设备皆有故障!

全面沦陷:噪声攻击与IOT设备的10+年战争

riscure

目录

概述

FI 的发展和现状

FI的实际应用

FI的预防

问题在哪?

```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(pin); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;


    printf("Please, introduce PIN\r\n");
    gets(pin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf(tmp);
    return result;
}
```

问题在哪?

```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(pin); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;


    printf("Please, introduce PIN\r\n");
    gets(pin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf(tmp);
    return result;
}
```

问题在哪?

```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(PIN); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;



    printf("Please, introduce PIN\r\n");
    fgets(pin,sizeof(pin),stdin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf("%s",tmp);
    return result;
}
```

它现在是否安全?

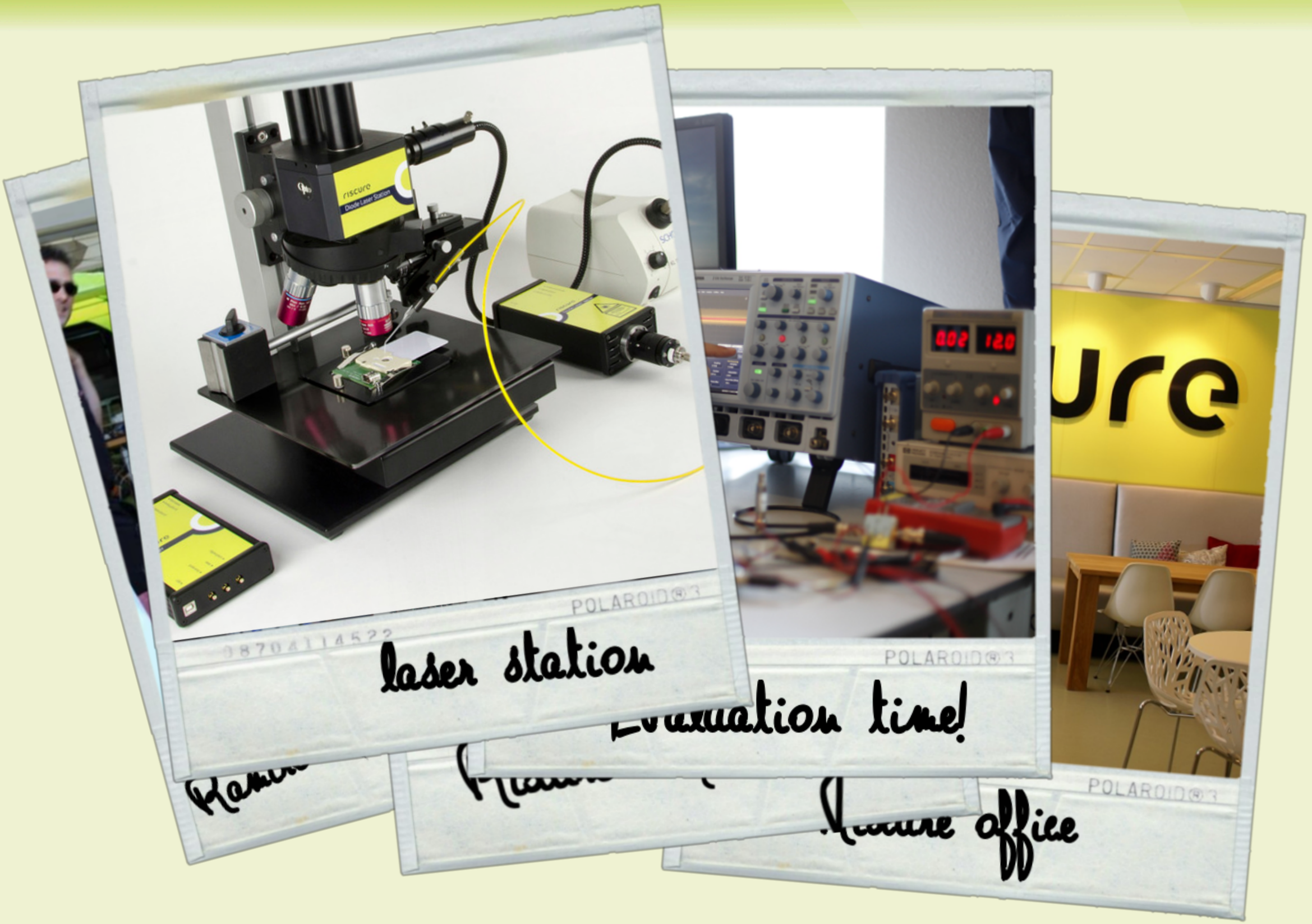
问题在哪?

```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(PIN); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;


    printf("Please, introduce PIN\r\n");
    fgets(pin,sizeof(pin),stdin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf("%s",tmp);
    return result;
}
```

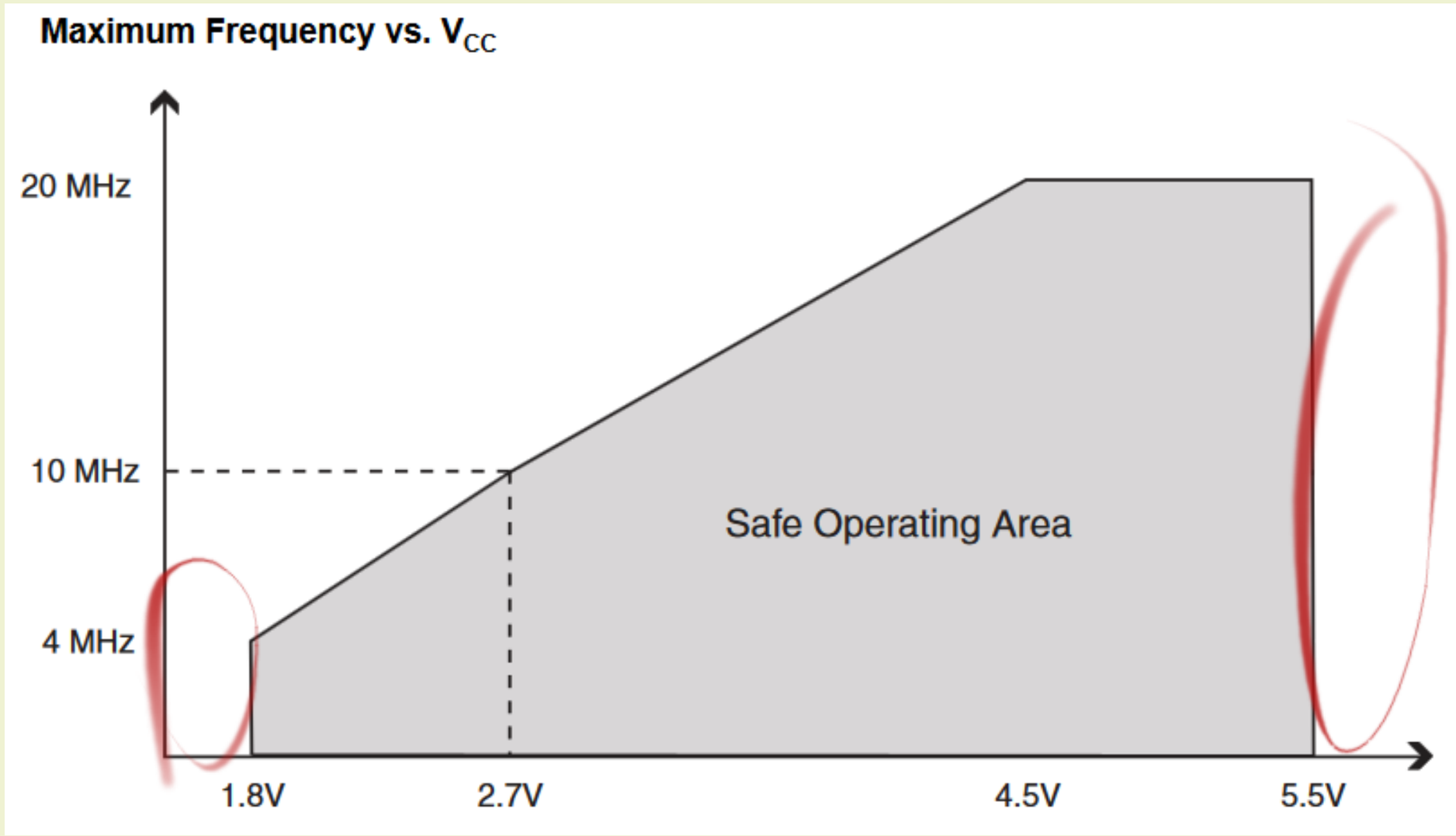
它现在是否安全?

我是谁?

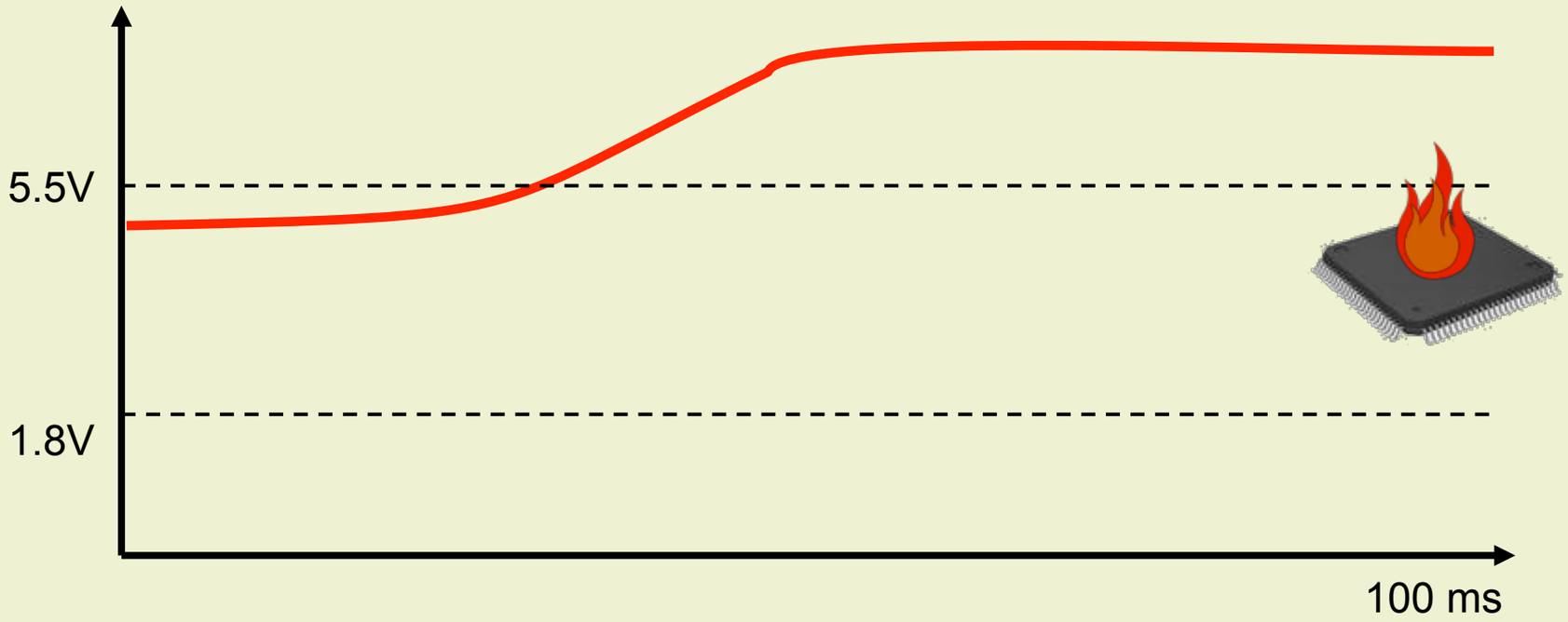

riscure
Diode Laser Station
laser station
Evaluation time!
office

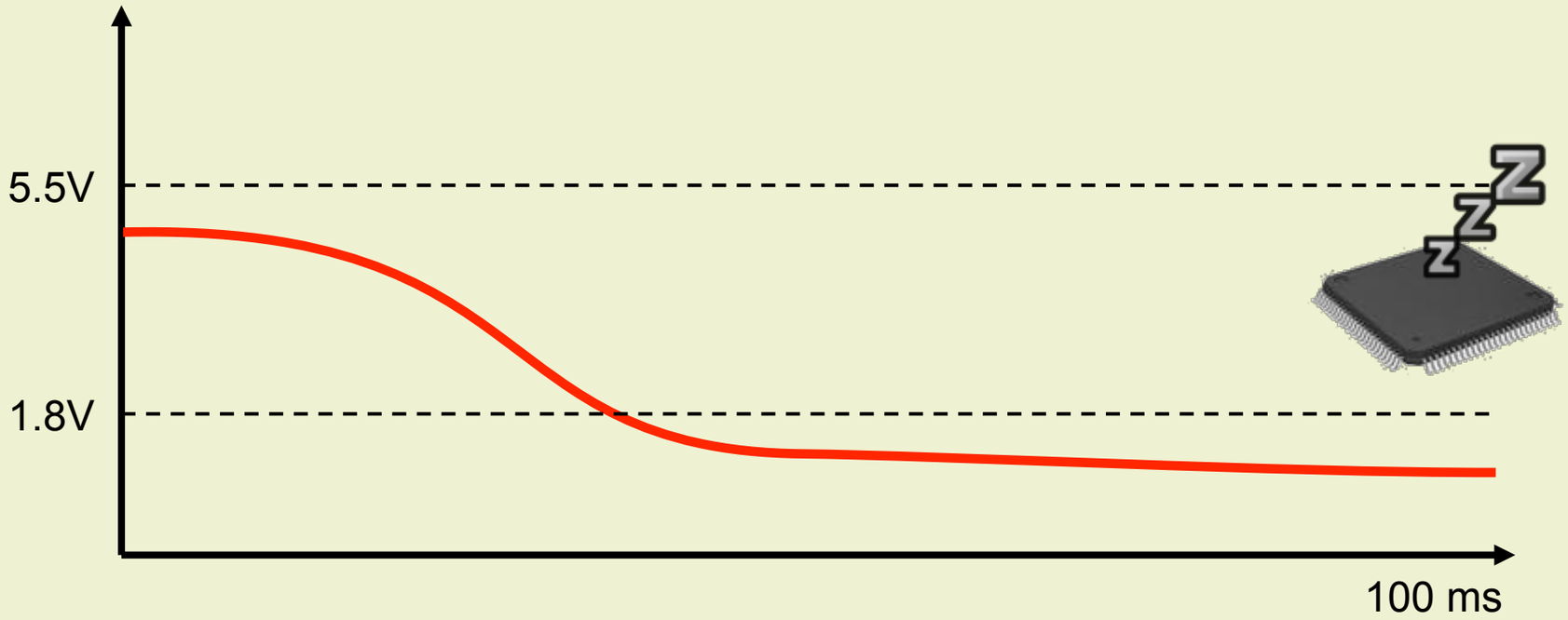
示例

故障注入攻击

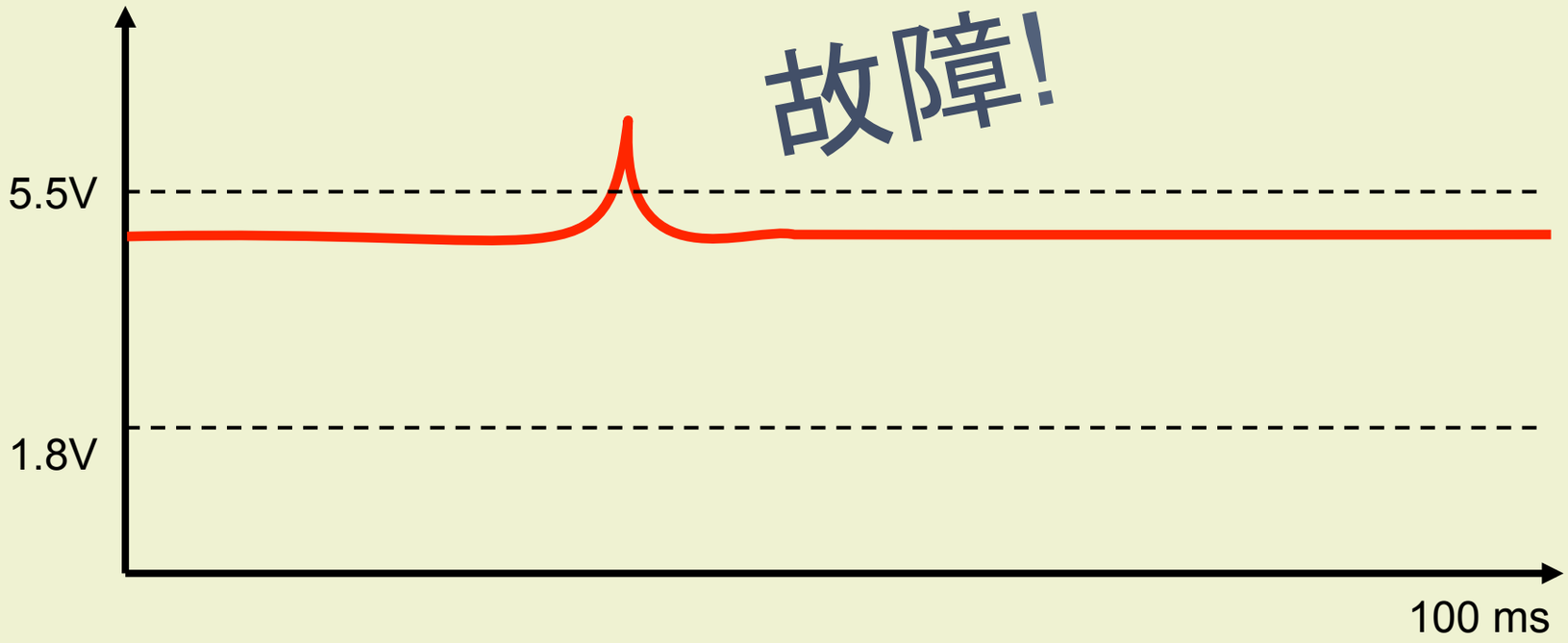

Maximum Frequency vs. V_{CC}
20 MHz
10 MHz
4 MHz
Safe Operating Area
1.8V
2.7V
4.5V
5.5V

