ECSE 444 Final Project Proposal

Interstellar Exploration Game

Group 1: Ao Shen, 260898863, Byron Chen, 260892558, Kevin Li, 260924898, Kua Chen, 260856888

I. Description of Project

A. Proposed Design

Based on the powerful functionality supported by STM32, we decided to design a classical video game that simulates the satellite dodges obstacles in space. This game is making players survive by getting scores. The STM32 board will be used as a steering wheel to control the satellite's movement in this game. As the survival time grows, the difficulty of the game will increase. For example, there may be multiple obstacles to dodge simultaneously. Players can also press the "emergency button" on the board to reset the game's difficulty.

B. Methodology

The whole game can be divided into four parts according to the functions supported by STM32: program logic, console display, input readings, and sound playback. We can read, process, and output the related data efficiently by applying the timers.

The logic of the whole program includes recording obstacles, recording of satellite position, collision detection, and score system. We can effectively describe the obstacle information by maintaining a one-dimensional array, where the index of the array indicates the position. The data located at this position will indicate the obstacle's existence and the distance of this obstacle to the satellite. A number ranging from 5 to 1 will be used to present the distance of the obstacles. Furthermore, the satellite's movement in the left and right directions can be expressed as the addition or subtraction to this number. The game screen will be presented using UART. The game screen will be animated smoothly by updating the array printed on the console at high speed. By applying the 3D accelerometer supported by STM32, we can read the board's movement and assign the thresholds for the satellite's movements. DAC will be used to create some sound effects of the game, and different actions and events will trigger different sounds. Furthermore, a soundtrack will be stored in QSPI Flash for playing as background music while the user is playing the game.

C. Tests

The project will be tested with two techniques: unit testing and exploratory testing. For each part we described in Methodology, we will do the unit testing, and after all the functions are done and put together, exploratory testing will be used.

The logical and UART parts will be combined for better testing since it is better to see the outputs when testing the logical part and the UART part is only about the connections. After that, the 3D accelerometer will be tested by running in an infinite loop while outputting the values it gets and collecting the threshold values for triggering the satellite's movements. After then, all the sound effects will be played, and the music will be played for a while for testing.

The primary testing technique will be exploratory testing. It is challenging to integrate unit testing perfectly into such an embedded system project. Therefore, exploratory testing will be used after the unit testing. Exploratory testing is the concurrent process of test design and test execution. As the name suggests, testers explore the application to identify potential edge cases whenever it is executable. The more the testers understand the players' actions, the more successful the tests will be.

II. Project Milestones

Here is the project milestones with the expected completion time:

1) Use the accelerometer of I2C to determine the angle of inclination of the board, which will be used to control the movement of the satellite and finish the testing (25^{th}, March).
2) Complete the logical part of the game and the connection to the UART with simple appearances (28^{th}, March).
3) Finish the testing for the logical part and the UART connection (30^{th}, March).
4) Use the DAC to generate some necessary sound effects, like the satellite crushing sound effect and finish the testing (1^{st}, April).
5) Use QSPI Flash to store background music and DAC to play the music and finish the testing (4^{th}, April).
6) Finish the exploratory testing with current work before the initial demonstration (6^{th}, April).
7) Use the functionality of UART to draw the satellite and obstacles based on the content of the arrays on the console. Furthermore, try to make the user interface more attractive (9^{th}, April).
8) Use the os threads to optimize the performance of this project if the time is allowed (11^{th}, April).
9) Finish the exploratory testing with current work before the Final demonstration (13^{th}, April).
10) Finish the report. (15^{th}, April).

III. Work Distribution

All the members will participate in all parts of the project. Ao is mainly responsible for the logical part of the game and the os threads implementation; Byron is mainly responsible for the report writing and the testing for each part; Kevin is mainly responsible for the I2C and QSPI implementation; Kua is mainly responsible for the UART connection and UI design and the exploratory testing.