AN EFFICIENT MANY-CORE ARCHITECTURE FOR ELLIPTIC CURVE CRYPTOGRAPHY SECURITY ASSESSMENT

Marco Indaco, Fabio Lauri, Andrea Miele, Pascal Trotta

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

POLITECNICO DI TORINO
1859 1906

INTRODUCTION

- **Public-key cryptography (PKC)**: tools for key-exchange and digital signature, use of (public key, private key) pairs

- Elliptic curves (standard) public-key tool: use group of points and security based on discrete logarithm problem hardness in group

- **Security: Pollard rho $O(\sqrt{|G|})$, using negation map $\approx \sqrt{\frac{\pi|G|}{4}}$**

- **But how long would it take to break it? Seconds, years, decades, aeons?**

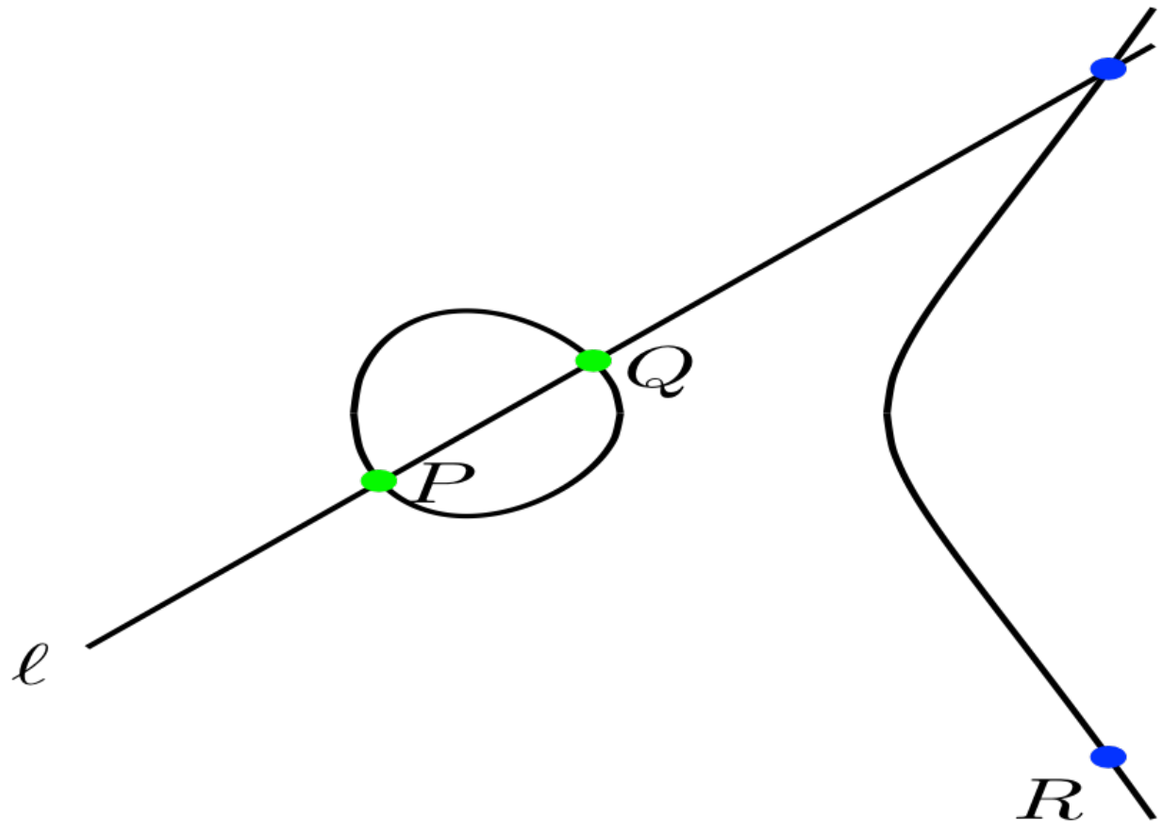
MOTIVATION

- Analyze performance of Pollard rho on new computing devices to assess what security level we can break in practice

