Rx & Tx

****

|  |  |  |  |
| --- | --- | --- | --- |
| **Digitado por:** | **Professor:** | **Versão** | **Ano** |
| Ricardo Kim 201100597 | Renato Giacommini | 1.1 | 2017 |

**ÍNDICE**

[1. Objetivos 3](#_Toc481796285)

[2. Funcionamento 4](#_Toc481796286)

[2.1. Transmissão de dados 4](#_Toc481796287)

[2.2. Recepção de dados 4](#_Toc481796288)

[3. Código FPGA comentado 5](#_Toc481796289)

# Objetivos

Conhecer os conceitos básicos da transmissão e recepção de dados através do barramento do RS232, Transmiter e Receiver. E o aprendizado no manejo da programação em VHDL na FPGA

# Funcionamento

## Transmissão de dados

A transmissão consiste no envio para o barramento do conector RS232 com o pulso de clock em na borda de descida transmitindo os níveis lógicos um a um, são transmitidos por transmissão, cada byte obedece o padrão de transmissão onde são enviados adicionais sendo 1 no começo da transmissão indicando ser o bit de início de transmissão sinalizado em nível lógico 0, e 2 bits no final do byte em nível lógico 1 indicando final da transmissão do byte, os bits a serem transmitidos são acionados pelos switches localizados na placa.

## Recepção de dados

A recepção consiste em detectar o primeiro bit da transmissão, o bit de inicio e os dois últimos bits indicando o fim da transmissão, para isso haverá um clock de leitura que dispara em aguardando o primeiro bit em nível lógico 0, após a leitura completa dos dados, os LEDs de cor vermelha acenderão correspondente a informação transmitida e se manterá acesa com a ultima informação transmitida.

# Código FPGA comentado

--Tx trabalhar o transmissor no Tx em 1 barramento

--Rx trabalhar o Rx com clock superior do Tx, normalmente com 4x~10x mais

--dessa forma o receptor pode detectar a borda de 0

--indicando o start bit

**ENTITY** Projeto4 **is**

**PORT**

**(**

--ELEMENTOS COMUNS A TODOS

clk**:** **in** BIT**;**

--ELEMENTOS DO Tx

sw0**,** sw1**,** sw2**,** sw3**,** sw4**,** sw5**,** sw6**,** sw7**,** sw8**,** sw9 **:** **in** BIT**;**

send **:** **IN** BIT**;**

Tx**,**ledTransmission **:** **out** BIT**;**

--ELEMENTOS DO Rx

led0**,** led1**,** led2**,**led3**,** led4**,** led5**,** led6**,** led7**,** led8**,** led9**:** **OUT** BIT**;**

receive **:** **OUT** BIT**;**

Rx**:** **IN** BIT**;**

-- elementos de debug

txview**,**rxview **:** **OUT** BIT

**);**

**END** Projeto4**;**

**ARCHITECTURE** Kim **OF** Projeto4 **is**

--PROCESSO 1

**CONSTANT** Cons\_Count **:** INTEGER **:=** 50000**;**

**SIGNAL** nClk**:** INTEGER **RANGE** 0 **to** Cons\_Count**;**

**SIGNAL** sClk**:** BIT**;**

--PROCESSO 2

**SIGNAL** transmit\_flag**:** INTEGER **RANGE** 0 **TO** 3 **:=** 0**;** --controla STEP de processo

**SIGNAL** byte**:** INTEGER **RANGE** 0 **to** 60**;**

--PROCESSO 3

**CONSTANT** rx\_speed**:** INTEGER **:=** **(**10**);**

**CONSTANT** rx\_count **:** INTEGER **:=** **(**Cons\_Count**/**rx\_speed**);** --clock de leitura 10x mais rapida do tx

