A HIGH SPEED SCALAR MULTIPLIER FOR BINARY EDWARDS CURVES.

A. P. Fournaris, *N. Sklavos and C. Koulamas*

Industrial Systems Institute Research Center ATHENA, Greece

Computer Informatics Engineering Dpt, Technological Educational Institute of Western Greece, Greece

Computer Engineering and Informatics Dpt, University of Patras, Patra, Greece

OUTLINE

- Elliptic Curve for Cryptography and Edwards Curves
- Motivation
- Proposed Concept
- Proposed Design Approach – Parallelism
- Employed Algorithms
- Proposed Architecture
- Results- Comparisons

ELLIPTIC CURVE FOR CRYPTOGRAPHY

- Scalar Multiplication main crypto-operation
- Elliptic Curves described in various forms:
  - Weierstrass form (most popular, standardized NIST)
  - Hessian form
  - Montgomery form
  - Edwards form
  - ….
- Popular Elliptic Curves (EC) defined over:
  - Prime Fields GF(p):
    Efficient software implementations
  - Binary Extension Fields $GF(2^k)$:
    Efficient hardware implementations

ELLIPTIC CURVE ARITHMETIC

Point Addition: add two points of the Elliptic Curve to get a third point of the Elliptic Curve $P_3=(x_3,y_3)=P_1+P_2$

Point Doubling: add one Elliptic Curve point with itself $P_3 = 2P_1$

Scalar Multiplication : add one Elliptic Point with itself e times $Q = e \cdot P$

- Can be analyzed in a series of point additions and point doublings

Point operations rely on GF(2^k) operations:

- GF(2^k) multiplication and inversion: computationally demanding
- Exchange GF(2^k) inversion with several multiplications to reduce computation cost by transforming point coordinates from the affine to the projective space

EDWARDS CURVES VS WEIERSTRASS CURVES

- Weierstrass EC equation do not provide unified symmetric approach for Point addition and Doubling. There are exception points (eg point at Infinity).
  - Problem: Exception points can be exploited for side channel and fault injection analysis attacks !!
- Weierstrass ECs are not complete. The Group law for point addition is different than the one for Point doubling.
- Edwards ECs have a unified, symmetric group law. The same equations can be used for point addition and for point doubling
- Edwards ECs have no exception points. There are complete.
- Unified Group Law + No exception Points = Edwards ECs intrinsically resistant against simple side channel attacks
- A point operation in Edwards Curves needs more $GF(2^k)$ operations than in Weierstrass Curves

DESIGN APPROACHES AND MOTIVATION

In Edwards curve projective coordinates, 2 GF(2^k) inversions (I) are exchanged with 13 Multiplications (M) for point addition (PA).

PA Total cost: **18M**+3S+6D+24A (*higher cost than PA in Weirstrass ECs*)

GF(2^k) multiplication approaches:

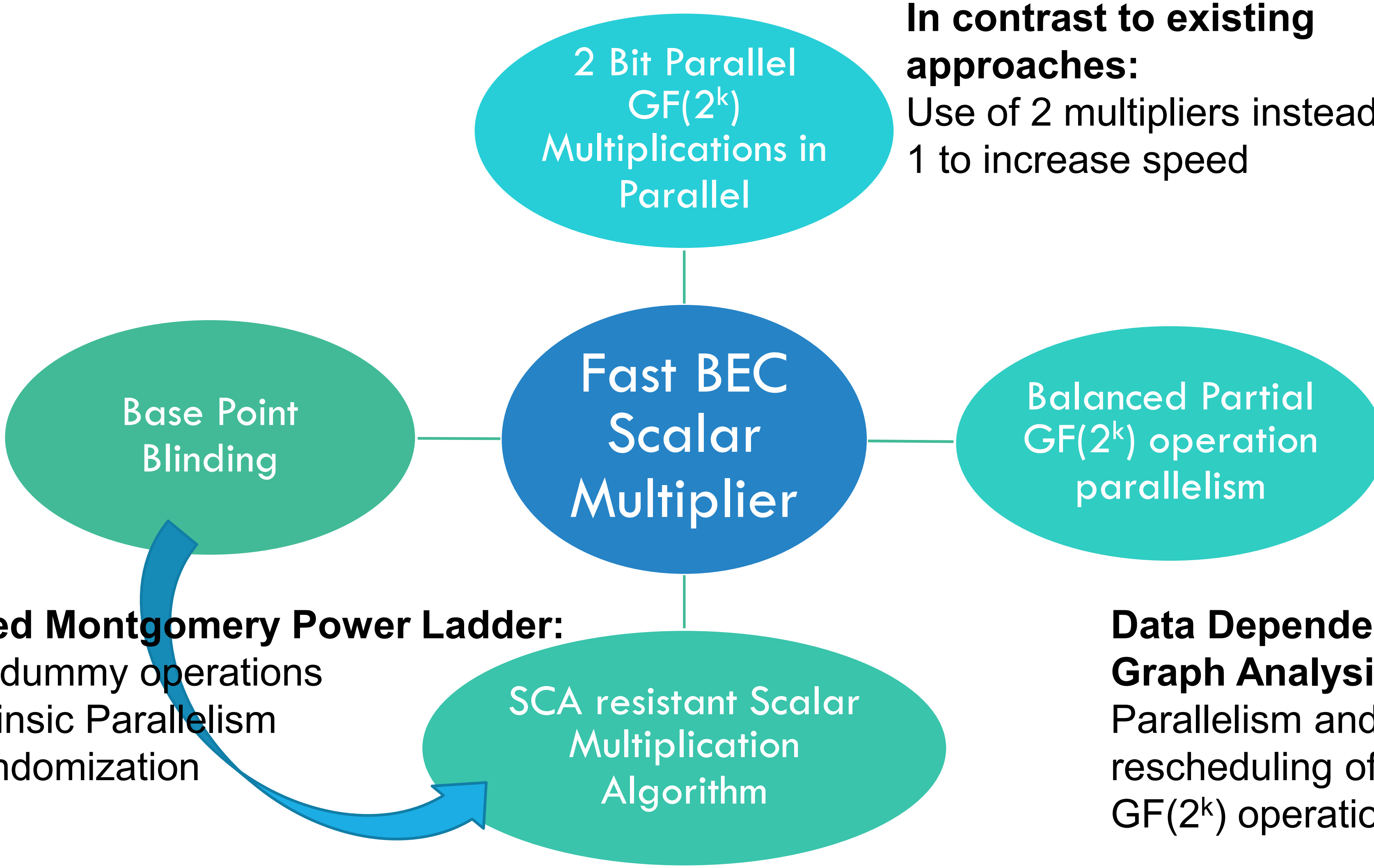
- Bit Serial multipliers: slow but small number of gates and flexible (can be reused for various curves and GF(2^k))
- Bit parallel multipliers: Fast but high number of gates and not flexible.
- Digit Serial multipliers: A compromise between bit serial and bit parallel approach

Can we design an Edwards curve scalar multiplier with similar performance characteristics as Weierstrass curve designs?

<u>WHY: BECs offer a solid base for strong Side Channel Attack Resistance.</u>

PROPOSED SOLUTION CONCEPT

In contrast to existing approaches:
Use of 2 multipliers instead of 1 to increase speed



Blinded Montgomery Power Ladder:

- No dummy operations
- Intrinsic Parallelism
- Randomization

Data Dependency Graph Analysis:
Parallelism and rescheduling of GF(2^k) operations

BLINDED MONTGOMERY POWER LADDER

Algorithm 2. SPA resistant MPL algorithm

Input: P : BEC base point $\in EC(GF(2^k))$,
$e = (e_{t-1}, e_{t-2}, ...e_0) \in GF(2^k)$
Output: $e \cdot P$
1. $R_0 = \mathcal{O}$, $R_1 = P$
2. **For** $i = t - 1$ **to** 0
   If $(e_i = 0)$ then
   (a) $R_1 = R_0 + R_1$, $R_0 = 2 \cdot R_0$
   else
   (b) $R_0 = R_0 + R_1$, $R_1 = 2 \cdot R_1$
   end if
3. **Return** R_0

