
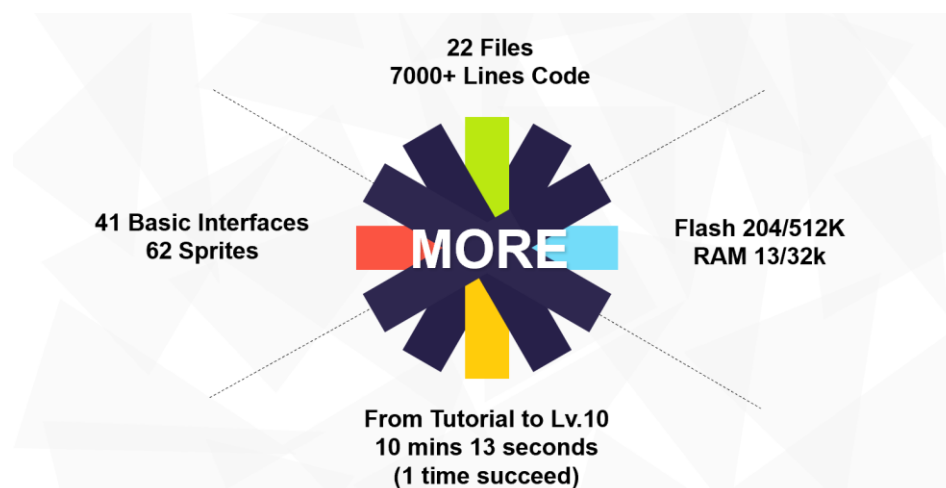
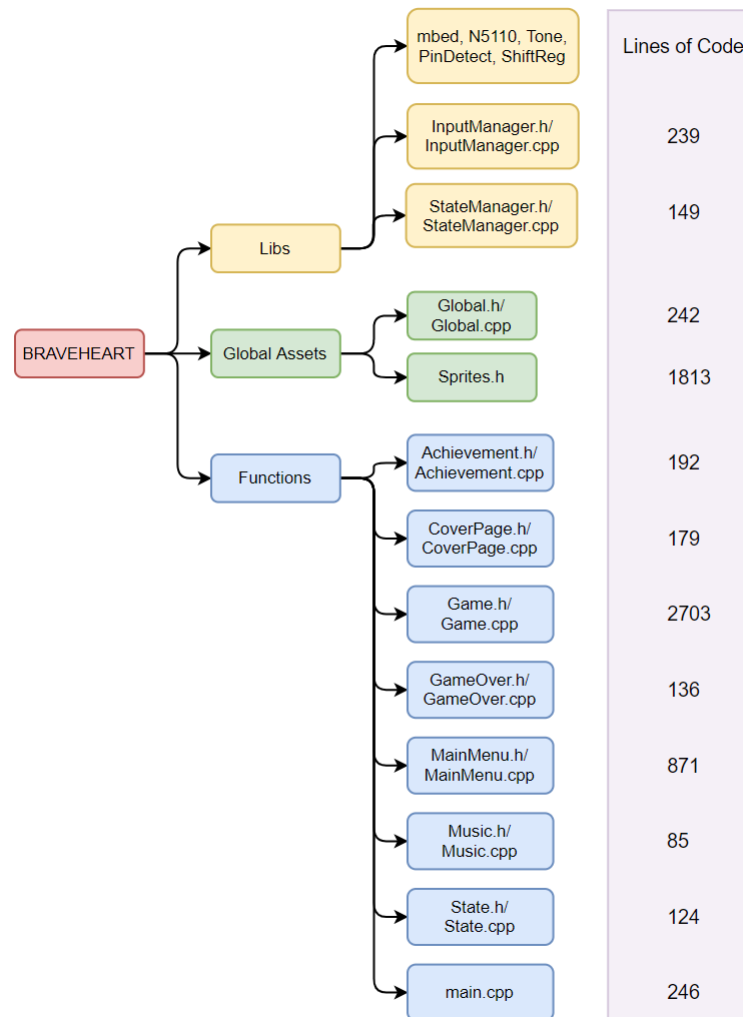


Project Reflection

Objectives

 1. Write, compile, run and de-bug complex C++ programs using appropriate software engineering techniques.

This task has been finished at least by the hard work I had done:



Several engineering techniques were used including but not limit to: Object-oriented programming, fsm-based state control mechanism, modular page-based separation design, rational use of interrupts and timers, use of light sensors, LCD, remote sensing, LEDs, and four buttons.

⚙️ 2. Implement advanced embedded software techniques on a microcontroller.

Since there were many advanced techniques used, I only name their functions but not introduce how they were implemented (weekly diary covered some of them).

- ♦ Random number generator based on analog read-in, random generation algorithm for game elements.
- ♦ Very complex map initialization process.
- ♦ Collision event handling (entity and bullets).
- ♦ AI algorithm for enemy movement.
- ♦ Design of endless levels.
- ♦ Achievement system based on local file storage.

🕒 3. Demonstrate initiative, independent learning, creativity, problem-solving, time management and project management skills.

Everything I did followed the development timetable (see task list in every diary). This game was completed designed and created by me, not borrowed from other finished games (fsm framework and music excluded). Every initiative of features was demonstrated in Project Highlights part.

During the development, many problems occurred:


First, due to the limited RAM space of the LPC1768 processor, I could not create a matrix of 84*48 size to save the randomly generated maps, so I had to save as many maps as possible in the form of const in ROM and then call them randomly. Inspired by the idea of hierarchy, the map is only used as the bottom layer, and by drawing enemies and gold coins etc. on the top layer, I can freely change their positions without storing them in a large array to render at once.

The next is the generation of true random numbers (instead of using the build-in rand() function) and collision body detection, these two implementations are explained in detail in the diary and will not be expanded here.

The most difficult problems to solve include how to design the tutorial interface. The ideal tutorial should show a text, then the player enters the game and gives feedback at the end. But the tutorial and the game are two event states of the same level, which can't call each other, and I can't copy the game flow with thousands of lines of code again because of the tutorial. My final thought was to design a global variable to add a judgment mechanism in all the different states of fsm, when the game is triggered by a tutorial, a specific level is shown and automatically returns to the main menu after it ends, and then to the next tutorial text screen without the user pressing any button.

Finally, a small question, because I limit the maximum frame rate of the game to 15fps (to prevent flicker caused by incomplete refresh due to too fast refresh), the background music does not play properly (the minimum period of note sounding and the chip each sleep

start period are not synchronized). My solution is very simple: the frame rate is still limited, but the processor no longer sleeps.

 4. Present their work in an effective manner.

Reflection

Through this mini-project, I fully understood and applied what I learned in class about event mechanisms, interrupting and sleeping, and the use of joysticks and screens. I have initial experience in how to design and follow a plan to implement a large project step by step. I also fully realized the importance of some engineer guidelines.

1. Have a solid foundation.

Without a solid professional foundation, it would be difficult for me to use data structures such as vector, map, file system, etc. correctly, use pointers properly and avoid memory leaks.

2. Have a rich imagination.

To design a game, the most important thing is to create something fun and it is worth to playing. The player can fire to the enemy, but how cannot a dragon shoot fire balls to a player? Based on this logic imagination, I enabled enemies to attack.

3. Good project management.

You must must must follow the timetable exactly. Otherwise, you have to relinquish some brilliant ideas because you have no time to implement. A wise and reasonable plan is vital, you cannot expect yourself finish a very complex part in one week.

 **PDF version:**