Comparaison d'algorithmes de réduction modulaire en HLS sur FPGA

Libey Djath[1], Timo Zijlstra[2], Karim Bigou[1], Arnaud Tisserand[2]

[1] Université de Bretagne Occidentale / Lab-STICC, UMR CNRS 6285
[2] CNRS / Lab-STICC, UMR 6285

Compas, 25-28 Juin 2019, Anglet, France

Contexte

La cryptographie asymétrique sert par exemple :

- signature numérique
- authentification
- échange de clés secrètes

Exemples de cryptosystèmes asymétriques :

- cryptographie basée sur les courbes elliptiques (ECC) [Mil85, Kob87]
- cryptographie post-quantique (PQC) [Reg05, LPR10]

Les calculs dans ces cryptosystèmes sont effectués sur :

- ECC : entiers modulo un grand nombre premier P (200–500 bits)
- PQC à base de réseaux euclidiens : polynômes de degrés $d \in [200, 1000]$ avec de petits coefficients (10–20 bits)

Residue Number System (RNS) pour ECC

RNS

- X représenté par ses restes dans une base $(a_1, a_2, ..., a_n)$
- représentation non positionnelle des nombres
- théorème chinois des restes (CRT) pour les conversions

Représentation du nombre X

$$\overrightarrow{X} = (X \bmod a_1 \ , \quad X \bmod a_2 \ , \quad \ldots \ , \quad X \bmod a_n \ )$$

Une primitive ECC requiert des milliers d'additions, soustractions et multiplications modulo P

En RNS, les calculs sur de grands entiers sont remplacés par des calculs en parallèle sur de petits restes mod a_i.

Cryptographie post-quantique (PQC)

Les cryptosystèmes actuels peuvent être cassés par l'algorithme quantique de Shor [Sho99] sur un ordinateur quantique

PQC

Basée sur des problèmes mathématiques pour lesquels il n'existe pas, a priori, d'algorithmes quantiques efficaces. Par exemple :

- réseaux euclidiens [LPR10]
- codes correcteurs d'erreurs

Opération la plus importante de PQC à base de réseaux euclidiens :

multiplication de polynômes

Les coefficients des polynômes sont dans $GF(q)$, avec q un premier de quelques dizaines de bits

Réduction modulaire

Les primitives ECC requièrent :

- réductions mod P
- réductions et accumulations de multiplications réduites si on utilise le RNS

Les primitives PQC à base de réseaux euclidiens requièrent :

- milliers de réductions de multiplications
- accumulations puis reductions

Comparaison d'algorithmes de réduction modulaire

Bibliothèque d'arithmétique modulaire

Développement d'une bibliothèque *C* d'arithmétique modulaire pour la cryptographie asymétrique :

- synthèse de haut niveau (HLS)
- motifs de calcul (réduction modulaire) rencontrés dans les cryptosystèmes asymétriques
- comparaison d'algorithmes de réduction sur FPGA

Motifs de calcul

Les motifs de calcul :

- $M1 : \sum_{i=1}^{N} x_i \bmod m$
- $M2 : \sum_{i=1}^{n} x_i \times y_i \bmod m$

Stratégies de réduction (pour $M2$) :

- réduction intermédiaire systématique (RIS) :
  $\left(\sum_{i=1}^{N}(x_i \times y_i \bmod m)\right) \bmod m$
- réduction seulement à la fin (RSF) :
  $\left(\sum_{i=1}^{N} x_i \times y_i\right) \bmod m$

Algorithmes implantés

Algorithme de référence :

réduction native de l'outil

Algorithmes implantés pour des moduli quelconques :

- réduction de Barrett
- réduction de Montgomery

Algorithmes implantés pour des moduli spécifiques :

- MSR : moduli de la forme $2^w - c$, avec $c < 2^{w/2}$
- MSC : moduli à écriture binaire très creuse
  (p. ex. 3 bits non nuls)

