© June 2019 | IJIRT | Volume 6 Issue 1 | ISSN: 2349-6002

UART Transmitter Experiment in FPGA

S. S. Limaye[1], Hema Kale[2]
[1]Professor, E & TC Department, SVPCET, Nagpur, India
[2]Assistant Professor, E & TC Department, SVPCET, Nagpur, India

***Abstract*- Universal Asynchronous Receiver Transmitter (UART) is a very commonly used device in microprocessors. But implementing it in VHDL is not an easy task for students. This paper proposes a simple way of implementation by using set-ups available in the college labs. From faculty point of view, it will help them to encourage lab research in VHDL for exploring students' knowledge. Here UART is developed using FPGA. The specifications are: UART should continuously read a byte from the DIP switches and serially transmit it on the SDO (Serial Data Output) line as long as a pushbutton is pressed and the baud rate should be maintained constant as specified. Testing is done in two parts, first simulation of the circuit is carried out using Xilinx ISE package and then Hardware testing is done on Spartan III FPGA kit (Basys)**

Index Terms- UART, DIP, VHDL, FPGA, SDO, ISE

I. INTRODUCTION

A Universal Asynchronous Receiver Transmitter is an integrated circuit used for serial communications over a computer or peripheral device serial port. It is the microchip with programming that controls a computer's interface to its attached serial devices. Specifically, it provides the computer with the RS-232C Data Terminal Equipment interface so that it can exchange data with modems and other serial devices.

II EXPERIMENTATION BACKGROUND

A. Block diagram

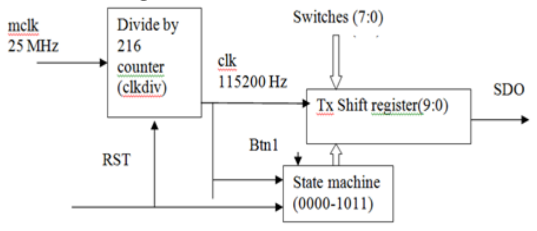

Figure 1: Block diagram of UART

The UART consists of 3 blocks as shown in fig. 1 above

1. Clock divider divides the on-board 25 MHz clock by 216 to provide approximately 115200 Hz signal, "clk".
2. State machine. It is a 4 bit state register. On reset, it is set to 0000 (Idle). If btn1 is pressed, the state changes to 0001, else it remains 0000. When the state is between 0001 and 1011, every clock pulse increments it. On reaching 1011, the state again becomes 0000. State machine is shown in fig. 2

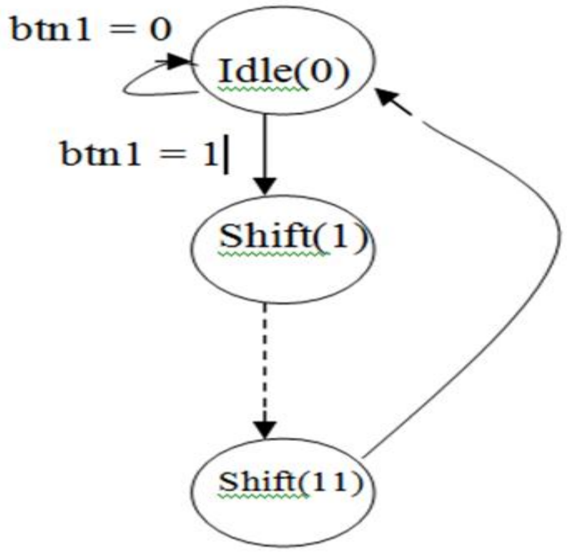

Figure 2: State machine

3. Transmitter shift register is shown in fig. 3. It is a 10 bit shift register named "tsr". In state 0000, it sets all bits to 1. In state 0001, reads the switches and stores them in tsr bits 8:1. Bit 9 (stop bit) is made 1 and bit 0 (start bit) is made 0. In states 0002 to 1011, it shifts tsr right by 1 bit, while filling the MSB with 1. The tsr(0) bit is given out as SDO.

