Efficient Hardware Implementations for Elliptic Curve Cryptography over Curve448

Mojtaba Bisheh Niasar[1], Reza Azarderakhsh[1,2], Mehran Mozaffari Kermani[3]

[1]CEECS Department, Florida Atlantic University
[2]PQSecure Technologies, LLC, Boca Raton, FL , USA
[3]CSE Department at University of South Florida

13-16 December 2020

IndoCrypt 2020

Outline

Introduction

- Motivation
  - Recently standardized by NIST
  - Offering higher security level after Curve25519 [1]
  - Suitable for hybrid key exchange with SIKEp434
- Curve448
  - Designed by Hamburg in 2015 [2]
  - Belongs to Safe-Curve [3]
  - Software-based design
  - Montgomery curve and also an untwisted Edwards curve (Edwards448) [4]

[1] Bisheh-Niasar, M., El Khatib, R., Azarderakhsh, R., Mozaffari-Kermani, M.: Fast, small, and area-time efficient architectures for key-exchange on curve25519. In: 2020 IEEE 27th Symposium on Computer Arithmetic (ARITH). 72–79 (2020)
[2] Hamburg, M.: Ed448-goldilocks, a new elliptic curve. IACR Cryptology ePrint Archive 2015, 625 (2015)
[3] Bernstein, D.J., Lange., T.: Safecurves: choosing safe curves for elliptic-curve cryptography. url: https://safecurves.cr.yp.to/. (2016)
[4] Langley, A., Hamburg, M., Turner, S.: Elliptic curves for security, Internet Engineering Task Force (IETF-RFC7748) (2016)

Background

- Work by [1] and [2] based on schoolbook multiplication
- LUT-based scheme [3]

- Gaps:
  - Few hardware implementations
  - Exploration of the different trade-offs between resource utilization and performance
  - Employing the Karatsuba-friendly property of Curve448

[1] Sasdrich, P., Güneysu, T.: Cryptography for next generation TLS: implementing the RFC 7748 elliptic curve448 cryptosystem in hardware. In: Proceedings of the 54th Annual Design Automation Conference, DAC 2017, Austin, TX, USA, June 18-22, 2017. 16:1–16:6 (2017)
[2] Sasdrich, P., Güneysu, T.: Exploring RFC 7748 for hardware implementation: Curve25519 and curve448 with side-channel protection. J. Hardware and Systems Security 2(4), 297–313 (2018)
[3] Shah, Y.A., Javeed, K., Shehzad, M.I., Azmat, S.: Lut-based high-speed point multiplier for goldilocks-curve448. IET Computers Digital Techniques 14(4), 149– 157 (2020)

Our Contributions

- Investigate three design strategies:
  - **Lightweight architecture** targets area-constrained applications.
  - **High-performance architecture** targets time-constrained applications.
  - **Area-Time Efficient architecture** targets area-time trade-off applications.
- Employing various optimization techniques to increase efficiency
  - Refined Karatsuba multiplication
  - Redundant number presentation
  - Interleaved multiplication
- Performing a precise schedule corresponding to each architecture
- Variable-base-point multiplications
- Extended by side-channel countermeasures

