ECE 574 – Cluster Computing
Lecture 23

Vince Weaver

https://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

2 November 2026

Announcements

- HW#8 was posted
- Don't forget project topics due Friday

Accelerators

What if CPU power isn't enough?

- We've been mostly looking at ways to get the most performance out of CPUs
- What else is there?

Accelerator Options – ASIC

- hard-coded custom hardware for acceleration
- quite possibly the fastest, as custom made for your workload
- expensive to make, as one-off
- need to hire ASIC designers and get things fabbed
- found in BitCoin mining?

Accelerator Options – FPGA

- Reprogramable logic
- can have fast in-hardware designs but can re-program when workload changes
- Need to have someone who can write FPGA code
- There has been work for having OpenMP and such be able to handle FPGAs
- Tend to be expensive

Accelerator Options – DSP

- Digital Signal Processors
- Can be good at certain workloads
  - linear algebra
  - FFTs
- Some supercomputers have had them in past

Google Tensor Processing Unit (TPU)

- For accelerating machine learning tasks
  - TPUs good at CNN (convolutional neural networks)
  - GPUs good at fully connected
  - CPUs good at RNN (recurrent)
- ISCA paper – In Datacenter Performance Analysis of a Tensor Processing Unit
- For high-volume low-prevision FP calculations (8 and 16-bit)
- Unlike GPU has no rasterizer or texture processor

Other TPU implementations

- Recently with the advent of AI people have been putting TPUs into everything
- Some recent NVIDIA GPUs have tensor units
- Apple M chips in laptops/cellphones have NPUs
- Intel "AI Boost" in Extreme chips have NPUs
- Intel's AMX instructions add NPU acceleration

Programming TPUs

- Was trying to research this
- Maybe with JAX? Pallas? Using Pytorch
- Might make an interesting project
- I don't necessarily have the proper hardware though

Accelerator Options – Cell Processor (Obsolete)

- Special IBM Power core that had many smaller helper cores
- Could be really fast if programmed well, hard to program
- In end people found it not worth the extra effort
- Was also in PlayStation 3
- Some groups would buy them up and make fast clusters with them. This annoyed Sony who eventually dropped Linux support

Accelerator Options – Xeon Phi (Obsolete)

- Intel, came out of the larabee design (effort to do a GPU powered by x86 chips)
- Large array of x86 chips(p5 class on older models, atom on newer) on PCIe card.
- Sort of like an internal mini cluster
- Runs Linux, can ssh into the boards over PCIe.
- Benefit: can use existing x86 programming tools and knowledge.
- Intel cancelled this

GPUs

- The most common accelerator these days are GPUs (graphics processing units)
- We'll look in detail how these are used

Graphics Background

- Graphics have often been of interest in HPC even outside GP-GPU
- Visualizing the results of experiments is important
- At SC they often have special presentations / awards for best graphics visualizations of HPC results

Old CRT Days

- Electron gun
- Horizontal Blank, Vertical Blank

LCD Displays (sic)

- Crystals twist in presence of electric field
- Asymmetric on/off times
- Passive (crossed wires) vs Active (transistor each pixel)
- Passive have to be refreshed constantly
- Use only 10% of power of equivalent CRT
- Circuitry inside to scale image and other post-processing
- Need to be refreshed periodically to keep their image
- New “bistable” display under development, requires no power to hold state

Coding for CRTs

- Atari 2600 – only enough RAM to do one scanline at a time
- Apple II – video on alternate cycles, refresh RAM for free
- Bandwidth key issue. SNES / NES, tiles. Double buffering vs only updating during refresh
- Multibanks of graphics (VGA and older) another way to deal with lack of bandwidth

Old 2D Video Cards

- Framebuffer (possibly multi-plane), Palette
- Dual-ported RAM, RAMDAC (Digital-Analog Converter)
- Interface (on PC) various io ports and a 64kB RAM window
- Mode 13h

Early Graphics Acceleration

- Sprites — small bitmaps that can be moved on screen
- Scrolling — pan across memory to quickly scroll screen
- Copper — small co-processor can modify graphics independent of main CPU
- 2d acceleration — drawing lines, rectangles, mouse cursor

3D Video Cards History

- At first only in high-end workstations (like SGI)
- 3dfx cards, with passthrough cable
- Became more mainstream
- Eventually NVIDIA and ATI (bought by AMD) with Intel distant third
- Embedded systems also have GPUs

Aside: Haswell-EP GPU

- It came with a Matrox MGA G200, from 1998
- Why? Traditionally servers weren't used for gaming or displays so performance not important so cheapest card available
- I do have an NVIDIA card installed for use on HW

Modern Graphics Cards

- Essentially high-end linear algebra / 3D rendering supercomputers
- Can draw a lot of power
- 2D (optional afterthought these days)
- Can contain other hardware accelerators (such as Video decoders, TPUs, NPUs)

Interface – Integrated vs Standalone

- Integrated
  - Built into motherboard/chipset/processor
  - Can share memory (and bandwidth) with CPU
  - Traditionally less capable, but that is changing
- Standalone
  - Usually in PCIe slot, bandwidth constrained
  - Can draw lots of power
  - Can have multiple

Video RAM

- VRAM (old) – dual ported. Could read out full 1024Bit line and latch for drawing, previously most would be discarded (cache line read)
- GDDR3/4/5 – traditional one-port RAM. More overhead, but things are fast enough these days it is worth it.
- Confusing naming, GDDR3 is equivalent of DDR2 but with some speed optimization and lower voltage (so higher frequency)
- HBM – high bandwidth memory, stacked on top of GPU or else next to it on interposer. Much higher bandwidth (1024 bit wide vs 64-bit for DDR)

Busses Connecting GPUs to CPUs

- Ancient busses parallel – ISA
- Move to 32 bit big debate (EISA/VLB/MCA)
- Intel's PCI won: PCI and PCI-X were 32/64-bit parallel bus
- Around this time people stopped wanting parallel busses
  hard to keep so many lines in sync, hard to route on motherboard

PCI-Express PCIe

- PCIe is a serial bus, sends packets
- One lane: differential pairs, one in each direction (so 4 wires)
  - 1x, 2x, 4x, 16x lanes can be grouped together for speed
  - Your CPU only has so many lanes available which can limit how many slots on motherboard
  - Especially as PCIe lanes needed for things like storage, network, etc, not just GPUs

PCI-Express PCIe Versions

- Versions of spec 1.0 up through 7.0
  - Each one faster and more complex
  - 5.0 can do 32 GT/s
  - NVIDIA Blackwell first PCIe 6.0 device
  - Your CPU possibly still stuck at PCIe 5.0
- In general PCIe is limiting factor to getting data to GPU.

PCIe Power

- Can power 25W
- additional power connectors to supply can have 75W, 150W and more
- Issues where NVIDIA cards melting

Connectors (Old)

- DDC – i2c bus connection to monitor, giving screen size, timing info, etc.
- RCA – composite/analog TV
- VGA – 15 pin, analog
- DVI – digital and/or analog. DVI-D, DVD-I, DVD-A

Connectors (Newer)

- HDMI – compatible with DVI (though content restrictions). Also audio. HDMI 1.0 – 165MHz, 1080p or 1920x1200 at 60Hz. TMDS differential signaling. Packets. Audio sent during blanking.
- Display Port – similar but not the same as HDMI
- Thunderbolt – combines PCIe and DisplayPort. Intel/Apple. Originally optical, but also Copper. Can send 10W of power.
- LVDS – Low Voltage Differential Signaling – used to connect laptop LCD

Interfaces for 3D Graphics

- OpenGL – SGI (Khronos)
- DirectX – Microsoft (Direct3d)
- Vulkan (sort of next gen OpenGL. Lower level, closer to hardware)
- Metal – from Apple
- WebGL – javascript/web
- OpenGL ES – embedded subset