Field Programmable Gate Array (FPGA) based Secure Serial Data Acquisition with Memory Support Using Smart Card as an Authentication Device

Researcher:
MUHAMMAD ABDULLAH
Registration No.
281-FET/MSEE/S12

Supervisor:
Prof. Dr. MUHAMMAD USMAN
Co Supervisor:
Dr. IHSAN UL HAQ

Department of Electronics Engineering,
Faculty of Engineering & Technology,
INTERNATIONAL ISLAMIC UNIVERSITY, ISLAMABAD

Accession No TH-14948 (F) 9.

MS
621.395
MUF

- Field programmable Gate Array
- Gate Array Circuits
- Smart Card
- Authentication device.

Field Programmable Gate Array (FPGA) based Secure Serial Data Acquisition with Memory Support Using Smart Card as an Authentication Device

MUHAMMAD ABDULLAH
281-FET/MSEE/S12

Submitted in partial fulfillment of the requirements for the Master of Science degree in the discipline of Electronics Engineering at the faculty of Engineering & Technology, International Islamic University Islamabad.

Supervisor **2014**

Prof. Dr. MUHAMMAD USMAN

DEDICATION

Dedicated to all those kind hearted who guided me at any stage of my life to achieve the success in this life and afterwards.

Title of Thesis	**Field Programmable Gate Array (FPGA) based Secure Serial Data Acquisition with Memory support Using Smart Card as an Authentication Device**
Name of Student	**MUHAMMAD ABDULLAH**
Registration No.	**281-FET/MSEE/S12**

Accepted by the Department of Electronics Engineering, Faculty of Engineering & Technology, INTERNATIONAL ISLAMIC UNIVERSITY, ISLAMABAD, in partial fulfillment of the requirements for the Master of Science Degree in the Discipline of Electronics Engineering.

<u>Supervisor</u>
Prof. Dr. Muhammad Usman
Director, AWC, NESCOM

[illegible]

<u>Internal Examiner</u>
Dr. Muhammad Amir
Chairman, DEE, FET, IIU Islamabad

[illegible]

<u>External Examiner</u>
Prof. Dr. ABDUL JALIL
DCS, PIEAS, Nilore, Islamabad

[illegible]

<u>Chairman, DEE, FET, IIUI</u>
Dr. Muhammad Amir

[illegible]

<u>Dean, FET, IIU Islamabad</u>
Prof. Dr. Aqdas Naveed Malik

[illegible]

ABSTRACT

The target is to achieve a customizable solution for acquiring data over universal asynchronous serial communication link. The serial link is configured by setting its baud rate, parity bits, data bits etc. for reliable end to end transfer of data. The data transfer is protected by using different algorithms that are to be implemented in separate block. Memory support is provided in order to save and retrieve the entire Configuration used during the data transfer. Smart card circuitry is added which can authenticate the user as well as the data.

It's a hierarchical design in which the Top module acts as the root of the design. All the modules are communicating via Top module. The current status of the device is displayed through the Display controller. Serial Communication protocols are implemented in the UART controller. User and Data Authentication are done using the Smart Card in the Smart Card Controller module. Encoding/Decoding Data Controller is for secure transmission of data over the serial interface. Data is saved in Flash and retrieved back from the Flash using Flash Controller.

COPYRIGHTS

DECLARATION

I hereby affirm that this thesis is prepared by my personal effort, made under the sincere guidance of my supervisor and colleagues. It is further added that all the practical work is done by me and no portion of work, presented in this thesis has been submitted in support of any application for any degree or qualification in any university.

MUHAMMAD ABDULLAH
281-FET/MSEE/S12

Department of Electronics Engineering,
Faculty of Engineering & Technology,
International Islamic University,
Islamabad.

ACKNOWLEDGEMENT

No words of gratitude will ever be sufficient for the ALLAH Almighty who made me capable of learning, blessed me with the knowledge & intellect and facilitated me with the finest of the mentors all through my academic year.

Dr. Muhammad Usman, who made me, realize that no matter how high you think of your work, there is always a room for development. I present my deep gratitude to him, for being the most marvelous and enduring supervisor.

I also appreciate my colleagues, for their consistent encouragement and continuous support especially in increasing my knowledge.

And finally, to my parents, most wonderful parents of the world who grew me up to never frantically fall upon a yearning other than knowledge and my truly adorable sister and brothers, and my family for high moral support.

Mr. MUHAMMAD ABDULLAH

FORWARDING SHEET

The thesis entitled "Field Programmable Gate Array (FPGA) based Secure Serial Data Acquisition with Memory support Using Smart Card as an Authentication Device", submitted by Mr. MUHAMMAD ABDULLAH in partial fulfillment of M.S degree in Electronics Engineering has been completed under my guidance and supervision. I am satisfied with the quality of student's research work and allow him to submit this thesis for further process as per IIU rules & regulations.

Supervisor

Prof. Dr. MUHAMMAD USMAN

TABLE OF CONTENTS

LIST of TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

FPGA	Field Programmable Gate Array
IC	Integrated Circuit
FIR	Finite Impulse Response
FFT	Fast Fourier Transform
DSP	Digital Signal Processing
IP Cores	Intellectual Property Cores
HDL	Hardware Descriptive Language
VHDL	VHISC Hardware Descriptive Language
VHISC	Very High Speed Integrated Circuits
Xilinx ISE	Xilinx Integrated Software Environment
CIC filter	cascaded integrator-comb filter
MHz	Mega Hertz
MRI	Magnetic resonance Images
CT scan	computed tomography scan
ADC	Analog-to-Digital Converter
GPS	Global Positioning System
GNSS	Global Navigation Satellite System
IO	Input Output
UART	Universal Asynchronous Receiver and Transmitter

IR	Infra-Red
RS232	serial binary single-ended data and control signals
DAQ	Data Acquisition
IER	Interrupt Enable Register
USB	Universal Serial Bus
PET	positron emission tomography
PMT	photo multiplier tubes
APD	avalanche photo diodes
AES	Advanced Encryption Standard
HF	High Frequency
VHF	Very High Frequency
UHF	Ultra High Frequency
LED	Light Emitting Diodes
IER	Interrupt Enable Register
IIR	Interrupt Identification Register
FIFO	First in First Out
FCR	FIFO Control Register
LSR	Line Status Register
LCR	Line Control Register
ROM	Read Only Memory
TTL	Transistor-Transistor logic

CHAPTER NO.1

INTRODUCTION

1. INTRODUCTION

Field Programmable Gate Array (FPGA) is an IC which comprises of a number of unattached gates. The programming of this device (IC) is basically connecting the gates together to perform different operations. These gates combine to form different multipliers, adders, comparators etc. Using the Xilinx Integrated Software Environment (ISE) this can be done at a block-diagram level. These blocks can vary from low to a very high level – ranging from a single gate to an FIR or FFT. Their performance is limited by the number of gates, and the clock rate. Recent FPGAs have included Multipliers especially for performing DSP tasks more efficiently [1].

Just as there are many design development languages — and some are more suitable than others w.r.t the applications — there are many ways to implement the FPGA design. The first design practice is designing using the IP cores. This is the easiest technique as the most optimized code for a specific core is used and definitely the best result is obtained using this technique.

Depending upon the application it is, however, not always likely to find an IP core that meets the exact specification of the proposed system. It might be possible to get an IP cores that cover only a part of its functionality. In this case, it is required to design the system hardware using one of the existing HDLs. The traditional HDLs are VHDL and Verilog. Hardware Descriptive Languages allow customizing the design, and can also be used along with IP cores [2].

Advantages of FPGA designing include:

- Design reuse at high level design.
- Reduced obsolescence risk
- Design change or update using simple methodology.
- Increased design implementation options through design modularization

Performance considerations include

- Design architecture of the targeted application.
- Size of the data and address bus.
- Pipelining and parallel processing.
- Strategies and plans to prevent stalls.

The tool used for designing the hardware in FPGA is Xilinx Integrated Software Environment (ISE). It provides the design environment for writing the hardware descriptive code. It synthesizes the HDL code and simulates it. Debugging is also supported in it. The design is finally downloaded to the target FPGA [3].

1.1 PROBLEM STATEMENT

This project is an implementation of System on Chip design. Verilog (Hardware Descriptive Language) is used as a hardware programming language in Xilinx Integrated Software Environment (ISE).

It is basically to implement such an interlinked modulo block diagram in which the data acquisition through serial communication link is free of error with highest level of security.

Sequential as well as combinational Verilog HDL programming is required to inter link the blocks input / output with one another e.g. Flash memory handling for data storage and data retrieving is linked with smart card application, UART transceiver and algorithm implementation block. The design is re-described just by reprogramming the HDL for any module and downloading it in FPGA to get the desired output.

1.2 SIGNIFICANCE

Modern Field Programmable Gate Arrays (FPGAs) can process complex algorithms of signal processing, communication over wired or wireless link. The ease of using FPGAs along with its low expense and selected dedicated hardware make it an ideal technology.

A key improvement in medicine over the last 100 years is the ability to get the images of the inside of a living creature without having intrusive surgery. The doctors and the scientists are able to view high- resolution images of the anatomical structures inside the body through the imaging techniques such as magnetic resonance imaging (MRI) and X-ray computer tomography (CT). Positron emission tomography (PET) is also included in this class of medical imaging. FPGAs are used for data acquisition of many current PET systems. FPGAs are used to collect the data and to perform/ apply filtering algorithm on it. Hence the desired information is obtained.

The use of modern FPGAs is discussed by the engineers and the scientists in PET scanner as well as in our next generation scanner to produce higher resolution images. In order to measure some metabolic reactions inside living creatures, PET is used. Radioactive decays are used in it to measure these results. This process can be divided into three stages. Generation and administration of a radioactive tracer is the first step. A tracer is actually an isotope which emits radio waves. It also includes a metabolically active molecule. The tracer is vaccinated into the body which needs to be scanned. After some time as this tracer is distributed in the body, it is then examined by scanning. How the process occurs: positron is emitted by this radioactive tracer during its decay process which is used in PET studies. Two 511 KeV anti parallel photons are eventually produced as this positron decays. These photons are actually scanned in the PET scanner. PET scanner is the second component of PET. It consists of a number of sensors that detect these photons. It also has the electronic device that processes those signals which are generated by the sensors. Scintillator crystal sand Avalanche photo diodes (APD) are used to make these sensors. They are often made of scintillator crystal sand photo multiplier tubes (PMT). The aim of the scintillator is to convert these emitted photons into many visible light photons. On the other side Avalanche photo diodes or photo multiplier tubes are used to generate an electrical pulse from these visible light photons. The electronic device processes these pulses to determine the information residing in it (i.e. energy, timing). The data is sent for tomography image reconstruction to view the information in a 3D image [4].

The significance of modern FPGAs is also proposed by the scientists Kole D.K. and Rahaman H. Using Universal Asynchronous Receiver Transmitter (UART), for secure transfer of data, they proposed the application of Advanced

Encryption Standard (AES) algorithm. AES-128 algorithm is implemented to encrypt the data before its transmission from the transmitter. The same algorithm is implemented to decrypt the data as UART receiver module receives it. The AES-128 design consists of a clock generator module. Different modules are clocked at different frequencies through it. The design is completely described using Hardware Description Language (HDL), Verilog and is functionally tested and verified using Xilinx ISE software. It results in taking on average of 47.2msec to transmit an encrypted 128 bit stream. On the receiving side it takes 36.7msec to receive the same amount of decrypted data. The proposed architecture is divided into different modules which are designed using FPGA technology. [5]

Global Positioning System (GPS) is another potential application of modern FPGAs. Global Positioning System (GPS) is a navigation system formed from a constellation of satellites and along with their ground stations. The satellite's location is transmitted by the satellite continuously along with the time. The GPS system uses this information coming from the satellite as a reference to calculate positions. A GPS system consists of a receiver which measures its distance from each of the satellites coming in its span. USB port or UART / serial communication is the main component of the GPS receiver. In this research, using Hardware Description Language (HDL), a data path is designed. This data path uses serial communication link which is designed, and implemented using

Field Programmable Gate Arrays (FPGAs). Using the Hardware Description Language (HDL) Verilog, Global Positioning System (GPS) receiver is designed which consists of data path with the serial communication port. VHDL is used to design the UART (Universal Asynchronous Receiver Transmitter) and implemented in FPGA. Xilinx ISE tools are used to synthesize and implement the design. The design is tested using the simulator ModelSim and chip scope pro .Finally the design is burned into a Xilinx FPGA [6].

GPS IF data can also be acquired through this system. Extra hardware is also required in it. An RF front end will receive the data from the GPS satellite. It is converted from radio frequency signal to an IF signal of few MHz's. This analog data is passed through ADCs to convert it into digital form so that it can be processed by the FPGA based system. Hence, the data acquisition device is used to digitize GPS signal in real-time also.

1.3 OBJECTIVE

The main objective of the project is to put efforts in the direction of indigenous development of an easily configurable, Plug & Play type, hardware (FPGA) based customized solution to protect data traffic between trusted hosts from eavesdropping.

Universal asynchronous serial communication link is the interface between the sources of the data. The parameters of the serial link are baud rate setting, parity bits implementation strategy, information bits etc. To make the communication secure over the serial link interface, algorithms are implemented in separate block. Configuration Management is implemented in support with Memory Module. Access level is also controllable by implementing the smart card device.

The project is divided into different modules. Top controller is the main module controlling the whole design. UART Controller is for Serial Communication. Smart Card Controller is for User and Device Authentication. Encoding/Decoding Data Controller is for secure transmission of data over the serial interface. Display Controller displays the current status of the device. Flash Controller is for configuration management.

1.4 SCOPE

The scope of project is to secure any type of link that might be:

- Wireless link from Air to Ground station
- Satellite link
- Critical wired link between two nodes
- HF/VHF/UHF radio link

CHAPTER NO.2

LITERATURE REVIEW

2 LITERATURE REVIEW

A few examples of the FPGA based practical problems, along with how they would be implemented:

1. Decimation filters in a digital wireless receiver. Typically, this is a CIC filter, operating at a sample rate of 50-100 MHz. A 5-stage CIC has 10 registers & 10 adders, giving an "add rate" of 500-1000 MHz. The CIC structure is very easy to implement it in an FPGA.

2. Digital radio receiver – baseband processing. Some receiver types require FFTs for signal acquisition. Matched filters are required once a signal is acquired. Both blocks can be easily implemented by FPGA approach.

3. Image processing. Most of the operations on an image are simple and very repetitive – best implemented in an FPGA [3].

4. Encryption algorithms can also be implemented on FPGAs achieving high data rates. User and device authentication can be implemented using smart card interfaced with FPGAs over serial interface.

5. FPGAs are also significant in medical applications such as ultrasound, MRI, CT scanners, and digital X-ray. Analog-to-digital converters (ADCs) are used to sample large chunk of data and serial interfaces are used to get the sampled data for further processing to extract the information [7].

6. Many e-mail systems uses regular expressions to filter for spam. Thousands of regular expressions are matched by these systems. This need parallel processing of data where FPGAs can do it quite effectively. The FPGA design methodology and execution model is used by the developers to create FPGA-based solutions for real-world filtering applications [8].

7. Global Positioning System (GPS) is a global navigation system, in which the information is coming from the satellite in space. The parameters it provides to the user anywhere anytime on the earth are time and location. The GPS system includes the data acquisition system, which can be implemented in Field Programmable Gate Array (FPGA). Data extraction unit for tracking and navigation can also be implemented in FPGAs [9].

8. The first open source GNSS receiver design was made the three scientists Peter J Mumford, Kevin Parkinson and Andrew G Dempster. It was based on FPGA technology for research and development. The design architecture consists of a custom circuit board, a baseband processor and the application

firmware. FPGA was used to design a GNSS tracking module and other custom logic. Using the FPGA's IO ports, memory chips were accessed. A configuration controller was the controlling device of the system. The system was clocked by a real time clock. A 3-axis accelerometer and the power supply was also the part of the system [10].

9. The IC used for serial communication is mainly Universal Asynchronous Receiver and Transmitter (UART) RS232 protocol. Using Hardware Description Language (HDL), Ching-Chang Wong & Yu-Han Lin proposed a reusable Intellectual Property (IP) of UART design method. The design was implemented on a Field Programmable Gate Array (FPGA) chip. The design mainly consists of a receiver module and a transmitter module, implemented in FPGA. The baud rate generation module was also implemented in it. FPGA provides a flexible implementation environment as the design is easily reconfigured and reused at different stages by the user. In order to receive the distance information of objects, the proposed design was applied to a data detector of Infra-red (IR) ranging system. The data detector of digital compass was used to receive the head direction of robot. And to communicate with a host computer and the other robots wireless modem was used. Most of the application circuits have been implemented on a FPGA chip successfully [11].

CHAPTER NO.3

PROPOSED SYSTEM

3 PROPOSED SYSTEM

This project is a customizable solution for acquiring asynchronous data over serial communication link at different baud rates using different parity settings. The level of security of data is also controllable using different algorithms for encoding and decoding. Memory support is there to save the customizable setting for the secure data communication over the serial link.

There are two main blocks in the design. Communication Block and the Algorithm Implementation Block. The communication block is further divided into the following blocks.

- UART controller, Encoder and Decoder interface modules, Display Controller, Flash Controller and Smart Card Controller

The Algorithm Implementation Block is divided into two modules.

- Encoding Algorithm and Decoding Algorithm

3.1 COMMUNICATION BLOCK

3.1.1 UART CONTROLLER

UART controller controls the data rate of the communication link along with the other parameters. Encoded data is received by the UART controller and is channeled to the algorithmic block via decoder.

Similarly sending data process involves the encoder which receives encoded data from the algorithmic block and the data is send over the communication link via UART controller.

3.1.2 ENCODER and DECODER MODULES

Encoder and Decoder modules just connect the communication block with the algorithmic block of the project.

3.1.3 DISPLAY CONTROLLER

Display controller outputs the status of the project. Result of the Smart card authentication process is also displayed via display controller. It consists of a series of dedicated LEDs and also seven segment LEDs.

3.1.4 FLASH CONTROLLER

1 mega bit Memory is used to save all the important data used during the communication. For example receiving bytes, sending bytes, link information, encoding decoding schemes data etc.

3.1.5 SMART CARD CONTROLLER

Smart card controller provides the device security as well as the user authentication. Only user with valid smart card can access the device.

