DVAFS: Trading Computational Accuracy for Energy Through Dynamic-Voltage-Accuracy-Frequency-Scaling

Bert Moons, Roel Uytterhoeven, Wim Dehaene and Marian Verhelst
Department of Electrical Engineering - ESAT/MICAS, KU Leuven, Leuven, Belgium

***Abstract*—Several applications in machine learning and machine-to-human interactions tolerate small deviations in their computations. Digital systems can exploit this fault-tolerance to increase their energy-efficiency, which is crucial in embedded applications. Hence, this paper introduces a new means of Approximate Computing: Dynamic-Voltage-Accuracy-Frequency-Scaling (DVAFS), a circuit-level technique enabling a dynamic trade-off of energy versus computational accuracy that outperforms other Approximate Computing techniques. The usage and applicability of DVAFS is illustrated in the context of Deep Neural Networks, the current state-of-the-art in advanced recognition. These networks are typically executed on CPU's or GPU's due to their high computational complexity, making their deployment on battery-constrained platforms only possible through wireless connections with the cloud. This work shows how deep learning can be brought to IoT devices by running every layer of the network at its optimal computational accuracy. Finally, we demonstrate a DVAFS processor for Convolutional Neural Networks, achieving efficiencies of multiple TOPS/W.**

***Index Terms*—Deep Learning, ConvNet, Convolutional Neural Network, CNN, Approximate Computing, DVAFS, Dynamic-Voltage-Accuracy-Frequency-Scaling, Digital Circuits**

I. Introduction

Several applications in machine-learning and human-to-machine interactions tolerate small deviations in output quality and are hence inherently fault-tolerant. This observation has led to a new class of hardware design techniques known as Approximate Computing [1] [2]. These allow digital processing to approximate results, hence enabling hardware to trade-off energy for accuracy. A processor could as such invest more energy when very accurate data processing is required, and save energy when less accurate processing is permitted. This effectively reduces average energy consumption and creates an extra degree of freedom for system-level power management. Early work allows this trade-off to be installed in digital arithmetic at design time [3] [4] [5] [6], while more recent work is focused on at run-time adaptable systems [7] [8] [9]. Note that this idea of dynamic scaling is analogous to the operation of the human brain, which can selectively increase its efforts when more difficult or precise computations are needed.

Multiple application domains have proven tolerance to such dynamic adaptations. [7] shows the DCT computations in a JPEG encoder can be performed at $4-bit$ accuracy with only 2dB SNR-loss. In [2], various machine learning algorithms ranging from Support Vector Machines to K-means clustering and specifically Deep Learning [10] Neural Networks are identified as fault-tolerant. There is hence an enormous opportunity for increased energy-savings through dynamic accuracy adaptation in aforementioned applications, omnipresent in today's mobiles, wearables and in future IoT applications.

This paper will introduce Dynamic-Voltage-Accuracy-Frequency-Scaling (DVAFS), a new technique which enables a rapid dynamic energy-accuracy trade-off with significant gains compared to existing state-of-the-art techniques in Approximate Computing (section II). After describing this new approach, section III will both evaluate the block and system-level energy gains of DVAFS and compare it to existing methods. Second, DVAFS' applicability to Deep Learning will be illustrated in section IV as an example of a fault-tolerant application. This is further supported with measurements from Envision [11] (section V), a Convolutional Neural Network (CNN) processor implemented in 28nm FDSOI, equipped with the proposed DVAFS principle as a proof-of-concept. Finally, section VI concludes this work.

II. Exploiting dynamic precision requirements

A. DAS: Dynamic-Accuracy-Scaling

An effective approach to dynamically scale the power consumption of e.g. a digital multiplier at constant throughput is through at run-time truncation or rounding of a variable number of its input bits [9]. This introduces deviations in output quality caused by higher quantization errors on the inputs and thus decreases computational accuracy, but also reduces the circuit's internal switching activity and hence energy consumption. If leakage power is neglected, the power of such a Dynamic-Accuracy-Scaling (DAS) system, P_{DAS}, is split into an accuracy-scalable part (as), of which the activity is modulated with scaled precision and a non-accuracy-scalable (nas) part which does not modulate with scaled precision:

$$P_{DAS} = \frac{\alpha_{as}}{k_0} C_{as} f V^2 + \alpha_{nas} C_{nas} f V^2 \qquad (1)$$

where α is the circuit's switching activity, f is the operating frequency, C is the technology dependent switching capacitance and V is the supply voltage. k_0 is a precision, circuit and architecture dependent scaling parameter.

