International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

Design and Implementation of High Security Optical Shaft Encoder for an Artificial Limb using Xilinx on FPGA

Jaskirat Kaur[Å*]

[Å]Electronics Communication and Engineering, Eternal University, Baru Sahib, India

Accepted 15 Nov 2014, Available online 20 Dec 2014, **Vol.4, No.6 (Dec 2014)**

Abstract

FPGA technology is used to make logic systems and it is majorly based on Computer-Aided Design (CAD). In this paper, the main aim is to introduce and implement our own logic for high security optical shaft encoder for an artificial limb using Xilinx software on FPGA. FPGA is used to design prototype hardware before final implementation. This will reduce cost as well as time. Now a day, Security is the most important issue for rucks. Therefore, to reduce their problems of security we implement high security optical shaft encoder for an artificial limb by providing security login system using finite state machine.

Keywords: *VHDL (Very High Integrated circuits Hardware Description Language), FSM (Finite State Machine), OSE (Optical Shaft Encoder), FPGA (Field Programmable Gate array), Xilinx14.2*

1. Introduction

Now a day, Robots play very important role in the field of technology and with robotic world, security is the basic need. It is very important to provide security to avoid unauthorized access. Therefore, matter of security is very important issue in any technology devices like movement of robots or any information related to the robots as well as optical shaft encoder is used in artificial limbs for the comfort of human that have any type of disability. In this paper, introduction of optical shaft encoder, which is used to make artificial limb by using advanced version of Xilinx software on FPGA board.

2. VHDL

VHDL (Very High Speed Integrated Circuits HDL) is an IEEE (Institute of Electrical and Electronic Engineers) standard language, which is widely used for designing hardware systems. VHDL (Very High Speed Integrated Circuits HDL) is an IEEE (Institute of Electrical and Electronic Engineers) standard language, which is widely used for designing hardware systems. VHDL is abbreviation of VHSIC Hardware Description Language. The acronym VHSIC refers to the Very High Speed Integrated Circuit Programs. VHDL emerged out of the United State Government's (VHSIC) Very High Speed Integrated Circuits Program [Amy Poh Ai Ling *et al*.2011]. Hardware description language overcomes some of the limitations of algorithmic languages, where the referencing time is not required [M.Moris 2008]. A VHDL model is basically a text based description of the system. The different model can be created with same system at different level of abstraction. The model with particular level of abstraction represents the detailed information related with it.

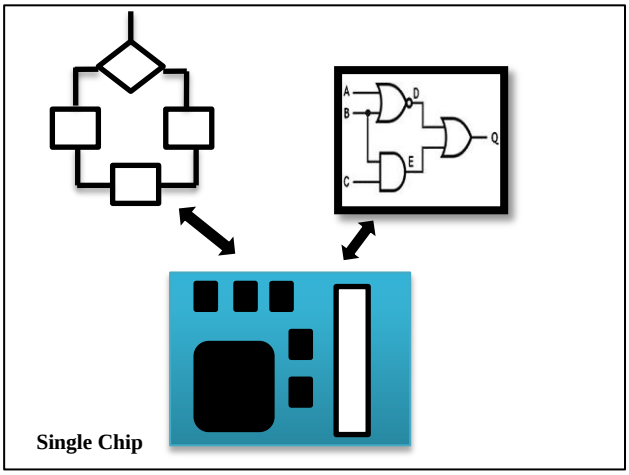

Figure No: 1 Multiple Levels of Abstraction of VHDL Language

The above diagram shows that VHDL language can be used to model software as well as hardware systems at multiple levels of abstraction. The language is independent of technology and methods of designs and styles, it promotes rapid prototyping. Methods of designing governed the application of these tools and flow of information between them. But as we know there are three types of different approaches to describing hardware named as Structural, Data Flow and Behavioral methods. Sometimes combination of three methods is also used as a

*Corresponding author **Jaskirat Kaur** is a M.Tech Student

mixed modeling. This can hinder interoperability. Hardware Description Language overcomes the limitations of algorithmic languages, where time is not important [Eric Simpson *et al*. 2006].

