Multiplierless Design of High-Speed Very Large Constant Multiplications

Levent Aksoy†, Debapriya Basu Roy‡, Malik Imran† and Samuel Pagliarini†

†Tallinn University of Technology, Tallinn, Estonia

‡Indian Institute of Technology Kanpur, Kanpur, India

Asia and South Pacific Design Automation Conference, Incheon, South Korea, 25th of January 2024

Outline

- Introduction
- Background
- Contributions
- Proposed Method
- Experimental Results
- Conclusions

Introduction

- Multiplication of constants by a variable is a ubiquitous operation
  - digital signal processing [1] and cryptography [2]
- Constant multiplications can be realized without multipliers
  - adders, subtractors, and shift operations under the shift-adds architecture
- In cryptography, constants can be very large
  - Elliptic Curve Cryptography (ECC): 204-521
  - Singular Isogeny Key Encapsulation (SIKE): 448-768
- The **Very Large Constant Multiplication (VLCM)** problem
  - find a minimum number of adders/subtractors which realize the multiplication of given very large constants by a variable

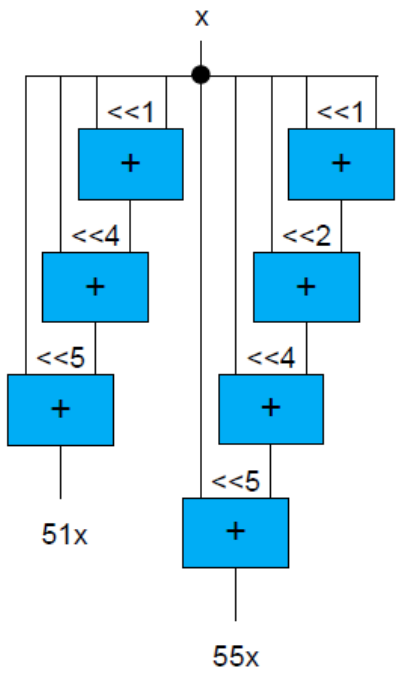
[1] H. Nguyen and A. Chatterjee, "Number-Splitting with Shift-and-Add Decomposition for Power and Hardware Optimization in Linear DSP Synthesis," IEEE TVLSI, 8(4), 419–424, 2000.
[2] D. B. Roy and D. Mukhopadhyay, "High-Speed Implementation of ECC Scalar Multiplication in GF(p) for Generic Montgomery Curves," IEEE TVLSI, 27(7), 1587–1600, 2019.

Background

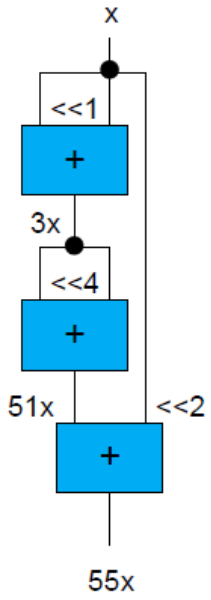
$51x = (110011)_{bin}x = x<<5 + x<<4 + x<<1 + x$

$55x = (110111)_{bin}x = x<<5 + x<<4 + x<<2 + x<<1 + x$

Digit-based Recoding [3]



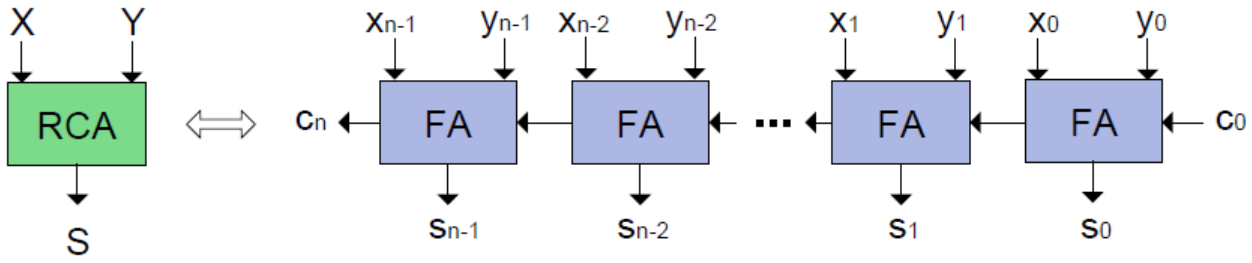
GB Algorithm [4]



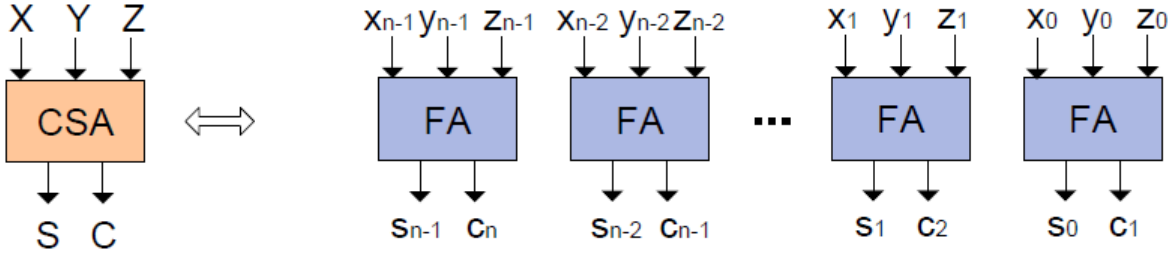
[3] M. Ercegovac and T. Lang, Digital Arithmetic. Morgan Kaufmann, 2003.
[4] Y. Voronenko and M. Püschel, “Multiplierless Multiple Constant Multiplication,” ACM Transactions on Algorithms, vol. 3, no. 2, 2007.

