PROJECT REPORT

Computer Organization & Assembly Language (CEL-221)

BS(CS)-3B

Project Title: HANGMAN

Group Members

Name	Enrollment
1.NOMAN TARIQ	**02-134211-010**
2.HOOND RAJ	**02-134212-108**
3.	
4.	

Submitted to:

BAHRIA UNIVERSITY KARACHI CAMPUS

Department of Computer Science

Module wise Work/Task Distribution

Name	Enrollment	Task
1.NOMAN TARIQ	**02-134211-010**	Does research on the project, programmed the game, and wrote some part of the report.
2.HOONDRAJ	**02-134212-108**	Experimented on how the program works in assembly language and wrote some part of the report & proposal.
3.		
4.		

Project file comprising of Title page, Introduction, Problem statement, methodology, objectives, diagrams, applications, scope, results, code and must be submitted to the LMS.

Abstract

The objective of this research is to know whether there is significant influence of using Hangman game toward students vocabulary mastery. This project describes the realization of the Hangman game in emu8086 using the assembly programming language. Knowledge of the work in the software tool is demonstrated, as well as advanced knowledge of assembler and work with library functions. The obtained game uses less CPU time than its realizations in other higher level programming languages.

TABLE OF CONTENT

Table of Contents

CHAPTER 1

INTRODUCTION

Project Description

Hangman is a popular word guessing game where the player attempts to build a missing word by guessing one letter at a time. After a certain number of incorrect guesses, the game ends and the player loses. The game also ends if the player correctly identifies all the letters of the missing word. The game is realized using the already known term which is to be guessed. At the beginning of the game, the player tries to guess the term by writing letters from the standard input. If the entered letter does not exist in the specified term, then the part of Hangman is added. However, if the letter exists in the term which a player tries to guess, then the letter is written to the appropriate position in the word. If the letter has already been hit, the user is informed necessarily about it, and if the user by accident writes it, then it does not affect the game. Every time a new letter is entered from the standard input, it is necessary to print it on the screen. When a user guesses the current term, they are allowed to hit the new one. If the user misses the letters enough times, the entire Hangman is drawn out, and a player loses the game.

Scope of the Project

The scope of this project is to polish our skills, entertainment purpose and to accomplish good grades. This is a simple Hangman game using assembly programming language. Beginners can use this as a small project to boost their programming skills and understanding logic. Suppose a random word the player gets limited chances to win the game by guessing the correct letters. When a letter in that word is guessed correctly, that letter position in the word is made visible. In this way, all letters of the word are to be guessed before all the chances are over. You'll also be doing some statistical analysis of the words used in the Hangman game. This game can have varied applications I the context of word formations and puzzles. Its

knowledge can be valuable to many other games like CROSSWORD PUZZLES, WHEEL OF FORTUNE, SCRABBLE etc.

Modules in Project(Individual working of each functionality)

Firstly a menu is displayed from which the user can select 3 options the first one is for new game to start the game by pressing 1. The second option is about the hardness level of the game by pressing 2 and the third one is about the high scores achieved in the game. The last one is ESC to quit the game. If a user wants to play the game again he/she can press any key to play again. 10 incorrect guesses are only allowed in this game after that the entire hangman is made and the user loses the game.

Project Features

Our project is very unique in its features. People can entertain thereselves by playing this easy game which is for all age group people. This game makes a person's brain strong and help in English vocabulary, learn new words and techniques. It includes the functions as MOV, CALL, CMP, JZ, ADD, DIV, POP, PUSH and many more.

CHAPTER 2

ANALYSIS

Existing System

The existing system of our project is that we can play the game easily by following the displayed functions to start or end the game by pressing numeric keys after that we have all the letters applicable which are suitable in need of the game by guessing words. We can also go back to menu after starting the game by pressing 1 and also reset by pressing 2. For checking score the user should press 3 and ESC any time to quit the game. The display of our game also shows all the incorrect letters and all the used vowels and left constants. If the user wins the game it displays CONGRATULATIONS in caps letters. When the user enters incorrect letter it displays TRY AGAIN.

