Implementation a Sieving Algorithm on a Dynamic Reconfigurable Processor

Takeshi Shimoyama (FUJITSU)

Tetsuya Izu (FUJITSU)

Jun Kogure (FUJITSU)

※This work is financially supported by a consignmemt research from the NICT,Japan

Outline

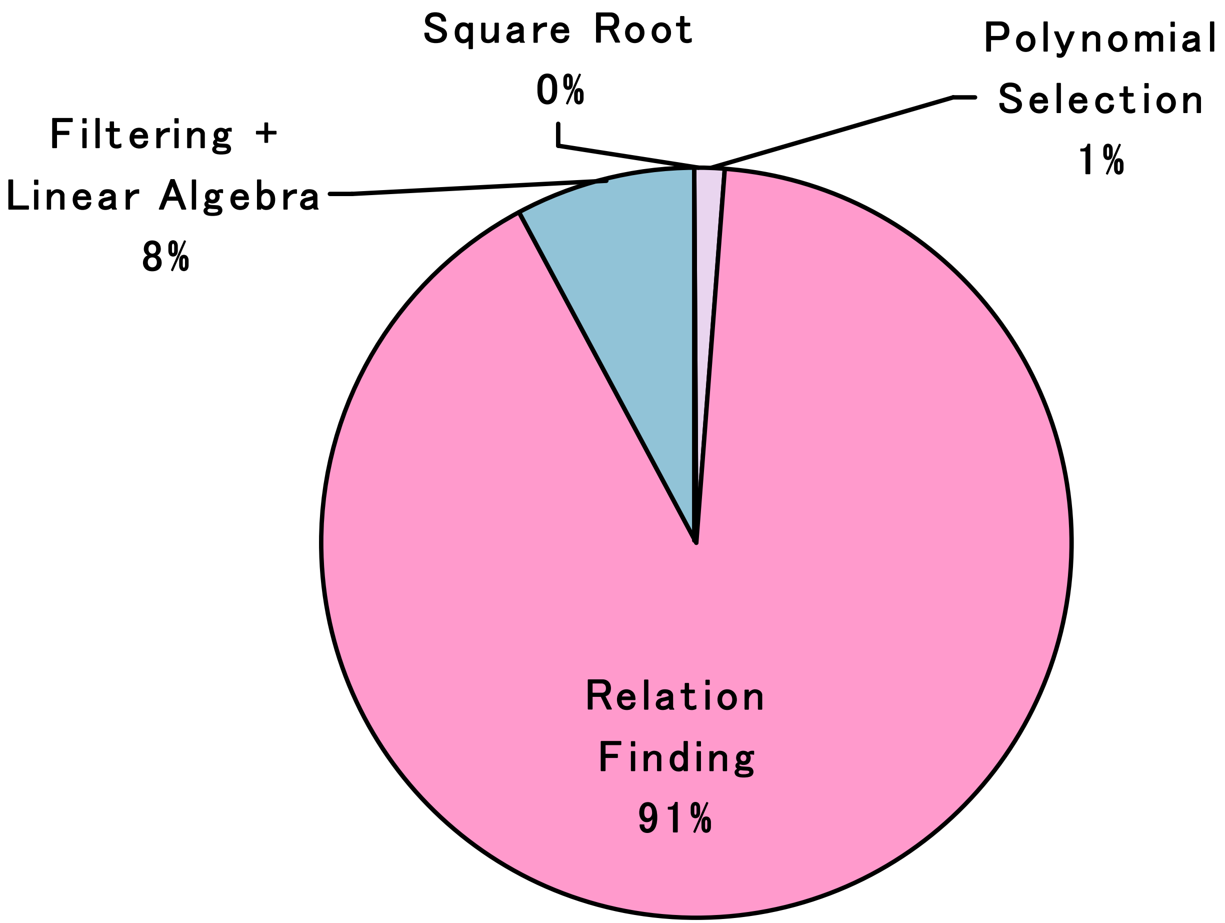
1. Estimation of each GNFS steps from software
2. Factorization hardware
3. Dynamic reconfigurable processor DAPDNA2
4. Sieving algorithm on DAPDNA2
5. Demo & Evaluation
6. Conclusions & Future works

Estimation of the Sieving Ratio from Software implementation

- GNFS steps
  1. Polynomial selection
  2. Relation finding (Sieving)
  3. Filtering + Linear algebra
  4. Square root
- Evironments
  - Test data: 164 digit composit number from cuningham number
  - Algorithm: Lattice sieve with 2+2 large prime variation, etc...
  - Machine: Pentium4 2.53GHz Northwood