3. FSM

FSM (Finite State Machine) is the basic component in the designing of the hardware. Finite State Machine theory deals with the transition and behavior between input and output states for sequential circuits, which can be apply to any specific object. The specific object can be cite as a device that stores the status of something at a given time and can operate on input to change the status and/or cause an action or output to take place for any given change [Amr T. Abdel-Hamid *et al*, 2004]. FSM are also called as finite state automata or simply automata. A sequential system is also known as finite state machines. A generic approach allows any sequential systems to be designed. Generally, a sequential system consists of three parts that is named as registers, present state and the next state. Registers represent the states of the system. Both next state logic and output logic are entirely combinational logic. A simple block diagram for a state machine is shown in Figure No.2, which contains a Present State Logic section, Next State logic section and an Output Logic section. Present state logic section provides all the information about its past which helps to determine its current state and next state. The basic function of this is to assign the next state to the present state at the active clock edge. The present state is stored in a binary value in state register.

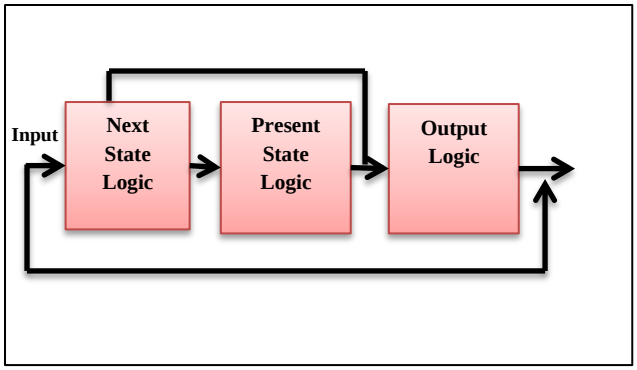

Figure No. 2 Block Diagram of simple FSM

Next state logic section computes the function to establish the next state of the system using FSM's inputs and present state. This section is implementing with sequential VHDL code with a process. Output Logic Section is used to generate the output of the system and this is implementing with concurrent VHDL code with conditional assignment statements [Jin, D *et al*2008]. FSM is one that has finite number of possible states. The maximum number of unique states that FSM have is 2^n. The numbers of states are also finite; hence, the name is finite state machine. Synchronous FSM and Asynchronous FSM are distinct concept or state of exchanging information between entities. Synchronous FSM having clock and its state can change value only at a triggering clock edge. Communication and computations occurs instantly at discrete time intervals. It may be difficult to design. However, in asynchronous FSM, there is no clock and its state can change value when its input value changes. They are free to proceed independently and do not execute a transition at same time. Implementation of asynchronous FSM is easy. They can operate faster and use less power as compared to synchronous FSM.

4. Optical Shaft Encoder

A rotatory encoder is known as Shaft encoder and the reason why it is called as optical is that it uses the light beams. Hence, it relates with optical and known as optical Shaft encoder. OSE is a device that coverts rotatory mechanical motion into a digital output. It includes two steps for conversion. Firstly, it converts the rotation of its shaft into interruptions of light beam. The second step is the conversion of light beam to electrical pulses.

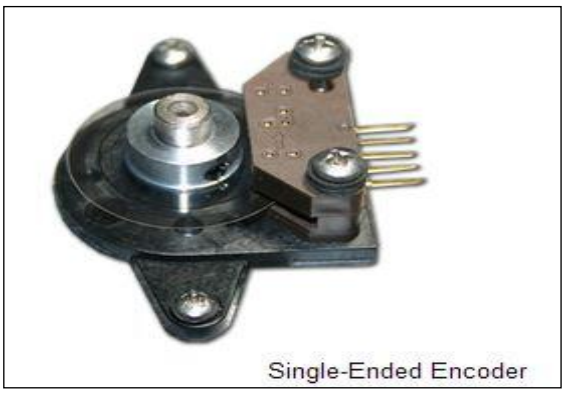

Figure No. 3 Single-Ended Shaft Encoder

The optical shaft encoder is used to measure both position as well as distance travelled by the shaft. It is also used to measure of the movement of any moving part. While rotating shaft, it produces two digital outputs waveforms. These output waveforms having a quadrature phase relationship with each other.

