXIe system. Different operation modes (states) of the CNN engine such as the reset state, weight loading state, MAC computation state, shifting state, and comparison state are triggered sequentially by the top-level control logic depending upon the operating phase of the test-chip.

D. Circuit Description

1) Pulse Generator and Pulse Selector Modules: The pulse generation and selection methodology is adapted from [16].

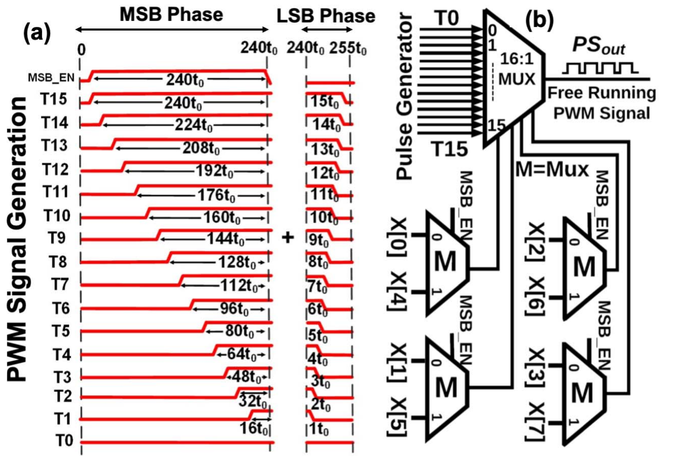


Fig. 7. (a) PWM signals generated by the pulse generator module. (b) Pulse selector circuit (adapted from [16]).

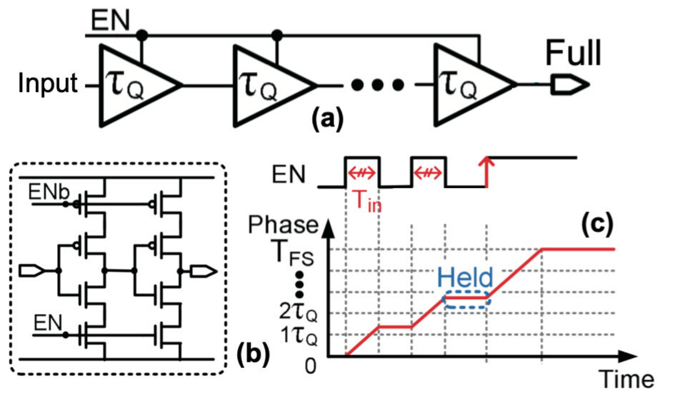


Fig. 8. (a) Time register circuit. (b) Gated delay cell. (c) Timing diagram [26].

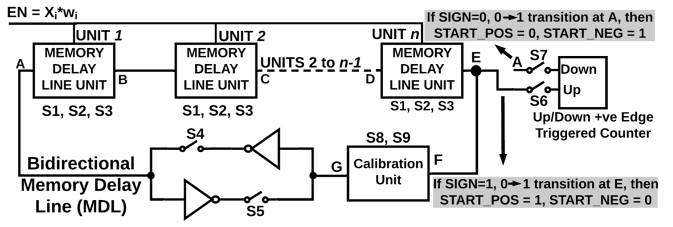

EN = Xi*wi
UNIT 1
UNIT 2
UNIT n
MEMORY DELAY LINE UNIT
S1, S2, S3
UNITS 2 to n-1
If SIGN=0, 0→1 transition at A, then START_POS = 0, START_NEG = 1
Down
Up
Up/Down +ve Edge Triggered Counter
Calibration Unit
Bidirectional Memory Delay Line (MDL)
If SIGN=1, 0→1 transition at E, then START_POS = 1, START_NEG = 0

Fig. 9. Proposed bi-directional MDL performing time-domain MAC.

Sixteen free-running PWM signals (*T0–T15*) are generated in a pulse generator module such that the pulse width of each signal increments by $17 \times t_o$ and varies in the range of $0 \times t_o$ to $255 \times t_o$, as shown in Fig. 7(a). Here, t_o represents the minimum possible pulse width and equals half period of input clock (e.g., for input clock frequency of 25 MHz, t_o equals 0.5×40 ns = 20 ns). These pulses are generated in two phases: the MSB phase and the LSB phase. In the MSB phase, the pulse width is incremented by $16 \times t_o$, whereas in the LSB phase, it gets incremented by t_o. Thus, $256 \times t_o$ or 128 input clock cycles (equals 1 *MAC_CLK* cycle) are consumed to generate these 16 PWM signals. To avoid overlapping of PWM signals between consecutive *MAC_CLK* clock cycles, an intentional delay of t_o is added at the start of each PWM signal ($T0$–$T15$). The input pixel value is then time-encoded using any of these 16 PWM signals in the MSB and LSB phases. 4 MSB bits of the input-pixel value (*X[7:4]*) select one of the 16 PWM signals for $240 \times t_o$ duration in the MSB phase (when *MSB_EN* is held high) and 4 LSBs (*X[3:0]*) select one of the PWM signals in the remaining $15 \times t_o$ duration in the LSB phase (when *MSB_EN* is held low). The selected PWM signals in both MSB and LSB phases are concatenated in the pulse selector module using one 16:1 multiplexer (MUX) and four 2:1 MUXes to time-encode the input pixel value *X[7:0]* as a free-running PWM signal (PS_{out}), as shown in Fig. 7(b). The pulse generation approach is not based on the logic path delay difference which is susceptible to pulse shrinking and expansion due to process variations. The proposed approach generates different PWMs based on the synchronous input clock operation. Thus, no shrinking and expansion effects are expected in the pulse generation circuit.

2) *Pulse Gating Logic:* As the pulse generator module operates continuously in every *MAC_CLK* period, a pulse gating logic is implemented to ensure that the product of input PWM pulse with the corresponding weight bit is applied only once to the MDL.

3) *Input/Weight Product Computation:* The PWM encoded pixel value is multiplied by the weight value using an AND gate. The weights are stored in a weight register, which is implemented as a 25-bit cyclic shift register. In the weight load phase, a shifter is initialized with filter weights for the convolution layer C1 or C3. In the MAC computation phase, each weight bit is serially shifted out to perform multiplication of the weight bit and the time-encoded input PWM signal using an AND gate.

4) *Time Accumulation Using Bi-Directional Memory Delay Line:* The proposed MDL design is derived from the concept of *time-register*, which is used in high precision TDC [26]. The property of time addition of two different input pulses is utilized to perform the accumulation operation in a MAC computation. The *time-registers* can perform the time addition and time storage using a string of gated delay cells, called the gated delay-line (GDL) which are controlled by an enable (*EN*) signal, as shown in Fig. 8(a) and (b) [26]. The phase increases when the *EN* pulse is high and held constant when *EN* is low [see Fig. 8(c)]. Hence, when an input pulse is received, the phase of the GDL is advanced by the amount of input pulse width and held when the input pulse becomes low. When the input pulse edge reaches the end of the GDL [phase equals the total full scale (TFS) value], a *Full* signal is asserted. The capacity of time-storage can be enlarged by increasing the number of gated delay cells. However, if *EN* is held low for a long duration, the gated delay cells can attain an arbitrary value due to tri-stated outputs. Thus, the *time-register* operation may pose a metastability risk when this GDL is used as a time accumulator during a MAC operation, which may often have zero product due to zero weight bit.

a) Bi-directional MDL: The time domain MAC computations are realized using the proposed Bi-directional MDL, which accumulates the product of weight bit and the time-encoded input pulse width (see Fig. 9). The MDL comprises

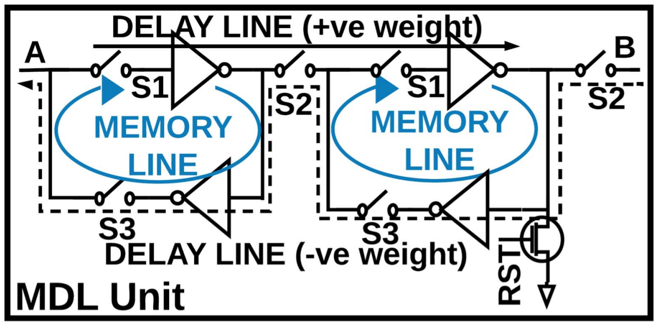


Fig. 10. Proposed MDL unit acting as a delay unit and a memory unit.

a string of delay units (MDL units) to perform signed time accumulation. The two ends of MDL (nodes A and E) are connected through an inverter. Each MDL unit comprises two cross-coupled inverter pairs, S1–S3 switches, and a reset logic (see Fig. 10). Each MDL unit is enabled by the *EN* signal. The product of time-encoded input pixel (X_i) and 1-bit weight (w_i) acts as a *EN* pulse and controls the MDL operating mode. During the time-accumulation phase, *EN* is held high and MDL acts either as a forward delay line (for positive products, S1 and S2 are ON) or a backward delay line (for negative products, S2 and S3 are ON), thus enabling bi-directional data flow emulating signed products. When *EN* becomes low, the MDL acts as a memory storage line and retains the MDL state vector using cross-coupled inverters (S1 and S3 are ON). Thus, the MDL unit can be realized as a memory or delay unit based on S1–S3 switch configurations (see Table I). The metastability risk during *EN* falling transition is resolved by the next incoming *EN* pulse, as the MDL is transformed into a chain of cascaded delay cells. In the case of positive weights (SIGN = 1), MDL state vector propagation takes place in the forward direction with switch S4 being turned on. However, in the case of negative weights (SIGN = 0), MDL state vector propagation takes place in the backward direction with switch S5 being turned on. An up-down counter is triggered (positive edge) whenever the MDL state vector string progresses towards either end of MDL (node A or node E), which translates time-domain product accumulation information into digital bits acting as a TDC. If the accumulated product pulse width exceeds the MDL full-length delay value, an overflow condition is detected, and the propagating edge is inverted (using S4 and S5, see Fig. 9) and applied at the beginning of MDL (node A). Thus, a finite length MDL can be used to perform long duration time domain accumulation using an up-down counter. The calibration unit consists of additional MDL units which can be added to the original MDL to mitigate the delay mismatch among concurrently running MDLs in the presence of process and temperature variations.