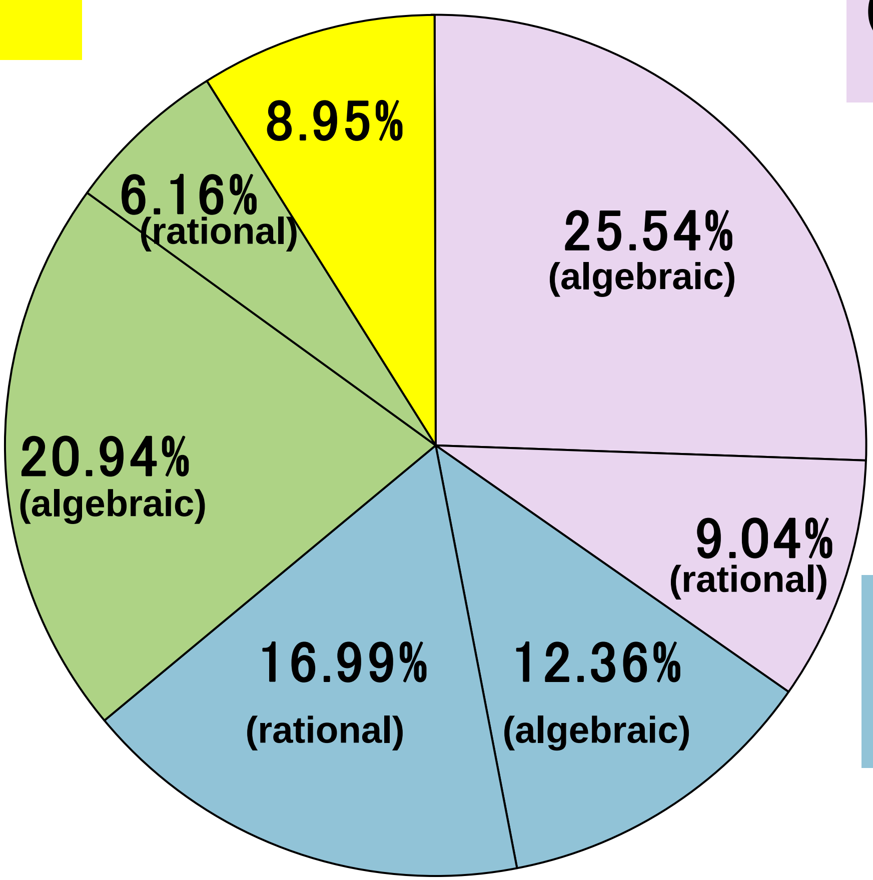
Ratio of each steps of GNFS

(164digit factoring timing data by Lattice Sieve in Software)



Ratio of the parts in Relation Finding

(164digit factoring timing data by Lattice Sieve in Software)