3.2 BLOCK DIAGRAM OF THE PROJECT

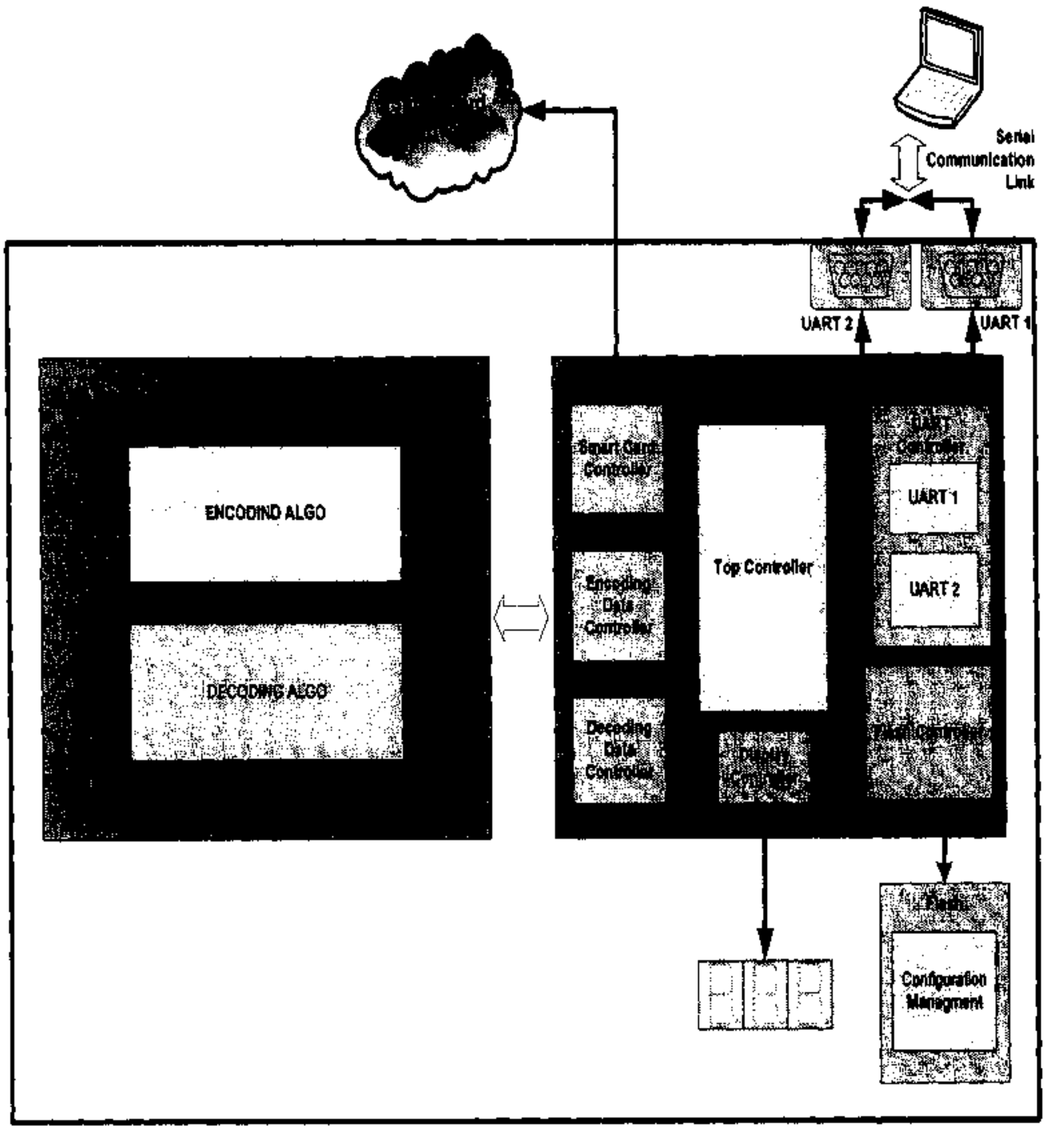


Figure 1 Block Diagram of the Project

CHAPTER NO.4

DATA ACQUISITION USING UART TRANSCEIVER AS A SERIAL INTERFACE

4 DATA ACQUISITION USING UART TRANSCEIVER AS A SERIAL INTERFACE

Data Acquisition model consists of the UART (Universal Asynchronous Receiver/Transmitter) which provides serial communication capabilities, allows communication with the external devices using a serial interface and RS232 protocol.

Serial communication protocols were used long before computers were available. In the 19th century for use in telegraphy a serial protocol was developed known as the Morse code [12]. It uses variable length sequences of short and long pulses to encode characters.

The technology has enhanced a lot nowadays. Every day comes with the more efficient solution for a particular application. Using minimum resources to get the required output is the area of research and development. In this era of Information technology, various kinds of devices use the communication protocols of serial nature like Universal Serial Bus (USB) and Ethernet. Common examples are RS232C, RS422, RS4851, I2C, Ethernet communication links.

In Serial communication voltage signals are sequentially transmitted after a specific time period. These signals are generally representing a character. A single periodic voltage signal is known as a bit in serial communication.

The protocol used in serial communication is asynchronous. Therefore, if the data is sampled in the perceived middle of the bit time, it is exempted to small differences in the clocks of the sending and receiving sides. For the accurate transmission and receiving of the information data, Read Only Memory and Random Access Memory blocks are also used in it.

The hardware for Data Acquisition model is described using Hardware Descriptive Language (HDL) in Field Programmable Gate Array (FPGA). Configurations for the Data Acquisition are set through the different register based modules. External Flash is attached as a corresponding hardware for storage of these configurations. Flash along with the Smart card is also used for authentication purpose i.e. only authenticated user will be given the access to established communication link using the device to receive or transmit the secure data.

4.1 FEATURES OF DAQ

The two important features of Data Acquisition module are

- First in First out operation.
- Direct access to Register level Modification.

4.2 INTERRUPT ENABLE REGISTER

UART Interrupt generation is enabled and disabled through the Interrupt Enable Register (IER). Table used for configuring 8-bit IER register is described below.

Bit #	Access	Description
0	Read Write	Interrupt to indicate that data is received 0 – disable the interrupt 1 – enable the interrupt
1	Read Write	Interrupt to indicate that Transmitter Holding Register is empty 0 – disable the interrupt 1 – enable the interrupt
2	Read Write	Interrupt to indicate Receiver Line Status 0 – disable the interrupt 1 – enable the interrupt
3	Read Write	Interrupt to indicate Modem Status Interrupt 0 – disable the interrupt 1 – enable the interrupt
7-4	Read Write	These bits are reserved and should have logic 0.

Table 1 Interrupt Enable Register Table

- **Reset Value for IER: 00h**

  Note: The default value set on the hardware reset is known as the reset value.

4.3 INTERRUPT IDENTIFICATION REGISTER

Interrupt Identification Register (IIR) is used to retrieve the highest priority interrupt status. Zero-bit of the register indicates its logic as 0 if an interrupt is pending. No interrupt is pending if it is 1. Interrupts related with IIR are Receiver Line Status interrupt, Receiver Data available interrupt, Timeout Indication interrupt, Transmitter Holding Register empty interrupt and Interrupt to indicate Modem Status.

Description of the Interrupts related with the IIR is given in the Table 2 along with the other information.

- **The default value on hardware reset is C1h**
- **Bits 4 and 5: Logic '0'.**
- **Bits 6 and 7: Logic '1' for compatibility reason.**

Interrupt Type	Priority	Bit 3	Bit 2	Bit 1	Interrupt Source	Interrupt Reset Control
Interrupt to indicate Receiver Line Status	1st	0	1	1	Break Interrupt or Framing errors, Parity, Overrun.	Line Status Register is read
Interrupt to indicate Receiver Data available	2nd	0	1	0	Trigger level of the FIFO is reached	Below the trigger level FIFO drops.
Interrupt to indicate Timeout Indication	2nd	1	1	0	For the last 4 Char times no character has been input to the FIFO or read from it and there is at least 1 character in the FIFO also.	(Receiver Buffer Register) FIFO is read
Interrupt to indicate Transmitter Holding Register empty	3rd	0	0	1	Transmitter Holding Register Empty	IIR is read or data is written to the Transmitter Holding Register.
Interrupt to indicate Modem Status	4th	0	0	0	DCD, DSR, CTS, or RI.	Modem status register is read.

Table 2 Interrupt Identification Register Table

4.4 FIFO CONTROL REGISTER

When the trigger level if the Receiver FIFO equals the number of bytes in FIFO, it generates an interrupt. The trigger level of the FIFO is adjusted through FIFO Control Register (FCR). Moreover FCR is used to clear the FIFOs also.

Bit #	Access	Description
0	Write Only	As the only FIFO mode is supported by this UART, therefore this bit is ignored able.
1	Write Only	In order to reset the logic of Receiver FIFO and to clear the Receiver FIFO, 1 is written to the bit 1. But shift register is not cleared, i.e. current characters are continuously received.
2	Write Only	In order to reset the logic of Transmitter FIFO and to clear the Transmitter FIFO, 1 is written to the bit 2. But shift register is not cleared, i.e. current characters are continuously transmitted.
5-3	Write Only	These bit are ignored
7-6	Write Only	Trigger level of the Receiver FIFO Interrupt is defined as 1 byte if 00 is written 4 byte if 01 is written 8 byte if 10 is written 11 byte if 11 is written

Table 3 FIFO Control Register Table

- **Reset Value: 11000000b**
- **Note: The default value set on the hardware reset is known as the reset value.**

4.5 LINE STATUS REGISTER

Receiver FIFO's and Transmitter FIFO's control signals are defined in a read only Line Status Register (LSR) which is eight bit long. All types of error during the transmission of the data or receiving of the data are defined in it.

Bit #	Access	Description
0	Read Only	This bit indicates the Data Ready (DR) 0 – FOFO is empty i.e. has no characters in it 1 – FIFO has received at least one character and it is present in the FIFO.
1	Read Only	This bit indicates Overrun Error (OE) 1 – The character is received in the receiver shift register while the FIFO is full. And the data in the FIFO will not be overwritten. Only the data in the shift register will be overwritten if the characters are received continuously. If a character is read from the FIFO, the bit is cleared. 0 – FIFO is not overrun

Bit #	Access	Description
2	Read Only	This bit indicates the Parity Error (PE) 1 – If the parity bit is not received as it is defined. Reading from the register will clear it. And interrupt for Receiver Line Status is also generated. 0 – If the parity bit is received as it is defined i.e. the current character has no parity error.
3	Read Only	This bit indicates the Framing Error (FE) 1 – There is no valid stop indicator or signal at the top of the FIFO for the received character. It means that the data received is corrupted. Reading from the register will clear it. And interrupt for Receiver Line Status is also generated. 0 – Valid stop bit is there in the received.
4	Read Only	This bit indicates the Break Interrupt (BI) 1 –When the current character has reached the break condition i.e. the line is held grounded for the time equal to receiving of one character. And the time for receiving a character is equal to the total time for receiving of start bit, data bits, parity bits and stop bit. If it is grounded for time less than the time for a character, the UART pauses for a valid start bit of the next incoming character. Reading from the register will clear it. And interrupt for Receiver Line Status is also generated. 0 – Character is received with no break condition in it.

Bit #	Access	Description
5	Read Only	This bit indicates that there is no character in Transmit FIFO. 1 – There is no character in the transmitter FIFO i.e. it is empty. An interrupt is generated indicating that the Transmitter Holding Register is empty. When the data is written to the transmitter FIFO, the bit is cleared. 0 – Transmitter FIFO is not empty.
6	Read Only	This bit indicates the Transmitter Status. 1 – If both the transmitter shift register and the transmitter FIFO are empty. When the data is written to the transmitter FIFO, the bit is cleared 0 – Transmitter FIFO is not empty.
7	Read Only	1 – If there is at least one character present in the FIFO that has any one of the above mentioned errors i.e. error in parity or break indications or framing error etc. Reading from the register will clear this bit. 0 – FIFO has character with no errors.

Table 4 Line Status Register Table

4.6 LINE CONTROL REGISTER

Universal asynchronous receiver transmitter accesses the serial data communication settings through Line Control Register (LCR). These configurations are Character length, Stop bits, Parity Enable and Divisor Latch Access for Baud rate settings.

Bit #	Access	Description
1-0	Read Write	These bits indicate the number of data bits in a character. 00 – data bits are 5 01 – data bits are 6 10 – data bits are 7 11 – data bits are 8
2	Read Write	This bit indicates the number of stop bits in a character. 0 – stop bit generated is 1 1 – stop bit generated is 1.5 if 5 data-bit are selected in bit 1, and bit 0 (= 00) 1– stop bits generated are 2 if data bits are 6,7 or 8. Note: Only the first stop bit is checked by the receiver.
3	Read Write	This bit enables or disables the Parity bit 0 – The outgoing character has no parity. 1 – Each received or transmitted character has a parity bit in it.

Bit #	Access	Description
4	Read Write	This bit selects the parity as Even Parity. 0 – If in a character the total numbers of 1's are even and parity bit (i.e. bit 3)of LCR is also 1. 1 – If in a character the total numbers of 1's are odd and parity bit of LCR is also 1.
5	Read Write	This bit selects the parity as Stick Parity. 0 – Either even or odd parity is selected. 1 –Parity bit generated is 1 if bit 3 and 4 of LCR 1. And the parity bit generated is 0 if bit 3 and 4 of LCR 1 and 0 respectively.
6	Read Write	This bit controls the Break Indication 1 – UART is forced into break state i.e. line is held grounded. 0 – break is not enabled
7	Read Write	This bit selects Divisor Latch or normal register access. 1 – Divisor latch is enabled and normal registers are disabled. 0 – Only Normal registers are enabled to be accessed.

Table 5 Line Control Register Table

- **Reset Value: 00000011b**

4.7 DIVISOR LATCHES

The seventh bit of LCR is used to access divisor latch. By setting this bit of LCR to '1', Divisor latch is enabled disabling the normal register access. In order to revert the current access to the normal registers having the same addresses, this bit is set to '0'. Divisor latch is 16 bits wide register. This register is internally accessed. And it is used as a single number. In order to ensure the normal operation both the bytes of the register are simultaneously set. External reset will set the register to its default value i.e. zero hence disabling the entire serial I/Os.

4.8 UART CONFIGURATIONS

Following are the customizable parameters of UART.

4.8.1 STOP BITs CONFIGURATION

It can be configured to stop bit 0 or stop bit 1.

4.8.2 NUMBER OF DATA BITS

It can be configured to No. of bits equal to 5 or 6 or 7 or 8.

4.8.3 PARITY BIT CONFIGURATION

It can be configured to Parity bit Odd, Even or None.

4.8.4 DATA RATE CONFIGURATION

Data rate configuration involves the input clock signal along with the divisor latch. Depending upon the real time input clock signal baud rate can either be set as elaborated below.

4.8.4.1 BAUDRATE SETTING

in order to set the baud rate for UART the following formula is used.

$$\text{Divisor Latch (2 Byte Reg.)} = \frac{\text{Input Clock}}{(16 \times \text{Desired Baud Rate})}$$

Implement the formula using the following steps.

- In order to access the divisor latch register, set LCR (Line Control Register) bit 7 to 1.
- The above mentioned formula is used to set its value.
- Set LCR bit 7 to 0 (disable Divisor Latch and) enable Normal Registers.

Set MSB first then LSByte, because as LSByte is Set it get starts its operation (counting).

4.8.4.2 DIVISOR LATCH VALUE TABLE

Following are the values calculated from the above mentioned formula to set the Divisor Latches using different baud rates for different clocks.

Baud Rate	Input Clock (MHz)				
(bit/sec)	24.576	9.216	7.3728	5.5296	3.6864
110	13963.63	5236.36	4189.09	3141.81	2094.54
300	5120	1920	1536	1152	768
1200	1280	480	384	288	192
2400	640	240	192	144	96
4800	320	120	96	72	48
9600	160	60	48	36	24
19200	80	30	24	18	12
38400	40	15	12	9	6
57600	26.67	10	8	6	4
115200	13.33	5	4	3	2
230400	6.67	2.5	2	1.5	1
460800	3.33	1.25	1	0.75	0.5
921600	1.67	0.625	0.5	0.375	0.25

Table 6 Divisor Latch Value Table

- **Note**: For a specific clock signal, baud rate with only integer values outcome can be achieved e.g. For24.576 MHz Input Clock, Only 300, 1200, 2400, 4800, 9600, 19200, 38400 Baud rates can be achieved.

4.9 UART TRANSCEIVER BLOCK DIAGRAM

UART Transceiver consists of three main modules. Rom, Control Unit and the Core module.

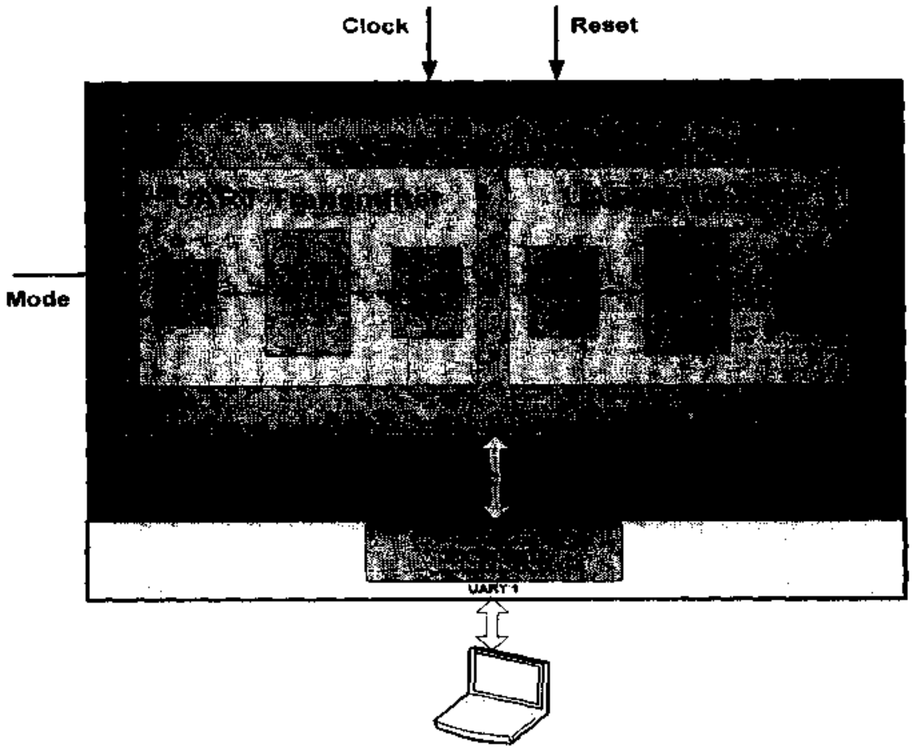


Figure 2 Block Diagram of the UART Transceiver

Control Unit is further divided into different modules; Initialization module, Establishment of connection, Interrupt identification, Line status, Data available and FIFO. The core module functions on Reset are; FIFOs of the receiver and transmitter are cleared. Secondly the Shift registers of the receiver and transmitter are cleared. The default value of Divisor Latch register is set to 0. The default values of Line Control Register are set to the communication of 8 bits of data, no parity, and 1 stop bit.