Algorithm 3. Blinded MPL (bMPL) algorithm

Input: P : BEC base point, random points
$R, -R \in EC(GF(2^k))$, $e = (e_{t-1}, e_{t-2}, ...e_0) \in GF(2^k)$
Output: $e \cdot P$
1. $R_0 = R$, $R_1 = R + P$, $R_R = -R$,
2. **For** $i = t - 1$ **to** 0
   (a) $R_R = 2R_R$
   If $(e_i = 0)$ then
   (b) $R_1 = R_0 + R_1$, $R_0 = 2 \cdot R_0$
   else
   (c) $R_0 = R_0 + R_1$, $R_1 = 2 \cdot R_1$
   end if
3. **Return** $R_0 + R_R$

2 PD
1 PA
In parallel
each round

STEP 1: BREAK PA AND PD INTO SINGLE GF(2^K) OPERATIONS

PA: 19 M +2 S + 22 A

PD: 4 M +6 S + 9 A

$(X_{3D} : Y_{3D} : Z_{3D}) = 2(X_1 : Y_1 : Z_1)$
$DA = X_1^2$
$DC = Y_1^2$
$DE = Z_1^2$
$DB = DA^2$
$DD = DC^2$
$DF1 = DE^2$
$DH = DA \cdot DE$
$DI = DC \cdot DE$
$DF = d_1 \cdot DF1$
$DG = DB + DD$
$DV2 = DH + DD$
$DV3 = DI + DB$
$DJ = DH + DI$
$DV1 = DF + DG$
$DK1 = d_2 \cdot DJ$
$DK = DG + DK1$
$Z_{3D} = DV1 + DJ$
$X_{3D} = DK + DV2$
$Y_{3D} = DK + DV3$

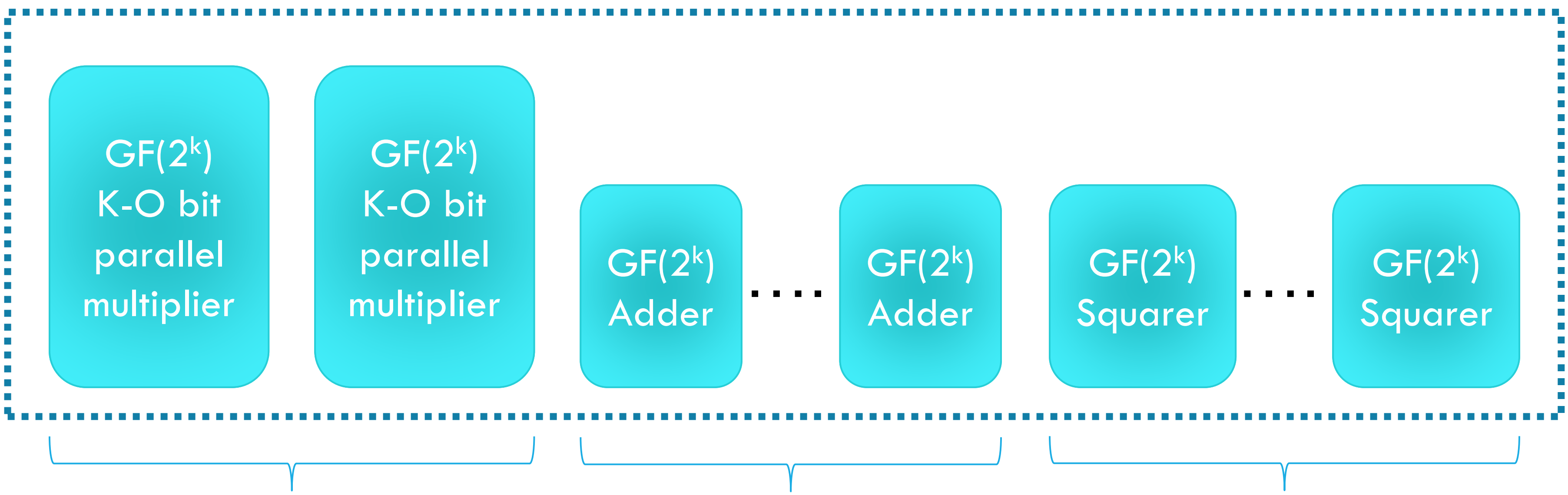
$(X_3 : Y_3 : Z_3) = (X_1 : Y_1 : Z_1) + (X_2 : Y_2 : Z_2)$	
$A = X_1 \cdot Y_1$	$V1 = A \cdot B$
$B = Y_1 \cdot Y_2$	$L2 = L1 + F$
$C = Z_1 \cdot Z_2$	$Z_3 = C \cdot L2$
$D = d_1 \cdot C$	$V2 = G \cdot H$
$E = C^2$	$V3 = d_1 \cdot E$
$F = D^2$	$V4 = V1 + V2$
$G1 = X_1 + Z_1$	$V5 = V3 + V4$
$G2 = X_2 + Z_2$	$L3 = L \cdot V5$
$H1 = Y_1 + Z_1$	$V6 = D \cdot F$
$H2 = Y_2 + Z_2$	$V7 = L3 + V6$
$G = G1 \cdot G2$	$V = V7 + U$
$H = H1 \cdot H2$	$S1 = A + D$
$I = A + G$	$S2 = G + D$
$J = B + H$	$S3 = S1 \cdot S2$
$K1 = X_1 + Y_1$	$S4 = D \cdot S3$
$K2 = X_2 + Y_2$	$X_3 = V + S4$
$K = K1 \cdot K2$	$T1 = B + D$
$L = d_1 \cdot K$	$T2 = H + D$
$U1 = K + I$	$T3 = T1 \cdot T2$
$U2 = J + C$	$T4 = D \cdot T3$
$U3 = U1 + U2$	$Y_3 = V + T4$
$L1 = L \cdot U3$	