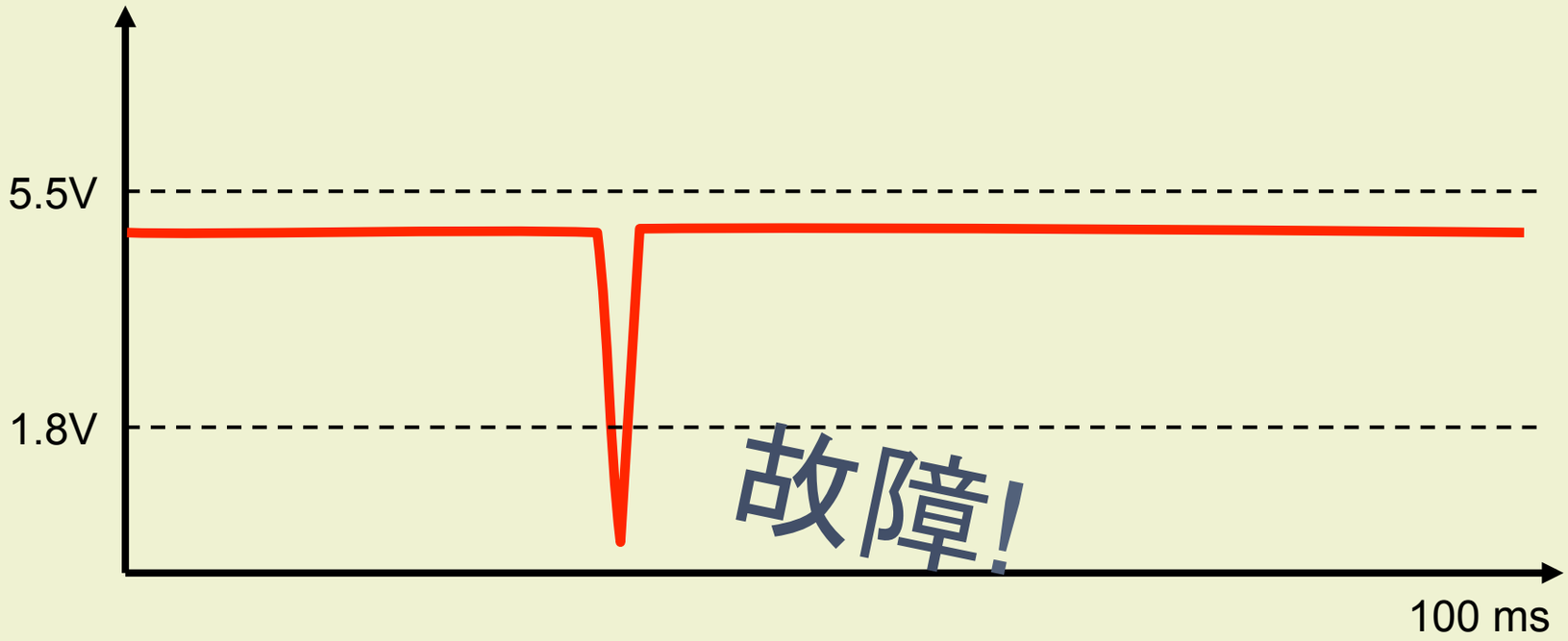
电压故障



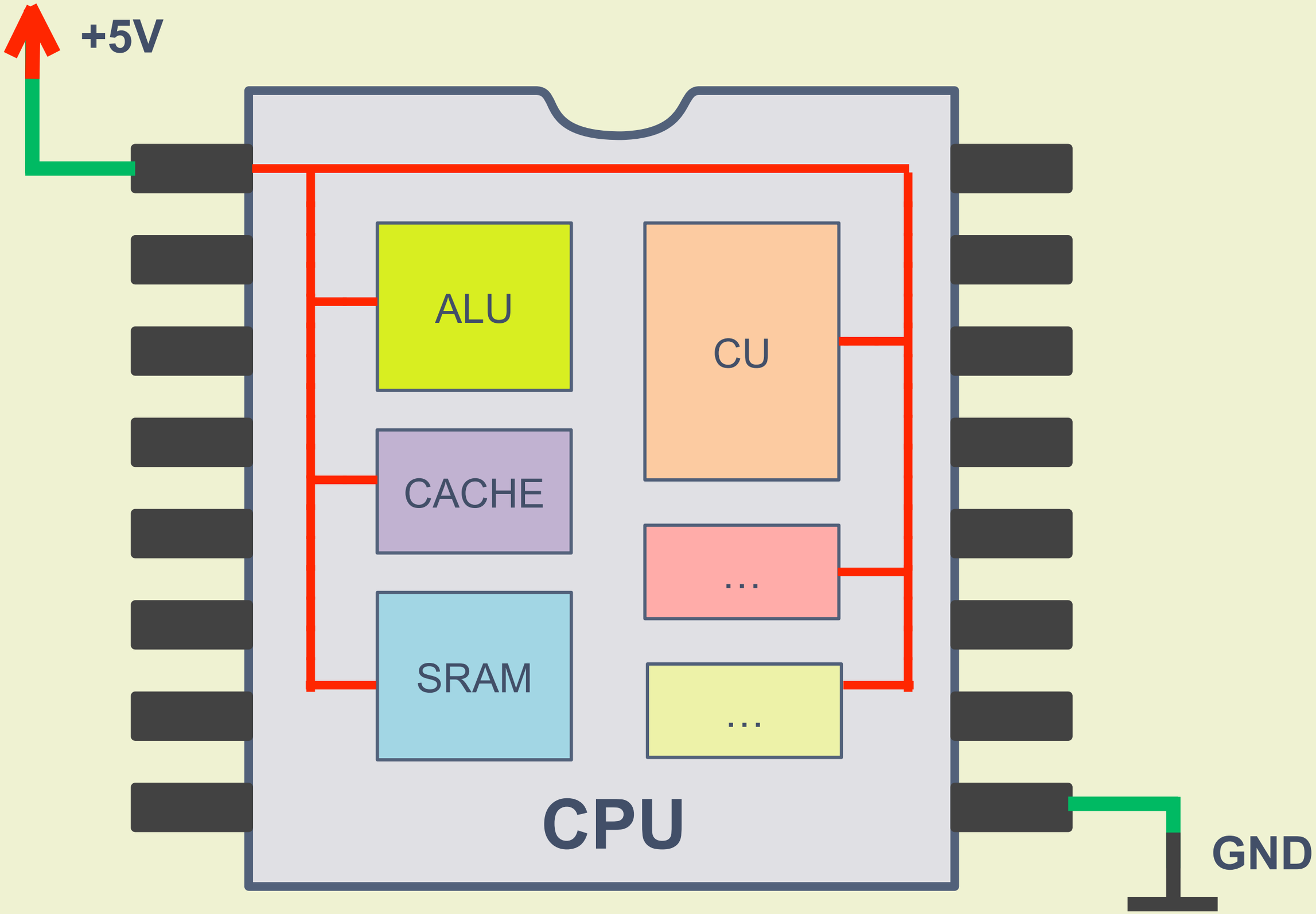


电压故障

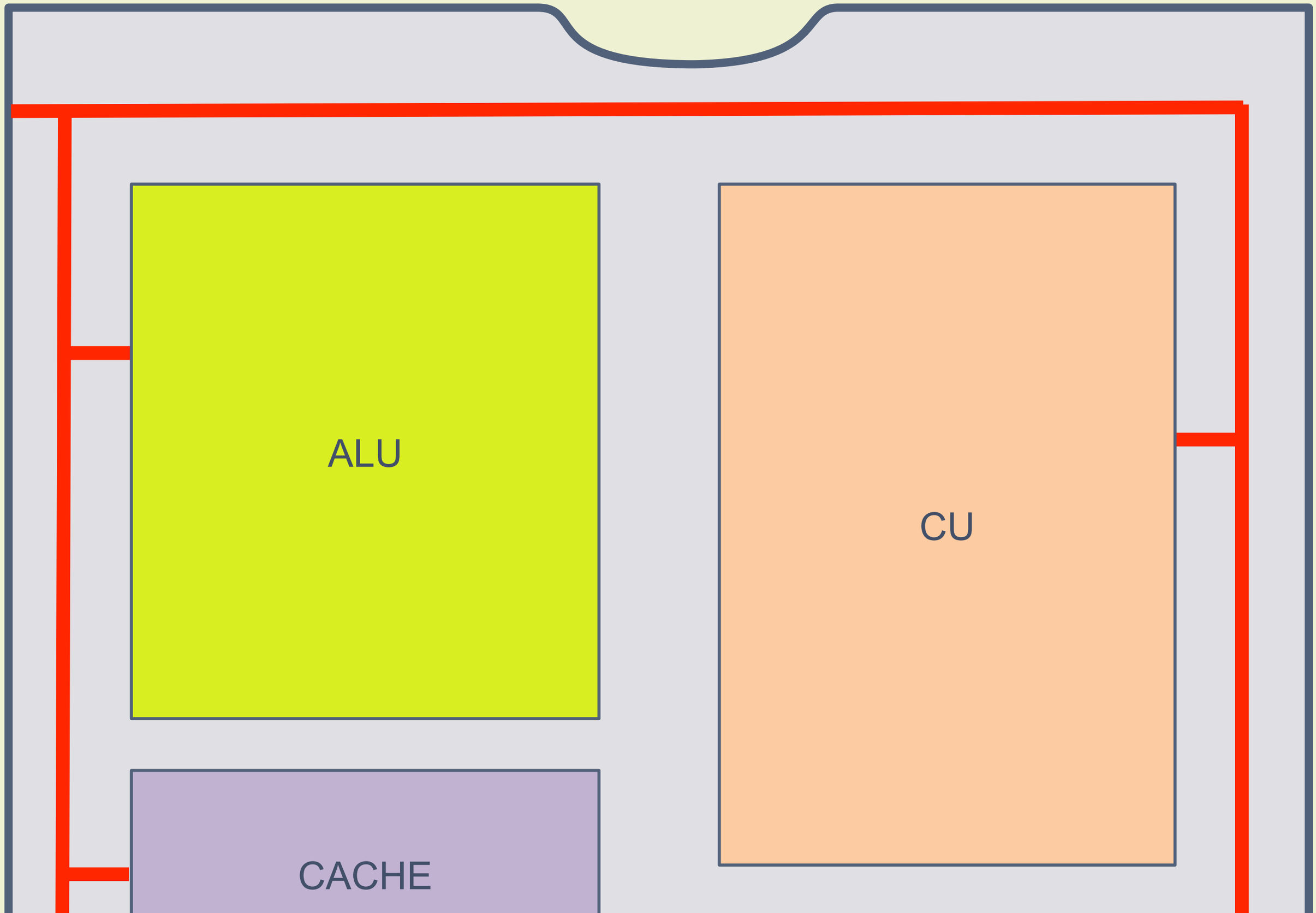




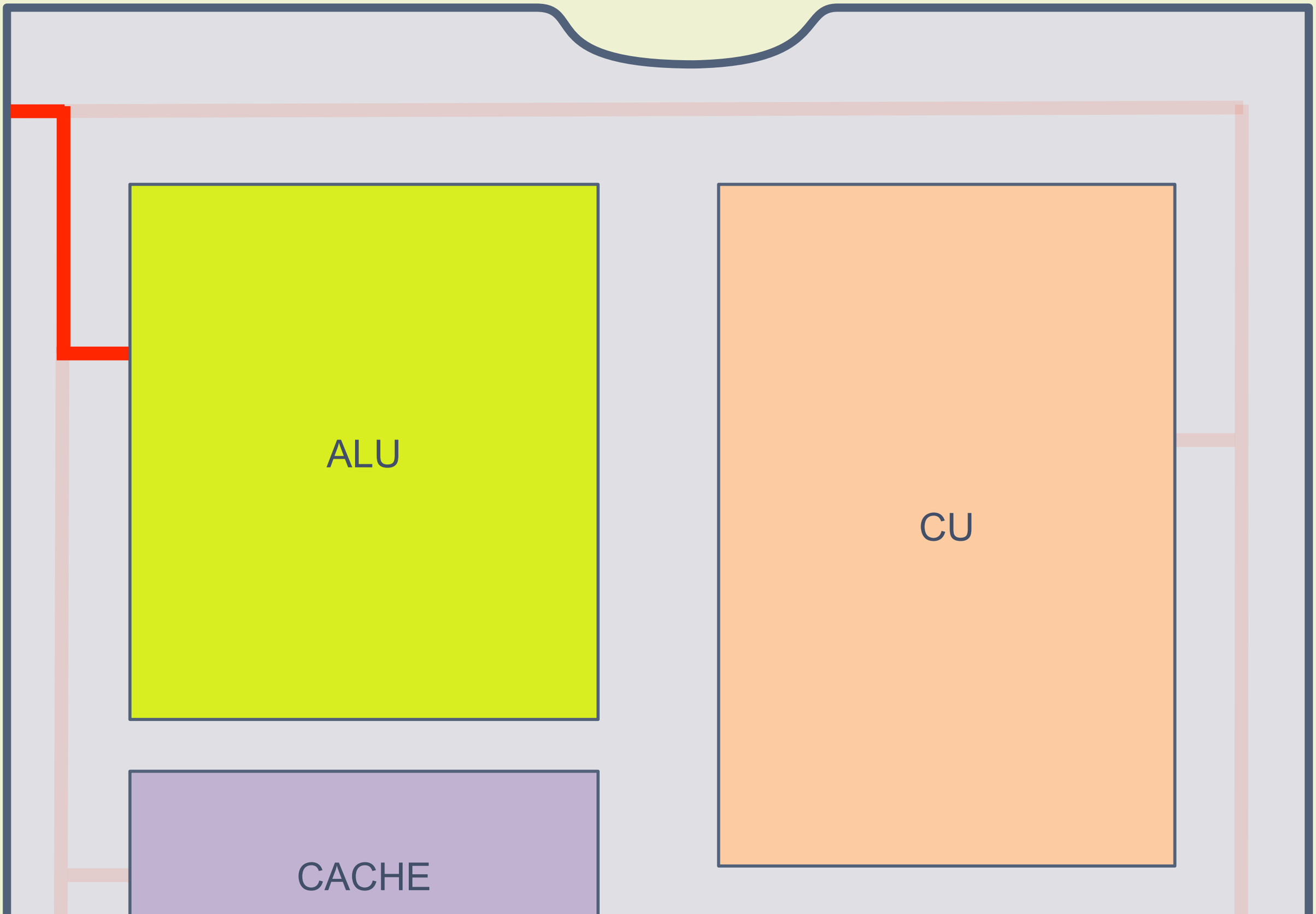
FI如何工作？



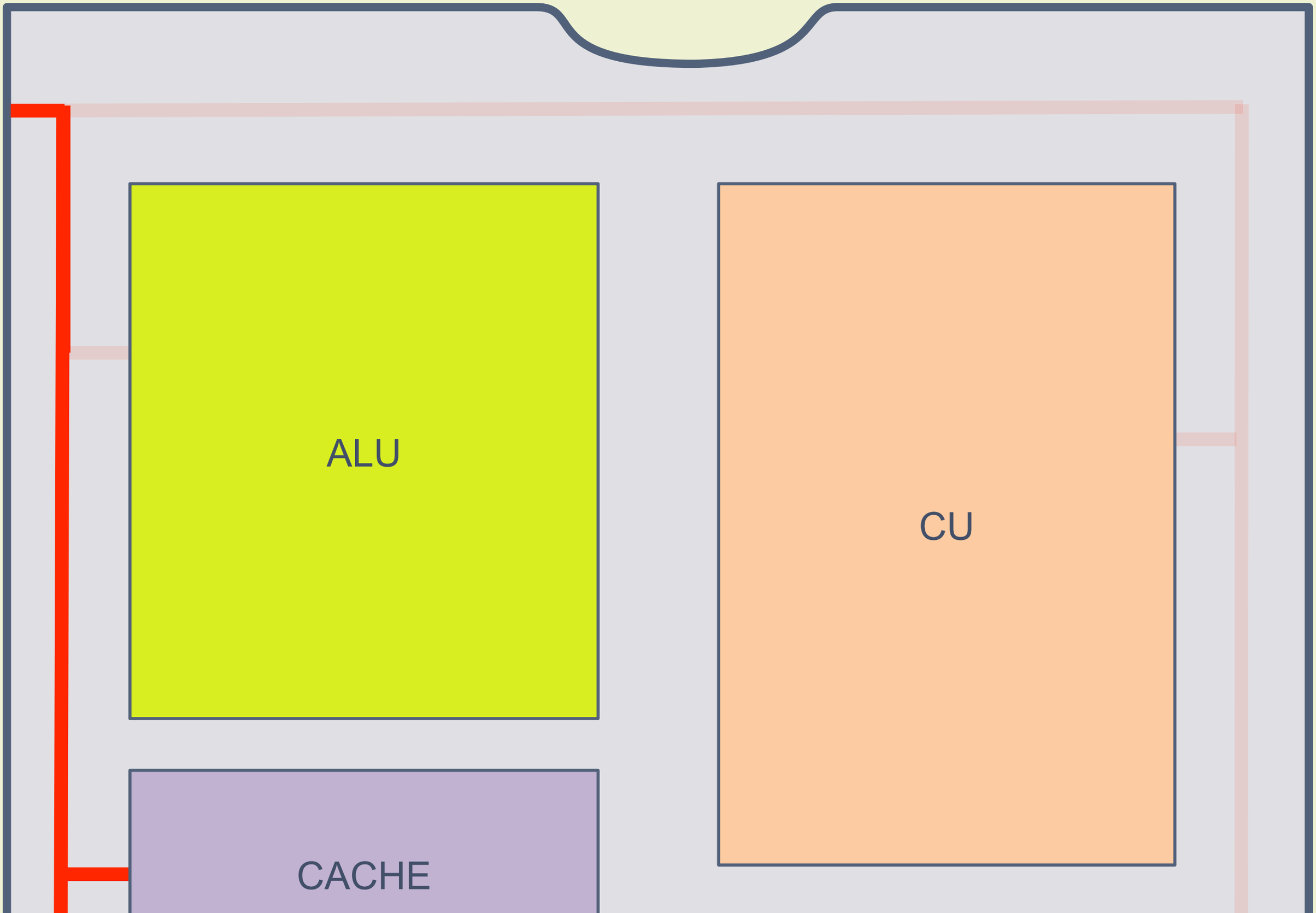
FI如何工作?



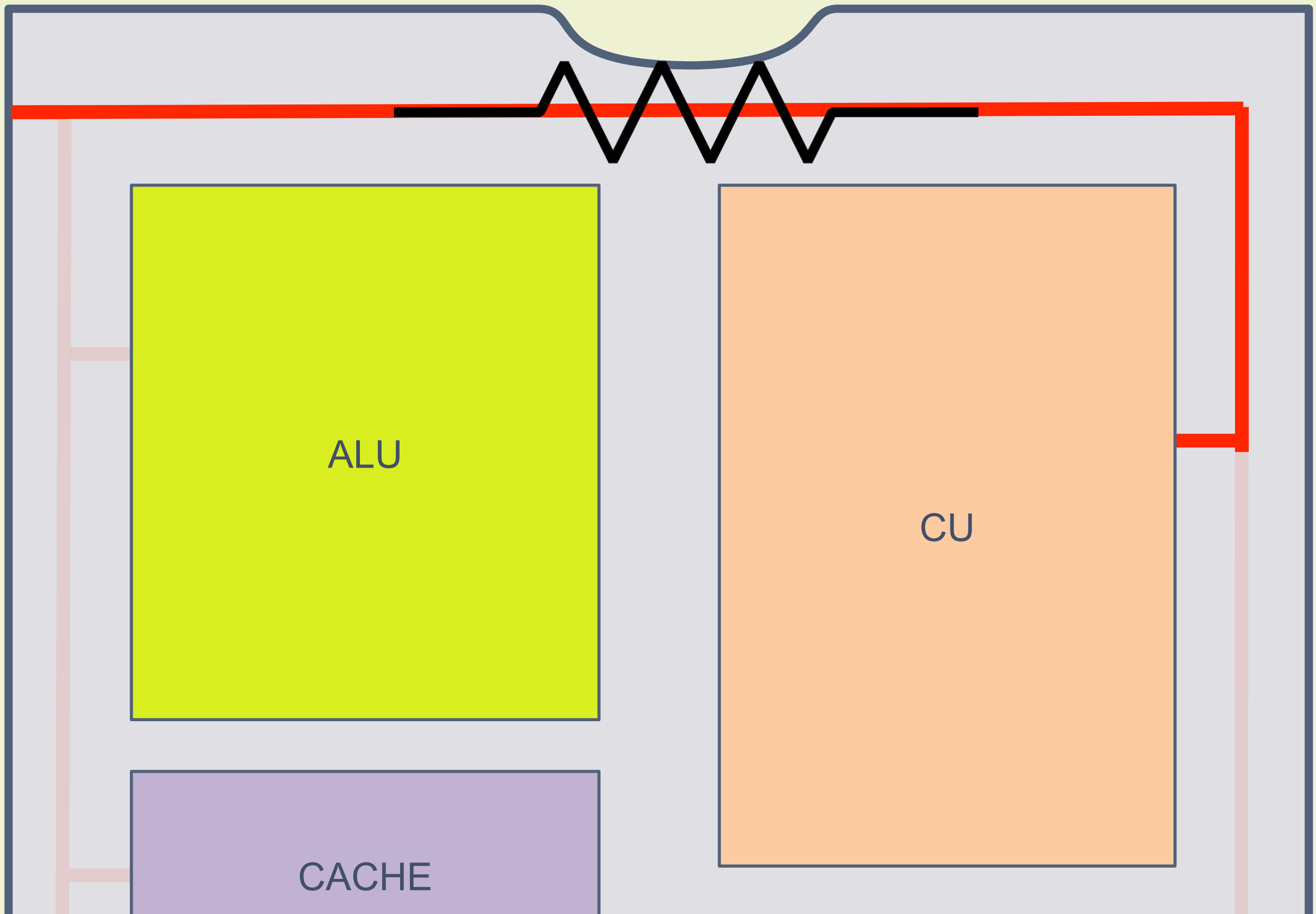
FI如何工作?



