

Milkymist™

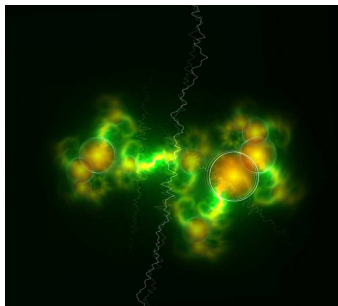
Open Source Hardware interactive VJ station

Yann Sionneau

OSHUG #8 – London

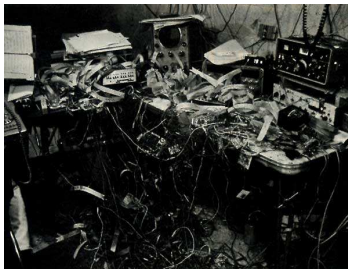
March 10, 2011

Some background...



- Sebastien Bourdeauducq started VJing in 2005
- Kludgy, ad hoc software inspired by MilkDrop, running on Linux PCs
- Tried to make performances more interactive

Why Milkymist was born

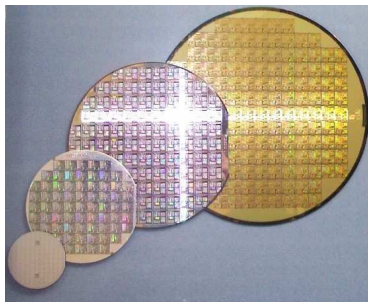


- PC towers feel heavy when carried around at 5am
- Long boot and setup time
- Power cut = 500 people see your computer rebooting
- Interfaces (MIDI, DMX, video input ...)
 - expensive
 - big mess of wires
 - driver problems at the wrong moment

Embedded would be cool, wouldn't it?



- Do all the VJing with a small box that contains everything you need
- Live seconds after power up
- Technically feasible? Yes, but needs high-performance SoC.
- Open Source Hardware is better...



Part I – The Chip

Open Hardware, SRSLY.

- Open source software is already well known...
- Among many hackers and artists: trend for “open source hardware”
- Famous projects based around open PCB designs: Arduino, Beagleboard, ...
- But all the magic is done by a proprietary chip!
- Not that Open Source. We want to go further!

You know, it's true, when something exceeds your ability to understand how it works, it sort of becomes magical. (iPad advertisement)

The Milkymist chip

- Instead of using a blackbox proprietary chip...
- ...Milkymist features a custom, free (System-on-)Chip design
 - Open Source Hardware (sources GNU GPL v3)
 - built-in custom interfaces
 - high performance
 - small size
- Works on FPGAs – reconfigurable silicon
 - enables everyone to modify and test the design
- Portable code – small effort to port to all FPGA vendors, or even to make a fully “hard” chip (and yes, we’ll do it)

You know, it's true, when something exceeds your ability to understand how it works, you sort of become stupid.

What makes it specific?

- The Milkymist System-on-Chip (SoC) contains a general-purpose Linux-capable CPU.



VJ-friendly on-chip features

- Video output (VGA)
- Video input (ITU-R BT.656, i.e PAL/SECAM/NTSC)
- AC'97 audio
- Ethernet (OpenSoundControl)
- MIDI
- DMX512
- USB
- IR remote control (R-5)
- Generic digital I/O
- Hardware graphics acceleration
 - A small subset of OpenGL, enough for our application.

Technical challenge

- The application requires memory to be big, fast, and cheap at the same time.
- The required memory size prohibits the use of SRAM
- ...then we have to use DRAM and face all its problems.

How bad is it?

