International Journal of Future Generation Communication and Networking
Vol. 12, No. 5, (2019), pp. 306- 318

FPGA BASED DIGITAL LOGIC ANALYZER FOR DIGITAL AUTOMATIC TEST EQUIPMENT

[1]Manjula C and [2]Jayadevappa D

[1]Oxford College of Engineering, Bengaluru, VTU, India

[2]JSS Academy of Technical Education, Bengaluru, VTU, India

Abstract

Simulation is necessary to identify the defects in most of the digital circuits before the final unit is formed. The process of simulation generally caters for displaying logic analysis. Usually, discrete logic circuits which are complex in nature are verified by input simulation and testing of outputs by boundary scan methods. Logic analysers used for such purposes may uncover hardware defects that are not found in simulation. This work covers design and simulation of Digital logic analyser and its sub modules Dual Port Random Access Memory (DPRAM), Parallel to serial converter (PTSC), Multi Standard Baud Rate Generator and Universal Asynchronous Receiver Transmitter (UART). Simulation is done using Xilinx ISE 14.5v EDA tool and FPGA implementation is carried out on Xilinx Spartan3 FPGA board.

Keywords: *Digital Pattern Generator, Digital Logic Analyser, Dual port RAM, Baud Rate Generator, UART.*

1. Introduction

Multiple analysis of signals from various digital circuits and systems can be analysed using an electronic instrument called logic analyser. These instruments converts the captured information to protocol decodes, timing diagrams machine traces and assembly language. When the users required to have relationships between many signals in digital systems [1], [2], there are advanced capabilities of triggering features in logic analysers. The overall ATE (Automatic Testing Equipment) system consists of Digital Pattern Generator (DPG), Digital Logic Analyser (DLA), Controller to configure DPG and DLA as per CUT (Circuit under Test).Whenever CUT is changed, controller will configure DPG and DLA accordingly to make ATE effective). Figure 1 shows the design of proposed Digital Test Pattern Generator (DTPG) implemented on Xilinx Spartan FPGA.

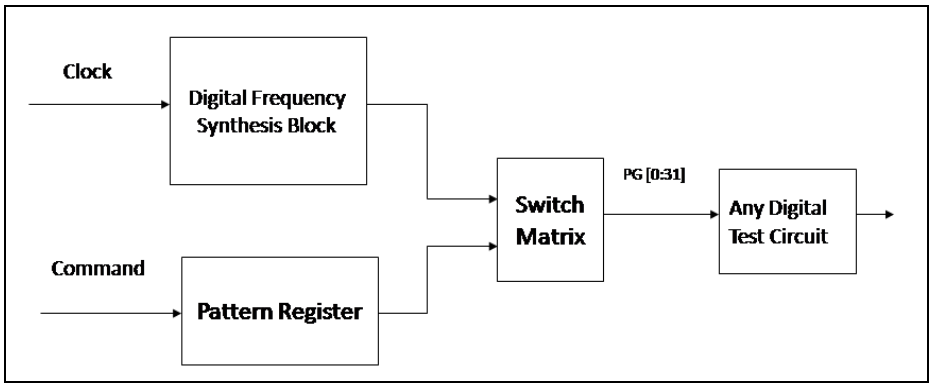


Figure 1. Digital Pattern Generator.

The design and simulation results of all individual sub modules; Digital Frequency Synthesizer, Pattern Generator, Switch Matrix and DUT of the FPGA based DTPG was published in IEEE Digital Library Journal.

When the system is under test, the logic analyser can be triggered to capture the data in a larger scale from the digital systems to analyse the complicated sequences of data. At the time when the logic analyser was put into use, it was a tradition to attach several hundred clips to a digital system. Later, due to the applications of specialized connectors, the development of probes of logic analysers have resulted to a common footprint that support multiple vendors, which enhanced added freedom to consumers. The connector less technology was demonstrated in 2002 and which was recognized by various vendors specifically with trade names such as Soft Touch, D-Max and Compression Probing [3]. Later they became very become popular also. These provided a strong, durable and reliable electrical and mechanical connection between the circuit board and probes with less than 0.5 to 0.7 pF loading per signal. Once the information is captured, logic analysers can present it in so many ways starting from simple like displaying waveforms or listing of states etc., to complex way i.e. displaying traffic of decoded Ethernet protocol. Similarly, some of the logic analysers can also work in a compare mode. In this case, captured data will be compared with previously loaded data set. This is helpful for the testing of empirical data for long time. Recent developed logic analysers can even adjusted to set to email a copy of the test information to the in charge on a successful trigger [4], [5].

2. Related Work

In 1960's integrated circuits and digital computing process started and at the same time, a new and complex issues have begun to file up because of inability of oscilloscopes for handling troubles. Early solutions made an attempt to combine hardware from multiple oscilloscopes into single unit, but a lack of definite data interpretation, screen clutter and the constraints of probes made this solution applicable for only marginal usage.

In 1973, the HP 5000A logic analyser was introduced, this was published in Hewlett Packard newsletter. Probably this was the first instrument commercially available namely Logic Analyser. However, the HP 5000A was limited to only two channels and information was usually presented by means of 32 LEDs with two rows. The HP 1601L was the first truly parallel instrument which has 12 channel. This was a plug-in for the HP 180 series oscilloscope mainframes and used the oscilloscope screen to present 16 rows of 12 bit words as 1s and 0s.

At present, there are 3 varieties of logic analysers are available in the market. The first one is the modular logic analyser which consist of either a chassis or mainframe and logic analyser modules [6]. In this case, the mainframe or chassis has display, controls, computer control and also multiple slots where the actual data capturing hardware is installed. In the modules, each have a specific number of channels and multiple modules may be combined to get a very high channel count. This type of logic analysers are generally very expensive. The cost justification for higher performance logic analysers depends on the ability to combine multiple modules to yield for the higher channel count. The user often must provide their own host PC or purchase an embedded controller compatible with the system for the very high end modular logic analysers [7], [8].