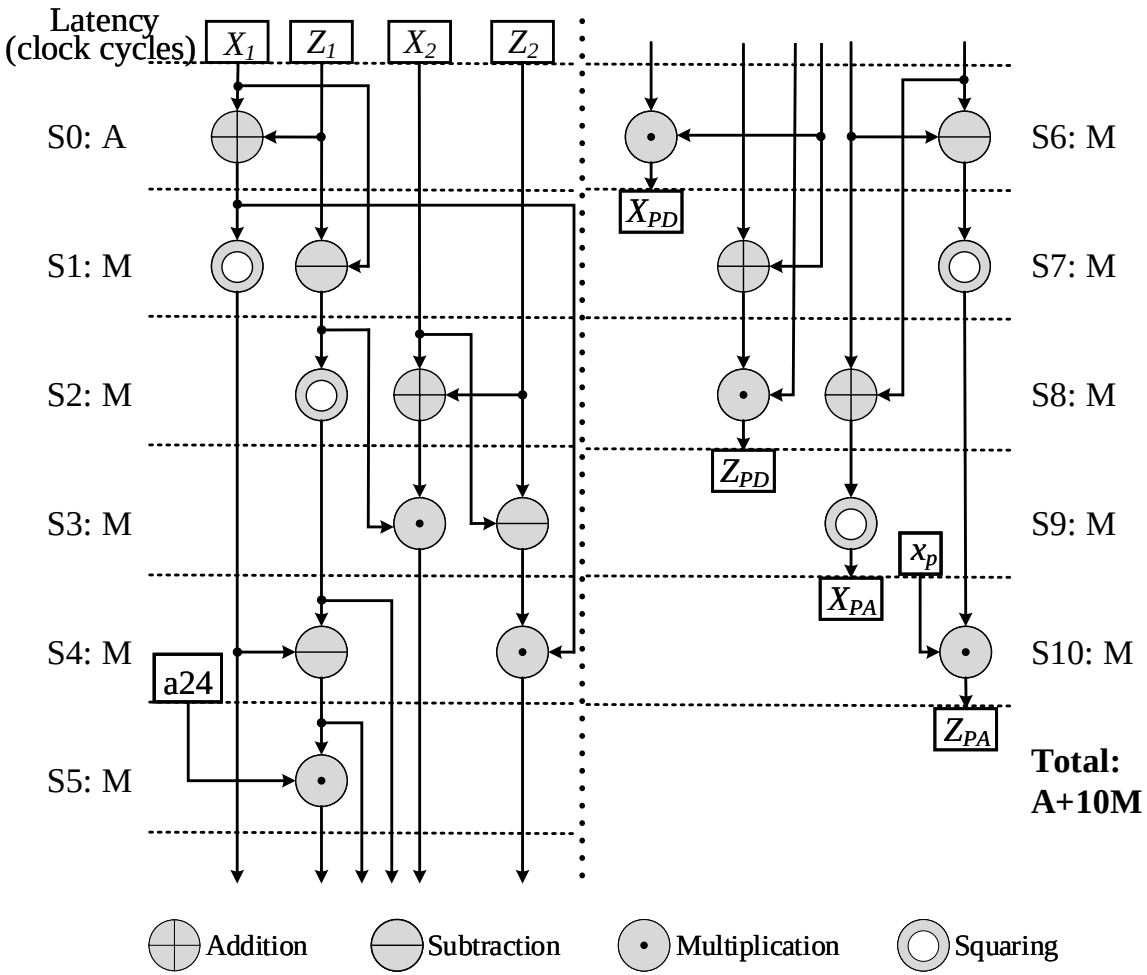
X448 Computation

- ECPM: elliptic curve point multiplication $Q = k \cdot P$ over Curve448
- Finite field arithmetic with $p = 2^{448} - 2^{224} - 1$
- All arithmetic is performed on Montgomery ladder for efficiency
  - point addition (PA)
  - point doubling (PD)
- Employing projective coordinate

$$
\begin{aligned}
X_{PD} &= (X_1 - Z_1)^2 \cdot (X_1 + Z_1)^2 \\
Z_{PD} &= 4X_1Z_1 \cdot (X_1^2 + 39081X_1Z_1 + Z_1^2) \\
X_{PA} &= 4(X_1X_2 - Z_1Z_2)^2 \\
Z_{PA} &= 4x_p(X_1Z_2 - Z_1X_2)^2
\end{aligned}
$$

Montgomery ladder

Data dependency diagram for one step Montgomery ladder execution over Curve448



X448 Side-Channel Protection

- Constant-time implementation against timing attack
- Secret-independent implementation against simple power analysis (SPA)
- Countermeasures against differential power analysis (DPA) attacks [1]
  - Point Randomization

