www.ijiarec.com

ISSN:2348-2079

Volume-7 Issue-4

International Journal of Intellectual Advancements and Research in Engineering Computations

Optimization of area and power in feed forward cut set free mac unit using ripple carry adder and EXOR full adder

V.MohanaPriya[1], S.Purushothaman[2], S.Tamilarasi[3], S.Vanitha[4]

Department of ECE, PGP College of Engineering and Technology, Namakkal.

ABSTRACT

MAC (Multiply Accumulate Unit) computation plays a important role in (DSP) Digital Signal Processing. The MAC is common step that computes the product of two numbers and add that product to an accumulator. Generally, the Pipelined architecture is used to improve the performance by reducing the length of the critical path. But, more number of flip flops are used when using the pipeline architecture that reduces the efficiency of MAC and increases the power consumption. On the basis of machine learning algorithm, this paper proposes a feed forward-cutset-free (FCF) pipelined MAC architecture that is specialized for a high-performance machine learning accelerator, and also proposes the new design concept of MFCF_PA using the concept of column addition stage with the 4:2 compressor. Therefore, the proposed design method reduces the area and the power consumption by decreasing the number of inserted flip-flops for the pipelining when compared to the existing pipelined architecture for MAC computation. Finally, the proposed feed forward cutset free pipelined architecture for MAC is implemented in the VHDL and synthesized in XILINX and compared in terms of area, power and delay reports.

Keywords: Hardware accelerator, Machine Learning, Multiply–Accumulate (MAC) unit, Pipelining

INTRODUCTION

In a machine learning accelerator, a large number of multiply–accumulate (MAC) units are included for parallel computations, and timing-critical paths of the system are often found in the unit. A multiplier typically consists of several computational parts including a partial product generation, a column addition, and a final addition. An accumulator consists of the carry-propagation adder. Long critical paths through these stages lead to the performance degradation of the overall system. To minimize this problem, various methods have been studied. The Wallace [8] and Dadda [9] multipliers are well-known examples for the achievement of a fast column addition, and the carry-look ahead (CLA) adder is often used to reduce the critical path in the accumulator or the final addition stage of the multiplier. Meanwhile, a MAC operation is performed in the machine learning algorithm to compute a partial sum that is the accumulation of the input multiplied by the weight. In a MAC unit, the multiply and accumulate operations are usually merged to reduce the number of carry-propagation steps from two to one [10]. Such a structure, however, still comprises a long critical path delay that is approximately equal to the critical path delay of a multiplier. It is well known that pipelining is one of the most popular approaches for increasing the operation clock frequency. Although pipelining is an efficient way to reduce the critical path delays, it results in an increase in the area and the power consumption due to the insertion of many flip-flops. In particular, the number of flip-flops tends to be large because the flip-flops must be inserted in the feed forward-cutset to ensure functional equality before and after the pipelining. The problem worsens as the number of pipeline stages

Author for correspondence:
Department of ECE, PGP College of Engineering and Technology, Namakkal.

is increased. The main idea of this paper is the ability to relax the feedforward-cutset rule in the MAC design for machine learning applications, because only the final value is used out of the large number of multiply–accumulations. In other words, different from the usage of the conventional MAC unit, intermediate accumulation values are not used here, and hence, they do not need to be correct as long as the final value is correct. Under such a condition, the final value can become correct if each binary input of the adders inside the MAC participates in the calculation once and only once, irrespective of the cycle. Therefore, it is not necessary to set an accurate pipeline boundary. Based on the previously explained idea, this paper proposes a feed forward-cutset-free (FCF) pipelined MAC architecture that is specialized for a high-performance machine learning accelerator.

LITERATURE SURVEY

- Krizhevsky, I. Sutskever, and G. E. Hinton, "Image Net classification with deep convolution neural networks," in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 1097–1105-We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce over fitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.
- K. Simonyan and A. Zisserman. (2014). "Very deep convolution networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556 -In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3x3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16-19 weight layers. These findings were the basis of our Image Net Challenge 2014 submission, where our team secured the first and the second places in the localization and classification tracks respectively. We also show that our representations generalize well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing Conv Net models publicly available to facilitate further research on the use of deep visual representations in computer vision.
- K. Simonyan and A. Zisserman. (2014). "Very deep convolution networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556 -We propose a deep convolutional neural network architecture codenamed "Inception", which was responsible for setting the new state of the art for classification and detection in the Image Net Large-Scale Visual Recognition Challenge 2014 (ILSVRC 2014). The main hallmark of this architecture is the improved utilization of the computing resources inside the network. This was achieved by a carefully crafted design that allows for increasing the depth and width of the network while keeping the computational budget constant. To optimize quality, the architectural decisions were based on the Hebbian principle and the intuition of multi-scale processing. One particular incarnation used in our submission for ILSVRC 2014 is called Google Net, a 22 layers deep network, the quality of which is assessed in the context of classification and detection.

