The performance of GRAPE-DR for dense matrix operations

Jun Makino, TiTech,
Hiroshi Daisaka, Hitotsubashi Univ.
Toshiyuki Fukushige, KFCR,

Yutaka Sugawara, Mary Inaba and Kei Hiraki, U. Tokyo

ICCS2011 Jun 1-3, 2011, Singapore

Talk structure

- GRAPE-DR
- Matrix multiplication
- LU decomposition
- Parallel LU decomposition
- Summary

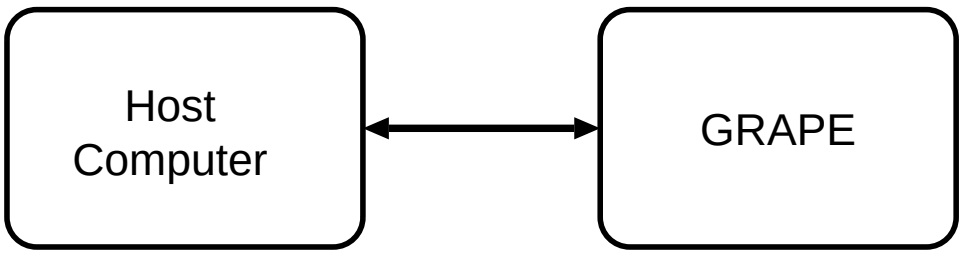
GRAPE-DR

- Accelerator for HPC
- Development: FY2004-2008
  (U-Tokyo+NAOJ+...)

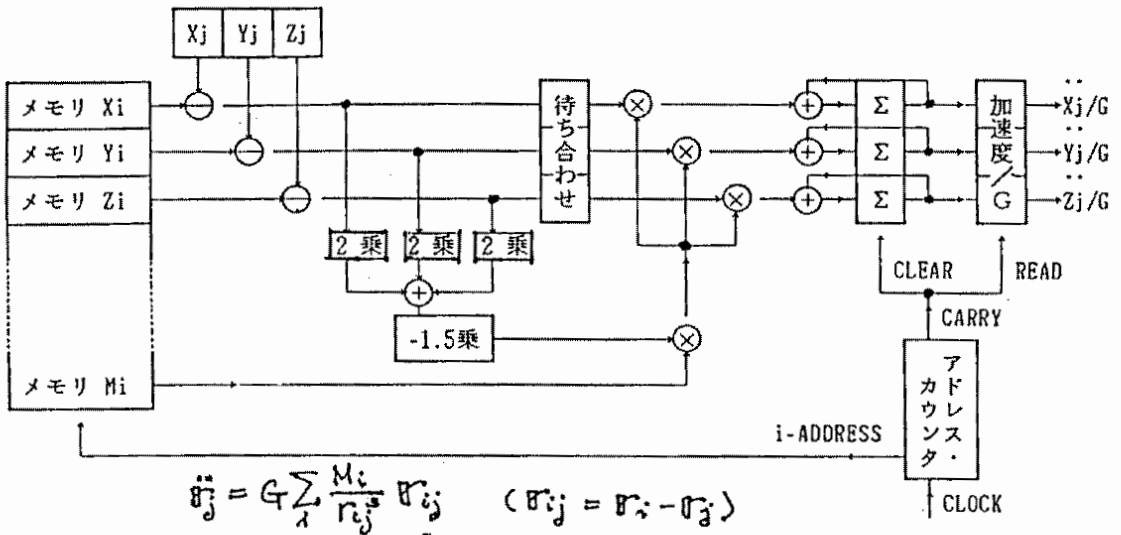
  I moved from UT to NAOJ in 2006 and to TiTech in 2011
- "Follow-up" for GRAPE (GRAvity PipE), special-purpose computer for gravitational many-body problems
- New architecture — wider application range than previous GRAPEs

Basic concept of GRAPE

- **With N-body simulation, almost all calculation goes to the calculation of particle-particle interaction.**
- **This is true even for schemes like Barnes-Hut treecode or FMM.**
- **A simple hardware which calculates the particle-particle interaction can accelerate overall calculation.**
- **Original Idea: Chikada (1988)**



Chikada's idea (1988)



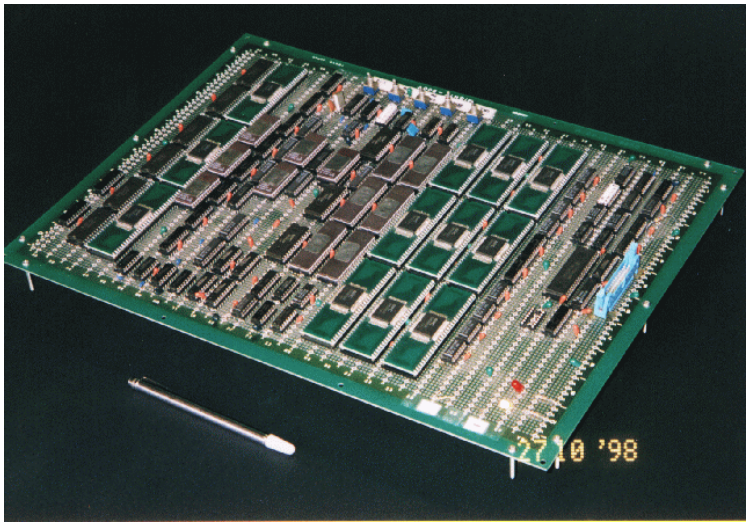
$$\ddot{\vec{r}}_j = G\sum_i \frac{M_i}{r_{ij}^3}\vec{r}_{ij} \qquad (\vec{r}_{ij} = \vec{r}_i - \vec{r}_j)$$