Trial Division
8.95%

Sieving
(Smallish Prime)
34.58%

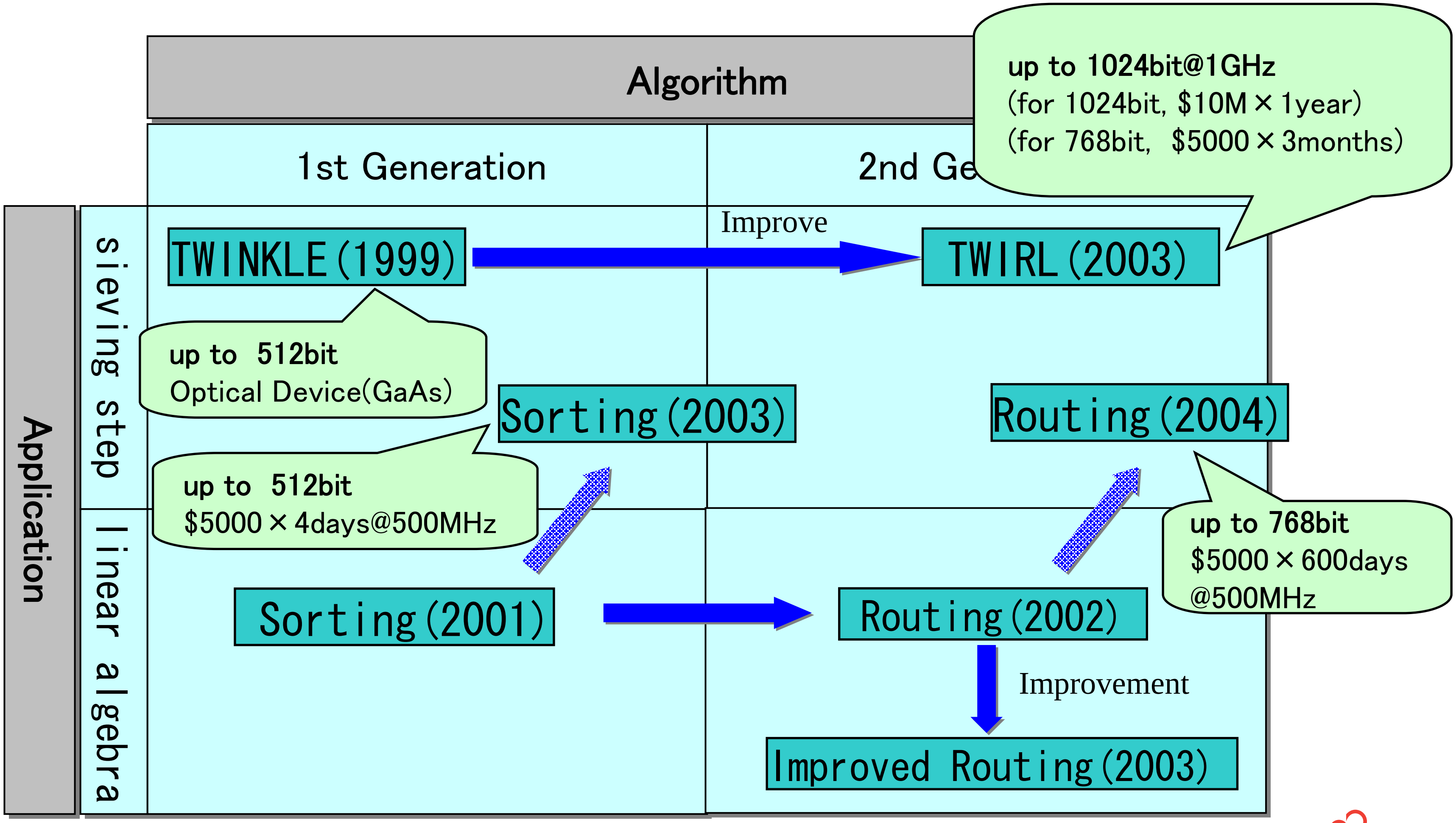
Lattice Base
27.10%

Sieving
(Largish Prime)
29.35%

Outline

1. Estimation of each GNFS steps from software
2. Factorization hardware
3. Dynamic reconfigurable processor DAPDNA2
4. Sieving algorithm on DAPDNA2
5. Demo & Evaluation
6. Conclusions & Future works

Factorization Hardware



Application	Algorithm: 1st Generation	Algorithm: 2nd Generation
sieving step	TWINKLE (1999) Sorting (2003)	TWIRL (2003) Routing (2004)
linear algebra	Sorting (2001)	Routing (2002) Improved Routing (2003)

Improve

Improvement

up to 1024bit@1GHz
(for 1024bit, $10M × 1year)
(for 768bit, $5000 × 3months)

up to 512bit
Optical Device(GaAs)

up to 512bit
$5000 × 4days@500MHz

up to 768bit
$5000 × 600days
@500MHz

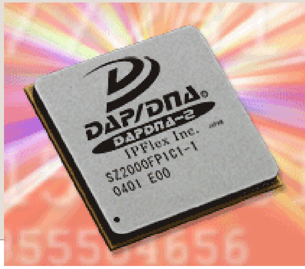
Produce and Run a Factorization Hardware!

- Motivation
  - There are many previous works in virtual world
  - But, these Hardware device of factorizations have not seen (in the real world), yet.
- Target Device
  - ASIC ⇒ too reckless!
  - FPGA ⇒ not easy (for me)!
  - DAPDNA2 ⇒ Current Target!

Outline

1. Estimation of each GNFS steps from software
2. Factorization hardware
3. What is DAPDNA2?
4. Sieving algorithm on DAPDNA2
5. Demo & Evaluation
6. Conclusions & Future works

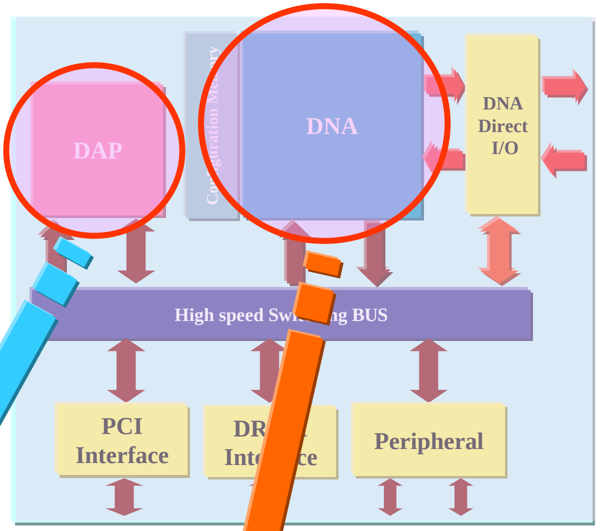
What is DAPDNA2(DD2)？

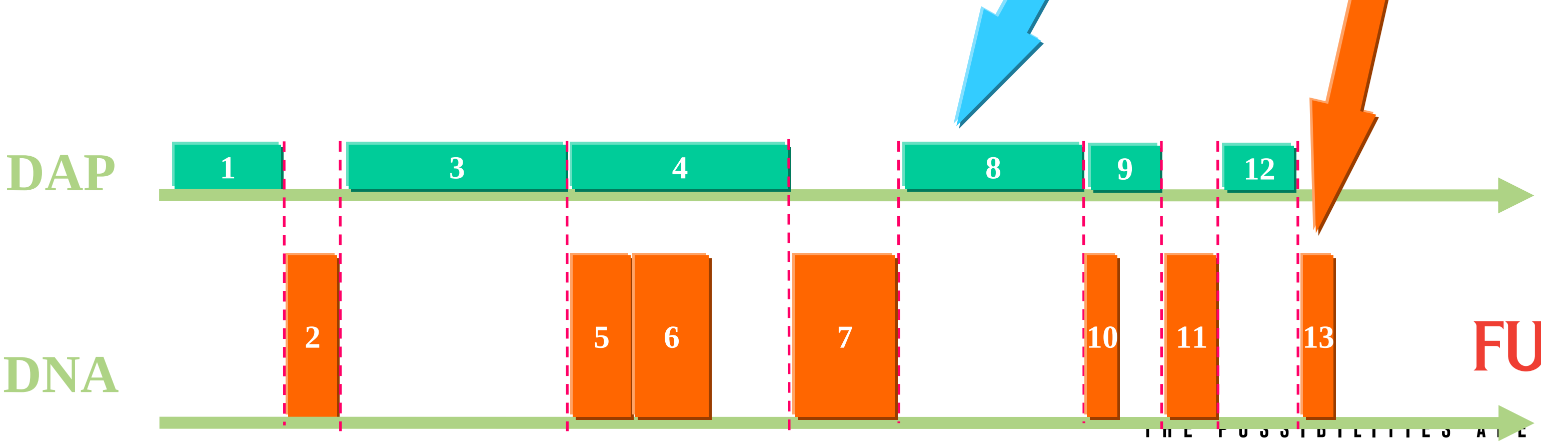


- A Dynamic Reconfigurable Processor introduced and manufactured by IPFlex Inc.
- Multi processor structure
  - DAP（RISC CPU） for System Control
  - DNA（Reconfigurable processor core） for high speed data processing

