Low Power Area Efficient High Data Rate 16-bit AES Crypto Processor

Habibullah Jamal
Vice Chancellor and
Professor in Electrical Engineering,
University of Engineering and
Technology Taxila, Pakistan
Email: drhjamal@uettaxila.edu.pk

Sheikh. M. Farhan
University of Engineering and
Technology, Taxila, Pakistan
Email:smfarhan@uettaxila.edu.pk

Shoab A Khan
Center for Advanced Studies in
Engineering, Islamabad, Pakistan
Email:shoab@case.edu.pk

***Abstract* --This paper presents a 16-bit AES architecture for low power and high bit rate applications. The novelty is in breaking the original 32-bit boundary based AES algorithm into a scalable architecture to work with 8-bit and 16-bit data set. 8-bit architecture is already developed. This new work offers a choice to the designer to use 8-bit or 16-bit algorithm for area and power efficient FPGA implementation. The novelty of the new development is still around the mix-column design. The complex matrix multiplication component and standard transformations of the 32-bit AES algorithm are transformed now to support 16-bit operations as well, simultaneously qualifying for applications requiring high data rates. The design has been further embellished by a memory based micro-programmed controller, which simplifies the control process of the algorithm and makes the FPGA platform viable for effective hardware utilization. The proposed architecture technique reuses same hardware resources for both key expansion and encryption**

***Index Terms*— AES, Encryption, Area Efficient, Low Power, Rijndael, Security**

I. Introduction

THE need for a low power, lesser area digital design is found in almost every application these days. As the number of components on a single chip increased exponentially, so did the need of compact and handheld devices. This led engineers to think on shrinking the size of the technology used in such devices and design practices.

Data path width plays significant role in any digital system design. The impact of data-path width is directly reflected on internal buses, their word-lengths and memories used in the design and hence the area required. In our earlier work, the 32-bit AES algorithm was modified to work on 8-bit architecture. The design was implemented in a 2Mbit telemetry modem. We wanted to design architecture to work with our new communication receiver design with more stringent data rate requirements. The design has been modified to fit on a 16-bit architecture to cater for the applications requiring high data rates consuming low area and power.

This paper is organized as follows: Section II briefly describes the 32-bit AES algorithm, section III, IV and V discuss the extended 16-bit AES architecture. Results and comparisons are drawn in section VI. Finally, the concluding remarks are presented in section VII. Section VIII lists the references.

Manuscript received May 3, 2006.

F. A. Author is with University of Engineering and Technology Taxila, Pakistan (ph: +92-51-9047401 ; Mob: +92-300-5008337; fax: +92-51-9047420; email: drhjamal@uettaxila.edu.pk).

S. B. Author is with University of Engineering and Technology Taxila, Pakistan;

T. C. Author is with Centre of Advanced Studies in Engineering Islamabad, Pakistan

II. AES 32-bit Algorithm

AES is a 32-bit, iterative symmetric block cipher and works on a fixed block size of 16 bytes (128 bits) as explained in [1,2]. The standard has the provision to work with variable key sizes of 128, 192 and 256 bits. Since it is an iterative block cipher, the same operations are performed many times on a fixed number of bytes. These operations can be categorized as Add Round Key (ARK), Byte Substitution (BS), Shift Rows (SR) and Mix Columns (MC) as shown in *Figure 1*. The data-path is 32-bit wide throughout the architecture. For more details on AES Algorithm specifications, refer to [3].

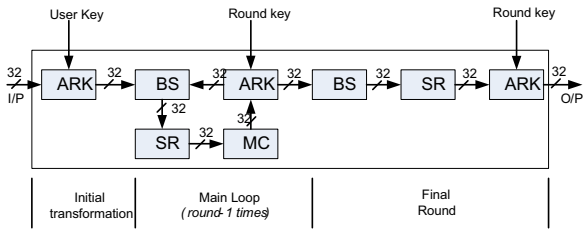

Figure 1 AES algorithm flow graph (32-Bit)

III. CUTTING THE ALGORTIHM DOWN TO 16-BIT

The 32-bit algorithm is studied to make it work for 16-bit architecture. As this is an extension to our earlier work [4], the design of the 8-bit AES processor was kept in mind. We concluded that the design can be transformed to work with 16-bit numbers. The modifications require similar changes in the existing 32-bit AES architecture as were done in 8-bit design. The proposed design works by eliminating the SR transformation from all round sequences. The original 32-bit AES algorithm requires 4 S-boxes implementation, however the design reduces it to 2 S-boxes. Instead of implementing the control logic in FPGA hardware resources (LUTs, FFs, MUXs), the controller is mapped as microprogrammed state machine on the available Block SelectRAM™ in FPGAs. Figure 2 presents the theoretical design flow of the proposed AES design.