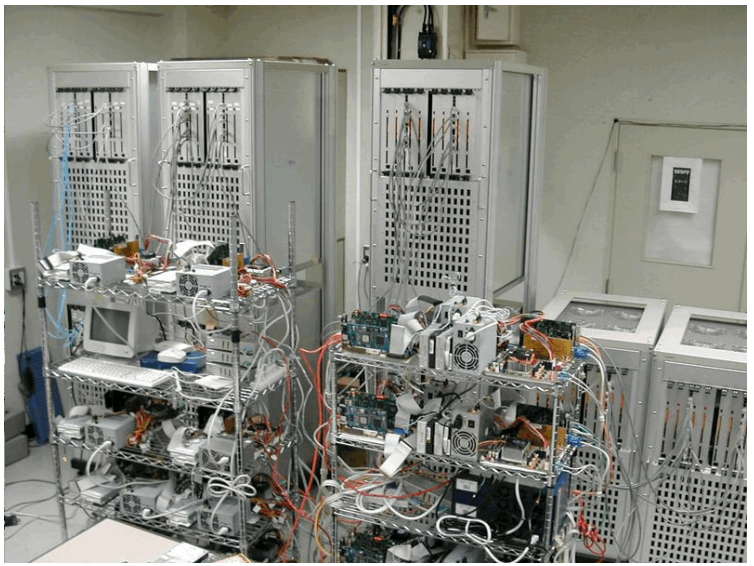
+, −, ×, 2乗は1 operation, -1.5乗は多項式近似でやるとして10operation 位に相当する.
総計24operation.
各operation の後にはレジスタがあって, 全体がpipelineになっているものとする.
「待ち合わせ」は2乗してMと掛け算する間の時間ズレを補正するためのFIFO(First-In First-Out memory).
「Σ」は足し込み用のレジスタ. N回足した後結果を右のレジスタに転送する.

図2. N体問題のj-体に働く重力加速度を計算する回路の概念図.

- Hardwired pipeline for force calculation (similar to Delft DMDP)
- Hybrid Architecture (things other than force calculation done elsewhere)

GRAPE-1 to GRAPE-6





GRAPE-1: 1989, 308Mflops

GRAPE-4: 1995, 1.08Tflops

GRAPE-6: 2002, 64Tflops

From GRAPE-6 to GRAPE-DR

Chip development cost has become too high.

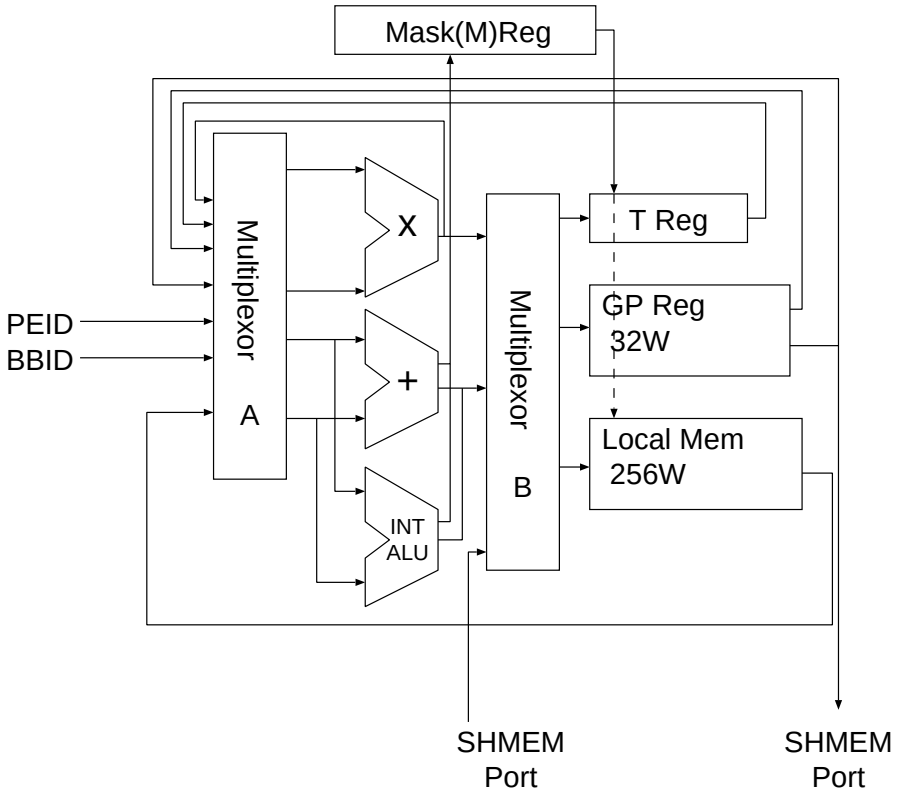
Year	Machine	Chip initial cost	process
1992	GRAPE-4	200K$	1μm
1997	GRAPE-6	1M$	250nm
2004	GRAPE-DR	4M$	90nm
2011?	GDR2?	$>$ 10M$	40nm?

How to deal with high initial cost?