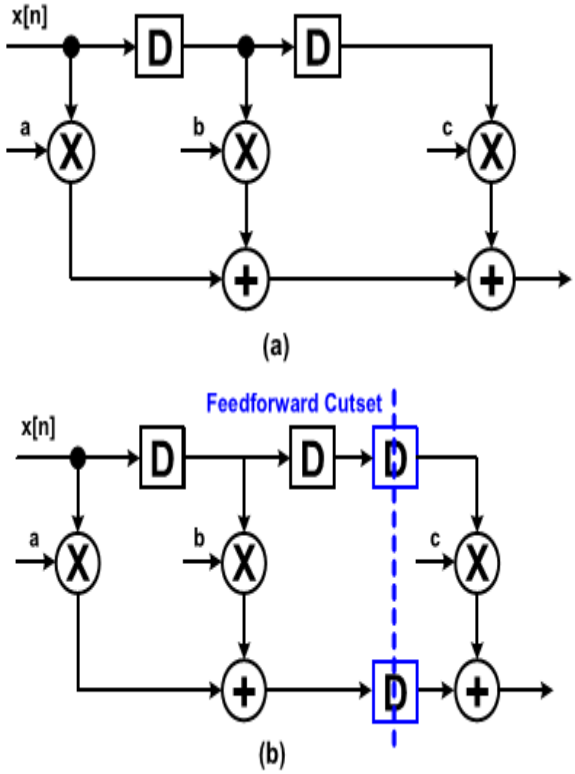
EXISTING SYSTEM

RECENTLY, the deep neural network (DNN) emerged as a powerful tool for various applications including image classification and speech recognition. Since an enormous amount of vector-matrix multiplication computations are required in a typical DNN application, a variety of dedicated hardware for machine learning have been proposed to accelerate the computations. In a machine learning accelerator, a large number of multiply–accumulate (MAC) units are included for parallel computations, and timing-critical paths of the system are often found in the unit.

A multiplier typically consists of several computational parts including a partial product generation, a column addition, and a final addition. An accumulator consists of the carry-propagation adder. Long critical paths through these stages lead to the performance degradation of the overall system. To minimize this problem, various methods have been studied. The Wallace and Dadda multipliers are well-known examples for the achievement of a fast column addition, and the carry-look ahead (CLA) adder is often used to reduce the critical path in the accumulator or the final addition stage of the multiplier.

The main idea of this paper is the ability to relax the feed forward-cutset rule in the MAC design for machine learning applications, because only the final value is used out of the large number of multiply–accumulations. In other words, different from the usage of the conventional MAC unit, intermediate accumulation values are not used here, and hence, they do not need to be correct as long as the final value is correct. Under such a condition, the final value can become correct if each binary input of the adders inside the MAC participates in the calculation once and only once, irrespective of the cycle. Therefore, it is not necessary to set an accurate pipeline boundary.

Based on the previously explained idea, this paper proposes a feed forward-cutset-free (FCF) pipelined MAC architecture that is specialized for a high-performance machine learning accelerator. The proposed design method reduces the area and the power consumption by decreasing the number of inserted flip-flops for the pipelining.



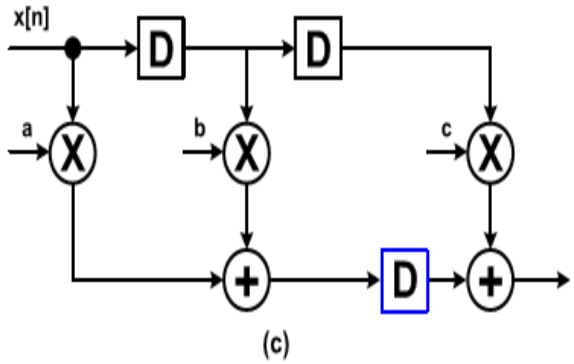


Figure 1: (a) Block diagrams of the three-tap FIR filter. (b) Valid pipelining. (c) Invalid pipelining. "D" indicates a flip-flop.

Preliminary: Feed forward-Cutset Rule for Pipelining

It is well known that pipelining is one of the most effective ways to reduce the critical path delay, thereby increasing the clock frequency. This reduction is achieved through the insertion of flip-flops into the data path.

