DESIGN OF EMBEDDED CONTROLLER FOR SAFETY CRITICAL SYSTEM

Summer Internship Report Submitted in partial fulfillment

of the requirement for under graduate degree of

Bachelor of Technology

In

Electronics and Communication Engineering

By

Karthik M.B

2210416132

Under the Guidance of

Mr. K. SATHISH, M.Tech

Assistant Professor

Department Of Electronics and Communication Engineering

GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

May 2019

DECLARATION

I submit this summer internship work entitled **"Embedded controller for safety critical system"** to **GITAM** (Deemed to be University) in partial fulfillment of the requirements for the award of Bachelor of Technology in Electronics and Communication engineering. I declare that this work is carried out by me with the help of my team members under the guidance of **MR. RAJA SINGH** (SC 'G') Scientist Brahmos laboratories DRDO, Kanchanbagh, Hyderabad.

The results embodied in this report are not submitted to any other University or Institute for award of any degree or diploma.

PLACE: HYDERABAD **Karthik M.B**

DATE:02/08/2019 2210416132

GITAM (Deemed to be University)

Hyderabad-502329, India

Dated: 2nd August 2019

CERTIFICATE

This is to certify that the summer internship report entitled **"DESIGN OF EMBEDDED CONTROLLER FOR SAFETY CRITICAL SYSTEM"** being submitted by **HM. Vishal Raj (2210416122), Ch. Sai Sumith (2210416115), T. Sai Teja (2210416162), A. Shivaji Rao(2210416101), K. Abbhinav (2210416127), Karthik M.B(2210416132)** for partial fulfillment of the requirement for the reward of **Bachelor of technology in Electronics and communication Engineering** to GITAM (Deemed to be University), Hyderabad campus during the academic year 2019-2020 is a record of work, undertaken by them under the supervision of the undersigned.

Mr. K. Sathish Chari
Assistant Professor
Department of ECE

Dr. K. Manjunatha
Professor and HOD
Department of ECE

Dr. G. RAJA SINGH THANGADURAI
SCIENTIST 'G'
Dy. PROGRAMME DIRECTOR, PJ -10

GOVERNMENT OF INDIA
MINISTRY OF DEFENCE
DEFENCE R&D ORGANISATION
DEFENCE RESEARCH AND DEVELOPMENT LABORATORY
P.O. Kanchanbagh, Hyderabad - 500 058
T.S., INDIA.

CERTIFICATE

This is to certify that **Mr. M.B. KARTHIK (Reg. No. 2210416132),** third year student in Electronics & Communication Engineering from **GITAM University, Hyderabad,** has completed his Summer Internship from 06th May 2019 to 15th June 2019, at DRDL, Hyderabad.

He has done his internship on **"Design of Embedded Controller for Safety of Critical Systems".** He also visited various work centers of Missile Development Laboratory to acquaint himself with the current technologies and simulations being used being used in the state of the art Missile Integration Complex.

During the entire course of Internship, his performance was good and his passion to learn new technologies was appreciable.

Place: Hyderabad
Date: 17.06.2019

(Dr. G Raja Singh), Sc "G"
Dr. G. Raja Singh Thangadurai
Dy. Programme Director, PJ-10
DRDL, Ministry of Defence
Hyderabad-500 058

MINISTRY OF DEFENCE * Defence Research & Development Laboratory * Kanchanbagh P.O. * HYDERABAD-500 058 *

Phone : 040-2408 7029, 2408 7030 Fax : 040 - 2408 7006 e-mail : grsingh@drdl.drdo.in

ACKNOWLEDGEMENT

Apart from our effort the success if this project largely depends on the encouragement and guidance of many others. We take this opportunity to express our gratitude to the people who helped us in the successful completion on this project.

We would like to thank respected **MR. RAJA SINGH** (SC 'G') sir defense research and is the as and development laboratories, Dwarakamai nagar, Kanchan bagh, Hyderabad. For their constant support in the Design of the project named **"DESIGN OF EMBEDDED CONTROLLER FOR SAFETY CRITICAL SYSTEM"** as part of our BTECH, Electronics and Communication curriculum.

We strongly record our deep sense of gratitude and thankfulness with outmost respect to our is an project guides **Mr. PRANITH** (SCIENTIST) and **Mr. PRAVEEN KUMAR AGM** (FCS) and the professors for providing the valuable information and help without which this project would not have been a success.

I would like to thank to respected faculties **Mr. K. Sathish** who helped me to make this seminar a successful accomplishment.

We are thankful to our friends and departments to make our project in more organized way and well stacked till the end.

Karthik M.B

2210416132

ABSTRACT

Universal Asynchronous Receiver Transmitter (UART) is usually an individual integrated circuit used for serial communication over a computer or peripheral device serial port. This UART is designed for making an interface between RS232 line and a microcontroller or an IP core. It works fine when connected to a serial port of a PC for data exchange with custom electronic. UARTs are now commonly included in microcontrollers.