Several options:

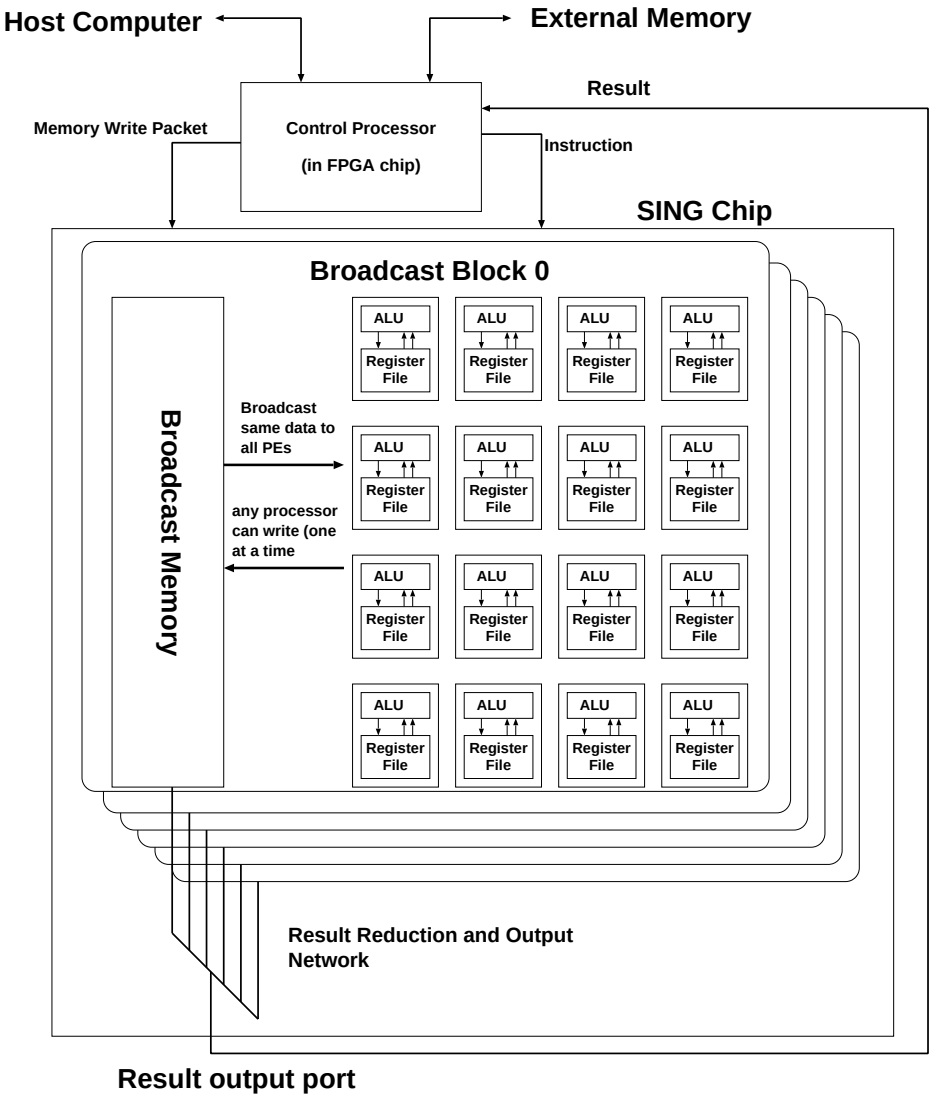
- Forget about making hardware, use x86 or GPU
- Use FPGA
- Develop hardware with wider range of application — our decision
  - an SIMD processor chip with very large number of processing cores (512)
  - simple on-chip network (broadcast/reduction tree)
  - particle-particle interaction, dense matrix operation, and other computationally expensive applications

GRAPE-DR Processor architecture



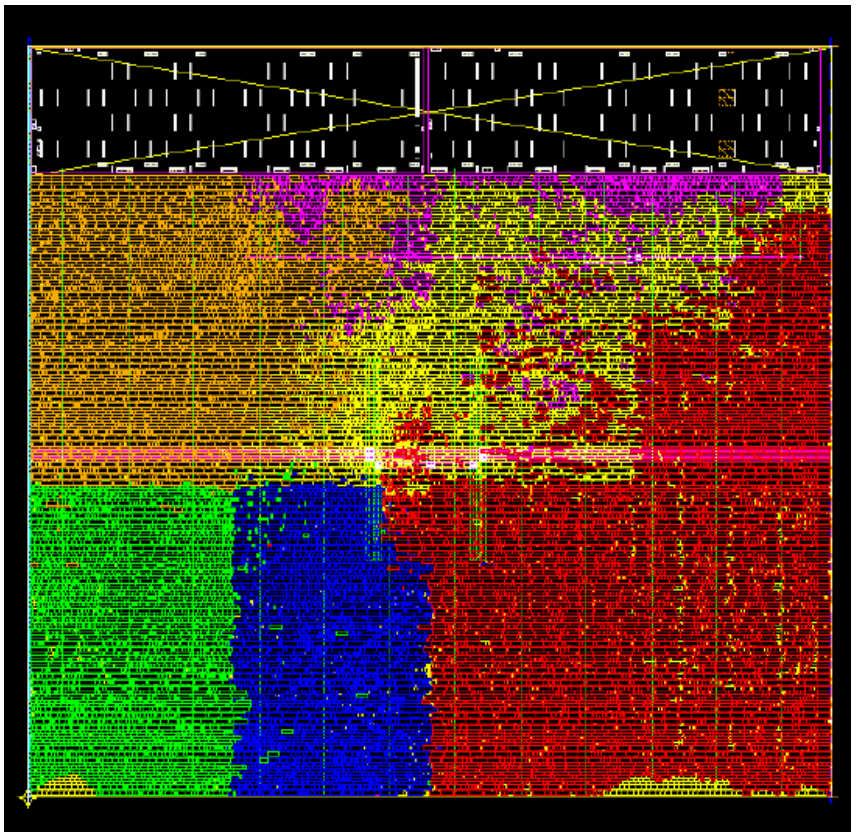
- DP Float Mult
- DP Float add/sub
- Integer ALU
- 32-word registers
- 256-word memory
- communication port