Code algorithme réduction de Barrett

```
1  #include "parameters.h"
2  #include "arithmod_internal.h"
3
4  word barrett(sumdword x)
5  {
6      sumword x1 = SUM_W(x >> width);
7      sumword q = SUM_W((RSW(x1) * RSW(R_const)) >> (shift - width));
8      word x0 = W(x);
9      counter c = 0;
10     if (x0 > M) c = 2;
11     else if (x0 != 0) c = 1;
12     q = q + c;
13     sumdword z = SUM_DW(q) * SUM_DW(m);
14     signword res = x - z;
15     if (res < 0) res = res + M;
16     if (res < 0) res = res + M;
17     return W(res);
18 }
```

Implantation matérielle

FPGA cible

Artix7 de Xilinx (xc7a15tcpg236-2l)

Outil

Vivado HLS (version 2017.4) de Xilinx

Implantation

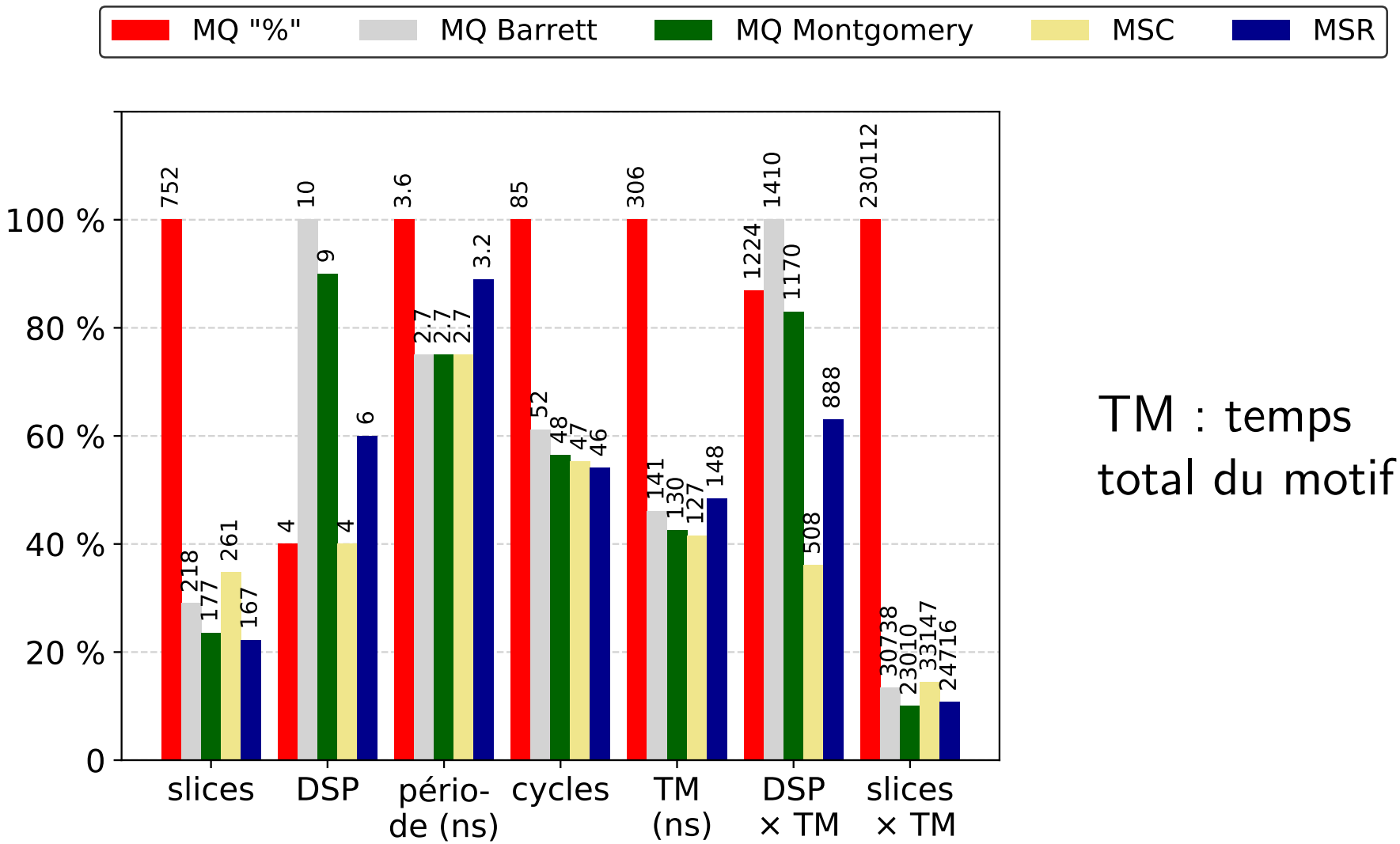
- moduli de tailles $w = 13, 17, 23, 30$ bits
- vecteurs de tailles $N = 10, 20, 40, 100$

