ECM on Graphics Cards

Tanja Lange

Department of Mathematics and Computer Science

Technische Universiteit Eindhoven

tanja@hyperelliptic.org

09.10.2008

joint work with Daniel J. Bernstein (UIC), Tien-Ren Chen (NTU),

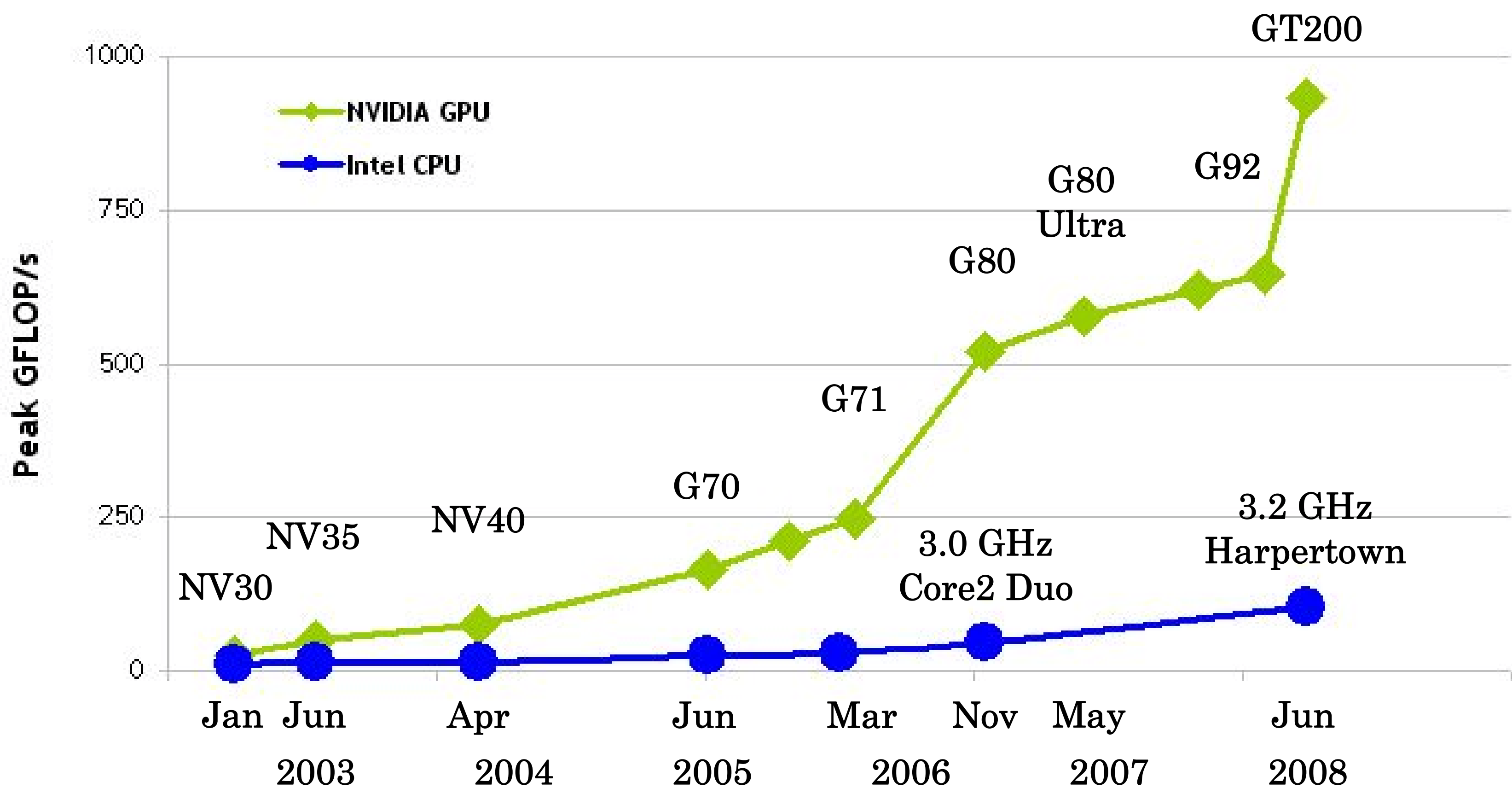
Chen-Mou Cheng (NTU), and Bo-Yin Yang (Academia Sinica)

Graphics Processing Units (GPUs)

Background on hardware (for more see Patrick Stach's talk yesterday):

- Massively parallel architecture.
- Allocates maximal silicon area to (floating-point) arithmetics rather than cache memory or control.
- Shared memory, not much in total.
- Example: NVIDIA GT200 chip
  - 240 cores @ 1296 MHz
  - 583.2 mm^2 die size @ TSMC 65nm process
  - 1.4×10^9 transistors
  - 933.1 GFLOPS (SP) and 77.8 (DP)

Why do People Care About GPUs?



