ECM speed records on CPU and GPU

D. J. Bernstein
University of Illinois at Chicago

NSF ITR–0716498

New EECM web site:
http://eecm.cr.yp.to

Joint work with:

1	2	3	Tanja Lange
1			Peter Birkner
1			Christiane Peters
	2	3	Chen-Mou Cheng
	2	3	Bo-Yin Yang
	2		Tien-Ren Chen
		3	Hsueh-Chung Chen
		3	Ming-Shing Chen
		3	Chun-Hung Hsiao
		3	Zong-Cing Lin

1. “ECM using Edwards curves.”

Prototype software: GMP-EECM.
New rewrite: EECM-MPFQ;
first announcement today!
Available now for download.

2. EUROCRYPT 2009:
“ECM on graphics cards.”

Prototype CUDA-EECM.

3. SHARCS 2009: “The
billion-mulmod-per-second PC.”

Current CUDA-EECM,
plus fast mulmods on
Core 2, Phenom II, and Cell.

Fewer mulmods

Measurements of EECM-MPFQ for $B_1 = 1000000$:

$b = 1442099$ bits in $s = \text{lcm}\{1, 2, 3, 4, \ldots, B_1\}$.

$P \mapsto sP$ is computed using 1442085 ($= 0.99999b$) DBL + 98341 ($0.06819b$) ADD.

These DBLs and ADDs use 5211333**M** ($3.61371b$**M**) + 5768340**S** ($3.99996b$**S**) + 9340897**add** ($6.47729b$**add**).

Compare to GMP-ECM 6.2.3:

$P \mapsto sP$ is computed using
2001915 (1.38820b) DADD +
194155 (0.13463b) DBL.

These DADDs use
8590140**M** (5.95669b**M**) +
4392140**S** (3.04566b**S**) +
12788124**add** (8.86772b**add**).

Compare to GMP-ECM 6.2.3:

$P \mapsto sP$ is computed using
2001915 (1.388820b) DADD +
194155 (0.134463b) DBL.

These DADDs use
8590140$\mathbf{M}$ (5.956669$b\mathbf{M}$) +
4392140$\mathbf{S}$ (3.045666$b\mathbf{S}$) +
12788124$\mathbf{add}$ (8.86772$b\mathbf{add}$).

Could do better! 0.134463$b\mathbf{M}$
are actually 0.134463$b\mathbf{D}$.
$\mathbf{D}$: mult by curve constant.
Small curve, small P, ladder
$\Rightarrow 4b\mathbf{M} + 4b\mathbf{S} + 2b\mathbf{D} + 8b\mathbf{add}$.
EECM still wins.

HECM handles 2 curves using
$2b\mathbf{M} + 6b\mathbf{S} + 8b\mathbf{D} + \cdots$
(1986 Chudnovsky–Chudnovsky, et al.); again EECM is better.

HECM handles 2 curves using
$2b\mathbf{M} + 6b\mathbf{S} + 8b\mathbf{D} + \cdots$
(1986 Chudnovsky–Chudnovsky, et al.); again EECM is better.

What about NFS? $B_1 = 1000$?
Measurements of EECM-MPFQ:

$b = 1438$ bits in s.

$P \mapsto sP$ is computed using
1432 ($0.99583b$) DBL +
211 ($0.14673b$) ADD.

These DBLs and ADDs use
6204$\mathbf{M}$ ($4.31433b\mathbf{M}$) +
5728$\mathbf{S}$ ($3.98331b\mathbf{S}$) +
10069**add** ($7.00209b$**add**).

Note: smaller window size
in addition chain,
so more ADDs per bit.

Compare to GMP-ECM 6.2.3:

Note: smaller window size
in addition chain,
so more ADDs per bit.

Compare to GMP-ECM 6.2.3:

$P \mapsto sP$ is computed using
8278**M** (5.756661b**M**) $+$
4305**S** (2.99374b**S**) $+$
12224**add** (8.50070b**add**).

Even for this small B_1,
EECM beats Montgomery ECM
in operation count.
Advantage grows with B_1.

Notes on current stage 2:

1. EECM-MPFQ jumps through the j's coprime to d_1.
GMP-ECM: coprime to 6.

2. EECM-MPFQ computes Dickson polynomial values using Bos–Coster addition chains.
GMP-ECM: ad-hoc, relying on arithmetic progression of j.

