Sistema de Comunicação Série Síncrono

Universidade de Aveiro

Engenharia de Computadores e Telemática

Laboratórios de Sistemas Digitais

Carlos Costa 88755

Diogo Correia 90327

Docente: Armando Nolasco Pinto

15/05/2018

Índice:

1. Especificações do Sistema

Um sistema de comunicação série síncrono, tem como principal objetivo a transmissão de dados de um nodo *Master* para um nodo *Slave*, utilizando dois sinais, um *data* referente à informação que vai ser transmitida, e um *clock* que determina o ritmo de envio da informação.

Este sistema é unidirecional (*simplex*), ou seja, a informação é enviada apenas pelo *Master* para o *Slave*, não existindo reciprocidade. Esta condição é especificada pelo guião referente ao Projeto Final a ser realizado.

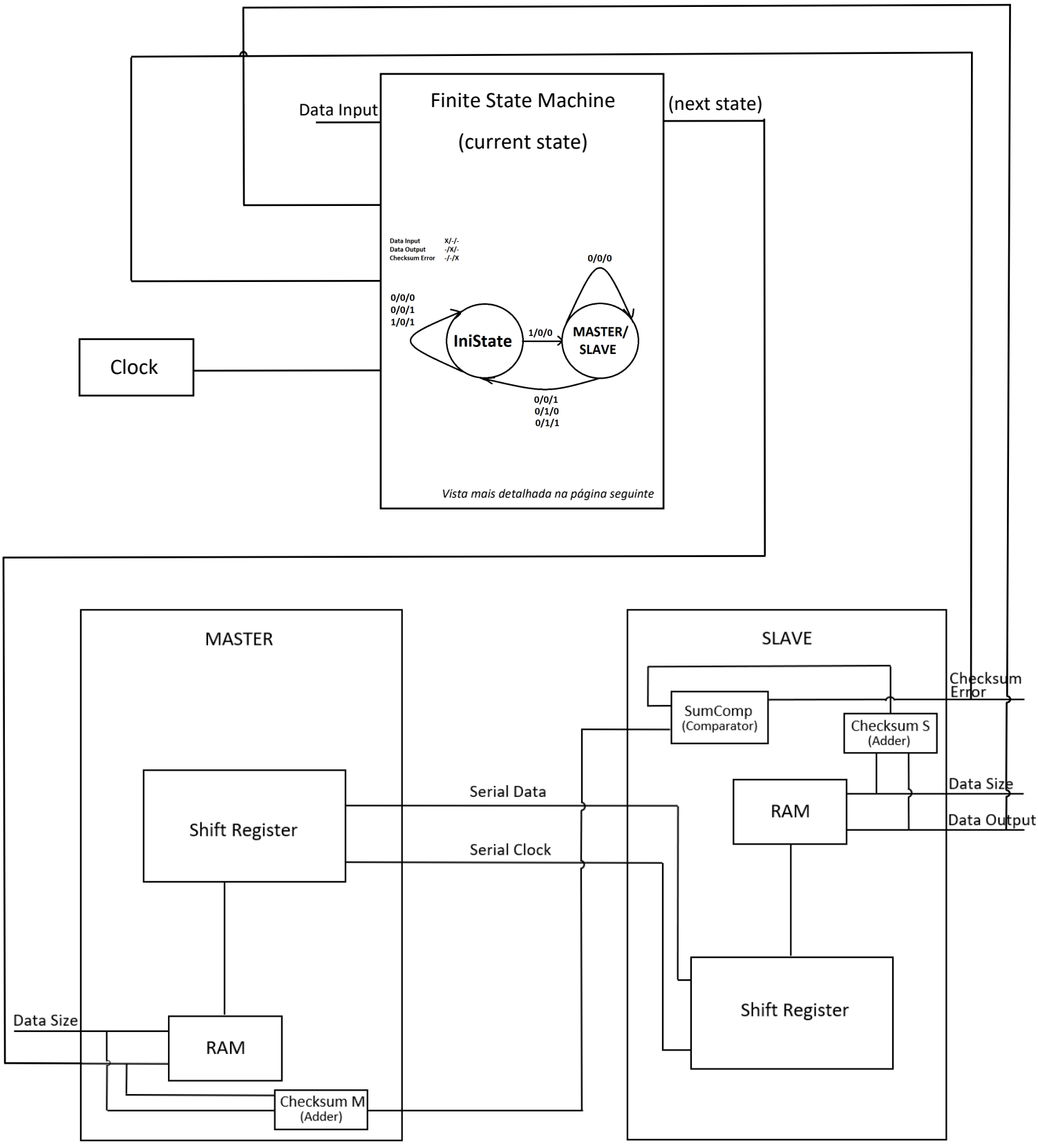
A estrutura de funcionamento do sistema pode ser dividida em quatro partes:

- **Start of Frame:** um bit com valor 0, que representa o início da transmissão dos dados;
- **Size:** contém a informação do número de bytes a utilizar pelo **Payload;**
- **Payload:** alberga os dados que estão a ser enviados do *Master* para o *Slave*;
- **Checksum:** deteta possíveis erros, através da soma de todos os *nibbles* (1 *nibble* = 4 *bytes*) enviados nos blocos **Size** e **Payload**.

