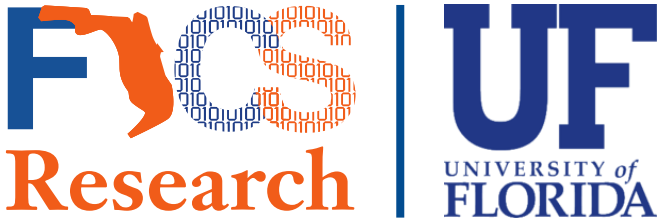
HOST Microelectronics Security Challenge: IP Security Track

Team: Gator Hardware Oriented Security Team (GHOST)

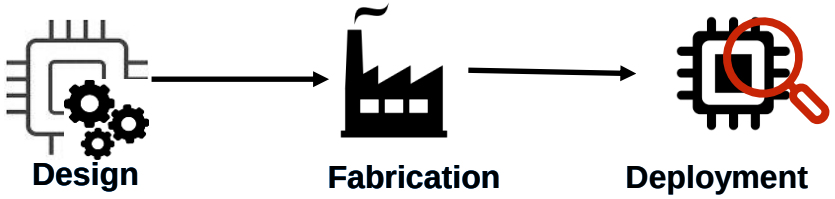
Advised by – Dr. Fahim Rahman

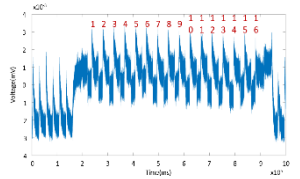
Members: Md. Kawser Bepary, Dipayan Saha, Amit Mazumder Shuvo, Pantha Pratim Sarker, Mashahedur Rahman, Shuvagata Saha, Hasan Al-Shaikh.



Secrets (not so) Hidden in Plain Sight

- **Motivation:**
  - Side channel leakage analysis
    - Unintentionally expose internal secrets due to weakness in hardware implementation.
  - Attack types: Power, electromagnetic, fault, timing, microarchitectural, acoustic and more.
  - **Assumes minimal knowledge about the victim system, cheap to implement and extremely effective!!**





Theoretically sound

Fabricated according to specification

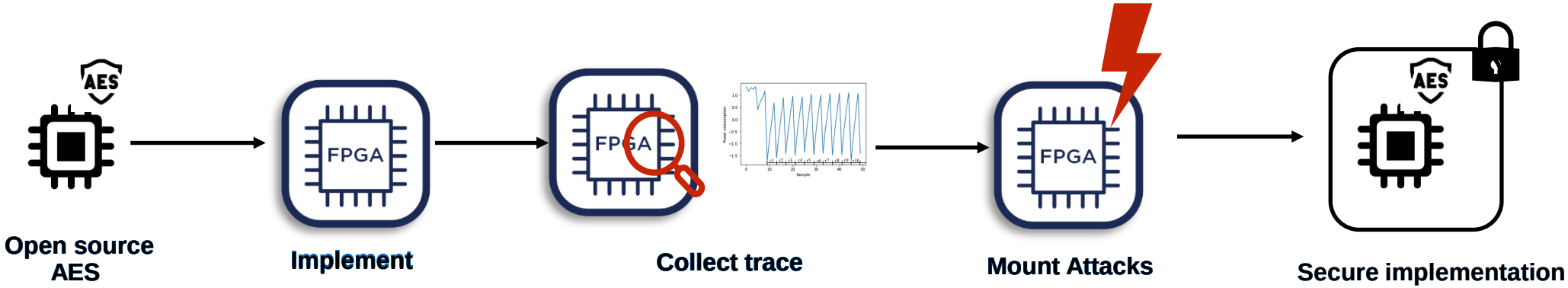
Still vulnerable!!!

June 2022

March 2022

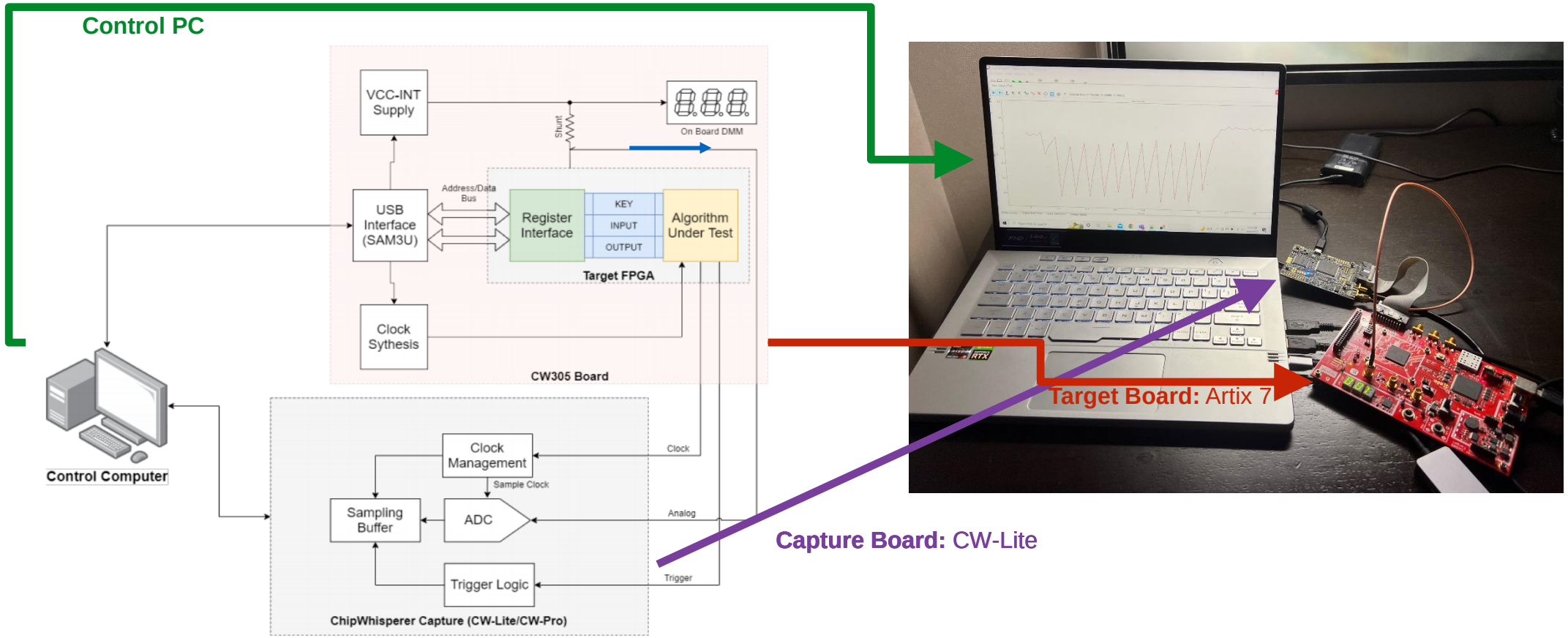
November 2020

Challenge Overview



- **Setup:**
  - An open-source implementation of 128-bit AES circuit
- **Tasks:**
  1. Implementation of the AES core on an FPGA
  2. Collect power traces
  3. Mount power side-channel and fault injection attacks.
  4. Find vulnerabilities and leakage information
  5. Propose and implement countermeasures.

Experimental Setup (ChipWhisperer)



- **Equipment:**
  - Laptop/PC (installed Chipwhisperer v4.0.1, Xilinx Vivado)
  - ChipWhisperer CW305 Artix-7 Target board
  - ChipWhisperer-Lite Capture board

Power Side-Channel Attacks

Simple Power Analysis (SPA)

- Simple power analysis (SPA) involves visual inspection of the power traces
- SPA is utilized to extract the feature points from the power traces
- Each power trace shows ten identifiable cycles of power consumption
- From visualization of a single power trace, each round can be identified along the time axis
- Such round information can be very helpful in the next DPA/CPA attack

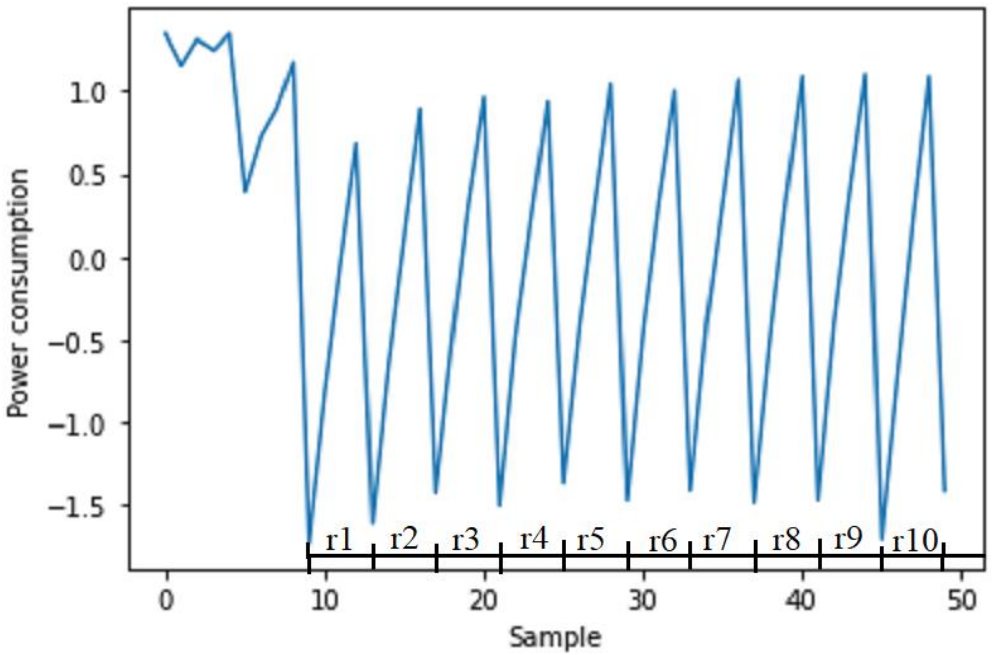


Figure: SPA on collected trace

Differential Power Analysis (DPA)

- Exploits data dependency of the power consumption of cryptographic devices using a large number of power traces

- DPA attack strategy:

  Step 1: Choosing intermediate result

  Step 2: Measurement of power consumption

  Step 3: Calculation of hypothetical intermediate values

  Step 4: Calculation of hypothetical power consumption using power model (HD/HW/MSB/LSB)

  Step 5: Statistical analysis

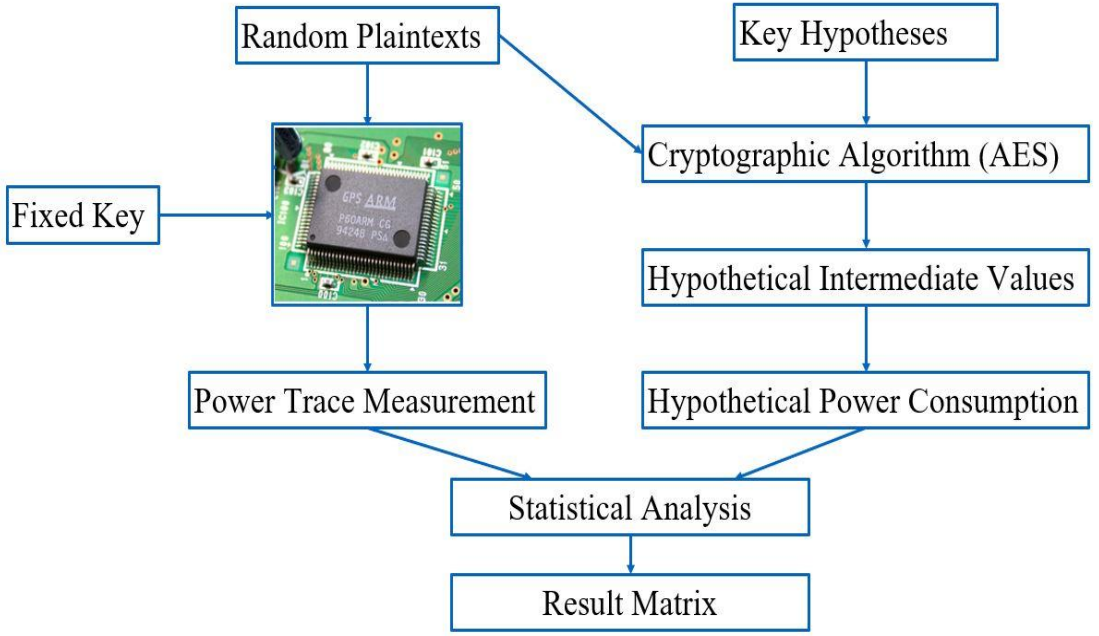


Figure: DPA attack methodology

DPA cont'd

- Depending on the statistical analysis different DPA attacks can be performed.
- **Difference of means:**

