Relatório Final:
Hardware de transmissão de dados

Alline Silva Domingos * Layssa Alves Pacheco †

Dezembro 2016

Resumo

Como projeto final da disciplina de Dispositivos Lógicos Programáveis I desenvolveu-se em linguagem de descrição de hardware VHDL um sistema de transmissão de dados em 1bps e 1000bps. Para isso, a correta transmissão exigiu a construção de blocos de verificação de paridade par, serializador, deserializador e a transformação dos dados binários em código hexadecimal para visualizar no led de sete segmentos da placa de FPGA DE2-115 da TERASIC. Por último, também conectou-se leds da placa para facilitar o compreensão dos sinais referentes as diferentes frequência, as atividades de transmissão e recepção, assim como o de erro na paridade dos dados de entrada e saída, além do próprio bit de paridade.

Palavras-chaves: FPGA. Transmissão. Serial. Dispositivos Lógicos Programáveis.

Introdução

O projeto final da disciplina de Dispositivos Lógicos Programáveis I da sexta fase do curso de Engenharia de Telecomunicações do semestre 2/2016 consiste em um circuito de transmissão e recepção serial implementado em linguagem de descrição de hardware VHDL.

*Aluna do curso de Engenharia de Telecomunicações. Contato: lline.domingosdmg@gmail.com

†Aluna do curso de Engenharia de Telecomunicações. Contato: layssapacheco@gmail.com

O circuito possui sete portas de entrada e de saída. As portas de entrada são:

- baud_rate_sel: std_logic;
- clk: std_logic;
- rst: std_logic;
- sel: std_logic;
- data_in_1: std_logic_vector(6 downto 0);
- data_in_2: std_logic_vector(6 downto 0);
- tipo_ssd: std_logic.

As portas de saída são:

- baud_rate_out: std_logic;
- sample_rate_out: std_logic;
- ssd_in_lsb: std_logic_vector(6 downto 0);
- ssd_in_msb: std_logic_vector(6 downto 0);
- ssd_out_lsb: std_logic_vector(6 downto 0);
- ssd_out_msb: std_logic_vector(6 downto 0);
- bit_error: std_logic.

Além disso, há um multiplexador e dez blocos compostos de seis componentes diferentes, sendo que dois deles também tem um componente cada. O circuito completo é apresentado nos RTLs as Figuras 01 e 02.

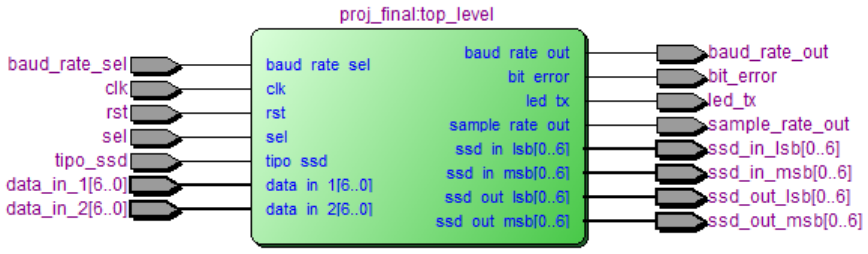

Figura 1 – RTL compactado

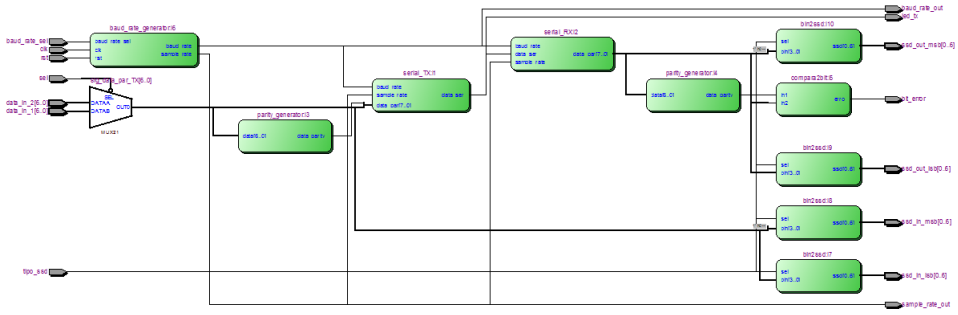

Figura 2 – RTL descompactado

Os blocos e o multiplexador que compõem o harware são descritos a seguir:

- Multiplexador: quando a entrada sel é '1' seleciona para o hardware as chaves SW3 até SW9 que correspondem a porta de entrada data_in_1, caso seja '0' seleciona as chaves SW10 até SW16 que correspondem a porta de entrada data_in_2;
- I3 (parity_generator): recebe na porta de entrada data o dado escolhido pelo multiplexador. Calcula a paridade par e envia pela porta de saída data_parity;
- I4 (parity_generator): recebe o dado da porta de saída data_par do bloco serial_RX na porta de entrada data. Calcula a paridade par e envia pela porta de saída data_parity;
- I1 (serial_TX): recebe na porta de entrada data_par o dado concatenado da porta de saída data_parity do bloco parity_generator com o escolhido no multiplexador. Além disso, o bloco possui as portas de entrada de controle, baud_rate e sample_rate. A baud_rate controla a frequência de entrada e saída dos dado e o sample_rate decide se o bloco transmitirá ou não. O bloco é responsável em transformar o dado de formato serial em paralelo, o qual sairá pela porta data_ser;
- I2 (serial_RX): recebe na porta de entrada data_par o dado da porta de saída data_ser do bloco serial_TX. Além disso, o bloco possui as portas de entrada de controle, baud_rate e sample_rate. A baud_rate controla a frequência de entrada e saída dos dado e o sample_rate decide se o bloco transmitirá ou não. O bloco é responsável em transformar o dado de formato paralelo em serial, o qual sairá pela porta data_par;
- I6 (baud_rate_generator): recebe os dados da porta de entrada do harware baud_rate_sel, clk e rst, as quais possuem o mesmo nome

no baud_rate_generator. O rst decide se o bloco funcionará ou permanecerá parado, o que implica em definir o funcionamento de todo o circuito. O clk é o sinal de clock proveniente da placa FPGA DE2-115. Já o baud_rate_sel é o sinal que seleciona qual será a frequência de transmissão dos dados, 1bps ou 1000bps, o qual sairá através da porta de saída baud_rate. Além disso, o bloco possui a porta de saída sample_rate, que é responsável por controlar os blocos seriais, ativando-os alternativamente;

- I7 (bin2ssd): recebe na porta de entrada bin os quatro bits LSB do dado selecionado no multiplexador. O bloco é responsável por traduzir o código binário recebido na entrada e enviar para a porta de saída ssd o código hexadecimal referente. Dependendo do sinal recebido na porta de entrada sel da pela porta de entrada do harware tipo_ssd, a saída é invertida, então os leds de sete segmento que acendem são aqueles que não representam o dado chaveado no circuito. A porta de saída ssd é conectada a porta de saída ssd_in_lsb do harware, que está ligada a um dos led de sete segmentos;

- I8 (bin2ssd): recebe na porta bin os três bits MSB da porta de entrada do harware concatenados com um bit '0' na. O bloco é responsável por traduzir o código binário recebido na entrada e enviar para a porta de saída ssd o código hexadecimal referente. Dependendo do sinal recebido na porta de entrada sel da pela porta de entrada do harware tipo_ssd, a saída é invertida, então os leds de sete segmento que acendem são aqueles que não representam o dado chaveado no circuito. A porta de saída ssd é conectada a porta de saída ssd_in_msb do harware, que está ligada a um dos led de sete segmentos;

- I9 (bin2ssd): recebe na porta de entrada bin os quatro bits LSB da porta de saída data_par do bloco serial_RX. O bloco é responsável por traduzir o código binário recebido na entrada e enviar para a porta de saída ssd o código hexadecimal referente. Dependendo do sinal recebido na porta de entrada sel da pela porta de entrada do harware tipo_ssd, a saída é invertida, então os leds de sete segmento que acendem são aqueles que não representam o dado chaveado no circuito. A porta de saída ssd é conectada a porta de saída ssd_out_lsb do harware, que está ligada a um dos led de sete segmentos;

- I10 (bin2ssd): recebe na porta bin os três bits MSB da porta de saída data_par do bloco serial_RX concatenados com um bit '0' na. O bloco é responsável por traduzir o código binário recebido na entrada e enviar para a porta de saída ssd o código hexadecimal referente. Dependendo do sinal recebido na porta de entrada sel da pela porta de entrada do harware tipo_ssd, a saída é invertida, então os leds de sete segmento que acendem são aqueles que não representam o dado chaveado no circuito.