4.1. Principle of Optical Shaft Encoder

The optical encoder is a device, which provides pulsed signals in response to the movement. Continuous optical signals are coded with the use of a rotating disc with code patterns called tracks [Trabajo Fin De *et al*2012]. The pattern on the disc consists of an opaque and transparent segment.

The basic work of optical encoder is to convert the angular displacement into the digital form directly. Optical encoders operate by means of a grating, which moves between a light source and a detector. When light passes through the transparent areas of the grating, an output is seen from the detector. It has a shaft mechanically coupled to an input driver, which rotates a disc rigidly fixed to it. A clear segment is marked on the surface of the disc. Light from infrared emitting diodes reaches the infrared receivers through the transparent slits of rotating disc. An analog signal is created. Then electronically, the signal is

amplified and converts into digital form. Figure No.3 shows the operating principle of an optical Shaft encoder.

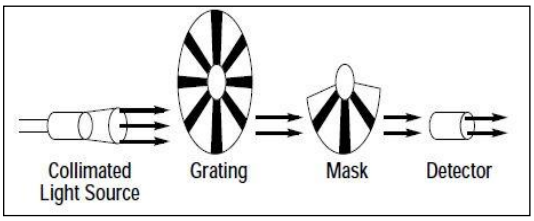

Figure No. 4 Operating principle of Optical Shaft Encoder

5. FPGA Field Programmable Gate Array

This type of device is based on Gate Arrays technology and is called as Field Programmable Gate Array. FPGA having the ability to reprogram that means the programmer can change their program according to their need and logic. It can be programmed repeatedly to correct the circuit. There is no large non-recurring engineering cost related to FPGAs. Many of programmers use this due to flexibility of FPGAs platform. FPGAs are generally the two dimensional array consists of AND array and OR array. There is an electrically programmable interconnection between logic blocks.

FPGA is more compact design because logic blocks are implemented using multiple level low fan in gates. It is easier as compared to an implementation with two-level AND-OR logic. Logic block of an FPGA can be configured in such a way that it can provide functionality as simple as that of transistor or as complex as that of microprocessor [Saroch.k 2012].

The reprogrammable devices such as field programmable gate arrays can easily do the hardware implementation of algorithms. Many of the software and traditional hardware can be support by FPGAs [Kaur, J. 2014 et.al]. The modern FPGAs are more capable to support technical complex products and the design, which is not possible before to implement. Many of the applications can be done using FPGAs.

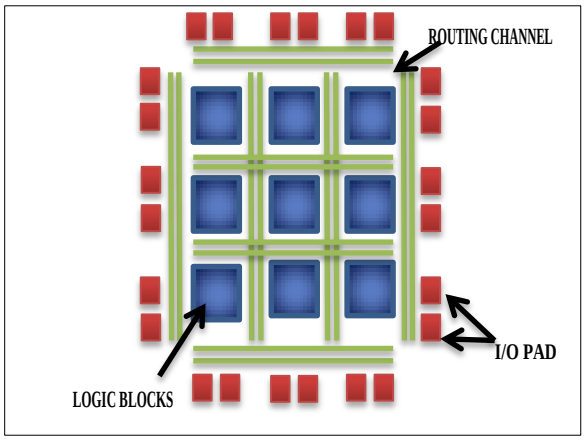

Figure No. 5 Basic Structure of FPGA

6. Programming Technology

Each FPGA depends on an underlying programming engineering that is utilized to control the programmable switches that give FPGAs their programmability. There are various programming innovations what's more; their disparities have a noteworthy impact on programmable logic architecture. The methodologies that have been utilized historically incorporate EPROM, EEPROM, flash, static memory, and anti-fuses. Of these methodologies, just the flash, static memory what's more anti-fuse methodologies are broadly utilized within cutting edge FPGA. This review centers fundamentally on static memory-based FPGA however, in this area, all these advanced programming innovations will be inspect to give a more finish understanding of the points of interest and detriments of static memory-based programming. There are three types of the programming technology, which is described below.