Portable LAs, are also called as standalone logic analysers [9], [10]. These analysers combines everything into a single unit, with options installed at the factory. But, portable logic analysers have lower performance as compared to their modular counterparts, they are often applied for general purpose debugging by cost conscious users.

In case of PC based logic analysers, the hardware connects to a computer through a USB or Ethernet connection and transmits the signals which are captured to the software on the computer. These devices are generally much smaller and cost effective because they make use of existing keyboard, display and CPU of personal computers.

3. Proposed Approach

The figure 2 shows the design of proposed Digital Logic Analyser (DLA) to be implemented on FPGA. The design consists of following blocks; Dual port RAM with simultaneous READ and

WRITE, Parallel to serial converter, UART, MAX232 level converter and RS232 cable. The design consideration of DLAL is as follows.

- The width of the FIFO DPRAM is fully programmable and is dependent on processor data bus.
- If there is a difference between the width of FIFO and processor data memory latency will be more
- In DPRAM, there is a separate port for data read & write with the condition with the condition first read and then write
- RS232 has a fixed frequency i.e., around 5KHz
- If processor speed is in KHz then DPRAM can be replaced by a single row shift register of data bus width.
- If processor is 10MHZ (data writing speed), data reading speed is 10KHZ, then data reading is slower by 10^3 compared to writing speed. Hence 10^3 rows of DPRAM is required for data missing.

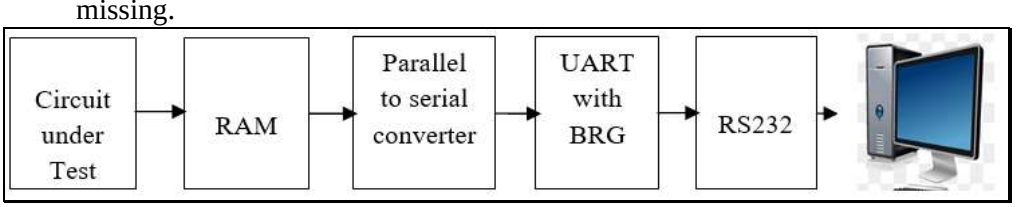


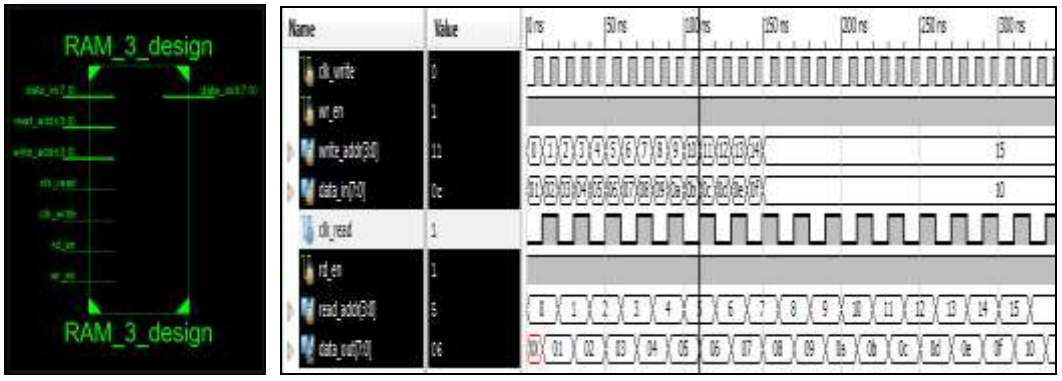
Figure 2. Proposed block diagram of Digital Logic Analyser.

4. Design and Simulation Results of DLA

4.1 DPRAM

Figure 4 (a) shows the RTL schematic of DPRAM (16x8) with 16 locations of each 8 bits, 2 separate ports, 2 address buses and 2 clocks for reading and writing data into the memory. This will perform read and write memory operations from the single RAM, with a condition that reading and writing operations will happen from different memory locations. There is a minimum prescribed data settling time (latency) of 3 write clocks time, before reading data from various locations of the RAM. The input and output pins/buses and their respective functionalities are as follows.

- data_in[7:0] – 8 bit input data bus or input data port to write data into the memory location as per corresponding write address.
- data_out[7:0] - 8 bit input data bus or input data port to read data into the memory location as per corresponding read address.
- read_addr[3:0] - 4 bit read address bus that specifies the memory location, from where the data is read
- write_addr[3:0] - 4 bit write address bus that specifies the memory location, from where the data is written
- clk_read - clock frequency at which data will be read from the memory
- clk_write - clock frequency at which data will be written into the memory
- rd_en- read enable signal to activate memory read operation
- wr_en–write enable signal to activate memory write operation



(a) **(b)**

Figure 4. (a) RTL schematic of DPRAM and (b) Simulation waveform of DPRAM.

This latency time between successive write and read operation can be noted only in post layout timing simulation and not indicated in behavioural simulation. The simulation waveform of DPRAM is illustrated in figure 4(b). As can be seen from the simulation waveforms,

- Write clock is running at 200MHz frequency or t=10ns.
- Read clock is running at 100MHz frequency or t=20ns.
- Write enable and read enable signal are held high from 0 to 400ns, all through the simulation snapshot window.
- Write clock frequency is more than read clock frequency.
- Between 0 to 150ns, 4 bit Write address is incremented in DPRAM and the write address is varied from 0 to 14, for each of the 14 clock pulses.
- During the same duration of 0 to 150ns, 8 bit-14 different data are written into respective data locations of DPRAM as per the addresses.
- Between 0 to 300ns, 4 bit read address is incremented in DPRAM and read address varied from 0 to 14, for each of the 14 clock pulses.
- During the same duration of 0 to 300ns, 8 bit-14 different data are read from respective data locations of DPRAM as per the addresses.
- DPRAM is having Dual ports and facilitate simultaneous read and write from different data locations and different read and write clock rates.
- Hence, the DPRAM successively designed, coded in Verilog and simulated in Xilinx ISE simulator is working and the functionality is fully verified.

