Blind Differential Cryptanalysis for Enhanced Power Attacks

Bart Preneel
COSIC – K.U.Leuven - Belgium

bart.preneel(AT)esat.kuleuven.be

Joint work with Helena Handschuh

SPANSION™

Concept

Differential cryptanalysis (DC) '90

Power analysis '99

Resistance against DC

Countermeasures (randomization, masking, circuit)

Combined attacks

Davies-Murphy

DC

?

Concept (2)

- Power analysis countermeasure: masking
  - Reduce RAM: mask only outer rounds
- Idea:
  - Apply DC on a few inner rounds
  - High probability characteristics, but filtering and key recovery impossible
  - Combine Hamming weight leakage with DC for key recovery

Outline

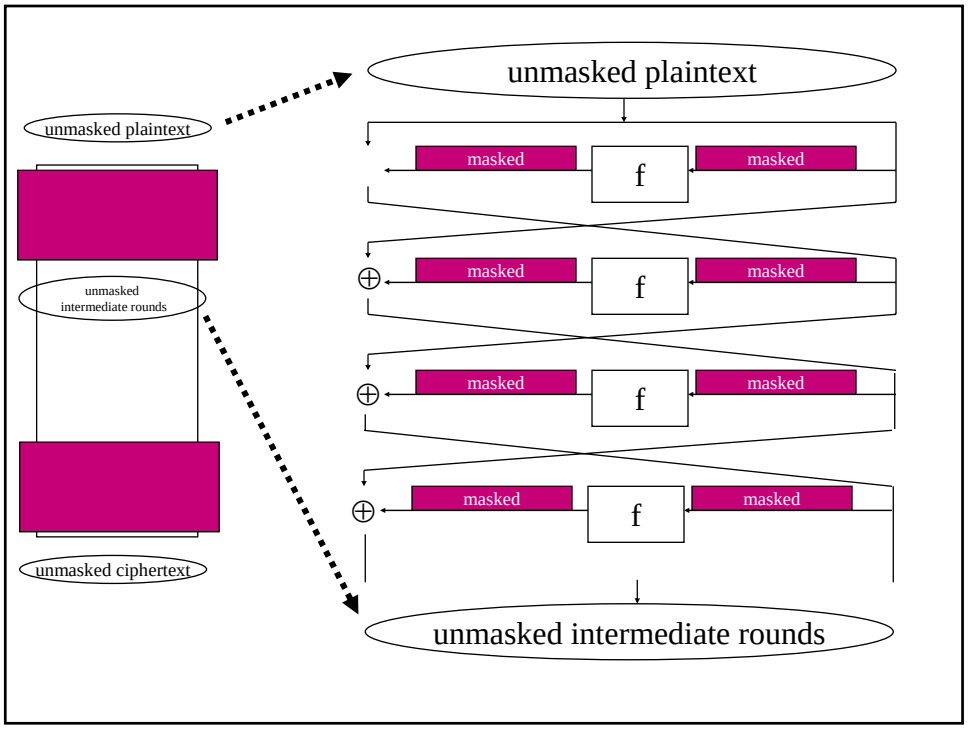
- Concept
- Related work
- Outer round masking of Feistel ciphers
- Differential cryptanalysis
- Blind differential cryptanalysis
- Filtering and key recovery
- Simulations
- Further research
- Conclusion

Related Work

- Akkar, Goubin. *A Generic Protection against High-Order Differential Power Analysis.* FSE 2003.
- Akkar, Bevan, Goubin. *Two Power Analysis Attacks against One-Mask Methods.* FSE 2004.
- Ledig, Muller, Valette. *Enhancing collision attacks.* CHES 2004.
- Schramm, Leander, Felke, Paar. *A Collision-Attack on AES: Combining Side Channel- and Differential-Attack.* CHES 2004.
- Osvik, Shamir, Tromer. *Cache attacks and countermeasures: The case of AES.* CT-RSA 2006.

