UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

Matheus Dalmolin da Silva

JOGO DA FORCA - ORGANIZAÇÃO DE COMPUTADORES

Santa Maria, RS
2018

SUMÁRIO

1 INTRODUÇÃO

O presente relatório foi elaborado com o propósito de descrever em detalhes o Trabalho I da disciplina de Organização de Computadores, o qual consistia na implementação de um simulador do jogo da forca. A simulação foi implementada utilizando a linguagem *assembly* e o software M.A.R.S (*MIPS Assembler and Runtine Simulator*).

O objetivo do jogo é que o jogador adivinhe a palavra que lhe é apresentada na tela através de espaços, revelando apenas seu tamanho. A palavra a ser descoberta é sorteada, de forma aleatória, a partir de um arquivo que contém ao todo 40 palavras e que é carregado para a região temporária (*buffer*) da memória.

O jogador tenta adivinhar chutando letras do alfabeto, sendo que cada letra só pode ser jogada uma vez. Se a palavra em questão contém a letra, essa é inserida na sua posição na tela. Caso contrário, a letra é colocada em um campo de letras erradas e é descontada uma vida do jogador. Uma vez que é esgotado número de vidas, o jogador perde o jogo. Por outro lado, se a palavra for preenchida antes do total de vidas ser esgotado, o jogador vence.

A elaboração deste trabalho busca aprimorar os conhecimentos do aluno e auxiliar na absorção, e compreensão, dos conceitos vistos em sala de aula, uma vez que a prática é a melhor forma de consolidar o estudo. O jogo da forca engloba todos os conteúdos vistos em sala de aula até a data de entrega do trabalho, logo, a implementação desse trabalho é fundamental para o desenvolvimento do aluno na disciplina.

2 OBJETIVOS

2.1 OBJETIVO GERAL

A proposta relatada nesse trabalho é a de implementação de uma simulação de um jogo da forca.

2.2 OBJETIVOS ESPECÍFICOS

2.2.1 Ler um arquivo com no mínimo 10 palavras não triviais e de tamanhos diferentes, as quais serão utilizadas pelo jogo.

2.2.2 Colocar as palavras em ordem alfabética em uma lista encadeada.

2.2.3 Sortear aleatoriamente uma palavra.

2.2.4 Solicitar ao jogador que tente adivinhar a palavra sorteada, inserindo letras que podem fazer parte da mesma.

2.2.5 A cada erro será desenhado parte da forca na ferramenta *bitmap display* do programa MARS.

2.2.6 Se a forca for desenhada antes da palavra ser completada, o jogador perde o jogo.

3 REVISÃO BIBLIOGRÁFICA

Um computador é uma máquina capaz de processar dados que são necessários para os usuários pelos mais diversos motivos. Esse processamento de dados é realizado através da coleta e manipulação, resultando nas informações fornecidas. O termo “dado” pode ser usado para definir a matéria-prima originalmente obtida e que por si só não transmite nenhuma mensagem significativa. Já a expressão “informação” é utilizada para designar o resultado da etapa de processamento. A partir desse resultado torna-se possível qualificar os dados (MONTEIRO, 2007).

No início da programação, os desenvolvedores utilizavam a notação binária para codificar suas aplicações. Essa forma de programar foi substituída por notações mais intuitivas e que se aproximavam da linguagem natural dos seres humanos. A tradução dessas notações para *assembly* era feita de forma manual, o que levou mais uma vez ao esgotamento dos programadores. Como solução, passou-se a utilizar a própria máquina como ferramenta de auxílio à programação. O primeiro programa criado para traduzir a notação simbólica para binário foi o montador, ou *assembler* (PATTERSON; HENNESSY, 2005).

