Sparse Linear Solver for Power System Analyis using FPGA *

J. R. Johnson† P. Nagvajara‡ C. Nwankpa§

1 Extended Abstract

Load flow computation and contingency analysis is the foundation of power system analysis. Numerical solution to load flow equations are typically computed using Newton-Raphson iteration, and the most time consuming component of the computation is the solution of a sparse linear system needed for the update each iteration. When an appropriate elimination ordering is used, direct solvers are more effective than iterative solvers. In practice these systems involve a larger number of variables (50,000 or more); however, when the sparsity is utilized effectively these systems can be solved in a modest amount of time (seconds). Despite the modest computation time for the linear solver, the number of systems that must be solved is large and current computation platforms and approaches do not yield the desired performance. Because of the relatively small granularity of the linear solver, the use of a coarse-grained parallel solver does not provide an effective means to improve performance. In this talk, it is argued that a hardware solution, implemented in FPGA, using fine-grained parallelism, provides a cost-effective means to achieve the desired performance.

Previous work [1, 2, 3] has shown that FPGA can be effectively used for floating-point intensive scientific computation. It was shown that high MFLOP rates could be achieved by utilizing multiple floating-point units,

*This work was partially supported by DOE grant #ER63384, PowerGrid - A Computation Engine for Large-Scale Electric Networks

†Department of Computer Science, Drexel University, Philadelphia, PA 19104. email: jjohnson@cs.drexel.edu

‡Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA 19104. email: nagvajara@ece.drexel.edu

§Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA 19104. email: chika@nwankpa.ece.drexel.edu

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 FEB 2005	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Sparse Linear Solver for Power System Analysis using FPGA	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Drexel University**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM00001742, HPEC-7 Volume 1, Proceedings of the Eighth Annual High Performance Embedded Computing (HPEC) Workshops, 28-30 September 2004 Volume 1., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**UU**	**21**	

and FPGA could outperform PCs and workstations, running at much higher-lock rates, on dense matrix computations. The current work argues that similar benefit can be obtained for the sparse matrix computations arising in power system analysis. These conclusions are based on operation counts and system analysis for a collection of benchmark systems arising in practice.

Benchmark data indicates that between 1 and 3 percent of peak floating point performance was obtained using a state-of-the-art sparse solver (UMFPACK) running on 2.60 GHz Pentium 4. The solve time for the largest system (50,092 unknowns and 361,530 non-zero entries) was 1.39 seconds.

A pipelined floating point core was designed for the Altera Stratix family of FPGAs. An instantiation of the core on an Altera Stratix with speed rating (-5) operates at 200 MHz for addition and multiplication and 70 MHz for divion. Moreover, there is sufficient room for 10 units. Assuming 100% utilization of eight FPUs, the projected performance for the FPGA implementation is 0.069 seconds, which provides a 20-fold improvement. While it is optimistic to assume perfect efficiency, hard-wired control should provide substantially better efficiency than available with a standard processor. Moreover, analysis of the LU factorization shows that the average number of updates per row throughout the factorization is 20.3, which provides sufficient parallelism to benefit from 8 FPUs. An implementation, and more detailed model, is being carried out to determine the attainable efficiency.

References

[1] J. Johnson, P. Nagvajara, and C. Nwankpa. High-Performance Linear Algebra Processor using FPGA. In *Proc. High Performance Embedded Computing (HPEC 2003)*. Sept. 22-24, 2003.

[2] K. Underwood. FPGAs vs. CPUs: Trends in Peak Floating-Point Performance. In *Proc. International Symposium on Field-Programmable Gate Arrays (FPGA 2004)*. Feb. 22-24, 2004.

[3] Ling Zhuo and Viktor K. Prasanna. Scalable and Modular Algorithms for Floating-Point Matrix Multiplication on FPGAs. In *Proc. International Parallel and Distributed Processing Symposium (IPDPS 2004)*, 2004.

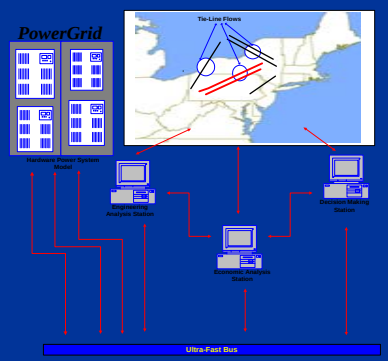
Poster

Agenda

Sparse Linear Solver for Power System Analysis using FPGA

Jeremy Johnson, Prawat Nagvajara, Chika Nwankpa

Drexel University



HPEC 2004

Goal & Approach

- To design an embedded FPGA-based multiprocessor system to perform high speed Power Flow Analysis.
- To provide a single desktop environment to solve the entire package of Power Flow Problem (Multiprocessors on the Desktop).
- Solve Power Flow equations using Newton-Raphson, with hardware support for sparse LU.
- Tailor HW design to systems arising in Power Flow analysis.