$$R = M1 - M0$$

- **Distance of means:**

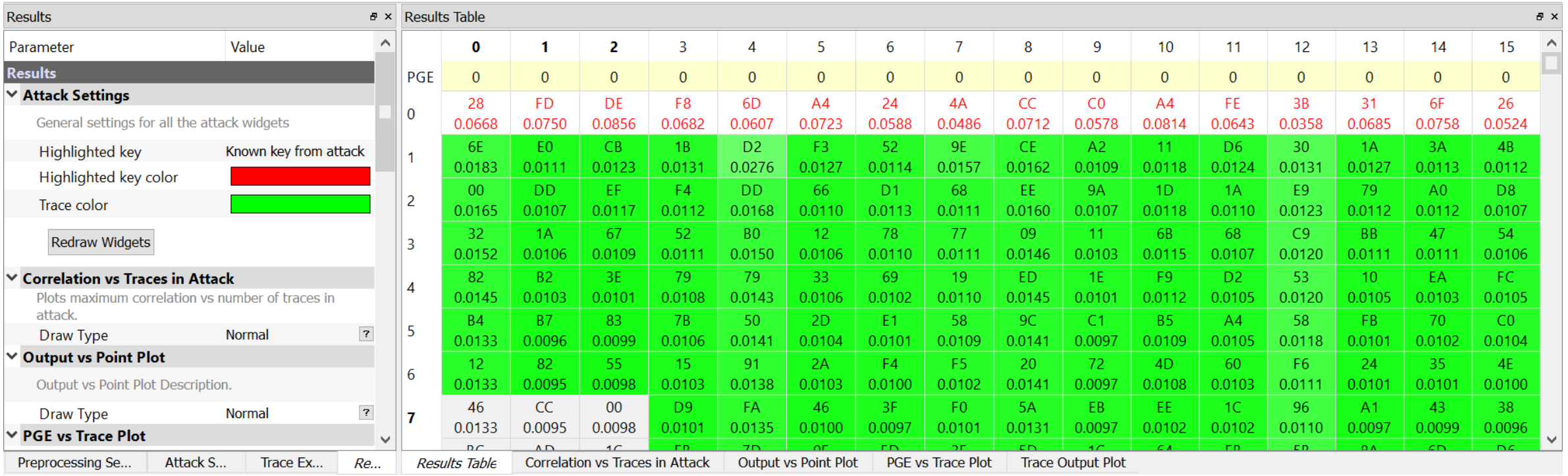
$$r_{ij} = \frac{m_{1ij} - m_{0ij}}{s_{ij}}$$

- **Correlation coefficient / Correlation Power Analysis (CPA):**

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i})(t_{d,j} - \overline{t_j})}{\sqrt{(\sum_{d=1}^{D}(h_{d,i} - \overline{h_i}))^2 \, (\sum_{d=1}^{D}(t_{d,j} - \overline{t_j}))^2}}$$

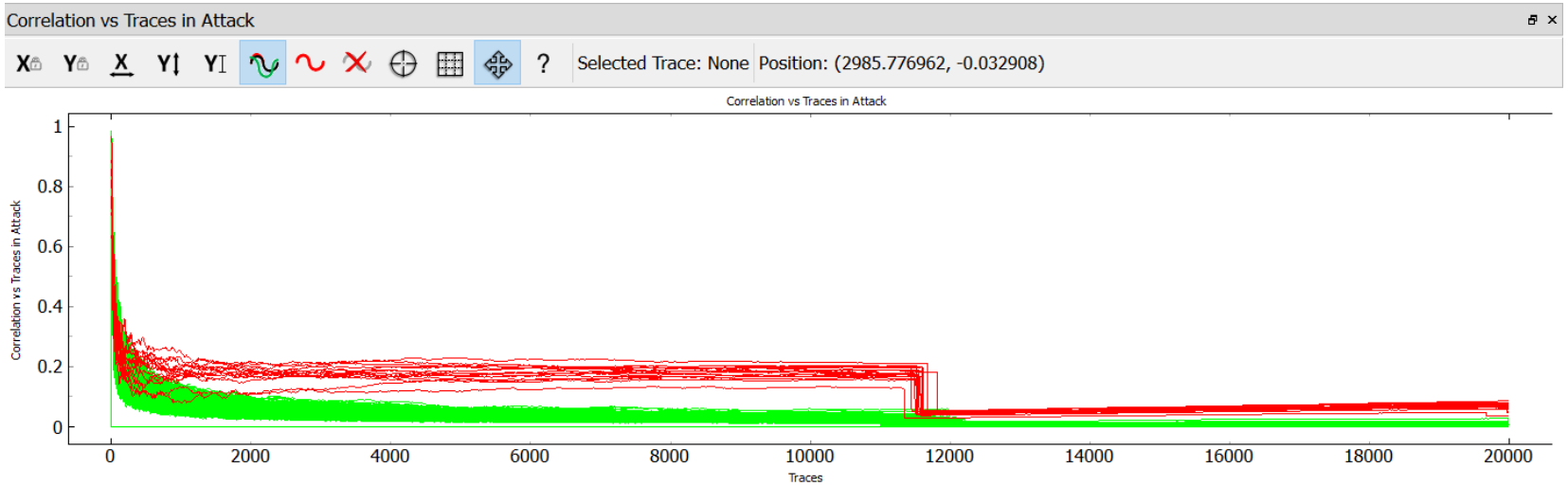
Result: CPA with hamming distance model

- CPA with HD model is performed on the last round
- Correct keys have been recovered for all sub-bytes



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	28 0.0668	FD 0.0750	DE 0.0856	F8 0.0682	6D 0.0607	A4 0.0723	24 0.0588	4A 0.0486	CC 0.0712	C0 0.0578	A4 0.0814	FE 0.0643	3B 0.0358	31 0.0685	6F 0.0758	26 0.0524
1	6E 0.0183	E0 0.0111	CB 0.0123	1B 0.0131	D2 0.0276	F3 0.0127	52 0.0114	9E 0.0157	CE 0.0162	A2 0.0109	11 0.0118	D6 0.0124	30 0.0131	1A 0.0127	3A 0.0113	4B 0.0112
2	00 0.0165	DD 0.0107	EF 0.0117	F4 0.0112	DD 0.0168	66 0.0110	D1 0.0113	68 0.0111	EE 0.0160	9A 0.0107	1D 0.0118	1A 0.0110	E9 0.0123	79 0.0112	A0 0.0112	D8 0.0107
3	32 0.0152	1A 0.0106	67 0.0109	52 0.0111	B0 0.0150	12 0.0106	78 0.0110	77 0.0111	09 0.0146	11 0.0103	6B 0.0115	68 0.0107	C9 0.0120	BB 0.0111	47 0.0111	54 0.0106
4	82 0.0145	B2 0.0103	3E 0.0101	79 0.0108	79 0.0143	33 0.0106	69 0.0102	19 0.0110	ED 0.0145	1E 0.0101	F9 0.0112	D2 0.0105	53 0.0120	10 0.0105	EA 0.0103	FC 0.0105
5	B4 0.0133	B7 0.0096	83 0.0099	7B 0.0106	50 0.0141	2D 0.0104	E1 0.0101	58 0.0109	9C 0.0141	C1 0.0097	B5 0.0109	A4 0.0105	58 0.0118	FB 0.0101	70 0.0102	C0 0.0104
6	12 0.0133	82 0.0095	55 0.0098	15 0.0103	91 0.0138	2A 0.0103	F4 0.0100	F5 0.0102	20 0.0141	72 0.0097	4D 0.0108	60 0.0103	F6 0.0111	24 0.0101	35 0.0101	4E 0.0100
7	46 0.0133	CC 0.0095	00 0.0098	D9 0.0101	FA 0.0135	46 0.0100	3F 0.0097	F0 0.0101	5A 0.0131	EB 0.0097	EE 0.0102	1C 0.0102	96 0.0110	A1 0.0097	43 0.0099	38 0.0096

Result: CPA with HD model cont'd



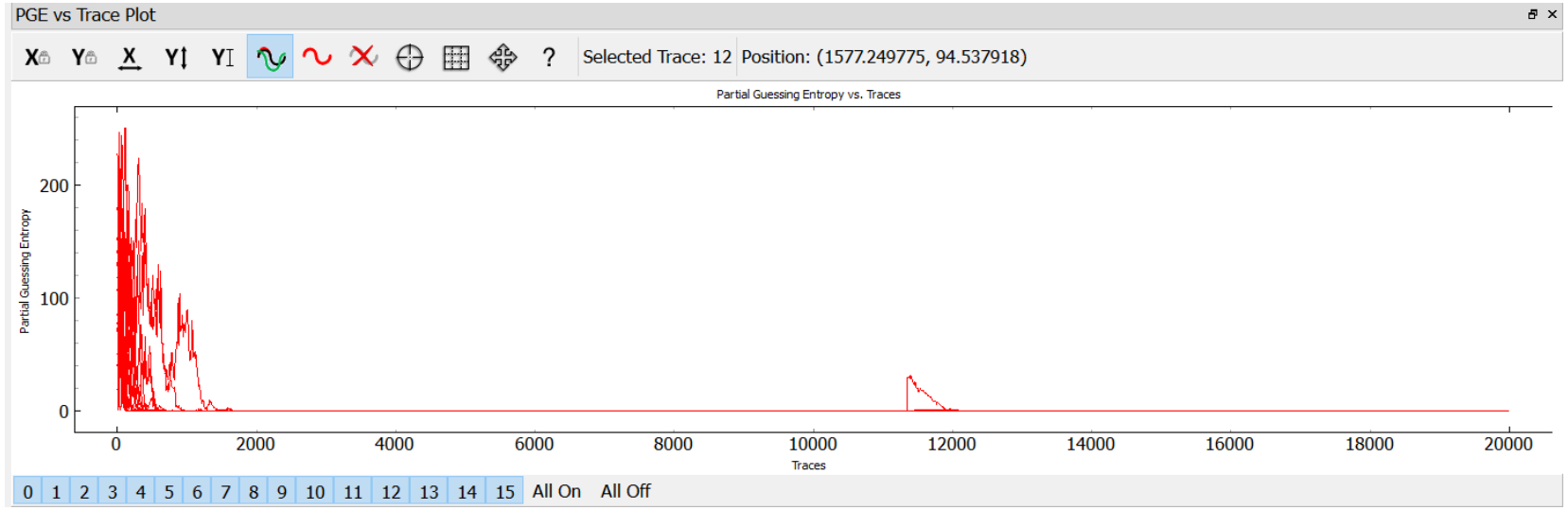
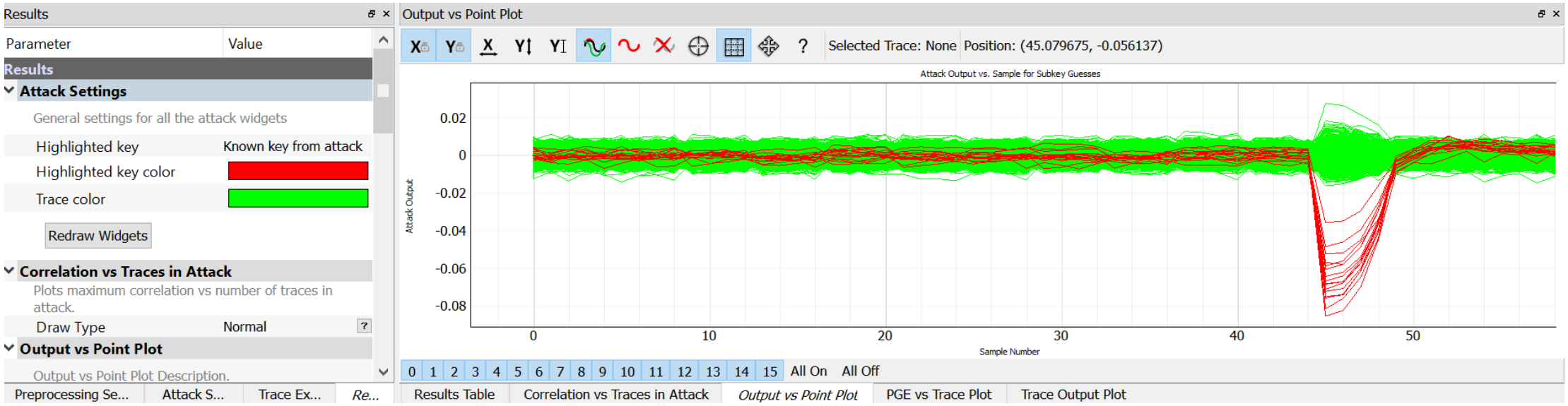


Figure: (top) correlation vs. number of traces; (bottom) partial guessing entropy vs. number of traces for last round CPA with HD model