Static RAM Programming: Static memory cells are the fundamental cells used for SRAM based FPGAs. Mostly SRAM based programming technology is used in the devices. SRAM cells are used for the following purposes. 1) To program the routing interconnect of FPGAs which are generally raised by small multiplexers. 2) To program configuration logic blocks that are used to implement logic functions.

In SRAM programming, memory bit controls a switch that connects/ disconnects two wires. It has the main advantage of reprogrammed easily.

Flash Programming Technology: Flash Programming Technology is an alternative technology to SRAM based programming technology. It uses the EPROM/EEPROM technology for programming. Therefore, it is called as EPROM/EEPROM programming technology**.** It uses Non-standard CMOS process. Power consumption is high.

Anti-Fuse Programming Technology: Anti-fuse programming technology to SRAM and EPROM/EEPROM programming technology. The devices in the technology program interconnection routes permanently. The primary advantage of this technology is its low area. In this, fuses makes or break link between two wires.

7. Xilinx 14.2

Xilinx 14.2 is a software tool produced by Xilinx. Its main purpose of existence for synthesis and analysis of HDL design. It enable the developers to synthesize their designs performing timing analysis examine RTL diagrams simulate a design's reaction to different stimuli and configure the target device with the programmer. This edition is fully support the low cost Spartan family of FPGA and family of CPLDs. Xilinx is basically used for enter a design through schematic capture, perform circuit simulation, assign pin locations, implement the design, generate FPGA configuration data.

Xilinx 14.2 have feature of Xilinx Plan-ahead. The Plan-ahead is used for Pre-planning of the hardware used to implement the code in the authentic project.

Xilinx Plan-ahead is used for floor- planning for all pin planning and view of design. Xilinx have capabilities to design and analysis the wide range of Plan-ahead RTL to Bit-stream.

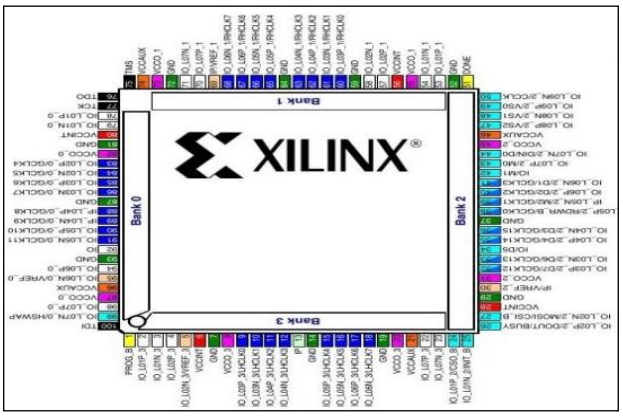

Figure No. 6 Block Diagram of Xilinx on FPGA board

8. Software Implementation

In this software, implementation of OSE for artificial limb is done successfully by using Xilinx software.

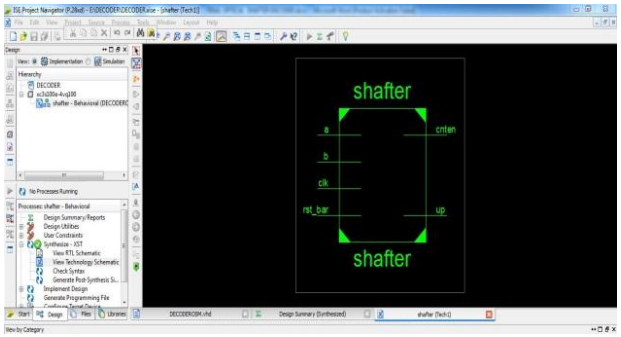

Figure No. 7 Technology Schematic of Optical Shaft Encoder

The Figure 7 shows that the technology schematic of optical shaft encoder. It shows the input and output used for Shafter. It is the schematic view of the project.

