Pop'n Engine: An FPGA Rhythm Game

Final Presentation

CSEE 4840 Embedded System Design

Presenter:

Vince-Arvin Magno (vm2787)

System Block Diagram

HPS - Linux userspace

- popn_main.c (the game loop)
- load_chart() parses .chart files
- 100 Hz game loop: spawn / advance / input
- Menu / Playing / Results state machine
- /dev/input/event0 ← Pop'n controller
- fork() + execl() → audio_play
- /dev/mem mmap → 0xC0000000

AXI

BRIDGE

HW

0xC0000000

16 × 32-bit regs

FPGA - Cyclone V fabric

- popn_engine.sv (custom peripheral)
- Avalon-MM slave: 16 × 32-bit registers
- VGA timing gen (640×480 @ 60 Hz)
- Sprite render functions (notes, font)
- 40-deep combinational pixel mux
- → VGA monitor (R,G,B,sync)
- audio_0 IP → WM8731 codec

USB Pop'n controller → HPS | FPGA → VGA monitor + WM8731 audio codec

Hardware-Software Interface: Register Map

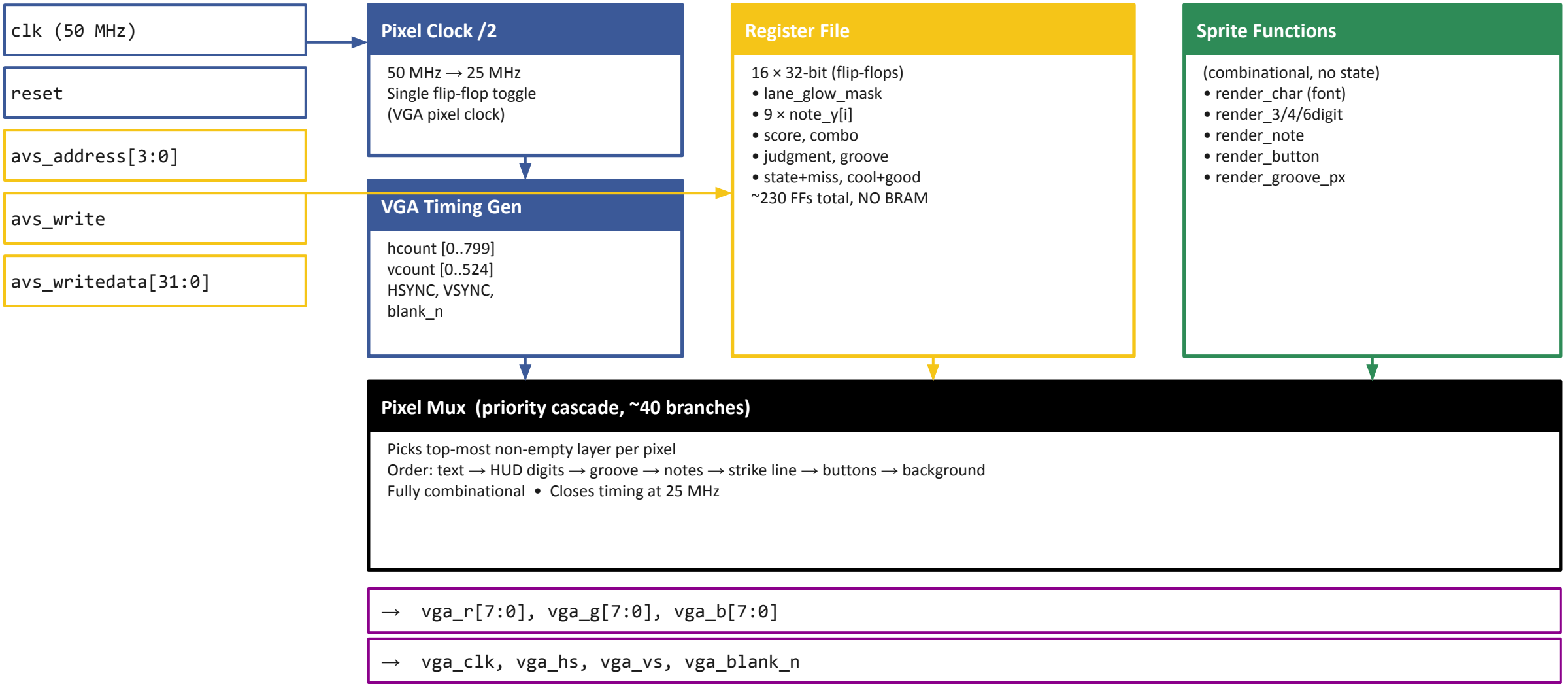
Base address: 0xC0000000 • 16 × 32-bit registers • 64 bytes total • Write-only (no readback)

Offset	Register	Width	Purpose
0x00	lane_glow_mask	9 bits	Which buttons are currently held, lights up bottom button row
0x04-0x24	note_y[0..8]	10 bits	Y position of head note in each lane (600 = off-screen)
0x28	score_bcd	24 bits	6-digit BCD score (top-left HUD)
0x2C	combo_bcd	16 bits	4-digit BCD combo counter (top-right HUD)
0x30	judgment	2 bits	0=none / 1=COOL / 2=GOOD / 3=MISS
0x34	groove	8 bits	Groove gauge fill level (0..255), starts at 128
0x38	state + miss	32 bits	PACKED: state[1:0] = Menu/Play/Results, miss_bcd[31:16]
0x3C	cool + good	32 bits	PACKED: cool_bcd[31:16], good_bcd[15:0]

ORGANIZING PRINCIPLE

• Things that update independently → own register • Things that update together → packed • BCD encoding → hardware extracts digits directly • Write-only → C is source of truth, no readback needed

Inside popn_engine: Hardware Block Diagram



0 block RAM

~230 flip-flops

25 MHz pixel clk

1 flat module (no submodules)

Software: popn_main.c Game Loop

Three states mirror the FPGA's game_state register (0x38, bits[1:0])

FALLING NOTE MATH

$$y = (JUDGE\_Y - 16) - (dt \times ((JUDGE\_Y - 16) - SPAWN\_Y)) / NOTE\_TRAVEL\_MS$$

JUDGE_Y=440 • SPAWN_Y=0 • NOTE_TRAVEL_MS=2200 • '-16' makes white stripe land on strike line

HIT WINDOWS

COOL	GOOD	MISS
±60 ms	±120 ms	+150 ms

CHART PARSING

Plain text format:
"timestamp_ms lane"
Offline gen via librosa
(tempo + onset + spectral
centroid ⟶ lane bin)

INPUT HANDLING

/dev/input/event0
EVIOCGRAB ioctl
EV_KEY codes 304-312
Non-blocking read
@ 100 Hz

AUDIO

fork() + execl()
⟶ /root/audio_play
starts at g_start_ms
48 kHz mono WAV
SIGTERM at end

Live Demo

WHAT WORKS

60 Hz
stable VGA frame rate

0 blocks
block RAM used by popn_engine

16 regs
memory-mapped registers

9 lanes
Pop'n controller buttons mapped

60 ms
COOL hit window, sub-frame latency