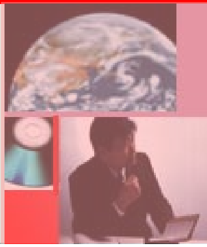
→1 chip solution

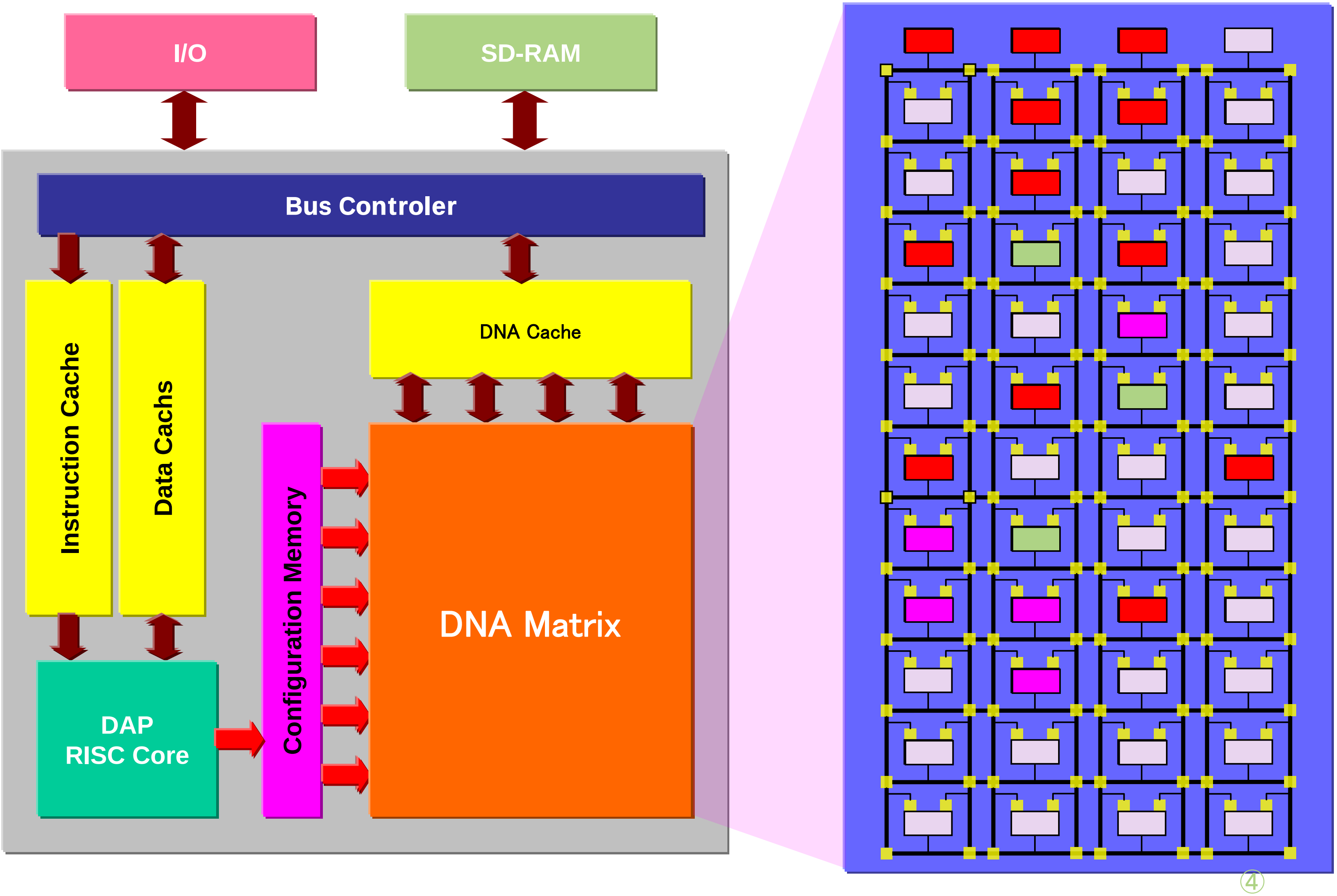
- DAP（Digital Application Processor）
  - 32bit RISC CPU
  - for control of configurations of DNA
- DNA（Distributed Network Architecture）
  - Two-dimensional array of 376 Processing Elements(PE)
  - Allows arbitrary configuration of the degree of parallelism and pipeline depth
  - Dynamic reconfiguration switch to another configuration in 1 clock cycle