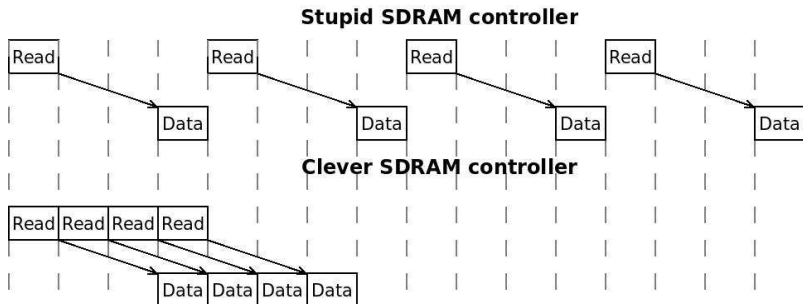
Task	Bandwidth
VGA framebuffer, 1024x768, 75Hz, 16bpp	950 Mbps
Distortion : texture mapping, 512x512 to 512x512, 30fps, 16bpp	250 Mbps
Live video : texture mapping, 720x576 to 512x512 with transparency, 30fps, 16bpp	300 Mbps
Scaling : texture mapping, 512x512 to 1024x768, 30fps, 16bpp	500 Mbps
Video echo : texture mapping, 512x512 to 1024x768 with transparency, 30fps, 16bpp	900 Mbps
NTSC input, 720x576, 30fps, 16bpp	200 Mbps
Software and misc.	200 Mbps
Total	3.3 Gbps

How bad is it?

- One DDR SDRAM chip running at 100MHz: (DDR-400 in marketing speak)
 - maximum of 2 transfers / cycle $\Rightarrow$ 200 Mt/s
 - 16 bits / transfer
 - 3.2 Gbps peak bandwidth
 - 64 MB capacity
 - a few dollars
- We have two of those chips, this is still manageable!

Peak bandwidth?

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



In Milkymist, memory transfers are always done using bursts of 4 consecutive words (FML bus). The bus master caches or discards the data it does not want.

Bursts of 4 consecutive words: a good heuristics?

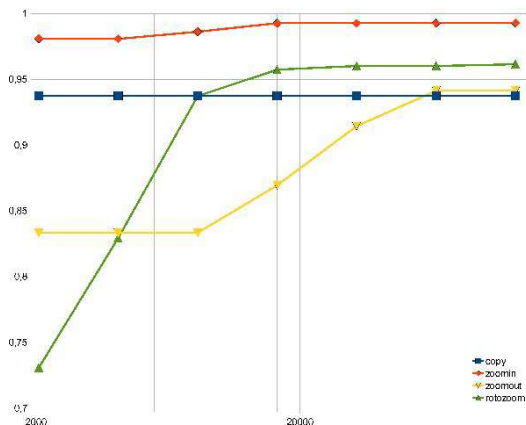
- Good for the VGA framebuffer (a big bandwidth consumer):
 - when it gets a burst of 4 consecutive chunks from memory...
 - those chunks also represent consecutive pixels (in scan order)
 - ...so it can just put them in its output FIFO and easily achieve 100% utilization!
- It is the same for PAL/NTSC video inputs.
- For image distortion: yes; more on this later.
- For software: principle of temporal/spatial locality, caches.
- And simplifies a lot (compared to a variable burst length):
 - SDRAM controller design
 - Bus signaling
 - Bus arbitration
 - Cache controllers

Performance constraints

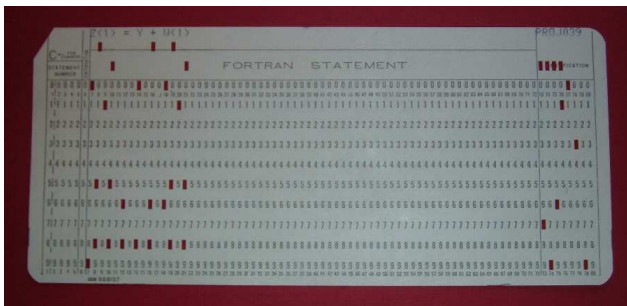
- To achieve 30fps at 1024x768, the Texture Mapping Unit (TMU) must process 24 million pixels per second.
- With a 100MHz clock, we have 4 cycles to process a pixel.
- With a naive implementation, each pixel requires (approx.):
 - math: 100 cycles
 - one memory read: 12 cycles
 - one memory write: 12 cycles
- Total: 124 cycles!!! Solutions for a 3000% speedup?!?