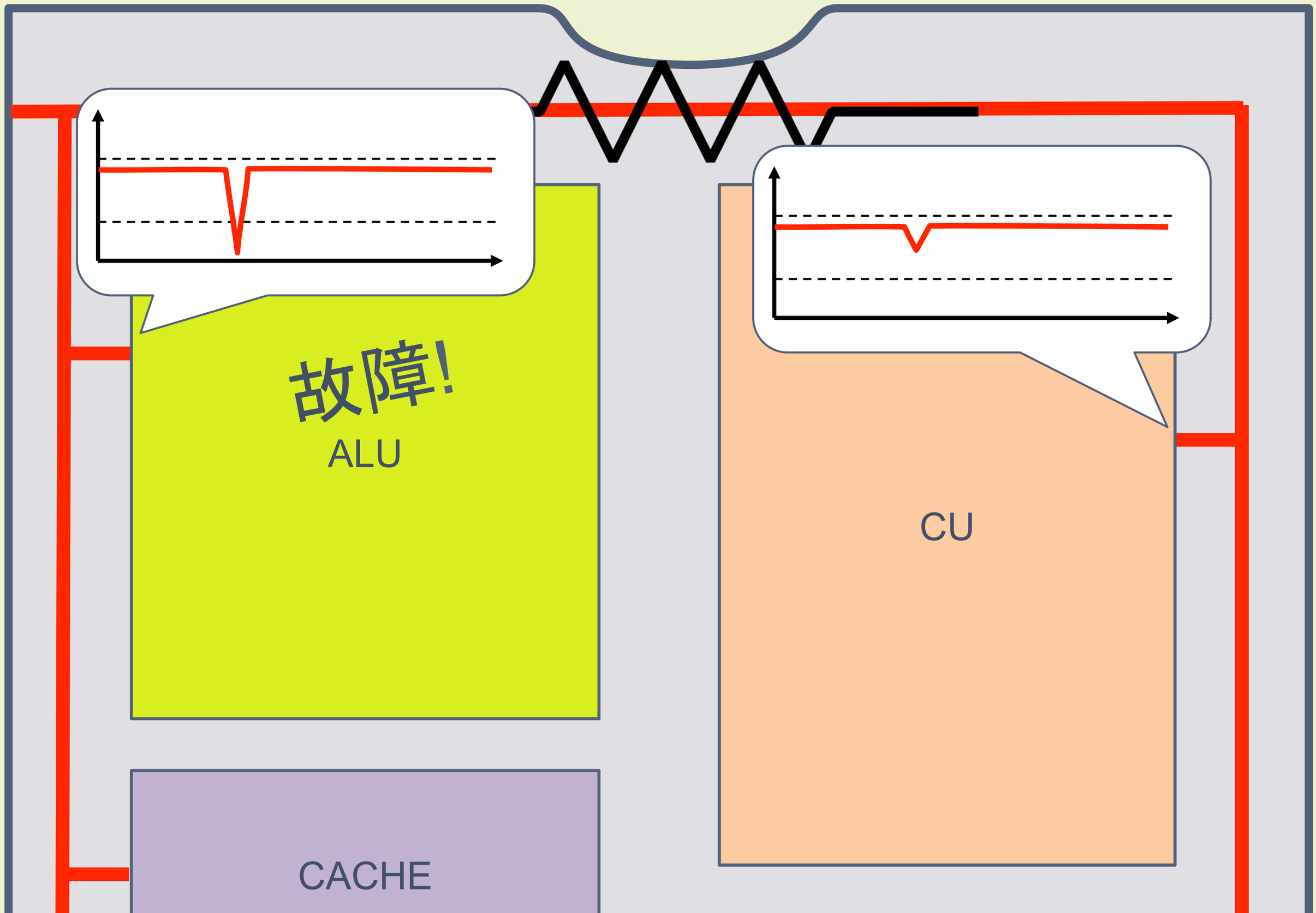
FI如何工作?



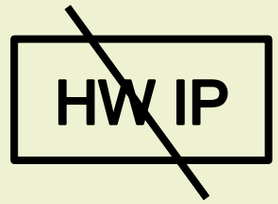
FI如何工作?



FI如何工作?



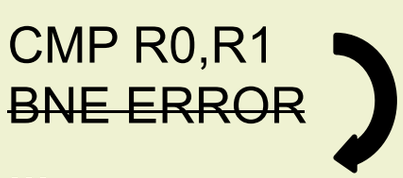
故障的影响



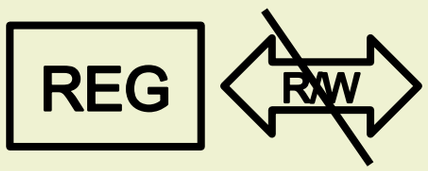
设备失灵



位翻转



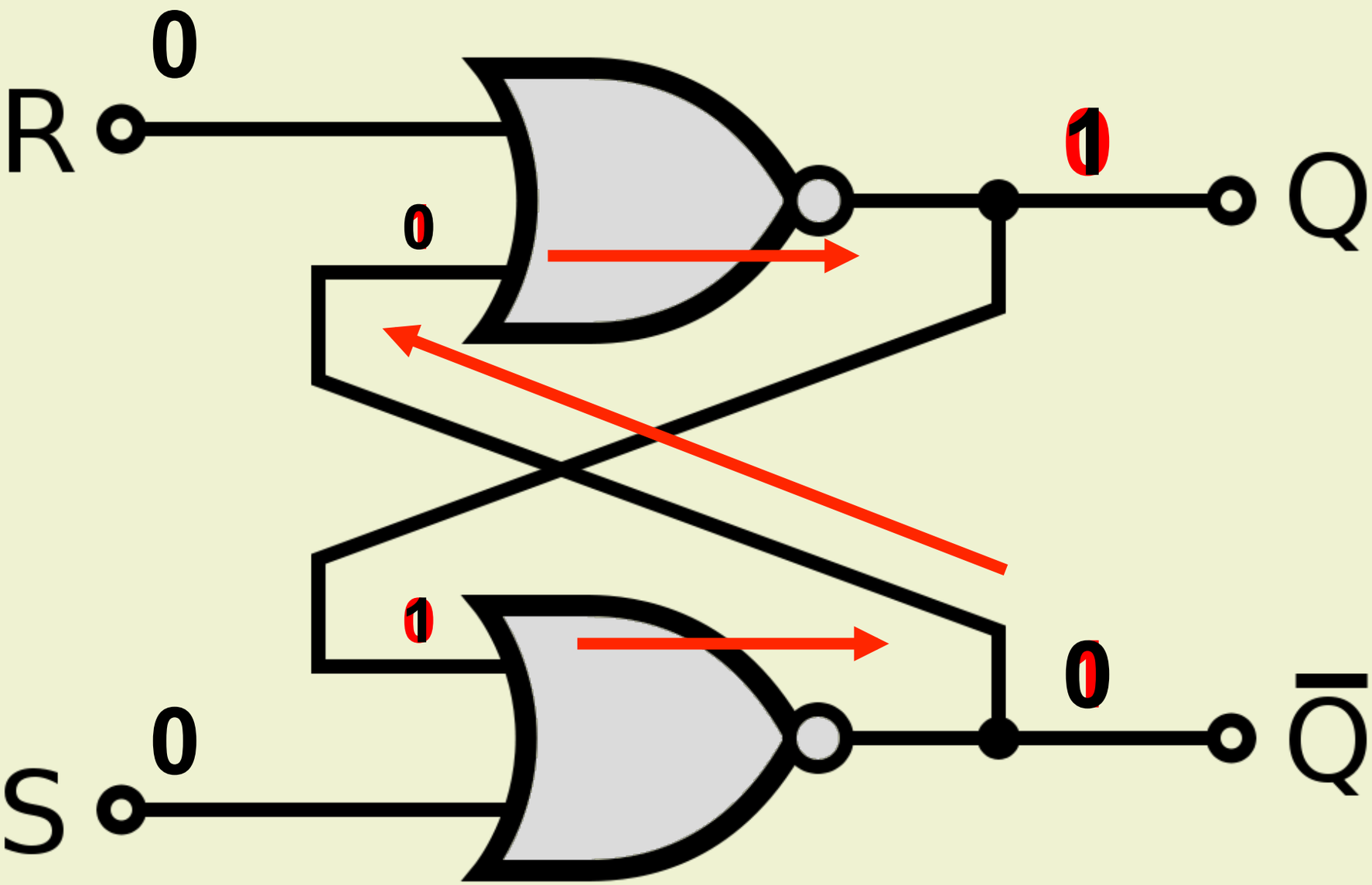
跳过/破坏指令



阻碍读/写

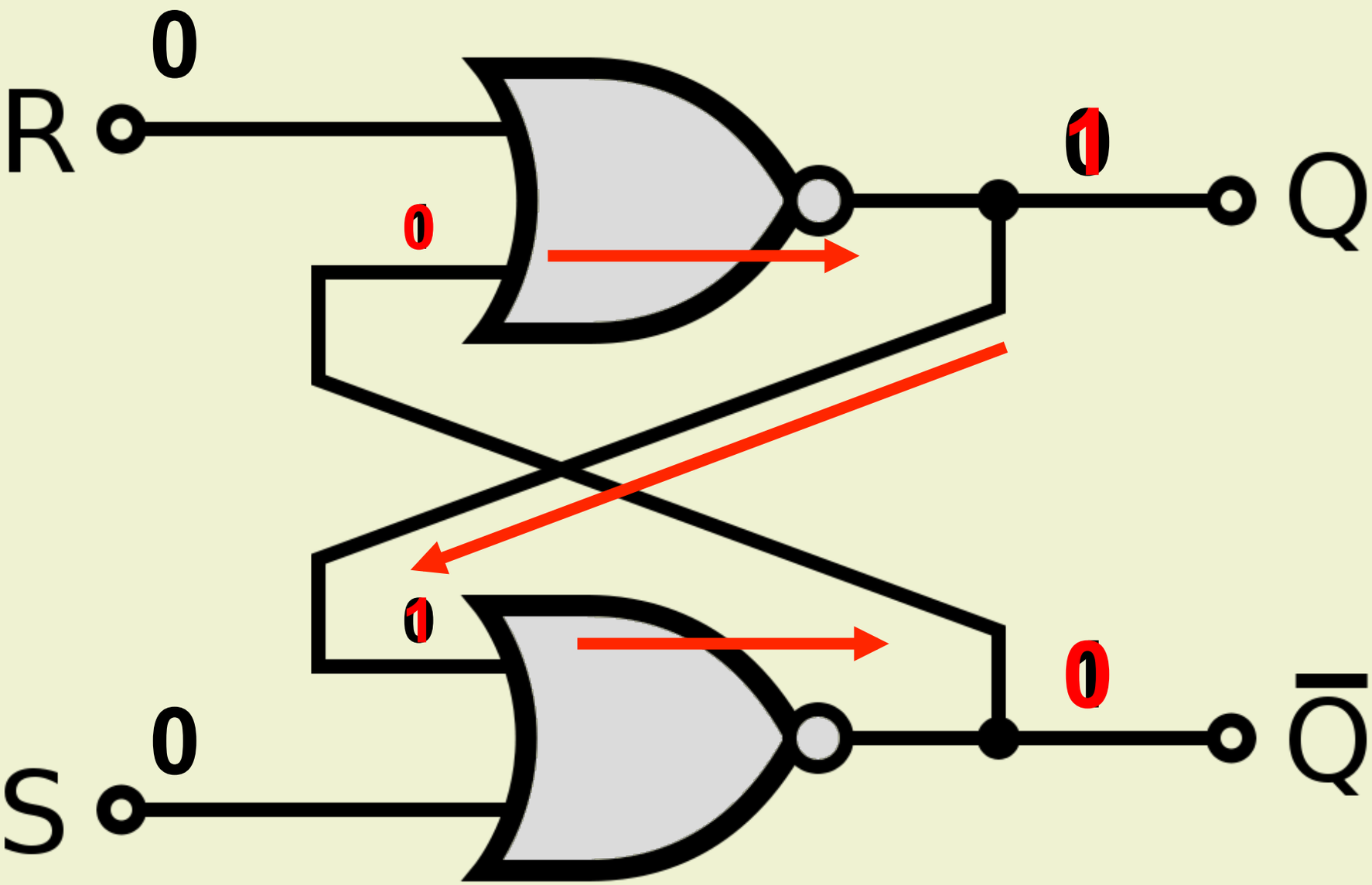
位翻转

1 ⟶ 0



位翻转

$0 \longrightarrow 1$



跳过/破坏指令

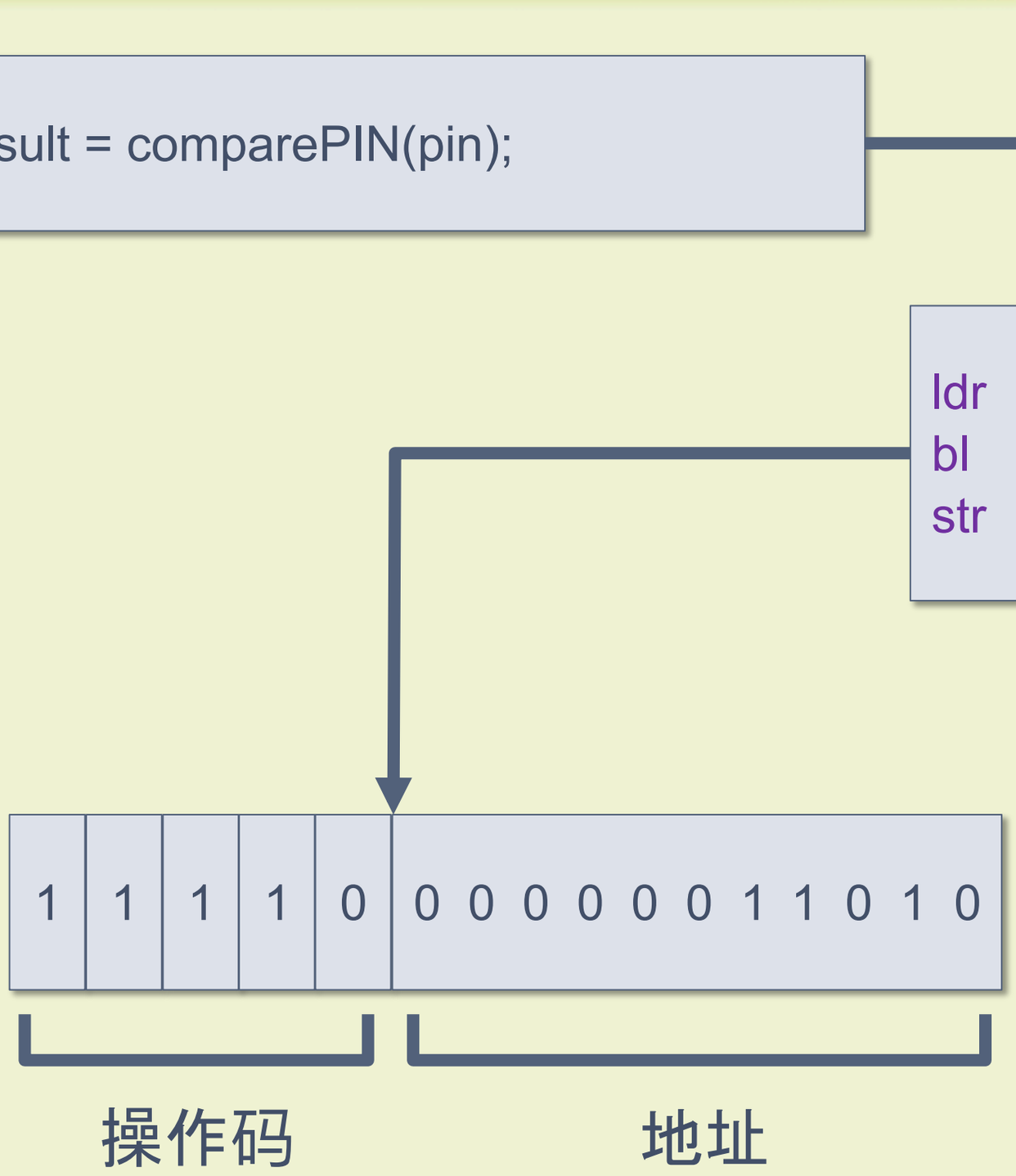
```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(PIN); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;



    printf("Please, introduce PIN\r\n");
    fgets(pin,sizeof(pin),stdin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf("%s",tmp);
    return result;
}
```

跳过/破坏指令

```
result = comparePIN(pin);
```

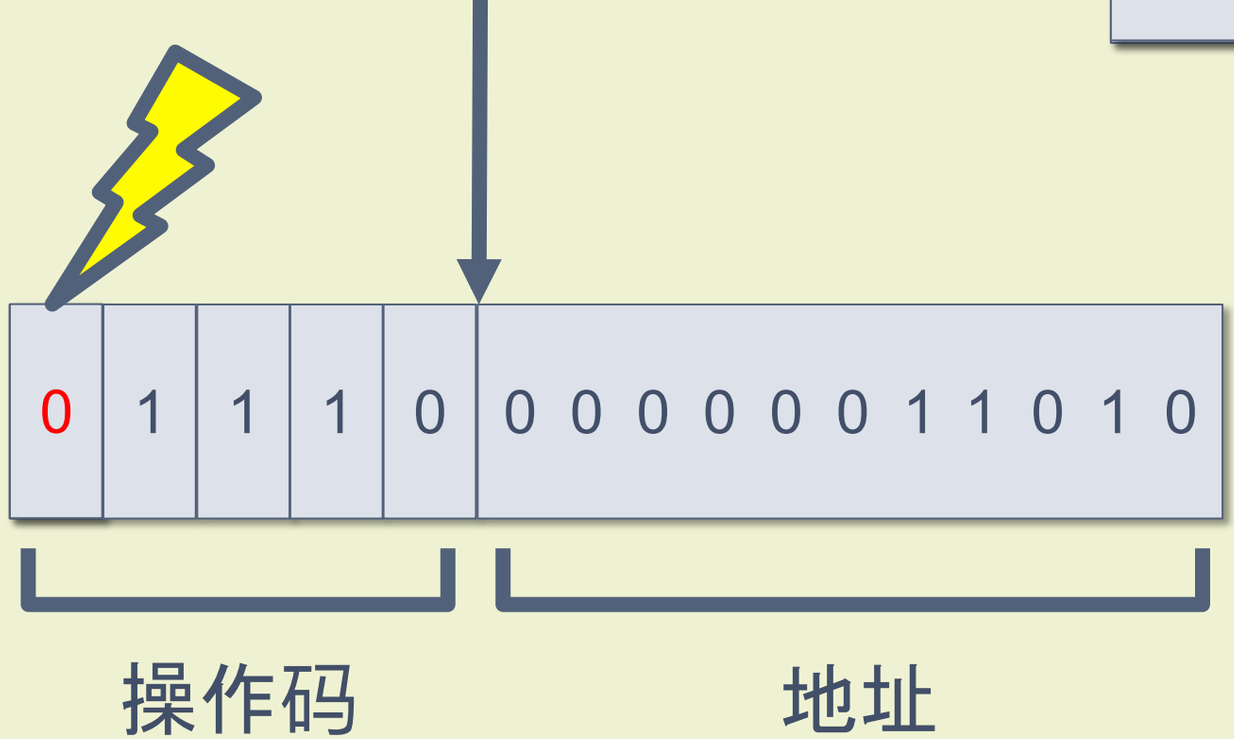
```
ldr   r0, [sp, #pin]
bl    comparePIN
str   r0, [sp, #result]
```



