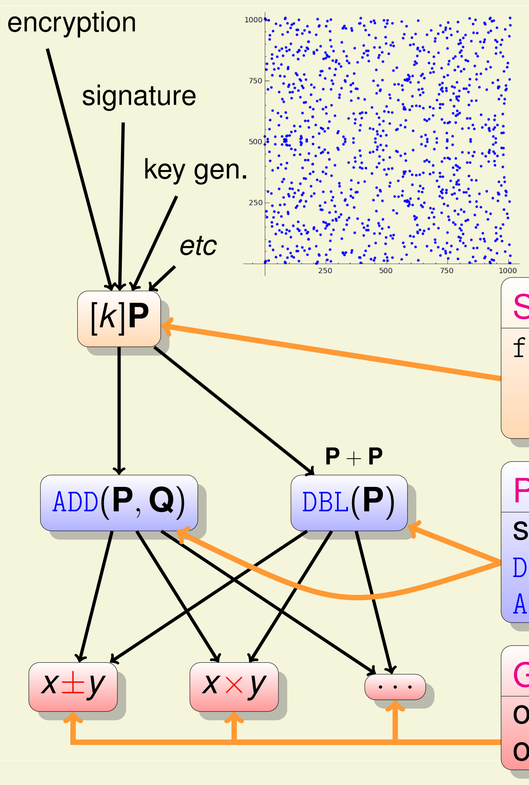
Finite Field Multiplier Architectures for Hyper-Elliptic Curve Cryptography

Gabriel GALLIN & Arnaud TISSERAND

1. Elliptic Curve Cryptography (ECC)

protocol level / curve level / field level



$E : y^2 = x^3 + 4x + 20$ over GF(1009)

Points on E: $\mathbf{P}$, $\mathbf{Q}$= (x, y) or (x, y, z)

Coordinates: $x, y, z \in$ GF$(\cdot)$

GF$(\mathcal{P})$, GF(2^m), t : 200–600 bits

$k = (k_{t-1}k_{t-2}\dots k_1k_0)_2 \in \mathbb{N}$

Scalar multiplication operation

```
for i from 0 to t − 1 do
    if k_i = 1 then Q = ADD(P, Q)
    P = DBL(P)
```

Point addition/doubling operations

sequence of finite field operations

DBL: $v_1 = z_1^2, v_2 = x_1 - v_1, \dots$

ADD: $w_1 = z_1^2, w_2 = z_1 \times w_1, \dots$

GF$(\mathcal{P})$ or GF(2^m) operations

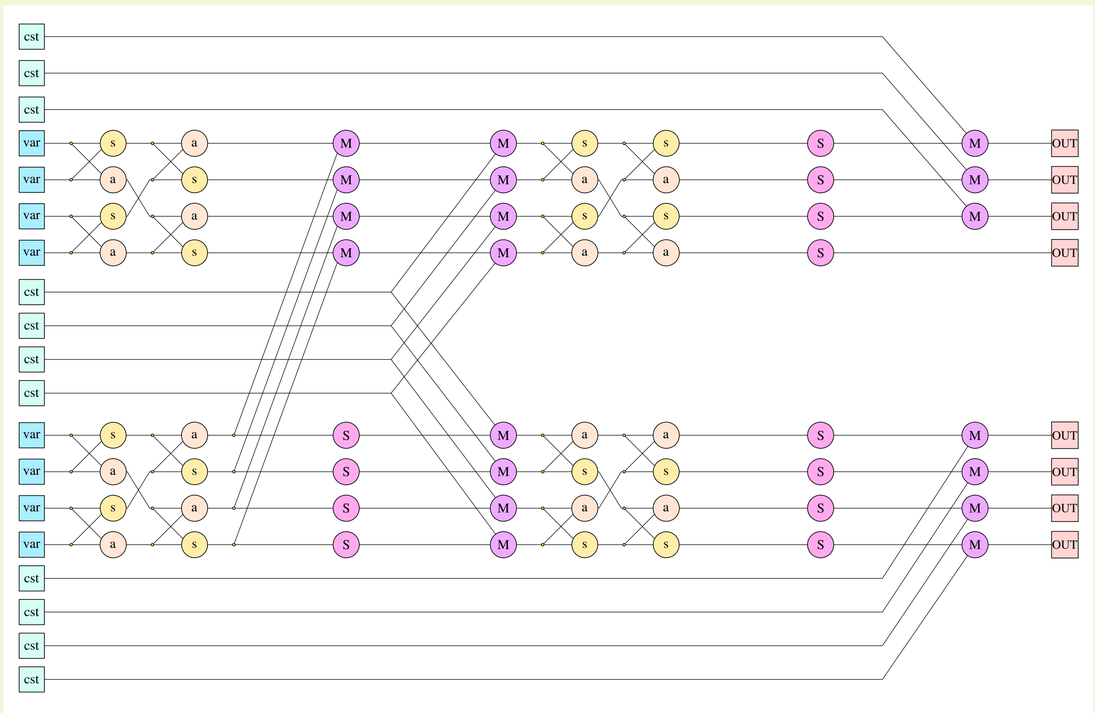
operation modulo large prime (GF$(\mathcal{P})$) or irreducible polynomial (GF(2^m))

