VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

SKRÝVÁNÍ PROUDOVÉ SPOTŘEBY
HIDING OF CURRENT CONSUMPTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **David Hirš**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Zdeněk Martinásek, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Student: David Hirš ***ID:*** 195151
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Skrývání proudové spotřeby

POKYNY PRO VYPRACOVÁNÍ:

V rámci bakalářské práce se seznamte s útoky proudovým postranním kanálem. Teoretickou část práce zaměřte na implementace protiopatření využívající hardwarové znáhodnění provádějících se instrukcí kryptografického algoritmu (platforma FPGA - Field Programmable Gate Array). V praktické části realizujte nejprve jednoduchou a diferenční proudovou analýzu, která bude cílena na nemaskovanou implementaci kryptografického algoritmu (př. AES). V rámci analýzy nalezněte různé zajímavé body pro mezivýsledky algoritmu (otevřený text, S-BOX, přičtení klíče atd.) a diskutujte výsledky útoku. Implementujte maskovací schéma využívající hardwarové znáhodnění a realizujte proudovou analýzu maskované implementace. Získané výsledky přehledně zpracujte.

DOPORUČENÁ LITERATURA:

[1] MENTENS, Nele. Hiding Side-channel Leakage through Hardware Randomization: a Comprehensive Overview. In: International Conference on Systems, Architectures, MOdeling and Simulation (IC-SAMOS 2017). IEEE, 2017.

[2] MANGARD, Stefan; OSWALD, Elisabeth; POPP, Thomas. Power analysis attacks: Revealing the secrets of smart cards. Springer Science & Business Media, 2008.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Proudová analýza představuje úspěšný typ útoku cíleného na důvěrná kryptografická zařízení, jako jsou například chytré karty. Popularita Internetu věcí (IoT) každým dnem roste, a je tedy nutné taková zařízení zabezpečit. Pouhá implementace kryptografických algoritmů nestačí, proto je nutné zavést dodatečná opatření, aby bylo možné zamezit takovému typu útoku, který představuje proudová analýza.

Tato bakalářská práce je zaměřená na implementaci odlehčené šifry PRESENT a realizace Jednoduché proudové analýzy spolu s Diferenční proudovou analýzou za účelem odhalení tajného klíče šifrovacího zařízení. Proudová analýza je realizována jak na nemaskovanou šifru PRESENT, kde bylo možné odhalit tajný klíč, tak i na maskovanou implementaci šifry, kde byla dokázána účinnosti navrhovaného protiopatření, které využívá techniku FPGA rekonfigurace.

KLÍČOVÁ SLOVA

PRESENT, odlehčená šifra, proudová analýza, SPA, DPA, protiopatření, hardwarová rekonfigurace, postranní kanály

ABSTRACT

Power analysis presents the typical example of successful attacks against trusted cryptographic devices such as smart cards or embedded devices. Nowadays, the popularity of Internet of Things (IoT) is growing therefore, designers sould implement cryptographic algorithms with countermeasures in order to defend against these types of attacks.

The bachelor's thesis focus on the implementation of ultra-lightweight block cipher PRESENT and execution of Simple power analysis and Differential power analysis to reveal the secret key. Power analysis is firstly performed on cipher PRESENT with no implemented countermeasures and the attack was successful. Then the countermeasures were implemented and there were no leakage about the secret key. The countermeasures are using the hardware randomization of FPGA boards.

KEYWORDS

PRESENT, lightweight cipher, power analysis, SPA, DPA, countermeasures, hardware reconfiguration, side channels

HIRŠ, David. *Skrývání proudové spotřeby*. Brno, 2019, 69 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Skrývání proudové spotřeby“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Zdeňku Martináskovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

podpis autora

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Elektronická zařízení se stala běžnou součástí lidských životů a s touto skutečností byl zvětšen důraz na ochranu soukromí a nutnost utajení některých citlivých informací (dat). Se zlepšujícím a zkvalitňujícím se zabezpečením citlivých informací vznikají stejně tak lepší a kvalitnější útoky na toto zabezpečení. Kupříkladu u standardizované blokové šifra AES prozatím nebyla odhalena žádná zranitelnosti, kterou by bylo možné využít k překonání zabezpečení, a je tedy odolná vůči konvenčnímu způsobu analýz zaměřeným na matematickou podstatu algoritmu. Díky velikosti klíče 2^{128} je, se současnou technologií, útok hrubou silou takřka nemožný. Překonání takového typu ochrany poskytuje útok využívající postranních kanálů.

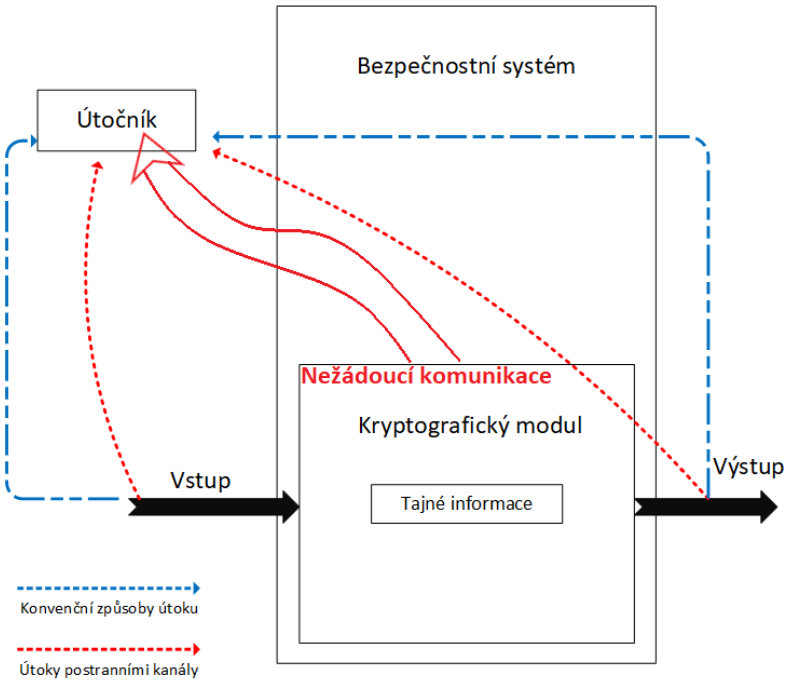
Pro definici útoků postranním kanálem můžeme využít kryptografický modul, který je ve své podstatě fyzicky implementovaný kryptografický algoritmus. Tento modul má ve své paměti uložený tajný klíč, kterým data šifruje i dešifruje. Šifrování, dešifrování, autentizace a další operace jsou přímo závislé na proudu odebíraném ze zdroje, modul také vyzařuje např. tepelné a elektromagnetické záření. Takovýchto stavů je mnohem více, všechny patří do kategorie postranních kanálů, tedy komunikace kryptografického modulu s okolním prostředím nežádoucím způsobem. Pomocí patřičné analýzy lze získat informace vedoucí k napadení kryptografického modulu, určení klíče a následné získání původní zprávy. Žádoucí způsob komunikace prochází po přesně daném kanálu s předem určeným vstupem i výstupem.

Proudová analýza (Power analysis, PA) představuje úspěšný typ útoku cíleného na dnes běžně používaná kryptografická zařízení. PA studuje proudovou spotřebu v závislosti na činnosti kryptografického modulu. Výsledkem analýzy jsou senzitivní informace vedoucí k odhalení šifrovacího klíče, kterých může útočník zneužít. Proudová analýza je velice populární, protože k její realizaci nepotřebuje útočník žádné specializované zařízení. Z tohoto důvodu by měly být kryptografické algoritmy implementovány s protiopatřeními, které zamezují realizaci proudové analýzy. Jedním z možných protiopatření je znáhodnění prováděných operací algoritmu.

Hlavním zaměřením této práce je proudový postranní kanál spolu s proudovou analýzou a protiopatřením proti jejímu úspěšnému provedení, jejichž popisu se věnuje teoretická část. Popsány jsou i další postranní kanály za účelem přiblížení problematiky postranních kanálů. Pro praktickou část, tedy vlastní realizaci PA, byla využita FPGA (Field Programmable Gate Array – Programovatelná hradlová pole), konkrétně vývojová deska SAKURA-G s implementovanou odlehčenou šifrou PRESENT. Dílčím cílem praktické části této práce je provedení jednoduché i diferenční proudové analýzy za účelem odhalení tajného klíče, a demonstrování nebezpečí, které proudová analýza přináší. Hlavní cíl práce je implementace protiopatření využívající hardwarové znáhodnění a prokázání jeho účinnosti na praktických výsledcích.

1 Popis postranních kanálů

Každý útok přes postranní kanál využívá jiný typ získané informace. Užitečné informace jsou v tomto případě nejčastěji měřitelné fyzikální veličiny nebo jiné skutečnosti, vznikající před, během nebo po dokončení šifrování. Pro lepší pochopení útoku je na obrázku 1.1 zvýrazněna komunikace modulu s okolím a tedy i s útočníkem.



Obr. 1.1: Zobrazení postranních kanálů a nežádoucí komunikace

Možnostmi, jak dělit útoky postranními kanály a krátké představením několika vybraných postranních kanálů je představeno v práci [1]. Jedním takovým způsobem je dělení podle způsobu zásahu útočníka do kryptografického modulu:

- aktivní způsob zásahu,
- pasivní způsob zásahu.

Aktivním zásahem útočník vytváří nestandardní situaci, například manipulací se vstupy nebo prostředím, ve kterém se modul nachází. V takto nestandardní situaci je modul nucen k „nestandardnímu“ chování. Na základě tohoto chování je útočník schopný zjistit tajný klíč. Pasivním zásahem je pouze sbírání informací z běžných činností a postupů modulu. Útočník žádným způsobem neovlivňuje běh modulu, pouze naslouchá. Dalším dělením je rozsah fyzického zásahu do zařízení [2]. Každý zásah z této kategorie můžu být jak aktivním, tak pasivním zásahem:

- invazivní zásah,
- semiinvazivní zásah,
- neinvazní zásah.

Invazivním zásahem do modulu je možné poškodit vodiče nebo zavést chyby. Upravením signálů změníme funkčnost zařízení. Po této úpravě nejsou žádné limity pro další manipulaci s modulem, vedoucí k získání tajného klíče. Pasivní možností by bylo pouze snímání proudících informací vně modulu, bez nutnosti zavádění chyb nebo poškození vodičů. Takto invazivní útok je ze zmíněné skupiny nejúčinější, avšak také nejnákladnější.

Semiinvazivní útok také proniká do implementace, ale nepoškodí čip přímo. Do čipu se zavedou chyby rentgenovým paprskem, světlem nebo elektromagnetickým polem, poté se sledují výstupní data. Pasivně se zavedou pouze elektromagnetické sondy, bez vytváření chyb.

