**EE 316 Computer Engineering Junior Lab**

**Design Project 3**

**Spring 2017**

* Specification:  **Digital Hangman**
* Lab Demonstration Due Date: March 2, 2017
* Written Report Due Date: March 6, 2017

**Parts List:**

1. HD-6402 UART
2. USB to Serial adapter
3. MAX232 Level changer
4. PS/2 Keyboard and PS/2 socket (6-Pin Mini DIN connector)
5. 1-line/20 or 2-line/20 character LCD display
6. TMS27C256 PROM (32K by 8 bit)
7. 74LS92 and a 1.8432 MHz oscillator
8. Other TTL or CMOS chips available in the Stock room (e.g., 74xx14, 74xx92, 74xx164, 74xx04, 74xx30, 74xx246, 74xx163, 74xx169, etc.), where xx = L, LS, ALS, HCT, HC, AHCT, etc.

You are to design an electronic system that allows a person to play hangman using a PS/2 keyboard and an LCD display panel connected to hardware. The hardware should communicate with a remote personal computer (PC) using its USB-serial adapter and display the words on a single line LCD display driven from the PC. The design must not use any CPLD or FPGA chip.

The system consists of display hardware (LCD and RS-232 serial line interface), a PS/2 keyboard (with its associated control/translation hardware), a standard serial cable, and Windows XP compatible software that provide a user interface to control the display. The hardware is separately powered, and when powered on the LCD should have a display that is blank except for a single cursor (underline located at the left side of the LCD Screen). The display hardware should be designed to interface with a serial port having the following characteristics: **Baud rate: 9600; parity: none; data bits: 8; stop bits: 1**

The software should have a graphical user interface (GUI) to display the status of the game. A picture with hangman’s noose, etc., should be shown on the GUI. The Software should be able to read a text file that contains a set of 5-10 words. The word puzzles should be of variable lengths, not exceeding 20 letters per word.

The text file is to be loaded at the beginning of the game. The software should display the message: “New Game?” on the GUI and on the LCD module. The player must reply in the affirmative by pressing the “Y”-key on the PS/2 keyboard connected to the hardware. Pressing any other key should do nothing. Note that this keyboard is not the one connected to the PC. The keyboard’s hardware should be able recognize, at the minimum, the English alphabets from A through Z. The software should then randomly pick one of the puzzle words from the text file and display the positions of every letter in the word with an underscore “\_”symbol. On the LCD display module the word would be left justified.

After the puzzle is displayed, the player should be allowed to guess a letter using the external PS/2 keyboard. A maximum of 6 guesses are allowed per puzzle. If a letter was guessed correctly, the system should display every instance of the letter in the word. For a wrong guess, the “picture” on the GUI should change. On the hardware, a 7-segment display should be used to indicate the number of guesses that remain.

If the player successfully picks every letter, the PC should then send a congratulatory message saying **“Well done! You have solved N puzzles out of M”**. If the player fails, the message should be **“Sorry! The correct word was XXXXX. You have solved N puzzles out of M”**. Since these messages have more than 20 letters and spaces, it should appear at the right edge of the LCD display and scroll across the screen at a rate appropriate for reading of the message. The system should then display: “New Game?” The player can always request a new puzzle by typing “Y”, unless all of the words from the text-file have been used up. At this point, the player should be able to end the game by typing the letter “N”. When the game ends, the LCD should display the final points (“4 correct out of 10”) and after a short duration, the message: “GAME OVER”.

**Performance objective: Minimize the number of ICs used. You may use TTL/CMOS chips and a PROM in your design. However, cannot use FPGAs or CPLD. The user interface should be easy to use, incorporate multi-threading, and it should be self-explanatory to any user familiar with basic Windows software.**

**Options:**

**1. (2.5 points)** You may provide an option that permits continuous “looping” display for messages longer than 20 characters. The loop should either display the message for a user set number of times, or allow the user to exit at any time.

**2. (2.5 points)** You may provide an option for your software to detect if your hardware is connected; the user should be informed as to the state of the connection.

**Documentation:**

Check on the LCD display in Moodle..

Here are a couple of links for serial port programming in JAVA and C++, respectively

<http://en.wikibooks.org/wiki/Serial_Programming/Serial_Java>

<https://msdn.microsoft.com/en-us/library/system.io.ports.serialport(v=vs.110).aspx?cs-save-lang=1&cs-lang=cpp#code-snippet-1>

Here is a good webpage for the PS/2 protocol:

<http://www.computer-engineering.org/ps2protocol/>

The UART (an HD-6402) is available from [www.intersil.com](http://www.intersil.com). Here is the direct link: <http://www.intersil.com/data/fn/fn2953.pdf>

The RS-232 voltage level converters are available from [www.maxim-ic.com](http://www.maxim-ic.com). The datasheet for MAX232: <http://datasheets.maxim-ic.com/en/ds/MAX220-MAX249.pdf>.

Teams:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Team1** | **Team2** | **Team3** | **Team4** | **Team5** | **Team6** | **Writer** |
| Bruska | Griffin | Bruce | Lowit | Oliver | Marsanskis |  |
| Straw | Trahan | Zander | Heck | Shippee | Craddock |  |
| strenk | Farden | Law | Michaels | Kuhns | Beyer |  |