EE 337: Microprocessors Laboratory (Spring 2021)
Indian Institute of Technology Bombay
Project: 20 points Date: March 8, 2021

Memory Game on Pt-51

1. [20 points] In this project, you will be writing a game program for a memory game. The player will type on a keyboard connected to the Pt-51 board via UART.

   - In the game, the LCD will display a sequence of 10 word-digit pairs, where each pair is displayed for 1 second. A random digit from `0` to `9` is shown next to each word.
     - For example, a sequence could be the following:

       ```
       king  5
       idea  7
       left  0
       good  1
       exit  5
       taxi  5
       chef  3
       lava  1
       cafe  2
       logo  9
       ```
     - Each word is 4 letters long.
     - Words should not repeat in the list, i.e. all the 10 words should be different.
     - The digits next to the word can repeat.
   - After the 10 word-digit pairs have been displayed, there is a delay of 3 seconds with the LCD displaying the string `Get Ready` on the first line. The second line is blank.
   - Then the same sequence of 10 words from the list appears one at time on the first line, with each word displayed for 3 seconds. In the 3 second duration, the player is expected to type the digit which appeared next to the word in the beginning of the game. The digit typed by the player should appear on the second line.
   - After all 10 words have been shown, the number of correct guesses is displayed on the first LCD line and the highest score so far is displayed on the second LCD line, for 3 seconds. For example, if the player get four digits correct in the current game and has a high score of 8, then the display will look like the following.

     ```
     Score:               4
     High Score:          8
     ```
   - After showing the score, the game restarts by showing a new sequence of 10 word-digit pairs.
   - To make the game interesting, the word-digit pairs need to change from game to game. We will use a linear feedback shift register (LFSR) of length 5 to generate pseudorandomness.
   - Choose any 31 four-letter words from the list given at `https://gist.github.com/fogleman/c4a1f69f34c7e8a00da8`. Note that $2^5 - 1 = 31$.

- Associate each of the 31 words with a unique non-zero 5-bit sequence. The zero 5-bit sequence is the one with all zeros `00000`.
- Consider a LFSR with 5 registers (each containing a bit). We can use a single byte to store the current state of the LFSR. If the current state of the 5 bits is $(b_4, b_3, b_2, b_1, b_0)$, the next state is

$$(b_3 \oplus b_0, b_4, b_3, b_2, b_1),$$

  where $\oplus$ represents bitwise XOR. With this next-state rule, the LFSR will cycle through all the non-zero 5-bit states with a period of 31. This is an example of a maximal-length LFSR `https://en.wikipedia.org/wiki/Linear-feedback_shift_register`.
- Initialize the LFSR state $(b_4, b_3, b_2, b_1, b_0)$ to a non-zero value (this is done only once when the program loads for the first time).
  - For the first word in the 10 word sequence, choose the word associated with the 5-bit sequence $b_4b_3b_2b_1b_0$.
  - For the digit to be associated with the word, take the remainder when the nibble $b_3b_2b_1b_0$ is divided by 10. For example, if $b_3b_2b_1b_0$ is 1110, the digit is 4.
  - Calculate the next state of the LFSR.
  - For the second word in the 10 word sequence, choose the word associated with the new 5-bit sequence $b_4b_3b_2b_1b_0$.
  - For the digit to be associated with the word, take the remainder when the new nibble $b_3b_2b_1b_0$ is divided by 10.
  - And so on, until 10 words have been displayed.
- Preserve the last state of the LFSR for the next game.