

Liquid-Physics-Simulator



Aidan Dodge (acd2243)

Da Won Kim (dk3311)

Daanish Khan (dk3472)

Project Overview



[Ref] <https://www.youtube.com/watch?v=s5Q8JY3kEb4>

- Real-time 2D water simulation running on FPGA hardware, displayed on a VGA monitor
- With an external USB mouse I/O, user can draw walls and spawn water by the calculation of liquid flows done on FPGA

High-Level System Design

HPS (ARM Cortex-A9)

- Read USB mouse → grid coords
- Load initial cell map at boot
- Write Avalon-MM registers
- Configure auto-run / frame-lock
- Brush commands (add/erase/wall)

Avalon
MM Bus

FPGA (Cyclone V)

Global Controller FSM Sequences evaluate → commit ticks, arbitrates Avalon

Physics Engine (PE) One cell/clock pipeline, 18-bit Q2.16 fixed-point

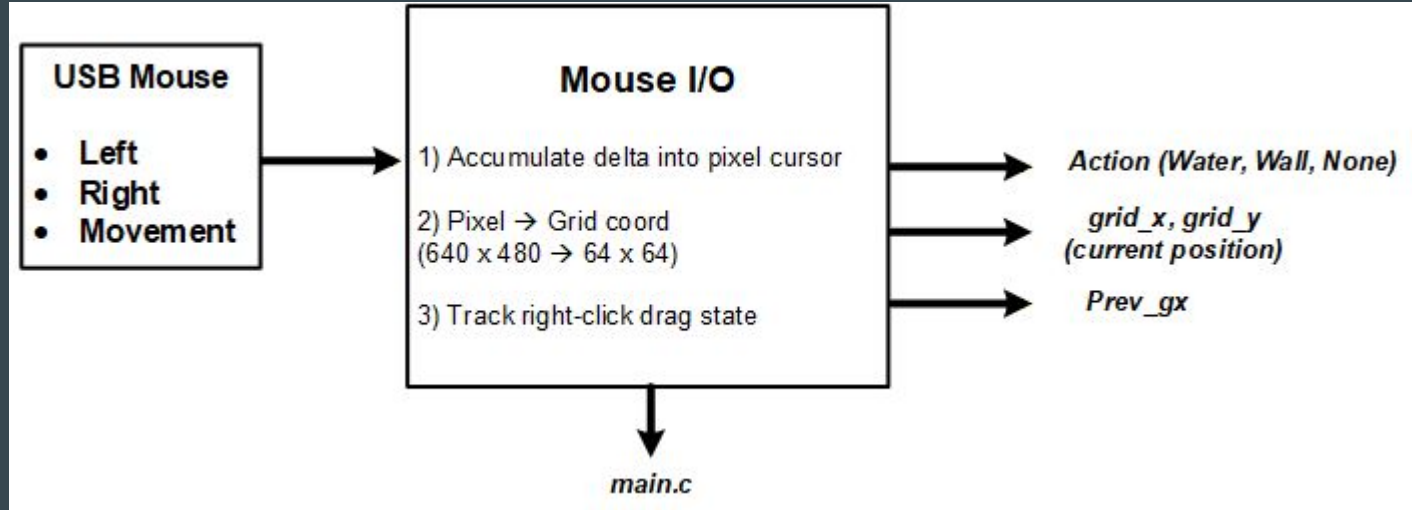
Cell BRAM 4096 × 32-bit words (64×64 grid)