b) MDL timing diagram: Fig. 11 shows the MDL timing diagram to illustrate the MDL operation in more detail. Assuming that the initial state of MDL outputs is all "0s," and node A is held high (logic value is "1"). The first time-encoded product is loaded onto MDL and the logic value "1" at A is propagated through MDL units for the time duration when *EN* is held high. When *EN* becomes low, the logic value "1" might not get latched properly in the latest MDL unit propagating the *EN* pulse, thereby arousing a metastability issue. Since the MDL state vector is represented as a string of 1s followed by a string of 0s or vice versa, the metastability risk latching the correct value is at the transition of 1s string to 0s string. Thus, MDL may represent the input time-encoded product, which is off by 1 MDL unit delay. In the next *MAC_CLK* period, the second time-encoded product is applied by holding *EN* high and the state vector of "1s" advances through MDL for the time duration when *EN* is held high. The metastability risk (if any) from previous product accumulation gets resolved when this *EN* is applied, since MDL now acts as a delay line and the metastable node gets driven by the previous MDL unit. In the next *MAC_CLK* cycle, the third product of the time-encoded input PWM signal and the negative weight is applied. As observed in Fig. 11, state-vector propagation occurs in the reverse (backward) direction, i.e., from right to left. Once the state vector of "1s," reaches the end of MDL (node E), the 0→1 transition occurs and the counter value is incremented. The overflow value is loaded back on MDL through an inverter and a string of "0s" starts propagating through MDL. Thus, the 0→1 transition is detected at either end of MDL (node A or node E) when a string of "0s" followed by a string of "1s" is propagated. Thus, a finite MDL length along with an up-down counter can be used to perform signed time accumulation of long duration.

c) MDL design considerations for NMOS/PMOS drive strengths' mismatch and slew rate: The mismatch between the

TABLE I
SWITCH CONFIGURATIONS OF MDL, CALIBRATION UNIT, AND TDC

Switch	Logic Equation
S1	$\overline{EN} + (EN.SIGN)$
S2	EN
S3	$\overline{EN} + (EN.\overline{SIGN})$
S4, S8	$SIGN$
S5, S9	$\overline{SIGN}$
S6	$SIGN.\overline{START\_NEG}$
S7	$\overline{SIGN}.\overline{START\_POS}$

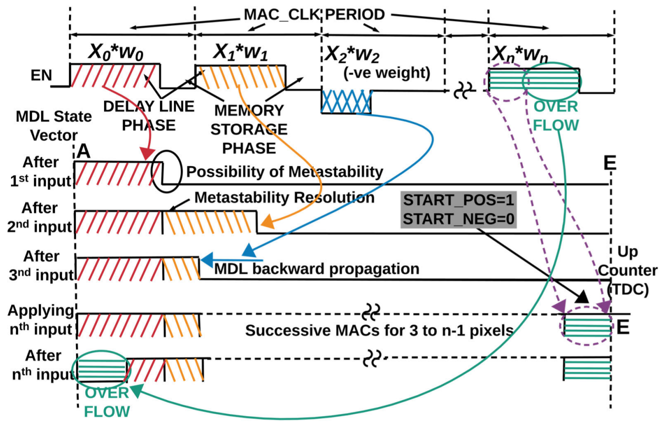


Fig. 11. Timing diagram of the proposed bi-directional MDL.

drive strengths of NMOS and PMOS devices in the MDL units does not affect the accumulation operation. The counter value gets incremented whenever the $0 \rightarrow 1$ transition is observed at either end of MDL (node A or node E). $0 \rightarrow 1$ transition occurs at either end of MDL when a string of 0s followed by 1s is propagated through MDL. Thus, the MDL full-length delay equals the sum of fall delay (when 0s propagate through MDL) and rise delay (when 1s propagate through MDL) for all the MDL units. Since the counter gets incremented/decremented by 1 when the *EN* of pulse width equal to 1 MDL full-length delay (sum of equal number of rise and fall delays for all MDL units) is propagated, the relative difference between the rise and fall delays of individual MDL units does not affect the MDL accuracy.

All the MDL units have a fanout of 1. The layout of MDL is done using the commercial place and route (P&R) engineering design automation (EDA) tools to optimize for the timing and meet the slew rate specifications. All the MDL units are placed close to each other to optimize for the performance-power-area (PPA) metric. The optimally designed MDL block is instantiated (replicated) four times, in a filter for performing 2×2 max-pooling. The proposed time-domain CNN engine instantiates 16 such filters. Thus, all the MDLs have identical placement and routing. Thus, any potential slew rate mismatch due to random MDL unit cell placement and routing is mitigated by using identically designed and placed individual MDL blocks.

5) Time-to-Digital Converter: A 20-bit positive edge triggered, up-down counter is used to convert the time-encoded MAC value into the digital domain. The counter value is incremented when the $0 \rightarrow 1$ transition occurs at node E (if switch S6 is turned on and SIGN $= 1$) and is decremented when the $0 \rightarrow 1$ transition occurs at node A (if switch S7 is turned on and SIGN $= 0$). The configuration of switches S6 and S7 is determined based on *START_POS* and *START_NEG* signal values (see Table I). Whenever the $0 \rightarrow 1$ transition occurs at node E while accumulating negative products (SIGN $= 0$), *START_NEG* is set, whereas *START_POS* is reset. Similarly, the *START_NEG* signal is reset and the *START_POS* signal is set when the $0 \rightarrow 1$ transition occurs at node E while accumulating positive products (SIGN $= 1$), as shown in Fig. 9. If the $0 \rightarrow 1$ transition occurs at node E while accumulating positive products (SIGN $= 1$) and *START_NEG* is high, switch S6 remains turned off and the counter value is not incremented, since the negative residue loaded on MDL from the previous stage has not been fully flushed. Thus, the control logic determining the configuration of S6 and S7 switches ensures that the counter value is incremented and decremented correctly to compute signed MAC operation in the time domain.

6) Scaling and MAV Computation Using Bi-Directional Barrel Shifter: The counter output value is fed to the 20-bit bi-directional barrel shifter (supporting the left shift and the right shift up to 7 bits) to compute MAV and scaling operations.

The counter output value represents how many times the MDL full-length delay has been traversed. It does not represent the absolute MAC output value since the time-encoded input-pixel value and the counter output value are represented in different time scales. Thus, a scaling operation needs to be performed to restore the correct MAC value by incorporating the scaling factor. This scaling operation ensures that the correct MAC outputs in terms of counter values are applied as inputs to the next convolution layer. The *EN* pulse width of a unit input-pixel value is t_o, where $2 \times t_o$ is the input clock period (as discussed in Section III-D1, Fig. 7). This pulse width for a unit pixel-value can be controlled by varying the input clock frequency. The full-length delay of MDL, which is the delay between consecutive $0 \rightarrow 1$ transitions (when a string of “0s” followed by a string of “1s” propagates through MDL), is $M \times (Unit_{Rise\_Delay} + Unit_{Fall\_Delay})$, where M is the number of MDL units, $Unit_{Rise\_Delay}$ is the rise propagation delay, and $Unit_{Fall\_Delay}$ is the fall propagation delay of each MDL unit. This MDL full-length delay value can be controlled by varying the supply voltage or number of MDL units. The *EN* pulse width to increment counter by 1 is equal to the MDL full-length delay value. Thus, the scaling factor (multiple of 2^n by controlling the input clock frequency and the supply voltage) needs to be accounted to scale back the counter value, as given by (3). The counter output needs to be multiplied by this scaling factor to reflect the actual MAC output. Typically, t_o is less than the MDL full-length delay value. Thus, the shifter performs left shifts (multiplies) by n bits to restore the MAC output.

To perform the averaging operation, the right shift by appropriate number of bits is performed. It is worth mentioning that the averaging factor in the MAV computation operation is a multiple of 2^m, such that $2^m \geq N$, where N is the number of products in a MAC. For example, in the C1 layer $N = 25$, whereas averaging is performed by 32 by shifting right the scaled counter output from the above step by 5 bits. The impact of different averaging factors (25 versus 32) is mitigated by training the CNN using 2^m as the averaging factor. This step avoids any loss in the classification accuracy. Thus, the left shift and the right shift by appropriate number of bits are performed in the shifter to perform scaling and averaging operations, respectively

$$\begin{aligned}\text{ScalingFactor} &\\ &= \frac{EN \text{ pulse width of a unit counter value}}{EN \text{ pulse width of a unit input pixel value}}\\ &= \frac{M \times (\text{Unit}_{\text{Rise\_Delay}} + \text{Unit}_{\text{Fall\_Delay}})}{t_o} = 2^n. \qquad (3)\end{aligned}$$

7) Pooling Using 8-bit Comparators: The sub-sampling operation using max-pooling across the 2×2 window is implemented to reduce the intermediate layer output memory footprint by 75%. This is achieved by four concurrent MDL operations and feeding the MDL shifter outputs to three 8-bit comparators (see Fig. 6). It should be noted that the max-pooling operation is not implemented in the time-domain since it requires MDL of very long length to accumulate pulse widths for all the products of a given MAC. Pooling in time-domain requires ORing of four long MDL state vectors to find the largest MDL state vector. Then, the largest MDL state vector needs to be converted back into the digital domain using finite length MDL and counter. Thus, the area