By default all interrupts are disabled in the Interrupt Enable Register.

4.10 TIMING DIAGRAM OF UART TRANSCEIVE

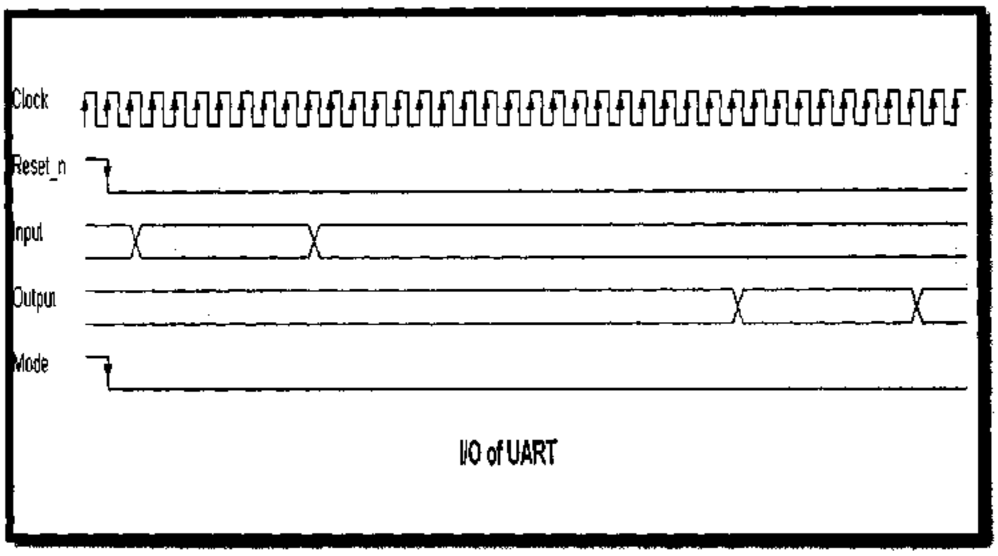


Figure 3 Timing Diagram of the UART Transceiver

UART initialization process requires the following procedures.

- Configure the UART by setting the Line Control Register to the desired line control parameters i.e. Character length, Stop bits, Parity Enable etc.
- For Baud rate configuration Set bit 7 to '1' to allow access to the Divisor Latches. Set the Divisor Latches, MSB first, LSB next.
- In order to disable access to Divisor Latches, set bit 7 of LCR to '0'. The communication process starts working after this setting and data can be sent and received.
- The mode control will shift between Normal input/output and loop back operation. In loop back mode the input data is transmitted back to the output port.

4.11 SECURE DATA FLOW THROUGH SERIAL COMMUNICATION LINK

Secure Data acquisition over the serial communication interface requires synchronization of all the configurable parameters on both the transmitter and receiver end.

4.11.1 SECURE DATA FLOW DIAGRAM OF SERIAL COMMUNICATION LINK

Secure communication hardware device basically comprises of two separate communication links. Each link communicates bi directionally. One link is dedicated for data encoding and the other one for data decoding as shown in the figure 4.

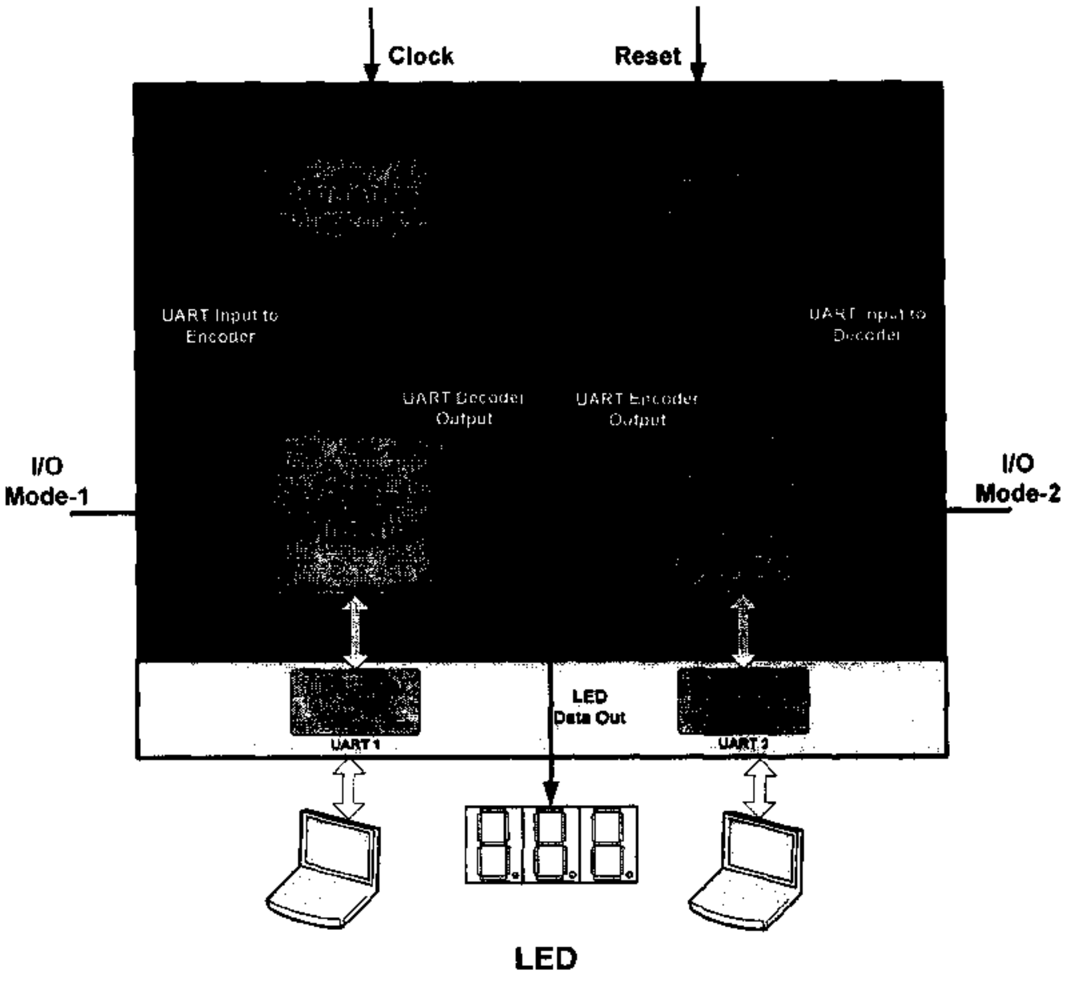


Figure 4 Data Flow Diagram of the Secure Serial Communication Link

After configuring the hardware serial ports, ensure that the same configuration are applied at the both ends.

Use Encoding and decoding algorithm for secure data transfer. Hence the security level is customizable depending upon the complexity of algorithm and hardware resources.

4.11.2 DATA FLOW CHART OF SECURE SERIAL COMMUNICATION LINK

When the hardware is up and the user is given the access for secure data communication over the serial interface, the data flows through the channel as shown in figure 5.

Data is divided into two categories.

4.11.2.1 UART INPUT TO ENCODER/ DECODER

UART Input to Encoder is basically the data which is to be encoded. After receiving that data over the serial interface the information is extracted from it. The information is routed to the encoder in order to encode it.

UART Input to Decoder is basically the data in which the information is only encoded. After receiving that data over the serial interface the information is extracted from it. The information is routed to the decoder in order to translate it.

4.11.2.2 ENCODER/ DECODER OUTPUT TO UART

UART Encoder data is actually the information data which is to be transmitted over the serial interface. It is the output of the Encoder. It is routed to the UART Transceiver module which makes it compatible with channel by adding extra bits to it.

UART Decoded data is the output of the Decoder. In order to transmit this information over the serial interface it is routed to the UART Transceiver module which makes it compatible with channel by adding extra bits to it.

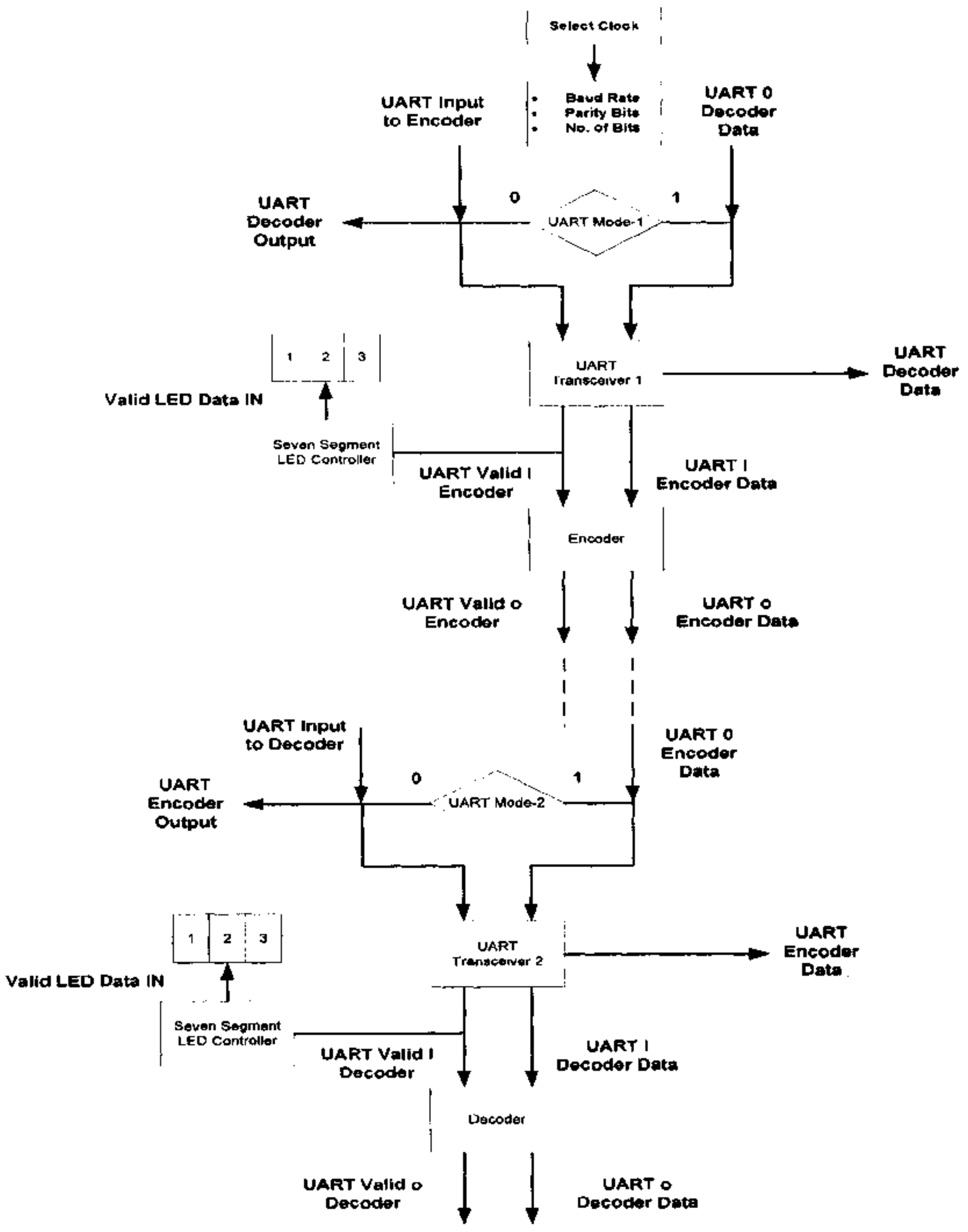


Figure 5 **Data Flow Chart for Secure Serial Communication Link**

4.12 TIMING DIAGRAM OF UART TRANSCEIVER

Figure 6 is the timing diagram implemented on UART Transceiver. UART receives encoded serial data. Information bits are extracted from the data. The information is decoded to get the desired response. For debugging the mode will shift between loop back and normal input/output operation. In loop back mode the received data is transmitted back from the output port.

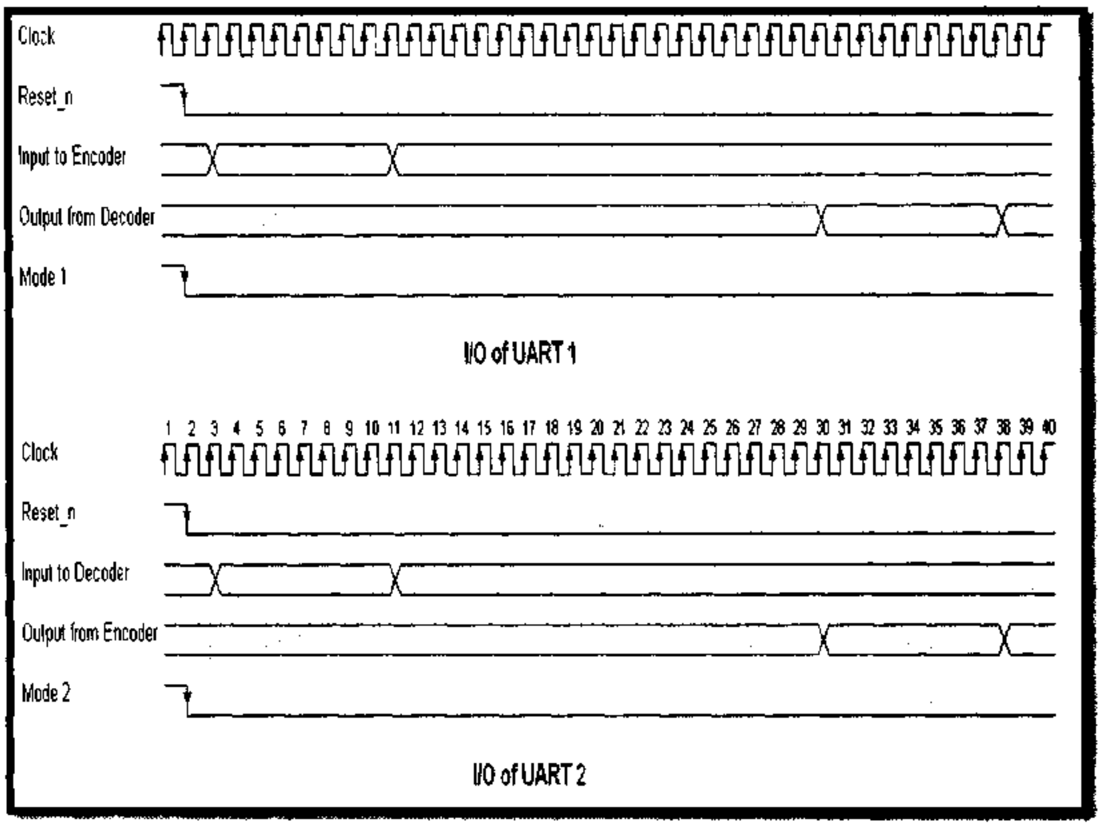


Figure 6 Timing Diagram of the UART Transceiver

4.13 HARDWARE DATA FLOW OF SECURE SERIAL COMMUNICATION LINK

Voltage Signal level for RS232 protocol is used for Universal Asynchronous Receiver Transmitter port.

General purpose input output FPGA ports uses the voltage signal level of 3.3V TTL. MAX RS232 level converter IC is used to convert from the 3.3V TTL to RS232 protocol voltage signal level. Same IC is used to convert from RS232 protocol voltage signal level to the voltage signal level of 3.3V TTL.

4.13.1 DATA FLOW FROM UART-1 TO FPGA

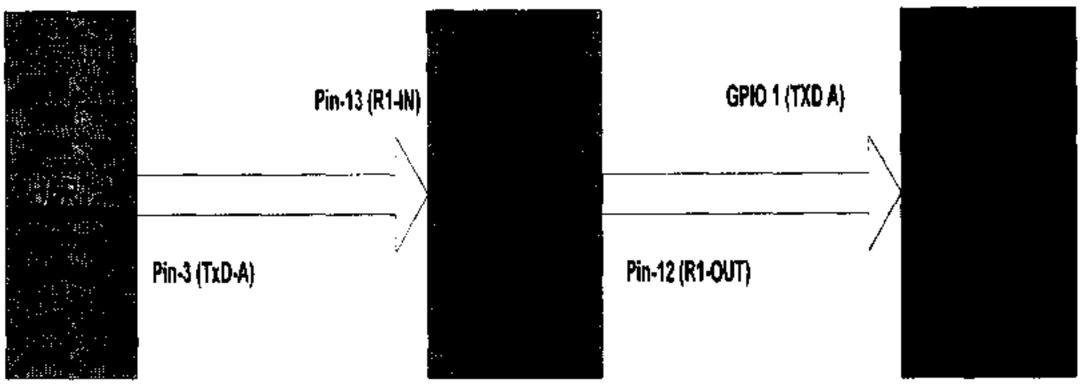


Figure 7 Data Flow from UART-1 to FPGA

UART-1 is basically the RS232 connector. Third pin of the connector is used for input. The data received on the third pin (TxD-A) is routed to the pin no. 13 of the IC MAX 232, which output the data to any specified pin of the General Purpose Input Output port (Let for example it is GPIO 1) of the FPGA according to the voltage level needed. This data is actually the UART input to Encoder as mentioned above in the article 4.11.2.1.

4.13.2 DATA FLOW FROM FPGATO UART-1

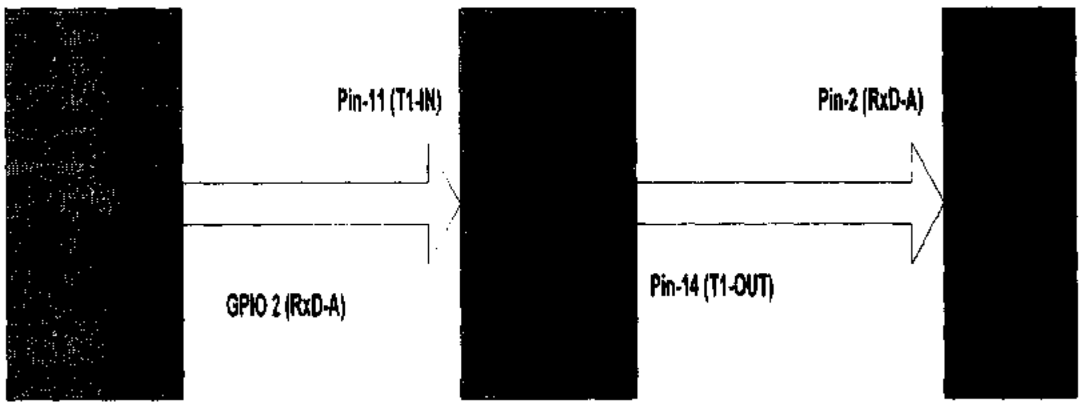


Figure 8 Data Flow from FPGA to UART-1

Data from the GPIO-2 of the FPGA is routed to the IC MAX RS232 which shifts its voltage level to the RS232 protocol voltage level required for transmission over the channel. UART-1 here is also the RS232 connector.