Results

- Software solutions (sparse LU needed for Power Flow) using high-end PCs/workstations do not achieve efficient floating point performance and leave substantial room for improvement.
- High-grained parallelism will not significantly improve performance due to granularity of the computation.
- FPGA, with a much slower clock, can outperform PCs/workstations by devoting space to hardwired control, additional FP units, and utilizing fine-grained parallelism.
- Benchmarking studies show that significant performance gain is possible.
- A 10x speedup is possible using existing FPGA technology

Application Specific Computing Research Group

Abstract

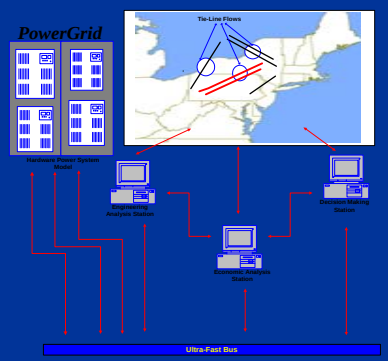
Agenda

CEPE

Sparse Linear Solver for Power System Analysis using FPGA

Jeremy Johnson, Prawat Nagvajara, Chika Nwankpa

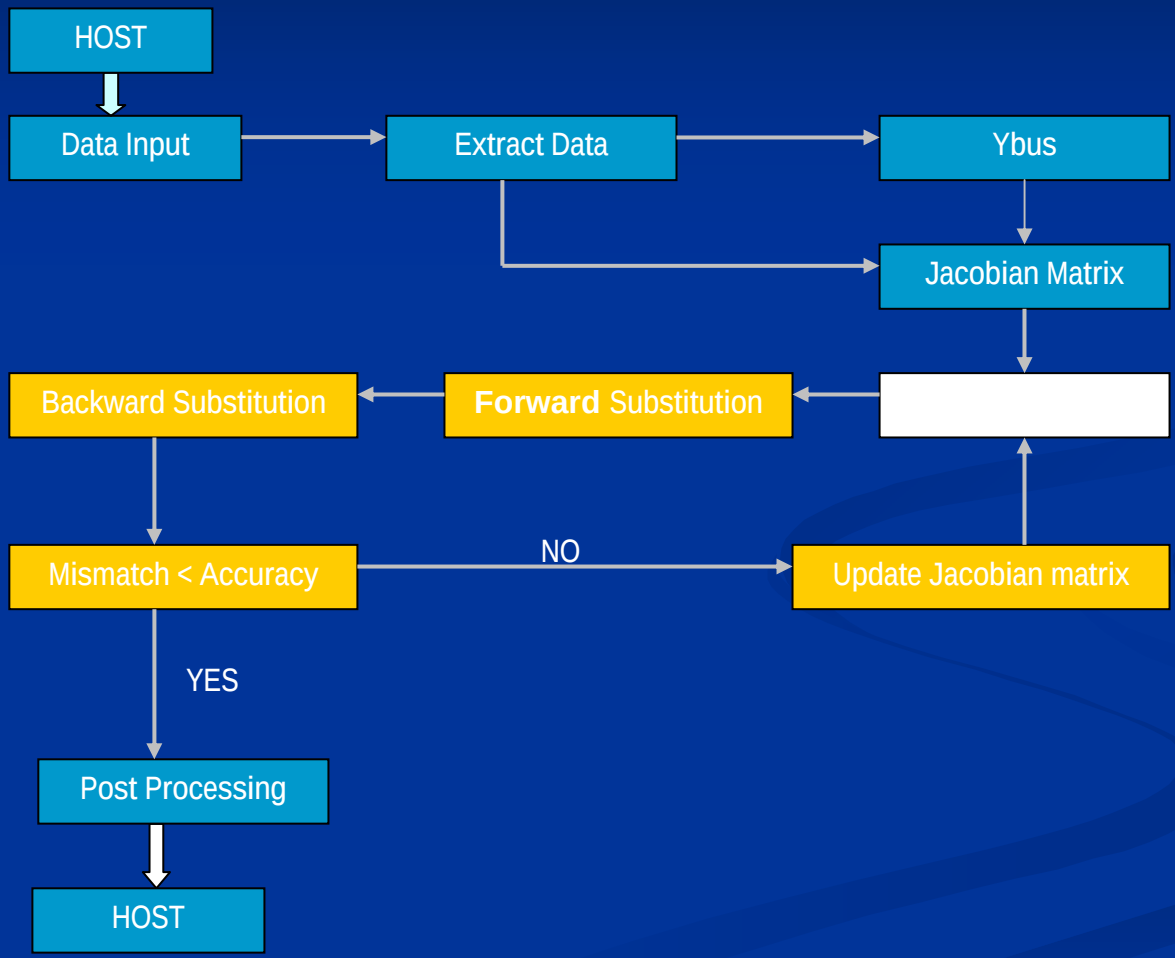
Drexel University



Goal & Approach

- To design an embedded FPGA-based multiprocessor system to perform high speed Power Flow Analysis.
- To provide a single desktop environment to solve the entire package of Power Flow Problem (Multiprocessors on the Desktop).
- Solve Power Flow equations using Newton-Raphson, with hardware support for sparse LU.
- Tailor HW design to systems arising in Power Flow analysis.