Diffs BRAM Scratch buffer for per-tick flow deltas

VGA Renderer 640×480 @ 60 Hz, 7×7 px per cell

Mouse I/O

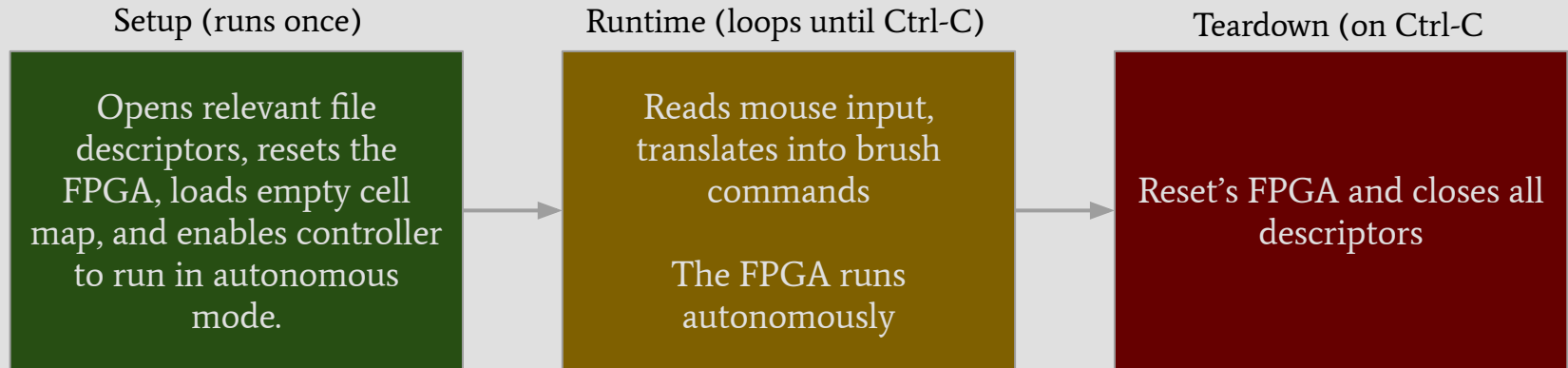
- Translates raw mouse packets into grid coordinates and user actions that the main loop sends to the FPGA



Register Mapping Overview (for our Hardware-Paired C)

Offset	Register	R/W	Field used by main.c	Used in
0x0000	CTRL_REG	W	[1] RESET [2] LOAD_MAP [4] BRUSH_APPLY	reset_fpga() load_empty_map() brush_apply()
0x0004	STATUS_REG	R	[0] BUSY [2] MAP_READY	reset_fpga() load_empty_map()
0x000C	MOUSE_POS_REG	W	[15:0] grid_x [31:16] grid_y	brush_apply()
0x0010	BRUSH_CFG_REG	W	[1:0] tool [15:8] radius = 0 [31:16] liquid	brush_apply()
0x0014	STEP_CFG_REG	R/W	[0] AUTO_RUN = 1 [1] FRAME_LOCK = 1 [15:8] STEPS_PER_FRAME = 1	configure_runtime()
0x0040	VGA_CTRL_REG	R/W	[0] ENABLE = 1	configure_runtime()
0x1000 - 0x4FFC	GRID_MEM	R/W	4096 × 32-bit cell words cell (x,y) at 0x1000 + 4·(y·64 + x)	load_empty_map()

Hardware-Paired C



Hardware-Paired C (Setup)

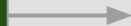
open_devices()

Opens file descriptors for /dev/water_sim (kernel driver) and /dev/input/mice (USB mouse)



reset_fpga()

Pulses CTRL_RESET, then polls STATUS_BUSY until the controller clears state and signals idle (3 way handshake)



configure_runtime()

Set AUTO_RUN, FRAME_LOCK(wait for Vsync), and number of STEPS_PER_FRAME. While finally also enabling the VGA.

From here on out, the FPGA ticks autonomously



load_empty_map()

Write all 4096 cells as blank, pulse CTRL_LOAD_MAP, then wait for STATUS_MAP_READY

Hardware-Paired C (Runtime)

Each event is an ioctl that writes cells into GRID_MEM.

Walls flip CellType → SOLID; water and erase update the Liquid field.

mouse_poll()
Block on /dev/input/mice until a button event occurs, or a SIGINT breaks the loop

handle_mouse_event()
Dispatched by the button with events like:
Add water, draw wall (through interpolation) and erase

Loop until Ctrl-C

Controller

IDLE

Waiting for:

- CTRL_STEP pulse
- or AUTO_RUN + VSYNC

EVALUATE

Walks all 4096 cells.

For each cell: reads self + 4 neighbours from Cell BRAM, runs PE, writes diffs to Diff BRAM.
~4096 clocks.