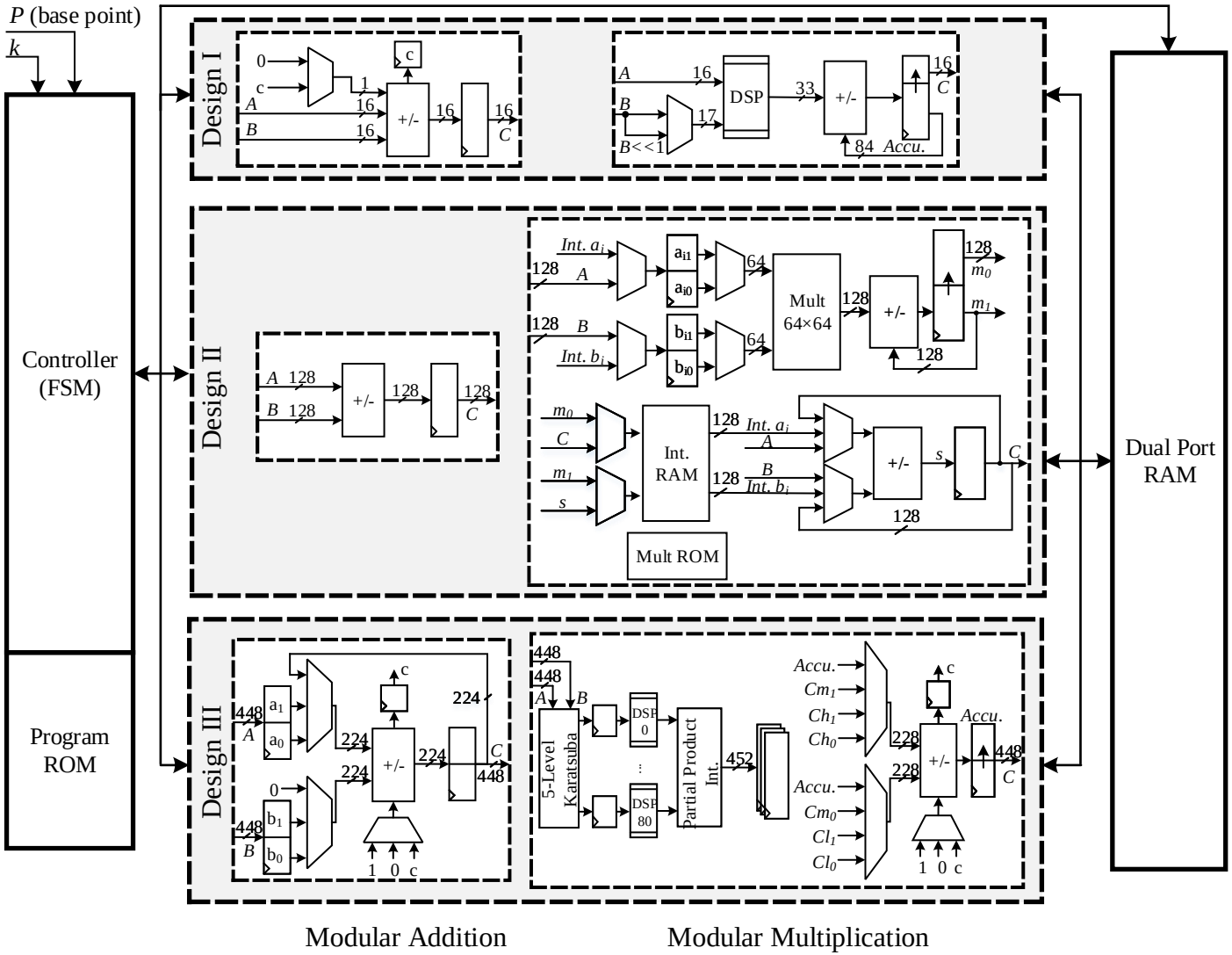
$$P = (X, Z) \rightarrow P_r = (\lambda \cdot X, \lambda \cdot Z)$$

