Differential Power Analysis of AES

F. De Santis
desantis@tum.de

Sichere Implementierung kryptographischer Verfahren WS 2015-2016
Lehrstuhl für Sicherheit in der Informationstechnik
Technische Universität München

02.12.2015

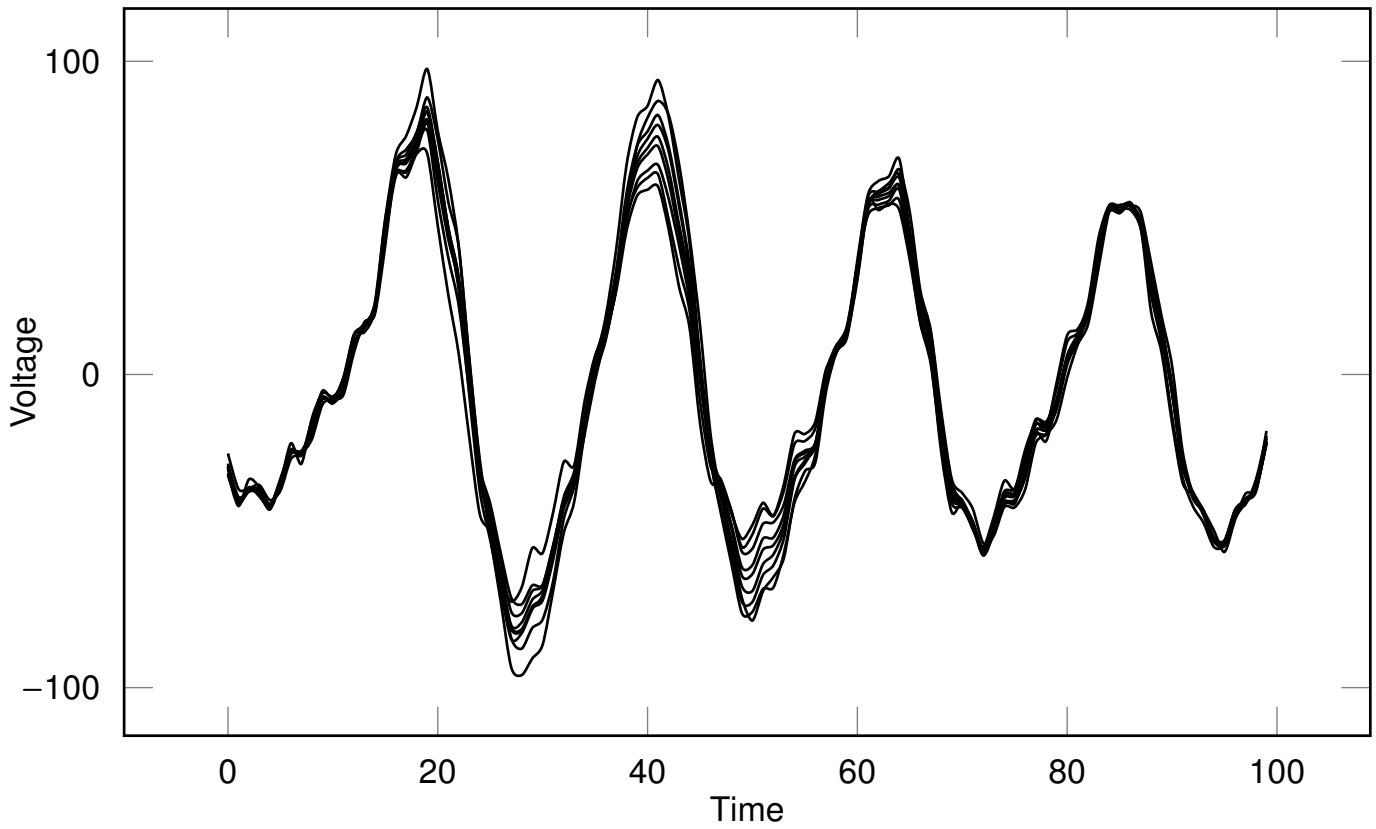
Outline

Section 1

Differential Power Analysis (DPA)