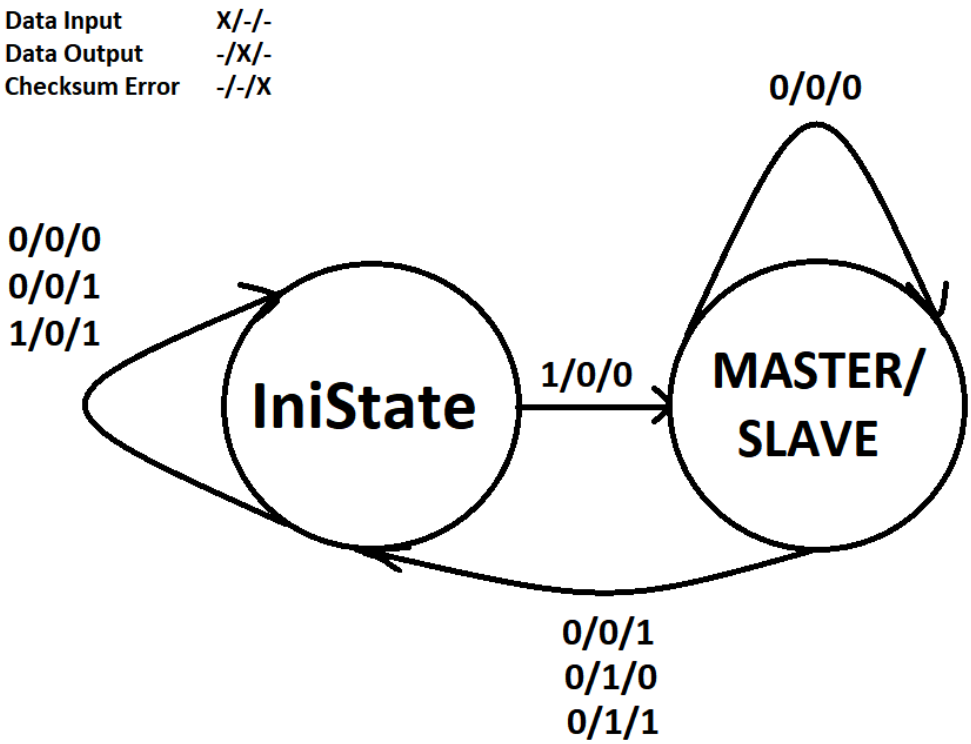
Blocos a utilizar:

- Shift Register;
- Clock;
- Adder;
- Comparator;
- State Machine;
- RAM;

2. Arquitetura do Sistema



2.1. Máquina de Estados



2.2. Tabela de Conclusões

Data Input	Data Output	Checksum Error	Initial State	Master/ Slave
0	0	0	1	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	X	X
1	1	1	X	X

3. Desenvolvimento e validação

Este sistema tem por base o funcionamento de uma máquina de estados, e por isso, a elaboração do código VHDL de uma Finite State Machine é o ponto de partida, não obstante ao trabalho já realizado relativo à arquitetura do sistema.

Prosteriormente, implementação da arquitetura num digrama de blocos, ou através da escrita de código.

Nesta fase inicial, é esperada ainda pouca complexidade nas características do trabalho em si, pois o foco é o funcionamente da FSM.

O resultado na FPGA deverá ser o seguinte:

- O *Data Size* é ainda predefinido fora da FPGA;
- Aparecimento do tamanho relativo ao *Data Size* inicial e final nas saídas HEX [7..4] da FPGA;
- Uso de um butão para ativar o input de dados;
- LED verde acende sempre que houver input de dados;
- LED vermelho acende sempre que houver erros de checksum.

Numa fase seguinte, os HEX [3..0] apresentarão conteúdo relativo ao checksum error e o *Data Size* inicial poderá ser configurado através de butões (incremento do tamanho).

