Accelerating Radiance with Novel Hardware

David Coulthurst

Graphics Group

University of Bristol

With thanks to:

Simon McIntosh-Smith

Director of Hardware

ClearSpeed

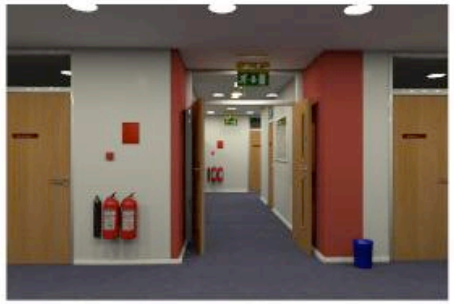
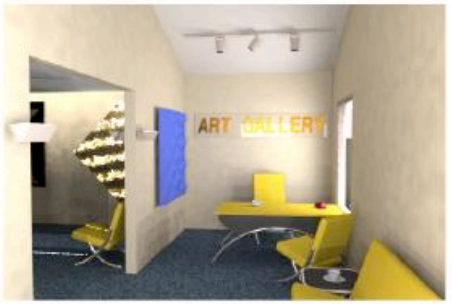
Overview

- Introduction
- Current hardware
- New hardware
- How this hardware can be harnessed
- Conclusions

Introduction

High fidelity graphics hardware

- We aren't using all the hardware that is out there.
- If we do, what speedup can we get.

Current Hardware

- CPU – Pentium4, Athlon – what Radiance runs on currently.
- GPU – nVidia, ATI – other ray-tracers have been GPU implementations.
- Clusters / Grids – Radiance and other ray-tracers have been implemented.

Current Hardware – CPU

- Good at branching & structure traversal – Acceleration structures such as k-d trees and octrees.

- They use double precision – necessary for complex geometry.

- 4-SIMD is the current maximum.

Current Hardware – GPU

- Can perform large volume of SIMD calculations in parallel.
- Simplistic branching – dynamic branching and looping is just starting to appear.
- Single precision floating point only at the moment.