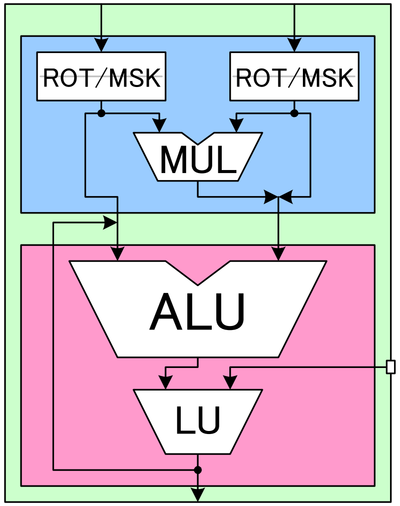
Concept of DAPDNA2





DAPDNA-2 Processing Elements (PEs)

	Processing Elements	#	Function
Data Operation	EXE	168	32bit, 2 input 1 output execution Include 56 multipliers (16-bit input, 32 bit output)
	DLE	136	32bit, 2 input 2 output deley Configurable delay length
	RAM	32	Internal memory, (16KB × 32=512KB)
Data IO	C16E	12	Address generation for LDB/STB access Generic 16bit counter
	C32E	12	Address generation for external memory access Generic 32bit counter
	LDB	4	Data input to DNA through the load buffer
	STB	4	Data output from DNA through the store buffer
	LDX	4	Data input to DNA through the Direct I/O
	STX	4	Data output from DNA through the Direct I/O
total		376	



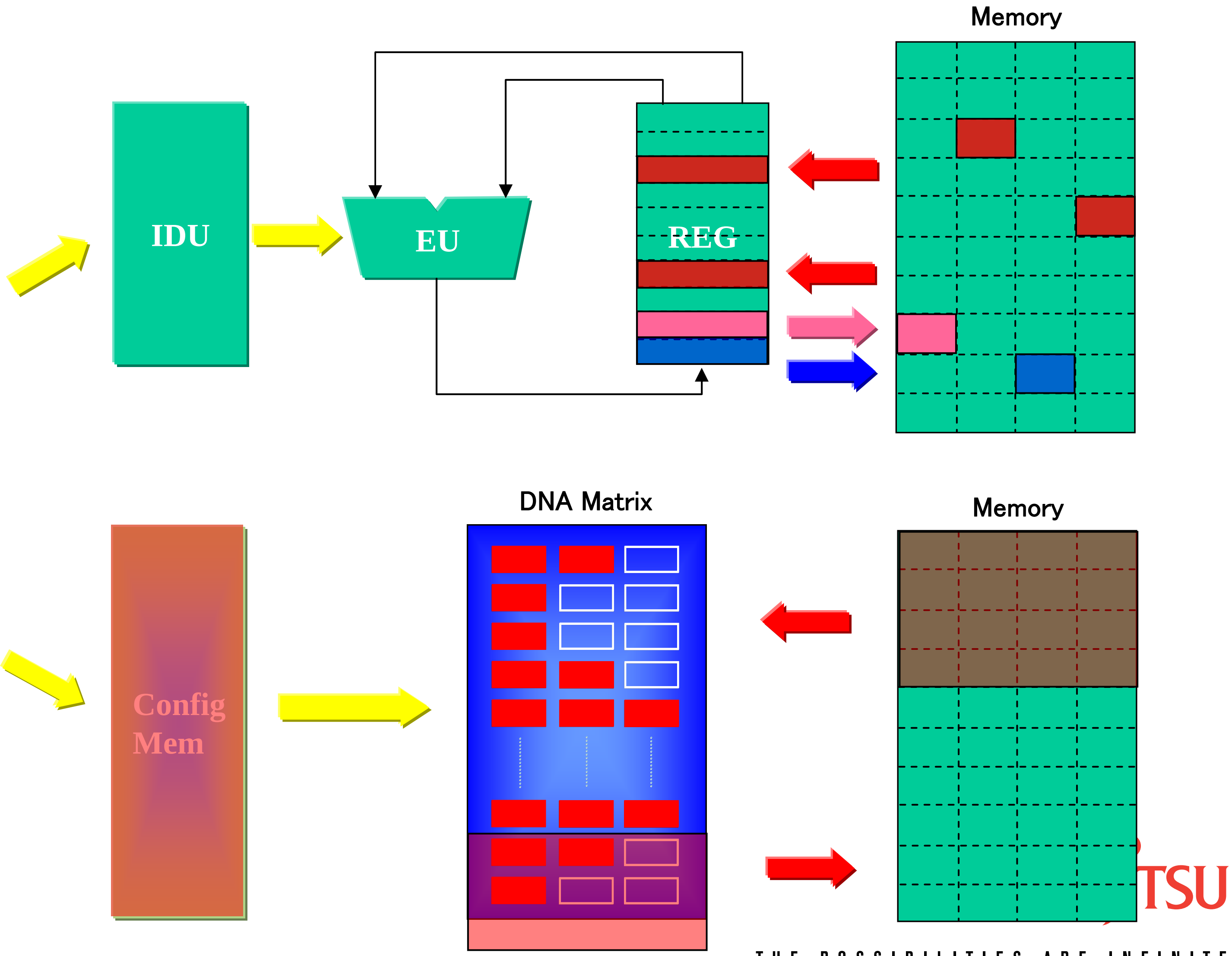
EXE (w/Multiply)

