Dynamic Bitstream Length Scaling Energy Effective Stochastic LDPC Decoding

Thomas Marconi and Sorin Cotofana
Computer Engineering Lab, TU Delft, The Netherlands
{T.Marconi, S.D.Cotofana}@tudelft.nl

ABSTRACT

Stochastic Computing (SC) is an attractive solution for implementing Low Density Parity Codes (LDPC) decoders due to its fault tolerance capability and low hardware requirements. However, in practical implementations, SC efficiency is limited by the Stochastic Bitstream (SB) length and by the computation inaccuracies due to non-unique SB representations. In this paper, rather than statically fixing the SB length at run-time, we propose a Dynamic Bitstream Length Scaling (DBLS) technique, which adjusts on-the-fly the SB length such that Quality of Service requirements for energy efficient LDPC decoding are fulfilled. In this way, depending on the communication channel condition, different SB lengths are adaptively utilized such that the best decoding performance vs energy consumption tradeoff is achieved. To evaluate the DBLS practical implications we selected an (1296,648) LDPC with $d_v = 3$ and $d_c = 6$ and implemented our approach and the best state-of-the-art stochastic LDPC decoder with 64-bit edge memory on a Virtex-7 FPGA. Experimental results indicate that our proposal requires 9% more FFs and 3% more LUTs while diminishing the energy consumption by 31-80% and providing 1.5-5.1x higher throughput.

1. INTRODUCTION

Low Density Parity Codes (LDPC) codes, introduced by Gallager in 1962 [6], have a decoding performance which is very close to the Shannon limit [7]. Due to their high error correction capability LDPC codes have been adopted by many communication standards, e.g., WiFi [2], WiMAX [3], DVB-S2 [1], 10GBase-T [4]. Although LDPC decoding is very powerful in terms of error correction it has high computation demands, which have a negative impact on the energy consumption. Given that in communication hand-held devices, a typical LDPC utilization case, to prolong the battery life, low energy LDPC decoding has received substantial attention in the recent past. One of the effective ways to construct LDPC decoders is by using Stochastic Computing (SC). Due to the SC efficient hardware implementation, SC based LDPC decoders power consumption is substantially smaller than the one relying on the traditional computation paradigm. However, given the fact that operands are represented as long Stochastic Bitstreams (SB) SC-based implementations inherently exhibit large latency. Consequently, the low power gain could be potentially nullified by the long computation time precluding the SC utilization for low energy LDPC decoder implementations.

In all existing SC-based machines, e.g., LDPC decoders, the SB length is fixed during decoding regardless of the communication channel Signal to Noise Ratio (SNR). To reduce energy consumption while fulfilling Quality of Service (QoS) requirements, in this paper, we propose a Dynamic Bitstream Length Scaling (DBLS), which adapts on-the-fly the SB length according to the channel conditions. Note that in existing SC-based LDPC decoders, the SB length is fixed and decided at design time. In particular, we utilize the communication channel conditions quantified by the current Signal-to-Noise Ratio (SNR) to dynamically adjust the SB length on-the-fly to the minimum value for which the QoS requirements are still satisfied. Although our focus in this work is on applying DBLS to synchronous designs, in principle, our approach is also suitable for asynchronous engines. To evaluate the practical implications of our proposal, we choose a fully parallel implementation of a 64-bit Edge Memory (EM) [12] stochastic (1296,648) LDPC decoder with $d_v = 3$ and $d_c = 6$ as an evaluation vehicle. Our extensive experiments on a Virtex-7 FPGA indicate that at the expense of an area overhead of 9% more FFs and 3% more LUTs, the DBLS technique improves the energy efficiency and throughput by 31-80% and 1.5-5.1x, respectively, while fulfilling the QoS requirements.

The rest of the paper is organized as follows. In Section 2, we discuss the proposed technique while its effectiveness is presented in Section 3. Finally, we close the discussion with some conclusions in Section 4.

2. PROPOSED TECHNIQUE

The Stochastic Computing paradigm proposed by Gaines [5], Ribeiro [11], and Poppelbaum et al. [10] in 1967, represents a number x as a Stochastic Bitstream (SB) $X = X_1X_2X_3...X_{L_s}$ such that $P(X_i = 1) = x$ where $0 \leq x \leq 1$ and L_s is the SB length. Thus an SB of L_s bits with N_1 comprising bits being equal to 1, encodes a number $x = \frac{N_1}{L_s}$. This number representation has two main advantages: (i) arithmetic computations can be done with low complexity hardware and (ii) has intrinsic high fault tolerant capabil-

GLSVLSI'15, May 20–22, 2015, Pittsburgh, PA, USA.

http://dx.doi.org/10.1145/2742060.2742117.