Chip architecture



- 32 PEs organized to "broadcast block" (BB)
- BB has shared memory.
- Input data is broadcasted to all BBs.
- Outputs from BBs go through reduction network (sum etc)

PE Layout

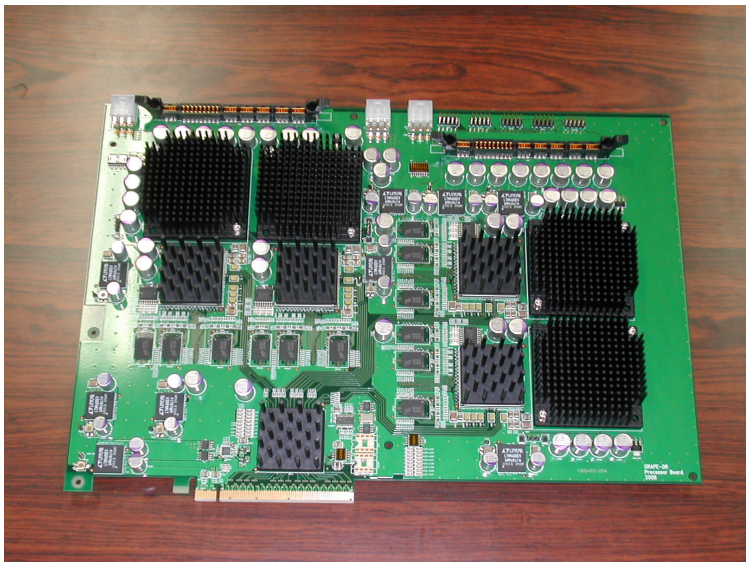


Black: Local Memory
Red: Reg. File
Orange: FMUL
Green: FADD
Blue: IALU

0.7mm by 0.7mm
800K transistors

0.1W@400MHz
800Mflops/400Mflops
peak (SP/DP)

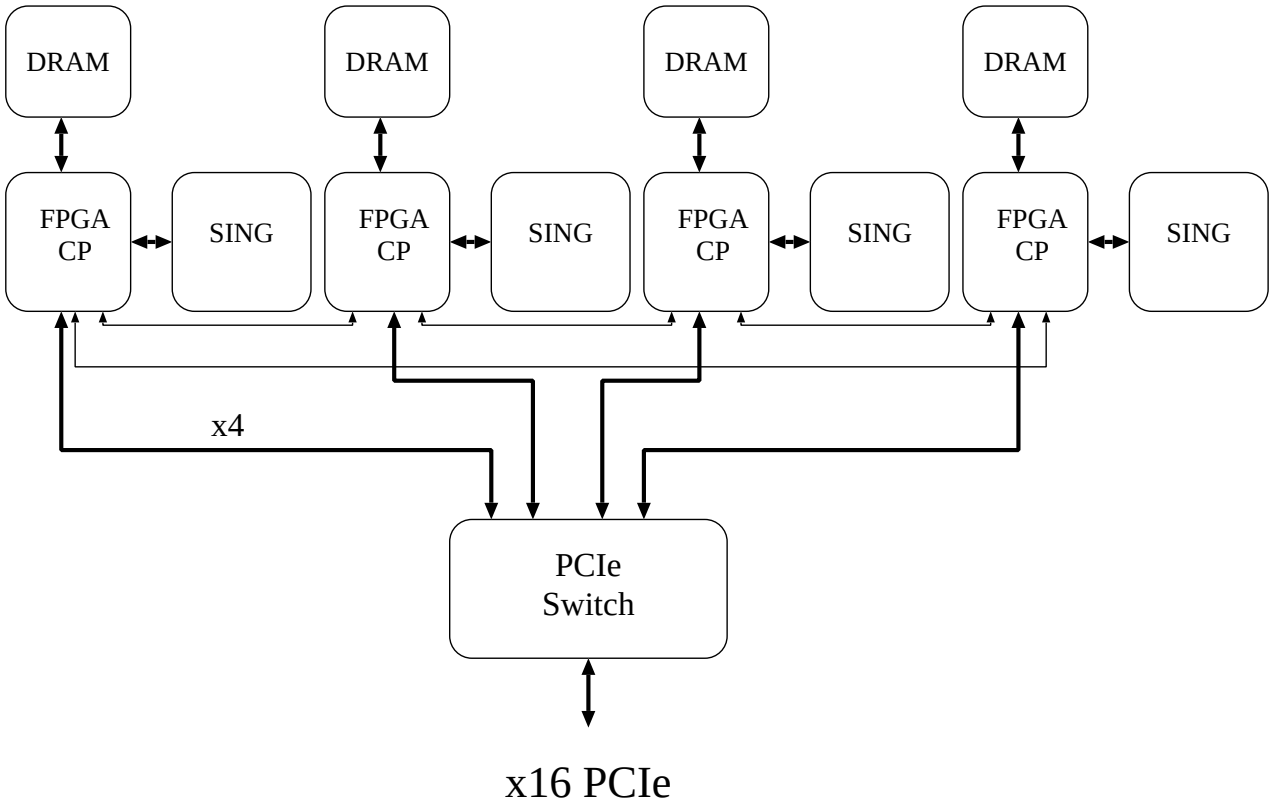
Processor board



PCIe x16 (Gen 1) interface
Altera Arria GX as DRAM controller/communication interface

- Around 200-250W power consumption
- 819Gflops DP peak (400MHz clock)
- Available from K&F Computing Research (www.kfcr.jp)

