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

# Implementation of AES Algorithm in a Microblaze Processor Using System C

**Rudo Duri[1], T. Madhavi Kumari[2]**

[1]MTech, ECE Department, Jawaharlal Nehru Technological University, Hyderabad, India

[2]Associate Professor, ECE Department, Jawaharlal Nehru Technological University, Hyderabad, India

**Abstract:** ***This research investigates the Advanced Encryption Standard (AES) encryption and decryption algorithm with regard to 256- bit message length and 192- bit key length. In Spartan3 EDK we implemented the AES algorithm through pipelined architecture through the soft core processor, the Microblaze. Xilinx XC3S200 device of the Spartan family of the FPGA is used for hardware evaluation. The code is translated, mapped, placed and routed in Spartan 3 EDK using Xilinx Platform Studio (XPS). The microblaze processor is a RISC machine which is highly reconfigurable, uses 5-stage pipeline and has a 32-bit instruction word. By using system C coding the implementation makes it very low complexity architecture, that is, in saving the hardware resources. This implementation is most suited for hardware critical applications.***

**Keywords:** AES encryption, decryption, XPS, microblaze processor, system C, FPGA, Spartan.

## 1. Introduction

For protection and security of sensitive information, encryption is becoming more and more important nowadays. Security of encrypted data depends on two things, the strength of the cryptographic key and the secrecy of the key while transmitting over a channel. AES offers greatest security to the sensitive data compared to other cryptographic algorithms. The AES was accepted as a standard in November 2001 [1]. NIST sought to "consider alternatives that offer a higher level of security" than that offered by the Data Encryption Standard (DES), [2], [3] which grew vulnerable to brute-force attacks due to its 56-bit effective key length. AES algorithm uses a varying cipher key size of 128, 192 and 256-bits to encrypt and decrypt data in blocks of 128-bits [4], [5]. Hardware based cryptography is considered superior to software based if implemented in a secure manner [6]. FPGAs provide a cost effective approach to building hardware accelerated computing platforms because they require lower development efforts and incur marginal nonrecurring engineering costs. The FPGA technology has much greater potential for providing higher security level because of its capability for dynamic reconfiguration [10], [12]. The FPGA uses highly pipelined architecture [13], and they are designed to overcome the limitations of existing micro processors and DSP processors.

The paper is organized as follows; the description of AES encryption algorithm is discussed in section 2, followed by the decryption algorithm in section 3. Implementation is discussed in section 4, then results and conclusion in section 5 and 6 respectively.

## 1. AES Encryption

Advanced Encryption Standard (AES) algorithm is a symmetric block cipher that encrypts and decrypts a 128-bit block message and a varying key size of 128, 192 and 256 bits. Encryption is a process of translating data into a secret code called ciphertext and decryption is converting ciphertext back to its readable format. The AES process consists of four stages that can be iterated a number of times depending on the size of key being used. For a key size of 128-bit, 10 rounds are required, 12 rounds for a 192-bit key and 14 rounds for a 256-bit key. These four stages are subBytes, shiftRows, mixColumns and addRoundKey. During encryption and decryption process the mixcolumn and invMixColumn processes are suppressed in the last rounds. If we take for instance, when using the AES-192, in the $12^{th}$ round the mixcolumn and the invmixcolumn will be missing in the encryption and decryption process respectively.

The first three functions of an AES round are designed to thwart cryptanalysis via the methods of "confusion" and "diffusion." The fourth function actually encrypts the data. Claude Shannon described the concepts of confusion and diffusion in his seminal 1949 paper, "Communication Theory of Secrecy Systems:"

"Two methods suggest themselves for frustrating a statistical analysis. These we may call the methods of *diffusion* and *confusion*."

**Confusion** is making the relationship between the ciphertext and the key as complex and involving as possible and **diffusion** refers to dissipating the structure of the plaintext over bulk of the ciphertext

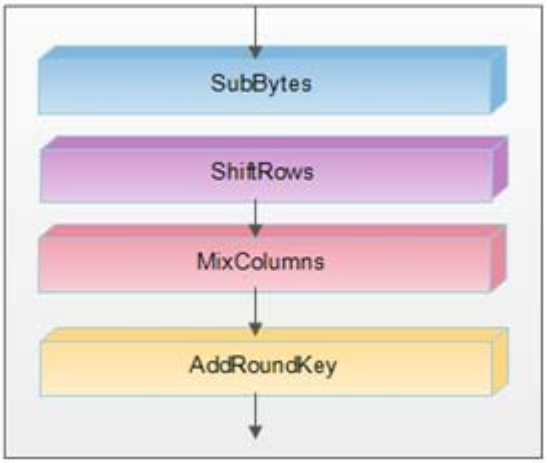



**Figure 2.1**: Stages of the AES process

### 2.1. SubBytes

The subbytes add confusion by processing each byte through an s-box. An s-box is a substitution table where one byte is substituted for another based on a substitution algorithm.