Fig. 1(a) shows the block diagram of a three-tap finite impulse response (FIR) filter

$y[n] = ax[n] + bx[n - 1] + cx[n - 2]$. .....(1)

Fig. 1(b) and (c) shows pipelining examples regarding the FIR filter. In addition to reducing critical path delays through pipelining, it is also important to satisfy functional equality before and after pipelining. The point at which the flip-flops are inserted to ensure functional equality is called the feed forward-cutset. The definitions of cutset and feed forward-cutset are as follows:

Cutset

A set of the edges of a graph such that if these edges are removed from the graph, and the graph becomes disjointed.

Feed forward-cutset: A cutset where the data move in the forward direction on all of the cutset edges.

Fig. 1(b) shows an example of valid pipelining. The two-stage pipelined FIR filter is constructed by inserting two flip-flops along feed forward-cutset. In contrast, Fig. 1(c) shows an example of invalid pipelining. Functional equality is not guaranteed in this case because the flip-flops are not inserted correctly along the feed forward-cutset.

Disadvantages

- Number of inserted flip flops increases the pipeline stages.
- Consumes larger area and high critical path delay.
- Power consumption is high.

PROPOSED SYSTEM

MAC (Multiply Accumulate Unit) computation plays a important role in (DSP) Digital Signal Processing. The MAC is common step that computes the product of two numbers and add that product to an accumulator. Generally, the Pipelined architecture is used to improve the performance by reducing the length of the critical path. But, more number of flip flops are used when using the pipeline architecture that reduces the efficiency of MAC and increases the power consumption. On the basis of machine learning algorithm, this paper proposes a feed forward-cutset-free (FCF) pipelined MAC architecture that is specialized for a high-performance machine learning accelerator. The proposed design method reduces the area and the power consumption by decreasing the number of inserted flip-flops for the pipelining when compared to the existing pipelined architecture for MAC computation. Finally, the proposed feed forward cutset free pipelined architecture for MAC is implemented in the VHDL and synthesized in XILINX and compared in terms of area, power and delay reports.

Pipelined Accumulator

While the conventional pipelining method is advantageous because it effectively reduces the critical path delays, it leads mostly to an increase in the area and the power consumption due to the insertion of a large number of flip-flops. Moreover, the constraint that requires the insertion of the flip-flops according to the feed forward-cutset rule tends to significantly increase the overhead. This section proposes a method for the selective elimination of the flip-flops from the conventional pipeline boundary by exploiting the unique characteristics of the machine learning algorithm.

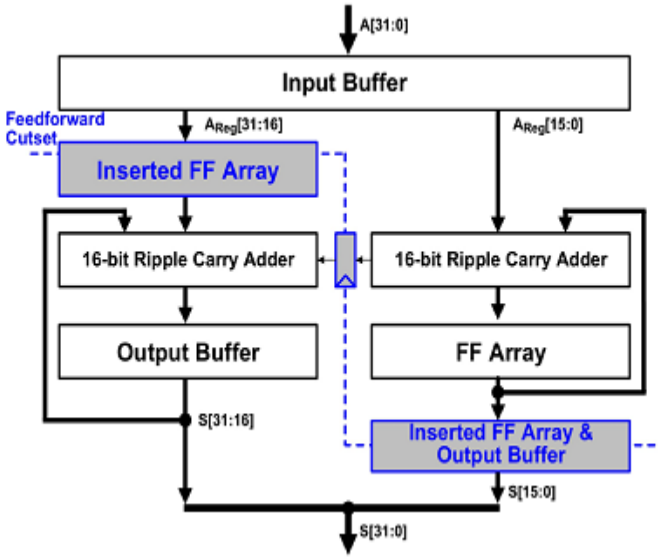
Proposed FCF Pipelining

Fig. 2 shows examples of the two-stage 32-bit pipelined accumulator (PA) that is based on the ripple carry adder (RCA). A[31 : 0] represents data that move from the outside to the input buffer register. A Reg[31 : 0] represents the data that are stored in the input buffer. S[31 : 0] represents the data that are stored in the output buffer register as a result of the accumulation. In the conventional PA structure [Fig. 2(a)], the flip-flops must be inserted along the feed forward-cutset to ensure functional equality. Since the accumulator in Fig. 2(a) comprises two pipeline stages, the number of additional flip-flops for the pipelining is 33 (gray-colored flip-flops). If the accumulator is pipelined to the n-stage, the number of inserted flip-flops becomes 33(n−1), which confirms that the number of flip-flops for the pipelining increases significantly as the number of pipeline stages is increased.