跳过/破坏指令

result = pin;

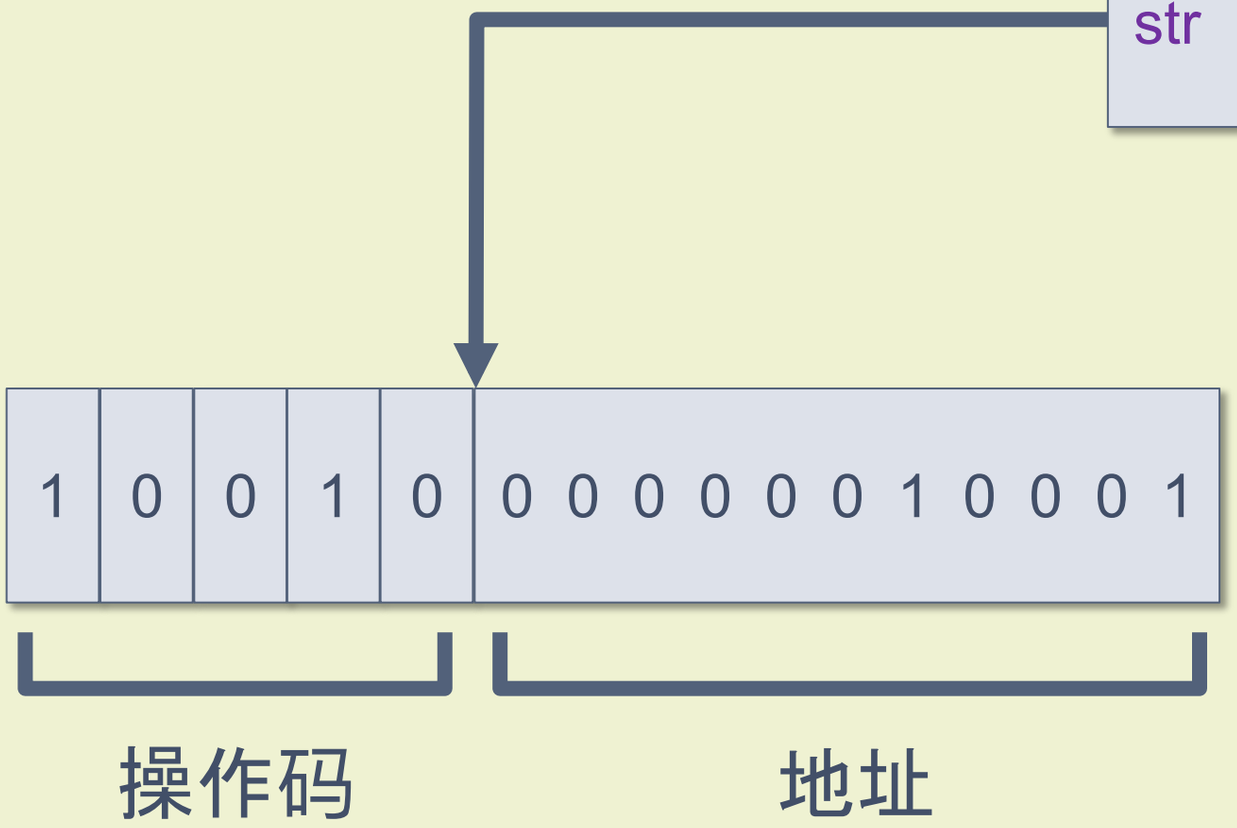
```
ldr    r0, [sp, #pin]
strb r2, [r3, #0]
str    r0, [sp, #result]
```



跳过/破坏指令

```
result = comparePIN(pin);
```

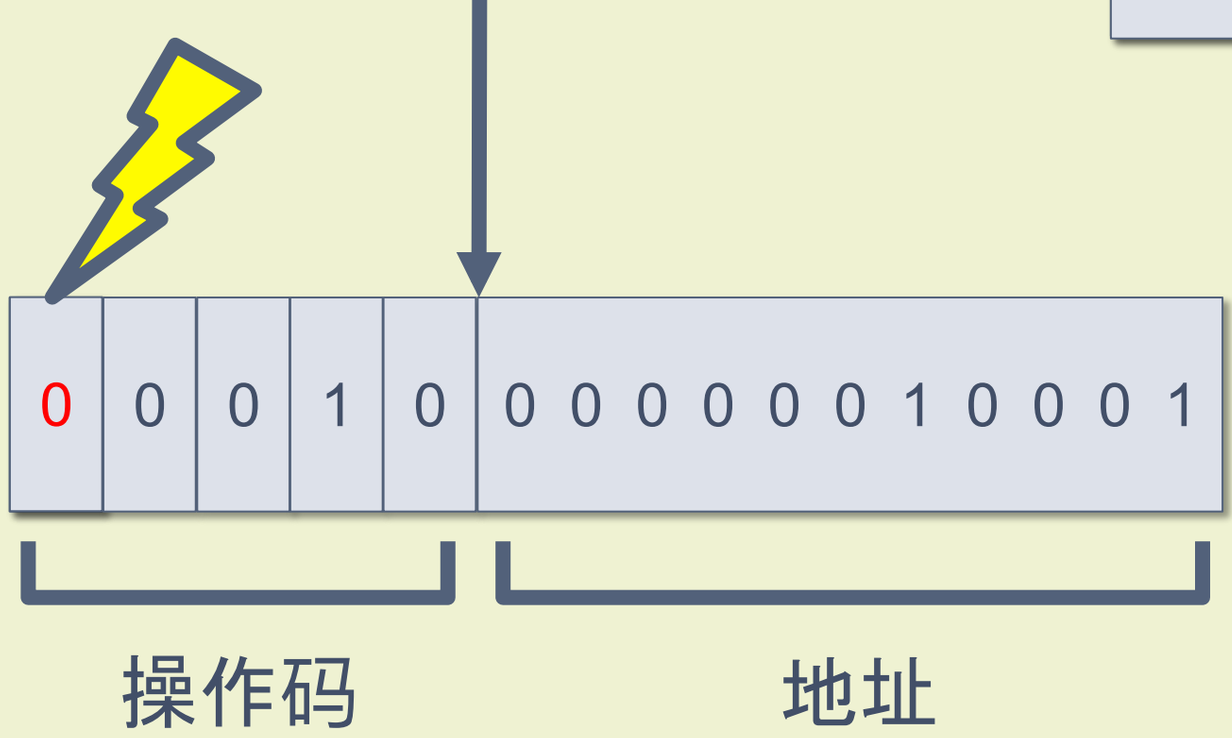
```
ldr   r0, [sp, #pin]
bl    comparePIN
str   r0, [sp, #result]
```



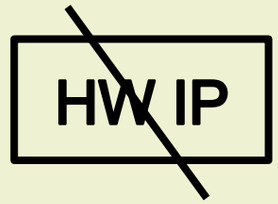
跳过/破坏指令

```
comparePIN(pin);
```

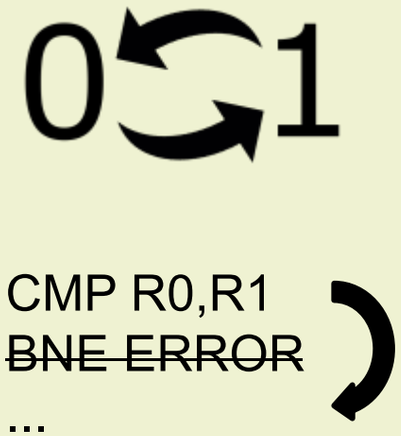
```
ldr    r0, [sp, #pin]
bl     comparePIN
asrs r1, r2, #32
```



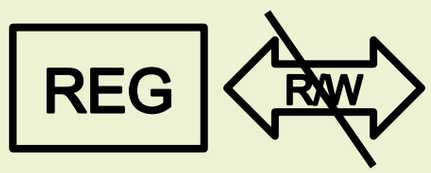
故障的影响



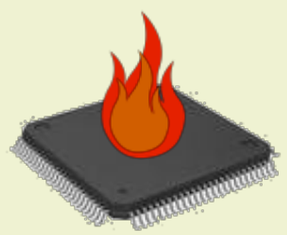
模块失灵



位翻转



阻碍读/写

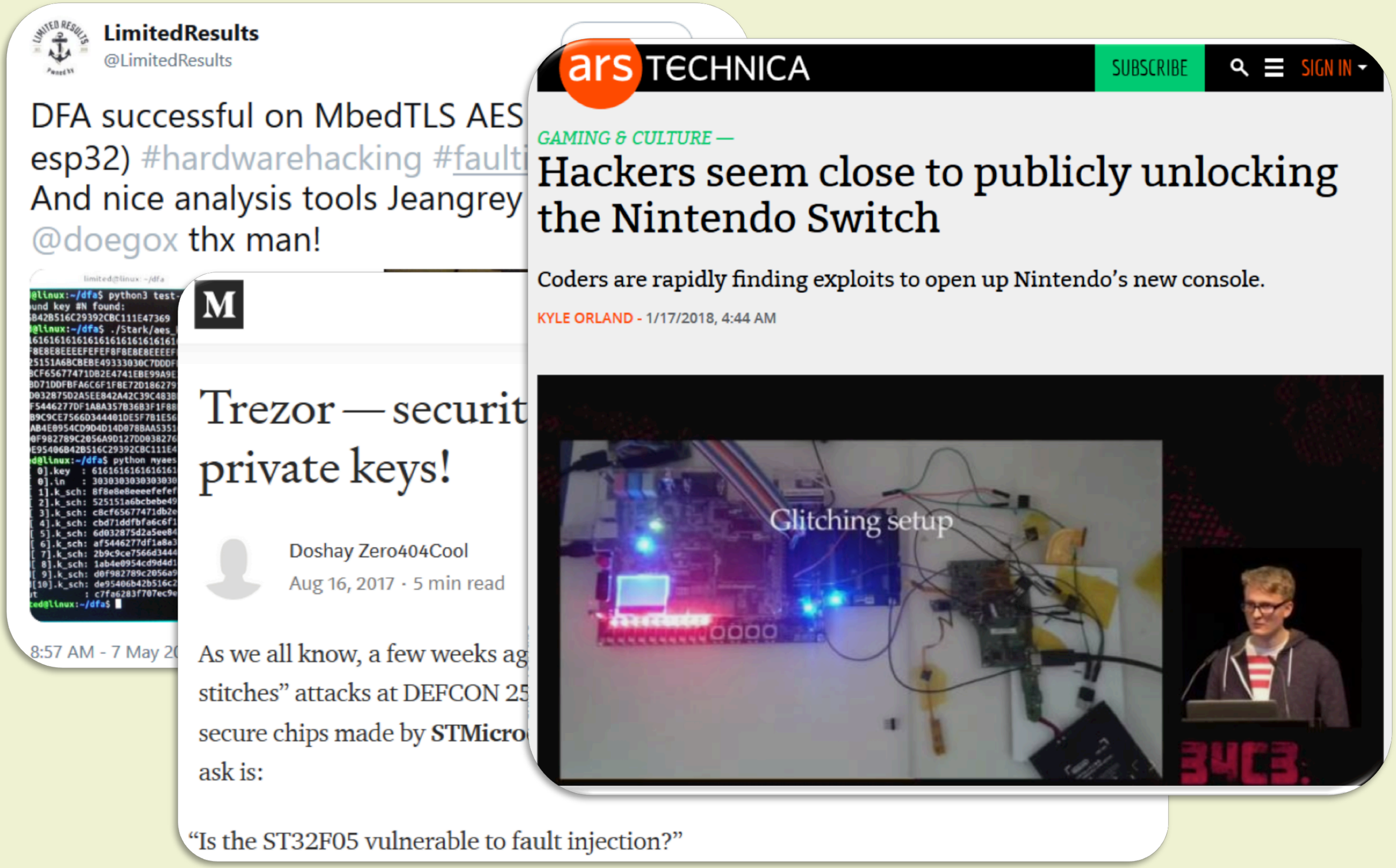


破坏芯片

跳过/破坏指令

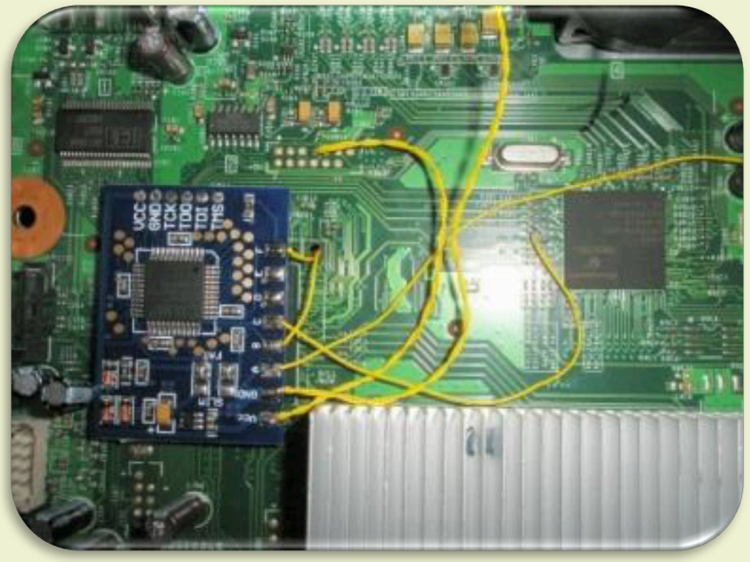
FI的发展和现状

FI 正在成为流行的测试技术!



但它已经存在很多年了...









年龄测试!

这两者之间有什么关系?



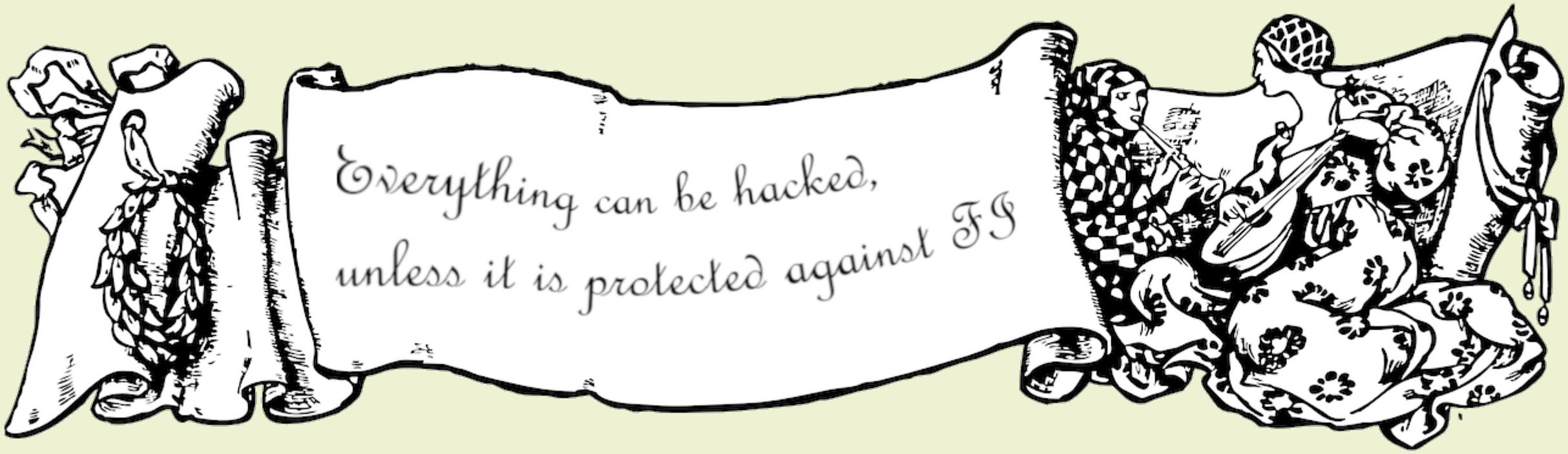


不要在家做这个!

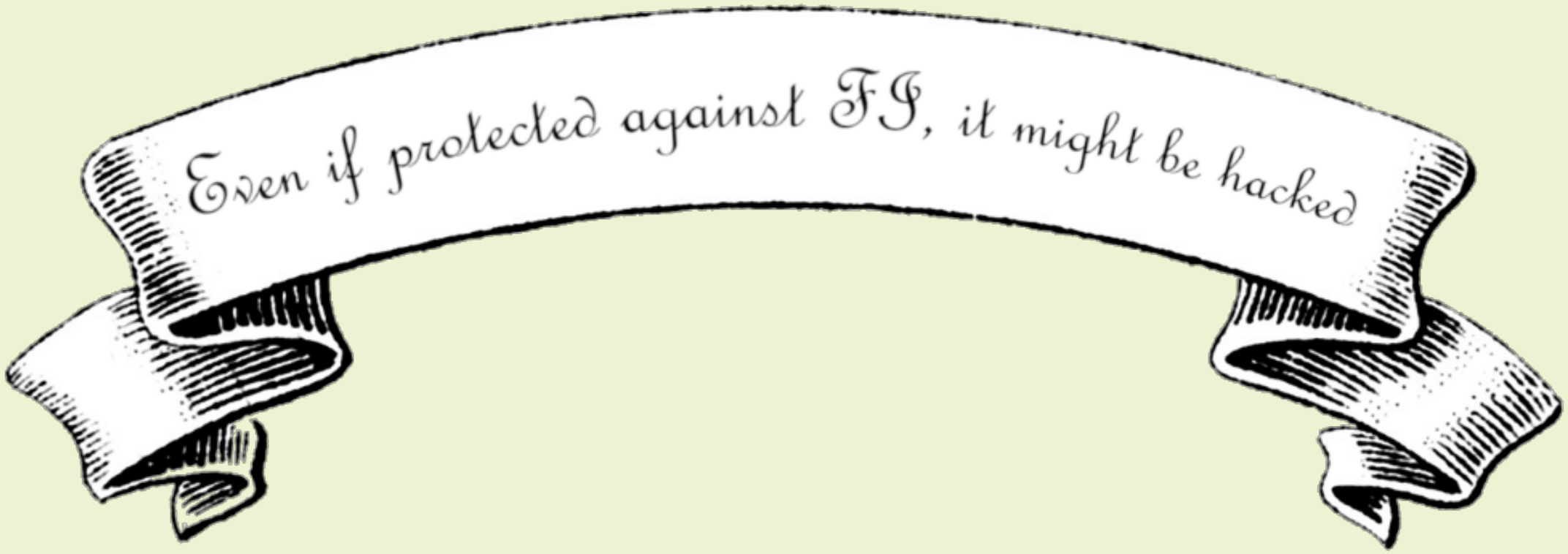


FI的实际应用

故障注入规则 #1



故障注入规则 #2



10+ 年存在的故障

绕过身份认证

绕过安全启动

升级权限

复原加密密钥

修改安全配置.

