Generalized Polynomial Decomposition for S-boxes with Application to Side-Channel Countermeasures

Dahmun Goudarzi, Matthieu Rivain, Srinivas Vivek, and Damien Vergnaud

Background

Secure Software S-box Implementations

- Higher-Order Masking

$$x = x_1 + x_2 + \cdots + x_d$$

- Main Challenge: S-box evaluations
  - Linear operations: $O(d)$
  - Non-linear operations: $O(d^2)$

- Goal: Find S-box representation with less non-linear operations

Polynomial Methods

- S-box seen as a polynomial over $\mathbb{F}_{2^n}$

$$S(x) = \sum_{i=0}^{n} a_i x^i$$

Generic Methods

$$S(x) = \sum_i (p_i \star q_i)(x)$$

- CRV decomposition, $\star = \times$
- Algebraic decomposition, $\star = \circ$

Specific Methods: example on AES

$$S_{\mathrm{AES}}(x) = \mathrm{Aff}(x^{254})$$

- RP 4-mult chain on $\mathbb{F}_{2^8}$
- KHL 5-mult chain on $\mathbb{F}_{2^4}$

Bitslice Methods

- S-box seen as a Boolean circuit

$$S\colon x \to (f_1(x), f_2(x), \ldots, f_m(x))$$

Generic Methods

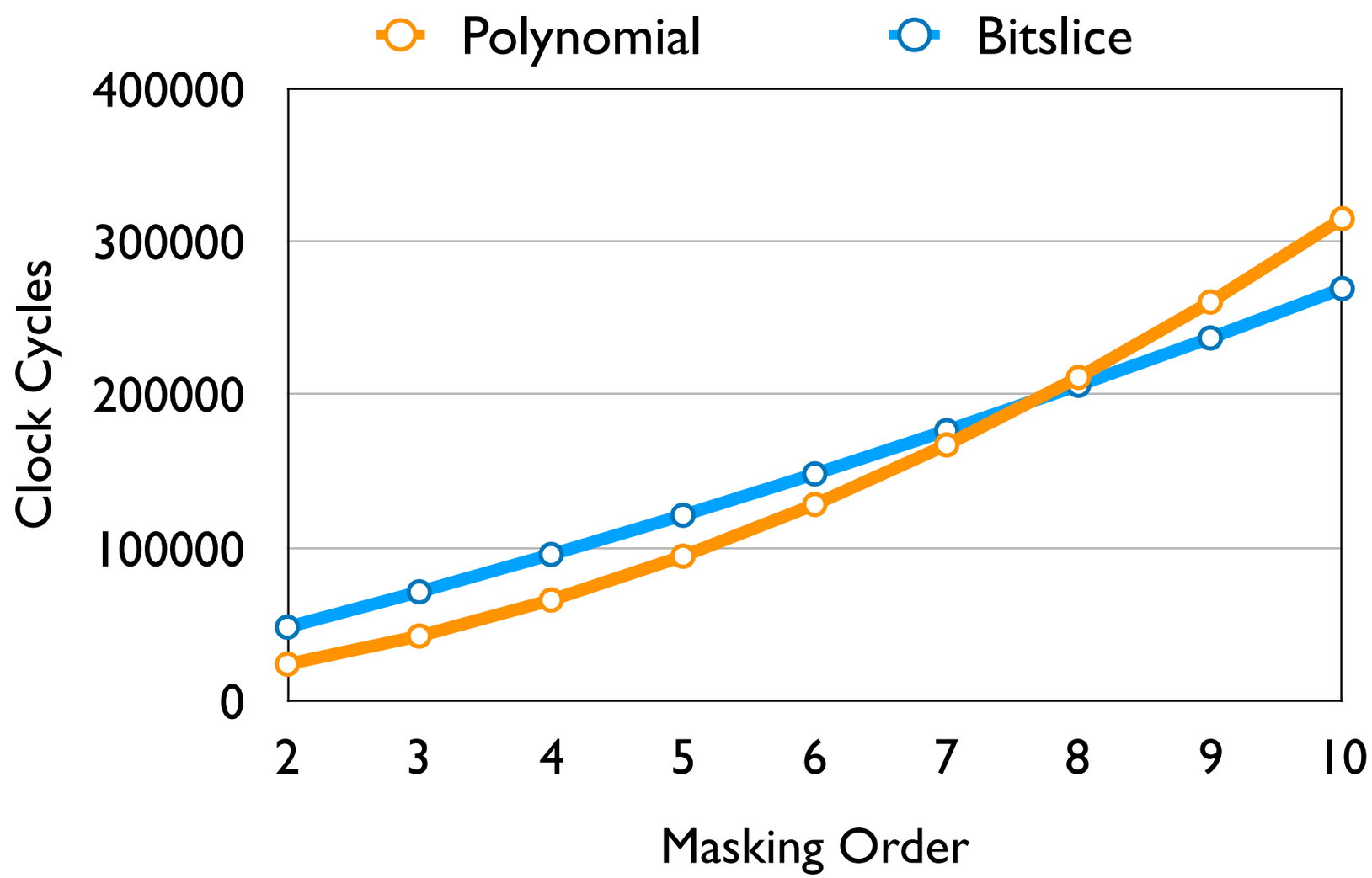
- Based on Boolean functions

Specific Methods: example on AES

- Based on a Boolean circuit (BMP13)

Polynomial vs Bitslice

- Generic 8-bit S-box evaluation



Full Field

CRV decomposition

Boolean Field

Boolean decomposition

Intermediate Field ?

This work

$$S(x) \rightarrow (S_1(y,z), S_2(y,z))$$