Background

Ripple Carry Adder (RCA)



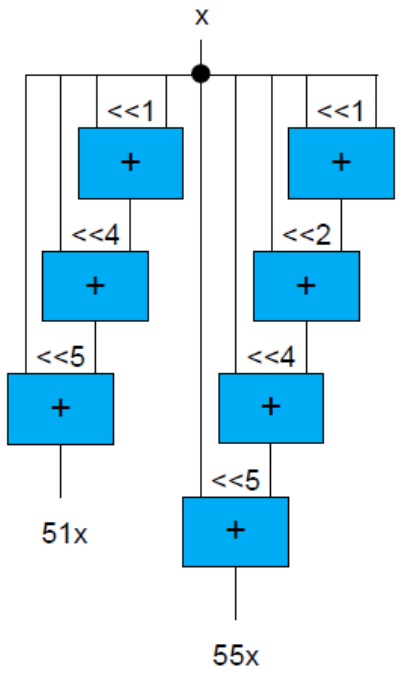
Carry Save Adder (CSA)



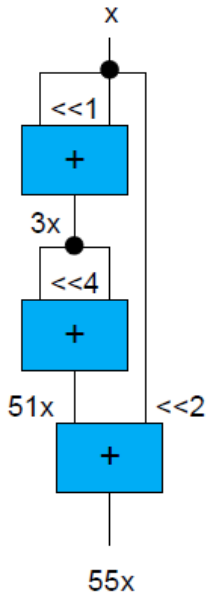
Background

$51x = (110011)_{bin}x = x<<5 + x<<4 + x<<1 + x$
$55x = (110111)_{bin}x = x<<5 + x<<4 + x<<2 + x<<1 + x$

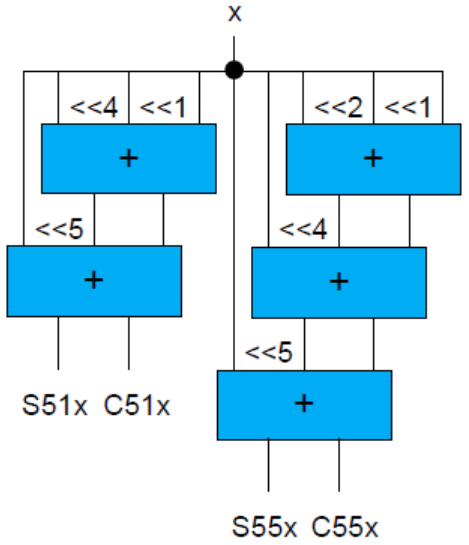
Digit-based Recoding [3]



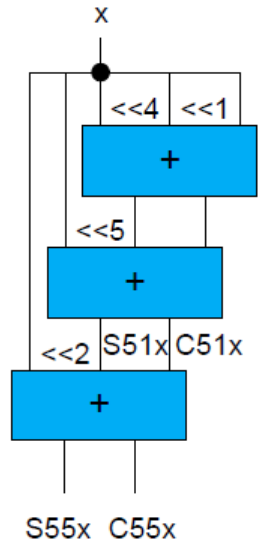
GB Algorithm [4]



Digit-based Recoding [3]



CSE Algorithm [5]



[3] M. Ercegovac and T. Lang, Digital Arithmetic. Morgan Kaufmann, 2003.
[4] Y. Voronenko and M. Püschel, "Multiplierless Multiple Constant Multiplication," ACM Transactions on Algorithms, vol. 3, no. 2, 2007.
[5] A. Hosangadi, F. Fallah, and R. Kastner, "Optimizing High Speed Arithmetic Circuits using Three-Term Extraction," in DATE, 2006, pp. 1–6.

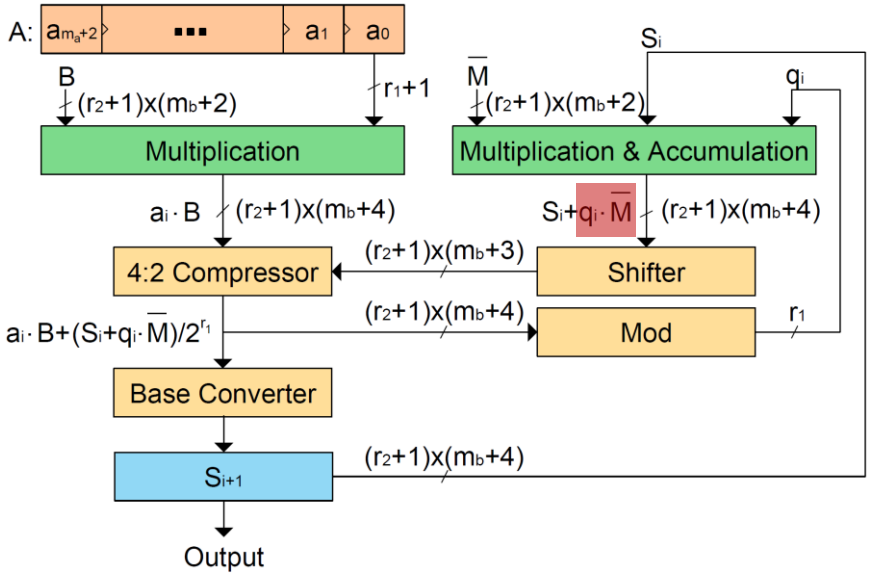
Background

Constant Time Montgomery Multiplication [2]

Input: $M = \sum_{i=0}^{m-1} m_i \cdot 2^{ri}$, $A = \sum_{i=0}^{m+2} a_i \cdot 2^{ri}$ with $a_{m+2} = 0$, $B = \sum_{i=0}^{m+1} b_i \cdot 2^{ri}$, $M' = -M^{-1} \bmod R$, $\overline{M} = (M' \bmod 2^r) \cdot M = \sum_{i=0}^{m+1} \overline{m}_i \cdot 2^{ri}$, $A, B < 2\overline{M}$, $4\overline{M} < 2^{rm}$, $R = 2^{r(m+2)}$
Output: $A \times B \times R^{-1} \bmod M$

1: S_0=0
2: **for** $i \leftarrow 0$ **to** $m+2$ **do**
3: $\quad q_i = S_i \bmod 2^r$
4: $\quad S_{i+1} = (S_i + q_i \cdot \overline{M})/2^r + a_i \cdot B$
5: **return** $S_{m+3} = A \times B \times R^{-1} \bmod M$

