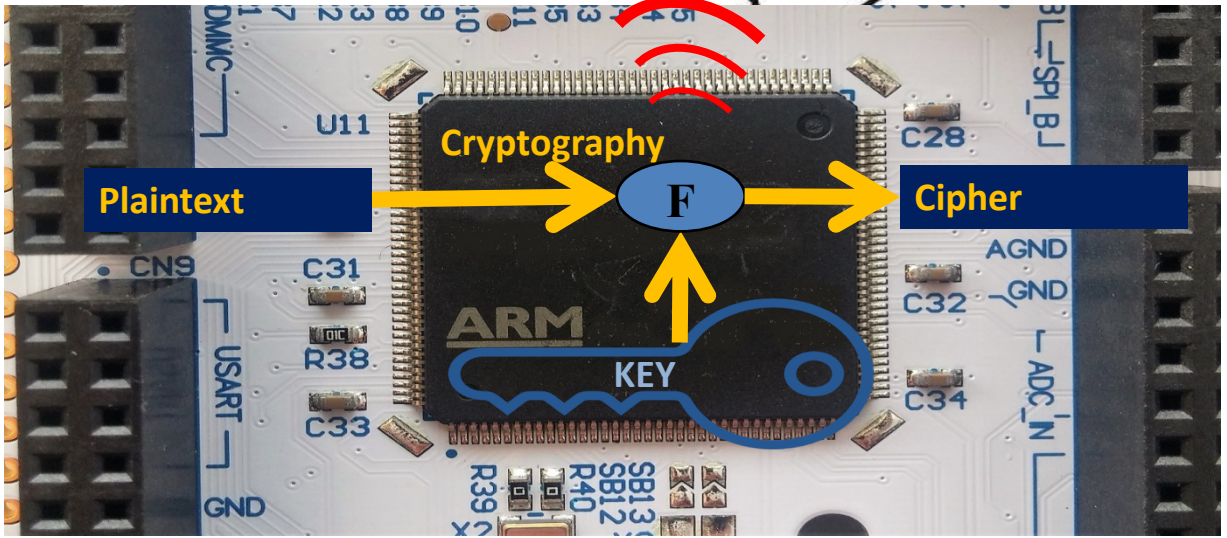
Hardware (Implementation) Attacks

- Side Channel Attack
- Fault Attack

Credit Ehsan Aerabi

Side Channel Attack to Cryptographic MCUs

- Exploiting side channel leakage:



Advanced Encryption Standard

AES

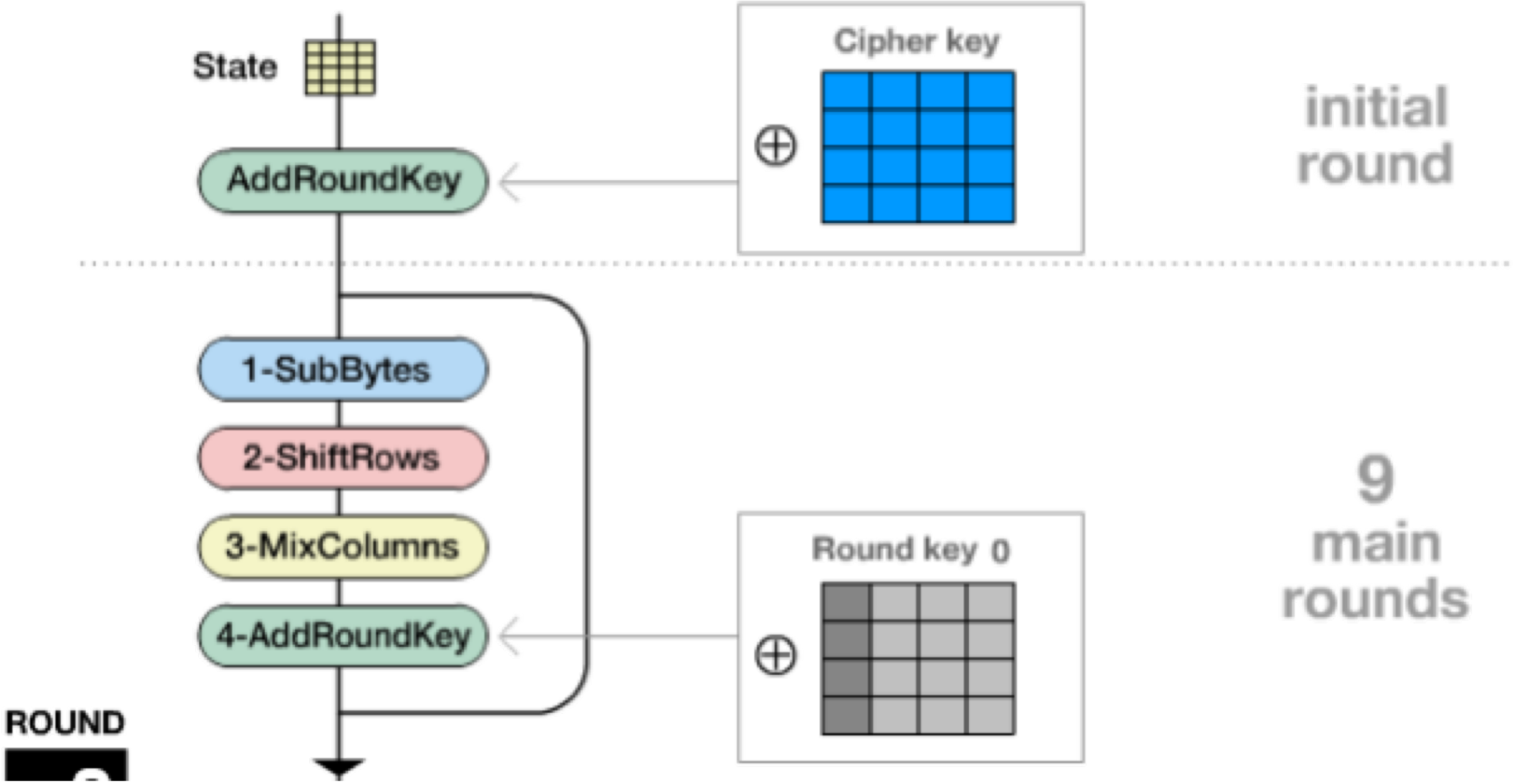
THE CASE STUDY

The AES process

- 1997: NIST publishes request for proposal
- 1998: 15 submissions. Five claimed attacks.
- 1999: NIST chooses 5 finalists
- 2000: NIST chooses Rijndael as AES (designed in Belgium)

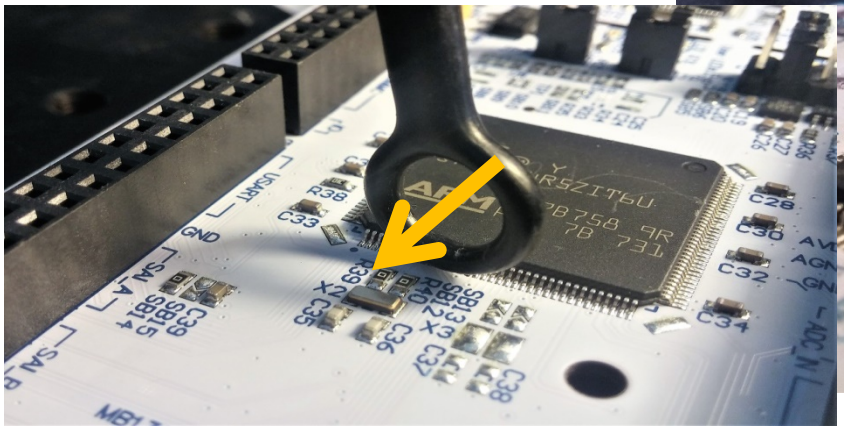
Key sizes: 128, 192, 256 bits. Block size: 128 bits

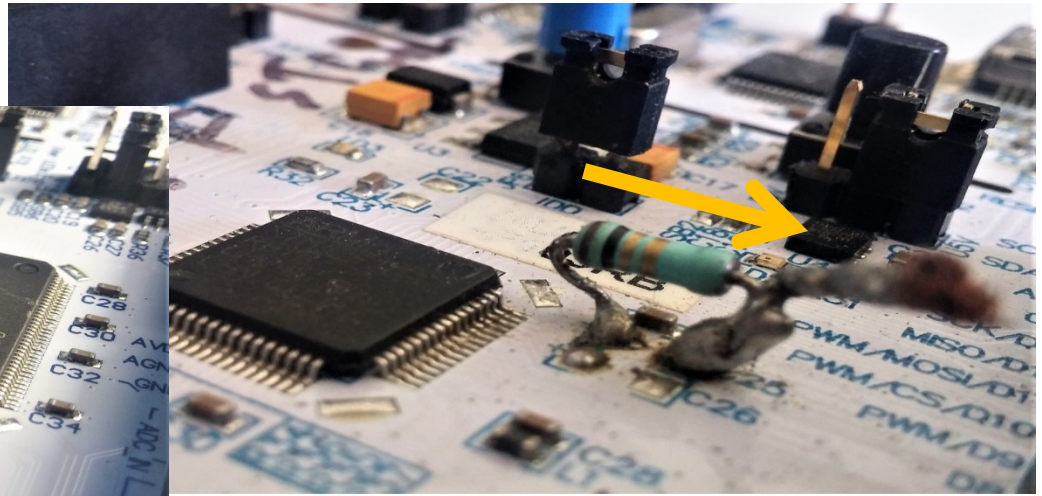
Encryption Process



SCA : Side Channel Attack to Cryptographic MCUs

- Exploiting side channel leakage:
  - Power consumption
  - Electromagnetic Emanation (EM)
  - Acoustic