1 8-bit function

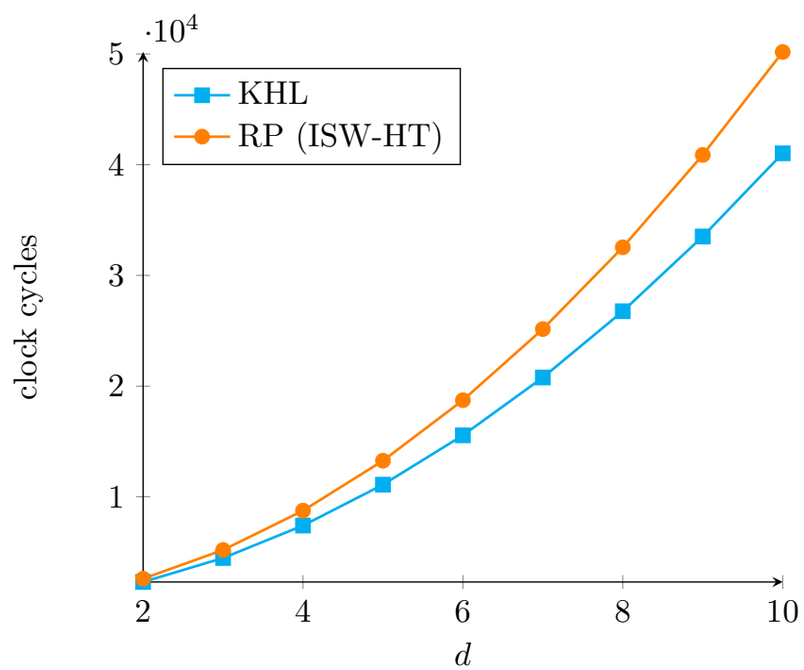
4-bit functions

Motivation

- Working on smaller fields
  - Degree of parallelisation increased (32-bit architecture)

	Boolean Case	4-bit field	8-bit field
Mult. in //	32	8	4

  - Example: 16 AES S-box with polynomial method



Our results

- Generalized decomposition method for any S-boxes w.r.t 3 parameters
  - n : number of inputs
  - m : number of output elements
  - λ : bit-size of the elements

- Study of the median case: example on 8-bit S-boxes:

$$S(x, y) = (f_1(x, y), f_2(x, y)) \text{ with } x, y \in \mathbb{F}_{2^4}$$

- Implementation in ARM assembly to compare with state of the art

Generalized Decomposition Method

S-box Characterization

- S-box seen as a $n\lambda$ -bit to $m\lambda$ -bit polynomial over $\mathbb{F}_{2^\lambda}$:

$$S(x) = (f_1(x), f_2(x), \ldots, f_m(x))$$

where $f_1, f_2, \ldots, f_m \in \mathcal{F}_{n,\lambda}$ (set of functions from $\mathbb{F}_{2^\lambda}^n$ to $\mathbb{F}_{2^\lambda}$)

Coordinate Function Decomposition

$$f(x) = \sum_{i=0}^{t} g_i(x) \cdot h_i(x)$$

- g_i : random linear combinations from a basis $\langle \bar{\mathcal{B}} \rangle$ with

$$\langle \bar{\mathcal{B}}_i \rangle = \left\{ g,\ g = \sum_{i=0}^{\lambda-1} \sum_{\phi \in \mathcal{B}} c_{\phi,i} \times \phi^{2^i} \right\}$$