Comparison between Software and DAPDNA2

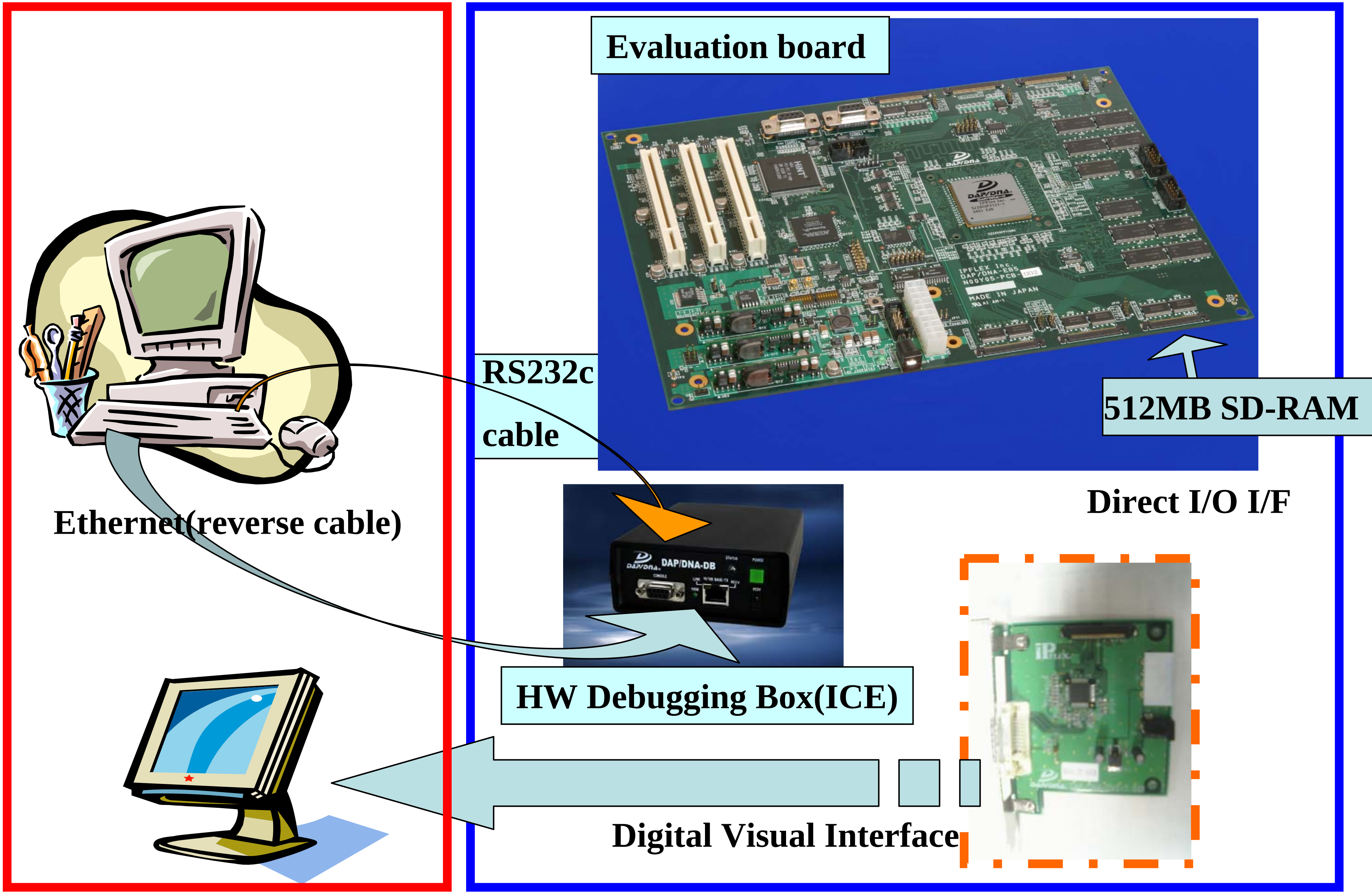
```
.L8:
        cmpge       t3,8
        br/t        .L15
.L9:
        mov         t8,cos_table1
        mov         t3,cos_table2
        addv        t2,t8,t2
        mov         cos_table3,t9
        lsl         t7,t3,3
        ld          t8,[fp+4]
        addv        t2,t8,t2

        lsl         t7,t7,1
                    :
                    :
                    :

        addv        t8,t8,t11
        addv        t7,t7,t10
        addv        t4,t4,a1
.L12:
        mov         t9,1
        addv        t2,t2,t9
        br          .L10
.L13:
.L14:
        mov         t2,1
        addv        t3,t3,t2
        br          .L8c
```



Evaluation Environment



Evaluation board

RS232c cable

512MB SD-RAM

Direct I/O I/F

Ethernet(reverse cable)

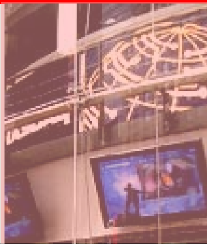
HW Debugging Box(ICE)

Digital Visual Interface

Outline

1. Estimation of each GNFS steps from software
2. Factorization hardware
3. Dynamic reconfigurable processor DAPDNA2
4. Sieving algorithm on DAPDNA2
5. Demo & Evaluation
6. Conclusions & Future works

Line Sieving



```
For  $b \leftarrow 1$ to $H_b$
        set $S[a]$ to $\log(F(a,b))$ for all a
        For prime $p \leftarrow 2$ to B
                compute the sieving points $a \geqq -H_a$
                While $a < H_a$
                        $S[a] \leftarrow S[a] - \log p$
                        $a \leftarrow a + p$
                end while
        end for
end for
```