Neinvazní útok nevniká do vnitřní implementace modulu, pouze naslouchá na fyzických rozhraních, která jsou volně přístupná. Při přivedení neobvykle vyššího napětí nebo během provozu čipu ve vysokých teplotách unikají informace, které jsou využitelné. Díky tomuto neinvaznímu přístupu takto realizovaný útok není možné odhalit a je častěji využívaný, také kvůli jeho „nižší“ nákladnosti.

Každý postranní kanál musí být důkladně analyzován. Takováto analýza je realizována prostřednictvím dvou základních typů:

- jednoduchá analýza (symple analysis),
- diferenční analýza (differential analysis).

Jednoduchá analýza, jak už název napovídá, je zpracování získaných výsledků jednoduchým (základním) způsobem. Útočník je schopen utajované informace vypozorovat a vyhodnotit. Pro práci s diferenční analýzou je potřeba využít matematických funkcí. Tato analýza je velice výhodná zvláště v případě, kdy jsou utajované informace skryté před jednoduchou analýzou. Hodnotí pouze ty faktory, které jsou odlišné nebo unikátní v porovnání s dalšími možnostmi.

Další část práce se bude podrobněji věnovat konkrétnějším druhům postranních kanálů. Vybrané kanály byly představeny již ve zmíněném článku [1], avšak pro představení problematiky byl vybrán jen jejich zlomek:

- akustický postranní kanál (acoustic side-channel),
- chybový postranní kanál (fault side-channel),
- elektromagnetický postranní kanál (electromagnetic side-channel),
- časový postranní kanál (timing side-channel),
- proudový postranní kanál (power side-channel).

1.1 Akustický postranní kanál

Množství zařízení, během svých výpočtu, produkuje zvukový šum kvůli vibracím některých hardwarových komponent. Tento šum, nebo hluk, není jen obtíž, ale obsahuje důležité informace spojené s výpočty během zabezpečovacího procesu. Hluk

z ventilátorů nebo pevného disku nese velmi „hrubé“ informace a nejsou moc vhodné pro kryptoanalýzu. Daniel Genkin, Adi Shamir a Eran Tromer se zaměřují na vibrace vytvořené elektrickými komponentami, které dle jejich práce [3] jsou mnohem účelnější. Vibrace jsou převážně tvořené napěťovými regulačními obvody, které korelují se systémovou aktivitou a CPU (Central Processing Unit – Centrální procesorová jednotka) významně mění odběr proudu podle aktuálně prováděné operace.

Další možností je odposlech stisku kláves na klávesnici nebo zvuky jehličkové tiskárny. Možného protiopatření při odposlechu kláves, je docíleno generováním zvuků, které náleží do stejného spektra a stejné formy jako stisk kláves. Takové zvuky jsou náhodně přehrávány. Pro větší účinnost se použije více různých variací, těchto „stisknutých kláves“.

1.2 Chybový postranní kanál

Z převážné části se analyzují fyzikální vlastnosti kryptografických zařízení. Chybový postranní kanál se však zaměřuje na chybová hlášení, která slouží jako oznámení pro uživatele. Útočník, za účelem odhalení tajného klíče, generuje chyby a analyzuje získaná chybová hlášení. Takováto hlášení jsou z bezpečnostního hlediska spíše přehlížena, ale obsahují i využitelné citlivé informace. Útočník nemusí invazivně přistupovat k čipu kryptografického modulu, aby navodil elektrický kontakt. Stačí odstranit pouzdro a vyslat optický puls, další možností je přistoupit k frézování, leptání. Postup bez jakéhokoliv fyzického dopadu, je možný při vystavení modulu změnám klimatických podmínek, dočasně jej přetaktovat na výší takt, než je uzpůsobený. Další z mnoha možností je zvýšení/snížení operačního napětí.

Míra úspěšnosti útočníka je velmi závislá na jeho schopnosti vložení chyb a následné vyhodnocení získaných chybových zpráv.

V roce `1997` vznikla práce [4], kde jsou, mimo jiné, prováděny testy na odolnost kryptografického algoritmu RSA (R. Rivest A. Shamir a L. Adleman – Jména autorů asymetrické šifry) útokem na ním vytvořené digitální podpisy, použitím jednoho špatného a jednoho správného podpisu stejné zprávy. Další implementace RSA, založené na čínské větě o zbytcích, mohou být také prolomitelné pomocí uměle vyvolaných hardwarových chyb.

Následujícího roku `1998`, Daniel Bleichenbacher [5], upravil formát šifrované zprávy takovým způsobem, aby vyvolalo chybové hlášení dešifrovacího zařízení, z čehož byl schopen dešifrovat komunikaci. Pro úspěšnou realizaci útoku, což představuje získání informací o tajném klíči, je však potřeba více než několik milionů dotazů na kryptografický modul.

1.3 Elektromagnetický postranní kanál

Každý kryptografický modul je soubor elektronických částí. Těmito částmi protéká elektrický proud, se kterým dále pracují. Průtok elektrického proudu je závislý na právě zpracovávaných datech. Elektronickou součástí je například tranzistor. Tranzistor, během své činnosti, mění svůj stav z logické 0, na logickou 1. Stav tranzistoru se mění podle aktuálně prováděné operace. Změna stavu je následně promítána do intenzity sledovaného elektromagnetického pole. Takto produkované elektromagnetické záření, je pro útok velmi užitečné. Část záření proniká do země, pokud je však dostatečná část vyzařována do okolí, kde ji můžeme odchytit, analyzovat a zneužít získané informace. Analýza může být provedena pomocí jednoduché elektromagnetické analýzy nebo účinější diferenciální elektromagnetické analýzy. Takto je útočník schopen získat tajné informace s pomocí speciálního vybavení.

V případě jednoduché elektromagnetické analýzy jsou kladeny velké nároky na útočníkovy znalosti v oblasti implementace kryptografického algoritmu a provedením i funkčností kryptografického zařízení. Na druhou stranu je výhodou malý počet potřebných naměřených průběhů. V práci [6] je podrobně popsán útok na implementaci asymetrického algoritmu RSA. RSA má základ v matematické operaci modulárního mocnění. Pro vypočítání modulárního mocnění se nejčastěji, v kryptografických modulech, používá algoritmus **Square and Multiply** 1.1. Algoritmus postupně pracuje s jednotlivými bity tajného klíče ($d[i]$). První se provádí modulární mocnění podmíněné modulárním násobením. Modulární násobení je podmíněné hodnotou bitu klíče. Pokud je právě zpracovávaný bit roven jedné, provede se modulární násobení, které zvýší hodnotu elektromagnetického pole. Počet kroků je omezen počtem bitů klíče, tento počet je reprezentován písmenem (b). Vstupní data (m), útočník zná a může je zaměňovat, dle libosti. Útočník je schopen jednoduše vypozorovat, z naměřeného elektromagnetického průběhu, ve kterém kroku bylo provedeno modulární násobení. Nakonec získá úplnou hodnotu tajného klíče.

Ukázka algoritmu 1.1: Square and Multiply

```
int R = m;              // globálně deklarovaná proměnná R
  for (int i = 0; i < (b-1); i++)
  {         // cyklus pro průchod jednotlivých bitů klíče
    R = (R*R) mod n;  // umocnění proměnné R "square"
    if (d[i] == 1)    // pokud je bit klíče roven 1
      R = (R*m) mod n;// provede se násobení "multiply"
  }
  return R;         // vrací výslednou hodnotu proměnné R
```

Další možností je využití kolize. Tento způsob útoku je také popsán v práci [6]. Útočník potřebuje provést alespoň dvě šifrování, které pozoruje. Každý průběh šifrování obsahuje jiný vstupní řetězec, ale stejný klíč. Využívá se skutečnosti, že ve dvou šifrovacích průbězích může být stejná specifická mezilehlá hodnota. Mezilehlá hodnota se srazí, pokud mezilehlá hodnota v prvním šifrovacím cyklu je rovna mezilehlé hodnotě druhého cyklu. Tato rovnost lze vyjádřit vztahem:

$$v_j = f(d_{i1}, k_j) = f(d_{i2}, k_j),$$

kde d_{i1}, d_{i2} jsou různé vstupy pro šifrování a k_j je tajný klíč.Kolize nenastává pro každou hodnotu klíče, ale pouze pro určitou podmnožinu. Jakákoliv kolize snižuje velikost prohledávaného prostoru pro výsledný tajný klíč. Čím více kolizí vyvoláme, tím je získání tajného klíče snazší.

Diferenciální elektromagnetickou analýzu využila skupina čínských vědců v práci [7], na symetrickém algoritmu AES. Největší únik citlivých informací je v tomto případě při provádění operace **SubBytes**.

Elektromagnetický postranní kanál je například používaný i pro útoky na kreditní karty, neboli **smart cards**, jako tomu je v práci [8]. Tento kanál je možné využít i pro rekonstrukci obrazu monitoru. Toto dokázal vědec Wim van Eck, z Nizozemska, v roce 1985 [9]. Také vědec Pankaj Rohatgi, ve své práci [10], popisuje Elektromagnetický postranní kanál, jeho analýzu i určitá protiopatření. Například jako jednu z možností uvádí přepracování obvodu, pro snížení nechtěných vyzařování, například pomocí elektromagnetického stínění. Možností je i přivedení dodatečného šumu, pro skrytí jednotlivých operací.

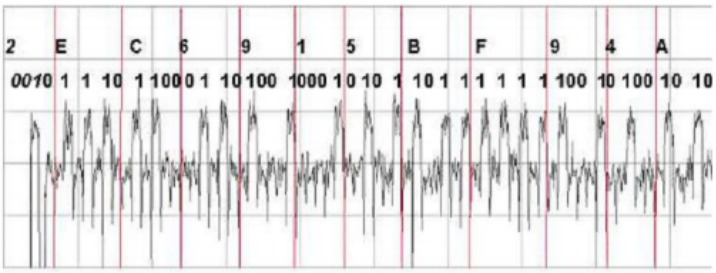
1.4 Časový postranní kanál

Prvním publikovaným postranním kanálem, roku 1996, je Časový postranní kanál [11]. Útok pomocí Časového postranního kanálu byl publikován na vědecké konferenci a byla publikována pouze jeho základní myšlenka, nikoli celé provedení. Tuto myšlenku publikoval a rozvinul, kryptolog amerického původu, Paul Carl Kocher.

Útok se zaměřuje na implementaci algoritmu a měří čas, který kryptografický modul potřebuje na vykonání operace. Naměřené hodnoty se analyzují, poté je možné přímo vyčíst, kdy a k jaké kryptografické operaci došlo. Tento útok je možný díky zkutečnosti, že každá logická operace potřebuje čas na vykonání, instrukce procesoru jsou časově odlišné, i větvení programu, čili vykonávání podmíněných operací, trvá nějaký čas. Logickou operací je například čtení dat a jejich zápis do paměti. Potřebné množství času se liší podle zadaného vstupu, nebo uloženém klíči pro šifrování/dešifrování. Vše tedy spočívá v souvislosti mezi hodnotou klíče, v kryptografickém modulu, a dobou potřebnou k výpočtu kryptografického algoritmu.