2. Hyper-Elliptic Curve Cryptography (HECC)

	GF$(\mathcal{P})$ size	ADD	DBL	source
ECC	ℓ_{ECC}	$12\mathtt{M} + 2\mathtt{S}$	$7\mathtt{M} + 3\mathtt{S}$	[EFD]
HECC	$\ell_{\mathrm{HECC}} \approx \frac{1}{2}\ell_{\mathrm{ECC}}$	$40\mathtt{M} + 4\mathtt{S}$	$38\mathtt{M} + 6\mathtt{S}$	[Lange 2005]
Kummer	ℓ_{HECC}	$19\mathtt{M} + 12\mathtt{S}$		[Renes et al. 2016]

HECC based on Kummer surfaces is more efficient than ECC:

- ARM Cortex M0: 75% clock cycles reduction for signatures
- AVR AT-mega: 32% cycles reduction for Diffie-Hellman

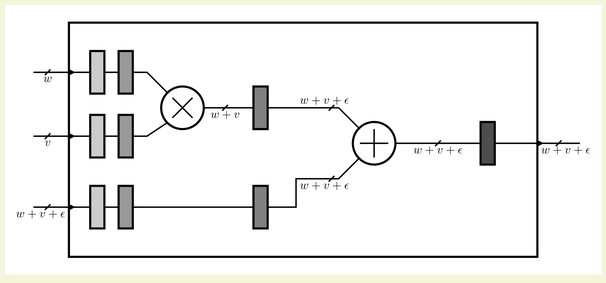


3. Montgomery Modular Multiplication (MMM)

Proposed by Peter L. Montgomery in 1985 (base for variations)

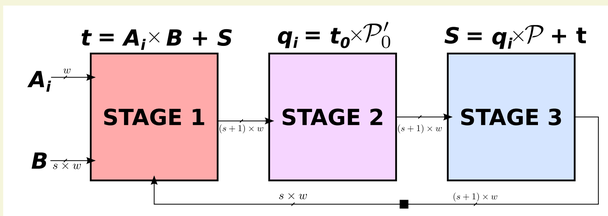
$R = A \times B$	[n bits $\times n$ bits $\rightarrow 2n$ bits]
$q = (R \times (-\mathcal{P}^{-1}))$ mod (2^n)	[n bits $\times n$ bits $\rightarrow n$ bits]
$q\mathcal{P} = q \times \mathcal{P}$	[n bits $\times n$ bits $\rightarrow 2n$ bits]