Outer Round Masking: Example of Feistel Cipher DES

- Unique Masking Method proposed by Akkar and Goubin
  - use new masks for every execution of the algorithm
  - left second round output unmasked
  - shown to be vulnerable at FSE2004
- Extended Unique Masking Method
  - all outer rounds are masked
  - independent round masks and different S-boxes used for DES
  - proposal: four rounds are enough since after that, intermediate data depends on all key bits
- We show how to recover the secret key from unknown unmasked intermediate values using blind differential cryptanalysis



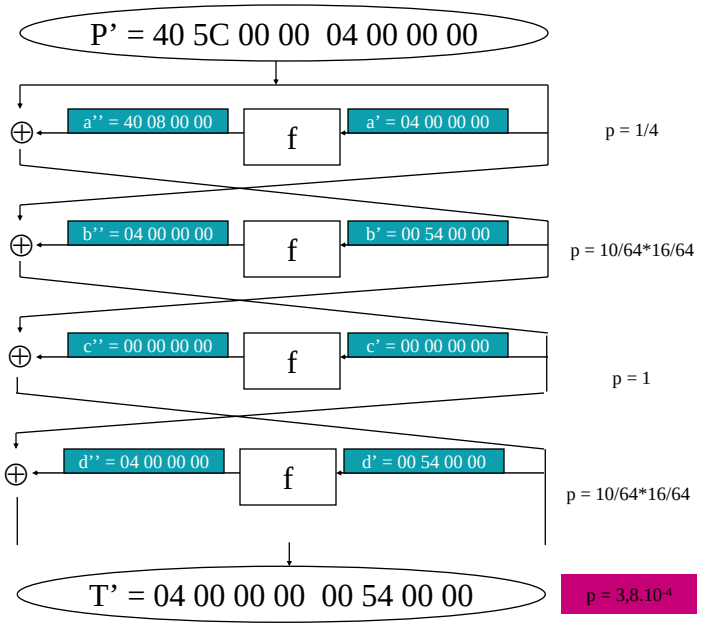
Differential Cryptanalysis

- [Biham-Shamir, 1990]
- Principle:
  - construct pairs of plaintexts with a fixed input difference
  - chose highly probability characteristic
  - filter right pairs based on their ciphertext difference
- Key recovery:
  - consider a right pair
  - only a few plaintexts from the difference distribution table of an S-box lead from a given input difference to a given output difference
  - deduce the key from these possible values and the ciphertext
  - wrong pairs add noise to the process

- Differential Cryptanalysis on inner rounds seems impossible since we have no access to intermediate values for filtering and key recovery

Blind Differential Cryptanalysis for Enhanced Power Attacks

- Power attacks [Kocher et al., 1999]
- Measure power consumption variations:
  - they depend on the secret data
  - Hamming weights: linear leakage model
  - cannot see actual intermediate values, but can measure Hamming weights
- UMM protects against power attacks and higher order power attacks
  - apply power attacks on inner unmasked rounds
  - combine with high probability differential characteristics
  - use Hamming weight information to estimate the intermediate unknown values.



Filtering Right Pairs

- Right pair = pair that follows the differential characteristic
- Our example has 7 equal 6-bit values on left half of intermediate text
  - implies equal Hamming weights for filtering
  - converse is not true!
  - introduces some noise in filtering process
  - works even when we don't have a 32-bit collision !
- Filtering:
  - probability of characteristic : 3.8×10^{-4}
  - probability of observing Hamming weight collisions : 3×10^{-5}
  - expected false alarms: 7.8%

Key Recovery on a Single Active S-box

- Regular differential key recovery:
  - deduce key from list of possible S-box input pairs and intermediate ciphertexts
  - here we only have Hamming weight of intermediate ciphertexts
  - list all possible texts having a given Hamming weight
  - **almost all key values are suggested: doesn't work !**
- Blind Key recovery :
  - use profiling technique
  - for a given key, partition possible round input pairs according to their Hamming weights.
  - every key has a different Hamming weight profile
  - **only depends on S-box and key value.**
  - can be pre-computed for every S-box and every cipher

Example Profiles for S-box S3 in DES

Profile of key 0x1f :

