Tutorial RS-232 da placa DE2-115 da Altera

Gustavo de Faria Silva - gustavofaria@ufu.br
João Paulo de Oliveira - joaopaulodeoliveira123@gmail.com
Lucas Rossi Rabelo - lucasrossi98@hotmail.com

9 de janeiro de 2018

Sumário

1 Introdução **3**

2 Caracteŕisticas físicas e elétricas **3**

3 Interfaceamento **4**

1 Introdução

O trabalho tem como objetivo mostrar como é feita a comunicação entre a DE2-115 e um computador de propósito geral. O material mostra o adaptador utilizado, bem como uma demonstração do hardware responsável pela troca de dados, algumas características da placa, além de mostrar o passo a passo da configuração no computador e no final é feita uma demonstração através de um exemplo do envio de caracteres em ASCII da DE2-115 para o computador através do PUTTY.

2 Caracteŕisticas físicas e elétricas

O RS-232 na Placa DE2-155 da Altera oferece um conector DB-9 fêmea para fazer o interfaceamento com o computador ou outro dispositivo de comunicação. Um computador conectado com a placa deve ter as seguintes configurações[2]

- **Taxa de transferência: 115200**
- **Bit de paridade: Nenhum**
- **Bits de dados: 8**
- **Bit de parada: 1**
- **Controle de fluxo: ON**

A DE2-115 usa o transceptor ZT3232 para gerenciar a conexão entre a placa. Para mais detalhes sobre o transceptor, consulte o datasheet.

A pinagem é conectada ao FPGA da seguinte forma:

Tabela 1: Tabela de associação de pinos

Pino	**No Pino FPGA**	**Descrição**	**Voltagem**
RXD	PIN_G12	Receptor da UART	3.3V
TXD	PIN_G9	Transmissor da UART	3.3V
CTS	PIN_G14	Limpar para enviar na UART	3.3V
RTS	PIN_J13	Requisição para enviar na UART	3.3V

A conexão será feita através do adaptador RS-232 <-> USB mostrado na figura 3:

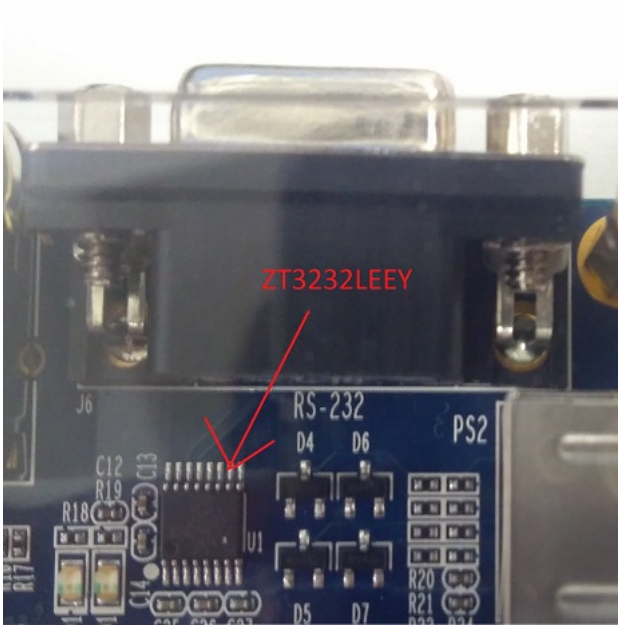

Figura 1: CI utilizado para o RS-232

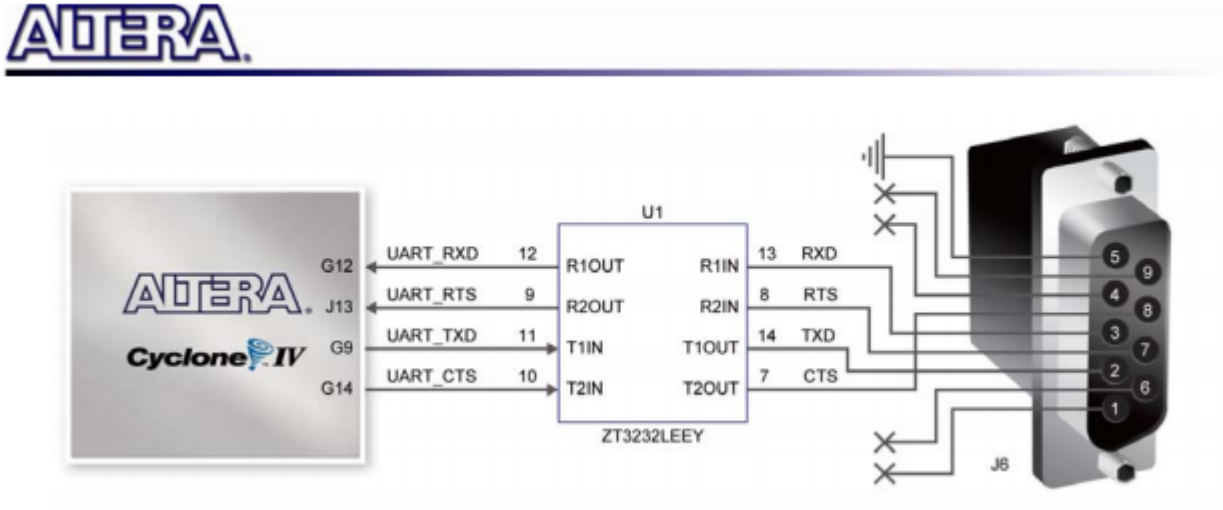

Figura 2: Pinagem utilizada entre o FPGA e o ZT3232LEEY

3 Interfaceamento

O PUTTY é um emulador de terminal e emulador serial que suporta vários protocolos de internet incluindo SPC, SSH, TELNET, rlogin e conexão de soquete bruto. Além disso, faz conexão com a porta serial. Ele é suportado

Figura 3: Adaptador utilizado para a comunicação

por Windows e Linux.

Para começar, deve-se configurar a porta serial da máquina para que o PUTTY consiga fazer a comunicação, deve-se descobrir a porta COM em que o conversor USB/RS-232 foi conectado. No Windows isso pode ser facilmente verificado no gerenciador de dispositivos, nos controladores USB como destacado na imagem.

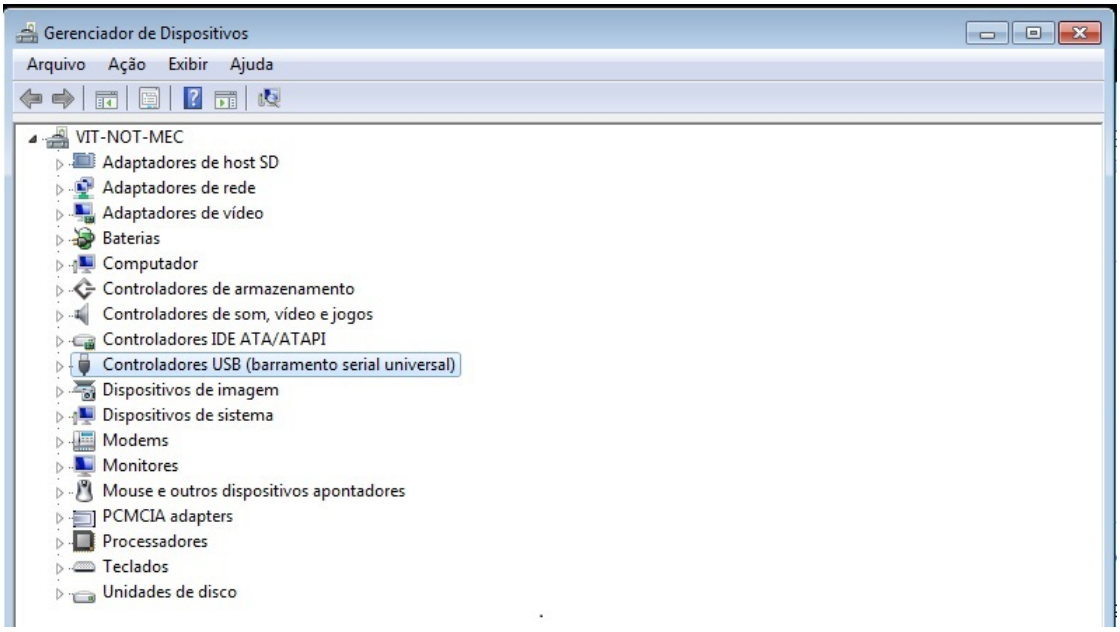

Figura 4: Gerenciador de dispositivos

Suponha que o nosso Adaptador está na porta COM4, assim devemos configurar o PUTTY para serial como mostra na Figura 5. Essa tela de configuração é exibida na inicialização do PUTTY.

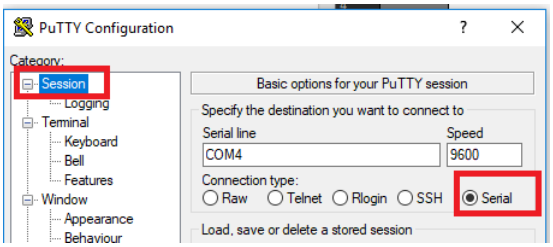

Figura 5: Configuração na aba Session no PUTTY

A única configuração necessária nessa aba é selecionar a comunicação serial como destacado na imagem acima.

Na aba serial, devemos utilizar as configurações mostradas na seção 1. Na figura 6, foi configurado exatamente para a DE2-115 e a porta COM4. Note que a taxa de transferência foi configurada para 9600 considerando o exemplo presente na seção 3.1, nele, o clock é dividido para resultar em uma taxa de transferência de 9600.

