GR8: CHIP-8 Emulator

GR8 is a CHIP-8 emulator implemented in Rust using the Macroquad game engine. It allows users to run classic CHIP-8 games and applications from the 1970s and 1980s in a modern graphical environment.

Overview

CHIP-8 was a simple interpreted programming language designed for early microcomputers, and this emulator recreates that environment, enabling users to experience retro games like Pong, Space Invaders, and Tetris.

The emulator implements the full CHIP-8 instruction set, handling the CPU emulation, memory management, and display rendering. It loads ROM files containing CHIP-8 programs and executes them instruction by instruction, producing visual output through Macroquad's rendering capabilities.

Project Structure

Core Systems

1. **Emulator Core** - Simulates the CHIP-8 hardware, including CPU, memory, and registers
2. **Opcode Handler** - Decodes and executes CHIP-8 instructions
3. **Graphics Interface** - Renders the CHIP-8 display using Macroquad

Main Files and Directories

- `src/main.rs` - Application entry point, sets up Macroquad and the main game loop
- `src/emulator/`
  - `mod.rs` - Defines the emulator module structure
  - `emulator.rs` - Contains the core `Emulator` struct that implements the CHIP-8 VM
  - `opcode.rs` - Defines the `Opcode` enum and ROM decoding functionality
- `src/examples/chip8-roms/` - Git submodule containing example CHIP-8 ROMs

Installation

Prerequisites

- Rust and Cargo (latest stable version)
- Dependencies for Macroquad (refer to Macroquad documentation for platform-specific requirements)

Building

```
cargo build
```

Usage

Running the Emulator

```
cargo run
```

Loading ROMs

The emulator can load ROM files from the `src/examples/chip8-roms` directory:

```
// Example code for loading a ROM
let mut emulator = Emulator::new();
emulator.load_rom("./src/examples/chip8-roms/games/Pong (1 player).ch8").unwrap();
```

Implementation Details

Emulator Core

The core `Emulator` struct implements the CHIP-8 virtual machine with these components:

- 4KB memory array
- 16 8-bit registers (V0-VF)
- Index register (I) for memory addressing
- 64x32 pixel display
- Delay and sound timers
- 16-key input array
- Stack for subroutine calls
- Program counter (PC)

Instruction Set

The emulator supports all standard CHIP-8 instructions, including:

- Display clearing and manipulation
- Flow control (jumps, calls, returns)
- Conditional operations
- Register operations
- Timer management
- Graphics drawing
- Input handling

ROM Execution

GR8 implements a core fetch-decode-execute loop pattern for ROM execution. Rather than decoding all instructions at once, the emulator:

1. Loads the ROM into memory
2. Processes instructions one at a time during execution
3. Updates the display and timers accordingly

This approach is necessary because CHIP-8 ROMs mix both code and data sections, making it impossible to decode an entire ROM ahead of time.

Available ROMs

A collection of CHIP-8 ROMs is included in the `src/examples/chip8-roms/` directory, organized into categories:

- Games (Pong, Space Invaders, Tetris, etc.)
- Demos and animations
- Test programs
- High-resolution variants

Testing

Run the tests with:

```
cargo test
```

License

This project is open source. Please refer to the repository for license information.

Acknowledgments

- The CHIP-8 ROMs are from Revival Studios' collection
- Built with Macroquad