WebRender Presentation

A Simple Graphics Accelerator for Rounded Rectangles

Alex Gajewski and Allison Ghuman
Motivation

- Rendering the web is slow
- Hardware is fast
- Goal: build a hardware accelerator to speed up common tasks in web rendering

Range: 1.44 s – 5.82 s

- 168 ms  Loading
- 594 ms  Scripting
- 2281 ms Rendering
- 64 ms  Painting
- 662 ms  System
- 607 ms  Idle
- 4376 ms  Total
Approach: rounded rectangles

- Almost every object on the web falls into two categories:
  - Bitmaps
    - (really fonts these days are vectors, but at least they used to be bitmaps)
  - Rounded rectangles

- Goal for this project:
  - Build an accelerator that can layer rounded rectangles on top of each other as quickly as possible
Implementation

- Initially, we thought we would need a dual-framebuffer design:
  - Read from one framebuffer
  - Write to the other
  - Switch
- Actually, you can render pixels in real time, in a single cycle
  - Use block RAM to read an entire row’s displaylist in one cycle
  - Purely combinatorial logic can compare the current column to the start and end column properties of the objects in the displaylist, and choose colors appropriately

```cpp
always_comb begin
    {VGA_R, VGA_G, VGA_B} = {line_r[7], line_g[7], line_b[7]};
    {VGA_R, VGA_G, VGA_B} = {line_r[6], line_g[6], line_b[6]};
    {VGA_R, VGA_G, VGA_B} = {line_r[5], line_g[5], line_b[5]};
    {VGA_R, VGA_G, VGA_B} = {line_r[4], line_g[4], line_b[4]};
    {VGA_R, VGA_G, VGA_B} = {line_r[3], line_g[3], line_b[3]};
    {VGA_R, VGA_G, VGA_B} = {line_r[2], line_g[2], line_b[2]};
    {VGA_R, VGA_G, VGA_B} = {line_r[1], line_g[1], line_b[1]};
  else if (hcount[10:1] == start_col[0] && hcount[10:1] < end_col[0])
    {VGA_R, VGA_G, VGA_B} = {line_r[0], line_g[0], line_b[0]};
  else
    {VGA_R, VGA_G, VGA_B} = {8’d255, 8’d255, 8’d255};
end
```
Hardware Interface

- **16-bit address space**
  - First 10 bits specify row
  - Next 3 bits specify object number (up to 8 objects per row)
  - Last 3 bits specify property
    - 0: Start column
    - 1: Start column (last 2 bits)
    - 2: End column
    - 3: End column (last 2 bits)
    - 4: R
    - 5: G
    - 6: B

- **Write all 8 objects sequentially**
  - Saved to block RAM when the last object's B property is written
Software Interface

- Renders an array of rounded rectangles
- Rounded rectangles have a number of properties:
  - X position: distance from left edge of display, ignoring rounding
  - Y position: distance from top edge of display, ignoring rounding
  - Width, ignoring rounding
  - Height, ignoring rounding
  - Border radius
  - Red
  - Green
  - Blue
- Builds a displaylist from the array of rectangles, then writes to hardware and renders to display
It works :)

![Image of a computer monitor with a tiger emoji on it]
Extensions

- Add support for bitmaps
  - Can probably also be done in a single cycle for each pixel
  - Load all bitmaps for a given row during dead time
  - Add a new “bitmap-pointer” object type to the displaylist

- Use accelerator for a simple web browser
  - At least be able to render local HTML files with simple styling (maybe no font scaling)
    - Though font scaling could also be done during dead time
  - Possibly attach to internet as well (more of a networks project at that point)