  - Scalar Blinding

$$Q = k \cdot P \rightarrow Q = k_r \cdot P$$

[1] Coron, J.: Resistance against differential power analysis for elliptic curve cryptosystems. In Koç, Ç.K., Paar, C., eds.: Cryptographic Hardware and Embedded Systems, CHES'99. 292–302 (1999, Worcester, MA, USA)

Hardware Architecture for Different Performance Levels over Curve448



Lightweight Architecture

- A 448-bit data is stored by splitting to 28 words in 16-bit chunks due to DSP block input size
- 16-bit datapath
- Program ROM consists of 2183 instruction lines
- Modular Addition/Subtraction:
  - Sequential addition by propagating carry/borrow to the next digit
- Modular Multiplication:
  - Based on the product scanning approach
  - Utilizing only one DSP
  - Interleaved reduction

Area-Time Efficient Architecture

- A 448-bit data is stored by splitting to 4 words in 112-bit chunks
- 128-bit datapath
- Program ROM consists of 480 instruction lines
- Fast Karatsuba multiplication [1] with golden ratio $\phi = 2^{224}$

$$A \times B = (A_0 B_0 + A_1 B_1) + 2^{224}(A_{10} B_{10} - A_0 B_0)$$

- Refined Karatsuba identity [2]

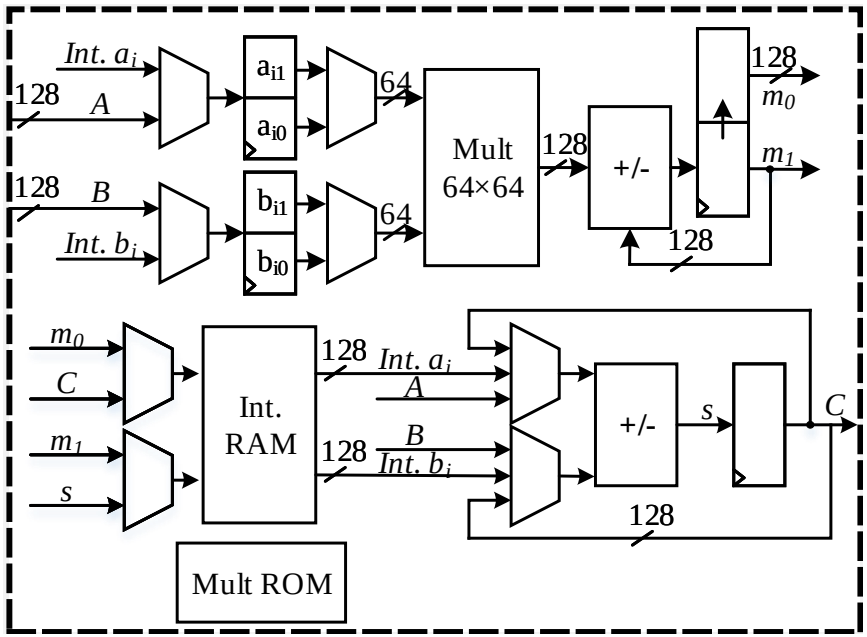
$$(a_0 + a_1 t^n) \cdot (b_0 + b_1 t^n) = (1 - t^n) \cdot (a_0 b_0 - t^n a_1 b_1) + t^n (a_0 + a_1) \cdot (b_0 + b_1)$$

[1] Hamburg, M.: Ed448-goldilocks, a new elliptic curve. IACR Cryptology ePrint Archive 2015, 625 (2015)
[2] Bernstein, D.J.: Batch binary edwards. In: Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings. 317–336 (2009)

