Vector Processors

Pratyusa Manadhata pratyus@cs
Vyas Sekar vyass@cs
References

- Hennessy/Patterson Appendix G: Vector Processing Appendix G
Introduction

- High-level operations on vectors - equivalent to an entire loop
- Uses
  - Large scientific applications
  - Multimedia
- Effectively exploiting data parallelism
Vector Processors

- Vector Processor: Scalar pipeline + vector unit
- Compiler makes sure that there are no data hazards in vector
- Architectures
  - vector -register
  - memory-memory vector - not as successful
Vector Processors...

- Use of memory banks to reduce load/store latency
- Strip Mining
- Vector Chaining: equivalent of forwarding
- Vector Mask Control: conditional execution of vector instruction
- Sparse matrices: Scatter-Gather
Vector Processors...

- Multiple Lanes:
  - popular technique to improve vector performance
  - traffic highway
Expensive than MP, SMP

- Sells few dozen copies, high design cost
- Needs fastest and hence expensive memory technology
- Packaging the processor
  - 20GB/s memory, CMOS allows pins at 133MHz => 1200 pins !!
- Relatively few architectural innovations to increase performance retaining low cost.
Semantic Advantages of Vector Processing

- Compact programs
  - Less no of instructions executed
  - Low fetch/decode bandwidth
  - Hiding execution of branches

- Highly parallel implementations

- Vector memory access has no wastage like in cache prefetch

- Low power: once a vector instruction starts operating
  - FU and buses needs to be powered
  - Power off fetch/decode/reorder buffers
Overcoming Limitations of Vector Architectures: CODE approach

- **CODE** = Clustered Organization for Decoupled Execution
- Complexity of central Vector register files
  - N VFUs => 3N access port
  - Area O(N*N), Power O(log N)
- Difficult to implement precise exception
  - In order commit needs large ROB
  - Virtual Memory needs large TLB
- Expensive on chip memory for low latency
Figure 1. The block diagram of CODE. The total number of clusters ($N$) is a design parameter. A cluster contains an integer, floating-point, or load-store VFU. We can also have a cluster without a VFU that merely introduces extra vector registers. The overall number of vector registers across all clusters can be larger than 32.
Clustered Register File

- Each cluster contains a VFU and a set of registers
- The total number of cluster registers must be \( \geq \) number of ISA registers
- Each CLVRF has 5 access ports
  - Note that this is independent of number of clusters
Communication Network

- Separate n/w for transferring vector operands across clusters
- This allows for a separate design decision for the n/w architecture
  - Bus/ring for low-cost processors
  - High b/w network for high end processor
- Tradeoffs: Power, Area, Complexity
Issue logic

- Selects the appropriate cluster to execute the instruction
- Generates inter-cluster transfer instructions
- Tracking of architectural registers using a renaming table
  - Also avoids the WAW/WAR dependencies
- Cluster Selection Policy?
  - Will affect the communication overhead
- The architecture looks similar to a message passing multiprocessor superscalar!!
Precise exception support

- History Buffers
  - Keeps track of the changes to the renaming table
  - Does not store the old vector values only the mappings

- This causes increased register pressure

- Impact on performance?
  - Not too significant from the results shown

- TLB requirement?
  - Allow partial completion of the instruction
Discussion:

- Decoupled execution across clusters
- Synchronize only when queues are full or there are data dependencies
- Hide communication overhead by forcing the output interfaces to look-ahead
- Simplified Chaining mechanism
- Compatible with Multi-Lane implementations
Conclusions

- Vector supercomputers are adapting commodity technology to improve their price-performance.
- Superscalar microprocessor designs have begun to absorb some of the techniques made popular in earlier vector computer systems.
- Multimedia apps require: low power, high performance
- Vectors: most effective way to exploit data-parallelism
Conclusions...

- Vector supercomputers are unviable but the vector instruction set architectures are still useful.
- Future ? new applications, commodity implementations
Thank You !!