International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 15
May 2018

Low Cost VLSI Architecture for Sample Adaptive Offset Encoder in HEVC

MD. Nasreen Sulthana
Email Id: md.nasreen7994@gmail.com
Nimra College of Engineering and Technology
MD.Shamshad Begum
Shas.shamshad @gmail.com
Hod ,Mtech,Phd Associated Professor
Nimra College of Engineering and Technology

ABSTRACT: Basically in high video efficiency coding (HVEC), a sample adaptive offset is used as in loop filtering block. This increases the compression efficiency in computer graphics up to 23%. So exhaustive operations are required to optimize the SAO parameters. In this paper a low cost high throughput VLSI implementation is proposed for the parameter estimation phase. Here the proposed architecture reduces the cost in terms of gates count and it is prototyped using 65nm CMOS technology.

Keywords—High efficiency video coding (HEVC), in-Loop filtering, and sample adaptive offset (SAO), VLSI, SAO encoder.

I. INTRODUCTION

High Definition (HD) and beyond-HD (4k or 8k) resolution videos are in demand for effective compression techniques. High efficiency video coding (HEVC/H.265) is standardized to be the next generation of video coding as a successor to AVC /H.264/MPEG-4. It was jointly developed by ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) standardization organizations. HEVC could achieve 50% bit rate encoding reduction for equal perceptual video quality compared with its predecessor.

In-loop filtering block has been adopted in the HEVC main profile to increase compression efficiency by subjectively improving the quality of the reconstructed picture. The In-loop filter mainly consists of two consecutive modules, De blocking Filter (DB) then Sample Adaptive Offset filter (SAO). It mainly works to eliminate block-based processing and quantization artifacts, such as blocking, ringing and color biases artifacts .SAO filter works to prevent ringing artifacts which appear when using large numbers of interpolation filter's taps or large transform size. Ringing artifacts obviously appear near object's edges or

Sharp transitions. It appears as sparkles near these edges.

A simplified block diagram of HEVC encoder is shown in Figure (1). Power hungry and real time video encoders are in demand for low cost high performance architectures. To the best of our knowledge, there is only two VLSI SAO encoder architecture. Zhu et. al. Implemented architecture on 65nm technology that processes 8Kx4K @120fps with 156.3 Kgates, while the architecture implemented by Mody et. al. on 28 nm technology can process 4K@60fps by 300 Kgates. In this paper, we propose high throughput, high performance, and low cost VLSI architecture for SAO Parameter estimation block.

Fig .1.Simplified Block Diagram of HEVC Encoder

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 15
May 2018

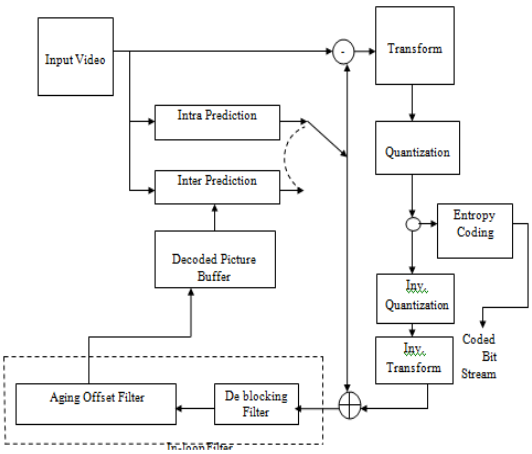


Fig. 3. Existed system

The frame is classified into Code tree units (CTUs) in HEVC. Each CTU contains three code tree blocks (CTBs) components, Luma CTB (L) and two chroma CTBs (Cband Cr) plus syntax element. From below figure (2) we can observe the merge mode and CTU component. AO is a processor based on code tree unit (CTU). AO is operated in three modes for the current CTU: processing a new parameter (New/No Merge), assumes parameters of the upper CTU (Merge Up), or assume parameters of the left CTU (Merge Left). In "New" mode, it is one of the three types, Band Offset (BO) type, Edge Offset (EO) type or AO not applied (NA). The optimum mode/type and the corresponding parameter are selected based on histogram analysis and rate distortion optimization.