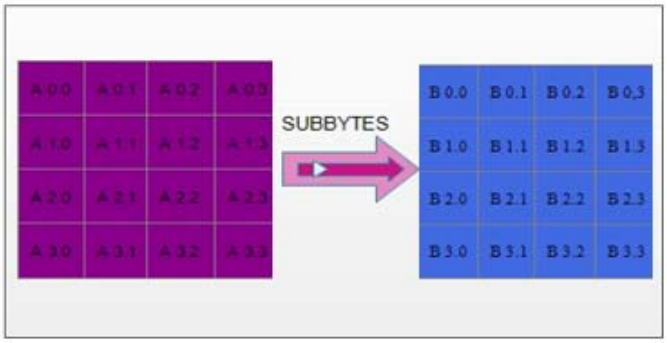



**Figure 2.2:** Subbytes process

### 2.2. ShiftRows

This is a simple shifting operation, it provides diffusion by mixing data within rows. The 1st row remain unchanged, the 2nd row is circularly left shifted by one byte, the 3rd row shifted by two bytes and the 4th row shifted by three bytes.

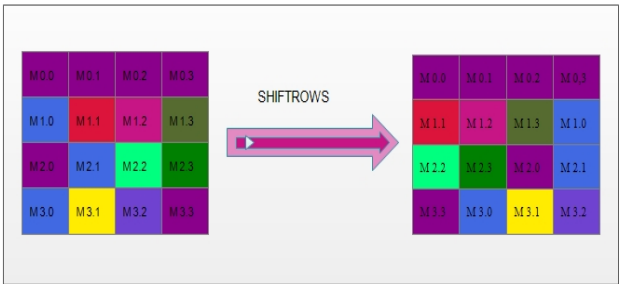



**Figure 2.3:** Shiftrows process

### 2.3. MixColumn

This also provides diffusion by mixing data within columns. The 4-bytes of each column in the message bits are treated as a 4-byte number and transformed to another 4-byte number through finite field mathematics.

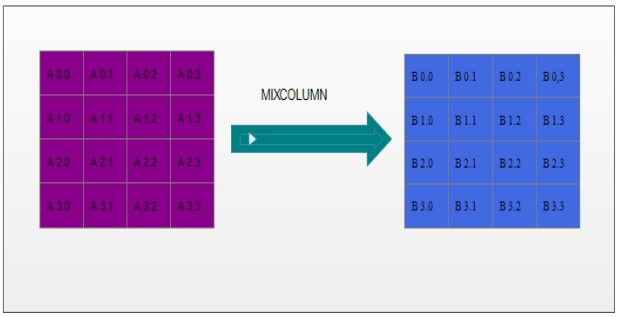



**Figure 2.4:** Mixcolumn process

### 2.4. AddRoundKey

A bitwise XOR of the block with the expanded key. The subkey is derived from the key according to the key expansion schedule shown in figure 2.6. The addRoundKey is the only stage that makes use of the key. The other three stages together provide confusion and diffusion but they do not provide security because they do not use the key.

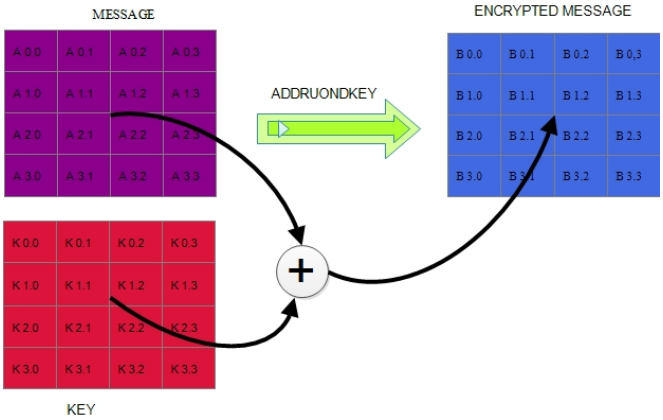



**Figure 2.5:** Addroundkey process

### 2.5. Key Expansion

As the key size defines the number of rounds in the encryption and decryption algorithm it also defines its expansion process. The process consists of three operation, that is, RotWord, SubWord and Xor operation. RotWord makes a one byte circular left shift on the word. SubWord performs a byte to byte substitution according to s-box. The last operation is an Xor operation.