Second pin of the connector is used for output. The data transmitted on the second pin (RxD-A) of the connector is actually the Decoder output to UART i.e. mentioned above in the article 4.11.2.2.

4.13.3 DATA FLOW FROM UART-2 TO FPGA

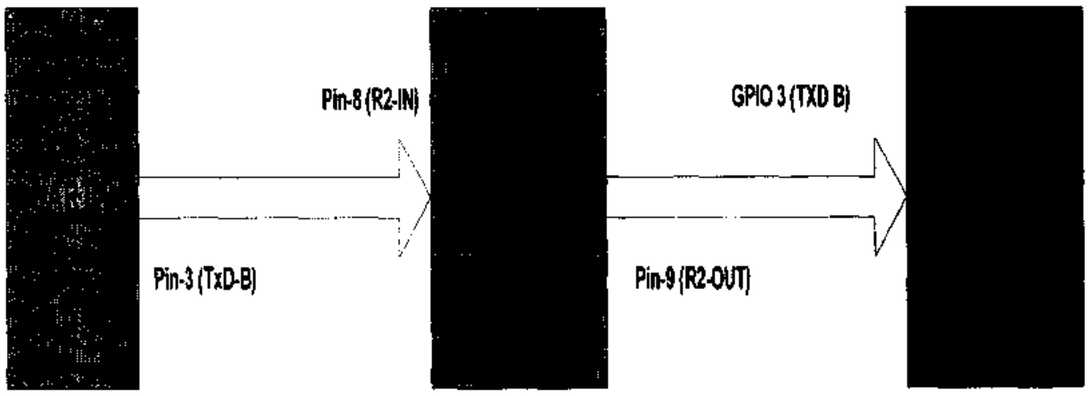


Figure 9 Data Flow from UART-2 to FPGA

UART-2 is basically the RS232 connector. Third pin of the connector is used for input. The data received on the third pin (TxD-B) is routed to the pin no. 8 of the IC MAX 232, which output the data to the GPIO 3 of the FPGA according to the voltage level needed.

4.13.4 DATA FLOW FROM FPGA TO UART-2

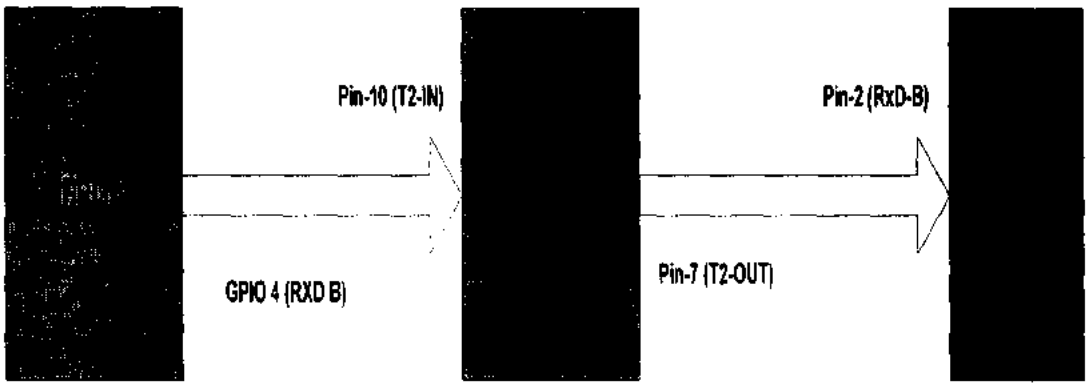


Figure 10 Data Flow from FPGA to UART-2

Data from the GPIO-4 of the FPGA is routed to the IC MAX RS232 which shifts its voltage level to the RS232 protocol voltage level required for transmission over the channel. Here UART-2 is also the RS232 connector. Second pin of the connector is used for output/ transmission of data over the channel. The data transmitted on the second pin (RxD-A) of the connector is actually the Encoder output to the UART i.e. mentioned above in the article 4.11.2.2.

4.13.5 THE SN74LVC4245A LEVEL CONVERTER

4.13.5.1 PIN DIAGRAM

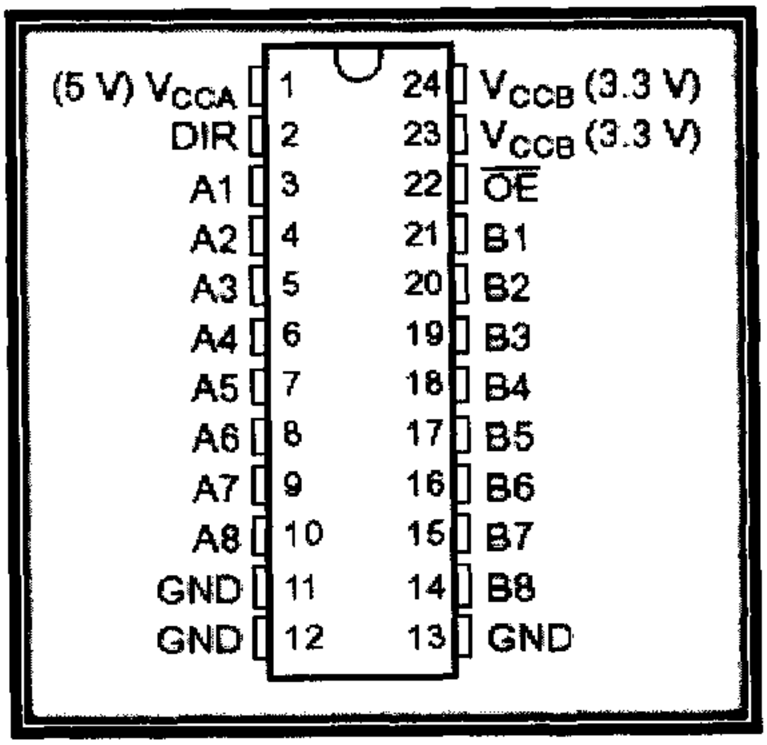


Figure 11 Pin Diagram of SN74LVC4245A Level Converter

4.13.5.2 FEATURES OF LEVEL CONVERTER

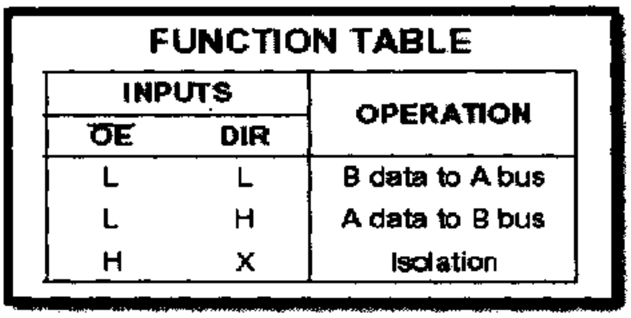
- Its Voltage Translator is bidirectional.
- Port A has voltage 5.5 V as high and port B has high voltage ranging from 2.7 V to 3.6 V.

4.13.5.3 DESCRIPTION

It is an 8 pin non-inverting IC with two supply rails. It is also used as transmitter or receiver; it has two ports, A and B. For port A high voltage is termed as VCCA and for port B it is termed as VCCB. VCCA is set at 5 V and VCCB is set at 3.3 V. The voltage level can be reduced from 5 V to 3.3V and amplified from 3.3 V to 5 V also.

Input output data busses communicate asynchronously using this IC. Direction-control (DIR) bit is used to control the data flow either from port A to port B or from port B to port A. The busses can be isolated using output-enable (OE) bit. VCCA is used as the power source for the IC.

4.13.5.4 FUNCTION TABLE



FUNCTION TABLE

INPUTS		OPERATION
$\overline{OE}$	DIR	
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

Figure 12 Functional Table of SN74LVC4245A Level Converter

CHAPTER NO.5

DISPLAY CENTER

5. DISPLAY CENTER

Display center consists of a number of light emitting diodes connected to the various function related to the smart card and the seven segments LED.

5.1 SEVEN SEGMENT LIGHT EMITTING DIODE

5.1.1 PIN DIAGRAM

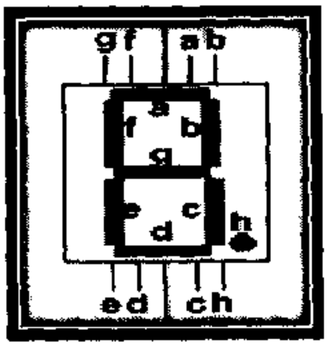


Figure 13 Pin Diagram of seven segments LED

5.1.2 FEATURES

Seven Segments LEDs consists of 10 pins. Each pin is associated with the LED segment in such a manner that it glows that specific segment on input to that pin. It is connected in series for different purposes. It displays number of bytes received, number of bytes sent, Baud rate and Operating Clock.

5.1.3 CONNECTION IN SERIES

Seven segment LED is connected in series in order to display the number in decimal format. The common pin of all the seven segment LEDs are connected with a single input as shown in the figure 14.

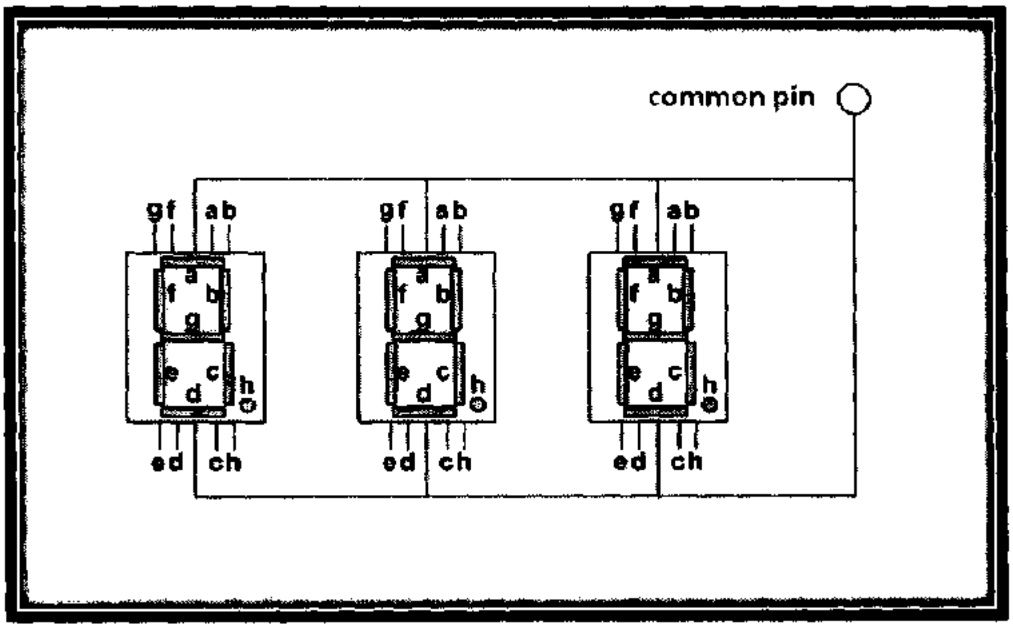


Figure 14 Series connection of Seven Segments LEDs

5.1.4 TYPES

There are two types of seven segment LEDs.

- Common Cathode Seven Segment LED.
- Common Anode Seven Segment LED.

5.1.4.1 COMMON CATHODE

In Common Cathode, Common Pin of seven segments LED is connected to Ground and other LED Pins should be connected to High Voltage Vcc. The figure 15 shows the direction of current flow from a diode i.e. from higher voltage level to the ground. The diode will glow on applying High voltage to it e.g. applying High voltage on pin a will lit the diode connected with pin a and the current will start flowing in the direction as shown in the figure 15.

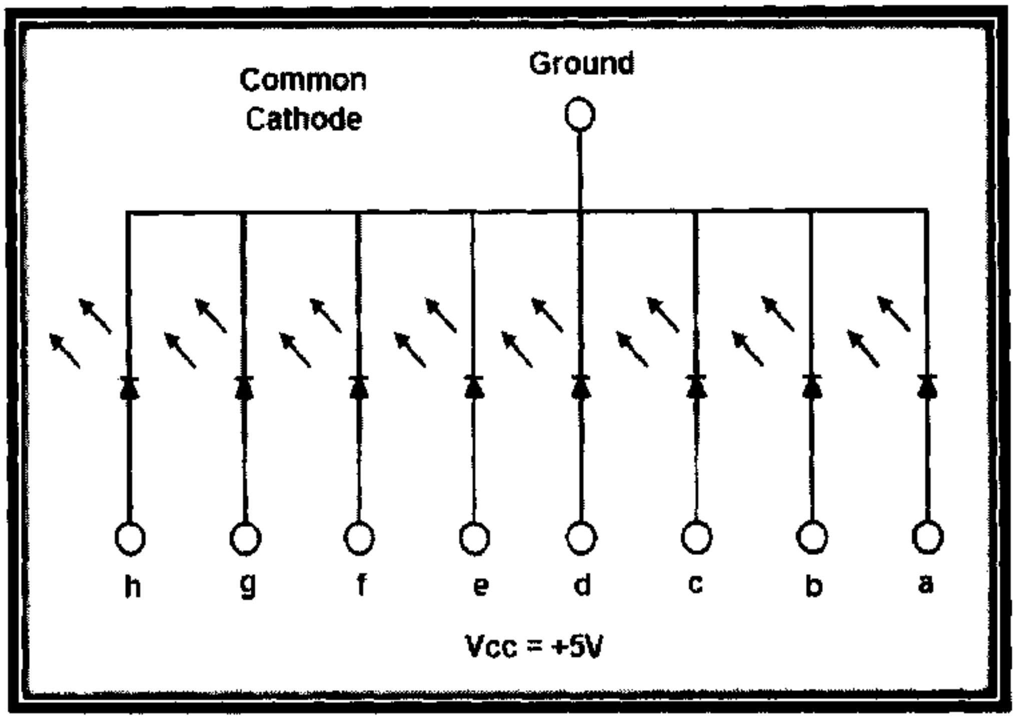


Figure 15 Common Cathode configuration of Seven Segments LED

5.1.4.2 COMMON ANODE

In Common Anode, Common Pin of seven segments LED is connected to Vcc and other LED Pins should be connected to Ground. The figure 16 shows the direction of current flow from a diode i.e. from higher voltage level to the ground. The diode will glow on applying Low voltage to it e.g. applying Low voltage on pin a will lit the diode connected with pin a and the current will start flowing in the direction as shown in the figure 16.

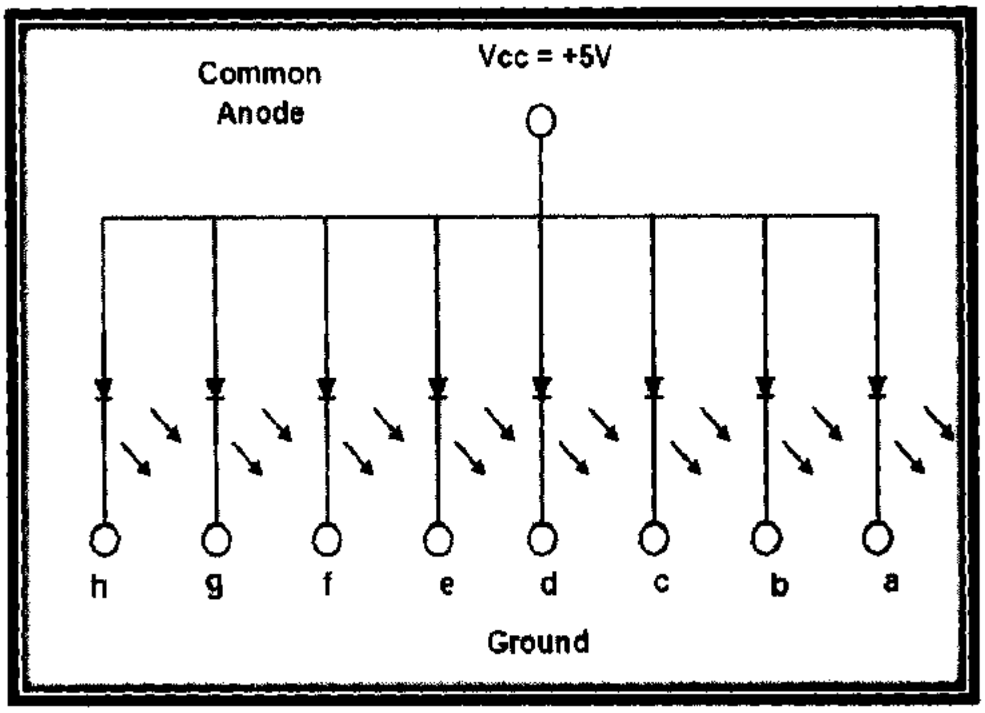


Figure 16 Common Cathode configuration of Seven Segments LED

5.2 FPGA CONNECTION WITH SEVEN SEGMENT LEDs

LED requires 9 pins of the general purpose input output port of FPGA. Eight GPIO pins are connected to the eight LED segments. The two common pins of LED are connected to the ninth GPIO pin of the FPGA.

For the LED with common cathode the common pin is given Low Voltage constantly. In order to glow any LED segment, the specific pin is given High Voltage.

For the LED with common anode the common pin is given High Voltage constantly. And in order to glow any LED segment. the specific pin is given Low Voltage.

The pin connection diagram of seven segments LED is shown in figure 17. It also shows the GPIO header connection with the FPGA .i.e. GPIO header B is connected with the FPGA-2.