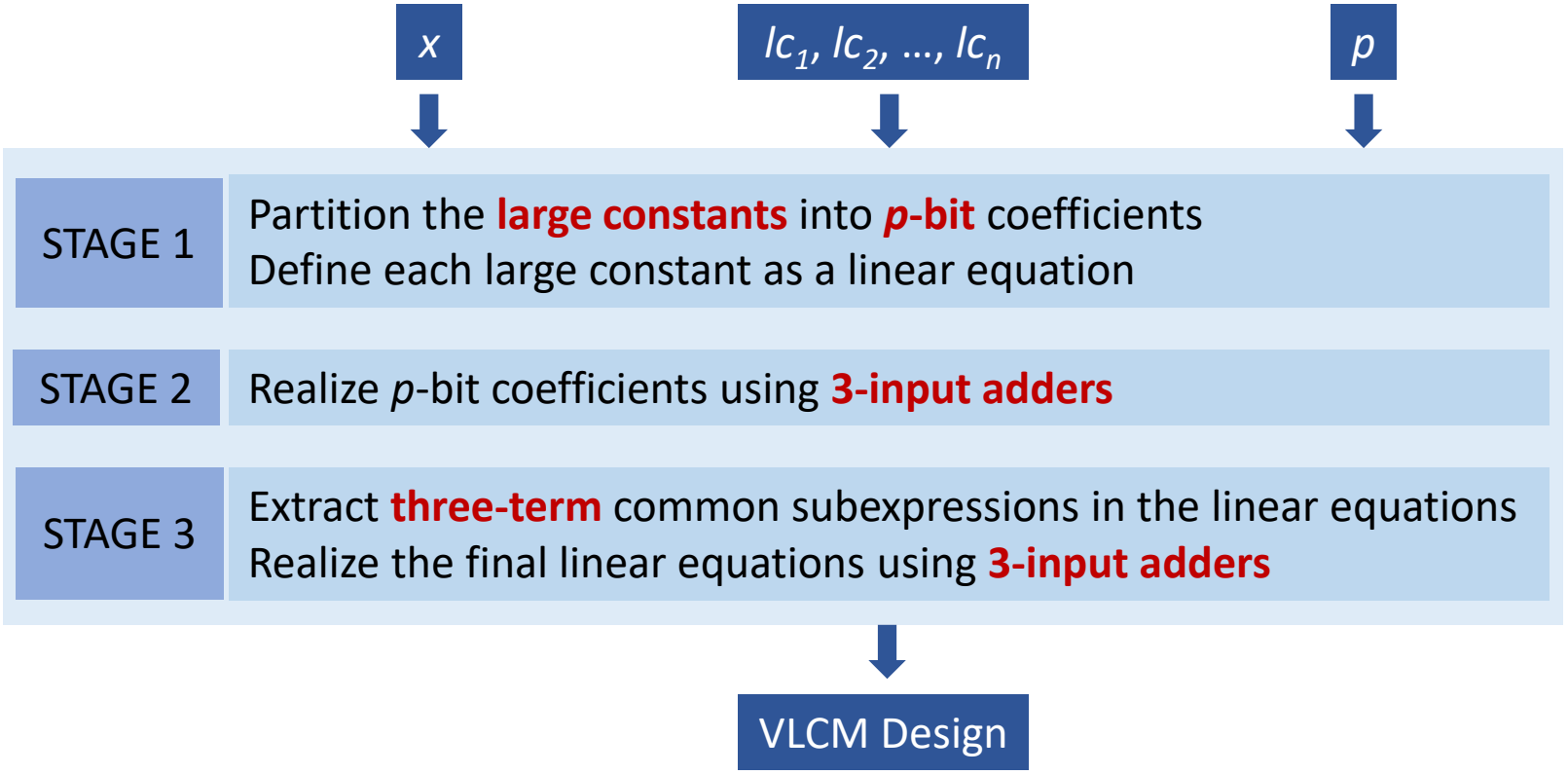
Montgomery Multiplier Architecture [2]



Contributions

- The CAD tool LEIGER that implements multiplierless high-speed VLCM operation
  - shift-adds architecture using CSAs (SA-CSA)
  - shift-adds architecture using 2- and 3-input adders (SA-Hybrid)
  - design architecture using compressor trees (CT)
- Implementation of the Montgomery multiplication using LEIGER