The figures suggest that minimum of 12k traces are required to recover the full key

Result: CPA with HD model cont'd

Figure shows that leakage is found at last round. Such leakage is expected since CPA with HD model was performed on the last round



Result: CPA with HW model

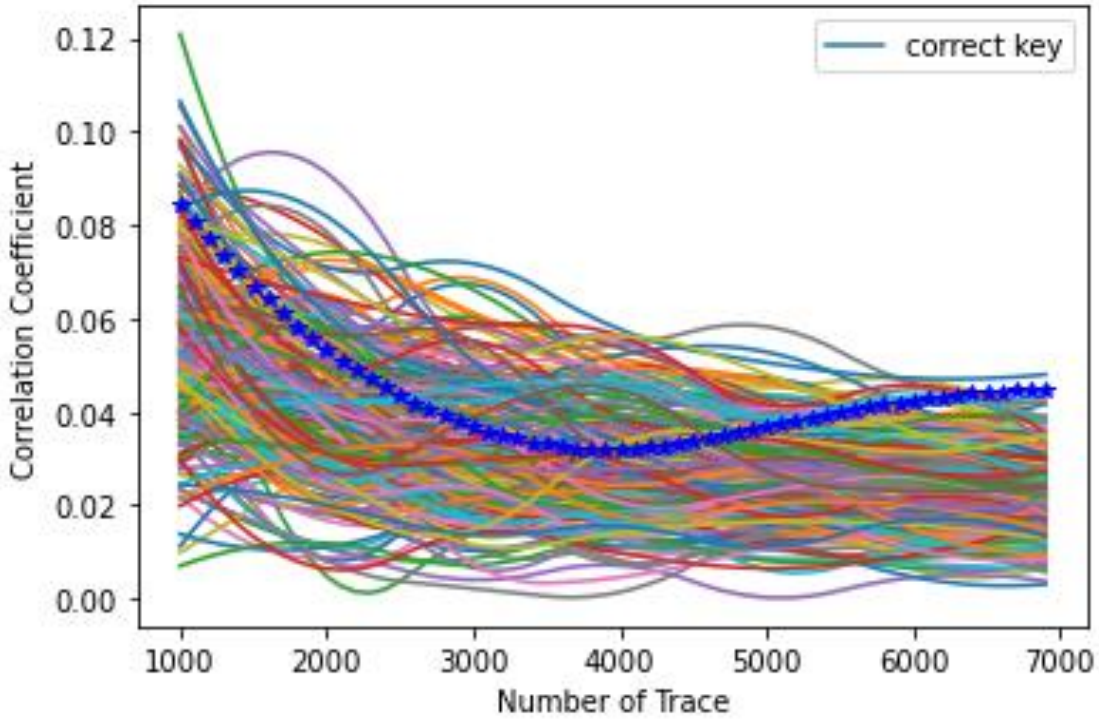


Figure: Correlation coefficient for variation of number of traces for CPA with HW model

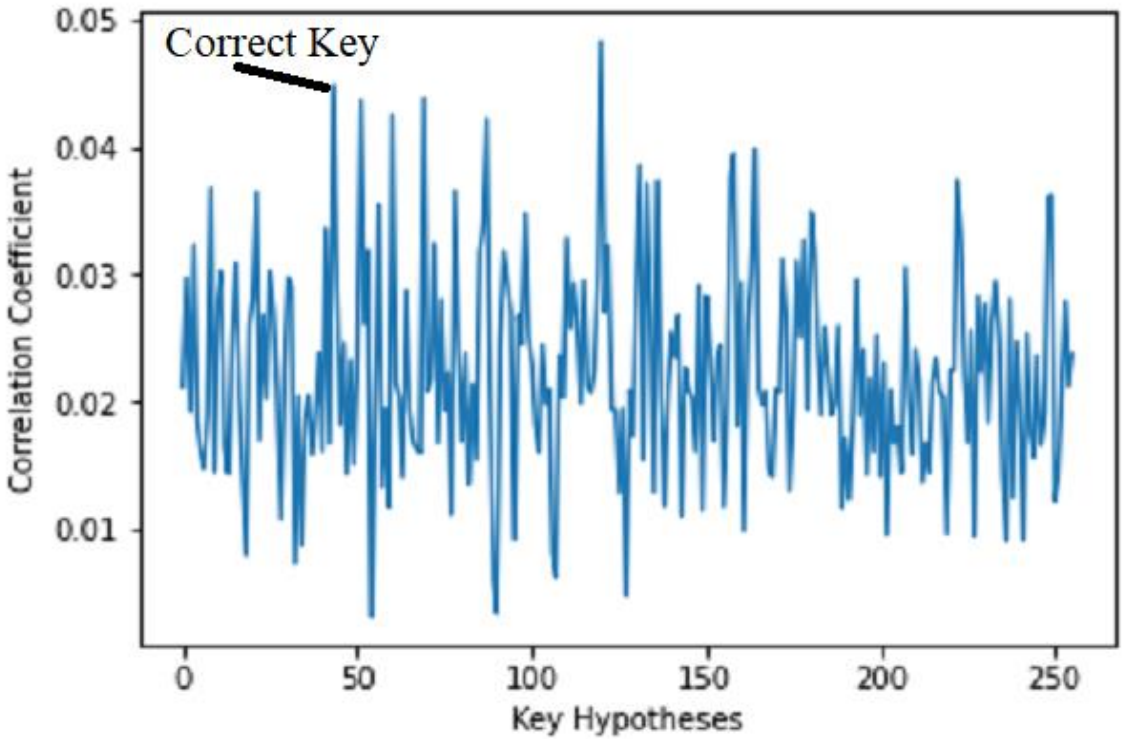


Figure: Correlation coefficient for different key hypotheses in case of CPA with HW model using 7000 power traces

Result: DPA with distance of means

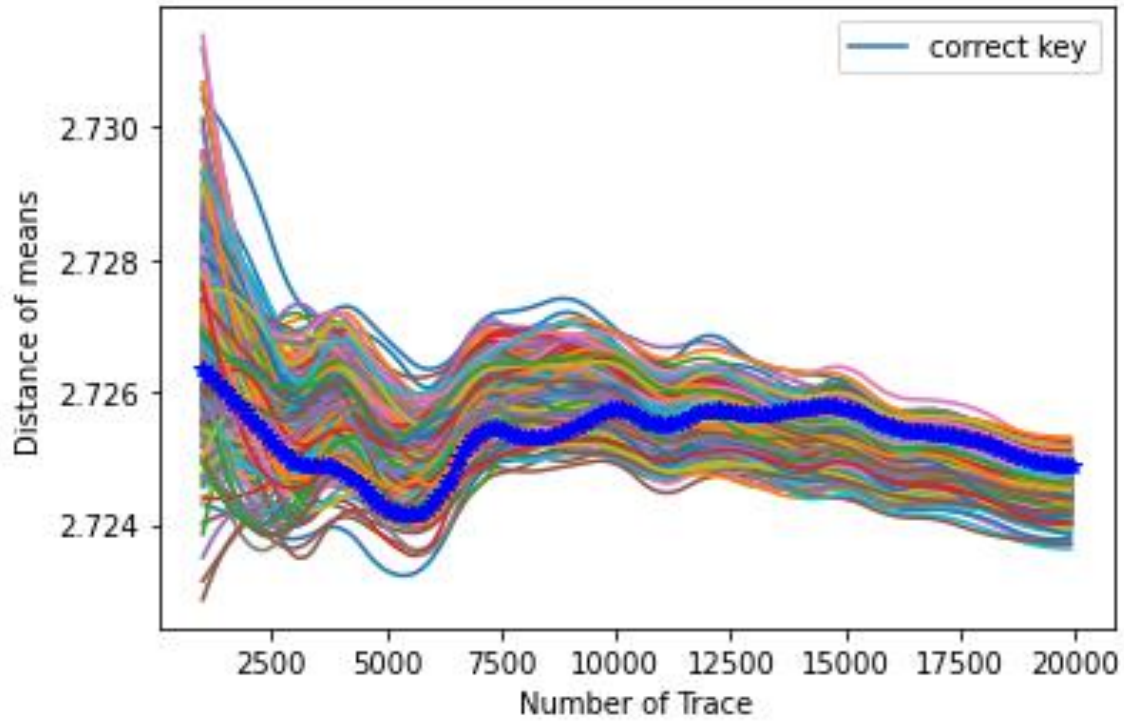


Figure: Correlation coefficient for variation of number of traces for DPA with distance of means

ML-based Side-Channel Attack

- Signal decomposition is integrated with ML-based side-channel attack methodology for unique and unseen feature extraction
- MLP networks are used with model ensembling

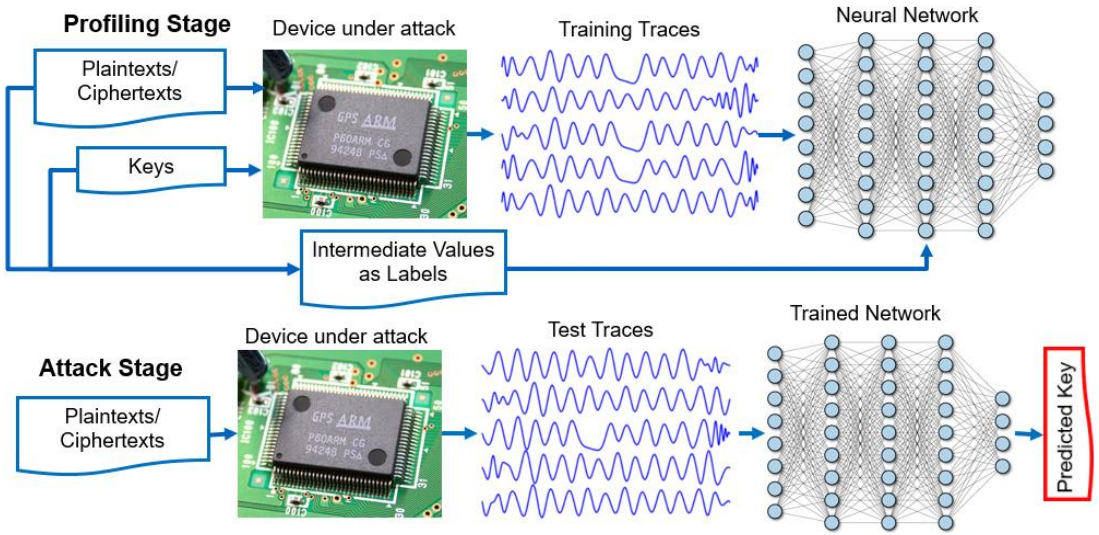


Figure: Overview of ML-based side-channel attack

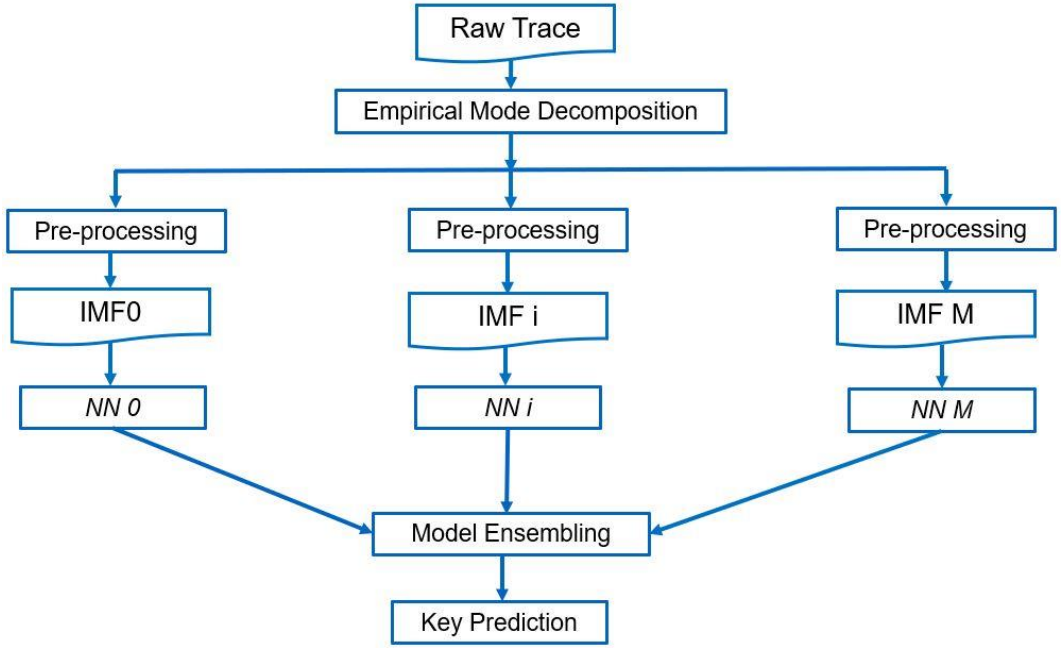


Figure: Overview of our ML-based attack

Result: ML-based Side-Channel Attack

- Dataset size: 40k
- Only around 1400 traces are required to extract key using ML-based approach
- Requires only 300k parameters