	0	1	2	3	4	5	6
0							
1		2					
2			2		1		
3				2		1	
4			1				
5				1			
6							

Profile of key 0x3c :

	0	1	2	3	4	5	6
0							
1				1			
2			2		2		
3		1		2			
4			2				
5							
6							

Note1: not all transitions can occur: some are impossible

Note2: there are only 10 different input/output pairs for the S-box S3 in DES

Key Recovery Procedure

1. Collect a sufficient number of right pairs; there will be some false alarms.
2. Compute an estimate for the Hamming weight profile PP*[i, j] based on the measurements.
3. Matching the observed profile and the profiles of all the values for the key k. Use a mean square error (MSE) as a matching criterion

$$MSE = \sum_{[i,j]} (PP_k[i, j] - PP_k^*[i, j])^2$$

4. The right 6 key bits are those for which the distance between the profiles is minimal.

Full Key Recovery

- Apply same technique to adjacent S-box S4 (16 input/output pairs).
- Allows to recover 12 bits of the secret key
- Select a new low-weight (and thus high probability) differential characteristic with different active S-boxes.
- Recover the whole key piece by piece.

Simulations

Plaintext Pairs	Right pairs	False alarms
2^{14}	3	1
2^{16}	26	4
2^{20}	408	69

- 20-30 right pairs suffice to recover the correct key in first position
- experiments simulated with several different keys
- ≈ 30.000 curves required
- computational complexity negligible

Improvements and Further Research

- Some keys work better than others (due to profiling)
- Optimize short characteristics for blind differential attacks

- Add noise to the leakage model
- Hamming weight transitions as a leakage model

- Attack works for Feistel Ciphers such as DES, 3DES but also Substitution Permutation Networks such as AES
- Attack needs good characteristics on a few inner rounds

Conclusion

- Masking only the outer rounds is insufficient
- Our attack does not depend on the masking technique

- Notion of blind differential attacks
- Notion of key profiling
- Combination between side-channel attacks and regular differential cryptanalysis

- Other combinations to be investigated

KATHOLIEKE UNIVERSITEIT LEUVEN

Superscalar Coprocessor for High-Speed Curve-based Cryptography

K. Sakiyama L. Batina
B. Preneel I. Verbauwhede

COSIC – K.U.Leuven – Belgium
IBBT

bart.preneel(AT)esat.kuleuven.be

Overview

- Introduction
- Curve-based Cryptography
- HW/SW Partitioning
- Superscalar Coprocessor
- Results
- Conclusions

Introduction - Motivation

- High-speed curve-based cryptography in HW/SW co-design
  - How much instruction-level parallelism can we obtain from coprocessor instructions?
- Performance improvement for different operation forms in datapath
  - AB+C mod P vs A(B+D)+C mod P A,B,C,D,P: polynomials
- Performance comparison three different curve-based cryptosystems
  - Which one is faster between ECC, HECC, ECC over a composite field?
- Programmability and scalability
  - Programmable in order to support different cryptosystems?
  - Scalable in field sizes?

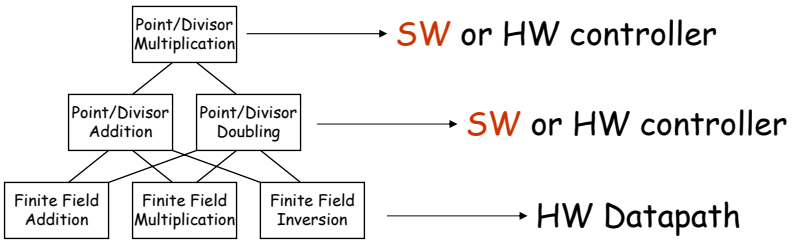
Introduction - Target Architecture

- Curve-based cryptography over binary fields
  - Hardware can be smaller and faster than prime field
  - ECC over binary field, e.g. $GF(2^{163})$
  - HECC of genus 2: field length 2x smaller, e.g. $GF(2^{83})$
  - ECC over composite field: field length 2x smaller, e.g. $GF((2^{83})^2)$