Binary Edwards EC equation:

$$d_1(x + y) + d_2(x^2 + y^2) = xy + xy(x + y) + x^2y^2$$

STEP 2: PARALLELISM SCHEME DDG ANALYSIS

One parallelism layer (all operations performed in parallel)



Strictly 2 Multipliers

Minimum to retain high use throughout all parallelism layers

DDG outcome: 3 adders

Minimum to retain high use throughout all parallelism layers

DDG outcome: 2 squarers

DATA DEPENDENCY GRAPH ANALYSIS
(CONSTRAINED FOR 2 GF(2^K) MULTIPLIERS PER LAYER)

One bMPL round:

2PD and 1 PA:

Needed GF(2^k) operations per round:

27 M + 14 S + 40 A

Parallelism layer dictated by M (2 per layer) →

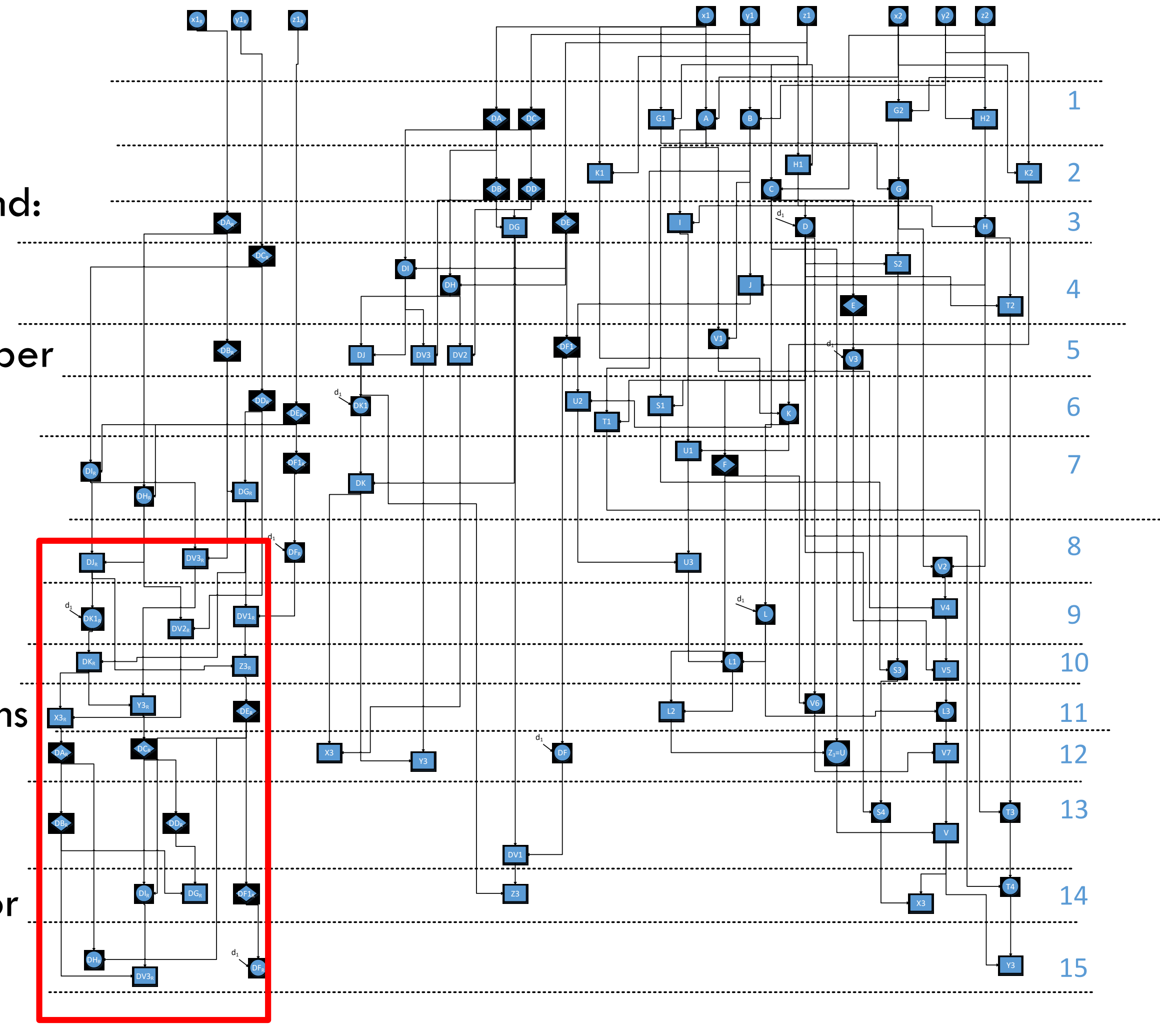
$\left\lceil \frac{27}{2} \right\rceil$ layers +1 final layer

i.e. **15 layers**

Assuming 3 adders and 2 squarers per layers:
Available 45 A and 30 S operations per round.

Needed 40 A and 14 S.

The unused operations employed for <u>precomputing next round's PD</u>



PROPOSED PARALLELISM

1^{st} bMPL round

Input		$(X_1 : Y_1 : Z_1)$		$(X_2 : Y_2 : Z_2)$		$(X_0 : Y_0 : Z_0)$	
layer	M1	M2	Sq1	Sq2	Ad1	Ad2	Ad3
1	A	B	DA	DC	G1	G2	H2
2	C	G	DB	DD	H1	$K2$	K1
3	D	H	DE	DA_R	I	DG	*
4	DI	DH	DC_R	E	J	S2	T2
5	V1	V3	$DF1$	DB_R	DJ	$DV2$	DV3
6	DK1	K	DE_R	DD_R	S1	T1	U2
7	DI_R	DH_R	$DF1_R$	F	U1	$DG1_R$	DK
8	$V2$	DF_R		*	DJ_R	$DV3_R$	U3
9	L	$DK1_R$	*	*	$DV2_R$	$DV1_R$	V4
10	S3	L1	*	*	DK_R	Z_R	V5
11	L3	V6	DE'_R	*	X_R	Y_R	L2
12	DF	Z_3	DA'_R	DC'_R	X_{3D}	Y_{3D}	$V7$
13	T3	$S4$	DB'_R	DD'_R	V	$DV1$	*
14	$T4$	DI'_R	$DF1'_R$	*	X_3	Z_{3D}	DG'_R
15	DH'_R	DF'_R	*	*	Y_3	$DV3'_R$	*
Out	$(X_3 : Y_3 : Z_3)$		$(X_{3D} : Y_{3D} : Z_{3D})$			$(X_R : Y_R : Z_R)$	