Časový útok na RSA

Nejčastější, a snad i nejpřehlednější, ukázkou časového postranního kanálu je u kryptografického modulu s implementovaným algoritmem RSA. Je to z důvodu přítomnosti algoritmu **square and multiply**, který je podmíněný hodnotou bitu klíče. Podrobněji popsán v předchozí kapitole se jménem **Elektromagnetický postranní kanál** s příkladem kódu 1.1. V kódu 1.1 vidíme smyčku, která se opakuje pro každý bit klíče. Časová náročnost je větší, pokud bit klíče odpovídá hodnotě jedna. Pokud je hodnota nula, neprovede se podmíněný úsek kódu a smyčka je dokončena dříve, než v opačném případě. Na naměřených průbězích je možné vidět, přesné hodnoty použitého klíče. Pro lepší vizuální pochopení použijeme obrázek 1.2 z práce [6], která se zabývá elektromagnetickou analýzou, ale pro přiblížení časového průběhu je obrázek dostačující.



Obr. 1.2: Naměřený průběh algoritmu square and multiply [6]

Při reálném pokusu o napadení kryptografického modulu touto metodou, je určení jednotlivých průchodů smyčkou velmi složité, přijatelnějším i snazším způsobem je vycházet z celkové doby při provádění operace a zjištění Hammingovi váhy. Hammingova váha reprezentuje počet nenulových bitů tajného klíče.

Útok pomocí časového postranního kanálu je využitelný i na jiné kryptografické algoritmy, nejen na RSA, ale i například na algoritmus Diffie-Hellman a další, jak uvádí pan Kocher ve své práci [11].

Protiopatření proti časovému útoku na algoritmus RSA, je více. Jednou z možností je „vycpání“ otevřené zprávy náhodnými bity. Metoda je pojmenována OAEP (Optimal Asymmetric Encryption Padding). Technika OAEP je směsí permutace otevřeného textu a vložení náhodných bitů. Příjemce zašifrované zprávy má instrukce o tom, jak postupovat při dešifrování a zbavit se této „vycpávky“. Rozsah zašifrované zprávy je nyní mnohem větší, a proto je její využití pro útok mnohem složitější. Její dešifrování je, pro útočníka bez znalosti pozic každého vyplňujícího bitu, skoro nereálné. Útočník si musí sám zvolit vyplňující bity, které odstraní.

To jej může přivést k vytvoření nesmyslného textu. Pokud rozluštěný text bude smysluplný, nemusí to však znamenat, že je správný.

Dalším způsobem je znáhodnění algoritmu [12], které je využité během výpočetních úkonů algoritmu. Znáhodnění způsobí rozmazání naměřených časových průběhů a jejich nepoužitelnost pro útočníka. Útočník není schopen ani poznat délku klíče, který byl použitý při šifrování. Tato informace je však klíčová, pro každý útok.

Časový útok na AES

Útok na algoritmus AES je možné realizovat pomocí vyrovnávací paměti **cache** [13]. Tato paměť ukládá kopii právě zpracovávaných dat a data jim následující, protože je velmi pravděpodobné, že budou vyžadována v dalším kroku procesoru. Tento způsob překonává zpoždění, které by nastalo při přistupování k datům na vyšší vrstvě paměťové architektury. Pokud procesorem požadovaná data, budou v cache paměti, doba přístupu k těmto datům bude krátká. V opačném případě, se přistoupí k vyšší vrstvě vyrovnávací paměti, která je pomalejší. Pokud ani zde nejsou vyžádaná data, je třeba přistoupit do hlavní paměti, ta je nejpomalejší. Dosažená data, také jim následující, budou znovu uložena do úrovní vyrovnávací paměti, pro případné další vyžádání od procesoru a dosažení optimalizace rychlosti. Paměťový prostor cache není nekonečný, proto nově ukládaná data jsou zaměněna s daty staršími. Tímto způsobem vzniká konkurence mezi nově ukládanými daty a těmi předchozími na stejném řádku paměti. Vznikly útoky postranními kanály, které využívají a analyzují chování cache paměti [14]. V této sekci je však důležité využití cache paměti z hlediska časové náročnosti.

Práce [13], popisuje také tento útok. Na stejném procesoru spustíme dva procesy. Jeden proces bude provádět kryptografické operace, pojmenujeme si jej „Crypto“, druhý se bude snažit ukrást jeho tajný klíč, to bude „Spy“. Tato pojmenování jsou často užívaná při popisu tohoto typu útoku. „Spy“ sleduje změny v čase, který „Crypto“ potřebuje k provedení kryptografické operace. Odvodí si potřebné informace k získání klíče, z odhadování počtu přístupů do paměti, které jsou spojené s tajným klíčem, a vedou k absenci procesorem vyžádaných dat, tedy k delší přístupové době. [15] Při odhadování kolizí cache paměti při posledním průběhu algoritmu AES, založených na známém zašifrovaném textu a odhadovaném bitu klíče, využijeme statistickou analýzu. Tato analýza porovná odhad s dobou potřebnou k výpočtu. Výsledkem bude nejpravděpodobnější kandidát na bit tajného klíče.

Algoritmus AES lze zabezpečit proti časovému útoku pomocí **Cache Partitioning**, neboli rozdělení vyrovnávací paměti. Přidělí se místa v paměti pro načtení hodnot využívaných tabulek. Tyto tabulky budou umístěni v prostoru paměti, kde

nedojde k jejich přepsání jiným procesem, dokud se nedokončí šifrování. Vzor přistupování k datům vyrovnávací paměti se změní, během šifrování.

Dalším způsobům, i výše zmíněnému, se věnuje podrobněji práce [16], kde je probrán například **Cache No Fill mode**. Tento mód způsobí, že vyrovnávací paměť obsluhuje pouze požadavky na data, která obsahuje. Pokud data obsažena nejsou, požadavek vyřizuje hlavní paměť, to zabrání přepisování hodnot uložených ve vyrovnávací paměti. V prvním kroku protiopatření se načte tabulka s-box (lookup table). Poté se zapne **no-fill mode**, spustí se proces šifrování. Nakonec se **no-fill mode** vypne. To vede ke každé shodě vyžadovaných dat procesorem, čas je co nejvíce konstantní a neunikají žádné informace použitelné útočníkem. Nevýhodou je, upřednostnění šifrovacího procesu, před procesy ostatními, dokud není **no-fill mode** vypnut.

1.5 Proudový postranní kanál

Stejně jako časový postranní kanál, proudový postranní kanál byl předveden kryptologem, Paulem Carlem Kocherem [17], v roce 1999. Útok proudovým postranním kanálem se zaměřuje na analýzu proudové spotřeby kryptografického modulu během prováděných operací. Získání tajného klíče je možné bez jakéhokoliv invazivního přístupu do kryptografického modulu. Základem těchto modulů je nejčastěji technologie CMOS (Complementary Metal-Oxide-Semiconductor). Důležitou charakteristikou těchto CMOS obvodů je velká odolnost vůči šumu a malý statický proudový odběr. CMOS obvod je složen z malých logických buněk. Součet proudové spotřeby každé jednotlivé buňky, je celková proudová spotřeba celého CMOS obvodu. Z tohoto důvodu je celková proudová spotřeba přímo úměrná počtu buněk v obvodu a jejich spojení.

Základní částí v CMOS obvodu je invertor složený ze dvou, napětím řízených tranzistorů, s opačnou vodivostí. Schématickou značku představuje obr. 1.3 upravený z knihy[18]. Složitější buňky obsahují více tranzistorů, ale pracují na stejném principu. Skrze tranzistory uvnitř invertoru protékají tři druhy proudů. Svodový proud vzniká ve chvíli, kdy je napájení vedeno do země. Tento děj nastává v případě, když dojde ke změně stavu invertoru, oba tranzistory jsou na malý okamžik otevřeny, a napájení je přes ně zkratováno. Tento proud je anglicky nazýván **direct path current**. Svodový proud způsobuje průměrnou výkonovou spotřebu, označenou P_{svod}, během času T. Okamžitá výkonová spotřeba invertoru, označená $p_{svod}(t)$. Svodový proud I_{svod} a časový úsek, po který jsou oba tranzistory otevřeny, t_{svod}. $P_{0\to1}$, znázorňuje pravděpodobnost přechodu do stavu 1. U_{nap} označuje konstantní

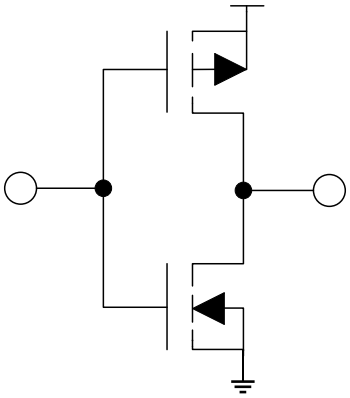
napájecí napětí. f značí kmitočet spínání. Tyto proměnné vložíme do rovnice:

$$P_{svod} = \frac{1}{T}\int_0^T p_{svod}(t)\mathrm{d}t = P_{0\to1} \cdot f \cdot U_{nap} \cdot I_{svod} \cdot t_{svod}. \tag{1.1}$$

Nabíjení/vybíjení parazitní kapacity, je dalším proudem. Spoje mezi buňkami a řídící elektrody dalších tranzistorů mají určité kapacity, ty jsou reprezentovány touto parazitní kapacitou **charge/discharge current**. Vzorec pro spočítání dynamické spotřeby invertoru během nabíjení parazitní kapacity [19]:

$$P_{dyn} = \frac{1}{T}\int_0^T p_{nab}(t)\mathrm{d}t = P_{0\to1} \cdot f \cdot C \cdot U_{nap}^2, \tag{1.2}$$

Stejně jako u předchozího vzorce, je U_{nap} konstantní napájecí napětí, f značí kmitočet spínání. $P_{0\to1}$, znázorňuje pravděpodobnost přechodu do stavu 1 a $p_{nab}(t)$ značí okamžitou výkonovou spotřebu během nabíjení parazitní kapacity C, za čas T. Rovnice (1.1) pro P_{svod}, a rovnice (1.2) pro P_{dyn}, patří pod dynamickou výkonovou spotřebu, která bude vysvětlena dále.



Obr. 1.3: Schéma invertoru CMOS

Třetí proud je zbytkový, jehož velikost udává statickou výkonovou spotřebu. Tento proud je využíván pro statickou analýzu, která vede k útoku na kryptografický modul. Na hodnotě, která se nachází na vstupu do obvodu, závisí velikost zbytkového proudu. Z důvodu opačných vodivostí tranzistorů, má zbytkový proud jinou hodnotu dle toho, který tranzistor je právě otevřený. Ze znalosti otevřeného textu a naměřených průběhů zbytkového proudu, je útočník schopen získat hodnoty tajného klíče. Zbytkový proud se nazývá **leakage current**.