LEIGER – SA-CSA



LEIGER – SA-CSA – Partitioning

Large constants

lc_1 = 1,309,873,459 = 0x4E131533

lc_2 = 2,623,935,253 = 0x9C661315

Partitioning when p is 8

lc_1 = | 4E | 13 | 15 | 33 |

lc_2 = | 9C | 66 | 13 | 15 |

Coefficients

0x33 = 51 | 0x15 = 21 | 0x13 = 19 | 0x4E = 78 | 0x66 = 102 | 0x9C = 156

C = {51, 21, 19, 78, 102, 156}

Linear equations

lc_1 = S&C78<<24 + S&C19<<16 + S&C21<<8 + S&C51

lc_2 = S&C156<<24 + S&C102<<16 + S&C19<<8 + S&C21

LEIGER – SA-CSA – Realization of Coefficients

Coefficients

C = {51, 21, 19, 78, 102, 156}

Subexpressions

s&c19 = 1<<4 + 1<<1 + 1

s&c39 = s&c19<<1 + 1

s&c21 = 1<<4 + 1<<2 + 1

s&c51 = s&c19 + 1<<5

Realization of coefficients

S&C51 = s&c51

S&C21 = s&c21

S&C19 = s&c19

S&C78 = s&c39<<1

S&C102 = s&c51<<1

S&C156 = s&c39<<2

LEIGER – SA-CSA – Realization of Linear Equations

Linear equations

lc_1 = S&C78<<24 + S&C19<<16 + S&C21<<8 + S&C51

lc_2 = S&C156<<24 + S&C102<<16 + S&C19<<8 + S&C21

Common subexpressions

$S\&Cexp_0$ = S&C19<<8 + S&C21

Final linear equations

lc_1 = S&C78<<24 + $S\&Cexp_0$<<8 + S&C51

lc_2 = S&C156<<24 + S&C102<<16 + $S\&Cexp_0$

Realization of final linear equations

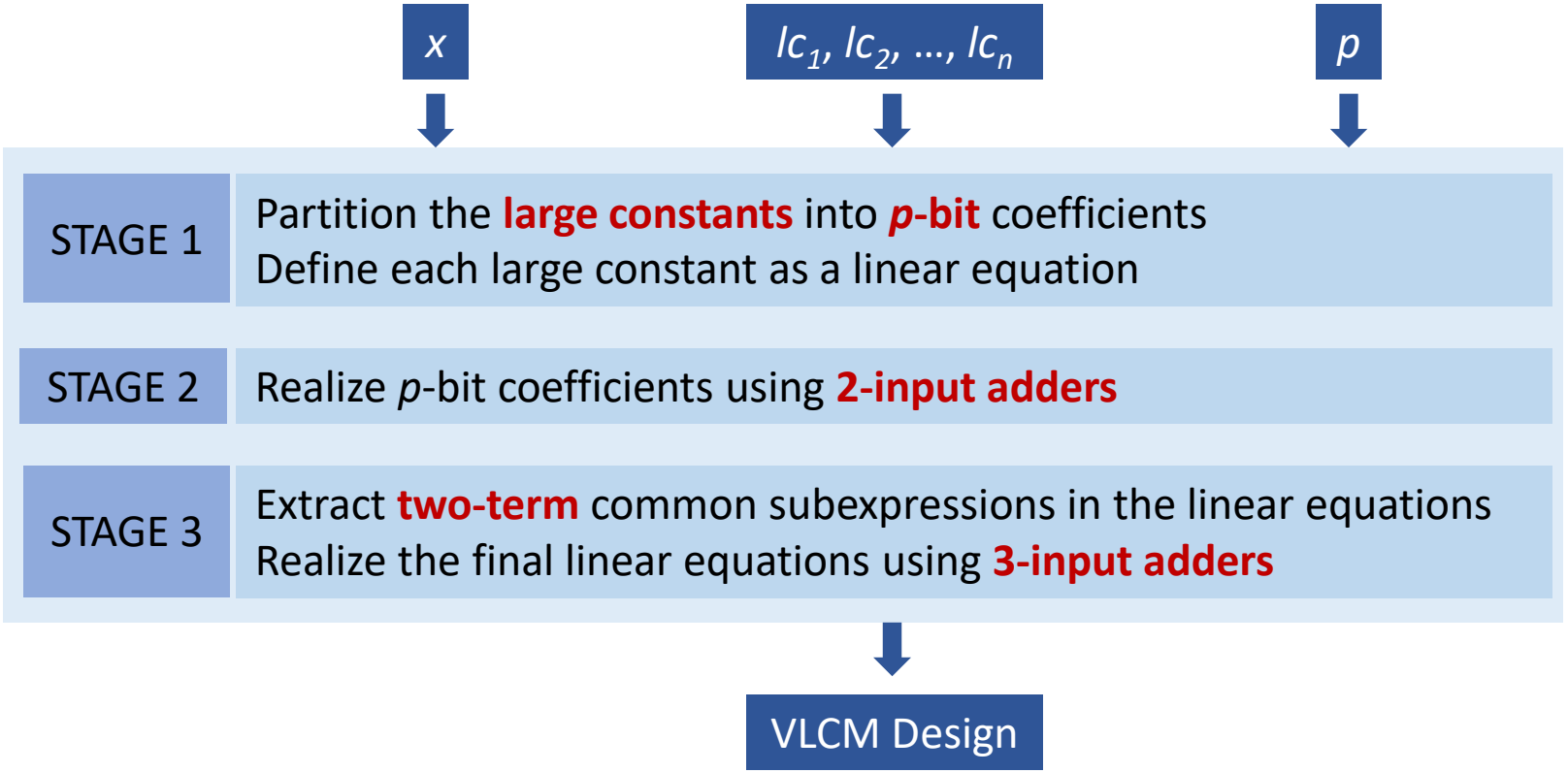
$S\&Cexp_1$ = $S\&Cexp_0$<<8 + S&C51

$S\&Clc_1$ = S&C78<<24 + $S\&Cexp_1$

$S\&Cexp_2$ = S&C102<<16 + $S\&Cexp_0$

$S\&Clc_2$ = S&C156<<24 + $S\&Cexp_2$

LEIGER – SA-Hybrid



LEIGER – SA-Hybrid – Partitioning

Large constants

lc_1 = 1,309,873,459 = 0x4E131533