Proposed System

The proposed system of our project was that how the game operates. After a certain number of incorrect guesses the game ends and the user loses the game. The game also end if the player or user correctly identifies all the letters of the missing word. If the entered letter does not exist in the specifies term a part of hangman is added. However, if the letter exists in the term which a player tries to guess, then the letter is written to the appropriate position in the word. If the letter has already been hit, the user is informed necessarily about it, and if the user by accident writes it, then it does not affect the game. Every time a new letter is entered from the standard input, it is necessary to print it on the screen. When a user guesses the current term, they are allowed to hit the new one. If the user misses the letters enough times, the entire Hangman

is drawn out, and a player loses the game. Our proposed system also includes file handling

CHAPTER 3

SYSTEM IMPLEMENTATION

Introduction

An algorithm for the realization of popular computer game Hangman. The basic idea of this work was the development of Hangman game using Emulator EMU8086 Environment for easier implementation of the game in assembly programming language. Assembler (assembly) language is a low-level programming language, and it is specific to a particular processor (microcontroller) architecture. As the oldest programming language, and of all languages, it bears the closest resemblance to a native machine language. It provides a direct access to computer hardware, requiring you to understand much about your computer's architecture and operating system. A large number of modern microcontrollers are based on a similar architecture as the microcontroller 8086 and have similar abbreviations for the names of assembly instructions. So, the study of microcontroller 8086 is reckoned not only for pedagogical reasons but also for usability. when designing microcontrollers it is necessary to know the algorithm which the hardware will use and that it is necessary to know the hardware on which the software will be executed to write an optimal algorithm.

Assembly Language Code

```
;HANGMAN GAME PROJECT -- CAT-3
data segment
 welcome_message db "                    HANGMAN GAME$"
 lives_msg db "                  LIVES:$"
 type_message db "                    Type a letter: $"
 new_line db 13,10,"$"
 win_message db "                     YOU WIN!$"
 lose_message db "                    YOU LOSE...BETTER LUCK NEXT TIME !!!$"
 space db "                   $"
 clue db "                CLUE:THIS IS A MUSICAL INSTRUMENT...$"
 ;word db "firefox$"
 discovered_word db 5 dup("-"),"$"
 word_size db 5
 lives db 5
 hits db 0
 errors db 0
 s1 db 1,"piano$"
 s2 db 2,"flute$"
 s3 db 3,"cello$"
 s4 db 4,"tabla$"
 s5 db 5,"viola$"
 s6 db 6,"bongo$"
 s7 db 7,"sitar$"
 s8 db 8,"organ$"
 s9 db 9,"brass$"
 r db 0h
ends

stack segment
 dw 128 dup(?)
ends
```

```
extra segment

ends

code segment

start:
 ;set segment registers

 mov ax, data
 mov ds, ax
 mov ax, extra
 mov es, ax
wordl:
   MOV AH, 00h
   INT 1AH
   mov  ax, dx
   xor  dx, dx
   mov  cx, 9
   div  cx       ; here dx contains the remainder of the division - from 0 to 9
   add  dl, '1'  ; to ascii from '0' to '9'
   mov r,dl
   mov ah, 2h   ; call interrupt to display a value in DL
   int 21h
   mov bx,00
   lea si,s1
   mov bl,[si]
   sub dl,30h
   cmp bl,dl
   je main_loop
   jmp c2
c2:
   mov dl,r
   mov bx,00
   lea si,s2
```

```
c9:
   mov dl,r
   mov bx,00
   lea si,s9
   mov bl,[si]

   sub dl,30h
   cmp bl,dl
   je main_loop
   jmp exit



main_loop:

 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
 lea dx, new_line
 call print
```