Solutions

- Improved algorithm
 - Bresenham's linear interpolation algorithm
- SIMD parallelism
 - the same operation on independent data can be done in parallel
- Pipelined parallelism
 - Milkymist's TMU has 14 pipeline stages
 - will grow as more features are added...
 - commercial GPUs can have hundreds
- Smart memory access
 - cache
 - write buffer



- Cache de 32Kio = 95% de hits
- Temps moyen = $0.05 \cdot (12+3) + 0.95 \cdot 1 = 1.7$ cycles!



Part II – The Software

It's the software, stupid!

- Well, some people are thinking...
- “pew, VJs ain’t gonna program FPGAs!”
- The FPGA is transparent (unless you go looking for it)
 - it will even disappear when we start engraving our own chips
- Only the software defines what the average user sees.
- If you get it right, you can hide any technical complexity.
 - people use Mac OS X on computers more complex than Milkymist, and think it’s simple.

Flickernoise, Milkymist's VJ application

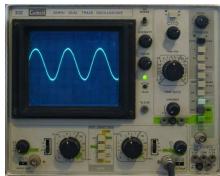
- Will boot and get ready in seconds
- Using MTK based on the Genode FX GUI toolkit
- Visual effect renderer inspired by MilkDrop
- GNU GPL licensed
- OS: RTEMS



How does the renderer work?

Based on a simple iterative process:

- Draw a waveform based on the music
- Distort the image
- Draw a waveform again...



Extra features

- Borders
- Motion vectors
- Video echo
- Blend with video input

Upcoming:

- Custom waves and shapes
- User defined pictures
- ...

Patch parameters

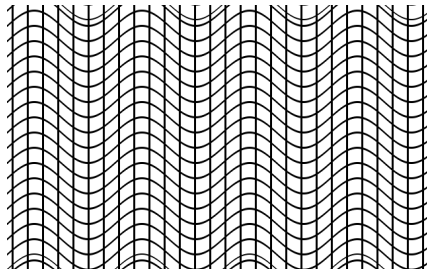
- Parameters to customize the patch, e.g. :
 - how the wave is traced: `nWaveMode`
 - wave color: `wave_r`, `wave_g`, `wave_b`
 - amount of zoom when distorting: `zoom`
 - displacement when distorting: `dx`, `dy`
 - ...

Per-frame equations

- Allow the parameters to change over time
- Examples:
 - $\text{per_frame_1} = dx = 0.1 * \text{bass}$
 - $\text{per_frame_2} = \text{wave_r} = 0.3 + 0.1 * \sin(6.28 * \text{time})$
 - ...

Per-vertex equations

- Configure the distortion on each control point (vertex)
- Example: $\text{per_vertex_1}=\text{dy}=0.1*\sin(40*x)$