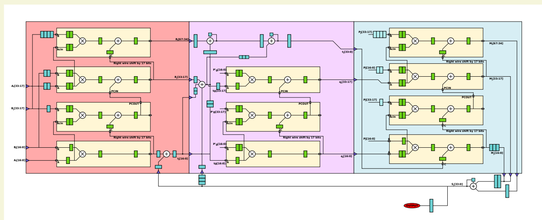
4. Pipeline in DSP Blocks

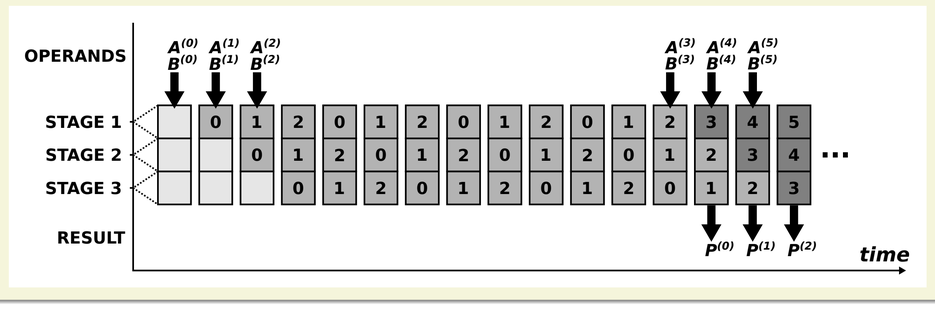


5. Hyper-Threaded Montgomery Multiplier (HTMM)

- One stage for each internal partial product
- 3 to 4 DSP slices in each stage







6. FPGA Implementation Results

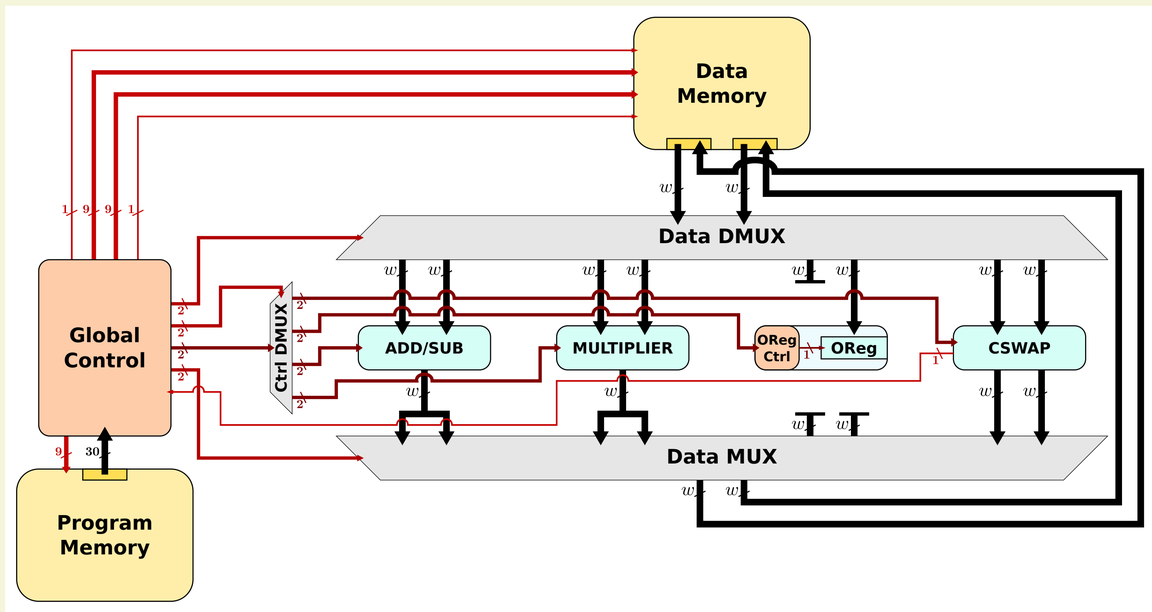
Results for 3M with $\mathcal{P}$ on 128 bits (4×34 b words), VHDL implementations for all versions, including fastest MMM implementation from [Ma et al. 2013]

Virtex 4 XC4VLX100, Virtex 5 XC5VLX110T, Spartan 6 XC6SLX75

HTMM_DRAM: operands in slices, HTMM_BRAM: operands in BRAMs

Version	FPGA	DSP	BRAM 18K/9K	FF	LUT	Slices	Freq. (MHz)	nb. cycles	Time (ns)
[Ma et al. 2013]	V4	21	6/0	1311	1201	879	252	65	258
	V5	21	6/0	1310	1027	406	296		220
	S6	21	0/6	1280	1600	540	210		309
HTMM_DRAM	V4	11	0/0	1638	1128	1346	330	79	239
	V5	11	0/0	1616	652	517	400		198
	S6	11	0/0	1631	1344	483	302		261
HTMM_BRAM	V4	11	2/0	615	364	449	328	79	241
	V5	11	2/0	593	371	249	357		221
	S6	11	0/2	587	359	180	304		260
HTMM_NQM	V4	8	2/0	667	432	524	329	94	286
	V5	8	2/0	643	371	234	360		261
	S6	8	0/2	629	356	206	308		305

7. 256b ECC vs 128b HECC (similar theoretical security)



FPGA.	Version	DSP	BRAM 36K/18K	Slices	Freq. (MHz)	nb. cycles	Time (ms)
V4	ECC	37	0/11	4655	250	109297	0.44
	HECC_1u	11	0/7	1413	330	183051	0.55
	HECC_2u	22	0/9	2356	330	115211	0.35
V5	ECC	37	10/0	1725	291	109297	0.38
	HECC_1u	11	0/7	873	360	183051	0.51
	HECC_2u	22	0/9	1542	360	115211	0.32

CNRS – Lab-STICC – IRISA http://h-a-h.inria.fr/