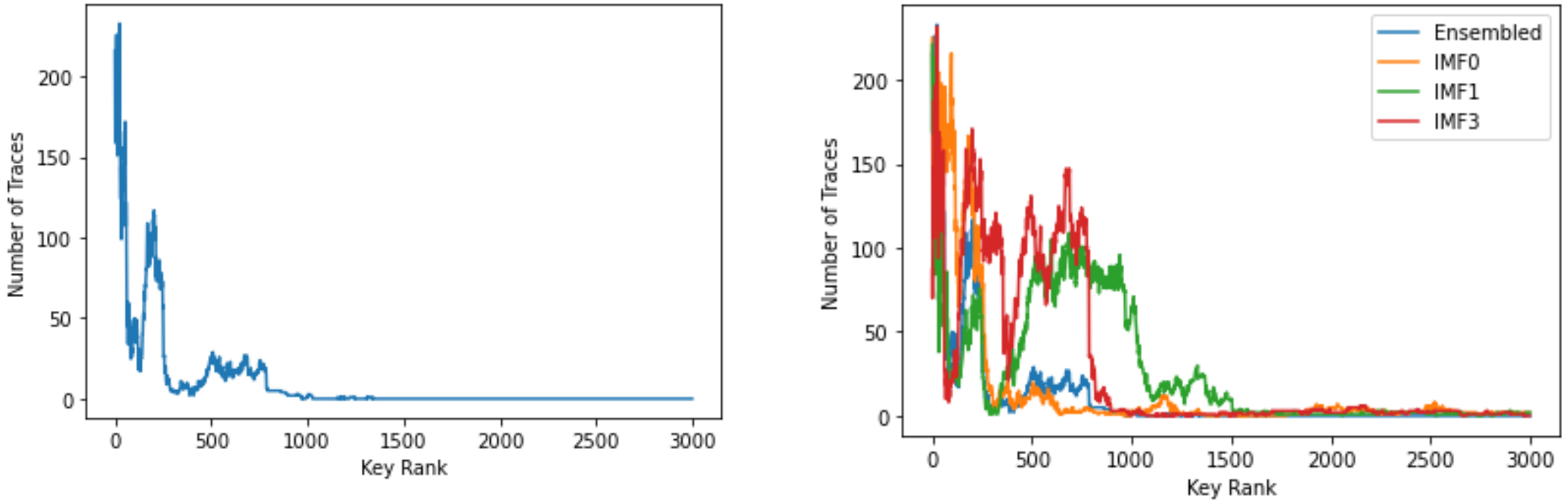


Figure: Key rank vs number of traces for ML-based side-channel attack

Power Side-Channel Countermeasures

Side-Channel Countermeasures

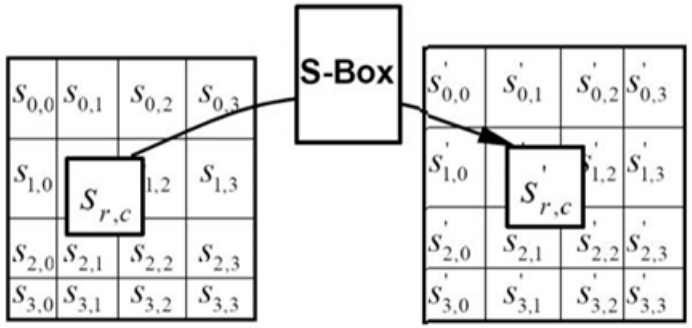
- Masking
  - Arguably most popular countermeasure against side-channel analysis.
  - Randomizes intermediate values and render power consumption independent of the associated data.
  - Incurs high overhead.
- Hiding
  - Increases the signal-to-noise (SNR) ratio.
  - Approaches-
    - Introduction of a noise generator in the circuit.
    - Implementation of dual-rail precharge (DRP) logic such as SABL and WDDL.
  - Incurs high overhead.

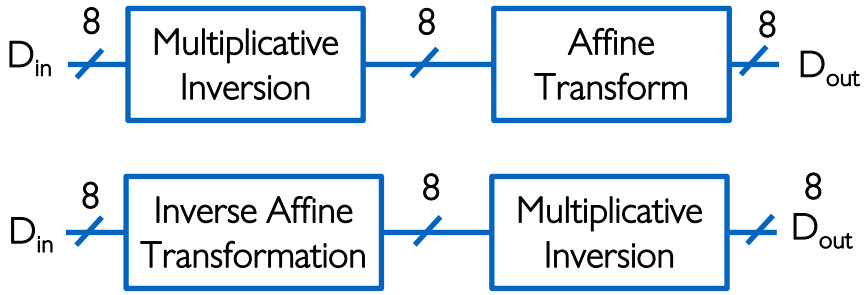
Side-Channel Countermeasures cont'd

- t-private Circuit
  - A provable theoretical framework for resisting side-channel analysis.
  - Gates are transformed in such a way that at least t+1 nodes need to be probed to retrieve a single bit of information.
- Constant Random Matrix Multiplication
  - The plaintext and key are multiplied with a special matrix **R** before starting of the rounds.
  - After the encryption, the ciphertext is again multiplied with **R** to get the correct ciphertext.
- Physical Design Level Countermeasure
  - Used in DRP logic.
  - Wire lengths for paired logic are made equal so that capacitances are matched, and power consumptions are the same regardless of the logic operation.
- Countermeasure by Modifying Clock Pulses
  - Skipping Clock Pulses, Randomizing Clock Freq., Multiple Clock Domain, etc.

Implemented Countermeasure: Approach 1 (Galois Field Transform)

- Sbox LUT block is the most vulnerable part of AES implementation
- Galois Field Transformation of Sbox LUT
- Isomorphic mapping can be employed to convert GF(2^8) to GF((((2^2)^2)^2), so that multiplicative inverse can be easily obtained



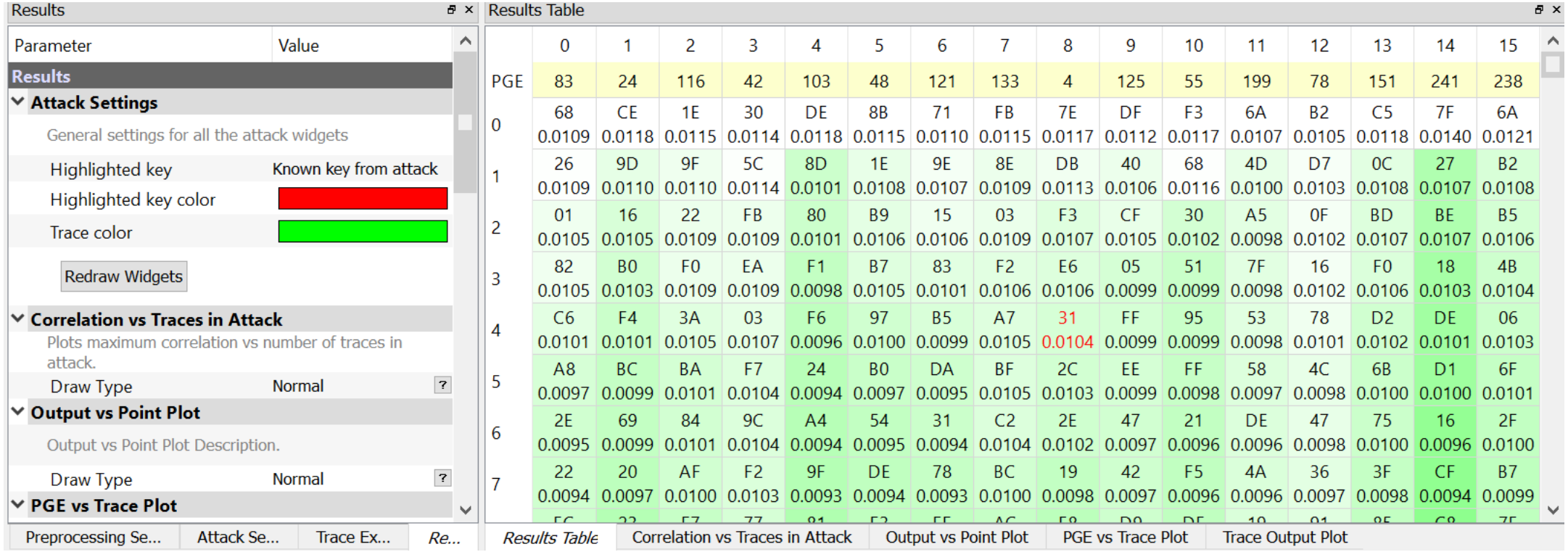


SubByte and Inverse SubByte Transformation in Galois Field

GF (2^2) →GF (2): $x^2 + x + 1$
GF ((2^2)^2) →GF (2^2): $x^2 + x + \phi$
GF ((2^2)^2)^2) →GF ((2^2)^2): $x^2 + x + \lambda$
Where ϕ = {10}2 and λ = {1100}2.

Countermeasure Result (GF): CPA with HD model

- CPA with HD model is performed on the last round
- None of the sub-keys have been recovered for all sub-bytes



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	83	24	116	42	103	48	121	133	4	125	55	199	78	151	241	238
0	68 0.0109	CE 0.0118	1E 0.0115	30 0.0114	DE 0.0118	8B 0.0115	71 0.0110	FB 0.0115	7E 0.0117	DF 0.0112	F3 0.0117	6A 0.0107	B2 0.0105	C5 0.0118	7F 0.0140	6A 0.0121
1	26 0.0109	9D 0.0110	9F 0.0110	5C 0.0114	8D 0.0101	1E 0.0108	9E 0.0107	8E 0.0109	DB 0.0113	40 0.0106	68 0.0116	4D 0.0100	D7 0.0103	0C 0.0108	27 0.0107	B2 0.0108
2	01 0.0105	16 0.0105	22 0.0109	FB 0.0109	80 0.0101	B9 0.0106	15 0.0106	03 0.0109	F3 0.0107	CF 0.0105	30 0.0102	A5 0.0098	0F 0.0102	BD 0.0107	BE 0.0107	B5 0.0106
3	82 0.0105	B0 0.0103	F0 0.0109	EA 0.0109	F1 0.0098	B7 0.0105	83 0.0101	F2 0.0106	E6 0.0106	05 0.0099	51 0.0099	7F 0.0098	16 0.0102	F0 0.0106	18 0.0103	4B 0.0104
4	C6 0.0101	F4 0.0101	3A 0.0105	03 0.0107	F6 0.0096	97 0.0100	B5 0.0099	A7 0.0105	31 0.0104	FF 0.0099	95 0.0099	53 0.0098	78 0.0101	D2 0.0102	DE 0.0101	06 0.0103
5	A8 0.0097	BC 0.0099	BA 0.0101	F7 0.0104	24 0.0094	B0 0.0097	DA 0.0095	BF 0.0105	2C 0.0103	EE 0.0099	FF 0.0098	58 0.0097	4C 0.0098	6B 0.0100	D1 0.0100	6F 0.0101
6	2E 0.0095	69 0.0099	84 0.0101	9C 0.0104	A4 0.0094	54 0.0095	31 0.0094	C2 0.0104	2E 0.0102	47 0.0097	21 0.0096	DE 0.0096	47 0.0098	75 0.0100	16 0.0096	2F 0.0100
7	22 0.0094	20 0.0097	AF 0.0100	F2 0.0103	9F 0.0093	DE 0.0094	78 0.0093	BC 0.0100	19 0.0098	42 0.0097	F5 0.0096	4A 0.0096	36 0.0097	3F 0.0098	CF 0.0094	B7 0.0099

