Milkymist SoC

A performance-driven SoC architecture for video synthesis

Sébastien Bourdeauducq

KTH

June 2010

How it all started

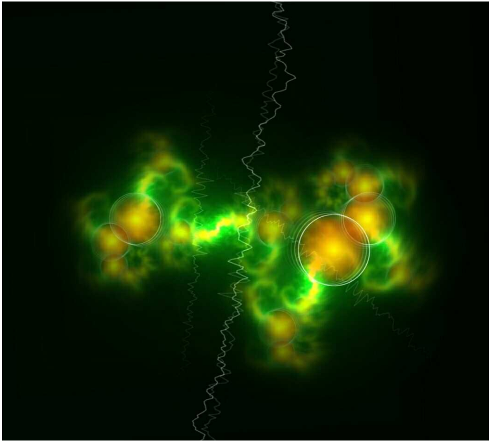
A device for video performance artists (VJs)...

- inspired by the popular MilkDrop program for PCs
- with many interfaces: MIDI, DMX, video in
- highly integrated

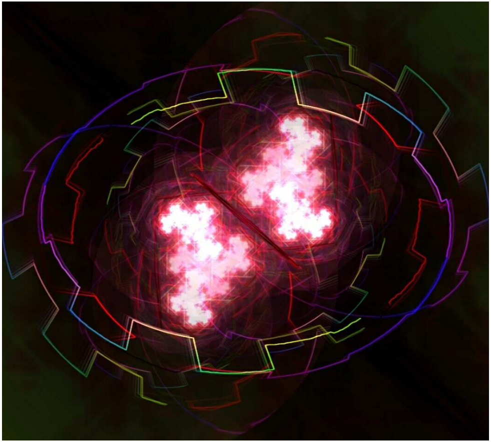
At the frontier between...

- big computers with software to render visual effects
- and small, handy microcontroller boards you connect to anything

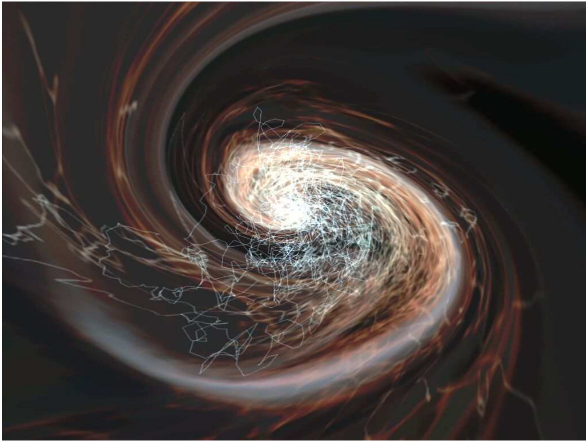
How does MilkDrop look?



How does MilkDrop look?



How does MilkDrop look?



How does MilkDrop work?

In two words:

- Take the current image, and distort it:
  - zoom
  - rotation
  - scaling
  - others...
- Draw waves and shapes.
- Display the result.
- Repeat the process (iterative rendering).

How does MilkDrop work?

- Distortion and waves are controlled by fully customizable equations.
- The set of those equations is called a “preset” or “patch”.
- Interaction of the visuals with sound is defined by those equations
- ...and also with DMX and MIDI in Milkymist.

Challenges

- The need for a CPU:
  - flexibility
  - ease of reprogramming, patching software bugs
  - software-friendly tasks: GUI, filesystems, protocols, ...
- Speed, size, and cost:
  - careful design
  - balance between hardware and software
  - software is cheap and slow, hardware is expensive and fast
- Memory problems: bandwidth, size.
- Compute-intensive operations:
  - distorting the image
  - evaluating the equations

SoC platform

- What is needed is a SoC with graphics acceleration.
- They are ubiquitous today:
    - Texas Instruments OMAP
    - Freescale i.MX
- However, those are closed and proprietary.
- This work: a new open source SoC that can run MilkDrop.

The memory problem

- A tough one.
- The application requires memory to be large, fast, and cheap.
- The required memory size prohibits the use of SRAM
- ...then we have to use DRAM and face all its problems.