- find $c_{i,j}$ s.t $h_i = \sum_j c_{i,j} \phi_j$ by solving :

$$f(x) = \sum_i (\sum_j a_{i,j} \phi_j(x))(\sum_j c_{i,j} \phi_j(x)),\ \forall x$$

Solving a Linear System

$$f(x) = \sum_i (\sum_j a_{i,j}\phi_j(x))(\sum_j c_{i,j}\phi_j(x)),\ \forall x$$

- $\{e_i\}_{i=1}^{2^{n\lambda}} = \mathbb{F}_{2^\lambda}^n$

  $A_1c_1 + A_2c_2 + \cdots + A_tc_t = (f(e_1), f(e_2), \ldots, f(e_{2^n}))$

$$A_i = \begin{pmatrix} \phi_1(e_1)\cdot g_i(e_1) & \phi_2(e_1)\cdot g_i(e_1) & \ldots & \phi_{|\mathcal{B}|}(e_1)\cdot g_i(e_1) \\ \phi_1(e_2)\cdot g_i(e_2) & \phi_2(e_2)\cdot g_i(e_2) & \ldots & \phi_{|\mathcal{B}|}(e_2)\cdot g_i(e_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(e_{2^n})\cdot g_i(e_{2^n}) & \phi_2(e_{2^n})\cdot g_i(e_{2^n}) & \ldots & \phi_{|\mathcal{B}|}(e_{2^n})\cdot g_i(e_{2^n}) \end{pmatrix}$$

Conditions

- $(t+1)|\mathcal{B}|$ unknowns, $2^{n\lambda}$ equations:

$$(t+1)|\mathcal{B}| \geq 2^{n\lambda}$$

- Condition on the sum: $t \geq \left\lceil \frac{2^{n\lambda}}{|\mathcal{B}|} \right\rceil - 1$

- Condition on the basis: $\langle \bar{\mathcal{B}} \times \bar{\mathcal{B}} \rangle$ has to span the entire space $\mathcal{F}_{n,\lambda}$

Spanning Property

$$\langle \bar{\mathcal{B}} \times \bar{\mathcal{B}} \rangle = \mathcal{F}_{n,\lambda} \iff \mathrm{rank}(\mathrm{Mat}(\bar{\mathcal{B}} \times \bar{\mathcal{B}})) = 2^{n\lambda}$$

with

$$\mathrm{Mat}(\mathcal{S}) = \begin{pmatrix} \varphi_1(e_1) & \varphi_2(e_1) & \dots & \varphi_{|\mathcal{S}|}(e_1) \\ \varphi_1(e_2) & \varphi_2(e_2) & \dots & \varphi_{|\mathcal{S}|}(e_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(e_{2^{n\lambda}}) & \varphi_2(e_{2^{n\lambda}}) & \dots & \varphi_{|\mathcal{S}|}(e_{2^{n\lambda}}) \end{pmatrix}$$

where $\{\varphi_1, \varphi_2, \dots, \varphi_{|\mathcal{S}|}\} = \mathcal{S}$ and $\mathcal{S} = \langle \bar{\mathcal{B}} \times \bar{\mathcal{B}} \rangle$

Basis Construction

- Start with $\mathcal{B} = \{1, x_1, x_2 \ldots, x_n\}$

- Pick ϕ, ψ in $\langle \bar{\mathcal{B}} \rangle$ at random, where

$$\langle \bar{\mathcal{B}} \rangle = \left\{ g \ , \ g = \sum_{i=0}^{\lambda-1} \sum_{\phi \in \mathcal{B}} c_{\phi,i} \times \phi^{2^i} \right\}$$

- Compute $\mathrm{rank}(\mathrm{Mat}(\mathcal{S} \times \mathcal{S}))$ with $\mathcal{S} = \mathcal{B} \cup \phi \cdot \psi$

- Redo N times and choose (ϕ, ψ) that increase the rank most

- Repeat until rank is at least $2^{n\lambda}$

Random Basis

	4-bit s-boxes			8-bit s-boxes			
(λ, n)	(1,4)	(2,2)	(4,1)	(1,8)	(2,4)	(4,2)	(8,1)
$\lvert\mathcal{B}_1\rvert$	7	4	3	26	14	8	5
r	2	1	1	17	9	5	3

- Improvements w.r.t previous methods:
  - Boolean case : initial basis from 25 to 17