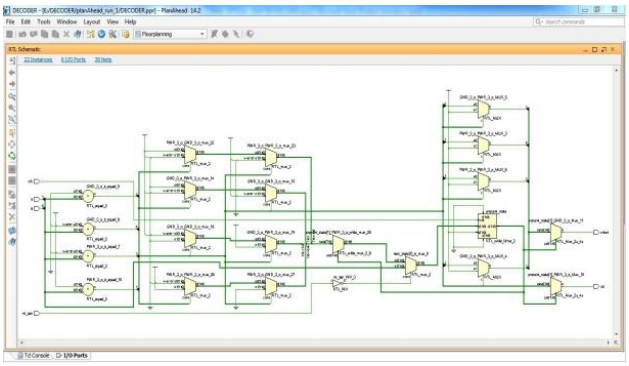

Figure No. 8 RTL Floor planning of Optical Shaft encoder

Figure 8 shows the floor planning of the Optical Shaft Encoder. It shows how register transistor logic is shown in the Xilinx, which is very helpful for planning the input and output.

The Figure 9 shows the planning of input and output with selected device used in optical Shaft Encoder. It shows the I/O planning with devices.

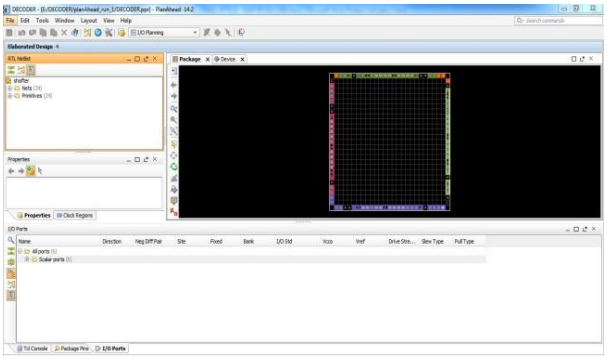

Figure No. 9 I/O planning with devices of Optical Shaft Encoder

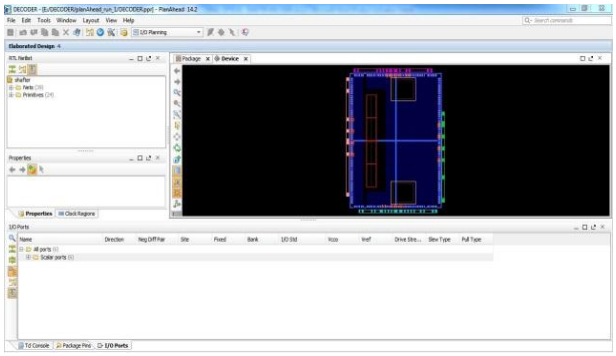

Figure No. 10 Planning with Package of Pins of Optical Shaft Encoder

Figure 10 shows that the planning of input and output with packages of pins. It shows the planning of input pins and output pins. It gives the package of input and output. It is shown for optical shaft encoder.

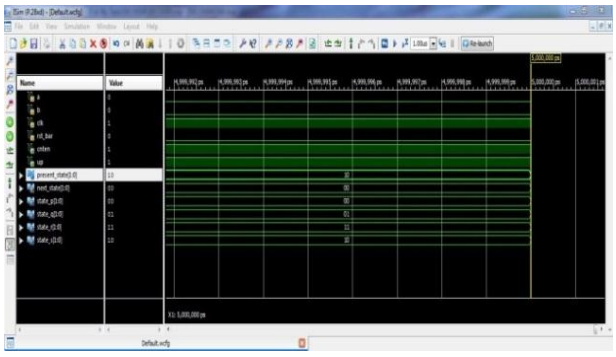

Figure No. 11 Waveform of Optical Shaft Encoder when input is 00

The above Figure no.11 shows that the waveform of optical shaft encoder. It shows how output is change in accordance with the input. When we give 00 input with rising edge of the clock then the present state is 10 and the next state is 00.

