Group-31 Final Abstract

Snake Game on DTEK-V Board

Abdulaziz and Jacob

October 2025

By Jacob Polanco Johansson (030726-7294) and Abdulaziz Hassan Farah (980627-0071)

1 Objective and Requirements

The goal of this project is to design and implement a simple Snake game on the DTEK-V embedded platform. The player controls the movement of a snake that collects apples to grow longer. The game ends when the snake collides with a wall or with its own body.

The main requirements are:

- The program must run on the DTEK-V board using the RISC-V toolchain (C/ASM).
- The graphics must be displayed on the VGA output.
- The snake must be able to move in four directions.
- Apples appear randomly on the screen and increase the snake's length when eaten.
- Collisions with walls or the snake itself must be detected.
- The current score must be shown on both the VGA display and the 7-segment display.
- The game can be restarted using a push button.

2 Solution

The solution is fully implemented in C without using interrupts. Instead, the game uses **polling** to read the input switches. SW0 and SW1 are used to rotate the snake right or left. The VGA interface is responsible for drawing the snake, apples, and text messages such as *GAME OVER*. The 7-segment displays show the current score in real time. The main loop updates the game state by repeatedly reading inputs, moving the snake, detecting collisions, and refreshing the VGA screen.

The logic, world map, and random apple generation are implemented in C using arrays to represent the game field. The VGA drawing functions use direct memory access to display pixels on the screen.

3 Verification

The game was verified by manual play-testing. Test scenarios included:

- Normal gameplay with multiple apple pickups.
- Collision with walls and self-collision to trigger *Game Over*.
- Checking that the score increases correctly when eating apples.
- Restarting the game with the push button.

Speed and timing were adjusted using software delay loops to maintain smooth movement. The game performed reliably during all tests.

4 Contributions

Jacob focused mainly on **input handling**, **game logic**, and **apple generation**, while Abdulaziz worked on **VGA graphics**, **text rendering**, and **score display**. Both contributed to debugging and integration.

5 Reflections

During development we learned how to combine low-level hardware control with game logic in C. The biggest challenges were managing VGA memory efficiently and synchronizing snake movement with screen updates. We also attempted to implement **interrupts** for input handling and timing, but we observed unstable and unexpected behavior. Because of this, we decided to continue using polling instead, which made the game more stable and easier to debug.

Through this project we gained a better understanding of embedded systems programming, memory-mapped I/O, and how simple games can be implemented directly on hardware without an operating system.