Area-Time Efficient Architecture

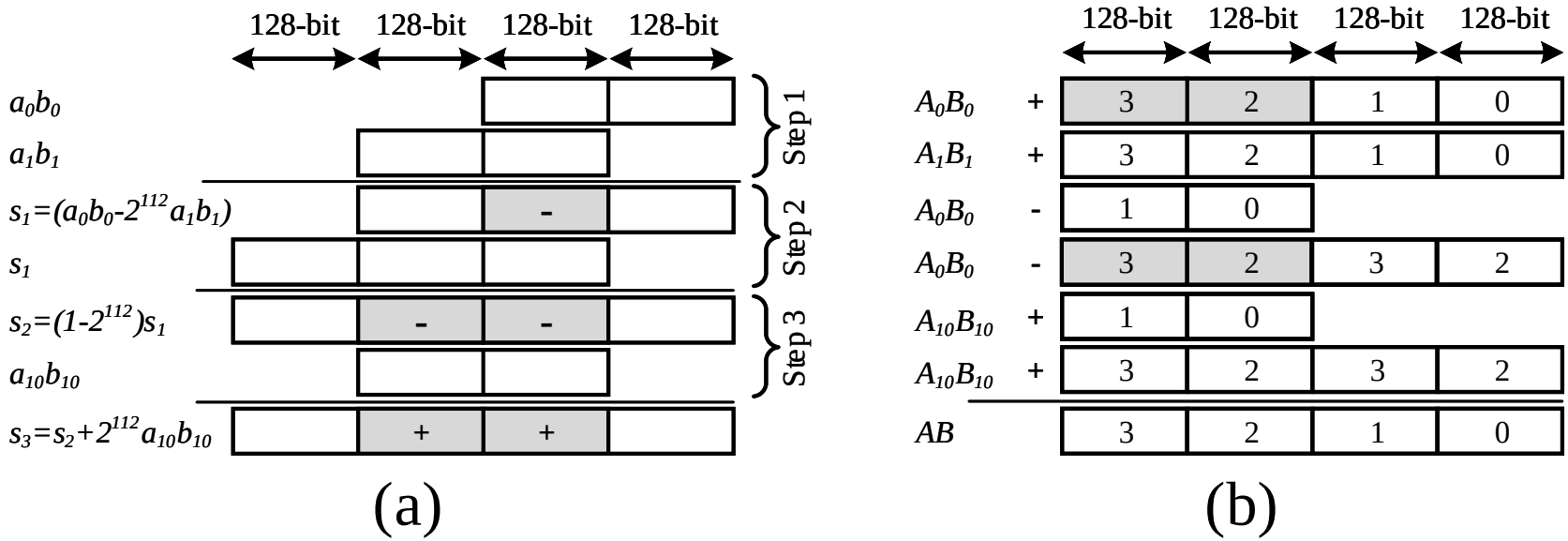
Modular Multiplication:

- 64×64-bit multiplication using 16 DSPs
- High throughput:
  - a 64×64-bit per cycle
  - a 128×128-bit per 4 cycles
  - $A_0 B_0$ in 12 cycles



$$A_0 B_0 = (1 - 2^{112}) \cdot (a_0 b_0 - 2^{112} a_1 b_1) + 2^{112}(a_0 + a_1) \cdot (b_0 + b_1)$$
$$= (1 - 2^{112}) \cdot (a_0 b_0 - 2^{112} a_1 b_1) + 2^{112}(a_{10} b_{10})$$

Area-Time Efficient Architecture



(a) Optimized middle-level recombination: only five highlighted operations are performed.

(b) Optimized top-level recombination: applying interleaved reduction cancels the highlighted digits.

