RanCompute: Computational Security in Embedded Devices via Random Input and Output Encodings

Georgia Tech | School of Electrical and Computer Engineering

KEVIN HUTTO^, SANTIAGO GRIJALVA*, AND VINCENT JOHN MOONEY III[&]
[&]ASSOCIATE *PROFESSOR, ^SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
[&]ADJUNCT ASSOCIATE PROFESSOR, SCHOOL OF COMPUTER SCIENCE
[^,*,&]INSTITUTE FOR INFORMATION SECURITY AND PRIVACY
GEORGIA TECH, ATLANTA, GA 30332-0250

presented at MECO’2022 and CPSIoT’2022, Budva, Montenegro
www.mecoconference.me

CPS&IoT

Outline

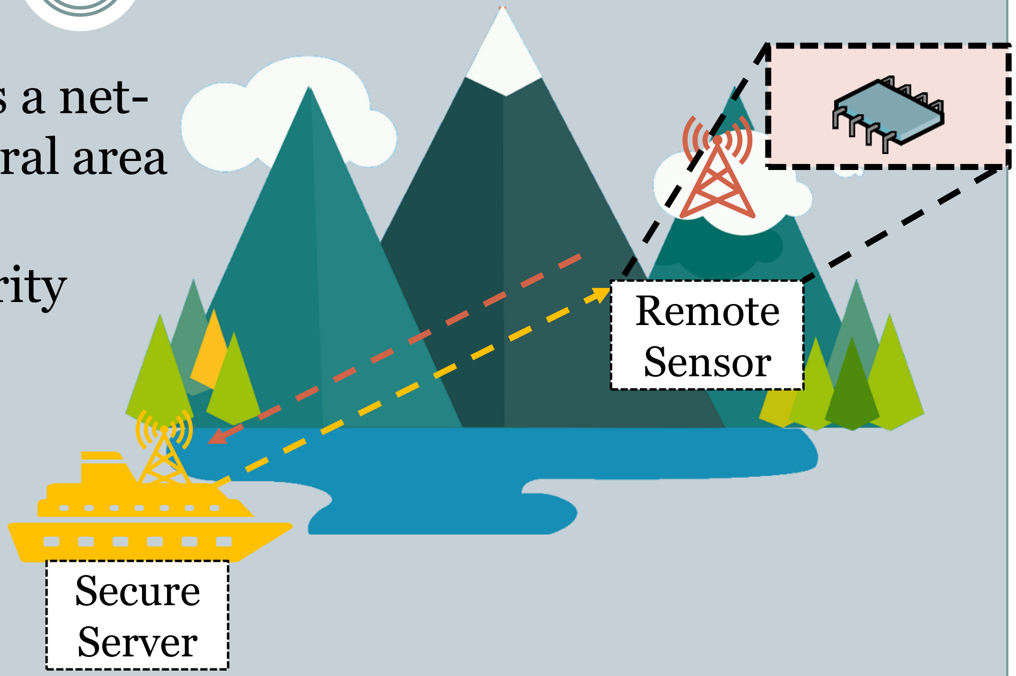
- Problem Definition
- Approach
- Example Computations
- Results
- Conclusions
- References

Outline

- Problem Definition
- Approach
- Example Computations
- Results
- Conclusions
- References

Problem Definition

- Consider a remote network, such as a network of sensors dispersed over a rural area
- The sensor is one of many in an environment with no physical security
- A capable adversary may capture one or more of the remote sensor(s) and attempt to reverse engineer the logic (including reconfigurable logic) and memory contents through state-of-the-art techniques [1][2]



- We consider a microchip architecture implementing one of two possible applications
- We aim to hide which of the two possible applications is being performed on the microchip given an adversary with complete white-box access at a specific time of capture

Outline

- Problem Definition
- Approach
- Example Computations
- Results
- Conclusions
- References