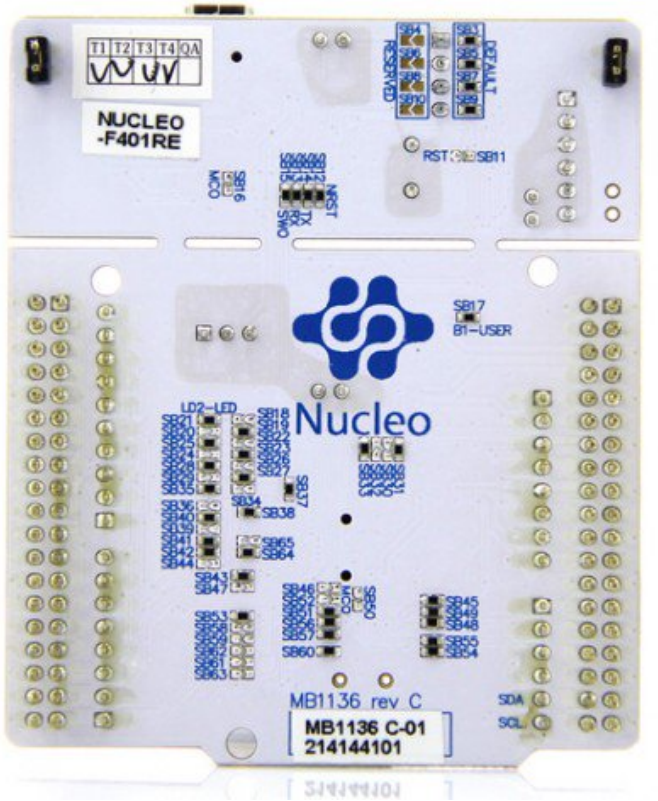


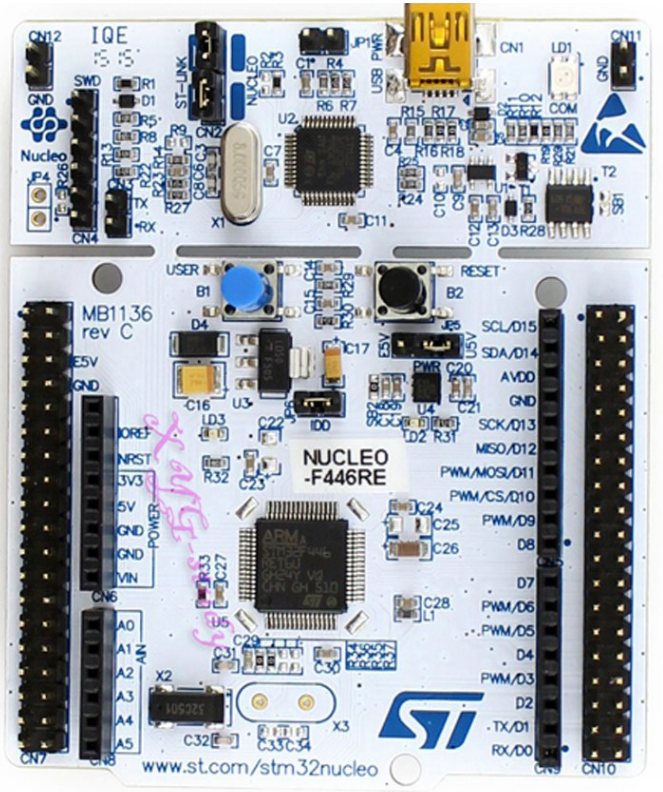


credit Ehsan Aerabi

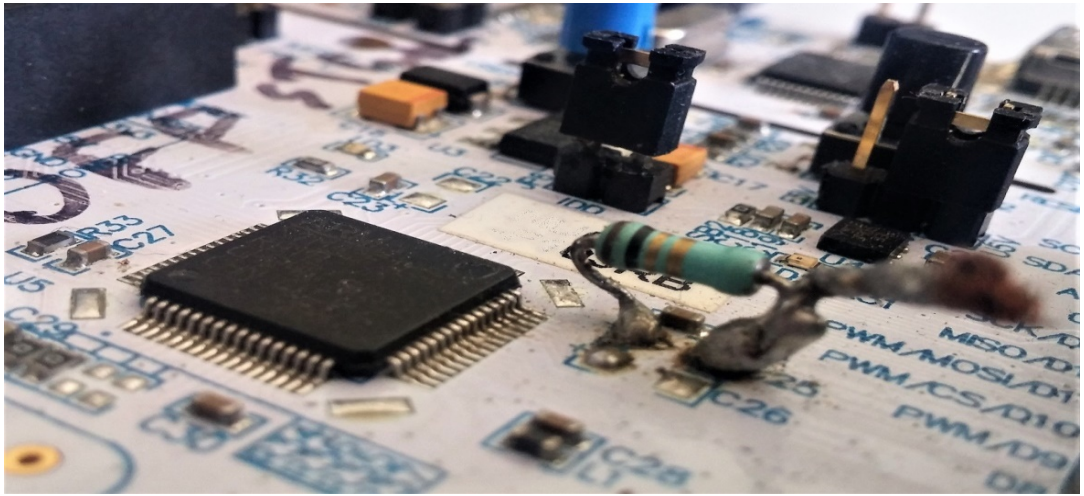
PREPARING THE BOARD

Nucleo ARM Board