- Only a few works on solving ECDLP on prime fields on FPGAs (no negation map)

- Main goal: cost of solving Certicom challenge ECCP-131 on cluster of FPGAs using Pollard rho with negation map

ELLIPTIC CURVES



$y^2=x^3+a_1x+a_0$ $\#E(F_p) \approx p$

Weierstrass coordinates: (x,y)

$P=(x_1,y_1)$, $-P=(x_1,-y_1)$

Affine addition: 2m+1s+6a+1i

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

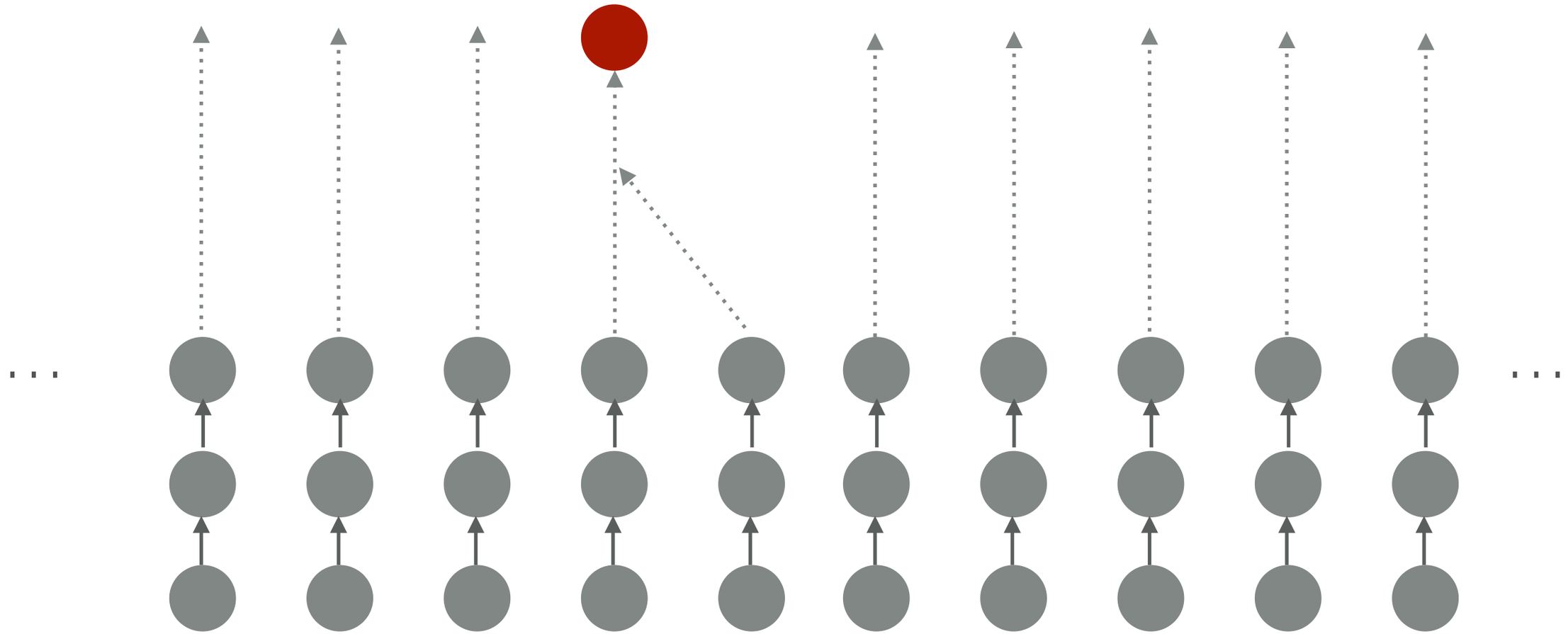
$x_3=t^2-x_1-x_2$, $y_3=t(x_1-x_3)-y_1$

$t=(y_1-y_2)/(x_1-x_2)$

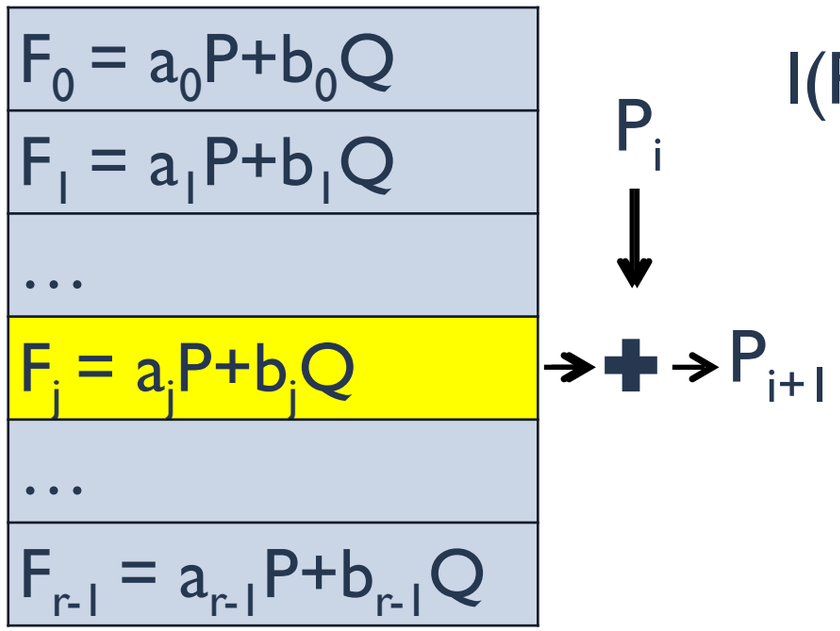
PARALLEL POLLARD RHO

- ECDLP in large prime order **<P>**: given **Q** in **<P>** find integer **k** s. t. **Q=kP**

- Run **m** (speed-up: **m**) walks producing pseudo-random points till 2 collide