As instruções simbólicas eram traduzidas para binário utilizando a linguagem *Assembly*, ou seja, o processamento de dados é realizado pelo computador através da execução de instruções em linguagem de máquina, as quais o processador tem a capacidade de executar. Em síntese, o primeiro passo desenvolver um algoritmo, o qual é composto por uma sequência de passos ou ações que determinam a solução de determinado problema computacional. Após é realizada sua codificação em uma linguagem de alto nível, resultando em um código inteligível por humano. Esse código, por sua vez, é traduzido para um código correspondente em linguagem *Assembly*, com o auxílio do interpretador, através da interpretação ou compilação do código-fonte. Nesse momento, as instruções em linguagem de alto nível são transladadas para instruções *assembly* equivalente, através do hardware da máquina. Por fim, o computador já é capaz de executar o programa (PATTERSON; HENNESSY, 2014). A MIPS Technologies, antiga MIPS Computer Systems, foi pioneira da arquitetura MIPS, trazendo para o mercado os primeiros processadores baseados nessa arquitetura. O conjunto de instruções do MIPS tem à seu serviço 32 registradores para realizar operações aritméticas e lógicas. Para programação em linguagem de montagem pode-se utilizar a IDE(*Integrated Development Environment*) MARS, a qual é baseada na arquitetura MIPS. Esse software disponibiliza para o usuário as opções de modo GUI (*Graphical User Interface*) ou modo texto (linha de comando). Sua implementação é em JAVA 1.4.2, usando técnicas de interação humano-computador para facilitar a programação (VOLLMAR; SANDERSON, 2006).

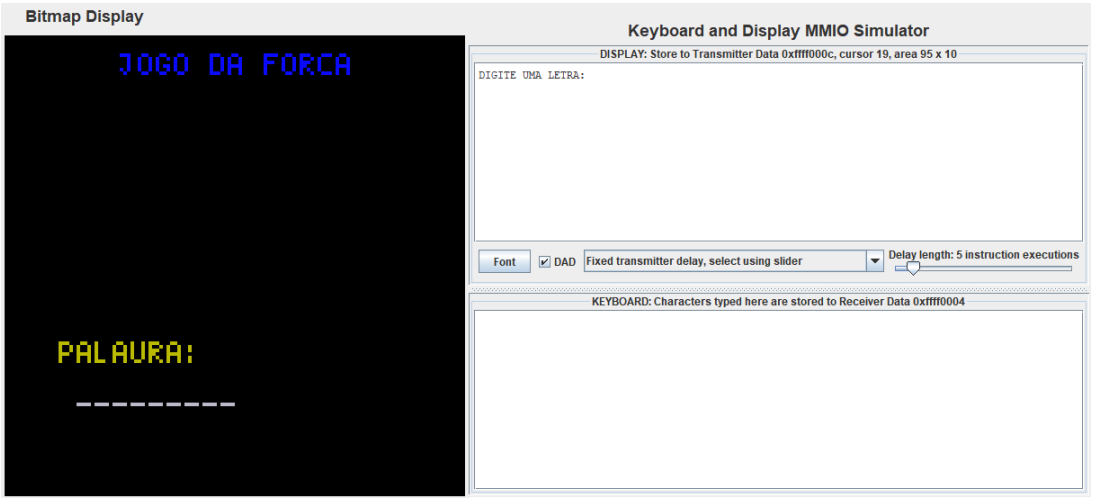
4 METODOLOGIA

Fundamentado na descrição do trabalho e nos conceitos aprendidos durante as aulas da disciplina, para o desenvolvimento da simulação do jogo da forca utilizou-se a linguagem de montagem *Assembly*. Além disso, o *software* usado para essa implementação foi o MARS, uma IDE para a programação em linguagem *Assembly* do MIPS.

5 EXPERIMENTO

A Figura 5.1 apresenta a tela inicial do jogo. Quando o usuário inicia a simulação, é apresentada essa tela. No campo do *KEYBOARD* é onde o jogador digitará a letra para testar na palavra.

