FPGA Design of Self-certified Signature Verification on Koblitz Curves

Kimmo Järvinen Juha Forsten Jorma Skyttä

Helsinki University of Technology
Signal Processing Laboratory
Otakaari 5A, FIN-02150, Finland
{Kimmo.Jarvinen,Juha.Forsten,Jorma.Skytta}@tkk.fi

September 12, 2007

Outline

Introduction

- Packet Level Authentication (PLA)[1]
  - Enormous speed requirements!
  - Elliptic curve cryptography because short signatures and fast performance are needed
  - Koblitz curve, NIST K-163, used to maximize speed
  - Self-certified ID based signatures because they are short and computationally less complex

[1] See http://www.tcs.hut.fi/Software/PLA/new/index.shtml

Introduction

- Packet Level Authentication (PLA)[1]
  - Enormous speed requirements!
  - Elliptic curve cryptography because short signatures and fast performance are needed
  - Koblitz curve, NIST K-163, used to maximize speed
  - Self-certified ID based signatures because they are short and computationally less complex
- Development in FPGA technology
  - Growth in resources enables massive parallelization
  - Point multiplication times $< 100\ \mu$s have been reported
- We focus on maximizing operations per second instead of minimizing computation time of a single operation

[1] See http://www.tcs.hut.fi/Software/PLA/new/index.shtml

Koblitz curves

- Koblitz curves have the form

$$E_K : y^2 + xy = x^3 + ax^2 + 1$$

- If $P = (x, y)$ is a point on E_K, then its Frobenius endomorphism, $\phi(P) = (x^2, y^2)$, is also on E_K.
- Very efficient point multiplication
  - Integer presented in τ-adic non-adjacent form (NAF)[2]
  - Point doublings replaced by Frobenius maps
  - Only $m/3$ point additions

[2] Solinas, Des. Codes Cryptogr. 19(2-3), 2000

Self-certified identity based signatures

- Used in the current version of the PLA
- Signature verification is the most critical operation

Self-certified identity based signatures

- Used in the current version of the PLA
- Signature verification is the most critical operation

Signature verification

A signature is verified by computing:

$$W_A = \text{DECOMPRESS}(r_A - \text{HASH}(ID_A), b_A) - r_A W_D, \text{ and}$$

$$\text{HASH}(\mathcal{M}) = c - [dG + cW_A]_x \pmod{r}$$

Self-certified identity based signatures

- Used in the current version of the PLA
- Signature verification is the most critical operation

Signature verification

A signature is verified by computing:

$$W_A = \text{DECOMPRESS}(r_A - \text{HASH}(ID_A), b_A) - r_A W_D, \text{ and}$$

$$\text{HASH}(\mathcal{M}) = c - [dG + cW_A]_x \pmod{r}$$

which simplify into the 3-term point multiplication:

$$dG + c(uG) - cr_A W_D$$

Self-certified identity based signatures

- Used in the current version of the PLA
- Signature verification is the most critical operation

Signature verification

A signature is verified by computing:

$$W_A = \text{DECOMPRESS}(r_A - \text{HASH}(ID_A), b_A) - r_A W_D, \text{ and}$$

$$\text{HASH}(\mathcal{M}) = c - [dG + cW_A]_x \pmod{r}$$

which simplify into the 3-term point multiplication:

$$dG + c(uG) - cr_A W_D = k_1 P_1 + k_2 P_2 + k_3 P_3$$

Point multiplication

$$Q = k_1 P_1 + k_2 P_2 + k_3 P_3$$

- Shamir's trick $\Rightarrow$ 3-term double-and-add algorithm
- 3-term τ-adic joint sparse form[3]

Simplified algorithm

1. Precompute all possible combinations $R_{k_1,k_2,k_3} = k_{1,j}P_1 + k_{2,j}P_2 + k_{3,j}P_3$
2. Perform $\phi(P)$ for all bits
3. If $k_{1,j}, k_{2,j}, k_{3,j} \neq 000$, add R_{k_1,k_2,k_3} to Q using mixed coordinate point addition[a]