High-Performance Architecture

- Full 448-bit datapath
- Program ROM consists of 1554 instruction lines
- 5-level consecutive Karatsuba multiplication
  - reducing considerably the required cycles at the cost of expanding addition tree
- the first level is designed pipeline due to the number of available DSP blocks
  - $Ch = A_1 B_1$, $Cl = A_0 B_0$, and $Cm = A_{10} B_{10}$
- the rest of the multiplications are performed parallel using 81 DSP blocks
  - 5 clock cycles for all three required values

Performance Results

FPGA implementation results for different performance level architectures in terms of clock cycles and latency requirements

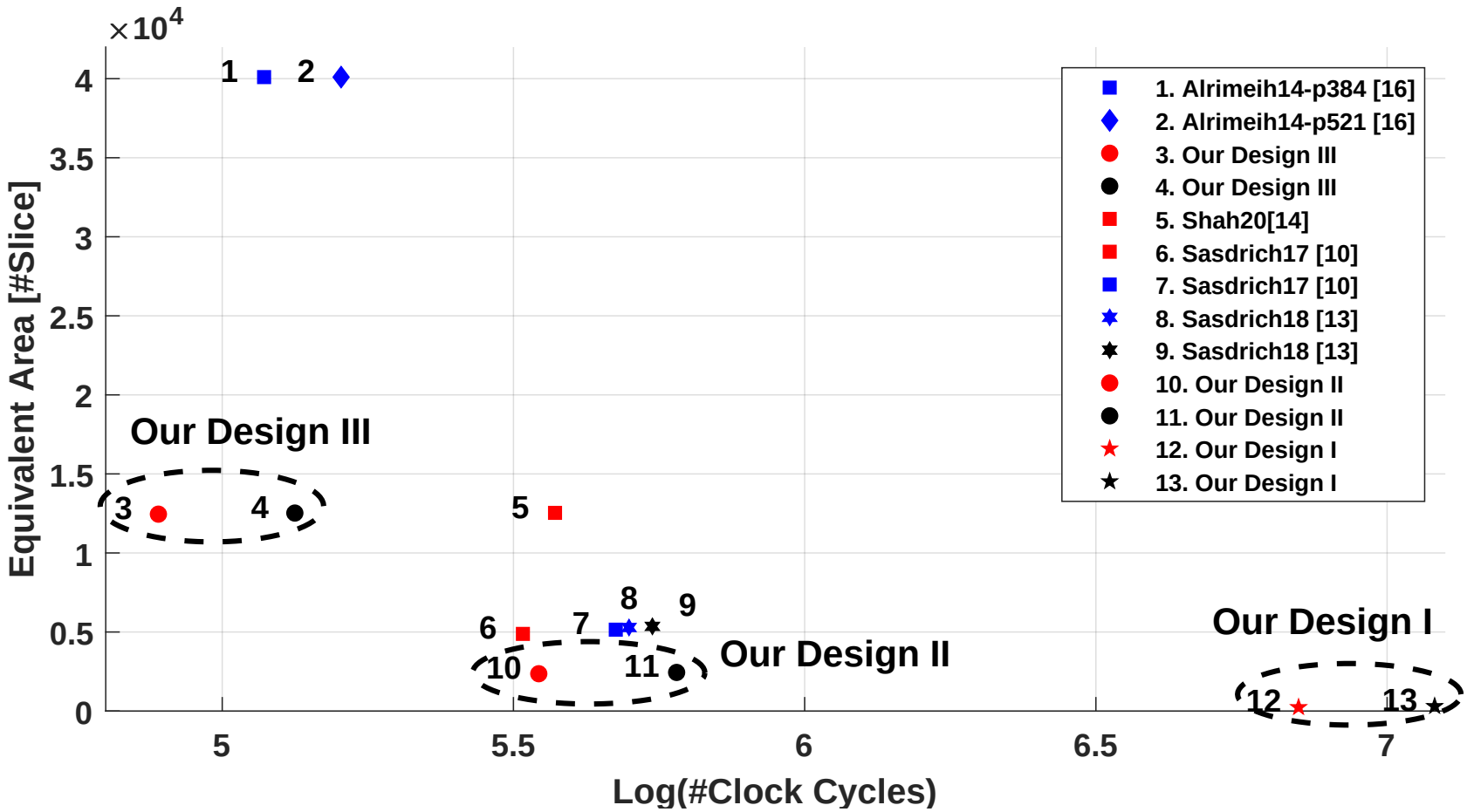
Proposed Architecture		Design I	Design II	Design III
Clock Freq. [MHz]		250	127	95
$\mathbb{F}_p$ Operation	Addition	112	8	7
	Subtraction	112	8	7
	Multiplication	1,233	69	15
	Inversion	620,047	32,272	6,917
Unprotected	Mont. step	14,346	708	158
	Point Multiplication	7,047,055	349,546	77,702
	Latency [ms]	28.19	2.75	0.82