Algorithm and HW/SW Partition



Results

- Software solutions (sparse LU needed for Power Flow) using high-end PCs/workstations do not achieve efficient floating point performance and leave substantial room for improvement.
- High-grained parallelism will not significantly improve performance due to granularity of the computation.
- FPGA, with a much slower clock, can outperform PCs/workstations by devoting space to hardwired control, additional FP units, and utilizing fine-grained parallelism.
- Benchmarking studies show that significant performance gain is possible.
- A 10x speedup is possible using existing FPGA technology

Benchmark

- Obtain data from power systems of interest

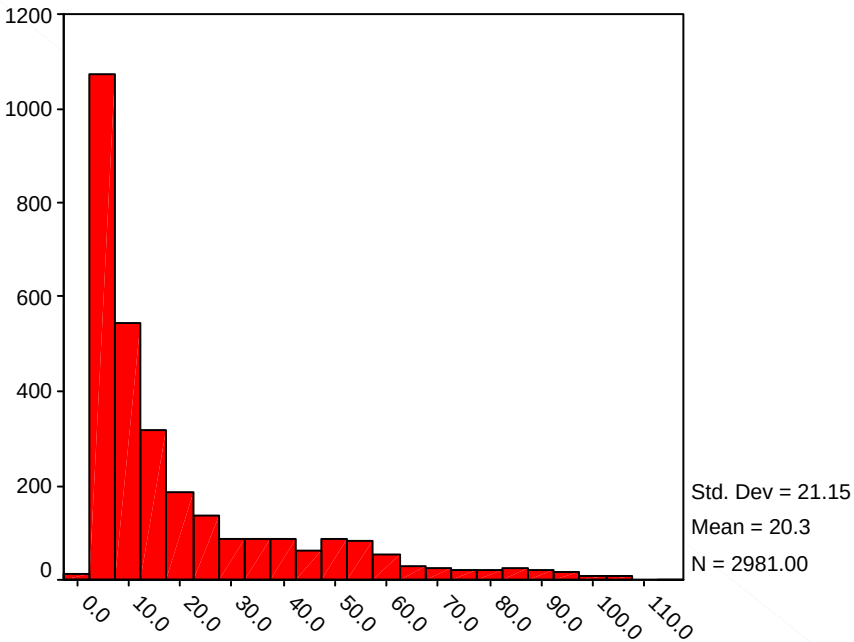
Source	# Bus	Branches/Bus	Order	NNZ
PSSE	1,648	1.58	2,982	21,682
PSSE	7,917	1.64	14,508	108,024
PJM	10,278	1.42	19,285	137,031
PJM	26,829	1.43	50,092	361,530

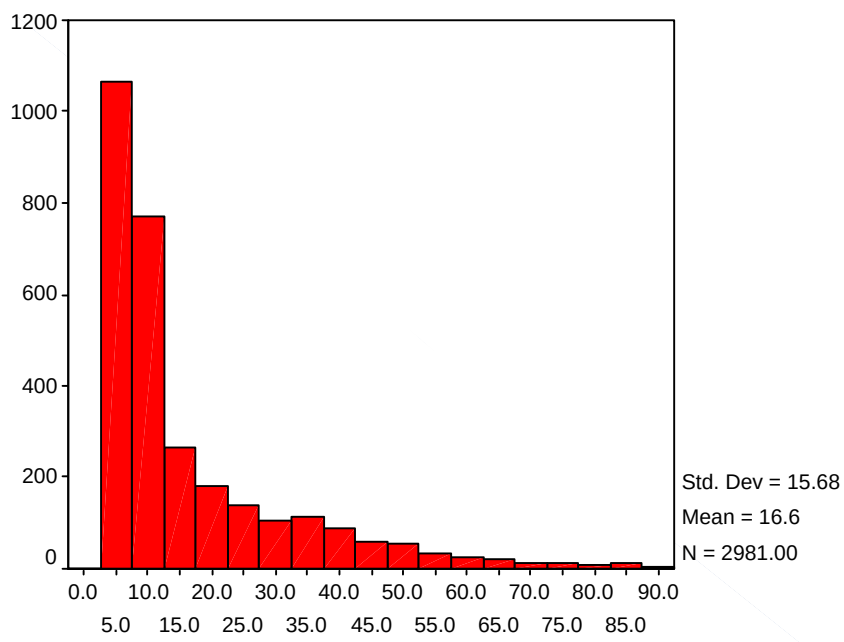
System Profile

# Bus	# Iter	#DIV	#MUL	#ADD	NNZ L+U
1,648	6	43,876	1,908,082	1,824,380	108,210
7,917	9	259,388	18,839,382	18,324,787	571,378
10,279	12	238,343	14,057,766	13,604,494	576,007
26,829		770,514	90,556,643	89,003,926	1,746,673