Processor board



4 FPGAs are connected in a bidirectional ring (used for broadcast/reduction)

Performance for Dense matrix operations

Accelerators can make DGEMM (matrix-matrix multiplication) fast.
Two practical problems

- **The actual efficiency of DGEMM**
  - **kernel efficiecy**
  - **communication/startup overhead**
- **Overall efficiency**
  - **Operations other than DGEMM (Amdahl's law)**

DGEMM implementation

Calculate: $C \rightarrow C + A \times B$, conceptually we do:

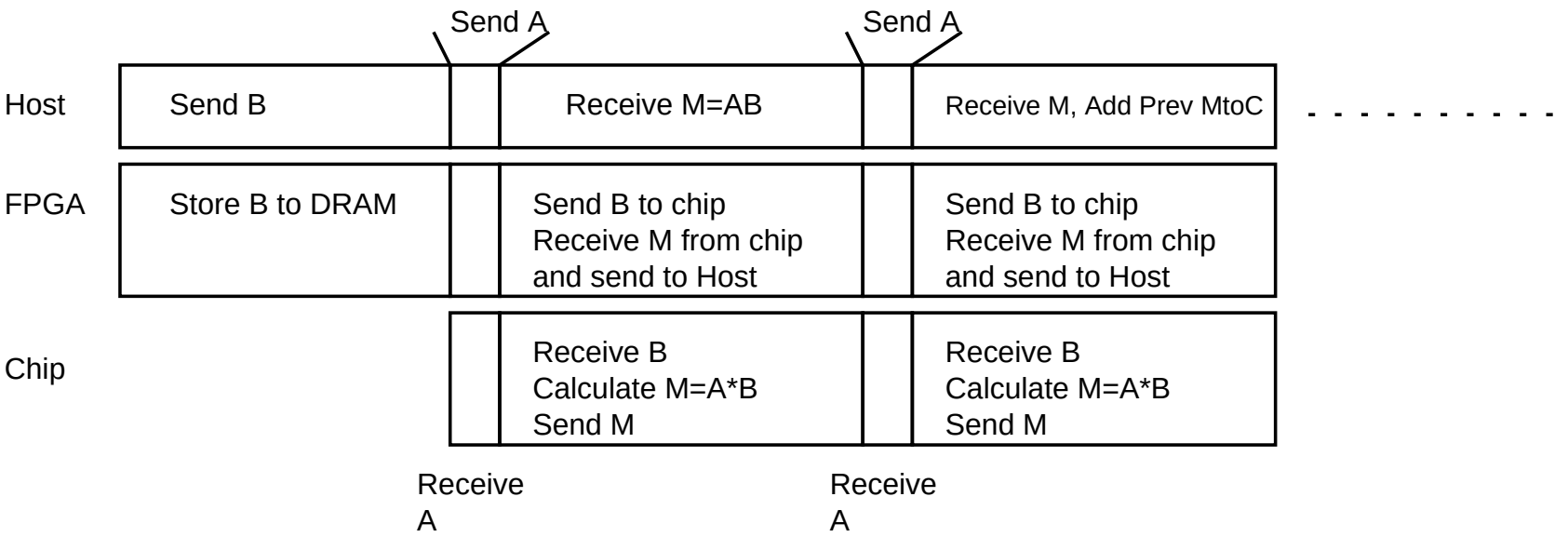
1. **Store B to on-board memory of GRAPE-DR**
2. **Load (part of) A to on-chip memory**
3. **load b (one vector of B) to registers of**
4. **calculate $m = A \times b$**
5. **output m (directly from register to PCIe interface)**

Steps 3-5 are done concurrently. In addition, addition ($C \rightarrow C + M$) is done on host CPU, also concurrently

Details:

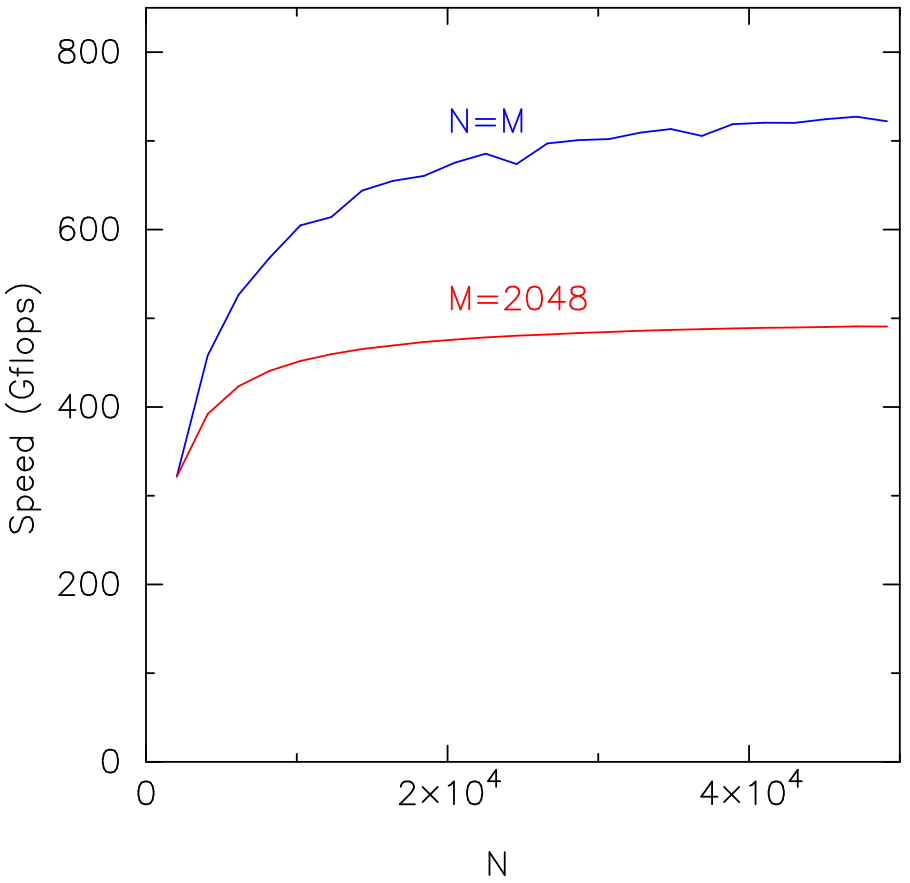
- **Each processing core stores 32×8 matrix and length 32 vectors**
- **Summation of 16 partial products on different cores is done by hardware adder tree, and thus no additional overhead**
- **Further summation of 4 results from 4 chips is also done in adders in FPGAs**

