American Journal of Engineering Research (AJER) 2015
American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-4, Issue-8, pp-212-215
www.ajer.org
Research Paper Open Access

Conversion of Number Systems using Xilinx.

Chinmay V. Deshpande[1], Prof. Chankya K. Jha[2].
[1,2](Electronics & Tele-communication Department,
Sahyadri Valley College of Engineering & Technology, S. P. Pune University, India)

ABSTRACT: *There are different types of number systems. Binary number system, octal number system, decimal number system and hexadecimal number system. This paper demonstrates conversion of hexadecimal to binary number using Xilinx software.*

KEYWORDS – *Analyzer, IEEE, Verilog, VHDL, Xilinx*.

I. INTRODUCTION

Numbers are used to represent whole integers for example 1, 2, 3, 11 and so on. In our daily life we use decimal number system for carrying out functions like addition, subtraction, multiplication and division.
Other types of number systems used in computers and calculators are:

1] Binary number system: This system is having only two digits, binary logic 0 and logic 1. Thus, base of this number system is 2.

2] Octal number system: This system is having digits 0 through 7. E.g. 0, 1, 2, 3, 4, 5, 6, 7, 12, 23, 34, 45, 56, 67 are octal numbers. These numbers are constructed using only 0, 1, 2, 3, 4, 5, 6, 7 integers. Thus base of the system is 8. These numbers are represented as: $(1234)_8$, $(5672)_8$, $(4563)_8$.

3] Decimal number system: Decimal number system is having integers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Thus base of this number system is 10. These numbers are represented as follows: $(123)_{10}$, $(904)_{10}$, $(456)_{10}$, $(786)_{10}$. We always consider base as 10 for writing numbers in system thus these numbers can also be written simply as: 123, 904, 456, 786 and so on.

4] Hexadecimal number system: Hexadecimal number system is having numbers from 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. This number system is used for microprocessor and micro-controller as well as ASICs programming as assembly level language. Instruction's address is given in hexadecimal number system using this language. E.g. 10A0, 0010, 001E, 001F etc.

II. CONVERSION OF NUMBERS.

1. Octal to binary number conversion: Octal numbers are having base or radix 8. These numbers are represented as: $(345)_8$, $(756)_8$ etc. For conversion of octal numbers to binary number, binary equivalent of octal number is grouped into three binary bits.

E.g. $(3)_8 = (011)_2$, $(4)_8 = (100)_2$.

2. Decimal to binary conversion: For conversion of decimal numbers to binary numbers divide each decimal number by 2 and write remainder quotient and remainder outside.
For example, take $(10)_{10}$:

TABLE 1: CONVERSION OF DECIMAL NUMBER TO BINARY NUMBER.

Divider	Number/Quotient	Remainder
	10	
2	5	0
2	2	1
2	1	0
2	0	1

Arrange remainder from bottom to top for getting binary equivalent of decimal number.

3. Hexadecimal to binary conversion: Hexadecimal numbers are 0 to 9 and A, B, C, D, E, F. The base of this number system is 16. These numbers are represented as follows: $(1AFE)_{16}$, $(1ACE)_{16}$ etc.
For conversion of hexadecimal numbers to binary numbers, make group of 4 binary equivalent bits of hexadecimal number.
For example: $(0)_{16} = (0000)_2$, $(1)_{16} = (0001)_2$, $(2)_{16} = (0010)_2$, $(3)_{16} = (0011)_2$, $(A)_{16} = (1010)_2$, $(B)_{16} = (1011)_2$, $(C)_{16} = (1100)_2$, $(D)_{16} = (1101)_2$, $(E)_{16} = (1110)_2$, $(F)_{16} = (1111)_2$.
Decimal equivalent of hexadecimal alphabets A = 10, B = 11, C = 12, D = 13, E = 14 and F = 15.

4. Decimal to Octal number system: For conversion of decimal numbers to octal number divide decimal number by 8. Remainders will give octal equivalent of decimal numbers.
For example: $(389)_{10} = (?)_8$

TABLE 2: CONVERSION OF DECIMAL NUMBER TO OCTAL NUMBER.

Divider	Number/Quotient	Remainder
	389	
8	48	5 ↑
8	48	0 |
8	0	6 |

Arrange remainder from down to up for getting octal equivalent of number.
Thus, $(389)_{10} = (605)_8$.

5. Hexadecimal to decimal system: For conversion of Hexadecimal number to decimal number convert hexadecimal number into binary equivalent by grouping into four bits first then multiply the binary bits stream by equivalent of 2^n position of bits.