绕过身份认证

绕过身份认证

```
const char *PIN = "19379148"

// Compare the PIN with the correct one
// Return true if different

bool comparePIN(char *pin) {

    for (int i=0; i<strlen(pin); i++) {
        if (pin[i]!=PIN[i])
            return true;
    }
    return false;
}
```

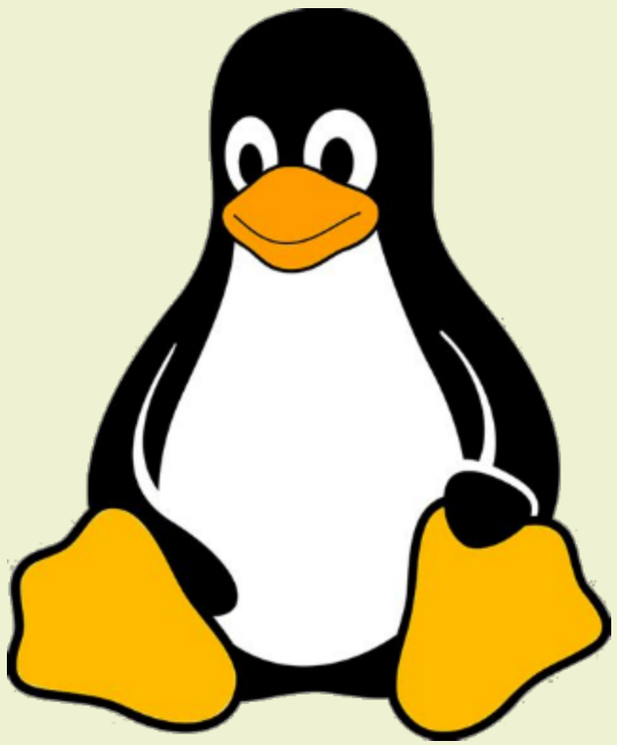
```
// Authenticate PIN.
// Return true if incorrect

bool pinAuthentication() {

    char pin[80];
    char tmp[80];
    bool result=false;


    printf("Please, introduce PIN\r\n");
    gets(pin);

    result = comparePIN(pin);

    if (result == false) {
        sprintf(tmp,"PIN correct\r\n");
    } else {
        sprintf(tmp,"PIN %s is
                    incorrect\r\n",pin);
    }

    printf(tmp);
    return result;
}
```

绕过身份认证


Visa Classic
4000 1234 5678 9010
4000
GOOD THRU 12/20
G. RAYMOND
VISA


RECOVERY SENTENCE
1.power 2.speak 3.east 4.detail 5.pitch
6.debate 7.page 8.solid 9.point 10.you
11.cart 12.cinnamon


Config
Scroll
1 Phase, 2 Wire, 240V, 10-100A, Cl 1.0, 50Hz
SILVERSPRING
Mesh Radio
N22834



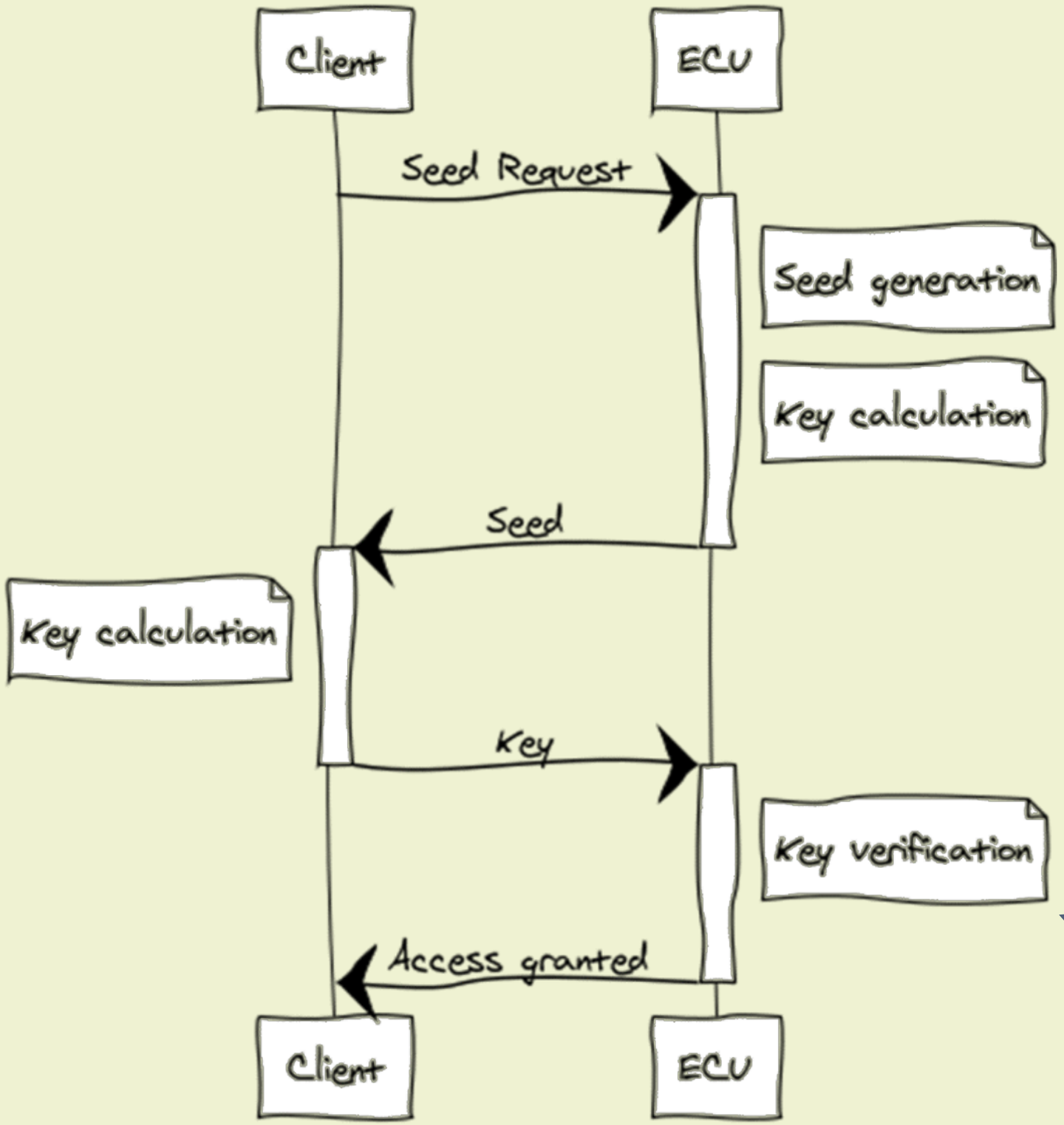


统一诊断服务

ISO 14229

统一
诊断
服务

UDS（通用数据系统）身份认证



FI

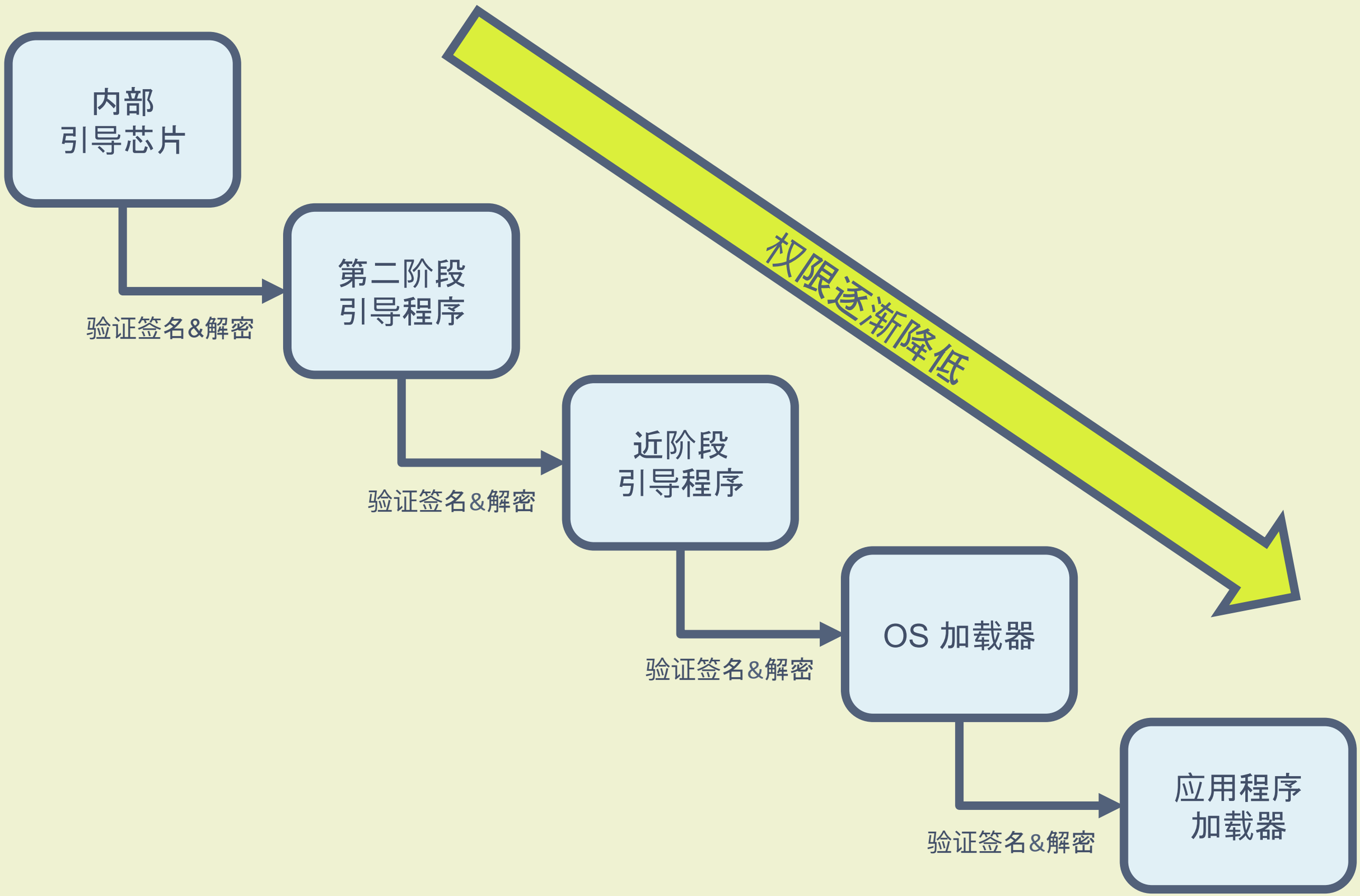
使用FI绕过UDS认证

```
if (receivedKey != expectedKey)
    not_authenticated();
else
    authenticated();
```



绕过安全启动

安全启动


内部
引导芯片
验证签名&解密
第二阶段
引导程序
验证签名&解密
近阶段
引导程序
验证签名&解密
OS 加载器
验证签名&解密
应用程序
加载器
权限逐渐降低

绕过安全启动


09:20
SAT, AUG 11
Google
Play Store
Phone
Messages
Chrome
Camera



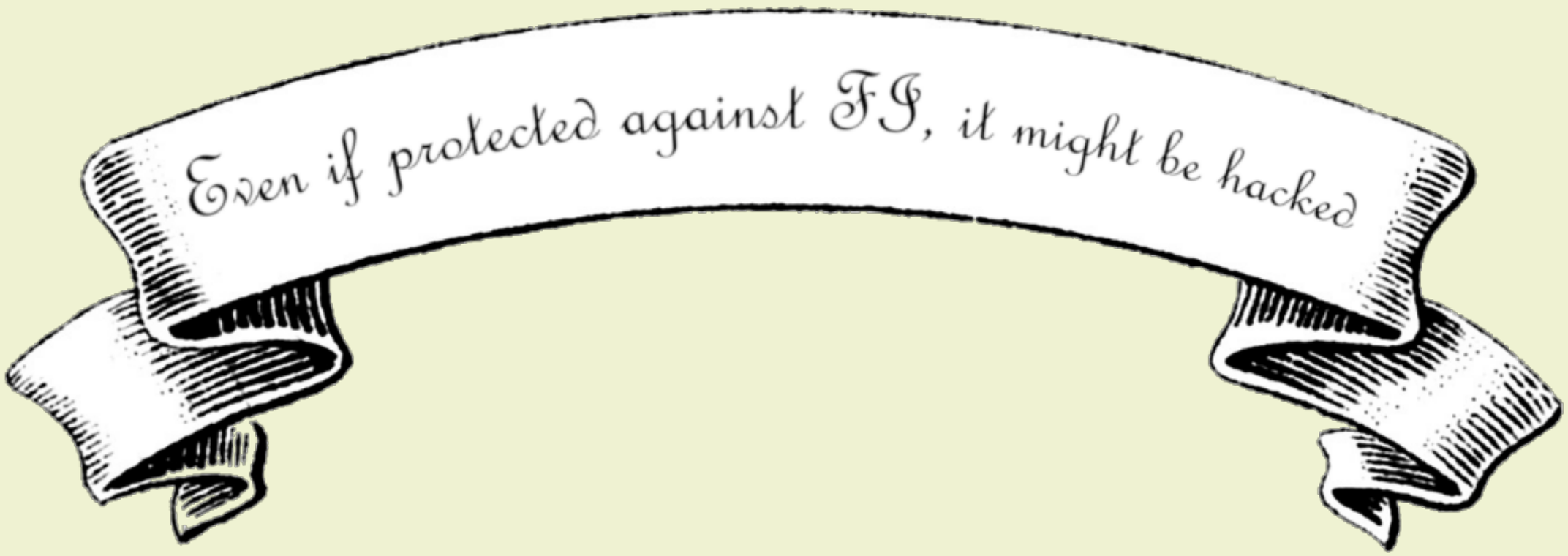

12:45
24°c

绕过安全启动

```
void boot() {

        void (*entryPoint)(void);
        ...

        // Load image to buffer
        uint32_t len = loadFlashImage(imgBuf)

        // Verify image
        if (verifyImage(imgBuf, len, sign,
                                    rsaKey) == false) {
                goto reset;      // Auth failed
        }

        // Jump to image
        entryPoint = imgbuf;
        (*entryPoint)();

        ...
}
```

```
//Verify image. Returns true is correct

bool verifyImage(uint8_t *image,
      uint32_t len, uint8_t *sign, rsaKey) {

   char *hash1, *hash2;
   ...

   // Calculate Hash
   calculateHash(image,len,hash1)

   // Verify signature
   verifySignature(sign, rsaKey, hash2)

   // Compare hashes
   if (memcmp(hash1, hash2, HASH_LEN) != 0)
      return false;
   else
      return true;

}
```