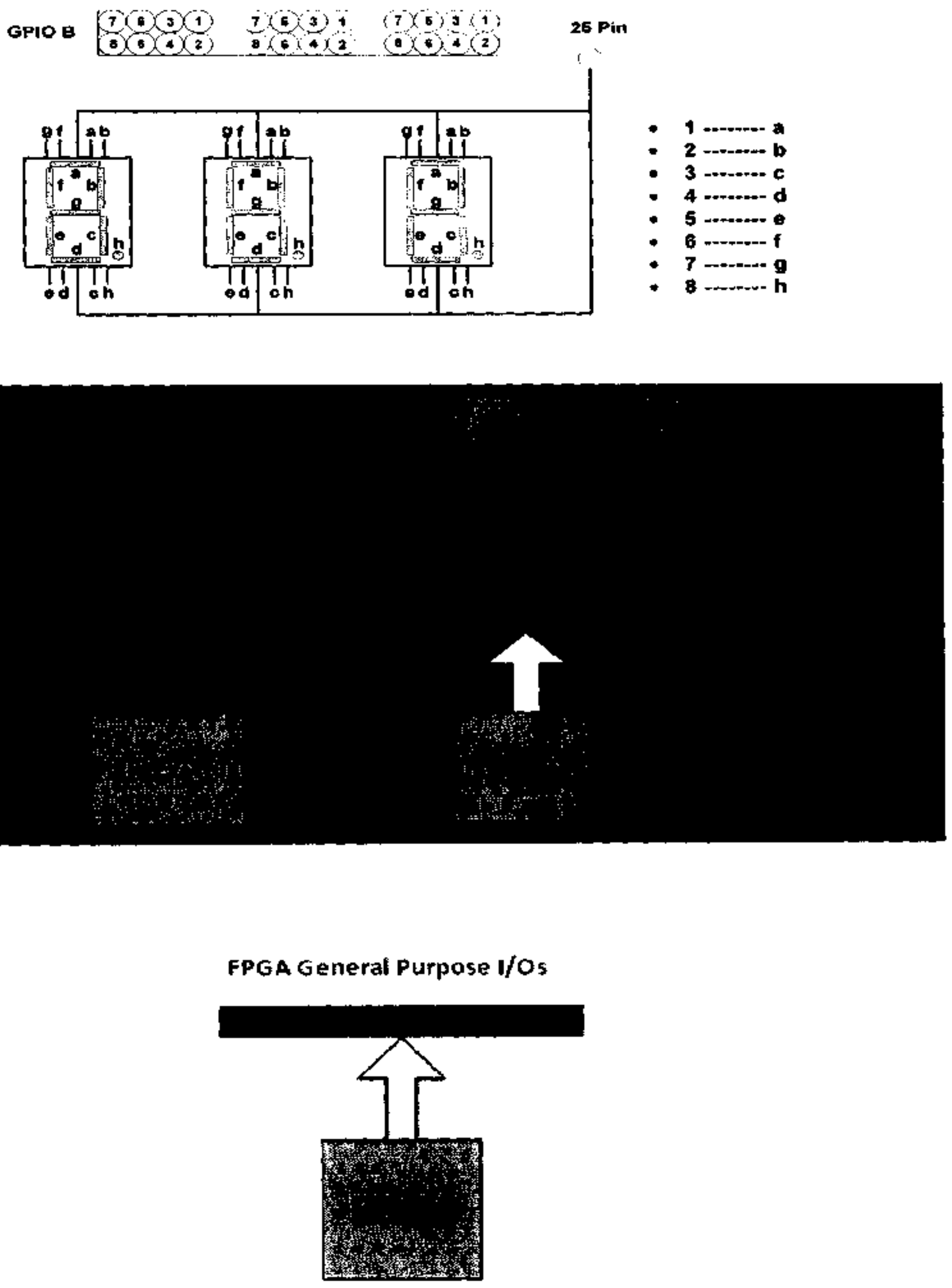


Figure 17 LED Pin Configuration

CHAPTER NO.6

CONFIGURATION MANAGMENT

6 CONFIGURATION MANAGEMENT

The device uses different parameters during the initialization of its modules. These parameters are also updated during the communication of the secure data over the serial link. Nonvolatile flash memory is used to store these parameters.

6.1 CONFIGURATION PARAMETERS OF THE MODULES

The entire configurations are stored by the user before the start of communication.

6.1.1 HARDWARE CONFIGURATION

The initialization parameter of the hardware is the Input clock selection

6.1.2 SERIAL LINK CONFIGURATION

The serial link connection parameters are Baud rate, Number of bits, Parity and Stop bits. After the successful transmission and reception of data the updated parameters are

- Number of bytes received
- Number of bytes transmitted

6.1.3 SMART CARD CONFIGURATION

Smart card parameter is the identification byte for authentication of user.

6.1.4 ENCODER and DECODER CONFIGURATION

Encoding and Decoding data used by the Encoder and Decoder respectively during the communication process.

6.2 FLASH CONTROLLER

Flash Controller defines the drivers to access the ATMEL AT29LV1024 IC Flash Memory. One mega bits of data can be stored in it. It is a non-volatile memory. It is used to write different types of data to their specific locations in Flash. It also reads the data from the Flash. The Hardware communicates with it while establishing the connection.

6.3 BLOCK DIAGRAM OF FLASH CONTROLLER

Flash controller is actually linked with all the controllers. UART controller communicates with it in order to connect on the channel. Display controller also reads data from Flash controller for its display. Smart card authentication process also involves smart card identification byte stored in Flash to be compared with the one in the smart card. Encoding/ Decoding data are used during the communication over the serial link to encode/ decode data.

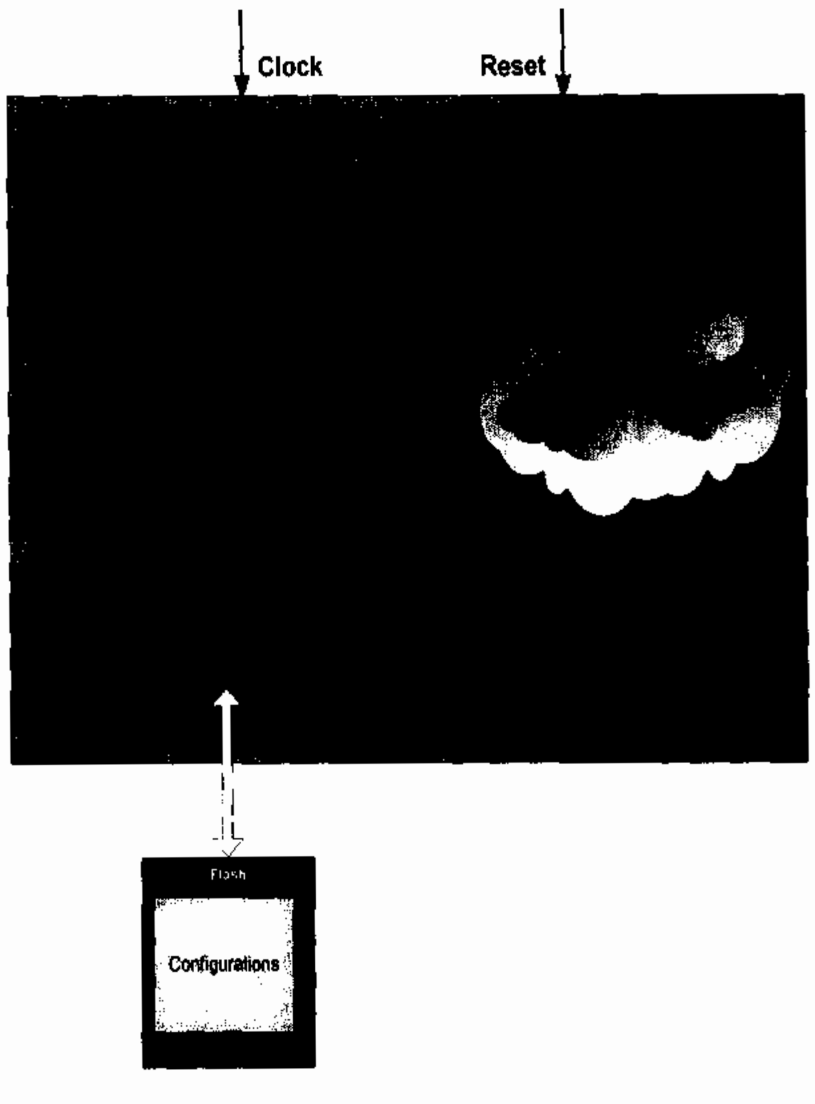


Figure 18 **Block Diagram of Flash Controller**

6.4 WRITING DATA INTO FLASH

Flash controller writes the data into the Flash. The Flash used consists of 128 word memory blocks called as sectors. The total number of sectors in the Flash is 512. Two bytes combines to form a word i.e. a word is the 16 bits wide.

Writing of data will always be 128 words every time even if a single bit is to be changed. In other words it's always a one complete sector to be programmed every time while writing data into flash.

If in any case a single word or byte of information is to be changed in any one of the 512 sectors, the complete data for sector will be loaded i.e. all the 128 words will be programmed into the sector of the Flash. While programming the Flash IC, the sector is erased automatically before writing the data into the sector. For erasing the data no command is required.

6.4.1 SOFTWARE DATA PROTECTION PROCEDURE

In order to write the data into the Flash a proper procedure is designed called software data protection. The main feature of this procedure is to protect the device from unintended programming. In this procedure three commands are issued serially to the specific addresses with specific data. After these command the 128 word data is loaded. The sector load data operation will always follow these commands of the software data protection procedure.

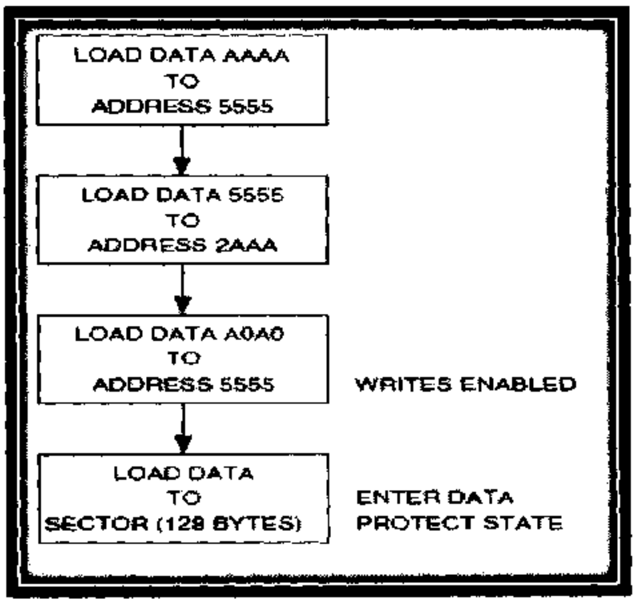

LOAD DATA AAAA TO ADDRESS 5555
LOAD DATA 5555 TO ADDRESS 2AAA
LOAD DATA A0A0 TO ADDRESS 5555
WRITES ENABLED
LOAD DATA TO SECTOR (128 BYTES)
ENTER DATA PROTECT STATE

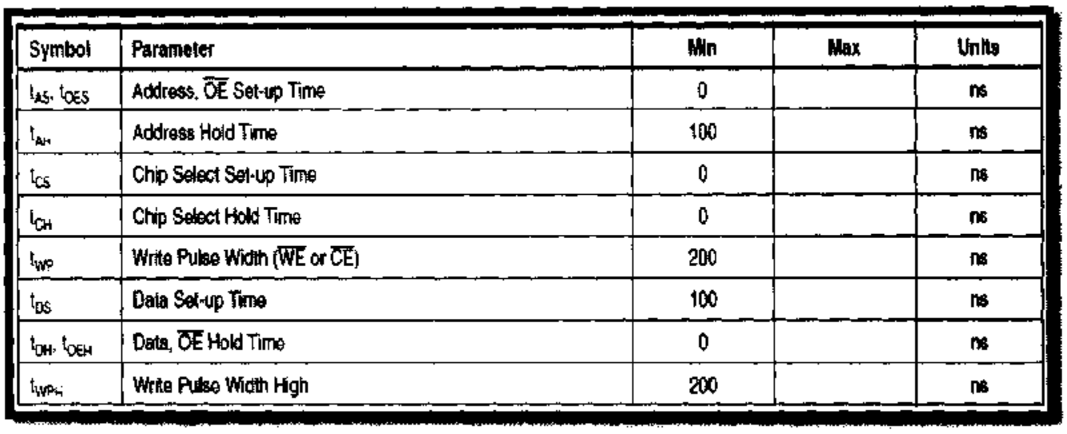
Figure 19 Software data protection commands

When the above mentioned three commands of specific data to the specific addresses are executed, programming of data will be started. In order to write a word, a low voltage is given on the Write Enable (WE) or Chip Enable (CE) control pin. Output Enable pin of the Flash should be kept high. The flash will automatically latch the address on the falling edge of WE or CE, whichever occurs last. While the data is latched on the rising edge of WE or CE, whichever occurs first. Second word of the data is written by handling the WE and CE control signals in the same manner as explained just above. And 128 words are loaded to program or reprogram a sector. When loading the 128 words into the device finishes, internal programming procedure is started to program all these words.

Entering the words onto the device is explained above. Note that there should be maximum of 150 micro seconds delay between the programming of two successive words. Sector address is from A7 to A15 which remains constant during the programming cycle. The word address will be specified by the pins A0 to A6. Sequential addressing is not required while programming the words.

6.4.2 TIMING SPECIFICATIONS FOR WRITING THE DATA

Timing specification for the programming of the sector data should be followed.



Symbol	Parameter	Min	Max	Units
t_{AS}, t_{OES}	Address, $\overline{OE}$ Set-up Time	0		ns
t_{AH}	Address Hold Time	100		ns
t_{CS}	Chip Select Set-up Time	0		ns
t_{CH}	Chip Select Hold Time	0		ns
t_{WP}	Write Pulse Width ($\overline{WE}$ or $\overline{CE}$)	200		ns
t_{DS}	Data Set-up Time	100		ns
t_{DH}, t_{OEH}	Data, $\overline{OE}$ Hold Time	0		ns
t_{WPH}	Write Pulse Width High	200		ns

Figure 20 Timing specifications for writing data into Flash

6.4.3 WRITING DATA PROCEDURES

Writing data into Flash can be done by the following two programming procedures.

- Write Enable controlled programming.
- Chip Enable controlled programming.

6.4.3.1 WRITE ENABLE CONTROLLED PROGRAMMING

In write enable controlled programming procedure CE is given low pulse before WE. When the WE is given low pulse, address is latched. And when the WE is changed from low to high data is latched. So it should occur before CE i.e. WE is given high voltage before CE. In the same manner 128 word are loaded. And the same technique will be followed during each program cycle.

6.4.3.1.1. TIMING DIAGRAM

Timing diagram of write enabled controlled programming (writing data into Flash) is given below.

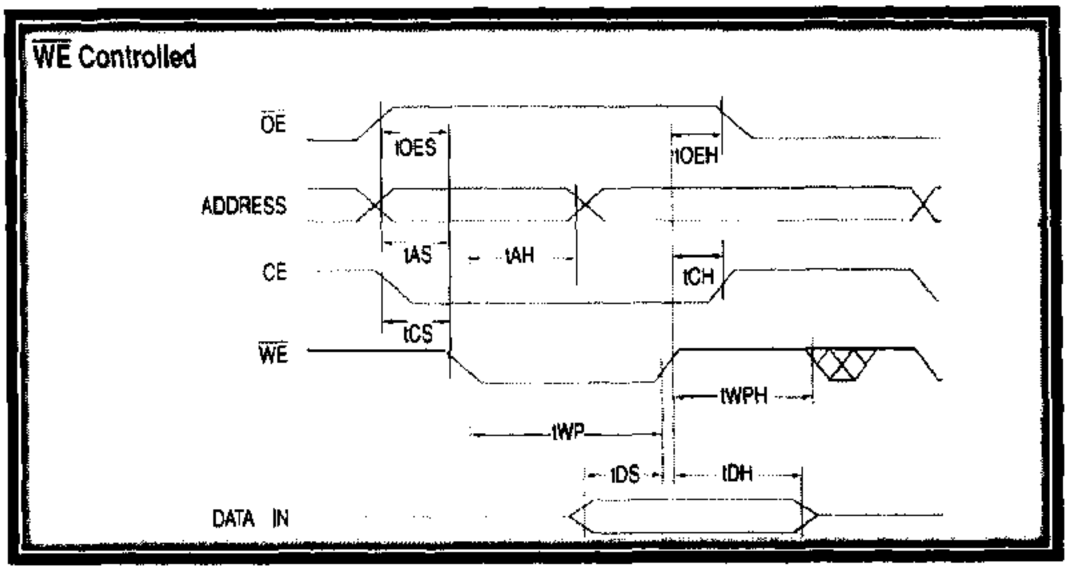


Figure 21 WE controlled Timing Diagram for writing data

6.4.3.2 CHIP ENABLE CONTROLLED PROGRAMMING

In chip enable controlled programming procedure CE is given low pulse after WE. When the CE is given low pulse, address is latched. And when the CE is changed from low to high data is latched. So it should occur before WE i.e. CE is given high voltage before WE. In the same manner 128 word are loaded. And the same technique will be followed during each program cycle.

6.4.3.2.1. TIMING DIAGRAM

Timing diagram of chip enabled controlled programming (writing data into Flash) is given below.

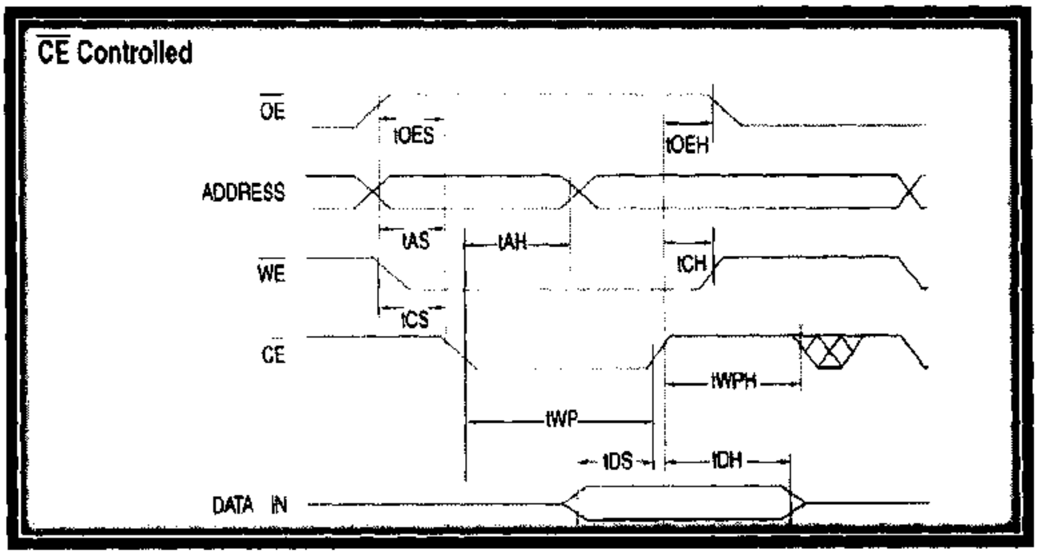


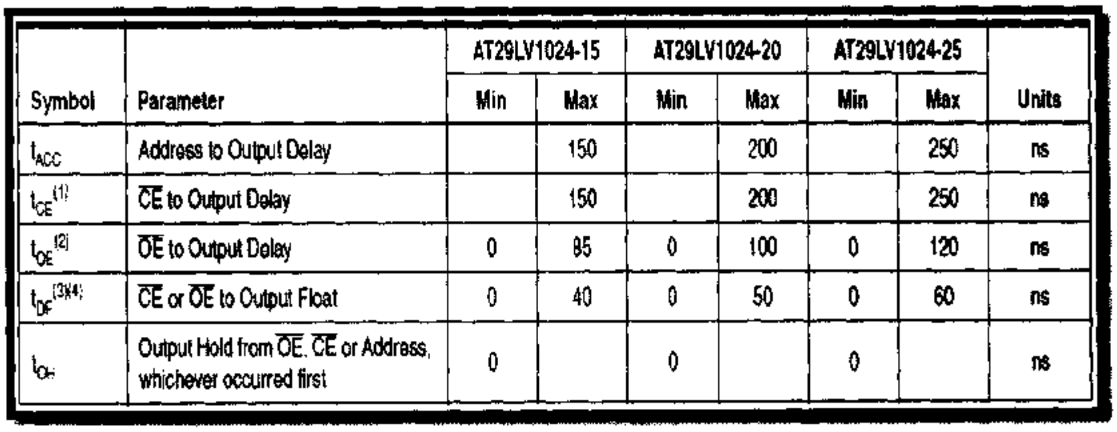
Figure 22 CE controlled Timing Diagram for writing data

6.5 READING THE DATA FROM THE FLASH

The Flash memory is like an Electrically Programmable Read Only Memory. In order to read the data from it, OE pin and CE pin are kept low while WE pin is at high voltage. The address is given on the address pins. And the data at the given address is output on data pins. Data pins show high impedance if OE or CE is at high voltage.