* : dummy operation

Precomputing operations for next round's R_R PD, reduces remaining rounds layers to 14

Remaining bMPL rounds

Inputs		$(X_1 : Y_1 : Z_1)$		$(X_2 : Y_2 : Z_2)$		$(X_0 : Y_0 : Z_0)$	
layer	M1	M2	Sq1	Sq2	Ad1	Ad2	Ad3
1	A	B	DA	DC	G1	G2	H2
2	C	G	DB	DD	H1	$K2$	K1
3	D	H	DE	*	I	DG	$DV1_R$
4	DI	DH	E	*	J	S2	T2
5	V1	V3	$DF1$	*	DJ	$DV2$	DJ_R
6	$DK1_R$	K	*	*	S1	$DV2_R$	U2
7	$DK1$	V2	*	F	U1	DK_R	T1
8	L	DF	*	*	Z_R	V4	U3
9	$S3$	$L1$	DE'_R	*	X_R	Y_R	V5
10	V6	L3	DA'_R	DC'_R	DK	$DV3$	$L3$
11	T3	Z_3	DB'_R	DD'_R	$DV1$	Y_{3D}	V7
12	T4	S4	$DF1'_R$	*	X_{3D}	Z_{3D}	V
13	DF'_R	DI'_R	*	*	X_3	Y_3	DG'_R
14	DH'_R	*	*	*	*	$DV3'_R$	*
Out	$(X_3 : Y_3 : Z_3)$		$(X_{3D} : Y_{3D} : Z_{3D})$			$(X_R : Y_R : Z_R)$	