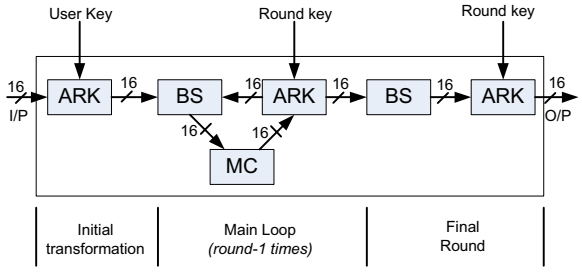

Figure 2 AES flow graph (16-Bit)

The data and key memories are dual ported memories. Elimination of SR transformation is achieved through simple address translation which writes the plain text into memory in an already row shifted format. This address translation is shown in Figure 3.

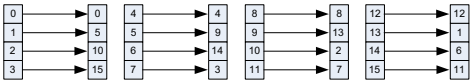

Figure 3 Address translation

The architecture proposed in this article implements AES algorithm for 128-bit cipher block and 256-bit cipher key. The proposed design has the ability to use the same encryption architecture for key expansion thus completely eliminating the need of a separate hardware unit for the same task. Memories, buses, registers and data-path are kept at the most, 16-bits wide. That is why the name, Crypto Processor-16 (CPR-16).

IV. BUILDING BLOCK OF CPR-16

CPR-16 mainly consists of the following major units: Dual ported 8-16 bit memories, 16-bit data-path and microprogrammed controller. The next section describes each unit in detail.

A. CPR-16 Memories

As stated earlier and as already employed in 8-bit AES architecture [4], CPR-16 uses dual ported Block SelectRAM™ (BRAMs) available in FPGAs. These BRAMs offer better access time and performance as compared to external onboard memories. For plain text and raw data, two dual ported BRAMs (BRAM1, BRAM2) are used. BRAM1 is 16x8 bits (16 byte locations) from one port and 8x16 bits (8 word locations) from the other port. BRAM2 is 8x16 bits from both the ports. The incoming plain text bytes are stored in BRAM1 after address translation. Once plain text is written into BRAM1, CPR-16 uses both BRAMs in a ping-pong fashion to process the data iteratively. Cipher and round keys are also stored and managed in two dual ported BRAMs (BRAM3, BRAM4). BRAM3 is 32x8 bits from one port and 16x16 bits from the other port. BRAM4 is 104x16 bits from both the ports. Once the cipher key is written into BRAM3, CPR-16 takes over dual ported key memories for key expansion and stores generated round keys into BRAM4. BRAM5 is a 32x16 bits memory, which is used to store the encrypted data. Figure 4 shows block diagram of CPR-16 with its associated memories.

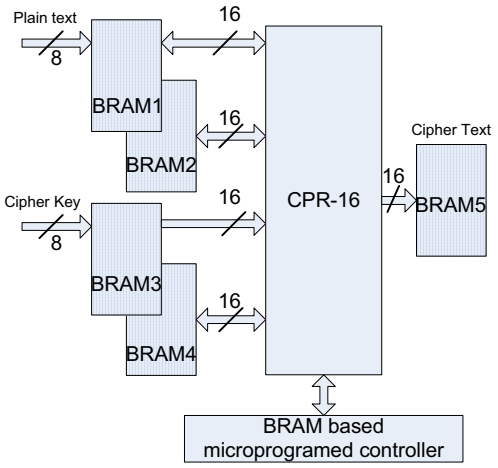

Figure 4 CPR-16 and memories

In the first 8 cycles of encryption, plain text is read as 16-bit words from BRAM1 into CPR-16 and is written back at the corresponding addresses of BRAM2 after completing first round. In the next 8 cycles, the partially processed data is read into CPR-16 from BRAM2 and written back to BRAM1 at the corresponding addresses at the end of second transformation. This process continues until all iterations are exhausted Round keys are also read as 16-bit words in the same manner from the key memory, which are then used in the ARK transformation. The key memory is divided into 15 pages, each page holding 8 words of round keys. For each iteration, a new page of round keys is read from the memory. Data memory (BRAM1, BRAM2) is initialized with round constants (RCONs) which are used in key expansion phase prior to any encryption request. Once the key expansion is

done, the data memory is available for storing plain text and cipher data processed through different round iterations.

B. The Data Path

The entire data path is kept 16-bits wide in CPR-16. The number of S-boxes has been cut down to two as compared to four in the original 32-bit AES. Figure 5 shows the complete data path section of CPR-16.