Our Approach

Combine two methodologies

1. Pipeline Method

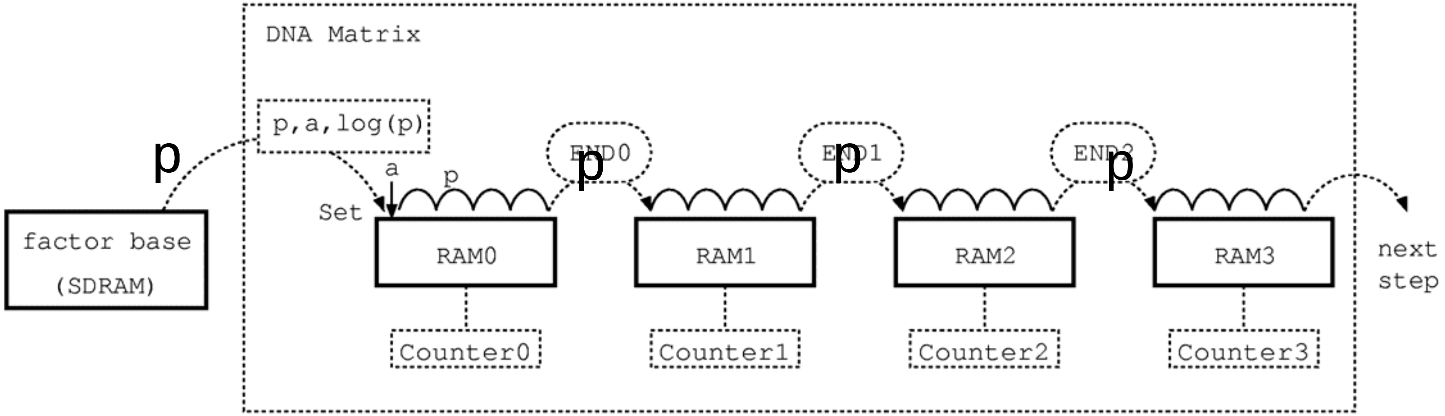
   ⇒ Sieving by Smallish Prime (P < one sieving size)

2. Bucket Sort Method

   ⇒ Sieving by Largish Prime (P ≧ one sieving size)