System Profile

- More than 80% of rows/cols have size < 30





Software Performance

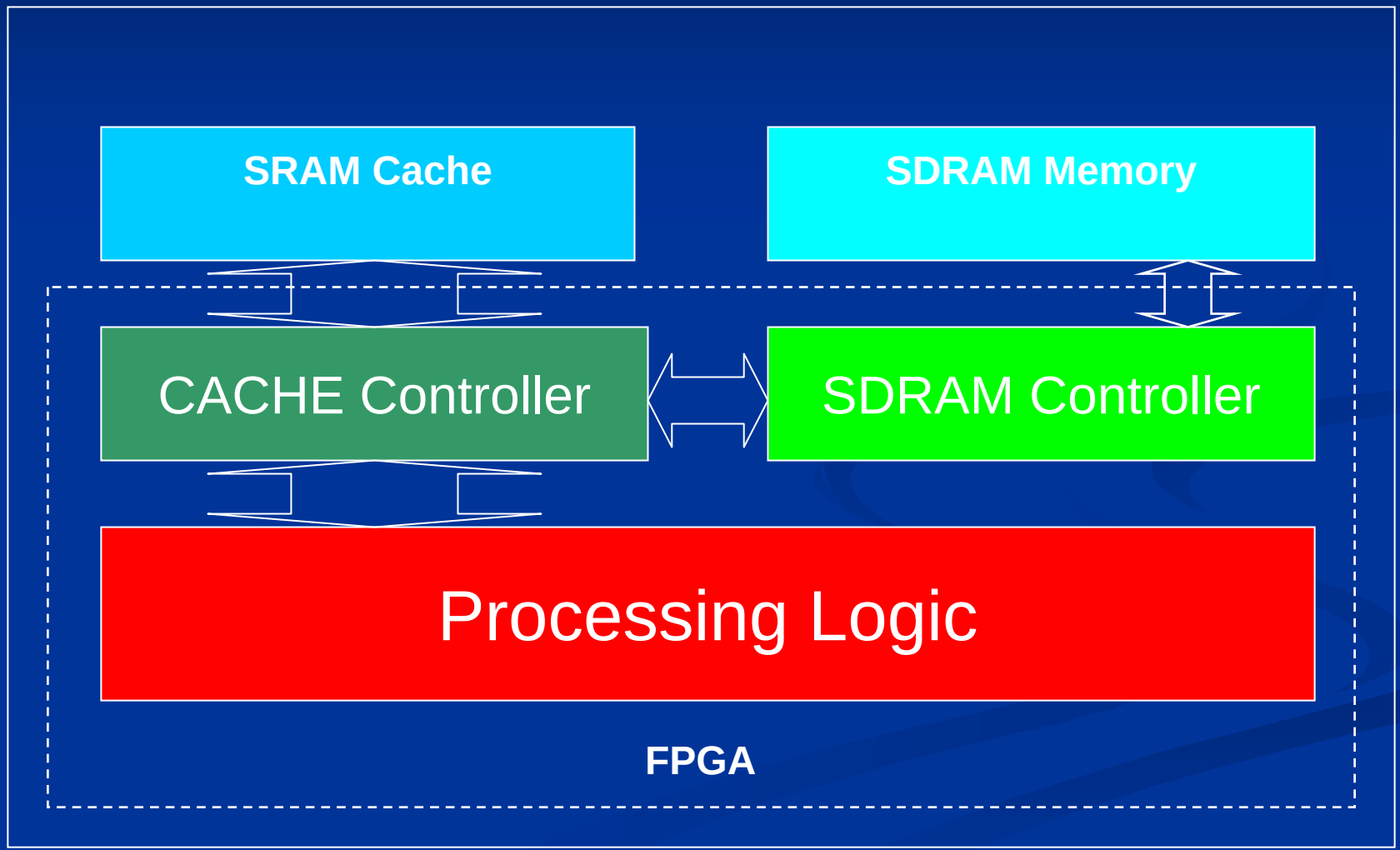
- Software platform
  - UMFPACK
  - Pentium 4 (2.6GHz)
  - 8KB L1 Data Cache
  - Mandrake 9.2
  - gcc v3.3.1

