Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina
Campus São José – Engenharia de Telecomunicações
Unidade Curricular – DLP29006
Prof. Marcos Moecke
ALUNA: Luiza Alves da Silva

Relatório do projeto final

Sistema de transmissão serial assíncrono

Transmissão serial assíncrona é responsável por comunicar dados entre duas interfaces diferentes que não possuem os clocks sincronizados. Essa comunicação é feita enviando bit a bit, o transmissor envia uma sequência que entre os dados a serem transmitidos contém sinais de start e stop bit, que prepara o receptor para receber dados.

O projeto proposto é um sistema de transmissão serial assíncrono que possuem quatro palavras pré definidas e quando se pressiona o botão é transmitido uma palavra por vez. Pode também além de definir as palavras a serem transmitidas, pode-se definir o tempo de transmissão de cada bit e a paridade dos dados, se o receptor quer que seja paridade par ou ímpar, a paridade serve para a verificação de erros dos dados.

No desenvolvimento do mesmo, foi utilizada a linguagem de programação VHDL na IDE Quartus II, testado na simulação HDL ModelSim e depois implementado fisicamente no kit DE2-115 da família FPGA Cyclone® IV E - ALTERA.

Execução do projeto

O projeto foi dividido em componentes para execução de determinadas tarefas, até que se chegue no objetivo final de transmitir os dados desejados. A figura 1 é composta por um circuito RTL geral do projeto, pode-se observar as entradas de informações, os componentes e as saídas respectivamente. O componente u2 é responsável por analisar a palavra que foi escolhida pelo usuário para ser transmitida, e enviar letra a letra para a saída após apertar o botão, e convertida para bits. No componente u5 é realizado a lógica para o tempo de transmissão dos dados que foram pré estabelecidos pelo usuário. O componente u3 realiza a lógica para ser exibido em um display de sete segmentos a palavra que está sendo transmitida, e finalmente o componente u3 é responsável por realizar o envio da informação.

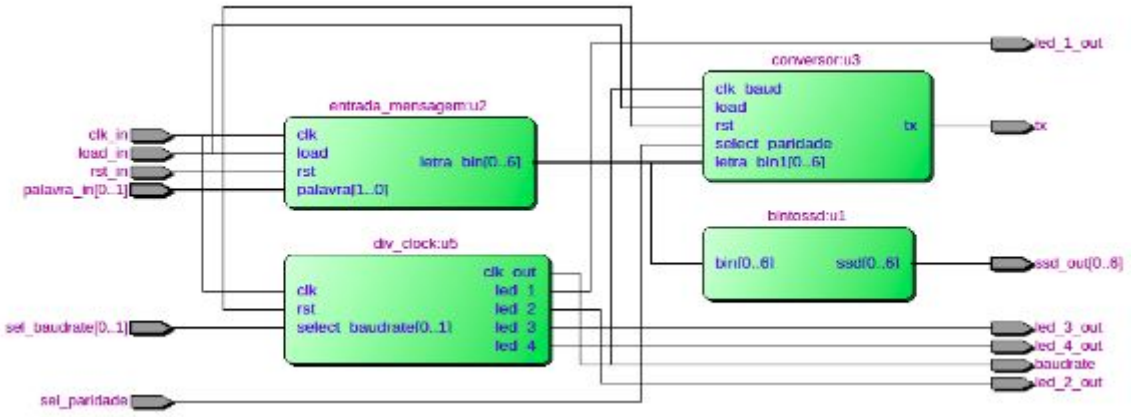

Figura 1 - Circuito RTL

Entradas:

clk_in → Clock de 50MHz vindo da placa.
load_in → Botão para a transmitir uma letra.
rst_in → Reiniciar a transmissão dos dados
palavra_in → Seleção da palavra
Sel_baudrate → Seleção do tempo de transmissão dos bits
Sel_paridade → Seleção da paridade

Saída:

tx → Saída do bit para ser transmitido
baud rate → saída do novo clock
ssd_out → visualização da letra transmitida
leds_out → visualização da seleção do baud rate

Componente u2: *entrada_mensagem*

Como dito anteriormente, este componente seleciona a palavra de acordo com o que foi escolhido pelo usuário, a entrada recebe dois valores através de uma chave e cada opção é uma palavra diferente que será transmitida. A seguir é visto uma tabela com as palavras.

Chaves	Palavra
00	Bala
01	Fago
10	Figo
11	Raso

Tabela 1 - Seleção da palavra

Na simulação deste componente pode ser observado o reset, que caso a pessoa queria reiniciar a transmissão desde a primeira letra. A cada pulso de clock é observado o comportamento do load, que simula o botão, e observa-se que ao ser pressionado, a palavra é separada em letras e enviada para a saída *letra_bin*.