Je vhodné uvést také vzorec pro průměrnou výkonovou spotřebu CMOS obvodu za čas T [20]:

$$P_{obv} = \frac{1}{T}\int_0^T p_{okam}(t)\mathrm{d}t = \frac{U_{nap}}{T}\int_0^T i_{okam}(t)\mathrm{d}t. \tag{1.3}$$

Zde U_{nap} značí konstantní napájecí napětí, $I_{okam}(t)$ je celkový okamžitý proud a okamžitá výkonová spotřeba $p_{okam}(t)$.

Celková výkonová spotřeba se skládá ze součtu dvou stavů, kterých nabývá invertor. Pokud je invertor ve stabilním stavu, odebírá statickou výkonovou spotřebu. Z toho plyne, pokud je například na výstupu $0 \rightarrow 0$, nebo $1 \rightarrow 1$, jedná se o stabilní stav, a tedy statickou výkonovou spotřebu. Dynamická výkonová spotřeba vzniká, když invertor přepne stav na svém vstupu nebo výstupu. Tedy $0 \rightarrow 1$, nebo $1 \rightarrow 0$. Lze tedy říci, že přechod z $0 \rightarrow 0$ má přibližně stejnou výkonovou spotřebu jako přechod $1 \rightarrow 1$. Na druhé straně, přechod $0 \rightarrow 1$, nebo $1 \rightarrow 0$ je rozdílný. Závěrem, dynamická spotřeba je přímo závislá na aktuálně zpracovávaných datech. Pro lepší přehlednost je zde tabulka 1.1

Tab. 1.1: Přechody invertoru a výsledná výkonová spotřeba

Přechod	výsledná výkonová spotřeba
$0 \rightarrow 0$	statická
$1 \rightarrow 1$	statická
$1 \rightarrow 0$	statická + dynamická
$0 \rightarrow 1$	statická + dynamická

Závislost dynamické výkonové spotřeby na vstupním textu je využívána pro útoky pomocí proudové analýzy. Závislost dynamické výkonové spotřeby je možné spočítat pomocí vzorce 1.2. Průběhy šifrování kryptografického modulu, pro různé vstupní texty, jsou sbírány, ukládány a poté vyhodnocovány. Vyhodnocení může být realizováno pomocí jednoduché proudové analýzy, tedy vyčtením tajného klíče přímo z naměřeného průběhu. Pokud není možné data přímo vyčíst, útočník může využít diferenční proudovou analýzu a její matematické zpracování, pro požadovaný výsledek. Proudová analýza také nemusí být příliš nákladná. Například pro měření není nutné nic víc, než osciloskop. Je možné použít přenosnou měřící kartu, nebo přímo integrovanou do počítače. Tyto měřící karty není těžké získat. Detailnějším vysvětlením zmíněných analýz, jejich použitím a zabezpečením kryptografického modulu, proti těmto analýzám, se bude v nadcházejících stránkách a kapitolách zabývat tato práce.

2 Proudová analýza

Proudová analýza se rozděluje na jednoduchou analýzu (SPA) a diferenční analýzu (DPA). Jednoduchý proudová analýza je anglicky nazývána Simple Power Analysis. Hodnoty se v této analýze odečítají přímo, tedy jedná se o přímou interpretaci dat, druhým způsobem je využití šablon. U této analýzy je nutná znalost kryptografického modulu, jeho práci a metody či funkce. Další nutná znalost se týká implementovaného kryptografický algoritmu, jeho průběhů a prováděných operací.

Diferenční proudová analýza je pojmenována Differential Power Analysis. Tato analýza se skládá z pěti kroků a matematických operací. Je účinnější než SPA. DPA může být založena na:

- korelačním koeficientu,
- rozdílu středních hodnot,
- vzdálenosti středních hodnot,
- využití šablon.

Při využití šablon, útočník musí mít pokročilé vědomosti o průběhu proudové spotřeby kryptografického modulu, u předchozích tří variant tomu tak není, což se může jevit jako výhoda. Použité šablony pro DPA jsou převážně rozšířené šablony použité pro SPA.

2.1 Jednoduchá proudová analýza

Jednoduchá proudová analýza pracuje s malým množstvím naměřených proudových průběhů. Je výhodné ji používat převážně na asymetrické algoritmy pracující s jednotlivými bity tajného klíče. Každá vykonaná operace odebírá jiné množství proudu. Pokud je možné rozeznat vykonávané operace, tak se jedná o velké bezpečností riziko. Například práce [21], jejímž cílem je využít možnosti SPA na „chytré karty“.

Přímá interpretace

V praxi není přímé odečtení klíče vždy možné, například kvůli velkému množství šumu. Častým příkladem je využití SPA na algoritmus RSA, kvůli existenci algoritmu **square and multiply**. Tento případ se již řešil v kapitole 1.3, kde byl ukázán algoritmus square and multiply 1.1. Výsledný naměřený průběh je zobrazen na obrázku 1.2. Z obrázku lze lehce vyčíst hodnoty tajného klíče. Práce [22] se zabývá získáním tajného klíče z „chytrých karet“ obsahujících algoritmus DES. V první části práce je použita jednoduchá proudová analýza. Naměřený proudový průběh a využití přímé interpretace dat pro získání tajného klíče zobrazuje také práce [23].

Práce popisuje jako možnosti protiopatření přidání šumu, pro skrytí tajného klíče, nebo vytvoření nadbytečných operací, které změní typický průběh algoritmu.

Využití šablon

Útok využívající šablony, neboli *Template-Based Attack*, je založen na možnosti charakterizovat proudový průběh vícerozměrným normálním rozložením. Toto rozložení je definováno vektorem středních hodnot (m) a kovarianční maticí (C), které dohromady tvoří šablonu (m, C). *Reduced Template Attack* je odvozený útok, který využívá například pouze vektor (m) představující redukovanou šablonu, avšak za cenu snížení informací o vztazích mezi body, což je nežádoucí [24]. Na vstup kryptografického zařízení, jehož tajný klíč chce útočník získat, jsou vložena pro každý průběh šifrování vždy jiná data d_i a různé odhadované klíče k_i. Výsledné proudové průběhy si útočník zaznamená. Dále je nutné vypočítat kovarianční matici a vektor středních hodnot. Tyto kroky vedou k vytvoření šablon pro data i klíče. Šablony jsou vyjádřené matematickým vztahem:

$$(d_i, k_i) : h_{d_i,k_i} = (m, C)_{d_i,k_i}.$$

V poslední části útočník vypočítá hustotu pravděpodobnosti vícerozměrného normálního rozložení a naměřeného proudového průběhu. K tomuto výpočtu slouží vzorec [20]:

$$p(t; (m, C)_{d_i,k_i}) = \frac{exp(-\frac{1}{2} * (t-m)' \cdot C^{-1} \cdot (t-m))}{\sqrt{(2 \cdot \pi)^T \cdot det(C)}}.$$

Útočník hledá největší hodnotu pravděpodobnosti. Při jejím nalezení útočník pravděpodobně získal tajný klíč. Rakouský student Marcel Medwed, ve své diplomové práci [25], popisuje jednoduchou i diferenční proudovou analýzu spolu s využitím šablon a aplikuje ji na ECDSA. Tedy na Elliptic Curve Digital Signature Algorithm, Algoritmus digitálního podpisu využívající eliptické křivky.

2.2 Diferenční proudová analýza

Pan Kocher v roce 2011 publikoval práci [26], kde popisuje diferenční proudovou analýzu, její druhy i provedení převážně na kryptografický algoritmus AES. Diferenční proudová analýza využívá velké množství naměřených průběhů, během procesu šifrování, nebo dešifrování s různými vstupními daty. DPA nezpracovává proudové průběhy v časové ose, jako tomu je u SPA, ale analyzuje závislost proudové spotřeby na vstupních datech, v daný časový okamžik. Tento typ analýzy je velmi závislý na počtu naměřených proudových průběhů z čehož vyplývá, že čím víc naměřených

průběhů, tím přesnější jsou výsledky spolu se shodou mezi skutečnou a odhadovanou proudovou spotřebou. Přímá ukázka na algoritmu DES je popsána krok po kroku v práci [27]. Další možností je práce [28] pánů Kocher, Jaffe a Jun, kteří sepsali představení do diferenční proudové analýzy, srovnávají ji s jednoduchou proudovou analýzou. Věnují se také opatření proti DPA, skrývání i maskování. Metody protiopatření budou popsány v následující kapitole této práce. Průběh diferenční proudové analýzy:

- výběr mezivýsledku aplikovaného algoritmu,
- měření proudové spotřeby,
- výpočet odhadovaných mezivýsledků,
- přiřazení odhadovaných mezivýsledků hodnotám proudové spotřeby,
- porovnání odhadovaných mezivýsledků s naměřenými průběhy.

Výběr mezivýsledku aplikovaného algoritmu

Výběr mezivýsledku použitého kryptografického algoritmu je první krok. Tento mezivýsledek musí být funkcí například otevřeného, nebo šifrovaného, textu d s tajným klíčem k. Není zapotřebí celý klíč, stačí jeden bajt. Matematicky zapsaná funkce: $f(d, k)$.

Měření proudové spotřeby

Při šifrování nebo dešifrování bloků dat, si útočník naměří proudovou spotřebu kryptografického modulu. Použité různé bloky dat označíme písmenem D, z důvodu označení zpracovávaných dat jako d v sekci 2.2. Zpracovávaná data d jsou využívána při výpočtu mezivýsledku určeného v předchozím kroku. Data d musí být útočníkovi během kryptografické operace známá a jejich hodnoty tvoří vektor $\mathbf{d} = (d_1, \ldots, d_D)'$. Během všech kroků šifrování nebo dešifrování si útočník ukládá naměřené proudové průběhy, které jsou shodné se zpracovávanými daty d_i, které jsou označením hodnoty prováděného kroku. Pořadové číslo prováděného kroku je označeno i. Tyto proudové průběhy si označíme jako $t_i' = (t_{i,1}, \ldots, T_{i,T})$, zde T značí délku naměřené spotřeby. Pro každý blok dat D jsou měřeny proudové průběhy, které umístíme do matice $\mathbf{T}$ velikosti $D \times T$. Získaná matice musí obsahovat zarovnané hodnoty tak, aby v každém sloupci byly hodnoty stejné operace. Pro takové zarovnání je potřeba správná synchronizace s měřícím zařízením.

Výpočet odhadovaných mezivýsledků

V této části útočník vypočítá odhadované mezivýsledky pro všechny možnosti tajného klíče k, které zapíše vektorem $\mathbf{k} = (k_1, \ldots, k_K)'$. Zde K je celkový počet klíčů.