4.2 Parallel to Serial Converter

In figure 5, the digital design of parallel to serial converter is described. This design contains 4 D-type flip-flops connected in cascade (output of each flip-flop is connected to input of next flip-flop via mux). Mux provides for serial or parallel loading of input data to each of the flip-flops. For serial input data loading, the select line of the mux [cntl =0], then, at the next active clock edge d[2]=q3=d3 i.e., d3 data is loaded to d2 and with next successive clock pulses, d3 will move from q3 to q2 to q1 and finally to q0, the last flip-flop in the 4 flip-flop cascade.

- If cntl=1, mux data [md] inputs - md2, md1 and md0 will be applied directly to flip-flop inputs d2, d1and d0 respectively and hence this method of data loading to a converter is called Parallel data loading. During the next immediate active clock edge, q3=d3, q2=d2=md2, q1=d1=md1 and q0=d0=md0.
- If the data output is read out from the 4 flip-flop outputs [q3, q2, q1 and q0] during this immediate active clock edge, we get parallel data output.
- If the data is parallel loaded and parallely read out, then such a system is called parallel in and parallel out converter (PIPO).
- If the data is loaded parallely with cntl=1 and data is read out from q0, after every active edge of 4 clock cycles, such a system is called Parallel to Serial converter.

Hence using the above circuit, we can realize SISO, PISO, SIPO and PIPO. But in the proposed research experiment requirement will be to implement parallel to serial converter only.

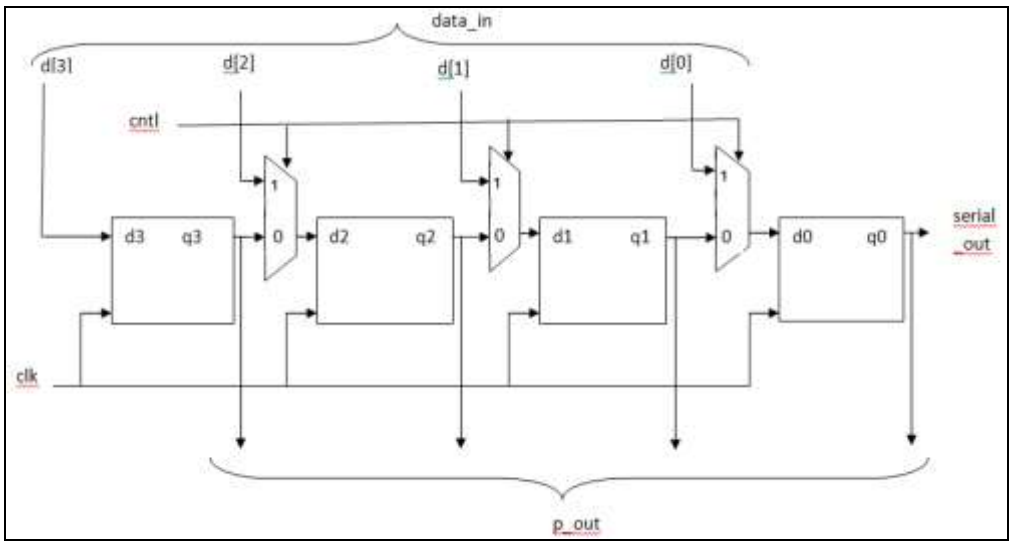
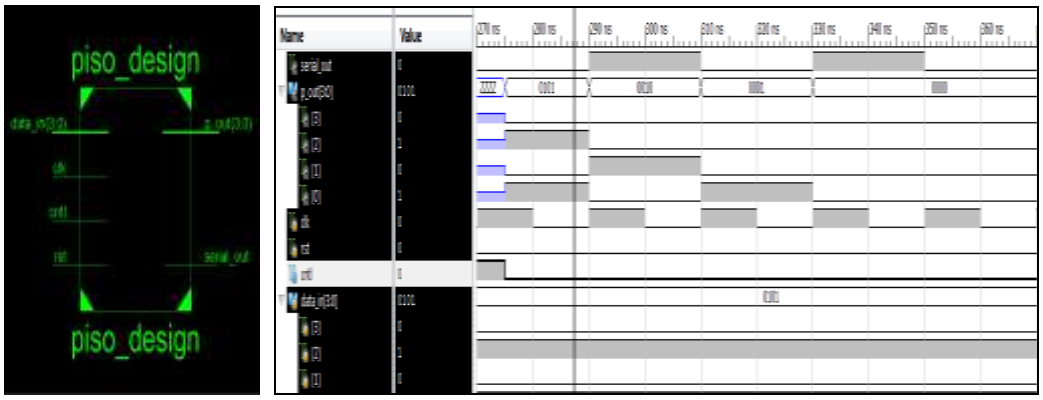


Figure 5. Digital circuit diagram of Parallel to Serial converter.

If the data is loaded parallelly with cntl=1 and data is read out from q0, after every active edge of 4 clock cycles, such a system is called Parallel to Serial converter. Hence using the above circuit, we can realize SISO, PISO, SIPO and PIPO.



(a) **(b)**

Figure 6. (a) RTL schematic of Parallel to serial converter and (b) Simulation result of Parallel to Serial Converter.

But our research experiment requirement will be to implement Parallel to Serial converter only. The figure 6(a) shows the RTL schematic of PISO after synthesis using Xilinx ISE tool. As can be seen,

- Data_in[3:0] is 4 bit data input to the PISO
- P_out[3:0] is 4 bit parallel output
- Serial_out is 1 bit serial output of PISO
- Cntl is 1 bit input signal to control the mux. [If cntl =0, serial in & if cntl=1, parallel input]
- Reset signal=1 will clear all 4 flip flops & Reset signal=0 will allow PISO to function normally.

The table1 shows data transition happening through all 4 flip-flops with respective parallel inputs is applied at various clock edges and the corresponding serial outputs, at various values of cntl signals and corresponding clock transitions.