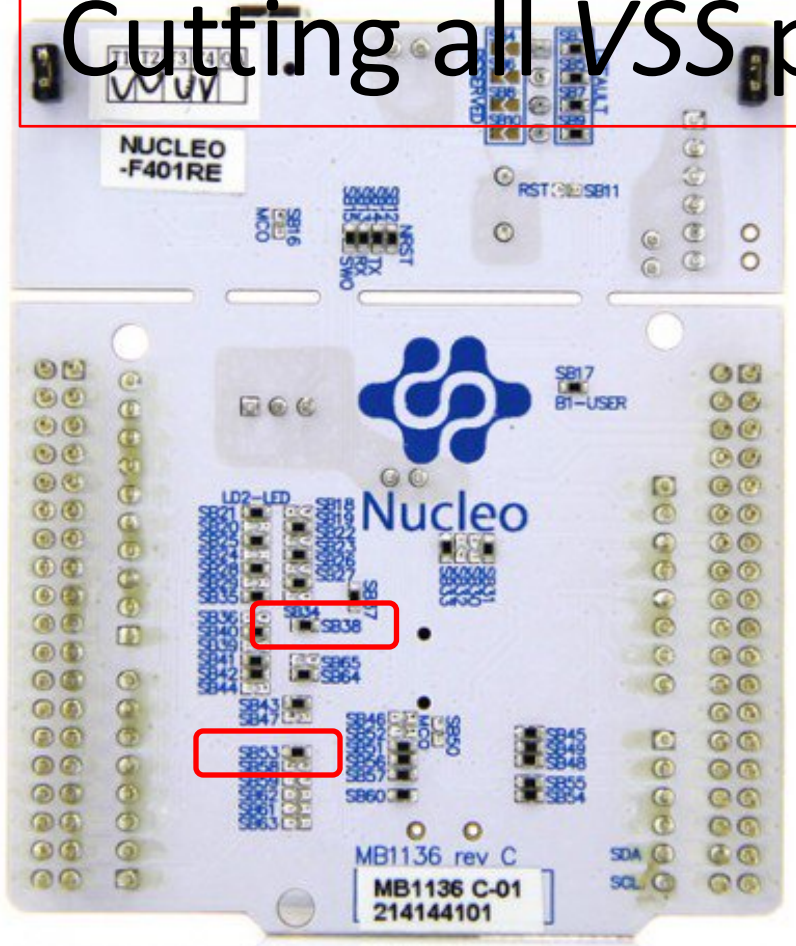


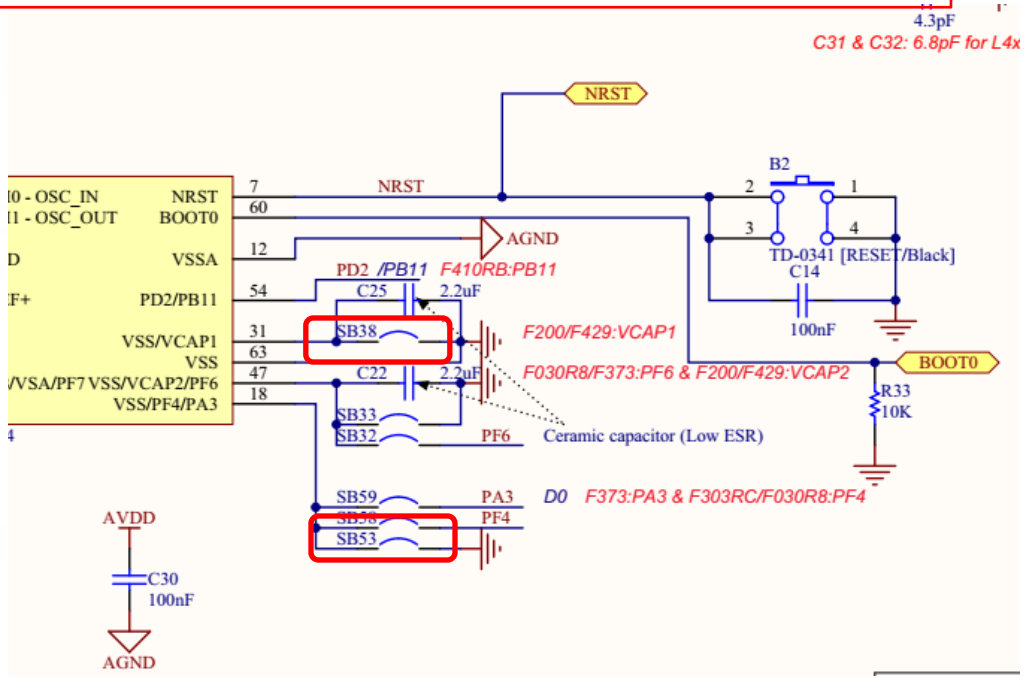


A shunt resistor to capture the power consumption

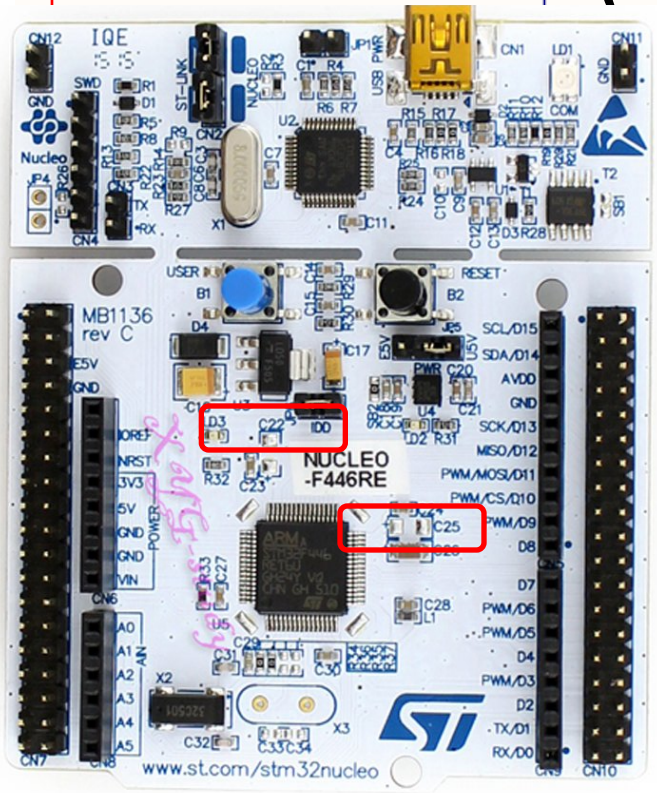