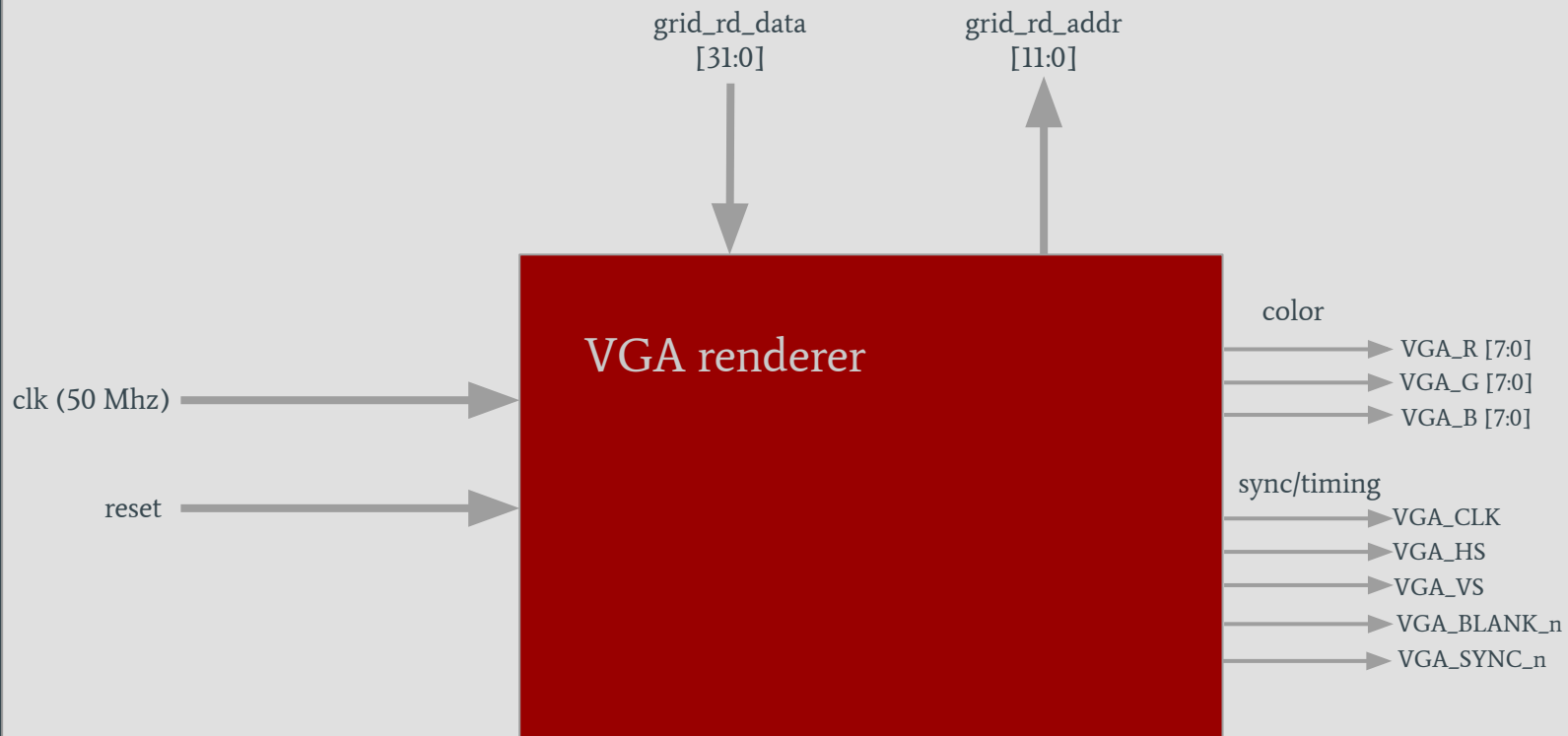
COMMIT

Walks all 4096 cells again.

Adds diffs to Cell BRAM, clamps liquid ≥ 0 , updates settled flags.
~4096 clocks.