Nyní pomocí vektoru **d**, z předchozího kroku, a vektoru **k** je útočník schopen vypočítat odhadované mezivýsledky $f(d, k)$ pro všechny odhady klíče K a šifrovací operace bloků dat D. Vzniká matice **B** s rozměry $D \times K$ ze vztahu

$$v_{i,j} = f(d_i, k_j); i = 1, \ldots, D; j = 1, \ldots, K. \tag{2.1}$$

Uvnitř matice **V** sloupce j obsahují hodnoty všech možných klíčů, tedy i toho, který byl použit v kryptografickém modulu. Hledaný tajný klíč si označíme jako k_{ck}.

Přiřazení odhadovaných mezivýsledků

Nyní je potřeba přiřadit odhadované mezivýsledky matice **V** k předpokládaným hodnotám proudové spotřeby, tedy k matici **H**. Nyní se využije simulace proudové spotřeby kryptografického modulu, k tomu slouží nejčastěji model Hammingovy vzdálenosti nebo model Hammingovy váhy. Vybraný model přiřadí každému výsledku $v_{i,j}$ odhadovanou hodnotu proudové spotřeby $h_{i,j}$.

Porovnání odhadovaných mezivýsledků

Poslední částí DPA je porovnání odhadovaných mezivýsledků s naměřenými průběhy, neboli hodnoty z matice **H**, sloupce h_i a matice **T** sloupce t_j. Porovnáním nám vzniká matice **R** velikosti $K \times T$. Pole $r_{i,j}$ obsahuje výslednou hodnotu srovnání sloupců h_i a t_j. Čím větší je hodnota pole $r_{i,j}$, tím je vyšší shoda sloupců. Zařízení během šifrování nebo dešifrování počítá hodnotu mezivýsledku v_{ck} pro různá vstupní data. Z tohoto důvodu jsou naměřené průběhy, v některých časových úsecích, závislé na hodnotě mezivýsledku. Tyto úseky označíme ct, tedy sloupec t_{ct} obsahuje závislé hodnoty proudové spotřeby na hodnotě mezivýsledku v_{ck}. Odhadované hodnoty h_{ck} útočník simuloval na základě hodnot v_{ck}, proto jsou sloupce h_{ck} a t_{ct} závislé a vedou k největší hodnotě v matici **R**. Největší hodnota matice **R** bude označena jako $r_{ck,ct}$. Ostatní hodnoty matice **R** již nejsou tak vysoké, kvůli jejich malé závislosti. Nalezením nejvyšší hodnoty v matici **R**, bude útočník schopen nalézt index tajného klíče ck a časového okamžiku ct, které jsou výsledkem útoku. Pokud by však útočník nemohl nalézt bezpochyby největší hodnotu, a všechny by si byly skoro rovny, znamenalo by to, že nemá dostatečné množství naměřených proudových průběhů.

Porovnání odhadovaných mezivýsledků s naměřenými proudovými průběhy je realizováno pomocí

- korelačního koeficientu,
- rozdílu středních hodnot,
- vzdálenosti středních hodnot,
- využití šablon – (velmi podobné využití v SPA, jen rozšířený [29]).

2.2.1 Korelační koeficient

Tato metoda je jedna z nejznámějších pro určení lineární závislosti dvou náhodných proměnných. Korelační koeficient může nabývat hodnot $-1 \leq k \leq 1$. Pokud se korelační koeficient rovná hodnotě -1, jde o nepřímou závislost. Nepřímá závislost znamená, že změna v jedné skupině má opačný dopad na skupinu druhou. Při výsledku 0, na sobě porovnávané hodnoty mohou záviset, ale závislost nelze vyjádřit lineární funkcí. Hodnota 1 značí přímou závislost, tedy bezchybnou korelaci. Pro výpočet korelace existuje obecný vztah, my si jej však upravíme pro potřeby DPA. Využijeme sloupců h_i a t_j na základě prvků D. Průměrné hodnoty sloupců h_i a t_j označíme $\overline{h_i}$ a $\overline{t_j}$. Výsledný vztah má podobu [20]:

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i}) \cdot (t_{d,j} - \overline{t_j})}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i})^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \overline{t_j})^2}}. \tag{2.2}$$

2.2.2 Rozdíl středních hodnot

Základem této metody je srovnání naměřených hodnot dvou skupin a výpočtem jejich středních hodnot. V první řadě vytvoří útočník matici $\mathbf{H}$, která je binární a rozděluje naměřené proudové průběhy do dvou skupin. Sloupce matice $\mathbf{H}$ jsou funkcí vstupních dat d a odhadů tajného klíče k_i. Dalším krokem je rozdělení matice $\mathbf{T}$ podle h_i na dva soubory tak, že jeden soubor bude obsahovat ty řádky matice $\mathbf{T}$, kterých index odpovídá pozici nul vektoru h_i. Druhý soubor obsahuje řádky zbylé. Útočník vypočítá průměr řádků obou souborů. Správnost odhadu tajného klíče se projeví tak, že existuje výrazný rozdíl mezi sousedícími řádky obou souborů. Z toho plyne, že výsledkem je matice $\mathbf{R}$, kde řádky odpovídají rozdílům mezi soubory. Práce [30], která se zabývá útokem na šifru AES pomocí diferenční analýzy, podrobněji popisuje postup Rozdílu středních hodnot, neboli *Difference of Means*.

2.2.3 Vzdálenost středních hodnot

O této metodě lze říci, že je vylepšením metody pracující s Rozdíly středních hodnot. Tato metoda pracuje i se směrodatnými odchylkami. Metoda pracuje skoro stejně, jako předchozí metoda, ale střední hodnoty jsou srovnávány pomocí testu vzdálenosti středních hodnot, který zapojuje směrodatnou odchylku pro rozdělení dvou skupin $s_{i,j}$ jako dělitele ve vztahu:

$$r_{i,j} = \frac{m_{1i,j} - m_{2i,j}}{r_{i,j}}, \tag{2.3}$$

zde $m_{1i,j}$ je průměr řádků prvního souboru a $m_{2i,j}$ je průměr řádků druhého souboru.

3 Protiopatření proti proudové analýze

Protiopatření proti proudové analýze je důležité, neznamená to však úplnou nutnost odstranit celý postranní kanál. Pokud docílíme toho, že útočník nebude schopen efektivně provést proudovou analýzu, nebo kanálem nebudou unikat žádné citlivé informace. Nebezpečí však vzniká nové a to, že vznikne nová technika útoku. Proudová spotřeba nesmí být závislá na mezivýsledku a operacích, které kryptografický modul provádí, takto je splněný cíl protiopatření. Dvěma hlavními skupinami protiopatření jsou *maskování (masking)* a *skrývání (hiding)*. Andreas Hoheisel ve své diplomové práci [31] popisuje a implementuje níže popsaná zabezpečení na algoritmus AES. Tato bakalářská práce se však bude podrobněji zabývat metodou skrývání z důvodu dodržení zadání a bližšího zaměření.

3.1 Skrývání

Metoda skrývání má za cíl skrýt závislost proudové spotřeby na zpracovávaných datech, jejich hodnotě nebo hodnotě mezivýsledku. První možností je ve výrobě, tedy vyrobit zařízení tak, aby vykazovalo náhodnou proudovou spotřebu. Náhodná proudová spotřeba je taková spotřeba, která při stejných prováděných instrukcích, v hodinovém cyklu, nabývá jiných hodnot. Druhá metoda je opakem, vytváří proudovou spotřebu konstantní v hodinovém cyklu, pro všechny instrukce i zpracovávaná data.

Pravdou ovšem je, že takovýchto stavů není reálně možné dosáhnout, jsou však metody, jak se k těmto požadovaných stavům alespoň co nejvíce přiblížit. Pro znáhodnění proudové spotřeby slouží návrhy, které mají vliv na časovou oblast této proudové spotřeby. Znáhodnění můžeme dosáhnout, když v různých časových okamžicích, při každém spuštění, provedeme kryptografické operace algoritmu. Návrhy, které mají vliv na okamžitou velikost proudové spotřeby, se snaží o konstantní, nebo alespoň náhodnou proudovou spotřebu. Každá operace má nějakou charakteristickou proudovou spotřebu, tato spotřeba je měněna návrhy tak, aby byla všech operací stejná.

Nele Mentens ve své práci [32] popisuje možnosti rekonfigurace hardwarových zařízení a jejich srovnání na různých hardwarových architekturách. Práce je srovnáním dalších publikací. Hlavní použité hardwarové architektury jsou FPGA (programovatelná hradlová pole) a ASIC (integrovaný obvod pro specifické použití). Nezaměřuje se pouze na proudový postranní kanál, zmiňuje také další kanály a rozsah popsaných protiopatření. Použitými algoritmy jsou převážně algoritmus AES, odlehčená šifra PRESENT a eliptické křivky.

Odlehčená šifra PRESENT je také popsána v práci [33], kde je implementována na FPGA vývojovou desku. Hlavním záměrem práce je rekonfigurace FPGA, kdy poskytuje možnost načtení hardwarových okruhů na vyžádání. Různé skupiny obvodů mohou pracovat na stejném místě FPGA, zároveň však v jiném časovém slotu. Tyto skutečnosti zabraňují útočníkovi v odhadu následujících operací a určení, která operace byla kdy spuštěna nebo provedena. Praktická část práce popisuje, že ani 10 miliónů naměřených průběhů nestačilo k odhalení jakékoliv užitečné informace.

Odlehčená šifra PRESENT je popsána v kapitole 4.1 z důvodu implementace do vývojové desky SAKURA-G na které bude realizována praktická část bakalářské práce. Hlavním zdrojem při realizaci protiopatření byla práce[33]. Implementované protiopatření popisuje kapitola 6

3.1.1 Znáhodnění proudové spotřeby

K vytvoření algoritmu, který vykonává operace v náhodném pořadí vedla především skutečnost, že naměřené proudové průběhy musí být synchronizovány. V opačném případě je útok skoro nerealizovatelný, nebo útočník potřebuje mnohem větší počet naměřených proudových průběhů. Následující dvě techniky jsou v praxi často kombinovány. Při správném využití obou těchto technik je možné se co nejvíce přiblížit úplnému znáhodnění proudové spotřeby.

Technika prázdných operací, neboli *dummy operations*, vkládá prázdné operace do prováděného algoritmu, vždy na jiná místa. Při spuštění algoritmu jsou vygenerována umístění, kam se prázdné operace vloží a jejich počet je vždy stejný. Pokud by se počet prázdných operací měnil, útočník by byl schopen zjistit jejich počet opětovným měřením a odečtením časové náročnosti algoritmu. Čím více prázdných operací, tím více se mění jejich umístění v algoritmu a proudová spotřeba je znáhodněná. Počet prázdných operací má však vliv také na časovou náročnost, tedy provedení algoritmu s více prázdnými operacemi je časově mnohem náročnější. Nutností je tedy zvolit mezi rychlostí provedení algoritmu, nebo jeho větším znáhodněním. V práci [31] kapitole 4.5 popisuje a znázorňuje časovou náročnost mezi nemaskovaným, maskovaným a maskovaným i znáhodněným algoritmem. Souhrnná tabulka ukazuje velký nárůst časové náročnosti na 8 bitovém čipu „chytrých karet“.