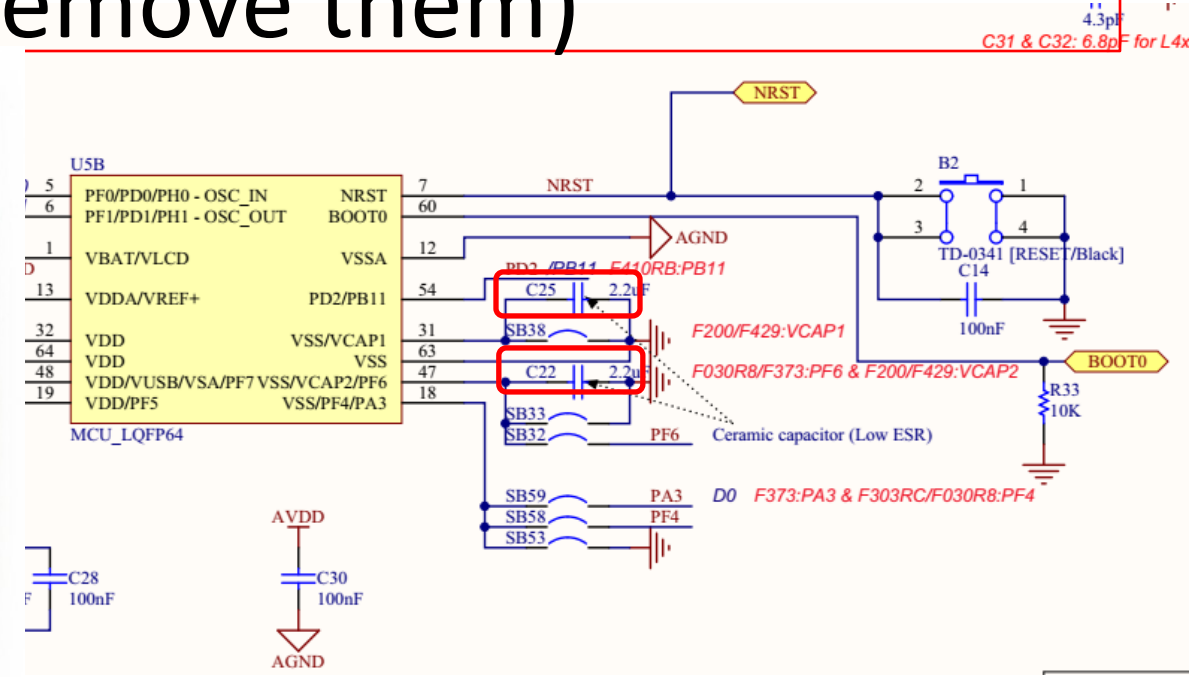
Cutting all *VSS* paths to the ground



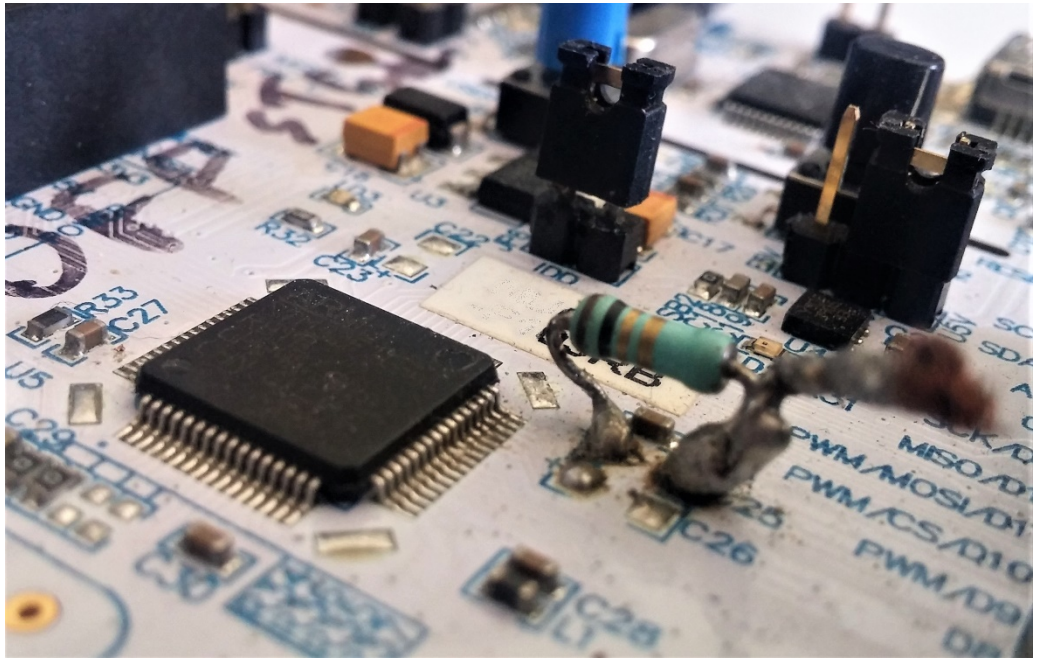


Capacitors are like filters (remove them)





