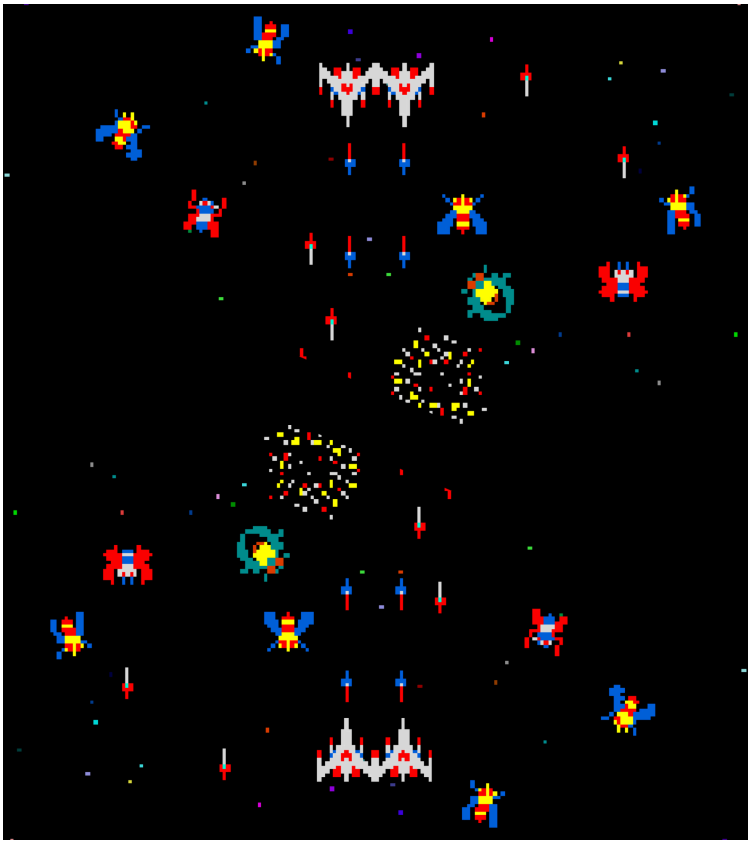
Airplane Battle Game Design Proposal

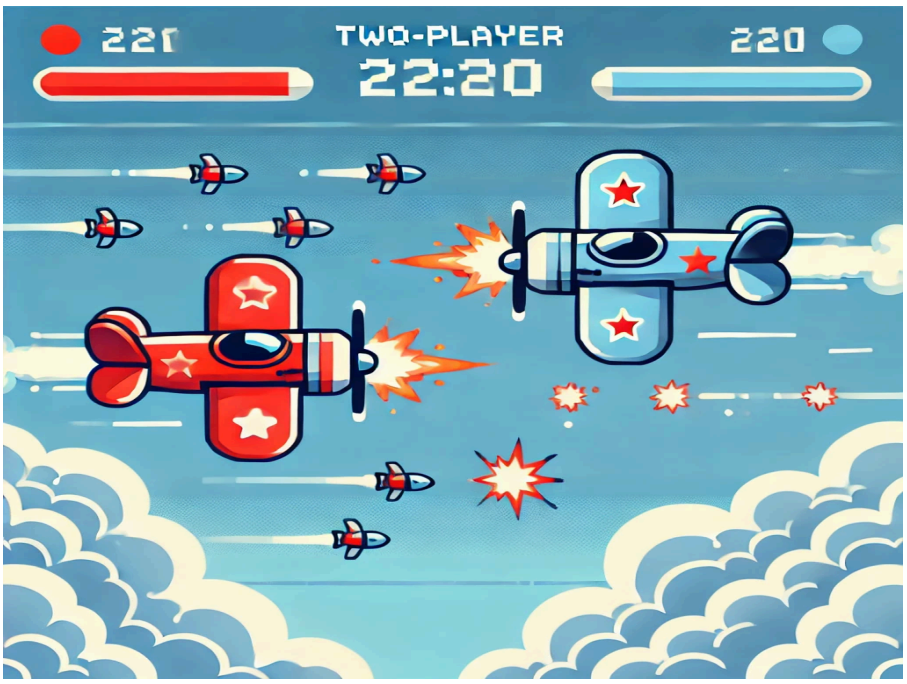
Mingzhi Li ml5160, Noah Hartzfeld (nah2178), Hiroki Endo(he2305), Jingyi Lai jl6932 Zhengtao Hu (zh2651)

Project Overview

The core of this project is a two-player airplane battle game, leveraging FPGA capabilities for hardware acceleration and VGA output for display. Players control their aircraft from opposite sides of the screen and engage in combat by firing projectiles toward each other while also avoiding and firing at enemies throughout the screen. Colision with enemies or opposite player's projectiles causes a life to be lost. The first player to run out of lives loses. The system will display the game interface on a VGA monitor and respond to user inputs through an external joystick.

The hardware accelerator will take a dictionary of coordinates and sprite designs and be responsible for creating, generating, and displaying the sprites on the screen at these coordinates. The software will handle the game logic, including receiving input from users, handling collisions, and movement speed. It will also generate, update, and pass sprite coordinates to the hardware.





Hardware Design

- **Framerate**: This will be 60 frames per second, which is common for arcade-style games.
- **VGA display**: This display will serve as the users' interface to monitor game action in real time, including aircraft positions, projectile trajectories, score, and game status.
- **Frame Buffer**: We will likely need to store bitmaps in our frame buffer. For less tearing and smoother transitions, we will try to use double-buffering.
- **USB Host Controller for Joystick:** Capture, de-bounce, and process external Joystick signals via FPGA.

Software Design

- **Game logic**: Implementation of the airplane battle game algorithms, including:
  - Aircraft movement and boundary limits
  - Synchronized projectile generation and movement physics
  - Collision detection between projectiles and aircraft
  - Power-up generation, acquisition, and effects
  - Win/loss condition detection and score management
- **Additional Features:**
  - Game State Management: Implement a state machine for initialization, main menu, gameplay, pause, and game over.
  - UI and Menus: Develop a user interface featuring a main menu, pause menu, scoreboard, and prompt notifications.

    - FPGA Communication Driver: Develop a driver that facilitates reliable data exchange between the software and the FPGA.

Hardware-Software Interface

- Share the updated sprite location dictionary with the hardware once per frame
- Syncronize the software so it supplies its data at exactly 60 frames per second.
- Because the software will be busy with the general game mechanics, it would likely benefit from using a DMA controller to perform actual data transfers. Because of this, a command-based style would most suit our design. In this case, the software will send a sequence of commands to the hardware through a single command port.

Development Tools

- **Hardware:**
    - DE1-SoC Board
    - Joystick/controller
    - VGA monitor
- **Software:**
    - C language
- **Development Environment**
    - Quartus for FPGA development
    - VS code for code editing and debugging

Milestones

Stage 1: Design the overall structure of the Airplane Battle game, then develop the core game logic (movement, firing mechanics) and basic display management.

Stage 2: Implement the VGA display functionality and handle storage/creation of sprites.

Stage 4: Establish the two-player simultaneous control system and implement the controller functionality.

Stage 4: Verify the the previous stages, then develop enhanced features such as power-ups and special weapons to increase gameplay variety and entertainment value.

Stage 5: Final testing, optimization, and polish of the game mechanics and visual elements.