B. DVAS: Dynamic-Voltage-Accuracy-Scaling

A DAS data path element, capable of modulating the computational precision of its input, will exhibit a different critical path length in function of the used computational precision. This is illustrated in Fig. 1a for a simplified $1-4$b multiplier with two operating modes (2b and 4b) highlighted in the figure. The operating modes can be configured by simply gating (or not gating) the least significant bits (LSB) of the inputs. In the high precision mode example of Fig. 1a all building blocks will be active, resulting in a high switching activity and a long critical path. However, in the low-precision

978-3-9815370-8-6/17/$31.00

case, only the most significant bits (MSB) are used, resulting in a lower switching activity and shorter critical path. The DVAS concept [7] [12] exploits this critical path scaling by dynamically scaling voltage in function of the used number of bits, without inducing timing errors. At a fixed frequency, this positive timing slack can then be transformed into energy savings through a lower supply voltage.

These combined effects - (1) reduction of switching activity and (2) shorter critical paths allowing lower supply voltages - have a major impact on the system's dynamic power consumption. The dynamic power P_{DVAS} of a DVAS-system is split into an accuracy scalable as and a non-accuracy-scalable nas part, given in equation 2.

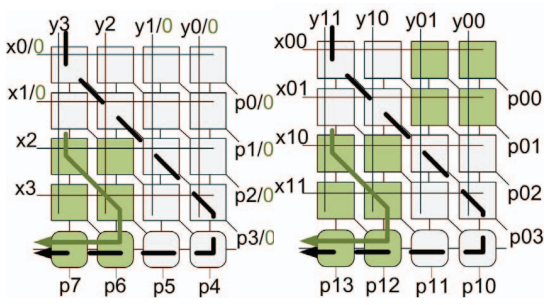
$$P_{DVAS} = \frac{\alpha_{as}}{k_1} C_{as} f (\frac{V_{as}}{k_2})^2 + \alpha_{nas} C_{nas} f V_{nas}^2 \quad (2)$$

As the critical path only scales in arithmetic building blocks, such as multipliers and adders and not in other blocks such as decoders and memory, a significant portion of the original power consumption nas does not scale under reduced precision. As the as and nas parts operate at different voltages, the design should be split into two separated power domains.

C. DVAFS: Dynamic-Voltage-Accuracy-Frequency-Scaling

Subword-parallel DVAFS [11], improves further upon the energy savings of DVAS by reusing inactive arithmetic cells at reduced precision. The example in Fig. 1b shows a $1-4$b subword parallel multiplier that can process two subword operations per cycle if precision is scaled to 2b or lower.

If computational throughput is kept constant, this allows to drop the full system's frequency and hence its voltage significantly below DVAS values. As a result, DVAFS is the first dynamic approximate computing technique which simultaneously lowers all run-time adaptable parameters influencing power consumption: activity α, frequency f and voltage V. In contrast to DVAS, which can only save energy in precision-scaled arithmetic blocks, DVAFS allows lowering f and V of the full system, including control units and memory, hereby shrinking nas energy overheads drastically at low precision.



(a) DVAS example (b) DVAFS example

Fig. 1. (a) A 4b DVAS multiplier example. When 2 MSB's are used, the other inputs are gated. In this case only the green blocks will switch and the critical path is shortened. (b) A 4b DVAFS multiplier example. When 2 MSB's or less are used, redundant logic is reused to make the multiplier subword-parallel. Here, both the critical path and the switching activity are reduced. If throughput is kept constant, the operating frequency can be reduced as well.

TABLE I
D(V)A(F)S PARAMETERS FOR THE MULTIPLIER OF SECTION III-A

parameter	4b	8b	12b	16b
k_0	12.5	3.5	1.4	1
k_1	12.5	3.5	1.4	1
k_2	1.2	1.1	1.02	1
k_3	3.2	1.82	1.45	1
k_4	1.53	1.27	1.02	1
N	4	2	1	1

These combined effects - (1) reduction of switching activity, (2) shorter critical paths allowing lower supply voltages, (3) subword parallel operation allowing lower operating frequencies at constant computational throughput - have a major impact on the system's power consumption. The dynamic power P_{DVAFS} of a DVAFS-system is given in equation 3,

$$P_{DVAFS} = \frac{\alpha_{as}}{k_3} C_{as} \frac{f}{N} (\frac{V_{as}}{k_4})^2 + \alpha_{nas} C_{nas} \frac{f}{N} (\frac{V_{nas}}{k_5})^2 \quad (3)$$

where N is the degree of subword parallelism.

DVAFS is analogous to Dynamic-Voltage-Frequency-Scaling (DVFS) [13], but in DVAFS, voltage and frequency are modulated with varying accuracy rather than varying throughput requirements.