# Bus	Time	FP Eff
1,648	0.07 sec	1.05%
7,917	0.37 sec	1.33%
10,278	0.47 sec	0.96%
26,829	1.39 sec	3.45%

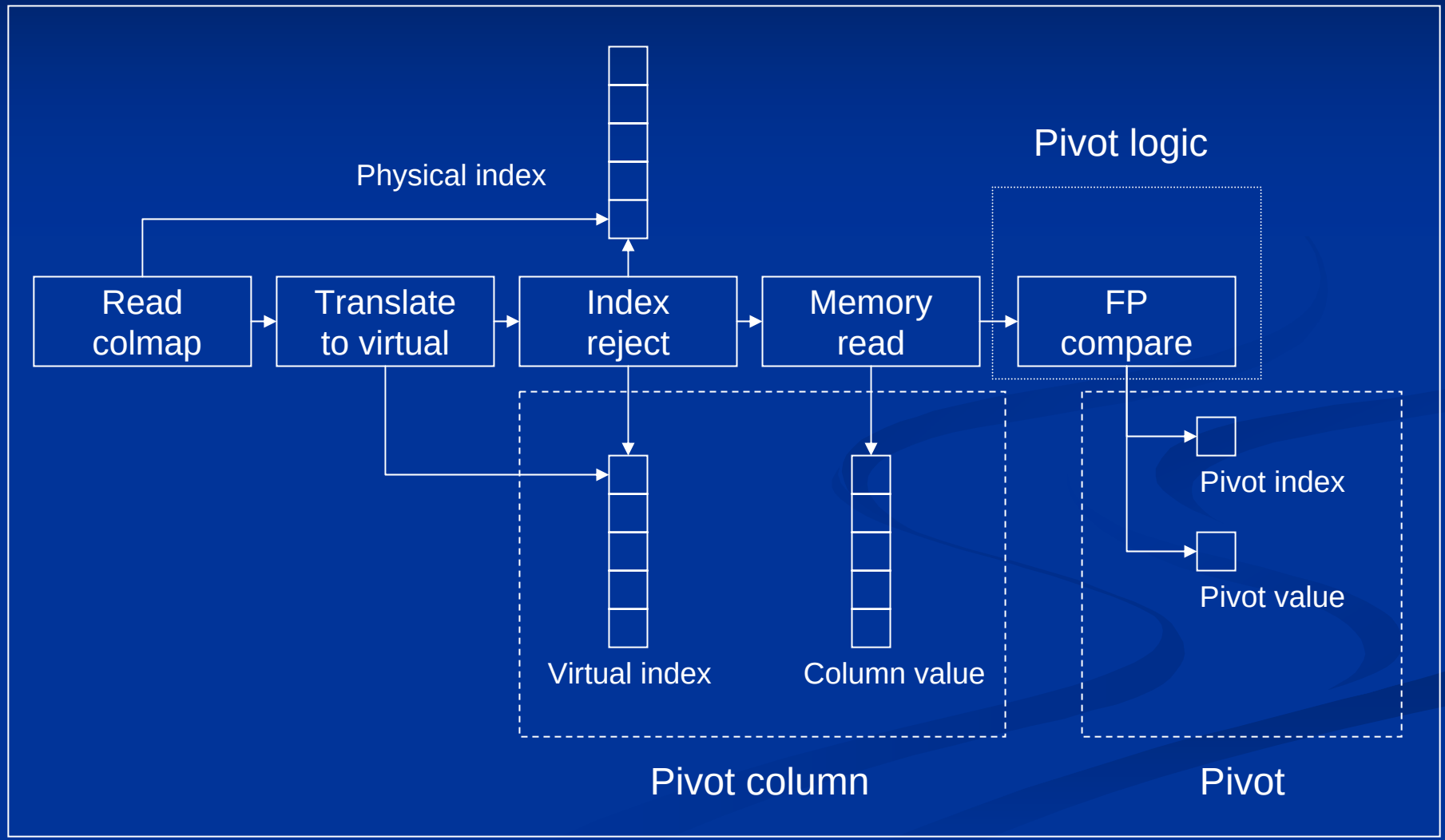
Hardware Model & Requirements

- Store row & column indices for non-zero entries
- Use column indices to search for pivot. Overlap pivot search and division by pivot element with row reads.
- Use multiple FPUs to do simultaneous updates (enough parallelism for 8 – 32, avg. col. size)
- Use cache to store updated rows from iteration to iteration (70% overlap, memory ≈ 400KB - largest). Can be used for prefetching.
- Total memory required ≈ 22MB (largest system)

