GameDroid: A GameBoy Emulator for Android

COMP 3004 – Deliverable 2

Team CreativeName:

Matthew Penny (100937331)
Hammad Asad (100854593)
Alan Wu (100942033)
Brendan Marko (100729104)

1.0 – Status report

The past few months have been very productive and our application is nearing its completion. The emulator core is able to properly run Nintendo GameBoy games' code, render graphics, and save/load game progress using both the games' native save functionality as well as our "save anywhere" feature (satisfying functional requirements 1, 2, and 3.1). Our user interface is able to display and organize the game library, can launch games, supports marking games as favorites, and allows searching (satisfying functional requirements 5.1, 5.4, and 5.5). The application is also light on memory usage, which satisfies non-functional requirement 2).

Currently, we are facing difficulties getting the emulator core to run at full speed (non-functional requirement 1). This is a high priority for us, since proper gameplay speed is a core part of the experience. We have been researching ways to improve rendering efficiency (the suspected bottleneck) and expect to solve this problem in the near future.

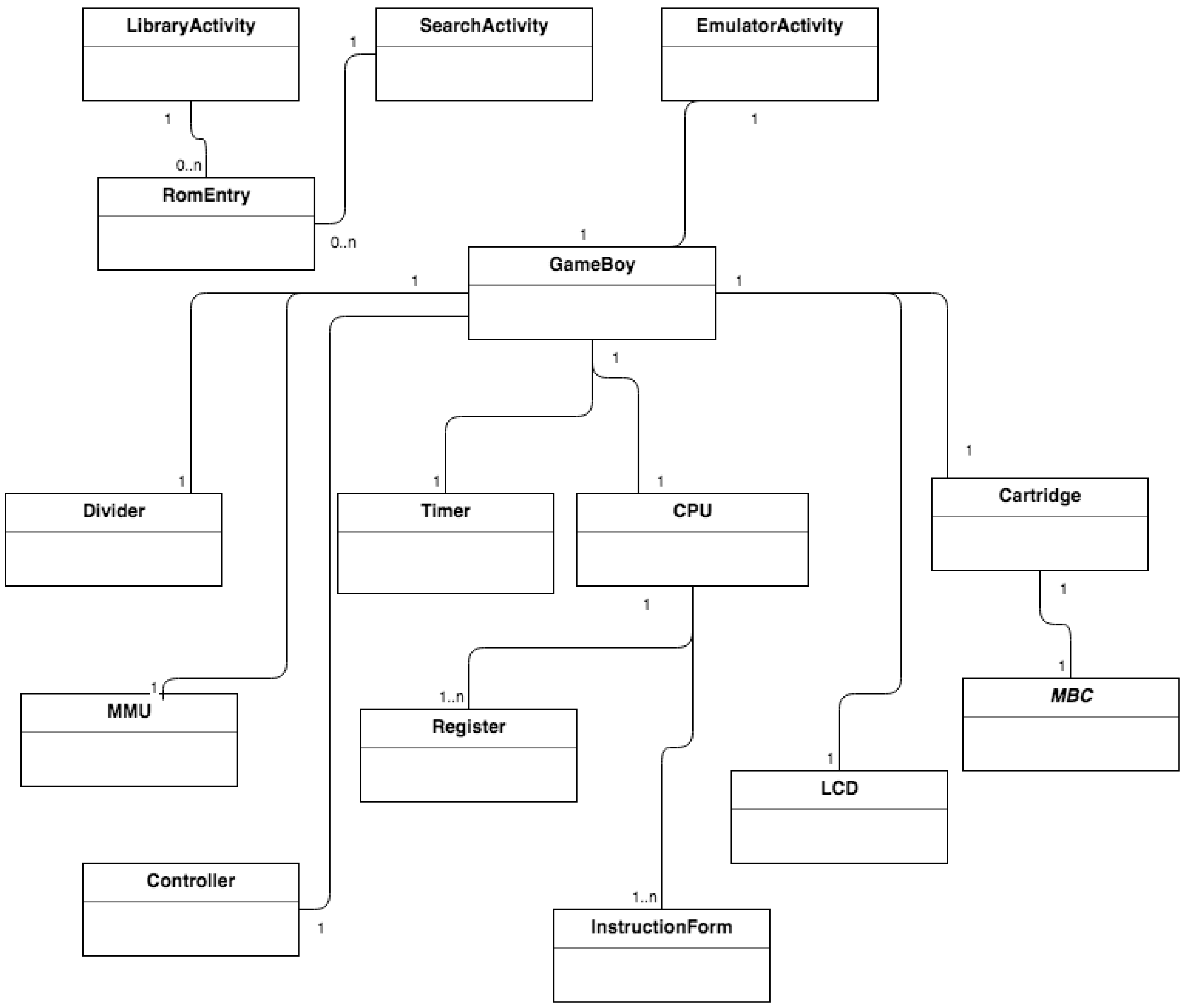
Our next month of development will consist of satisfying the remaining requirements for the application. This entails implementing the rewind feature (functional requirement 3.2), user-configurable settings (functional requirement 4), and game file management (functional requirements 5.2 and 5.3). We will also focus on getting emulated games to run at full speed (non-functional requirement 1) as mentioned above.

2.0 – Demo description

The user scenario that will be presented will be scenario 2.0 (loading, playing, and saving a Nintendo GameBoy game). The user will open the application and use the search feature to find a game to play (marking it as a favorite for easier searching in the future). They will start the game, which will load from where the user previously left off. After playing for a short while, the user will stop at a point where saving is not natively supported, demonstrate our "save anywhere" feature, and exit the game. Finally, the user will find the game again (this time by looking through their favorites) and load the previously saved state. This will demonstrate functional requirements 1.0 (emulation requirements), 2.0 (game ROM file requirements), 3.1 (save states), 5.1 (the game library), 5.4 (favoriting), and 5.5 (searching). Everything we will demonstrate is actually implemented. Nothing is simulated. However, some requirements are not currently satisfied (functional requirements 4, 5.2, and 5.3).

3.0 – Component diagram

Our application is divided into two distinct packages: the emulator core and the user interface. The emulator core implements the virtual GameBoy hardware (i.e., the Sharp LR35902 processor, proprietary LCD controller, gamepad input, extra cartridge hardware, etc.) while the user interface handles loading/saving games, rendering the view, library features, and feeding user input to the emulator. The GameBoy class is the main entry point to the emulator core. The EmulatorActivity class keeps an instance of it for connecting it to the user interface. Other UI classes (LibraryActivtiy, SearchActivity, and RomEntry) handle library navigation.


LibraryActivity
SearchActivity
EmulatorActivity
RomEntry
GameBoy
Divider
Timer
CPU
Cartridge
MMU
Register
MBC
LCD
Controller
InstructionForm
1
0..n
1..n