[a] Al-Daoud et al. IEEE Tran. Comp. 51(8), 2002

[3] Brumley, ICICS 2006, LNCS 4307

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j$, and
  2. $R_{k+1} = R_i - R_j$

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}01$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j$, and
  2. $R_{k+1} = R_i - R_j$

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j$, and
  2. $R_{k+1} = R_i - R_j$

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j$, and
  2. $R_{k+1} = R_i - R_j$

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j,$ and
  2. $R_{k+1} = R_i - R_j$

Precomputed points

$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point	$k_3k_2k_1$	Point
000	$R_0 = \mathcal{O}$	$10\bar{1}$	$R_7 = R_3 - R_1$	n/a		$\bar{1}01$	$-R_7$
001	$R_1 = P_1$	110	$R_8 = R_3 + R_2$	$00\bar{1}$	$-R_1$	$\bar{1}\bar{1}0$	$-R_8$
010	$R_2 = P_2$	$1\bar{1}0$	$R_9 = R_3 - R_2$	$0\bar{1}0$	$-R_2$	$\bar{1}10$	$-R_9$
100	$R_3 = P_3$	111	$R_{10} = R_8 + R_1$	$\bar{1}00$	$-R_3$	$\bar{1}\bar{1}\bar{1}$	$-R_{10}$
011	$R_4 = R_2 + R_1$	$11\bar{1}$	$R_{11} = R_8 - R_1$	$0\bar{1}\bar{1}$	$-R_4$	$\bar{1}\bar{1}1$	$-R_{11}$
$01\bar{1}$	$R_5 = R_2 - R_1$	$1\bar{1}1$	$R_{12} = R_9 + R_1$	$0\bar{1}1$	$-R_5$	$\bar{1}1\bar{1}$	$-R_{12}$
101	$R_6 = R_3 + R_1$	$1\bar{1}\bar{1}$	$R_{13} = R_9 - R_1$	$\bar{1}0\bar{1}$	$-R_6$	$\bar{1}11$	$-R_{13}$

- Precomputations require 10 point additions(/subtractions)
- Pairs (R_k, R_{k+1}) are computed so that
  1. $R_k = R_i + R_j$, and
  2. $R_{k+1} = R_i - R_j$
- Unified point addition and subtraction: $(R_k, R_{k+1}) \leftarrow R_i \pm R_j$

Unified point addition and subtraction

Point addition

$(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2}$$
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

Point subtraction

$(x_4, y_4) = (x_1, y_1) - (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2 + x_2}{x_1 + x_2}$$
$$x_4 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_4 = \lambda(x_1 + x_4) + x_4 + y_1$$

Unified point addition and subtraction

Point addition

$(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2}$$
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

Point subtraction

$(x_4, y_4) = (x_1, y_1) - (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2 + x_2}{x_1 + x_2}$$
$$x_4 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_4 = \lambda(x_1 + x_4) + x_4 + y_1$$

- Inversion is the same[4]

[4] Mentioned by Okeya et al. in ACISP 2005, LNCS 3574

Unified point addition and subtraction

Point addition

$(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2}$$
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

Point subtraction

$(x_4, y_4) = (x_1, y_1) - (x_2, y_2)$

$$\lambda = \frac{y_1 + y_2 + x_2}{x_1 + x_2}$$
$$x_4 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_4 = \lambda(x_1 + x_4) + x_4 + y_1$$

- Inversion is the same[4]
- Some additions can be saved by rearranging operations

[4] Mentioned by Okeya et al. in ACISP 2005, LNCS 3574

Unified point addition and subtraction

Point addition

$$(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$$

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2}$$
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

Point subtraction

$$(x_4, y_4) = (x_1, y_1) - (x_2, y_2)$$

$$\lambda = \frac{y_1 + y_2 + x_2}{x_1 + x_2}$$
$$x_4 = \lambda^2 + \lambda + x_1 + x_2 + a$$
$$y_4 = \lambda(x_1 + x_4) + x_4 + y_1$$

- Inversion is the same[4]
- Some additions can be saved by rearranging operations
- Total cost reduces from $2I + 4M + 2S + 17A$ to $I + 4M + 2S + 14A$