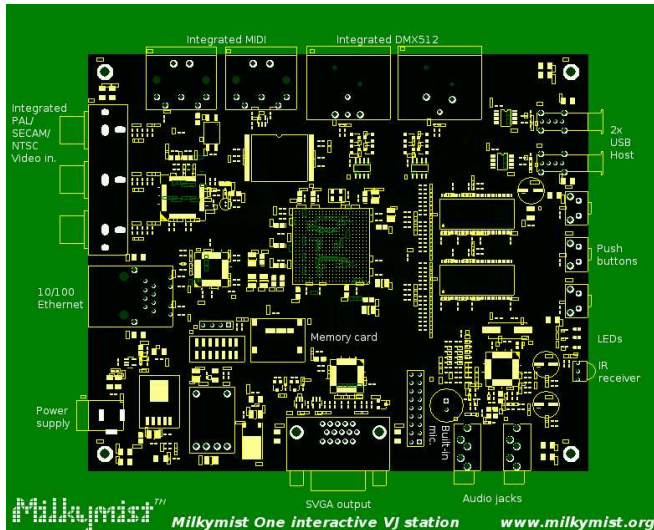
Part III – Milkymist One

Milkymist One

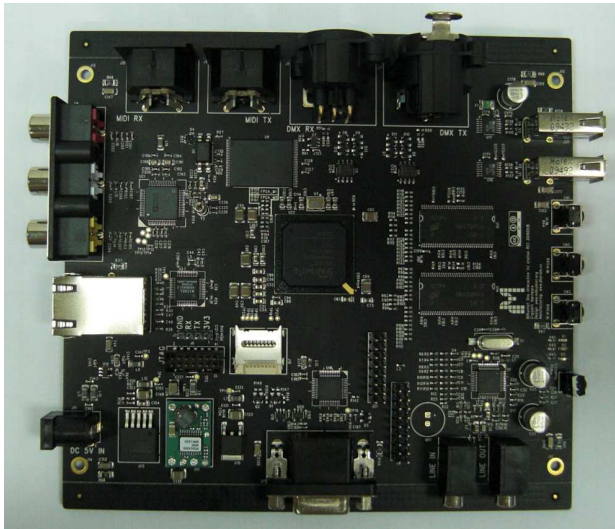
- We want to make a complete, high quality end user product
 - An interactive VJ station, not (only) a FPGA development board!
 - With case and packaging
 - Works out of the box
- Open design as well (like everything else)
- Developed, manufactured and sold in collaboration with Sharism at Work Ltd. (known for Ben Nanonote)



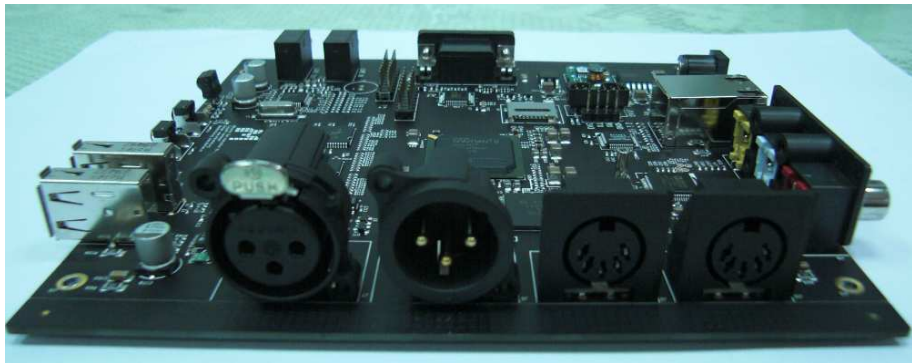
Milkymist One (M1)



Milkymist One (M1)



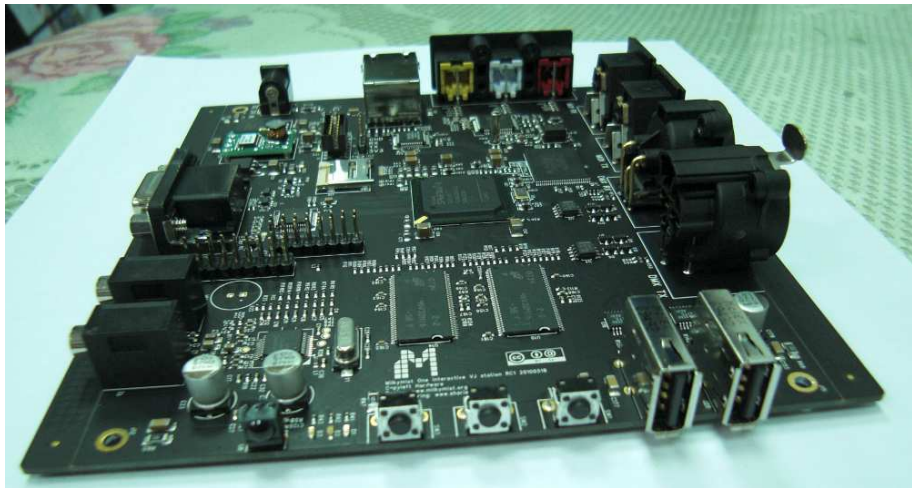
Milkymist One (M1)