A UART is a full duplex transmitter or receiver. UART is the kind of serial communication protocol. In parallel communication the cost as well as complexity of the system increases due to simultaneous transmission of data bits on multiple serial communications alleviates this drawback of parallel communication and emerges effectively in many applications for long distance communication as it reduces the signal distortion because of its simple structure.

TABLE OF CONTENTS:

LIST OF FIGURES

ABBREVATIONS USED

UART: Universal asynchronous receiver transmitter

DRDO: Defence research and development organisation.

DRDS: Defence research and development service

TXD: Transmitter

RXD: Receiver

RS232: Recommended standard 232

IP: Internet protocol

PC: Personal computer

LSB: Least significant bit

MSB: Most significant bit

FPGA: Field programmable gate array

ASM: Algorithmic state machine

RTL: Register transfer level

FIFO: First in first out

LCR: Liquidity coverage ratio

CHAPTER 1

1.1 DRDO:

The Defence Research and Development Organisation (DRDO) is an agency of the Government of India, charged with the military's research and development, headquartered in New delhi, was formed in 1958 by the merger of the Technical Development Establishment and the Directoratate of Technical Development and Production with the Defence Science Organisation. It is under the administrative control of the Ministry of Defence, Government of India.

With a network of 52 laboratories, which are engaged in developing defence technologies like in the various fields, aeronautics, armaments, electronics, landcombat engineering, life and India's largest and most diverse research organisation. The organisation includes around 5,000 scientists belonging to the Defence Research & Development Service (DRDS) and about 25,000 scientific, technical and supporting personnel.

1.2 Vision:

Make India establishing world-class science and technology base provide our Defence Services decisive edge by equipping competitive systems and solutions.

1.3 Mission:

Design develop and lead to production state-of-the-art sensors, weapon systems, platforms and allied equipment for our Defence Services.

Develop infrastructure and committed quality manpower and build strong technology base.

1.4 INTRODUCTION:

Asynchronous serial communication has advantages of less transmission line, high reliability, and long transmission distance , therefore is widely used in data

exchange between command and peripherals. Asynchronous serial communication is usually by Universal Asynchronous Receiver Transmitter (UART).

UART allows fullduplex communication in serial link, thus has been widely used in the data communications and control system. In actual applications, usually only a few key features of UART are needed.

Specific interface chip will cause waste of resources and increased cost. Particularly in the field of electronic design, SOC technology is recently becoming increasingly mature. This situation results in the requirement of realizing the whole system function in a single or a very few chips. Basic UART communication needs only two signal lines (RXD, TXD) to complete full-duplex data communication.

TXD is the transmit side, the output of UART; RXD is the receiver, the input of UART. UART basic features are there are two states in the signal line, using logic 1 (high) and logic 0 (low) to distinguish respectively. For example, when the transmitter is idle, the data line is in the high logic state.

Otherwise when a word is given to the UART for asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted. The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter.

These two clocks must be accurate enough to not have the frequency drift by more than 10% during the transmission of the remaining bits in the word. After the Start Bit, the individual databits of the word are sent, with the Least Significant Bit (LSB) being sent first. This transmitted is for exactly the same amount of time as all of the other bits, and the receiver loop approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0.

For example, if it takes two seconds to send each bit the receiver will examine the signal to is to be determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on.

When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates. When the receiver has received all of the bits in the data word, it may check for the Parity Bits (both sender and receiver must agree on whether a Parity Bit is to be used), and then the receiver looks for a Stop Bit.

If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be garbled and will report a Framing Error to the host processor when the data word is read. The of usual cause of a Framing Error is that the sender and receiver clocks were not running at the is a same speed, or that the signal was interrupted. Regardless of whether the data was received as a the correctly or not, the UART automatically discards the Start, Parity and Stop bits. If the bit of sender and receiver are configured identically, these bits are not passed to the host.

If another word is ready for transmission, the Start Bit for the new word can be sent as soon as is the Stop Bit for the previous word has been sent. Because of "self-synchronizing", if there are no data to transmit, the transmission line can be of idle.

CHAPTER 2

1.1 UART:

A UART is sometimes referred to as a Serial Port, RS-232 Interface, or COM Port. It's one of the simplest methods of communication with your FPGA. Other methods communication you might have heard of are PCI, PCI-Express, USB, etc, but the Go Board uses a UART is of the because it's the easiest and best to learn with.

The UART consists of a receiver component and a transmitter component. For the UART of the Receiver portion, we need to take in the serial data from the computer. This is sent by the of its computer one bit at a time. The receiver converts it back to the original byte.

