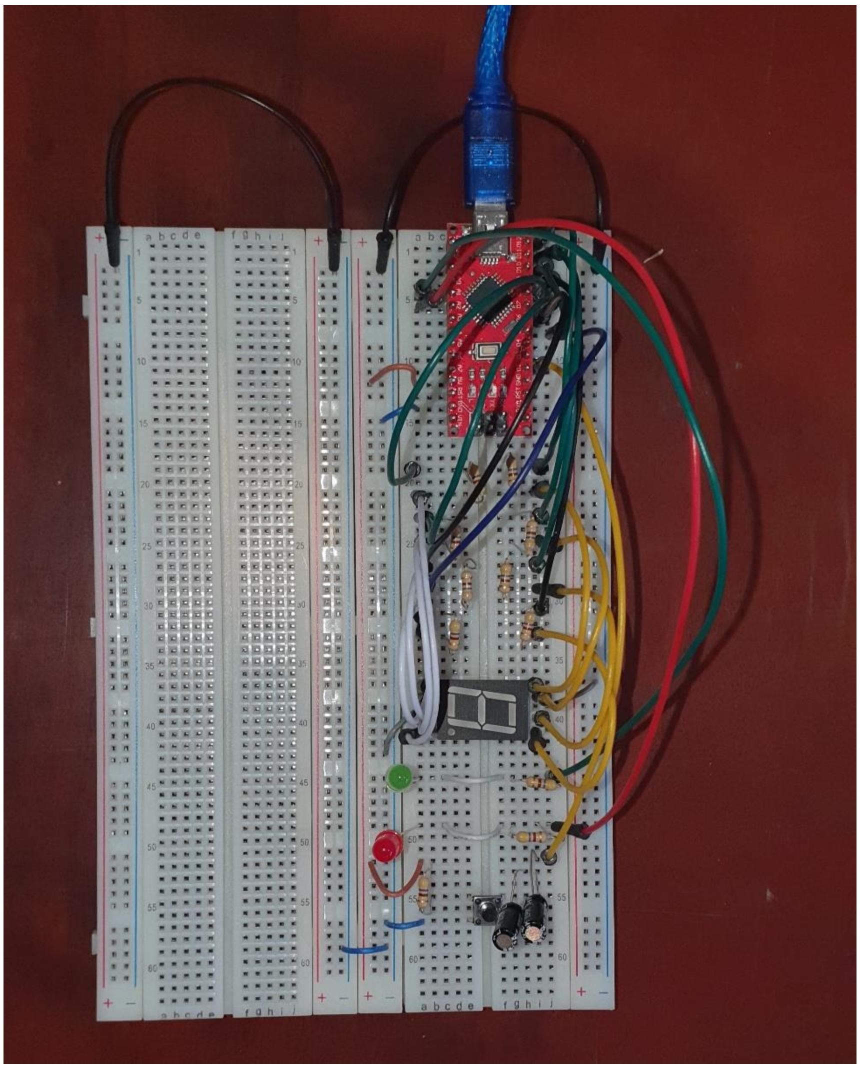
Assembly Number Game

By: Shen Reddy



Functionality and Implementation

This project has implemented code to create a simple mental maths game. The program generates a random number sequence at the push of a button and displays each number for 3 seconds on a 7-segment LED. The program determines whether the sum of the numbers displayed, since the last time the button was pushed, is divisible by 3, based on the push of a button. The program flashes a green LED, for every correct push, and flashes a red LED for every incorrect push. The program keeps track of the user's score. The score will be increased by one point for every correct push, and decreased by 2 points for every incorrect push, with the lowest possible score set as 0. The game ends after 60 seconds have elapsed since the game started, and the user's score will be displayed and flashed every 2 seconds for the user to see. If the score is less than 9, the program will simply flash the value of the score on the display. If the score is equivalent or greater than 10, the score is split into two segments, where the high digit of the score is flashed for 2 seconds, after which the low digit of the score will flash for 2 seconds, accompanied by a decimal point.

The functions that work as described without any issues are the push to start function, the random number generator, the displaying of the random numbers on the 7-segment LED, the timer to flash a random number every 3 seconds, the timer to determine the end of 60 seconds and the displaying of the score for any value between 0 and 9.

Not all of the listed functions work correctly as desired. The check to determine a correct push sometimes does not flash the correct LED due to switch bounce. The function to display the score if it is equal to or greater than 10 does not correctly display the lower digit of the score.

Instruction Manual

- The game begins when the push button is pressed for the first time. Once the push button has been initially pressed the random numbers will begin to flash.
- The user must keep track of the numbers displayed and determine if the sum of the numbers are divisible by 3(since the last time the button was pressed).
- If the user thinks the sum is divisible by 3, they must then press the push button again. A green LED will flash if the answer was correct, and a red LED will flash if the answer was incorrect.
- Once 60 seconds have elapsed, the game is over and the push button is disabled and the random numbers will no longer be displayed.
- The user's score will then flash every 2 seconds on the Display.