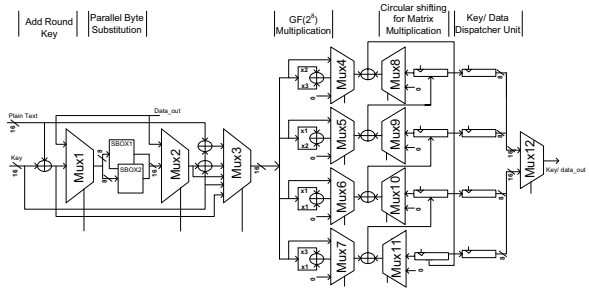

Figure 5 CPR-16 Data path

Data path for ARK and BS is straightforward. The mix column data-path implements the matrix multiplication in a word systolic manner. The mix-column section can be treated as a hybrid of 8 and 16-bit data path whose input and output are 16 bits wide whereas internal operations are performed by splitting 16-bit data path into 8-bit buses. Mix Column stage is a linear transformation based on GF(2^8) multiplication with a modular polynomial x^4+1 which can be implemented with XOR operations [3]. As shown in Figure 5, four partial results are computed by multiplying the resulting data, after parallel byte substitution, with the first and second column of the matrix [3], resulting in a word wide matrix multiplication. In the next cycle, same process continues for the next data word which is then multiplied with the third and fourth column of the matrix.

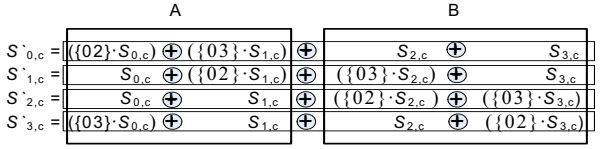

Figure 6 Equations resulting from GF(2^8)Matrix Multiplication [3]

Since the same multiplier coefficients are repeated in the Mix Column stage therefore they have been re-used. After every two clock cycles, a complete matrix multiplication occurs. In the first cycle, section A of figure 6 is computed. In the second cycle, section B is computed which is then XORed with the circularly shifted partial products computed in the first cycle. The resulting words are dispatched for the next transformation through the dispatcher unit.

Another feature of CPR-16, that makes it unique and a true candidate for a low area design is its microprogrammed controller implementation using BRAMs. A number of multiplexers are inserted at different data feeding points, which are controlled by the microprogrammed controller allowing both key expansion and encryption with the same hardware. Following sections describe how the same data-path is re-used for both key expansion and encryption.

1) Data Path- Key Expansion

After writing the initial 256-bits of cipher-key into the key memory, the key expansion takes place. The last four bytes of cipher key are read into CPR-16 as two 16-bit words while bypassing the ARK and BS stages. These last two words are directly fed into the MC stage, which rotates these words giving them a left circular shift of one byte. This rotated word is the output from the MC stage, which is fed back into Mux1 for parallel byte wise substitution thus implementing the BS transformation. In the same cycle when the parallel byte substitution takes place, key[i- Nk] is also fetched and the two words are XORed together in each cycle. Parameter Nk is the number of 32 bit words comprising the cipher key. The resulting word is then XORed with Rcon[i], which is made available from the data memory. This will result in two words of generated round keys, which are written back into the key memory. The same process continues for other iterations as per the AES algorithm [3]. The 60th iteration will give 120 words of round keys written into the key memory.

2) Data Path –Encryption

Data encryption can only take place after key expansion. On receiving the encryption request, the cipher data and round keys are fetched from the memory word by word at every cycle. The two words (key, data) are XORed with each other for ARK transformation. Mux1 directs the resulting data to two Sbox ROMs. The lower 8-bits become the address to S-box1 and higher 8-bits become the address to S-box2 and get substituted with the corresponding data bytes residing at those addresses. After BS, the date is directed to MC stage for matrix multiplication as described in part *B* of section IV. This completes one round and the output of the mix column is written back into the data memory. At the end of 15 rounds, the fully encrypted data is available in the output memory.

V. MICROPROGRAMMED CONTROLLER

Micro programmed state machine based controller design provides further saving in area. Instead of implementing the state machine using valuable FPGA resources, the state machine is microprogrammed using two BRAMs (512x36 bits) available in FPGA, cutting down the FPGA slices usage by 30%. The microprogrammed controller drives CPR-16 either in key expansion mode or data encryption mode by generating status/control signals for data transportation, key expansion, encryption and addresses for BRAMs for reading and writing the data from/into the memory. Figure 7 shows the microprogrammed state machine implementation [5].

