OpenCL on the FPGA

So founders can drink beer now

OPENCL ON THE FPGA

So founders can drink beer now

Hi! I'm Vincent Hindriksen

- Founder of a company called StreamComputing
- My colleagues know how to make software fast, really fast!
- We use everything, from algorithm-optimisations to duct-tape.
- One of the target devices we use are FPGAs

SEE OUR HOMEPAGE FOR BRAGGING POSTS ON SPEEDUPS WE GOT

From 33 seconds to 0.068 seconds

OpenCL – The Compute Language

- Young language, started only in 2009
- Enables compute parallelism and explicit memory management in one C-like language.
- Open standard supported by many big companies, like AMD, NVidia, Intel, Altera, Xilinx, ARM, Vivante, Imagination, Texas Instruments, Qualcomm, etc.
- Now at version 2.1, but no drivers available yet.
- OpenCL 1.2 is the most used version, where 2.0 is getting traction quickly.

OpenCL – The Compute Language

- Works on CPUs – you know
- Works on GPUs – graphics units that now also process matrices
- Works on DSPs – Digital Signal processors – like your soundcard
- Works on FPGAs – Our subject
- Works on Steroids – because it's fast

OpenCL parallelism in 3 simple steps

- Take a small function (kernel) that operates on one data element
- Define the data to operate on
- Combine the above two and Tada!
- (Then optimise the hell out of it, which takes most of the effort)

APPLY KERNEL-FUNCTION TO *EACH* ELEMENT

2

What is an FPGA?

- It's a processor where the program doesn't run on it, but the processor is the software itself. So... it is actually hardware-only programs.
- It runs programs with very low latency – the time to process simple input can be done under 0.1 microsecond = 0.0001 millisecond = 0.0000001 second.
- It's much cooler than a GPU. Mostly because it uses 10x less power.
- Many brands, two biggest are Altera and Xilinx

WHY PROGRAM ALSO FPGAS WITH OPENCL?

- Standard languages VHDL and Verilog are for designing IP (functionality)
- OpenCL is for designing full systems (connect IP to Resources like memory)

- OpenCL is good in defining parallelism

OpenCL on FPGAs

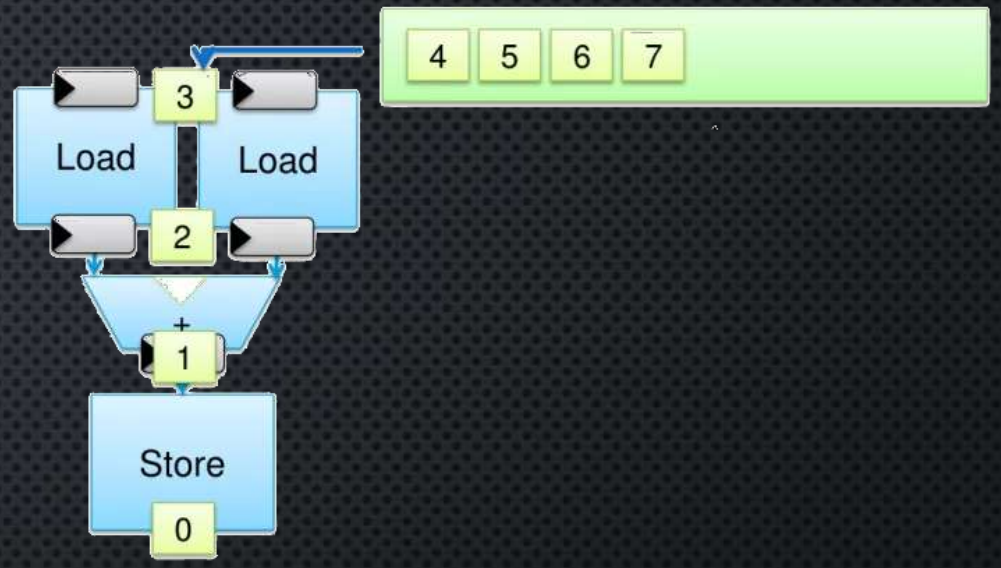
- Only on Altera and Xilinx
- Altera is slowly getting to version 1.2
- Xilinx is still pretty much beta-state, but it works.

Standard "SIMD" parallelism

- core 1 processes task 1
- core 2 processes task 2
- core 3 processes task 3
- core 4 processes task 4
- Etc

Thng is: FPGA doesn't have "cores". So what now?