[4] Mentioned by Okeya et al. in ACISP 2005, LNCS 3574

Montgomery's trick

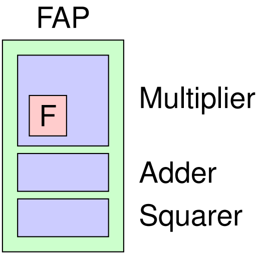
Method	Cost	$I = 9M$
Naïve	$10\,(I + 2M + S + 8A) + 5A$	110M
Unified	$5\,(I + 4M + 2S + 14A)$	65M

Montgomery's trick

Method	Cost	I = 9M
Naïve	$10\,(I + 2M + S + 8A) + 5A$	110M
Unified	$5\,(I + 4M + 2S + 14A)$	65M
Unified + Montgomery	$I + 17M + 2S + 9A + 5\,(4M + 2S + 14A)$	46M

- Trades inversions to multiplications
- $1/x_1$ and $1/x_2$ computed so that $1/x_1 = x_2/(x_1 x_2)$ and $1/x_2 = x_1/(x_1 x_2)$
- n inversions computed with $3(n-1)$ multiplications and 1 inversion

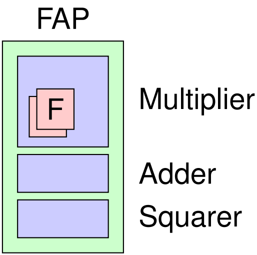
Architecture



Massey-Omura multiplier

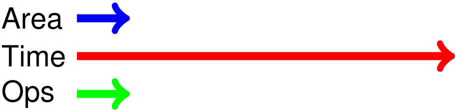
- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles

Architecture

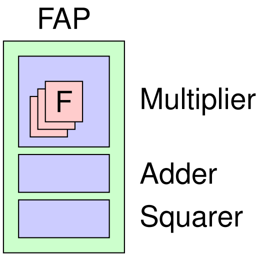


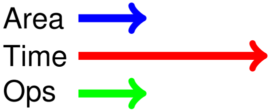
Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$



Architecture

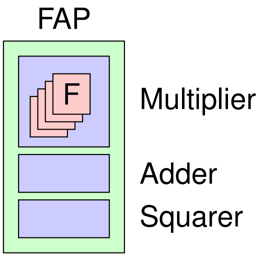


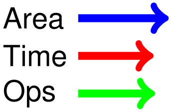


Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$

Architecture

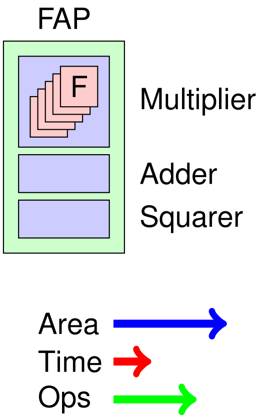




Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$

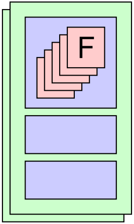
Architecture



Massey-Omura multiplier

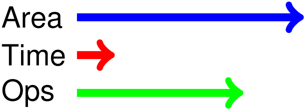
- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$

Architecture

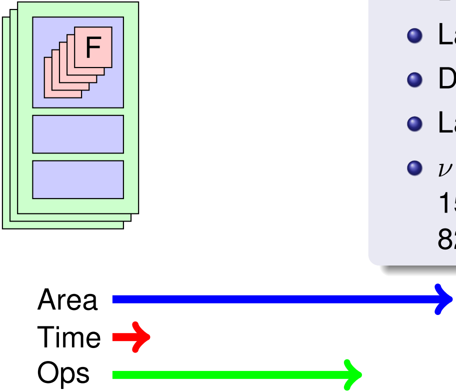


Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$



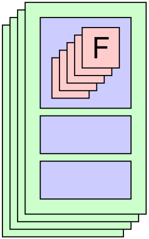
Architecture



Massey-Omura multiplier

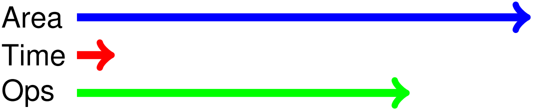
- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$