lc_2 = 2,623,935,253 = 0x9C661315

Partitioning when p is 8

lc_1 = | 4E | 13 | 15 | 33 |

lc_2 = | 9C | 66 | 13 | 15 |

Coefficients

0x33 = 51 | 0x15 = 21 | 0x13 = 19 | 0x4E = 78 | 0x66 = 102 | 0x9C = 156

C = {51, 21, 19, 78, 102, 156}

Linear equations

lc_1 = S78<<24 + S19<<16 + S21<<8 + S51

lc_2 = S156<<24 + S102<<16 + S19<<8 + S21

LEIGER – SA-Hybrid – Realization of Coefficients

Coefficients

C = {51, 21, 19, 78, 102, 156}

Subexpressions

s3 = 1<<1 + 1

s21 = s3<<3 – s3

s51 = s3<<4 + s3

s19 = 1<<4 + s3

s39 = s19<<1 + 1

Realization of coefficients

S51 = s51

S21 = s21

S19 = s19

S78 = s39<<1

S102 = s51<<1

S156 = s39<<2

LEIGER – SA-Hybrid – Realization of Linear Equations

Linear equations

lc_1 = S78<<24 + S19<<16 + S21<<8 + S51

lc_2 = S156<<24 + S102<<16 + S19<<8 + S21

Common subexpressions

exp_0 = S19<<8 + S21

Final linear equations

lc_1 = S78<<24 + exp_0<<8 + S51

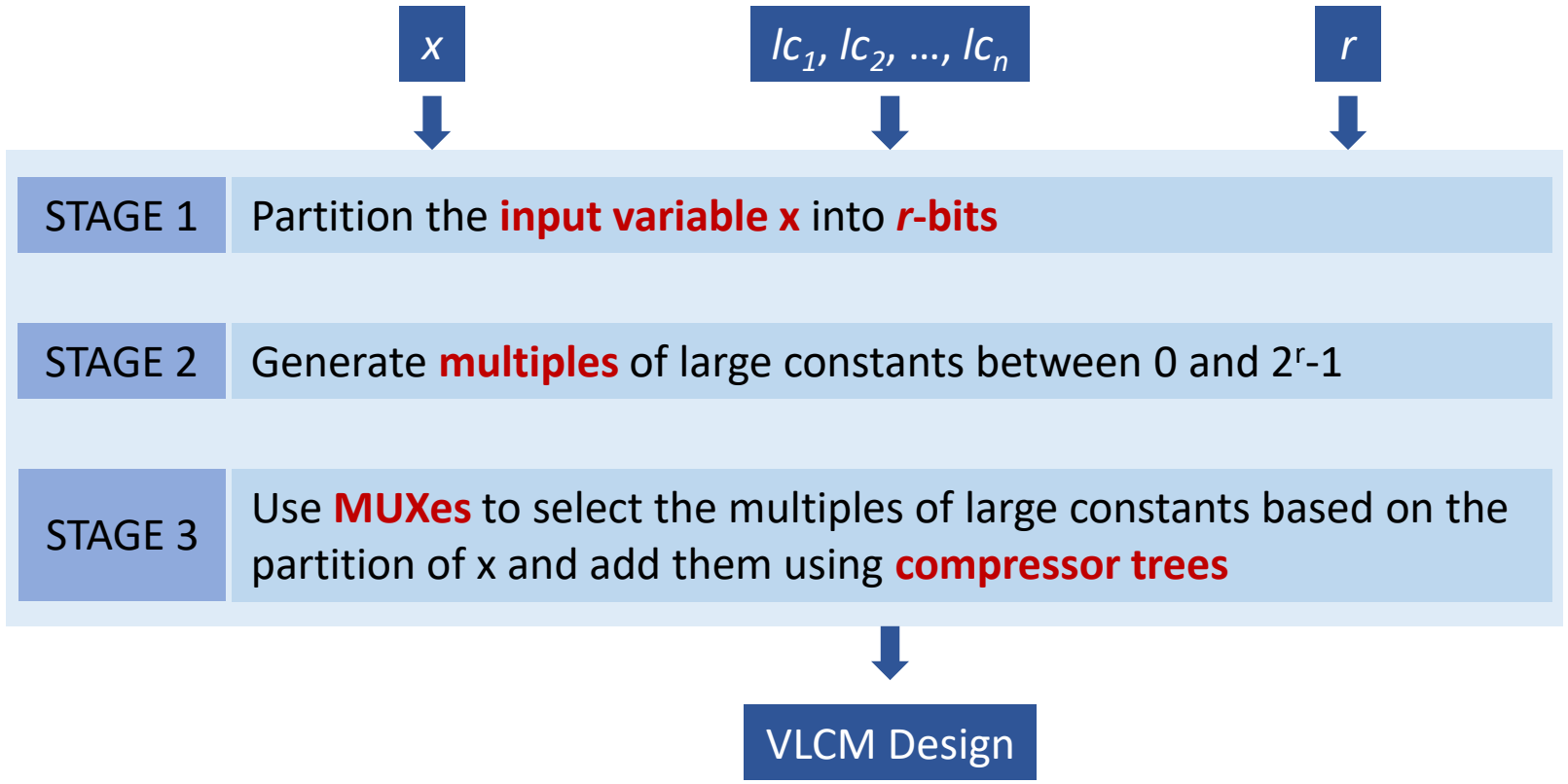
lc_2 = S156<<24 + S102<<16 + exp_0

Realization of final linear equations

S&Clc_1 = S78<<24 + exp_0<<8 + S51

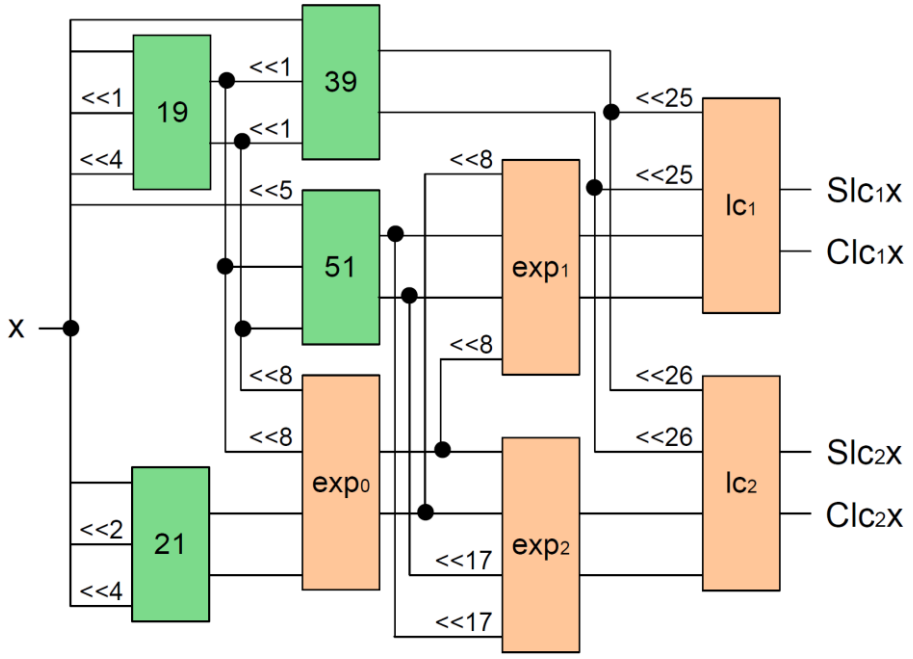
S&Clc_2 = S156<<24 + S102<<16 + exp_0

LEIGER – CT

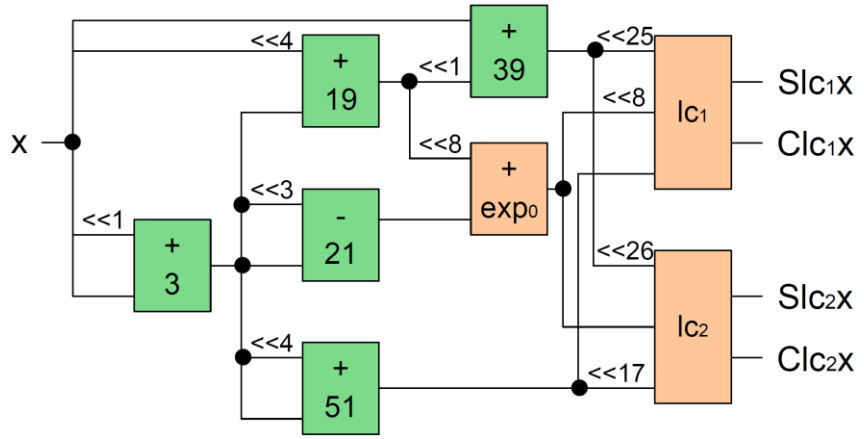

x
$lc_1, lc_2, \ldots, lc_n$
r
STAGE 1
Partition the input variable x into r-bits
STAGE 2
Generate multiples of large constants between 0 and 2^r-1
STAGE 3
Use MUXes to select the multiples of large constants based on the partition of x and add them using compressor trees
VLCM Design