This is a conversion of serial data to parallel data. 12 The transmitter works in the opposite way. It sends out one byte at a time across a single wire. A byte is 8-bits wide, so in order to send and byte over a single wire, you need to convert the data from parallel data to serial. This is what we will be doing in the UART Transmitter decided to break up the UART into two parts because it's the most complicated project thus far.

1.1.1 UART Configuration Parameters:

A UART has several parameters that can be set. The transmitter and the receiver need to agree on the settings below or data corruption occurs. Let's discuss the UART parameters that are if an settable: Baud Rate (9600, 19200, 115200, others) Number of Data Bits (7, 8) Parity Bit (On, an Off) Stop Bits (0, 1, 2) Flow Control (None, On, Hardware) Baud Rate is the rate at which the if the serial data is transmitted. 9600 Baud means 9600 bits per second.

Number of Data Bits is in almost always set to eight. A Parity Bit can be appended after theis a to be data is sent, to know if the data was received correctly or not. A Stop Bit always set to 1, and there can be 0, 1, or 2 Stop Bits. Flow Control is not typically used in present day is to set a applications and will likely be set to None.

1.2 Project Description:

Create a UART Receiver that receives a byte from the computer and displays it on the 7-bit an Segment Displays. The UART receiver should operate at 115200 baud, 8 data bits, no parity,1 stop bit, no flow control. Here are the settings that you will be using for this project: Baud Rate 9600 Number of Data Bits 8 Parity Bit Off Stop Bits 1 Flow Control None.

1.3 UART Serial Data Stream:

The UART receiver will use a state machine to keep track of the data being received. The first of receiver first looks for the falling edge of the start bit. This indicates that a byte is to transmitted. It then waits for half of a bit period to align to the center of the UART data. The reason to use the center of the bits is that this is when you're least likely to see transitions, and most likely to get a on good sample of the UART data. Once the receiver is aligned to the center, it samples the data on the line and stores each bit into a byte register. It increments an index to keep track of which bit is being received. Once all 8 data bits are received, it receives a stop bit, then returns to the on IDLE state to wait for the next data byte.

The UART data stream and go board of UART are shown in figure 1 and figure 2 respectively.

