International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Implementation of Secured Data Transmission System on Customized Zynq SoC

D. Dhanalaxmi [1], V. Roopa Reddy[2]

[1]M.Tech Scholar, VLSI-SD, Department of ECE, Teegala Krishna Reddy Engineering College, JNTUH, India
[2]Assistant Professor, Department of ECE, Teegala Krishna Reddy Engineering College, JNTUH, India

Abstract: ***Data security and privacy are major concerns for communication in a channel or storing it in storage devices. The need of an information encryption has become increasing today, to prevent it from unauthorized reception or thefts during communication. Modern communication channel address this problem by using cryptographic algorithms, such as DES, 256-bit Advanced Encryption Standard (AES). This Encryption / Decryption system is presently proposed in the Missile Telemetry system for sending telemetry signal to ground system after proper encryption. Instead of sending telemetry frame as it is, it is first encrypted and then sends. Suitable decryption scheme is performed at ground system to recover the original information from encrypted information after the reception. Most important concepts behind security is Encryption. In this system secrecy of the key is most important. The Advanced Encryption Standard (AES) is the main algorithm used to ensure security and privacy in several different applications. The Xilinx Zynq-7000 All Programmable SoC (AP SoC) and Vivado Design Suite together provide hardware/software development platform for implementation of encryption algorithms. This new methodology facilitates hardware/software system function partitioning for advanced performance, and the optimized joint hardware/software design flow provides increased efficiency.***

Keywords: ZYNQ 7000 AP SOC, AES, AXI interconnect, Xilinx, Vivado, Key, encryption

1. Introduction

The purpose of a telemetry system is to collect data at a place that is remote or inconvenient and to relay the data to a point where the data may be evaluated. Typically, telemetry systems are used in the testing of moving vehicles such as aircraft, and missiles. Telemetry systems are a special set of communication systems. During Development flights, Telemetry system is necessary to evaluate the Performance of the flight vehicle. A large number of physical parameters i.e., Temperature, Pressure, Strain, Vibration, Electrical parameters from various subsystems are monitored [1].The telemetry data acquired is used for post flight data analysis. On-board telemetry system consists of Sensors, Signal Conditioners, MIL-STD-1553 interfaced PCM Encoder, S-band transmitter, Power Divider and Antenna System. The output of signal Conditioners are sampled, multiplexed, encoded using A/D Converters and converted to a serial PCM bit stream of 1 Mbps. This project describes Secured Telemetry data transmission in Missile Telemetry system for sending telemetry signal to ground system after proper encryption. Instead of sending telemetry frame as it is, it is first encrypted and then send. An information encryption is a process of encoding information in such a way that no decryption is process of converting the encoded (encrypted) signal back to original information. In order to implement security in Telemetry system this paper discusses ZYNQ SoC implementation of the Advanced Encryption Standard (AES). AES is based on the block cipher Rijndael algorithm. It is a symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits.

A SoC is specially designed to meet the standards of incorporating the required electronic circuits of numerous computer components onto a single integrated chip. SoC fabricates all necessary circuits into one unit. Because SOC includes both the hardware and software, it uses less power, has better performance, requires less space and is more reliable than multi-chip system.

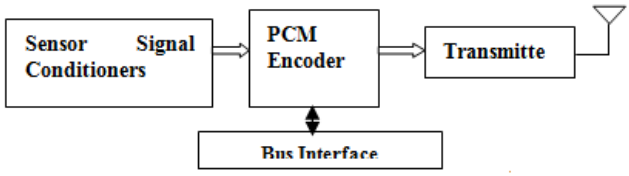

Figure 1: Telemetry System

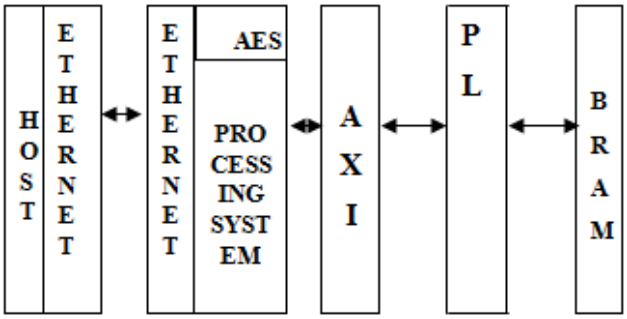

Figure 2: Block Diagram of secure data transmission system on zynq soc

The following tasks are carried out to fulfill the objective of the paper are study the architecture of zynq-7000 SOC and its interfaces. Understanding and implementation AES algorithm. Understandings of AXI interconnect and their read and write signals. Configuration and Booting of PS and PL for Hardware and software. Processor electronics and its interfaces were designed using Vivado software. Implementation of PL logics in Xilinx ISE 13.2v.