LEIGER

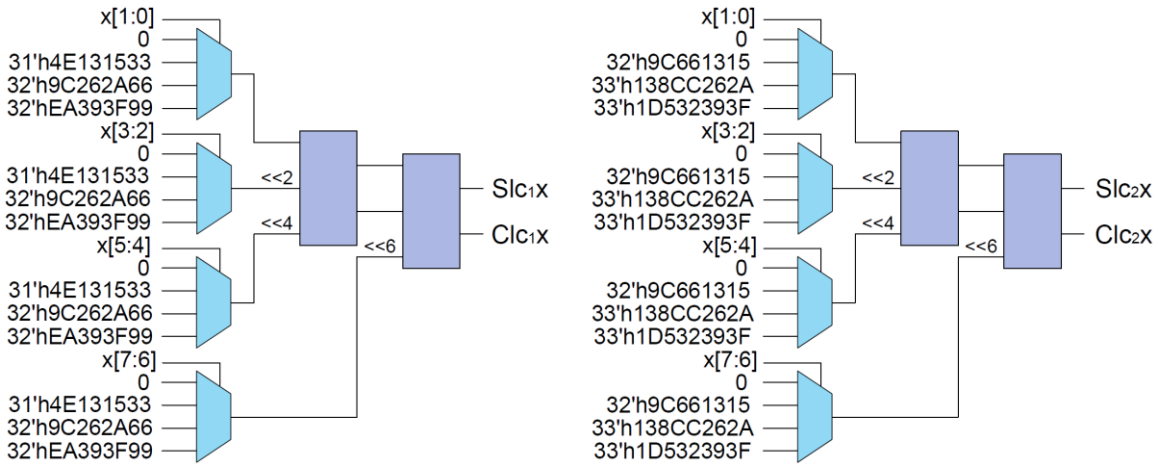
SA-CSA – p=8



SA-Hybrid – p=8



CT – r=2



Hardware Complexity

Architecture	#2-input Adders	#3-input Adders	#4x1 MUXes
SA-CSA	0	14	0
SA-Hybrid	6	2	0
CT	0	4	8

Experimental Results

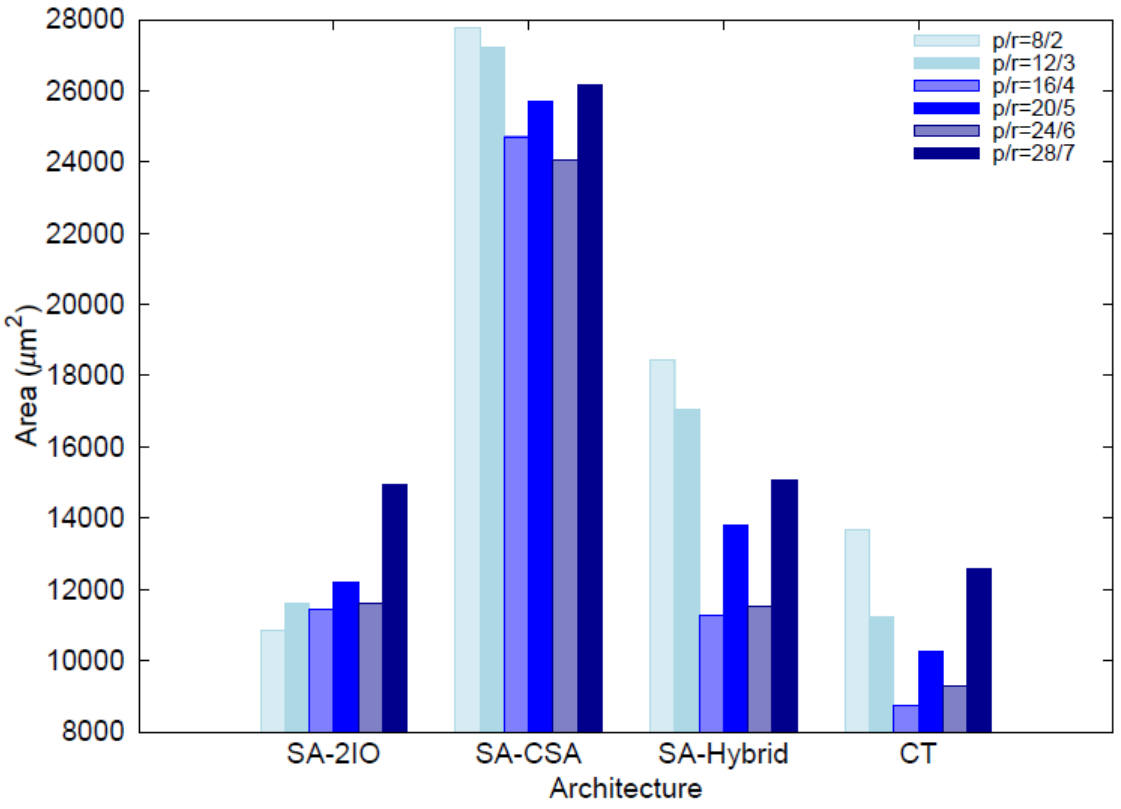
- LEIGER was applied to well-known cryptographic prime numbers [6]
  - bit-width of primes ranges between 204 and 384
  - corresponding VLCM operations and Montgomery multiplications were designed
- TÕLL [7] was also applied to realize these operations and Montgomery multiplications under the shift-adds architecture with 2-input operations (SA-2IO)
- Logic synthesis was performed by Cadence Genus using a commercial 65 nm cell library

[6] D. J. Bernstein and T. Lange. SafeCurves: Choosing Safe Curves for ECC. [Online]. Available: https://safecurves.cr.yp.to
[7] L. Aksoy, D. B. Roy, M. Imran, P. Karl and S. Pagliarini, “Multiplierless Design of Very Large Constant Multiplications in Cryptography, ” IEEE TCAS II, vol. 69, no. 11, pp. 4503-4507, 2022.