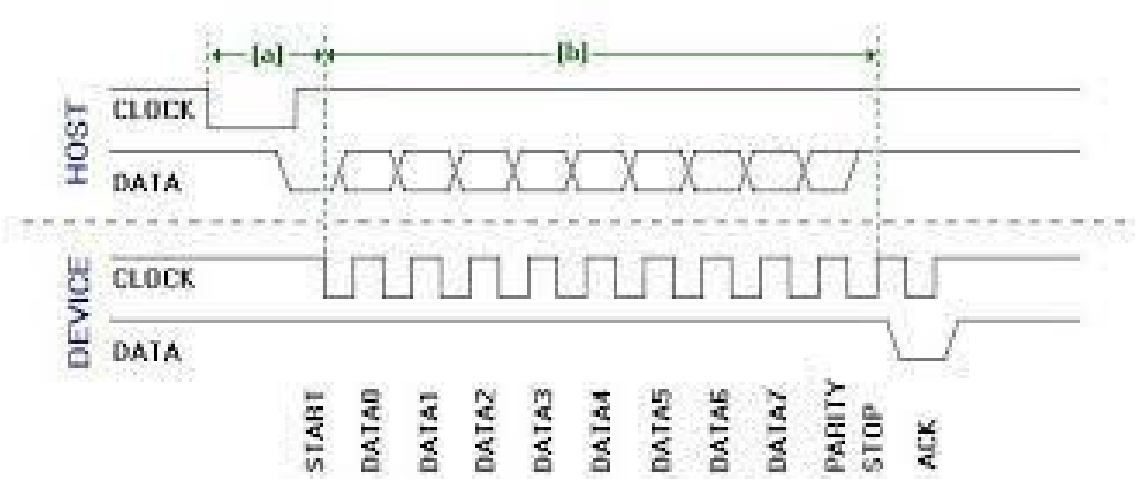

Figure 1: Serial data stream of UART

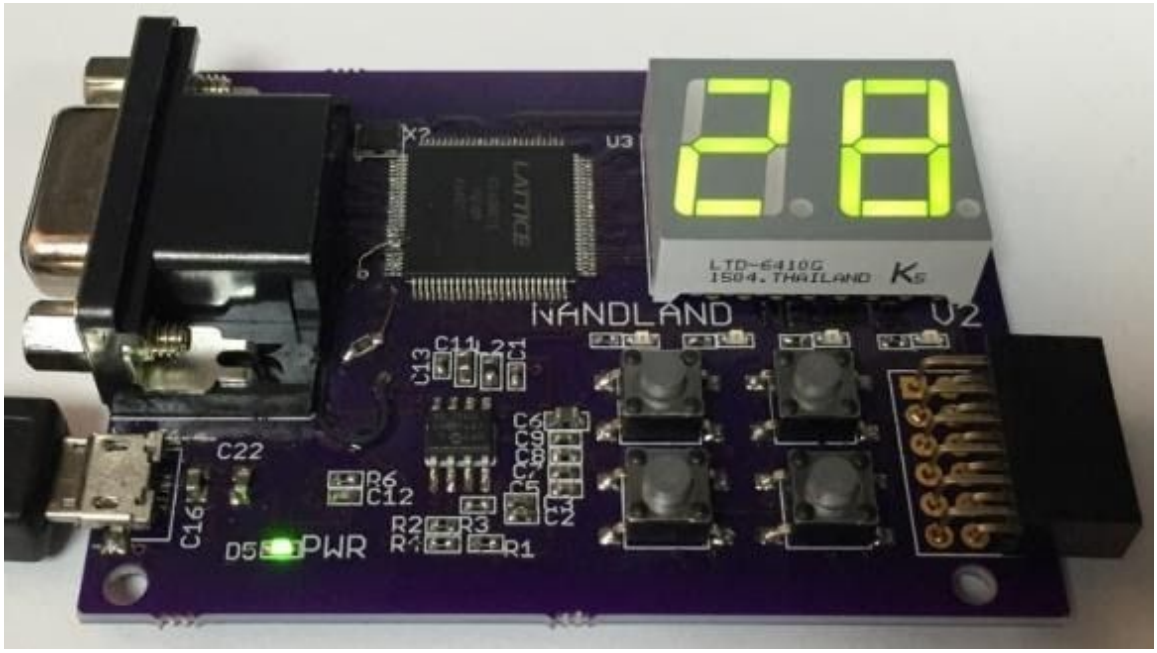

Figure 2: Go board of UART

CHAPTER 3

2.1 ASM CHART:

1. When the UART detects a start bit, it reads in the remaining bits serially and shifts them into the RSR.
2. When all the data bits and the stop bit have been received, the RSR is loaded into the RDR, and the Receive Data Register Full flag (RDRF) in the SCSR is set.
3. The microcontroller checks the RDRF flag, and if it is set, the RDR is read and the flag is to be cleared.

CHAPTER 4

3.1 RECEIVER VERILOG CODE:

```
module UART_RX
 (
 input i_Clock,
 input i_RX_Serial,
output o_RX_DV,
output [7:0] o_RX_Byte
 );
 parameter CLKS_PER_BIT = 217;
 parameter IDLE = 3'b000;
parameter RX_START_BIT = 3'b001;
parameter RX_DATA_BITS = 3'b010;
 parameter RX_STOP_BIT = 3'b011;
 parameter CLEANUP = 3'b100;
reg [7:0] r_Clock_Count = 0;
reg [2:0] r_Bit_Index = 0;
reg [7:0] r_RX_Byte = 0;
regr_RX_DV = 0;
reg [2:0] r_SM_Main = 0;
always @(posedgei_Clock)
begin
case (r_SM_Main)
IDLE :
16 begin r_RX_DV<= 1'b0;
 r_Clock_Count<= 0;
r_Bit_Index<= 0;
if (i_RX_Serial == 1'b0) r_SM_Main<= RX_START_BIT);
else
 r_SM_Main<= IDLE;
end
 RX_START_BIT :
 begin
 if (r_Clock_Count == (CLKS_PER_BIT-1)/2)
```

```
begin
 if (i_RX_Serial == 1'b0)
 begin
r_Clock_Count<= 0;
r_SM_Main<= RX_DATA_BITS;
end
 else
r_SM_Main<= IDLE;
end
 else
 begin
r_Clock_Count<= r_Clock_Count + 1);
r_SM_Main<= RX_START_BIT;
 end
 end
 RX_DATA_BITS
 begin
if (r_Clock_Count< CLKS_PER_BIT-1)
 begin
r_Clock_Count<= r_Clock_Count + 1;
r_SM_Main<= RX_DATA_BITS;
end
 else
 begin r_Clock_Count<= 0;
r_RX_Byte[r_Bit_Index] <= i_RX_Serial;
if (r_Bit_Index< 7)
begin
r_Bit_Index<= r_Bit_Index + 1;
r_SM_Main<= RX_DATA_BITS;
 end
 else
begin r_Bit_Index<= 0;
r_SM_Main<= RX_STOP_BIT;
end
 end
 end
```

```
RX_STOP_BIT : 18
Begin
 if (r_Clock_Count< CLKS_PER_BIT-1)
 begin
r_Clock_Count<= r_Clock_Count + 1;
r_SM_Main<= RX_STOP_BIT;
 end
else
 begin
r_RX_DV<= 1'b1;
r_Clock_Count<= 0;
r_SM_Main<= CLEANUP;
 end ; end
 CLEANUP :
begin
r_SM_Main<= IDLE;
r_RX_DV<= 1'b0;
end
default :
r_SM_Main<= IDLE;
 end
case
 end
 assign o
_RX_DV = r_RX_DV;
 assign o_RX_Byte = r_RX_Byte;
endmodule
```

