COL215P

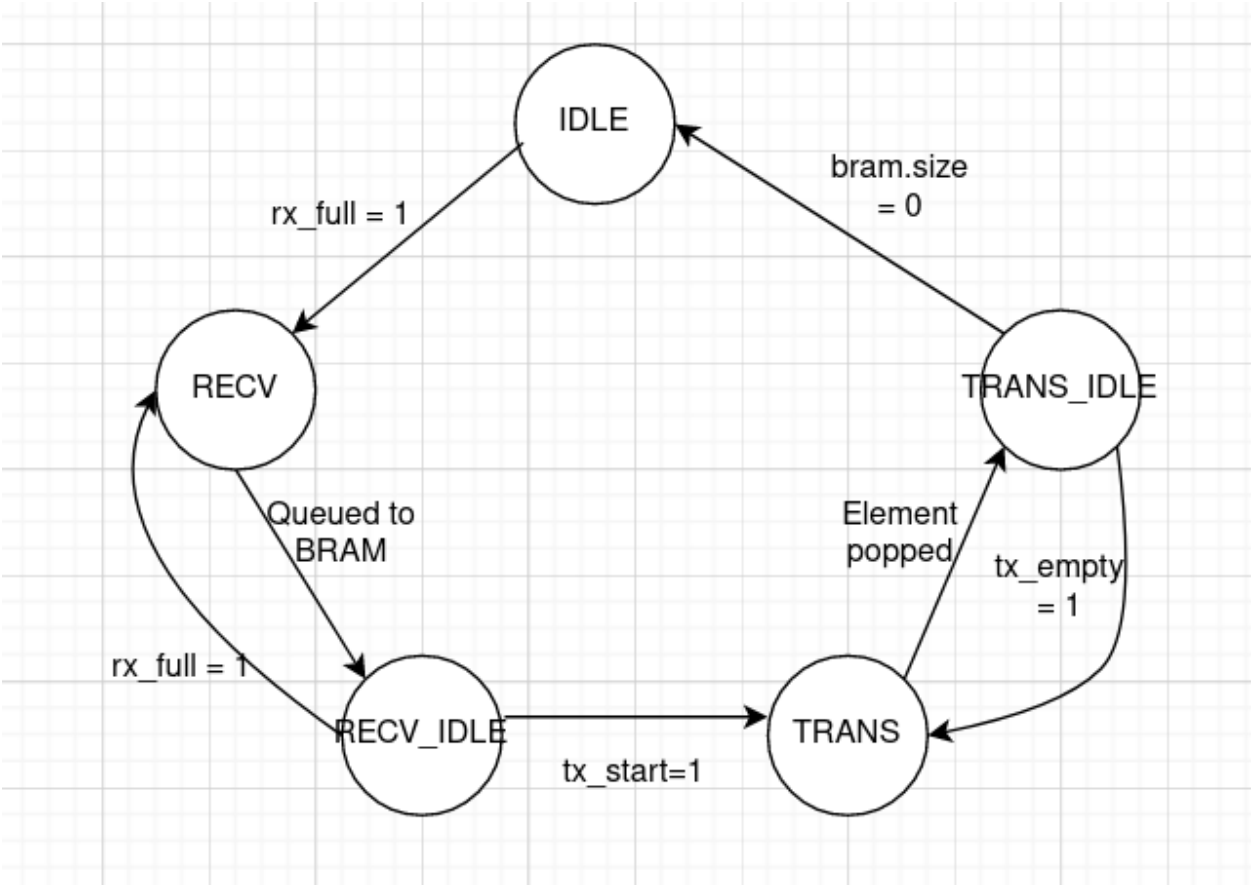
Assignment 10 Report

Nalin Wadhwa 2019CS10375

Sidharth Agarwal 2019CS50661

In the 10th Assignment, we designed a file transferer between board and PC. The assignment was built on the work done in modules of serial trasnferer, display and fifo buffer. In our master.vhd we used 5 states as described in the diagram below. We basically queued the input read in RECV and waited in the IDLE or RECV_IDLE to read the 8 bit input. And for transferring, we first switched to TRANS state from RECV_IDLE. Later we popped the element in TRANS and sent the date back to PC in TRANS_IDLE, until bram size is again 0.

In this code, we used 516 Flip flops, 375 LUTs, 0.5 BRAM, and 0 DSPs. A complete list of resources can be found in the attached master_util_synth.rpt file, alongside the source files(master.vhd, rec.vhd, trans.vhd, bram.vhd,singlessd.vhd,display.vhd,btn_debouncer.vhd), constraint file(Basys-3-Master.xdc) and the bitstream file(master.bit).