Calculation timechart



- **Transfers of A and B from host are not hidden**
- **Everything else is done concurrently with calculation**
- **We made transfer of A hidden, but X58 chipset became unstable...**

DGEMM performance



M=N, K=2048: 722 Gflops (88% peak)
N=K=2048, 490 Gflops

FASTEST single-card performance on the planet.

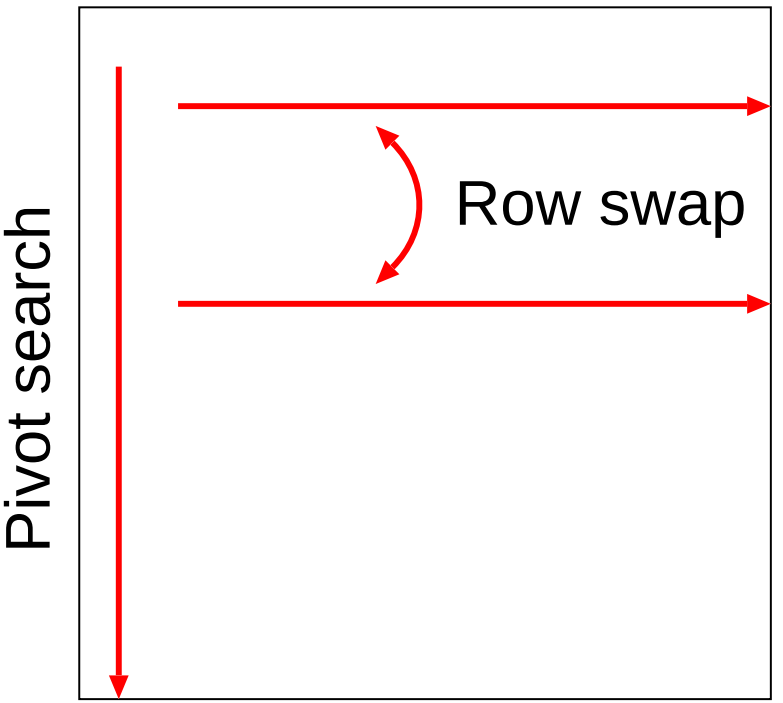
Fermi: 300Gflops (60% peak)
AMD Cypress: 470Gflops (87% peak)

LU-decomposition tuning

Almost every previously known techniques

- Use large block (NB=2048)
- right-looking form
- TRSM converted to GEMM

Problem: row swap is very slow – stride access



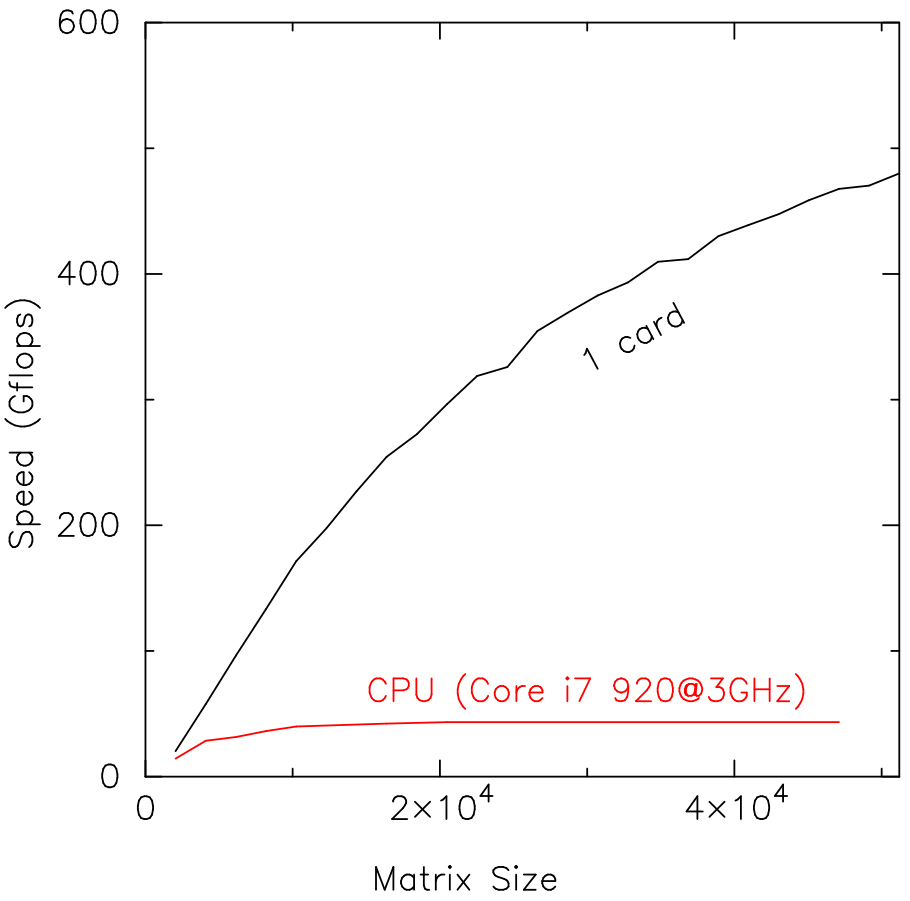
Accelerate row-swapping

- Use row-major order to make row swapping fast
- Transpose matrix during recursive column decomposition to make pivot search and narrow band matrix operation fast