Approach

- One aspect which helps to hide the identity of a digital computation is to have truth tables with identical output frequencies
  - Output frequency – the number of times (multiplicity) a specific output appears in all possible outputs (including repeat values) resulting from a function $F_m()$ given a finite input set [3]
- We add a minimum number of encodings to ensure matching output frequencies of two target computations

A	B	$F_1(A,B) = A+B$
0	0	0 (S_0^1)
0	1	1 (S_1^1)
1	0	1 (S_1^1)
1	1	2 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

→

A	B	$F_1(A,B) = A+B$
0	0	00 (S_0^1)
0	1	01 (S_1^1)
1	0	01 (S_1^1)
1	1	10 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	00 (S_{0a}^2)
0	1	01 (S_{0b}^2)
1	0	01 (S_{0b}^2)
1	1	10 (S_1^2)

Approach (continued)

A	B	$F_1(A,B) = A + B$
0	0	0 (S_0^1)
0	1	1 (S_1^1)
1	0	1 (S_1^1)
1	1	2 (S_2^1)

A	B	$F_2(A,B) = A * B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

S_0^1 = Symbol representing zero for function F_1

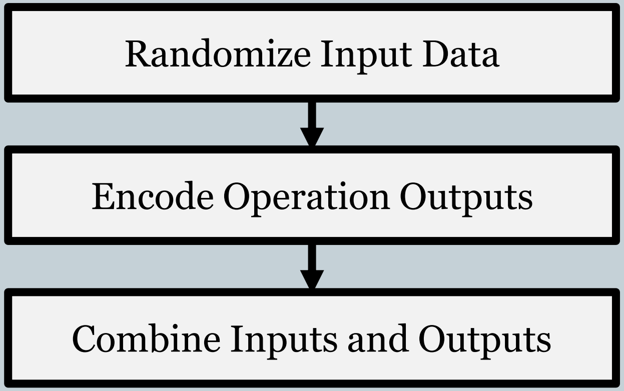
S_{0a}^2 = Symbol representing zero for function F_2, version *a*

S_{0b}^2 = Symbol representing zero for function F_2, version *b*

A	B	$F_1(A,B) = A + B$
0	0	00 (S_0^1)
0	1	01 (S_1^1)
1	0	01 (S_1^1)
1	1	10 (S_2^1)

A	B	$F_2(A,B) = A * B$
0	0	00 (S_{0a}^2)
0	1	01 (S_{0b}^2)
1	0	01 (S_{0b}^2)
1	1	10 (S_1^2)

Approach (continued 2)



A	B	$F_1(A,B) = A+B$
0	0	00 (S_0^1)
0	1	01 (S_1^1)
1	0	01 (S_1^1)
1	1	10 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

(a) Standard Unsigned Binary Operations

A	B	$F_1(A,B) = A+B$
0	0	S_0^1
0	1	S_1^1
1	0	S_1^1
1	1	S_2^1

A	B	$F_2(A,B) = A*B$
0	0	S_{0a}^2
0	1	S_{0b}^2
1	0	S_{0b}^2
1	1	S_1^2

(b) Outputs Assigned Symbols to Match Frequencies

A	B	$F_1(A,B) = A+B$
0	0	10 (S_0^1)
0	1	11 (S_1^1)
1	0	11 (S_1^1)
1	1	00 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	10 (S_{0a}^2)
0	1	11 (S_{0b}^2)
1	0	11 (S_{0b}^2)
1	1	00 (S_1^2)

(c) Output Symbols Replaced With New Bit-Representations

Approach (continued 3)

(b) Encode Operation Outputs

A	B	$F_1(A,B) = A+B$
0	0	00 (S_0^1)
0	1	01 (S_1^1)
1	0	01 (S_1^1)
1	1	10 (S_2^1)

→

A	B	$F_1(A,B) = A+B$
0	0	S_0^1
0	1	S_1^1
1	0	S_1^1
1	1	S_2^1

→

A	B	$F_1(A,B) = A+B$
0	0	10 (S_0^1)
0	1	11 (S_1^1)
1	0	11 (S_1^1)
1	1	00 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