Video games bring science forward!

GPUs in Cryptology

Implementations using OpenGL (need to understand graphics programming):

- Attacks on symmetric ciphers.
- Implementations of AES; many parallel executions.
- Cook, Keromytis, CryptoGraphics: Exploiting Graphics Cards For Security, Advances in Information Security, 20, Springer, 2006.
- Moss, Page, Smart, Toward Acceleration of RSA Using 3D Graphics Hardware, in Cryptography and Coding 2007.

NVIDIA developed CUDA, a C-like language, and an assembly-like language; published first version 2007. asm does not give full machine control.

More Public Key Crypto on GPU

Szerwinski, Güneysu, Exploiting the Power of GPUs for Asymmetric Cryptography. CHES 2008:

- Using NVidia GeForce 8800 GTS 320 (G80).
- Use CUDA for coding.
- 224-bit scalar.
- 224-bit modulus.
- Special modulus: $2^{224} - 2^{96} + 1$.
- 1412 elliptic-curve scalar multiplications per second.

CHES'08 implementation had hard time filling all cores; no side-channel protection. This motivated us to look for other applications, namely ECM as in NFS.

Preview of our Results

Aim for medium size numbers to be factored; settle on 280-bits.

Different application from CHES'08 but can still look at throughput.

- Also using 8800 GTS 320 (G80).
- 280-bit scalar.
- 280-bit modulus.
- General 280-bit moduli.
- 2414 elliptic-curve scalar multiplications per second (compare to the 1412 elliptic-curve scalar multiplications per second in CHES'08 with smaller and special modulus).