Table 1. Truth table for all 4 flip-flops.

Clk	Cntl	Data in	D_3	D_2	D_1	D_0	Serial Out
1	1	0101	0	0	0	1	-
2	0		0	0	1	0	1
3	0		0	0	0	1	0
4	0		0	0	0	0	1
5	0		0	0	0	0	0
6	1	1111	1	1	1	1	-
7	0		0	1	1	1	1
8	0		0	0	1	1	1
9	0		0	0	0	1	1
10	0		0	0	0	0	1

The following aspects can be observed from the simulation waveform.

- 500MHZ frequency Clock is applied to the PISO converter
- Reset is held low all through the simulation.
- From 270ns to 400ns, parallel input data_in[3:0] is held at 0101.
- Cntl is held high initially between 270 to 280ns and parallel data output pout [3:0] is tristated.
- From 280ns to 400ns, Cntl =0, to facilitate serial loading of data to the cascade of 4 flip-flops in the PISO converter
- At 280ns, parallel data input data_in[3:0]=0101, pout[3:0]= 0101 and serial_out=0.
- At the clock edge of 290ns, parallel data input applied at 270ns will get shifted right once or by 1 bit and hence pout= 0010 and serial_out=1.
- At the clock edge of 310ns, parallel data input applied at 290ns will get shifted right once or by 1 bit and hence pout= 0010 and serial_out=0.
- At the clock edge of 330ns, parallel data input applied at 310ns will get shifted right once or by 1 bit and hence pout= 0010 and serial_out=1.
- And the similar chain of events repeats. Thereby converting 4 bit parallel data input applied into 4 bit serial data output.
- The total number of clock pulses required to convert parallel input data into serial output data= the number of bits of parallel data.
- Hence PISO functionality is verified.

4.3 Standard Baud Rate Generator for FPGA

The multiple standard baud rate generator (BRG) for the FPGA board having 10 MHz as master clock is used in the design of DLA. The table 2 indicates multiple standard baud Rates and strategy to derive the multiple specific baud clocks from master clock on the FPGA board of 10MHz.

For instance, to transmit data from UART at 1200bps, a baud rate clock of 600Hz has to be derived from the master clock of 10MHz. This is achieved by dividing master clock of 10MHz by a factor of 16666 and thereby the duration between pulses of 1.66ms is achieved. Likewise, all other standard baud rates can be achieved by using suitable dividing factor, to thereby achieve respective baud rates and pulse duration.

Table 2. Multiple Standard Baud Rates and Clock for FPGA.

Standard Baud Rate in bps	Master clock to be divided a factor	Specific Baud clock in HZ	Duration b/w pulses
1200	16666	600	1.66 ms
2400	8333	1200	0.833ms
4800	4167	2400	0.416 ms

9600	2083	4800	208.35 µs
19200	1041	9600	104.14 µs
38400	520	19200	52.0 µs
57600	260	38400	26.05 µs
115200	174	57600	17.45 ms

The figure 7 shows multiple BRG clock design wherein master clock is divided by suitable division factor to achieve required baud clock.

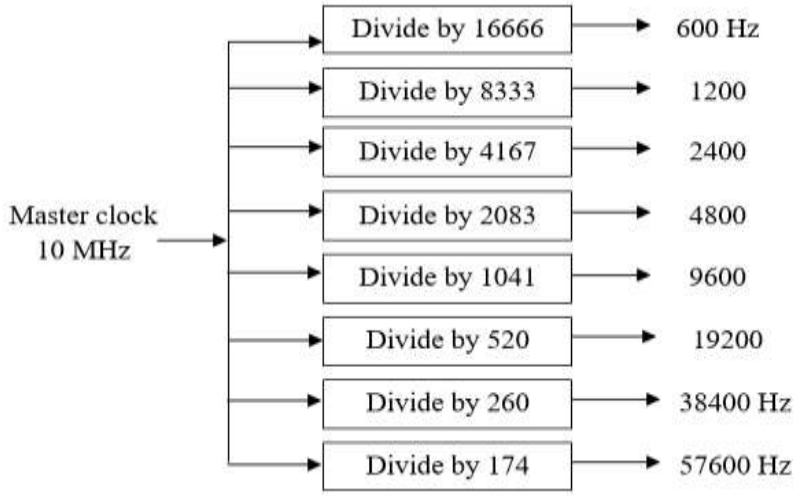


Figure 7. Baud Rate Generator (BRG) clock Design.

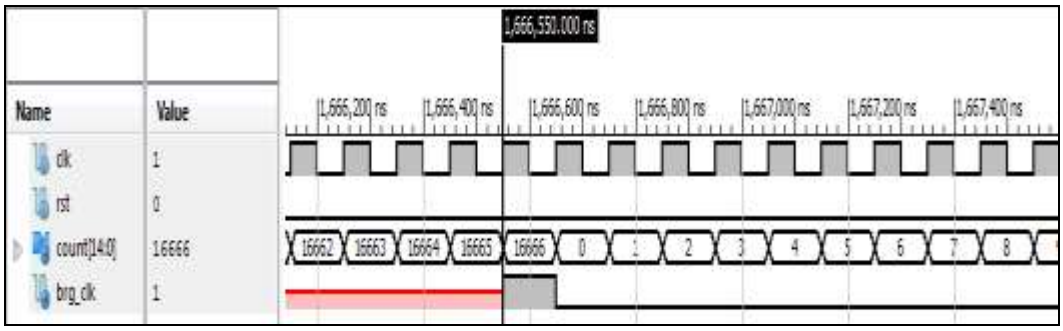
4.3.1 Simulations results for different Baud rate and Baud rate clock

A. Baud rate 1200 bps and baud rate clock 600 Hz

The simulation of Baud rate generator (1200 bps) of baud clock 600 is shown in figure 8. The following are the simulation results implications.