**SIGNAL** rxNClock **:** INTEGER **RANGE** 0 **to** rx\_count**;**

**SIGNAL** rxSClock **:** BIT**;**

--PROCESSO 4

**SIGNAL** init\_flag**:** INTEGER **RANGE** 0 **to** 3 **:=** 0**;** --controla STEP de processo

**SIGNAL** rxByte**:** INTEGER **RANGE** 0 **to** 2200 **:=** 0**;**

-- registrador dos bytes de entrada;

**SIGNAL** byte\_receiver**:** BIT\_VECTOR **(**0 **TO** 22**);**

**BEGIN**

rxview **<=** rxSClock**;**

txview **<=** sCLk**;**

-- ====================

-- == PROCESSO I ==

-- ====================

-- Reducao do clock de 50MHz para 500Hz

**PROCESS(**clk**)**

**BEGIN**

**IF(**clk'**EVENT** and clk**=**'0'**)** **THEN**

**IF** **(**nClk **>** Cons\_Count**/**2**)** **THEN**

nClk **<=** 0**;**

sClk **<=** not sClk**;**

**ELSE**

nClk **<=** nClk **+** 1**;**

**END** **IF;**

**END** **IF;**

**END** **PROCESS;**

-- ====================

-- == PROCESSO II ==

-- ====================

-- Ler os dados do Switch

**PROCESS** **(**sClk**)**

**BEGIN**

-- Em eventos de clock de borda de decida

**IF** **(**sClk'**EVENT** and sClk **=**'0'**)** **THEN**

-- Se transmissao for ativado (Push Bottom)