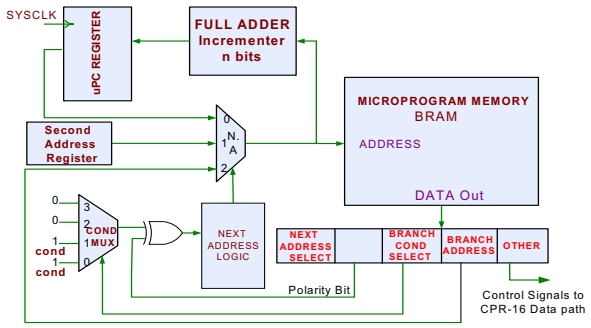

Figure 7 Microprogrammed Controller Implementation

VI. RESULTS

CPR-16 was synthesized using ISE 8.1i and simulated using Modelsim 5.7g. The target device selected was Virtex2 XC2V1000-6 [11]. The operating speed achieved after place and route and strict timing constraints was far better as compared to many reported designs. The synthesis results show that the core utilizes only 228 slices. The CPR-16 core can be operated at frequencies up to 150 MHz yielding data rates in the range of 110 Mbps. A comparison of number of slices between different 32 bit, 8-bit and the proposed 16-bit AES architecture is shown in Table 1.

Table 1 Comparison Table

Design	Process	Device	Slices	Speed(MHz)
[4]	CPR-8	Xilinx (XC2V1000)	337	110
[6]	Encr	Xilinx (XCV1000)	5302/10992	14.1/31.8
[7]	Encr	Xilinx (not specfied)	5673	--------
[8]	Decr	Altera(EPF10K)	2885	41.5
[9]	Encr/Decr	Altera(APEX1K4001)	845LE	---------
[10]	Encr/Decr	Xilinx (Virtex)	2902	25.9
CPR-16 Exp/ Enc		Xilinx(XC2V1000)	228	150

VII. CONLCUSION

The paper presented our work on designing 16-bit AES architecture. The algorithm is mapped to a 16-bit architecture for area and power efficient implementation. The technique of transforming non-linear algorithms like AES to digit serial architecture is a novel addition to the existing techniques of digital serial architectures and displays a promising methodology of designing area and power efficient architectures for this type of algorithms. The proposed architecture is best suited for applications requiring high data rates in ranges of ~ 110 Mbits/s. Future work includes mapping of AES algorithm on a run-time reconfigurable platform.

VIII. REFERENCES

[1] [1] J. Daemen, V.Rijmen “The Rijndael Block Cipher” AES proposal, First Candidate conference (AESI), August 20-22, 1998.

[2] [2] Joan Daemen, Vincent Rijmen “ The Design of Rijndael, AES-The Advanced Encryption Standard” Springer-Verlag Berling Heidelberg New York 2002.

[3] [3] “Announcing the Advanced Encryption Standard (AES)”, Federal Information Processing Standards Publication 197 November 26, 2001

[4] [4] Sheikh Muhammad Farhan, Shoab A Khan, Habibullah Jamal, “Mapping of High-bit Algorithm to Low-bit for Optimized Hardware Implementation”, Proceedings of the 16th International Conference on Microelectronics, ICM 2004, December 6-8, 2004, Tunis, pp. 148 – 151, TUNISIA.

[5] [5] Michael A Lynch,” Microprogrammed State Machine Design”, ISBN 0849344646, CRC Press, January 1993.

[6] [6] A.J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, “An FPGA Based Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists”, Proc. Third Advanced Encryption Standard (AES) Candidate Conf., Apr. 2000.

[7] [7] A. Dandalis, V.K. Prasanna, and J.D.P. Rolim, “A Comparative Study of Performance of AES Final Candidates Using FPGAs”, Proc. Third Advanced Encryption Standard (AES) Candidate Conf., Apr. 2000. (This work has also been published in the Proc. CHES 2000, Aug. 2000).

[8] [8] P. Mroczkowski, “Implementation of the Block Cipher Rijndael Using Altera FPGA”, http://csrc.nist.gov/encryption/aes/round2/pubcmnts.htm, 2001.

[9] [9] V. Fischer and M. Drutarovsky, “Two Methods of Rijndael Implementation in Reconfigurable Hardware”, Proc. CHES 2001, May 2001.

[10] [10] K. Gaj and P. Chodowiec, “Comparison of the Hardware Performance of the AES Candidates Using Reconfigurable Hardware”, Proc. Third Advanced Encryption Standard (AES) Candidate Conf., Apr. 2000.

[11] [11] Xilinx VirtexTM II Platform FPGAs. URL: www.xilinx.com, October 14, 2003