COMP2121 Project: G12 Racer

Semester 1, 2014. Due: Friday of Week 13

Task

You are going to write the code to create a game that can be played on the AVR board called Speed Bracer. This will integrate many of the tasks you have been performing so far in your other labs into one program.

Synopsis

You are a driver in the *Universal Nova Speed Wing* Team competing in the local *Collosal Space Event* races. Yor goal is to race to the finish, avoiding obstacles along the way while collecting powerups to gain more fame for your team. Will you rise to glory before you destroy your three vehicles, or sink into obscurity?

Specification

The description of the game is as follows. Most of the play happens on the LCD, although other components are utilised.

Screen

A sample in-game view is as follows:

L	:	2		C	:	3	|	C		O					S
S	:			1	3	7	|						O		

The left 8 columns are a scoring area. There is room to display the current level (L), the number of cars remaining (C) and the current score (S). The 8th column has Pipe characters (|) or any other separator. The right 8 columns are the gameplay area. C is your car. O represents obstacles to avoid. S is a powerup that you might be able to collect.

Gameplay

There are two buttons. One resets the game into a default mode. The 2nd Allows you to start a new level, or skip to the next level if pressed again. At the start of a new game:

- The score area is updated to show Level 1, three cars and Score is 0.
- The rest of the screen is set up for level 1 (as stated below)

At the start of a new level:

- The level in the score area is updated
- C will be placed at the top row in the 9th column (where it is above).
- The rest of the play area will be blank.

The game is over when you run out of Cars. Your final score should be displayed when this occurs and the game should wait until it is reset to begin the next game. You should program the game to reach at least Level 9, with further levels possible if you wish.

At regular intervals, obstacles or powerups may appear on the right side of screen. They work as follows:

- At each time step of the current level, all current obstacles are shifted left one space
- To generate a new column in the rightmost column, the following rules are followed:
    - If there were no obstacles in the previous column:
        - 5% chance of powerup in top row
        - 5% chance of powerup in bottom row

        - 25% chance of Obstacle in top row
        - 25% chance of Obstacle in bottom row
        - 40% chance of nothing in either row
    - If there was an obstacle, the column then nothing appears in either row.
- If the Obstacles intercept the Car, then a Car is lost and the level restarts.
- Powerups disappear halfway through the screen. They disappear if they reach column 12.
- You get a number of points equal to the level when an obstacle leaves the play area to the left, so 1 point per Obstacle at level 1, 2 at level 2, etc.
- If a powerup and the Car are in the same spot, you get 10*level bonus points.
- The level ends after 30 seconds of game time pass. The game then begins the next level.
- Time steps for each level are as follows:

Level	Time step duration (s)
1	1
2	0.9
3	0.8
4	0.7
5	0.6
6	0.5
7	0.25
8	0.1
9 (and onwards)	Half previous level

The car is moved with the 2, 4, 6 and 8 buttons of the keypad, representing UP, LEFT, RIGHT and DOWN respectively. You may want to only allow movement at the time step interval, but full marks will require you to be able to move whenever a button is pushed. You cannot move UP in the top row or DOWN in the bottom row. If you move the Car into an obstacle or Powerup, then the appropriate thing happens as specified above.

Other inputs/outputs

In addition to the Keypad inputs specified above, the following are needed:

- Push buttons are used to reset the game or skip to the next level. Skipping to the next level keeps your score intact.
- The LED Bar Graph always displays your current score
- Whenever you crash into an obstacle, the Motor will run at 70rpm for 2 seconds to represent vibration feedback.

Additional powerups

To get full marks for the assignment, you will also need to implement one or more new powerups that fit the game. These should make sense in the context of the game. Marks will be awarded based on creativity and suitability for the game.

Background Information

This section provides some information about parts of the design that were not introduced in labs.

Random number generation

Some sample code with an implementation of a Linear Congruential Generator is provided to generate random numbers where needed. There are two functions you can call within this code:

InitRandom: Initialises the random number generator based on the current time provided by Timer 1. Thus, Timer 1 needs to be running for this to function.

GetRandom: Generates a random number and returns an 8-bit representation in r16.

Both functions require the stack to be initialized, of course.

Submission information

The following items should be submitted:

1. Source code via the subject webpage, and its printed copy. Your program should be well commented.
2. A printed copy of a User Manual. The User Manual describes how a user plays your game, including how to wire up the AVR board. Make sure you indicate which buttons perform what actions and how the LCD should be interpreted.
3. A printed copy of the design manual. The design manual describes how you designed the code that implements the game. It must contain the following components.
    a. *System control flow,* describing the control flow of the system at the module level using a diagram. This also shows how and when you service the inputs.
    b. *Data Structures,* describing the main data structures used in the system.
    c. *Algorithms,* describing how you maintain the state of the game.
    d. *Module specification*, describing the functions, the input and the output of each module.

Overall, a person with knowledge about the subject and AVR board should understand how your system is designed after reading user and design manuals.

Assessment of your project working on the AVR board will be performed in week 13.

Grading

The project is worth 15% of your course mark and will be marked under the following criteria (which is approximate for now):

- Implementation (65%):
    - Adherence to specification
    - Implementation of all input/output functionality
- Code Style (10%):
    - Easy to read
    - Well documented
- User Manual (10%)
    - Accurately describes set up of board
    - Correctly describes functionality
- Design Manual (15%)
    - Adherence to specification
    - Readability and Completeness