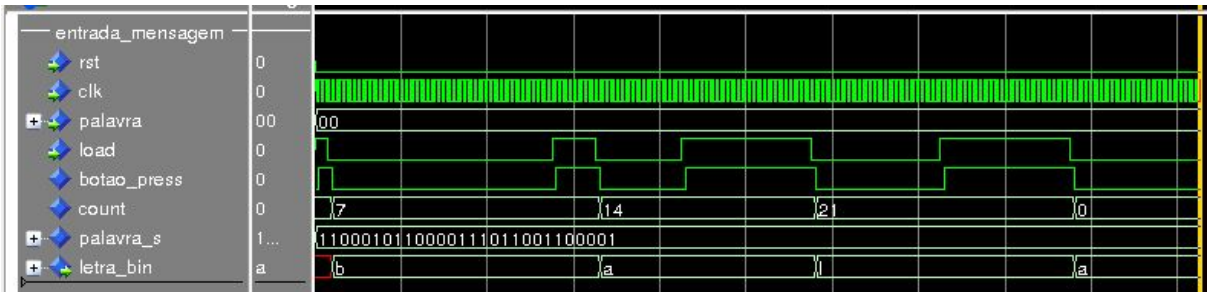

Figura 2 - Simulação no Modelsim do componente u2

Componente u5: *div_clock*

Nesta etapa pode ser visto a seleção do usuário em relação tempo de transmissão dos dados. A lógica baseia-se no clock de entrada que vem da placa FPGA de 50MHz, na simulação, Como pode ser visto na Figura 2, o valor foi alterado para 5 para melhor visualização da simulação, que a cada quantidade de pulsos de clock tem na saída *clock_out* um clock mais lento. O select_baudrate seleciona duas chaves e a cada seleção, tempo um tempo de transmissão diferente, conforme mostrado na tabela 2. Os quatro leds é para indicar visualmente qual baud rate foi escolhido.

Chaves	Tempo(s)
00	1
01	0,5
10	0,25
11	384,6u

Tabela 2 - Seleção do baud rate

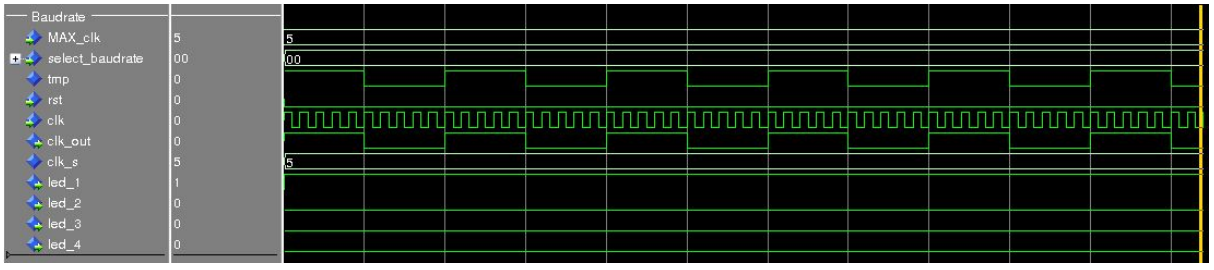

Figura 3 - Simulação no Modelsim do componente u5

Componente u3: *conversor*

Nesta etapa, a cada clock de saída do componente u2, ele analisa se o botão foi pressionado ou não, quando o botão for solto, ele transmite bit a bit para a saída tx. Neste código foi implementado a função de paridade dos bits e a transmissão do start e stop bit. Na figura a seguir observa-se a simulação do mesmo.