Optimisation

Mêmes directives d'optimisation pour tous les algorithmes :

- pipeline
- déroulage de boucles

Comparaison des différents algorithmes de réductions pour $w = 23$ bits, M2-RSF, $N = 20$



TM : temps total du motif

Impact des directives d'optimisation

Impact des directives d'optimisation pour M2-RSF, MSR, $w = 23$ bits et $N = 20$

RSF : réduction seulement à la fin

directives	surface		temps (ns, cycles)			surface × temps	
	slices	DSP	période	cycles	TM	DSP × TM	slices × TM
aucune	136	4	3.1	216	670	2680	91120
`pipeline`	142	4	3.2	64	205	820	29110
`pipeline + unroll2`	167	6	3.2	46	148	888	24716
`pipeline + unroll4`	228	10	3.3	37	123	1230	28044
`pipeline + unroll10`	526	22	3.1	39	121	2662	63646

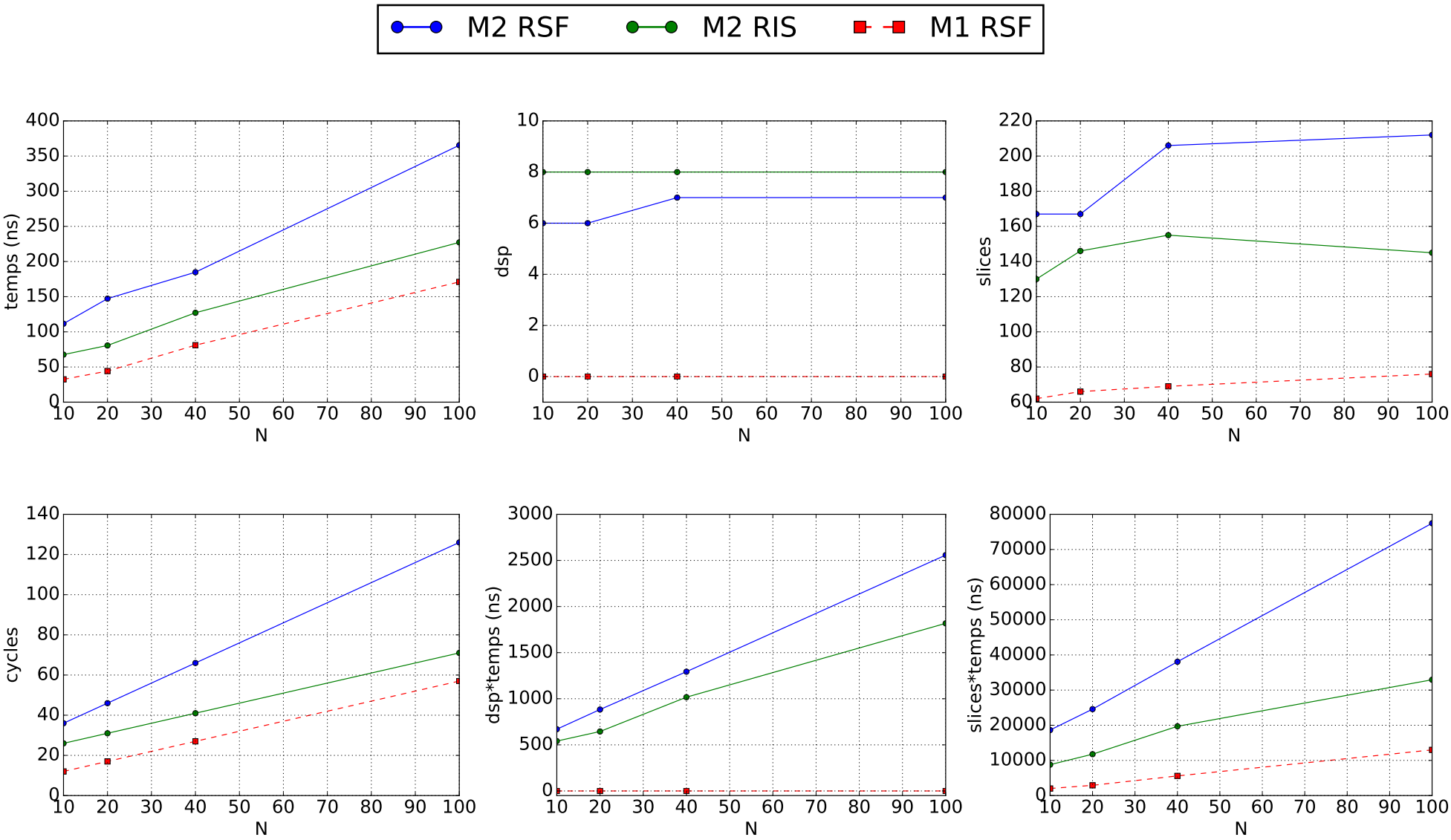
Impact de la stratégie de réduction

Impact des stratégies de réduction pour $M2$, $w = 23$ bits et $N = 20$

RIS : réduction intermédiaire systématique
RSF : réduction seulement à la fin

motif	algorithme et stratégie	surface		temps (ns, cycles)			surface×temps	
		slices	DSP	période	cycles	TM	DSP×TM	slices×TM
M2	Montgomery RIS	194	12	2.6	60	156	1872	30264
	Montgomery RSF	149	7	2.6	64	167	1165	24794
	Barrett RIS	259	12	2.8	53	149	1781	38436
	Barrett RSF	218	10	2.7	52	141	1404	30608
	MSC RIS	403	4	2.7	55	149	594	59846
	MSC RSF	261	4	2.7	47	127	508	33121
	MSR RIS	146	8	2.6	31	81	645	11768