你是否记得FI规则#2？

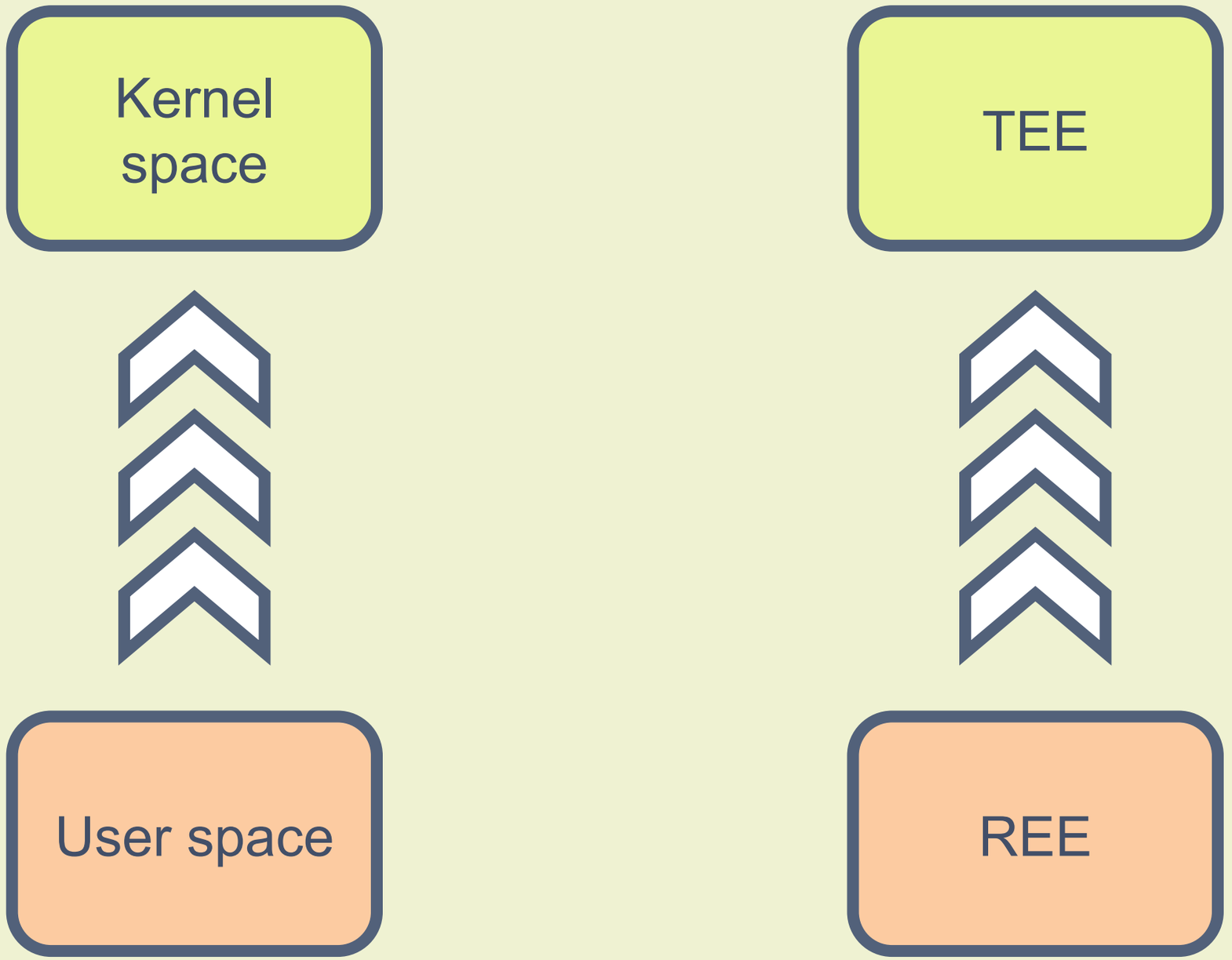


保护安全启动可能是一个噩梦

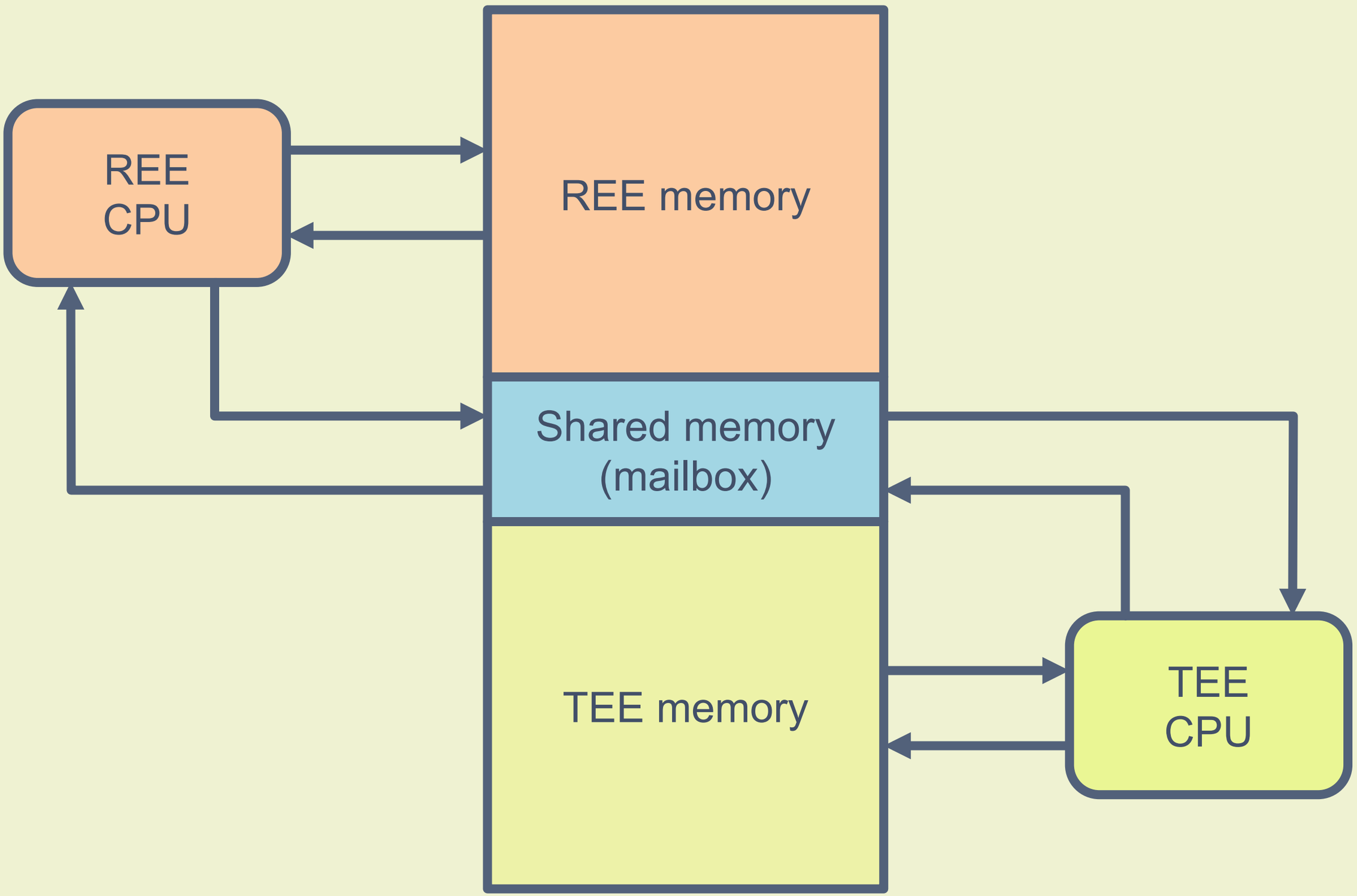
FISim(金融中介服务)案例

升级权限

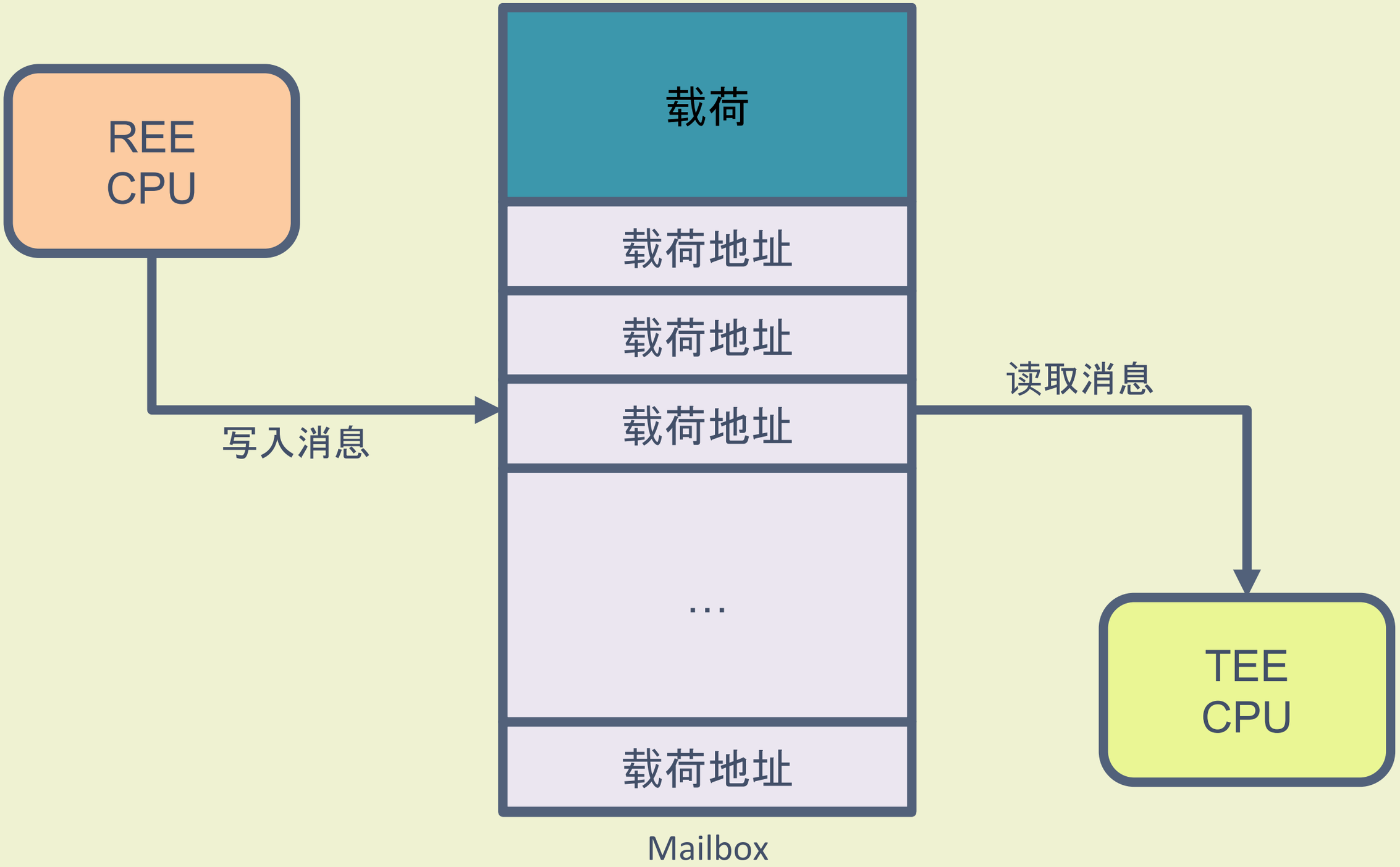
升级权限



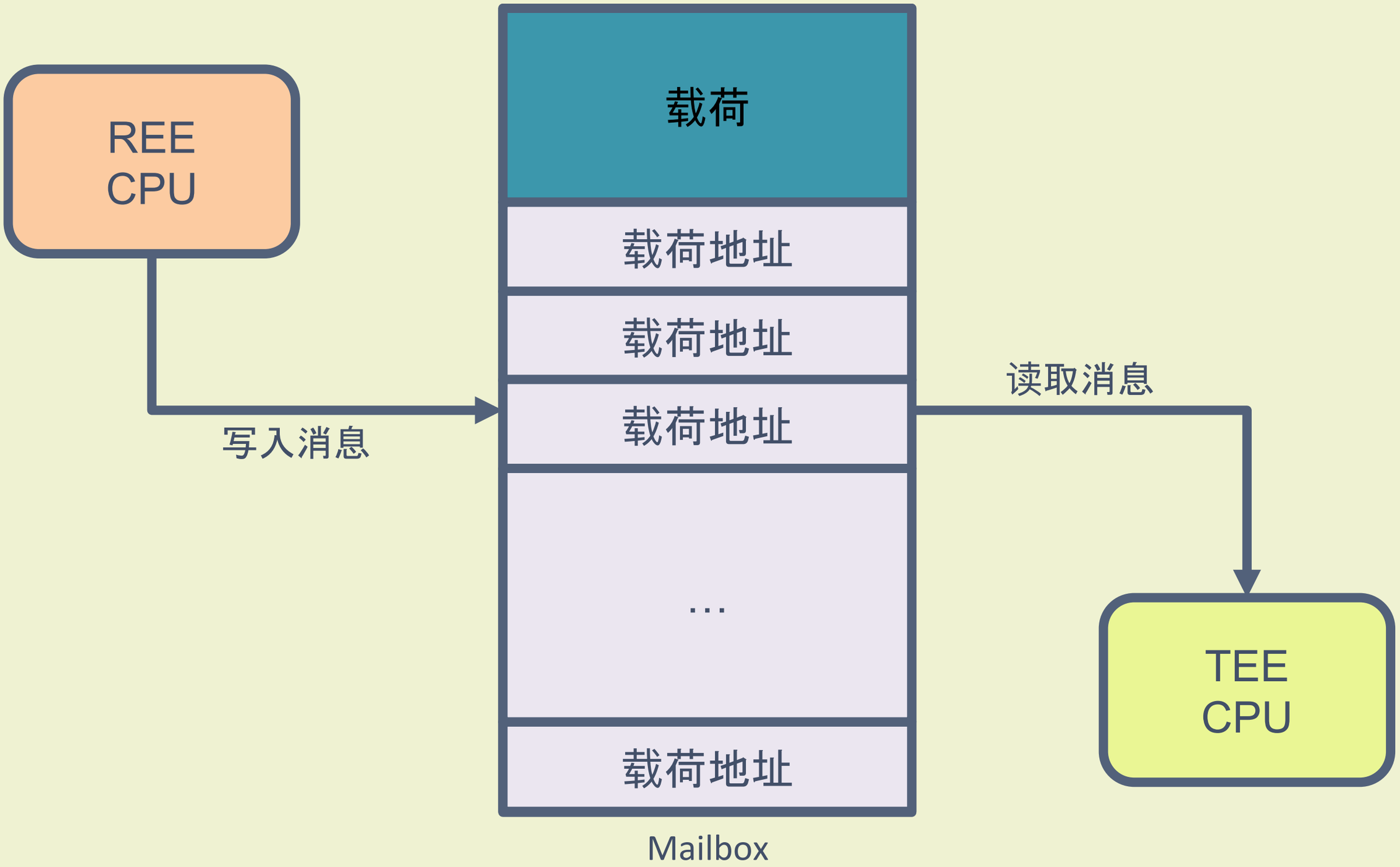
REE-TEE 分离



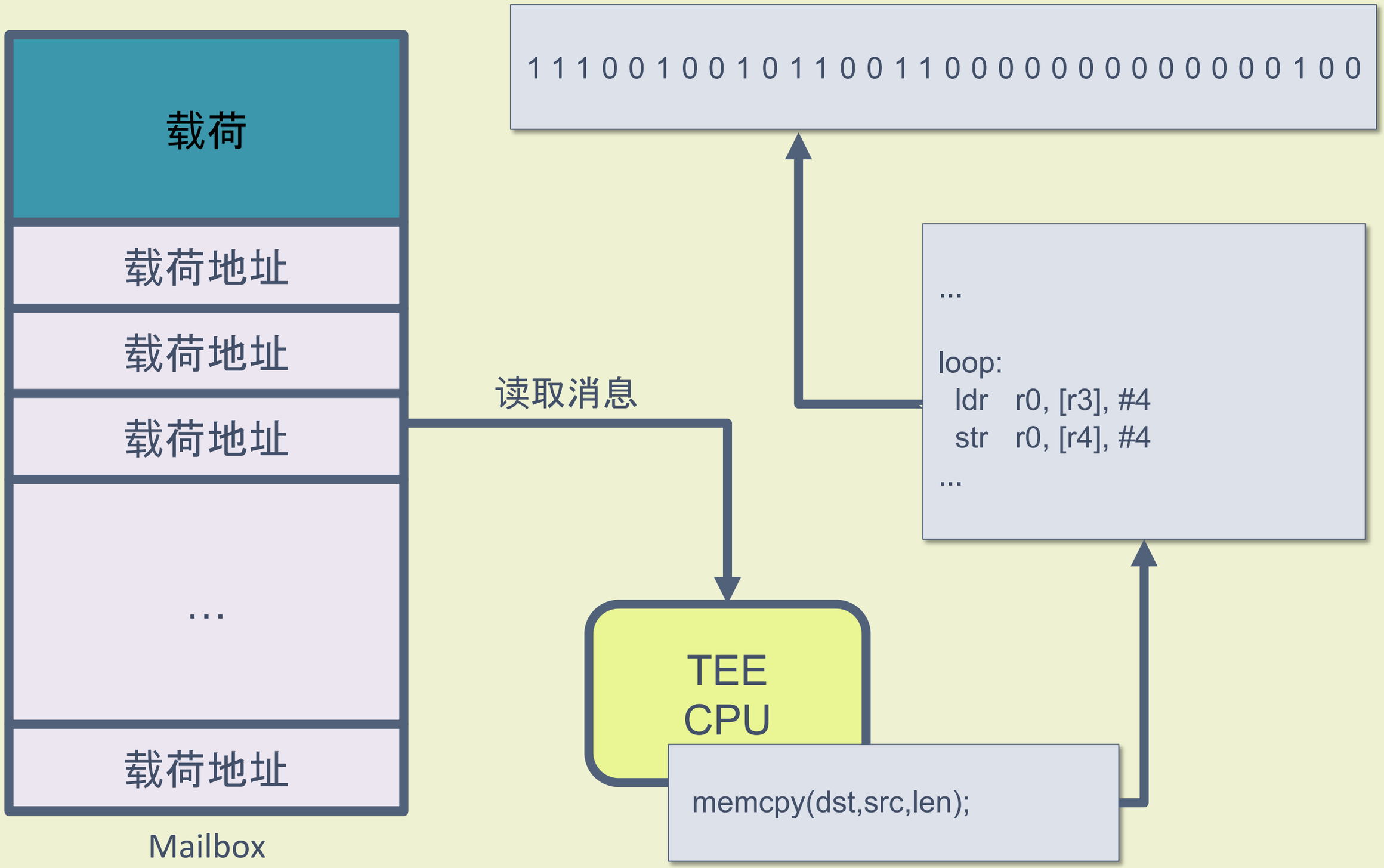
随机跳跃攻击



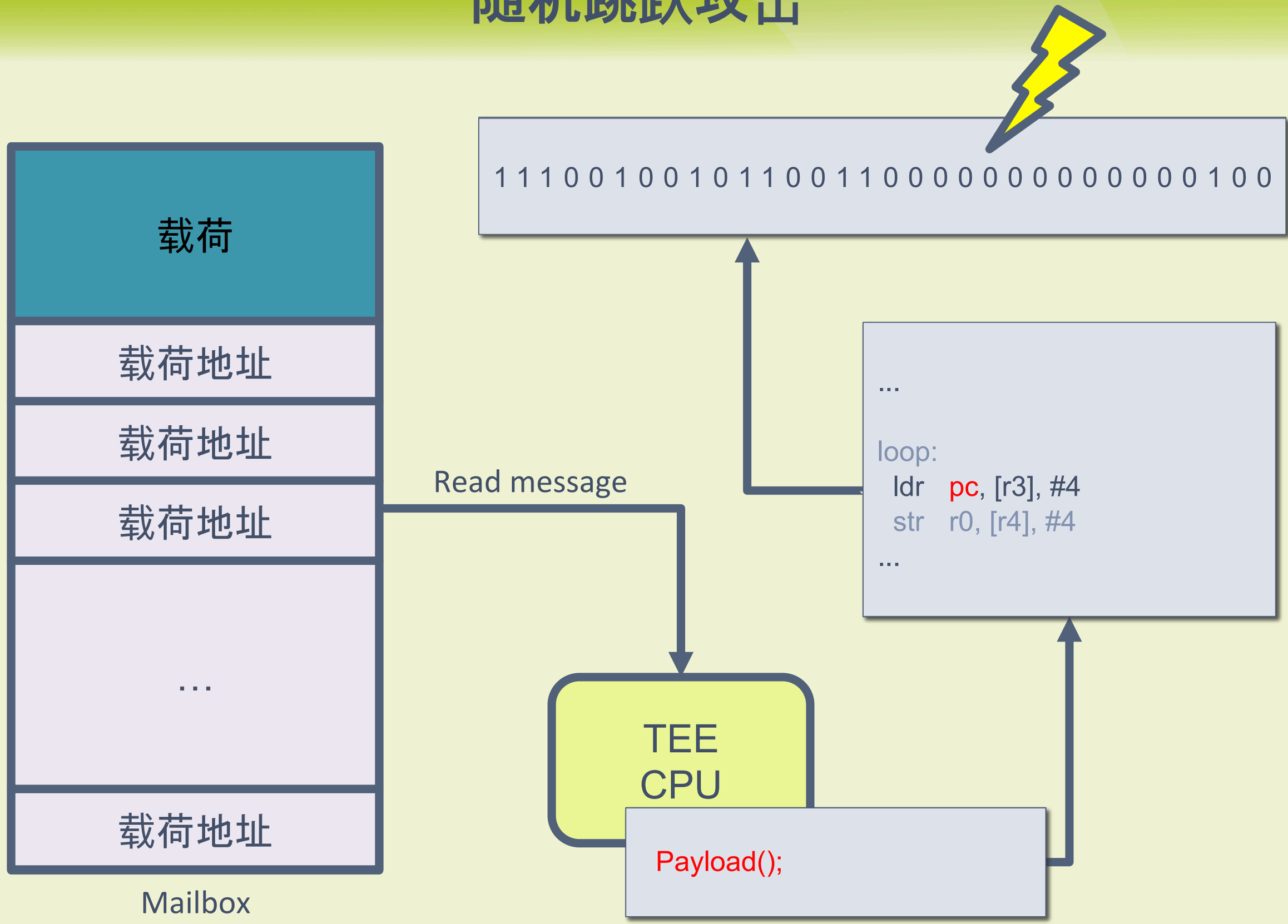
随机跳跃攻击



随机跳跃攻击

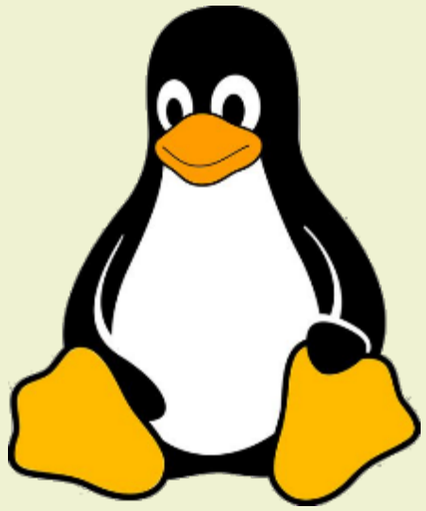

载荷
载荷地址
载荷地址
载荷地址
…
载荷地址
Mailbox
读取消息
1 1 1 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
...
loop:
ldr r0, [r3], #4
str r0, [r4], #4
...
TEE
CPU
memcpy(dst,src,len);

随机跳跃攻击


1 1 1 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
载荷
载荷地址
载荷地址
载荷地址
...
载荷地址
Mailbox
Read message
TEE
CPU
...
loop:
ldr pc, [r3], #4
str r0, [r4], #4
...
Payload();