Fig. 2(b) shows the proposed FCF-PA. For the FCF-PA, only one flip-flop is inserted for the two-stage pipelining. Therefore, the number of additional flip-flops for the n-stage pipeline is n − 1 only.

In the conventional PA, the correct accumulation values of all the inputs up to the corresponding clock cycle are produced in each clock cycle as shown in the timing diagram of Fig. 2(a). A two-cycle difference exists between the input and the corresponding output due to the two-stage pipeline. On the other hand, in the proposed architecture, only the final accumulation result is valid as shown in the timing diagram of Fig. 2(b).

Fig. 3 shows examples of the ways that the conventional PA and the proposed method (FCF-PA) work. In the conventional two-stage PA, the accumulation output (S) is produced two clock-cycle after the corresponding input is stored in the



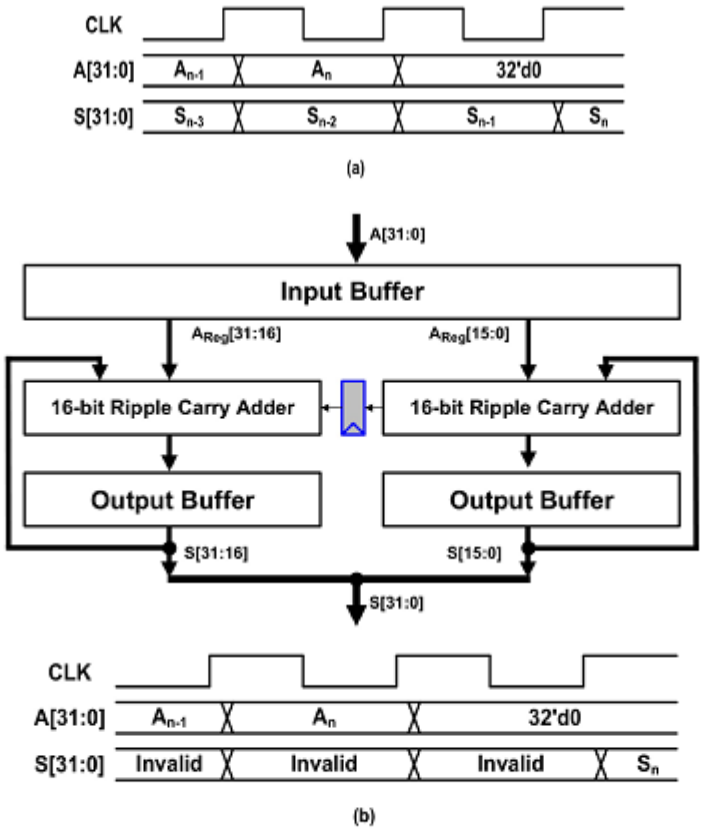


Figure 2: Schematics and timing diagrams of two-stage 32-bit accumulators. (a) Conventional PA. (b) Proposed FCF-PA.

Input buffer. On the other hand, regarding the proposed structure, the output is generated one clock cycle after the input arrives. Moreover, for the proposed scheme, the generated carry from the lower half of the 32-bit adder is involved in the accumulation one clock cycle later than the case of the conventional pipelining. For example, in the conventional case, the generated carry from the lower half and the corresponding inputs are fed into the upper half adder in the same clock cycle as shown in the cycles 4 and 5 of Fig. 3 (left). On the other hand, in the proposed FCF-PA, the carry from the lower half is fed into the upper half one cycle later than the corresponding input for the upper half, as depicted in the clock cycles 3-5 of Fig. 3 (right). This characteristic makes the intermediate result that is stored in the output buffer of the proposed accumulator different from the result of the conventional pipelining case

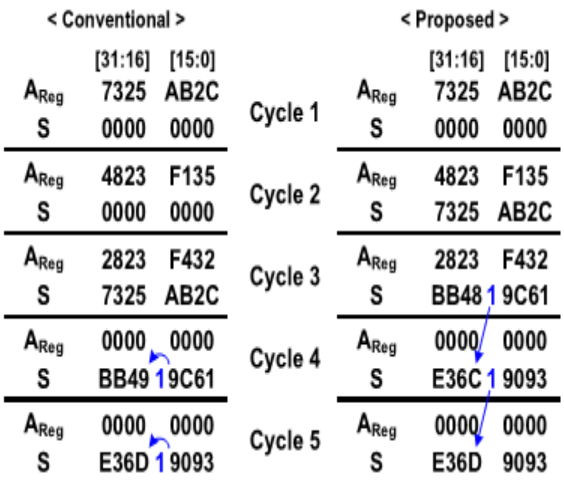


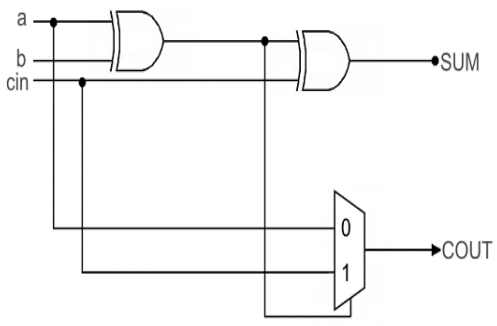
Figure 3: Two-stage 32-bit pipelined-accumulation examples with the conventional pipelining (left) and proposed FCF-PA (right). Binary number "1" between the two 16-bit hexadecimal numbers is a carry from the lower half