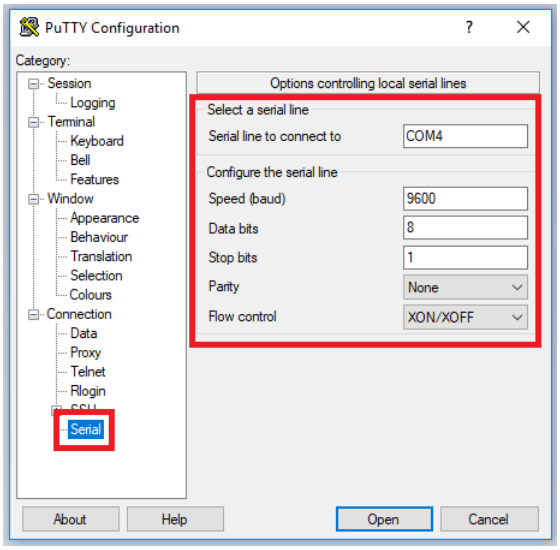

Figura 6: Configuração na aba Session no PUTTY

Feito isso, podemos iniciar abrir a comunicação com a DE2-115 clicando no botão Open, daí será aberto o terminal de comunicação do PUTTY.

Resta agora programar a placa para fazer a comunicação. Veja o exemplo:

Exemplo

O código a seguir está na linguagem System Verilog e pode ser gravada na placa através do Quartus II, no código está descrito uma máquina de estado finito que efetua o envio de 1 byte de dados, que é recebido pelo PUTTY na forma de carácter na tabela ASCII.

O exemplo está mapeado para a placa DE2-115 com os assignments disponibilizados pela Altera. As entradas e saídas utilizadas são [1]

- **SW:** As chaves de 0 até 7 serão mapeadas para o byte a ser enviado.
- **KEY:** São os botões 0 e 1 na placa:
  - **KEY0:** Corresponde ao Clock da placa. Quando pressionado, a placa envia o carácter representado nas chaves, porém, não é feito o debounce, então quando pressionado, pode ocorrer o envio de vários caracteres com um único clique.
  - **KEY1:** Corresponde ao reset da comunicação.
- **UART TXD:** É a saída serial do conector DB-9 da placa.

```
/* Modulo de transmissao UART na placa DE2-115 da Altera*/
module rs232(input CLOCK_50,
    input  [1:0] KEY,
    input  [7:0] SW,
    output reg UART_TXD);

  reg[4:0] bitCounter;       // conta o numero de bits que foram
   sendo enviados

  reg[31:0] counter;         // conta o numero de pulsos de clock
   , usado para dividir o clock interno(115200)

  reg state, nextState;      // estado atual e proximo estado

  reg[9:0] rightShiftReg;     // regtistrador usado para
   armazenar o valor que esta sendo atualmente enviado

  reg shift, load, clear;     // determina as operacoes que
   dever ser terminadas no estado atual
  always @ (posedge CLOCK_50)
  begin
    if (KEY[1]) begin
      state <= 0;
      counter <= 0;
      bitCounter <= 0;
    end
```

```
   else begin
     counter <= counter+1;
     if (counter >= 5208) begin            // divide o colock
  para uma taxa de transferencia de 9600
       state <= nextState;              // se o valor foi
  alcancado, vai para o proximo estado
       counter <= 0;                 // da reset no contador e
  efetua a operacao do estado atual
       if (load)                     // preparam o bit nas chaves
   incluindo o start bit (0), e o stop bit (1)
         rightShiftReg <= {1'b1, SW[7:0], 1'b0};
       if (clear)
         bitCounter <= 0;
       if (shift) begin
         rightShiftReg <= rightShiftReg>>1;
         bitCounter <= bitCounter+1;
       end
     end
   end
end

always @(state or bitCounter or KEY[0]) // maquina de estados
  do transmissor
begin
    load <= 0;
    shift <= 0;
    clear <= 0;
    UART_TXD <= 1;
    case (state)
     0: begin             // Estado inicial, se KEY0 esta
  pressionada, inicializa a transmissao das chaves (SW)
       if (KEY[0] == 1) begin
         nextState <= 1;
         load <= 1;
         shift <= 0;
         clear <= 0;
       end
       else begin
         nextState <= 0;
         UART_TXD <= 1;
       end
     end

     1: begin             // faz as operacoes para o estado e
  mantem nesse estado ate todos os bits serem enviados
         if (bitCounter >= 9) begin
           nextState <= 0;
           clear <= 1;
         end
```

```
            else begin
              nextState <= 1;
              shift <= 1;
              UART_TXD <= rightShiftReg[0];
          end
        end
      endcase
    end
endmodule
```

Referências

[1] R. Jacob Baker. *CMOS: Mixed-signal Circuit Design*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[2] Terasic Technologies. *DE2-115 User Manual*. Terasic Technologies, 2010.