Vector Processors

C. Scott Ananian

2/15/00

Architectures Anonymous

Chronology:

Cray 1. 1976. (8) 64-word 64-bit Vector registers. 80 MHz.

...

NEC SX-5. 144kB of Vector Registers.

Imagine. 64kB "Stream Register File."

Pentium III. 1k SSE unit.

G4. 4k AltiVec unit.

Playstation 2000. 12k in three sep. FUs.

Design of a supercomputer: CRAY-1

CPU: 230,000 gates.

Memory: 4M 64-bit words

Clock: 80 MHz

Power: 130 kW

Speed: 250

1 “gate time” is 5in microstrip or 1ft twisted pair. 1500 ps. 8 gate times=1 clock cycle. (10% logic used for delay)

Challenges (then):

- System reliability. Cray used 8 bits in each word for a single-error-correction, double-error-detect ECC. Despite high component complexity and 67 of interconnect, 100h MTBI.
- Signal propagation. Three inches of interconnect is not much when you're working with discrete parts. But managed 80MHz clock.
- Cooling. New technologies required.

Future Directions (then):

- Signal propagation: smaller IC packages, smaller traces, better PC board materials.
- Si ECL will be replaced with GaAs.
- Cooling: liquid nitrogen systems; Josephson junctions w/ liquid helium.

From their summary: “Key hardware problems...are heat removal, circuit interconnection, IC packaging, circuit density, and device speed.”

What happened?

Cray was bought by SGI.

The commodity microprocessor entered the market.

Amdahl's law caught up with the compiler technology.

"Mini-vector" processors have proliferated (MMX, AltiVec, MIPS MDMX, etc)

Challenges (now):

- Finding parallelism.
- Finding applications.
- Writing applications.

  (Languages/compilers/programmers.

The Application Problem:

Commodity processor technology (and the cost benefits it entails) has followed general-purpose usage patterns.

Compiler technology has not advanced to the point where we can extract significant amounts of parallelism from ‘gcc’.

The scalar/vector trade-off.

From Hwang: “Vector balance point” of historical vector processors.

- Cray 1S: 0.90
- Cray 2S: 0.93
- IBM 3090/VF: 0.20
- NEC SX-3 (1990): 0.98

This requires some fancy compiler technology to pull off.

There are applications with lots of inherent vector parallelism.

- Weather forecasting.
- Atom bomb testing.

Microsoft word does not have much vector parallelism.

The ‘Imagine’ solution.

The applications market will drive demand for **multimedia-capable** devices.

Imagine’s engine bears *some* similarity to vector processing engines. And in fact, the Imagine ‘kernels’ would deal with typical vector ops very well.

What differentiates a vector supercomputer from a multimedia machine?

- Memory capacity / size of working set.
- Operation granularity / locality.
- Memory and I/O bandwidth (and access modes)
- Application programming environment.

Commodity microprocessors: Intel

The Pentium III SSE instruction set has:

- (8) 128-bit logical registers (64-bit physical); floating-point capable.

Application-specific instructions:

- PMAX/PMIN for speech recognition
- PAVG for video decode
- PSADBW for video encoding

Commodity microprocessors, cont.

Microprocessor Report: “The problem with SIMD engines is that they can easily increase the processing rate to a level above the memory system’s ability to supply data.”

Solution: “memory streaming” -- prefetch, “streaming stores”, etc.

Other commodity uP:

AltiVec:

- (32) 128-bit registers, which can be used as 16-element vector of 1-byte data, 8 x 2-byte, 4 x 4-byte, etc.

Playstation 2000:

- 300MHz superscalar MIPS + 128-bit integer SIMD. Two add'l vector units w/ (32) 128-bit floating point registers which execute independently. 16 parallel pixel processors @ 150 MHz.

Other related solutions:

Digital signal processors. Traditionally, these place a premium on "streaming memory" -- multiple memory ports, no caches, etc. Ex. TMS320C32.

These also have compiler problems.

Questions:

- What are some of the primary reasons of why vector processors are not as popular today as they used to be...are there some fundamental reasons?
- Why don't we see vector processing in desktop PCs? It's twenty-five-year old (at least) technology, and seems like it would be useful to graphics, at least.

More questions:

- I know the new Apple G4's have a "Velocity Engine" that does vector processing, how does it relate to the processors in the readings? How do its capabilities compare?
- How do you program these vector machines?What sort of tricks or stylistic changes must a programmer adopt in order to have programs typically execute more quickly on a vector machine?