Decomposition of the S-box

- Sbox: $S\colon x \to (f_1(x), f_2(x), \ldots, f_m(x))$

- Apply m coordinate decompositions on the f_i's

- Basis update:
  - Start with a basis $\mathcal{B}_i$
  - At each step: $\mathcal{B}_{i+1} \leftarrow \mathcal{B}_i \cup \{g_i \cdot h_i\}_{i=0}^{t_i}$

Decomposition example

$$S : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$$

$$n = 2, m = 2, \lambda = 4 \to S(x, y) = (f_1(x, y), f_2(x, y))$$

$$\mathcal{B}_1 \quad | \quad \mathcal{B}_1 \times \mathcal{B}_1 \text{ spans } \mathcal{F}_{n,\lambda}$$

$$f_1(x) = \sum_{i=0}^{t_1} g_{1,i}(x) \cdot h_{1,i}(x)$$

$$\mathcal{B}_2 = \mathcal{B}_1 \cup \{g_{1,i} \cdot h_{1,i}\}_{i=0}^{t_1} \longrightarrow f_2(x) = \sum_{i=0}^{t_2} g_{2,i}(x) \cdot h_{2,i}(x)$$

Experimental Results and Implementations

Optimal Parameters

- Cost of the decomposition:

$$r + \sum_{i=1}^{m} t_i$$

with $t_i \geq \left\lceil \frac{2^{n\lambda} - 1}{\lambda |B_i| + 1} \right\rceil - 1$

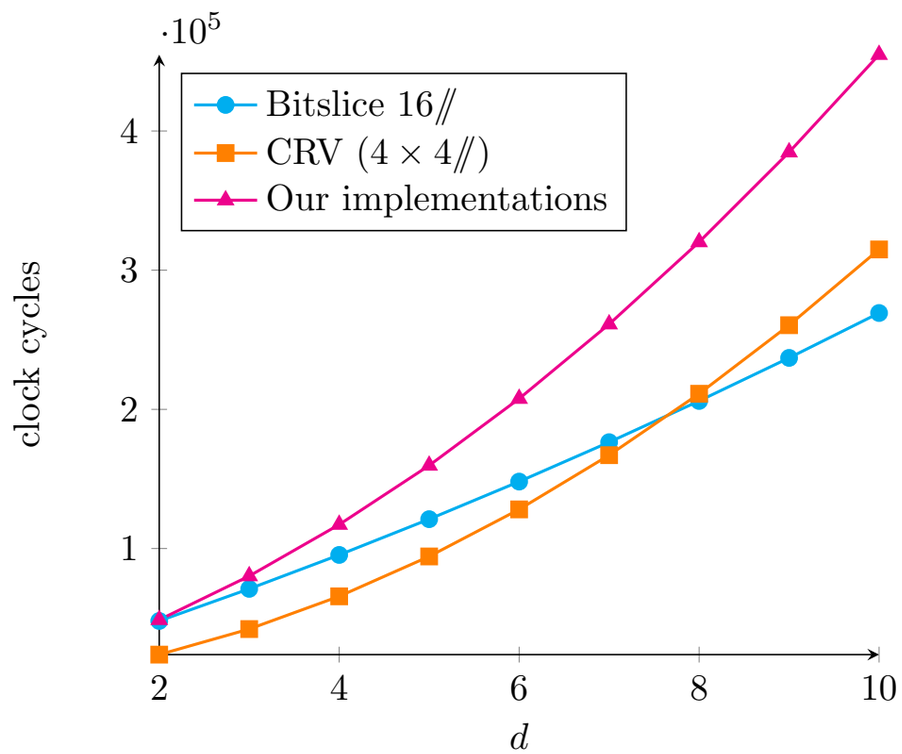
- Optimal parameters:

$$t_i = \left\lceil \frac{2^{n\lambda} - 1}{\lambda |B_i| + 1} \right\rceil - 1$$

Achievable Results for Median Cases

Optimal/Achievable	(λ, n)	$\lvert\mathcal{B}_1\rvert$	r	$t_1, t_2, \ldots, t_n$	C^*
4-bit s-boxes					
Optimal	(2,2)	5	2	1,1	4
Achievable	(2,2)	5	2	1,1	4
8-bit s-boxes					
Optimal	(2,4)	16	11	8,5,4,3	31
Achievable	(2,4)	16	11	9,6,5,3	34
Optimal	(4,2)	10	7	6,4	17
Achievable	(4,2)	10	7	7,4	18