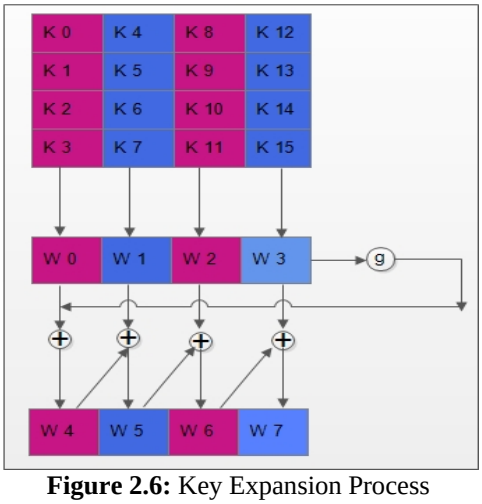



**Figure 2.6:** Key Expansion Process

## 2. Decryption

The decryption process makes use of the inverse of the encryption process stages, that is, the **invSubBytes, invShiftRow**, **invMixColumn** and the **AddRoundKey**.

AddRoundKey does not require an inverse function, as it XORs the state with the sub key (XOR encrypts when applied once, and decrypts when applied again).

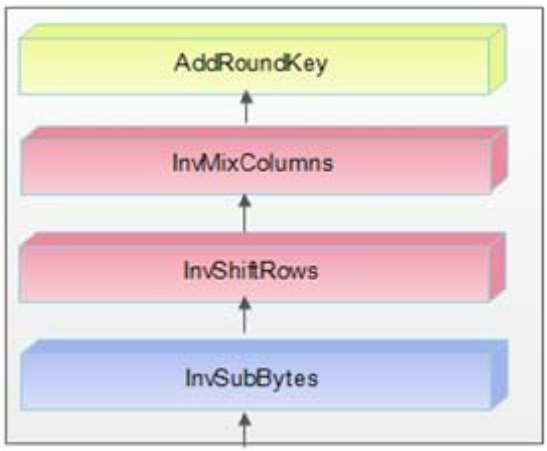



**Figure 2.6:** The decryption process

## 3. Implementation

A field-programmable gate array **(FPGA)** is a reprogrammable semi-conductor device designed to be configured by a customer or a designer after manufacturing. FPGAs are completely reconfigurable and they provide hardware timed speed and reliability. FPGAs also offer more speed in execution process.

**Table 4.1:** Configuration of FPGA

| Property Name | Value |
|---|---|
| Device | XC3S200 |
| Package | TQG144 |
| Speed Grade | -4 |

### 4.1. System C coding

System C coding is an open $C^{++}$ class library used for hardware system level modeling, design and verification. System C can be used to describe a system at several level of abstraction. Systems C model consists of communicating processes and in this respect it is the same as many other modeling languages, including VHDL and verilog. System C allows models of computation to be mixed in the same model. The model can be defined for the hardware down to the RTL and it is possible to make a synthesis of the system C code for FPGA development.

### 4.2. Xilinx Platform Studio

The Xilinx Platform Studio (XPS) is the development atmosphere or user interface used for planning the hardware portion of your embedded processor system. Xilinx Embedded Development Kit (EDK) is associated integrated software system tool suite for developing embedded systems with Xilinx MicroBlaze and PowerPC CPUs. EDK includes a spread of tools associated applications to help the designer to develop associated embedded system right from the hardware creation to final implementation of the system on an FPGA. System style consists of the creation of the hardware and software system, parts of the embedded processor system and also the creation of a verification element is elective. A typical embedded system style project involves: hardware platform creation, hardware platform verification (simulation), software system platform creation, software system application creation, and software system verification. Base System Builder is that the wizard that's want to mechanically generate a hardware platform in keeping with the user specifications that's defined by the MHS (Microprocessor Hardware Specification) file. The MHS file defines the system design, peripherals and embedded processors. The Platform Generation tool creates the hardware platform mistreats the MHS file as input.

