UNIVERSIDAD DE PUERTO RICO

RECINTO UNIVERSITARIO DE MAYAGUEZ

DEPARTAMENTO DE INGENIERIA ELECTRICA

Project Report: “Simon Says”

José Antonio Rodríguez Rivera

802-12-6715

INEL4206-030

Dr. Rogelio Palomera

Project Specification

The task is to recreate the popular game “Simon Says”. The game consists in repeating the pattern generated by the microcontroller. To illustrate said pattern, 4 LEDs and a buzzer will be implemented.

Game Description

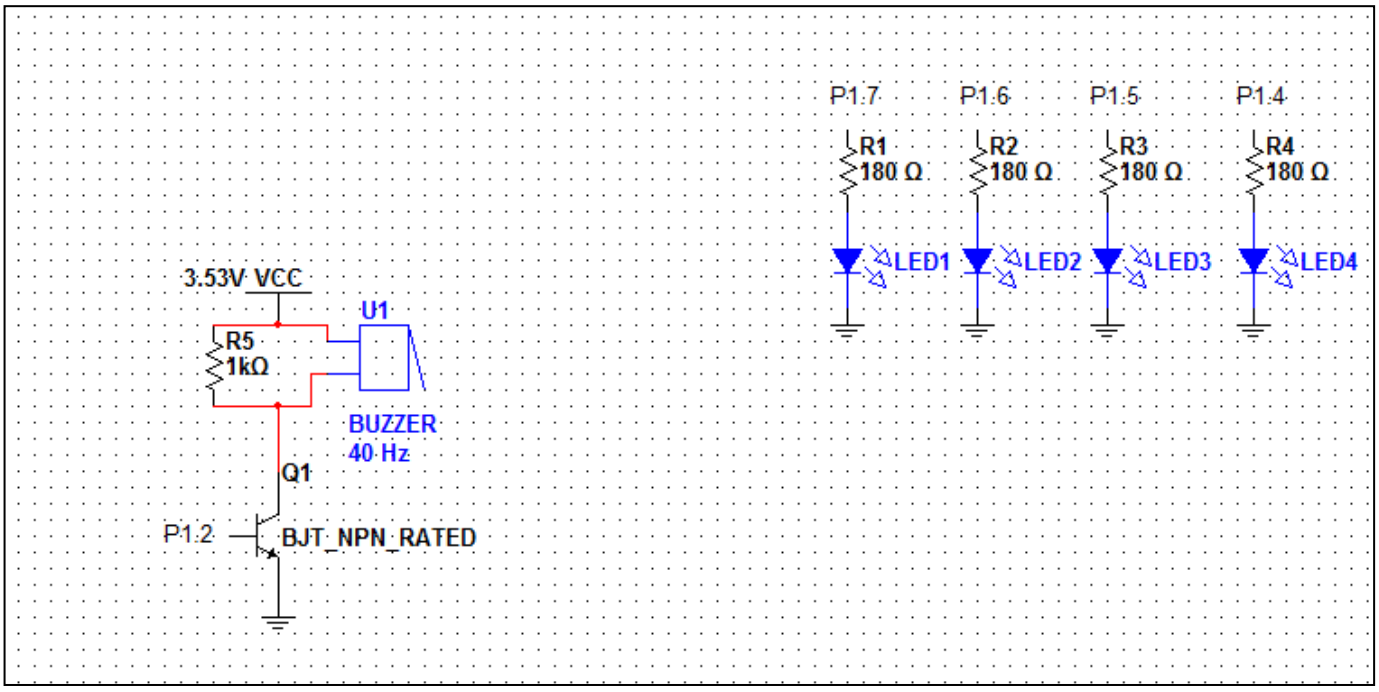
When the game is turned on, it will request the difficulty to be played. At the start of a round, the microcontroller will generate a pattern that will be shown with the LEDs and their corresponding sounds using the buzzer. The player must replicate the sequence given. If the player fails, he/she would have to start all over again, as the game will end. However, if the player enters the correct pattern, he/she advances to the next round. Every 5 rounds, the number of LEDs in the pattern will increase, until all 20 rounds are over.