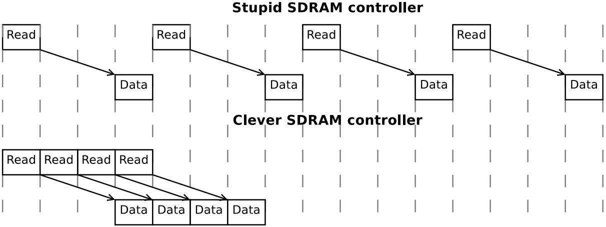
Bandwidth estimation

Task	Required bandwidth
VGA frame buffer	950Mb/s
Distortion	250Mb/s
Live video	300Mb/s
Scaling	500Mb/s
Video echo	900Mb/s
NTSC input	200Mb/s
Software and misc.	200Mb/s
Total	**3.3Gb/s**

- One DDR SDRAM chip running at 100MHz:
  - 3.2Gb/s peak bandwidth
  - 32MB capacity
  - a few dollars

Peak bandwidth?

Performance of SDRAM depends a lot on the cleverness of its controller.
Simplified example:



Memory transfers are always done using bursts of 4 consecutive words.
The bus master caches or discards the data it does not want.

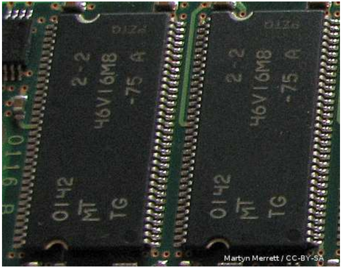
Techniques used in Milkymist

- Single SDRAM and system clock domains (reduces latency)
- Bursts
- Critical word first
- Pipelining
- Page mode DRAM control

Bursts: a good heuristics?

- Good for the VGA framebuffer (a big bandwidth consumer):
  - when it gets a burst of consecutive chunks from memory...
  - those chunks also represent consecutive pixels (in scan order)
  - ...so it can just put them in its output FIFO and easily acheive 100% utilization!
- It is the same for video inputs.
- For image distortion: yes; more on this later.
- For software: principle of temporal/spatial locality, caches.

Our memory system



- 2 chips of 32-bit DDR SDRAM at 100MHz.
- Peak bandwidth of 6.4Gb/s.
- Oversized – but this is necessary.

Performance measurement

Patch	BW	AMAT	Max. BW bound
Idle	292	5.51	3932
Bright Fiber Matrix 1	990	6.37	3474
Swirlie 3	1080	6.71	3320
Spacedust	1021	6.47	3427
Snowflake Delight	1399	6.28	3516
Balk Acid	1427	6.38	3469

What is “distortion”?

More precisely...

- Tessellate the source image with rectangles.
- Compute the source (texture) coordinates on each vertex.
- Fill each rectangle in the destination picture.
- Interpolate linearly the source (texture) coordinates.
- This is called *texture mapping*.

Speed constraints

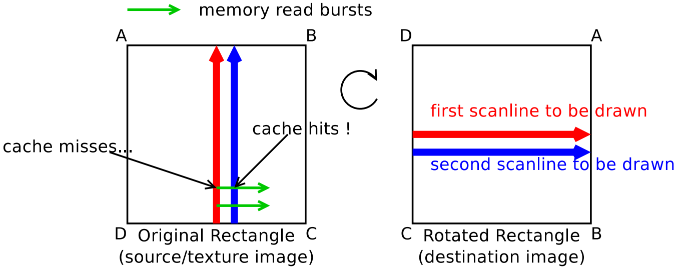
- Good system performance: must fill > 31 million pixels per second.
- With a 100MHz clock, we have < 3.2 cycles to put out a pixel.
- Precludes any software implementation (more than 40 times too slow).