For example: $(BCD)_{16} = (?)_{10}$
$(BCD)_{16} = (1011\ 1100\ 1101)_2 = (1\times2^{11} + 0\times2^{10} + 1\times2^9 + 1\times2^8 + 1\times2^7 + 1\times2^6 + 0\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0)_{10} = (3021)_{10}$.

6. Hexadecimal to Octal number system: For conversion of hexadecimal numbers to octal number, first convert hexadecimal number to binary number then group binary bits into group of three bits for getting equivalent conversion of hexadecimal number.
E.g. $(A23F)_{16} = (1010\ 0010\ 0011\ 1111)_2 = (1010001000111111)_2 = (001\ 010\ 001\ 000\ 111\ 111)_2 = (1\ 2\ 1\ 0\ 7\ 7)_8 = (121077)_8$.
If in grouping of three binary bits there are less bits then add zeros to left side for completing grouping of three bits.

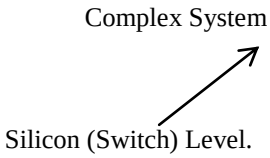
7. Hexadecimal to binary number system: For conversion of hexadecimal to binary number system convert each hexadecimal number into binary group of four bits. Resulting number is binary equivalent of hexadecimal number.
For example: $(A123)_{16} = (1010\ 0001\ 0010\ 0011)_2$.

III. LITERATURE SURVEY.

History of Verilog Language: Verilog HDL was also consisting of Verilog Simulator. Verilog-XL, a new version of simulator with the enhanced language and simulator was introduced in 1985.
1989:
- Cadence bought Gateway.

1990:
- In early 90's Cadence split Verilog HDL and Verilog-XL simulator into separate products.
- Verilog was then released to public domain.
- OVI (Open Verilog International) was formed to control the language specification.
  - 1993:
- Of all design submitted to ASIC foundries in this year, 85% were designed and submitted using Verilog.
  - 1995:
- The Verilog language was reviewed and adopted by the IEEE as IEEE standard 1364.
  - Verilog can handle all levels of design abstraction.

- Verilog HDL allows different levels of abstraction to be mixed in the same model.



IV. FEATURES OF VERILOG

Universality:
Verilog is universal. It allows the entire design process to be performed within one design environment.

Industrial Support:
Verilog supports switch level modeling, hence has always been popular with ASIC designers, as it allows fast simulation & effective synthesis.

Extensibility:
The IEEE standard 1364 contains definition of PLI (Programming Language Interface) that allows for extension of Verilog capabilities.

Similarity with C:
Syntax is similar to the C programming language. Hence C programmers find it easy to learn Verilog.
"VERILOG is case – sensitive language."

Levels of Abstraction:
Verilog is both, behavioral and structural language. Designs in Verilog can be described at all the four levels of abstraction depending on the needs of design.

Behavioral Level:
It is used to model behavior of design without concern for the hardware implementation details. Designing at this level is very similar to C programming.

Dataflow Level [Register Transfer Level: RTL]:
Module is specified by specifying the data flow. The designer is aware of how the data flows between registers.

Gate Level:
Module is implemented in terms of logic gates & interconnections between these gates.
Design at this level is similar to describing design in terms of gate level logical diagram.

Switch Level:
Lowest level of abstraction is provided by Verilog. Module can be implemented in terms of switches, storage nodes & interconnection between them.

V. SOFTWARE REQUIRED.

Xilinx ISE 9.1

For conversion of hexadecimal number to binary number ⊕ = XOR operation is used.
The conversion of plaintext into binary code is done using Xilinx software and programming is done in Verilog.

Xilinx ISE tools:
Xilinx have different tools for viewing snapshots. Snapshots are the outputs of RTL schematic and technology schematic.
Other tools are: Constraints editor, core generator, schematic viewer, timing analyzer FPGA editor, FPGA editor, XPower Analyzer, iMPACT, SmartXplorer and so on.

VI. RESULTS OF EXPERIMENT.

For converting hexadecimal text i.e. plaintext into cipher text i.e. binary output stream of bits two processes are required which are implementation process and simulation process. Simulation process is required to simulate

i.e. to provide inputs to Behavioral Model. IEEE 1532 programming is supported by iMPACT when provided by an .ISE file.