III. DVAFS PERFORMANCE ANALYSIS

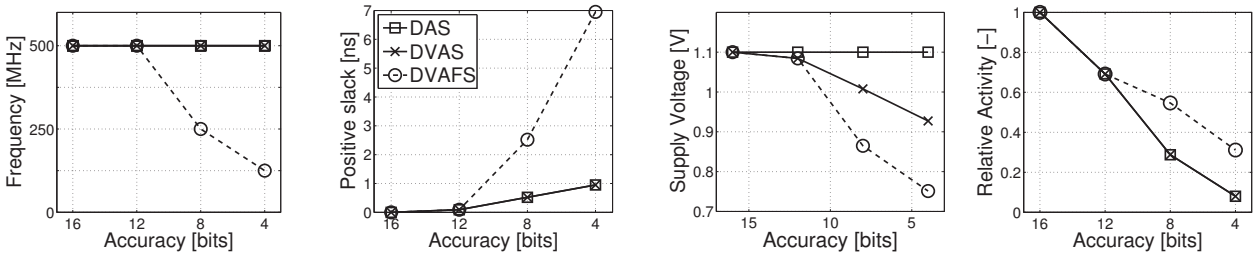
A. Performance of a DVAFS multiplier

We assess the performance and energy-accuracy trade-off of a DVAFS-equipped Booth encoded Wallace tree multiplier through detailed simulations in a $40nm$ LP LVT technology with a nominal supply voltage of $1.1V$. The multiplier is synthesized with commercially available cells in a standard digital flow, using a multi-mode optimization, ensuring the critical path indeed decreases when less bits are used. We use conservative wire models for synthesis and power estimations. In our simulations, multiplier throughput is kept constant at T = 1words/cycle$\times$500MHz= 2words/cycle$\times$250MHz= 4words/cycle$\times$125MHz= 500MOPS, as shown in Fig. 2a.

Figure 2b shows the effect of D(V)A(F)S on the positive circuit delay slack. Without voltage scaling positive slack increases up to $1ns$ if 4b words are processed in the D(V)AS case and up to $7ns$ due to the 125MHz clock in the 4×4b D(V)AFS case. Fig. 2c shows this slack can be compensated for through lowering the supply voltage at constant throughput. For DVAS, a reduction down to $0.9V$ is achievable, resulting in a 36% energy reduction. In DVAFS, supply can go down to $0.75V$, or an additional decrease in energy of 55%.

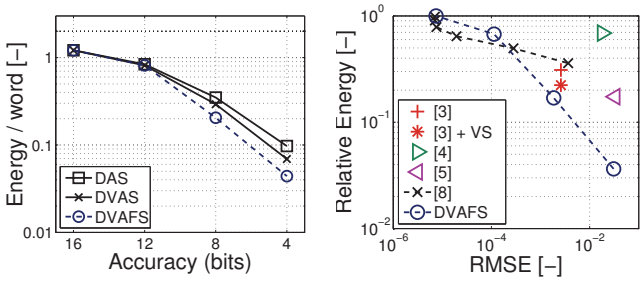
Fig. 2a and Fig. 2d show the respective operating frequencies and activity reduction of the multiplier computing at different levels of accuracy. Switching activity drops $12.5\times$ in the DVAS-case and $3.2\times$ in the DVAFS-case at 4b. However, in DVAFS the frequency is lowered to 125 MHz while it remains constant at 500MHz for all accuracies in DVAS. This analysis allows extracting parameters k_i and N for this multiplier, as given in Table I.

The combination of these reductions in energy consumption leads to global energy-accuracy curves shown in Figure 3a. In this figure we compare the total energy consumption



(a) Operating frequency f @ constant throughput (b) Positive slack / scaling of the critical path @ 1.1V (c) Supply voltages V @ 0 positive slack (d) Activity α modulation

Fig. 2. Operating frequency, slack, supply voltage and switching activity in a custom Booth-encoded Wallace-tree subword-parallel DVAFS multiplier in DAS-, DVAS- and DVAFS-modes. In DVAFS, all run-time adaptable parameters in the digital power equation are modulated: α, f and V.



(a) D(V)A(F)S (b) DVAFS vs [3]–[5], [8].