⇒
- The datapath can be shared
- Programmable coprocessor supporting three curve-based cryptography by defining coprocessor instruction(s)
- (Coprocessor) instruction-level parallelism by superscalar

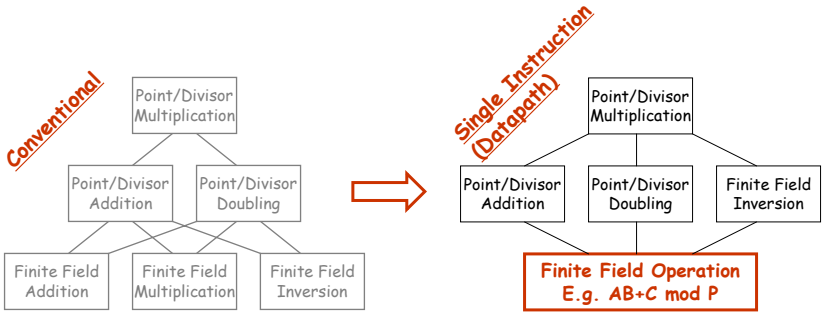
Curve-based Cryptography
HW/SW partitioning (1)

- General hierarchy in coprocessor for curve-based cryptography



Curve-based Cryptography
Proposed Hierarchy (1)

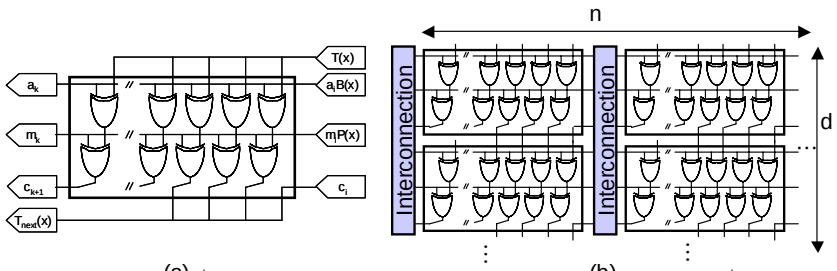
- Single instruction for all finite field operations
- Fixed-cycle execution enables efficient implementation



Curve-based Cryptography

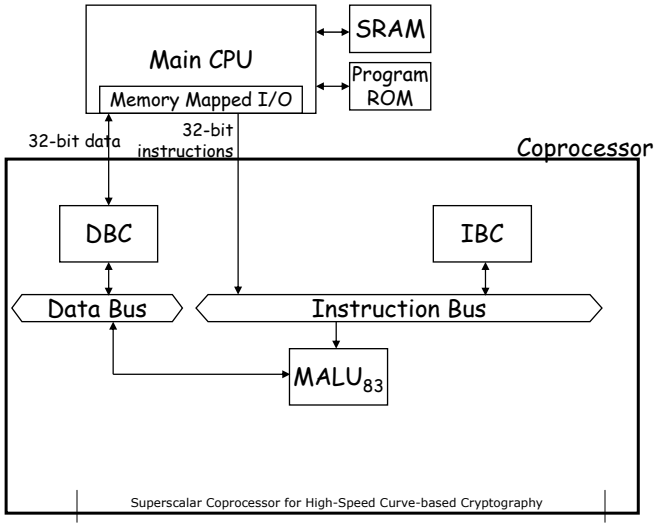
Modular Arithmetic Logic Unit (MALU)

- (a) Building block: Regular XOR chains
- (b) Scalable in digit size (d) and field size (k) by interconnecting several building blocks
- We use $MALU_{83}$ (n=83, d=12) as building block
- $2xMALU_{83}$ can be configured as $1xMALU_{163}$