3.2 RECEIVER TESTBENCH :

```
module UART_RX_TB();
 parameter c
CLOCK_PERIOD_NS = 40;
 parameter c_CLKS_PER_BIT = 217;
 parameter c_BIT_PERIOD = 8600;
regr_Clock = 0;
regr_RX_Serial = 1;
 wire [7:0] w_RX_Byte;
 task UART_WRITE_BYTE;
 input [7:0] i_Data;
 integer i;
 begin r_RX_Serial<= 1'b0;
 #(c_BIT_PERIOD);
#1000;
 for (ii=0; ii<= i_Data[i];
#(c_BIT_PERIOD);
end
r_RX_Serial<= 1'b1;
 #(c_BIT_PERIOD);
end
endtask
UART_RX
 #(.CLKS_PER_BIT(c_CLKS_PER_BIT))
 20 (.i_Clock(r_Clock),
.i_RX_Serial(r_RX_Serial),
.o_RX_DV(),
 .o_RX_Byte(w_RX_Byte) );
always
 #(c_CLOCK_PERIOD_NS/2) r_Clock<= !r_Clock);
initial
begin @(posedger_Clock);
UART_WRITE_BYTE(8'b00010101);
 @(posedger_Clock);
 if (w_RX_Byte == 8'b00010101)
```

```
$display(“Test Passed – Correct Byte Received”);
 else
$display(“Test Failed – Incorrect Byte Received”);
 $finish();
end
endmodule
```

CHAPTER 5

4.1 RTL SCHEMATIC :

RTL Coding **RTL** is an acronym for register transfer level. This implies that your **VHDL** code describes how data is transformed as it is passed from register to register. The transforming of the data is performed by the combinational logic that exists between the registers.

In simple terms **RTL design** or Register Transfer Level **design** is a method in which we can be of transfer data from one register to another. OR. Constructing a digital **design** using the of an Combinational and Sequential circuits in HDL like Verilog or VHDL which can model logical if and hardware operation.

In digital circuit **design**, register-transfer level (**RTL**) is a **design** abstraction which models a of the synchronous digital circuit in terms of the flow of digital signals (data) between hardware is a registers, and the logical operations performed on those signals.

4.2 TECHNOLOGY SCHEMATIC :

In telecommunication and data transmission, **serial communication** is the process of the and of sending data one bit at a time, sequentially, over a communication channel or computer bus. This is in contrast to parallel communication, where several bits are sent as a whole; on a link with is several parallel channels.

Serial communication is used for all long-haul communication and most computer networks, of where the cost of cable and synchronization difficulties makes parallel communication and one is impractical.

Serial computer buses are becoming more common even at shorter distances, as the improved signal integrity and transmission speeds in newer serial technologies have begun to is outweigh the parallel bus's advantage of simplicity (no need for serializer and deserializer, power or SerDes) and to outstrip its disadvantages (clock skew, interconnect density). The migration is from PCI to PCI Express is an example.

Many serial communication systems were originally designed to transfer data over relatively is an large distances through some sort of data cable.

Practically all long-distance communication transmits data one bit at a time, rather than in and parallel, because it reduces the cost of the cable. The cables that carry this data (other than "the" serial cable) and the computer ports they plug into are usually referred to with a more specific name, to reduce confusion.

Keyboard and mouse cables and ports are almost invariably serial—such as PS/2 port and Apple Desktop Bus and USB.

The cables that carry digital video are almost invariably serial—such as coax cable plugged into a HD-SDI port, a webcam plugged into a USB port or Firewire port, Ethernet cableconnecting to an IP camera to a Power over Ethernet port, FPD-Link, etc.

Other such cables and ports, transmitting data one bit at a time, include Serial ATA, Serial SCSI, Ethernet cable plugged into Ethernet ports, the Display Data Channel using previously reserved is pins of the VGA connector or the DVI port or the HDMI port.