Fig. 3. Comparison of energy gains in a custom Booth encoded Wallace tree multiplier under DAS, DVAS and DVAFS to previous examples in Approximate Computing [3]–[5], [8] as a function of the used precision.

per computed word of our multiplier (normalized to a non-reconfigurable 16b multiplier) in different modes and as a function of computational accuracy. Note that processing at 16b in the DVAFS multiplier comes at a slight energy penalty, as this reconfigurability leads to a 21% overhead at full precision. At full 16b resolution te reconfigurable multiplier consumes $2.63pJ/word$ compared to the $2.16pJ/word$ baseline in this technology. The DAS case illustrates the sole effect of the decrease in α due to accuracy scaling. Energy drops more significantly in DVAS due to the added voltage scaling. DVAFS achieves energy savings of more than 95% of the baseline at 4×4 DVAFS processing.

The energy-accuracy trade-off in DVAFS is superior to other Approximate Computing techniques, both in terms of energy-reduction as well as in terms of energy-accuracy dynamic range. Figure 3b compares the curve of our multiplier with [3]–[5], [8], with energy relative to the respective fully accurate implementation in each reference and accuracy expressed in terms of Root-Mean-Square-Error (RMSE). Although [8] consumes less energy at high accuracy, the energy-consumption is higher at accuracies lower than 1e-4 RMSE. [3]–[5] do not allow a run-time adaptable trade-off and consume more energy per word at the same level of accuracy.

B. Performance of a DVAFS SIMD-processor

To show the advantages of DVAFS at the system level we implemented and simulated a DVAFS compatible SIMD RISC vector processor in an Application-Specific Instruction set Processor (ASIP) design tool [14], using the same $40nm\ LP\ LVT$ technology as in section III-A. The processor has a parametrized SIMD width SW, denoting the number of data path units and memory banks. In order to be DVAFS compatible, it is subword-parallel and contains multiple power domains. Every SIMD-unit can scale its precision across $1 \times 1 - 16$b, $2 \times 1 - 8$b, $4 \times 1 - 4$b DVAFS modes.

All memories are in a separate power-domain (V_{MEM}), at a fixed $1.1V$ to maintain reliable operation. All other parts of the processor are in two power-domains with variable supply voltage (V_{as} and V_{nas}). As a benchmark, we run a large convolution kernel on the SIMD processor. The performance of the SIMD-processor is illustrated in figure 4, where the energy of the complete processor per processed word is plotted against the used computational accuracy at constant throughput for processor instantiations with different SIMD widths SW. The $SW-$processor in the 1×16b mode at 500MHZ is used as a baseline. The maximal decrease in energy consumption is achieved at the 4×4b DVAFS mode, with a reduction of 85% compared to the baseline. Gains are more modest in the DAS and DVAS (60%) case as the contribution of the as data path to the full processor's energy consumption is smaller.

Table II gives a more detailed overview of the energy distribution of this processor. The energy consumption is distributed over memory (mem), instruction decode & fetch and other overhead (nas) and the vector arithmetic (as). The

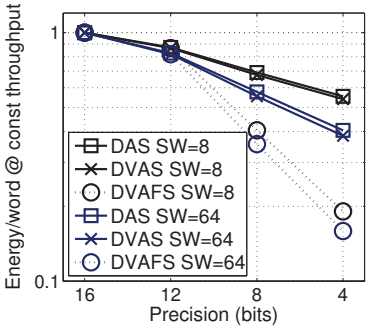


Fig. 4. Energy per word as a function of computational precision consumed by the SIMD processor and memory in different cases at constant throughput. The baseline is the SW used 16b processing units at 500MHz.

TABLE II
POWER DISTRIBUTION AND CONSUMPTION FOR DIFFERENT SETUPS AT $T = SW \times N$WORDS/CYCLE$\times 500/N$MHZ. VOLTAGES ROUNDED TO CLOSEST DECIMAL POINT.

SW	Mode	V_{nas}	V_{as}	mem	nas	as	P [mW]
8	$1 \times 16b$	$1.1V$	$1.1V$	31%	46%	23%	36
8	$1 \times 8b$	$1.1V$	$1.0V$	24%	64%	12%	24
8	$1 \times 4b$	$1.1V$	$0.9V$	17%	77%	6%	20
8	$\mathbf{2 \times 8b}$	$0.9V$	$0.9V$	**39%**	**48%**	**13%**	**15**
8	$\mathbf{4 \times 4b}$	$0.8V$	$0.7V$	**47%**	**44%**	**9%**	**7**
64	$1 \times 16b$	$1.1V$	$1.1V$	31%	32%	37%	289
64	$1 \times 8b$	$1.1V$	$1.0V$	29%	49%	22%	160
64	$1 \times 4b$	$1.1V$	$0.9V$	23%	64%	13%	111
64	$\mathbf{2 \times 8b}$	$0.9V$	$0.9V$	**41%**	**39%**	**20%**	**103**
64	$\mathbf{4 \times 4b}$	$0.8V$	$0.7V$	**53%**	**33%**	**14%**	**45**

highly parallel $SW = 64$ processor has a larger percentage of its energy consumption in the arithmetic circuits and thus has a larger potential for energy reduction than its $SW = 8$ counterpart. This is the case for DAS, DVAS and DVAFS. However, DVAFS shows a much larger energy-accuracy trade-off than the other mentioned techniques, due to its frequency scalability and improved voltage scalability. Thus, in a sole DVAS-system, the parallellism should be large for significant energy gains. In DVAFS, energy scales down significantly also at low levels of parallellism (low SW), as the supply voltage is also lowered in the non-accuracy-scalable (nas) parts.