2. Implementation of AES Algorithm on ZYNQ SOC

This section presents the design of the sub modules in implementing Fig 2. This covers the features of Zynq SoC, interfacing of AXI Interconnect, Implementation of AES algorithm.

2.1 Introduction of zynq

The Zynq-7000 family is based on the Xilinx All Programmable SoC (AP SoC) architecture. These families integrated with a feature-rich dual-core ARM Cortex-A9 MPCore based processing system (PS) and Xilinx programmable logic (PL) in a single device, which is built on a state-of-the-art of High-performance & Low-Power (HPL) and 28 nm technology. The ARM Cortex-A9 MPCore CPUs are the heart of the PS which also includes on-chip memory, external memory interfaces, and a rich set of I/O peripherals. The Zynq-7000 family offers the flexibility and scalability of an FPGA, while providing performance, power, and ease of use typically associated with ASIC and ASSPs. The range of devices in the Zynq-7000 AP SoC family enables designers to target cost-sensitive as well as high-performance applications from a single platform using industry-standard tools. While each device in the Zynq-7000 family contains the same PS, the PL and I/O resources vary between the devices. The Zynq-7000 AP SoC contains a large number of fixed and flexible I/O. Zynq-7000 AP SoC has a constant 130 pins dedicated to memory interfaces (DDR I/O), multiplexed peripherals (MIO), and control. Programmable logic provides additional pins for SelectIO resources (SIO) and multi-gigabit serial transceivers (GTX) that scale by device as well as fixed pins for configuration and analog-to-digital conversion (XADC). SIO can be used to extend the MIO to further leverage the fixed peripherals of the processing system (PS) [2].

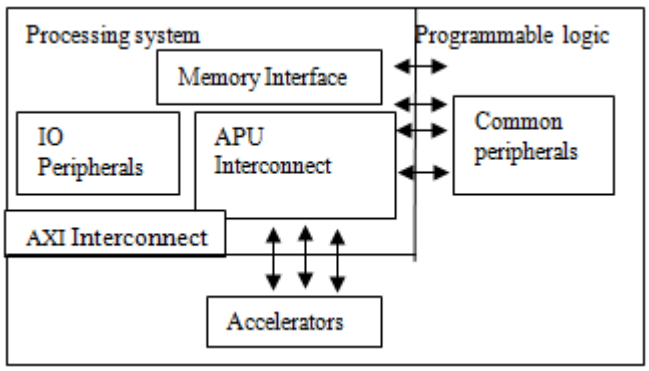

Figure 3: Zynq FPGA Overview

Zynq-7000 AP SoC is composed of the following major functional blocks as shown in the Fig. 3.

1. Processing System (PS)

A) Application processor unit (APU)
B) Memory interfaces

2. I/O peripherals (IOP)
3. Interconnect
4. Programmable Logic

2.1.1 Processing System

Application processor unit (APU)

The application processing unit (APU), located within the PS, contains two ARM Cortex A9 processors with NEON co-processors that are connected in an MP configuration sharing a 512 KB L2 cache. Each processor is a high-performance and low-power core that implements two separate 32 KB L1 caches for instruction and data. The Cortex-A9 processor implements the ARMv7-A architecture with full virtual memory support and can execute 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java byte codes in the Jazelle state. The NEON coprocessor's media and signal processing architecture adds instructions that target audio, video, image and speech processing, and 3D graphics. These advanced single instruction multiple data (SIMD) instructions are available in both ARM and Thumb states.