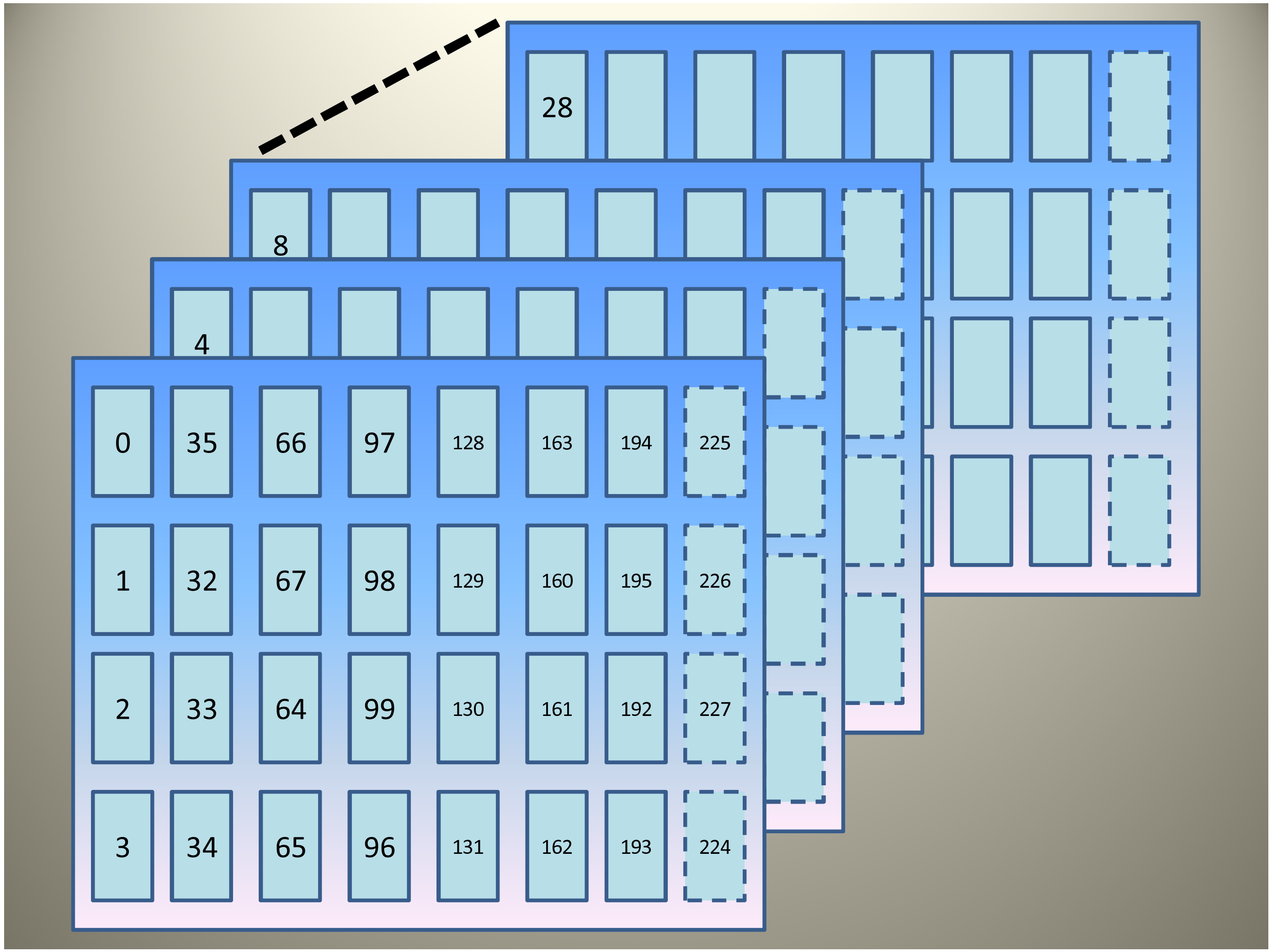
Modular Multiplications

- Decided to use floating point instructions; still experimenting with integer instructions (high bits missing).
- Try to have as much parallelization as possible in mult.
- 28-limb, radix 2^{10}, schoolbook multiplication:
  - Karatsuba is slower because of inefficient use of the native MAD (multiply-and-add) instructions.
- Montgomery's modular reduction.
- Montgomery representation implies that “small” integers turn into full-size modular values.
- Basically turns each tiny 8-core processor on GPU into an 8-way modular arithmetic unit (MAU).

Thread Organization Design

- A group of 32 threads work on multiplying two 28-limb, 280-bit integers.
- Each thread works on a 7-by-4 region.
  - 21 loads from and 10 stores to on-die fast memory.
  - 28 multiplications and 18 additions.
- Each multiprocessor executes 256 threads and hence works on 8 curves at a time.
- Which thread works on what region is carefully designed so that memory addresses accessed by the threads within a same half warp (16 threads) are coalesced properly, avoiding bank conflict in reading from and writing to on-die fast memory.

Thread Organization Design



Thread Organization Design

- A group of 32 threads work on multiplying two 28-limb, 280-bit integers.
- Each thread works on a 7-by-4 region.
  - 21 loads from and 10 stores to on-die fast memory.
  - 28 multiplications and 18 additions.
- Each multiprocessor executes 256 threads and hence works on 8 curves at a time.
- Which thread works on what region is carefully designed so that memory addresses accessed by the threads within a same half warp (16 threads) are coalesced properly, avoiding bank conflict in reading from and writing to on-die fast memory.

ECM on GPU

- Use many curves for attempted factorization of the same integer. (In sequential ECM many curves are tried for the same integer, we use many (e.g. 120 for the GTX280) in parallel).
- In NFS applications we could also choose to use the same curve with different numbers to be factored; our choice allows sharing the modulus between different processing units. (8 processors share memory).
- Generally, memory turns out to be largest restriction.
- Reconsider all choices from software implementations (GMP-ECM and GMP-EECM).

Current Design Choices

- Use Edwards curves!

Current Design Choices

- Use Edwards curves!
- Our field arithmetic does not make multiplications by small integers faster (we use Montgomery representation of integers).
- Multiplications by curve constants and point coordinates count as full multiplications.
- No reason to use twisted Edwards curves.
- No use in using the 100 nice curves mentioned in Peter Birkner's talk.