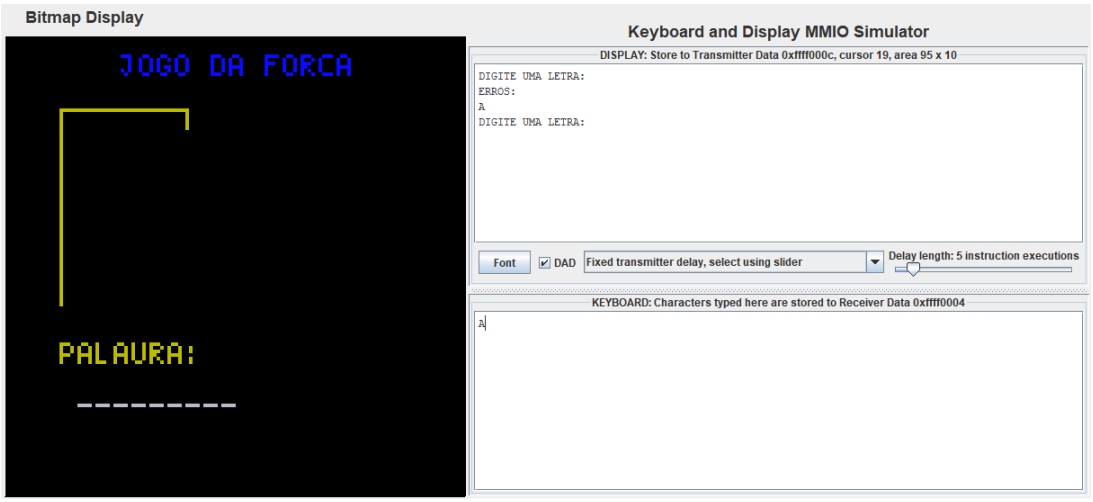
Figura 5.1 – Início do Jogo



Fonte: Arquivo Pessoal

Assim que o usuário tenta uma letra e a palavar não contém essa letra, é impresso no *Bitmap Display* uma parte da forca. E no *Display MMIO Simulator* é impressa a letra que não contém na palavra, ilustradas na Figura 5.2.

Figura 5.2 – Primeira Jogada

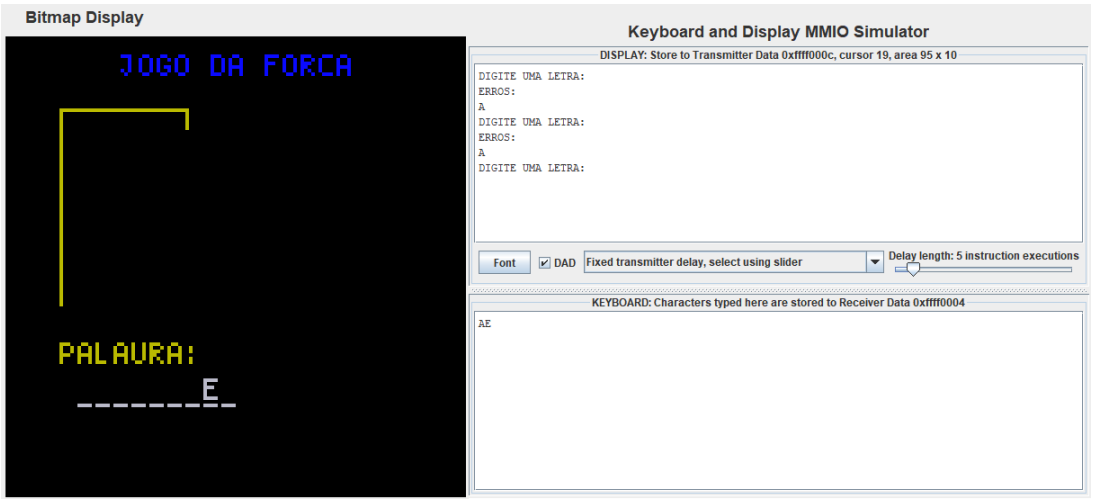


Fonte: Arquivo Pessoal

A figura 5.3 mostra a tela quando o jogador digita uma letra que contém na

palavra, sendo impressa no *Bitmap Display* nas respectivas posições da palavra.