The software system platform is defined by MSS (Microprocessor software system Specification) file that defines driver and library customization parameters for peripherals, processor customization parameters, customary one hundred and ten devices, interrupt handler routines, and different software system connected routines. The MSS file is associate input to the Library Generator tool for personalization of drivers, libraries and interrupts handlers.

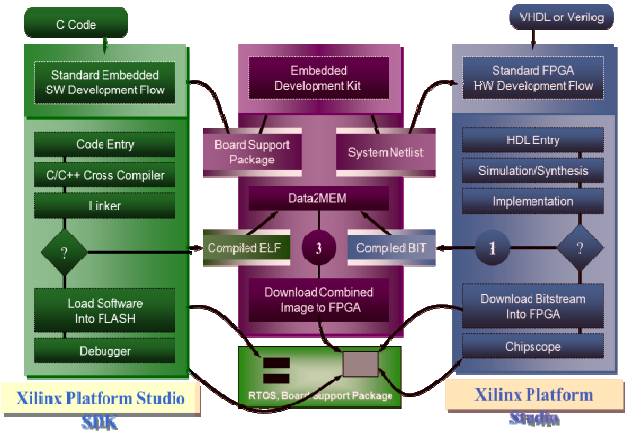



**Figure 4.1:** Embedded Development Kit Design Flow

The creation of the verification platform is facultative and is predicated on the hardware platform. The MHS file is taken as Associate in Nursing input by the Siegen tool to make simulation files for a particular machine. 3 varieties of simulation models will be generated by the Siegen tool: behavioral, structural and temporal arrangement models. Another helpful tool on the market in EDK is Platform Studio that provides the GUI for making the MHS and MSS files produce or import IP wizard that permits the creation of the designer's own peripheral and imports them into EDK. Platform Generator customizes and generates the processor system within the sort of hardware net lists. Library Generator tool configures libraries, device drivers, file systems and interrupt handlers for embedded processor system. Bit stream initializer tool, initializes the instruction memory of processors on the FPGA. Antelope Compiler tools are used for collection and linking application executable for every processor within the system [6]. There

are 2 choices on the market for debugging the appliance created victimization EDK namely: Xilinx microchip correct (XMD) for debugging the appliance package employing a microchip correct Module (MDM) within the embedded processor system, and package programmer that invokes the package programmer appreciate the compiler getting used for the processor. C. package Development Kit Xilinx Platform Studio package Development Kit (SDK) is Associate in Nursing integrated development atmosphere, complimentary to XPS, that's used for C/C++ embedded package application creation and verification. SDK is made on the Eclipse open source framework. Soft Development Kit (SDK) may be a suite of tools that allows you to style a package application for elite Soft IP Cores within the Xilinx Embedded Development Kit (EDK).The package application will be written during a "C or C++" then the entire embedded processor system for user application are completed, else correct and download the bit file into FPGA. Then FPGA behaves like a processor implemented on it in a Xilinx Field Programmable Gate Array (FPGA) device.

### 4.3. The Microblaze Processor

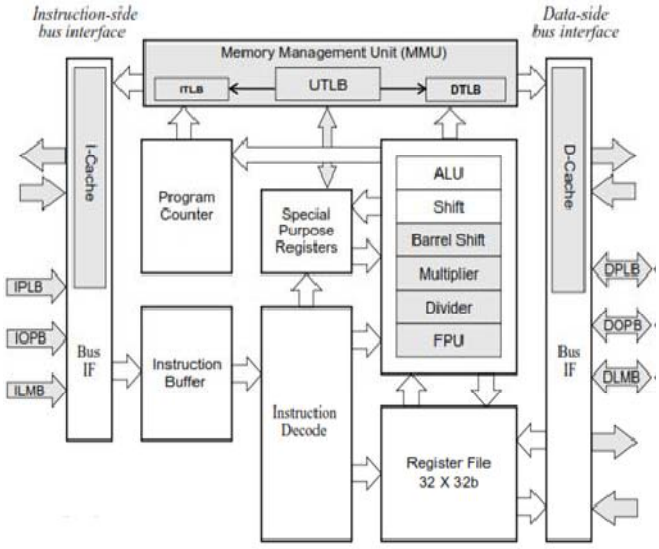



**Figure 4.2:** The microblaze processor core block diagram

The Micro Blaze soft core processor is highly configurable and the feature set of the Processor includes:

- Five Stage Pipeline
- 32-bit general purpose registers
- 32-bit instruction word with three operands and two addressing modes,
- 32-bit address bus and Single issue pipeline

## 4. Results and Analysis

Hardware implementation was through system C coding which we synthesized, mapped placed and routed using XPS.