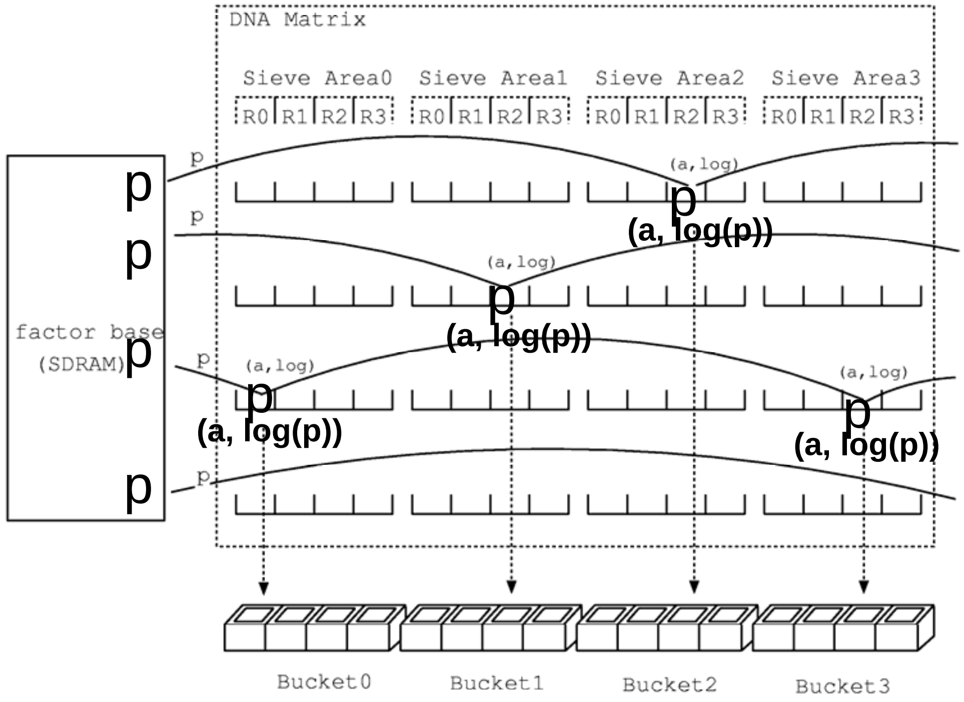
Pipeline Sieving

※ Suitable for smallish primes

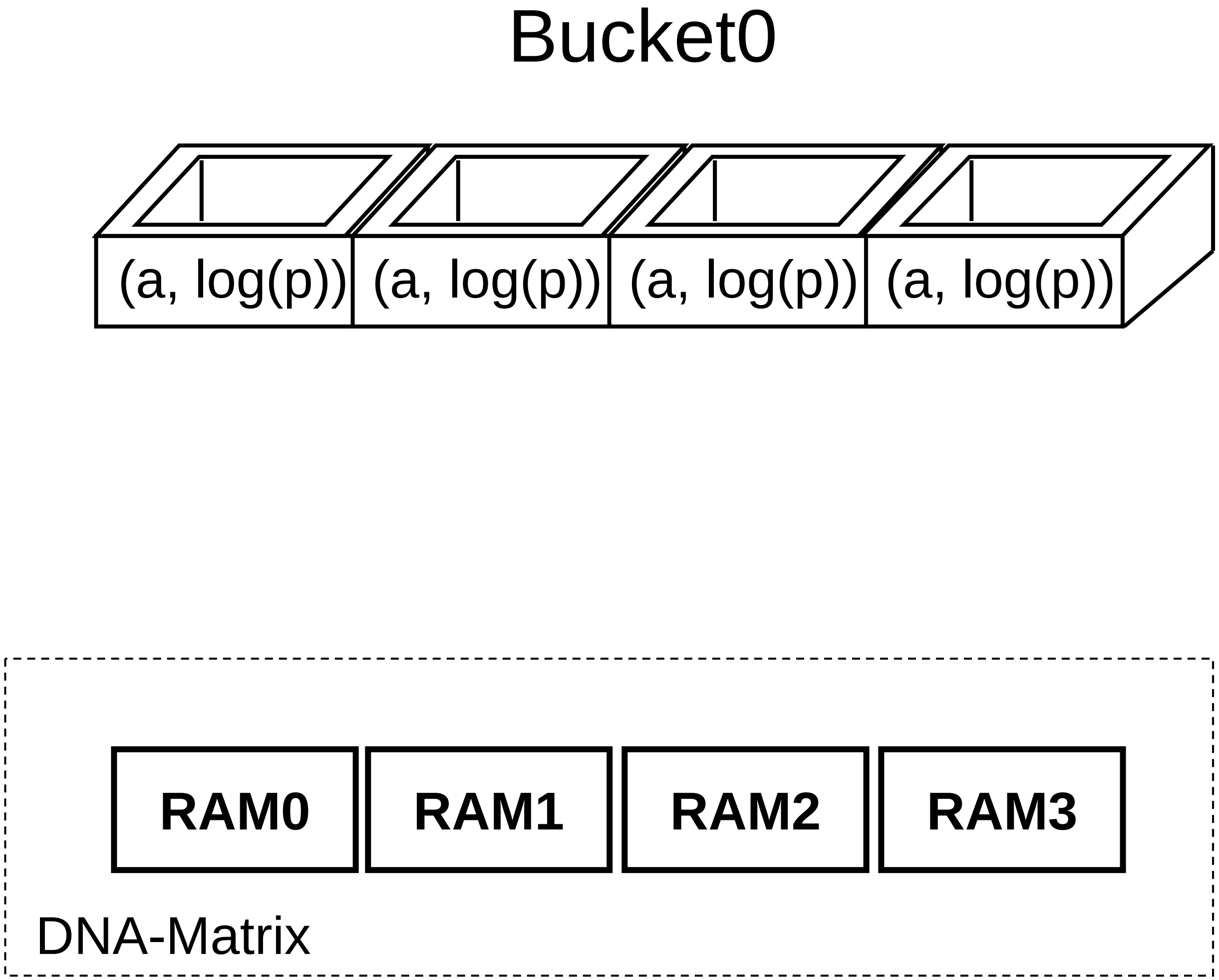


Bucket Sort Sieving (Phase 1)

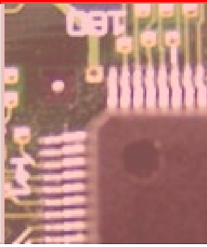
※ Suitable for largish primes



Bucket Sort Sieving (Phase 2)



Current status of our Implemenataion into DAPDNA2



- Phases in sieving

		status
1. memory setup phase	→	completed (05/2/14)
2. pipeline sieving phase	→	completed (05/2/21)
3. bucket sort sieving phase	→	not completed ※
4. extract relation phase	→	completed (05/2/14)

※ Since "the arbitration" of memory interface becomes too complex.

Demonstration

Simulator : DAPDNA2 FrameWork2

Evaluation on DAPDNA2
(sample data. RSA100)

- Timing data for one sieving area (2^{16})

  (parameter RSA100, ap: 2200000, rp: 300000)

		clock	timing
algebraic	memory setup	65818	0.396ms
	sieving	7477661	45.046ms
	extract relation	65611	0.395ms
rational	memory setup	65818	0.396ms
	sieving	3633901	21.890ms
	extract relation	65611	0.395ms
total		11374420	68.520ms

clock cycle
: 166MHz

- Estimation of whole sieving time without trial division : 682.16 hours

  It's more than **40 times slower than software**...

※Software : 16.7h on one Pentium4(2.8GHz) by using lattice sieve

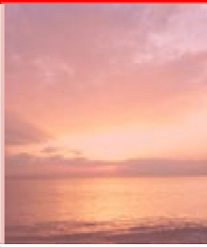
Outline

1. Estimation of each GNFS steps from software
2. Factorization hardware
3. Dynamic reconfigurable processor DAPDNA2
4. Sieving algorithm on DAPDNA2
5. Demo & Evaluation
6. Conclusions & Future works

Conclusion

- Showed a ratio of the sieving time in GNFS by software
- Introduced the dynamic reconfigurable processor DAPDNA2
- Proposed the sieving algorithm for DAPDNA2
- Evaluated our implementation

Future Works



- Continue Implementations of
  - Bucket Sort Sieving
  - Trial Division
  - Lattice Sieve
  - etc...
- Try it on another target devices
  - FPGA
  - ASIC

FUJITSU

THE POSSIBILITIES ARE INFINITE