Countermeasure Result (GF): CPA with HD model

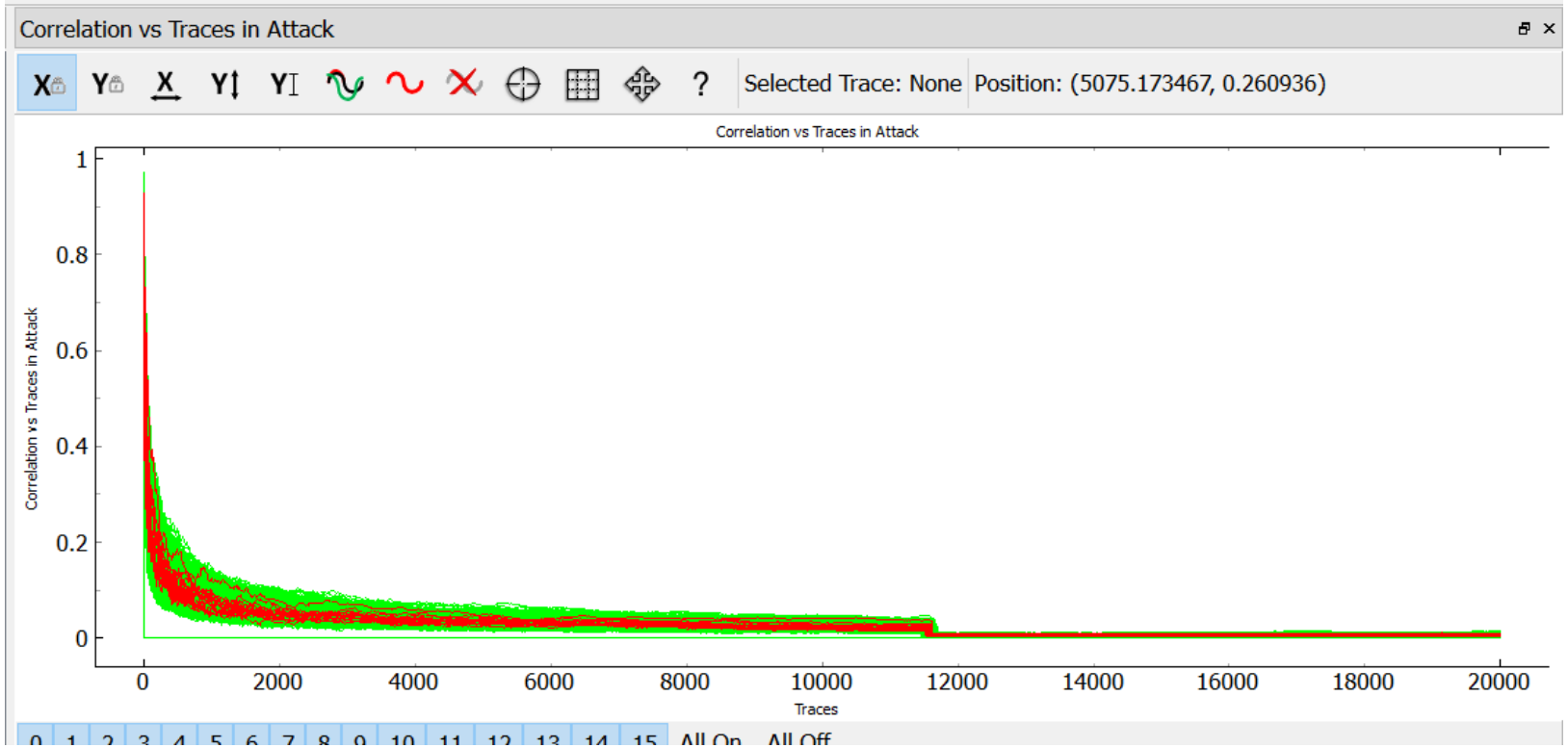


Figure: correlation vs number of traces for last round CPA with HD model. The figure suggests that none of the sub-bytes could be recovered

Countermeasure Result (GF): CPA with HD model

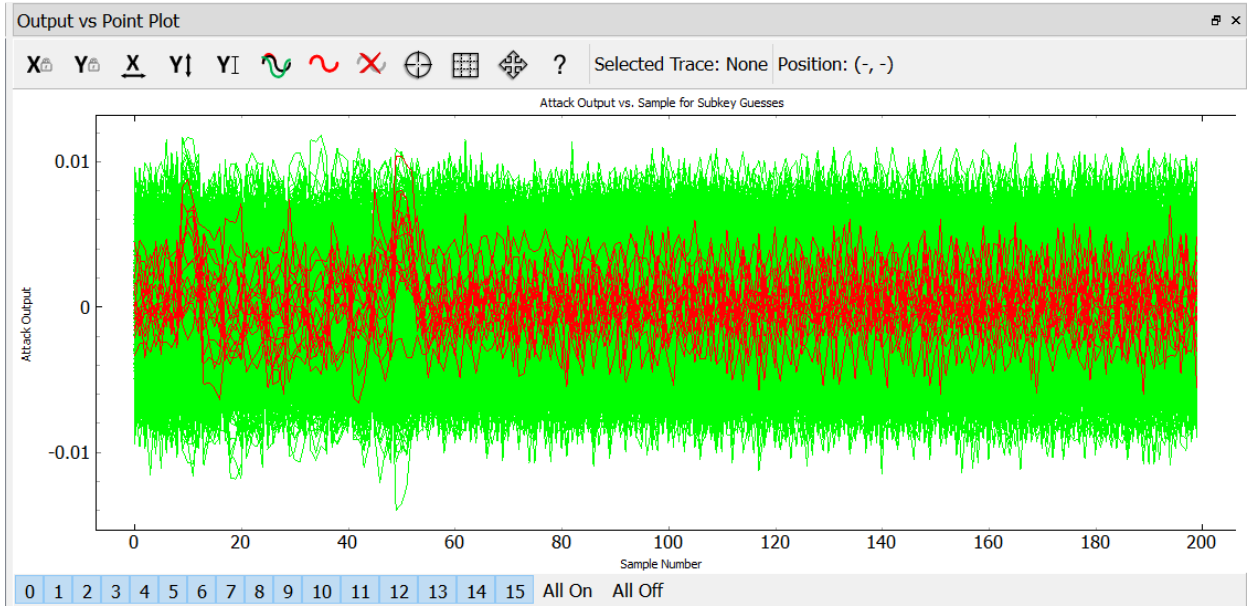


Figure: Attack output vs sample for subkey guesses

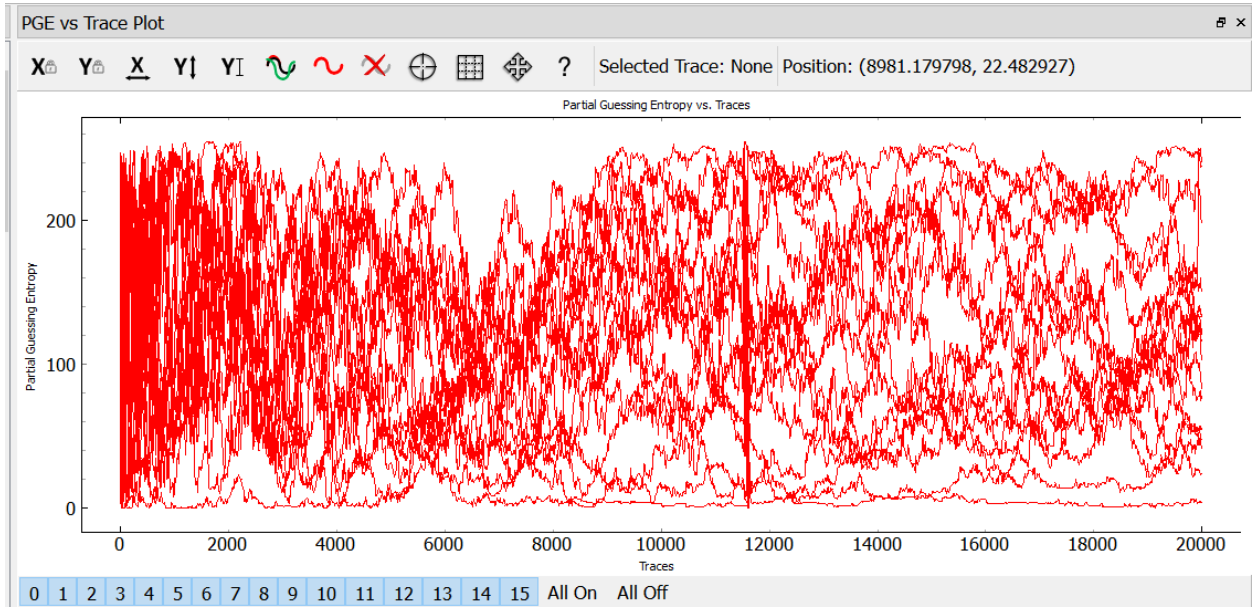


Figure: partial guessing entropy vs. number of traces for last round CPA with HD model

Countermeasure 1 (GF): Implementation Overhead

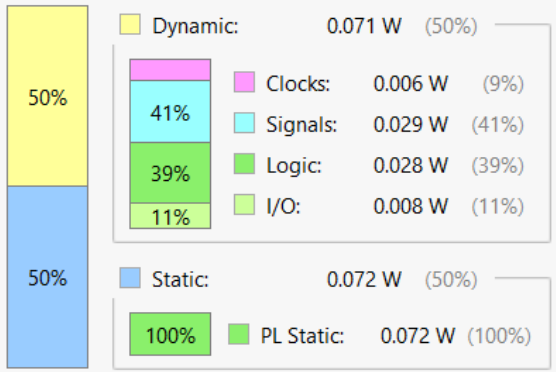
- **LUT Utilization:** 1912 vs 2722 => Overhead 42% of AES module
- **Total Power:** 0.143W vs 0.161W => Overhead is 14% of AES module
- Minimal Overhead to achieve significant side-channel robustness

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	1912	20800	9.19
FF	1275	41600	3.06
IO	42	170	24.71
BUFG	4	32	12.50

Power

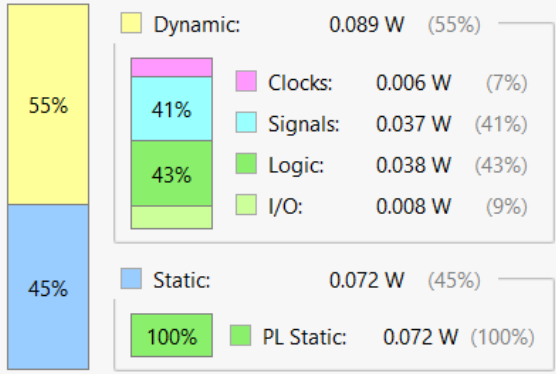


Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

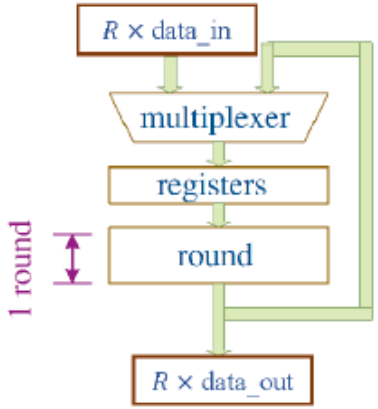
Resource	Utilization	Available	Utilization %
LUT	2722	20800	13.09
FF	1275	41600	3.06
IO	42	170	24.71
BUFG	4	32	12.50

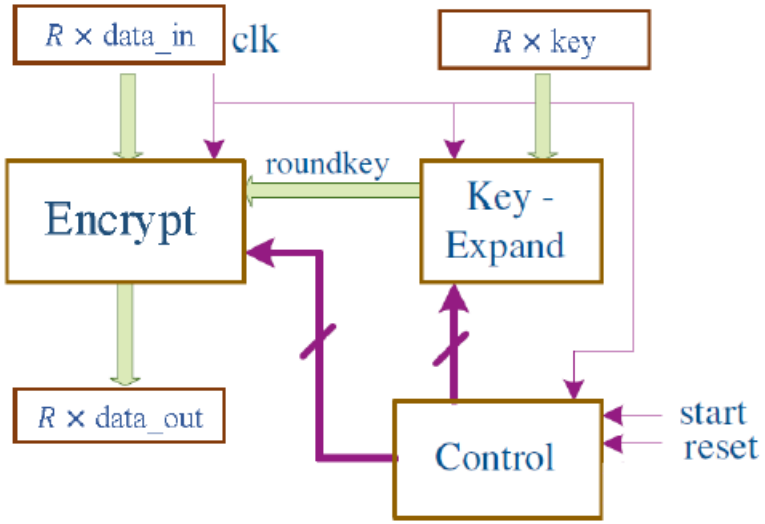
Power



Attempted Countermeasure: Approach 2 (Random Masking)

- A constant matrix is multiplied with the input plaintexts and key bytes to mask the power signature.