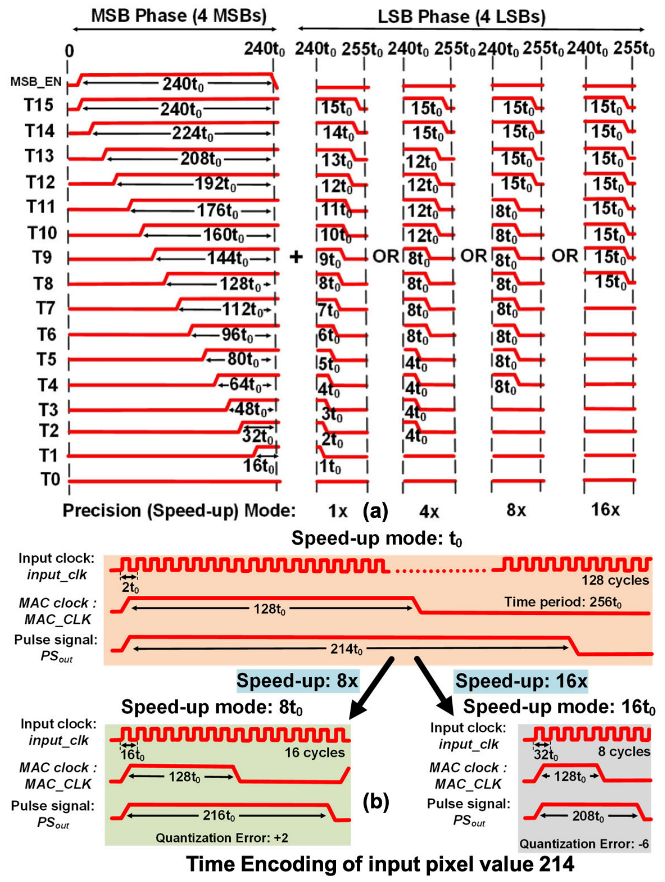


Fig. 12. (a) Quantization of input PWM signals supporting 1–16× speed-up. (b) Demonstration of 1×, 8×, and 16× speed-up modes to time-encode the input pixel value of 214.

Speedup Mode	1x	4x	8x	16x
Input clock period	$2t_o$	$8t_o$	$16t_o$	$32t_o$
MAC clock period	$256t_o$	$256t_o$	$256t_o$	$256t_o$
# input clock cycles/MAC clock cycle	128	32	16	8
Quantization error (input: 0-255)	0	±2	±4	±8

(a)

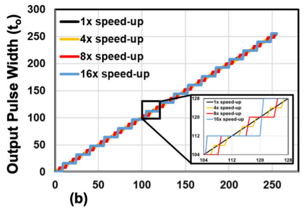


Fig. 13. (a) Throughput versus quantization error. (b) Linearity (pulse width versus input activation value) for 1–16× speed-up modes.

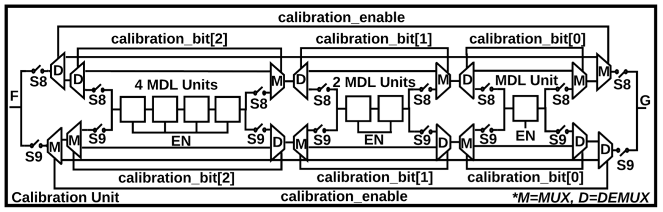


Fig. 14. Calibration unit circuit to mitigate process variations.

overhead of implementing max-pooling in the time domain would be very high. In the proposed work, MDL of finite length is deployed by making use of counter which converts the partially computed time-domain MAC value to the digital domain (as discussed in Section III-D4b). The pooled output from each filter is stored off-chip and reused as the input to the next convolution layer.

IV. Design Trade-Offs and Mitigation Strategies

A. Throughput Improvement Versus Input Quantization

One of the main trade-offs of the time-domain-based MAC computation approach is the low throughput. For a n-bit input pixel value, 2^{n-1} input clock cycles are required to time-encode the input pixel value and perform its product with a weight bit. For example, for a 16-bit input pixel value, *MAC_CLK* period $= 32768 \times (2 \times t_o)$, which would require a longer duration to perform accumulation in the time domain. Hence, the throughput for larger bit-width inputs would be very small in the time-domain approach. To address the low throughput issue associated with the proposed time-domain approach, four speed-up modes are implemented to support 1–16× throughput improvement by quantizing 4-LSBs in the PWM signal. The varying pulse width pulses (*T0–T15*) are generated based on the speed-up mode in the LSB phase and concatenated with *X[7:4]* MSB pulses generated in the MSB phase [see Fig. 12(a)]. Thus, the speed-up mode results in 1–16× throughput improvement to time-encode the input pixel value, as shown in Fig. 12(a). With a higher speed-up mode, the input clock period represents a higher input magnitude. The LSB pulse width quantization steps are chosen to limit the quantization error to ± 0.5 × speed-up ratio [see Fig. 13(a)]. With a higher speed-up mode, the quantization step increases and the linearity between the pulse widths of time-encoded input and the input activation value decreases [see Fig. 13(b)].

B. Process Variation Mitigation Using Calibration Unit

To achieve correct max-pooling output due to concurrent MDL operation, the delay paths in all four MDLs need to be accurately matched for a given input PWM. However, the full-length delay value might not be the same in all four MDLs due to process mismatch or temperature variations. Therefore, a calibration unit is added to each MDL to offset the delay variations, which may exist due to process or temperature variations (see Fig. 14). Due to temperature variations, the full-length delay value of MDLs might get affected. To mitigate such delay variations (systematic delay error component) across all MDLs, the input clock frequency and the supply voltage can be tuned, to ensure that the equation (3) holds true for the same value of scaling factor n under temperature variation. The delay mismatch (random delay error component) due to random process variations among all MDLs for the same input is minimized by appropriately configuring *cal_bit[0:2]* and *cal_enable* signals to add a configurable delay of 1–7 MDL units.

C. Variable MDL Length for Residue Reduction

In the proposed CNN engine, configurable length MDLs are implemented to optimize the MDL full-length delay value. This enables controlling the residual time value in the MDL depending on the input clock frequency. Each MDL comprises 1–4 MDL blocks, where each MDL block comprises 16 MDL

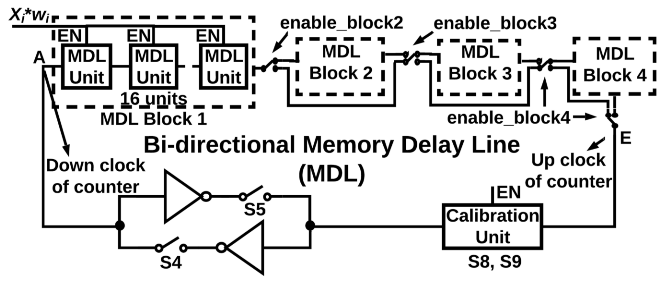


Fig. 15. Configurable MDL length (16–64 MDL units) to address accuracy versus throughput tradeoff.

units (see Fig. 15). A longer MDL results in larger time-residue, which may lower the classification accuracy of the CNN engine. The residual time loss in MDL equals the remainder of total accumulated time/MDL full-length delay. Thus, a longer MDL length implies higher residual time loss, which translates into lower classification accuracy. MDL with a longer length comprises more MDL units. Thus, it leads to more area and higher leakage power, since leakage for more number of MDL units is added.

However, longer MDL supports higher number of products per MAC for the same capacity of counter, as MDL full-length delay is increased. The dynamic power of counter decreases since it is incremented or decremented less number of times due to higher MDL full-length delay. Furthermore, longer MDL can effectively mitigate the process variations since the delay mismatch of each MDL unit gets averaged over a larger number of MDL units, resulting in a consistent full-length delay value across the MDLs. There is no effect on the dynamic power of MDL. Thus, by controlling the MDL length, the residual error can be optimized, so that classification accuracy loss is restricted to a minimal value while operating at low power.

D. MDL Half-Length Delay Offset

As discussed in Section III-D4, the initial state of each MDL unit output node value is held at the logic value "0," and node A is held at the logic value "1." Then, the first positive edge transition (0→1) at node E occurs by propagating node A value "1" for the *EN* duration until it reaches node E. Thus, the counter value gets incremented by propagating only a string of "1s" in the first round, instead of propagating "0s" followed by "1s." Thus, the intentional half of MDL full-length delay value is added upfront. For subsequent counter increments, the string of 0s followed by string of 1s needs to reach node E. This ensures that the final residual time value of the MDL is ± half of the MDL full-length delay value, instead of 1 full-length delay value.

V. Measured Results

A. Measurement Setup and Test-Chip Summary

Fig. 16 shows the overall measurement setup of the test-chip implemented using commercial 40-nm CMOS process technology [27]. The test-chip implements the LeNet-5 CNN architecture and occupies a total area of 0.124 mm^2

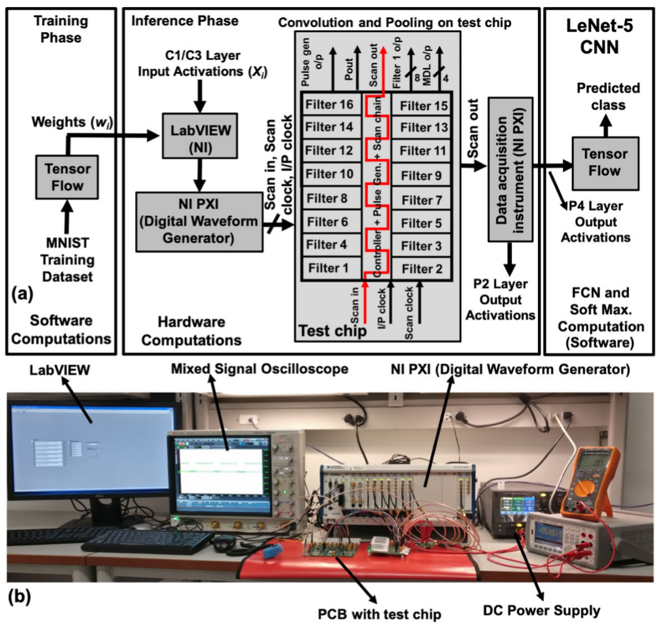
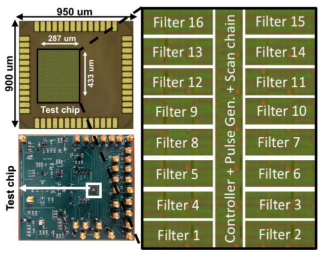


Fig. 16. (a) Block diagram of the proposed time-domain CNN engine setup. (b) Laboratory measurement setup.