Choice of Curves

- Use Atkin-Morain curves with torsion structure $\mathbb{Z}/2 \times \mathbb{Z}/8$ – easy to generate, could do precomputations in $\mathbb{Q}$ and then reduce them modulo n.
- Investigating other torsion groups, e.g. Montgomery's $\mathbb{Z}/12$ construction.
- Use affine rather than projective base point and precomputed points. Then coordinates have full size but there is no penalty for that.

Choice of Coordinates

- Projective Edwards coordinates more suitable than inverted Edwards coordinates. They need 1D (multiplication by curve constant) less in DBL.
- Use addition formulas due to Hisil, Wong, Carter, Dawson without multiplications by curve constants (not unified, no problem for ECM).

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1 y_1 + x_2 y_2}{x_1 x_2 + y_1 y_2}, \frac{x_1 y_1 - x_2 y_2}{x_1 y_2 - y_1 x_2} \right).$$

Elliptic Curves

- Try to use windowing with large window – scalar is way longer than modulus ($B_1 = 2^{13}$).
- Use affine base point and precomputations.

Elliptic Curves

- Try to use windowing with large window – scalar is way longer than modulus ($B_1 = 2^{13}$).
- Use affine base point and precomputations.
- Severe storage restrictions! No precomputations possible.
- Only possible to use NAF of scalar.

Elliptic Curves

- Try to use windowing with large window – scalar is way longer than modulus ($B_1 = 2^{13}$).
- Use affine base point and precomputations.
- Severe storage restrictions! No precomputations possible.
- Only possible to use NAF of scalar.
- Way out: parallelize formulas, then 2 processors share memory.
- Problem:
  - DBL: 4M+3S,
  - mADD: 9M,

  both odd; seems to ask for idle stages.

Elliptic Curves

- Develop new formulas; pipeline two operations:
  - DBL-DBL: 4M+3S+6a,
  - mADD-DBL: 7M+1S+7a,
  - DBL+mADD: 6M+2S+8a.
- These numbers are even – and we manged to get perfect parallelism, i.e. no wait-stages for multiplications.
- Result: This freed up enough storage so that we can store 8 points: P, [3]P, [5]P, . . . , [15]P
- Given the size of the moduli, even larger windows would be desirable.

DBL-DBL

Step	MAU 1	MAU 2	
1	$A = X_1^2$	$B = Y_1^2$	S
2	$X_1 = X_1 + Y_1$	$C = A + B$	a
3	$X_1 = X_1^2$	$Z_1 = Z_1^2$	S
4	$X_1 = X_1 - C$	$Z_1 = Z_1 + Z_1$	a
5	$B = B - A$	$Z_1 = Z_1 - C$	a
6	$X_1 = X_1 \times Z_1$	$Y_1 = B \times C$	M
7	$A = X_1 \times X_1$	$Z_1 = Z_1 \times C$	M
8	$Z_1 = Z_1^2$	$B = Y_1^2$	S
9	$Z_1 = Z_1 + Z_1$	$C = A + B$	a
10	$B = B - A$	$X_1 = X_1 + Y_1$	a
11	$Y_1 = B \times C$	$X_1 = X_1 \times X_1$	M
12	$B = Z_1 - C$	$X_1 = X_1 - C$	a
13	$Z_1 = B \times C$	$X_1 = X_1 \times B$	M
			4M+3S+6a

Horizontal line indicates beginning of second operation.

mADD-DBL

Step	MAU 1	MAU 2	
1	$B{=}x_2 \times Z_1$	$C{=}y_2 \times Z_1$	M
2	$A{=}X_1 \times Y_1$	$Z_1{=}B \times C$	M
3	$E{=}X_1 - B$	$F{=}Y_1 + C$	a
4	$X_1{=}X_1 + C$	$Y_1{=}Y_1 + B$	a
5	$E{=}E \times F$	$Y_1{=}X_1 \times Y_1$	M
6	$F{=}A + Z_1$	$B{=}A - Z_1$	a
7	$E{=}E - B$	$Y_1{=}Y_1 - F$	a
8	$Z_1{=}E \times Y_1$	$X_1{=}E \times F$	M
9	$Y_1{=}Y_1 \times B$	$A{=}X_1 \times X_1$	M
10	$Z_1{=}Z_1^2$	$B{=}Y_1^2$	S
11	$Z_1{=}Z_1 + Z_1$	$C{=}A + B$	a
12	$B{=}B - A$	$X_1{=}X_1 + Y_1$	a
13	$Y_1{=}B \times C$	$X_1{=}X_1 \times X_1$	M
14	$B{=}Z_1 - C$	$X_1{=}X_1 - C$	a
15	$Z_1{=}B \times C$	$X_1{=}X_1 \times B$	M
			7M+1S+7a