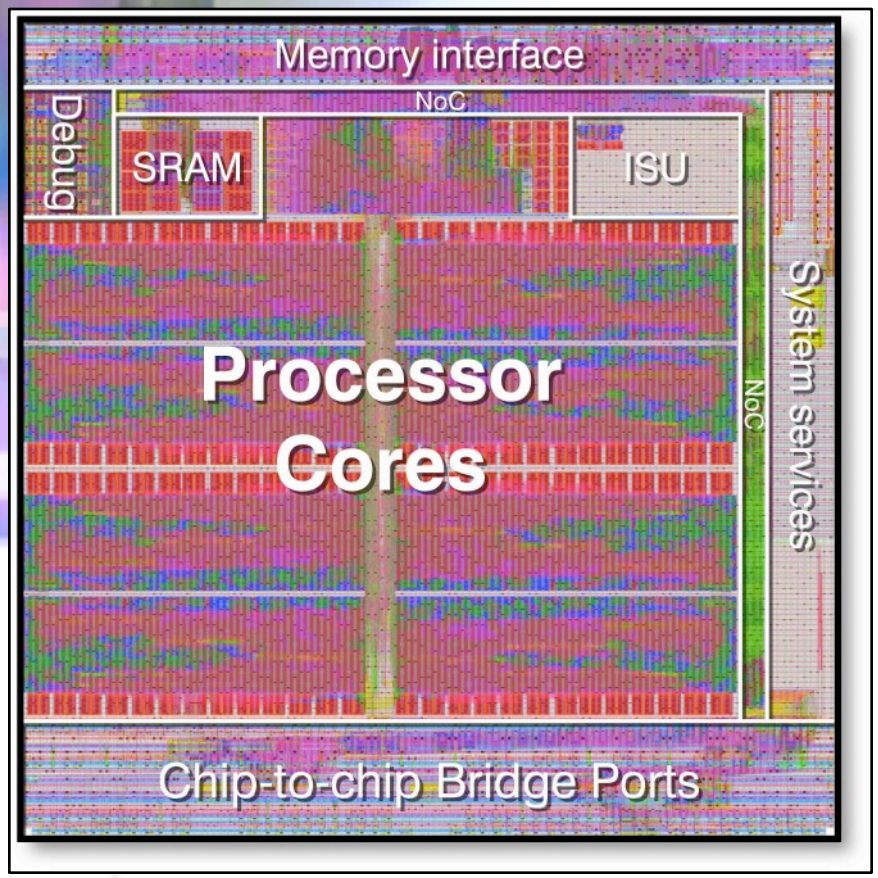
Current Hardware – Clusters and Grids

- Clusters –
  - multiple computers acting as one.
  - one frame per computer
  - portions of a frame per computer
- Grids –
  - Similar to clusters, but more separation between computers.
- Latency, granularity and load spreading.

New Hardware – ClearSpeed Boards

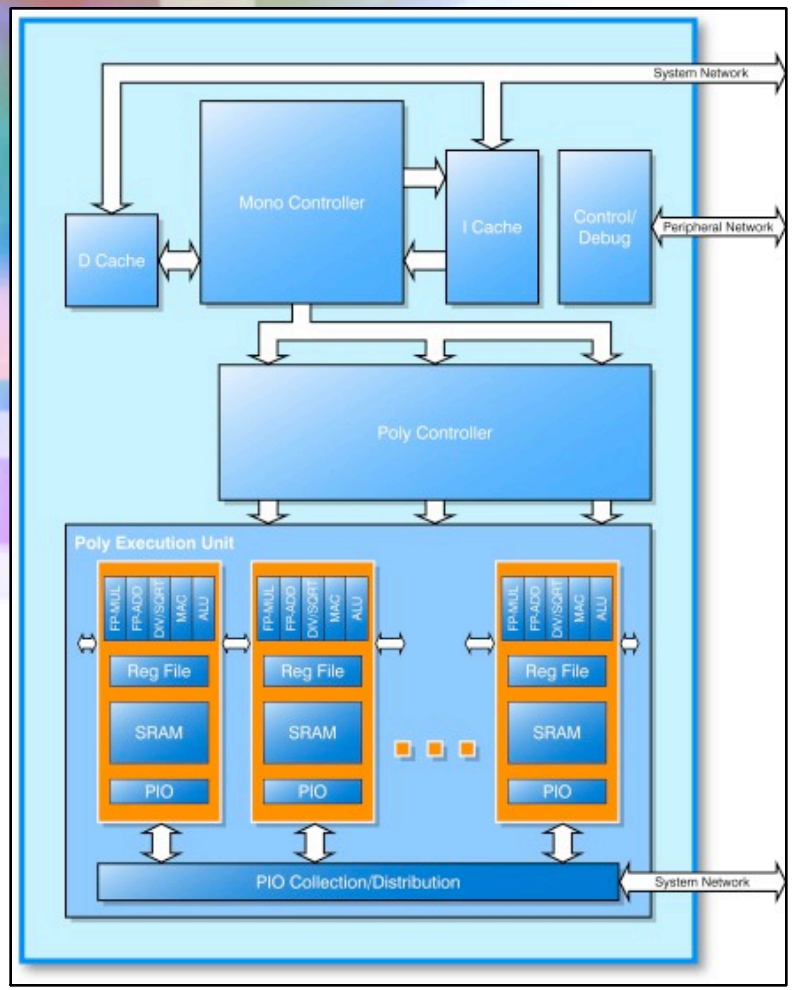
- Massively SIMD co-processors.
- Full IEEE 734 single and double precision floating point.
- Multi-threaded and high level of branching.
- Complementary to C.P.U. and G.P.U.