Technology	40nm CMOS
Area	0.124mm^2
# Standard cells	~40,000
# Filters supported	16
Data type supported	Signed/Unsigned
Voltage range	0.375V - 1.10V

Fig. 17. Die micrograph and a summary of 40-nm test-chip implementing the time-domain CNN engine.

(see Fig. 17). The overall CNN inference and training methodology consists of: 1) training input data using the TensorFlow/Keras [28] software framework; 2) feeding trained filter weights and MNIST validation dataset image pixel values to the test-chip using LabVIEW-PXIe data acquisition instruments [29]; 3) performing on-chip MAC, averaging, and pooling operations for both C1 and C3 convolution layers; and 4) FCN layers and soft-max computation in software (TensorFlow). Due to the absence of on-chip global buffers (SRAM), the input activations, the weights, and the output activations (pooled MAV value) are applied/captured using the scan chain interface for each MAC computation. The National Instrument PXIe instrument is used to record the scan chain outputs. Each image is classified as two steps—storing output activations for layer C1 and for layer C3.

B. Test-Chip Characterization

The experimental demonstration of a 64-unit long MDL behavior under the delay phase and the memory phase is shown in Fig. 18(a). Three output node values (after 32 units, 48 units, and 64 units) are observed as shown in the oscilloscope capture. These waveforms are phase shifted in time

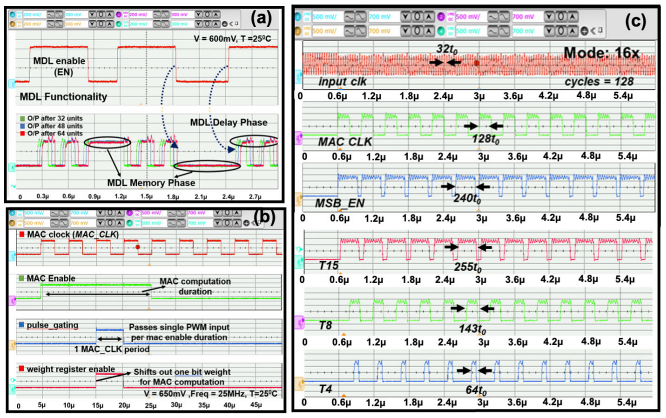


Fig. 18. Measured (a) MDL waveforms demonstrating delay and memory phases, (b) pulse gating logic module waveforms, and (c) PWM signals generated from a pulse generator module.

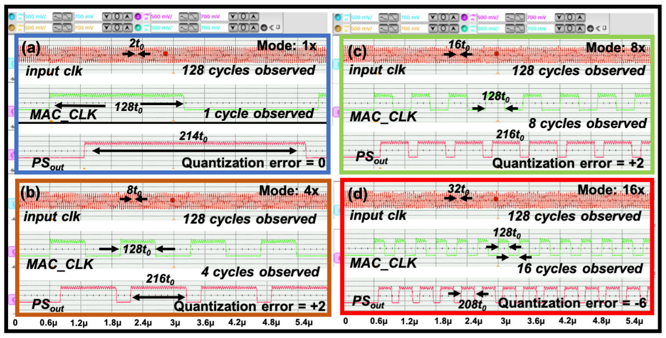


Fig. 19. Experimental demonstration of (a) 1×, (b) 4×, (c) 8×, and (d) 16× speed-up modes.

which confirms the successful operation of the proposed MDL concept. During the MDL delay phase (when *EN* is held high), the state vector on MDL is advanced, thereby resulting in transitions, whereas the MDL state is held constant during the memory phase (when *EN* is low). The pulse gating logic waveforms are shown in Fig. 18(b), confirming that the time-encoded input pixel value is used only once in the MAC computation. The pulse generator module functionality is verified with the correct toggling of *MSB_EN*, *T15*, *T8*, and *T4* outputs in the 16× speed-up mode, as shown in Fig. 18(c). 1–16× speed-up in PWM input representation is validated with multiple speed-up modes for a test-case input of 214 (see Fig. 19). In the 1× speed-up mode, 128 input clock cycles are used to time-encode the input pixel value of 214 accurately. However, the pixel value of 214 is encoded four times in 4× speed-up mode for the same number of 128 input clock cycles, thereby increasing the throughput by 4×, but with a quantization error of 2. Similarly, 16 and eight cycles of input clock are required in 8× and 16× speed-up modes to time-encode the 214-pixel value as 216 and 208, respectively. Thus, the PWM throughput increases with a higher speed-up mode at the expense of higher quantization error.

C. Accuracy, Throughput, and Energy Efficiency Results

The classification accuracy is measured for LeNet-5 CNN over 100 MNIST dataset images. Table II lists key LeNet-5 network parameters. 8-bit fixed point input pixel values and

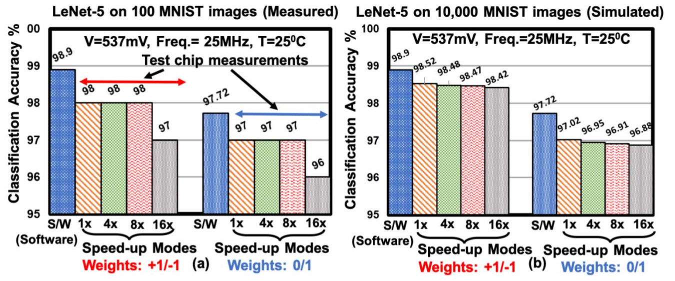


Fig. 20. Classification accuracy on LeNet-5 CNN for 4 speed-up modes using signed/unsigned weights at 537 mV (a) measured over 100 images and (b) simulated over 10 000 images accounting for MDL non-ideal effects.

TABLE II
PARAMETERS OF LeNeT-5 CONVOLUTION LAYERS C1 AND C3

Parameters for LeNet-5	C1	C3
Filter Size	5*5*1*6	5*5*6*16
Input/Filter Size	8bits/ 1bit	8bits/ 1bit
Input Size	32*32*1	14*14*6
Output Size	14*14*6	5*5*16
#Filters	6	16
#Operations/convolution$	(25*6*4)*2	(150*16*4)*2

$1 MAV = 2 operations: 1 multiply + 1 add/average, #MACs computed in parallel = 4

binary signed/unsigned weights are used in the convolution layers, whereas 16-bit floating point inputs and weights are used in the fully connected layer and software implementation. For signed binary weights (±1), 98% classification accuracy is obtained in 1–8× speed-up modes, whereas it is degraded by 1% in 16× speed-up mode [see Fig. 20(a)]. 16× speed-up mode resulted in lower accuracy because of input quantization and increased sensitivity of the MDL residue. For unsigned weights (0 and 1), 1% accuracy drop with respect to the signed weights case is observed. The measured accuracy values are within 1-2% of the state-of-the-art 16-bit floating point software implementation, which confirms the overall functionality of the proposed MDL based time-domain MAC computing approach.

The measurements are performed over 100 validation set MNIST images for signed and unsigned weights in all the 4 speed-up modes. In total, classification accuracy is experimentally measured for 100 × 2 × 4 = 800 images. As mentioned in Section V-A, the output activations (pooled MAV value) are captured for each MAC value from the test-chip using a scan chain due to the absence of on-chip SRAM. The National Instrument PXIe instrument is used to interface with the scan chain resulting in very slow data acquisition from the test-chip.

To fully validate the effectiveness of the proposed time-domain approach, LeNet-5 simulations are performed on the entire 10 000 MNIST validation dataset images taking into account the non-ideal effects of MDL, shifter, and pulse generation circuits. The non-ideal effects modeled in the simulation are: 1) quantization of input activations in different speed-up modes; 2) addition of half-length delay value as an offset in MDL; 3) residual time loss in each MAC value depending upon the MDL full-length delay value; and 4) loss

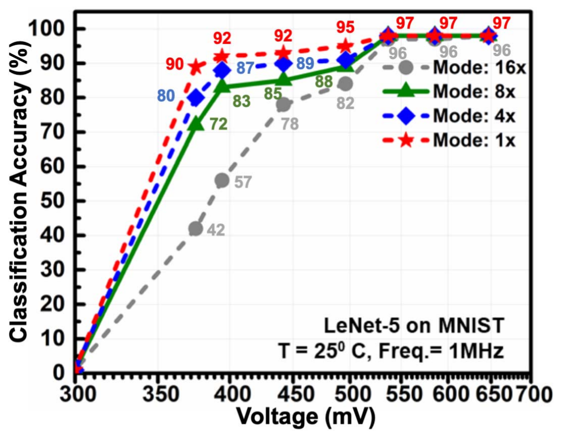


Fig. 21. Measured classification accuracy on LeNet-5 CNN over 100 MNIST dataset test images for 4 speed-up modes and signed weights for the 300–700-mV range.