Architecture

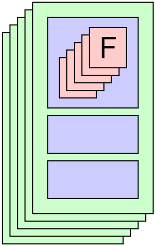


Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$

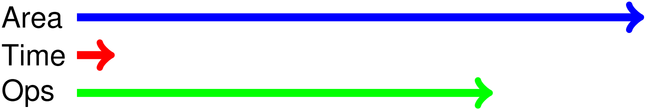


Architecture

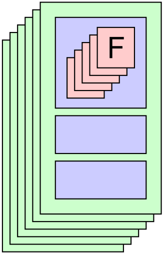


Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$

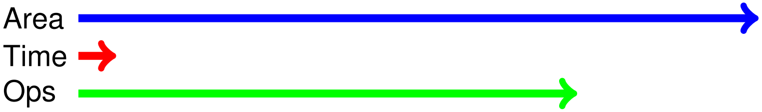


Architecture

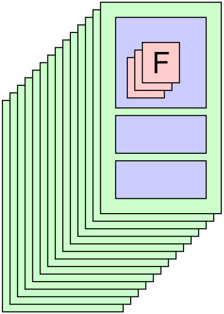


Massey-Omura multiplier

- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F}$: $\{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$

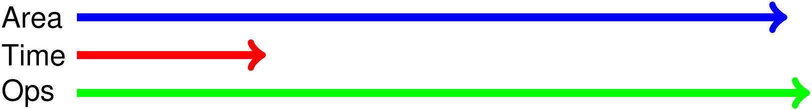


Architecture

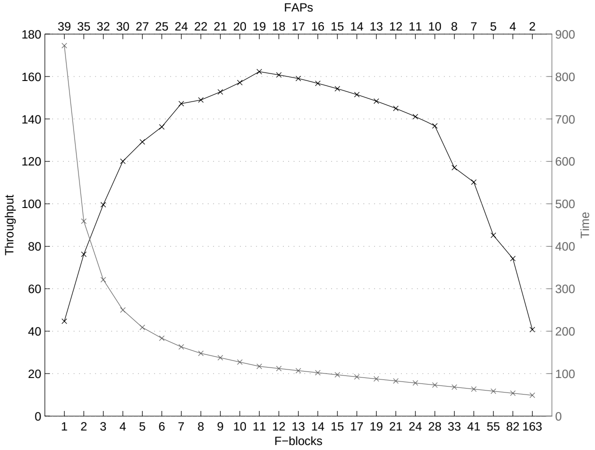


Massey-Omura multiplier

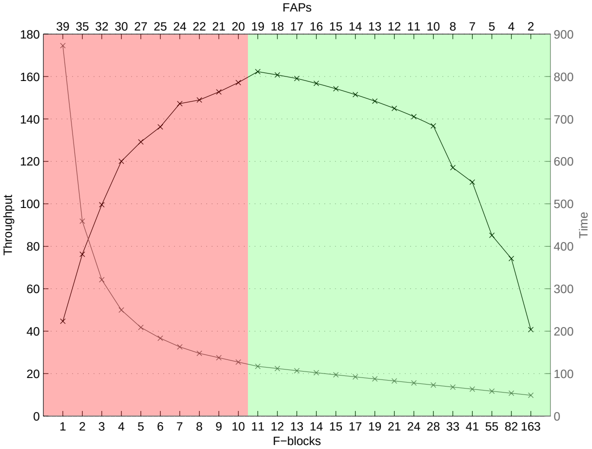
- Bit-serial, only one F-block
- Latency: $m + c + 1$ clock cycles
- Digit-serial, ν F-blocks
- Latency: $\lceil \frac{m}{\nu} \rceil + c + 1$
- ν should be from the set $\mathcal{F} : \{1 - 15, 17, 19, 21, 24, 28, 33, 41, 55, 82, 163\}$