	MSR RSF	167	6	3.2	46	148	884	24583

Impact de la taille N des vecteurs pour *MSC*



Conclusion

Conclusion

La bibliothèque proposée offre :

- support d'algorithmes de réduction modulaire adaptés à la cryptographie asymétrique, et pas supportés par les outils HLS actuels

  Les implantations sont 2 fois plus rapides, avec un meilleur compromis surface $\times$ temps

- possibilité de générer des circuits optimisés en temps et en surface pour des moduli quelconques et spécifiques

Travaux à venir

Dans la suite, nous souhaitons ajouter à notre bibliothèque :

- autres opérations
- autres formes de moduli

References I

[Kob87] N. Koblitz.
Elliptic curve cryptosystems.
volume 48, pages 203–209. American Mathematical Society, 1987.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev.
On ideal lattices and learning with errors over rings.
In *Proc. 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 1–23. Monaco, June 2010.

[Mil85] V .S. Miller.
Use of elliptic curve in cryptography.
In *Advances in Cryptology*, volume 218, pages 417–426. Springer, 1985.

[Reg05] O. Regev.
On lattices, learning with errors, random linear codes, and cryptography.
In *Proc. 37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, MD, USA, May 2005.

[Sho99] P. W. Shor.
Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer.
volume 41, pages 303–332, 1999.