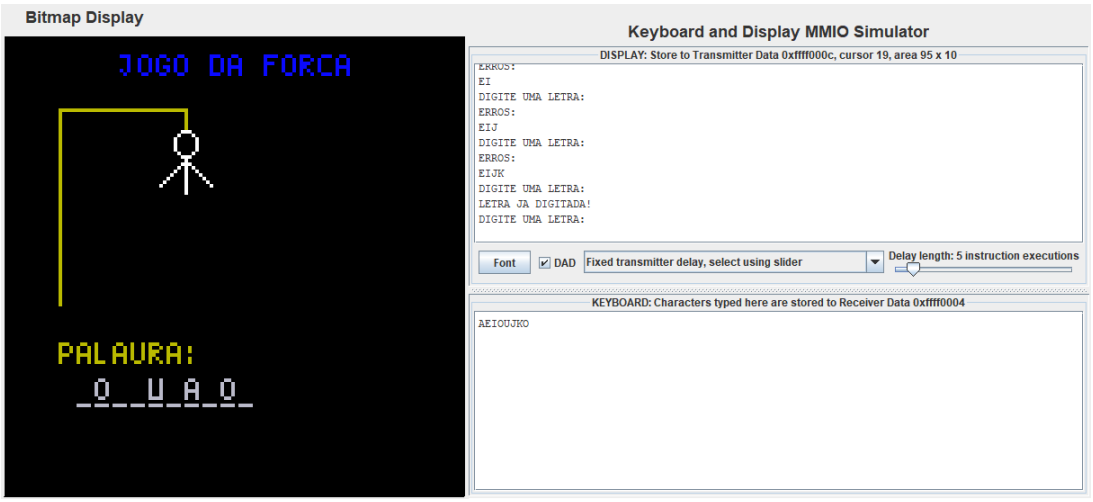
Figura 5.3 – Segunda Jogada



Fonte: Arquivo Pessoal

A figura 5.4 retrata a tela quando o usuário tenta digitar uma letra que já foi digitada anteriormente, imprimindo no *Display MMIO Simulator* a mensagem "LETRA JÁ DIGITADA!", e em seguida pedindo uma outra letra, prosseguindo com a simulação.

Figura 5.4 – Letra Repetida

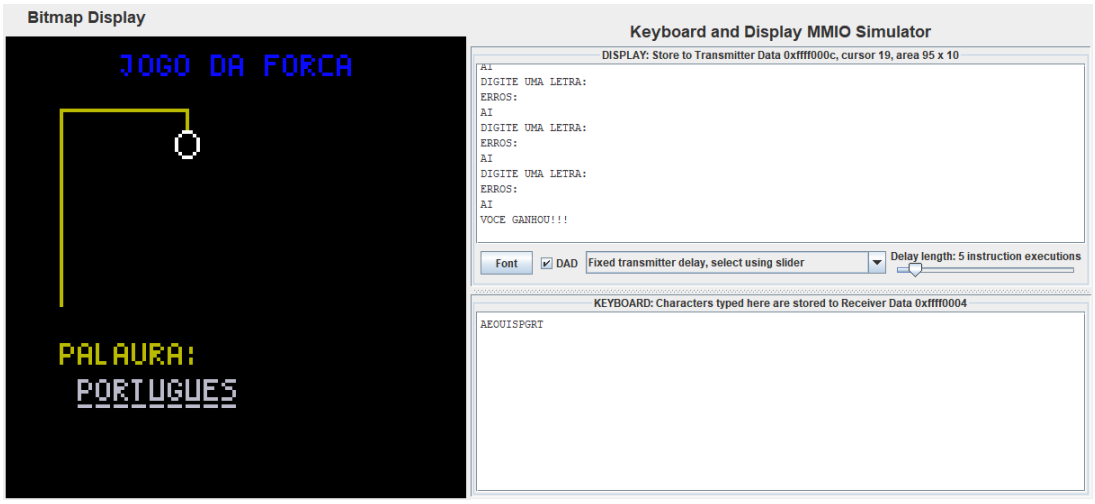


Fonte: Arquivo Pessoal

A figura 5.5 ilustra a tela quando o jogador acerta a palavra que foi disposta no jogo. É impresso no *Display MMIO Simulator* a mensagem "VOCÊ GANHOU!!!", e em seguida a simulação é finalizada.

Caso o jogador não acerte a palavra e perde o jogo, a palavra é impressa no *Bitmap Display* na cor vermelha, sinalizando que ele não completou a palavra.