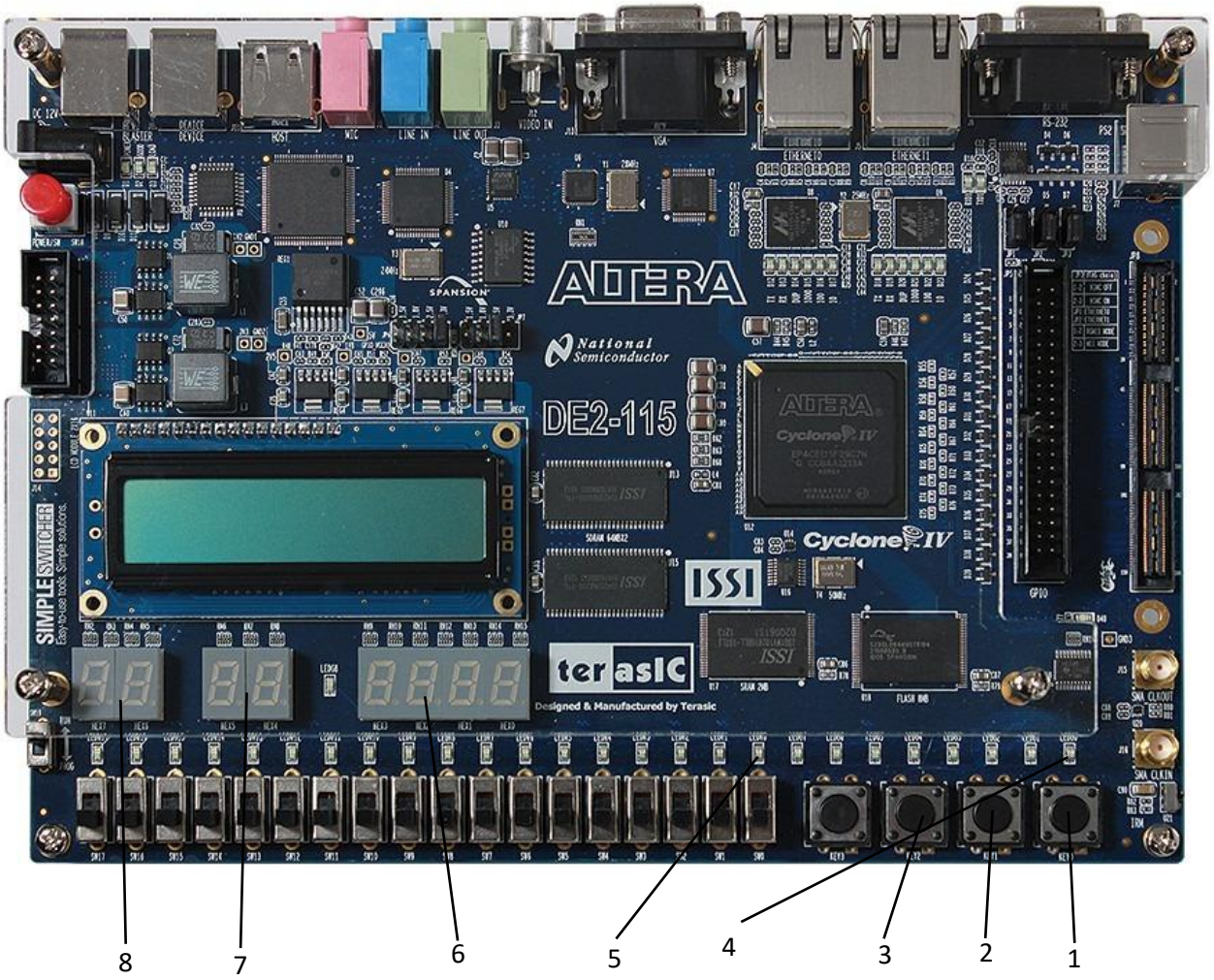
Numa fase final, como requesitado no guião do Projeto Final, será implementada uma comunicação de *multislave*, havendo a possibilidade de o utilizador escolher para que *slave* a *master* enviará os dados.

4. Divisão do Trabalho

O trabalho será divido em duas partes: código da arquitetura *Behavioral*; código da arquitetura *Estructural*.

5. Manual do Utilizador

Imagem de:
https://www.mercadolivre.com.br/



1 – Butão de input de dados.
2/3 – Butões que definem o tamanho do payload.
4 – LED verde que acende sempre que são transmitidos dados com sucesso entre o *MASTER* e o *SLAVE*.
5 – LED vermelho que acende quando há um erro.
6 – Complementa 5, mostrando alguma mensagem.
7 – Tamanho dos dados transmitidos na última transferência.
8 – Tamanho dos dados transmitidos na primeira traferência.