The two Cortex A9 processors within the APU are organized in an MP configuration with a snoop control unit (SCU) responsible for maintaining L1 cache coherency between the two processors and the ACP interface from the PL. To increase performance, there is a shared unified 512 KB level-two (L2) cache for instruction and data. In parallel to the L2 cache, there is a 256 KB on-chip memory (OCM) module that provides a low-latency memory. An accelerator coherency port (ACP) facilitates communication between the programmable logic (PL) and the APU [9]. This 64-bit AXI interface allows the PL to implement an AXI master that can access the L2 and OCM while maintaining memory coherency with the CPU L1 caches.

Memory interfaces
Processing System include 128kB On-chip boot ROM, 256 KB on-chip RAM (OCM) (shared between the CPUs) with Byte-parity support. It also supports 2 types of external memory interfaces-Multiprotocol dynamic memory controller and Static memory controller

2.1.2 I/O Peripherals (IOP)

The general purpose I/O (GPIO) peripheral provides software with observation and control of up to 54 device pins via the MIO module. It also provides access to 64 inputs from the Programmable Logic (PL) and 128 outputs to the PL through the EMIO interface. The GPIO is organized into four banks of registers that group related interface signals. Each GPIO is independently and dynamically programmed as input, output, or interrupt sensing. Software can read all GPIO values within a bank using a single load instruction, or write data to one or more GPIOs using a single store instruction.

2.1.3 Interconnect

The APU, memory interface unit, and the IOP are all connected to each other and to the PL through a multilayered

ARM AMBA AXI interconnect. Interconnect is non-blocking and supports multiple simultaneous master-slave transactions. The interconnect is designed with latency sensitive masters, such as the ARM CPU, having the shortest paths to memory, and bandwidth critical masters, such as the potential PL masters, having high throughput connections to the slaves with which they need to communicate.Zynq-7000 AP SoC devices uses a variety of interconnect technologies, optimized to the specific communication needs of the functional blocks.

PS-PL Interfaces

The PS-PL interface contains all the signals available to the PL designer for integrating the PL-based functions and the PS. There are two types of interfaces between the PL and the PS. Functional interfaces which include AXI interconnect, extended MIO interfaces (EMIO) for most of the I/O peripherals, interrupts, DMA flow control, clocks, and debug interfaces. These signals are available for connecting with user-designed IP blocks in the PL. Configuration signals which include the processor configuration access port (PCAP), configuration status, and single event upset (SEU) and Program/Done/Init. These signals are connected to fixed logic within the PL configuration block, providing PS control.

2.1.4 Programmable Logic (PL)

The PL of Z-7020 is derived from Xilinx's Kintex-7 series FPGA technology. The PL is used to extend the functionality to meet specific application requirements. The PL includes many different types of resources including configurable logic blocks (CLBs), port and width configurable block RAM (BRAM), DSP slices with 25 x 18 multiplier, 48-bit accumulator and pre-adder (DSP48E1) etc. The PL provides a rich architecture of user-configurable capabilities.

2.1.5 Interrupts

The PS is based on ARM architecture, utilizing two Cortex-A9 processors (CPUs) and the GIC pl390 interrupt controller. The interrupt structure is closely associated with the CPUs and accepts interrupts from the I/O peripherals (IOP) and the programmable logic (PL). Each CPU has 5 sets of private peripheral interrupts (PPIs) with private access using banked registers. The PPIs include the global timer, private watchdog timer, private timer, and FIQ/IRQ from the PL. Software generated interrupts (SGIs) are routed to one or both CPUs. The SGIs are generated by writing to the registers in the generic interrupt controller (GIC). There are 16 software generated interrupts. The shared peripheral interrupts (SPIs) are approximately 60 generated by the various modules can be routed to one or both of the CPUs or the PL. The SPI interrupts from the PS peripherals are also routed to the PL.

