Projeto Final - Sistema de Transmissão Serial Assíncrono

Daniel Tatsch, Mário Allan L. de Abreu, Schaiana Sonaglio
Dispositivos Lógicos Programáveis I
Engenharia de Telecomunicações
IFSC - Instituto Federal de Santa Catarina, São José, SC
Dezembro de 2017

1 Introdução

Nesse projeto, foi desenvolvido um sistema de transmissão digital de dados, utilizando o *software* Quartus em conjunto com o ModelSim, tendo como base o FPGA Cyclone IV - EP4CE115F29C7.

O projeto foi dividido em quatro partes, tendo como foco inicial o desenvolvimento dos componentes que formam a parte de transmissão e recepção de dados. Nessa etapa, cada um dos componentes foi simulado separadamente de modo a diminuir a complexidade do sistema como um todo. Após a validação dessas duas partes separadamente, as mesmas foram simuladas em conjunto, formando assim todo o sistema de transmissão e recepção de dados.

Por último, foi desenvolvido outro componente, que foi ajustado nas partes transmissora e receptora, responsável por apresentar ao usuário os bits de informação que foram utilizados na comunicação, assim como uma saída de erro para o caso destes terem sido transmitidos incorretamente.

A Figura 1 representa o formato do quadro que foi transmitido durante os testes e simulações. O primeiro e o último bit são chamados de *start* bit e *stop* bit (Sb), eles são responsáveis pela indicação de chegada da informação no receptor e para indicação do fim da transmissão da informação, respectivamente. Quando houver informação para ser transmitida ao receptor, o *start* bit ficará em nível lógico baixo, caso contrário, ficará sempre em nível lógico alto, acompanhando o valor do *stop* bit. Os demais bits correspondem aos bits de paridade (P), endereçamento (A2, A1 e A0) e de dados (D7 a D0) e serão melhor detalhados nos tópicos seguintes.

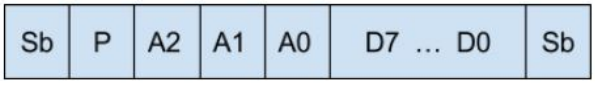

Figura 1 - Formato da mensagem transmitida

2 Descrição da Aplicação Desenvolvida

2.1 Transmissão de dados

O componente responsável pela transmissão de dados foi fragmentado em 3 partes, sendo estas o geradores de bit de paridade, o gerador de *baudrate* e, por último, o conversor de dados paralelo para serial.

A partir do acionamento das chaves mecânicas, é feita a seleção do bit de paridade, que é o responsável por detectar erros nas transmissões, a seleção dos bits de configuração da taxa de transmissão de dados (*baudrate*) e também a definição dos dados a serem enviados; a partir das chaves mecânicas, também há um *push-button* responsável pelo *reset* do sistema, sendo este ativo em nível lógico baixo, ou seja, anulando a transmissão caso assuma valor zero. Após estas configurações, essa sequência de bits é atribuída à entrada do conversor paralelo serial, que acrescentará o *start* bit e *stop* bit e encaminhará para a saída do componente TX o sinal resultante serializado. O sinal completo serializado só será enviado mediante o acionamento de outro *pushbutton*, chamado de *load*. O esquemático do componente TX do sistema pode ser observado na Figura 2.