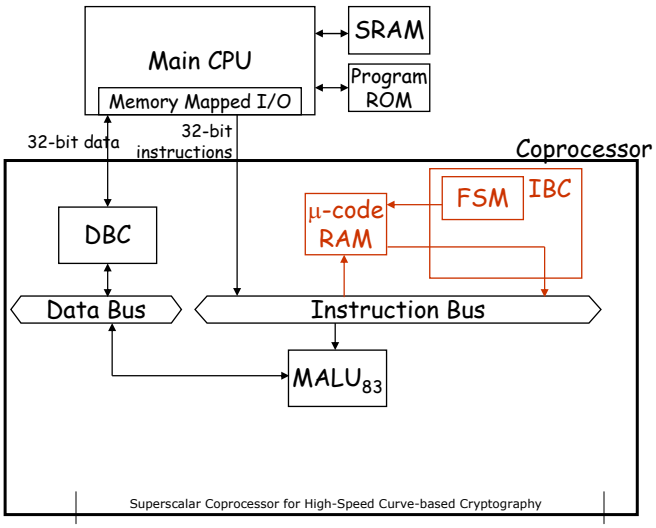
HW/SW Partitioning

TYPE I: Smallest implementation (baseline)



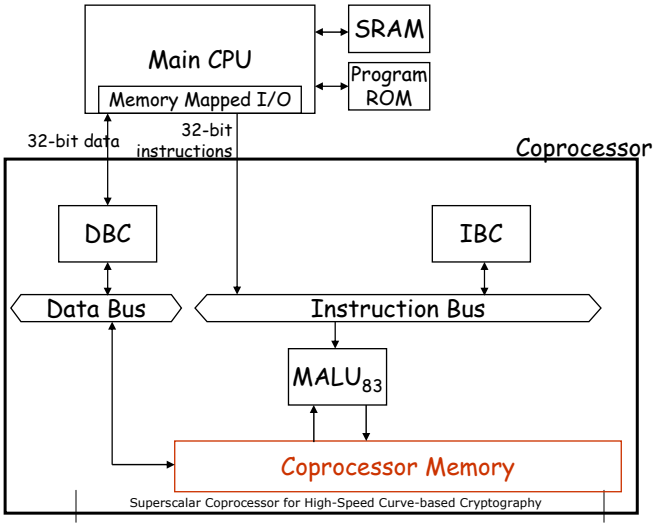
HW/SW Partitioning

TYPE II: TYPE I + μ-code RAM



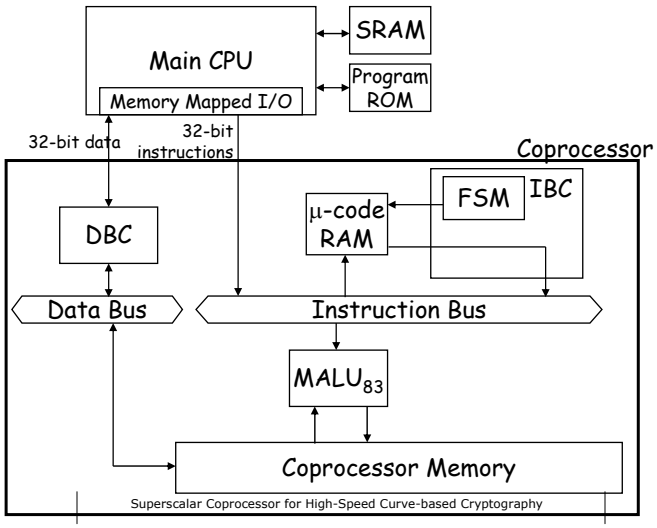
HW/SW Partitioning

TYPE III: TYPE I + Coprocessor Memory



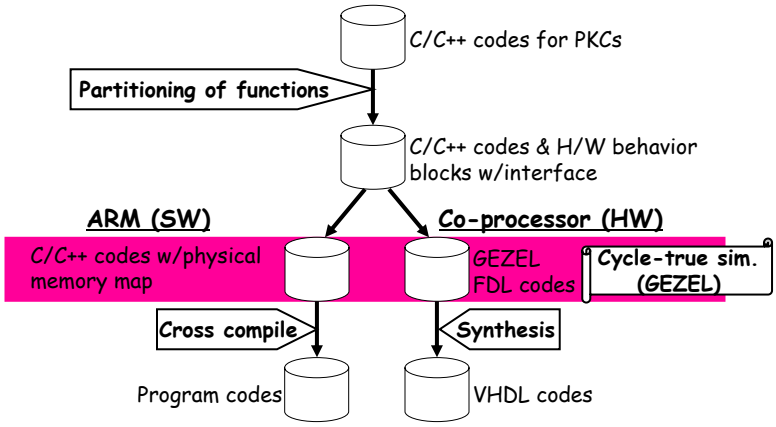
HW/SW Partitioning

TYPE IV: TYPE I + Copro. Mem.& μ-code RAM



HW/SW Partitioning