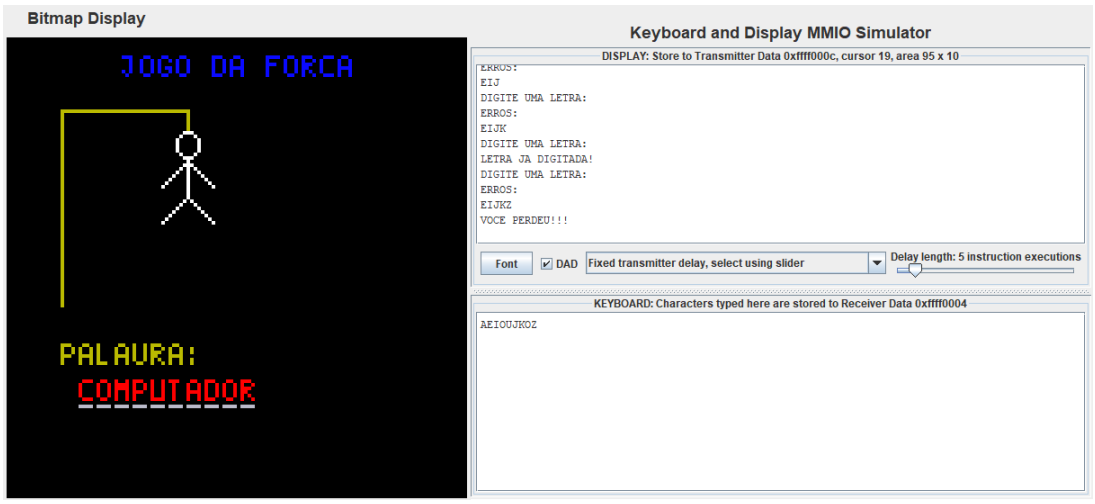
Figura 5.5 – Fim do Jogo: Ganhou



Fonte: Arquivo Pessoal

Também é impresso no *Display MMIO Simulator* a mensagem "VOCÊ PERDEU!!!", ilustradas na Figura 5.6, e em seguida é finalizada a simulação.

Figura 5.6 – Fim do Jogo: Perdeu



Fonte: Arquivo Pessoal

6 RESULTADOS

Conforme os objetivos iniciais definidos nesse relatório, foram alcançados os seguintes resultados: i) Implementação da lógica base do jogo; ii) Implementação das funções que interagem com a parte gráfica do MARS (*Bitmap Display*); iii) Implementação das funções que interagem com o usuário (*Keyboard and Display MMIO Simulator*).

A partir das funcionalidades implementadas, o jogo permite: i) Digitar a(s) letra(s) desejadas; ii) Verificar se a letra faz parte ou não da palavra atual; iii) Exibir as letras já inseridas e, dentre essas, as que estão incorretas; iv) Avisar o usuário se a letra já tiver sido digitada; v) Imprimir na interface gráfica a letra na posição que ele ocupada da palavra; vi) Imprimir partes da forca quando o usuário erra; vii) Se o usuário errou esgotou as suas tentativas, perde o jogo; viii) Imprimir a palavra no caso de derrota; ix) Se o usuário completa a palavra, ganha o jogo.

7 DISCUSSÃO

A meta inicial de implementação da simulação do jogo da forca foi alcançada com sucesso, sendo possível implementar todas as funcionalidades utilizando a linguagem de montagem. O único objetivo específico que não foi atendido foi o 2.2.2, que era de organizar as palavras em ordem alfabética em uma lista encadeada, pois foi uma dificuldade que não consegui vencer por conta da falta de tempo. As demais dificuldades e dúvidas foram esclarecidas com o professor da disciplina ou consultando a documentação do *software* MARS.

A codificação do trabalho contou com a implementação de procedimentos que foram utilizadas para estruturar o programa a fim de torná-lo mais claro e receptivo à outras pessoas que possam utilizá-lo como referência para seus respectivos trabalhos.

8 CONCLUSÕES E PERSPECTIVAS

A simulação relatada nesse trabalho serviu para fixar os conteúdos apresentados na disciplina de Organização de Computadores. Através da implementação dessa simulação, foi possível colocar em prática o que foi aprendido sobre a linguagem *Assembly*.

Portanto, considera-se que trabalhos desse escopo são essenciais para a formação do aluno, tanto para a melhor compreensão da linguagem, quanto para a compreensão dos componentes de hardware.

REFERÊNCIAS BIBLIOGRÁFICAS

MONTEIRO, M. A. **Introdução à Organização de Computadores**. 5 ed. Rio de Janeiro: Editora LTC, 2007. 510 p.

PATTERSON, D. A.; HENNESSY, J. L. **Organização e Projeto de Computadores: a interface hardware/software**. 3ed. Rio de Janeiro: Elsevier, 2005.

____. **Organização e Projeto de Computadores: a interface hardware/software**. 4ed. Rio de Janeiro: Elsevier, 2014.

VOLLMAR, K.; SANDERSON, P. Mars: An education-oriented mips assembly language simulator. **ACM SIGCSE Bulletin**, v. 38, n. 1, p. 239–243, 2006.