A porta de saída ssd é conectada a porta de saída ssd_out_msb do harware, que está ligada a um dos led de sete segmentos;

- I5 (compara_2bits): recebe na porta de entrada in2 o último bit da porta de saída data_par do bloco serial_RX, esse bit corresponde ao bit de paridade. Já na porta de entrada in1 recebe o bit da porta de saída data_parity do bloco I4 (parity_generator). O compara_2bits é responsável por comparar os dois valores de paridade para verificação de erro. Com isso, caso os valores sejam diferentes a porta de saída erro receberá '1', caso contrário '0'. A porta de saída erro é conectada a porta de saída bit_error do harware.

Resumidamente a função do circuito de transmissão serial é transmitir dados de setes bits para leitura nos leds de sete seguimentos da placa de FPGA DE2-115 da TERASIC. Isso acontece através de um circuito capaz de variar a frequência de envio e promover a verificação de erro por meio da paridade.

1 Simulações

Uma das etapas do projeto consistiu na realização de simulações, sendo divididas em três:

- Blocos: são as simulações de cada bloco de componente individualmente. Essas sismulações explicam visualmente o comportamento de cada bloco e facilitam que se encontre possíveis erros que comprometam todo o funcionamento do circuito.
- Circuito completo: é a simulação do circuito completo, ou seja, do trabalho em conjunto dos blocos de componente. Essa sismulação explica visualmente o comportamento do circuito, mostrando os sinais de entrada e suas saída, além dos sinais internos quando desejado.
- Placa: depois de promovidas as simulações anteriores, a simulação na placa é a que apresenta o trabalho completo, funcionando de acordo com o simulado no item anterior. Além disso, possibilita perceção do funcionamento pelos leds usados, que foram: os leds de sete segmentos, e leds para a saída de baud_rate_out (saída baud_rate do baud_rate_generator), bit_error (saída erro do compara_2bits), sample_rate (saída sample_rate do baud_rate_generator) e led_tex (saída data_parity do I3 parity_generator).

1.1 Blocos

O hardware construído consiste em seis blocos de componentes, os quais são: um deserializador (serial_RX), um serializador (serial_TX), um gerador de paridade par (parity_generator), um comparador de 2 bits (compara_2bits), um conversor de binário para hexadecimal (bin2ssd) e um conversor de frequência (baud_rate_generator).

1.1.1 serial_RX

O serial_RX possui três portas de entrada: data_ser (tipo std_logic), sample_rate (std_logic) e baud_rate (std_logic); e uma porta de saída data_par (std_logic_vector(n-1 downto 0)).

O serial_RX possui mais um componente, o comp_rx, e juntos trabalham para receber na entrada um bit em formato std_logic e realizar a saída de um dado em formato std_logic_vector(n-1 downto 0). Cada bit recebido na ordem n-1 até 0 é adicionado nas posições do dado de forma 0 até n-1.

Esse componente é controlado pelo sinal de sample_rate, no qual quando baixo transmite e quando é alto não. Além disso, a frequência de transmissão é definida pela entrada baud_rate.

Com o intuito de exemplificar melhor, as Figuras 03, 04 e 05 são simulações do serial_RX em ModelSim para a saída dos dados "10101010", "10110010"e "01100011".

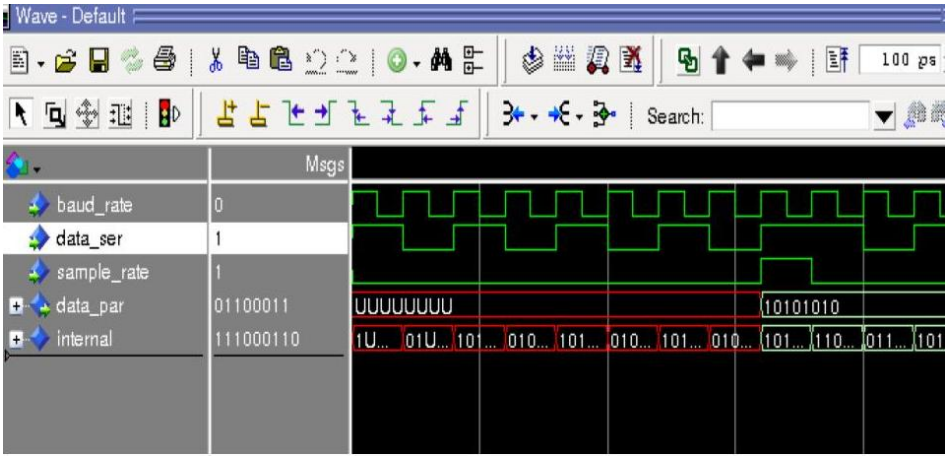

Figura 3 – Simulação "10101010"em serial_RX

1.1.2 serial_TX

O serial_TX possui três portas de entradas: data_par (std_logic_vector(0 to n-1)), sample_rate(std_logic) e baud_rate (std_logic); e uma porta de saída data_ser (std_logic).