→

A	B	$F_2(A,B) = A*B$
0	0	S_{0a}^2
0	1	S_{0b}^2
1	0	S_{0b}^2
1	1	S_1^2

→

A	B	$F_2(A,B) = A*B$
0	0	10 (S_{0a}^2)
0	1	11 (S_{0b}^2)
1	0	11 (S_{0b}^2)
1	1	00 (S_1^2)

(a) Standard Unsigned Binary Operations

(b) Outputs Assigned Symbols to Match Frequencies

(c) Output Symbols Replaced With New Bit-Representations

Approach (continued 4)

(a) Randomize Input Data

A	B	$F_1(A,B) = A+B$
0	0	0 (S_0^1)
0	1	1 (S_1^1)
1	0	1 (S_1^1)
1	1	2 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

A	B	$F_1(A,B) = A+B$
1	0	00 (S_0^1)
1	1	01 (S_1^1)
0	0	01 (S_1^1)
0	1	10 (S_2^1)

A	B	$F_2(A,B) = A*B$
1	0	00 (S_{0a}^2)
1	1	01 (S_{0b}^2)
0	0	01 (S_{0b}^2)
0	1	10 (S_1^2)

(b) Encode Operation Outputs

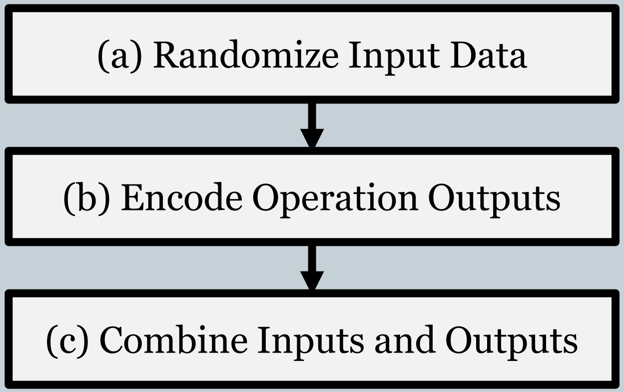
A	B	$F_1(A,B) = A+B$
0	0	0 (S_0^1)
0	1	1 (S_1^1)
1	0	1 (S_1^1)
1	1	2 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

A	B	$F_1(A,B) = A+B$
0	0	10 (S_0^1)
0	1	11 (S_1^1)
1	0	11 (S_1^1)
1	1	00 (S_2^1)

A	B	$F_2(A,B) = A*B$
0	0	10 (S_{0a}^2)
0	1	11 (S_{0b}^2)
1	0	11 (S_{0b}^2)
1	1	00 (S_1^2)

Approach (continued 5)



A	B	$F_1(A,B) = A + B$
0	0	0 (S_0^1)
0	1	1 (S_1^1)
1	0	1 (S_1^1)
1	1	2 (S_2^1)

A	B	$F_2(A,B) = A * B$
0	0	0 (S_0^2)
0	1	0 (S_0^2)
1	0	0 (S_0^2)
1	1	1 (S_1^2)

A	B	$F_1(A,B) = A + B$
1	0	10 (S_0^1)
1	1	11 (S_1^1)
0	0	11 (S_1^1)
0	1	00 (S_2^1)

A	B	$F_2(A,B) = A * B$
1	0	10 (S_{0a}^2)
1	1	11 (S_{0b}^2)
0	0	11 (S_{0b}^2)
0	1	00 (S_1^2)