Experimental Results – VLCM Operation

Impact of ***p*** and ***r*** on area of the VLCM operation

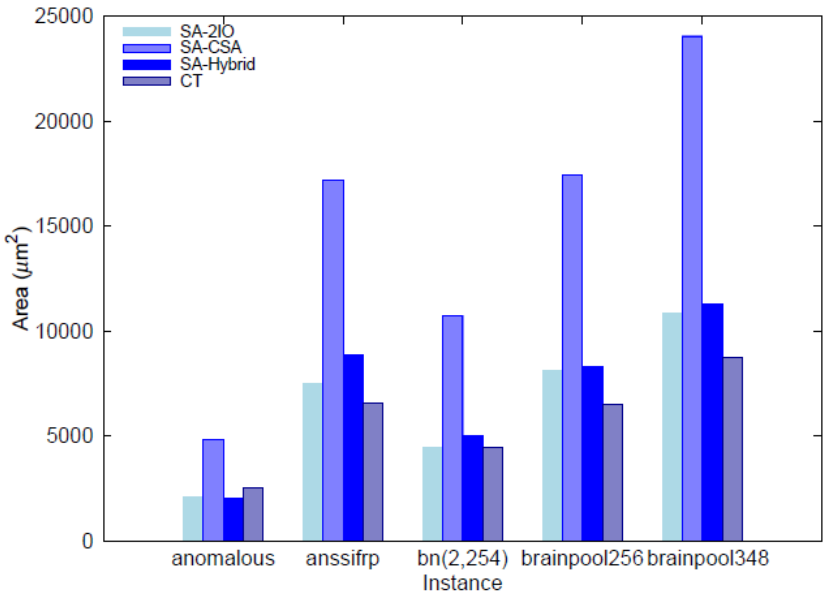
anomalous - iw=16



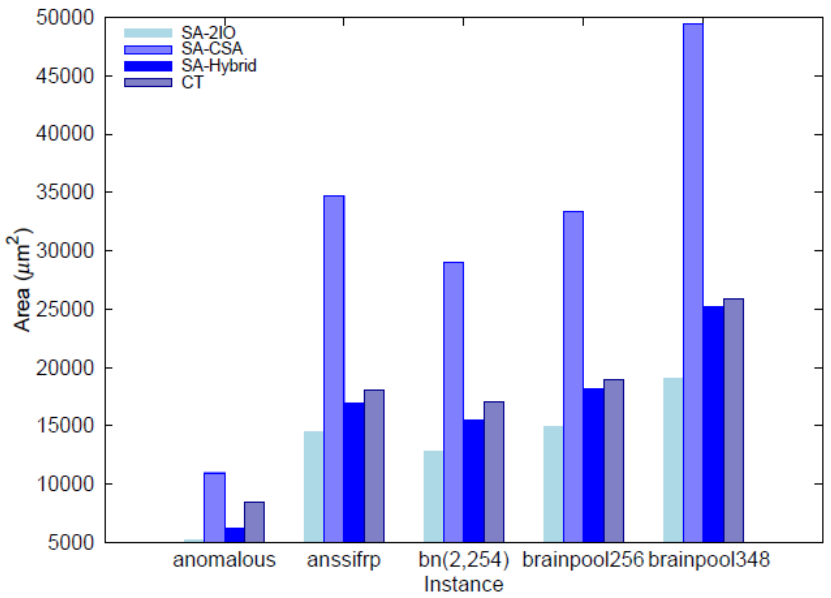
Experimental Results – VLCM Operation

Impact of design architectures on area of the VLCM operation

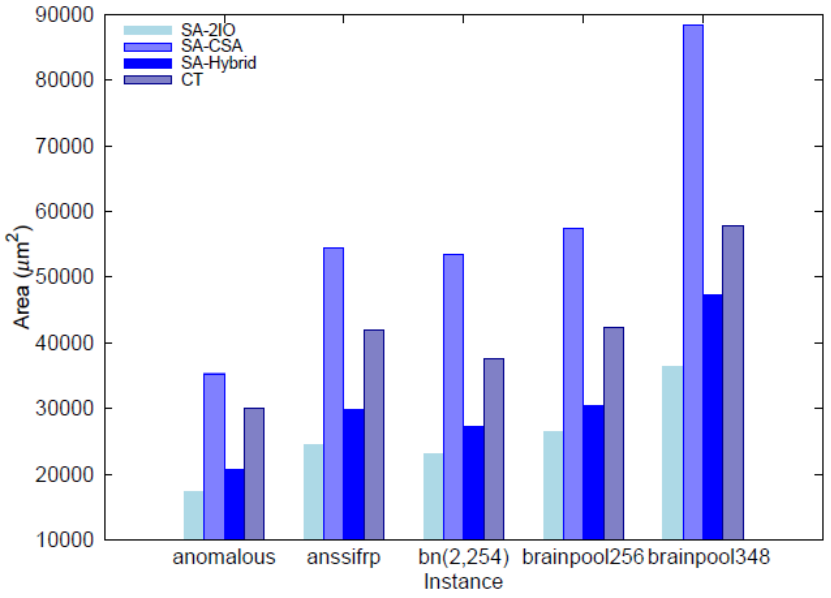
iw=16



iw=32



iw=64



Experimental Results – VLCM Operation

VLCM designs with minimum achievable delay (MAD) values

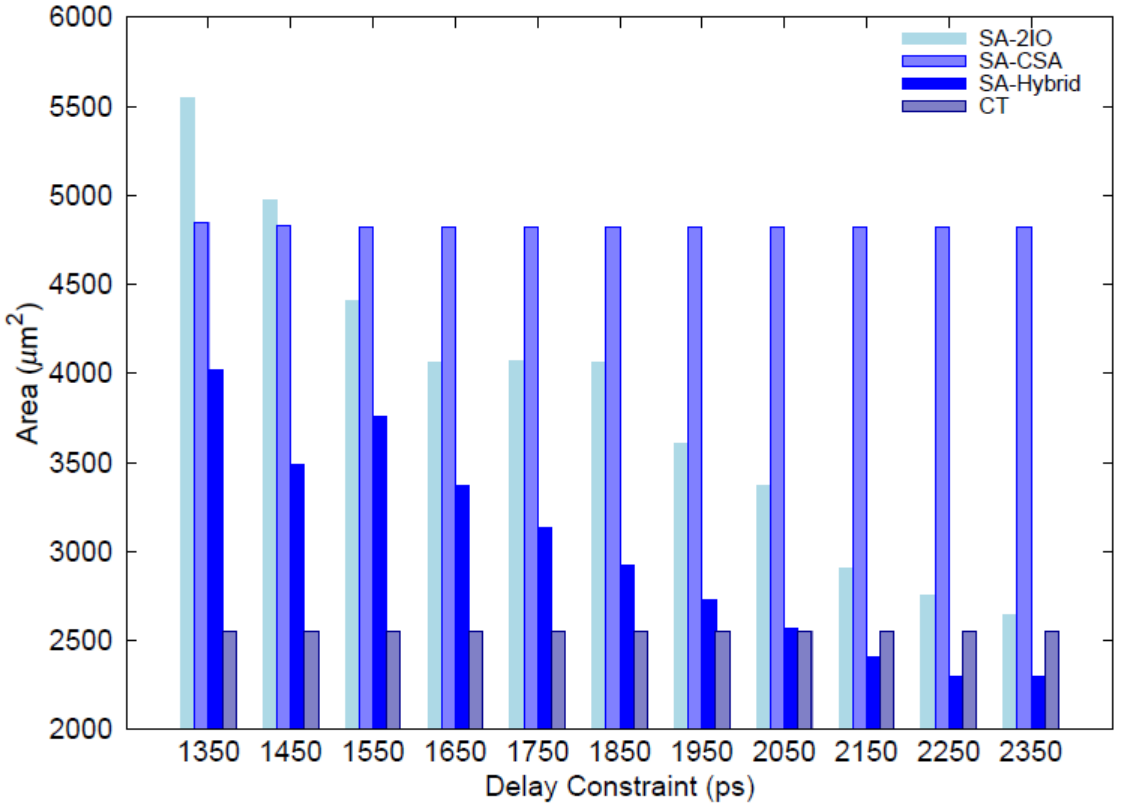
anomalous

Architecture	iw=16				iw=32				iw=64			
	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	Power (μW)	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	Power (μW)	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	Power (μW)
SA-2IO	5546	1341	7.4	1642	13847	1972	27.3	5601	39662	2719	107.8	15959
SA-CSA	8978	738	6.6	2735	17711	942	16.6	6702	95565	1014	96.9	34713
SA-Hybrid	4873	1154	5.6	1410	13704	1777	24.3	5745	46374	2140	99.2	18874
CT	5863	302	1.7	889	17684	505	8.9	4555	68342	703	48.0	18750

Experimental Results – VLCM Operation

Impact of delay constraint on area of VLCM operation

anomalous - iw=16



Experimental Results – Montgomery Multiplication

Montgomery multiplication designs with minimum achievable delay (MAD) values

anomalous

Architecture	iw=16							iw=32						
	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	#Clock cycles	Latency (ns)	Power (mW)	Energy (pJ)	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	#Clock cycles	Latency (ns)	Power (mW)	Energy (pJ)
SA-2IO	74220	1680	124.6	51	85.6	6.5	560	117214	2166	253.8	30	64.9	8.7	567
SA-CSA	94632	1087	102.8	51	55.4	8.8	489	197744	1197	236.6	30	35.9	17.1	616
SA-Hybrid	73685	1550	114.2	51	79.0	6.3	504	123590	1914	236.5	30	57.4	12.3	709
CT	106287	875	93.0	51	44.6	7.8	352	174710	1158	202.3	30	34.7	12.5	436

Conclusions

- We introduced LEIGER proposed for high-speed design of VLCM operations
  - is equipped with area optimization techniques
  - can describe VLCM designs under different multiplierless architectures