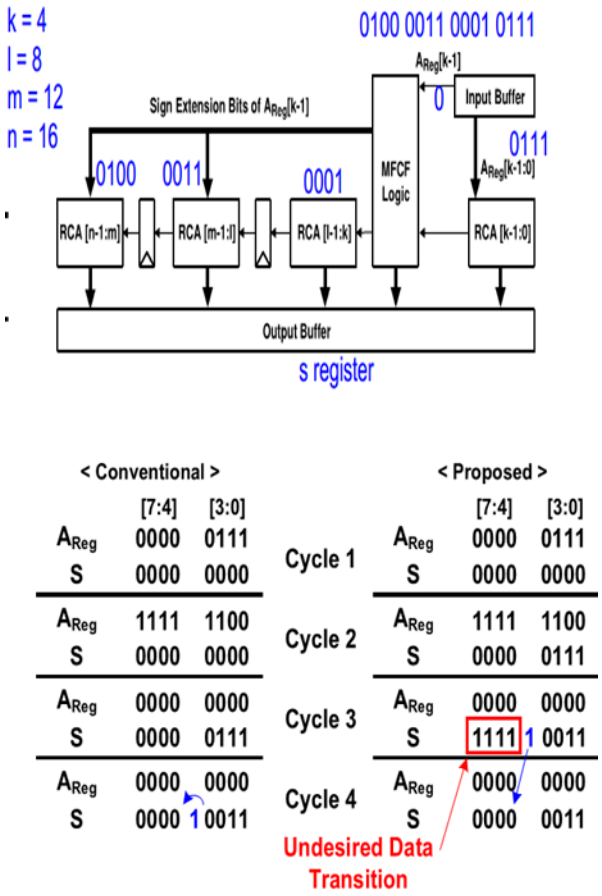
The proposed accumulator, however, shows the same final output (cycle 5) as that of the conventional one. In addition, regarding the two architectures, the number of cycles from the initial input to the final output is the same. The characteristic of the proposed FCF pipelining method can be summarized as follows: In the case where adders are used to process data in an accumulator, the final accumulation result is the same even if binary inputs are fed to the adders in an arbitrary clock cycle as far as they are fed once and only once.

Meanwhile, the CLA adder has been mostly used to reduce the critical path delay of the accumulator. The carry prediction logic in the CLA, however, causes a significant increase in the area and the power consumption. For the same critical path delay, the FCF-PA can be implemented with less area and lower power consumption compared with the accumulator that is based on the CLA.

Full adder designs using XNOR and XOR gates for sum logic

A full adder design employing two stages of XNOR gates for the sum logic, while that employing two successive stages of XOR gates for the sum logic is depicted.