A 1-12 Ω shunt instead of C25



PROGRAMMING THE STM32 AND CAPTURING OSCILLOSCOPE DATA

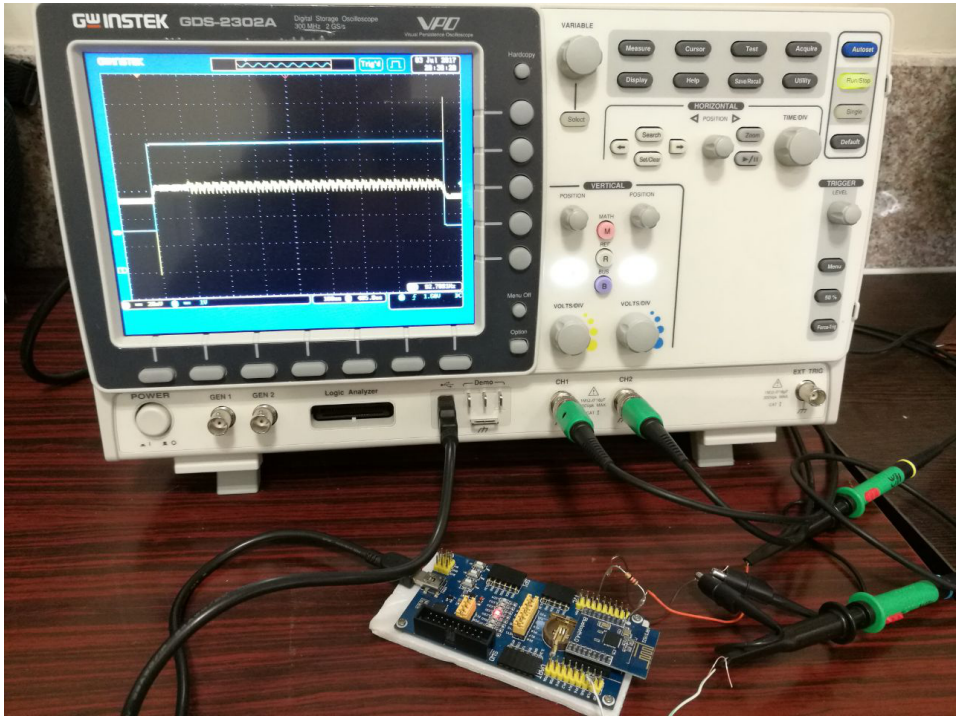
credit Ehsan Aerabi

Place the Trigger

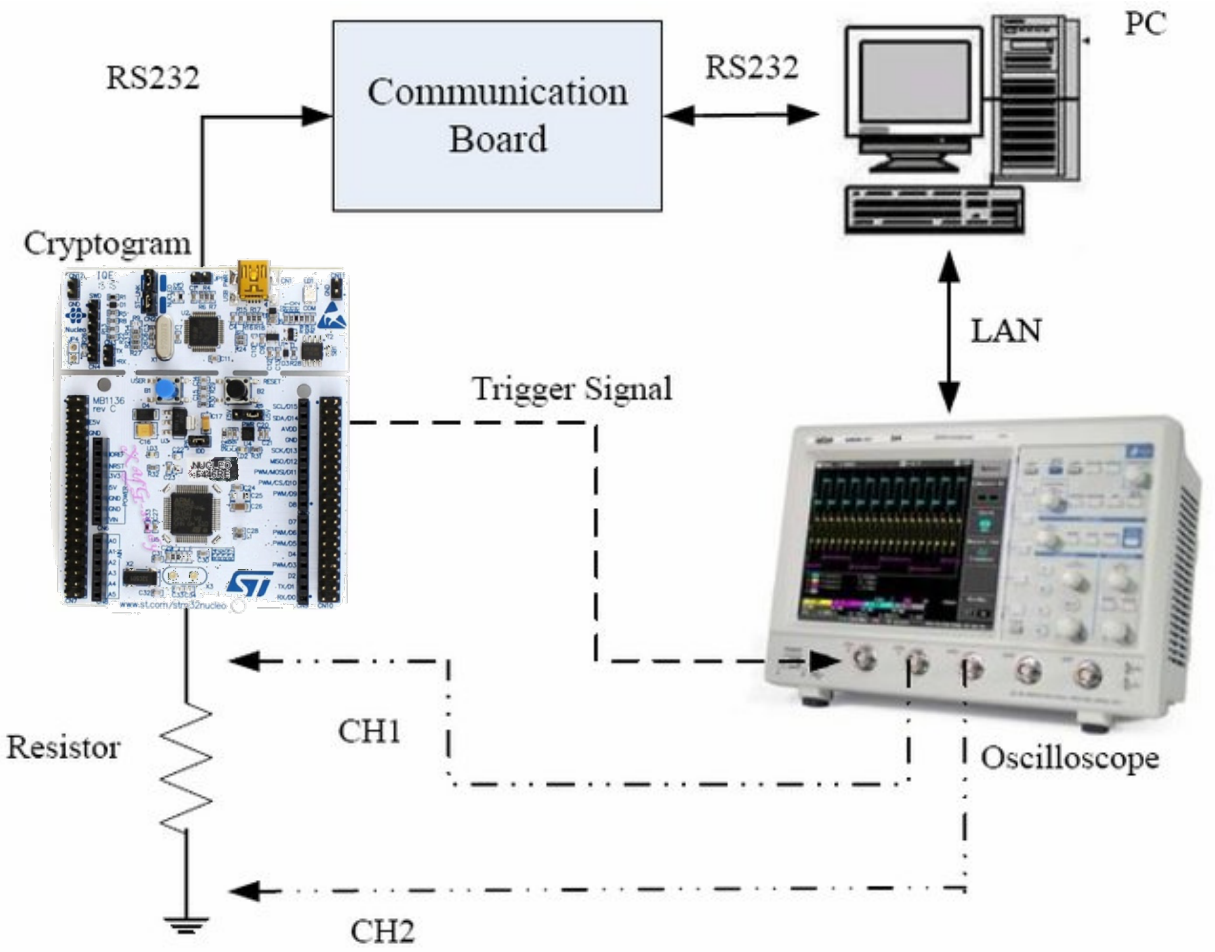
```
static void Cipher()
{
  uint8_t round = 0;

  // Add the First round key to the state before starting the rounds.
  AddRoundKey(0);

  // There will be Nr rounds.
  // The first Nr-1 rounds are identical.
  // These Nr-1 rounds are executed in the loop below.
  for (round = 1; round < Nr; ++round)
  {
    SubBytes();
    ShiftRows();
    MixColumns();
    AddRoundKey(round);
  }

  // The last round is given below.
  // The MixColumns function is not here in the last round.
  SubBytes();
  ShiftRows();
  AddRoundKey(Nr);
}
```

```
HAL_GPIO_WritePin(LED2_GPIO_PORT, TRIGGER_PIN_D8, GPIO_PIN_SET);

HAL_GPIO_WritePin(LED2_GPIO_PORT, TRIGGER_PIN_D8, GPIO_PIN_RESET);
```