Comparison

FPGA implementation results for different ECPM cores above 128-bit security

Work	SCA	Area					Time			
		LUTs	FFs	Slices	DSPs	BRAMs	Latency [CCs]	Freq [MHz]	Total time [ms]	OP/s
NIST P-384 (192-bit security)										
Alrimeih et al. (2014)	(+)	32900	~	11,200	289	128	~	100	1.18	847
Ananyi et al. (2009)	(-)	31946	~	20,793	32	1	~	60	17.50	57
NIST P-521 (260-bit security)										
Alrimeih et al. (2014)	(+)	32900	~	11,200	289	128	~	100	1.60	625
Ananyi et al. (2009)	(-)	31946	~	20,793	32	1	~	60	39.90	25
Curve448 (224-bit security)										
Sasdrich et al. (2017)	(-)	2,555	7,049	1,580	33	14	328,286	357	0.92	1,087
	(+)	3,583	7,423	1,648	35	14	473,926	335	1.41	708
Sasdrich et al. (2018)	(+)	4,624	8,209	1,985	33	14	499,344	341	1.46	685
	(++)	~	~	2,056	33	14	547,728	341	1.61	622
Shah et al. (2020)	(-)	50,143	~	~	0	~	372,742	325	1.15	869
Design I	**(-)**	**321**	**174**	**137**	**1**	**2**	**7,047,055**	**250**	**28.19**	**35**
	(++)	**509**	**438**	**203**	**1**	**2**	**12,068,239**	**250**	**48.27**	**21**
Design II	**(-)**	**2,233**	**1,152**	**760**	**16**	**9**	**349,546**	**127**	**2.75**	**363**
	(++)	**2,587**	**1,629**	**842**	**16**	**9**	**602,801**	**127**	**4.75**	**211**
Design III	**(-)**	**13,132**	**4,035**	**4,354**	**81**	**0**	**77,702**	**95**	**0.82**	**1,219**
	(++)	**13,415**	**4,610**	**4,424**	**81**	**0**	**133,254**	**95**	**1.40**	**713**

Comparison



Efficiency comparison between FPGA-based ECPM architectures in terms of $A \cdot T$ (Area×Time) in a fixed low-frequency.

red: unprotected, blue: protected, black: high-protected

Comparison

- **Design I**
  - Saving 96% of resources with the competitive performance
- **Design II**
  - Improving 48% and 50% efficiency compared to [1] and [2]
  - Occupying 52% fewer resources
- **Design III**
  - Improving efficiency by 40% and 43% compared to [1] and [2]
  - Increasing 12% throughput

[1] Sasdrich, P., Güneysu, T.: Cryptography for next generation TLS: implementing the RFC 7748 elliptic curve448 cryptosystem in hardware. In: Proceedings of the 54th Annual Design Automation Conference, DAC 2017, Austin, TX, USA, June 18-22, 2017. 16:1–16:6 (2017)
[2] Sasdrich, P., Güneysu, T.: Exploring RFC 7748 for hardware implementation: Curve25519 and curve448 with side-channel protection. J. Hardware and Systems Security 2(4), 297–313 (2018)

Conclusion

- Three hardware implementations for security level 224-bit
  - area-constrained applications
  - time-constrained applications
  - area and time trade-off applications
- Implemented on a mid-range Xilinx FPGA XC7Z7020
- Extended by side-channel countermeasures
- Computing 1219, 363, and 35 ECDH operations per second.

Thanks for your attention.