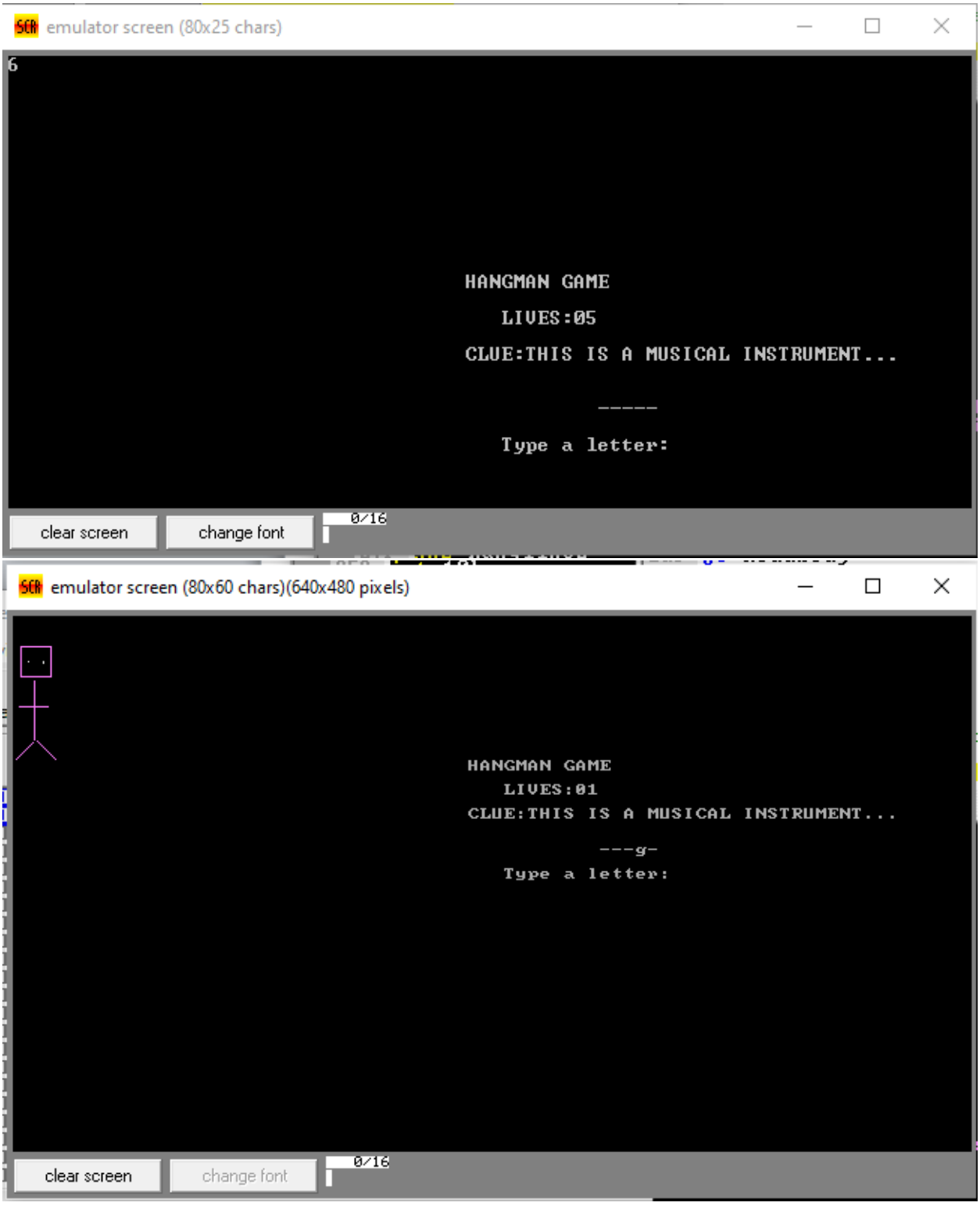
```
lea dx, new_line
call print
lea dx, new_line
call print

lea dx, welcome_message
call print

lea dx, new_line
call print

lea dx, new_line
call print

lea dx,lives_msg
call print

mov al,lives
sub al,errors
mov bl,100
mov ah,00
div bl

mov dl,al
mov cl,ah
add dl,30h
mov ah,02
int 21h

mov dl,cl
add dl,30h
mov ah,02
int 21h

lea dx,new_line
call print
```

```
 lea dx,new_line
 call print
 lea dx,clue

 call print
 lea dx,new_line
 call print
 lea dx,new_line
 call print
 lea dx,new_line
mov dx,83

rline:
int 10h
inc cx
inc dx
cmp cx,16
cmp dx,97
jnz rline
ret



hang:
mov cx,20        ;setting row pixel for hanging
mov dx,6         ;setting column pixel
mov bx,10        ; line size

hangline:
int 10h
inc dx
dec bx
jnz hangline

mov cx,20        ;setting row pixel
mov dx,6         ;setting column pixel
```