2.1.6 Clock Interface

1. Processing system gets clock from external port, clock range from 30MHz to 50MHz.
2. PL gets clock from PS through clock generation module.
3. The bit rate clock in generated internally from PS & PL logic

2.2 AES Algorithm

In October 2000, the National Institute of Standards and Technology (NIST) announced the Rijndael algorithm as the Advanced Encryption Standard. Rijndael is a powerful private key encryption algorithm, which supports any combination of 128, 192, or 256-bit key and plaintext blocks. However, the Advanced Encryption Standard only recognizes plaintext blocks of 128-bits with key lengths of 128, 192, or 256-bits, which are referred to as AES-128, AES-192, and AES-256 respectively. Due to the systematic and parallel structure of the Rijndael algorithm, efficient hardware implementations are possible.

This standard specifies the Rijndael algorithm a symmetric block cipher that can Process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits. Rijndael was designed to handle additional block sizes and key lengths; however they are not adopted in this standard. Throughout the remainder of this standard, the algorithm specified herein will be referred to as "the AES algorithm." The algorithm may be used with the three different key lengths indicated above, and therefore these different "flavours" may be referred to as "AES-128", "AES-192", and"AES-256".

The AES algorithm operates on a 2-dimensional four-by-four array that is called the state of the data-block or in short state. The state represents the status of the data-block after a certain transformation. Each element of the state has a length of a byte (8-bits). This section will describe how the state is changed through the cipher. The AES algorithm diagram flow is depicted in figure 2.4. The encryption algorithm consists of an initial key addition and 12 rounds of arithmetic and logical operations. First, sixteen bytes of the plain text are copied to the buffer as a 32 bit size of four blocks. Secondly, the 128-bit key is added to the state. Afterwards, the state enters the first round operation. Each round is composed of the following four transformations.

- Byte Sub substitutes a byte by using a S-box substitution table.
- Shift Row changes the sequence of the input bytes.
- Mix column maps a word (4-bytes) to a single byte.
- AddRoundKey performs modulo-2 addition of a key with the input bytes.

In case when all fourteen rounds are performed, the state enters the final step. This final step is a reduced round operation and consists of the Byte Sub, Shift Row and the AddRoundKey transformations [5].

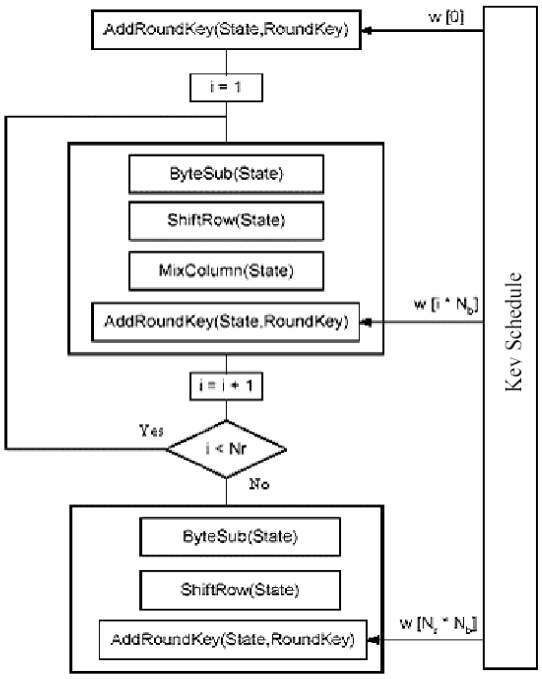

Figure 4: AES-Encryption Algorithm

2.3 Hardware Implementation of AES

The basic building block of the AES cipher block is the pipeline architecture acts as an interface between the AES core and external interfaces to the FPGA. For the ease of communication the data is received at the falling edge of the enable signal and transmitted at the rising edge of the delay enable. The load and the start signals are generated with each event in the enable signal by the driver block. The architecture guarantees data to the cipher block at the end of the each block enable signal. As the cipher block outputs the data in just 13 clock cycles it remains idle for the next 115 clock cycles, the transmitter pipeline waits and latches the cipher data with respect to the enable signal.

2.3.1 Design of Individual Modules of AES IP Core

There are mainly around eight sub modules in it and among those four major transformation modules are there in this Core. They are of the following.

a. AES Top Level Module
b. AES State Machine Controller
c. Adder Key Module
d. Loading Key access Module
e. Loading Bytes into the S-Box Module
f. Bytes Sub by the S-Box Module
g. Shifting Rows Module
h. Mixing Columns Module

The Main building blocks of AES-256 IP CORE on SOC are of the following kind, which are developed by the RTL Code of VHDL making as synthesizable code.