Project Summary | master.vhd | bram.vhd | rec.vhd | trans.vhd | btn_debouncer.vhd

/home/dual/cs5190661/vivado/assignment10/assignment10.srcs/sources_1/new/master.vhd

```
94  get_btnR: entity work.btn_debouncer(Behavioral)
95  port map (clk, btnR, debounced_btnR);
96
97  --for displaying input on the board..
98  get_display: entity work.display(Behavioral)
99  port map (clk, display_input, display_brightness, c, a);
100
101 process(clk)
102 begin
103     if rising_edge(clk) then
104         case state is
105         -- shifts from IDLE when recieves the full 8bits
106             when IDLE =>
107                 if rx_full = '1' then
108                     state <= RECV;
109                     bram_read_value <= recv_data_out;
110                     display_input <= recv_data_out;
111                     bram_read_flag <= '1';
112                 end if;
113         -- first adds the input to memory queue and switches to RECV_IDLE
114             when RECV =>
115                 bram_read_flag <= '0';
116                 if rx_full = '0' then
117                     state <= RECV_IDLE;
118                 end if;
119         -- waits to recieve the 8 bits or shit to TRANS on btnR pressing.
120             when RECV_IDLE =>
121                 if rx_full = '1' then
122                     state <= RECV;
123                     bram_read_value <= recv_data_out;
124                     display_input <= recv_data_out;
125                     bram_read_flag <= '1';
126                 elsif tx_start = '1' then
127                     state <= TRANS;
128                     bram_write_flag <= '1';
129                 end if;
130         -- first pops the element from queue and shifts to TRANS_IDLE
131             when TRANS =>
132                 bram_write_flag <= '0';
133                 if sent_flag = '0' then
134                     send_data_in <= bram_write_value;
```

Project Summary | controller.vhd | bram.vhd | btn_debouncer.vhd | IP_bram_wrapper.vhd

/home/dual/cs5190661/vivado/assignment9/assignment9.srcs/sources_1/new/bram.vhd

```
72
73  get_IP_bram_module: entity work.IP_bram_wrapper(Behavioral)
74  port map (clk, bram_en, bram_we, bram_addr, bram_din, bram_dout);
75  process(clk)
76  begin
77      if rising_edge(clk) then
78          case state is
79          --default state where both push and pop are available
80              when IDLE =>
81                  read_done <= '0';
82                  write_done <= '0';
83                  if read_flag = '1' then
84                      state <= ST_PUSH;
85                  elsif write_flag = '1' then
86                      state <= ST_POP;
87                      bram_addr <= std_logic_vector(to_unsigned(tail, 3));
88                  end if;
89          -- only pop is available in FULL
90              when FULL =>
91                  write_done <= '0';
92                  if write_flag = '1' then
93                      read_done <= '0';
94                      state <= ST_POP;
95                      bram_addr <= std_logic_vector(to_unsigned(tail, 3));
96                  elsif read_flag = '1' then
97                      read_done <= '1';
98                  else
99                      read_done <= '0';
100                 end if;
101         -- only push is available in EMPTY
102             when EMPTY =>
103                 read_done <= '0';
104                 if read_flag = '1' then
105                     state <= ST_PUSH;
106                     write_done <= '0';
107                 elsif write_flag = '1' then
108                     write_done <= '1';
109                 else
110                     write_done <= '0';
111                 end if;
112         -- pushes read_value in FIFO queue at head
113         -- goes to FULL no space
114             when ST_PUSH =>
115                 bram_addr <= std_logic_vector(to_unsigned(head, 3));
116                 bram_din <= read_value;
117 --                bram(head) <= read_value;
118                 read_done <= '0';
```

Project Summary | master.vhd | bram.vhd | **rec.vhd** | trans.vhd | btn_debouncer.vhd