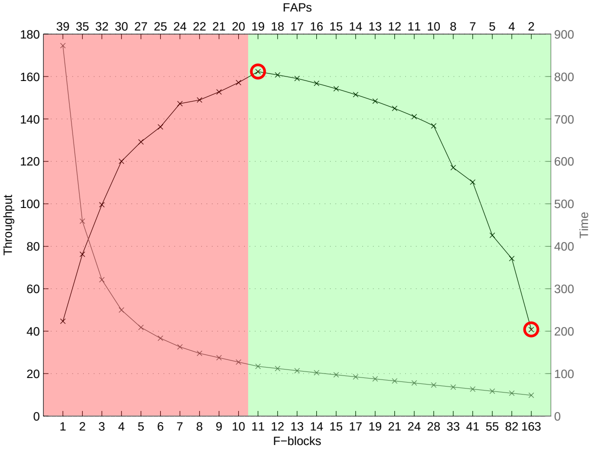
Parameters


FAPs
39 35 32 30 27 25 24 22 21 20 19 18 17 16 15 14 13 12 11 10 8 7 5 4 2
Throughput
0 20 40 60 80 100 120 140 160 180
Time
0 100 200 300 400 500 600 700 800 900
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 19 21 24 28 33 41 55 82 163
F−blocks

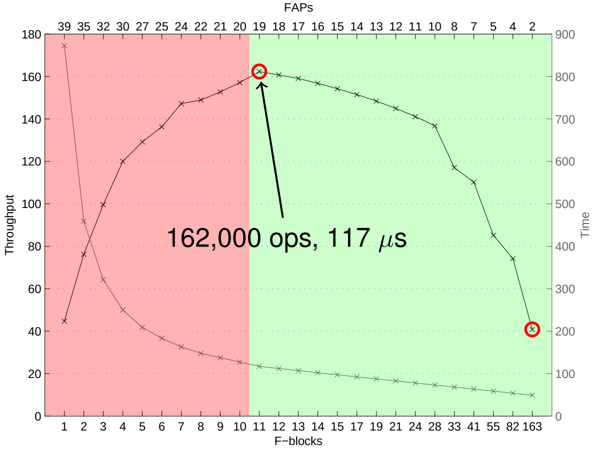
Parameters


FAPs
39 35 32 30 27 25 24 22 21 20 19 18 17 16 15 14 13 12 11 10 8 7 5 4 2
Throughput
0 20 40 60 80 100 120 140 160 180
Time
0 100 200 300 400 500 600 700 800 900
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 19 21 24 28 33 41 55 82 163
F−blocks

Parameters


FAPs
39 35 32 30 27 25 24 22 21 20 19 18 17 16 15 14 13 12 11 10 8 7 5 4 2
Throughput
0 20 40 60 80 100 120 140 160 180
Time
0 100 200 300 400 500 600 700 800 900
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 19 21 24 28 33 41 55 82 163
F−blocks

Parameters


FAPs
39 35 32 30 27 25 24 22 21 20 19 18 17 16 15 14 13 12 11 10 8 7 5 4 2
Throughput
0 20 40 60 80 100 120 140 160 180
Time
0 100 200 300 400 500 600 700 800 900
162,000 ops, 117 µs
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 19 21 24 28 33 41 55 82 163
F−blocks

Results on an Altera Stratix II S180C3

- VHDL
- Altera Quartus II 6.0 SP1

Results on an Altera Stratix II S180C3

- VHDL
- Altera Quartus II 6.0 SP1

Results from Quartus II

- 67,467 ALMs (94 %)
- 240 M512 (26 %), 305 M4K (40 %)
- Two clocks: 164 MHz and 82 MHz

Results on an Altera Stratix II S180C3

- VHDL
- Altera Quartus II 6.0 SP1

Results from Quartus II

- 67,467 ALMs (94 %)
- 240 M512 (26 %), 305 M4K (40 %)
- Two clocks: 164 MHz and 82 MHz

Performance

- One verification 114.2 μs (average)
- Up to 166,000 ops!

Conclusions and future work

- Very high ops achievable with modern FPGAs
  - Development in FPGAs: speed and area
  - Parallelization
  - Time of single operation vs. ops

Conclusions and future work

- Very high ops achievable with modern FPGAs
  - Development in FPGAs: speed and area
  - Parallelization
  - Time of single operation vs. ops
- Future work:
  - Polynomial basis?
  - Counterpart implementations for signature generation
  - Other operations (hash, modular arithmetic)
  - Possible problems (side channel attacks, power, etc.)

Thank you.
Questions?