**IF(**send **=** '0' AND transmit\_flag **=** 0**)** **THEN**

transmit\_flag **<=** 1**;**

**ELSIF(**transmit\_flag **=** 1**)** **THEN**

-- se transmissao

**IF** byte **=** 22 **THEN**

byte **<=** 0**;**

transmit\_flag **<=** 2**;**

**ELSE**

byte **<=** byte **+** 1**;**

**END** **IF;**

**CASE** byte **is**

-- =============================

-- = Byte de envio numero 1 =

-- =============================

**WHEN** 0 **=>**

Tx **<=** '0'**;**

**WHEN** 1 **=>**

Tx **<=** sw7**;**

**WHEN** 2 **=>**

Tx **<=** sw6**;**

**WHEN** 3 **=>**

Tx **<=** sw5**;**

**WHEN** 4 **=>**

Tx **<=** sw4**;**

**WHEN** 5 **=>**

Tx **<=** sw3**;**

**WHEN** 6 **=>**

Tx **<=** sw2**;**

**WHEN** 7 **=>**

Tx **<=** sw1**;**

**WHEN** 8 **=>**

Tx **<=** sw0**;**

**WHEN** 9 **=>**

Tx **<=** '1'**;**

**WHEN** 10 **=>**

Tx **<=** '1'**;**

-- =============================

-- = Byte de envio numero 2 =

-- =============================

**WHEN** 11 **=>**

--start

Tx **<=** '0'**;**

**WHEN** 12 **=>**

--7

Tx **<=** '0'**;**

**WHEN** 13 **=>**

--6

Tx **<=** '0'**;**

**WHEN** 14 **=>**

--5

Tx **<=** '0'**;**

**WHEN** 15 **=>**

--4

Tx **<=** '0'**;**

**WHEN** 16 **=>**

--3

Tx **<=** '0'**;**

**WHEN** 17 **=>**

--2

Tx **<=** '0'**;**

**WHEN** 18 **=>**

--1

Tx **<=** sw9**;**

**WHEN** 19 **=>**

--0

Tx **<=** sw8**;**

**WHEN** 20 **=>**

Tx **<=** '1'**;**

**WHEN** 21 **=>**

Tx **<=** '1'**;**

**WHEN** **others** **=>**

**end** **case;**

ledTransmission **<=** '1'**;**

**ELSIF** **(**send **=** '1' AND transmit\_flag **=** 2**)** **THEN**

transmit\_flag **<=** 0**;**

ledTransmission **<=** '0'**;**

**ELSE**

**END** **IF;**

**END** **IF;**

**END** **PROCESS;**

-- ====================

-- == PROCESSO III ==

-- ====================

-- Clock 5x mais rapido que o Tx

**PROCESS(**clk**)**

**BEGIN**

**IF(**clk'**EVENT** and clk**=**'0'**)** **THEN**

**IF** **(**rxNClock **>** rx\_count**/**2**)** **THEN**

rxNClock **<=** 0**;**

rxSClock **<=** not rxSClock**;**

**ELSE**

rxNClock **<=** rxNClock **+** 1**;**

**END** **IF;**

**END** **IF;**

**END** **PROCESS;**

-- ====================

-- == PROCESSO IV ==

-- ====================

-- Rx

**PROCESS(**rxSClock**)**

**BEGIN**

--Na do clock na borda de descida

**IF** **(**rxSClock'**EVENT** and rxSClock **=**'0' **)** **THEN**

--Detectar a primeira entrada do Rx

**IF** **(**Rx **=** '0' AND init\_flag **=** 0**)** **THEN**

-- Mantem preservado a informacao da ultima

init\_flag **<=** 1**;**

rxByte **<=** 0**;**

receive **<=** '0'**;**

-- detectar a entrada do startBit

**ELSIF(**init\_flag **=** 1**)** **THEN**

-- retorna para o estado de espera do startBit

-- se (chegar o ultimo bit)

**IF** **(**rxByte **=** 215**)** **THEN**

--resetar o receptor de bytes

byte\_receiver**(**0**)<=** '0'**;**

byte\_receiver**(**1**)<=** '0'**;**

byte\_receiver**(**2**)<=** '0'**;**

byte\_receiver**(**3**)<=** '0'**;**

byte\_receiver**(**4**)<=** '0'**;**

byte\_receiver**(**5**)<=** '0'**;**

byte\_receiver**(**6**)<=** '0'**;**

byte\_receiver**(**7**)<=** '0'**;**

byte\_receiver**(**8**)<=** '0'**;**

byte\_receiver**(**9**)<=** '0'**;**

byte\_receiver**(**10**)<=** '0'**;**

byte\_receiver**(**11**)<=** '0'**;**

byte\_receiver**(**12**)<=** '0'**;**

byte\_receiver**(**13**)<=** '0'**;**

byte\_receiver**(**14**)<=** '0'**;**

byte\_receiver**(**15**)<=** '0'**;**

byte\_receiver**(**16**)<=** '0'**;**

byte\_receiver**(**17**)<=** '0'**;**

byte\_receiver**(**18**)<=** '0'**;**

byte\_receiver**(**19**)<=** '0'**;**

byte\_receiver**(**20**)<=** '0'**;**

byte\_receiver**(**21**)<=** '0'**;**

byte\_receiver**(**22**)<=** '0'**;**

rxByte **<=** 0**;**

init\_flag **<=** 0**;**

-- continua leitura dos bits

**ELSE**

receive **<=** '1'**;**

rxByte **<=** rxByte **+** 1**;**

**END** **IF;**

**CASE** rxByte **is**

-- =============================

-- = Byte de entrada numero 1 =

-- =============================

**WHEN** 4 **=>**

-- [y]xxxxxxxx[xx]

byte\_receiver**(**10**)** **<=** Rx**;**

**WHEN** 14 **=>**

-- [x]yxxxxxxxx[xx]

byte\_receiver**(**9**)** **<=** Rx**;**

**WHEN** 24 **=>**

-- [x]xyxxxxxx[xx]

byte\_receiver**(**8**)** **<=** Rx**;**

**WHEN** 34 **=>**

-- [x]xxyxxxxx[xx]

byte\_receiver**(**7**)** **<=** Rx**;**

**WHEN** 44 **=>**

-- [x]xxxyxxxx[xx]