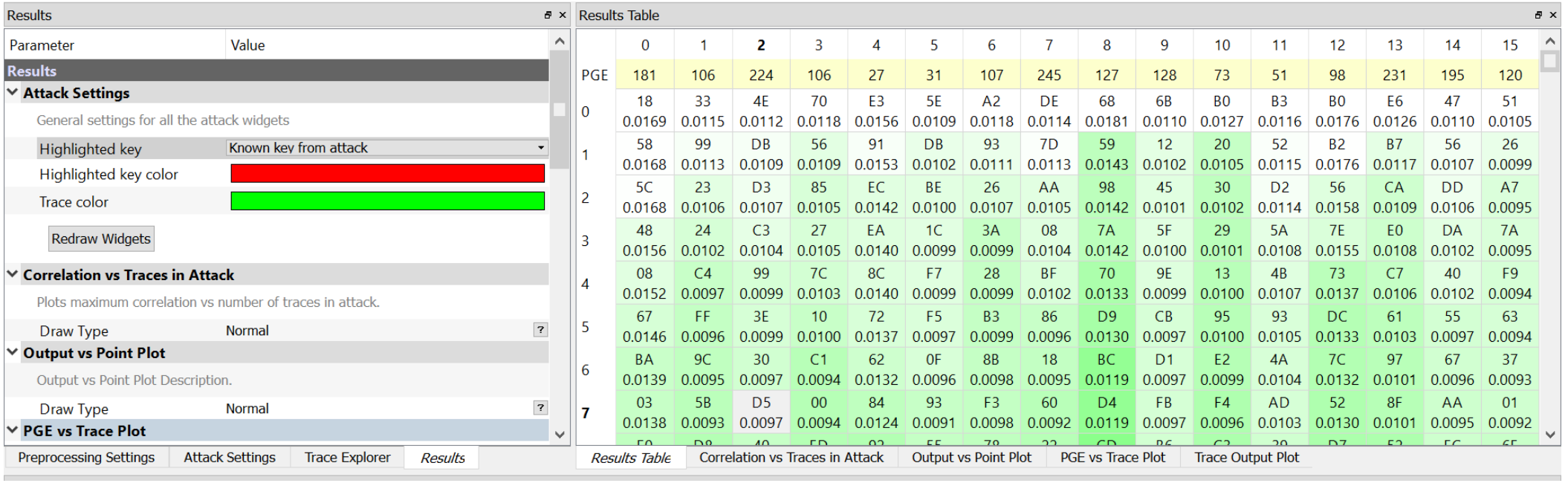




Zhang, Xinmiao, et al. "Hardware obfuscation of aes through finite field construction variation." *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2019.

Countermeasure Result (Random Masking) : CPA with HD model

- CPA with HD model is performed on the last round
- None of the sub-keys have been recovered for all sub-bytes



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	181	106	224	106	27	31	107	245	127	128	73	51	98	231	195	120
0	18 0.0169	33 0.0115	4E 0.0112	70 0.0118	E3 0.0156	5E 0.0109	A2 0.0118	DE 0.0114	68 0.0181	6B 0.0110	B0 0.0127	B3 0.0116	B0 0.0176	E6 0.0126	47 0.0110	51 0.0105
1	58 0.0168	99 0.0113	DB 0.0109	56 0.0109	91 0.0153	DB 0.0102	93 0.0111	7D 0.0113	59 0.0143	12 0.0102	20 0.0105	52 0.0115	B2 0.0176	B7 0.0117	56 0.0107	26 0.0099
2	5C 0.0168	23 0.0106	D3 0.0107	85 0.0105	EC 0.0142	BE 0.0100	26 0.0107	AA 0.0105	98 0.0142	45 0.0101	30 0.0102	D2 0.0114	56 0.0158	CA 0.0109	DD 0.0106	A7 0.0095
3	48 0.0156	24 0.0102	C3 0.0104	27 0.0105	EA 0.0140	1C 0.0099	3A 0.0099	08 0.0104	7A 0.0142	5F 0.0100	29 0.0101	5A 0.0108	7E 0.0155	E0 0.0108	DA 0.0102	7A 0.0095
4	08 0.0152	C4 0.0097	99 0.0099	7C 0.0103	8C 0.0140	F7 0.0099	28 0.0099	BF 0.0102	70 0.0133	9E 0.0099	13 0.0100	4B 0.0107	73 0.0137	C7 0.0106	40 0.0102	F9 0.0094
5	67 0.0146	FF 0.0096	3E 0.0099	10 0.0100	72 0.0137	F5 0.0097	B3 0.0099	86 0.0096	D9 0.0130	CB 0.0097	95 0.0100	93 0.0105	DC 0.0133	61 0.0103	55 0.0097	63 0.0094
6	BA 0.0139	9C 0.0095	30 0.0097	C1 0.0094	62 0.0132	0F 0.0096	8B 0.0098	18 0.0095	BC 0.0119	D1 0.0097	E2 0.0099	4A 0.0104	7C 0.0132	97 0.0101	67 0.0096	37 0.0093
7	03 0.0138	5B 0.0093	D5 0.0097	00 0.0094	84 0.0124	93 0.0091	F3 0.0098	60 0.0092	D4 0.0119	FB 0.0097	F4 0.0096	AD 0.0103	52 0.0130	8F 0.0101	AA 0.0095	01 0.0092

Countermeasure Result (Random Masking): CPA with HD model

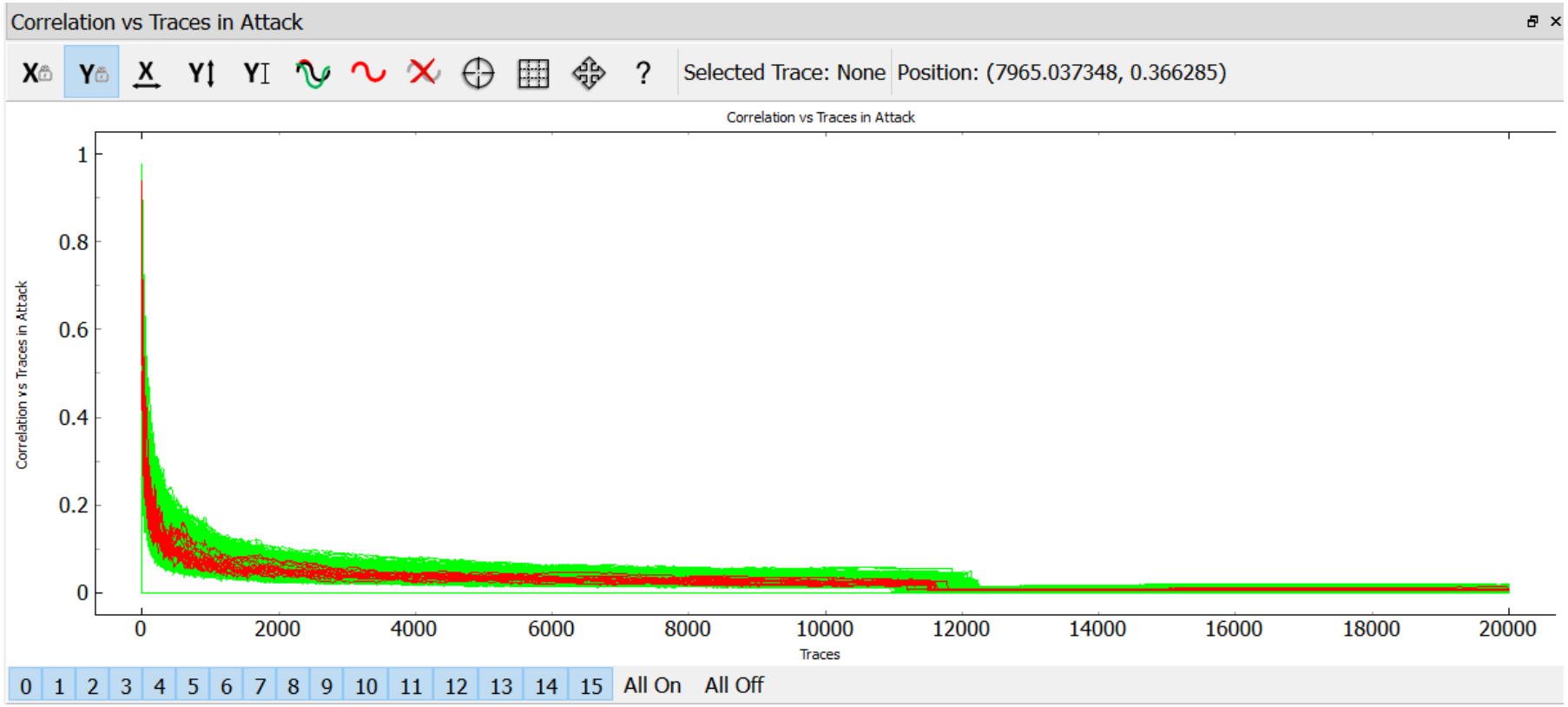


Figure: correlation vs number of traces for last round CPA with HD model. The figure suggests that none of the sub-bytes could be recovered

Countermeasure Result (Random Masking): CPA with HD model

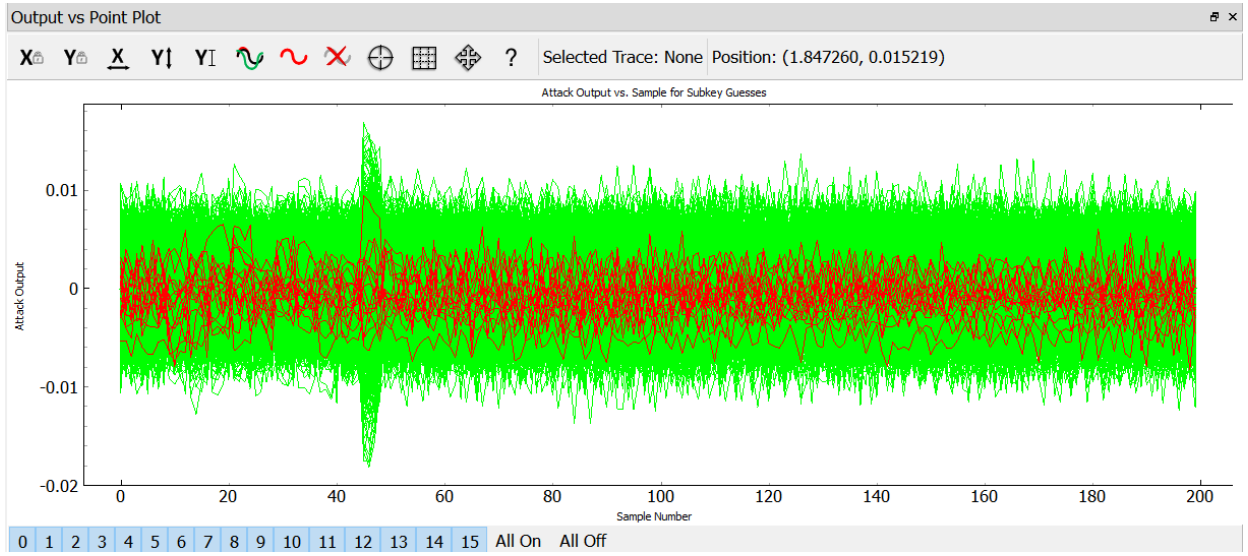


Figure: Attack output vs sample for subkey guesses

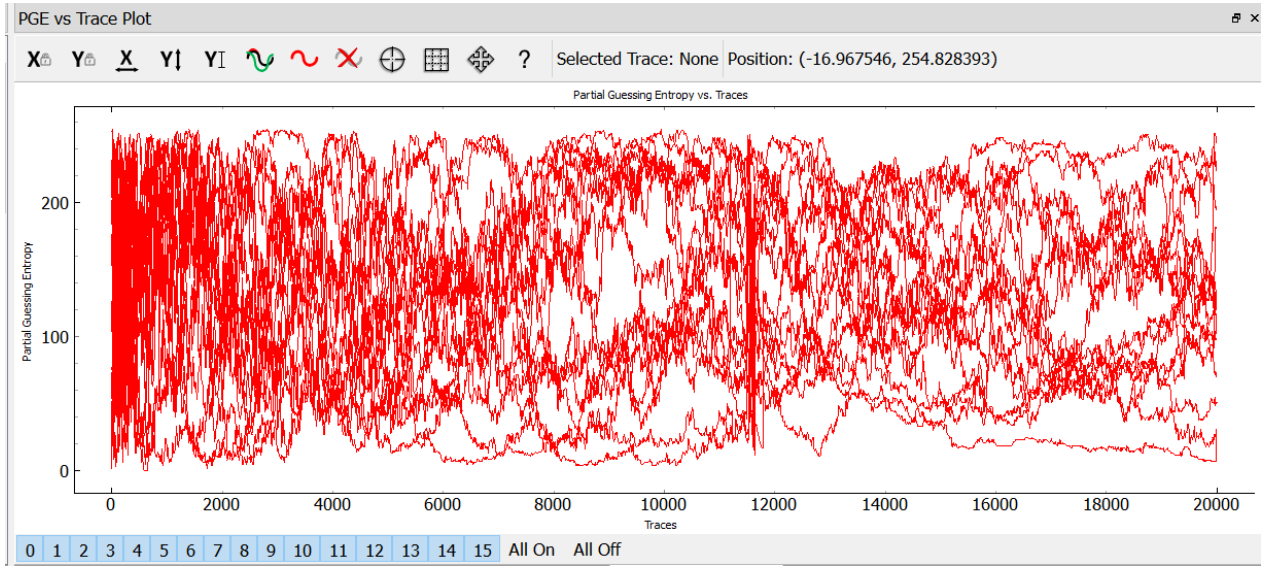


Figure: partial guessing entropy vs. number of traces for last round CPA with HD model

Attempted Countermeasure: Approach 3 (Frequency Randomization)

- Before sending to AES, the clock frequency is randomized with the help of an LFSR.
- This desynchronizes the power traces→ more complicated DPA attack.