O serial_TX possui mais um componente, o comp_tx, e juntos trabalham para receber na entrada um dado em formato std_logic_vector(n-1 downto 0) e realizar a saída de um bit por vez em formato std_logic de cada posição do dado de entrada.

Esse componente é controlado pelo sinal de sample_rate, no qual quando alto transmite e quando é baixo não. Além disso, a frequência de transmissão é definida pela entrada baud_rate.

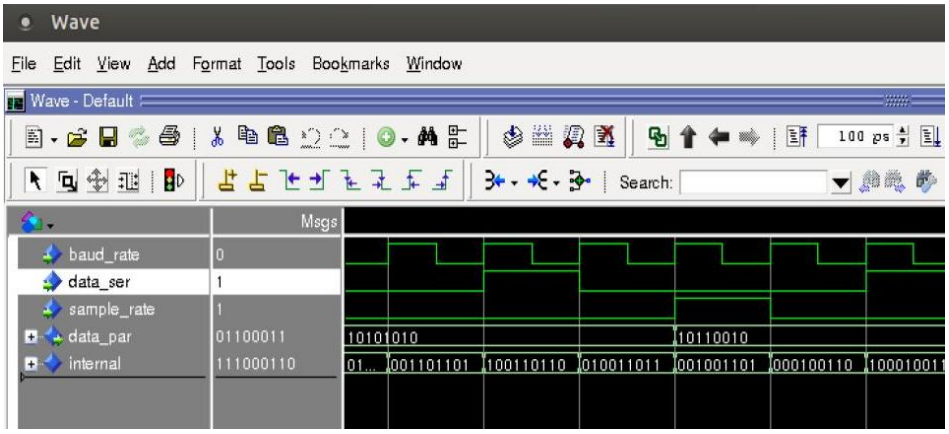

Figura 4 – Simulação "10110010"em serial_RX

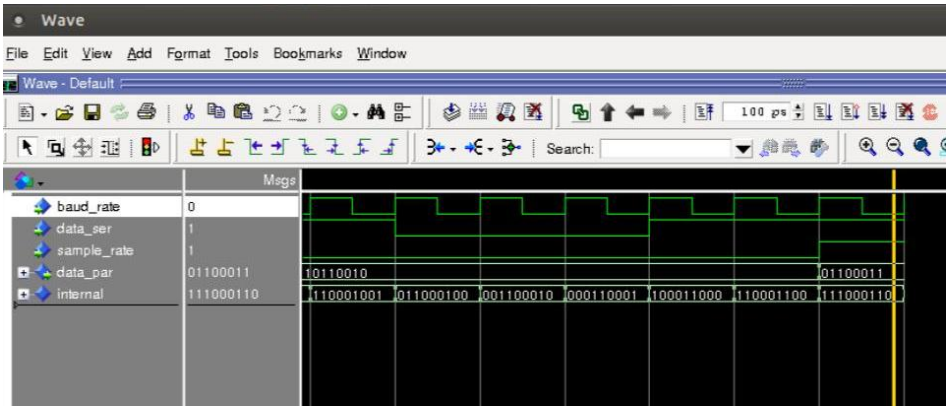

Figura 5 – Simulação "01100011"em serial_RX

Para exemplificar melhor, a Figura 06 é a simulação do serial_TX em ModelSim para as entradas dos dados "10101010", "1111111", "01010101"e "0000000".

1.1.3 parity_generator

O parity_generator possui uma porta de entrada data_par (std_logic_vector(0 to n-1)) e uma porta de saída data_parity (std_logic).

O parity_generator trabalha a paridade par do dado de entrada e a apresenta na saída. Se o dado tiver quantidade ímpar de bits '1', a saída é um bit '1'. Caso contrário, a saída é um bit '0'. No bloco I3, a saída é apresentada no LEDR11 da placa.