ClearSpeed CSX600 Coprocessor



- Array of 96 Processor Elements
- 250 MHz
- IBM 0.13µm FSG process, 8-layer metal (copper)
- 47% logic, 53% memory
  - More logic than most processors!
  - About 50% of the logic is FPUs
  - Hence around one quarter of the chip is floating point hardware
- 15 mm x 15 mm die size
- 128 million transistors
- Approx. 10 Watts

CSX600 Processor Core



Multi-Threaded Array Processing

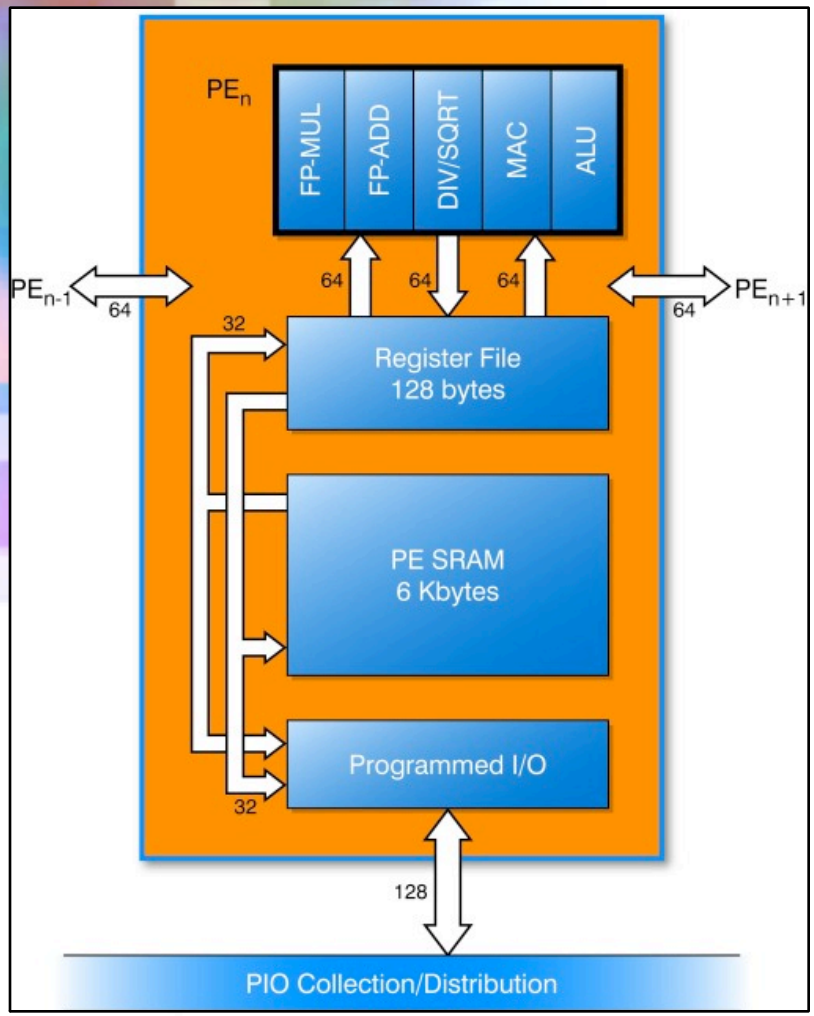
- Programmed in high-level languages
- Hardware multi-threading for latency tolerance
- Asynchronous, overlapped I/O
- Run-time extensible instruction set
- Bi-endian (compatible with host CPU)

Array of 96 Processor Elements (PEs)

- Each is a Very Long Instruction Word (VLIW) core, not just an ALU
- Flexible data parallel processing
- Built-in PE fault tolerance, resiliency