Druhou technikou je přesouvání prováděných operací, neboli *shuffling of operations* [34]. Některé operace algoritmu nelze přesunout kvůli jejich návaznosti, avšak přesouvání operací neovlivní časovou náročnost. Jako příklad poslouží kryptografický algoritmus AES. Algoritmus využívá operaci *SubBytes*. Tato operace provádí šestnáct substitucí podle substituční tabulky a pořadí provedení není pevně dané, proto lze substituce provádět v náhodném pořadí. Stejně jako u techniky prázdných operací, jsou generována náhodná čísla pořadí, dle kterých se substituce provedou.

Práce [35] se zabývá softwarovým provedením metody skrývání, která využívá přesouvání operaci, jako protiopatření. Práce rozebírá několik možných provedení této metody a porovnává je mezi sebou. Testovaný modul má implementovaný algoritmus AES. Práce také poukazuje na fakt, že zjednodušené protiopatření, například použití pouze náhodných počátečních indexů nemá velké účinky a výsledek je stejný jako u nechráněného kryptografického modulu.

3.1.2 Konstantní proudová spotřeba

Informace, které unikají postranním kanálem musí být co nejmenší, nebo alespoň nerozeznatelné. Nutností je, aby odstup signálu od šumu byl co nejmenší, tedy hodnota *signal-to-noise ratio* (SNR) byla rovna nule. Možností je snížit hodnotu užitečného signálu na 0, nebo zvýšit hodnotu šumu na nekonečno. Ani jedna možnost není možná na 100%, avšak k požadovanému stavu se můžeme znatelně přiblížit [20]. Pro zvýšení šumu můžeme využít rušičky, které náhodně přepínají mezi paralelními a aktivními operacemi. Můžeme také provádět několik nezávislých operací zároveň. Pokud bychom dosáhli stejné proudové spotřeby pro všechna data i prováděné operace, užitečný signál by se rovnal nule. Musíme tedy vytvořit speciální buňky v logické struktuře tak, aby měli stejnou proudovou spotřebu, poté bude výsledkem konstantní proudová spotřeba celého zařízení. Možností je také filtrování proudové spotřeby. K tomu se využívají filtry, které odstraňují z proudové spotřeby datovou závislost a závislost ostatních komponent na prováděných operacích.

Signal-to-noise ratio podrobněji popisuje článek [36], kde je zkoumána pro algoritmy DES a AES a uváděna jako míra odolnosti vůči útoku postranními kanály. Možností jak zvýšit ochranu proti DPA je využít generátor šumu, který je zaměřen na odbourání korelace mezi odběrem proudu a použitým tajným klíčem [37].

3.2 Maskování

Jak název napovídá, využívá se tajná náhodná hodnota, která maskuje hodnotu mezivýsledku kryptografického algoritmu. Pomocí této hodnoty, masky, se mezivýsledek jeví jako náhodný. Hodnota masky je v každém průběhu šifrování nebo dešifrování jiná a útočník ji není schopen znát. Tato metoda zapříčiní, že výsledná proudová spotřeba nebude závislá na zpracovávaných datech i v případě, že bez aplikovaného maskování, na nich měla výraznou závislost.

Maska je tedy náhodná hodnota m, která je přidávána k mezivýsledku v. Maska se aplikuje buď exkluzivním součtem, tedy operací XOR ($\oplus$), což je nazýváno jako aditivní maskování. Multiplikativní maskování je prováděno operací násobení ($\otimes$). Dále jsou dva typy maskování, *booleovský* a *aritmetický*. Aritmetické maskování

je realizováno rovnicí s operací sčítání nebo násobení s modulem. Rovnice je vyjádřena jako $v_m = v + m(mod\ n)$. *Booleovské* maskování využívá operace XOR $v_m = v \oplus m$. Ne všechny používané algoritmy využívají jen aritmetických nebo boolean operací, a proto je potřeba maskování kombinovat. Kombinování však vyžaduje velké množství dalších operací, které přechody mezi maskováním umožňují. Nakonec výsledkem algoritmu je maska, která musí být odstraněna pro získání kryptogramu. Pro využívání masek je nutné pozměnit implementaci kryptografického algoritmu.

Při zpětném pohledu na rovnici $v_m = v \oplus m$ vidíme, že pro výpočet mezivýsledku v potřebujeme znát hodnoty v_m a m. Zjištění pouze hodnoty v_m nás nepřivede k jakékoliv informaci o mezivýsledku v. Na tomto principu pracuje maskování se sdílením tajemství, kdy je použito více masek na jednu hodnotu mezivýsledku a útočník je tedy nucen znát všechny hodnoty použitých masek.

Celým tématem maskování, provedením a detailnějším rozborem se zabývá disertační práce, kterou napsal Reparaz Oscar v roce 2016 [38], také značná část práce [34] je věnována maskování algoritmu AES. Maskování šifry PRESENT popisuje práce [39], kde roku 2016 skupina z Technologického institutu vědy a elektroniky v Pekingu realizovala úspěšný útok diferenční proudovou analýzou. Poté aplikovaly maskování na hodnoty S-boxu spolu s maskováním funkce permutace. Toto protiopatření zamezilo odhalení tajného klíče a zvýšení zabezpečení.

3.3 Provedení protiopatření

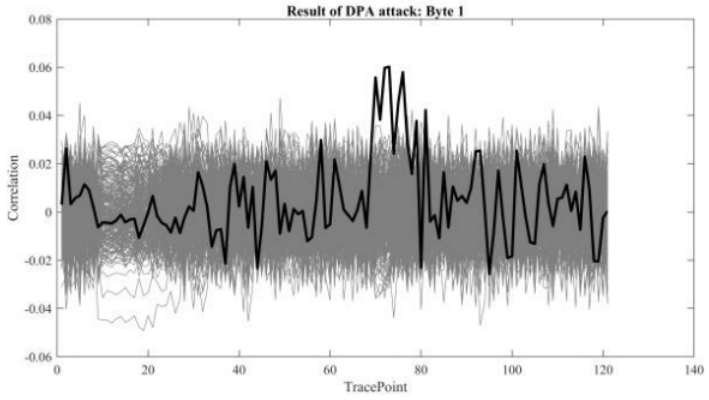
Protiopatření lze implementovat jak hardwarové, tak i softwarové. Softwarové je levnější a jednodušší pro přidání k již využívaným kryptografickým zařízením. Má však značná omezení a není tak účinné, jako hardwarové protiopatření, které je sice nákladnější, zato však účinější.

3.3.1 Hardwarové provedení

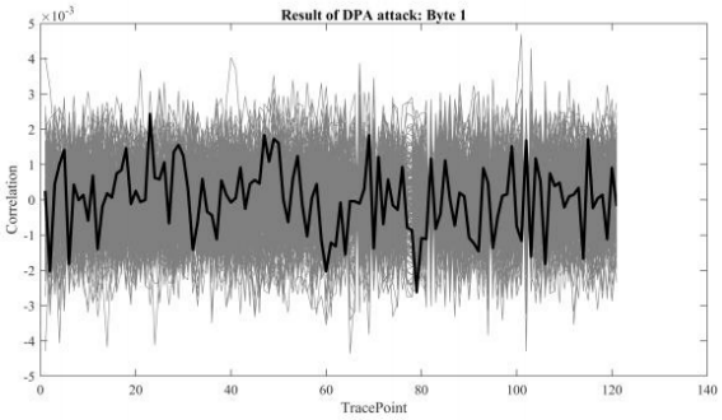
Implementovat protiopatření hardwarovým způsobem nám přináší mnohem více možností. Jako protiopatření můžeme využít vkládání prázdných operací a přesouvání operací. Realizace je stejná jako u softwarového provedení. Musíme však také myslet na fakt, že protiopatření proti jednomu typu útoku nemusí mít vliv na útok realizovaný jiným způsobem. Na hardwarové úrovni máme však navíc možnost, například náhodně měnit frekvenci hodinového pulzu, vynechat hodinový pulz a přepínat mezi více zdroji hodinového signálu. Z názvů metod lze vyčíst hlavní záměr, a tedy upravit hodinový signál takovým způsobem, aby nebylo možné synchronizovat proudovou spotřebu. Možnou volbou je zapojení šumových rušiček, které skryjí užitečný signál. Umístění více rušiček na jednom místě není nejlepší variantou, měli

by být rozmístěné v celém obvodu zařízení. Další možné protiopatření je vytvořit speciální obvod, který potlačuje užitečný signál unikající ze zařízení, jak je blíže popsáno v práci [40]. V neposlední řadě filtrování proudové spotřeby je účinné protiopatření, ke kterému se využívá spínacích kapacitorů, zdrojů konstantního proudu a prvků regulujících proudovou spotřebu.

Využití RLC filtrů se věnuje práce [41], kde je tento filtr vložen do napájecí cesty zařízení. Názorně je zde ukázáno, jak filtr skrývá průměrnou proudovou spotřebu, která je neskrytá na obrázku 3.1, kdy není možné rozpoznat ani počet provedených iterací 3.2.



Obr. 3.1: Nechráněný proudový průběh [41]



Obr. 3.2: Skrytý proudový průběh [41]

Podrobnějším popisem hardwarové implementace se zabývá práce [42], která popisuje redukci SNR a znáhodnění vykonávání algoritmu. Hlavním cílem, je vytvoření rovnice pro tvůrce kryptografických zařízení, která poslouží jako sumarizační prvek který říká, zda je použité protiopatření dostatečné, již v rané fázi výroby.

Nejčastěji se využívá kombinace hardwarové i softwarové implementace, nebo pouze implementace hardwarová. Dostatečné protiopatření se odvíjí od počtu naměřených proudových průběhů potřebných pro získání užitečné informace o tajném klíči.