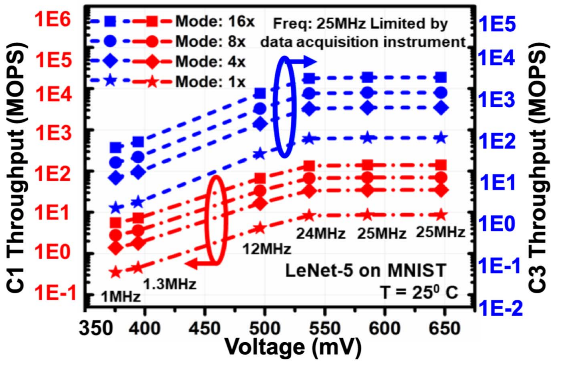


Fig. 22. Measured throughput for convolution layers C1 and C3 of LeNet-5 CNN for different voltages and speed-up modes.

of precision in averaging since it is performed by right shift operation in the bi-directional shifter. The predicted class from this simulation model matches with the experimental results obtained over 100 MNIST images. Fig. 20(b) shows the classification accuracy results obtained using a simulation model over 10 000 MNIST validation set images in all the 4 speed-up modes. 98.42% accuracy is obtained in the 16× mode, which is ≈0.5% lower than the state-of-the-art floating point software accuracy. These results confirm the overall functionality of the proposed MDL based time-domain MAC computing approach. Moreover, MDL supports the ultra-low voltage operation; is functional up to 375 mV with more than 90% accuracy in the 1× speed-up mode. 97% classification accuracy is observed at voltages down until 537 mV in 16× speed-up mode (see Fig. 21). As the supply voltage scales down and approaches the threshold voltage (V_T), increased delay variations are observed in the MDL due to: 1) intrinsic V_T variations in the NMOS and PMOS devices of the MDL unit and 2) variable and degraded input slew rate because of preceding stage MDL transitions. The increased delay variations result in variable MDL full-length delay value. Thus, MAC values are not computed accurately and the classification accuracy is degraded.

For LeNet-5 CNN, both convolution layers C1 and C3 measured throughput increases with higher speed-up mode and with increasing supply voltage, achieving a peak

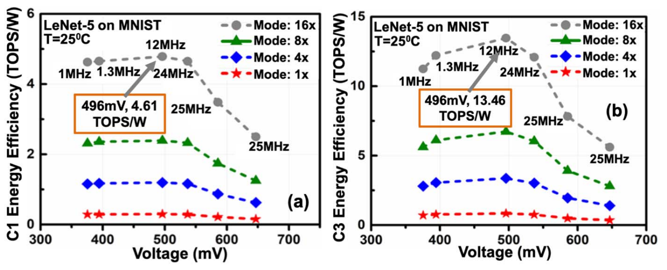


Fig. 23. Measured energy efficiency for convolution layers C1 and C3 of LeNet-5 CNN for different voltages and speed-up modes.

TABLE III

Performance Summary of Proposed Time-Domain CNN Engine Implementing Convolution Layers C1 and C3 of LeNet-5

LeNet-5 Results/Metrics	Convolution Layer – C1				Convolution Layer – C3			
Speedup Mode	1x	4x	8x	16x	1x	4x	8x	16x
Input clock frequency (MHz)	24.0	24.0	24.0	24.0	24.0	24.0	24.0	24.0
MAC clock frequency (MHz)	0.19	0.75	1.50	3.00	0.19	0.75	1.50	3.00
Convolution Cycle Time (μs)	149.3	37.33	18.67	9.33	842.67	210.67	105.33	52.67
Operating Voltage (V)	0.537	0.537	0.537	0.537	0.537	0.537	0.537	0.537
Power (μW)	28.67	28.67	28.67	28.67	30.17	30.17	30.17	30.17
Throughput (GOPS)	0.008	0.032	0.064	0.128	0.023	0.091	0.183	0.365
Energy Efficiency (TOPS/W)	0.29	1.16	2.33	4.65	0.76	3.02	6.04	12.08

throughput of 0.38(0.128) GOPS for the C3(C1) layer at 585 mV (see Fig. 22). The maximum frequency is limited to 25 MHz due to the test equipment limitation. Thus, no increase in throughput is observed above 550 mV. The measured energy efficiency peaks with supply voltage scaling and reaches a maximum of 13.46(4.61) TOPS/W for C3(C1) layer at 496 mV (see Fig. 23). With the increase in voltage, the energy efficiency increases first until 550 mV and then decreases. This trend is observed since test equipment supports the maximum input clock frequency of 25 MHz, and thus power increases while throughput remains constant above 550 mV. Table III summarizes the test-chip performance. For a supply voltage of 537 mV and an input clock frequency of 24 MHz, an energy efficiency of 12.08 TOPS/W and a throughput of 0.365 GOPS for convolution layer C3 are observed in the 16× speed-up mode.

D. Scalability Analysis for Multi-Bit Weights

In this section, we propose two MDL design approaches to perform time-domain MAC computations for multi-bit inputs and multi-bit weights.

1) Approach 1: Using Multiple Independent MDLs: The time-accumulation of products of the m-bit weight and the time-encoded input pixel value is performed using m MDLs (1 for each weight bit). The counter values from these m MDLs are scaled appropriately (multiplied by powers of 2 depending upon weight bit position), and then added to compute the final MAC value, as shown in Fig. 24. The main limitations of using multiple independent MDLs for computing multi-bit weight/activation MAC operation are: 1) significant loss in classification accuracy due to accumulation of residual time loss across all MDLs and 2) multiple MDLs (m MDLs are required for a m-bit weight value) and an adder circuit to add

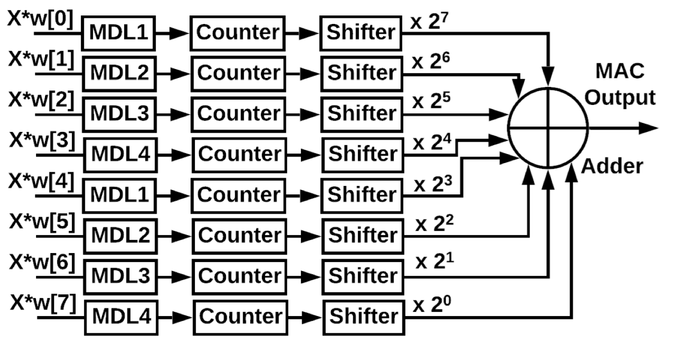


Fig. 24. Illustration of 8-bit weight and 8-bit input MAC operation using eight independent MDLs.

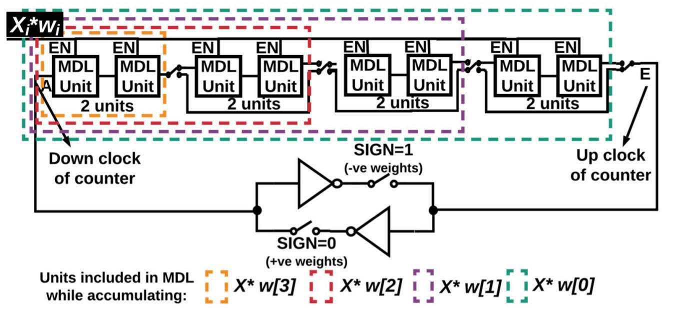


Fig. 25. Illustration of 4-bit weight and 8-bit input MAC operation using MDL with a configurable full-length delay.

the partial sums for each weight bit, thereby increasing the area and power dissipation.

2) Approach 2: Using Single MDL of Configurable Length: The limitations associated with multiple MDLs approach described above can be mitigated by designing a MDL of configurable full-length delay value, in which MDL units are dynamically added depending upon the weight-bit position. The full-length delay value of MDL is scaled by a factor of 2^{-i}, where i is the weight bit position, as shown in Fig. 25. In this approach, all the products of time-encoded input activation with MSB weight bit are first applied on the MDL (minimum MDL length). Then, the MDL length is doubled (full-length delay is doubled) and products of input activations with next significant weight bit are applied. This process is repeated until products with all the weight bits are applied on MDL. Thus, a single MDL is used to accumulate products of multi-bit input activations and weights, and the residual time loss is minimized to improve the classification accuracy. However, this approach requires a minimum of 2^{m-1} MDL units, where m is the bit-width of the weight signal. For example, a 16-bit weight value would require 32 768 MDL units, which can consume a large area and high power. Thus, this approach may pose design challenges to perform MAC operations for weight values with higher bit-widths. Moreover, careful design analysis is required to ensure that the MDL full-length delay value is scaled exactly by powers of 2 in the presence of process and temperature variations. These MDL design trade-offs for higher bit-widths inputs/weights are discussed in Section V-F3.