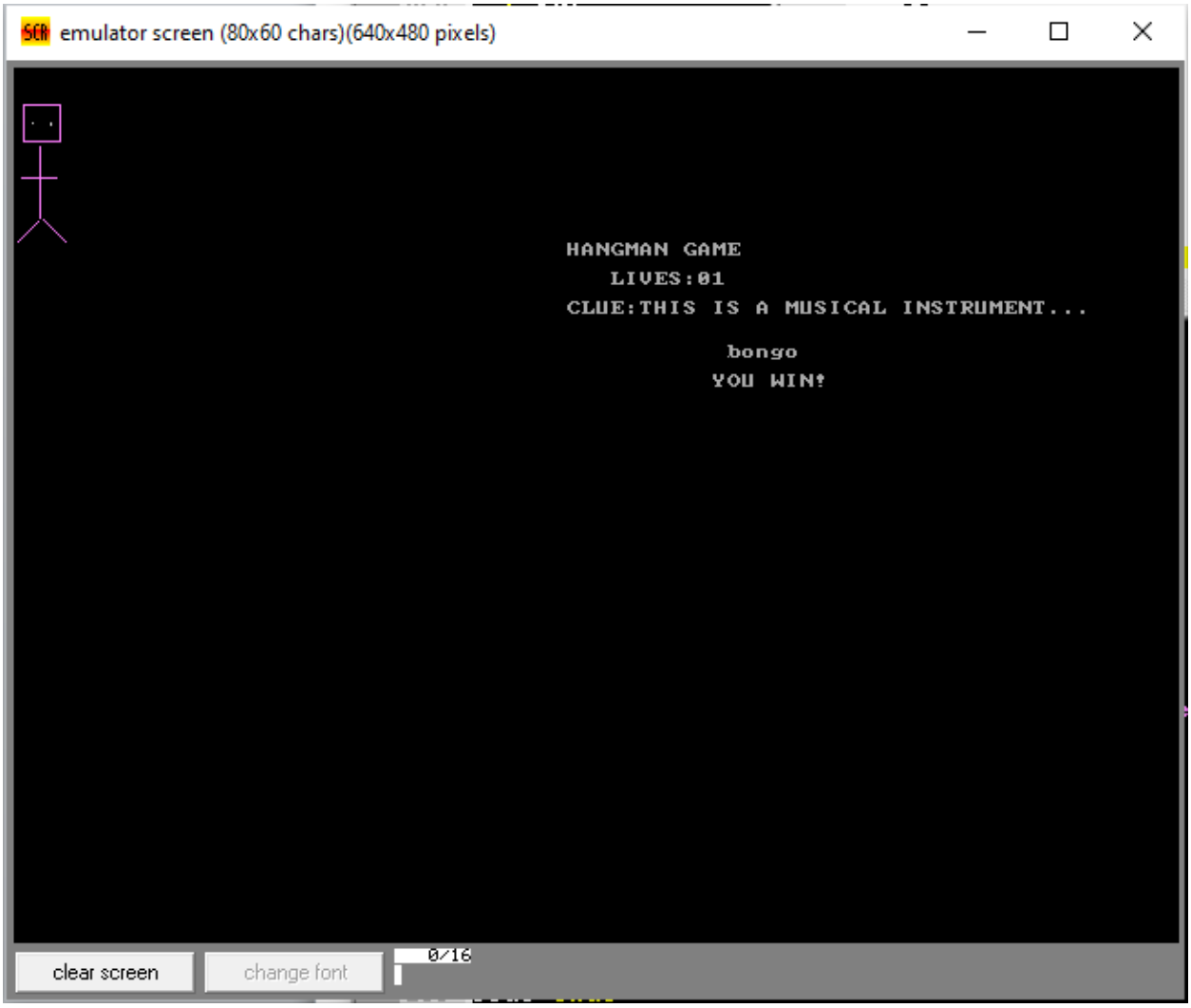
```
mov bx,20

hangline1:
int 10h
inc cx
dec bx
jnz hangline1

mov cx,40          ;setting row pixel
mov dx,7           ;setting column pixel
mov bx,80

hangline2:
int 10h
inc dx
dec bx
jnz hangline2

ret



code ends

end start
```

CHAPTER 4

SAMPLE SCREEN SHOTS OF OUTPUT




emulator screen (80x60 chars)(640x480 pixels)
HANGMAN GAME
LIVES:01
CLUE:THIS IS A MUSICAL INSTRUMENT...
-o-go
Type a letter:
clear screen
change font
0/16


emulator screen (80x60 chars)(640x480 pixels)
HANGMAN GAME
LIVES:01
CLUE:THIS IS A MUSICAL INSTRUMENT...
bongo
YOU WIN!
clear screen
change font
0/16