The RTL schematic of the UART is shown in figure 4

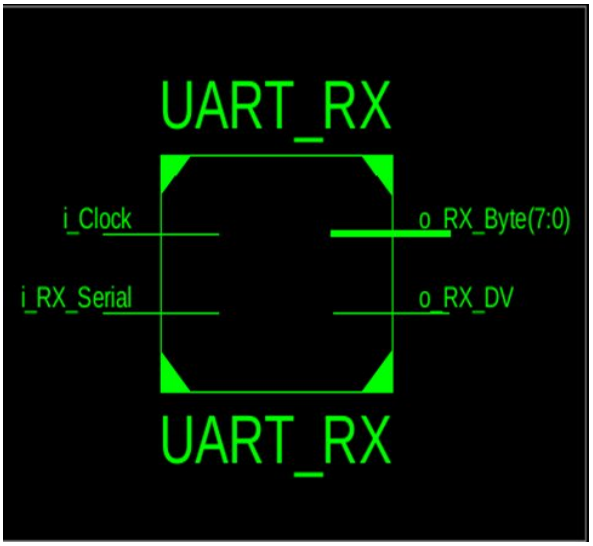

Figure 4: RTL schematic

The technology schematics are shown in Figure 5 and 6 respectively.

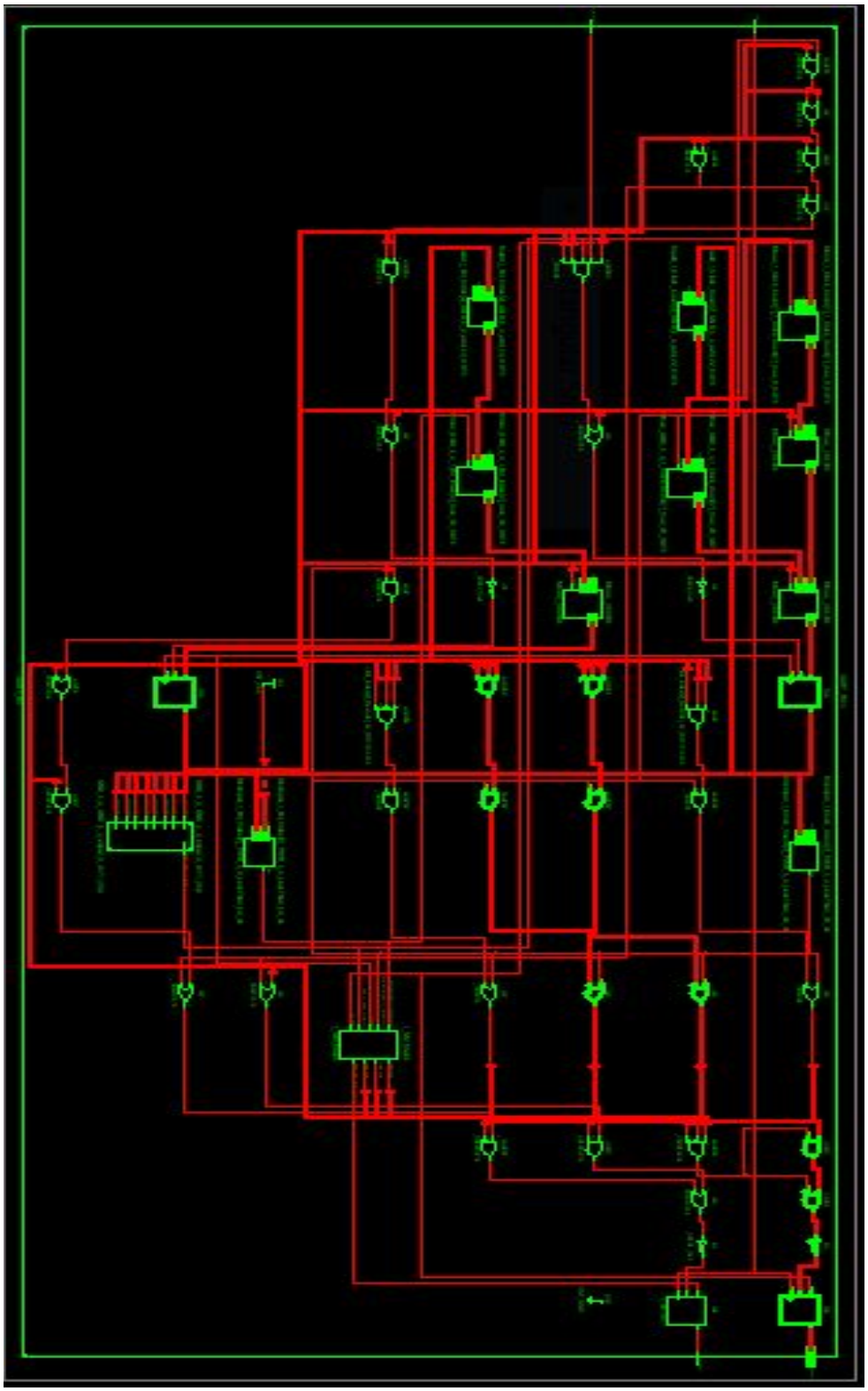

Figure 6: Technology schematic

4.3 SIMULATION :

Simulation Result of Baud Rate Generator B. Receiver Simulation During receivers in the of simulation, the receiving sampling clock frequency generated by the baud rate generator is set to 153600 Hz, UART receiving baud rate is set to 9600bps.