High performance, low power dissipation

CSX600 Processing Elements

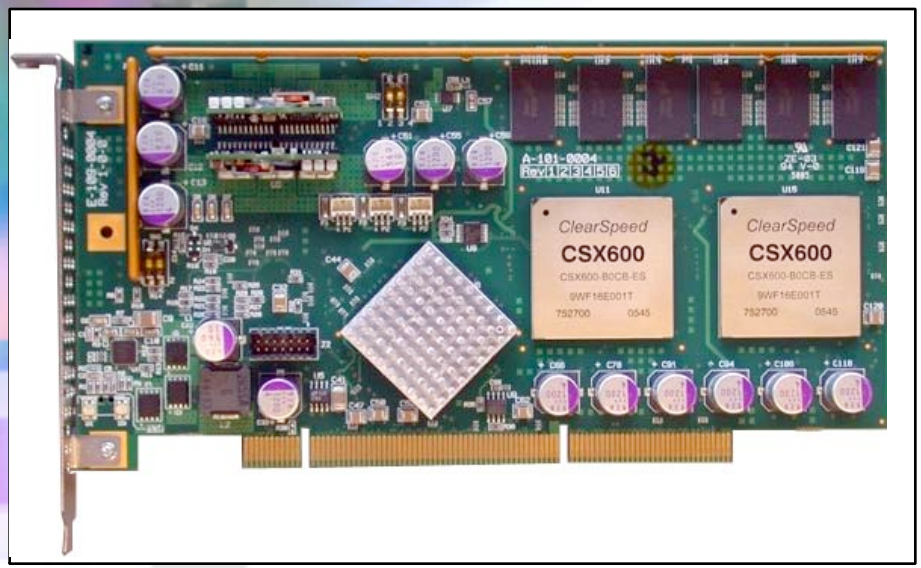


Each PE is a VLIW core:

- Multiple execution units
  - 4-stage floating point adder
  - 4-stage floating point multiplier
  - Divide/square root unit

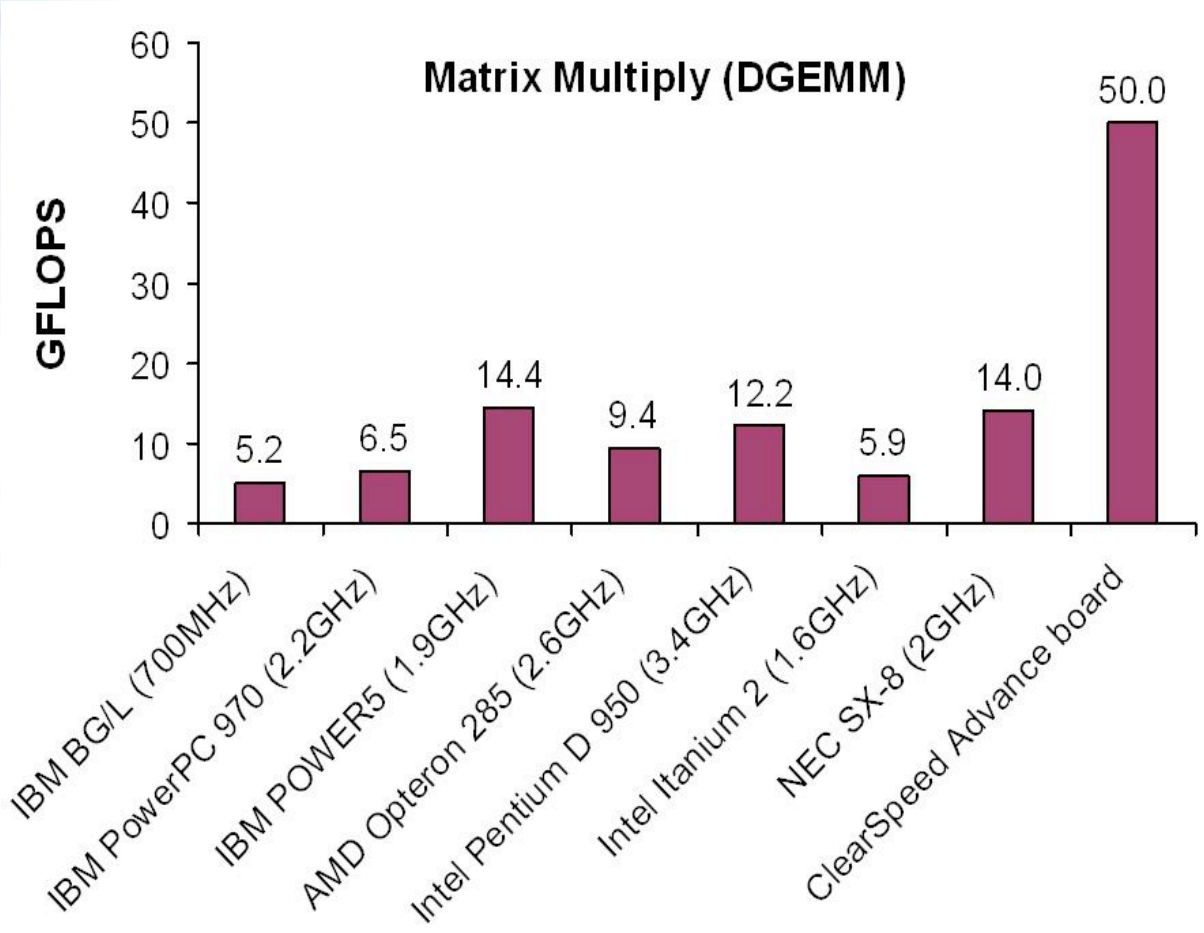
  } 32/64-bit IEEE 754
  - Fixed-point MAC 16x16 → 32+64
  - Integer ALU with shifter
  - Load/store
- High-bandwidth, 5-port register file (3r, 2w)
- Closely coupled 6 KB SRAM for data
- High bandwidth per PE DMA (PIO)
- Per PE address generators
  - Complete pointer model, including parallel pointer chasing and vectors of addresses