POLLAR RHO ITERATION

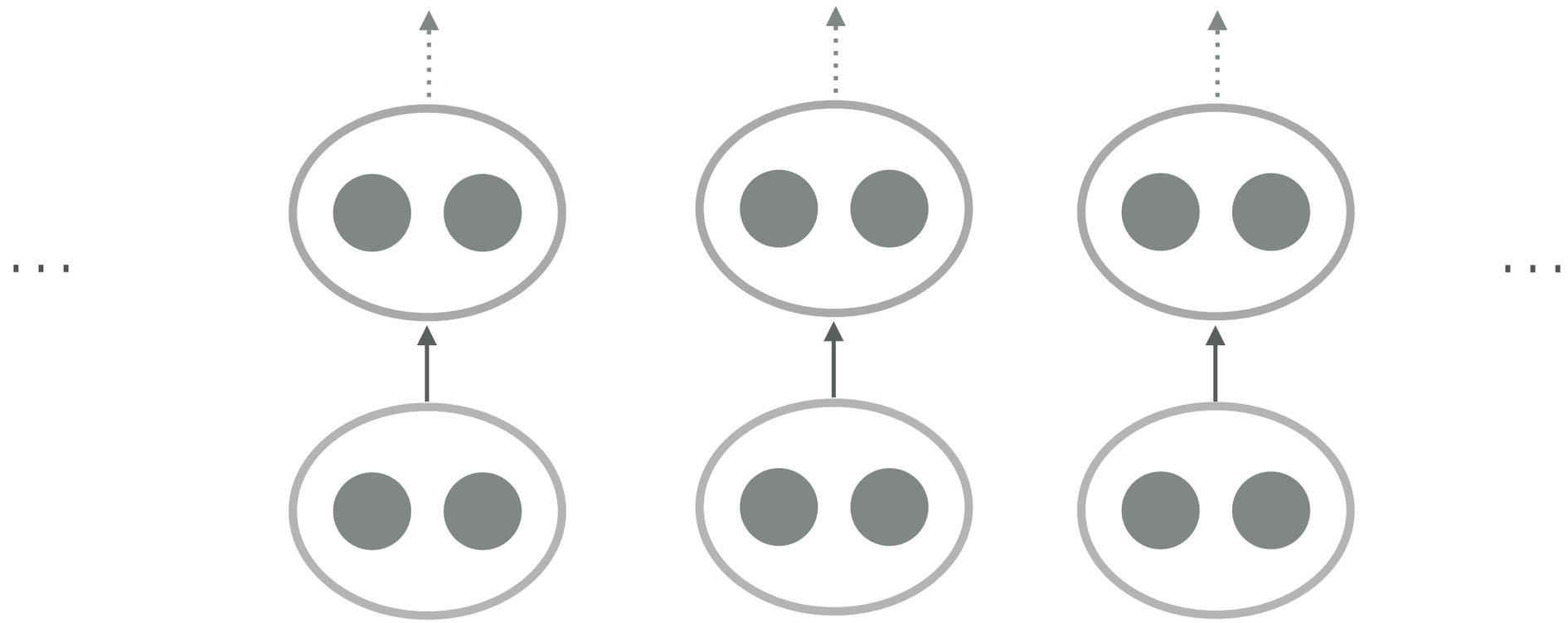


- Run **m** independent walks using the same table. Define distinguished points (easy to check property). Check for distinguished point collision

USING THE NEGATION MAP

- Search for collision of equivalence classes of size **2** induced by the negation map: **p ~ -p,** search for collision of **±p**

- Theoretical $\sqrt{2}$ speed-up. Practice: little overhead + **fruitless cycles**



POLLARD RHO ON FPGAS

- Prime field of size **k** (e.g., **k=131**)