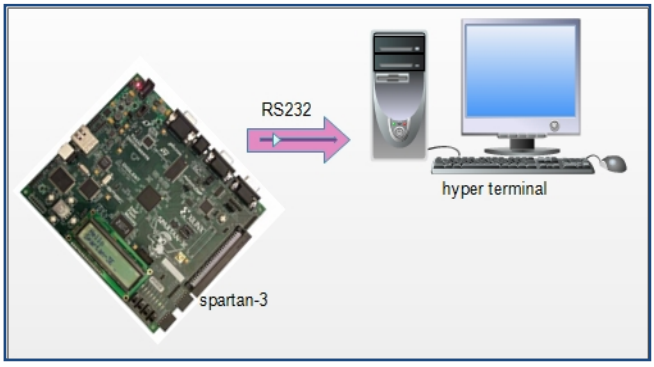



**Figure 5.1:** Overview of the implementation set-up

The synthesis report shows that we can implement this AES encryption and decryption in the microblaze processor as seen from figure 5.1.

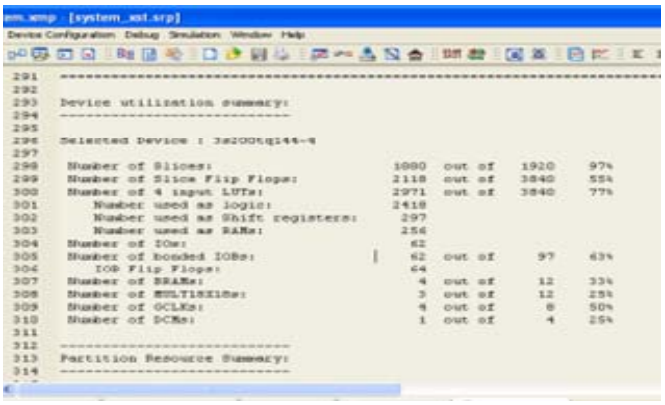
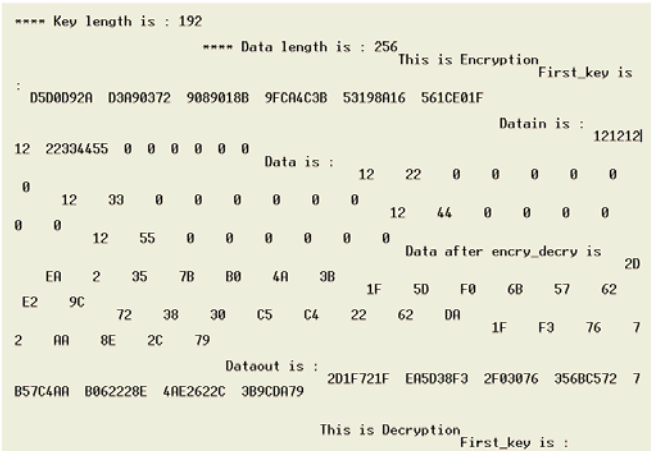
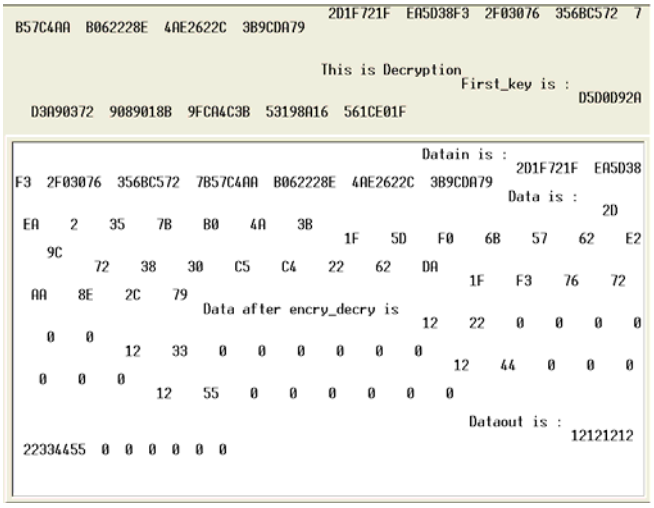


**Figure 5.2**: Synthesis Report

The data was then sent to the personal computer (PC) through the HyperTerminal and the results were obtained as shown in the figures below.