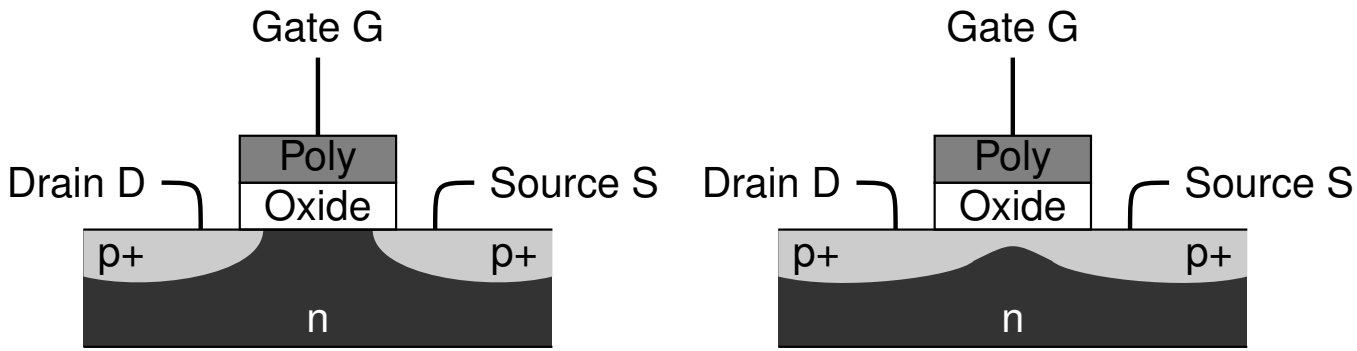
Differential Power Analysis (DPA)

- Differential Power Analysis (DPA) exploits key-dependent differences in the instantaneous power consumption of cryptographic CMOS devices to recover the secret key



CMOS Technology

- Transistors are used as switches to realize digital cells
- Digital cells computes basic logic functions (*e.g.*, NOT, AND, . . .)
- Logic levels are defined by the logic style (*e.g.*, VML, CML, . . .)

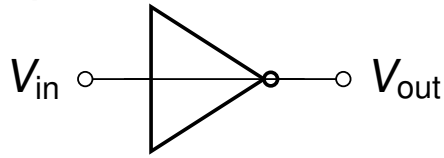


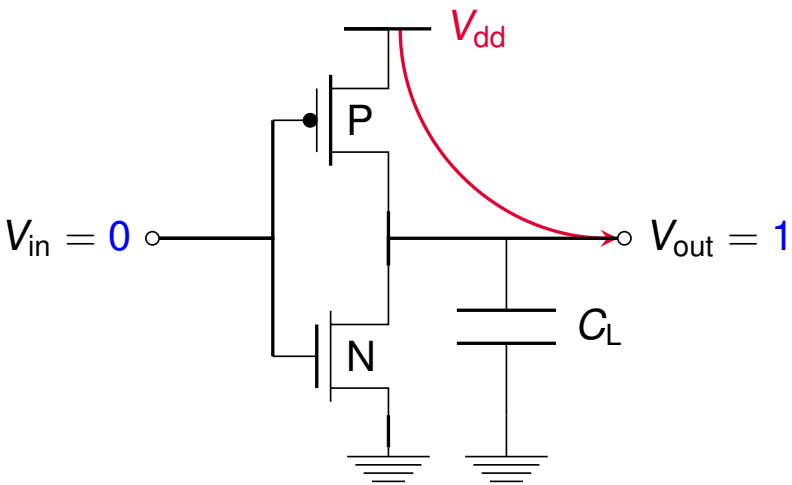
(a) $V_{GD} \geq V_T$. (b) $V_{GD} < V_T$.