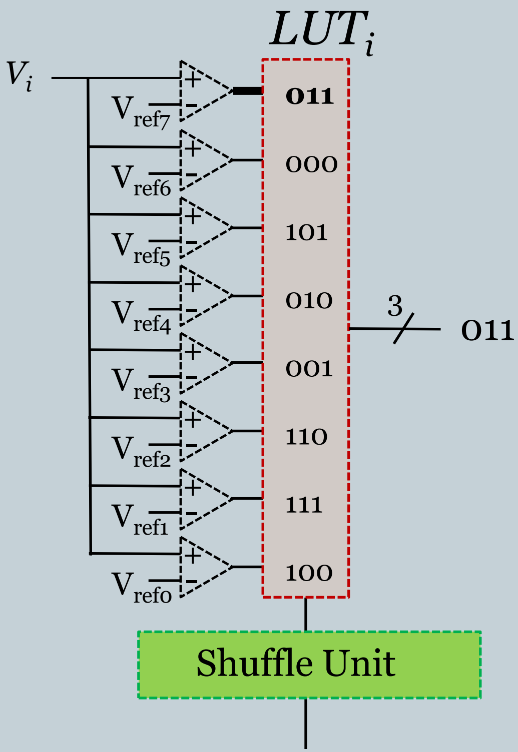
Approach (continued 5)

- Look-Up Table (LUT) result for each of the operations with randomized inputs and randomized outputs equalized for frequency

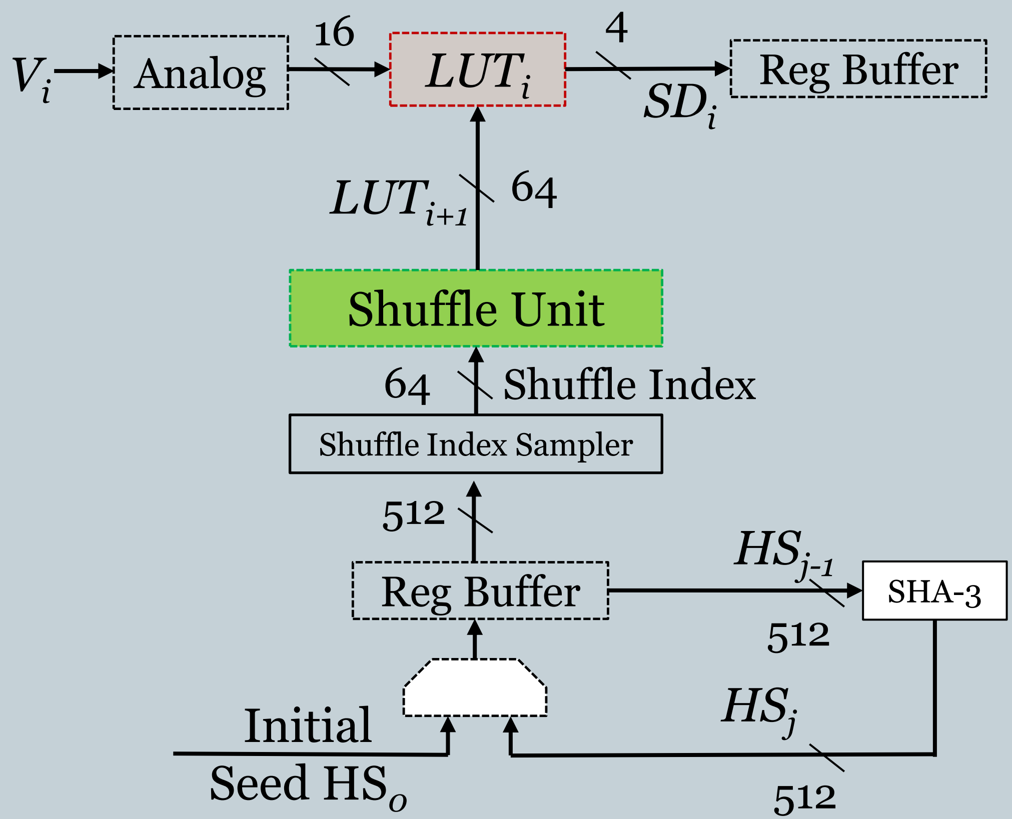
A	B	$F_1(A, B) = A + B$
0	0	11 (S_1^1)
0	1	00 (S_2^1)
1	0	10 (S_0^1)
1	1	11 (S_1^1)