Com o intuito de exemplificar melhor, a Figura 07 é a simulação do parity_generator em ModelSim para as entradas dos dados "1111111", "0000000",

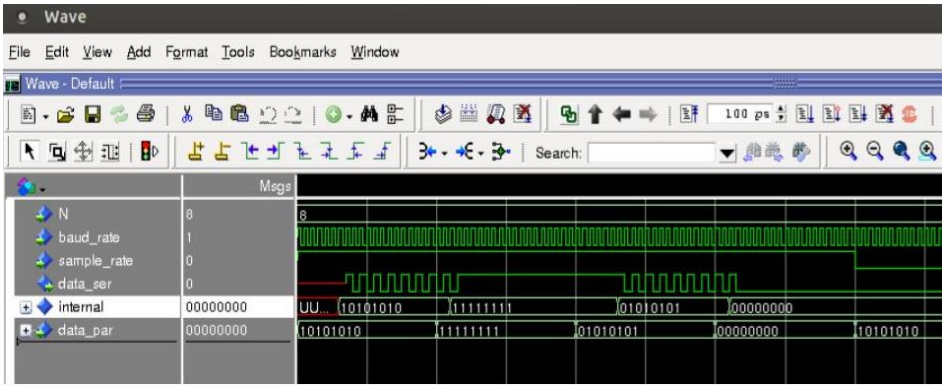

Figura 6 – Simulação serial_TX

"0000001"e "0000011".

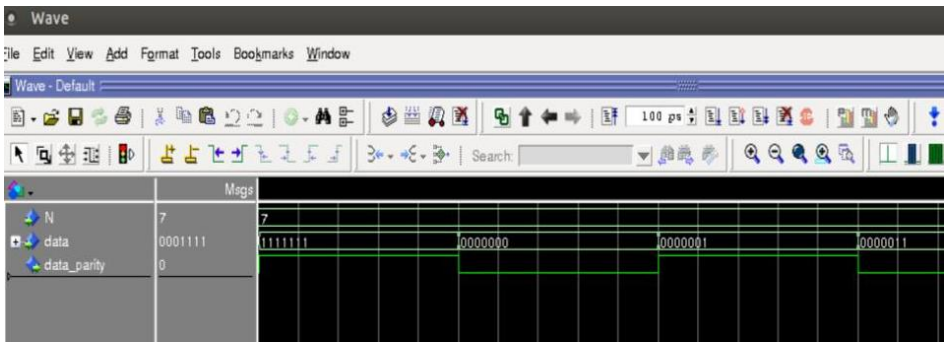

Figura 7 – Simulação parity_generator

1.1.4 compara_2bits

O compara_2bits possui duas portas de entrada, in1 (std_logic) e in2 (std_logic), e uma porta de saída, error (std_logic).

O compara_2bits é responsável por comparar dois bits de paridade, um que sai do I3 (parity_genertor) e outro do I4 (parity_generator) a fim de verificar se há erro na transmissão do dado de entrada para a saida, que será apresentanda no led de sete segmentos. O resultado é visível no LEDR15 da placa, caso tenha erro, o led acenderá, caso contrário, permanecerá apagado.

Para exemplificar melhor, a Figura 08 é a simulação do compara_2bits em ModelSim para as entradas dos dados iguais e diferentes.

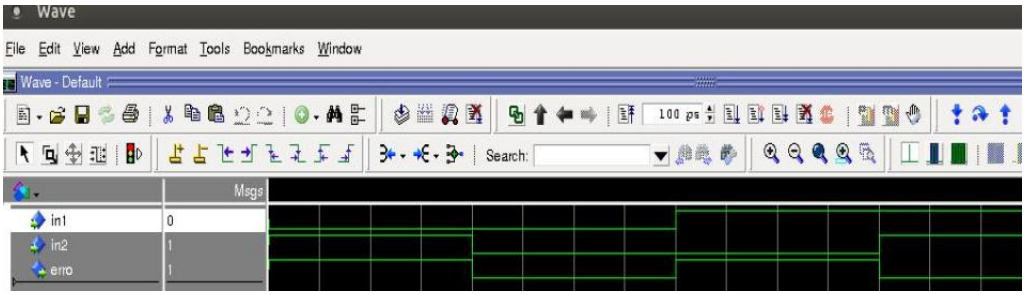

Figura 8 – Simulação compara_2bits

1.1.5 bin2ssd

O bin2ssd possui duas entradas: bin (std_logic_vector(0 to 3)) e sel (std_logic). Além disso, apresenta a saída ssd (std_logic_vector(0 to 6)).

O bin2ssd recebe na entrada um dado de quatro bits em código binário e o converte para um dado de sete bits em código hexadecimal, o qual é a saída do bloco, e inclusive visível no led de sete segmentos da placa.