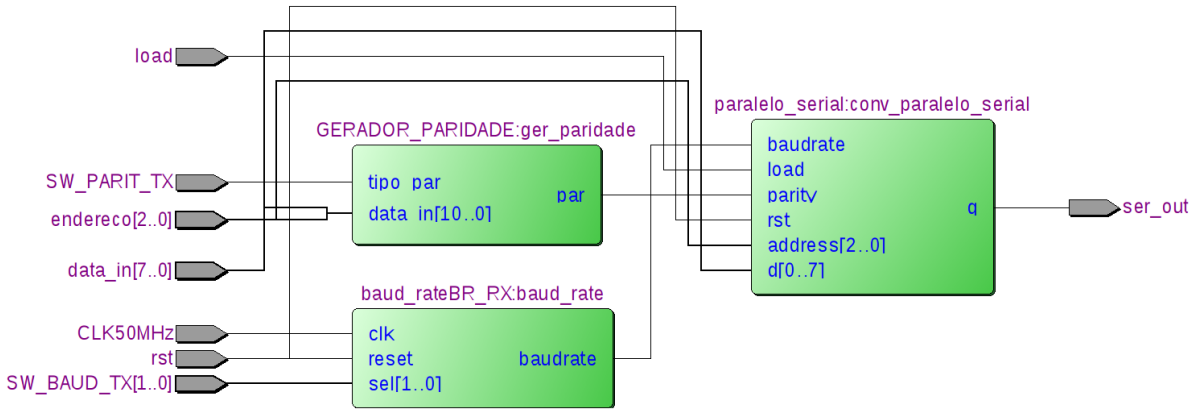

Figura 2 - RTL do componente de transmissão.

Na entrada do gerador de *baudrate*, temos o *clock* do sistema, que atua em 50MHz. A partir da seleção dos bits na entrada SW_BAUD_TX, define-se qual será a taxa de transmissão de dados do sistema de transmissão, podendo atuar em 100kHz, 1kHz, 10Hz ou 0,1Hz. A saída desse módulo é utilizada como base de tempo para a conversão dos bits de paralelo para serial, sendo agrupados na saída a cada borda de subida do *baudrate*.

No gerador de paridade, há a possibilidade de ser selecionado o tipo de paridade que se deseja, através da chave SW_PARIT_TX, sendo ’0’ a paridade par e ’1’ a ímpar. O sinal recebido só será verificado com sucesso se o bit de paridade do componente de recepção for o mesmo do que o de transmissão. Caso ocorra erro na verificação de paridade, a saída RX_ERROR fica em nível lógico alto.

Para simular o componente de transmissão de dados, foi utilizado o programa ModelSim, onde foram selecionadas as paridades par e ímpar, através da inserção dos valores ’0’ e ’1’ (no SW_PARIT_TX), respectivamente, a taxa de transmissão de dados em 100kHz (SW_BAUD_TX em "11") e os bits de endereço "010"e "101". Os bits de dados foram setados para os valores "10011100", "11111111", "00000000"e "10101010"e o *reset* foi desabilitado (ativo em ’0’). É gerado um pulso na entrada *load* em 2,5 até 10μs, com o objetivo de carregar toda a informação na variável de saída *ser_out* que, após a serialização, envia o *stop* bit e permanece em nível lógico alto até a próxima transmissão, que só será realizada quando ocorrer outro pulso no *load*. A figura 3 mostra a simulação realizada e os respectivos valores de entrada e saída.

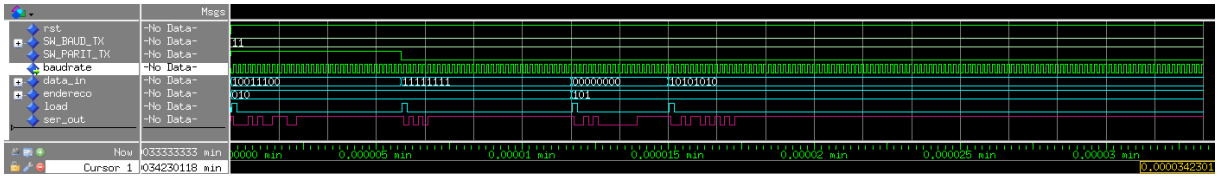

Figura 3 - Simulação do componente TX.

2.2 Recepção de dados

O sistema de recepção de dados é responsável por receber o quadro que foi transmitido, analisar se houve erro durante a transmissão e verificar se essa informação realmente é endereçada ao receptor. Ele possui como entrada o *clock* do sistema e o seletor de *baudrate*, que deverá estar configurado com a mesma taxa do transmissor. Além dessas duas entradas, o receptor possui também o seletor de paridade, *reset* e a entrada serial.

Após o ajuste do *baudrate*, equivalente ao processo realizado na transmissão, e após a conversão de serial para paralelo, que é realizada de acordo com os pulsos recebidos no *ser_in*, o sinal resultante é analisado no detector de paridade, onde é verificada a paridade selecionada com os pulsos recebidos para que, caso ocorra um erro na transmissão, a saída RX_ERROR seja ativada. Os bits de dados são recuperados e mostrados na saída DATA_OUT. A Figura 4 apresenta o esquemático do componente de recepção.