6.5.1 TIMING SPECIFICATIONS OF READING THE DATA

Timing specification for the reading of the sector data should be followed.



Symbol	Parameter	AT29LV1024-15		AT29LV1024-20		AT29LV1024-25		Units
		Min	Max	Min	Max	Min	Max	
t_{ACC}	Address to Output Delay		150		200		250	ns
t_{CE}[1]	$\overline{CE}$ to Output Delay		150		200		250	ns
t_{OE}[2]	$\overline{OE}$ to Output Delay	0	85	0	100	0	120	ns
t_{DF}[3][4]	$\overline{CE}$ or $\overline{OE}$ to Output Float	0	40	0	50	0	60	ns
t_{OH}	Output Hold from $\overline{OE}$, $\overline{CE}$ or Address, whichever occurred first	0		0		0		ns

Figure 23 Timing specifications for reading data from the Flash

6.5.2 TIMING DIAGRAM OF READING THE DATA

In order to read data from the Flash, CE is given low pulse before OE. When the OE is given low pulse, address is latched. The Flash outputs the data on the output/ data pins after 85 ns as shown in figure 23. The output remains asserted until the signal CE or OE, whichever is given high voltage at first.

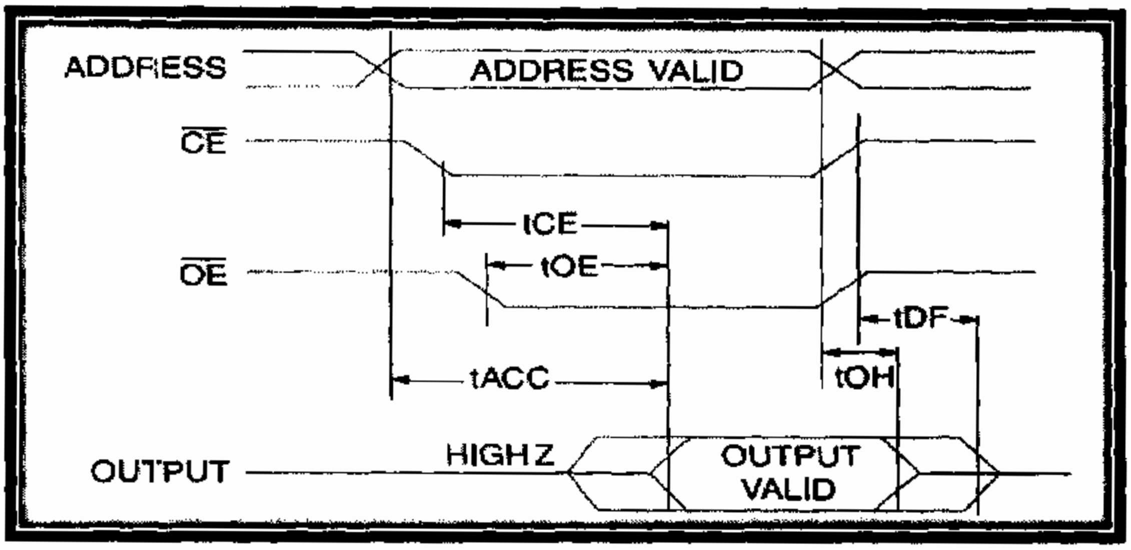


Figure 24 Timing Diagram for writing data into Flash

CHAPTER NO.7

SMART CARD OPERATION

7 SMART CARD OPERATION

7.1 INTRODUCTION

Smart Card is used as an authentication device. Only the user with the valid card can have the access to communicate securely over the channel using this hardware. Smart Card Controller is designed in HDL in order to communicate with Smart Card.

7.2 DESCRIPTION

Smart card is an EMV Integrated Circuit Card with a processor fixed in it. The most important aspect of processor card design is security. The processor protects access to the memory, using tamper-proof hardware design coupled with high-security software algorithms.

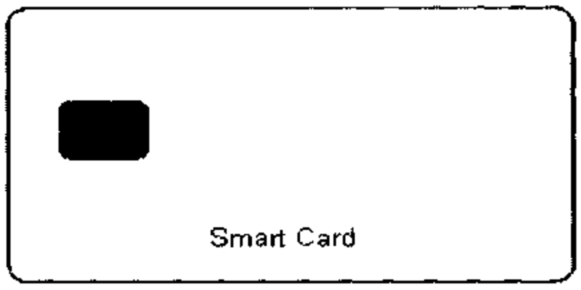


Figure 25 Smart Card

7.3 FEATURES

The important features of Smart Card are.

- Complete miniature processor. The metallic contact area shown in figure 26 is a complete miniature computer.
- The Electrically Erasable &Programmable Read Only Memory (EEPROM) is the 'hard disk' of the card. Data written to EEPROM retains its value when the card is powered down.
- Random Access Memory (RAM).

Figure 26 Smart Card processor

7.4 PROGRAMMING SMART CARD

The hardware required for programming a Smart Card is Personal Computer (PC) and a Smart Card Reader as shown in figure 27.

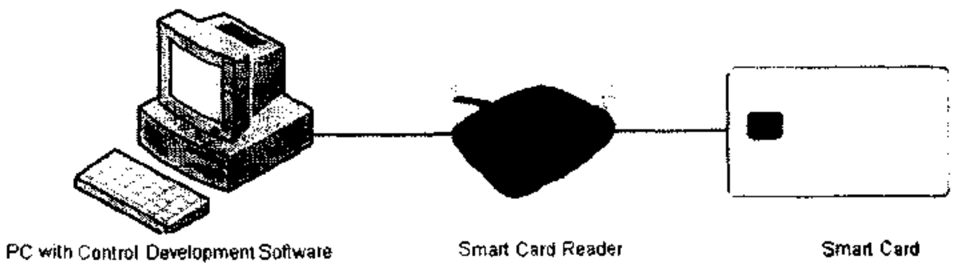


Figure 27 Smart Card Programming Environment

Smart Card Reader is connected to the personal computer through a serial RS232 data port with DB9 connection. It can also use PS2 port of keyboard for its communication. Hence for using the mouse the PC should have a USB data port. Smart card development software should be installed on the windows based personal computer.

In order to program the smart card through the smart card reader, smart card development environment is used. It is an interface between the programmer and the smart card via smart card reader. The communication is done through serial interface and the default communication baud rate is 9600bps, no parity, eight bits and one-stop bits. The smart card is inserted in the smart card's socket in such a way that its metallic area should face upward. Both the two ports, serial RS232 port and PS2 port, are used for its connection with the PC. The figure 28 shows the diagram of Smart Card reader.

Figure 28 Smart Card Reader

The purpose of smart card is to allow only authorized user to communicate over the channel establish by the hardware. The smart card is programmed with the smart card identification byte in order to authenticate the user. This smart card identification byte is also stored in Flash memory of the hardware as a part of the configuration. Flash controller is used to program it in the flash memory. User is given the access when smart card identification byte matches the stored one in the flash.

7.5 SMART CARD PROGRAMMING FLOW CHART

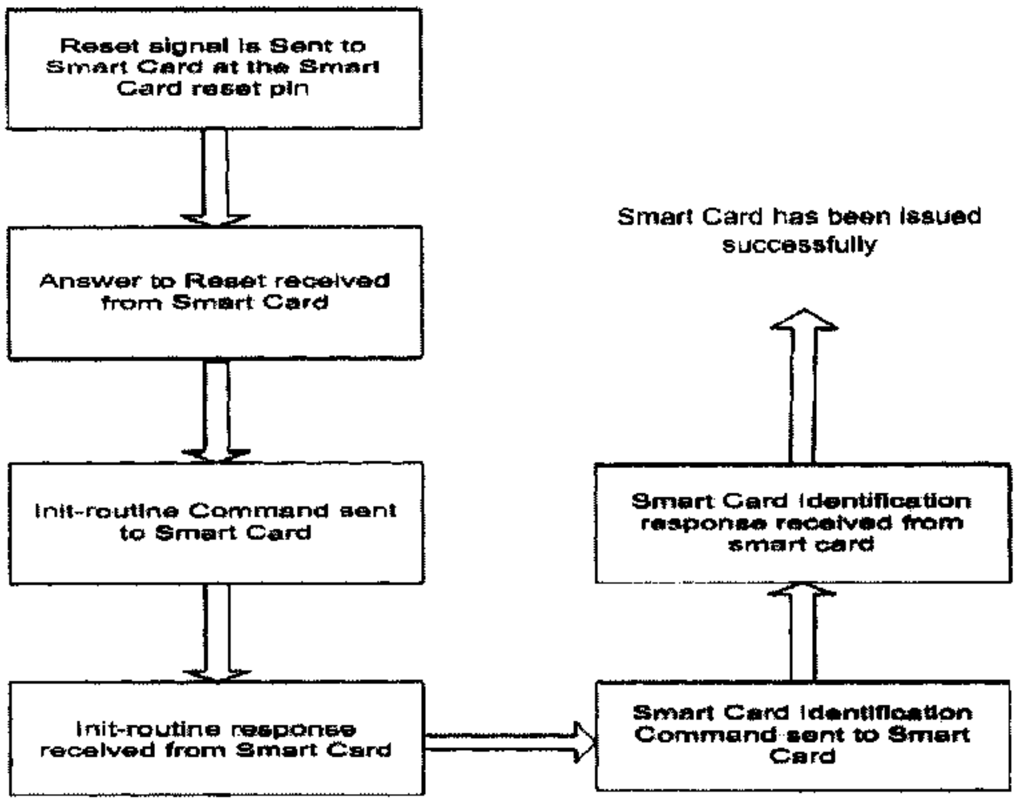

Reset signal is Sent to Smart Card at the Smart Card reset pin
Answer to Reset received from Smart Card
Init-routine Command sent to Smart Card
Init-routine response received from Smart Card
Smart Card Identification Command sent to Smart Card
Smart Card Identification response received from smart card
Smart Card has been issued successfully

Figure 29 Programming Smart Card

7.6 SMART CARD CONTROLLER

Smart Card Controller defines the drivers to read the smart card identification key with using smart card reader and PC. An interface board is designed to replace smart card reader and smart card development environment running on the PC. HDL is used to write the drivers which are embedded on FPGA.

7.6.1 BLOCK DIAGRAM

Smart card is inserted in the smart card socket which is fixed on the interface board. The interface board is connected with the FPGA through its GPIOs.

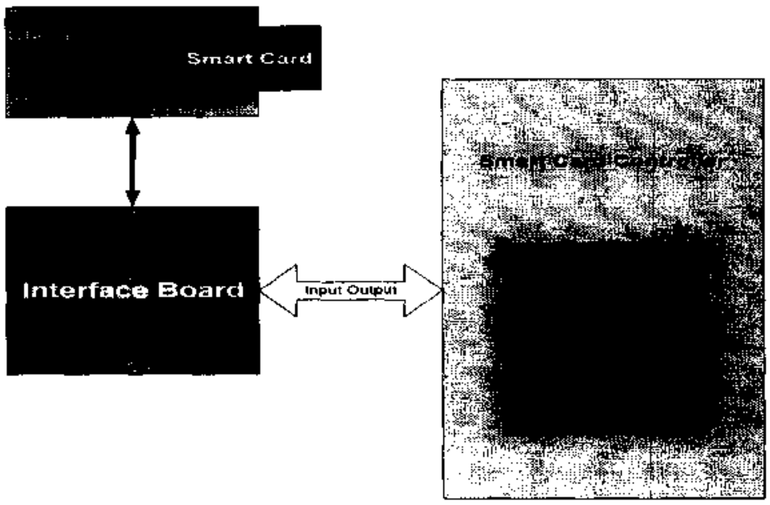


Figure 30 Block Diagram of Smart Card Controller

7.6.2 DESCRIPTION

When the smart card is inserted in the socket, smart card detection (logic High) signal is received by the smart card controller. The signal routes from interface board to a specific GPIO of the FPGA.

7.6.2.1 SMART CARD INITIALIZATION

To initialize communication, a logic high signal is to be sent to the RESET pin of the Smart Card. The signal routes from a specific GPIO of the FPGA to the interface board. In response to this RESET signal, Smart Card sends an Answer to Reset. After receiving the answer to reset, INIT command is sent to the Smart Card by the smart card controller. In response, Smart Card sends an Answer to Init response. This completes the smart card initialization process.

7.6.2.2 USER AUTHENTICATION

Read smart card identification command is send to the smart card by the smart card controller. And in response the smart card identification byte is received from the smart card. During the initialization of the system/ device the stored smart card identification byte is read from the memory. And it is compared with the one received from the smart card to authenticate that the user has the valid smart card. Whether the byte is matched or mismatched, the status is shown by the display center.

7.6.2.3 SMART CARD COMMUNICATION

The protocol used for communication with smart card is called **command and response** protocol.

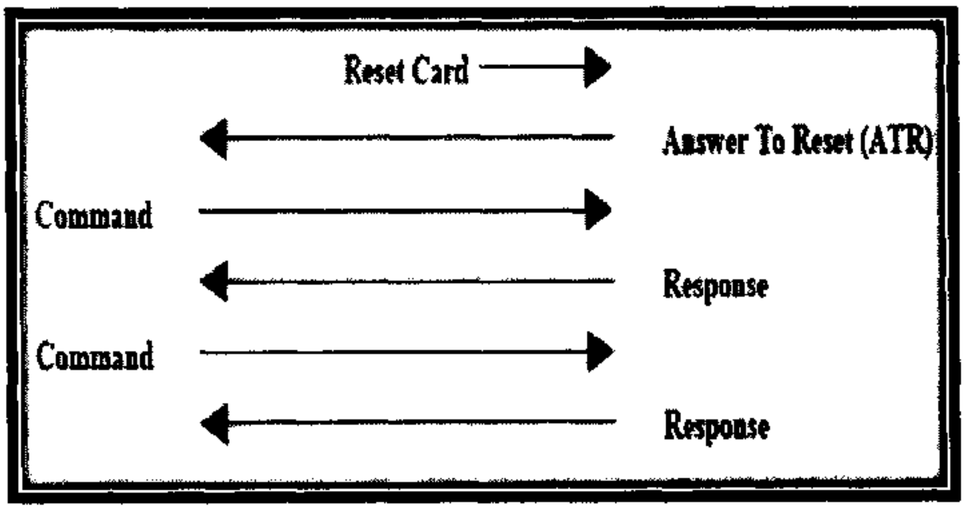


Figure 31 Smart card communication

7.6.2.4 COMMUNICATION PROTOCOLS OF SMART CARD

Smart card has bi directional contact with the outer world. This contact allows it to respond to the incoming command. It follows T0 and T1 protocols maintaining the standards 7816-3 and 7816-4 of the ISO, according to which the communication data rate is 9600 mpbs or more.

A command is defined in the card and programmed using the card reader. The command is then called from the FPGA. The FPGA is not only controlling all the commands but is also used for authenticating the responses against the different commands during its communication.

7.6.2.4.1 THE T0 PROTOCOL

This protocol is a standard protocol defined in the document ISO/IEC 7816-3. In this protocol the character are send to the integrated circuit cards for its response. The response is also character-based over the serial RS232 interface.

7.6.2.4.2 THE T1 PROTOCOL

This protocol is also a standard protocol defined in the document ISO/IEC 7816-3. In this protocol a block of characters is send to the integrated circuit cards for its response. The response is also a character-based block over the serial RS232 interface.

7.6.2.4.3 COMPARISION BETWEEN T0 AND T1

- T1 is faster, easier to use, and less error-prone.

- T0 protocol should be used if a pre-existing T0 command set is to be implemented or if the available that does not support the T1 protocol.