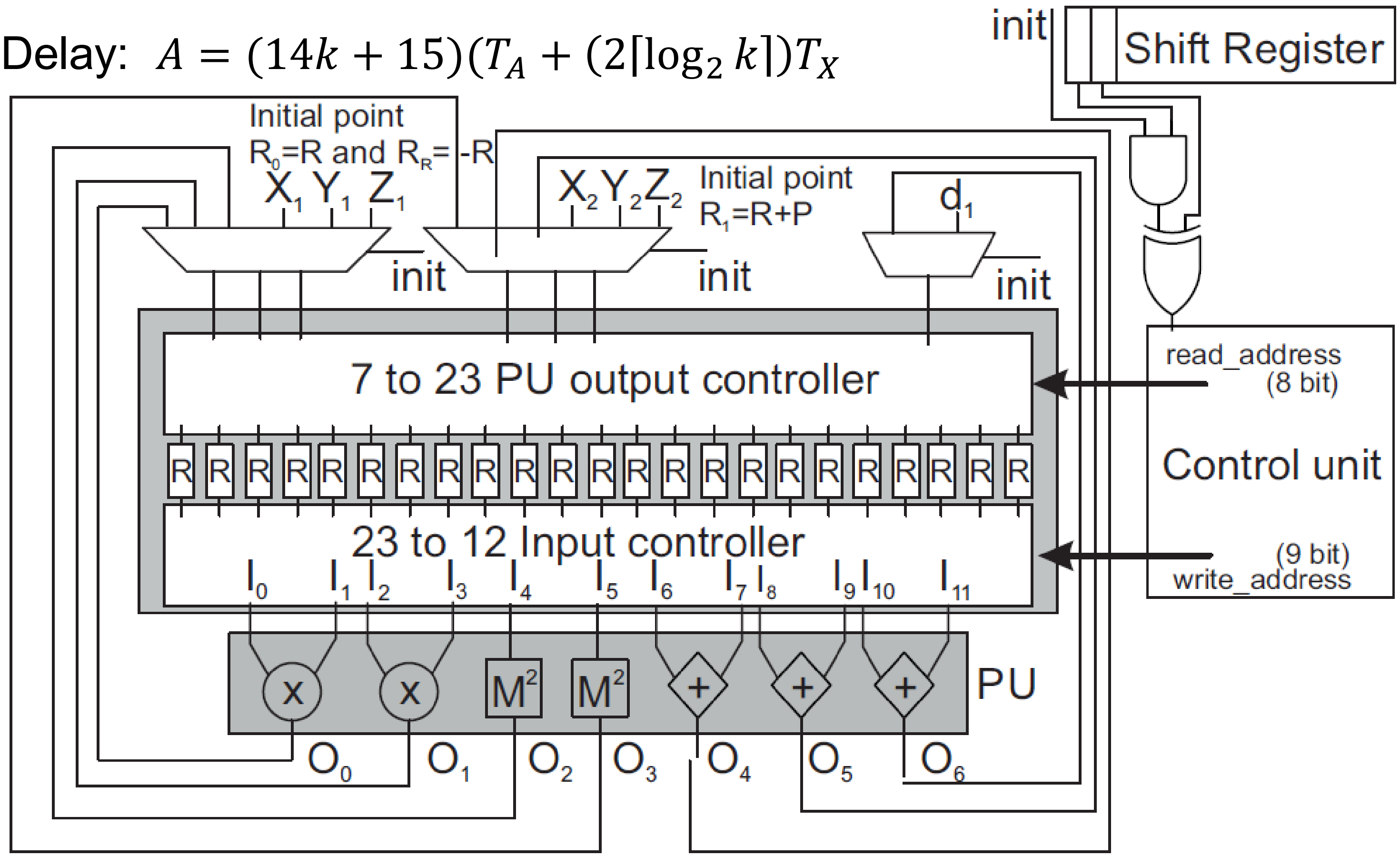
* : dummy operation

Each layer needs 1 clock cycle to come up with a result.

clock cycle time period dictated by the $GF(2^k)$ multiplier

PROPOSED BEC SCALAR MULTIPLIER ARCHITECTURE

Total Delay: $A = (14k + 15)(T_A + (2\lceil \log_2 k \rceil)T_X$



Multiplier unit: bit parallel Karatsuba-Ofman based on

H. Fan, J. Sun, M. Gu, and K.-Y. Lam, "Overlap-free Karatsuba–Ofman polynomial multiplication algorithms," *IET Information Security*, vol. 4, no. 1, p. 8, 2010.

BEC SCALAR MULTIPLIER IMPLEMENTATION RESULTS-COMPARISONS

arch.	techn.	k	Area	max Freq.	time delay	effic.	SCA resist.	
prop.	XC5VLX110	233	32874	132	0.025	0.81	Point Rand	BEC
prop.	XC4VFX140	233	40793	67	0.049	1.97	Point Rand	BEC
[11]	XC4V140	233	35003	47	0.19	6.65	intrinsic SPA	BEC
[29]	XC5VLX110	233	18097	156	0.012	0.2	No	WS
[3]	XC5VLX110	163	17305	262	0.013	0.22	intrinsic SPA	BEC
[3]	XC5VLX110	233	~25000	~200	~0.025	0.6	Intrinsic SPA	

Prop. in Xilinx Virtex 4 better than BEC [11] (faster + better SCA resistance)

Prop. In Xilinx Virtex 5 same speed as normalized BEC [2] but worst Area (note that very rough estimations are made) but offers better SCA resistance.

Prop. In Xilinx Virtex 5 speed close to Weierstrass ECs of [29]. Still more optimizations are needed but [29] results achieved with no SPA/SCA resistance.

CONCLUSIONS – FUTURE WORK

Come close to Weierstrass ECs scalar multiplier performance through parallelism and increasing the number of parallel components (2 $GF(2^k)$ multipliers instead of 1).

Future Work:

- Explore more compact multipliers to save chip covered areas like hybrid or digit serial multipliers
- Explore Different, less costly randomization approaches exploiting BEC intrinsic resistance (randomized projective coordinates??)

QUESTIONS?

Funded by GSRT Action KRIPIS: "ISRTDI: Industrial Systems for Sustainable Development and Wellbeing - Research, Technological Development and Innovation"

cofunded by EU COST action IC1204 Trustworthy Manufacturing & Utilization of Secure Devices

End of Presentation

Thank You!