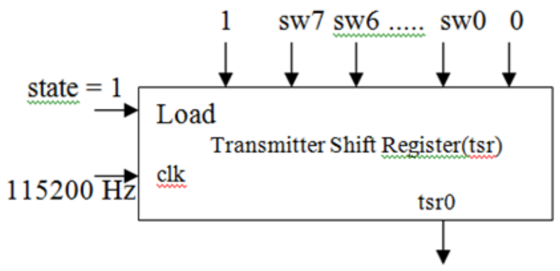

Figure 3: Transmitter Shift Register

III VHDL IMPLEMENTATION

A VHDL implementation for above three blocks i.e. is given below,

1. Clock divider
2. State machine
3. Transmitter shift register

VHDL code:

```
--tx main entity
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity uart is
    port(rst,mclk,btn1: in std_logic;
        sw: in std_logic_vector(7 downto 0);
        sdo: out std_logic
         );
  end uart;
architecture v1 of uart is
              signal tsr:std_logic_vector(9 downto 0);
              signal state:std_logic_vector(3 downto 0);
              signal clkdiv:std_logic_vector(8 downto 0);
              signal clk:std_logic;
begin
              sdo <= tsr(0);
              --Clock divides mclk 25 MHz to produce
baud rate 115200
              process(mclk,rst)
              begin
      if rst ='1' then
        clkdiv <= (others =>'0');
                              clk <= '0';
      elsif mclk'event and mclk = '1' then
                            clkdiv <= clkdiv +1;
if clkdiv = 108 then
                            clkdiv <= (others =>'0');
                            clk <= not clk;
                                          end if;
                              end if;
              end process;
              --TSR
              process(clk,rst,btn1)
              begin
      if rst ='1' then
        tsr <= (others =>'1');
      elsif clk'event and clk = '1' then
                                if state = "0000" then
                                      tsr <= (others =>'1');
        elsif state = "0001" then
                                            tsr <= '1' & sw & '0';
        elsif state > "0001" and state < "1011"then
                              tsr <= '1' & tsr(9 downto 1);
                                end if;
                              end if;
              end process;
              -- State machine
              process(clk,rst,btn1)
              begin
      if rst ='1' then
        state <= (others =>'0');
      elsif clk'event and clk = '1' then
                                if state = "0000" then
                                            if btn1= '0' then
                                            state <= "0000";
                                            else state <= "0001";
                                            end if;
        elsif state >= "0001" and state < "1011"then
                                            state <= state +1;
                                elsif  state = "1011" then
                                            state <= (others =>'0');
                                end if;
                              end if;
              end process;
end v1;
```