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 1200bps [frequency=600Hz], hence the clock has to be divided by a factor of 16666.
- Using multiple clock dividers, the master clock is divided by 16, again divided by 10 three times.
- Alternative method a 16666 counter's terminal counter signal will provide the required 9600bps or 4800Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 16666 clock pulses and are indicated by B_clk dotted signals.



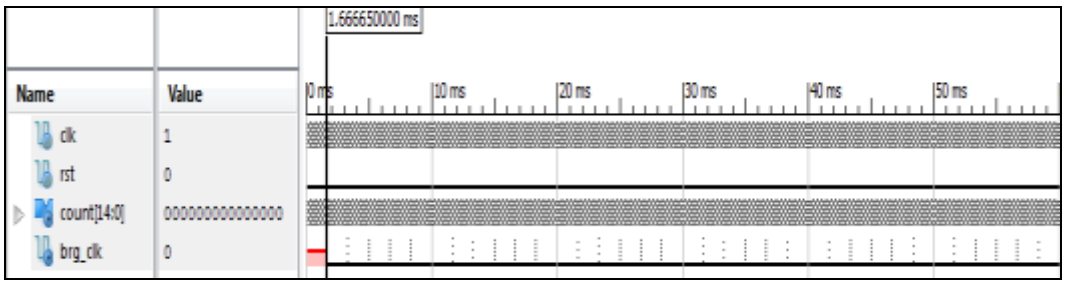


Figure 8. Simulation of Baud rate generator of baud clock 600.

B. Baud rate 2400 bps and baud rate clock 1200 Hz

The simulation of Baud rate generator (2400 bps) of baud clock 1200 is shown in figure 9.

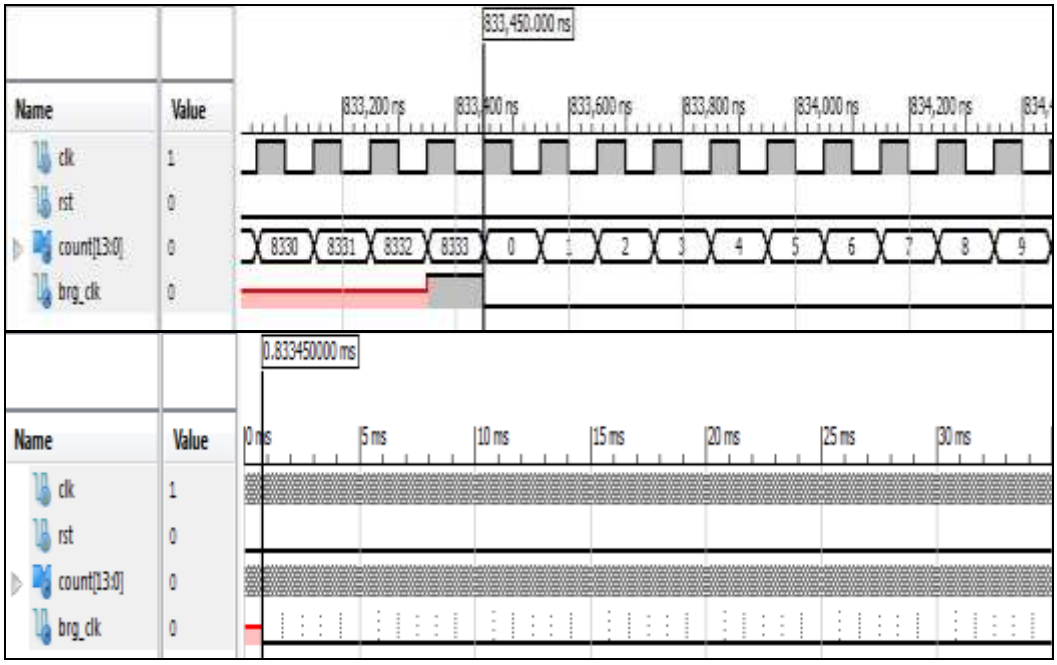


Figure 9. Simulation of Baud rate generator of Baud clock 1200.

The inferenceof the simulation results are as follows

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 2400bps [frequency=1200Hz], hence the clock has to be divided by a factor of 8333.
- Using multiple clock dividers, the master clock is divided by 8, again divided by 10 three times.
- Alternative method 8333 counter's terminal counter signal will provide the required 2400bps or 1200Hz more accurately.
- As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 8333 clock pulses and are indicated by B_clk dotted signals.

C. Baud rate 4800 bps and baud rate clock 2400 Hz

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 4800bps [frequency=2400Hz], hence the clock has to be divided by a factor of 4167.

- Using multiple clock dividers, the master clock is divided by 4, again divided by 10 three times.
- Alternative method a 4167 counter's terminal counter signal will provide the required 4800bps or 2400Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 4167 clock pulses and are indicated by B_clk dotted signals.