Effect of the Trigger on Oscope



The Complete Test-bed



Receiving and Encrypting PT, and Transmitting the Cipher

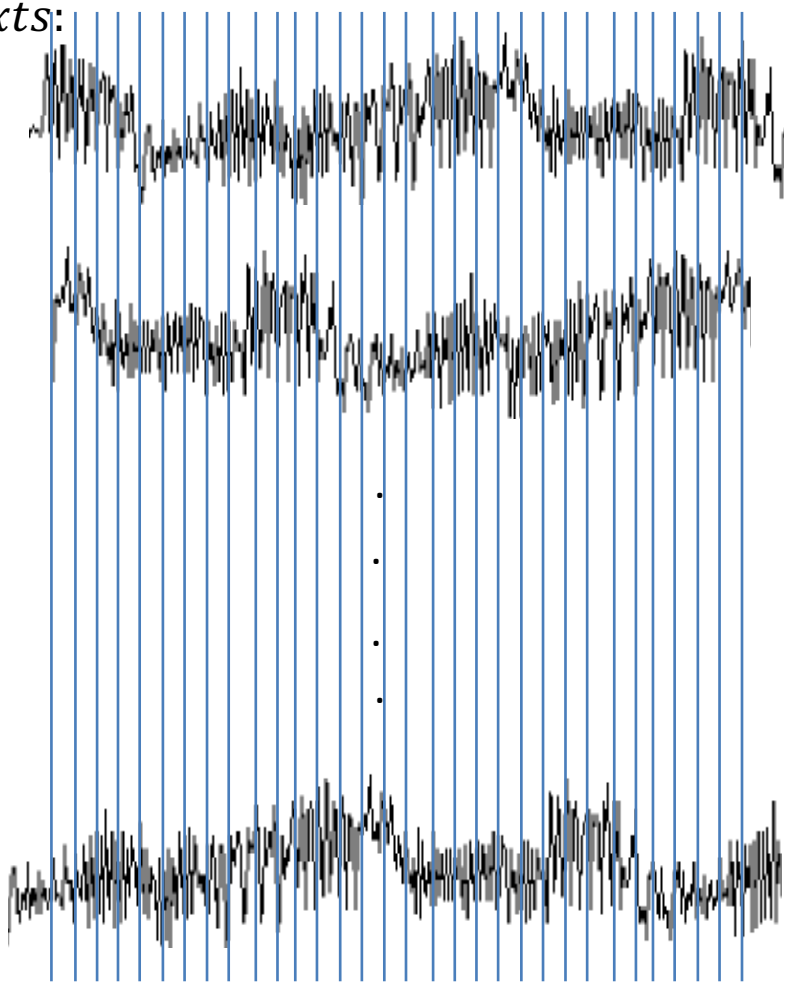
```
HAL_UART_Receive(&UartHandle, (uint8_t *)&keyin, 16, 0xFFFF);
HAL_UART_Receive(&UartHandle, (uint8_t *)&datain, 16, 0xFFFF);
AES_ECB_encrypt(datain, keyin, buffer, 16);
HAL_UART_Transmit(&UartHandle, (uint8_t *)&buffer, 16, 0xFFFF);
```

$Plaintexts$:

d_1
d_2
d_3
.
.
.
d_D

Plaintexts:

d_1
d_2
d_3
.
.
.
d_D



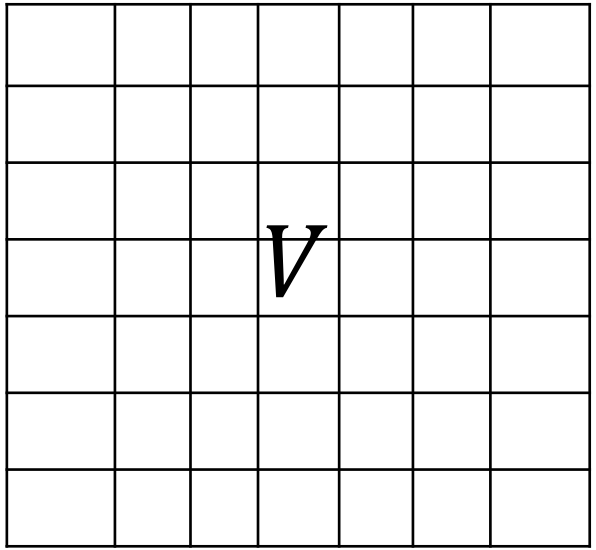
$t_{1,1}$						
				T		

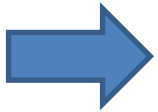
data points

```
clear all
load('attack_data.mat');
load('constants.mat');
%##########################################################

% Prepare data
D = plaintexts_SCA(:,1);

% Prepare power traces
traces = datapoints2(1:samples,:);
```