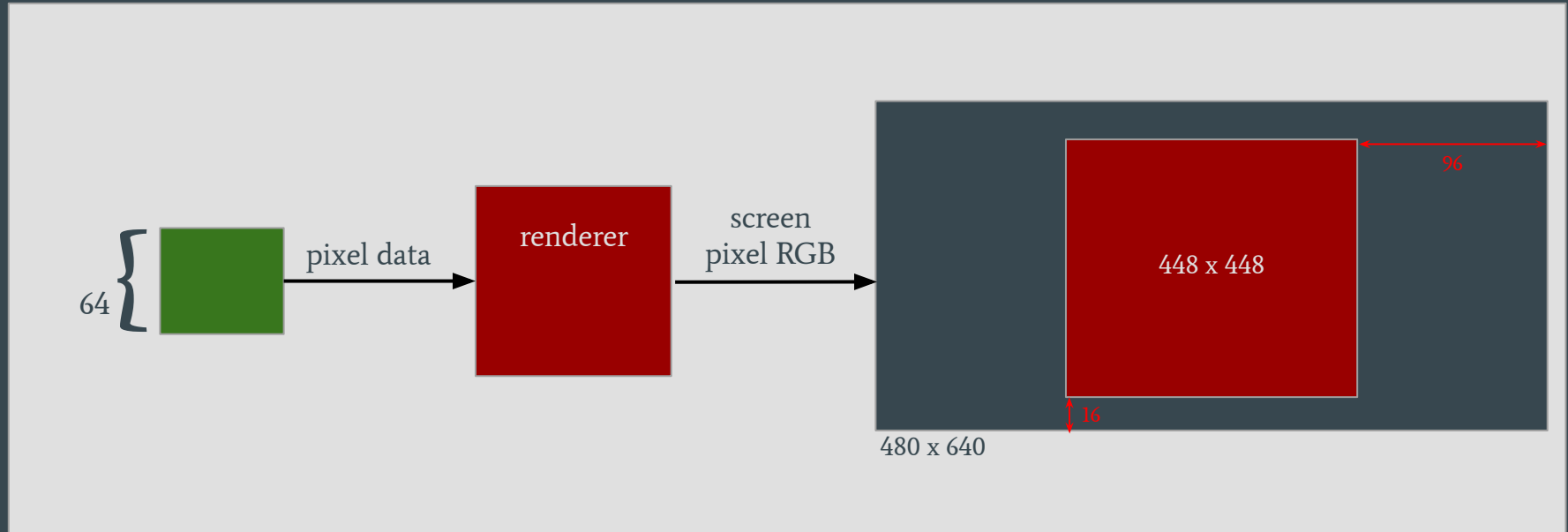
← back to IDLE (tick complete; STATUS_DONE pulse; BUSY cleared)

VGA implementation



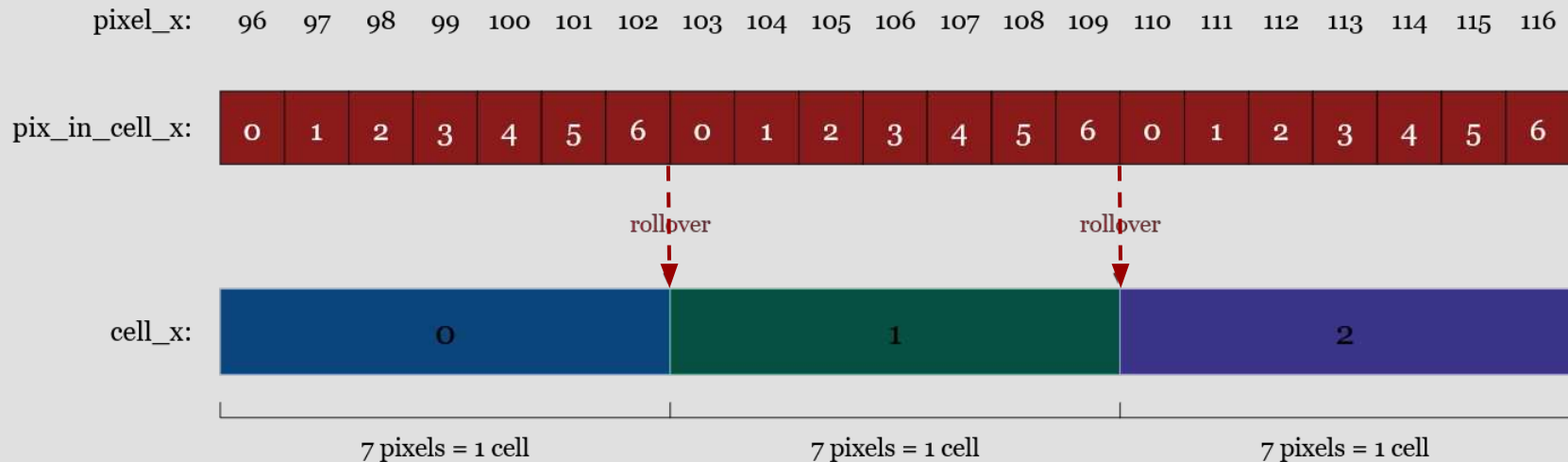
VGA Implementation

Decision 1: Mapping indices onto a 7x7 pixel block.