A	B	$F_2(A, B) = A * B$
0	0	11 (S_{0b}^2)
0	1	00 (S_1^2)
1	0	10 (S_{0a}^2)
1	1	11 (S_{0b}^2)

Method of Input Encoding

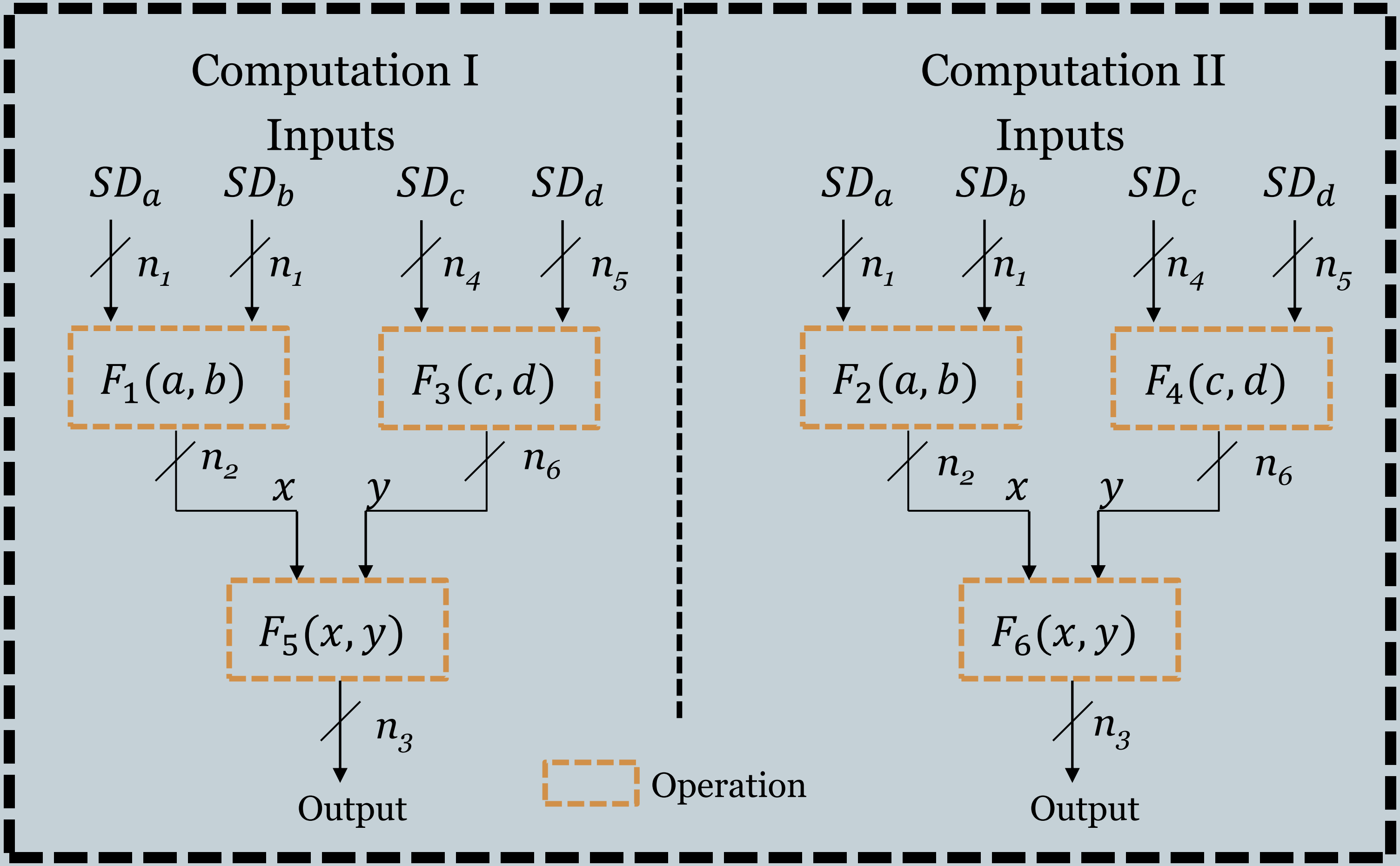


Input Encoding [4]