Esse componente tem sua saída controlada pelo sinal de sel, o qual tem origem na entrada do hardware como tipo_ssd e no formato std_logic. Quando sel é '1' a saída é como esperado, já quando sel é '0', a saída é invertida devido a negação no final do bloco, assim os leds que ficariam apagados acendem e vice-versa.

Para exemplificar melhor, as Figuras 09 e 10 são as simulações do bin2ssd em ModelSim para quando o sel é '1' e '0' com diferentes valores de entrada.

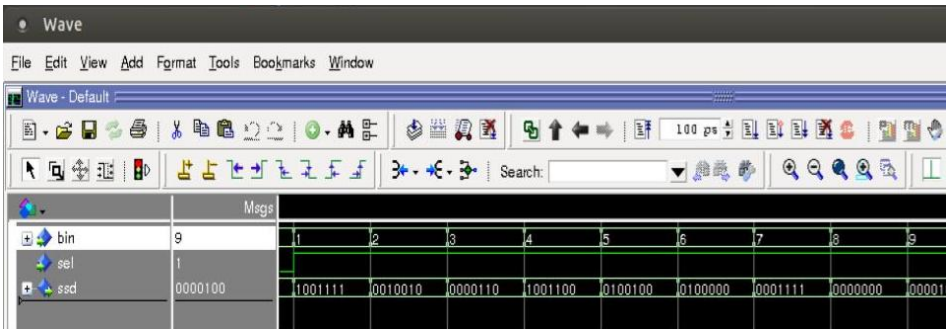

Figura 9 – Simulação bin2ssd com sel 1

No hardware final há quatro instanciações desse bloco, a fim de traduzir de binário para hexadecimal dos bits MSB e LSB do dado de entrada no circuito e o de saída do componente serial_RX.

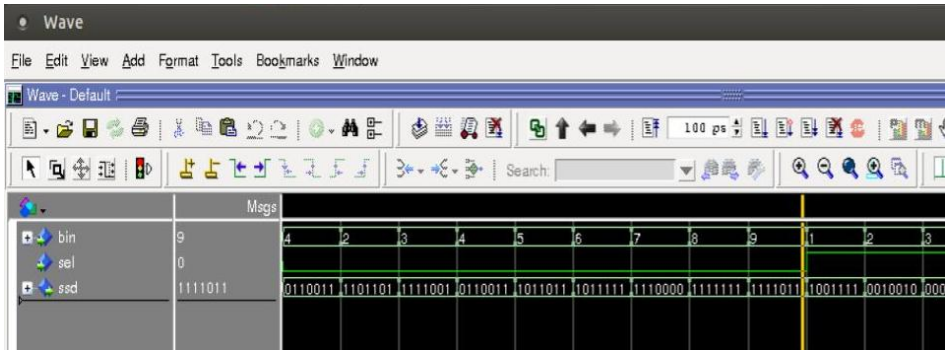

Figura 10 – Simulação bin2ssd com sel 1

1.1.6 baud_rate_generator

O baud_rate_generator possui três portas de entrada: baud_rate_sel (std_logic), clk (std_logic) e rst (std_logic); e duas portas de saída: baud_rate (std_logic) e sample_rate (std_logic).

O baud_rate_generator é o componente que recebe da entrada do harware o clock da placa usada e, a partir disso, produz duas frequências de clock diferentes. As frequências geradas saem pela porta baud_rate e são para taxas de transmissão de 1bps ou 1000bps. A escolha das frequências depende do sinal da porta baud_rate_sel, sendo que quando essa é '0' a transmissão ocorre em 1000bps e quando em '1' ocorre em 1bps. Além disso, a taxa de transmissão é visível pelo LEDR17, visto que a saída baud_rate do bloco é ligada a baud_rate_out que é conectada ao led da placa.

O rst é responsável por ativar ou desativar o trabalho do bloco e, portanto, controlar a transmissão ou não transmissão de todo o harware.

O sample_rate é a saída encarregada de controlar as atividades dos blocos serial_RX e serial_TX, sendo manipulado de acordo com o clock produzido pelas frequências. Quando o sinal interno mux_baud é '1', ou seja, quando o quando a frequência de trabalho é de 1bps e o baud_rate_sel é '1', o sample_rate recebe '1' a cada oito ciclos, ou seja, o serial_TX é permitido de funcionar e o serial_RX não. Já quando o sinal interno mux_baud é '0', ou seja, quando o quando a frequência de trabalho é de 1000bps e o baud_rate_sel é '0', o sample_rate recebe '0' a cada oito ciclos, ou seja, o serial_RX é permitido de funcionar e o serial_TX não. Assim como o baud_rate, o sample_rate também tem uma saída par um led da placa, LEDR13.