Some other tunings, such as

- Use recursive scheme for TRSM (calculation of L^{-1})

LU-decomposition performance



Speed in Gflops as function of Matrix size

Top: GRAPE-DR
Bottom: host CPU

480 Gflops (58% of theoretical peak) for N=50K

x11 speedup over host CPU

HPL (parallel LU)

- **Everything done for single-node LU-decomposition**
- **Both column- and row-wise communication are hidden**
- **TRSM further modified: calculate LT^{-1} instead of $T^{-1}U$**
- **x2 performance compared to HPL 1.04a**

HPL performance

Date	N	# Nodes	Speed Efficiency	Green500
Jun 2010	**240K**	**64**	**24TF** **50%**	**#1** **(Little)**
Nov 2010	**432K**	**81**	**37.4TF** **56%**	**#2**

Comparison with other works

(From Nov 10 Top 500 list)

Accelerator /System	CPU /Clock	Performance /Efficiency	Acceleration over host
Fermi Tianhe-1A	Xeon 6c 2.93GHz	2.566PF 54.4%	2.83
Fermi Tsubame 2.0	Xeon 6c 2.93(3.19?) GHz	1.192PF 53.5%	6.13
GRAPE-DR	Core i7 4c 3GHz	37.4TF 53.2%	10.6

Similar efficiency with much higher acceleration ratio.

Dark side of tuning...

- X58 DMA performance seems to be limited to 6.4GB/s (sum of upstream and downstream, theoretical limit is 19.2GB/s)
- It starts to drop data silently when busy.
  - PIO write
  - DMA write

Workaround we used:

- Do not use PIOW
- Do not use DMA read and write concurrently

Similarity and Difference with GPUs

	GRAPE-DR	GPU (Fermi)
SIMD	Yes	Yes
Design rule	90nm	40nm
# FPUs	512	448
Memory bandwidth	$\sim$ 5GB/s	$>$ 100GB/s
# transistors	400M	3G
Peak DP performance	205GF	515 Gflops
Power consumption	50W	250W
Performance per watt	4.0GF/W	2.1GF/W
DGEMM Efficiency	$\sim$ 90%	$\sim$ 60%

Similarity and Difference with GPUs

- Both GRAPE-DR and GPUs achieved very high performance (and performance per watt) using SIMD many-core architecture
- The design of GRAPE-DR is much more extreme, with 1/10 transistors per FPU.
- Part of the reason of this difference is the limited memory bandwidth.
- Reduction in transistor count resulted in high performance/W.

Summary

- GRAPE-DR is an SIMD accelerator for scientific computing
- With 90nm technology, one GRAPE-DR chip integrates 512 cores and provides 205Gflops (Double precision)
- In our DGEMM implementation, all data transfers, except the transfer of input matrices from host to GRAPE-DR card, are hidden.
- 4-chip card DGEMM performance 722 Gflops, LU decomposition $\sim$ 500Gflops
- Accelerators require new algorithms, not just porting and tuning

Detailed breakdown of calculation time

Nswap=0 cpsec = 184.784 wsec=108.456 488.994 Gflops
swaprows time= 5.09831e+09 ops/cycle=0.181402
scalerow time= 1.3279e+08 ops/cycle=6.9647
trans rtoc time= 3.79496e+09 ops/cycle=0.243703
trans ctor time= 2.42686e+09 ops/cycle=0.381087
trans mmul time= 2.74357e+09 ops/cycle=5.05642
tr nr cdec time= 3.68971e+09 ops/cycle=0.250655
trans vvmul time= 7.16809e+08 ops/cycle=1.29022
trans findp time= 2.97246e+09 ops/cycle=0.311138
solve tri u time= 5.95504e+09 ops/cycle=7.22212e-06
solve tri time= 4.00307e+10 ops/cycle=94.6313
trans mmul8 time= 9.15249e+08 ops/cycle=8.08387
trans mmul4 time= 4.9365e+08 ops/cycle=7.49393
trans mmul2 time= 1.33296e+09 ops/cycle=1.38765

Detailed breakdown of calculation time (cont'd)

DGEMM2K time= 2.77404e+11 ops/cycle=184.353
DGEMM1K time= 1.75294e+10 ops/cycle=54.0258
DGEMM512 time= 1.64471e+10 ops/cycle=28.7905
DGEMMrest time= 3.16284e+10 ops/cycle=14.9713
col dec t time= 1.26994e+10 ops/cycle=2.33042
Total time= 3.65573e+11 ops/cycle=145.072

Next-Generation GRAPE

Question:
Any reason to continue hardware development?