2.3.2 Top Level AES IP-Core Module

The AES-256 IP Core module consists of the power reset, load key, microcontroller address, data in microcontroller, read write signals of the microcontroller, enable clock, these are the input signals for the aes_core and ciper_out is the output signal of this core. The encryption to be done, when the encryption clock and power reset is low is activated.

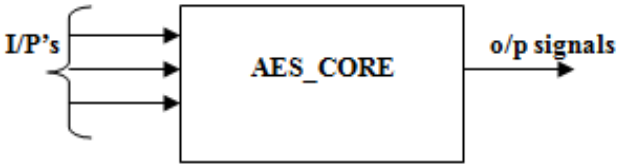

Figure 5: Top Level of AES IP Core

2.3.3AES State Machine Controller Module

The AES State Machine Controller module performs the transformations of the AES encryption algorithm, such as adder, substitution, shifting, mixing, key controls depending on the control signals with respect to the encryption clock to synchronize with the tasks and functions performed by the module.

2.3.4Adder Key Module

This module loads or stores the AES-256 Key values into it to perform the initial key addition for the key schedule generation.

2.3.5Loading Key access Module

This module loads or stores the values of the predefined look up table into it for the substitution transformation after initial key addition process.

2.3.6Loading Bytes into the S-Box Module

This module loads or stores the values of the predefined look up table into it for the substitution transformation after initial key addition process.

2.3.7AES Shifting Rows Transformation Module

This module performs the transformation by rotating left shift operation shifts the bytes accordingly for further process.

2.3.8 AES Mixing Columns' Transformation Module

This module performs the transformation by mixing columns with a fixed polynomial to ensure irreducible polynomial of degree less than GF 2 power 8.

2.4 AXI Interconnect

Getting familiar with zynq design flow we need the Knowledge of AXI Good knowledge on AXI makes Fast and efficient development with Vivado

AXI is part of ARM AMBA, a family of micro controller buses first introduced in 1996. The first version of AXI was first included in AMBA 3.0, released in 2003. AMBA 4.0, released in 2010, includes the second version of AXI, AXI4 [3].

There are three types of AXI4 interfaces:

- AXI4 - for high-performance memory-mapped requirements.
- AXI4-Lite - for simple, low-throughput memory-mapped communication (for Example, to and from control and status registers).
- AXI4-Stream - for high-speed streaming data.

Working of AXI Interconnect

Both AXI4 and AXI4-Lite interfaces consist of five different channels:

- Read Address Channel
- Write Address Channel
- Read Data Channel
- Write Data Channel
- Write Response Channel

Data can move in both directions between the masters and slave simultaneously, and data transfer sizes can vary. The limit in AXI4 is a burst transaction of up to 256 data transfers.AXI4-Lite allows only 1 data transfer per transaction [3].