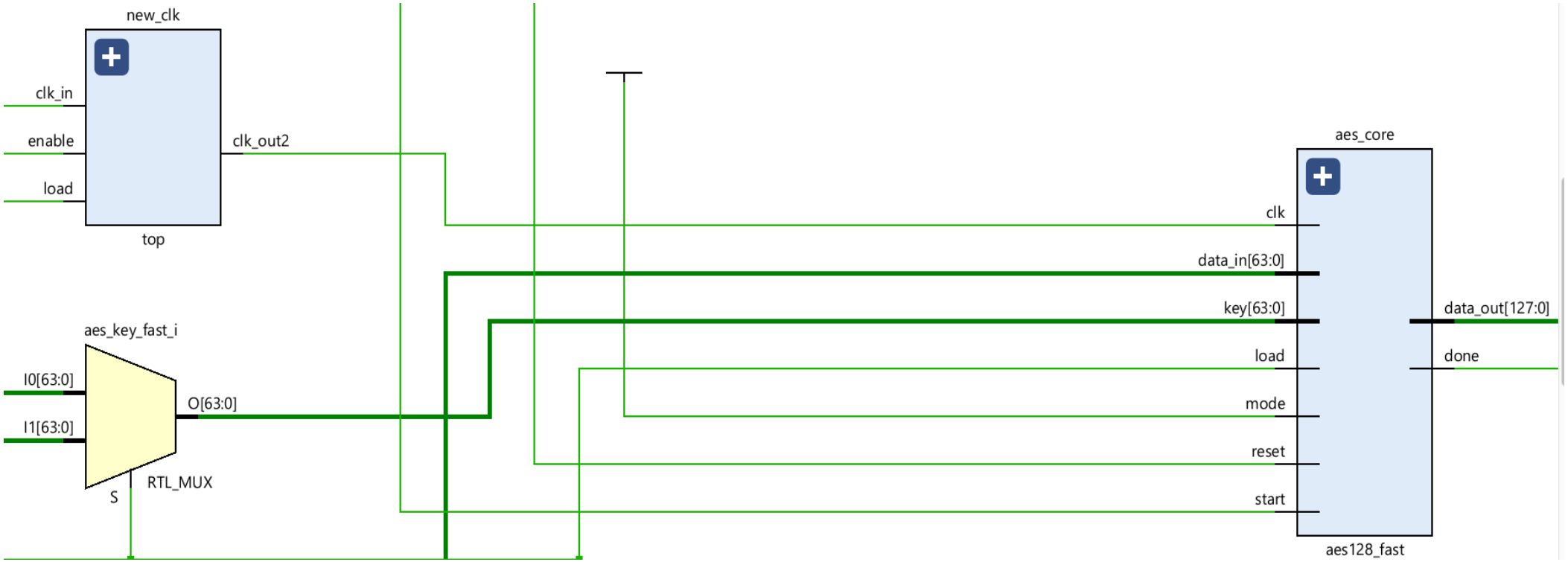


Fig: Block Level Diagram of our Countermeasure

Attempted Countermeasure: Approach 3 (Frequency Randomization)

- AND chain is implemented in such a way that it doesn't cause setup and hold violation.

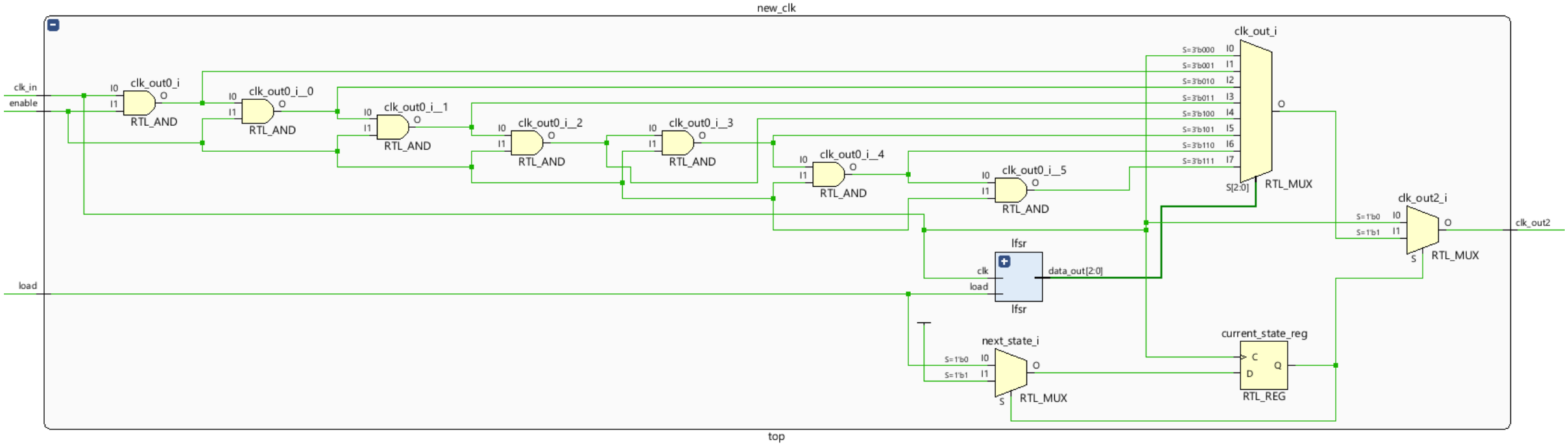
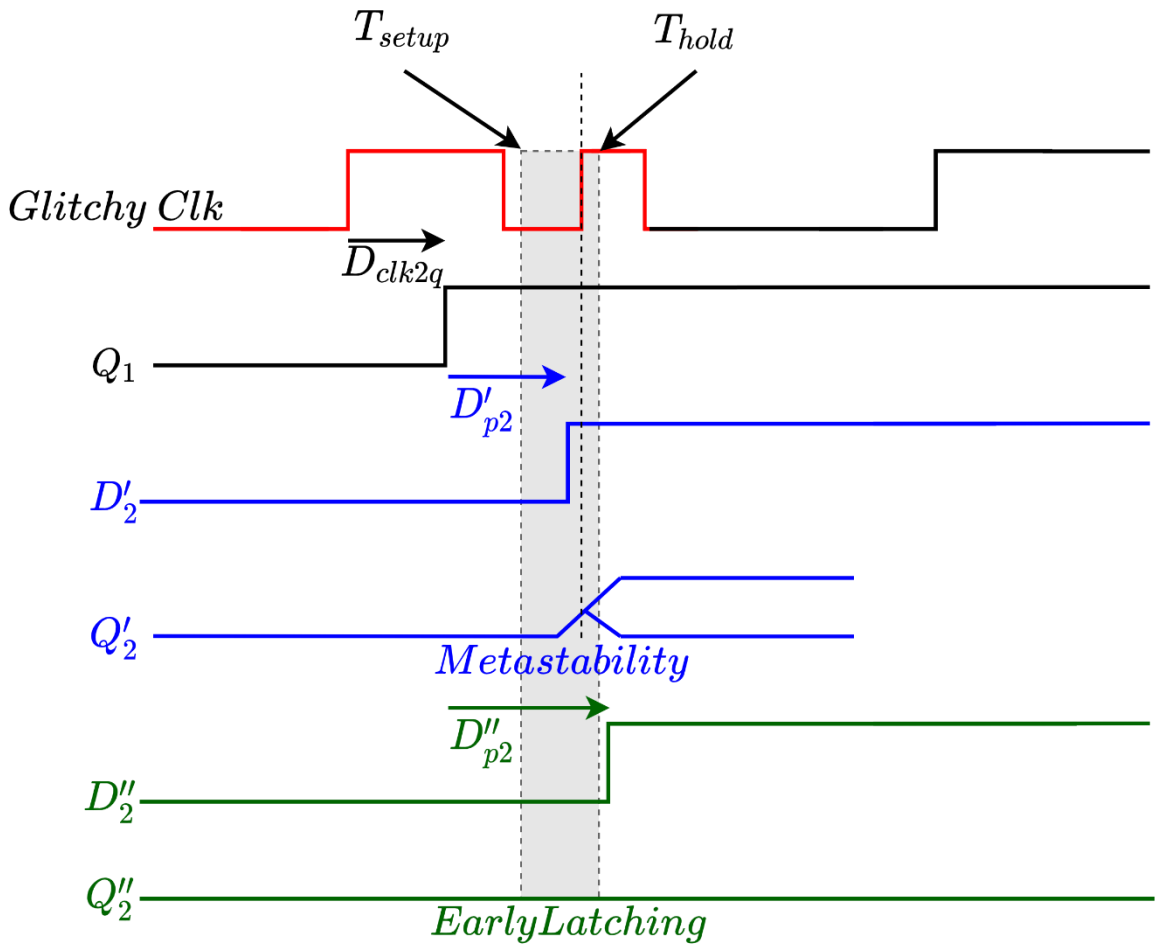


Fig: Schematic of our proposed method.

Fault Injection Attacks

Threat Model for Fault Injection

- Clock glitch insertion
  - Setup-time and hold-time violations
  - Early latching
- Differential Fault Analysis (DFA)
- Extraction of secret key of crypto modules.
- Assumptions-
  - Unlimited physical access to the design.
  - Complete knowledge of the design's functionality.
  - Complete control over the timing, duration and position of a clock glitch.



Simple Fault Analysis

- Attack on Crypto IP
  - Fault Injection
  - Fault Propagation
  - Simple Mathematical Analysis
  - Direct Extraction of key

- Example
  - Assertion of done signal after 1st round of AES by a fault injection.
  - Propagation of the 1st round value to the cipher output.

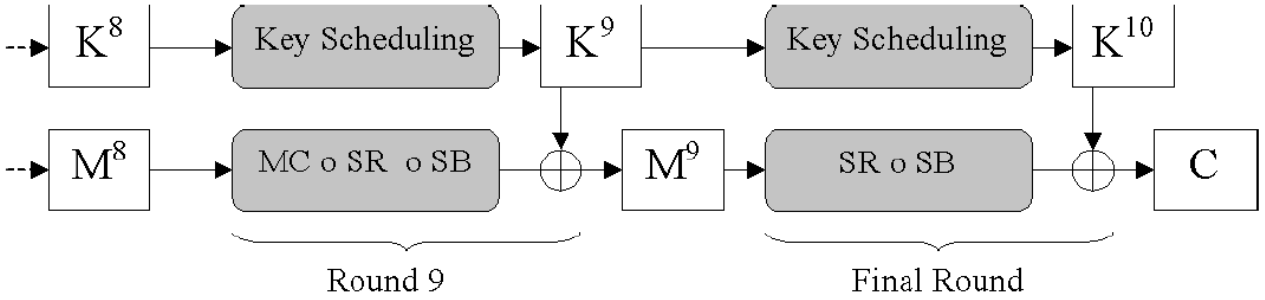
$$state\_reg\_1 = plaintext \oplus key$$
$$ciphertext = state\_reg\_1$$
$$key = plaintext \oplus ciphertext$$

Differential Fault Analysis (DFA)

- Attack on Crypto Modules
  - Fault Injection
  - Fault Propagation
  - Faulty Cipher
  - Analysis on the faulty cipher and original cipher.
  - Extraction of key.

- Example
  - A single-bit fault at the input to the AES 10th round helps to extract a byte of 10th round key.

