Universidade Federal de Minas Gerais
Escola de Engenharia – Departamento de Engenharia Eletrônica
Laboratório de Arquitetura e Organização de Computadores
Assunto: Projeto de um módulo UART
Autoria: Prof. Ricardo de Oliveira Duarte

Objetivos:

Explicar o funcionamento de um módulo UART.
Apresentar os passos necessários ao projeto de um módulo UART.

Seção 1: Funcionamento de um módulo UART:

UART significa Receptor e Transmissor Assíncrono Universal de dados seriais. A função do módulo UART é a de converter dados paralelos (8 bits) para dados seriais e vice-versa. O módulo UART transmite *bytes* de dados sequencialmente um *bit* de cada vez via um pino de GPIO programado com saída (pino TX) e recebe o *byte* de dados serializado *bit* a *bit*, via outro pino de GPIO programado como entrada (pino RX). Como os processos de envio e recebimento dos dados não requerem nenhuma entrada de *clock* para sincronizar a transmissão serial, ele é caracterizado como um módulo de comunicação serial assíncrona. Na comunicação de dados via UART, a velocidade de transmissão de *bits* por segundo (*bps*) é caracterizada pela taxa de transmissão (*baud rate*). A taxa de transmissão descreve o número total de *bits* enviados pela comunicação serial por segundo. Inclui *bit* de inicialização, *byte* de dados, *bit* de paridade (opcional) e *bit* de parada. Tanto o módulo transmissor, quanto o receptor precisam ser mantidos configurados com a mesma taxa de transmissão para o correto funcionamento. A forma de onda da figura 1 mostra o protocolo de comunicação do módulo UART. Quando não há nenhum dado a ser transmitido, a saída do módulo fica em nível lógico 1. No protocolo de comunicação serial o primeiro *bit* 0 é chamado de *bit* de início (*start bit*) indica que a transmissão de um *byte* será iniciada via comunicação serial. Os próximos 8 bits são bits de dados. O *bit* de dado menos significativo do *byte* é o primeiro *bit* a ser enviado serialmente e o módulo prossegue a transmissão continua enviando outros 7 *bits*. O décimo *bit* a ser enviado é um *bit* de paridade que é usado para detectar erros na comunicação. O *bit* de paridade vai ser 0 ou 1, dependendo do número de 1´s presentes na transmissão. Se paridade par for configurada, o número de *bits* deve ser par. Se a paridade ímpar for configurada, o número de *bits* deve ser ímpar. O módulo pode ser também configurado para não enviar nenhum *bit* de paridade no protocolo. O último *bit* do protocolo é o *bit* de parada (*stop bit*) que deve ser 1. Como a taxa de transmissão é fixa e configurada pelo programador, cada *bit* do protocolo terá uma duração de tempo fixa durante a transmissão ou recepção. O atraso de transmissão/recepção para cada *bit* será de 104,16 µs para uma taxa de transmissão de 9600 *bps*. O término da transmissão poderá ou não estar associado a uma sinalização de interrupção.

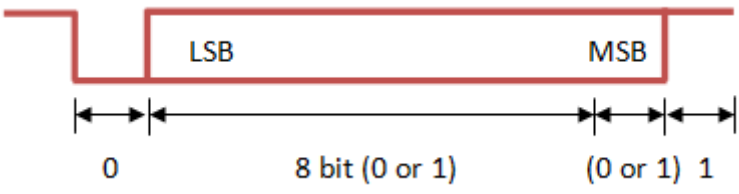

Figura 1 – Protocolo de uma transmissão serial via UART

Observação: consulte os códigos de UART_TX, UART_RX e UART_TB disponibilizados por mim. Adapte-os às suas necessidades e use-os no seu projeto do microcontrolador (*MCU*).

Universidade Federal de Minas Gerais
Escola de Engenharia – Departamento de Engenharia Eletrônica
Laboratório de Arquitetura e Organização de Computadores
Assunto: Projeto de um módulo UART
Autoria: Prof. Ricardo de Oliveira Duarte

Seção 2: Projeto de um módulo UART:

Passo 1: Defina os componentes que esse módulo precisará.

Passo 2: Faça um desenho completo interligando todos os componentes do seu módulo.

Passo 3: Projete em VHDL o módulo desenhado no passo 2.

Passo 4: Compile o seu módulo. Corrija eventuais erros de compilação.

Passo 6: Crie um *testbench* para testar o seu módulo. Simule o seu sistema com o *testbench*.

Passo 7: Escreva um programa em linguagem C que configure o seu módulo para alguma aplicação.

Passo 8: Converta o programa que você escreveu no passo anterior para a linguagem *assembly*.

Passo 9: Converta o programa que você escreveu no passo anterior para a linguagem de máquina.

Passo 10: Compile a sua CPU com o novo módulo e as modificações realizadas na controladora de interrupções. Corrija eventuais erros de compilação.

Passo 11: Teste o seu sistema no kit DE10-Lite.

Passo 12: Observe no osciloscópio a sequência de dados transmitida e meça com o osciloscópio a duração de transmissão de um bit.