When clock is 1 and the value of input a and b is 1,0 respectively then the present state is 00 and next state is 10. When this synthesis process is over then Floor planning are also made by using Xilinx 14.2 Plan-Ahead. Xilinx Plan-Ahead is used to make pre-planning of the components used in hardware implementation. In this

components are used for decoder are 16 multiplexers, 1 inverter and 1 flip-flop is used.

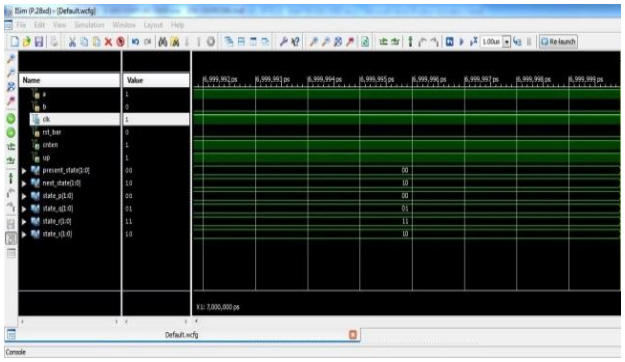

Figure No. 12 Waveform of Optical Shaft Encoder when 01

8. Hardware Implementation

After the process of software implementation of optical shaft encoder, we check our prototype before final implementation. FPGA platform is used for testing the prototype of code.

Figure No.13 Hardware Implementation of optical shaft encoder

9. Results

The optical shaft encoder for an artificial limb is done successfully on FPGA using Xilinx 14.2 software. The hardware is verified on FPGA Spartan 3 development board and it is running properly. By implementing this, power consuming becomes less and according to physics, time and power are directly proportional to each other. Therefore, by reducing power consumption, consumption of time is reducing. There is one significant result comes out while implementing is that the wastage of component is also less because we are checking our prototype on FPGA board.

Conclusions

In this modern era of technology, optical shaft encoder is used vastly in all robotic industry. With this technology security issues comes automatically. Security is very important issue and it is an art of restricting admittance to certain entries. So in this paper we are reduces the security issues of the rucks (people) by implementing a high security optical shaft encoder for an artificial limb. This project can be converted into a fully commercial project with little or no more modification.

Acknowledgments

Firstly, I would like to thank my parents for respecting my dreams and for unconditional love and care. I am also very thankful to my grandmother and brother for supporting me in every single step in my life. I am also thankful to Er. Gagandeep Singh, Software Engineer who helped me to enhance my programming skills.

References

Jin, D. and Y. Ito (2008). Application of Finite State Machine Theory to the Simulation of Reversed Non-Linear Hysteretic Relationships. Tsinghua Science & Technology 13: 46-52.

Kaur, J. and A. Sharma (2014). An overview of hardware approach of FPGA design by using XILINX.

Ling, A. P. A., *et al*. (2012). Enhancing smart grid system processes via philosophy of Security-case study based on information security systems. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 3(3): 94-112.

Saroch, K. and A. Sharma FPGA Based System Login Security Lock Design Using Finite State Machine.

Simpson, E. and P. Schaumont (2006). Offline hardware/software authentication for reconfigurable platforms. Cryptographic Hardware and Embedded Systems-CHES 2006, Springer: 311-323.

Books

Kenneth L. Short, VHDL for Engineers, Pearson, 2009

Floyd and Jain, Digital Fundamentals, Pearson, 2005

M. Mooris Mano, Michael D.Cliletli, Digital Design, Pearson,2008

R.D. Sudhakar Samuel, an illustrative approach to logic design, Pearson, 2005

Yalamanchili, Introductory VHDL, from simulation to synthesis, 2004

Sites

Wong (2011) VHDL 3 Finite State Machines: www.cse.cuhk.edu.hk/~khwong

Kamal (2005) Welcome to the ECE 449 Computer Design Lab: teal.gmu.edu/courses/ECE449.

file:///E:/A%20My%20Data/VOICEMAIL/Voicemail%20%20Wikipedia,%20the%20free%20encyclopedia.htm.