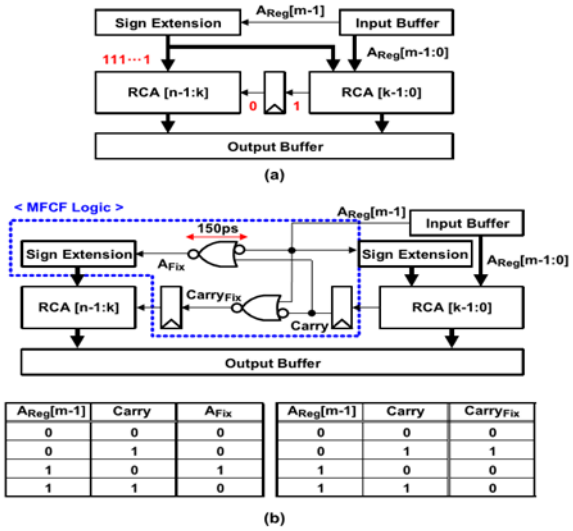




Full adder using XOR gates and a MUX

Modified FCF-PA for Further Power Reductions

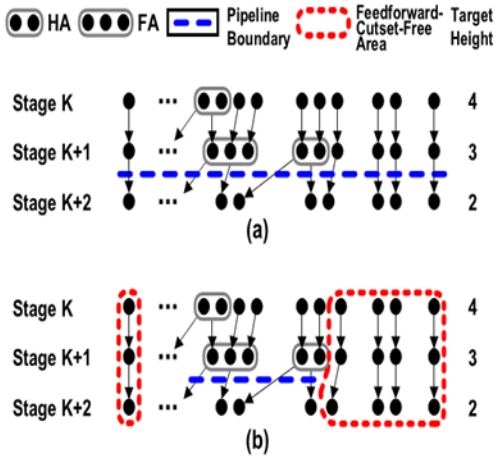
Although the proposed FCF-PA can reduce the area and the power consumption by replacing the CLA, there are certain input conditions in which the undesired data transition in the output buffer occurs, thereby reducing the power efficiency when 2's complement numbers are used. Fig. 4 shows an example of the undesired data transition. The inputs are 4-bit 2's complement binary numbers. AReg [7 : 4] is the sign extension of AReg [3], which is the sign bit of ARe g [3 : 0]. In the conventional pipelining [Fig. 4 (left)], the accumulation result (S) in cycle 3 and the data stored in the input buffer (AReg) in cycle 2 are added and stored in the output buffer (S) in cycle 4. In this case, the "1" in AReg [2] in cycle 2 and the "1" in S[2] in cycle 3 are added, thereby generating a carry. The carry is transmitted to the upper half of the S, and hence, S[7:4] remains as "0000" in cycle .Block diagram of MFCF-PA Block diagram of MFCF-PA



AReg[m-1]	Carry	AFix
0	0	0
0	1	0
1	0	1
1	1	0

AReg[m-1]	Carry	CarryFix
0	0	0
0	1	1
1	0	0
1	1	0

Proposed (a) FCF-PA and (b) MFCF-PA for the improvement of the power efficiency.



Pipelined column addition structure with the Dadda multiplier.
(a) Conventional pipelining. (b) Proposed FCF pipelining. HA: half-adder. FA: full adder.

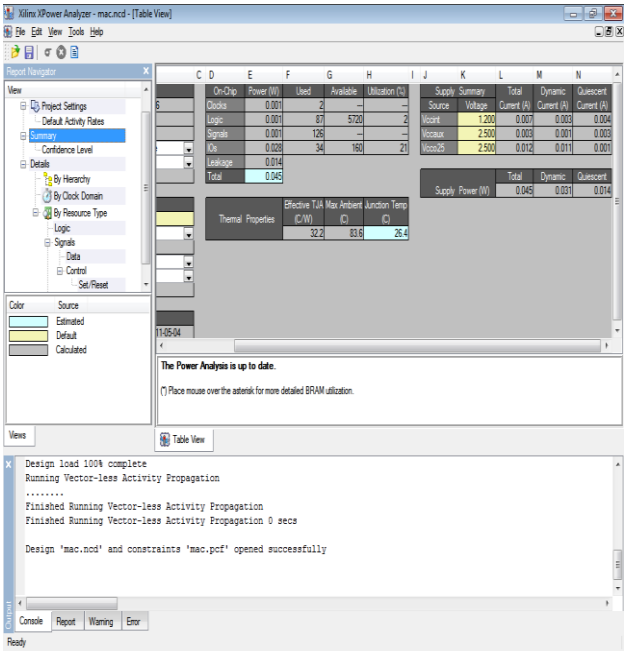
Advantages

- Feed Forward Cutset Free technique decreases the Pipeline stages.
- Less area and shorter critical path delay when using the concept of DADDA multiplier.
- Power consumption is low.