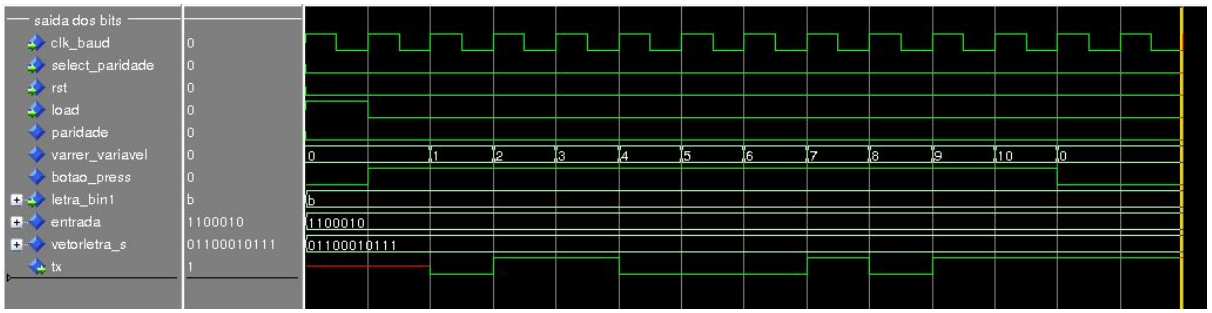

Figura 4 - Simulação no Modelsim do componente u3

Componente u1: *bintossd*

Neste componente a cada palavra transmitida é exibida no display, como pode ser visto na figura 5.

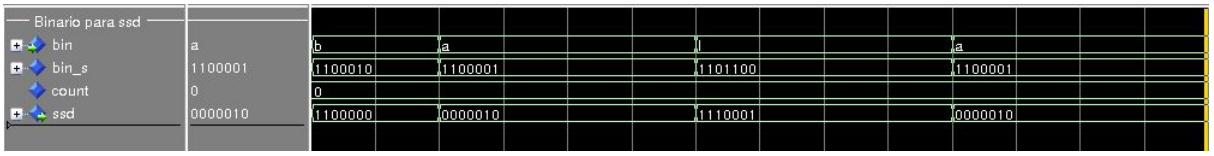

Figura 5 - Simulação no Modelsim do componente u1

Resultados obtidos

Na imagem a seguir, pode ser visualizado a simulação geral do projeto que pode ser visto no RTL, com todos os componentes citados anteriormente juntos nesta mesma simulação.

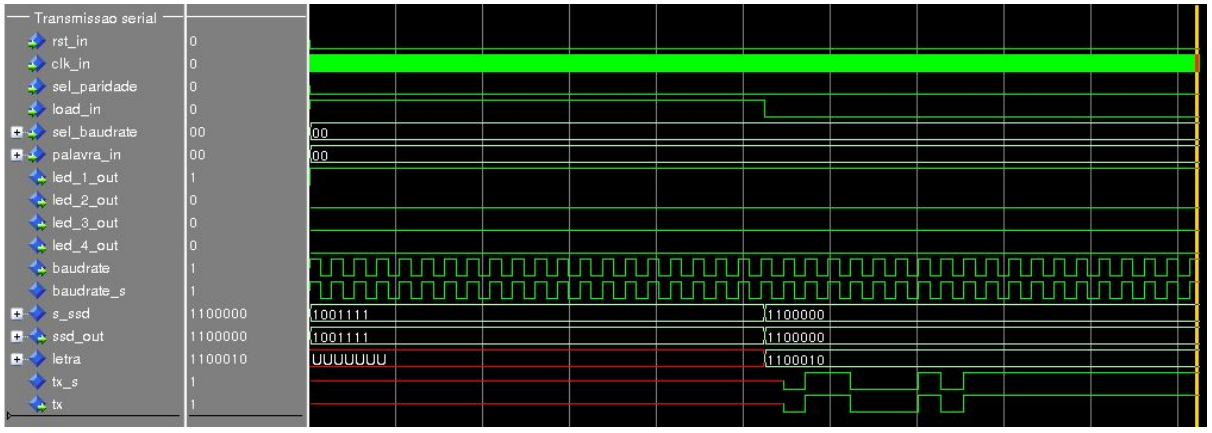

Figura 6 - Simulação no Modelsim do projeto

Pinagem

clk_in → PIN_Y2
load_in → PIN_M23
rst_in → PIN_R24
palavra_in → PIN_Y23 e PIN_Y24
Sel_baudrate → PIN_AA23
Sel_paridade → PIN_AB23
tx → PIN_AB22

Baudrate → PIN_AC15

ssd_out (0 to 6) → HEX0[0], HEX0[1], HEX0[2], HEX0[3], HEX0[4], HEX0[5] e HEX0[6]

leds_out → PIN_E21, PIN_E22, PIN_E23 e PIN_E24

No total, para a execução deste projeto se fez uso de 221 elementos lógicos, 21 pinos, como observado na figura a seguir.

Flow Status	Successful - Tue Jul 9 11:44:11 2019
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	transmissao_serial
Top-level Entity Name	transmissao_serial
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	221 / 114,480 (< 1 %)
··· Total combinational functions	219 / 114,480 (< 1 %)
··· Dedicated logic registers	46 / 114,480 (< 1 %)
Total registers	46
Total pins	21 / 529 (4 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figura 7 - Compilação do projeto no QUARTUS