Register Transfer Level (RTL) schematic:

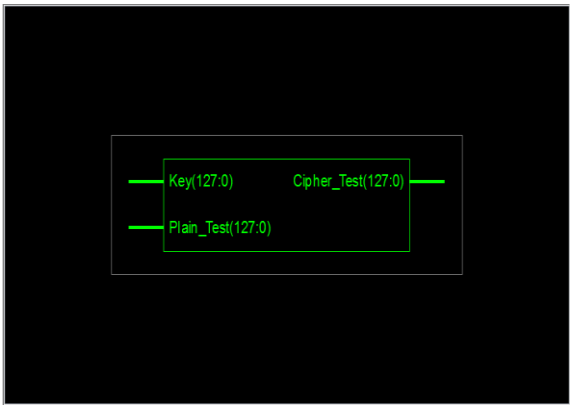


Figure 1: RTL Schematic.

Cipher output:

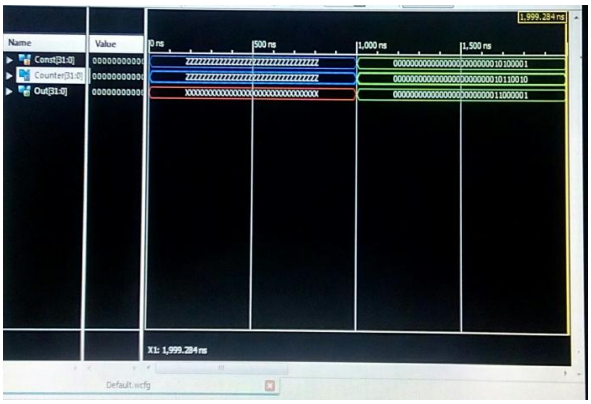


Figure 2: Binary output view.

VII. CONCLUSION.

In this project, for conversion of hexadecimal number given as input to cipher output i.e. a binary equivalent code of hexadecimal input Verilog programming is done using Xilinx ISE 9.1.
RTL schematic can be viewed by schematic viewer as shown in Figure 1. Binary output is given in Figure 2.

VIII. ACKNOWLEDGEMENTS

I place on record, my sincere thanks to all authors mention in references, my guide Prof. Chankya K. Jha and all my friends for their support.

REFERENCES

[1] J. Aumasson. On a bias of Rabbit, presented at the State of the Art of Stream Ciphers Workshop (SASC 2007).
[2] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christensen, and O. Scavenius, "Rabbit: A new high-performance stream cipher," LectureNotes in Computer Science, vol. 2887, pp. 307–329, 2003.
[3] D. J. Bernstein, "Which eSTREAM ciphers have been broken?" Dept. Math., Statistics Comput. Sci. (M/C 249), Univ. Illinois, Chicago, IL, USA, 2008.
[4] S. Fischer, W. Meier, C. Berbain, J. Biasse, and M. J. B. Robshaw, "Non-randomness in eSTREAM candidates Salsa20 and TSC-4," FHNW 5210 Windisch, Switzerland, FTRD, 92794 Issy les Moulineaux, France, 2008.
[5] P. Crowley, "Truncated differential cryptanalysis of five rounds of Salsa20," presented at IACR Cryptology ePrint Archive, pp.375-375, Sep. 2005.
[6] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki, and H. Nakashima, "Differential cryptanalysis of Salsa20/8," in *Proc. Workshop Rec. SASC 2007: StateArt Stream Ciphers*, eSTREAM Rep. 2007/010, 2007.
[7] J. Hernandez-Castro, J. Tapiador, and J. Quisquater, "On the Salsa20/8 core function," *Lecture Notes in Computer Science*, vol. 5086, pp. 462–469, Feb. 2008.
[8] Mohsen A. M. El-Bendary*a*, AtefAbou El-Azm*b*, Nawal El-Fishawy*b*, FaridShawki*b*,Mostafa A. R. El-Tokhy*a*, Fathi E. Abd El-Samie*b*, and H. B. Kazemian*c*: SVD Audio Watermarking: A Tool to Enhance the Security of Image Transmission over ZigBee Networks.
[9] S. Sadoudi, C. Tanougast, and M. S. Azzaz: First experimental solution for channel noise sensibility in digital chaotic communications.
[10] Vimalathithan R. and M. L. Valarmathi : Cryptanalysis of Simplified-AES using Particle Swarm Optimization.
[11] Xiaoling Huang: A new digital image encryption algorithm based on 4d chaotic system.

Chinmay V. Deshpande is pursuing his M. E. in Electronics and Tele-communication Department (VLSI and Embedded Systems) from S. P. Pune University, India. He graduated in Electronics and Tele-communication Engineering from Sant Gadge Baba Amravati University, India. His areas of interest are wireless sensor networks, micro-processor and micro-controller and VLSI technology.