- **GPUs are fast, and getting faster**
- **FPGAs are also growing in size and speed**
- **Custom ASICs practically impossible to make**

Next-Generation GRAPE

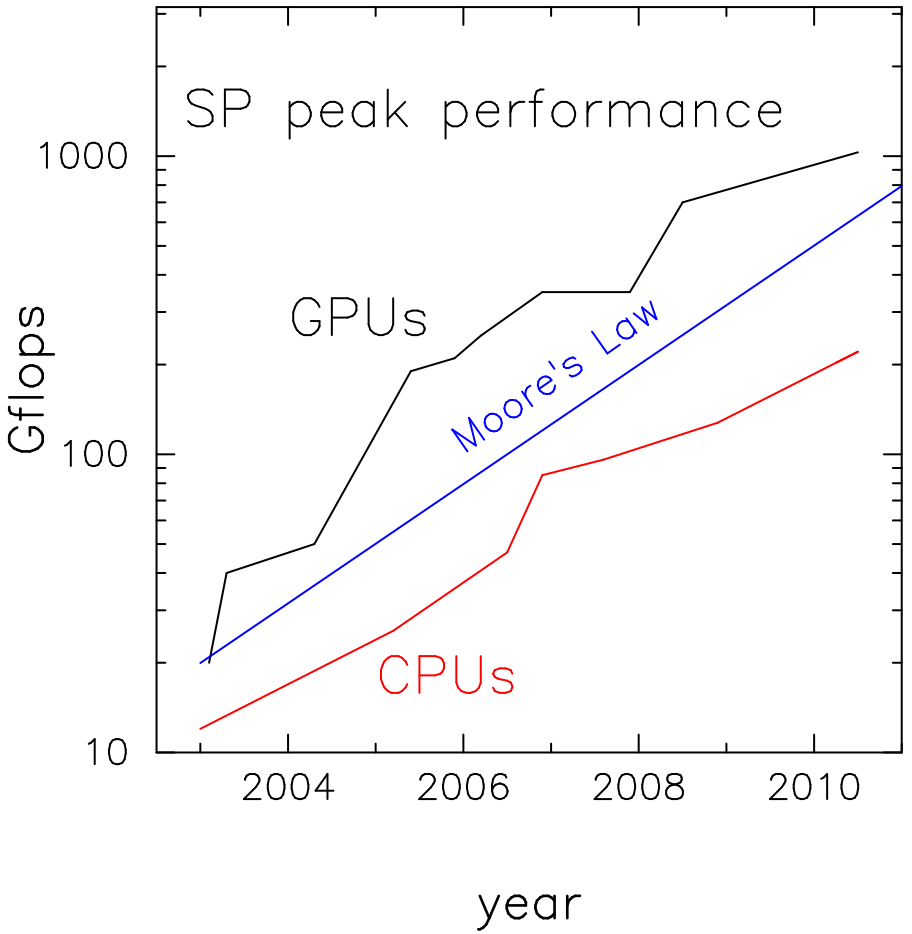
Question:
Any reason to continue hardware development?

- **GPUs are fast, and getting faster**
- **FPGAs are also growing in size and speed**
- **Custom ASICs practically impossible to make**

Answer?

- **GPU speed improvement might have slowed down**
- **FPGAs are becoming far too expensive**
- **Power consumption might become most critical**
- **Somewhat cheaper way to make custom chips**

GPU speed improvement slowing down?



Clear "slowing down" after 2006 (after G80)

Reason: shift to more general-purpose architecture

Discrete GPU market is eaten up by unified chipsets and unified CPU+GPU

But: HPC market is not large enough to support complex chip development

FPGA

"Field Programmable Gate Array"

- **"Programmable" hardware**
- **"Future of computing" for the last two decades....**
- **Telecommunication market needs: large and fast chips (very expensive)**

Power Consumption

1kW · 1 year ∼ 1000 USD

You (or your institute) might be paying more money for electricity than for hardware.

Special-purpose hardware is quite energy efficient.

Chip	Design rule	Gflops/W
GRAPE-7(FPGA)	65nm	> 20
GRAPE-DR	90nm	4
GRAPE-6	250nm	1.5
Tesla C2050	40nm	< 2
Opteron 6128	45nm	< 1.2

Structured ASIC

- **Something between FPGA and ASIC**
- **eASIC: 90nm (Fujitsu) and 45nm (Chartered) products.**
- **Compared to FPGA:**
  - **3x size**
  - **1/10 chip unit price**
  - **non-zero initial cost**
- **Compared to ASIC:**
  - **1/10 size and 1/2 clock speed**
  - **1/3 chip unit price**
  - **1/100 initial cost ($>$ 10M USD vs $\sim$ 100K)**

GRAPEs with eASIC

- Completed an experimental design of a programmable processor for quadruple-precision arithmetic. 6PEs in nominal 2.5Mgates.
- Started designing low-accuracy GRAPE hardware with 7.4Mgates chip.

Summary of planned specs:

- around 8-bit relative precision
- 100-200 pipelines, 300-400 MHz, 2-5Tflops/chip
- small power consumption: single PCIe card can house 4 chips (10 Tflops, 50W in total)

Will this be competitive?

Rule of thumb for a special-purpose computer project:

> **Price-performance goal should be more than 100 times better than that of a PC available when you start the project.**

— x 10 for 5 year development time
— x 10 for 5 year lifetime

Compared to CPU: Okay
Compared to GPU: ??? (Okay for electricity)

Will this be competitive?

Rule of thumb for a special-purpose computer project:

> **Price-performance goal should be more than 100 times better than that of a PC available when you start the project.**

— x 10 for 5 year development time
— x 10 for 5 year lifetime

Compared to CPU: Okay
Compared to GPU: ??? (Okay for electricity)

Will GPUs exist 10 years from now?