Design and Simulation of UART IP Core for FPGA Implementation

Abstract—Universal Asynchronous Receiver Transmitter (UART) is a popular two wire serial communication interface between two microcomputer based systems. The programmable logic devices can be used for such application by developing core for UART. This design included transmitter, receiver and baud rate generator. By using hardware descriptive language UART simulation can be tested before it can be loaded on programmable device. In simulation and results part, this paper will describe the functionality of UART. From the designing point of view, we can say that this design is most efficient and cost effective.

Index Terms: UART, FIFO, FSM, FPGA

I. INTRODUCTION

Universal Asynchronous Receiver Transmitter (UART) is the heart of serial communication. Synchronous communication requires a common clock which should have same phase and frequency for the entire duration of communication [1]. But it is very difficult to maintain same clock frequency and phase for a longer time to all devices which are involved in serial communication. So to avoid this problem, asynchronous communication is used, which operates on different clocks where to check the order of communication extra synchronization bits are added [2].

UART resides in computer and this computer can handle parallel data only [3]. So in transmitter side, UART takes parallel data from computer and converts it in serial form and in receiver side, another UART takes this serial data from transmitter and converts back in parallel form so that computer can handle it.

In this paper we describe the simulation of UART core for implementation of such application on Field Programmable Gate Arrays (FPGA). The FPGA can be a good solution for reconfiguration of the system hardware modification.

II. DESCRIPTION OF EFFICIENT UART DESIGN

The internal architecture of UART mainly consists of two modules, transmitter module and receiver module as shown in figure 1. FIFO, parallel to serial shift register, parity generator and multiplexer are the sub modules of UART transmitter. Same way UART receiver consists of serial to parallel shift register, parity detector and FIFO. FIFO is used for temporary data storage purpose [4].

![Figure 1 UART model](image)

UART 16550 has programmable options [6]. It consists of 1 low start bit, 6/7/8 data bits, 1 parity bit and 1 high stop bit [5]. Odd parity or even parity can be used to detect any error occurred during transmission of data bits. In a given design I have used 7 data bits and even parity bit. Low start bit indicates to the receiver that new block of data is received and a high stop bit is used to indicate the end of a block. Between these start and stop bit data bits are transmitted followed by a parity bit. The data is transmitted LSB first.

III. RESULTS AND SIMULATION
A. BAUD RATE GENERATOR

The rate at which the data is transmitted is known as Baud Rate [1]. UART receiver operates on the frequency which is 8 or 16 times higher than transmitter. Baud rate generator provides two clocks one for transmitter and another for receiver to maintain data integrity between transmitter and receiver. Baud rate generator is shared between UART transmitter and receiver.

Different values of baud rates are available: ex 300, 1200, 4800, 9600, 19200, 38400, 57600 etc [3]. The value of divisor is equal to 13 (divisor = 8 MHz / (38400 X 16) for 8 MHz system frequency and 38400 baud rate value [1], [6]. Reset and sysclk are inputs of baud rate generator. It generates two clocks: bclk is used as transmitter clock and bclk16 is used as receiver clock. For receiver clock it will go high after 13 X sysclk and for transmitter it will go high after 16 X bclk16 as shown in figure 10.

B. UART Transmitter

The UART transmitter consists of FIFO, shift register, parity generator, multiplexer and FSM.

1) FIFO (First in first out memory)

From figure 3 the FIFO (first in first out) buffer provides more buffering space and further reduces the chance of data overrun. This FIFO is designed using UART 16650 specification. So FIFO is 7 bits wide and 16 in depth. When write enable signal is high the corresponding data is written to the FIFO at the rising edge of sysclk and when read enable is high, data word is removed at the rising edge of BCLK.