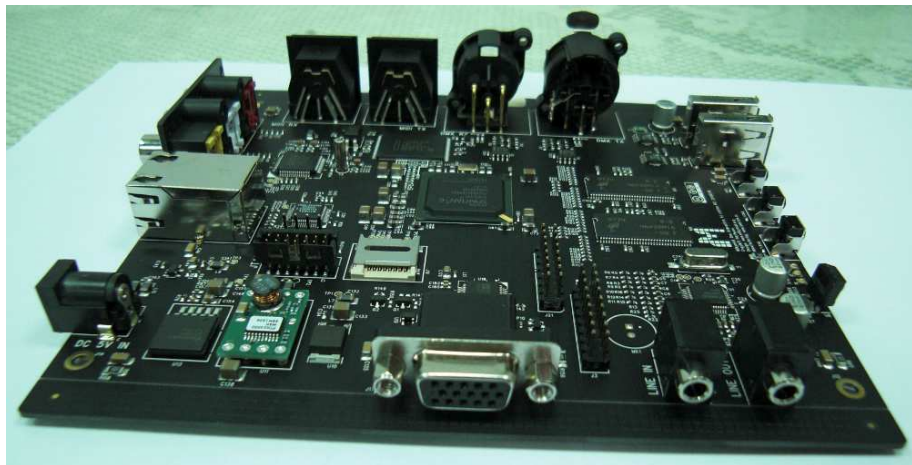
Milkymist One (M1)



Milkymist One (M1)



Milkymist One (M1)



Status

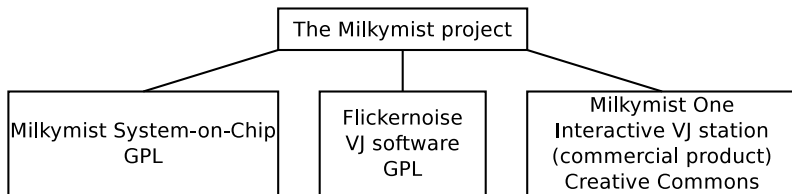
- Milkymist One RC-1 and RC-2 have been manufactured, RC-3 is coming soon

Active contributors

- Sebastien Bourdeaducq (Germany) – project mangement, electronic design, SoC and Flickernoise
- Takeshi Matsuya (Japan) – Linux port
- Yanjun Luo (China) – JTAG daughterboard design
- Wolfgang Spraul (China) – manufacturing and distribution
- Michael Walle (Germany) – QEMU, RTEMS port
- Adam Wang (Taiwan) – electronic development and production
- Joachim Steiger (Germany) – DIY steampunk case and acrylic case
- Yann Sionneau (France) – RTEMS port (GSoC 2010)

Community

- Want to develop on a truly Open Source platform ?
- **We are searching for software developers.**
- You don't need any special tool or FPGA.
- “Classical” tasks, such as user interface.
- Even if you don't care about VJ, we encourage you to reuse the code (FPGA, OS, software, ...).
 - You can help tracking bugs, improve OS support and toolchain.
 - Example: NASA's CoNNeCT experiment (software defined radio, Milkymist's DRAM controller).



- Web: <http://www.milkymist.org>
 - documented source code (GPLv3 licensing)
 - mailing list, wiki, blog, IRC (#milkymist on Freenode : <http://goo.gl/EwPu7>)
- Mail: [yann.sionneau \[AT\] gmail DOT com](mailto:yann.sionneau@gmail.com)
- project leader : [sebastien.bourdeauducq \[AT\] lekernel DOT net](mailto:sebastien.bourdeauducq@lekernel.net)
- These slides are under CC-by-SA Sebastien Bourdeauducq & Yann Sionneau
- Source of the slides : http://sionneau.net/oshug8/milkymist_oshug8.tex