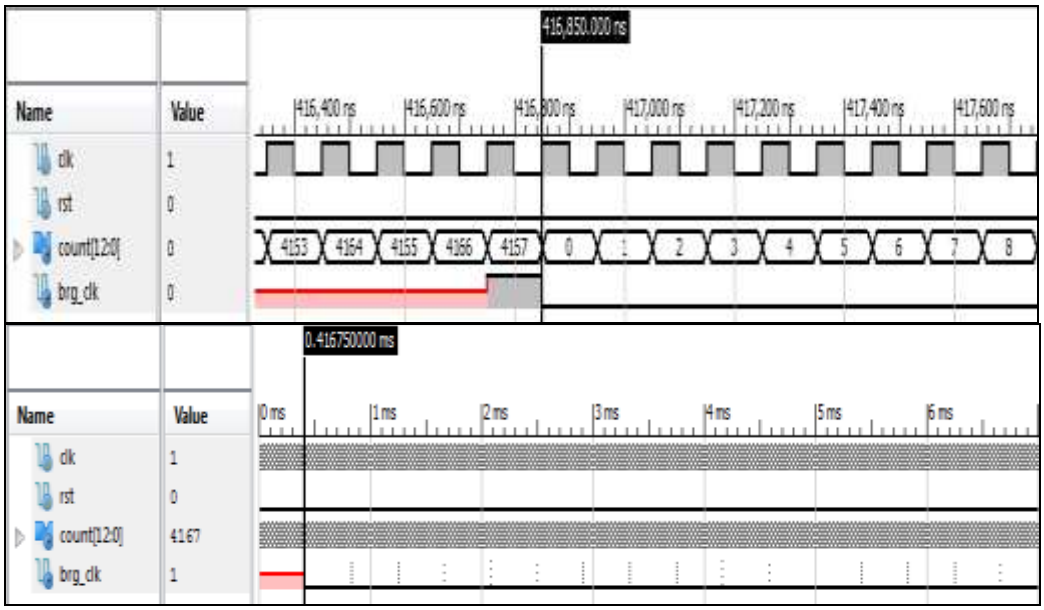
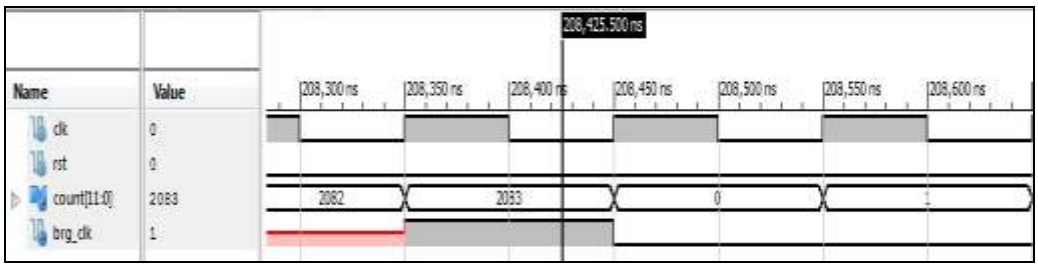


Figure 10. Simulation of Baud rate generator of Baud clock 4800.

D. Baud rate 9600 bps and baud rate clock 4800 Hz

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 9600bps [frequency=4800Hz], hence the clock has to be divided by a factor of 2083 [approximately 2000].
- Using multiple clock dividers, the master clock is divided by 2, again divided by 10 three times.
- Alternative method a 2083 counter's terminal counter signal will provide the required 9600bps or 4800Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 2083 clock pulses and are indicated by B_clk dotted signals.



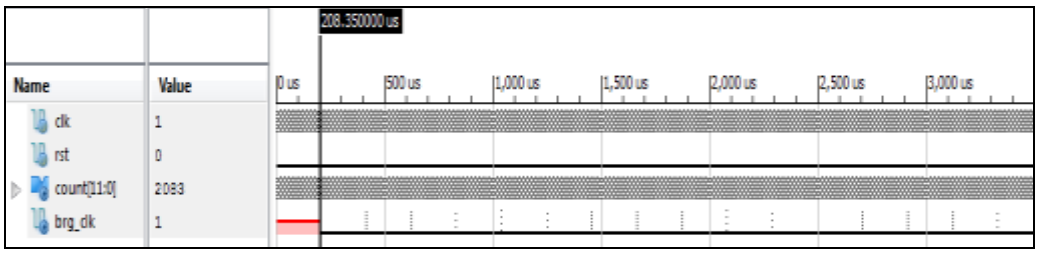


Figure 11. Simulation of Baud rate generator of Baud clock 4800.

E. Baud rate 19200 bps and baud rate clock 9600 Hz

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 19200bps [frequency=9600Hz], hence the clock has to be divided by a factor of 1041.
- Using multiple clock dividers, the master clock is divided by 10 three times.
- Alternative method a 1041 counter's terminal counter signal will provide the required 19200bps or 9600Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 1041 clock pulses and are indicated by B_clk dotted signals.

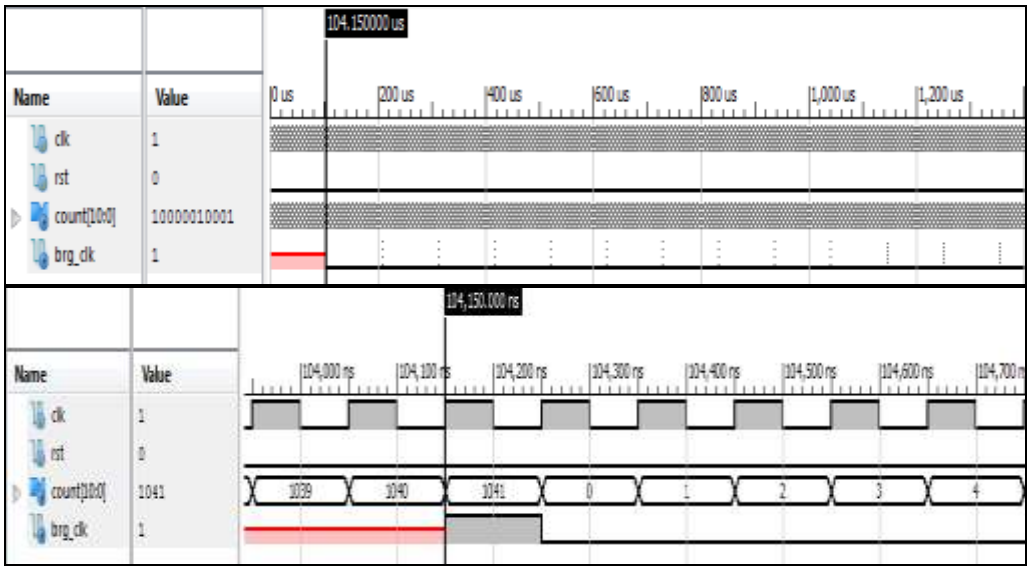


Figure 12. Simulation of Baud rate generator of Baud clock 9600.

F. Baud rate 38400 bps and baud rate clock 19200 Hz

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 38400bps [frequency=19200Hz], hence the clock has to be divided by a factor of 520.
- Using multiple clock dividers, the master clock is divided by 5, again divided by 10 two times.
- Alternative method a 520 counter's terminal counter signal will provide the required 38400bps or 19200Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 520 clock pulses and are indicated by B_clk dotted signals.