IV. EMBEDDED DEEP LEARNING THROUGH DVAFS

Deep Learning networks, and more specifically ConvNets or Convolutional Neural Networks (CNN), have come up as state-of-the-art classification algorithms in Computer Vision and Automatic Speech Recognition. Although powerful, these networks are also very computationally and memory intensive, requiring hundreds of megabytes for network storage and billions of operations per input. However, the energy consumption of neural networks can be significantly reduced by explicitly exploiting their error resilience, allowing their employment on battery-constrained systems.

A. Convolutional Neural Network Background

CNNs, visualized in Fig. 5, are a type of artificial neural networks inspired by visual neuroscience. They are a cascade of multiple stacked convolutional-, non-linearity- and pooling-layers used for feature extraction, followed by a smaller number of fully-connected neural network layers used for classification. A **convolutional layer** (CONV), with topology parameters listed in Fig. 5, transforms input feature maps (I) into output feature maps (O), each containing multiple units. Each unit in an output feature map ($M \times M \times 1$) is connected to local patches of units ($K \times K \times C$) in the input feature maps through a filter $W[F]$ ($K \times K \times C \times 1$) in a filter bank W ($K \times K \times C \times F$) existing out of a set of machine-learned weights and a bias (B) per output feature map. A formal mathematical description is given in the following equation:

$$O[f][x][y] = \sum_{c=0}^{C} \sum_{i=0}^{K} \sum_{j=0}^{K} I[c][Sx+i][Sy+j] \times W[f][c][i][j] + B[f] \quad (4)$$

Where S is a stride and x, y, f are bounded by: $x, y \in [0, \ldots, M[$ and $f \in [0, \ldots, F[$.

The result of the local sum computed in this filter bank is then passed through a **non-linearity layer**, typically a Rectified Linear Unit (ReLU), using the nonlinear activation function $f(u) = max(0, u)$, where u is a feature map unit.

Max-pooling layers compute and output only the maximum of a local patch (typically 2×2 or 3×3) of output units in a feature map. They thereby reduce the dimension of the feature representation and create an invariance to small shifts and distortions in the inputs.

Finally, **fully connected layers** (FC) are used as classifiers in the CNN algorithm. An FC layer is described as the matrix-vector product $O[z] = \sum_{m=0}^{M} W[z, m] \times I[m] + B[z]$.

The optimal network architecture, characterized by the number of cascading stages and the values of model parameters F, H, C, K and M, varies for each specific application.

B. Increasing energy-efficiency in CNNs

The energy consumed by the basic algorithms discussed in section IV-A can be minimized in three distinct ways. First, hardware implementations of neural networks can be made more efficient by exploiting algorithm-level parallelism, as well as temporal and spatial data-locality [15]–[19]. Second, as in [20]–[22], CNNs are extremely sparse and can be compressed to reduce their memory footprint [18] and be made more efficient by skipping all insignificant computations [23] [12]. Finally, numerous references [22] [24] [25] show how CNNs are inherently fault-tolerant and can be operated at low computational accuracy with limited recognition loss. Typical benchmarks can be run at 1-9 bit fixed-point rather than 32b floating-point at less than 1% accuracy loss, without any network retraining. Courbariaux et. al [24] even show how networks can be specifically trained to operate using 1b network weights and activations. This observation can lead to major energy-savings, as current CPU and GPU architectures operate using $32 - 16$b Floating-Point number formats. Reducing precision from 32b Floating-Point to low precision Fixed-Point reduces the storage needed for network weights and intermediate results and allows diminishing the energy consumed in any SOC's computational units.