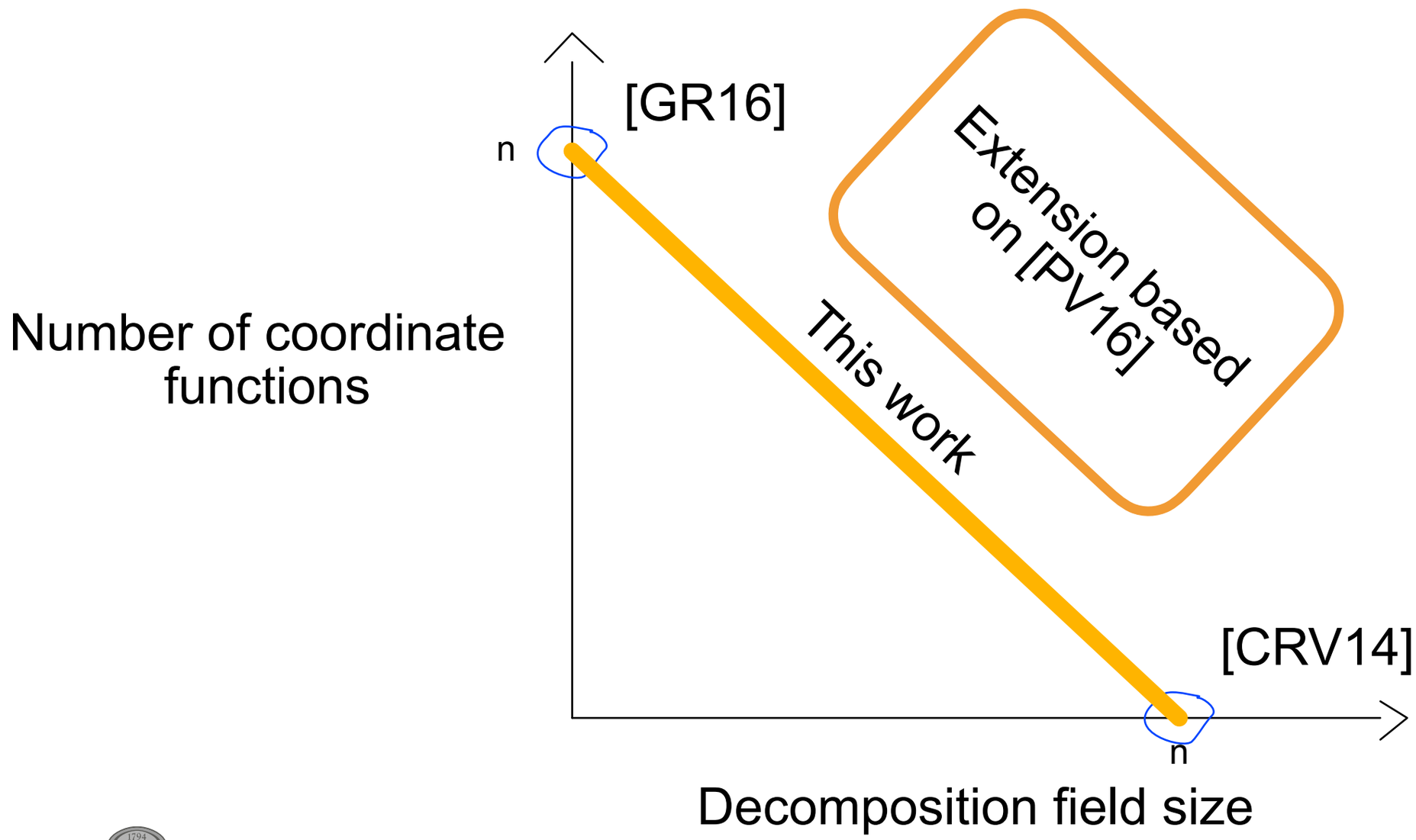
Implementation Results



	Code Size	RAM
CRV	27.5 KB	$80d$ B
Boolean Dec	4.6 KB	644d B
Our impl.	8.7 KB	92d B

Conclusion

- Generalized decomposition method well suited for any s-boxes or target architectures against side-channel attacks



Conclusion

- Case study on 32-bit ARM
  - Median case 8-bit S-box => 2 4-bit functions
  - Implementation comparison with state of the art
  - Memory trade-off
- Can suit low end device with smaller architecture
  - Parallelisation level decreased => poor bitslice performances
  - Few memory requirements