3. EECM-MPFQ doesn't bother converting to affine coordinates until the end of stage 2.

4. EECM-MPFQ uses NTL for poly arith in "big" stage 2.

More primes per mulmod

1987/1992 Montgomery,
1993 Atkin–Morain
had suggested using torsion
$\mathbf{Z}/12$ or $(\mathbf{Z}/2) \times (\mathbf{Z}/8)$.

GMP-ECM went back to $\mathbf{Z}/6$.

"ECM using Edwards curves"
introduced new small curves
with $\mathbf{Z}/12$, $(\mathbf{Z}/2) \times (\mathbf{Z}/8)$.

Does big torsion really help?
Let's look at what matters:
number of mulmods used
to find an average prime.

e.g. Try all 7530 primes
between $2^{25} - 2^{17}$ and 2^{25}.

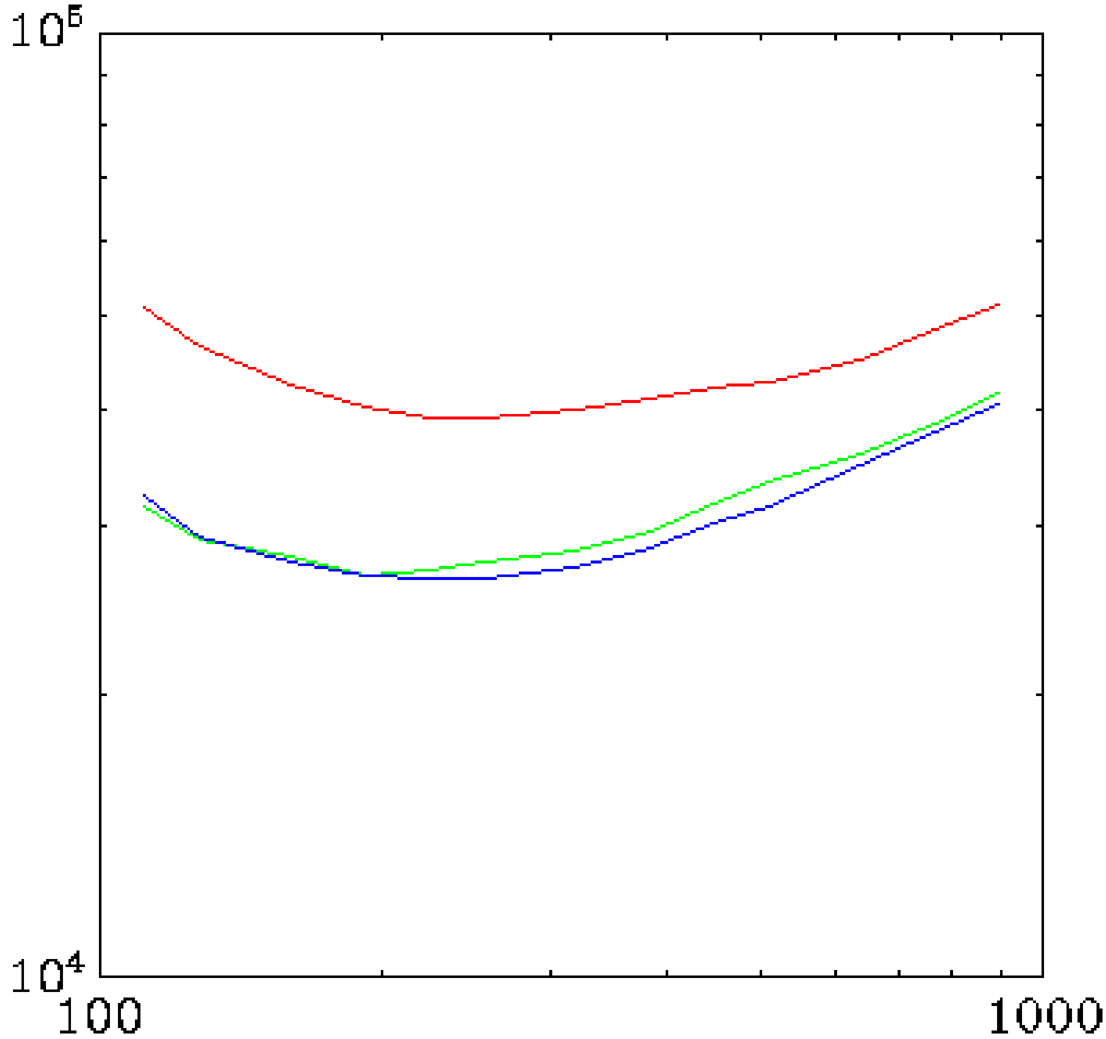
EECM-MPFQ $B_1 = 128$ $d_1 = 120$
with a $\mathbf{Z}/4$ Edwards curve
uses $21774749\mathbf{M} + 5509272\mathbf{S}$
to find 2070 of these primes.
Cost per prime found:
$10519\mathbf{M} + 2661\mathbf{S}$.

e.g. Try all 7530 primes between $2^{25} - 2^{17}$ and 2^{25}.