IV SIMULATION RESULTS

Simulation of the circuit is carried out using Xilinx ISE package. A test bench is written here. It instantiates the UART as a component. It generates mclk signal, RST signal, sw signal and btn1 signal. Note that the mclk generator stops after producing 8000 pulses. This automatically stops simulation. If we don't do it, then the simulator continues to run indefinitely and we have to stop it manually. The waveforms of clk and SDO can be observed. Change sw positions and observe the waveform. The test bench program is as follows.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
ENTITY uart_tb IS
END uart_tb;
ARCHITECTURE behavior OF uart_tb IS
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT uart
  PORT(
    rst : IN  std_logic;
    mclk : IN  std_logic;
    btn1 : IN  std_logic;
    sw : IN  std_logic_vector(7 downto 0);
    sdo : OUT  std_logic
   );
END COMPONENT;
--Inputs
 signal rst : std_logic := '0';
 signal mclk : std_logic := '0';
 signal btn1 : std_logic := '0';
 signal sw : std_logic_vector(7 downto 0) := (others => '0');
  signal clkcount : integer:=0;
--Outputs
signal sdo : std_logic;
-- Clock period definitions
constant mclk_period : time := 40 ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: uart PORT MAP (
    rst => rst,
    mclk => mclk,
    btn1 => btn1,
    sw => sw,
    sdo => sdo
   );
-- Clock process definitions
  mclk_process :process
  begin
          mclk <= '0';
          wait for mclk_period/2;
          mclk <= '1';
          wait for mclk_period/2;
          clkcount <= clkcount + 1;
          if clkcount > 8000 then
                    wait;
                    end if;
end process;
-- Stimulus process
 stim_proc: process
  begin
          rst <= '1';
          btn1 <= '0';
          sw <= "00010101";
  wait for 2 us;
          rst <= '0';
          wait for 2 us;
          btn1 <= '1';
          wait for 20 us;
          --btn1 <= '0';
  wait for mclk_period*20000;
  wait;
 end process;
END;
```

Simulation Results are shown below in fig. 4 below,

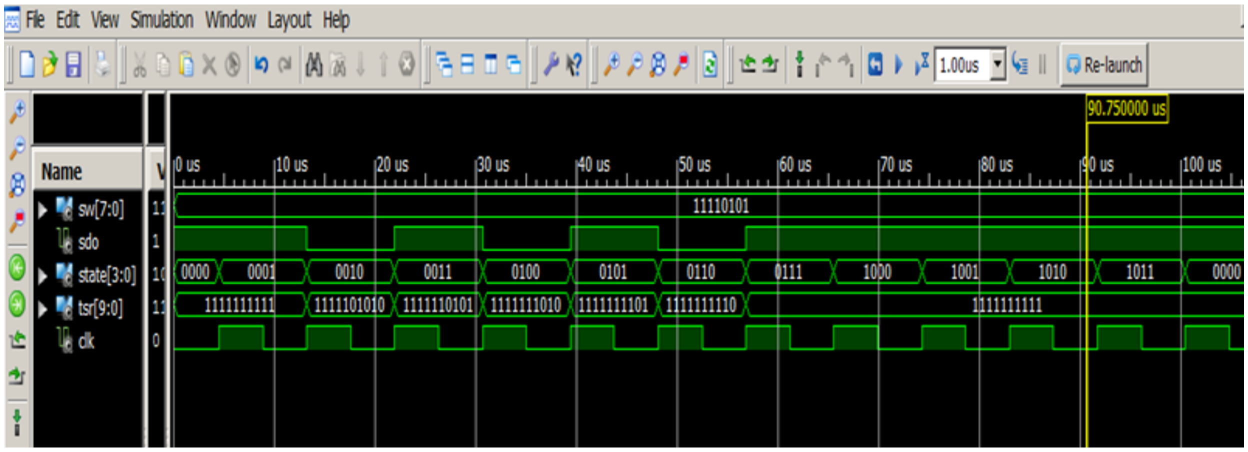

Figure 4: Simulation Result

V HARDWARE IMPLEMENTATION

Hardware testing is done on Spartan III FPGA kit (Basys)
UCF file was created for pin definitions on BASYS Spartan III kit as follows.

```
NET "sw<7>" LOC = "N3";
NET "sw<6>" LOC = "E2";
NET "sw<5>" LOC = "F3";
NET "sw<4>" LOC = "G3";
NET "sw<3>" LOC = "B4";
NET "sw<2>" LOC = "K3";
NET "sw<1>" LOC = "L3";
NET "sw<0>" LOC = "P11";
NET "rst"       LOC = "G12";  #Button 0
NET "btn1"      LOC = "C11";  #Button 1
NET "mclk"      LOC = "B8";
NET "sdo"       LOC = "C9";
NET "clkout"    LOC = "A9";
```

The program was downloaded on SPARTAN III kit. SDO pin was connected to RxD of USB-TTL converter and GND to GND. The other end of USB-TTL converter was connected to a USB port on PC. The hardware setup is shown below,

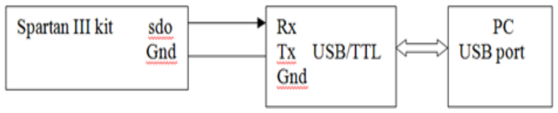

Figure 5. Hardware Implementation

The proposed work is carried out by using the software & hardware available in the college labs. The main idea is to clear the basic technical concepts by innovative experimentation in the labs and to make teaching learning process more interesting. Faculty of UG departments can use this paper as a content beyond syllabus manual for experimentation with VHDL.

IX. CONCLUSION

This paper presented the UART Transmitter design using VHDL & Implementation in FPGA, This was the innovative experimentation done from the student point of view. This work can be further extended to UART Receiver implementation in FPGA.

REFERENCES

[1] Dr. S. S. Limaye, "VHDL: A Design Oriented Approach", Macgraw Hill, 2008

[2] https://reference.digilentinc.com/_media/basys2:basys2_sch.pdf,Basys 2 Reference Manual

[3] Ramesh Gaonkar, "Microprocessor Architecture, Programming and Applications with The 8085, PENRAM,2013