升级权限









arm

TRUSTZONE

复原密钥

复原密钥

故障分析

DFA

差分故障分析

DFIA

差分故障
强度分析

SFA

计故障分析

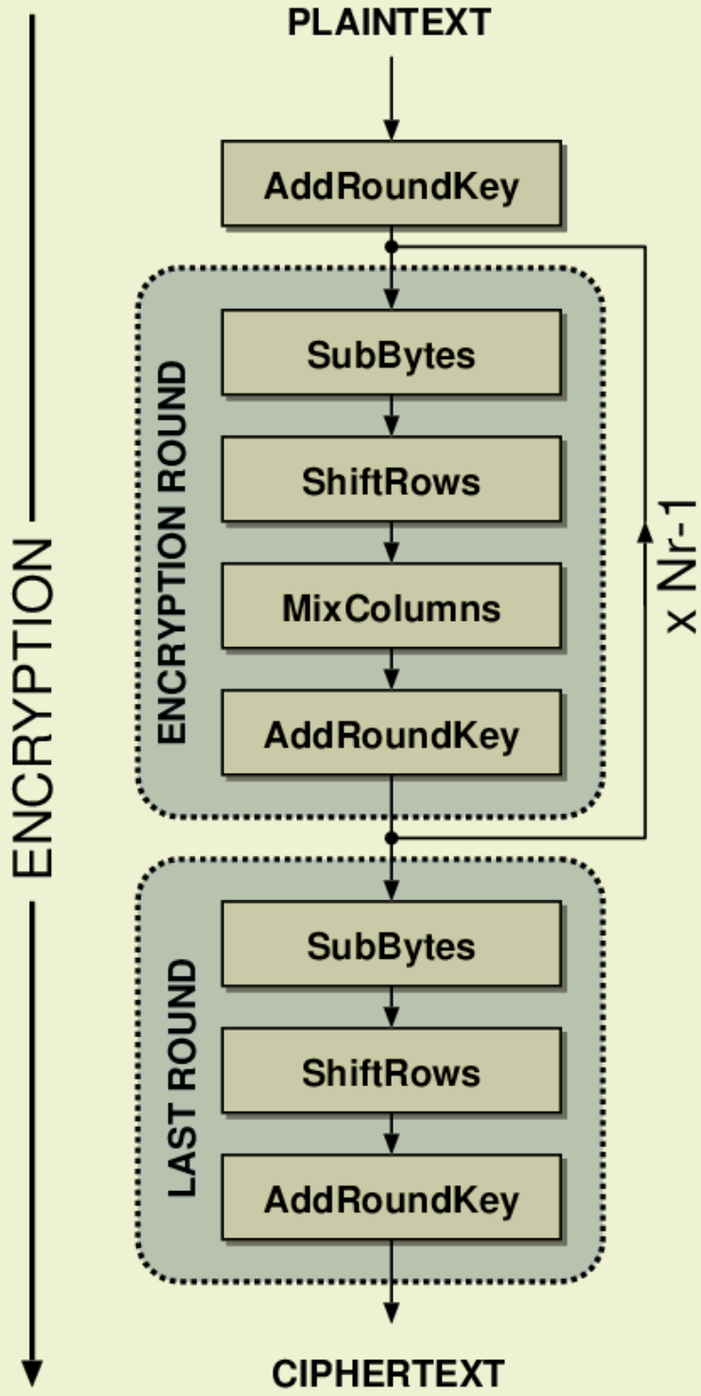
IFA

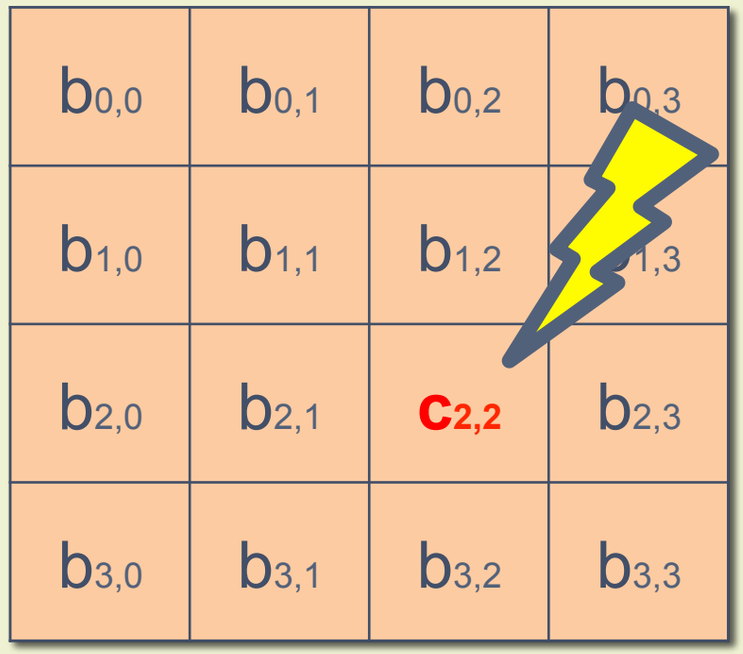
失效的故障分析

SIFA

统计失效故障分析

AES故障分析

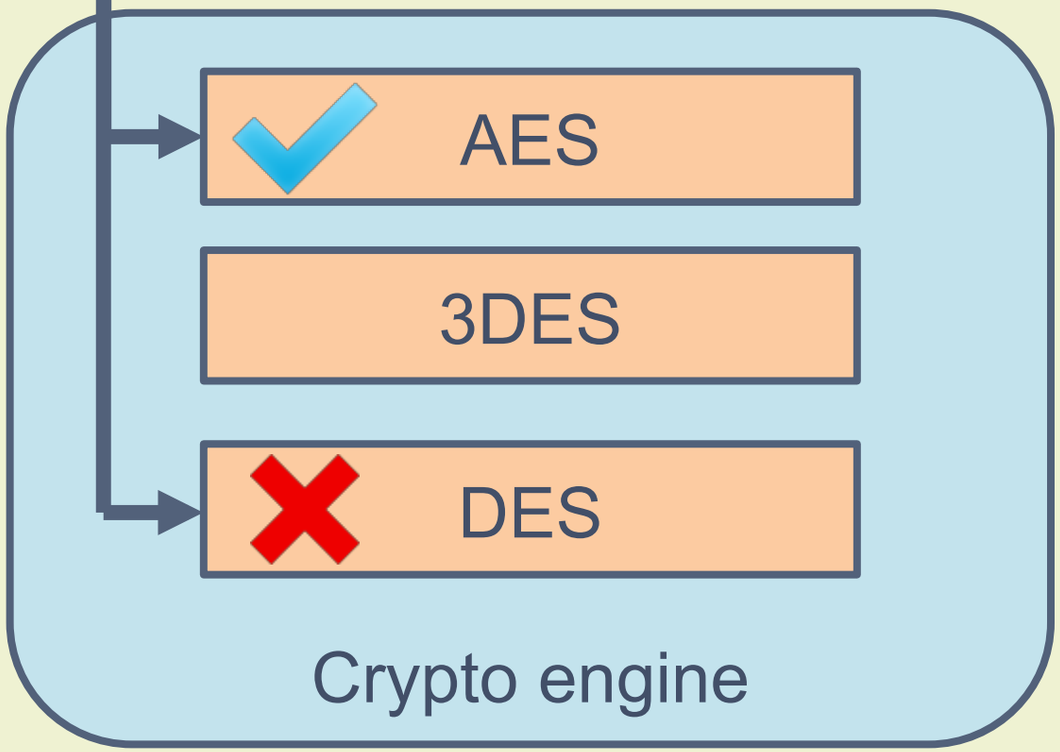

PLAINTEXT
AddRoundKey
SubBytes
ShiftRows
MixColumns
AddRoundKey
ENCRYPTION ROUND
x Nr-1
SubBytes
ShiftRows
AddRoundKey
LAST ROUND
ENCRYPTION
CIPHERTEXT


b0,0 b0,1 b0,2 b0,3
b1,0 b1,1 b1,2 b1,3
b2,0 b2,1 c2,2 b2,3
b3,0 b3,1 b3,2 b3,3

基于关键属性的故障注入

Key	Attributes	
Key 1	AES	ENC
Key 2	3DES	DEC
Key 3	AES/3DES	ENC/DEC
…	…	…

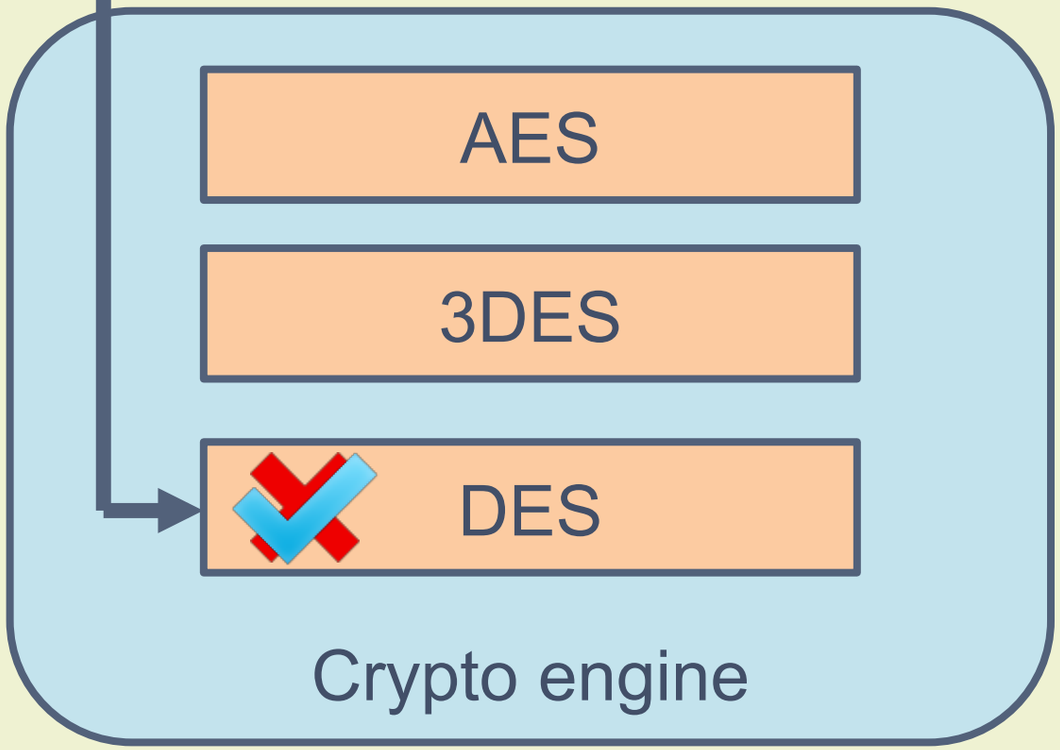
Keyslot



基于关键属性的FI

Key	Attributes	
Key 1	DES	ENC
Key 2	3DES	DEC
Key 3	AES/3DES	ENC/DEC
…	…	…

Keyslot



复原密钥




CRYPTO


Visa Classic
4000 1234 5678 9010
4000
GOOD THRU 12/20
G. RAYMOND
VISA


09:20

修改安全配置

阻碍安全配置

```
void boot() {

        void (*entryPoint)(void);
        ...

        // Load image to buffer
        uint32_t len = loadFlashImage(imgBuf)

        // Verify image
        if (verifyImage(imgBuf, len, sign,
                                    rsaKey) == false) {
                goto reset;      // Auth failed
        }

        // Jump to image
        entryPoint = imgbuf;
        (*entryPoint)();

        ...
}
```

```
// Read fuses and configure security modules

bool disableJtag = readFuse(DISABLE_JTAG);
bool enableMemoryScrambler =
                        readFuse(ENABLE_MEM_SCRAMBLER);
bool enableTEE = readFuse(ENABLE_TEE);


if (disableJtag==true)
    closeJtag();

if (enableMemoryScrambler==true)
    configureMemoryScrambler();

if (enableTEE==true)
    configureTEE();
```

局部故障注入

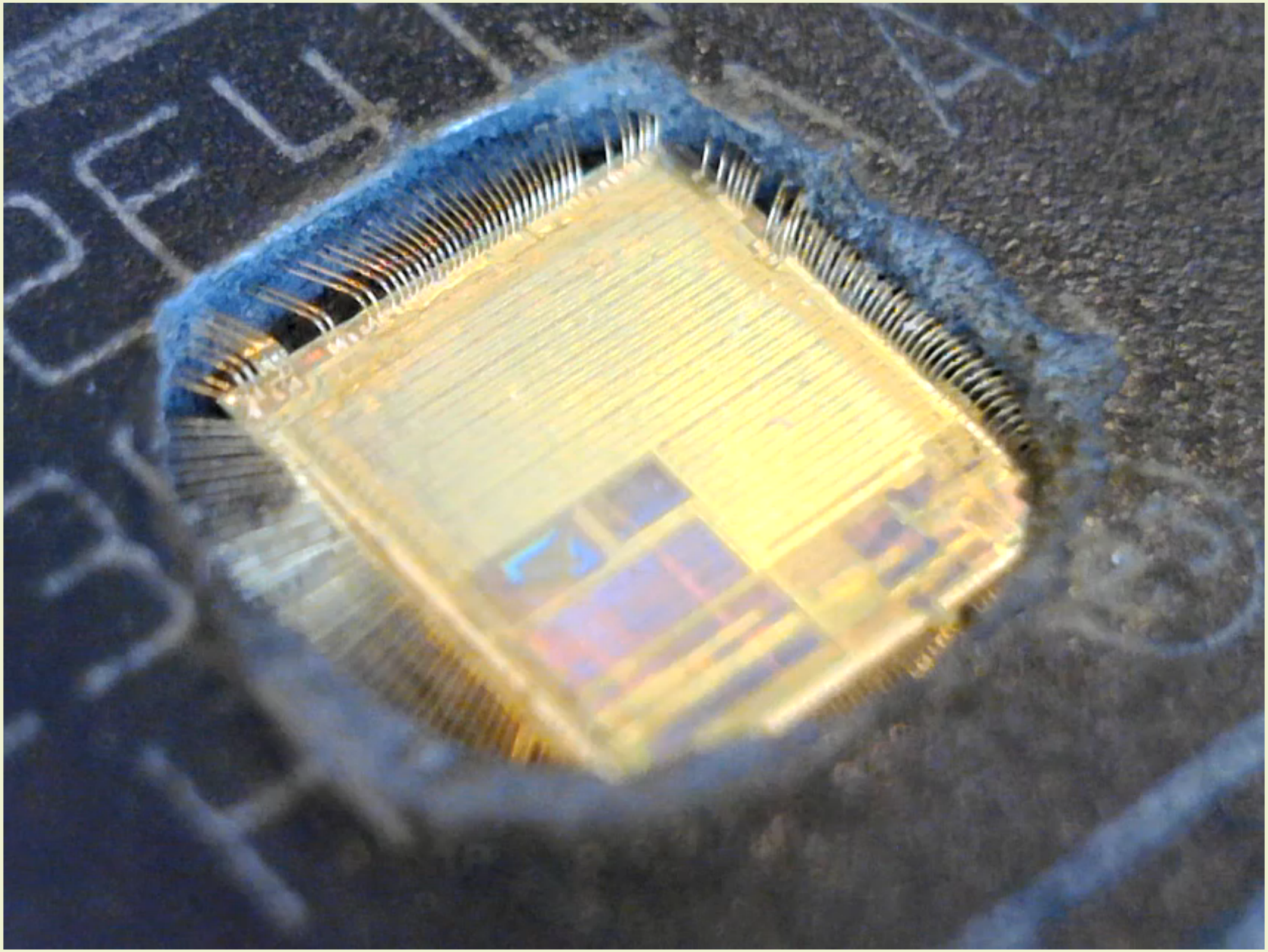


电磁故障注入

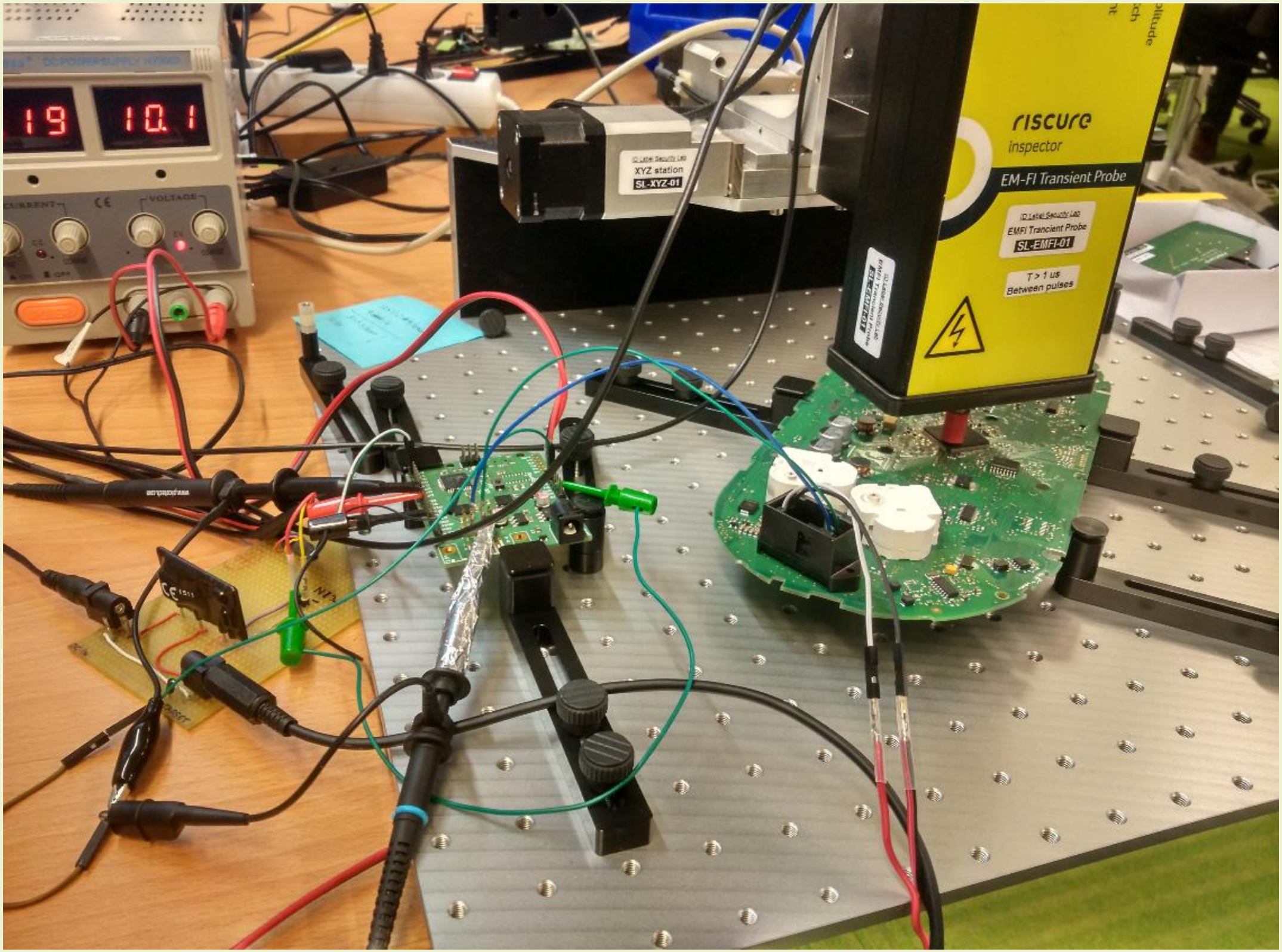


激光故障注入

激光故障注入

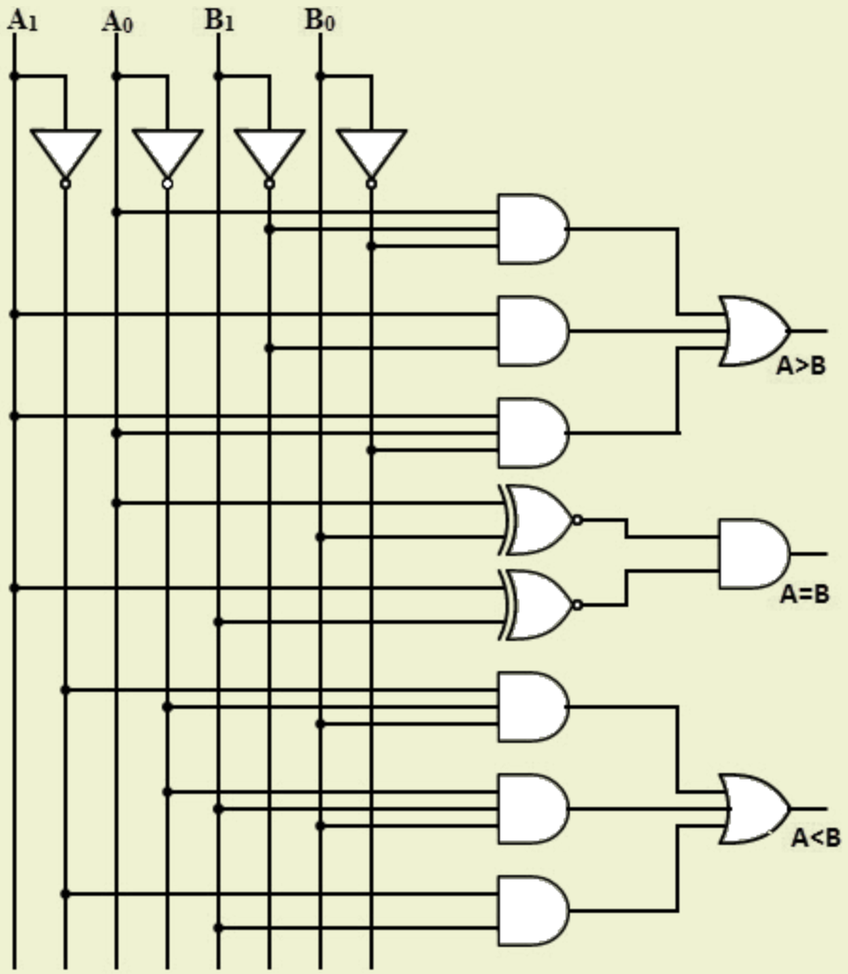


电磁故障注入


riscure
inspector
EM-FI Transient Probe
EMFI Trancient Probe
SL-EMFI-01
T > 1 us
Between pulses
XYZ station
SL-XYZ-01
VOLTAGE