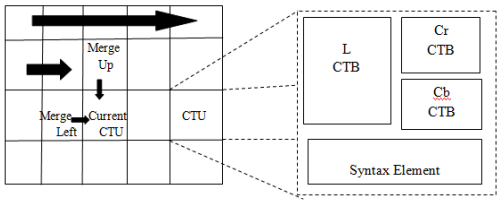
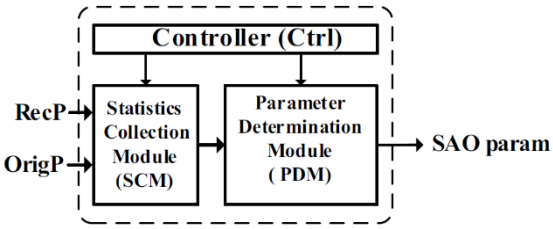


Fig .2.Merge Mode and CTU Component

II. EXISTED SYSTEM

The below figure (3) shows the architecture of existed system. The existed system is divided mainly into three modules they are given as Statistics collection module, parameters determination module and controller as shown in figure (3). Let us discuss each module in detail manner.



a) SCM Module:

SCM module produces N and E of 4X4 Pixels per cycle. In this each cycle 6X6 RecP and 4X4 OrigP are registered to SCM. An *allocator* is used to reorder RecP and D to be inputted to both EO and BO engine. Each EO class x engine works on the corresponding allocator output, it mainly consists of four Cat.x matching sub modules. The output of each Cat.x matching sub modules is summed up for corresponding N of this category. Next coming to BO engine, it consists of 30 cores. Each core works on the same 4X4 allocator output block of pixels.

b) Parameters determination module:

PDM works for 56 cycles to produce the optimum SAO parameters of the current CTB. PDM mainly consists of four cores, E and N components which generators (ECG and NCG respectively), Minimum cost and offset generator (MCOG) and Group and Compare core (GAC) plus storage elements. In this 48 N&E pairs stack is processed on per cycle to get minimum Cost. Another eight N&E pairs which correspond to the stored parameters of the Upper and Left CTUs are processed too to get costs in case of reusing of upper left CTU's parameters.

c) Controller:

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 15
May 2018

Controller works for reducing switching activity, consequently the power consumption of the SAO encoder. It inputs data and triggers clock only when required. The Clock gating technique is adopted to ensure that pipelining.

From all these it can be observed that existed system does not provide better security. So a new system is proposed which is discussed in below section.

III. PROPOSED SYSTEM

The main objectives of AES are high level security, adoptable to diverse application, efficient and exportable. In this project work, the plain text of 128 bits is given as input to encryption block in which encryption of data is made and the cipher text of 128 bits is throughout as output. The key length of 128 bits is used in process of encryption. The AES algorithm is a block cipher that uses the same binary key both to encrypt and decrypt data blocks is called a symmetric key cipher. A commonly accepted definition of a good symmetric key algorithm, such as the AES, is that there exists no attack better than key exhaustion to read an encrypted message.

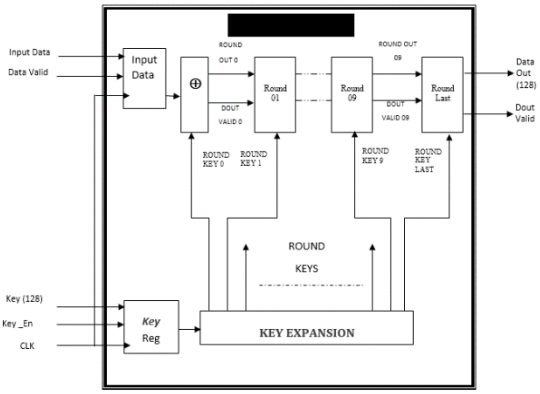


Fig. 4. Top Level Block Diagram of AES Algorithm

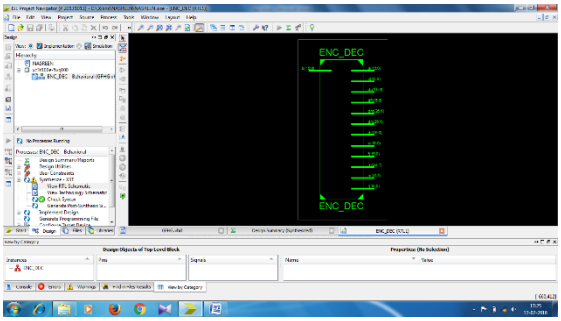
Fig. 5. RTL schematic

were carried out in which the last round will be performed separately. For both its Cipher and Inverse Cipher, the AES algorithm uses a round function that is composed of four different byte-oriented transformations:

- ➢ Byte substitution using a substitution table (S-box)
- ➢ Shifting rows of the State array by different offsets
- ➢ Mixing the data within each column of the State array
- ➢ Adding a Round Key to the State

Above mentioned functions were carried out for every individual round and in the last round the third function, that is, Mixing the data within each column of the State array will not be performed. Hence the last round is carried out separately. Based on the key provided, the new set of keys will be generated in the Key Expansion block and is given to the each round as input.