2) Shift register

As shown in figure 4 the output of FIFO data_out is given to shift register. In this design signal TSR is used to load and shift data in shift register. In shift register, when load signal is high, TSR is loaded with 7 bit parallel data which is the output of FIFO. When shift enable signal indicated by shift_en is high, loaded value of TSR is moved one position right and MSB is filled with ‘0’ bit and the output of shift register gets the LSB of TSR. This process is continue till all bits of TSR are shifted right and recovered by shift register output. At this time TSR fill with all zeros and new value of parallel data from FIFO output is loaded into TSR. Shift_en signal is used for the controlling purpose.

3) Multiplexer

Multiplexer is used to select one out of many data inputs. From figure 5, simulated wave form of 4:1 line multiplexer is shown. Relationship between select bits and multiplexer output is as shown in table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value at 11.7 ns</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESET</td>
<td>0 1</td>
</tr>
<tr>
<td>LOAD</td>
<td>0 0</td>
</tr>
<tr>
<td>BCLK</td>
<td>0 0</td>
</tr>
<tr>
<td>SHF_T,EN</td>
<td>0 1</td>
</tr>
<tr>
<td>DATA,0</td>
<td>0 0</td>
</tr>
</tbody>
</table>

Fig. 3. FIFO for UART transmitter

Fig. 4. Parallel to serial shift register for transmitter

Fig. 5. Simulated waveform of baud rate generator

Table 1. Relation between select bits and multiplexer output
Select bits | Output of multiplexer
--- | ---
S2 | S1
0 | 0 | Start bit '0'
0 | 1 | Serial data bit
1 | 0 | Parity bit
1 | 1 | Stop bit '1'

4) **Parity Generator**

The parity bit is used for error detection. In this design I have used even parity. Parity bit is generated after xoring of 7 data bits with parity itself as shown in figure 6. When parity_en is high and after 7 clock cycles of bclk parity bit is generated.

5) **FSM (Finite State Machine)**

Finite state machine is used to generate control and status signals for all blocks of transmitter. Reset, BCLK, Parity bit enable and FIFO empty are its input signals and at provides Load, shift enable, parity enable and selection bits to control whole transmitter section.

6) **Top transmitter module**

After interfacing all blocks of transmitter we get simulated waveform of top UART transmitter as shown in figure 8.

C. **UART RECEIVER**

The UART receiver consists of FIFO, shift register, parity detector and FSM.

1) **Shift Register**

As shown in figure 9, serial data RXD is given to shift register. It is used signal TSR of 7 bit. When rising edge of BCLK and shift enable is high, MSB of TSR gets RXD and shift one position right. This process continues till all bits of TSR gets the values of RXD. Whenever load_fifo signal goes high, output of shift register gets the value of TSR parallely.
2) Parity detector

The parity detector detects the parity bit which is transmitted with data bits. When parity enable is high parity is generated by xoring of parity and RXD (serial input of parity detector). When the value of count becomes “0111”, output of parity detector which is specified by parity_bit gets the value of parity of previous clock.

3) FSM (Finite State Machine)

Finite state machine is used to generate control and status signals for all block of receiver. Reset, TXD, Parity bit enable and BCLK are its input signals and at output it generates Load, shift enable, parity enable, RXD and clear old parity signals to control all blocks of receiver. Simulated waveform of FSM are shown in figure 11.

4) FIFO (First in first out memory)

FIFO size same as that of transmitter. When write enable signal is high the corresponding data is written to the FIFO at rising edge of BCLK and when read enable signal is high written values are read out at rising edge of sysclk.

5) Top receiver module

After port mapping (interfacing) of all blocks of receiver we get simulated waveform of top UART receiver as shown in figure 13.

C. Top UART model

After interconnecting top transmitter, top receiver, and simulated waveform of full UART model are generated as shown in figure 14. After 7 clock pulses of bclk, first data bits corresponding to input are generated.
VII. CONCLUSION

UART is efficiently designed using VHDL. Functionality of UART is also verified. Baud rate generator is also designed and simulated to provide data integrity between transmitter and receiver. In the future one can also implement on FPGA or CPLD for hardware implementation of UART functionality.

REFERENCES