Para exemplificar melhor, as Figuras 11 e 12 correspondem as simulações do
baud_rate_generator em ModelSim.

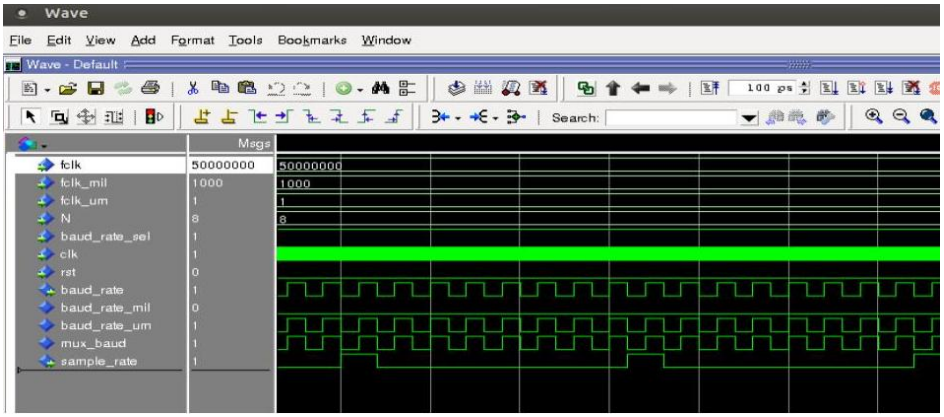

Figura 11 – Simulação baud_rate_generator para 1bps

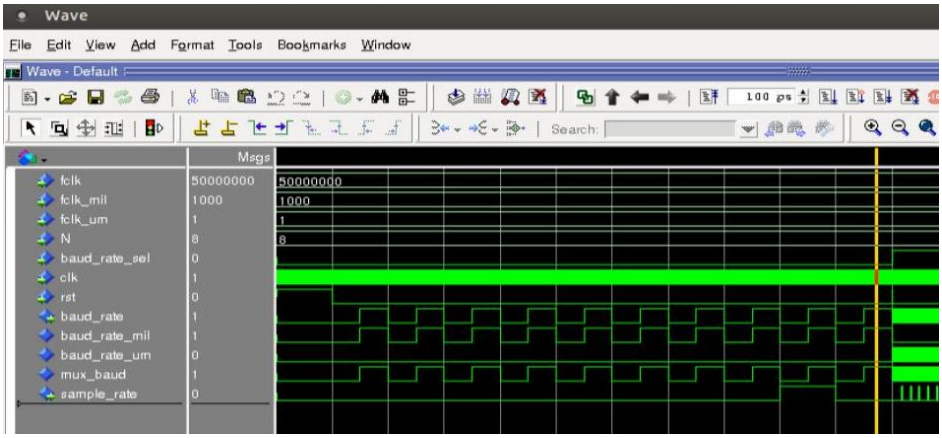

Figura 12 – Simulação baud_rate_generator para 1000bps

1.2 Circuito completo

O circuito completo consiste na escolha de dois dados, data_in_1 ou data_in_2, pelo multiplexador de entrada, e a transmissão pela sequência controlada pelo I6 (baud_rate_generator): I3 (parity_generator), I1 (serial_TX), I2 (serial_RX), I4 (parity_generator) e os ssd2bin (I7, I8, I9 e I10).

As simulações são apresentadas nas Figuras a seguir:

1.3 Placa

AS simulaç es na placa foram feitas em apresentação para o professor da disciplina no dia 21 de Dezembro de 2016.