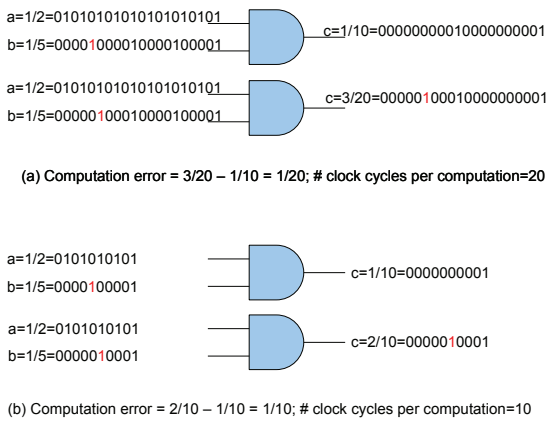

Figure 1: Tradeoff of Computation Error and Latency in Stochastic Computing (SC)

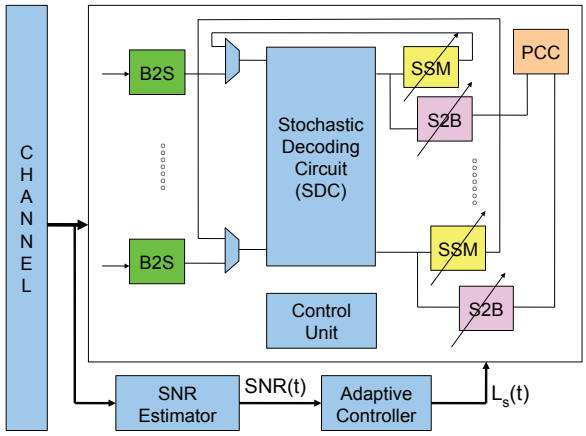

Figure 2: LDPC Decoding with DBLS Technique

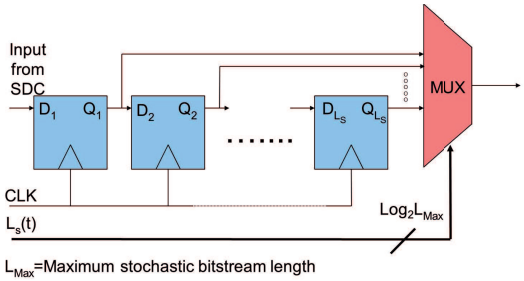

Figure 3: SSM for supporting DBLS technique

ity. This representation allows for the evaluation of complex arithmetic operations by using simple logic gates. For instance, we can use an AND gate to implement a multiplication as illustrated in Figure 1. Moreover, by relying on an identically-weight representation, SC has a high fault tolerant capability, e.g., flipping one bit at position n of an SB with length L_s causes only an $\frac{1}{L_s}$ difference irrespective of the n value while the same bit flip in traditional binary representation creates a 2^{n-1} difference. The main SC drawbacks are: (i) long computation latency due to the required long SB representations and (ii) inaccurate computation results caused by the non-unique number representation. For example, illustrated in Figure 1, a number $a = \frac{1}{2}$ can be represented as a 20-bit SB A=01010101010101010101 or a 10-bit SB A=0101010101 or any other SB that has an equal number of bit 1 and 0. This non-unique representation creates the premises for producing incorrect results. In Figure 1(a), when we change B_5B_6 from 10 to 01, the 2nd AND gate produces an incorrect result; $\frac{1}{2}$ times $\frac{1}{5}$ should be equal to $\frac{1}{10}$ as produced by the 1st AND gate, however, the 2nd AND gate's result is $\frac{3}{20}$. It can be easily observed that the computation error of Fig 1(a) is $\frac{3}{10} - \frac{1}{10} = \frac{1}{20}$ and the computation latency is 20 clock cycles. If we shorten the SB length L_s from 20 to 10 to shorten the computation time from 20 to 10 clock cycles as shown in Fig 1(b), the computation error will increase to $\frac{2}{10} - \frac{1}{10} = \frac{1}{10}$. This simple motivational example clearly demonstrates that various computation error vs latency trade-offs are possible in the SC context. Using longer (shorter) SBs produce more (less) accurate results with longer (shorter) latencies. Since the energy consumption proportionally depends on latency, trading accuracy for energy is also possible.