- Addition/subtraction modulo prime: **1** clock cycle

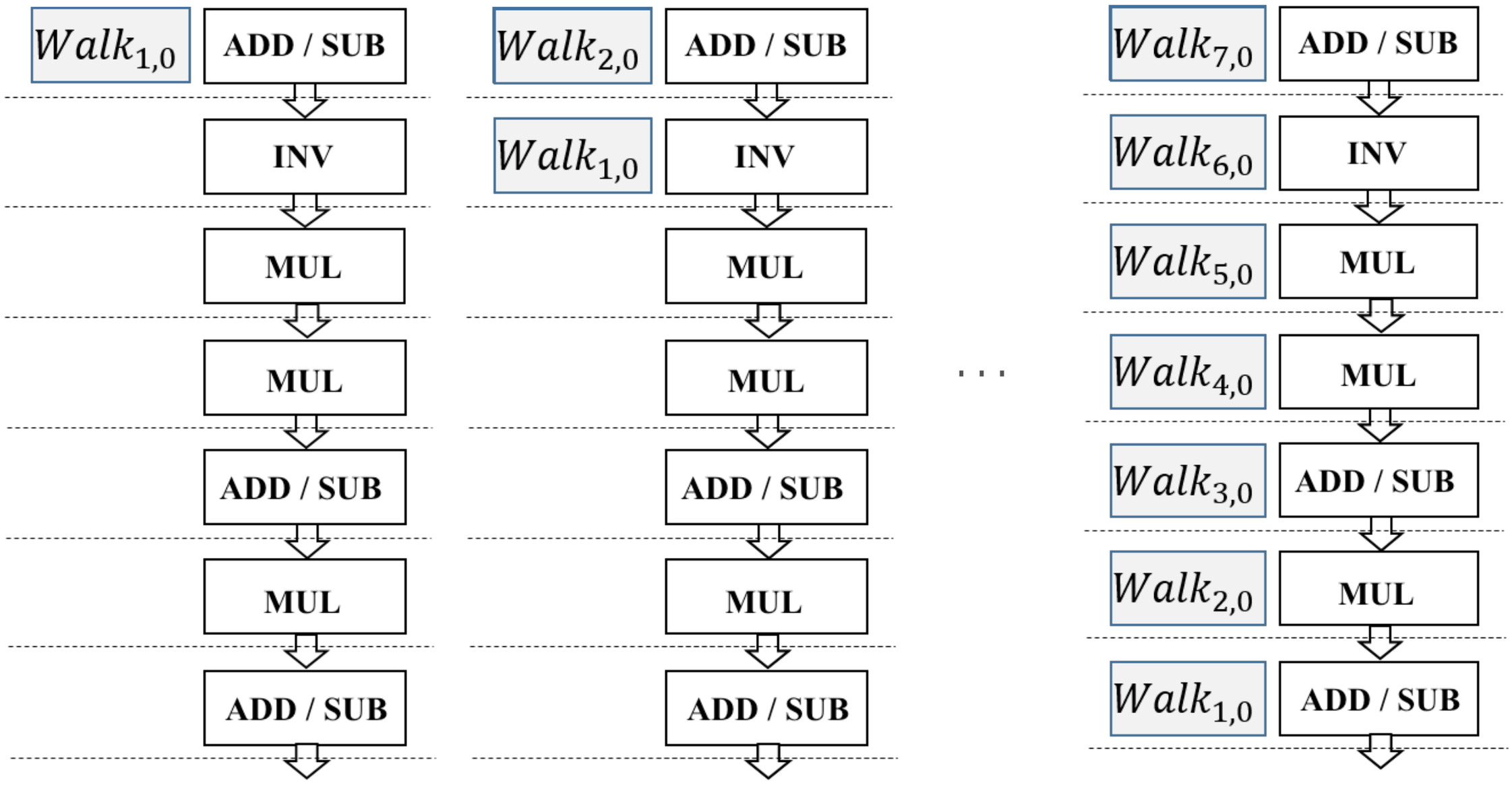
- Montgomery multiplication: **k** clock cycles

- Kaliski inversion: **2k** clock cycles

- Negation map, cycle handling and analysis with large **r** (e.g., 2^{14})