Randomized Data Circuit

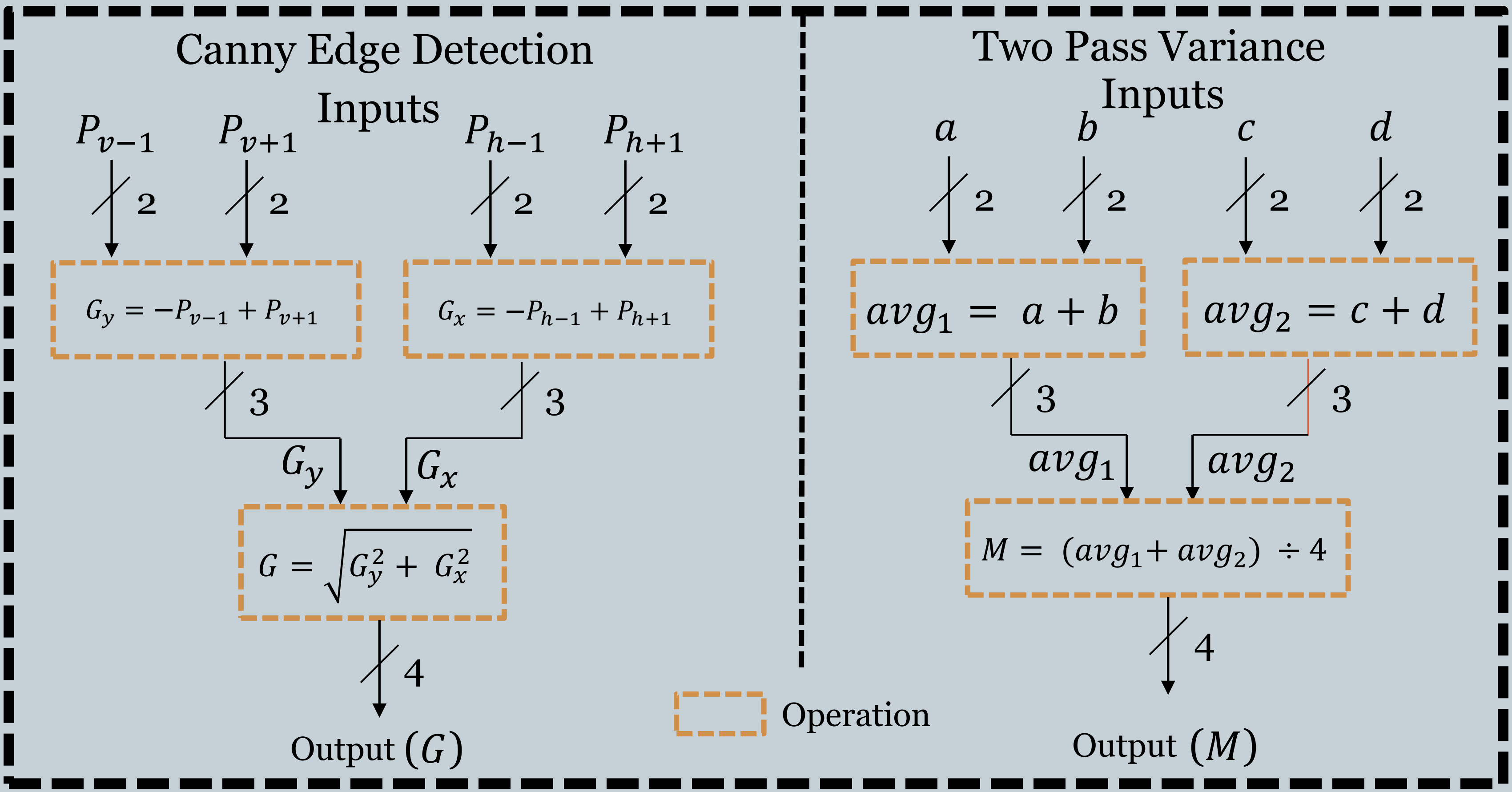
Performing Computations



Outline

- **Problem definition**
- **Approach**
- **Example Computations**
- **Results**
- **Conclusions**
- **References**