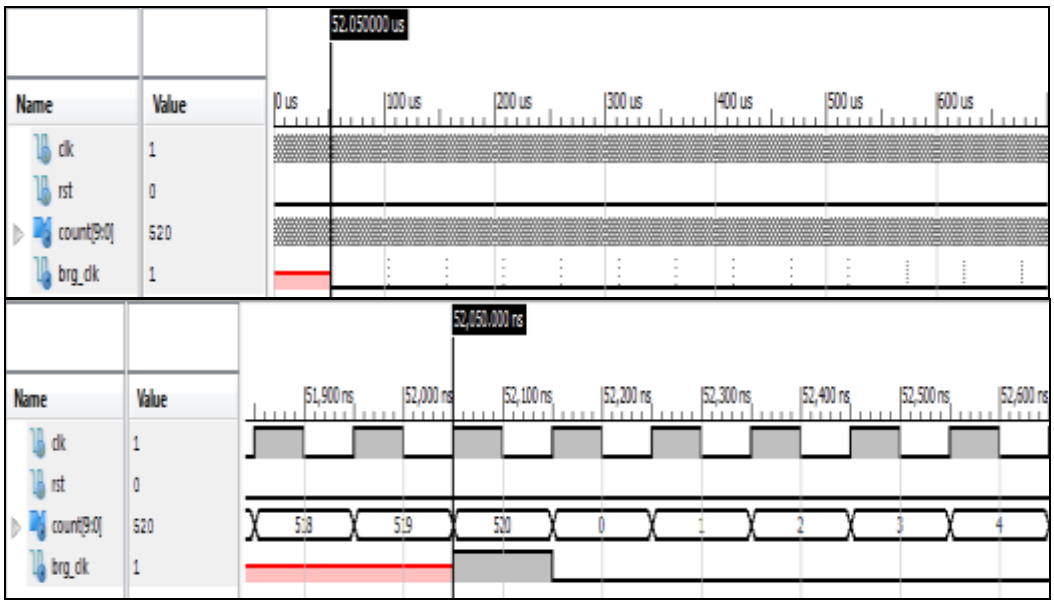


Figure 13. Simulation of Baud rate generator of Baud clock 19200.

G. Baud rate 57600 bps and baud rate clock 38400 Hz

- Master Clock frequency is 10 MHz=20Mbps.
- BRG to be generated at 57600bps [frequency=38400Hz], hence the clock has to be divided by a factor of 260.
- Using multiple clock dividers, the master clock is divided by 2, divided by 13 and again divide by 10.
- Alternative method a 260 counter's terminal counter signal will provide the required 57600bps or 38400Hz more accurately.

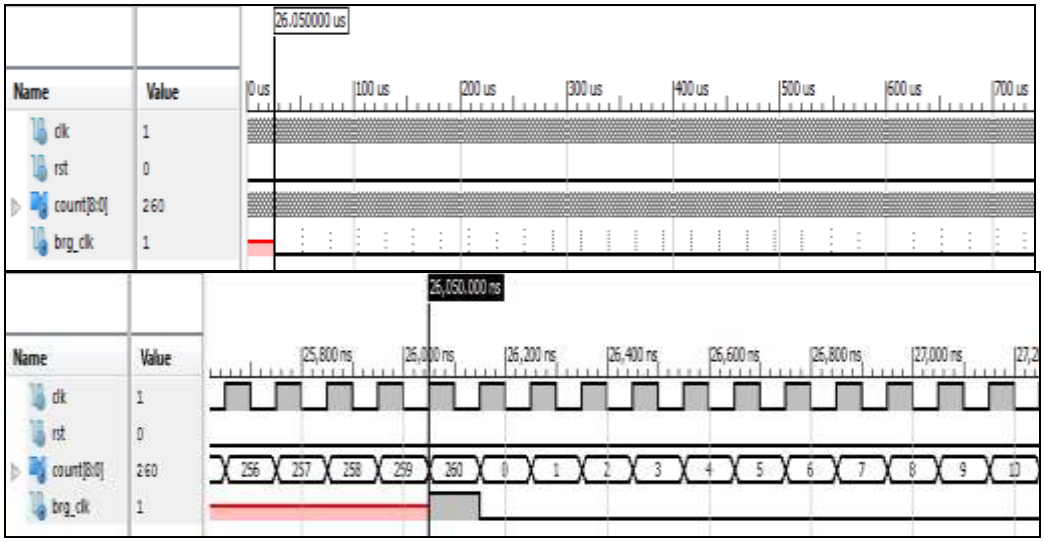


Figure 14. Simulation of Baud rate generator of frequency 38400.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 260 clock pulses and are indicated by B_clk dotted signals.

H. Baud rate 115200 bps and baud rate clock 57600 Hz

- Master Clock frequency is 10 MHz=20Mbps.

- BRG to be generated at 115200bps [frequency=4800Hz], hence the clock has to be divided by a factor of 174.
- Using multiple clock dividers, the master clock is divided by 13, again divided by 13.
- Alternative method a 174 counter's terminal counter signal will provide the required 115200bps or 57600Hz more accurately.

As can be seen from the simulation waveforms, Baud rate pulses or frequency are generated after every 174 clock pulses and are indicated by B_clk dotted signals.