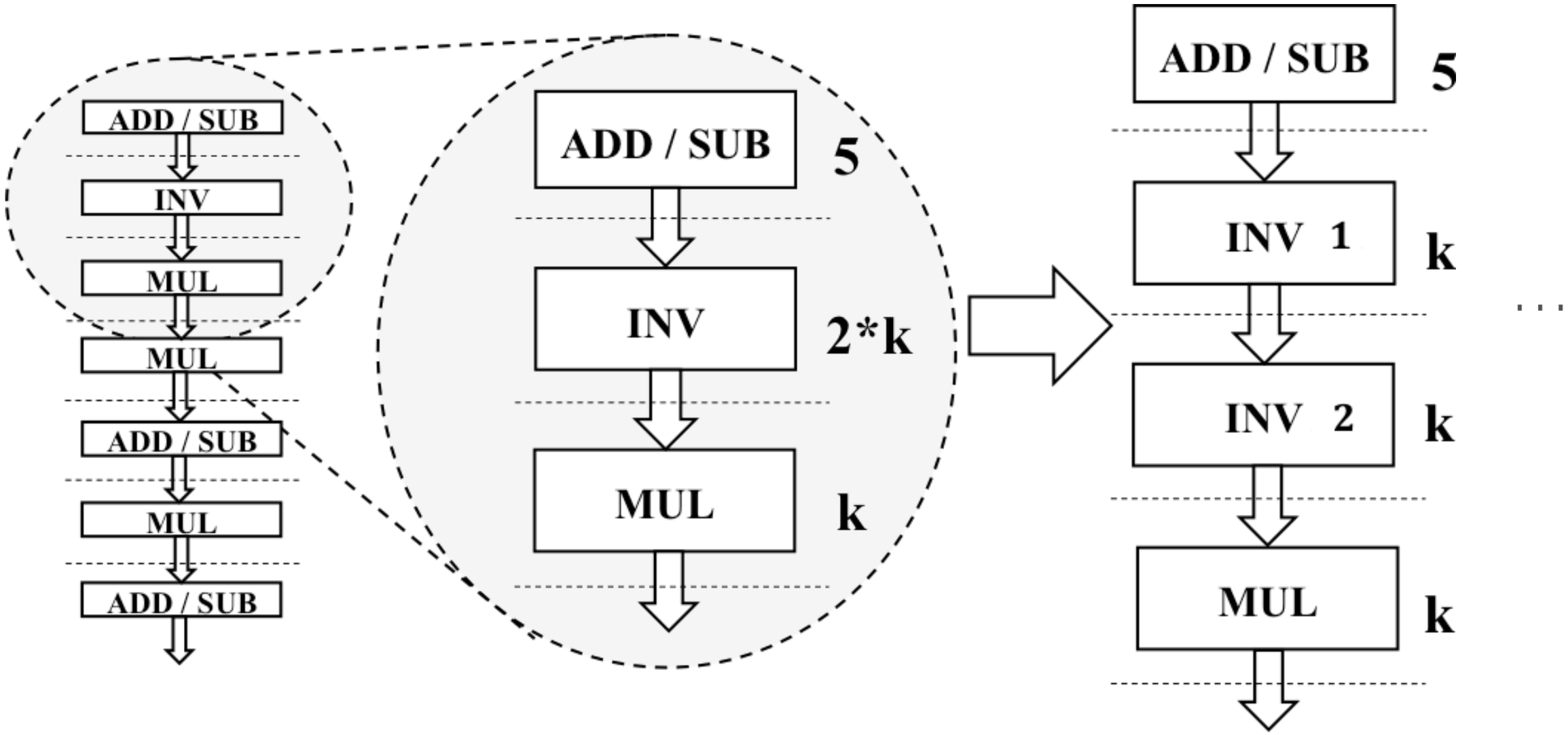
SINGLE-PIPE-MULTIPLE-WALK (SPMW) CORE

Walk <=> "HW" thread: pipeline elliptic curve addition



OPTIMIZATION: UNROLLING

- **k**-bit prime, MUL (**k** cycles), INV (**2k** cycles): $t_{max} = 2k$, throughput $\cong 1/t_{max}$
- **Idea**: split slowest stage into **2** stages (replicas) running for $t_{max}/2$ cycles