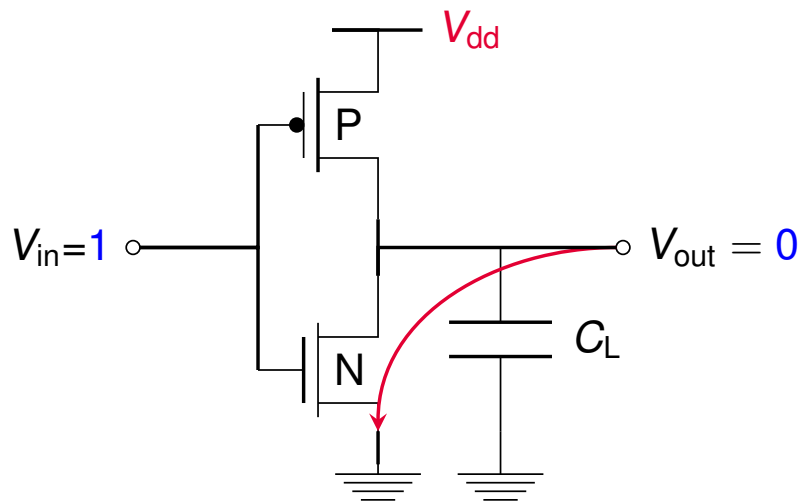
Power Consumption in CMOS Devices

- NOT is the simplest CMOS cell:

V_{in}	V_{out}
0	1
1	0

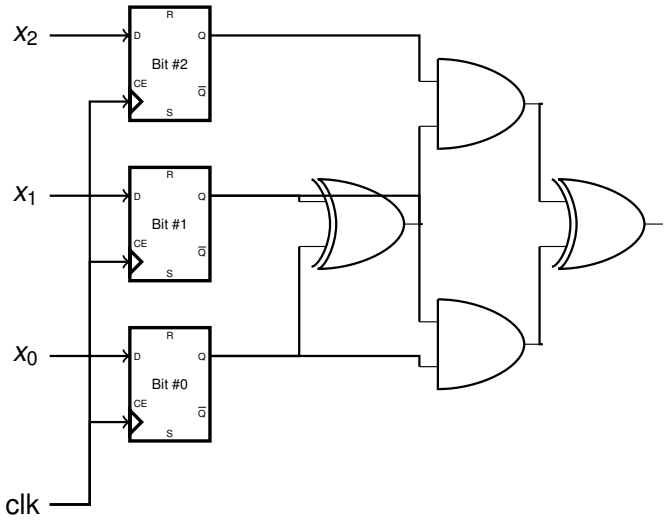


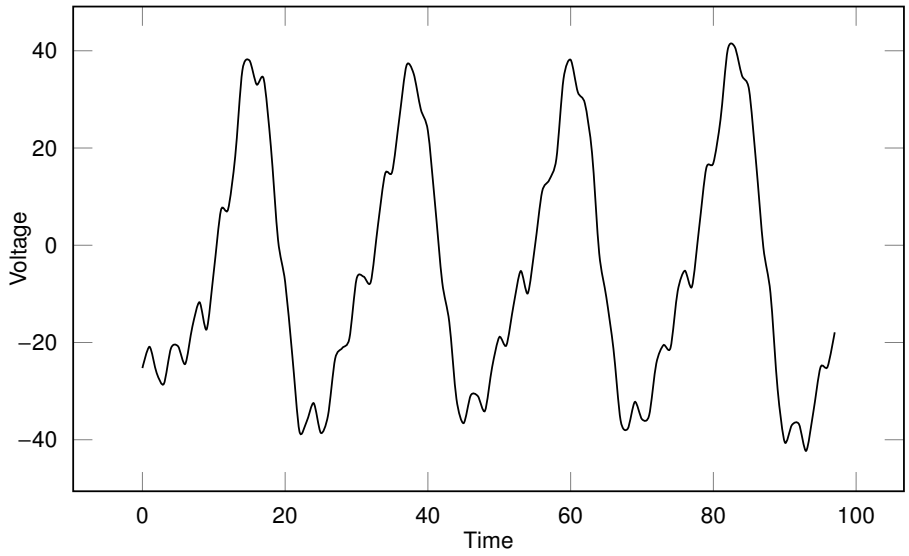




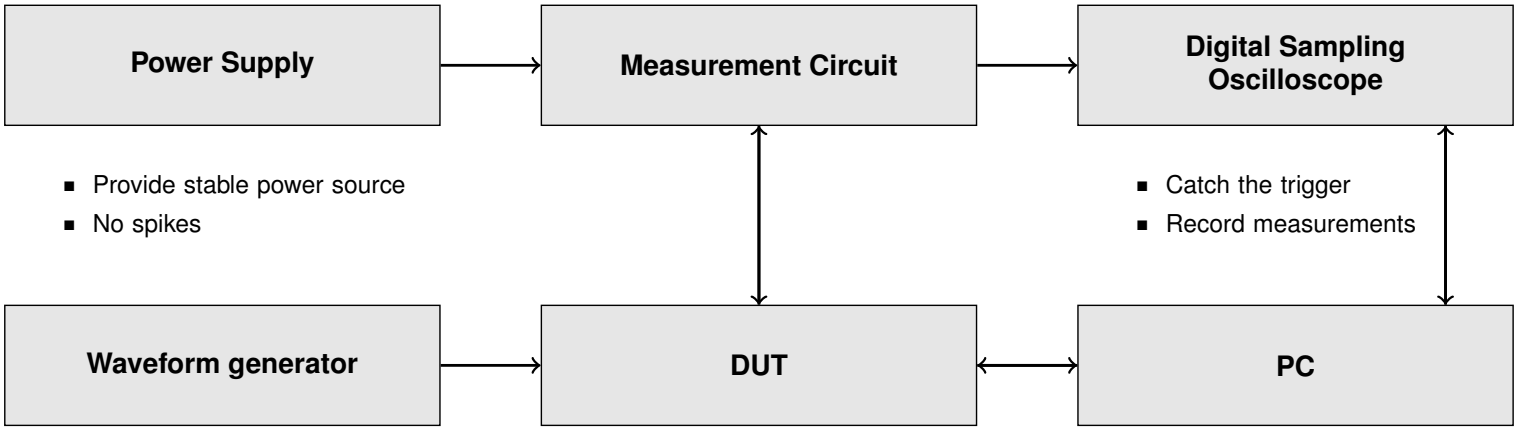
Power Consumption in CMOS Devices

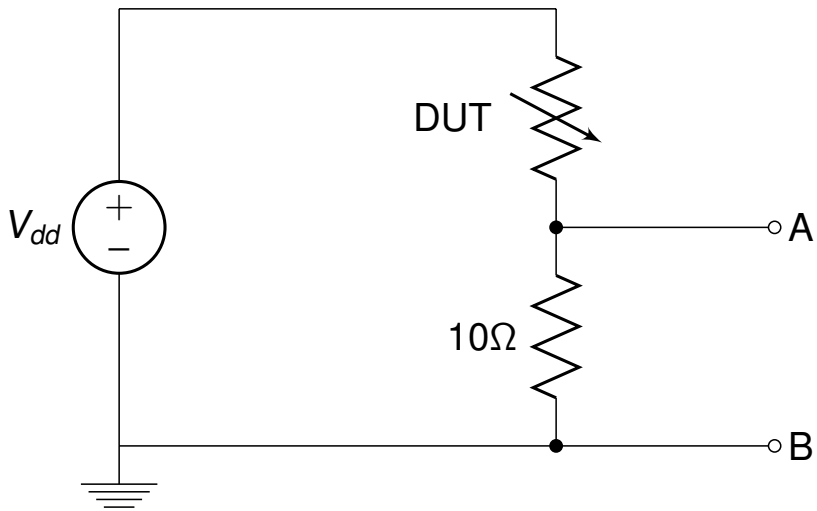
- Synchronous register updates
- Asynchronous switching activity of the combinational path