DFA on AES (10th round)



$$C = ShiftRows(SubBytes(M^9)) \oplus K^{10}$$

$$C_{ShiftRow(i)} = SubByte(M_i^9) \oplus K^{10}_{ShiftRow(i)}$$

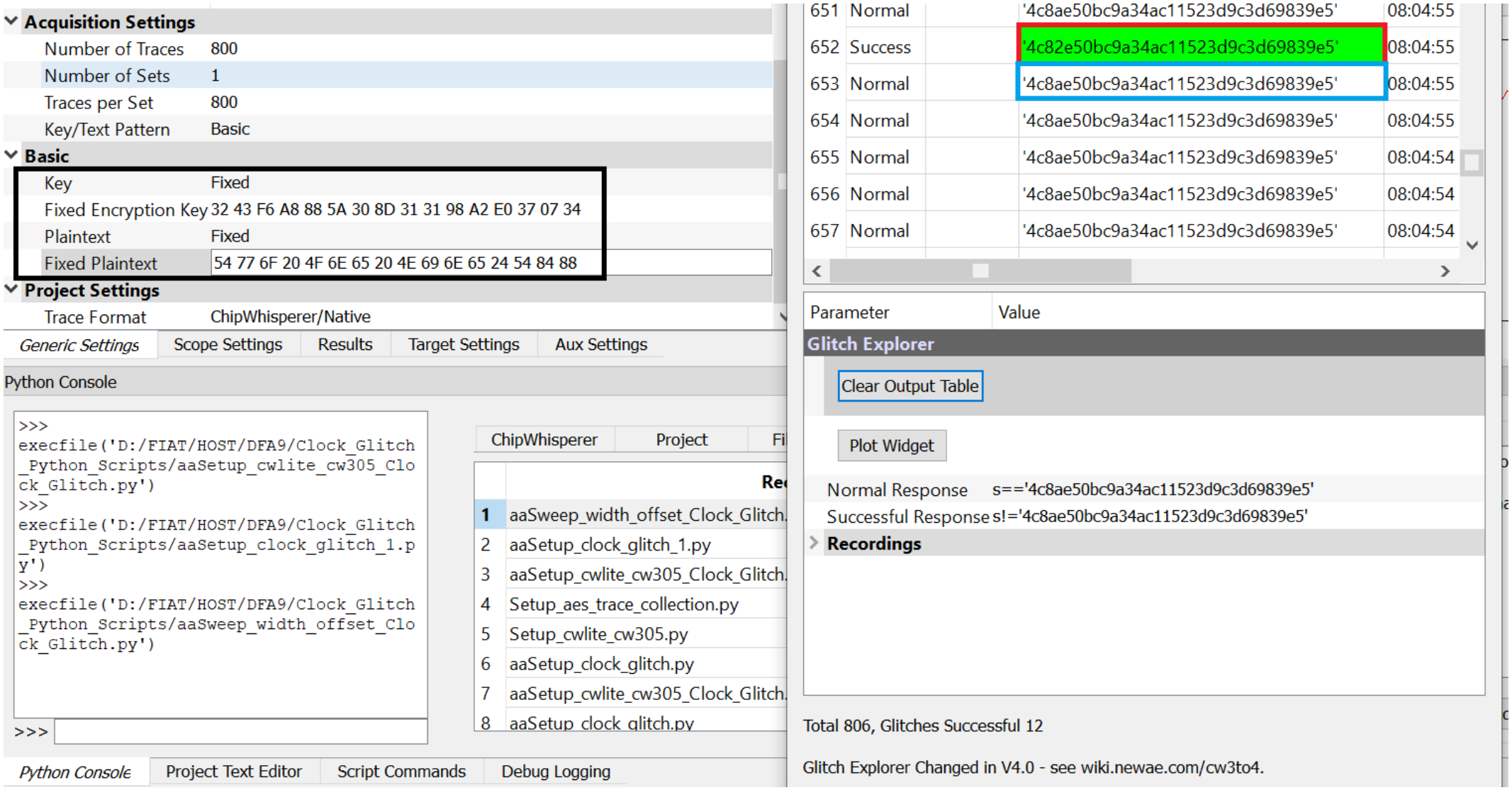
$$D_{ShiftRow(j)} = SubByte(M_j^9 \oplus e_j) \oplus K^{10}_{ShiftRow(j)}$$

$$D_{ShiftRow(i)} = SubByte(M_i^9) \oplus K^{10}_{ShiftRow(i)}$$

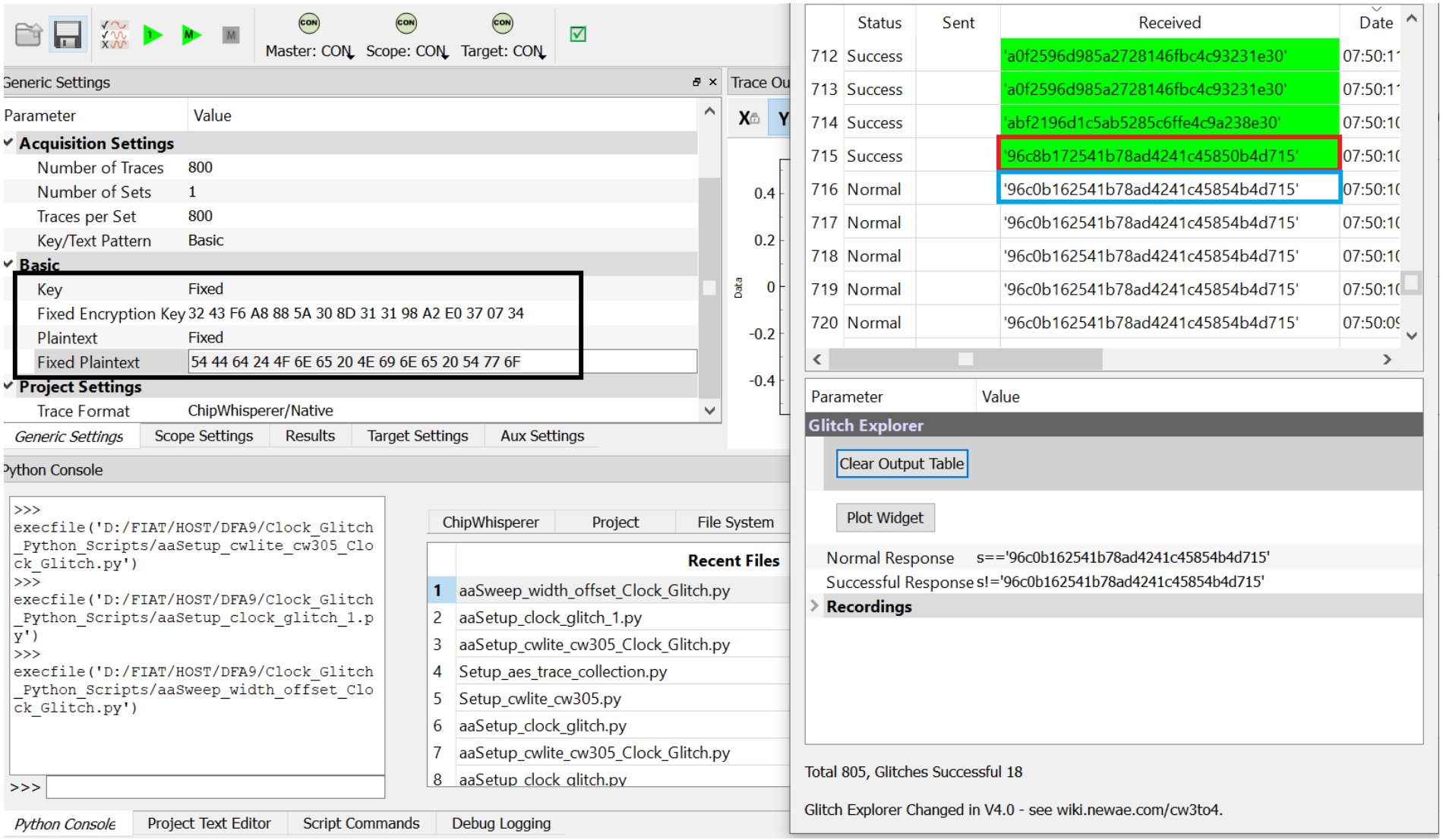
$$C_{ShiftRow(i)} \oplus D_{ShiftRow(i)} = 0$$

$$C_{ShiftRow(j)} \oplus D_{ShiftRow(j)} = SubByte(M_j^9) \oplus SubByte(M_j^9 \oplus e_j)$$

Fault Injection Attack on AES



Fault Injection Attack on AES (Cont.)



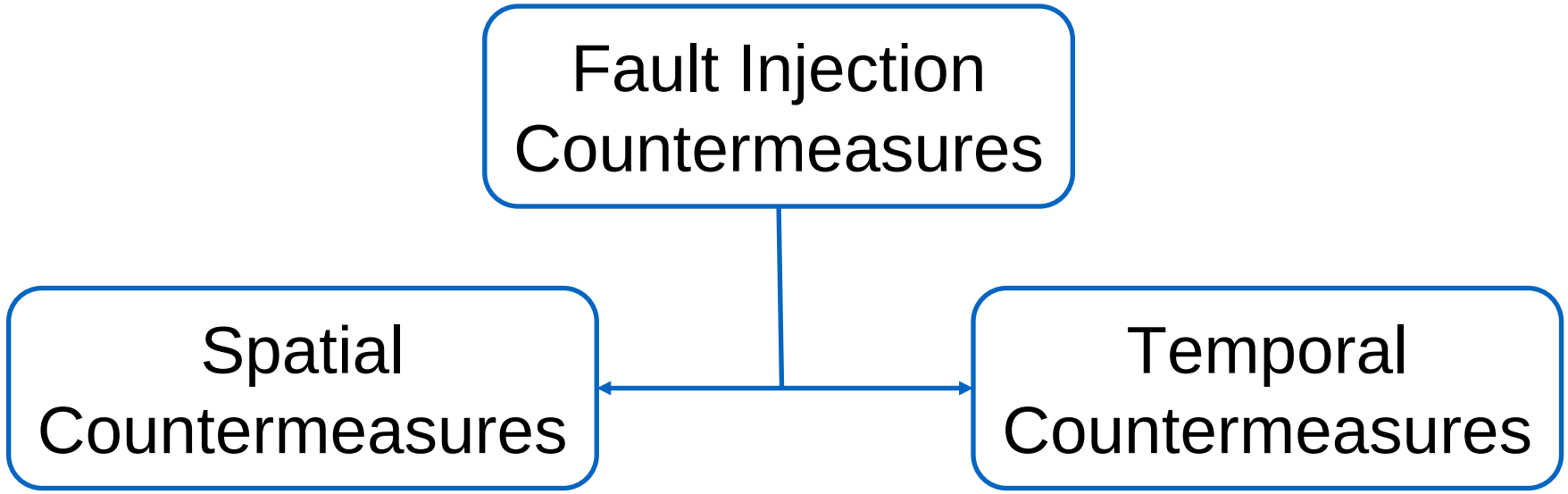
Experimental Result for DFA

```
Fixed Key  0x3243f6a8885a308d313198a2e0370734
10th Round's Key  0xcbf39708ac73d39523cf94af518317ac

Plaintext1  0x54776f204f6e65204e696e6524548488
Original Cipher 0xe2f9a2d17155c1be7a894dd7a7fdc182
Faulty Cipher   0xe2f9a2d17155c1be7a894dd7a790c182
C_orig xor C_faulty   0x6d0000

Plaintext2  0x544464244f6e65204e696e652054776f
Original Cipher 0x5b5c8b518cf0dd3f0f425c3a7139abc6
Faulty Cipher   0x5b5c8b518cf0ddd50f425c3a026babc6
C_orig xor C_faulty   0xea0000000073520000
```

```
Analysis of the possible search space

82 0xee 8a 0x83 8 6d   1
a4 0xb4 a6 0xd9 2 6d   1
e0 0x1c f0 0x71 10 6d   1


3d 0x1e 3f 0x4c 2 52   1
41 0xba 51 0xe8 10 52   1
c0 0x83 c8 0xd1 8 52   1
c7 0xff e7 0xad 20 52   1

Key search space for Plaintext1
['0xee', '0x83', '0xb4', '0xd9', '0x1c', '0x71']

Key search space for Plaintext2
['0x1e', '0x4c', '0xba', '0xe8', '0x83', '0xd1', '0xff', '0xad']

key_byte
['0x83']
```

Fault Injection Countermeasures

Fault Injection Countermeasures



- Spatial countermeasures:
  - Multiple instances of the same implementation placed in the silicon.
  - After each computation, the outputs from all the instances are compared.
  - If there is a mismatch, fault injection has occurred.
  - Incurs two to three-fold overhead, based on the number of instantiations implemented.

Fault Injection Countermeasures cont'd

- Temporal Countermeasures:
  - Same operation is performed on the same implementation multiple times, and the outputs are compared.
  - If there is a mismatch, a fault attack has occurred.
  - Minimal area overhead, but timing becomes slower.
- Comparison between countermeasures:

Spatial Countermeasures	Temporal Countermeasures
Two to three-fold area overhead, but no timing overhead.	High timing overhead, but minimal area overhead.
Suitable for high-speed circuits where area requirement is not much of a concern.	Suitable for slower mobile devices where timing requirement is not stringent, but area requirement is.

Conclusion

- Various Power Side-Channel Analysis was attempted on the provided AES module
- CPA and ML-based attacks were successful in retrieving the secret key
- Successfully injected fault through clock glitch, and generated threat model environment for DFA
- However, due to low observability of the hardware environment, key extraction through DFA is incomplete
- Multiple countermeasures are discussed that can make the cryptographic module robust against side-channel attacks

Thank You!