Co-design flow with GEZEL

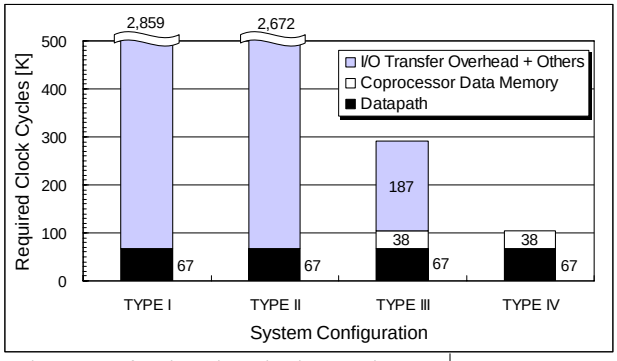


HW/SW Partitioning
Result: Vertical Exploration of System

- HECC Performance for different HW/SW partitioning
  (Performance: Point/Divisor multiplication)

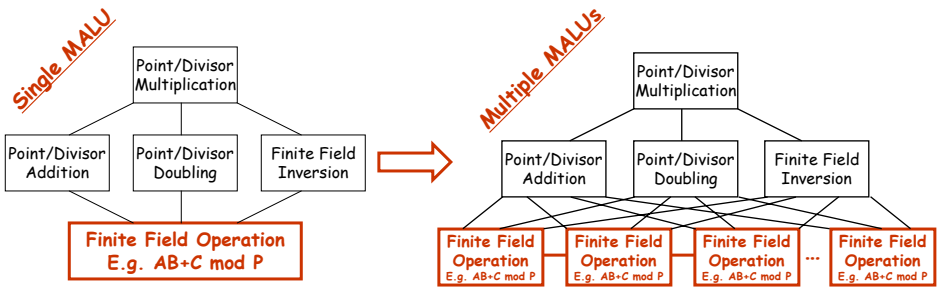
Coprocessor Configuration

	μ-code RAM	Data Mem.
TYPE I		
TYPE II	X	
TEPE III		X
TYPE IV	X	X



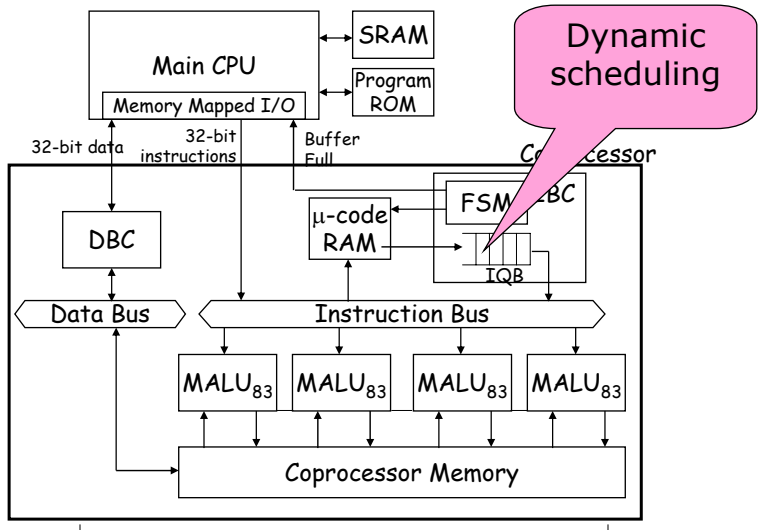
Superscalar Coprocessor
Proposed Hierarchy (2)