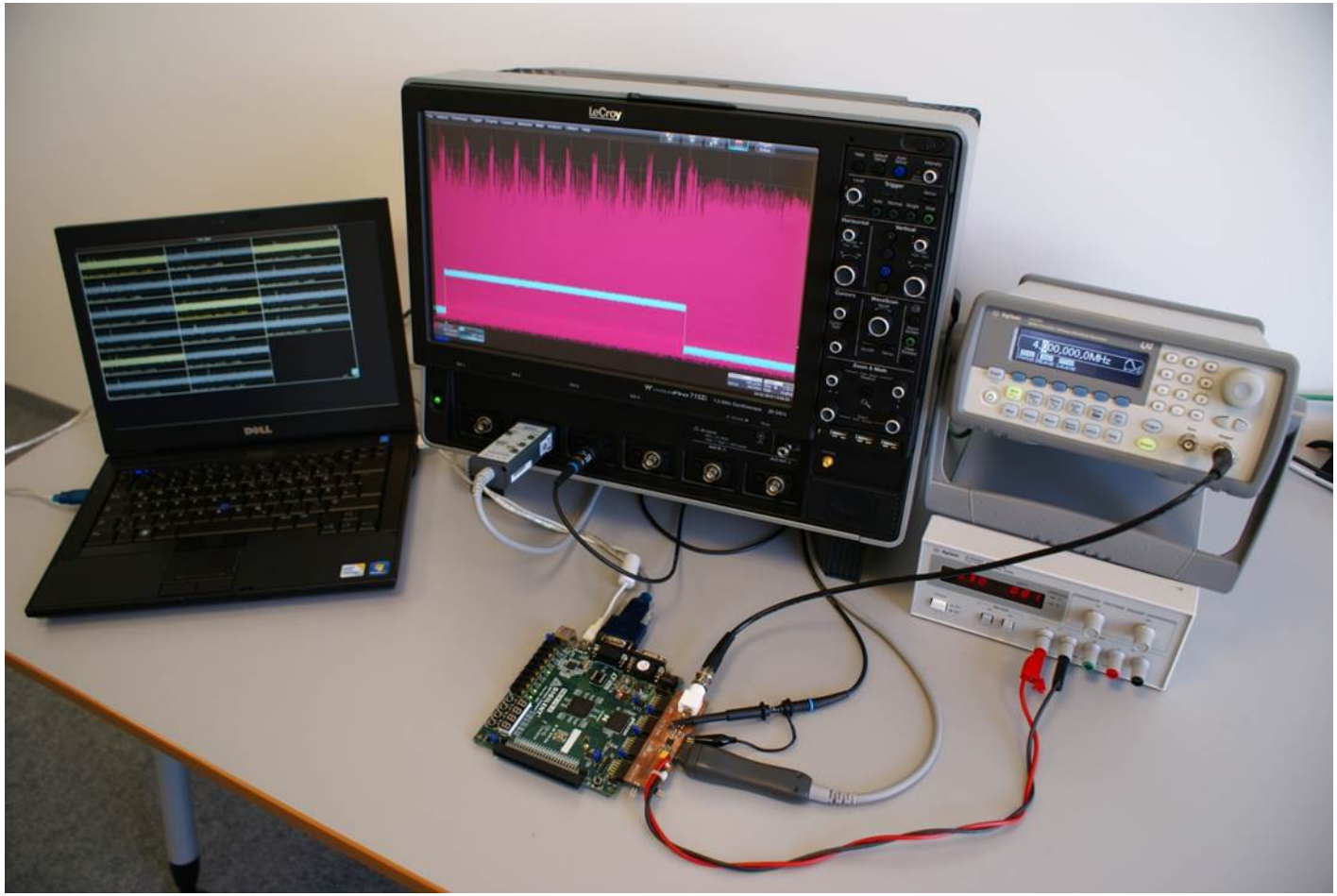


Power Measurements

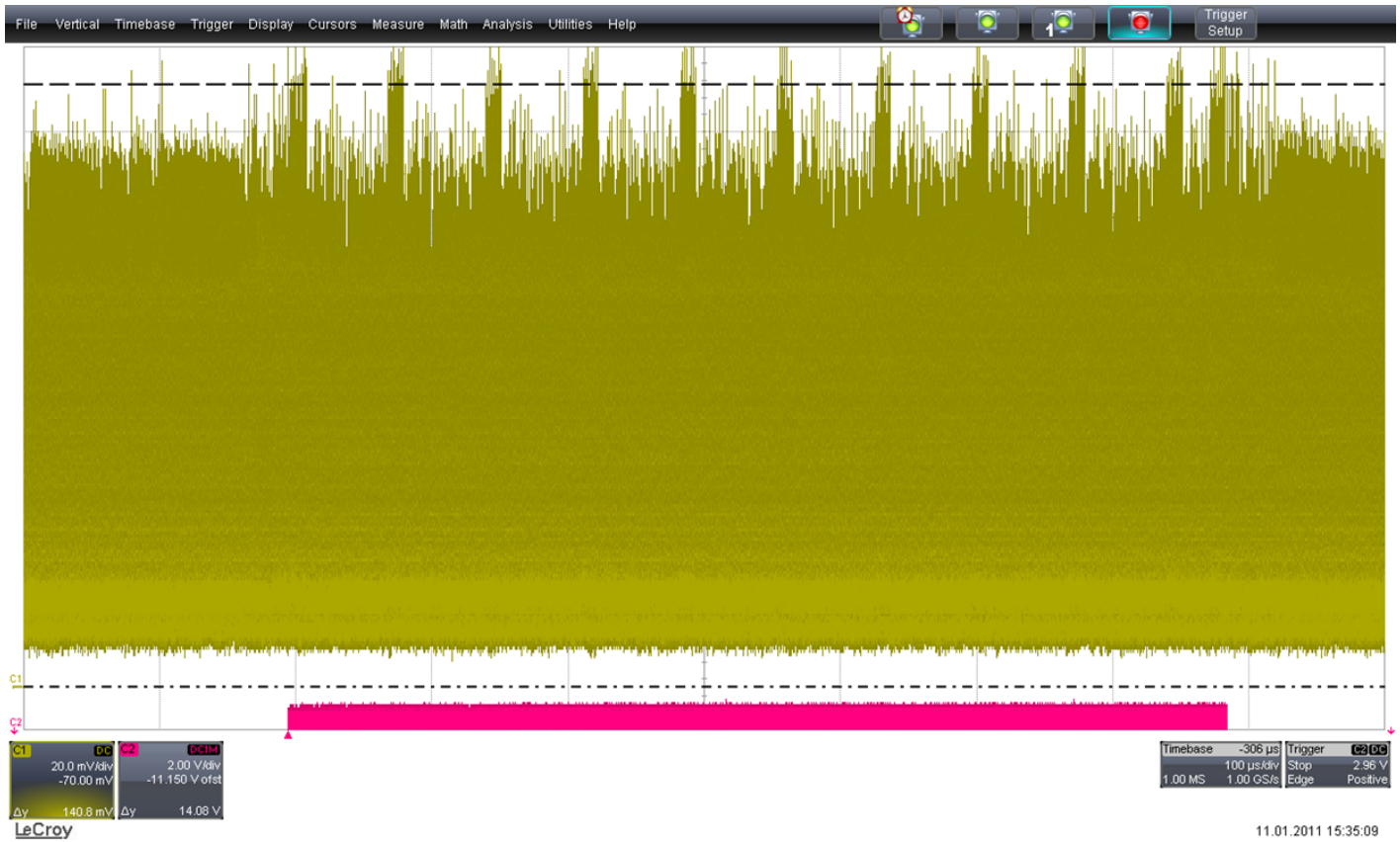




Power Measurements



Power Measurements



DPA Attacks

