الجامعة الإسلامية بالمدينة المنورة
ISLAMIC UNIVERSITY OF MADINAH
كلية الحاسب الآلي ونظم المعلومات
Faculty of Computer and Information Systems

Lab Project

Introduction

This section presents an overview of the project requirements and constraints. Specific details are discussed later. This project is a teamwork project. Each team consists of two students.

Searching is one of the most important operations in many applications. In this project, we are interested in implementing the searching operation in Assembly language. The operation is to find whether a certain word exist in the Video RAM or not.

This is an assembly language project; hence you are only allowed to use the emulator emu8086 for your application. No GUI implementations are accepted. To enhance the graphical presentation, you may make use different functionalities, in text mode, such as character attributes (for foreground and background coloring), and special ASCII characters (for organizing the screen). No further details for this issue are provided; we will leave it to your imagination. But note that a part of the grading is: how you present your results?

The rest of this document describes exact details of the functional requirements and some guidelines to their implementations. In addition, grading criteria hints are given to help you get the highest possible mark in case of not completing the full list of requirements. Please read it very carefully. For any inquiries, please return to the Lab instructor.

Functional Requirements

In this section we divide the project requirements into a set of functionalities. Each function is described separately, and then we provide an overall system description that combines all functions together.

At the beginning of any session, an opening screen should appear (OpenMenu). The user is asked to enter one of the following menu options:

- To Select the length of word searched for, press 1
- To Enter the word, press 2
- To Find whether the entered word exist in V-RAM, press 3
- To Find the number of occurrences of the entered word in V-RAM, press 4
- To Exit, press 5

The user should press 1 for selecting the length of the word to be searched for in Video RAM., 2 for entering the word to be searched for, 3 to find whether the entered word exist in V-RAM or not, 4 to find the number of occurrences of the entered word in V-RAM, and 5 to exit the program.

Length of Word

If the user selects 1 in the previous menu (OpenMenu), the user will be allowed to select the length of the word to be searched for within the V-RAM. The system should allow to enter any length from 1 to 10 characters.

Once length of the word is entered, the system should go back to OpenMenu.

The user can select 1 again to change the previously entered length or select 2 to enter the Word.

Entering the Word

In this part, the system should ask the user to enter the word to be searched for. We will leave it to you to decide about the messages used to enter the Word clearly.

After entering the Word, the system should go back to OpenMenu where he is allowed to press 1 to change length of the Word, 2 to change the Word, 3 to find whether the entered Word exist in V-RAM or not, 4 to find the number of occurrences of the entered Word in V-RAM, and 5 to exit the program.

Searching for the Word

If the user selects 3 in the OpenMenu, the program should search for the entered Word within the V-RAM area. V-RAM starts at 0b800h:0000h and it has 4000 bytes to keep the information for 2000 characters

You can use the CMP instruction for comparison. Or you can use the instruction CMPSB. The instruction CMPSB compares one byte in memory location addressed by the DS:SI with the contents of the byte in memory location addressed by ES:DI. After the comparison, the SI and DI are incremented/decremented depending on the Direction Flag.

The program should respond with YES if the Word exist in V-RAM and responds with NO if the Word does not exist in V-RAM.

Number of Occurrences

If the user selects 4, the program should display the number of times the Word exist in V-RAM. If the user presses a space bar, the program should return to the main menu (OpenMenu).

Summary

This section tries to connect all the previous components into one fully integrated system in a group of points:

- Selecting the length of the Word is done by selecting 1 from OpenMenu.
- Entering the Word is done by selecting 2 from OpenMenu.
- Performing the search for the existence of the Word in V-RAM is done by selecting 3 from OpenMenu.
- If the user wishes the number of occurrences of the Word in V-RAM. This is done be selecting 4.
- If a user wishes to quit the program, he/she could press 5 in OpenMenu. A quit is only accepted when the user is in the OpenMenu mode.
- If the user selects the same option again from Open Menu, then he/she can change the previously entered data for that option.

Guidelines

In this section we present some helping guidelines for the implementation. You are not obligated to follow it; you are free to design your own alternatives.

- For a good and easy message handling, design all your messages to have one static size.
- You must very carefully organize your program, by using labels, as necessary, for easier code maintenance, and bug tracking.
- Try to think of the program as a C/C++ program, and then convert all high-level language to their corresponding low-level language.
- Try delivering complete atomic functions. For example, you could start by implementing a part of the required functionality. When you are 100% sure it is working fine, take a snapshot of that program, then move on to implementing the next part of functionality.
- Delivering concrete requirements is awarded more than partially incomplete ones. You can consider the set of concrete requirements (and corresponding grading criteria) as follows:
    - Selecting length of Word
    - Entering the Word
    - Performing the searching
    - Computing number of occurrences
    - Overall logic of the program

Honor System

READ CAREFULLY. Any identified cheating of any type is simply graded a big ZERO. This project is a teamwork project, but at the end it is an academic material, team members should be completely aware of every piece of code written by the team. On the project delivery day, any question could be asked to any team member at random.