FI的预防

FI的应对措施



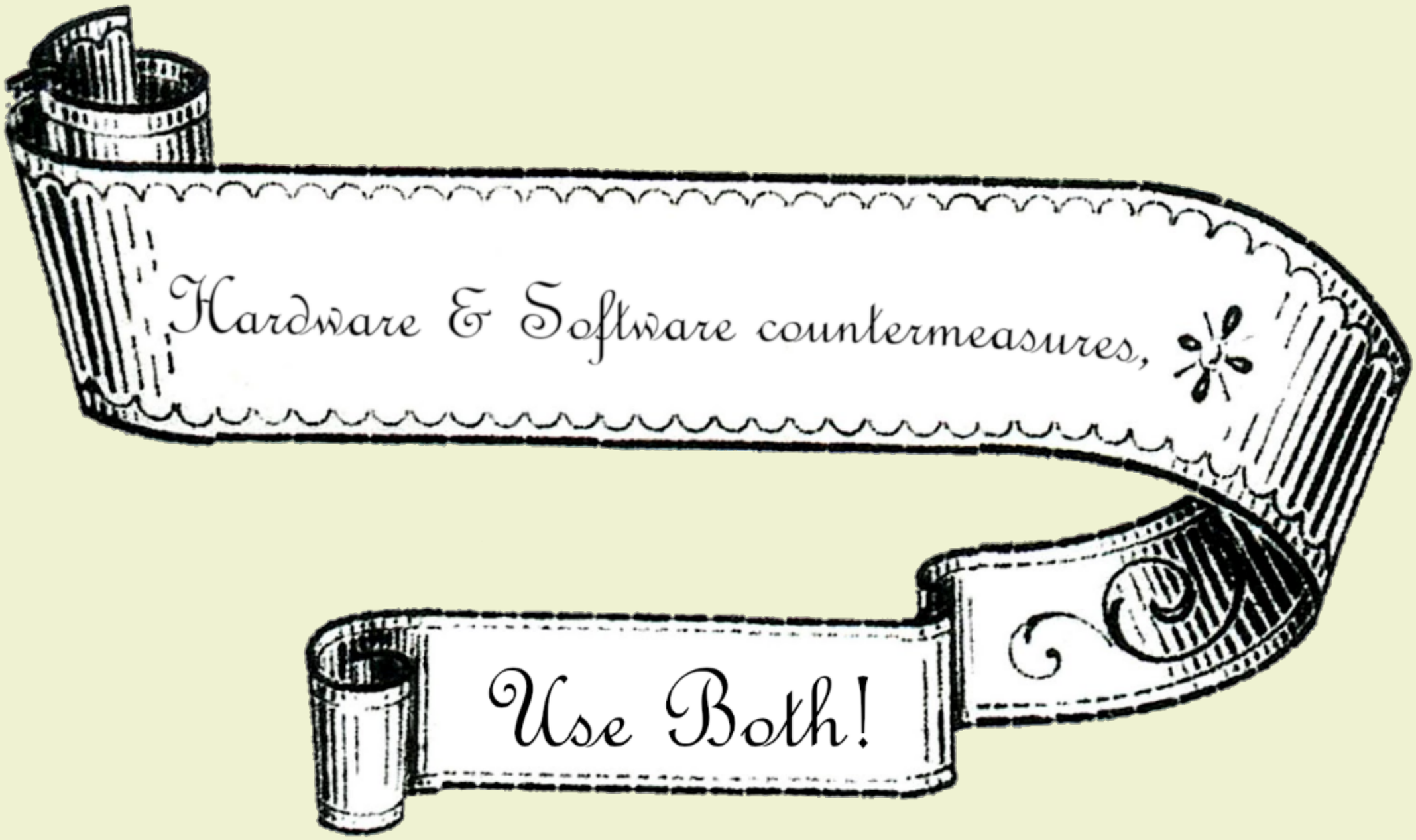
硬件

VS

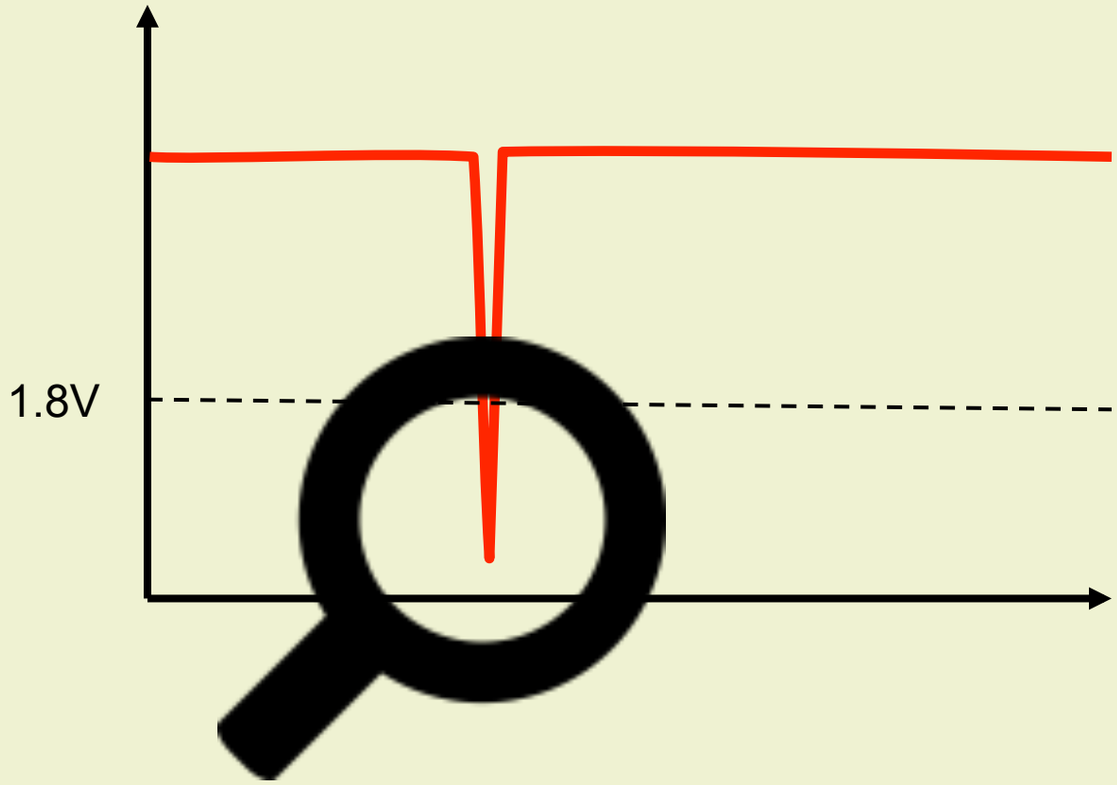
```
00000000                push    ebp
00000001                mov     ebp, esp
00000003                movzx   ecx, [ebp+arg_0]
00000007                pop     ebp
00000008                movzx   dx, cl
0000000C                lea     eax, [edx+edx]
0000000F                add     eax, edx
00000011                shl     eax, 2
00000014                add     eax, edx
00000016                shr     eax, 8
00000019                sub     cl, al
0000001B                shr     cl, 1
0000001D                add     al, cl
0000001F                shr     al, 5
00000022                movzx   eax, al
00000025                retn
```

软件

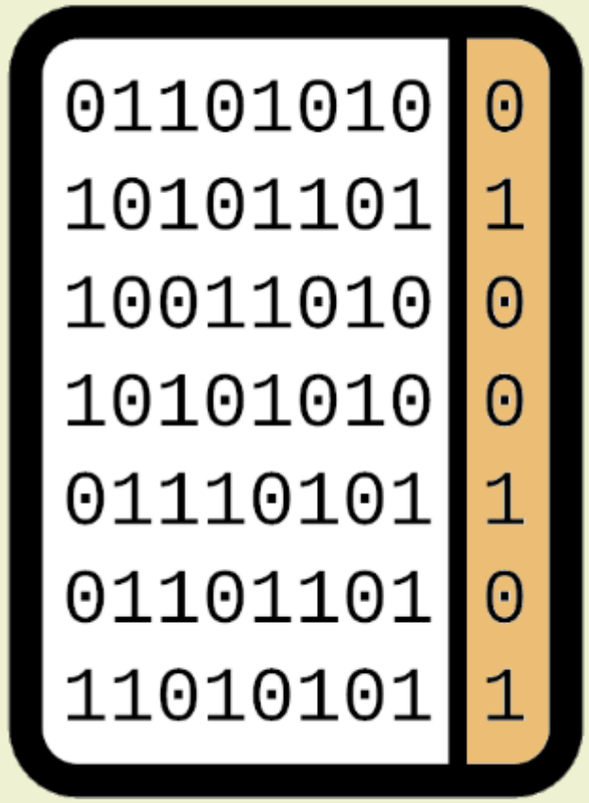
故障注入规则 #3



基于硬件的措施



故障感应器



冗余

基于软件的措施#1

```
if (receivedKey != expectedKey)
    not_authenticated();
else
    authenticated();
```



基于软件的措施 #1



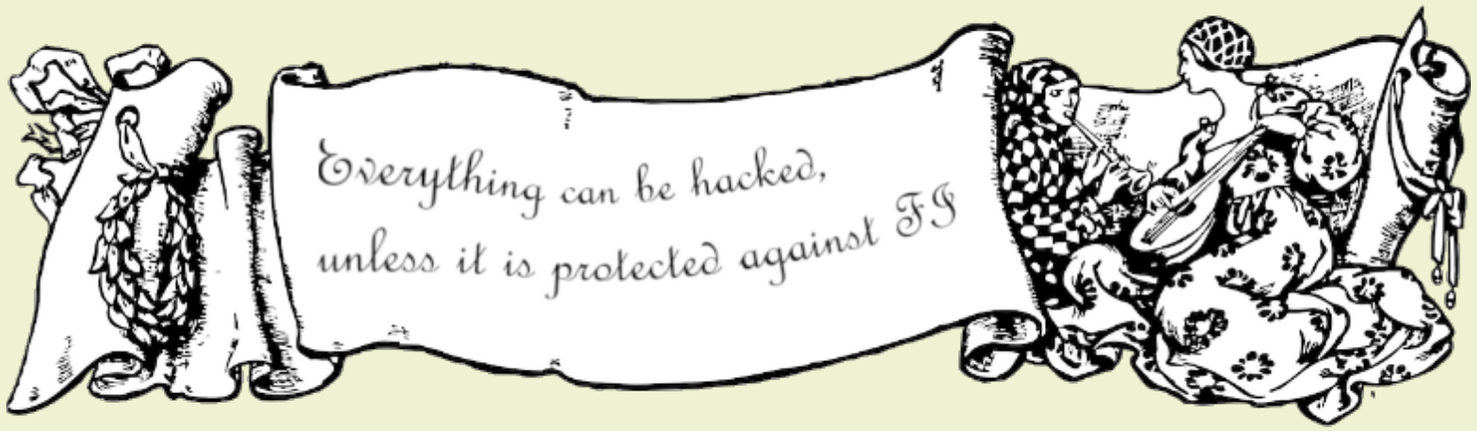
```
if (receivedKey != expectedKey)
    not_authenticated();

...
// Do something else
...

if (receivedKey != expectedKey)
    not_authenticated();
else
    authenticated();
```

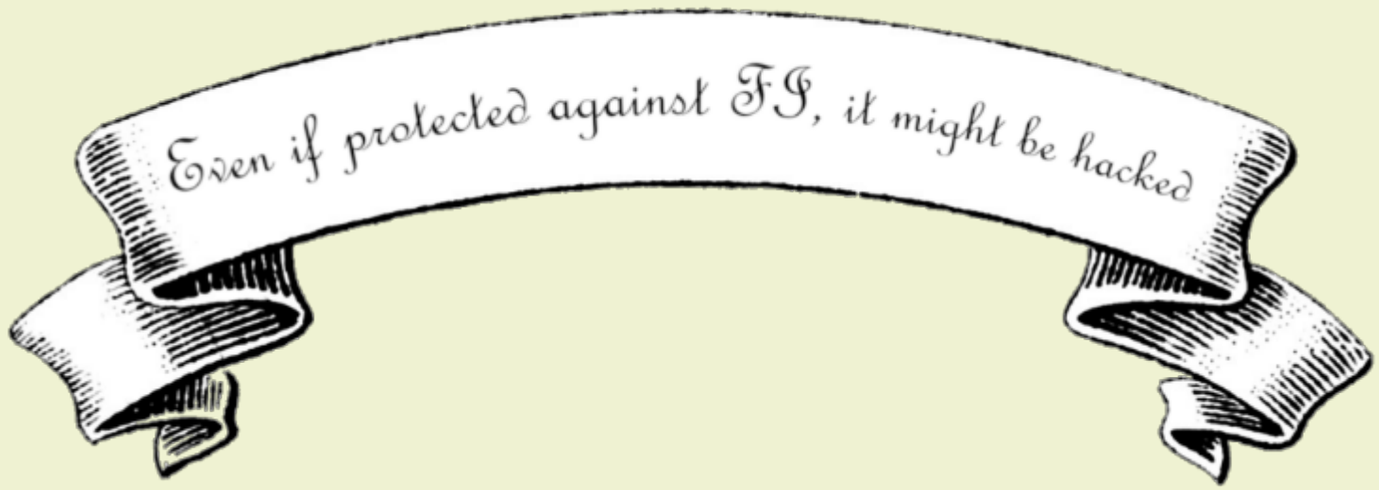
基于软件的措施 #2

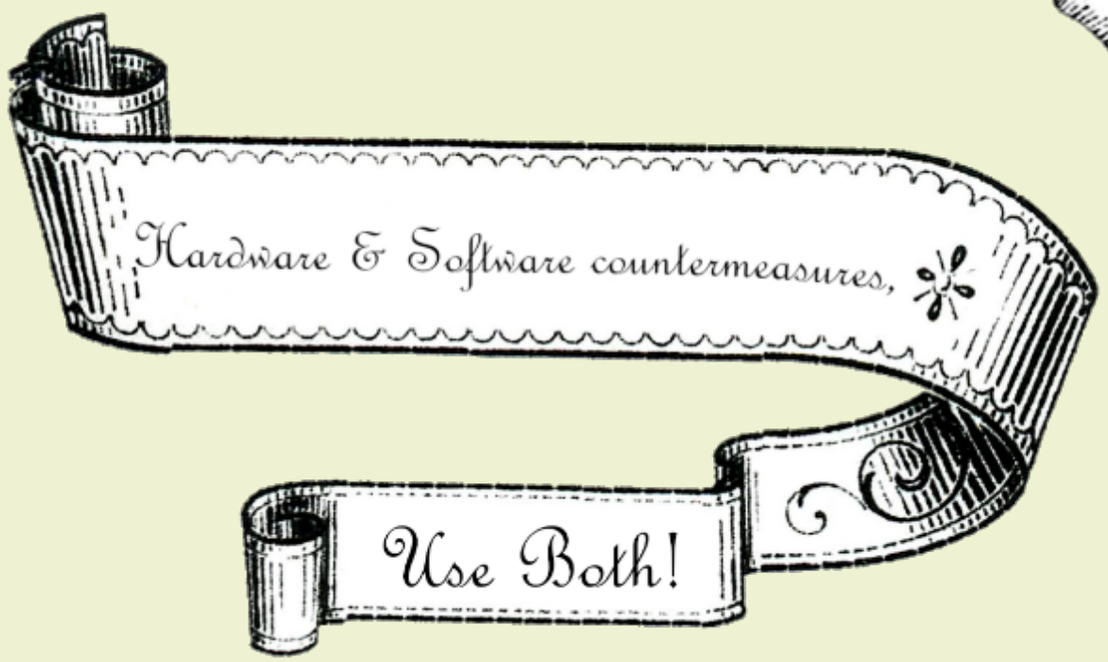
```
sleep(rand()); // Random delay

if (receivedKey != expectedKey)
    not_authenticated();
else
    authenticated();
```

总结

记住这些规则!


Everything can be hacked,
unless it is protected against FI


Even if protected against FI, it might be hacked


Hardware & Software countermeasures,
Use Both!

更多信息:

Protecting software against FI
https://www.riscure.com/uploads/2018/11/201708_Riscure_Whitepaper_Side_Channel_Patterns.pdf

FI on UDS:
https://www.riscure.com/uploads/2018/06/
Riscure_Whitepaper_Fault_injection_on_automotive_diagnostic_protocols.pdf

Bypassing secure boot
https://www.riscure.com/uploads/2017/10/eu-16-Timmers-Bypassing-Secure-Boot-Using-Fault-Injection.pdf

Linux privileges escalation:
https://www.riscure.com/uploads/2017/10/
Riscure_Whitepaper_Escalating_Privileges_in_Linux_using_Fault_Injection.pdf

Optical FI
https://www.riscure.com/uploads/2017/09/Practical-optical-fault-injection-on-secure-microcontrollers.pdf

Wild Jungle Jump attack
https://www.riscure.com/uploads/2017/09/Controlling-PC-on-ARM-using-Fault-Injection.pdf

Practical DFA
https://www.slideshare.net/secret/K8jlDL4os1evrW

招聘进行中!

上海·中国

代尔夫特·荷兰

旧金山·美国

riscure

Contact: Ramiro Pareja
pareja@riscure.com

Riscure China
2081, No.989, Changle Road
200031 Shanghai
China
Phone: +86 21 5117 5440
Wechat: RiscureChina

Riscure B.V.
Frontier Building, Delftechpark 49
2628 XJ Delft
The Netherlands
Phone: +31 15 251 40 90

www.riscure.com

Riscure North America
550 Kearny St.
Suite 330
San Francisco, CA 94108
+1 (650) 646 9979

inforequest@riscure.com