IV. RESULTS



algorithm. Also the basic inputs to the system and the outputs from the system were clearly represented. As per the standard, 10 rounds for 128 bits key length

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 15
May 2018

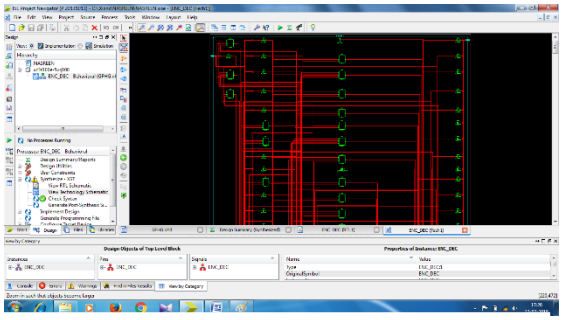


Fig. 6. Technology schematic

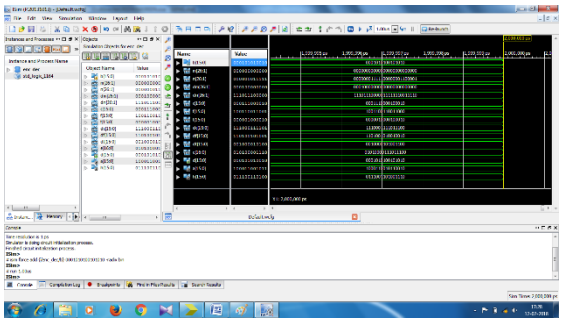


Fig. 7. Output

V. CONCLUSION

We have presented a VLSI architecture for AES algorithm that performs both the encryption and decryption. S-boxes are used for the implementation of the multiplicative inverses and shared between encryption and decryption. The round keys needed for each round of the implementation are generated in real-time. The main purpose of encryption is to hide data from unauthorized usage. In this paper, we purposed a method to employ the crypto processor run in integration with a General Purpose Processor. In this direction, we have presented a pipeline version of AES algorithm that can encrypt data.

VI. REFERENCES

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[2] C.-M. Fu, E. Alshina, A. Alshin, Y.-W. Huang, C.-Y. Chen, C.-Y. Tsai, C.-W. Hsu, S.-M. Lei, J.-H. Park, and W.-J. Han, "Sample Adaptive Offset in the HEVC Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1755–1764, Dec. 2012.

[3] J. Zhu, D. Zhou, S. Kimura, and S. Goto, "Fast SAO estimation algorithm and its VLSI architecture," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 1278–1282.

[4] M. Mody, H. Garud, S. Nagori, and D. K. Mandal, "High throughput VLSI architecture for HEVC SAO encoding for ultra HDTV," 2014, pp. 2620–2623.

[5] El Gendy, Sayed, Ahmed Shalaby, and Mohammed S. Sayed. "Fast parameter estimation algorithm for sample adaptive offset in HEVC encoder." In 2015 Visual Communications and Image Processing (VCIP), pp. 1-4. IEEE, 2015.

[6] C.-M. Fu, C.-Y. Chen, Y.-W. Huang, and S. Lei, "Sample adaptive offset for HEVC," 2011, pp. 1–5.

[7] https://hevc.hhi.fraunhofer.de/trac/hevc/browser/tags/HM-16.2.

[8] Y. Voronenko and M. Puschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, no. 2, p. 11–es, May 2007.

[9] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 10." Jan-2013.

[10] Qing Wu, M. Pedram, and Xunwei Wu, "Clock-gating and its application to low power design of sequential circuits," *IEEE Trans. Circuits Syst. Fundam. Theory Appl.*, vol. 47, no. 3, pp. 415–420, Mar.