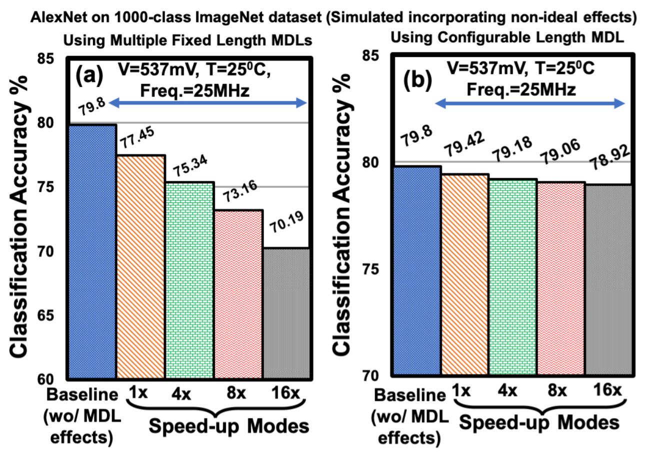


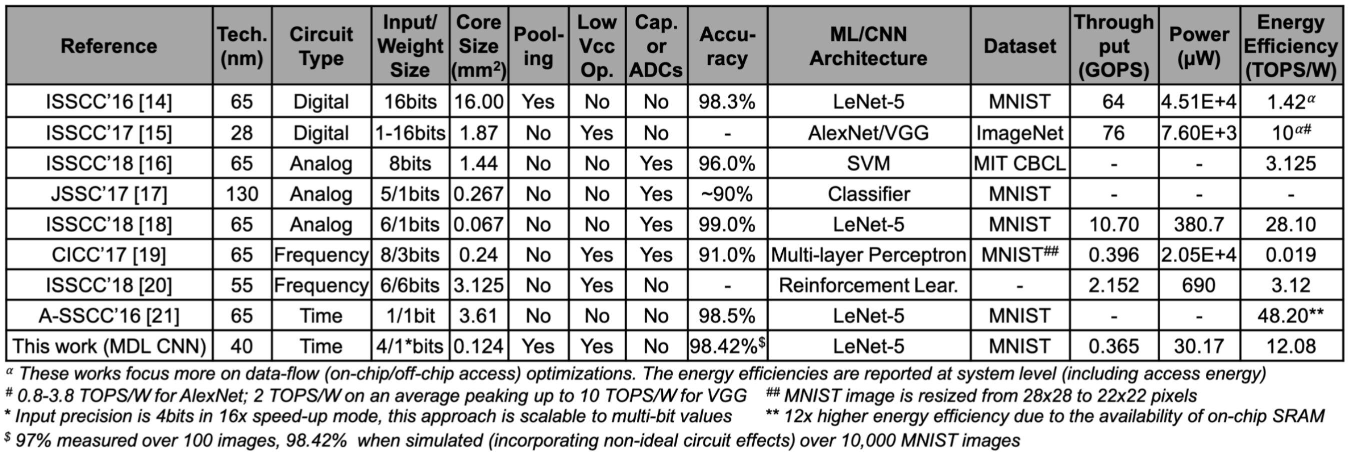
Fig. 26. Simulated classification accuracy on AlexNet CNN over 1000-class ImageNet dataset for different speed-up modes operating at 537 mV, 25 ^{o}C, and 25-MHz input clock frequency using (a) 8 independent MDLs (each 64 MDL units long) and (b) a single MDL with configurable length (1-128 MDL units).

3) Simulation Results on AlexNet CNN: The classification accuracy of the AlexNet network on the 1000-class ImageNet validation dataset (50 000 images) using the proposed multi-bit weight time-domain approaches for all the speed-up modes is evaluated by incorporating the non-ideal effects of MDL, shifter, and pulse generation circuits. The non-ideal effects modeled in the simulation are: 1) quantization of input activations in different speed-up modes; 2) addition of half-length delay value as an offset for each MDL; 3) residual time loss in all the MDLs depending upon the MDL full-length delay value and input clock frequency; and 4) loss of precision in averaging since it is performed by right shift operation in the bi-directional shifter. The full-length delay of MDL is obtained by simulating the MDL at 537 mV and 25 °C. 8-bit fixed point input and weights are used in convolution and pooling layers, whereas 16-bit floating point inputs and weights are used in FCN and soft-max layers. In approach-1, 8 independent MDLs each 64 MDL units long are used. ≈10% classification accuracy loss is observed in 16× speed-up mode when compared with the 16-bit floating point software accuracy [see Fig. 26(a)]. This loss can be attributed to the fact that residue loss in timing accumulation occurs for each weight bit and gets added since eight MDLs are used to compute each MAC value. Moreover, residual time loss is significantly high for MSB bits of weight vector. For approach-2 with configurable MDL length, less than 1% classification accuracy loss is observed using a single MDL with configurable length (1-128 MDL units) in the 16× mode when compared with the 16-bit floating point software accuracy [see Fig. 26(b)], since the residual time loss is minimized in this approach.

E. Comparison With Prior Approaches

Table IV compares the proposed time-domain approach with the earlier proposed digital [14], [15], analog [16]–[18], and frequency domain [19], [20] and time-domain [21] approaches. The comparison includes different metrics such

TABLE IV
COMPARISON OF THE PROPOSED TIME-DOMAIN CNN ENGINE WITH PRIOR ENERGY EFFICIENT ML ACCELERATORS



Reference	Tech. (nm)	Circuit Type	Input/ Weight Size	Core Size (mm^2)	Pool-ing	Low Vcc Op.	Cap. or ADCs	Accu-racy	ML/CNN Architecture	Dataset	Through put (GOPS)	Power (μW)	Energy Efficiency (TOPS/W)
ISSCC'16 [14]	65	Digital	16bits	16.00	Yes	No	No	98.3%	LeNet-5	MNIST	64	4.51E+4	1.42$^{\alpha}$
ISSCC'17 [15]	28	Digital	1-16bits	1.87	No	Yes	No	-	AlexNet/VGG	ImageNet	76	7.60E+3	10$^{\alpha\#}$
ISSCC'18 [16]	65	Analog	8bits	1.44	No	No	Yes	96.0%	SVM	MIT CBCL	-	-	3.125
JSSC'17 [17]	130	Analog	5/1bits	0.267	No	No	Yes	~90%	Classifier	MNIST	-	-	-
ISSCC'18 [18]	65	Analog	6/1bits	0.067	No	No	Yes	99.0%	LeNet-5	MNIST	10.70	380.7	28.10
CICC'17 [19]	65	Frequency	8/3bits	0.24	No	Yes	Yes	91.0%	Multi-layer Perceptron	MNIST$^{\#\#}$	0.396	2.05E+4	0.019
ISSCC'18 [20]	55	Frequency	6/6bits	3.125	No	Yes	No	-	Reinforcement Lear.	-	2.152	690	3.12
A-SSCC'16 [21]	65	Time	1/1bit	3.61	No	No	No	98.5%	LeNet-5	MNIST	-	-	48.20**
This work (MDL CNN)	40	Time	4/1*bits	0.124	Yes	Yes	No	98.42%$^{\$}$	LeNet-5	MNIST	0.365	30.17	12.08

$^{\alpha}$ *These works focus more on data-flow (on-chip/off-chip access) optimizations. The energy efficiencies are reported at system level (including access energy)*
$^{\#}$ *0.8-3.8 TOPS/W for AlexNet; 2 TOPS/W on an average peaking up to 10 TOPS/W for VGG* $^{\#\#}$ *MNIST image is resized from 28x28 to 22x22 pixels*
* *Input precision is 4bits in 16x speed-up mode, this approach is scalable to multi-bit values* ** *12x higher energy efficiency due to the availability of on-chip SRAM*
$^{\$}$ *97% measured over 100 images, 98.42% when simulated (incorporating non-ideal circuit effects) over 10,000 MNIST images*

as the technology node, input/weight bit precision, core size, accuracy, CNN type, dataset on which classification accuracy is reported, low Vcc operation support, throughput, power, and energy efficiency. The classification accuracy of 98.42% (simulated, incorporating circuit non-ideal effects) is achieved over 10 000 MNIST images using the proposed time-domain approach in 16× speed-up mode implementing LeNet-5. This classification accuracy is comparable with the earlier proposed approaches which also perform only on-chip convolution and off-chip fully connected layer operations of LeNet-5: 98.3% in [14], 99% over 100 images in [18], and 98.5% (performing convolution layer C2) in [21]. The proposed MDL based time-domain design implements 16 filters and occupies 0.124 mm^2 in 40-nm CMOS technology. Unlike analog domain approaches [16]–[18], the proposed time-domain approach enables low voltage operation with the MDL operational down until 375 mV. The energy efficiency of the earlier proposed works [14]–[21] ranges from 0.019 to 48.20 TOPS/W. It should be noted that the energy efficiency numbers for earlier approaches as well as for the proposed MDL based time-domain approach depend upon the complexity of the implemented CNN, number of on-chip filters, on-chip global buffer (SRAM) capacity, input and weight bit-widths, operating voltage range, and CMOS technology node implementation.

F. Case Study: Time Domain Versus Digital Domain MAC

As the time-domain approach utilizes standard digital gates with sequential MAC computation, it is worthwhile to compare it with a conventional digital implementation performing sequential partial sum-based MAC computation.

1) Methodology and Design Configurations: A detailed post-PNR (P&R) analysis is performed to compare the average net length (post-Route wirelength), C_{DYN}, area, and energy efficiency of designs performing 1-bit and 8-bit weight MAC operation in conventional digital and proposed time-domain approaches. Both digital-domain and time-domain circuits are synthesized using the Cadence synthesis tool (Genus) and physically implemented using the Cadence P&R tool (Innovus) with standard cell utilization around 70% (see Table V). 40-nm commercial typical design corner libraries (nominal voltage of 1.1 V) are used in both the design approaches. The designs in both approaches are routed using metal layers M1–M4.

To perform MAC operation of 8-bit input and 1-bit weight values in the digital domain, a 15-bit sequential adder is used along with a 15-bit register to add the partial sum value and the product of 8-bit input and 1-bit weight [see Fig. 27(a)]. The input pixel values from 0 to 255 are sequentially added to perform the 1-bit weight MAC operation. The 15-bit adder and the register are used to ensure no overflow occurs when adding these 256 values. The MAC operation of the 8-bit input and 8-bit weight values is computed using a 8-bit × 8-bit multiplier, 28-bit adder, and 28-bit register in the digital domain [see Fig. 27(b)]. The input and weight values are selected randomly and the MAC operation is computed for a filter size of 3 × 3 × 256 (equals AlexNet convolution layer C3 filter size). The input and weight values are randomly selected 2304 times, and each product of input and weight value is added to compute a MAC value. Thus, 28-bit adder/register sizes are chosen to ensure no overflow occurs in adding 2304 16-bit input/weight product values. In the time-domain approach, the proposed circuit (16-unit MDL) is used along with the 15-bit counter, as shown in Fig. 27(c). Two time-domain approaches are implemented to perform the MAC operation of 8-bit input and 8-bit weight values. Eight copies of 16-unit MDL, eight copies of 28-bit counter/shifter (acts as a counter in the normal mode and a shifter in the shifting mode), and a single 20-bit adder are used to compute MAC operation using time-domain approach-1 [see Fig. 27(d)], whereas the configurable 1–128 unit MDL and 28-bit counter are used in time-domain approach-2 [see Fig. 27(e)] to compute multi-bit input/weight MAC operation in the time domain. These multi-bit weight time-domain approaches are discussed earlier in Section V-D.