RESULT AND DISCUSSION

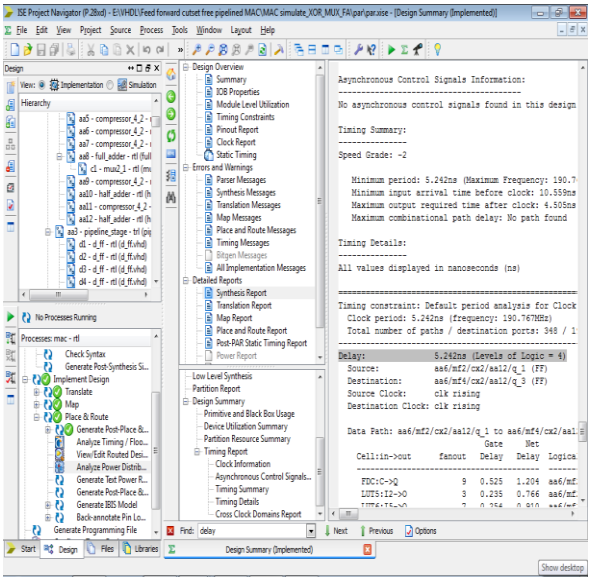
POWER REPORT

MAC _XOR_MUX_FA



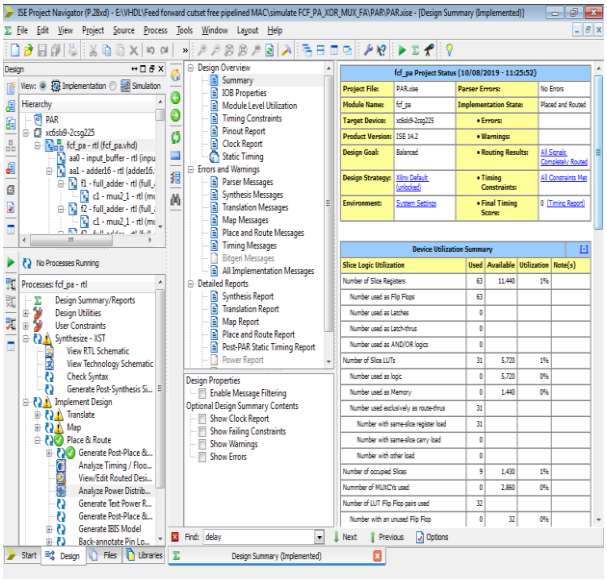
Delay Report

MAC_XOR_MUX_FA:

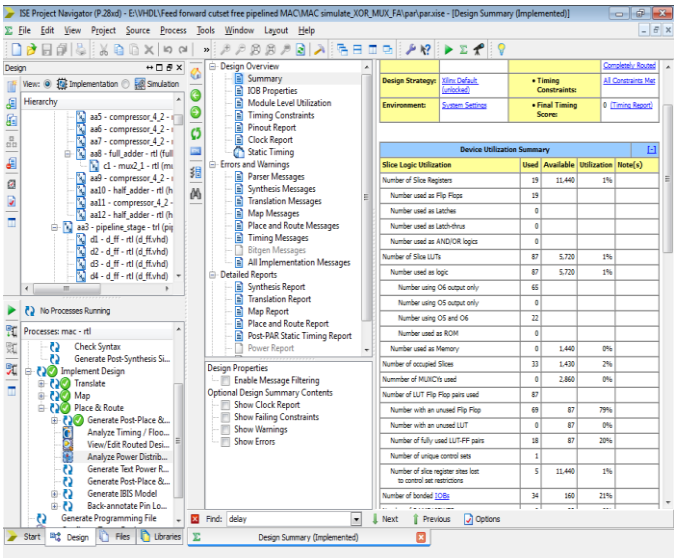


Area Report

FCF_PA_XOR_MUX_FA

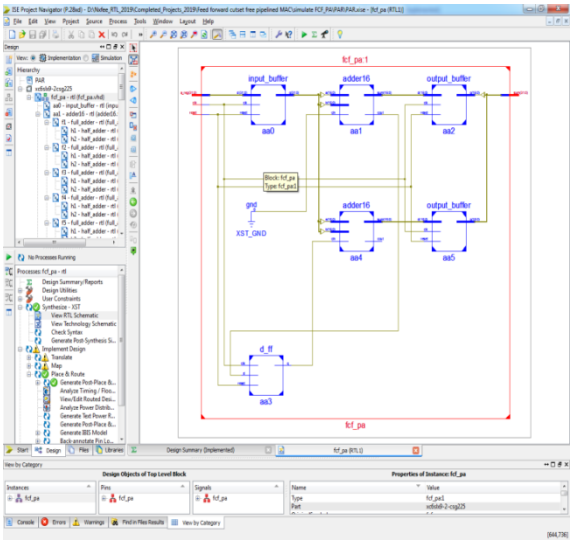


MAC_XOR_MUX_FA



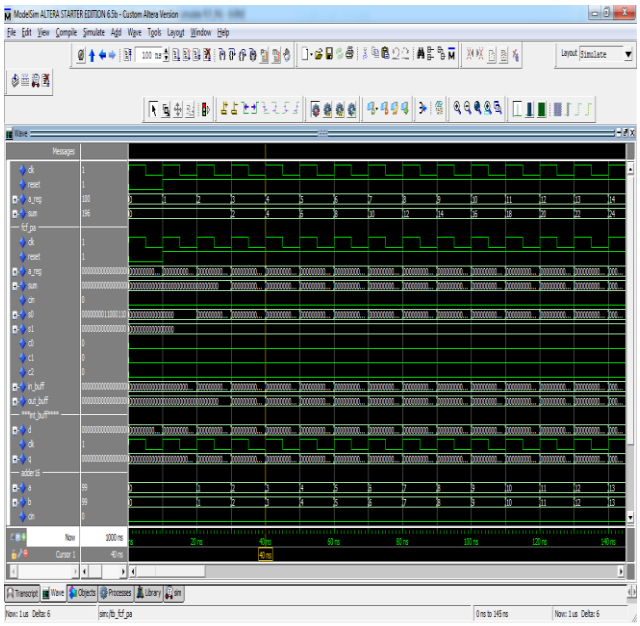
RTL View

FCF_PA_XOR_MUX_FA



Simulation Output

FCF_PA_XOR_MUX_FA



CONCLUSION

We introduced the FCF pipelining method in this paper. In the proposed scheme, the number of flip-flops in a pipeline can be reduced by relaxing the feed forward-cutset constraint, thanks to the unique characteristic of the machine learning algorithm. We applied the FCF pipelining method to the accumulator (FCF-PA) design, and then optimized the power dissipation of FCF-PA by reducing the chance of undesired data transitions (MFCF-PA). The proposed scheme was also expanded, and applied to the MAC unit (FCF-

MAC). For the evaluation, the conventional and proposed MAC architectures were synthesized in a 65-nm CMOS technology. The proposed accumulator showed the reduction of area and the power consumption by 17% and 19%, respectively, compared with the accumulator with the conventional CLA adder-based design. In the case of the MAC architecture, the proposed scheme reduced both the area and power by 20%. we will design MAC Unit using MCF_PA with 4:2 compressor and XOR MUX Full adder with compared Conventional full adder designs in the future.We believe that the proposed idea to utilize the unique characteristic of 4:2 compressor computation for more efficient MAC design can be adopted in many hardware accelerator designs.

REFERENCES

[1]. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Image Net classification with deep convolution neural networks," in Proc. Adv. Neural Inf. Process. Syst., 2012, 1097–1105.
[2]. K. Simonyan and A. Zisserman. "Very deep convolution networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556. 2014.
[3]. C. Szegedy et al., "Going deeper with convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, 1–9.
[4]. A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), 2013, 6645–6649.
[5]. J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in Proc. Adv. Neural Inf. Process. Syst., 2015, 577–585.
[6]. Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolution neural networks," IEEE J. Solid-State Circuits, 52, 2017, 127–138.
[7]. B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10tops/w sub word-parallel dynamic-voltage-accuracy frequency-scalable convolution neural network processor in 28nm FDSOI," in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC), 2017, 246–247
[8]. C. S. Wallace, "A suggestion for a fast multiplier," IEEE Trans. Electron. Comput., 1, , 1964, 14–17, Feb. 1964.
[9]. L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, vol. 34, no. 5, pp. 349–356, Mar. 1965.
[10]. P. F. Stelling and V. G. Oklobdzija, "Implementing multiply-accumulate operation in multiplication time," in Proc. 13th IEEE Symp. Comput. Arithmetic, Jul. 1997, pp. 99–106.
[11]. K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. New Delhi, India: Wiley, 1999.
[12]. T. T. Hoang, M. Sjalander, and P. Larsson-Edefors, "A high-speed, energy-efficient two-cycle multiply-accumulate (MAC) architecture and its application to a double-throughput MAC unit," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 12, pp. 3073–3081, Dec. 2010.
[13]. W. J. Townsend, E. E. Swartzlander, and J. A. Abraham, "A comparison of Dadda and Wallace multiplier delays," Proc. SPIE, Adv. Signal Process. Algorithms, Archit., Implement. XIII, vol. 5205, pp. 552–560, Dec. 2003, doi: 10.1117/12.507012.
[14]. M. Courbariaux, Y. Bengio, and J.-P. David, "Binary Connect: Training deep neural networks with binary weights during propagations," in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 3123–3131.
[15]. M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: Image Net classification using binary convolution neural networks," in Proc. Eur. Conf. Comput. Vis. Springer, 2016, pp. 525–542.
[16]. M. Gao, J. Pu, X. Yang, M. Horowitz, and C. Kozyrakis, "Tetris: Scalable and efficient neural network acceleration with 3d memory," in Proc. 22nd Int. Conf. Archit. Support Program. Lang. Oper. Syst., 2017, pp. 751–764. [17] A. Parashar et al., "SCNN: An accelerator for compressed-sparse convolution neural networks," in Proc. 44th Annu. Int. Symp. Comput. Archit., 2017, 27–40.