CHAPTER NO.8

DESIGN AND IMPLEMENTATION

8 DESIGN AND IMPLEMENTATION

8.1 DESIGN AND IMPLEMENTATIOP FLOW CHART

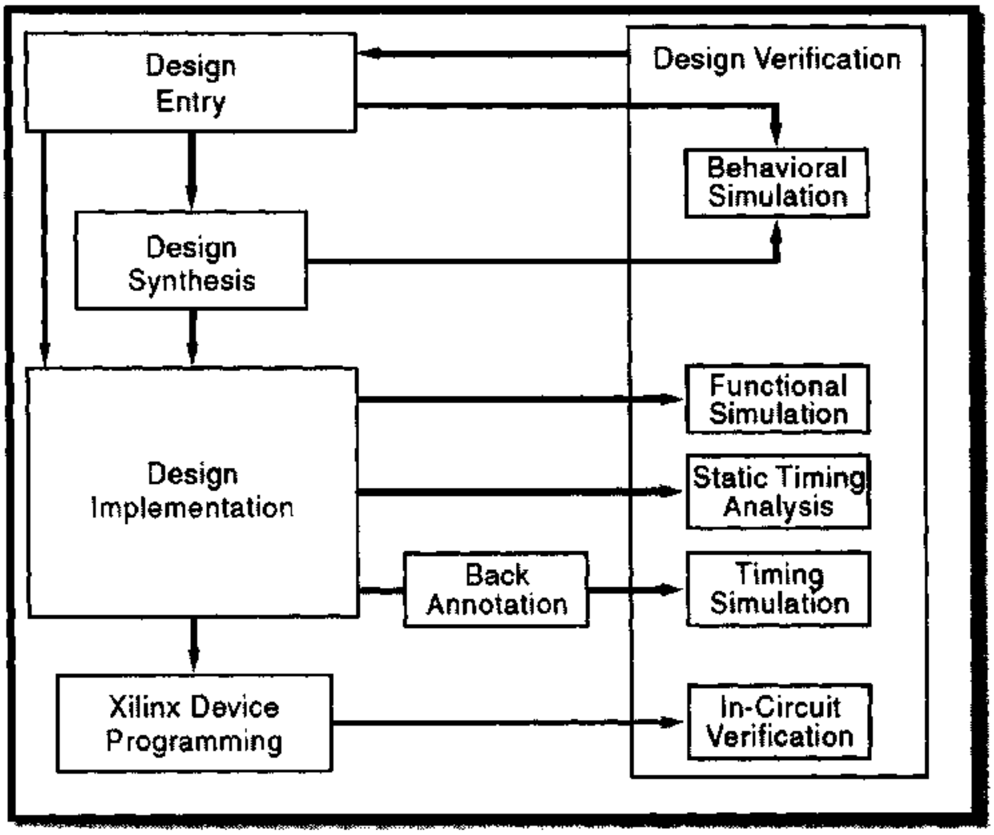


Figure 32 Flow chart for implementation of design using Xilinx ISE

8.2 SYSTEM ON CHIP DESIGN

The design phase consists of two stages. One is implementing the design using Verilog (Hardware Descriptive Language) as a hardware programming language in Xilinx Integrated Software Environment (ISE). The second is the design synthesization in Xilinx Integrated Software Environment (ISE).

8.3 DESIGN VERIFICATION

Model-Sim is used for debugging at different stages during the design verification phase of the project.

8.3.1 MODEL SIM AS DEBUGGING TOOL

Stimulus is written in Hardware Descriptive Language to examine the system. This test code comprises all the possible real scenarios which might occur during the communication process. And the output is examined to verify the system.

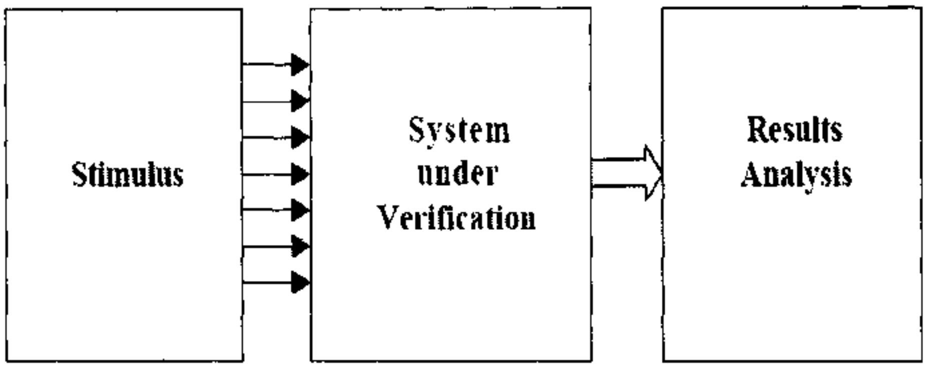


Figure 33 Debugging flow chart

Stimulus is written in verilog (hardware descriptive language). The design is instantiated in it. Stimulus should be intelligent to test all the possible cases. Model-sim is limited to the stimulus and is the best tool for hardware design verification. A snap shot of Model Sim based debugging is shown in the figure 34. The wave form gives the picture of the designed system. The output is thoroughly examined by numerous inputs.

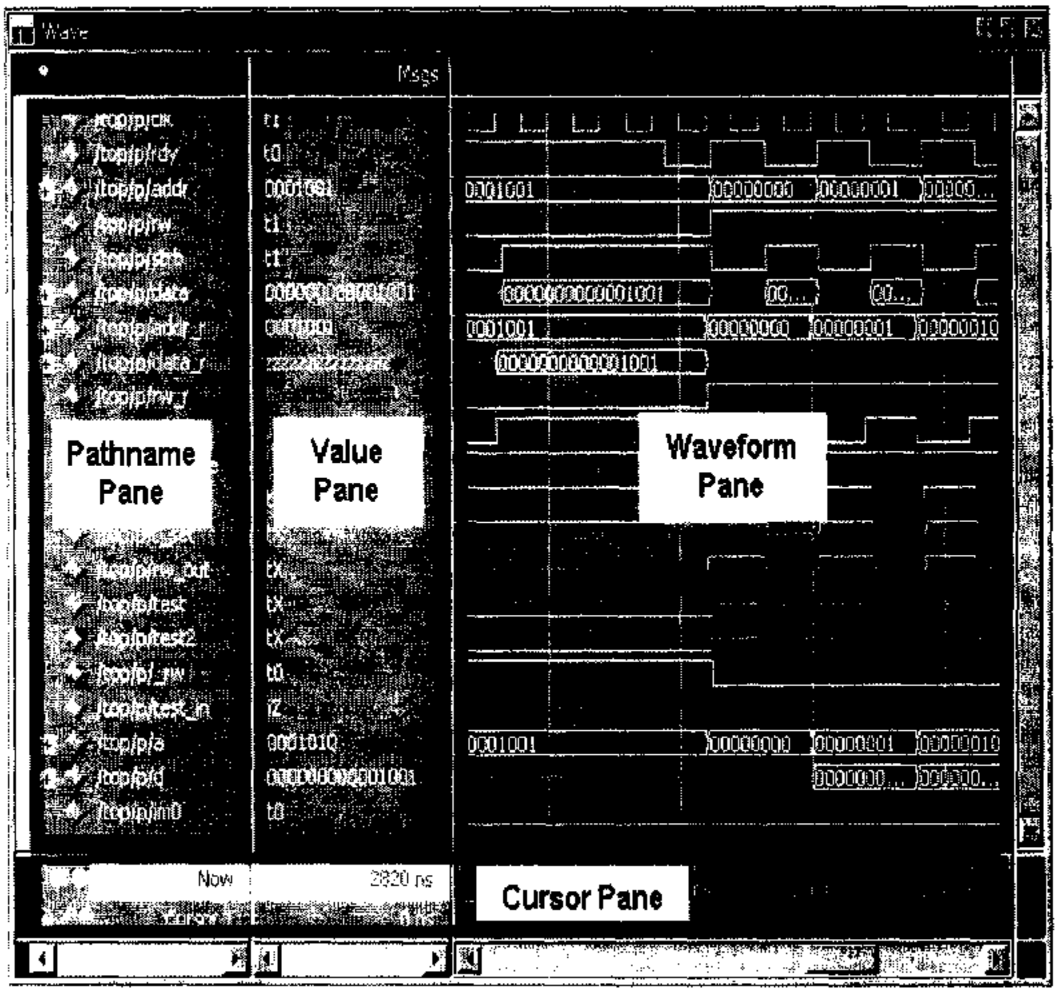


Figure 34 Debugging flow chart

8.4 DESIGN IMPLEMENTATION

After the design has been verified, Xilinx development kit is used to implement it.

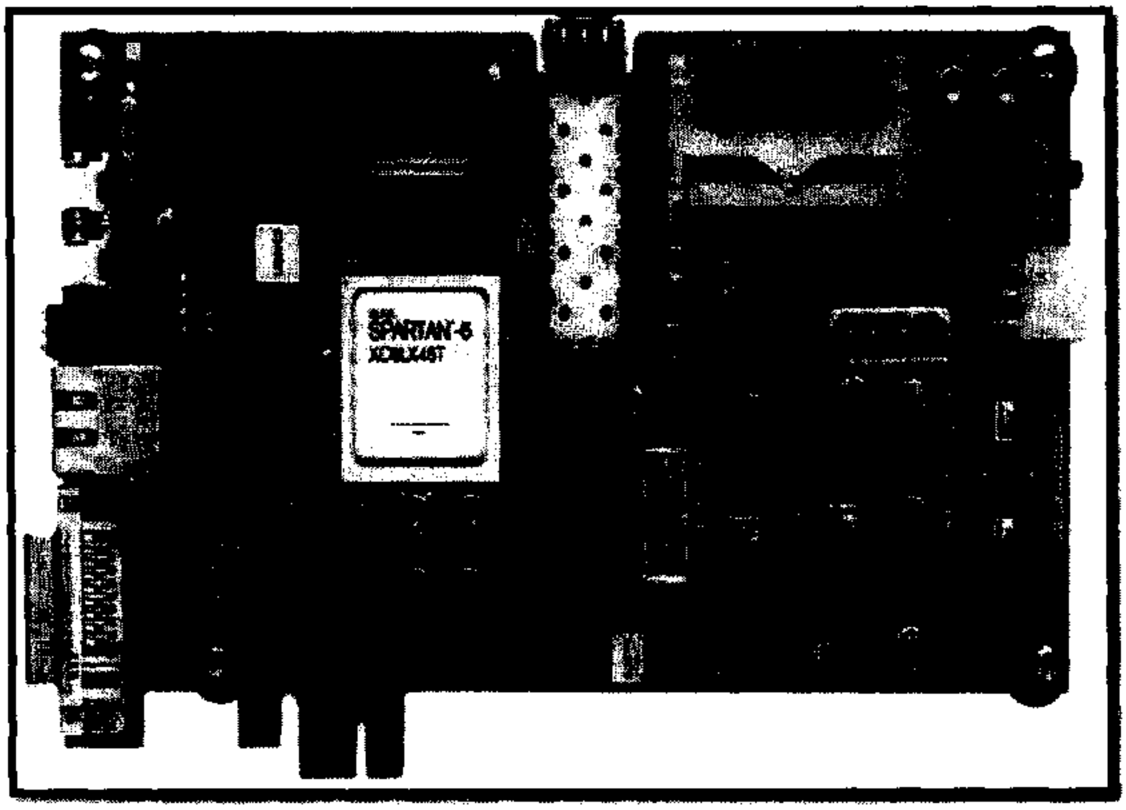
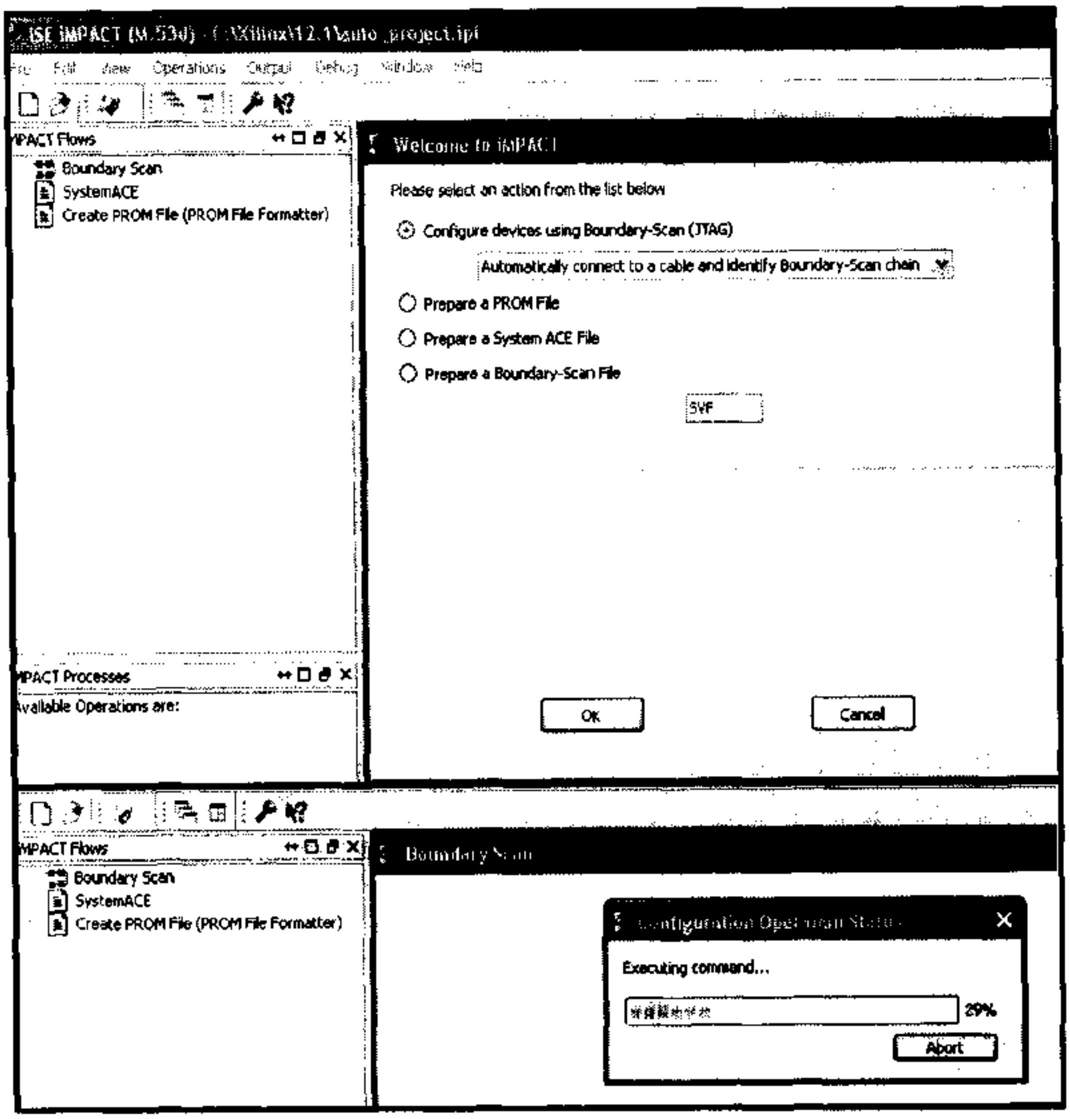


Figure 35 Xilinx development kit

The Impact tool of the Xilinx ISE is used to burn the actual design in Spartan 6 FPGA.


ISE iMPACT (M.53d) - C:\Xilinx\12.1\auto_project.ipf
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File Formatter)
iMPACT Processes
Welcome to iMPACT
OK
Cancel
Boundary Scan
Configuration Operation Status
Abort

Available Operations are:

Please select an action from the list below

[✓ Configure devices using Boundary-Scan (JTAG)]

Automatically connect to a cable and identify Boundary-Scan chain

Prepare a PROM File

Prepare a System ACE File

Prepare a Boundary-Scan File

SVF

Executing command...

29%

Figure 36 Xilinx ISE tool iMPACT

CHAPTER NO.9

REAL TIME SIGNAL AND DATA ANALYSIS

9 REAL TIME SIGNAL AND DATA ANALYSIS

9.1 TIMING ANALYSIS USING XILINX CHIP SCOPE PRO

Using Chip Scope Pro, functional simulations and timing analysis is done. It is possible only if the desired hardware is also available. During this process a binary file is generated and the Xilinx FPGA is configured with it. The project is implemented on Spartan 6 Field Programmable Gate Array (FPGA).

ChipScope Pro

Figure 37 Xilinx Chip Scope Pro as a debugging Tool

9.2 REAL TIME SIGNAL AND DATA OF FLASH

9.2.1 WRITE DATA CYCLE ANALYSIS

Install Xilinx Chip Scope Pro Analyzer on the PC. Connect the FPGA using its parallel/usb connection with the PC. The cable will show its status by lighting the LED as Green for proper connection.

Open Cable/ search JTAG device chain as shown in figure 38.

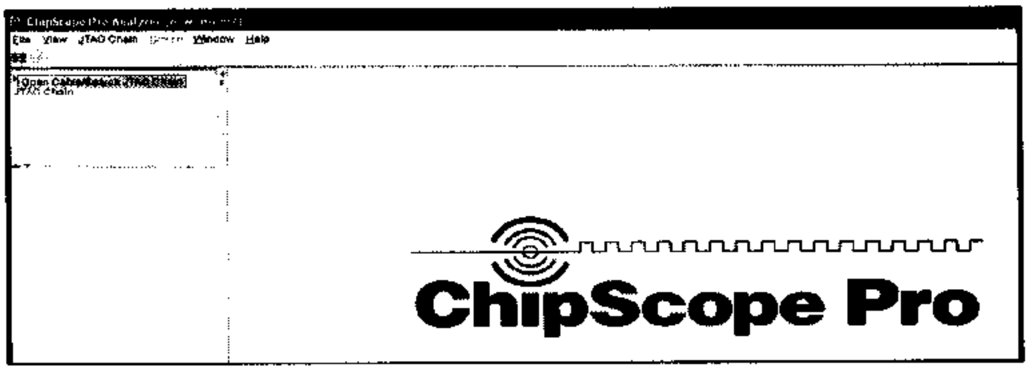
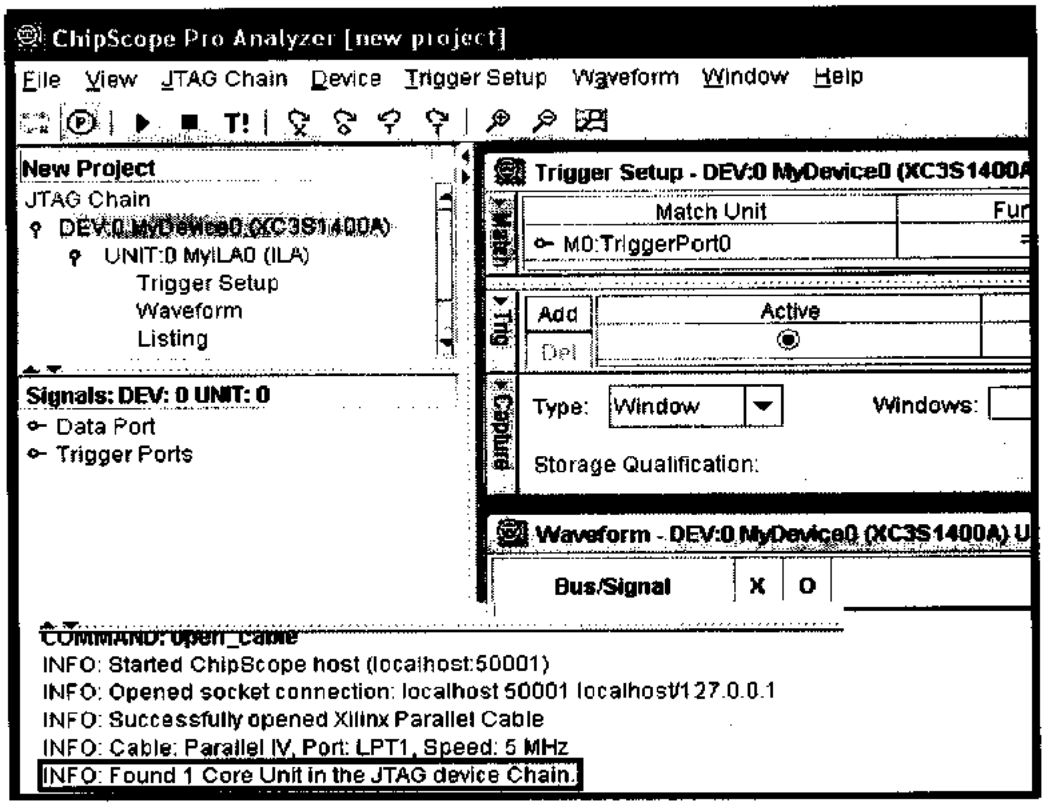

File View JTAG Chain Window Help
Open Cable/Search JTAG Chain
JTAG Chain
ChipScope Pro

Figure 38 Xilinx Chip Scope Pro environment

It will find the core unit in JTAG device chain as shown in figure 39.