2) C_{DYN}, Area, and Energy Efficiency Analysis: Table V lists the important design metrics for both these approaches using the same 40-nm CMOS standard digital gates. In the

TABLE V

AREA, C_{DYN}, AND ENERGY EFFICIENCY COMPARISON OF MAC COMPUTATION IN THE DIGITAL DOMAIN AND THE PROPOSED TIME-DOMAIN APPROACH

Metrics	MAC Computation - Inputs: 8-bit, Weight: 1-bit					MAC Computation - Inputs: 8-bit, Weight: 8-bit								
	Digital Domain	Proposed Time Domain 1-bit Weight			Comparison Digital/Time	Digital Domain	Proposed Time Domain Multi-bit Weight Approach 1			Proposed Time Domain Multi-bit Weight Approach 2			Comparison Digital/ Approach1	Comparison Digital/ Approach2
		MDL + TDC	Pulse Gen. Overhead	Total			MDL + TDC	Pulse Gen. Overhead	Total	MDL + TDC	Pulse Gen. Overhead	Total		
Number of nets	194	44	13	57	3.40	990	450	7	457	286	7	293	2.17	3.38
Average Net length (um)	6.35	5.64	8.97	6.40	0.99	7.63	7.19	8.97	7.22	7.17	8.97	7.21	1.06	1.06
Total Capacitance (pF)	0.610	0.134	0.046	0.180	3.39	5.119	2.1755	0.023	2.1985	1.4388	0.023	1.4618	2.33	3.50
Total CDYN (α*C) (pF)	0.283	0.0799	0.0423	0.1222	2.31	1.879	0.876	0.0215	0.8975	0.779	0.0215	0.8005	2.09	2.34
Standard Cell Util. (%)	72.45%	72.08%	72.88%	-	Same (~1)	70.10	70.60	72.88	-	69.25	72.88	-	Same (~1)	Same (~1)
Area (um^2)	529.00	169.00	37.51	206.51	**2.56**	3600	1600	18.76	1618.76	1024	18.76	1042.76	**2.22**	**3.45**
Power (uW)	570.60	9.62	5.118	14.738	38.72	1516.20	105.94	2.60	108.54	11.34	2.60	13.94	13.97	108.77
F_{max} (GHz)	1.67	0.10	0.10	0.10#	16.70	0.667	0.10	0.10	0.10	0.0125	0.10	0.0125##	6.67	53.36
Energy – Power/F_{max} (pJ)	0.3417	0.0962	0.05118	0.1474	**2.32**	2.273	1.059	0.026	1.085	0.907	0.026	0.933	**2.09**	**2.29**

# Pulse Generation and Time Accumulation are Pipelined. Hence, total time taken is 10ns. ## Pulse Generation takes 10ns, Time Accumulation takes 80ns; both are pipelined. Hence, total time taken is 80ns.

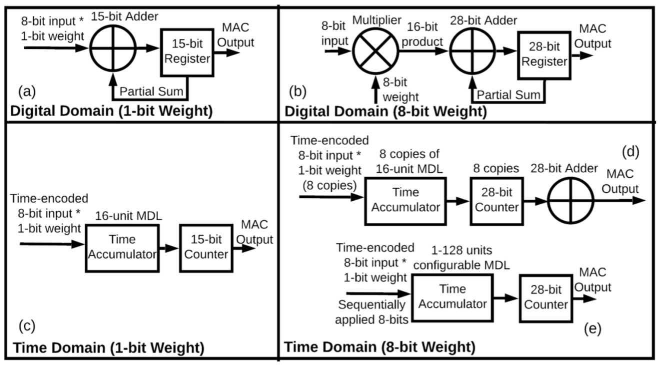


Fig. 27. Illustration of designs performing MAC operation for 8-bit inputs. (a) 1-bit weight in the digital domain. (b) 8-bit weight in the digital domain. (c) 1-bit weight in the proposed time domain. (d) 8-bit weight in the proposed time-domain approach-1. (e) 8-bit weight in the proposed time-domain approach-2.

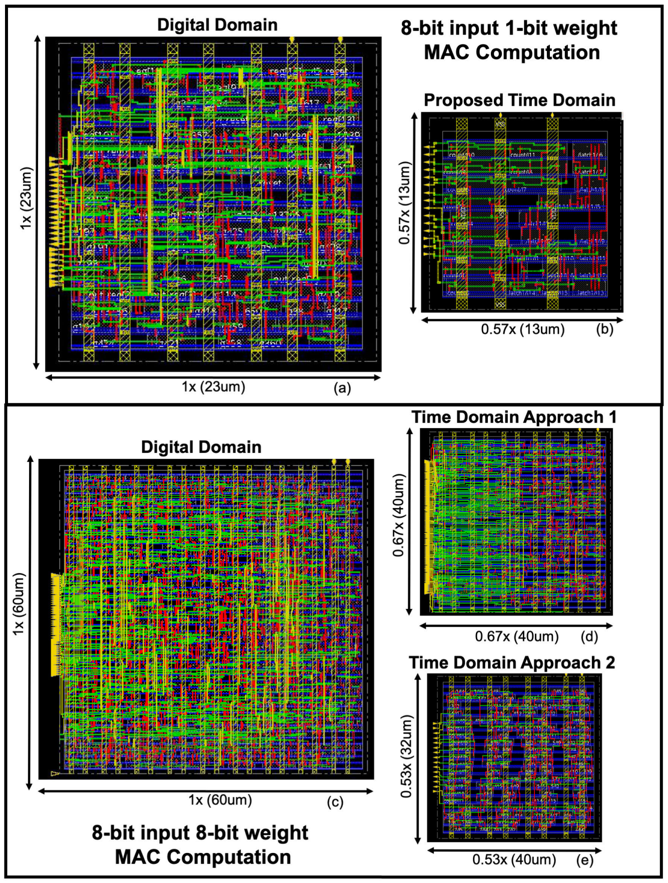


Fig. 28. Post-PNR floorplan of designs performing MAC operation for 8-bit inputs. (a) 1-bit weight in the digital domain. (b) 1-bit weight in the proposed time domain. (c) 8-bit weight in the digital domain. (d) 8-bit weight in the proposed time-domain approach-1. (e) 8-bit weight in the proposed time-domain approach-2.

proposed time-domain approach, the input pixel value in the digital domain is converted into a time-domain signal using the pulse generator and selector modules. This overhead is taken into consideration while evaluating the net length, C_{DYN}, area, power, and energy efficiency of the time-domain approach. It is assumed that 16 filters are implemented to compute MAC operation for 1-bit weight MAC operation (implementing LeNet-5), and 32 filters are implemented to compute MAC operation for 8-bit inputs and weights (implementing AlexNet). Each filter comprises four MDLs to compute four MAC values. Thus, the pulse generation overhead for each MDL is evaluated by dividing its value by 64 (16×4) and 128 (32×4) for 1-bit weight and 8-bit weight MAC computation, respectively.

The proposed time-domain implementations to compute MAC operation for 1-bit and 8-bit weight values result in 2.17–3.40$\times$ reduction in the number of nets, 2.33–3.50$\times$ reduction in total capacitance, and 2.09–2.34$\times$ reduced switching capacitance when compared with designs implemented in the digital domain (see Table V). The time-domain design implementations are compact and results in 2.22–3.45$\times$ smaller area when compared with digital design implementations, as evident in post-PNR layouts shown in Fig. 28. Since the time-domain approach operates at the *MAC_CLK* frequency which is smaller than the input clock frequency of the digital domain approach, 28-bit adder implementation in the proposed time-domain approach-1 [see Fig. 28(d)] is compact than the 28-bit adder implementation in the baseline digital approach [see Fig. 28(c)]. This translates into significant ($\approx 2\times$) area benefit for the proposed time-domain approach-1 over the digital domain approach.

The vector-based power analysis using the Value Change Dump (VCD) file [30] (generated from functional simulation) is performed in the Synopsys primetime (PTPX) tool to accurately estimate the power in both digital and time-domain approaches. For 1-bit weight MAC computation, input pixel values from 0 to 255 are sequentially applied in both digital- and time-domain approaches to estimate the power of adding an 8-bit input-pixel value to the partial sum value. The input toggle rate (including both rise and fall transitions) varies in the range of 0.0078-1 for all the input pins (1 for the LSB and decreases by a factor of 2 for the next significant bit).

For 8-bit weight MAC computation, the input and weight values are selected randomly and the MAC operation is computed for a filter size of 3 $\times$ 3 $\times$ 256 (equals AlexNet convolution layer C3 filter size). The input and weight values are randomly selected 2304 times, and each input $\times$ weight product is added sequentially to compute a MAC value. The average power varies by less than 1% when estimated over 30 runs (each run with different random number generator seed values), which ensures that the reported average power in Table V is a good representation of typical power consumption. The input toggle rate varies in the range of 0.20-0.35 for all the input pins. Using these power values and the maximum clock frequency supported by these design configurations, the energy for MAC operation is quantified. The proposed MDL based time-domain circuits result in 2.09–2.32$\times$ better energy efficiency when compared with its digital design implementations at an iso-voltage (see Table V).

3) MDL Design Limitations for Higher Input/Weight Bit-Widths: The pulse generation scheme in the proposed time-domain approach requires 2^{n-1} input clock cycles to time-encode a n-bit input pixel value, thereby significantly degrading the throughput (lower *MAC_CLK* frequency) for inputs with higher bit-widths (e.g., 16 or 32 bit). The power consumption of MDL and TDC scales down appropriately at lower *MAC_CLK* frequencies such that the energy efficiency (Power/Fmax) of MDL remains the same. However, pulse generation/selection logic power consumption almost remains the same since the input clock frequency is unchanged. Thus, the power of pulse generation/selection modules starts dominating the overall time-domain CNN engine power, thereby reducing the energy efficiency of the proposed time-domain approach. Moreover, the proposed time-domain approach using multiple independent MDLs (approach-1) to perform the MAC operation of weights with higher bit-widths may result in significant classification accuracy loss, since residual loss would be accumulated for higher number of weight bits. Using a single MDL with configurable length (approach-2) requires 2^{m-1} MDL units to perform the m-bit weight MAC operation. Hence, it may lead to a significant increase in area for higher values of m (e.g., 16 or 32 bit). Thus, better MDL/pulse generation design solutions need to be devised to scale the proposed MDL based time-domain approach to larger bit-width inputs and weights.