Trading accuracy at the right time is the key idea behind our proposed technique to diminish energy consumption while fulfilling Quality of Service (QoS) requirements. Trading accuracy in the moment accuracy is needed to fulfill QoS requirements is not a good option. To guaranty QoS while diminishing energy consumption, we need to trade accuracy only in the time moments when QoS can be provided without requiring high computation accuracy. To be able to properly adjust at run time the SB length to the channel status we perform at design time a decoder pre-characterization, i.e., we measure decoder's Frame Error Rate (FER), Bit Error Rate (BER), energy/bit and throughput under SB length scaling on a variety of noisy channels.

The way we apply the DBLS technique during the LDPC decoding process is presented in Figure 2. We utilize the SNR Estimator present in any standard communication systems, to provide SNR information to the Adaptive Controller. This information is essentially needed to assist the Adaptive Controller in adjusting the SB length $L_s(t)$ based on the instantaneous channel condition represented by SNR(t). Based on SNR(t) and the pre-characterization results, the Adaptive Controller tunes $L_s(t)$ to the minimum value for which the QoS requirements (e.g., BER, FER) are fulfilled. We propose 3 strategies in pre-determining the SB length at design time. Choosing $L_s(t)$ at SNR(t) is done by selecting SB length from pre-characterization results that minimizes (i) BER denoted as Best BER (BB), (ii) FER called as Best FER (BF), or (iii) energy/bit named as Lowest Energy (LE) strategy.

In the following we describe the modus operandi of the proposed adaptive stochastic LDPC decoder. For each received message we first determine the BS length value $L_s(t)$ according to the actual SNR value. This adaptation is done on-the-fly, no decoding stop nor restart is required. Subsequently, in the first $L_s(t)$ clock cycles, soft messages (i.e., probabilities) are converted by the Binary to Stochastic Converter (B2S) blocks to SBs with length $L_s(t)$, bit by bit in each clock cycle. In this first iteration, the multiplexers direct the SBs to the Stochastic Decoder Circuit (SDC), i.e.,

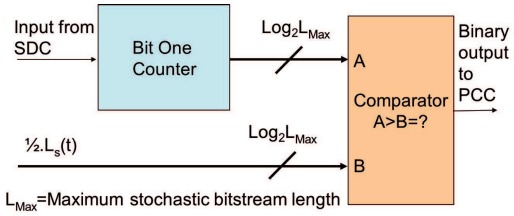

Figure 4: S2B for supporting DBLS technique

Variables Nodes (VNs) and Check Nodes (CNs), for processing. The $L_s(t)$-bit SB produced at each SDC output is stored in the Stochastic Stream Memory (SSM). This SSM should have adjustable size to accommodate our DBLS technique. The architecture of this adjustable SSM is presented in Figure 3. Different from fixed-length SSM, this tunable SSM makes use of a multiplexor to select $Q_{L_s(t)}$ for output. In this way an SB length of up to L_{Max} can be accommodated. In the subsequent $L_s(t)$ clock cycles, the multiplexer inside SSM in collaboration with the multiplexer in Figure 2 feeds back $L_s(t)$-bit stored in SSM as inputs to the SDC for subsequent iterations. At the end of each iteration, the Parity Check Circuit (PCC) verifies the parity constraints for early termination purpose. To do this verification, the PCC makes use of the hard bit values provided by the Stochastic to Binary Converter (S2B). Since DBLS technique needs to support adjustable SB length, the S2B needs to facilitate adjustable SB length hard decisions as presented in Figure 4. By letting the comparator making hard decisions controllable by the Control Unit as indicated in the figure, the variable SB length is supported by S2B. The SB iterative processing stops either when all check nodes are satisfied or when the maximum number of iterations is surpassed.

3. EVALUATION

Although our technique is in principle applicable to any SC machines for which the SB length vs performance relation can be pre-characterized, in this paper, we evaluate its effectiveness by applying it to a fully parallel implementation of the 64-bit EM [12] stochastic LDPC decoder reported as the best SC-based decoder in terms of decoding performance in, e.g., [9]. More precisely, a circuit decoding regular (1296, 648) LDPC code with VN degree of 3 ($d_v = 3$), and PN degree of 6 ($d_c = 6$) is utilized. Input messages in form of probabilities are 4-bit quantized. Each probability is represented by a 16-bit SB. The maximum number of iterations is set at 100. Each iteration takes 16 decoding cycles and for each decoding cycle one clock cycle is needed. The EM decoders with DBLS technique using proposed strategies, denoted as BB, BF, and LE, are evaluated against their original version, named as EM, in terms of: (i) decoding performance, specifically FER and BER, (ii) area - the number of occupied FPGA hardware resources (i.e., FFs and LUTs), (iii) throughput (Mb/s), and (iv) energy/bit (pJ/bit). An Additive White Gaussian Noise (AWGN) channel with Binary Phase Shift Keying (BPSK) modulation (with mapping $0 \rightarrow 1$ and $1 \rightarrow -1$) is generated to mimic a realistic telecommunication channel in our experiments. We apply the Noise-Dependent Scaling (NDS) technique [12] with $\alpha = 3$ and Y=6 to the AWGN channel