Logic signals as shown in the simulation results of the different modules of the UART prove its correct The operation of the proposed UART follows the standard protocols of the serial data is communication and it is compatible with any other standard UART.

Simulation Result of UART Working principle of this UART has been tested a ISE simulator, which can be implemented on FPGA.

The simulation results of the UART is shown in figure 7

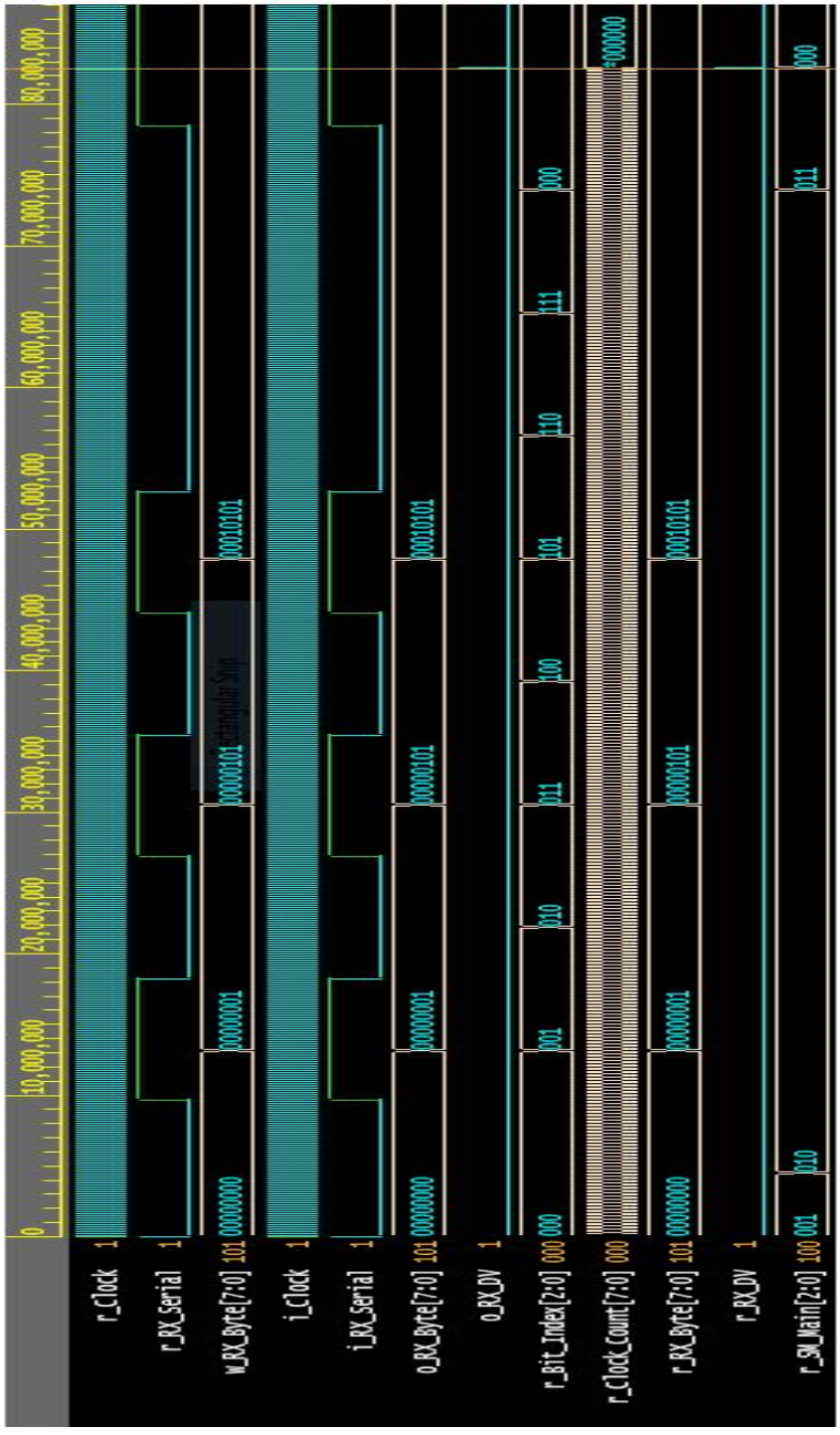

Figure 7: simulation result of UART

4.4 APPLICATIONS :

A universal asynchronous receiver transmitter (UART) emulation stage for modem communication uses a digital signal processor containing a software UART control program for sending UART control signals to hardware based UART emulation circuitry. The software UART control program communicates to a modem application interface program that is under control of a host processor. The UART emulation circuitry that is responsive to the control signals from the digital signal processor, includes dedicated transmit and receive FIFO buffer memory for storing modem data and also includes interrupt generation logic to generate an interrupt for the digital signal processor when the received FIFO buffer memory is at a predetermined threshold. The UART emulation circuitry also includes programmable control logic for facilitating host processor interrupt pacing to maintain high compatibility with legacy applications, namely DOS based applications.