Materials to be used in this project

The materials that will be used in this project are as follows:

- Resistors
    - $1\ k\Omega\ (1)$
    - $180\ \Omega\ (4)$
- 4 LEDs
- 4 Push Buttons
- NPN Transistor (2N3904)
- Buzzer
- MSP430 Launchpad

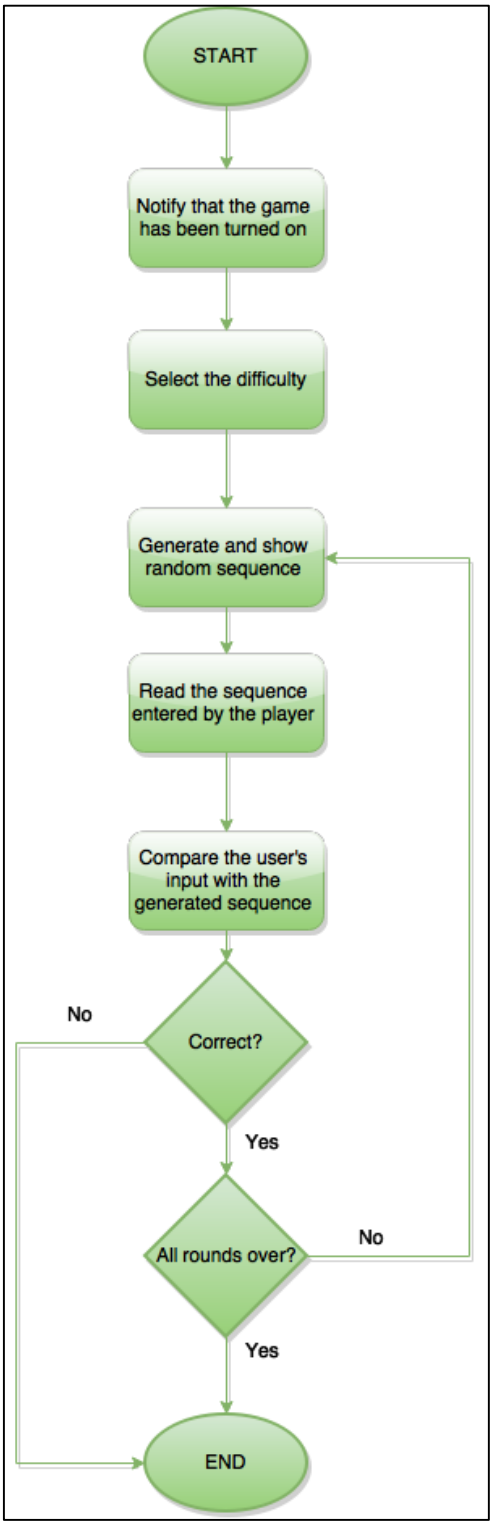
Output Schematic



Note: P1.2 is configured as a PWM output.

As for the inputs, P2.3, P2.2, P2.1 and P2.0 are used. One pin of the push buttons is connected to Vcc and the other to the corresponding pin. In order for it to work, the pins have their internal resistors as pull-down resistors.

Flowchart of the general processes



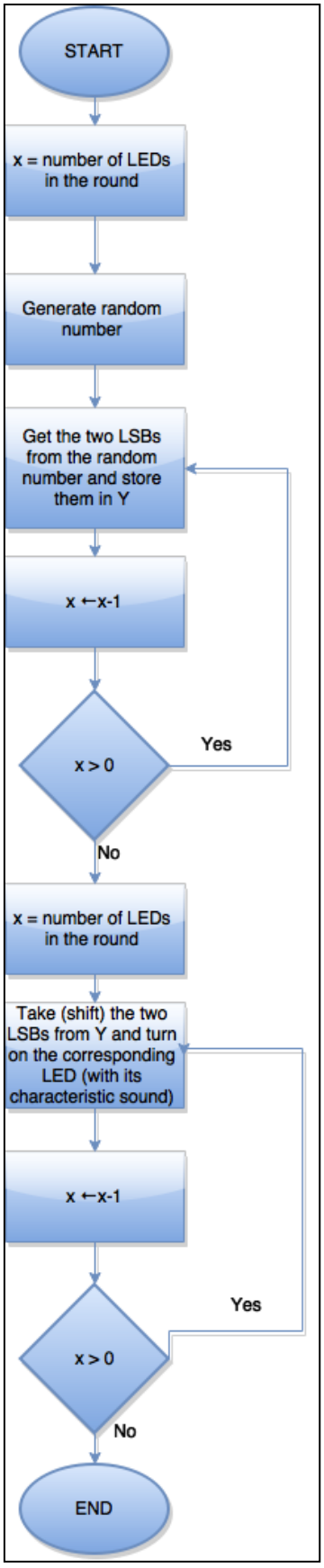
General processes of the game

- When the game is turned on, the player will be notified with the LEDs turning on and the buzzer will sound.
- After this is done, the player must press a button to select the difficulty, which will be explained in detail later on.
- After selecting the difficulty, the sequence will be shown to the player.
- The player must enter the sequence shown, using the buttons provided, and it will be compared with the generated sequence. If they match, the player will be notified that the sequence was correct, and will proceed to the next round. If not, the player will be notified that the sequence was wrong, and the game will end.