Design Structure

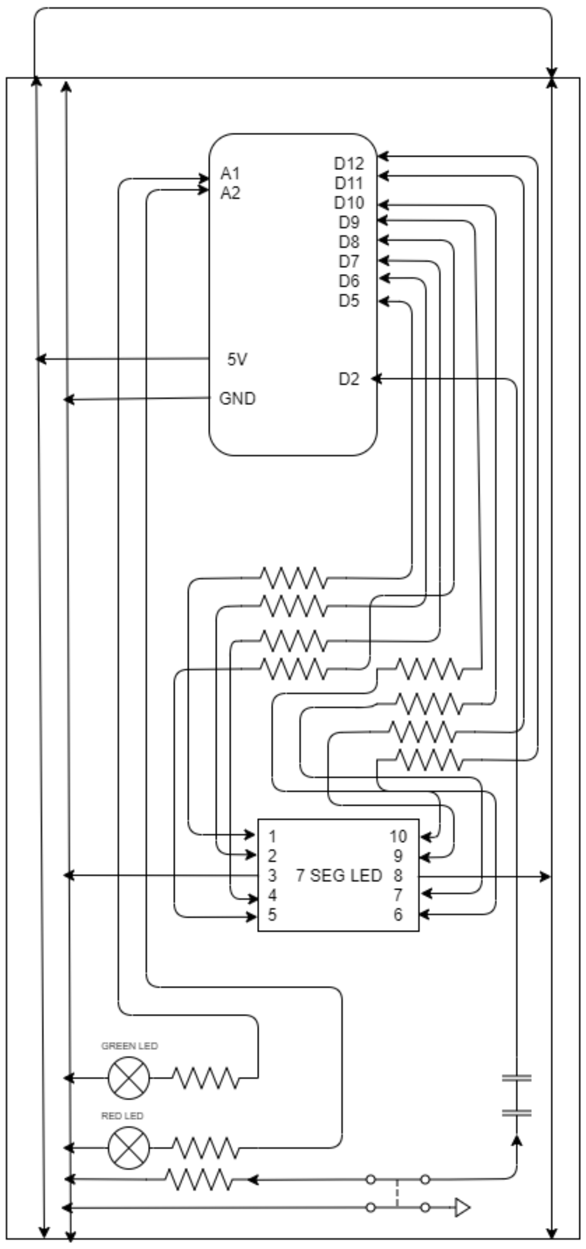
Both timer interrupts and external input interrupts were used in this program. An external interrupt was used to control the use of the push button. The functionality of the push button was split into 3 phases. The first phase only allowed the user to use the button to start the game, after which the button functionality was moved to phase 2. The button only allowed the user to check if the sum of numbers was divisible by 3. In the third phase the button was disabled to prevent the user from giving input. The push button was connected via a pull-up resistor which keeps the button sitting at a logical high. The interrupt is triggered by a falling edge on the button which detects that that button has been pressed and functions can be performed. The button was also connected to a capacitor to help filter out the noise caused by switch bounce.

The Timer 1 interrupt was used to change the random numbers every 3 seconds, count up to 60 seconds, and flash the user's score for 2 seconds. The Timer 1 output compare register A was used to track the 3 second interval by setting it to count to clear mode. Every 3 seconds the interrupt would trigger and call the function to generate and display the random number. A register was incremented every time the interrupt was triggered so that it could track when the program had been running for 60 seconds, which occurred when the register was incremented to 15. The Timer 1 output compare register B was used to track 2 second intervals by the same process when the user's score was needed to be displayed.

The pseudo random number generator made use of a seed formula. At the start of the program the 2 counters were set to infinitely increment. When an interrupt was triggered by the push button for the first time, the random number generator would take in the 2 values from these counters at that point in time and used them as input variables for the generator. This allows a different number sequence to be generated every time the program is started.

Timer 0 was used as an overflow counter. Every time the counter overflowed a counter register was incremented and compared to the value of 1 second. This was then used in a routine to create a small time delay to allow the green and red LEDs to flash.

Schematic



This schematic includes, a breadboard with the Arduino nano with the utilised pins labelled accordingly. The 7-segment LED. The Resistors that were used. 2 LEDs(green and red). As well as the push button switch. The 5V source and Ground were connected to the rails on the sides of the breadboard.

Citations and Acknowledgements

- https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard-2/
- https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard-3/
- https://pdf4pro.com/view/introduction-to-pic-programming-talking-14be0.html
- https://www.best-microcontroller-projects.com/switch-debounce.html
- https://stackoverflow.com/questions/55196692/mod-in-avr-assembly-divmodhi4
- https://www.avrfreaks.net/forum/asm-modulus-12-integer-constant
- http://www.rjhcoding.com/avr-asm-8bit-division.php
- https://www.arxterra.com/10-atmega328p-interrupts/
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fcomponents101.com%2Fdisplays%2F7-segment-display-pinout-working-datasheet&psig=AOvVaw26CzVolh8ZFoT-VsBNj35J&ust=1622093558311000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCKj32sTP5vACFQAAAAAdAAAAABAD
- https://sites.google.com/site/qeewiki/books/avr-guide/external-interrupts-on-the-atmega328
- https://www.instructables.com/LED-Blinking-With-a-Push-Button/

Documentation Used:

- AVR Instruction Manual ATMEGA328P
- AVR DATASHEET ATMEGA328P

Acknowledgements:

- Jenna Dunford – helped with external and general interrupts, advice on timers, advice on capacitors and switch bounce, and advice on modulus calculations.