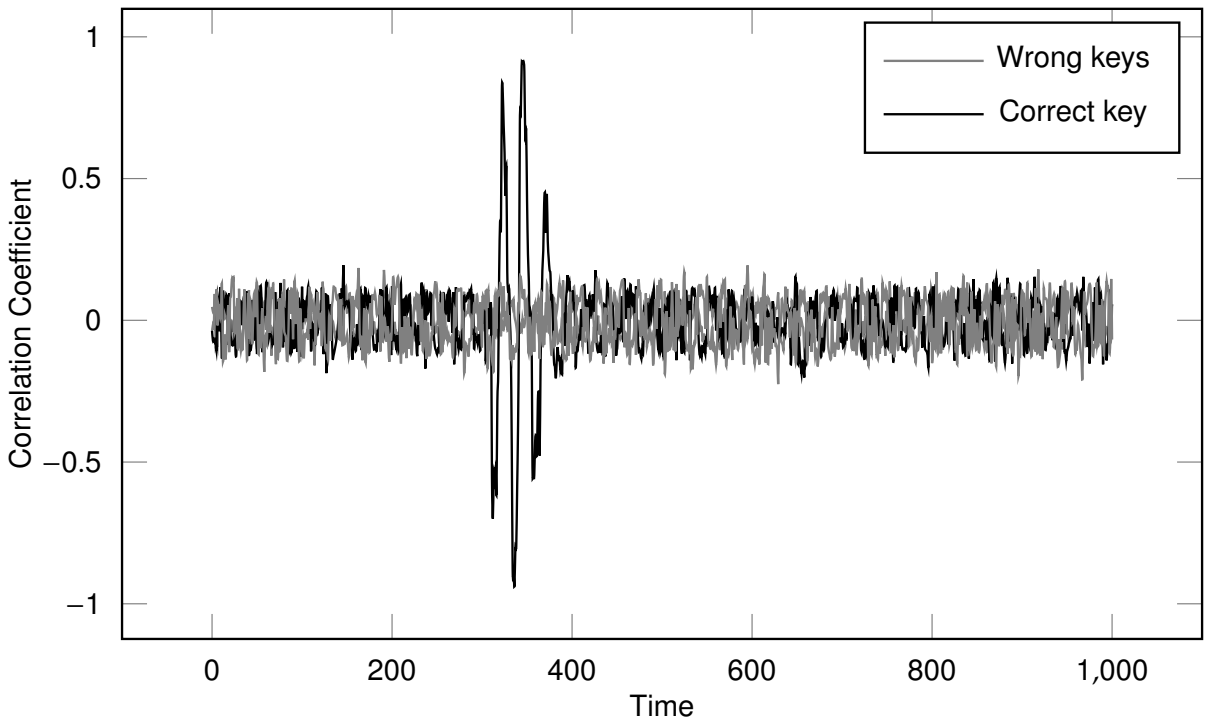
Idea: for each key hypothesis compute intermediate values, apply a power model and verify the key hypothesis by statistical means