Difficulty Selection

- After the game is turned, the player will be greeted by the LEDs turning on and sounds in the buzzer. At this moment, the player is required to press a button to select the difficulty:
    - Leftmost Red LED Button – Normal Difficulty
        - LEDs will turn on during the sequence and their corresponding sounds will be heard.
    - Green LED Button – Hard Mode
        - LEDs will turn on during the sequence and their sounds will be heard, but they will be much faster.
    - Yellow LED Button – Expert Mode
        - LEDs will **not** turn on, only their sounds will be heard.
- The player will be notified after selecting a difficulty, and the rounds will begin. Each round will consist of 20 sequences, starting with 1 LED for the first 5 sequences, and adding 1 LED for every 5 sequences (maximum of 4 LEDs per sequence).

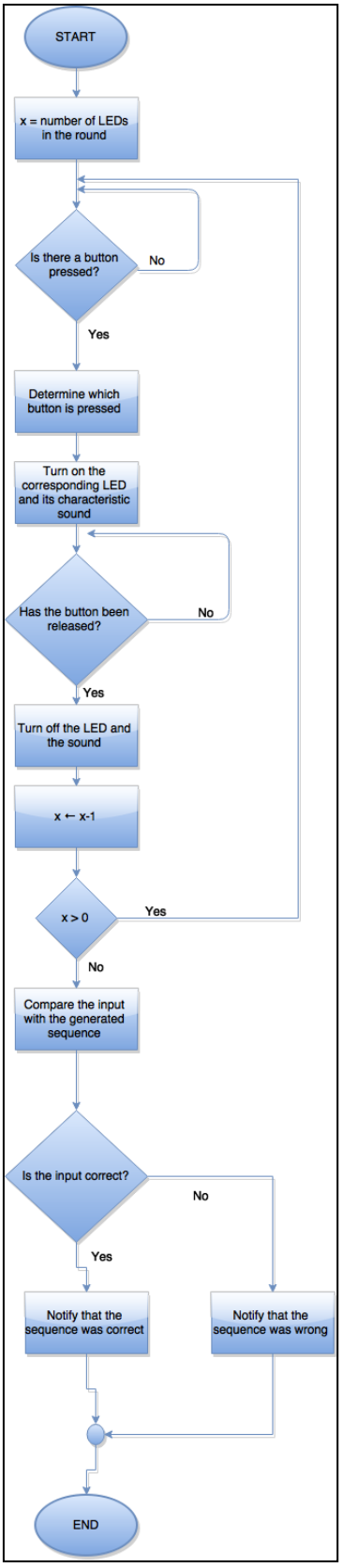
Generating the random sequence



Generating the random sequence

- To generate the random number, a variation of the Linear Congruential Generator algorithm is used. The concept behind it is to take a number and multiply it by a large prime number, and usually add (sum) another number. In my case, I used the number 48271 and added 0.
- After the number was generated, we take (shift) the two LSBs from the number generated for each LED in the current round.
- Since we took two bits, there are only 4 combinations possible, which are used to determine the LED to turn on. The determination process is as follows:
    - 00 – Rightmost Red LED
    - 01 – Yellow LED
    - 10 – Green LED
    - 11 – Leftmost Red LED
- While the corresponding LED is turned on, the sequence is saved in a format that is similar to the format seen when reading the input, in order to make easier the comparison process later on. The format is as follows:
    - 0001 – Rightmost Red LED
    - 0010 – Yellow LED
    - 0100 – Green LED
    - 1000 – Leftmost Red LED

Reading the player's input



Reading the player's input

- After the sequence is shown, it's time to read the input. The microcontroller keeps checking (polling) if one of the 4 LED buttons or the reset button (P1.3) is pressed.
- When the MSP430 detects that a button has been pressed, it carries on to an individual test to see the exact button that's being pressed.
- After detecting which button was pressed, the input is saved and converted to the following format:
    - 0001 – Rightmost Red LED Button
    - 0010 – Yellow LED Button
    - 0100 – Green LED Button
    - 1000 – Leftmost Red LED Button
- Afterwards, the corresponding LED is turned on, and the sound is generated in the buzzer, and now the microcontroller keeps checking (polling) if the button has been released. When it is released, the LED and the sound are turned off. This process is repeated until the number of buttons pressed equal the number of LEDs in the round.
- When the reading process is finished, the user's input and the generated sequence are compared. If they are equal, the player will be notified with a specific pattern and sound, and will go on to the next round. If they're not equal, both red LEDs will turn on, along with a sound from the buzzer, indicating that the game is over.