byte\_receiver**(**6**)** **<=** Rx**;**

**WHEN** 54 **=>**

-- [x]xxxxyxxx[xx]

byte\_receiver**(**5**)** **<=** Rx**;**

**WHEN** 64 **=>**

-- [x]xxxxxyxx[xx]

byte\_receiver**(**4**)** **<=** Rx**;**

**WHEN** 74 **=>**

-- [x]xxxxxxyx[xx]

byte\_receiver**(**3**)** **<=** Rx**;**

**WHEN** 84 **=>**

-- [x]xxxxxxxy[xx]

byte\_receiver**(**2**)** **<=** Rx**;**

**WHEN** 94 **=>**

-- [x]xxxxxxxx[yx]

byte\_receiver**(**1**)** **<=** Rx**;**

--end bit

**WHEN** 104 **=>**

-- [x]xxxxxxxx[xy]

byte\_receiver**(**0**)** **<=** Rx**;**

--end bit

-- byte\_receiver(10) -- start[0]

led7 **<=** byte\_receiver**(**9**);**

led6 **<=** byte\_receiver**(**8**);**

led5 **<=** byte\_receiver**(**7**);**

led4 **<=** byte\_receiver**(**6**);**

led3 **<=** byte\_receiver**(**5**);**

led2 **<=** byte\_receiver**(**4**);**

led1 **<=** byte\_receiver**(**3**);**

led0 **<=** byte\_receiver**(**2**);**

-- byte\_receiver(1) -- end bit;[11]

-- byte\_receiver(0) -- end bit;[11]

---------------------------------------

-- =============================

-- = Byte de entrada numero 2 =

-- =============================

**WHEN** 114 **=>**

-- [y]xxxxxxxx[xx]

byte\_receiver**(**21**)** **<=** Rx**;**

**WHEN** 124 **=>**

-- [x]yxxxxxxx[xx]

byte\_receiver**(**20**)** **<=** Rx**;**

**WHEN** 134 **=>**

-- [x]xyxxxxxx[xx]

byte\_receiver**(**19**)** **<=** Rx**;**

**WHEN** 144 **=>**

-- [x]xxyxxxxx[xx]

byte\_receiver**(**18**)** **<=** Rx**;**

**WHEN** 154 **=>**

-- [x]xxxyxxxx[xx]

byte\_receiver**(**17**)** **<=** Rx**;**

**WHEN** 164 **=>**

-- [x]xxxxyxxx[xx]

byte\_receiver**(**16**)** **<=** Rx**;**

**WHEN** 174 **=>**

-- [x]xxxxxyxx[xx]

byte\_receiver**(**15**)** **<=** Rx**;**

**WHEN** 184 **=>**

-- [x]xxxxxxyx[xx]

byte\_receiver**(**14**)** **<=** Rx**;**

**WHEN** 194 **=>**

-- [x]xxxxxxxy[xx]

byte\_receiver**(**13**)** **<=** Rx**;**

**WHEN** 204 **=>**

-- [x]xxxxxxxx[yx]

byte\_receiver**(**12**)** **<=** Rx**;**

**WHEN** 214 **=>**

-- [x]xxxxxxxx[xy]

byte\_receiver**(**11**)** **<=** Rx**;**

--byte\_receiver(21) start bit;

--led15 <= byte\_receiver(20);

--led14 <= byte\_receiver(19);

--led13 <= byte\_receiver(18);

--led12 <= byte\_receiver(17);

--led11 <= byte\_receiver(16);

--led10 <= byte\_receiver(15);

led9 **<=** byte\_receiver**(**14**);**

led8 **<=** byte\_receiver**(**13**);**

--byte\_receiver(12) end bit;

--byte\_receiver(11) end bit;

**WHEN** **others** **=>**

**END** **case;**

**END** **IF;**

**END** **IF;**

**END** **PROCESS;**

**END** Kim**;**