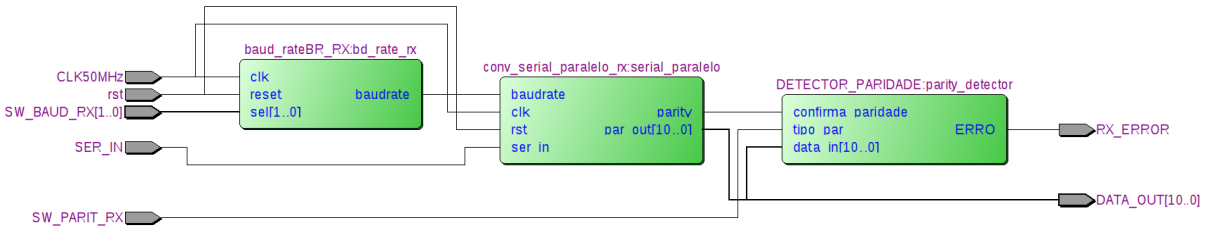

Figura 4 - RTL do componente de recepção.

Assim como no componente TX, foi utilizado o ModelSim para simulação do componente de recepção de dados, conforme Figura 5. Para isso, foi simulada na entrada *ser_in* algumas sequências, respeitando o formato do quadro presente na Figura 1. O *clock* do circuito foi setado para 50MHz na entrada CLK50MHz e ajustado com os bits "11"no gerador de *baudrate* SW_BAUD_RX para ser equivalente à taxa de bits transmitida, 100kHz. A entrada SW_PARIT_RX foi ajustada para detectar a paridade ímpar e o *reset* ficou desativado para que pudéssemos analisar a as duas saídas.

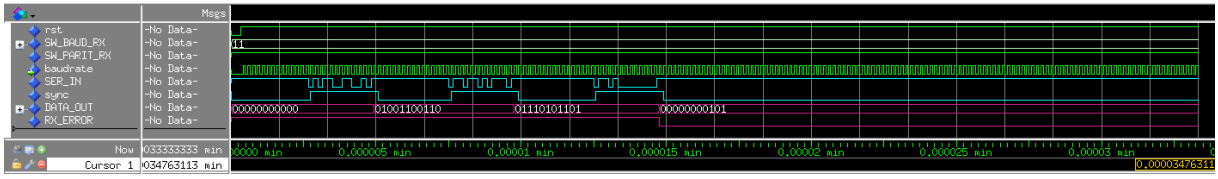

Figura 5 - Simulação do componente RX

2.3 União dos componentes de transmissão e recepção

Após a validação das partes de transmissão e recepção separadamente, ambas foram integradas no mesmo programa, instanciando componentes em um código principal. A entrada do sistema, como observado na Figura 6, consiste nas mesmas entradas do componente TX, na sua saída os bits de dados são recuperados e, em caso de falha, uma porta é acionada, sinalizando o erro na transmissão.

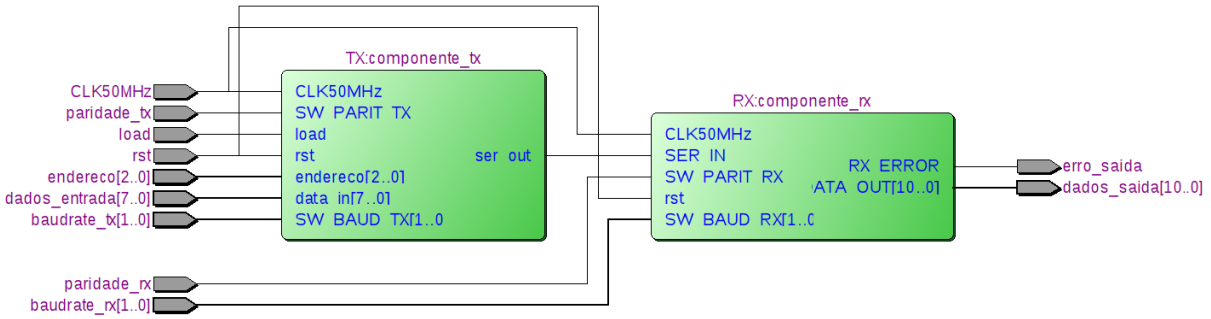

Figura 6 - Sistema de transmissão serial.