[22] shows the fixed-point precision should not be chosen uniformly for a single neural network accelerator, as the necessary precision varies between applications, networks and even from layer-to-layer within a single network. This is illustrated

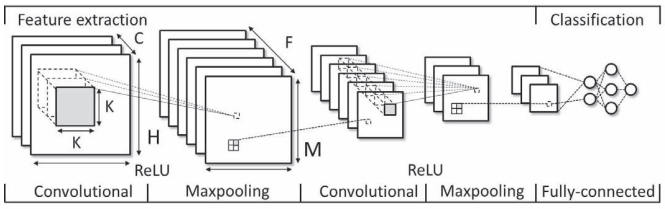
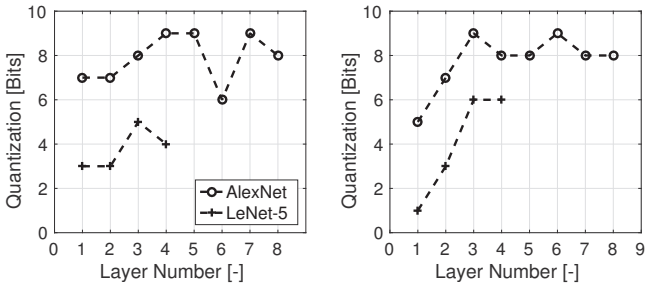


Fig. 5. Overview of a typical CNN architecture: 2 sequences of convolutional - ReLU - Maxpool layers perform hierarchical feature extraction. This feature extraction is followed by a multi-layer fully connected classification stage.



(a) Weights (b) Input Feature Maps

Fig. 6. Necessary number of bits for (a) weights and (b) input feature maps. With these quantization settings [22], the network achieves 99% relative accuracy compared to full precision 32b floating point operation.

Technology	28nm FDSOI
Area	1.87 mm²
On-chip MEM	148kB
# MAC's	256
# gates (NAND-2)	1.95M
Frequency	6-200MHz
Voltage	0.6-1.1V
Precision	Nx1-16/N
Performance	256 x N x f MACS
Power scalability @ 76GOPS nominal	300 mW (1x16b) 7.5 mW (4x4b,sparse)

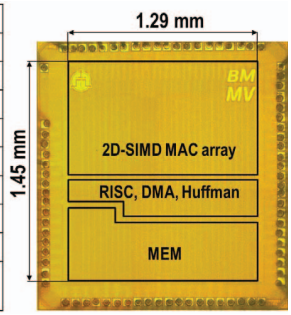


Fig. 7. (a) High level specifications of Envision. (b) Chip photograph of Envision. N is the level of subword-parallelism.

in Fig. 6, which plots the minimum required computational precision in every layer of LeNet-5 and AlexNet at 99% relative benchmark accuracy. Only 1-6b quantization suffices for LeNet-5 [26] on the MNIST data set and 5-9b quantization for AlexNet [27] on the ImageNet data set. Furthermore, [28] shows only 2-8b quantization is necessary for VGG16 [29] on ImageNet, while [11] uses 4-7b for VGG16 on the Labeled Faces in the Wild (LFW) data set.

An optimal energy-efficient Deep Learning accelerator should hence be able to dynamically tune its precision to minimize the energy consumption in function of the application, neural network and neural network layer. Below, we discuss and analyse Envision, a DVAFS-compatible CNN processor capable of this, which achieves state-of-the-art energy-efficiency.

V. Envision: a DVAFS-compatible CNN processor

Envision [11], an efficient and energy-scalable CNN chip, based on the DVAFS principle, was implemented in a 28nm FDSOI technology. This chip, illustrated in Fig. 7 is a C-programmable processor, both exploiting algorithmic sparsity [12] [22] and the dynamic accuracy scaling discussed in this work. At 200MHz, its 256 processing units achieve a peak 102GOPS in the $1 \times 1 - 16$b mode and up to 408GOPS in the $4 \times 1 - 4$b mode. The chip has 132kB on-chip data memory and 16kB on-chip program memory for its 16b instruction set. Due to the DVAFS functionality, energy-effiency varies from 0.3TOPS/W in the high precision 1×16b mode up to 4.2TOPS/W for non-sparse CONV layers and up to more than 10 TOPS/W for sparse CONV layers in the 4×4b mode when maintaining the 76GOPS nominal throughput.

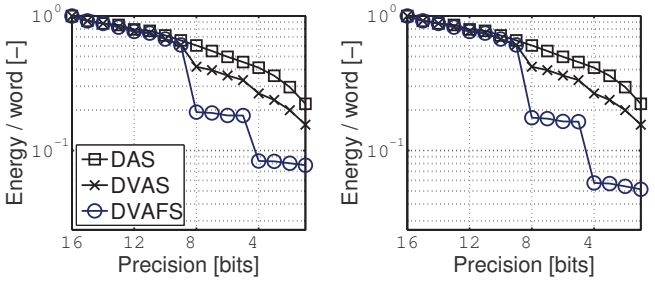
The energy-efficiency of Envision is illustrated in Fig. 8, both at constant 200MHZ frequency and at a 76GOPS constant throughput. These measurements are done on a 5×5 CONV layer with a typical MAC-efficiency of 73% or $0.73 \times 256 \times 2 \times f$ operations per second.