4.5 ADVANTAGES :

A **UART** protocol which takes advantage of an line space state to optimize communication between computers. A delay is provided by sending a byte of actual data followed by a dummy byte. The dummy byte is produced by placing the SDO line in the mark state, thus freeing up CPU time.

Among other advantages, overall power reduction. In use, the **UART** is controlled to initially receive data as samples. The microcontroller examines the samples to determine the baud rate.

The peripherals of the UART receiver is shown in figure 8

Figure 8: peripherals of UART

The interface diagram of the UART is shown in figure 9

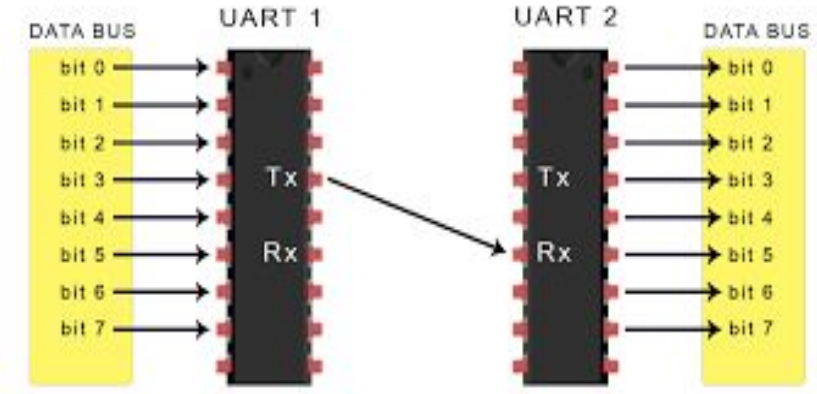

Figure 9: interface of UART

The interfaces of communication in drones is shown in figure 10

Figure 10: interfaces of communication in drones

The impulse defence are shown in figure 11

Figure 11: impulse defence

The electronic communication protocols are shown in figure 12

Figure 12: electronic communication protocols

The communication modules are shown in figure 13

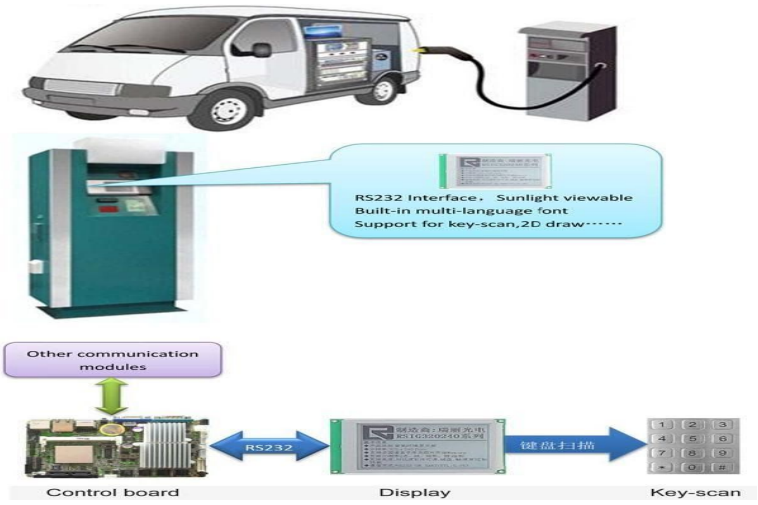

Figure 13: Communication modules

CHAPTER 6

5.1 Conclusion :

The design has been proposed using the resources available to meet the system requirements and has been tested to give satisfactory results. The project can be extended to include more is an of interlocks and to improve technology used. For example – The RS232 cable connected to a and to CC2500 module (for short ranges) and the PC connected to an AVR ISP would facilitate is a of wireless transfer of character commands from the PC. The instructions given to the hardware should additionally involve defining device addresses and path address for wireless transfer.

References

[1] Amanpreet Kaur, Amandeep Kaur, "An Approach for Designing a Universal synchronous is Receiver Transmitter (UART)", International Journal of Engineering Research and Applications. Vol. 2, Issue 3, May-Jun 2012.

[2] Sandeep Singh Rawat, &Prof.VishalRamola, " UART design using FIFO ram and LCR and circuit with BIST capability at different baud rate", International Journal of Engineering as a in Development and Research. 2015 IJEDR | Volume 3, Issue 3 | ISSN: 2321-9939.

[3] https://www.drdo.gov.in/drdo/English/Defence_Research_and_Development_Organisation. Pdf.