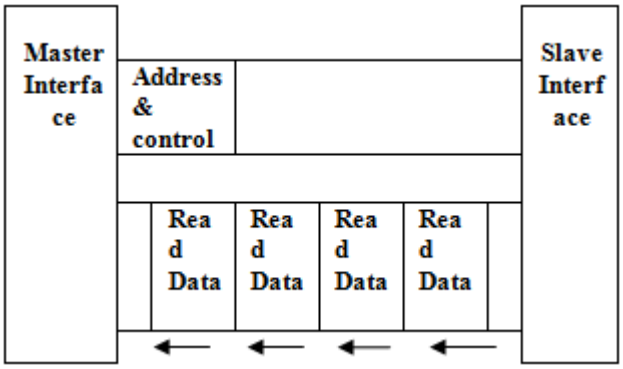

Figure 6: Channel Architecture of Reads

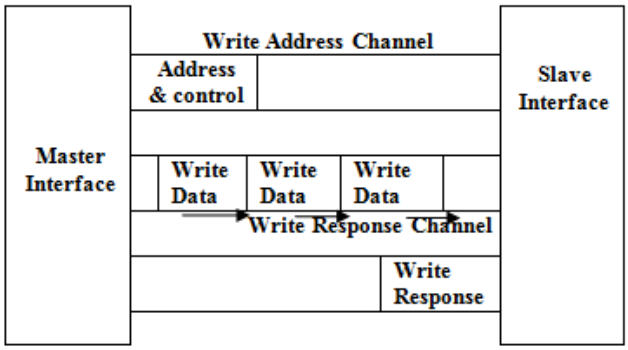

Figure 7: Channel Architecture of write

2.5 Bram Controller

The LogiCORE IP AXI Block RAM (BRAM) Controller is a soft IP core for use with the Xilinx Vivado Design Suite, Embedded Development Kit (EDK), and ISE Design Suite. The core is designed as an AXI Endpoint slave IP for integration with the AXI interconnect and system master devices to communicate to local block RAM. The core supports both single and burst transactions to the block RAM and is optimized for performance [13].

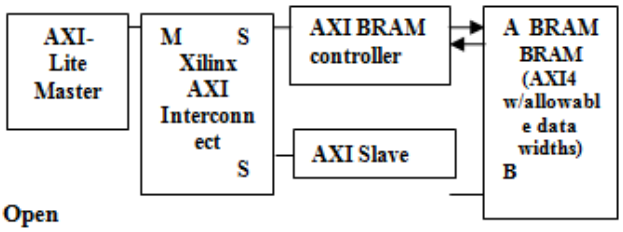

Figure 8: AXI configuration of single port BRAM controller

2.6 Vivado Design Suite

Vivado is the tool suite for Xilinx FPGA design and includes capability for embedded system design.

- IP Integrator is part of Vivado and allows block level design of the hardware part of an embedded system Integrated into Vivado.
- Vivado includes all the tools, IP, and documentation that are required for designing systems with the Zynq-7000 AP SoC hard core and Xilinx soft core processor.
- Vivado + IPI replaces ISE/EDK.

SDK is an Eclipse-based software design environment. Enables the integration of hardware and software components, it Links from Vivado.

Vivado is the overall project manager and is used for developing non-embedded hardware and instantiating embedded systems [11].

2.7 Vivado Components

Vivado/IP Integrator

Design environment for configuration of PS, and hardware design for PL.

- Hardware Platform (xml)
- Platform, software, and peripheral simulation
- Vivado logic analyzer integration

Software Development Kit (SDK)

- Project workspace
- Hardware platform definition
- Board Support Package (BSP)
- Software application
- Software debugging

3. Results and Discussion

3.1 IP Integration

The following chapter explains about Vivado software design suite and simulation results of AES. The software implementation is done on the VIVADO Design suit. Firstly Zynq device is configured as per given requirements and a hardware platform of VIVADO tool generates ARM IP cores (PS hardware). The defined IP core is also added to the tool generated hardware to meet the specified requirements. When IP is added then corresponding HDL code is generated automatically. After then required pins of different IPs interconnected and routed to I/Os. The hardware configuration information is exported to the SDK (software development kit).