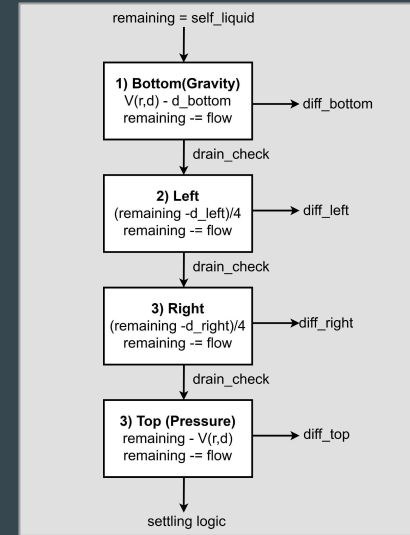
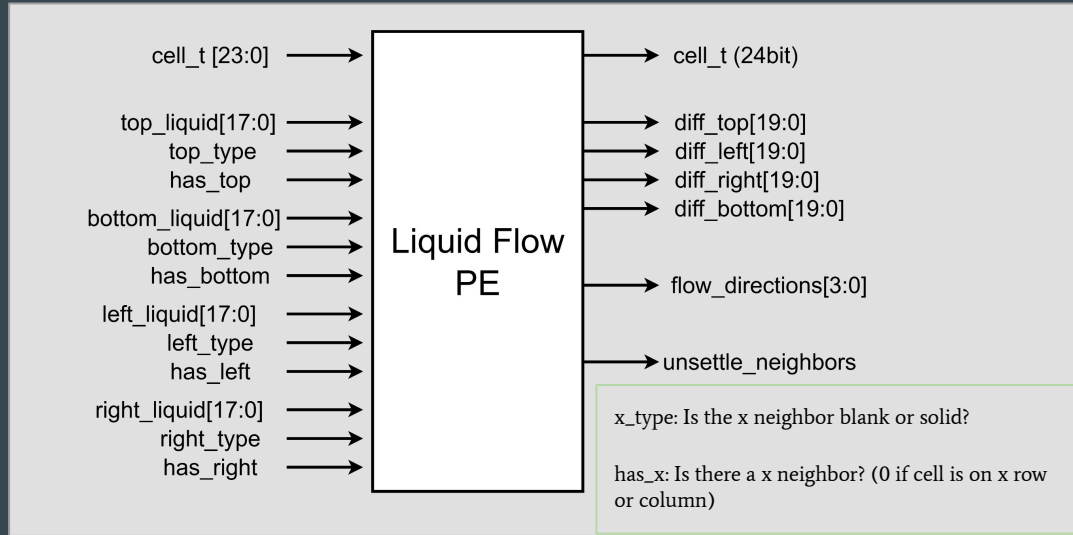


VGA Implementation

Decision 2: Pixel Mapping Via Counter



Liquid Flow Processing Element (Particle Accelerator)



- Given a cell and its neighbors, processing element compute how much liquid flows down (gravity), left, right (spreading), and up (pressure), and output the changes
- Purely Combinational, Q2.16 fixed-point (18 bit unsigned liquid, 20 bit signed diffs)
- Output diffs only - commit pass applies them later

Timing Analysis

Clock Constraint

50 MHz

20 ns period

Required by 640x480 @ 60 Hz VGA

Achieved Fmax

23.53 MHz

Slow 1100mV 85°C model

clock_50_1: from soc_system_sta.rpt

Worst Setup Slack

-22.5 ns

clock_50_1 (TNS -4419 ns)

Significant violation: design not timing-closed

Worst Hold Slack

-0.784 ns

clock_50_1

Hold also violated

Thank you!