- Multiple Modular Arithmetic Logic Units (MALUs) in coprocessor



Superscalar Coprocessor

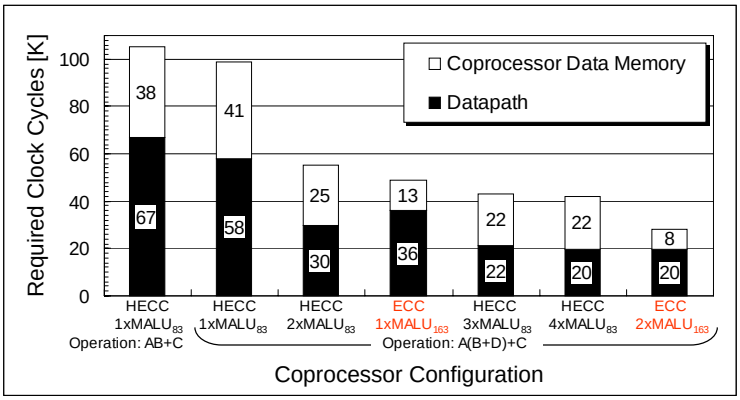
Parallel Processing Architecture (TYPE IV-based)



Superscalar Coprocessor

Horizontal Exploration of System

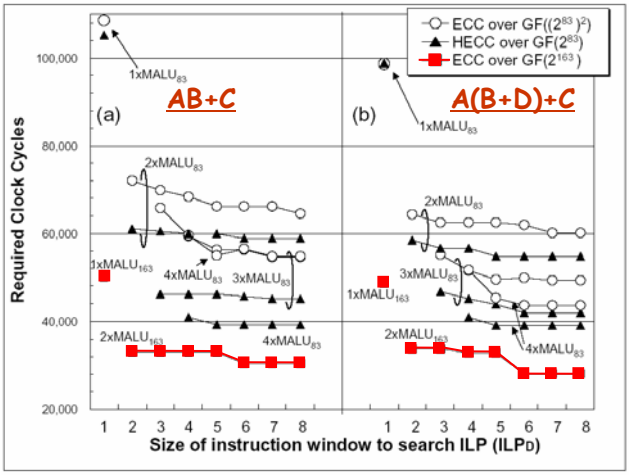
- Performance of ECC and HECC



- Further improvements: Montgomery's powering ladder (ECC) and improved memory management

Results

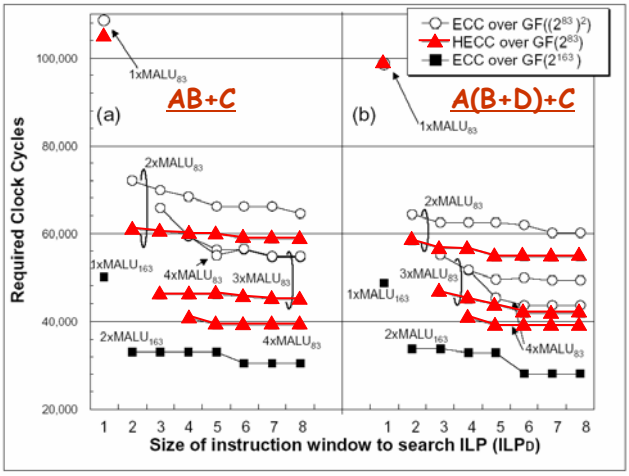
Performance of ECC over GF(2^{83})



- Fastest of three
- x1.8 speed-up by 2-way superscaling ($ILP_D P=6$) with A(B+D)+C
- Still more improvement is possible by adding MALUs