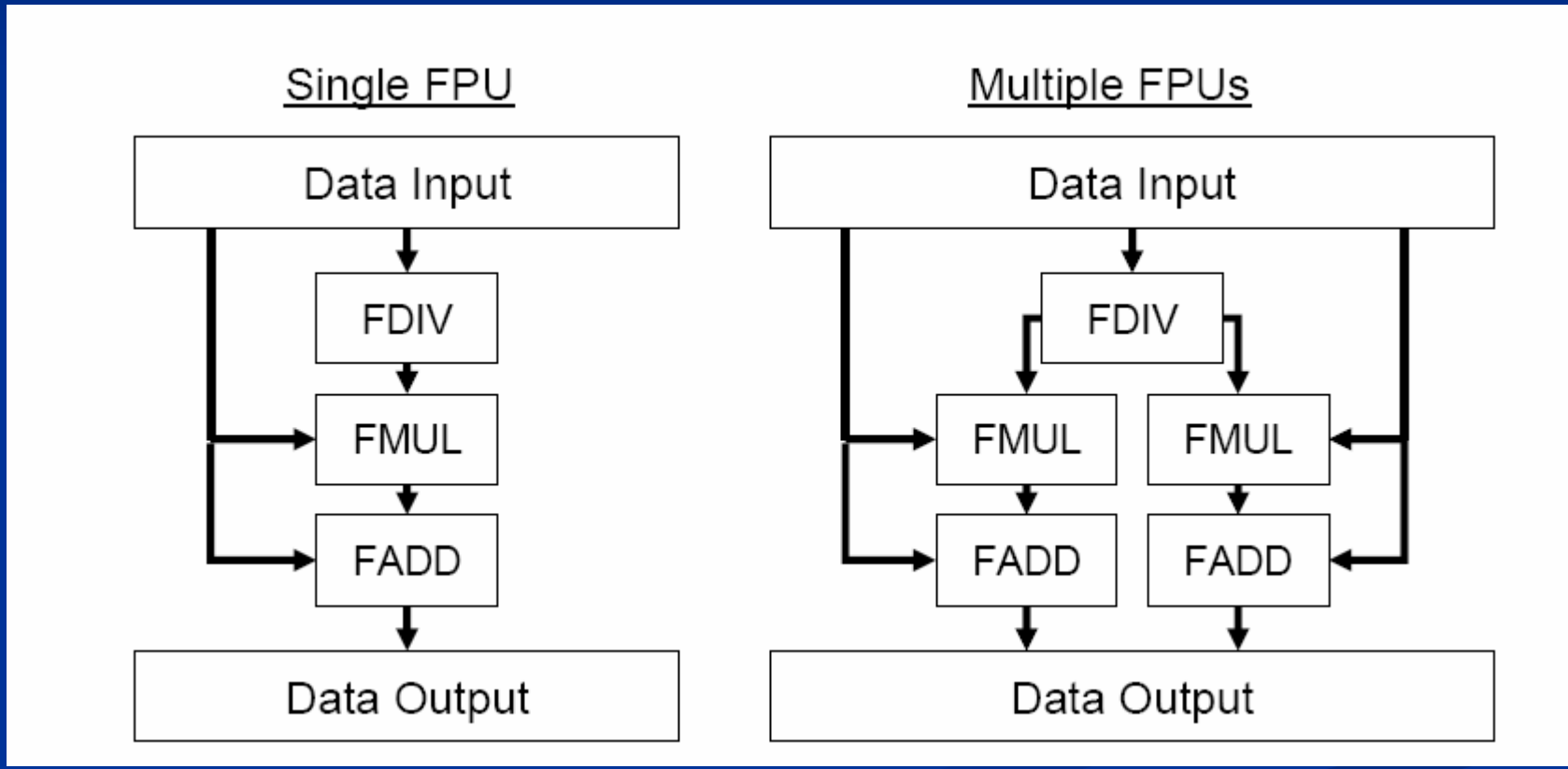
Architecture



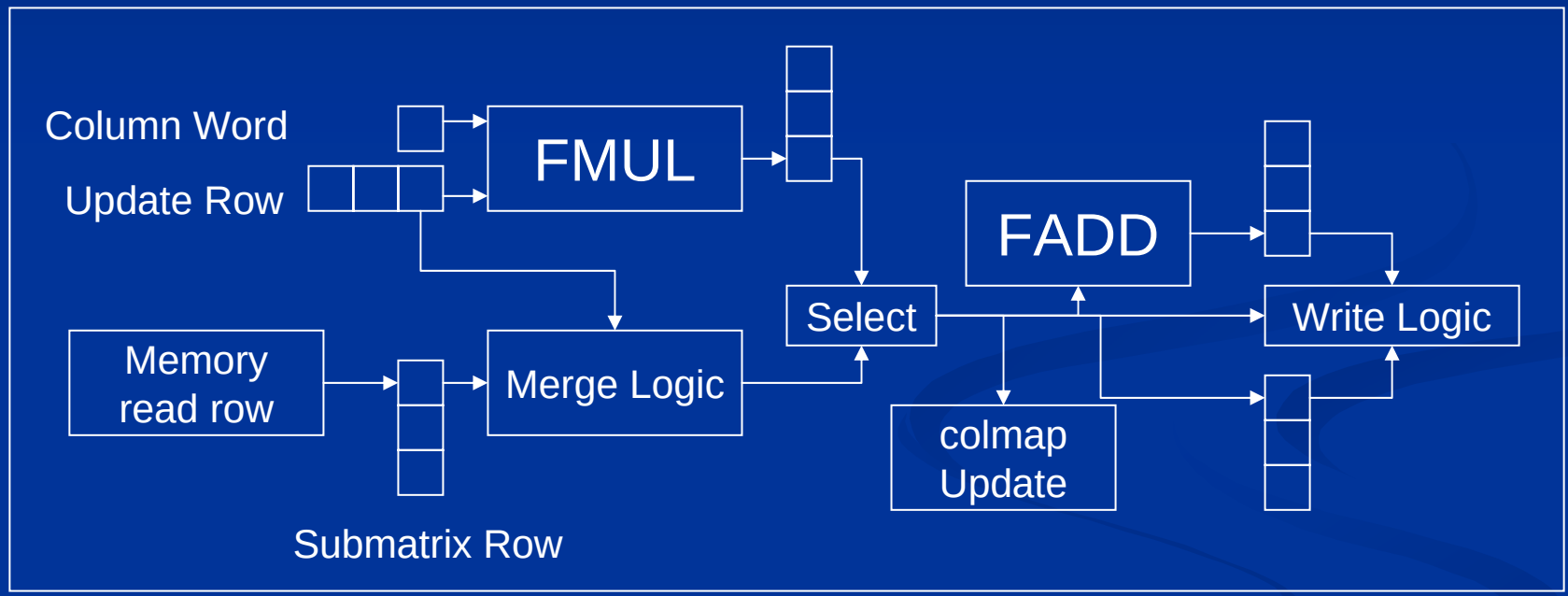
Pivot Hardware



Parallel FPUs



Update Hardware



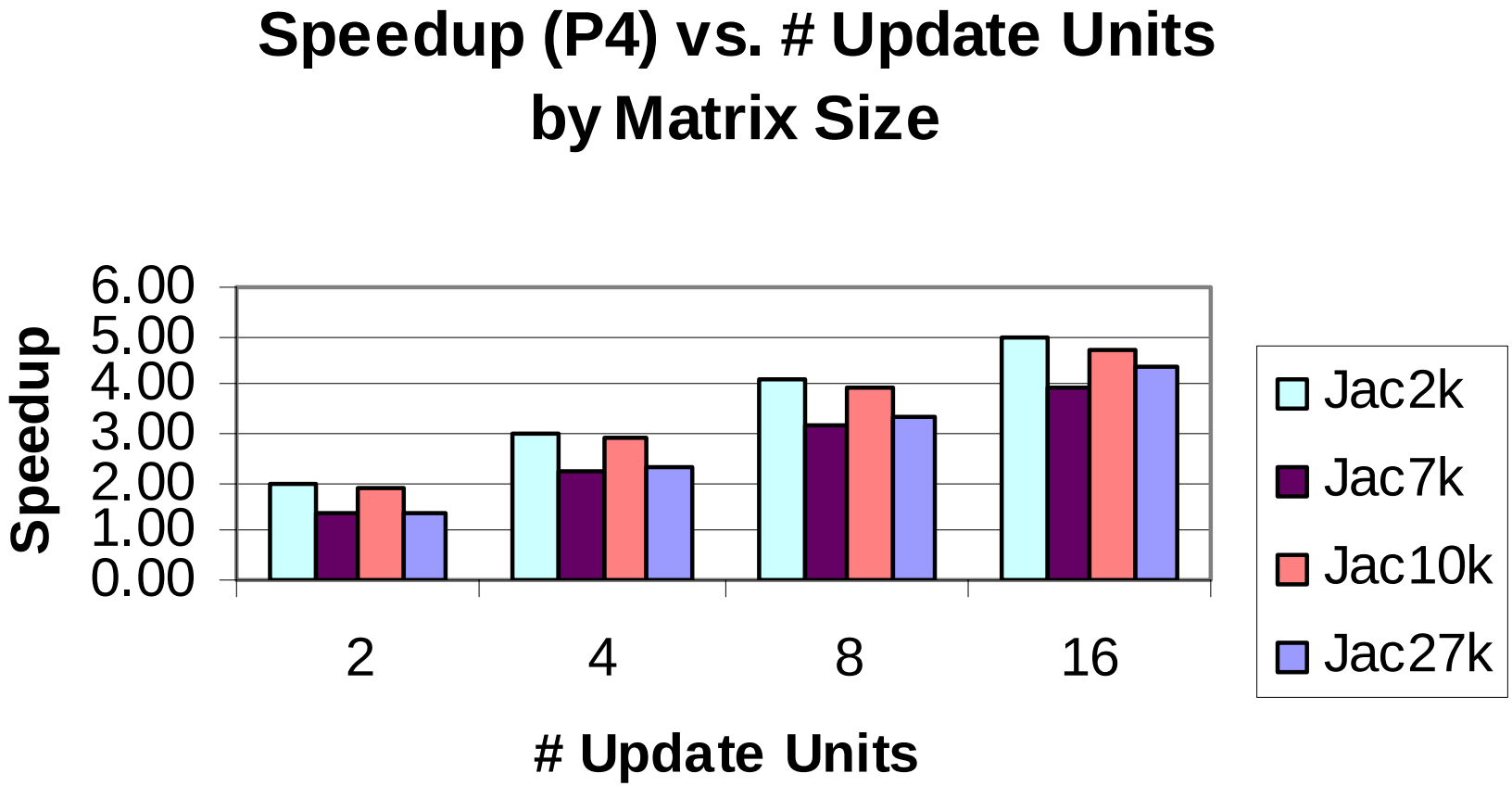
Performance Model

- C program which simulates the computation (data transfer