Com os componentes unidos, foi realizada mais uma simulação para visualização da atuação geral do sistema de transmissão serial assíncrono. Com a mesma configuração de transmissão apresentada na Figura 3, a saída se comportou como o esperado no momento em que as configurações do módulo RX foram ajustadas corretamente. A Figura 7 apresenta a simulação realizada no ModelSim.

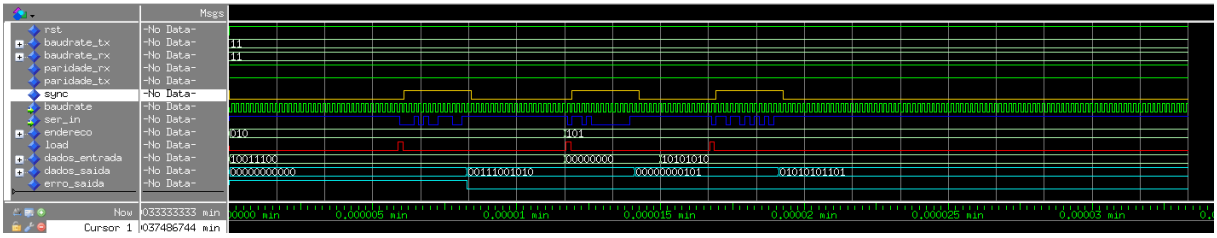

Figura 7 - Simulação do sistema de transmissão serial.

Com a conclusão do sistema de transmissão serial (TOP LEVEL), foram obtidos os valores de frequência máxima, número de elementos lógicos e de pinos, de acordo com a Tabela[1] 1. Ao fazer a compilação do sistema completo, o compilador do Quartus conseguiu fazer associações que aproveitassem os pinos e elementos lógicos já utilizados pelos componentes de transmissão e recepção.

Tabela 1 - Dados obtidos no desenvolvimento do projeto

	Frequência máxima	Nº de elementos lógicos	Nº de pinos
Componente TX	296,74MHz	83	18
Componente RX	287,69MHz	122	18
Sistema completo	297,18MHz	197	32

[1] Valores obtidos para o modelo Cyclone IV - EP4CE115F29C7

3 Conclusões

Neste projeto, foi desenvolvido um sistema de transmissão serial assíncrono, dividido, basicamente, entre as partes de transmissão e recepção de dados. Este sistema possui estrutura hierárquica e, por isso, facilitou o entendimento e desenvolvimento de cada parte dos componentes TX e RX. Após cada etapa concluída, um sistema de nível hierárquico maior era gerado e testado, validando a etapa em questão com suas devidas simulações.

Para fins de experimento e validação dos dados, foram feitas apenas as simulações no *software* ModelSim, não realizando o procedimento de criação do componente que ficaria responsável por dispor nos *displays* e *leds* as informações trocadas.

As maiores dificuldades encontradas durante o desenvolvimento ocorreram nas conversões de paralelo serial e serial paralelo. Preocupou-se inicialmente com a necessidade de sincronia dos bits que eram serializados e separados para transmissão. Para contornar esse problema utilizou-se uma porta conectada a uma chave mecânica do tipo *pushbutton* para gerar um pulso e carregar as informações. Na recepção, houveram dificuldades durante a simulação. Após alguns ajustes no código e um maior estudo do mesmo as simulações foram efetivas, validando assim, o componente RX do sistema.

Este projeto, além de exercitar tanto a programação sequencial quanto a concorrente em VHDL, fez amadurecer os conceitos abordados durante a disciplina, como a verificação do número de elementos lógicos em um circuito e o que pode ser feito para minimizá-lo e análise do *hardware* gerado através de esquemáticos RTL.

Outro fator positivo na realização do projeto foi a utilização de conceitos presentes em outras disciplinas, através da montagem do quadro de informação que é gerado para a transmissão.