Project FINALBYTE

Name: FINALBYTE **Version:** 1.2 **Subtitle:** Unified audio-visual expansion system for 8-bit platforms (Atari 800XL, Commodore 64, ZX Spectrum)

□ Project Goal:

FINALBYTE is a cross-platform enhancement system for classic 8-bit computers, providing modern audio and visual capabilities while preserving full compatibility with original hardware and software.

FINALBYTE delivers:

- □ Wavetable and sampled audio with FX (reverb, echo, pitch-shift)
- □ Graphical overlay with alpha channel, color palettes and sprites
- ⚙ Unified address/control interface across platforms

□ Supported Platforms:

- Atari 800XL / 130XE
- Commodore 64
- ZX Spectrum (128k / AY)

□ Architecture:

▶ FINALBYTE Module

- External hardware (ESP32 / RPi Pico / STM32 / FPGA / RP2040)
- Intercepts bus or I/O instructions from host system
- Includes:
  - □ **Sound engine**: multisample, WAV, FX, stereo output
  - □ **Overlay engine**: 320x200px, 16/32 color, 4-bit alpha
  - □ **Memory bank**: SD card / SPI flash for samples and graphics

▶ Communication with Host System

- Passive sniffer (listens to POKEY/SID/Beeper instructions)
- Active control via:
  - `$D700+` (Atari)
  - `$DF00+` (C64)
  - `OUT (n),a` with prefix (ZX)

□ □ FINALBYTE Enhancements:

□ 1. Video Overlay (Amiga-style)

- Full overlay system:
  - Captures original video signal (ZX/C64/Atari)
  - Syncs to VSYNC
  - Adds FINALBYTE HUD, cursors, effects
- Chip variant: RP2040 / STM32 / FPGA
- Modes:
  - Overlay HUD – additional layer only
  - Overlay Full – full-screen overlay (cutscenes, maps)
  - Overlay Off – passthrough only

□ 2. USB Keyboard and Mouse

- Direct USB HID connection via USB host (ESP32-S3)
- Mouse moves overlay cursor independently from host CPU
- HID support for mouse movement/clicks + key scanning for GUI/adventure/strategy games

□ 3. "Lite" Tile System (16×16 tiles, 32×32 maps)

- Lightweight background engine:
  - Tile size: 16×16 px
  - Visible grid: 20×14 tiles
  - 2 layers (parallax + HUD)
- Easy entry point for new devs
- Smooth scrolling without DMA

□ 4. FPS Optimization + Cinematic Modes

- Recommended: 30 fps overlay refresh, 24 fps animation
- Host CPU can run as low as 8 fps without visible drop
- FINALBYTE cinematic loop™ – optional filmic mode

□ Sound Engine:

- 8/12/16-bit samples
- Wavetable playback (velocity, pitch, offset, loop)
- Effects: reverb, delay, filter, saturation
- Command set: `NOTE_ON`, `PLAY_SAMPLE`, `SET_VOLUME`, `FX_ON`, etc.

□ Overlay Engine:

- 320x200px, 4-bit alpha, 16/32 color palettes ("ST" and "Amiga" modes)
- Sprite support:
  - Positioning, Z-order, animation

    - HUD, icons, labels
- Transparency, blinking, fade
- Controlled via mapped registers or ports

□ Collision Detection (in external system):

- Hundreds of sprite collisions handled in real-time
- Host system reads back collision results via buffer:

□ Collision feedback:

1. **Memory-mapped collision buffer** (recommended):

    - Example: `$D780–$D79F` (Atari), `$DF80–$DFA0` (C64)
    - Entries in pairs: `SPRITE_ID_A`, `SPRITE_ID_B`, ends with zero
    - Host polls buffer and reacts (hit, overlap, pickup)

2. **Optional: Bitmask matrix**

    - 256×256 bit matrix stored in FINALBYTE module
    - Any pair can be queried

□ Compatibility & Philosophy:

- □ No modification to original hardware/software
- ✉ All commands are non-invasive or in unused I/O ranges
- □ Games remain playable on stock machines in "Lite mode"
- □ Full version activates automatically if module is detected (e.g. handshake on `$D7FF`)

□ Use Cases:

- Cross-platform remakes (Bruce Lee, Saboteur, The Last Ninja)
- New ambient/adventure games with music and voice
- Real-time demos (beat sync, layered effects)
- VJ/live setups with retro computers as controllers

□ Status:

- ✏ Architecture designed
- □ Address and protocol documentation in progress
- ⚖ Planned open-source release (HW + FW + dev libs)

FINALBYTE = One spirit, three legends.

Together, we bring 8-bit creativity into a new golden age.