and arithmetic operations) and estimates the architecture's performance (clock cycles and seconds).

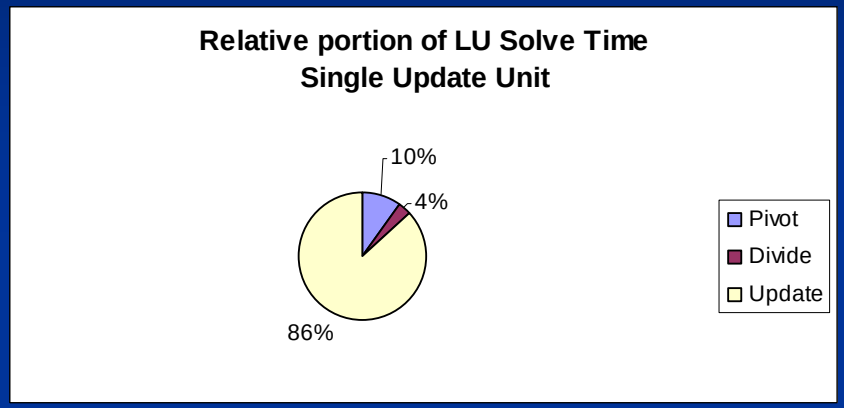
Model Assumptions

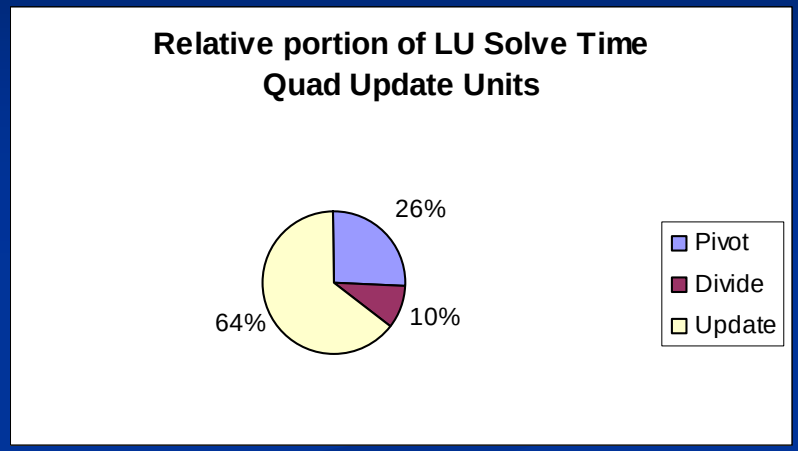
- Sufficient internal buffers
- Cache write hits 100%