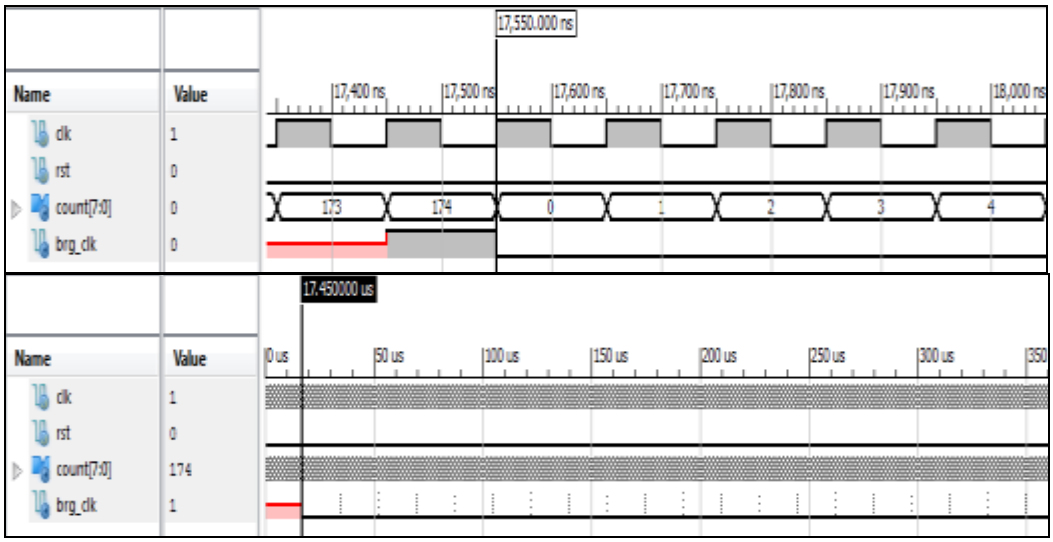


Figure 15. Simulation of Baud rate generator of Baud clock 57600.

4.4 UART Design

The top level block diagram of UART is depicted in figure 16. It consists of digital modules, Parallel to serial converter, Multiple Baud Rate Generator which provides Universal data rates [1200, 2400, 4800, 9600, 19200, 38400, 57600 and 115200]. In UART, the transmission Buffer Register [TBR] sends transmission data serially at desired baud rates and the Baud Rate select Mux is used to select required Baud clock for data transmission based on specified Baud rate.

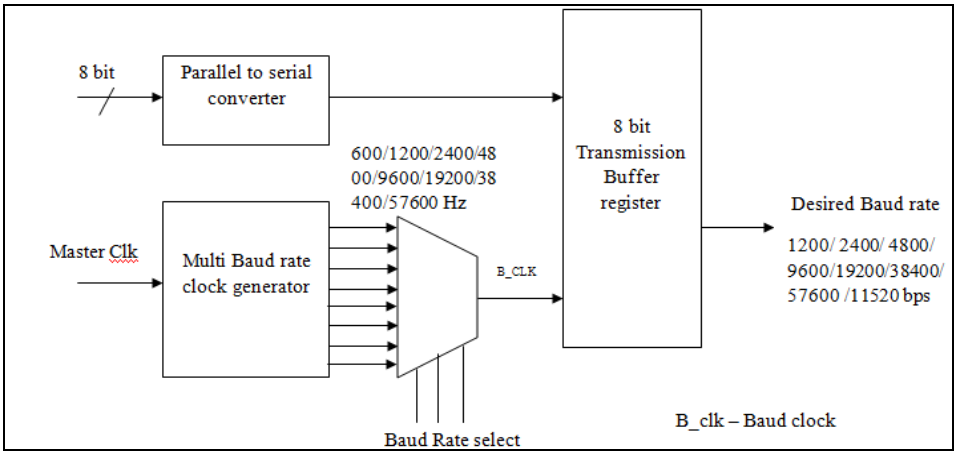


Figure 16. Block diagram of UART.

5. Conclusion

In this paper, novel and innovative design was undertaken to implement all the sub modules of a Digital Logic Analyzer. Multi Baud rate generator is a specialized module that would make this DLA acquire data from low or high frequency DUT channels and enable high speed data acquisition and display. Till date, several products are available in the market with varied specifications and functionalities with respect to Digital Logic Analyzers. But no research has been undertaken to design and develop a Digital ATE on FPGA. Hence this research experiment is very unique and important as compared to the existing systems.

References

[1] Amit Dhir, "Programmable logic solutions enabling digital consumer technology", the digital consumer technology handbook, chapter 19, (2004), pp. 516-537.
[2] S.Adilakshmi, K.Rajasekhar and T.B.K.Manoj kumar, "Integrating Logic Analyzer Functionality into VHDL designs", International Journal of Computer Science and Information Technologies, vol. 3, no. 1, (2012), pp. 3107-3111.
[3] D Nanda Kishore and S K Ghoshal, "Design and implementation of a PC-based FPGA Tester", IETE Technical Review, vol.12, (2015), pp. 119-132.
[4] Sonam M. Tiple, Swetansha C, Aishwarya S, V. P. Mulik, A. P. Yadav , "Arm Based Logic Analyzer", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 5, Issue 3, (2015).
[5] Chaitanya Kilaru, J K R Sastry and K RajaSekhara Rao, "Testing distributed embedded systems through logic analyzer", International Journal of Engineering and Technology, 7 (2.7), (2018), pp. 297-302.
[6] Ranjan Kumar Sen, Ajit Pal and A. K. Choudhury, "A Programmable Logic State Analyser", IETE Journal of Research, vol. issue 7, (2015), pp. 434-439.
[7] Sheng-Luen Chung , Cheng-Wei Chu , Yu Fu and , Embedded control system design and practice", Journal of the Chinese Institute of Engineers, vol. 30, issue 6, (2011), pp. 1103-1107.
[8] W F Clocksin, "Logic Programming and Digital Circuit Analysis", Journal of Logic Programming, (1987), pp. 59-82.
[9] Fei Tao, Y. Tang, X. Zou, Q. Qi, "A field programmable gate array implemented fibre channel switch for big data communication towards smart manufacturing", Robot. Comput.-Integr. Manuf. vol.57, (2019), pp. 166–181.
[10] J. Li, X. Dai, Z. Meng, "Automatic reconfiguration of petri net controllers for reconfigurable manufacturing systems with an improved net rewriting system-based approach", IEEE Trans. Autom. Sci. Eng. vol. 6, (2009), pp. 156–167.