Wide range of DPA attacks employing different statistical tools

- Difference of means
- Pearson's correlation coefficient
- Mutual information
- ...

DPA Attacks on AES

- The target is either the **first round** or the **last round** depending on whether the plaintexts or the ciphertexts are known
- The attack is performed independently for each byte of the state (only 2^8 key hypothesis must be verified)



Section 2

Task Description

Task Description

1. Implement the DPA attack on the **last round** of AES in Python

   - ➥ Framework @ https://tueisec-sica.sec.ei.tum.de/
     - ✯ `trace_measurement.py`: take measurements
     - ✯ `sample_trace_plot.py`: tool for plotting traces
     - ✯ `main.py` **not** to be modified
     - ✯ `student.py` for implementing the attack
     - ✯ `test_key.py` for computing the secret key

   - ➥ Sample measurements
     - ✯ Plaintexts, Ciphertexts and Secret Key: they can be used to verify the attack script

Task Description

2. Collect measurements of your AES implementation and run the DPA attack on the **last round** to extract the **last round** key
   - ➥ Schematic of the measurement board
   - ➥ Firmware of your own implementation and of a reference implementation @ https://tueisec-sica.sec.ei.tum.de/hex/ with a different secret key for each student
   - ➥ The correctness of the last round key can be tested against the plaintexts and ciphertexts saved with the traces

Section 3

Measurements

Lab Equipment

Taking Measurements

1. Flash the firmware onto the MCU with
   $ avrdude -p m644 -c avrisp2 -P usb -B 5 -U flash:w:student.hex -F
2. Check all the connections against the schematic
3. Disconnect the programmer and re-insert the card into the reader
4. Modify the script `trace_measurement.py`
   - ➥ Output directory
   - ➥ Sampling rate
   - ➥ Sampling window
5. Run the script and analyse the measurements
   - ➥ Average traces and identify the rounds
   - ➥ Set the acquisition window on the last round
   - ➥ Take measurements using different sampling rates
   - ➥ ...

Section 4

Submitting Results

Handing in Results

- Hand in `student.py` @ https://tueisec-sica.sec.ei.tum.de/handin/
  - ➥ Authentication using the LRZ Kennung required

- Multiple submissions are possible
  - ➥ Only the last submission is considered (files are overwritten)

- Deadline for submission fixed in 6 weeks
  - ➥ **13.01.2016 at 23:59:59 CET**

Final Remarks

- The assignment is passed if the `key.txt` is correct **and** the `student.py` works correctly on freshly generated new measurements

- Take measurements as early as possible
- Reserve an appointment (30 minute) with the tutors
- Store traces on your own USB stick
- DUT is leaking the Hamming Weight
- Do DPA attacks offline
- Do not include plotting commands in `student.py`
- Use numpy for numerical computations

Stairway to Heaven

1. Download the project skeleton
2. Go to the lab and collect measurements
3. Implement the DPA attack
4. Run the attack on collected measurements
5. Submit the source code and the `key.txt`
   before 13.01.2016 23:59:59 CET

Reserve your time slot per E-mail with the tutors