In DAS and DVAS, the energy gains at reduced computational accuracy are limited, as the non-scalable parts of the processor, such as the instruction decoder and memories become dominant. Fig. 8a shows the processor consumes $2.4\times$ and $3.8\times$ less energy per 4b operation than for 16b full precision, in the DAS and DVAS cases respectively.

If frequency is kept constant at 200MHz, the chip consumes 300mW at full 16b precision and 76 real GOPS, leading to an efficiency of 250GOPS/W. At the nominal frequency of 200MHz and in the $N = 4$, 4×4b mode, the chip consumes 104mW and achieves $4 \times 76 = 304$ real GOPS or 2.8TOPS/W.

Efficiency can be further improved at low computational accuracy through exploiting the frequency scaling capabilities of DVAFS. If frequency is scaled under constant computational throughput, the supply voltage of the whole system can be lowered further, leading to higher efficiency gains. This is illustrated in Fig. 8b, where in the 4×4b full DVAFS mode, frequency is scaled down from 200MHz to 50MHz to maintain the 76GOPS throughput. In this case, power consumption scales from 300mW down to 18mW or 4.2TOPS/W, or a $6.9\times$ and $4.1\times$ improvement over DAS and DVAS respectively.

As indicated in section IV, a simple handwritten digit classification task can be performed using 1-6b fixed-point precision, while more complex classification on ImageNet and LFW requires 4-9b operation. The DVAFS-based CNN processor thus allows performing simple tasks at a much higher energy-efficiency compared to more complex tasks. This is illustrated in Tab. III showing the energy-efficiency of the different convolutional layers of VGG16 (3.3fps), AlexNet (47fps) and LeNet-5 (13kfps) on Envision. The table also



(a) Constant f=200MHz (b) Constant T=76GOPS

Fig. 8. (a) Relative energy consumption of Envision at 200MHz. These numbers are normalized to the maximal power consumption of 300mW at 16b. DVAFS outperforms DVAS, as the system becomes subword-parallel at lower computational accuracy. At 4×4b, the power consumed is 108mW at 304 effective GOPS, or 2.8 TOPS/W. (b) Energy consumption per word at 76GOPS. At constant throughput, frequency and supply voltage can be lowered further at lower accuracy leading to higher efficiency gains. At 4×4b, the power consumed is 18mW at 76 effective GOPS, or 4.2 TOPS/W.

TABLE III
POWER CONSUMPTION IN VGG16, ALEXNET AND LENET-5 ON ENVISION [11] DUE TO SPARSITY [12] AND DVAFS.

Layer	Mode	f [MHz]	V [V]	Wght. [b]	In. [b]	Wght. sp. [%]	In. sp. [%]	MMACS/ frame	Power [mW]	Eff. [TOPS/W]
VGG1	2×8b	100	0.80	5	4	5	10	87	25	2.1
VGG2-13	2×8b	100	0.80	5	6	25-75	30-82	462-1850	19-35	1.5-2.8
Total	–	–	–	–	–	–	–	15346	26	2
AlexNet1	2×8b	100	0.80	7	4	21	29	104	37	2.7
AlexNet2	2×8b	100	0.80	7	7	19	89	224	20	3.8
AlexNet3	1×16b	200	1.03	8	9	11	82	150	52	1
AlexNet4-5	1×16b	200	1.03	9	8	4	72	112	58-62	0.8-0.9
Total	–	–	–	–	–	–	–	666	44	1.8
LeNet1	4×4b	50	0.65	3	1	35	87	0.3	5.6	13.6
LeNet2	2×8b	100	0.80	4	6	26	55	1.6	29	2.6
Total	–	–	–	–	–	–	–	1.9	25	3

shows the operating frequency (f), the main core voltage (V), the weight (wght.) and input (in.) precision in bits (b) and the weight and input sparsity (sp.) levels for every layer.

Envision achieves 1.8TOPS/W on AlexNet, significantly outperforming non-scalable [15] (0.16TOPS/W) and DVAS-only [12] (0.94TOPS/W) implementations.

VI. CONCLUSION

In this work, we first introduce the concept of DVAFS, a circuit level technique allowing to trade-off energy for computational accuracy, outperforming previous Approximate Computing techniques and showing a dynamic power range of $20\times$ in a multiplier and up to $8\times$ in a full SIMD-processor, when going from 16b to 4b operation.