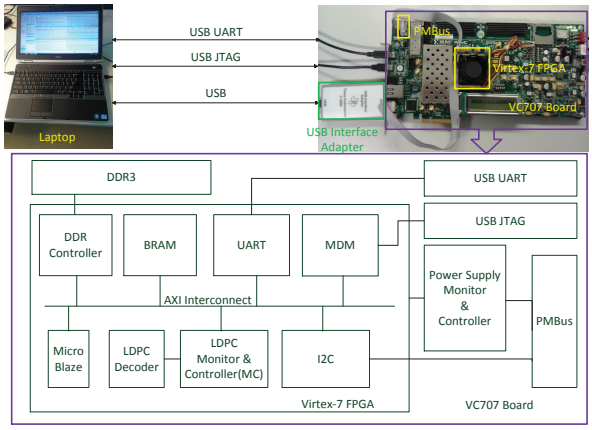

Figure 5: Experimental Hardware Platform

Table 1: Area and Power Consumption

Decoders	Area		Power
	# FFs	# LUTs	(W)
EM	189896	146300	1.85
EM with DBLS	206730	150189	1.855

prior to decoding. Similar to [8], all evaluation figures are obtained from direct FPGA measurements.

To facilitate a real hardware evaluation, we develop an experimental hardware platform that consists of a laptop and a Xilinx VC707 board as depicted in Figure 5. The laptop is dedicated to: (i) designing the hardware platform targeting Xilinx Virtex-7 FPGA: XC7VX485TFFG1761-2 inside the Xilinx board and generating the bitstream files, (ii) downloading the bitstream files for FPGA hardware and the software files for the MicroBlaze through the USB JTAG interface, (iii) monitoring/capturing the number of iterations and decoding outcomes, FER, and BER through the USB UART, and (iv) measuring Energy/bit using the Fusion Digital Power Designer from Texas Instrument through Texas Instrument USB Interface adapter by reading PMBus, and accessing Power Supply Monitor and Controller inside the board. To build the evaluation hardware support (e.g., AWGN channel emulator, BPSK modulator, and other functionality for evaluation purpose) and the decoders, the Xilinx board is utilized. The MicroBlaze processor, resident on the FPGA, in collaboration with the LDPC Monitoring and Controller (MC) mainly acts as the evaluation hardware support by accessing software and data stored in BRAM and DDR3. The MC monitors the decoding outcome and conveys 4-bit quantized probabilities from the MicroBlaze to the evaluated decoder. The two possible outcomes of the decoding process are: (i) "success" and (ii) "give up". If all check nodes are satisfied, the outcome is "success". In the case that all check nodes cannot be satisfied after the maximum number of iterations, the system reports "give up". The MicroBlaze processor utilizes these outcomes for computing the statistical results of the experiments (i.e. BER, FER). The laptop displays the experimental results received from FPGA board through USB UART.

The experimental results in terms of FER, BER, energy efficiency, and throughput are presented in Figures 6 and