Dual CSX600 PCI-X accelerator board



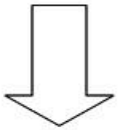
- 50 DGEMM GFLOPS sustained
- 0.4 M 1K complex single precision FFTs/s (20 GFLOPS)
- ~200 Gbytes/s aggregate B/W to on-chip memories
- 6.4 Gbytes/s aggregate B/W to local ECC DDR2-DRAM
- 1 Gbyte of local DRAM (512 Mbytes per CSX600)
- ~0.8 Gbytes/s to/from board via PCI-X @133 MHz
- 25 watts for entire card (8” single-slot PCI-X)

Matrix Multiplication



Porting Code

```
void daxpy(double *c, double *a, double alpha, uint N) {
  uint i;
  for (i=0; i<N; i++)
    c[i] = c[i] + a[i]*alpha;
}
```



```
void daxpy(double *c, double *a, double alpha, uint N) {
  uint i;
  poly double cp, ap;
  for (i=0; i<N; i+=num_pes) {
    memcpym2p(&cp, &c[i+pe_num], sizeof(double));
    memcpym2p(&ap, &a[i+pe_num], sizeof(double));
    cp = cp + ap*alpha;
    memcpyp2m(&c[i+pe_num], &cp, sizeof(double))
}
```

How this hardware can be harnessed

- Floating point precision significance.
- Different characteristics of computation.
- Using multiple pieces of hardware.
- Hardware resource Allocator

How this hardware can be harnessed

Floating point precision significance

- What is the effect of using single precision floating point versus double precision in ray-tracing.
- The general view is that double precision is needed for geometry calculations, and single precision for colour calculations.
- Create variations on radiance using double for all, single for all, and a mixture. Test on typical and atypical scenes.
- Compare the results using image metrics.

How this hardware can be harnessed

Different Characteristics of computation

- Different parts of ray-tracing have different characteristics.
- Acc. Structure creation may need to be fast, high level of branching and double precision – so suited to CPU.
- Ray-scene traversal / ray-triangle intersection may require a high level of parallelism, branching and double precision – CSC.
- Shading may require high level of parallelism, but only require single precision – GPU.

How this hardware can be harnessed

Using multiple pieces of hardware

- A current pc may contain for example 1-2 CPUs, 1-2 GPUs, 1+ CSC

- If we know which pieces are best at which type of ray-tracing work, that type of work can be allocated to them.

- Have to decide how much work to send given the latency to send, process and receive work.

How this hardware can be harnessed

Hardware Resource Allocator

- Spread work out among pieces of hardware.

- Take the knowledge about different pieces of hardware, different work in ray-tracing, etc, and use it to write a lightweight piece of code to harness this.

- Producer - Consumer framework that can farm out the work to the most appropriate piece of hardware, then collect and process the results.

Conclusions

- What hardware suits which work, based on f.p.p., acc. structure traversal, etc.

- Techniques for farming out work in ray-tracing based on this – HRA, or other.

- Extending from a single computer to a cluster or larger – what are the scaling issues.

Further Contact

ClearSpeed

David Coulthurst

Computer Graphics group

University of Bristol

davec@cs.bris.ac.uk

Simon McIntosh-Smith

Director of Hardware

ClearSpeed

simon@clearspeed.com