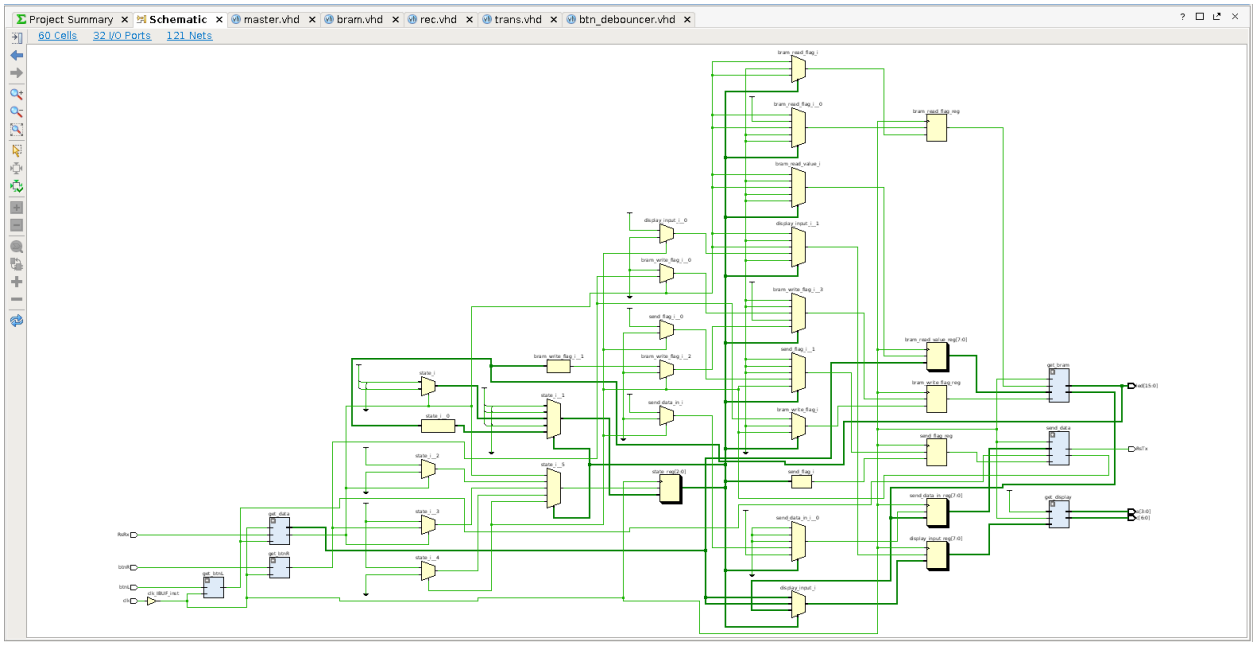
/home/dual/cs5190661/vivado/assignment10/assignment10.srcs/sources_1/imports/vivado/2019CS10375_2019CS50661_8/rec.vhd

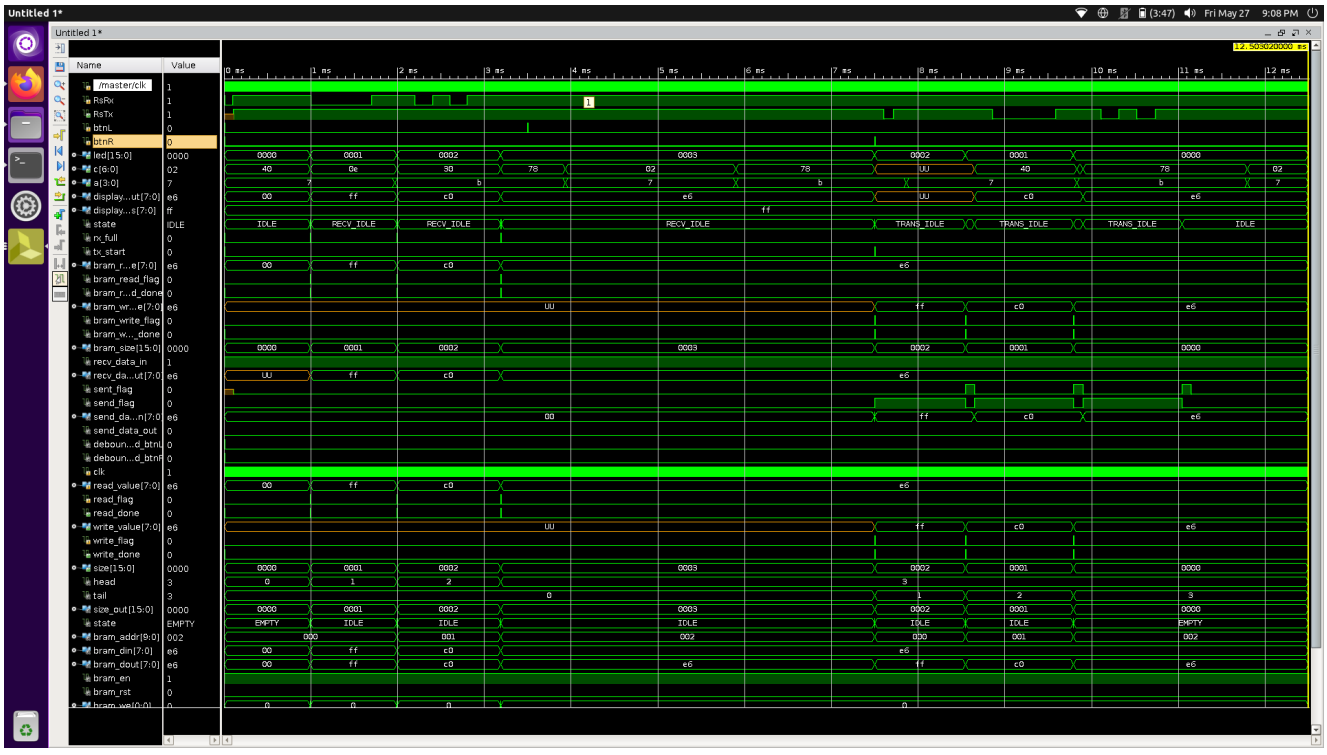
```
68      end if;
69  end process;
70
71  --clk which updates at 16 * baud rate
72  fast_baud <= '1' when counter = fast_baud_lim else '0';
73
74  process(clk)
75  variable bit_ctr: integer := 0;
76  variable bits: integer := 0;
77  begin
78      if rising_edge(clk) then
79          if reset = '1' then
80              -- reset button (btnC) sets any ongoing condition back to the IDLE position and forgets any on going input
81              state <= IDLE;
82              stored_data <= (others => '0');
83              data_out <= (others => '0');
84              bit_ctr := 0;
85              bits := 0;
86              recv_flag <= '0';
87          else
88              if state = IDLE then
89                  -- wait till it finds 0 as the start bit
90                  -- it detects at 16*baud rate to as to later be able to more accurately detect bits
91                  if fast_baud = '1' then
92                      recv_flag <= '0';
93                      stored_data <= (others => '0');
94                      bit_ctr := 0;
95                      bits := 0;
96                      if data_in = '0' then
97                          --start checking for data since start bit in
98                          state <= START;
99                      end if;
100                 end if;
101             elsif state = START then
102                 if fast_baud = '1' then
103                     -- wait 8 more fast_bauds to get to mid bit of start bit
104                     -- after verifying the start bit 8 times, it switched to the data state
105                     if data_in = '0' then
106                         if bit_ctr = 7 then
107                             state <= GET;
108                             bit_ctr := 0;
109                         else
```