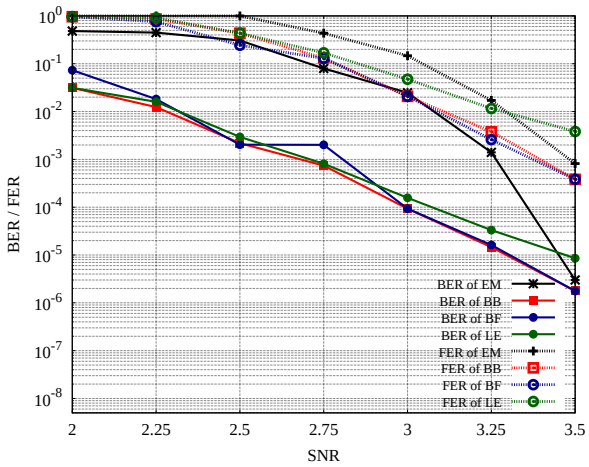

Figure 6: BER and FER

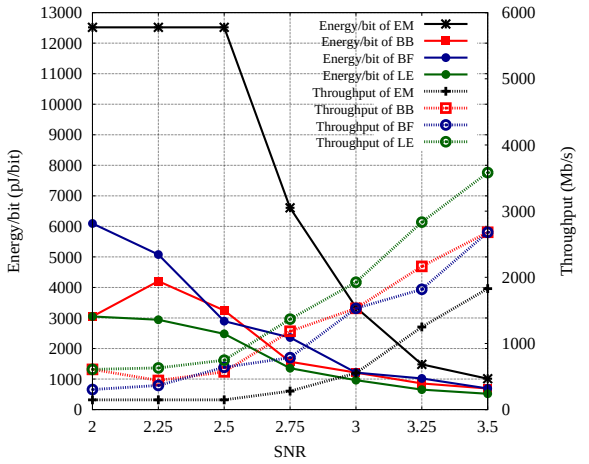

Figure 7: Energy/bit and Throughput

7, respectively. Tables 1 summarizes the amount of FPGA hardware resources required by the decoders and the corresponding power consumption. From these experimental results, one can observes that: (i) The area overhead associated with the DBLS technique in terms of the number of occupied FFs and LUTs is 9% and 3%, respectively. This area overhead is mainly attributed to the additional hardware to support adjustable SB length. The overhead does not affect the maximum clock frequency, which is 182.449 MHz, but it slightly increases power consumption by 0.3%. (ii) Since the computation latency is lowered by shortening SB length, the throughput is increased. Because the energy consumption is proportional to the computation latency, despite the power consumption increase due to (i), the energy/bit is minimized as depicted in Figure 7. (iii) Although the BB strategy is mainly designed to minimize BER, it also reduces the energy/bit by 32-76% when compared to EM due to the higher throughput (1.7-4.2x) achieved by SB length reduction. (iv) The BF strategy is minimizing FER while lowering the energy by 31-77% and increasing the throughput by 1.5-4.3x thanks to its SB length adaptability to SNR. (v) The LE strategy provides the largest energy reduction (49-80%) and boosts throughput (2.3-5.1x) for a slightly higher FER and BER at SNR=3.5dB. The BER is 8.54×10^{-6}, which is higher than EM's BER, 3.0×10^{-6}. However, if the BER target is higher than 8.54×10^{-6}, the LE scheme still can fulfill the targeted QoS.

4. CONCLUSIONS

In this paper, we proposed a Dynamic Bitstream Length Scaling (DBLS) technique to automatically adjust Stochastic Bitstream length to achieve energy efficient stochastic LDPC decoding. Subject to specified decoding performance constraints and adaptive to communication channel conditions, the technique leverages available performance-energy tradeoffs to deliver satisfying performance while minimizing the energy. When implemented on a Virtex-7 FPGA, despite its area overhead of 9% more FFs and 3% more LUTs, the technique outperforms the best equivalent state of the art counterpart in terms of energy efficiency by 31-80% and throughput by 1.5-5.1x while fulfilling the QoS requirements.

Acknowledgment

This work was supported by the Seventh Framework Programme of the European Union, under the Grant Agreement number 309129 (i-RISC project).

5. REFERENCES

[1] The digital video broadcasting standard.
[2] The ieee 802.11n working group std.
[3] The ieee 802.16 working group std.
[4] Ieee p802.3an (10gbase-t) task force.
[5] B. R. Gaines. Stochastic computing. In *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*, AFIPS '67 (Spring), pages 149–156, New York, NY, USA, 1967. ACM.
[6] R. Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, January 1962.
[7] D. J. MacKay and R. M. Neal. Near shannon limit performance of low density parity check codes. *Electronics letters*, 32(18):1645–1646, 1996.
[8] T. Marconi, C. Spagnol, E. Popovici, and S. Cotofana. Towards energy effective ldpc decoding by exploiting channel noise variability. In *Very Large Scale Integration (VLSI-SoC), 2014 22nd International Conference on*, pages 1–6, Oct 2014.
[9] A. Naderi, S. Mannor, M. Sawan, and W. Gross. Delayed stochastic decoding of ldpc codes. *Signal Processing, IEEE Transactions on*, 59(11):5617–5626, Nov 2011.
[10] W. J. Poppelbaum, C. Afuso, and J. W. Esch. Stochastic computing elements and systems. In *Proceedings of the November 14-16, 1967, Fall Joint Computer Conference*, AFIPS '67 (Fall), pages 635–644, New York, NY, USA, 1967. ACM.
[11] S. T. Ribeiro. Random-pulse machines. *Electronic Computers, IEEE Transactions on*, EC-16(3):261–276, June 1967.
[12] S. Tehrani, S. Mannor, and W. Gross. Fully parallel stochastic ldpc decoders. *Signal Processing, IEEE Transactions on*, 56(11):5692–5703, Nov 2008.