(a)




(b)

**Figure 5.3: (a)** encryption results and **(b)** decryption results

## 5. Conclusion

We have successfully implemented the AES encryption and decryption algorithm in the FPGA as shown from the results in figure 5.3 (a) and (b). The system C code was developed for the implementation of the encryption and decryption process. The microblaze processor uses a 5-stage pipeline which gives a high speed implementation of the AES algorithm. The synthesis report shows that space consumption is low, this permits the implementation of this method over inexpensive FPGAs.

## 6. Future Scope

I propose that this work be used as part of larger projects, including protecting sensitive data in the military and in the banks. This system can also be adopted in data terminal equipment with less demand on throughput.

## References

[1] FIPS FIPS-197, Federal Information Processing Standards Publication FIPS-197,Advanced Encryption Standard(AES),http://csrc.nist.gov/publications/fips/fips 197/fips-197.pdf, 1999.
[2] Daemen, J. and Rijmen, V., The design of Rijndael: AES — The Advanced Encryption Standard. Springer-Verlag, 2002.
[3] SCHNEIER, B., Applied Cryptography: Protocols, Algorithms and Source Code in C. John Wiley & Sons, Inc. 2nd Ed, 1996.
[4] William Stallings, Cryptography and Network Security, Principles and Practices, 4th ed. Pearson Education, pp. 134-161, 2006
[5] [5] Charlie Kaufman, Radia Perlman, Mike Speciner, Network Security, Private Communication in a Public World, 2nd ed. Pearson Education, pp. 41-114, 2006
[6] *on Circuits and Systems(LASCAS-2011),* February 201[6] Gomes, O. S. M.; Pimenta, T. C.; Moreno, R. l., "a highly efficient FPGA Implementation", *2nd Latin America Symposium* 1.
[7] Daemen, J. and Rijmen, V. A Specification for The AES Algorithm. NIST (National Institute of Standards and Technology).http://csrc.nist.gov/archive/aes/rijndael/wsd index.html, 2010.
[8] Ahmad, N.; Hasan, R.; Jubadi, W.M; "Design of AES S-Box using combinational logic optimization", IEEE Symposium on Industrial Electronics & Applications (ISIEA), pp. 696-699, 2010.
[9] J. V. Dyken and J. G. Delgado-Frias, "FPGA schemes for minimizing the power-throughput trade-off in executing the Advanced Encryption Standard algorithm" School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164-2752, USA, Available online 16 December 2009 .
[10] K. Joarvinen, "Study on high-speed hardware implementation of cryptographic algorithms" Department of Signal processing and Acoustics, Helsinki of technology, 10 University Feb 2009.
[11] A. Hodjat and I. Verbauwhede, "A 21.54 Gbits/s Fully Pipelined AES Processor on FPGA", 12th IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2004), pages 308-309, IEEE Computer Society, 2005.
[12] Standaert et al, "Efficient Implementation of Rijndael Encryption in Reconfigurable Hardware: Improvements and Design Tradeoffs. ches 2003, LNCS 2779, pp. 334-350, 2003.
[13] Saggese et al, "An FPGA-Based Performance Analysis of the Unrolling, Tiling, and Pipelining of the AES Algorithm", FPL 2003, LNCS 2778, pp. 292-302, 2003.
[14] Jarvinen et al, "A fully pipelined memory less 17.8 Gbps AES-128 encryptor", International Symposium on Field Programmable Gate Arrays, pp. 207-215. 2003.

## Author Profile

**Rudo Duri** attained her B. Tech. Degree in ECE from the Harare Institute of Technology 2010, Zimbabwe. Currently she is studying towards M. Tech Digital Systems and Computer Electronics at JNTUH, India.. She is a Harare Institute of Technology staff development research fellow. Her research interests are in the area of Microcontroller Design, ADHoc and Wireless Sensor networks, VLSI Design, Advanced Data Communications, Real Time Operating Systems and Digital System Design.

**Mrs.Thoomati Madhavi Kumari** obtained B.Tech. in ECE and M.Tech. in Computer Science from JNT University. Presently Mrs. Madhavi is pursuing Ph.D. in Data security for natural languages using FPGA. Mrs. Madhavi joined the faculty of ECE Department of JNU College of Engineering, Kukatpally, Hyderabad as Assistant Professor and served the department for 13 years. Later she was appointed as Assistant Director, UGC-ASC, JNT University, and Hyderabad. She was associated with the implementation of AICTE sponsored project on computer networks.