Results

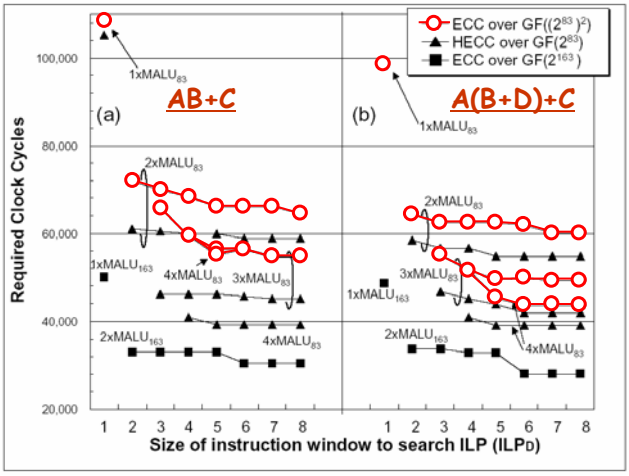
Performance of HECC over GF(2^{83})



- Faster than ECC over a composite field
- x2.7 speed-up by 4-way superscaling ($ILP_D P=5$) with A(B+D)+C
- Less improvement as increasing # of MALU

Results

Performance for ECC over GF((2^83)^2)



- Slowest of three
- x2.5 speed-up by 4-way superscaling ($ILP_DP=6$) with A(B+D)+C
- Less improvement as increasing # of MALU

Results

Comparison of ECC/HECC implementations on FPGAs

Ref. Design	Field	Target Platform	Area [slices/gates]	f_{max} [MHz]	Perform. [μsec]	Polynomial $P(x)$	Comments
HECC							
This work	$GF(2^{83})$	Virtex-II Pro	2,446 4,749 6,586	100.0	989 549 420	Arbitrary	$1\times MALU_{83}$ $2\times MALU_{83}$ $3\times MALU_{83}$
[11]	$GF(2^{81})$	Virtex-II Pro	4,039 7,737	57.0 60.7	787 387	Fixed	$2\times MULT, 1\times INV$ $3\times MULT, 2\times INV$
ECC							
This work	$GF(2^{163})$	Virtex-II Pro	4,749 8,450	100.0	488 280	Arbitrary	$1\times MALU_{163}$ $2\times MALU_{163}$
[14]	$GF(2^{163})$	Virtex E	19,508	66.5	1,554 143	Arbitrary Fixed: $x^{163}+x^7+x^6+x^3+1$	López-Dahab scalar mult.
[13]	$GF(2^{167})$	Virtex E	3,002 (+ 10 BRAMs)	76.7	210	Fixed: $x^{167}+x^6+1$	López-Dahab scalar mult.
[29]	$GF(2^{191})$	Virtex E	19,626 (+ 26 BRAMs)	9.99	59.26	Fixed: $x^{191}+x^9+1$	López-Dahab scalar mult.

[11] T. Wollinger, PhD thesis, 2004.
[13] G. Orlando and C. Paar, CHES 00.
[14] N. Gura *et al.*, CHES02.
[29] Nazar A. Saqib *et al.*, International Journal of Embedded Systems 2005

Results (2)

Comparison of ECC/HECC implementations for 0.13µ CMOS

ECC GF(2^{193})

f_{max}	Area	Performance (µsec)	comments
357	250 Kgates	86/61	6 MALU + 6 RF
357	121 Kgates	99/80	4 MALU + 2 RF'
416.7	0.59 mm^2	-/30	HW CU
510.2	117.5 Kgates	190/-	GF(p) support

HECC GF(2^{97})

f_{max}	Area	Performance (µsec)	comments
357	397 Kgates	122	6 MALU + 6 RF
357	250 Kgates	131	4 MALU + 6 RF'
357	250 Kgates	169	2 MALU + 1 RF

Conclusions

- **Performance improvement / Comparison**
  - ECC: improved by a factor of 1.8 (2-way) to 2.5 (8-way)
  - HECC (genus 2): improved by a factor of 2.7 (4-way)
  - ECC over a composite field: improved by a factor of 2.5 (4-way)
  - A(B+D)+C offers better performance than AB+C
  - ECC is the fastest in this case study
- **Programmability & flexibility**
  - Support 3 different curve-based cryptosystems over binary field
  - Arbitrary irreducible polynomial
  - Field size up to 332 bits by using 4x$MALU_{83}$ (582 bits using 6x$MALU_{97}$)