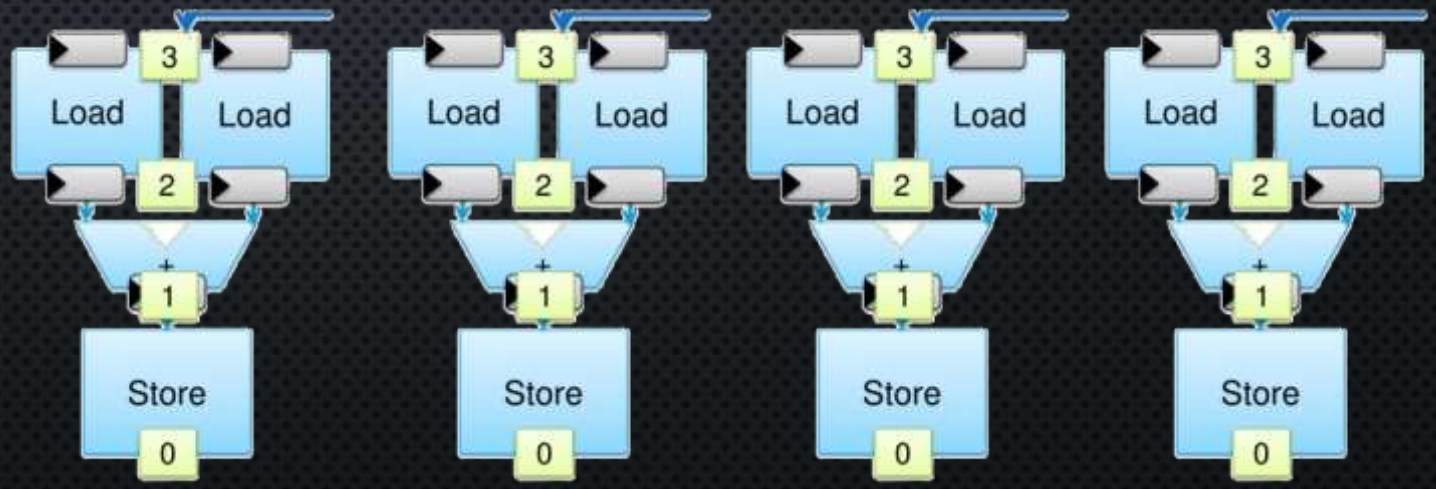
PARALLELISM ON FPGAS 1/2



- FPGA USES PIPELINE
- EACH TASK IS PIPED THROUGH. WHEN A PHASE IS FINISHED, ALL TASKS HOP FORWARD ONE POSITION.
- IF THERE IS SPACE, MULTIPLE IDENTICAL PIPELINES CAN BE CREATED.

PARALLELISM ON FPGAS 2/2

- MULTIPLE PIPELINES



EXAMPLE! DEVICE-SIDE

```
__kernel void vector_add(__global const float *x,
                         __global const float *y,
                         __global float *restrict z)
{
   // get index of the work item
   int index = get_global_id(0);

   // add the vector elements
   z[index] = x[index] + y[index];
}
```

Host-side boilerplate (standard)

1. Get a list of available platforms
2. Select device
3. Create Context
4. Create command queue
5. Create memory objects
6. Read kernel file
7. Create program object
8. Compile kernel
9. Create kernel object
10. Set kernel arguments
11. Execute kernel (Enqueue task)
12. Read memory object
13. Free objects

Host-side boilerplate (when FPGA-only)

1. Select FPGA-device
2. Create Context and command queue
3. Create memory objects
4. Read pre-compiled kernel file
5. Create Kernel object of pre-compiled kernel
6. Set kernel arguments
7. Execute kernel (Enqueue task)
8. Read memory object
9. Free objects

Or:

1. Copy host-code from previous project
2. Alter till it works

TRY YOURSELF FPGA

- CHEAP BOARD - $250
- GIVES ONE YEAR LICENCE (NOT SURE IF STILL APPLICABLE…)
- IT RUNS "HELLO WORLD" AND MORE!

Want to know more?

- Uhm, uh… Google?
- Follow @OpenCLonFPGAs on Twitter
- Check websites of Altera and Xilinx
- Ask us