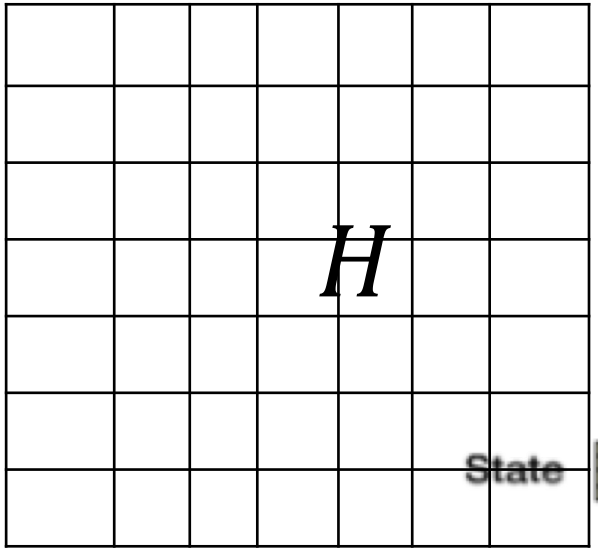
k_1	k_2	.	.	.					k_K

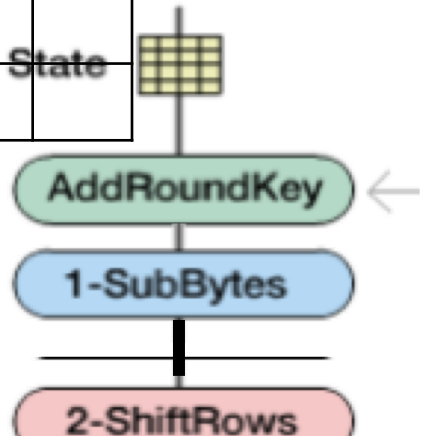
d_1
d_2
d_3
.
.
d_D





Power
Model





SBOX (d_i xor k_j)

```
% Prepare keys
K = uint8(0:255);

% Calculate hypothetical intermediate values

% Calculate hypothetical power consumption
```

k_1	k_2	.	.		k_K

Correlation Coefficient

$h_{1,1}$	$h_{1,2}$	.	.	.	$h_{1,K}$
$h_{2,1}$					
.					
.		H			
.					
$h_{D,1}$					$h_{D,K}$

Corr (,)

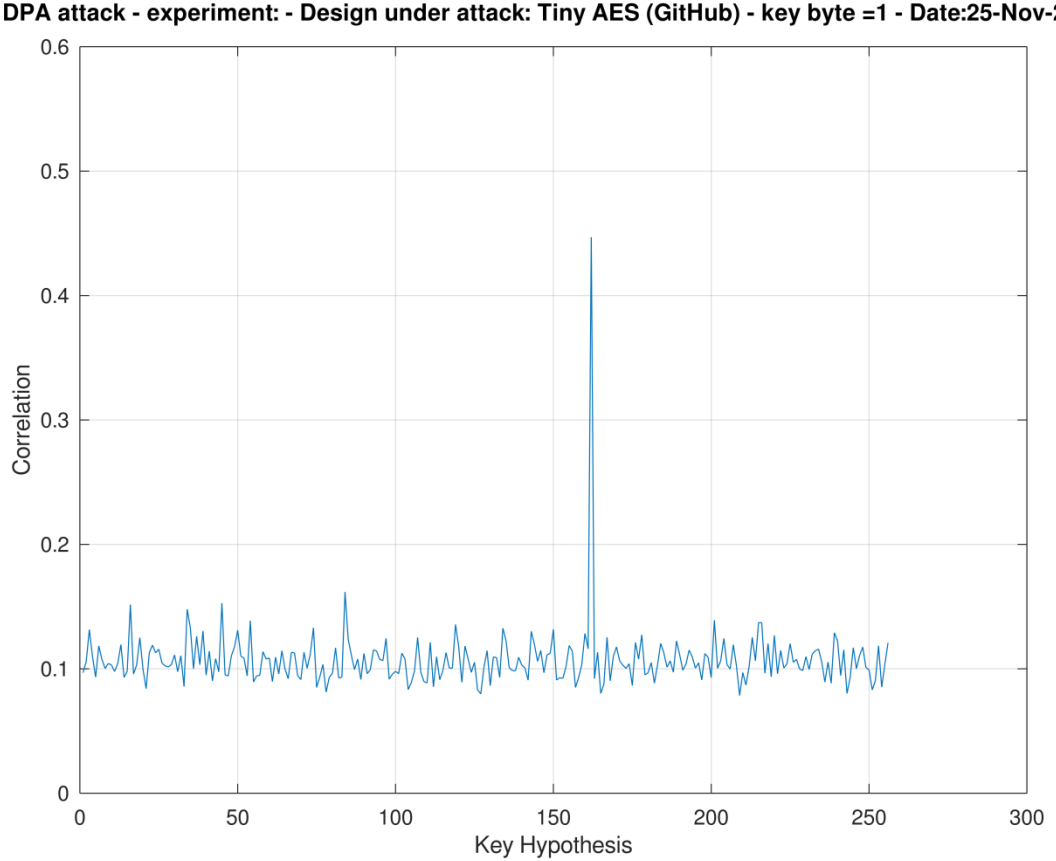
$t_{1,1}$	$t_{1,2}$	.	.	.		$t_{1,T}$
$t_{2,1}$						
.						
.			T			
.						
$t_{D,1}$						$t_{D,T}$

		R				

```
% Calculate the correlation
```

```
% Find the maximum correlation and corresponding row and
column in matrix R
```

```
% Find Maximum correlation for each key and plot it
% write your code here...
```


DPA attack - experiment: - Design under attack: Tiny AES (GitHub) - key byte =1 - Date:25-Nov-2019
Correlation
Key Hypothesis
0
0.1
0.2
0.3
0.4
0.5
0.6
0
50
100
150
200
250
300

Pearson Correlation Coefficient

$$r_{i,j} = \frac{\sum_{d=1}^{D}\left(\left(h_{d,i} - \bar{h}_i\right) \cdot \left(t_{d,j} - \bar{t}_j\right)\right)}{\sqrt{\sum_{d=1}^{D}\left(h_{d,i} - \bar{h}_i\right)^2 \cdot \sum_{d=1}^{D}\left(t_{d,j} - \bar{t}_j\right)^2}}$$

$h_{1,1}$	$h_{1,2}$	.	.	.	$h_{1,K}$
$h_{2,1}$					
.					
.					
.					
$h_{D,1}$					$h_{D,K}$

$t_{1,1}$	$t_{1,2}$	.	.	.		$t_{1,T}$
$t_{2,1}$						
.						
.						
.						
$t_{D,1}$						$t_{D,T}$