EECM-MPFQ $B_1 = 128$ $d_1 = 120$ with a $\mathbf{Z}/4$ Edwards curve uses $21774749\mathbf{M} + 5509272\mathbf{S}$ to find 2070 of these primes. Cost per prime found: $10519\mathbf{M} + 2661\mathbf{S}$.

EECM-MPFQ $B_1 = 96$ $d_1 = 60$ with a $\mathbf{Z}/12$ Edwards curve uses $10607297\mathbf{M} + 3883056\mathbf{S}$ to find 1605 of these primes. Cost per prime found: $6608\mathbf{M} + 2419\mathbf{S}$.

Cost per prime found
for 30-bit primes,
as function of B_1:



Between $2^{35} - 2^{17}$ and 2^{35}:

$B_1 = 640$ $d_1 = 210$ $\mathbf{Z}/4 \Rightarrow$
107045**M** per prime found.

$B_1 = 384$ $d_1 = 150$ $\mathbf{Z}/12 \Rightarrow$
75769**M** per prime found.

Some upcoming experiments:

1. Try $a = -1$ curves.

2. Replace some **M** with **D**;
account for resulting speedup.

3. Check many more primes
for robust statistics.

Faster mulmods

ECM is bottlenecked by mulmods:

- practically all of stage 1;
- curve operations in stage 2 (pumped up by Dickson!);
- final product in stage 2, *except* fast poly arith.

GMP-ECM does mulmods with the GMP library.
. . . but GMP has slow API,
so GMP-ECM has ≥ 20000 lines of new mulmod code.

```
$ wc -c<eecm-mpfq.tar.bz2
8853
```

Obviously EECM-MPFQ doesn't include new mulmod code!

```
$ wc -c<eecm-mpfq.tar.bz2
8853
```

Obviously EECM-MPFQ doesn't include new mulmod code!

MPFQ library (Gaudry–Thomé) does arithmetic in $\mathbf{Z}/n$ where number of n words is known at compile time. Better API than GMP: most importantly, n in advance.

EECM-MPFQ uses MPFQ for essentially all mulmods.

GMP-ECM 6.2.3 (2009.04)
using GMP 4.3.1 (2009.05),
both current today:

Tried 1000 curves, $B_1 = 1024$,
typical 240-bit n,
on 2.4GHz Core 2 Quad 6fb.

Stage 1: $5.84 \cdot 10^6$ cycles/curve.

GMP-ECM 6.2.3 (2009.04)
using GMP 4.3.1 (2009.05),
both current today:

Tried 1000 curves, $B_1 = 1024$,
typical 240-bit n,
on 2.4GHz Core 2 Quad 6fb.

Stage 1: $5.84 \cdot 10^6$ cycles/curve.

EECM-MPFQ,
same 240-bit n, same CPU,
1000 curves, $B_1 = 1024$:
$3.92 \cdot 10^6$ cycles/curve.

Some speedup from Edwards;
some speedup from MPFQ.

What about stage 2?

GMP-ECM,
100 curves, $B_2 = 443706$,
Dickson polynomial degree 1:
$28.2 \cdot 10^6$ cycles/curve.
Degree 3: $34.7 \cdot 10^6$.

Some speedup from Edwards;
some speedup from MPFQ.

What about stage 2?

GMP-ECM,
100 curves, $B_2 = 443706$,
Dickson polynomial degree 1:
$28.2 \cdot 10^6$ cycles/curve.
Degree 3: $34.7 \cdot 10^6$.

EECM-MPFQ, 100 curves,
$d_1 = 990$, range 506880
for primes $990i \pm j$:
$23.8 \cdot 10^6$ cycles/curve.
Degree 3: $30.9 \cdot 10^6$.

Summary: EECM-MPFQ uses fewer mulmods than GMP-ECM; takes less time than GMP-ECM; and finds more primes.

Summary: EECM-MPFQ uses fewer mulmods than GMP-ECM; takes less time than GMP-ECM; and finds more primes.

Are GMP-ECM and EECM-MPFQ fully exploiting the CPU? No!