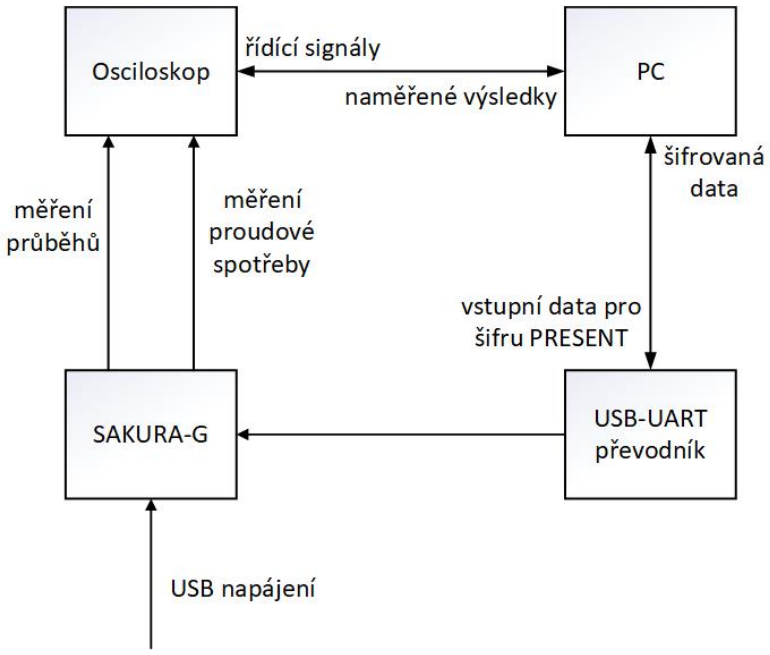
3.3.2 Softwarové provedení

Omezení softwarového provedení pramení ze skutečnosti, že charakteristická proudová spotřeba je určená hardwarem na nižší úrovni architektury. Vkládání prázdných operací, či přesouvání operací lze provádět i softwarově, a proto jsou tyto dvě metody nejvyužívanější. Obě metody však potřebují generátor náhodných čísel, který musí být již v zařízení implementován. Takto realizovaná ochrana proti útoku proudovým postranním kanálem není velmi úspěšná. Naopak snaha o konstantní proudový průběh je účinnější proti jednoduché proudové analýze (SPA), protože software vybírá instrukce k realizaci algoritmu a tedy může přímo ovlivňovat okamžitou hodnotu proudové spotřeby. Jako protiopatření proti diferenční proudové analýze (DPA) ani tato metoda není dostatečně účinná.

Práce [43] je zaměřená na skrývání tajného klíče FPGA desky pomocí metody náhodného výpočtu, neboli přesouvání operací a instrukcí. Instrukce jsou prováděny náhodně a pořadí je měněno dynamicky. Jsou zde také výsledné grafy, které ukazují, jaký vliv má skrytí informace o tajném klíči. Nechráněný algoritmus AES odkrývá informaci o tajném klíči po sedmi tisících průběhů, kdežto u chráněného algoritmu nebyl dostatečný ani milión použitých průběhů.

4 Využité experimentální pracoviště

Samotnému provedení útoku pomocí proudové analýzy předchází sestavení a zapojení experimentálního pracoviště. Základem pracoviště je kryptografický modul, v našem případě FPGA deska SAKURA-G. Nemaskovaná odlehčená šifra PRESENT je hardwarově implementována na FPGA desku a její odolnost vůči proudové analýze bude následně podrobněji testována i s realizovaným protiopatřením. Samotná šifra je podrobněji rozepsána v podkapitole 4.1 se všemi jejími funkcemi.



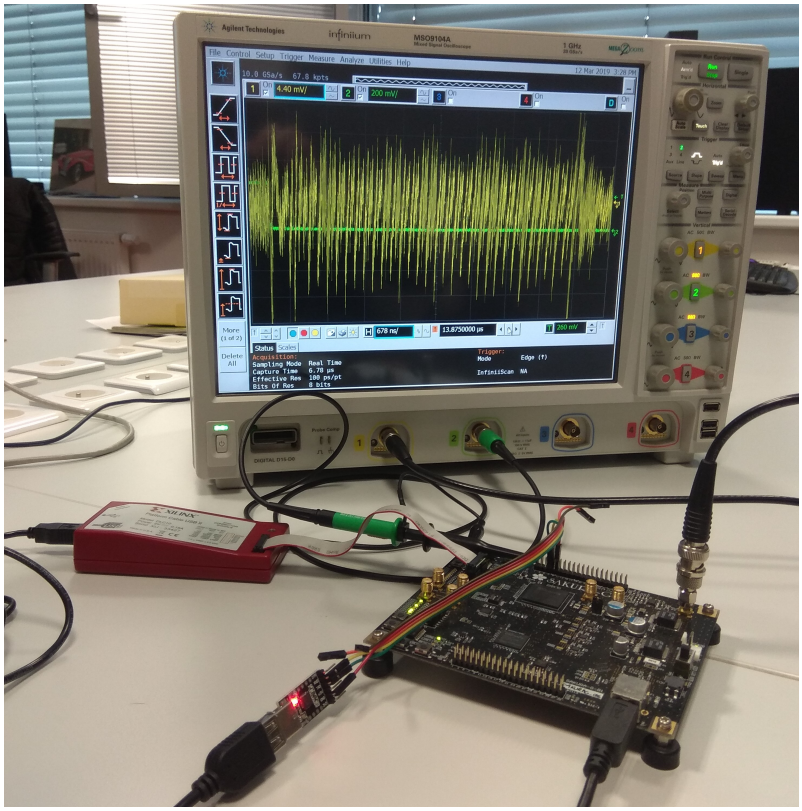
Obr. 4.1: Blokové schéma pracoviště

Experimentální pracoviště sestává ze čtyř bloků. Počítače, FPGA vývojové desky SAKURA-G, osciloskopu a UART převodníku. Počítač obsahuje kód psaný v programovacím jazyce C a generuje vstupní data pro odlehčenou blokovou šifru PRESENT, který je implementován na FPGA vývojové desce SAKURA-G. Bližší specifikace vývojové desky SAKURA-G[44] jsou popsány v tabulce 4.1.

Tab. 4.1: Bližší specifikace vývojové desky SAKURA-G

FPGA pro šifrování	Xilinx Spartan-6 LX75
FPGA kontrolér	Xilinx Spartan-6 LX9
Napětí jádra	1,2V
Zdroj hodinového signálu	48MHz
Propojení FPGA čipů	51 bitů

Komunikace počítače s vývojovou deskou je realizována přes USB typu B se sériovou linku RS-232 na vývojové desce. Z tohoto důvodu je zde převodník UART. SAKURA-G je napájena pomocí USB z externího zdroje. Proudová spotřeba je měřena osciloskopem přes bočník, který je již součástí desky. Osciloskopem je také měřený počet proudových průběhů. Synchronizace je zajištěna přes snímání synchronizačního signálu na uživatelských vstupních pinech pomocí pasivní napěťové sondy. Použitý byl osciloskop Agilent Technologies MSO9104A. Osciloskop dále posílá naměřené hodnoty do počítače pomocí USB rozhraní a přes stejné rozhraní osciloskop přijímá řídící signály. Proudové průběhy jsou ukládány do složek. Každá složka obsahuje 100 proudových průběhů. Pro bližší představu je zde přiložené foto použitého pracoviště 4.2. Výsledné naměřené proudové průběhy spolu se vstupními texty jsou zpracovány na počítači pomocí programu MATLAB.



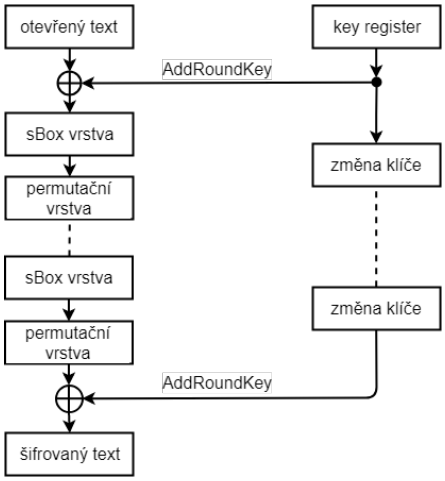
Obr. 4.2: Použité pracoviště pro realizaci DPA

4.1 Odlehčená šifra PRESENT

Jak již bylo zmíněno v podkapitole 3.1, šifra PRESENT je základem pro praktickou část této práce a je nutné ji více představit. Postup při vývoji šifry, vlastnosti, tabulky s-box i permutace, implementaci a porovnání implementace s ostatními

šiframi popisuje práce [45], kde byla šifra poprvé představena. PRESENT je odlehčená šifra vyvinuta v roce 2007, především kvůli potřebám šifrovat data i z velmi omezených zařízení, jako jsou například RFID (Radio Frequency Identification – Identifikace na rádiové frekvenci) a zařízení v síti senzorů. Je tedy vhodná převážně pro zařízení s nižšími nároky na bezpečnost. PRESENT je příkladem substitučně-permutační sítě (SP-síť). Několik let po svém vzniku je šifra častým tématem a roku 2012 byla standardizována organizací ISO v dokumentu označeném jako ISO/IEC 29192-2:2012. Standardizace zvýšila povědomí a vzrůstá množství odborných prací, které ji podrobují testům na odolnost proti útokům, zejména postranními kanály. Na druhé straně však stojí vědci, kteří se snaží o její možná vylepšení. Příkladem je práce [46], která vyšla roku 2017 a klade si za cíl snížit množství odebíraného proudu, nároky na velikost úložiště a výpočetní možnosti zařízení. Práce popisuje a srovnává 7 různých hardwarových architektur z toho jsou 2 nově navrhované.

Následující popis je věnován originálnímu návrhu šifry, kdy její průběh sestává z 31 rund. Délka bloků je 64 bitů. Klíče mohou být buď 80 nebo 128 bitů dlouhé. Pokud pohlédneme zpět na drobná zařízení v síti senzorů, je doporučené využít 80 bitové klíče z důvodu nižších požadavků na úložiště a výpočetní výkon. Každá z 31 rund má implementovanou operaci XOR, pro zavedení rundovního klíče K_i, kde $1 \leq i \leq 32$. K_{32} se využívá pro *post-whitening*[1], a vrstvy lineární bitové permutace, nelineární substituce. Nelineární vrstva využívá jeden čtyřbitový S-box, který je paralelně aplikován 16krát v každé rundě. Názorný popis ukazuje blokové schéma na obrázku 4.3 převzaté z již zmíněné práce [45].




```
generateRoundKeys()
for i = 1 to 31 do
    addRoundKey(STATE, K[i])
    sBoxLayer(STATE)
    pLayer(STATE)
end for
addRoundKey(STATE, K[32])
```

Obr. 4.3: Algoritmus a blokové schéma šifry PRESENT.

Rundovní klíč vyjádřený $K_i = k^i_{63} \dots k^i_0$ pro $1 \leq i \leq 32$ má aktuální stavy $b_{63} \dots b_0$. Operace *AddRoundKey* obsahuje operace v rozmezí $1 \leq j \leq 63$. Vzniká vztah: $b_j \rightarrow b_j \oplus k^i_j$.

[1]Kombinování dat s částmi klíče.

Vrstva S-box je tvořena paralelními čtyřbitovými S-boxy. Tabulku S-boxu reprezentuje tab. 4.2. Aktuální hodnota je značena x, *S[x]* značí hodnotu po substituci. Uvnitř S-box vrstvy se na aktuální stav $b_{63} \dots b_0$ pohlíží jako na šestnáct čtyřbitových slov $w_{15} \dots w_0$, kde $w_i = b_{4 \cdot i+3}||b_{4 \cdot i+2}||b_{4 \cdot i+1}||b_{4 \cdot i}$ pro $0 \leq i \leq 15$. Výsledný kousek je označený jako $S[w_i]$.