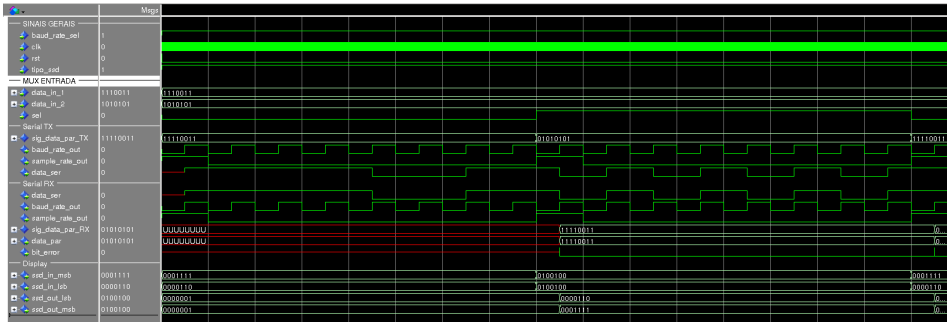

Figura 13 – Simulação parcial 01

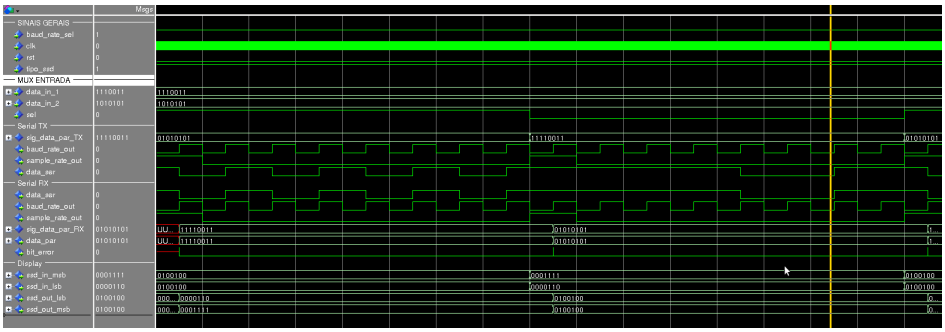

Figura 14 – Simulação parcial 02

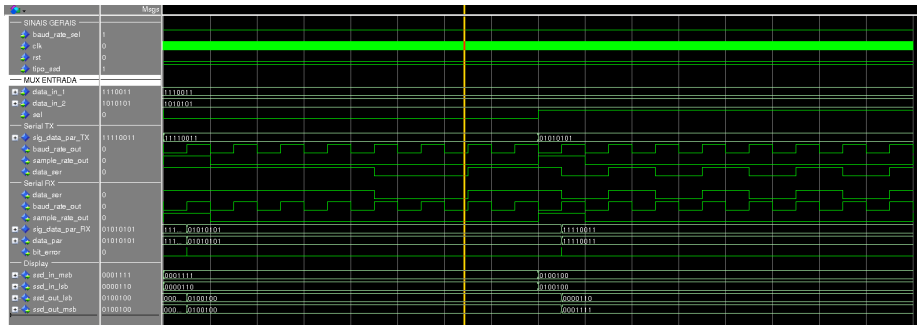

Figura 15 – Simulação parcial 03

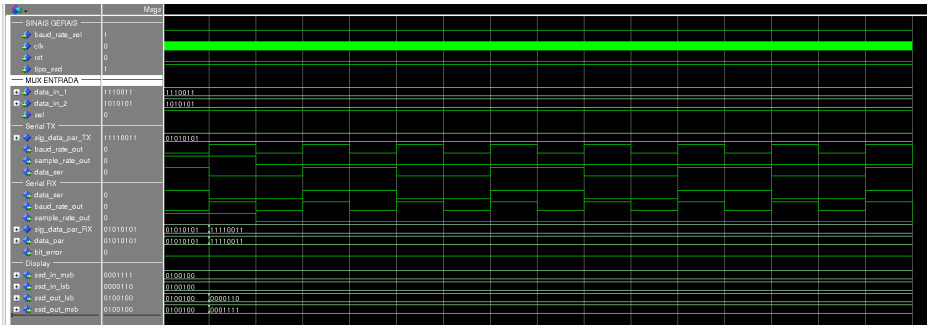

Figura 16 – Simulação parcial 04

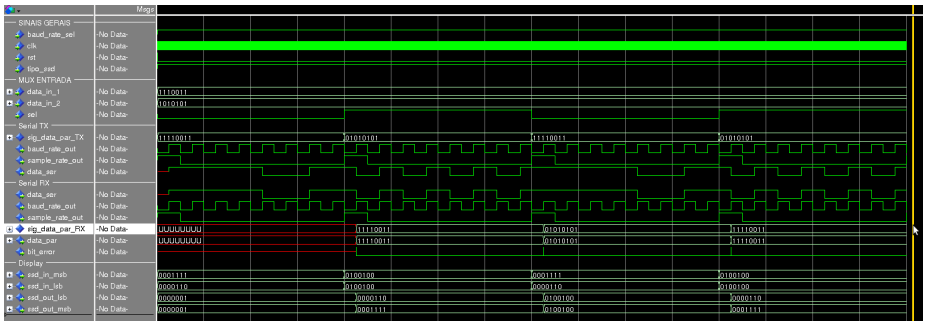

Figura 17 – Simulação completa