VI. Conclusion

In this article, an energy efficient time-domain CNN engine suitable for edge computing is demonstrated. The proposed time-domain CNN engine deploys a bi-directional MDL to perform the signed accumulation of input and weight products. The fully digital and technology scaling friendly design is compact and does not use any capacitors and data converters (DACs and ADCs). It supports near-threshold voltage operation and is useful for low power edge computing applications. Four speed-up modes and a configurable MDL length are supported to address the throughput versus accuracy trade-off. The proposed design is tolerant to process variations, and the delay mismatch among MDLs are offset by the calibration unit. A 40-nm CMOS test-chip implementing the LeNet-5 CNN achieved an energy efficiency of 12.08 TOPS/W and a throughput of 0.365 GOPS at 537 mV in the 16$\times$ speed-up mode. The classification accuracy of 97% measured over 100 MNIST images and of 98.42% by simulating (incorporating circuit non-ideal effects) over 10 000 MNIST images is achieved. Furthermore, two MDL design approaches are proposed to perform MAC operations with multi-bit inputs and multi-bit weights. Simulation results taking into account MDL circuit non-idealities for 1000-class AlexNet over 50 000 ImageNet validation set images show a classification accuracy loss within 1% when compared with the 16-bit floating point software implementation. The proposed MDL based time-domain approach performing 1-bit/8-bit weight and 8-bit input MAC operations when compared with the corresponding baseline digital implementations show 2.09–2.32$\times$ higher energy efficiency and 2.22–3.45$\times$ smaller area.

Acknowledgment

The authors would like to thank V. Radhakrishnan, N. Maheshwari, and J. Rohan for helping with the test-chip measurements. They would also like to thank the Analog Circuit Research Group for helpful discussions during the test-chip implementation and AMD and TSMC University Shuttle Program for the test-chip fabrication support.

References

[1] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[2] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 1701–1708.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 2012, pp. 1097–1105.

[4] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.

[5] B. Moons and M. Verhelst, "An energy-efficient precision-scalable ConvNet processor in 40-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 903–914, Apr. 2017.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2014, *arXiv:1409.1556v6*. [Online]. Available: https://arxiv.org/abs/1409.1556v6

[7] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5687–5695.

[8] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2014, pp. 10–14.

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[10] S. Pan and A. Kak, "A computational study of reconstruction algorithms for diffraction tomography: Interpolation versus filtered-backpropagation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-31, no. 5, pp. 1262–1275, Oct. 1983.

[11] P. Koistinen and L. Holmstrom, "Kernel regression and backpropagation training with noise," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Singapore, vol. 1, Nov. 1991, pp. 367–372.

[12] J. Zhou, W. Xu, and R. Chellali, "Analysing the effects of pooling combinations on invariance to position and deformation in convolutional neural networks," in *Proc. IEEE Int. Conf. Cyborg Bionic Syst. (CBS)*, Beijing, China, Oct. 2017, pp. 226–230.

[13] C.-Y. Lee, P. Gallagher, and Z. Tu, "Generalizing pooling functions in CNNs: Mixed, gated, and tree," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 863–875, Apr. 2018.

[14] J. Sim, J.-S. Park, M. Kim, D. Bae, Y. Choi, and L.-S. Kim, "A 1.42TOPS/W deep convolutional neural network recognition processor for intelligent IoE systems," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Jan./Feb. 2016, pp. 264–265.

[15] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28 nm FDSOI," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2017, pp. 246–247.

[16] S. K. Gonugondla, M. Kang, and N. Shanbhag, "A 42pJ/decision 3.12TOPS/W robust in-memory machine learning classifier with on-chip training," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2018, pp. 490–492.

[17] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6T SRAM array," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[18] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2018, pp. 488–490.

[19] M. Liu, L. R. Everson, and C. H. Kim, "A scalable time-based integrate-and-fire neuromorphic core with brain-inspired leak and local lateral inhibition capabilities," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Austin, TX, USA, Apr./May 2017, pp. 1–4.

[20] A. Amravati, S. B. Nasir, S. Thangadurai, I. Yoon, and A. Raychowdhury, "A 55nm time-domain mixed-signal neuromorphic accelerator with stochastic synapses and embedded reinforcement learning for autonomous micro-robots," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2018, pp. 124–126.

[21] D. Miyashita, S. Kousai, T. Suzuki, and J. Deguchi, "Time-domain neural network: A 48.5 TSOp/s/W neuromorphic chip optimized for deep learning and CMOS technology," in *Proc. IEEE Asian Solid-State Circuits Conf. (A-SSCC)*, Toyama, Japan, Nov. 2016, pp. 25–28.

[22] R. J. D'Angelo and S. R. Sonkusale, "A time-mode translinear principle for nonlinear analog computation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 9, pp. 2187–2195, Sep. 2015.

[23] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, Mar. 2016, pp. 525–542.

[24] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 3123–3131.

[25] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 4107–4115.

[26] K. Kim, W. Yu, and S. Cho, "A 9 bit, 1.12 ps resolution 2.5 b/stage pipelined time-to-digital converter in 65 nm CMOS using time-register," *IEEE J. Solid-State Circuits*, vol. 49, no. 4, pp. 1007–1016, Apr. 2014.

[27] A. Sayal, S. Fathima, S. S. T. Nibhanupudi, and J. P. Kulkarni, "All-digital time-domain CNN engine using bidirectional memory delay lines for energy-efficient edge computing," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2019, pp. 228–230.

[28] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," Mar. 2016, *arXiv:1603.04467*. [Online]. Available: https://arxiv.org/abs/1603.04467

[29] *National Instruments LabVIEW and PXIe*. Accessed: Sep. 20, 2019. [Online]. Available: http://www.ni.com/en-us/shop/labview.html

[30] *ModelSim Command Reference Manual, Mentor Graphics*. Accessed: Sep. 20, 2019. [Online]. Available: https://www.microsemi.com/document-portal/doc_view/136660-modelsim-me-10-5c-reference-manual-for-libero-soc-v11-8

Aseem Sayal (S'18) received the bachelor's degree in electrical and electronics engineering from Delhi Technological University (formerly Delhi College of Engineering), New Delhi, India, in 2013, and the M.S. degree from the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA, in 2017, where he is currently pursuing the Ph.D. degree.

From 2013 to 2015, he was a Physical Design CAD Engineer with Qualcomm India Pvt., Ltd., Bengaluru, India. He was an Intern with Apple Inc., Cupertino, CA, USA, and Google LLC, Sunnyvale, CA, USA, where he was involved in developing clock tree simulation methodology and average power estimation solutions for long running arbitrary workloads. His research is focused on designing energy-efficient hardware accelerators for machine learning applications and developing engineering design automation (EDA) design methodologies for heterogeneously integrated secured application specific integrated circuits (ASICs).

Mr. Sayal was a recipient of the Chancellor Gold Medal and the Meritorious Student Award from Delhi Technological University in 2013.

S. S. Teja Nibhanupudi (S'17) received the bachelor's degree in electrical engineering from IIT Gandhinagar, Ahmedabad, India, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA.

From 2016 to 2017, he was a Project Associate with IIT Gandhinagar, where he was involved in the development of high-voltage process flow at the Semiconductor Laboratory. His research is focused on circuit-device co-optimization using novel materials, such as RRAM, insulator-metal transition (IMT), and STTRAM.

Shirin Fathima received the bachelor's degree in electronics and communication engineering from the Birla Institute of Technology and Science at Pilani (BITS Pilani), Pilani, India, in 2017, and the M.S. degree in electrical and computer engineering from The University of Texas at Austin, Austin, TX, USA, in 2019.

She worked on designing energy-efficient hardware accelerators for image classification in the master's thesis project. She is currently an SoC Physical Design Timing Engineer with Apple Inc., Cupertino, CA, USA.

Jaydeep P. Kulkarni (M'09–SM'15) received the B.E. degree from the University of Pune, Pune, India, in 2002, the M.Tech. degree from the Indian Institute of Science (IISc), Bengaluru, India, in 2004, and the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 2009, all in electronics/electrical engineering.

From 2009 to 2017, he was with the Intel Circuit Research Lab, Hillsboro, OR, USA, where he worked on energy-efficient integrated circuit technologies. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, The University of Texas at Austin, and currently holds the AMD endowed Chair position in computer engineering. He has filed 35 patents and published 75 articles in referred journals and conferences. His research is focused on machine learning hardware accelerators, in-memory computing, emerging nano-devices, hardware security, and radiation hardened electronics.

Dr. Kulkarni was a recipient of the Best M.Tech. Student Award from IISc, the Intel Foundation Ph.D. Fellowship Award, the SRC Best Paper and Inventor Recognition Award, the Purdue Outstanding Doctoral Dissertation Award, seven Intel Divisional Recognition Awards, the 2015 IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS Best Paper Award, and the SRC Outstanding Industrial Liaison Award. He has served as the Conference General Co-Chair for the International Symposium on Low Power Electronics Design (ISLPED) 2018. He is participating in the Technical Program Committee of Custom Integrated Circuits Conference (CICC), Design Automation Conference (DAC), International Conference on Computer Aided Design (ICCAD), and International Conference on Artificial Intelligence Circuits and Systems (AICAS) conferences. He is also serving as an Associate Editor for the IEEE SOLID-STATE CIRCUIT LETTERS and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS.