Disclaimer: "The contents of this document are scribe notes for The University of Texas at Austin EE382V Spring 2007, Computer Architecture: User System Interplay. The notes capture the class discussion and may contain erroneous and unverified information and comments."*

Fault Tolerance in Merrimac

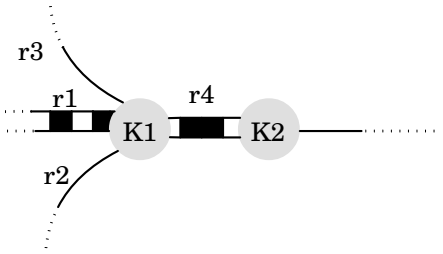
Lecture #8: Wednesday, 14 February 2007
Lecturer: Dr. Mattan Erez
Scribe: Senthil K Chellappan
Reviewer: Min Kyu Jeong

The class began with a brief overview of stream processors and fault tolerance. The discussion progressed with the problem, solution and its users but was more centered around evaluation of the problem and solution. During the discussion, many questions about stream computing, fault detection and critical evaluation flaws were asked and clarified. Few of them are presented in the Miscellaneous section and few are interleaved in the respective sections.

1 Overview

- "Stream", strictly speaking, is an infinite stream of data elements with heads and tails continuously emptied and filled. However, in the context of the paper, stream is not an infinite sequence of words but a very long sequence of blocks.
- Kernel, in the context of the paper, is a long loop of code that consumes blocks of data, takes processing time and produces blocks of output. Kernels represent bulk computation and are usually loops that process blocks and produces new blocks (to and from the same level of the storage hierarchy). To run a kernel, a bulk load between storage hierarchy levels (e.g., off-chip to on-chip) has to be performed.
- When kernels are cascaded and stream of data is consumed by each kernel, numerous temporary variables are produced. Both the data produced by one kernel and that consumed by the cascaded kernel can be at different rates.
- Applications expose this producer-consumer locality (dataflow locality, remember – WaveScalar) at a coarser granularity, i.e., the short term producer-consumer locality is encapsulated in the kernels.

- The architecture is primarily used for throughput oriented computation both in terms of compute and memory system. Batch processing more than interactive processing.
- Usage model in the paper: Scientific Applications, Finite element/volume methods, fluid/aerodynamics, data/image processing or in general, bulk operations – without tight data/control interdependencies.



■ Blocks of data
K1, K2 - Kernels
r1-r4 - data production/consumption rate

- **Soft-Errors:** Soft errors occur on data nodes in the system but the system does not change physically. Soft errors originate out of various sources (e.g., radiation, noise, interference) and cause unpredictable failures. Soft error rate (SER) is the rate at which soft errors occur in a system.

2 Problems being solved

- Soft-Error Fault Tolerance for compute-intensive computers.
- Stream processors are compute intensive and hence are more prone to silent errors compared to the control intensive superscalars. Silent error, for example, could occur when computations may not affect control flow or generated addresses but a floating-point result that may go unnoticed.
- Fault Tolerance while saving off-chip bandwidth and on-chip memory.
- Cost Efficiency.

3 Intended Users and the Usage Model

- Supercomputer Architects and Users.
- Programmers needing Fault Tolerance.

- Kernel Developers.
- Scientific Application Users.
- VLSI Designers.
- **Usage Model:** Long running non-interactive batch jobs is assumed.

4 Uniqueness

Unique features of the solution are listed below and other suggestions that were categorized not unique are listed as non-unique items.

- A balance of hardware and software redundancy.
- Reconfigurable – Allows to simply change a state machine through the addition of local switches and comparators.
- Reconfigurability at an user level.
- Configures Fault Tolerance scheme dynamically? Yes.
- A solution for compute intensive applications.
- Takes advantage of the coarser-grained stream execution model to apply effective fault checking in the softare system (kernel replication).
- Suggestions that were concluded not unique:
  - Reliability with error correcting codes and fault exception for 2 errors.
  - AVF (Architecture Vulnerability Factor) - not unique.

5 Evaluation

- Using photographs for area estimation. What does that mean?
  - *Answer: One can identify structures similar to the design in existing chips and use existing die photos to estimate the area. For example, Imagine has similar architecture to Merrimac and has actually been built. The data was used in conjunction with scaling rules with technology and error could have been 10%. Similarly, IBM Cell processor uses the same XDR-DRAM that Merrimac was designed to use.*
- How was the cluster size 4x4 chosen, any evaluation?

  - *Answer: A rough choice based on applications. This decision was made very early in the work. A later work that has not been published addresses the issue and arrives at an optimal cluster size between 4 and 16. The tradeoff is mostly between shared state with a larger intra-cluster switch to partitioned state and a smaller switch (with a larger inter-cluster switch that scales better).*

- No fault injection.
  - *Clarification: All single errors will be caught. Probability of missing a fault is too less and coverage was 100%.*
- No change in granularity or any other model during comparison.
- No analysis of usage model. i.e., in which situation (batch vs. single) would the user use the solution.
  - *If the job duration takes 2 days to execute and at the end of <4 days, re-execution based results fail?*
  - *Solution: Do checkpoint-to-checkpoint re-execution after every specified time slice.*
- Phase behavior of programs considered?
  - *Yes. But the feature was not tested or evaluated.*
- Slowdown in recovery is 2-3%.
- Comparison against other varied stream models was not done.

6 Evaluation inline with user requirements?

- Yes and No.
  - Paper does not discuss recovery mechanism.
    - Clarification: Paper mentions checkpoint based recovery. In the paper, fault tolerance is only at fault detection level.
- Compare Cell vs. Merrimac – *Not done due to space constraints.*
- Why is power important and why is it not discussed in this context?
  - *Importance: Power hurts in terms of supply and cooling in a data-center. Question is always power or performance. Power limits all segments of the market. Microsoft and Google are building clusters in relatively cooler geographical areas and places with hydro and nuclear power sources.*

  - *Discussed in this context?: New mechanisms don't consume more power, so power and performance improvements have gone together.*

- SER related to temperature?

  - *No. Particle strikes, which is the main source of soft-errors are not affected by temperature (the particles are typically cosmic rays).*
  - *No way to know this ahead of time.*
  - *Noise induced errors are more likely as temperature increases.*

7 Technology a factor in the problem or solution?

- Problem: Susceptibility of SER with shrinking technology.
- I/O pins don't scale with technology (bandwidth impact).
- More the storage, better locality and better performance.

8 New Tools or S/W Techniques

The technique did not introduce new tools as such, however, it brought few new algorithms to optimize compilers and existing tools.

- Kernel replication is a new compiler technique that is based on the stream execution model.

9 Other Users affected

- Usage model is restricted, other usage models (not long running batch jobs) will require different techniques. An assumption in the paper is that the workload is a single job for a longer duration of time. Simple usage model is not common.
- Binary Compatibility is affected.

  - *Clarification: Stream processing is not a solution targetting binary compatibility. It is more application specific and the relatively smaller number of applications can be recompiled with dedicated compilers.*

- Area overheads.
- Depends on which granularity user expects.

10 Generalization in the context of the class

- Evaluation using real models or estimations. Is it good to quantify or analyze?
- A hardware-software hybrid technique. Keep in mind: software costs a lot!

11 Miscellaneous

- **Programming Stream Processors:** Programming stream processors is a burden than normal C programming. Stream computers are good alternative if applications expose parallelism and locality. An analogical example in a superscalar context: If the program's working-set does not fit in the cache, software owns the burden to pull the data into the cache.
- **Usage Model:** Well suited for a single job that runs for a long period of time.
- **Question:** What is register pressure?
  - *Answer: Register pressure occurs when temporary state exceeds available registers and requires spill/refill code. When more temporary variables are produced, reload would cause performance overheads.*
- No RTL or real model as such?
  - A cycle accurate Merrimac stream simulator was used and the applications were compiled with own software chain. Tools similar to CacTI were used.

12 Conclusion

Few aspects concerning architects with respect to fault tolerance are whether to design for performance or fault tolerance. Where to draw the line in redundancy between hardware and software for fault tolerance and how to leverage the advantages of both? Compute intensive applications provide architects an opportunity to exploit locality and parallelism. The paper utilizes a hybrid approach to fault tolerance which is dynamically hardware configurable and saves the user from that trouble. The paper does a fairly minimal analysis to prove the core point but utilizes results obtained by prior work. This is a key point for the class, esp., when students develop solutions to problems that have already been worked on. Dr. Erez emphasized that granularity need not necessarily be fine grained as it may not always be optimal. It was also accepted that the paper had its flaws as mentioned in the above sections. Few of them were discussed and clarified in the class. Overall, the paper serves as a good introduction to hybrid approach to fault tolerance and explicitly answers all the questions for the discussion.