- Simple static memory allocation
- No penalty on cache write-back to SDRAM

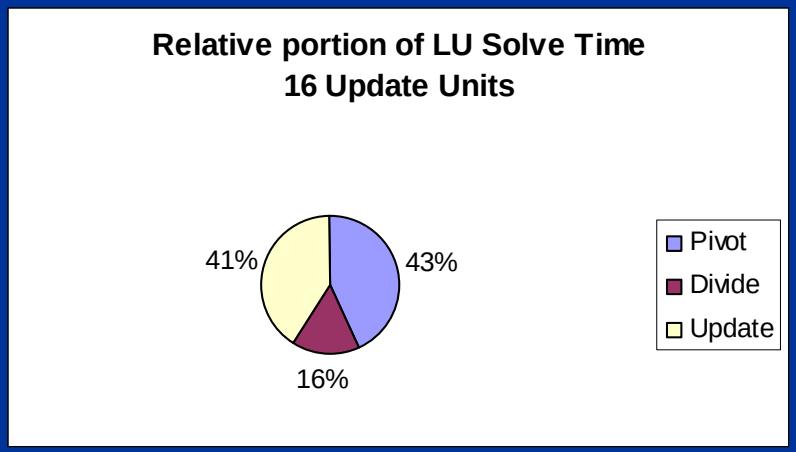
Performance



GEPP Breakdown







Cycle Count
By # of Update Units

	1	4	16
Pivot	259401	259401	259401
Divide	96439	96439	96439
Update	2409312	642662	248295