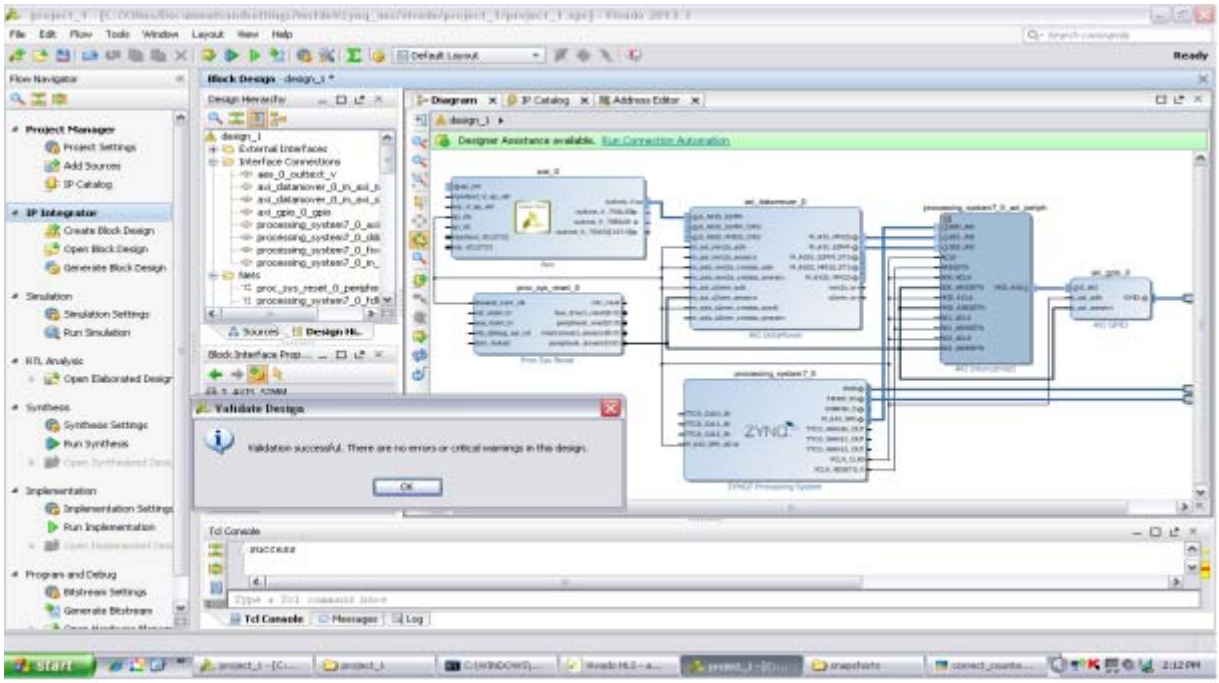

Figure 9: IP Integration

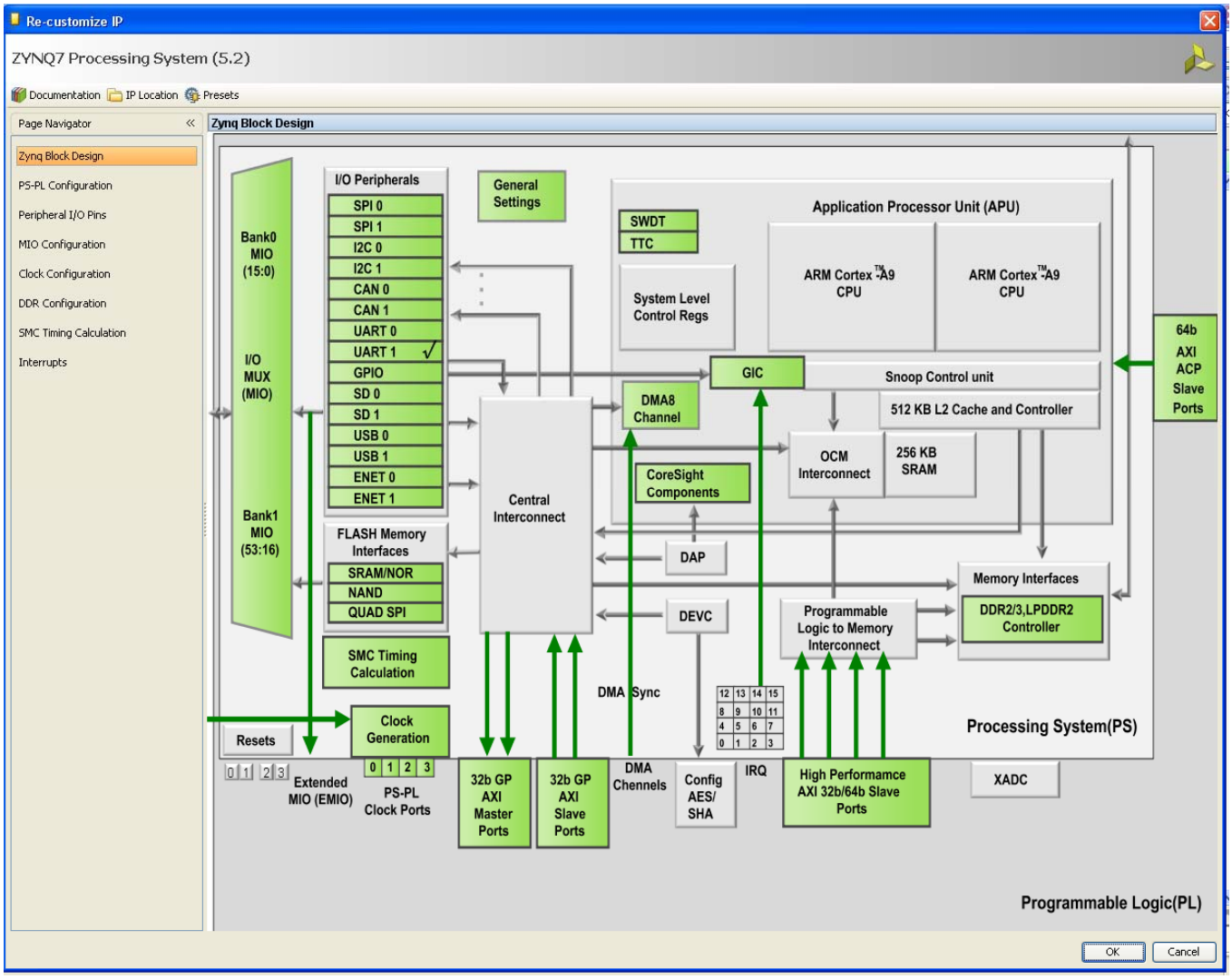

Figure 10: FPGA Interface

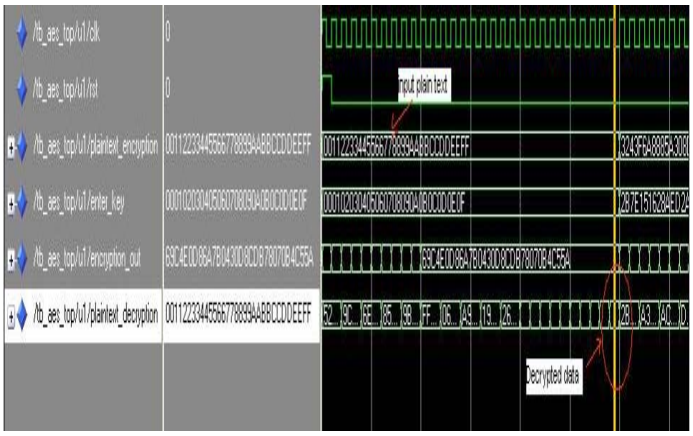

Figure 11: AES Simulation Results

4. Conclusion

Secured data transmission systems on Zynq SoC are presented in this paper. It consists of several different modules. Each of these modules are, implemented and tested on different development platforms. This hardware implementation is relatively simple and easily integrated to any platform due to the independence to Vendor specific macros. The advanced encryption standard (AES) algorithms are used. Thus it achieves high speed at low area cost. Algorithms and implementations are presented in above figures. In present systems Telemetry systems are occupies more space, size and weight. Using zynq Soc it occupies less space and weight.

References

[1] Telemetry System Engineer .pdf
[2] Xilinx Inc., UG865 (v1.2), "Zynq-7000 All Programmable SoC Packaging and Pinout", February 14, 2013
[3] Xilinx AXI Reference Guide (UG761)
[4] J. Nechvatal, et. al., Report on the Development of the Advanced Encryption Standard(AES), National Institute of Standards and Technology, October 2, 2000, available at[1].
[5] AES implementation of Fips-197 .pdf
[6] A. Lee, NIST Special Publication 800-21, Guideline for Implementing Cryptography in the Federal Government, National Institute of Standards and Technology, November 1999

[7] AXI Reference Guide, UG761 January 18, 2012, http://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf
[8] Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques, http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_3/ug873-zynq-ctt.pdf
[9] Xilinx Inc., UG585 (v1.6), "Zynq-7000 All Programmable SoC Technical Reference Manual", June 28, 2013
[10] Reference Designs/Tutorials of Zed board, http://www.zedboard.org
[11] Ug937-Vivado–design – suite – simulation-tutorial .pdf
[12] Ug939– Vivado-designing with IP tutorial .pdf
[13] Ds777_axi_bram_ctrl.pdf