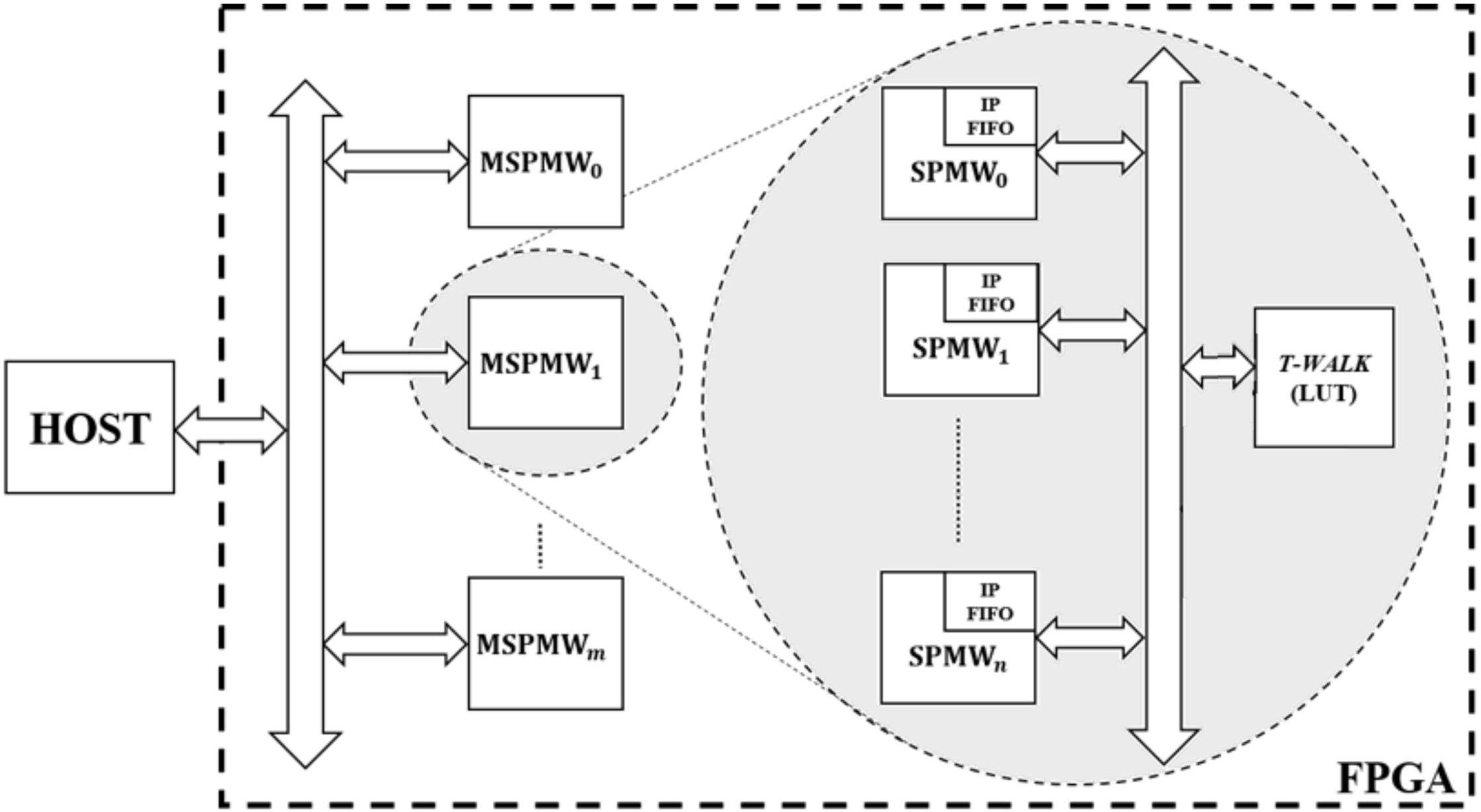
UNROLLING VS MORE CORES

- **We want to maximize throughput under area constraints**
- We can find the optimal values for **#Stages** and **#Cores**

For k=131 on a Xilinx Virtex 7-xc7v2000t

#Cores	#Stages (#walks)	t_{max}	Freq	Power
11	78	9	192 Mhz	26.9 W

SYSTEM OVERVIEW



RESULTS

Speed-up: 3.9x vs [GPP08] and 4.8x vs [JMS12]

Certicom ECCp-131 challenge

FPGA	Unit price	Points/sec	Overall cost to solve in 1 year	#Devices
Virtex-5 vlx330t	8.4 K$	20.5 M	453 M$	44K
Virtex-6 vlx760	12.6 K$	67.3 M	207 M$	15K
Virtex-7 v2000t	17.4 K$	211.2 M	91 M$	4.6K
RIVYERA V7	500 K$	8448 M	65 M$	114
Virtex UltraScale 440	25 K$	738 M	34 M$	1.3K

CONCLUSION

- Solving Certicom ECCp-131 on FPGA cluster infeasible (near future?)
- Future work:

1. Estimate performance of ASIC implementation
2. Optimize arithmetic for k=131 and specific FPGA (e.g., low cost)

THANKS FOR YOUR ATTENTION!