Project Summary | master.vhd | bram.vhd | rec.vhd | **trans.vhd** | btn_debouncer.vhd

/home/dual/cs5190661/vivado/assignment10/assignment10.srcs/sources_1/imports/vivado/2019CS10375_2019CS50661_8/trans.vhd

```
55  begin
56
57  process(clk)
58  begin
59      if rising_edge(clk) then
60          if counter = baud_lim then
61              counter <= (others => '0');
62          else
63              -- counter is set to count the time period for baud rate (9600 Hz) from the clock (100 MHz)
64              counter <= counter + 1;
65          end if;
66      end if;
67  end process;
68
69  -- set 1 when one baud passes
70  baud <= '1' when counter = baud_lim else '0';
71
72  process(clk)
73  variable bits: integer := 0;
74  begin
75      if rising_edge(clk) then
76          -- reset button (btnC) sets ongoing Tx back to idle whichever state it is in
77          if reset = '1' then
78              data_out <= '1';
79              sent_flag <= '0';
80              bits := 0;
81              state <= IDLE;
82          elsif baud = '1' then
83              case state is
84                  when IDLE =>
85                      -- wait till send_flag from mgr is 1 ( till then send '1' on line)
86                      data_out <= '1';
87                      sent_flag <= '0';
88                      bits := 0;
89                      if send_flag = '1' then
90                          state <= START;
91                      end if;
92                  when START =>
93                      -- send start bit initially before sending the bits
94                      data_out <= '0';
95                      state <= SEND;
96                  when SEND =>
```


Project Summary
Schematic
master.vhd
bram.vhd
rec.vhd
trans.vhd
btn_debouncer.vhd
60 Cells
32 I/O Ports
121 Nets


Untitled 1*
Fri May 27
9:08 PM
Name
Value
/master/clk
RsRx
RsTx
btnL
btnR
led[15:0]
c[6:0]
a[3:0]
state
rx_full
tx_start
read_value[7:0]
read_flag
read_done
write_value[7:0]
write_flag
write_done
size[15:0]
head
tail
size_out[15:0]
bram_addr[9:0]
bram_din[7:0]
bram_dout[7:0]
bram_en
bram_rst
IDLE
RECV_IDLE
TRANS_IDLE
EMPTY