Second, the exploitation of DVAFS is shown for Deep Learning applications, enabling their deployment on IoT. We illustrate variable precision requirements for state-of-the-art recognition algorithms, with precisions ranging from 1-9b for a number of benchmarks, in function of the application, network and even the particular layer of the network.

Finally, we show a real-life DVAFS implementation is feasible through the design and measurement of Envision: a DVAFS-enabled CNN processor implemented in 28nm FD-SOI. Power consumption in a number of CNN benchmarks ranges from $5.6 - 62$mW, mainly dependent on the computational accuracy required in the specific convolutional layer. Ultimately, the DVAFS technique allows achieving real-time processing at multiple TOPS/W.

REFERENCES

[1] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," *European Test Symposium*, 2013.

[2] V. K. Chippa *et al.*, "Analysis and characterization of inherent application resilience for approximate computing," *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2013.

[3] C. Liu *et al.*, "A low-power, high performance approximate multiplier with configurable partial error recovery," *DATE*, 2014.

[4] P. Kulkarni *et al.*, "Trading accuracy for power with an underdesigned multiplier architecture," *VLSID*, 2011.

[5] K. Y. Kyaw *et al.*, "Low-power high-speed multiplier for error-tolerant application," *Electron devices and solid-state circuits (EDSSC))*, 2011.

[6] V. Camus *et al.*, "A low-power carry cut-back approximate adder with fixed-point implementation and floating-point precision," *2016 53rd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2016.

[7] B. Moons and M. Verhelst, "Dvas: Dynamic voltage accuracy scaling for increased energy-efficiency in approximate computing," *International Symposium on Low Power Electronics and Design (ISLPED)*, 2015.

[8] M. de la Guia Solaz *et al.*, "A flexible low power dsp with a programmable truncated multiplier," *TCAS-I*, 2012.

[9] S. Venkataramani *et al.*, "Quality programmable vector processors for approximate computing," *MICRO*, 2013.

[10] Y. LeCun *et al.*, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] B. Moons *et al.*, "Envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi," *ISSCC Dig. of Tech. papers*, 2017.

[12] B. Moons and M. Verhelst, "A 0.3-2.6 tops/w precision-scalable processor for real-time large-scale convnets," *Proceedings of the IEEE Symposium on VLSI Circuits*, pp. 178–179, July 2016.

[13] K. Usami and M. Horowitz, "Clustered voltage scaling technique for low-power design," *International symposium on low power design (ISLPED)*, 1995.

[14] B. Wu and M. Willems, "Rapid architectural exploration in designing application-specific processors," in *ASIP Designer whitepaper*, 2015.

[15] Y.-H. Chen *et al.*, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *ISSCC Dig. of Tech. papers*, pp. 262–263, 2016.

[16] Z. Du *et al.*, "Shidiannao: Shifting vision processing closer to the sensor," *International Symposium on Computer Architecture (ISCA)*, pp. 92–104, 2015.

[17] L. Cavigelli *et al.*, "Origami: A convolutional network accelerator," *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*, pp. 199–204, 2015.

[18] S. Han *et al.*, "EIE: efficient inference engine on compressed deep neural network," *International symposium on Computer Architecture (ISCA)*, 2016.

[19] M. Peemen *et al.*, "Memory-centric accelerator design for convolutional neural networks," *Proceedings of the IEEE 31st International Conference on Computer Design (ICCD)*, pp. 13–19, 2013.

[20] F. N. Iandola *et al.*, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size," *CoRR*, vol. abs/1602.07360, 2016.

[21] S. Han *et al.*, "Learning both weights and connections for efficient neural network," *Proceedings of Advances in Neural Information Processing Systems*, pp. 1135–1143, 2015.

[22] B. Moons *et al.*, "Energy-efficient convnets through approximate computing," *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–8, 2016.

[23] R. Dorrance and D. Markovic, "A 190gflops/w dsp for energy-efficient sparse-blas in embedded iot," *2016 IEEE Symposium on VLSI Circuits (VLSIC)*, pp. 1–2, June 2016.

[24] M. Courbariaux and Y. Bengio, "Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1," *CoRR*, vol. abs/1602.02830, 2016.

[25] P. Gysel *et al.*, "Hardware-oriented approximation of convolutional neural networks," *Workshop contribution to International Conference on Learning Representations (ICLR)*, 2016.

[26] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] A. Krizhevsky *et al.*, "Imagenet classification with deep convolutional neural networks," *Proceedings of Advances in neural information processing systems*, pp. 1097–1105, 2012.

[28] F. Sun *et al.*, "Intra-layer nonuniform quantization for deep convolutional neural network," *arXiv preprint arXiv:1607.02720*, 2016.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.