DBL-mADD

Step	MAU 1	MAU 2	
1	$A{=}X_1^2$	$B{=}Y_1^2$	S
2	$X_1{=}X_1+Y_1$	$C{=}A+B$	a
3	$X_1{=}X_1^2$	$Z_1{=}Z_1^2$	S
4	$X_1{=}X_1-C$	$Z_1{=}Z_1+Z_1$	a
5	$B{=}B-A$	$Z_1{=}Z_1-C$	a
6	$X_1{=}X_1\times Z_1$	$Y_1{=}B\times C$	M
7	$Z_1{=}Z_1\times C$	$A{=}X_1\times Y_1$	M
8	$B{=}x_2\times Z_1$	$C{=}y_2\times Z_1$	M
9	$E{=}X_1-B$	$F{=}Y_1+C$	a
10	$X_1{=}X_1+C$	$Y_1{=}Y_1+B$	a
11	$E{=}E\times F$	$Z_1{=}B\times C$	M
12	$F{=}A+Z_1$	$B{=}A-Z_1$	a
13	$E{=}E-B$	$Z_1{=}X_1$	a
14	$A{=}Z_1\times Y_1$	$X_1{=}E\times F$	M
15	$A{=}A-F$		a
16	$Z_1{=}E\times A$	$Y_1{=}A\times B$	M
			6M+2S+8a

New Speed Records for ECM

- 604.99 curves/sec for ECM stage 1 with $B_1 = 8192$ for 280-bit integers on a single PC.
  - Achieved using two NVIDIA GeForce 280 GTX graphics cards and a CPU.
  - A single 280 GTX can do 22.66×10^6 modular multiplications per second.
- Compare to (almost) speed leader on CPU, the GMP-ECM: 171.42 curves/sec on a 2.4GHz Core 2 Quad Q6600.
  - 17.91×10^6 modular multiplications per second.

Performance Comparison

We timed the basic implementation (using NAF) and the windowing method using parallel formulas on GPUs vs. GMP-ECM on CPUs.

Coprocessor	#Cores	Freq (GHz)	Rmax (GFLOPS)	Mulmods (10^6/sec)	Curves (1/sec)
CHES 2008 (scaled)	96	1.2	230.4		26.81
8800 GTS (G80)	96	1.2	230.4	7.51	57.30
8800 GTS (G92)	128	1.625	416.0	13.64	104.14
GTX 260	192	1.242	476.9	14.97	119.05
GTX 280	240	1.296	622.1	19.53	155.29
Core 2 Duo E6850	2	3.0	48.0	11.19	107.14
Core 2 Quad Q6600	4	2.4	76.8	17.91	171.42
Core 2 Quad Q9550	4	2.83	90.7	21.77	208.39
GTX 260 (parallel)	192	1.242	476.9	16.61	165.58
GTX 280 (parallel)	240	1.296	622.1	22.66	216.78
Q6600+GTX 280$\times$2				63.23	604.99

More Detailed Price Analysis

Coprocessor	Component-wise Cost ($)	Component-wise P/C (1/(sec·$))	System-wise Cost ($)	System-wise P/C (1/(sec·$))
8800 GTS (G80)	119	0.48	1006	0.1139
8800 GTS (G92)	178	0.59	1124	0.1853
GTX 260	250	0.48	1268	0.1878
GTX 280	400	0.39	1568	0.1981
Core 2 Duo E6850	200	0.54	972	0.1102
Core 2 Quad Q6600	185	0.93	957	0.1791
Core 2 Quad Q9550	325	0.64	1097	0.1900
GTX 260 (parallel)	250	0.66	1268	0.2612
GTX 280 (parallel)	400	0.54	1568	0.2765
Q6600+GTX 280$\times$2	985	0.61	1889	0.3203

Prices based on Pricegrabber, 03 September 2008.

Cheaper: educational discounts from NVIDIA etc. or . . .

…fly to Taiwan!

The end!