ChipScope Pro Analyzer [new project]
File View JTAG Chain Device Trigger Setup Waveform Window Help
New Project
JTAG Chain
DEV:0 MyDevice0 (XC3S1400A)
UNIT:0 MyILA0 (ILA)
Trigger Setup
Waveform
Listing
Signals: DEV: 0 UNIT: 0
Data Port
Trigger Ports
Trigger Setup - DEV:0 MyDevice0 (XC3S1400A
Match Unit
M0:TriggerPort0
Add
Del
Active
Type: Window
Windows:
Storage Qualification:
Waveform - DEV:0 MyDevice0 (XC3S1400A) U
Bus/Signal
X
O

```
COMMAND: open_cable
INFO: Started ChipScope host (localhost:50001)
INFO: Opened socket connection: localhost 50001 localhost/127.0.0.1
INFO: Successfully opened Xilinx Parallel Cable
INFO: Cable: Parallel IV, Port: LPT1, Speed: 5 MHz
INFO: Found 1 Core Unit in the JTAG device Chain.
```

Figure 39 Xilinx Chip Scope Pro connection establishment

Configure the device by loading the bit file shown in figure 40.

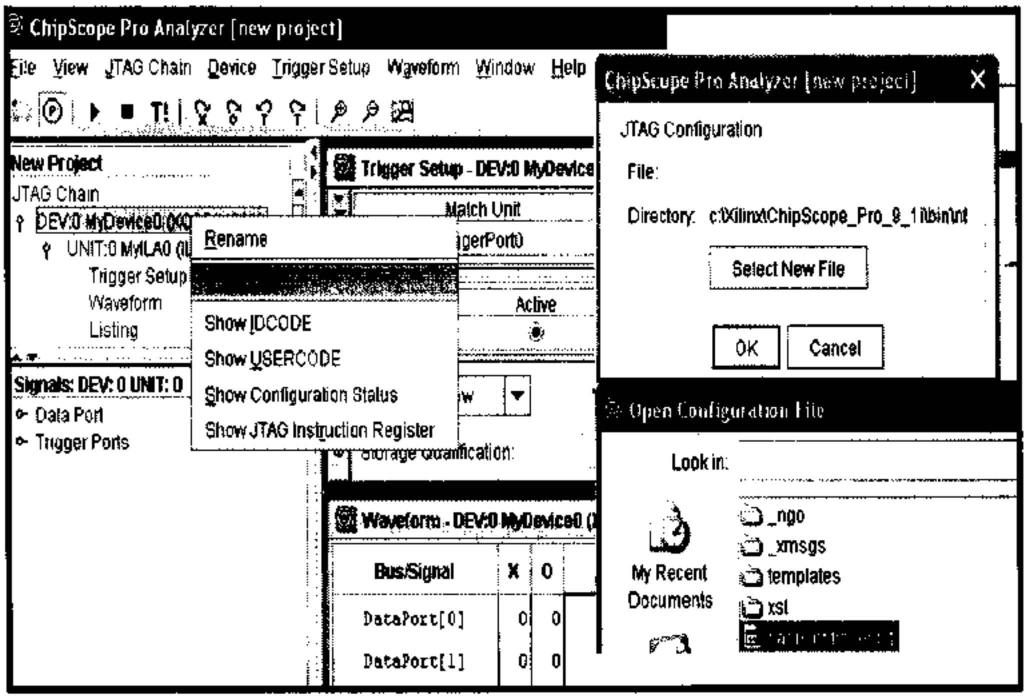


Figure 40 FPGA configuration using Xilinx Chip Scope Pro

Open the project for real time analysis purpose as shown in figure 41. Set the trigger to load the wave form.

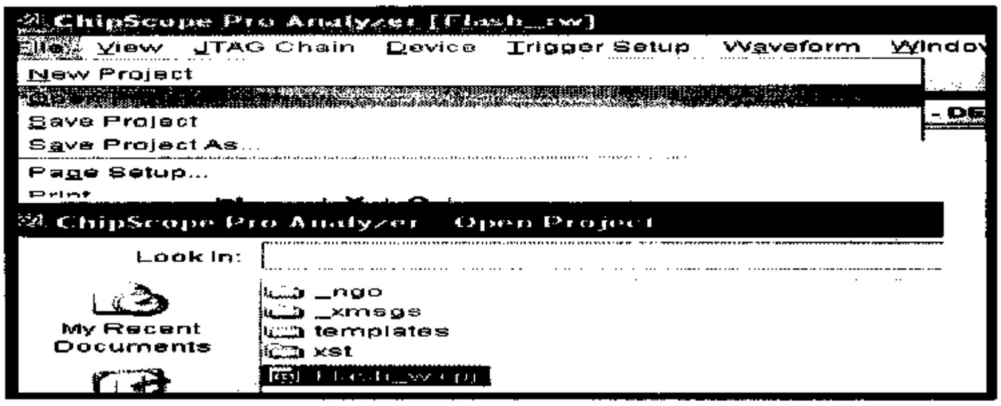

ChipScope Pro Analyzer [Flash_rw]
File View JTAG Chain Device Trigger Setup Waveform Windo
New Project
Save Project
Save Project As...
Page Setup...
Print
ChipScope Pro Analyzer Open Project
Look In:
My Recent Documents
_ngo
_xmsgs
templates
xst

Figure 41 Xilinx Chip Scope Pro trigger setup

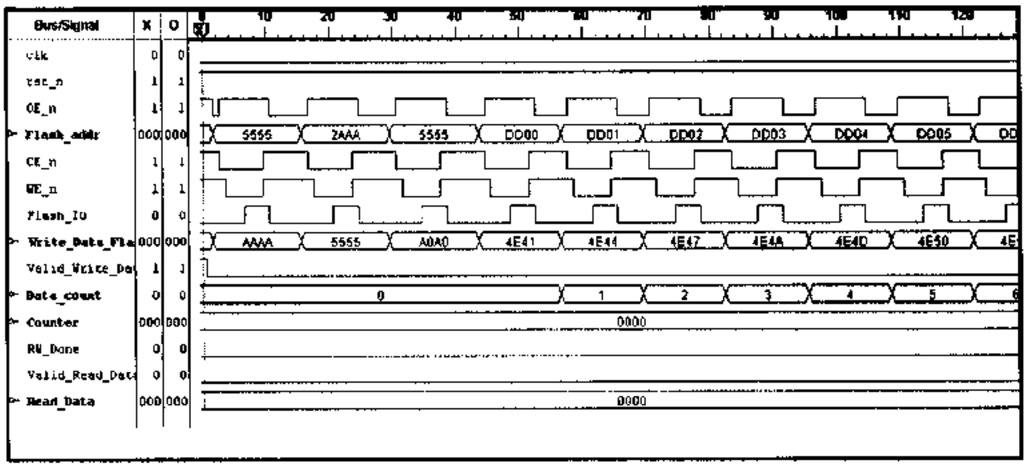


Figure 42 Real time signal analysis for writing data into Flash

9.2.2 READ DATA CYCLE ANALYSIS

Follow the first three steps as mentioned in article 9.2.1. Configure the device by loading the bit file. Open the project and set the trigger to load the wave form. Figure 43 shows the wave form for reading data from the flash.

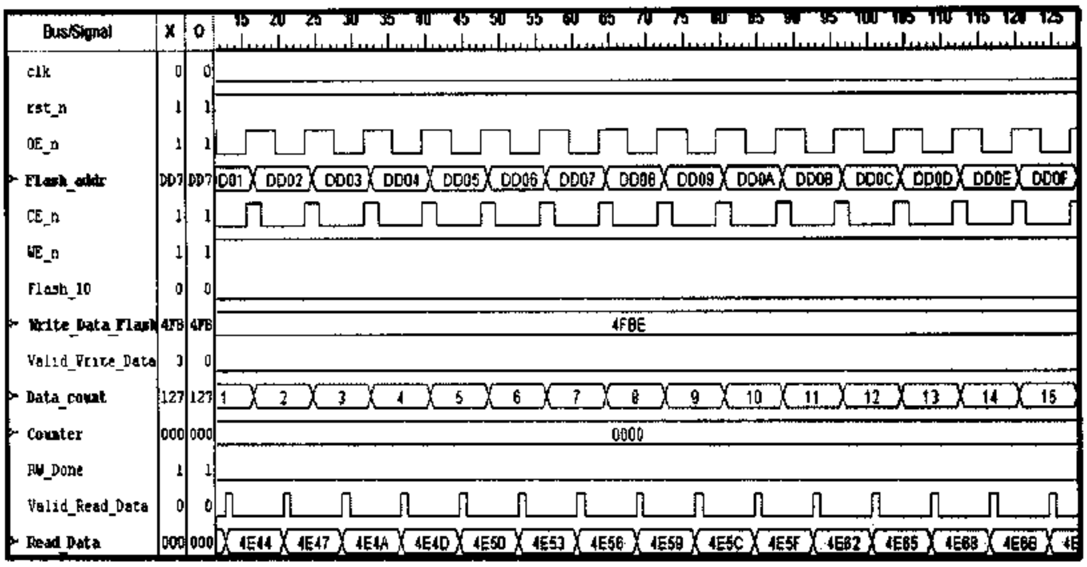


Figure 43 Real time signal analysis for reading data from the Flash

9.2.3 CRITICAL PARAMETERS

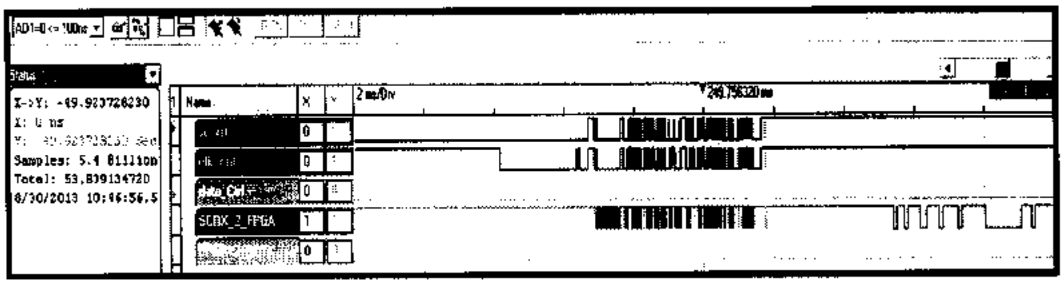
Clock	=	24.576 MHz
Sector Data Writing Time	=	20 ms
Sector Data Reading Time	=	0 sec

9.3 REAL TIME SIGNAL AND DATA OF FLASH

The four Smart card control signals are, Smart card detection signal, Smart card reset signal, Smart card clock control signal and Smart card data control signal. The two Smart card data signals are, Smart card data input signal and Smart card data output signal. For real time smart card data and control signal analysis the debugging tools used are DigiView Logic Analyzer and Xilinx Chip Scope-Pro.

9.3.1 TIMING ANALYSIS USING LOGIC ANALYSER

The figure 44 shows the real time data captured over serial interface while communicating with the smart card.



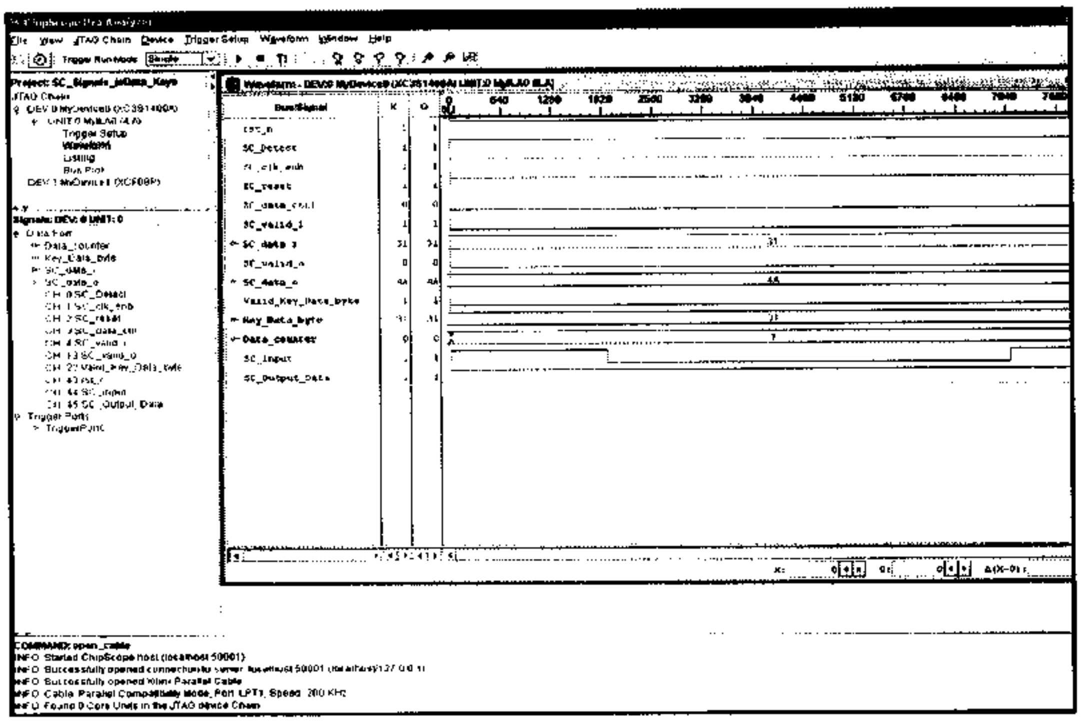
```
X->Y: -49.923726230
X: 0 ns
Samples: 5.4 Billion
Total: 53,839134720
8/30/2013 10:46:56.5
```

Digital view of smart card data

Figure 44 DigiView Logic Analyzer real time testing

9.3.2 TIMING ANALYSIS USING CHIP SCOPE PRO

Real time data and signal analysis on smart card command and response communication using Xilinx chip scope pro is shown in figure 45.



```
COMMAND: open_cable
INFO Started ChipScope host (localhost:50001)
INFO Successfully opened connection to server localhost:50001 (localhost/127.0.0.1)
INFO Successfully opened Xilinx Parallel Cable
INFO Cable: Parallel Compatibility Mode, Port: LPT1, Speed: 200 KHz
INFO Found 0 Core Units in the JTAG device Chain
```

Figure 45 Wave form for smartcard using Xilinx Chip Scope Pro

CHAPTER NO.10

CONCLUSION

10 CONCLUSION

In this research, endeavor is to develop an easily configurable Field Programmable Gate Array based customized solution for secure pear to pear communication link. Universal asynchronous serial communication link is the interface at both the ends. Link is synchronized by setting baud rate, parity, data bits etc. to make the communication unswerving. Algorithms are implemented to control the level of security of the data in an independent manner.

Xilinx integrated Software Environment provides the best practical environment for hardware description using combinational as well as sequential logic in hardware descriptive language Verilog. Using "Chip scope pro" the design has been tested and verified. The design is reconfigured by reprogramming any module and downloading it in FPGA to get the desired application output.

The FPGA based system can be used for data rates between 460800 mbps to 110 mbps. Whether the data is block cipher or stream cipher, the level of security is adjustable depending upon the algorithm.

CHAPTER NO.11

FUTURE WORK

11 FUTURE WORK

The system can be composed to transfer data at higher data rates. The data acquisition is done in parallel mode using field programmable gate arrays. It is only possible by modifying the hardware. Moreover the design complexity will also increase. Hence, the data rate will increase at the cost of design complexity and enhancement of the hardware.

The project can be divided into different modules. Top controller will be controlling the whole design. The main task is to implement the UART Controller. Data acquisition from the serial communication links is a standalone job. Further algorithms can be implemented in order to minimize the delays.

CHAPTER NO.12

BIBLOGRAPHY

BIBLOGRAPHY

[1] Pete Warnes, Choosing FPGA or DSP for your Application: [Online]: http://www.hunteng.co.uk/info/index.htm

[2] Susannah Martin, Speeding digital signal processing solutions with FPGAs: Senior Design Engineer, Xilinx Inc., San Jose, Calif. [Online]: http://www.eetimes.com

[3] FPGA Soft Processor Design Considerations. [Online] http://www.design-reuse.com

[4] Michael Haselman, Robert Miyaoka, Thomas K.Lewellen, Scott Hauck, "FPGA-Based Data Acquisition System for a Positron Emission Tomography (PET) Scanner", Seattle, West America: University of Washington.

[5] Kole D.K., Rahaman H.. "Implementation of AES Algorithm in UART Module for Secured Data Transfer", International Conference on Advances in Computing and Communications (ICACC), 2012: Cochin, Kerala India.

[6] Wael M El Medany, "Serial Data Path Implementation on a Xilinx Spartan 3 FPGA for Global Positioning System." Fayoum University, Egypt, and University Of Bahrain, Bahrain.

[7] Choosing serial interfaces for high speed ADCs in medical apps: [Online]: www.eetimes.com/design/medical-design/4402426/Choosing-serial-interfaces-for-high-speed-ADCs-in-medical-apps

[8] Ken Eguro, "Automated Dynamic Reconfiguration for High-Performance Regular Expression Searching": International Conference on Field-Programmable Technology, December 2009.

[9] Radi H.R., Vivian L.W.B., M.N.Shah Zainudin., M.Muzafar Ismail, "FPGA based Global Positioning System.": International Journal of Electrical& Computer Sciences IJECS-IJENS Vol: 12 No: 05

[10] Peter J Mumford, Kevin Parkinson, Andrew G Dempster, "Open GNSS Receiver Platform": University of New South Wales, New South Wales, Australia.

[11] Ching-Chang Wong & Yu-Han Lin, "A Reusable UART IP Design and its Application in Mobile Robots", Tamkang University, Taipei, Taiwan

[12] ITU-R Recommendation M.1677: International Morse Code, May 2004, International Telecommunication Union, Geneva, Switzerland.