  - can also realize the Montgomery multiplication
- Experimental results showed that
  - it can generate alternative designs which may help a designer to choose the best fit for design requirements in a given application

Questions

THANKS for YOUR ATTENTION

Contact: Levent Aksoy
E-mail: levent.aksoy@taltech.ee
LEIGER GitHub Repository: https://github.com/leventaksoy/vlcm

Motivation

- There exist **no** methods proposed for high-speed design of the VLCM problem
  - earlier works [3-5] target DSP applications with a limited number of bit-widths
  - TÕLL [6] targets reduction of the number of operations and number of operations in series

[3] R. Hartley, “Subexpression Sharing in Filters Using Canonic Signed Digit Multipliers,” IEEE TCAS II, vol. 43, no. 10, pp. 677–688, 1996.
[4] A. Hosangadi, F. Fallah, and R. Kastner, “Reducing Hardware Complexity of Linear DSP Systems by Iteratively Eliminating Two-Term Common Subexpressions,” in ASP-DAC, 2005, pp. 523–528.
[5] L. Aksoy and E. O. Günes, “Area Optimization Algorithms in High-Speed Digital FIR Filter Synthesis,” in SBCCI, 2008, pp. 64–69.
[6] L. Aksoy, D. B. Roy, M. Imran, P. Karl and S. Pagliarini, “Multiplierless Design of Very Large Constant Multiplications in Cryptography, ” IEEE TCAS II, vol. 69, no. 11, pp. 4503-4507, 2022.

Experimental Results – Montgomery Multiplication

High-speed Montgomery multiplication designs

CT

Instance	iw=16							iw=32						
	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	#Clock cycles	Latency (ns)	Power (mW)	Energy (pJ)	Area (μm^2)	Delay (ps)	ADP (μm^2*ps*10^6)	#Clock cycles	Latency (ns)	Power (mW)	Energy (pJ)
anomalous	106287	875	93.0	51	44.6	7.8	352	174710	1158	202.3	30	34.7	12.5	436
anssifrp	133381	955	127.3	63	60.1	11.8	712	224574	1199	269.2	36	43.1	20.4	882
bn(2,254)	133850	944	126.3	60	56.6	10.9	621	211212	1139	240.5	36	41.0	18.2	750
brainpool256	140497	935	131.3	63	58.9	12.2	722	230367	1176	270.9	36	42.3	23.0	977
brainpool348	178034	914	162.7	87	79.5	19.2	1534	308665	1213	374.4	48	58.2	34.2	1992