Solutions

- Efficient algorithm
  - Inspired by Bresenham's linear interpolation algorithm
- "SIMD" parallelism
  - the same operation on independent data can be done in parallel
  - example: computing the interpolated X and Y in the texture
- Pipelined parallelism
  - Milkymist's TMU has about 20 pipeline stages
- Smart memory access
  - cache
  - write buffer

Using a cache

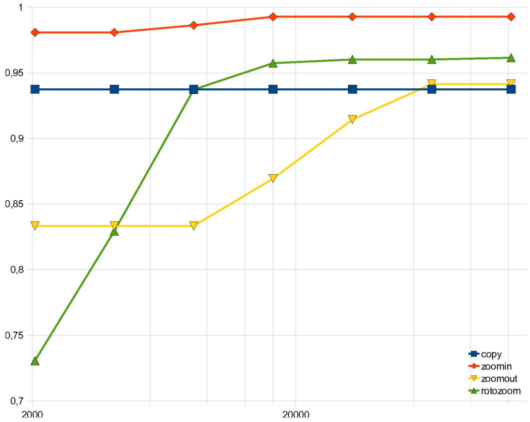
Example: rotation of a rectangle.



It can work !

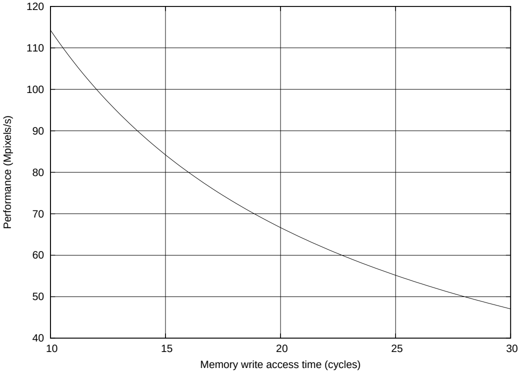
How big should the cache be?

- Simulation with different sets of texture coordinates:



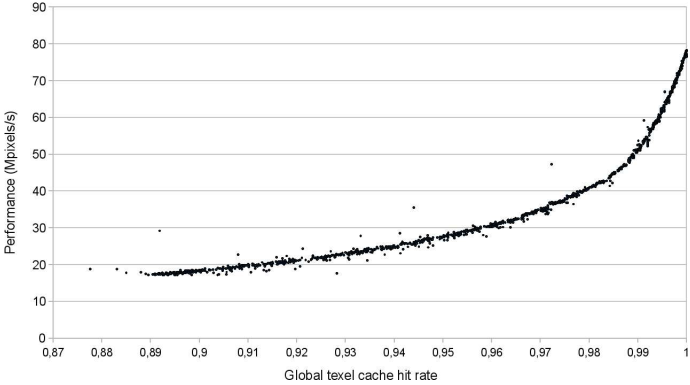
Write buffer

- "Double buffering", stores two bursts.
- 1 pixel/clock up to 12 cycles of memory access time.



Performance results

- Depends on cache hit rate.
- Enough performance for MilkDrop in 640x480 30fps.



The problem

- Intensive floating point processing, for each vertex.
- At least ≈ 58 million operations per second needed.
- Cannot be met with an in-order FPU at 100MHz in FPGAs (CPI < 1.73).
- We need parallelism.

Levels of parallelism

- Two approaches:
  - Vertex-level parallelism.
  - Instruction-level parallelism.
- Vertex-level parallelism requires more on-chip storage for temporary values.
- Instruction-level parallelism is potentially slower.
- The two approaches are not mutually exclusive.
- We focused on instruction-level parallelism only (simpler).

Instruction-level parallelism

- Out-of-order execution.
- Relatively expensive and complex hardware structures.
- We avoid them with instructions statically scheduled by the compiler.
  - like VLIW architectures.
- Works well, because:
  - all delays are known (negligible memory accesses).
  - no control hazards.

Results

Patch	Instructions	Cycles	CPI
Default	192	259	1.35
The Tunnel	208	286	1.38
Warp of Dali 1	220	292	1.33
Digital Flame	216	293	1.36
Wormhole Pillars	231	326	1.41

- We needed CPI < 1.73.
- Success!

Conclusion

- We have developed a working MilkDrop rendering program for the SoC.
  - proof of concept
- Further development
  - Interfaces support: video input, DMX, MIDI, USB ...
  - Operating system support.
  - End user application.
  - “Packaged” device.
- Further research
  - Out-of-order memory subsystem.
  - Texture mapping unit prefetching.
  - High level synthesis.

Thank you for your attention

- Web: `http://www.milkymist.org`
- Mailing list: devel [AT] lists [DOT] milkymist [DOT] org

Demonstration & questions