Three ongoing efforts to speed up mulmods for ECM: Thorsten Kleinjung, for RSA-768; Alexander Kruppa, for CADO; and ours—see next slide.

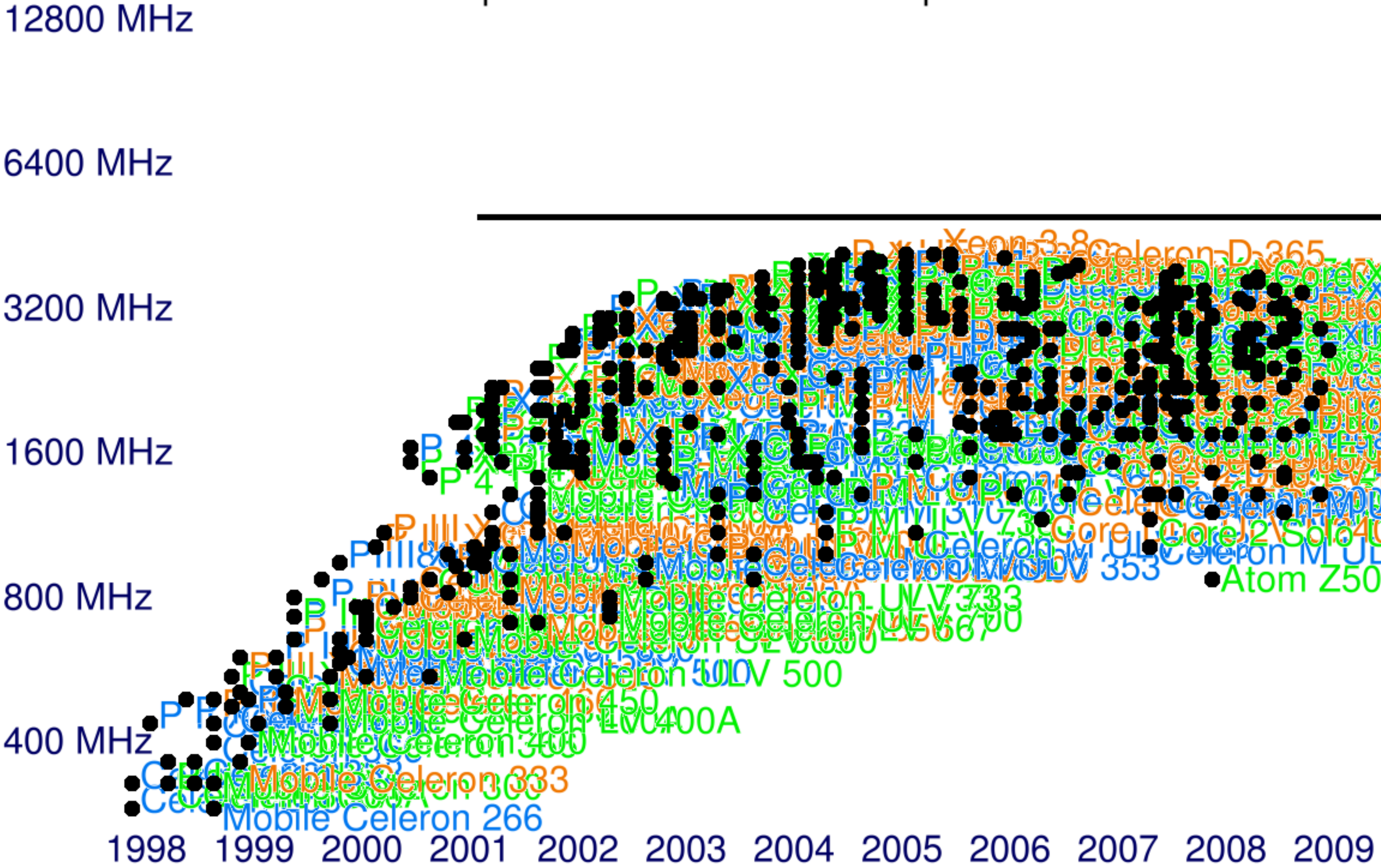
Our latest mulmod speeds,
interleaving vector threads
with integer threads:

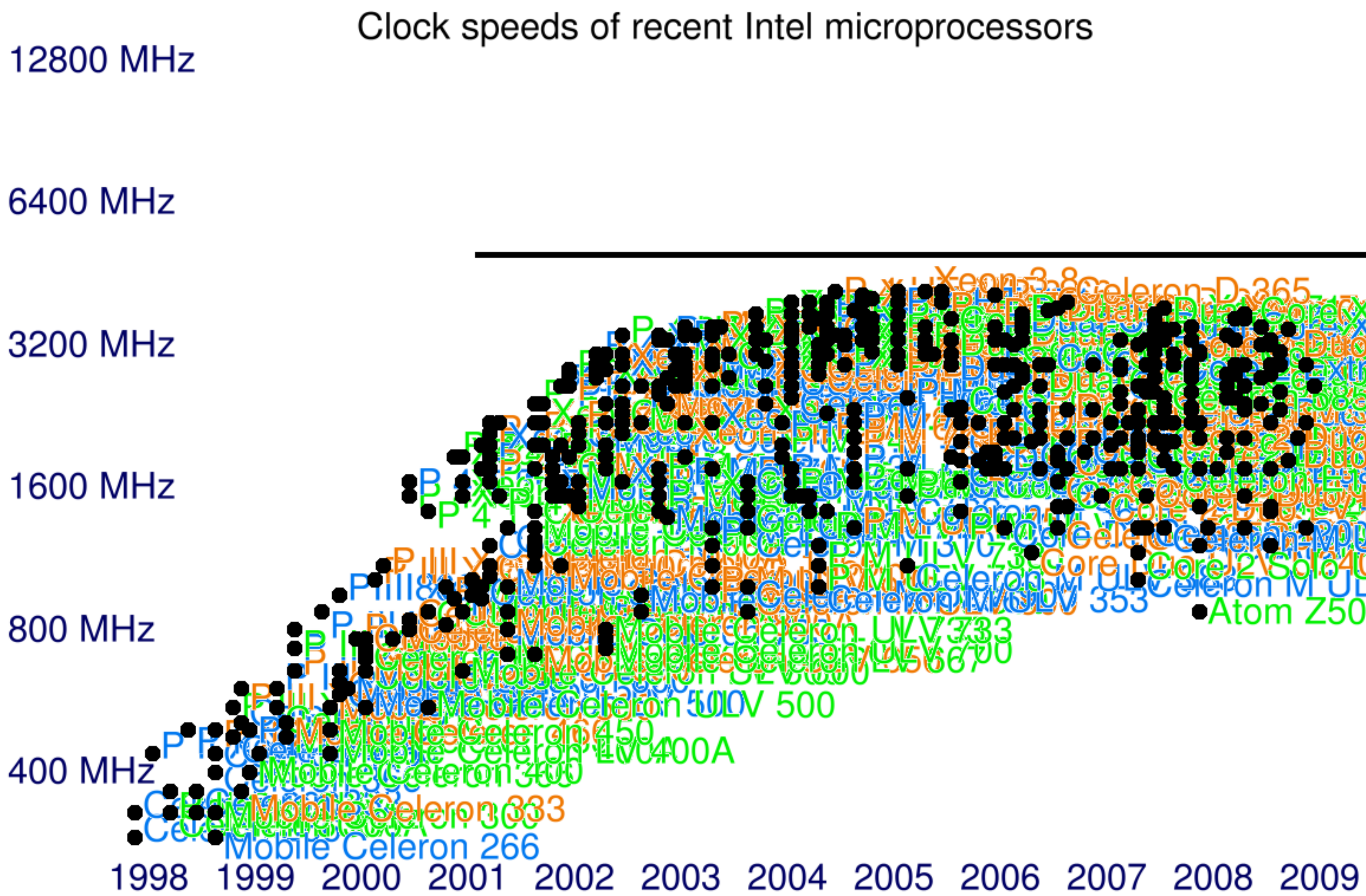
$4\times$3GHz Phenom II 940:
$202 \cdot 10^6$ 192-bit mulmods/sec.

$4\times$2.83GHz Core 2 Quad Q9550:
$114 \cdot 10^6$ 192-bit mulmods/sec.

$6\times$3.2GHz Cell (Playstation 3):
$102 \cdot 10^6$ 195-bit mulmods/sec.

Clock speeds of recent Intel microprocessors





How do we gain more speed
if clock speeds have stalled?
Answer: Massive parallelism!

$500 GTX 295
is one card with two GPUs.

Total 480 32-bit ALUs
running at 1.242GHz.

Our latest CUDA-EECM speed:
$481 \cdot 10^6$ 210-bit mulmods/sec.

For $\approx$ $2000 can build PC
with one CPU and two GPUs:
$1300 \cdot 10^6$ 192-bit mulmods/sec.