Example Computations



Example Computations

Canny Edge Detection (Two 3-bit Input Operation)		
(a) Unencoded Output (G)	(b) f_i^1	(c) $f_i^1 \cup f_{ij}^1$
0	4	2, 2
1	8	8
1.5	4	4
2	16	1, 2, 6, 7
3	20	1, 3, 3, 6, 7
3.5	8	8
4	4	4

Two-Pass Variance (Two 3-bit Input Operation)		
(a) Unencoded Output (M)	(b) f_i^2	(c) $f_i^2 \cup f_{ij}^2$
0	1	1
0.25	2	2
0.5	3	3
0.75	6	6
1	7	7
1.25	8	8
1.5	10	2, 4, 4
1.75	8	8
2	7	7
2.25	6	6
2.5	3	3
2.75	2	2
3	1	1

- Output frequencies for the operations with two 3-bit Inputs for Canny Edge Detection and Two-Pass Variance
- Resulting output frequencies: (1,1,2,2,2,3,3,4,4,6,6,7,7,8,8)
- Note that the total number of outputs is 64

Outline

- Problem definition
- Approach
- Example Computations
- Results
- Conclusions
- References

Results

- For each developed RanCompute application, we tested 10,000 iterations
- All simulations showed expected functionality, with the output of each RanCompute application equaling the expected encoding

FGPA UTILIZATION OF RANCOMPUTE

Application	Slice LUTs	Slice Regs	Bonded IOB	Max Freq.
(a)	2	3	10	450 MHz
(b)	6	5	14	450 MHz
(c)	5	1	18	380 MHz
(d)	8	4	15	450 MHz
(e)	13	1	22	380 MHz
(f)	120	7	26	380 MHz

(a) Two 2-bit input logic functions (add, multiply)
(b) Two 3-bit input logic functions (add, multiply)
(c) Two 4-bit input logic functions (add, multiply)
(d) 2-bit Edge Detection / Variance
(e) 3-bit Edge Detection / Variance
(f) 4-bit Edge Detection / Variance

Outline

- Problem definition
- Approach
- Example Computations
- Results
- Conclusions
- References

Conclusions

- In this paper we introduced a novel methodology to perform computations which are indistinguishable from each other from the point of view of an adversary with reverse engineering capabilities.
- We believe this is an important first step in the development of a framework for a general purpose method to perform indistinguishable computations on a microchip.

Outline

- Problem definition
- Approach
- Example Computations
- Results
- Conclusions
- References

References

[1] “Technical Capabilities,” 2021. [Online.] Available: https://www.techinsights.com/technical-capabilities

[2] A. Duncan et al., “FPGA Bitstream Security: A Day in the Life,” *2019 IEEE International Test Conference* (ITC ’19), 2019, pp. 1-10.

[3] W. D. Blizard et al., “Multiset Theory,” Notre Dame Journal of Formal Logic,” Vol. 30, No. 1, pp. 36-66, 1989.

[4] K. Hutto and V. Mooney, “Sensing with Random Encoding for Enhanced Security in Embedded Systems,” *2021 10th Mediterranean Conference on Embedded Computing* (MECO ’21), Vol. 10, pp. 809-814, 7 June 2021.

[5]-[16] *Please see the paper for these references.*

THANK YOU

Q&A

Kevin Hutto
khutto30@gatech.edu

Santiago Grijalva
sgrijalva@ece.gatech.edu

Vincent Mooney
mooney@ece.gatech.edu