Tab. 4.2: Substituční tabulka šifry PRESENT

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S[x]	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Vrstva permutace je realizována pomocí tabulky 4.3. Řádek označený písmenem i značí původní pozici bitu, řádek *P(i)* označuje pozici, na kterou se bit posunul.

Tab. 4.3: Permutační tabulka šifry PRESENT

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P(i)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P(i)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P(i)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P(i)	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

Celý klíč K je uložený v paměti do *key register*. Při použití 80 bitů, je klíč prezentovaný jako $k_{79}, k_{78} \dots k_0$. Během provádění šifrování/dešifrování má klíč stejnou délku, jako bloky dat. Těchto 64 bitů je bráno zleva, tedy $K_i = k_{63}, k_{62} \dots k_0 = k_{79}, k_{78} \dots k_{16}$. Bity klíče K jsou posunuty o 61 bitů doleva. První 4 bity zleva jsou vyjmuty a provede se substituce. Písmeno i značí číslo rundy, označený jako *round_counter*, toto číslo se vloží do funkce XOR s bity $k_{19}, \dots, k_{15}$. Postup:

1. $[k_{79}, k_{78}, \dots, k_1, k_0] = [k_{18}, k_{17}, \dots, k_{20}, k_{19}]$,
2. $[k_{79}, k_{78}, k_{77}, k_{76}] = S[k_{79}, k_{78}, k_{77}, k_{76}]$,
3. $[k_{19}, k_{18}, k_{17}, k_{16}, k_{15}] = [k_{19}, k_{18}, k_{17}, k_{16}, k_{15}] \oplus round\_counter$.

Při použití 128 bitového klíče do operací vstupují vyšší bity a postup je pouze rozšířen o další 4 bity vstupující do operace substituce:

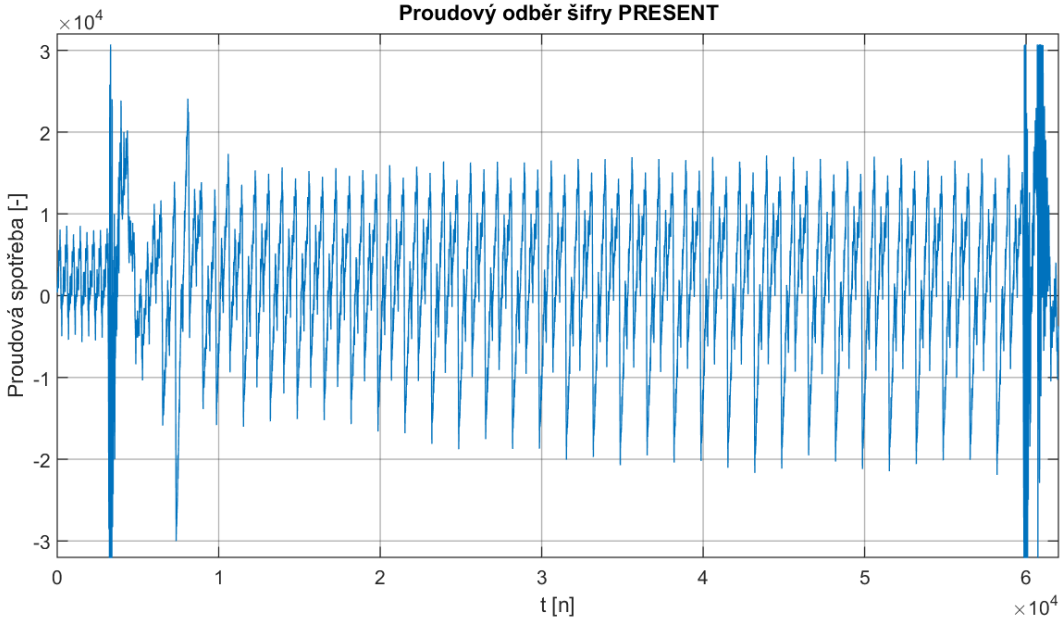
1. $[k_{127}, k_{126}, \dots, k_1, k_0] = [k_{66}, k_{65}, \dots, k_{68}, k_{67}]$,
2. $[k_{127}, k_{126}, k_{125}, k_{124}] = S[k_{127}, k_{126}, k_{125}, k_{124}]$,
3. $[k_{123}, k_{122}, k_{121}, k_{120}] = S[k_{123}, k_{122}, k_{121}, k_{120}]$,
4. $[k_{66}, k_{65}, k_{64}, k_{63}, k_{62}] = [k_{66}, k_{65}, k_{64}, k_{63}, k_{62}] \oplus round\_counter$.

5 Realizace útoku PA

Dosažený přehled o operacích a krocích nemaskované implementace šifry PRESENT, spolu s naměřenými proudovými průběhy z experimentálního pracoviště, jsou klíčové pro samotnou realizaci proudové analýzy. Tato kapitola je věnována postupům při snaze o odhalení tajného klíče. Naměřené proudové průběhy, zobrazené viz obr. 5.1, z experimentálního pracoviště jsou dále přeneseny a uloženy v počítači. Prvním krokem je realizace SPA, tedy jednoduché proudové analýzy představené v sekci 2.1, která má za cíl zjisti co nejvíce informací z omezeného množství naměřených proudových průběhů šifry PRESENT. Poté následuje provedení DPA, která vyžaduje mnohem více proudových průběhů, za účelem odhalení tajného klíče.

5.1 Jednoduchá proudová analýza

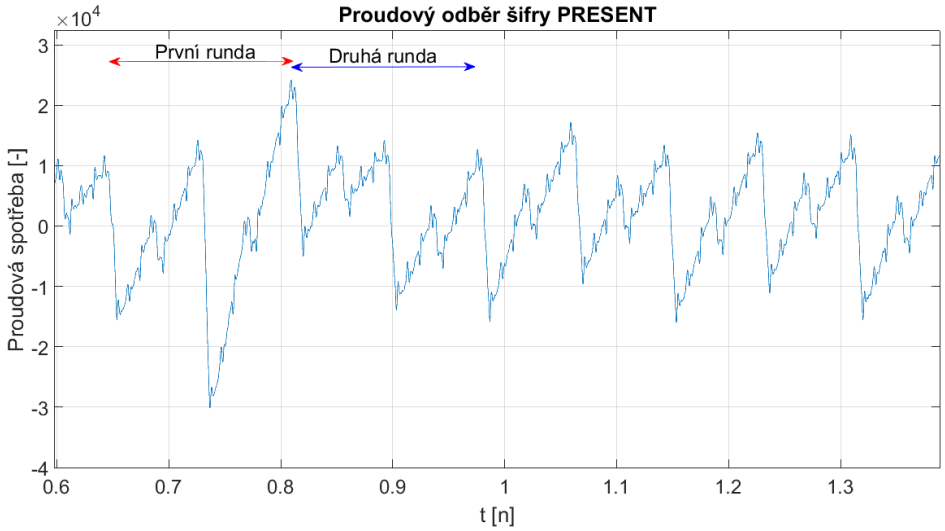
Základem SPA je vizuální inspekce, tedy prozkoumání naměřeného proudového odběru šifry PRESENT. Pro SPA bylo použito 500 proudových průběhů, které byly vybrány z celkové skupiny 199 000 proudových průběhů použitých dále pro DPA. Tyto proudové průběhy byly vloženy do programu MATLAB. Vykreslený graf programem MATLAB vyobrazuje proudový odběr šifry PRESENT 5.1. Na první pohled je viditelný počet rund, které šifra provádí. Konkrétně tedy 32 rund. Po přiblížení



Obr. 5.1: Proudový odběr šifry PRESENT

celkového proudového průběhu a zaměřením na jednotlivé rundy, viz obr. 5.2, je znatelný rozdíl proudového odběru pro první rundu šifrování. Takto zvýšený proudový

odběr je způsobený především zpracováváním vstupních dat a přípravou kryptografického algoritmu na šifrování/dešifrování, které neprobíhají přímo uvnitř první rundy šifrování, ale mají vliv na proudový odběr šifry, jak je na průběhu znatelné. FPGA deska v prvé řadě musí přijmout vstupní data a poté kryptografický algoritmus vypočítá rundovní klíče. I přes tyto přípravné operace je možné tento časový úsek označit za zajímavý pro proudovou analýzu. Při pohlédnutí na výsledky DPA, viz obr. 5.7, je viditelné, že nejvyšší hodnota korelace se nachází v tomto časového úseku. Z první rundy pohlédneme na rundu poslední. Navzdory faktu, že poslední

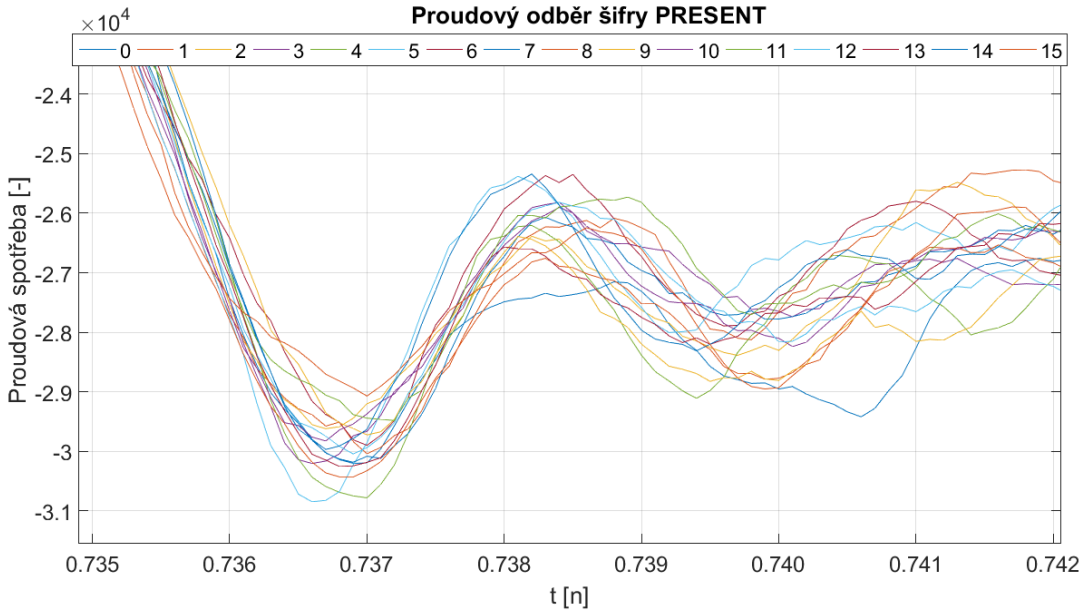


Obr. 5.2: Přiblížený proudový odběr šifry PRESENT zaměřený na první a druhou rundu.

runda šifry PRESENT je zkrácená o operace S-box a Permutace, výsledný proudový odběr je, na vykresleném grafu 5.1, nezměněný.

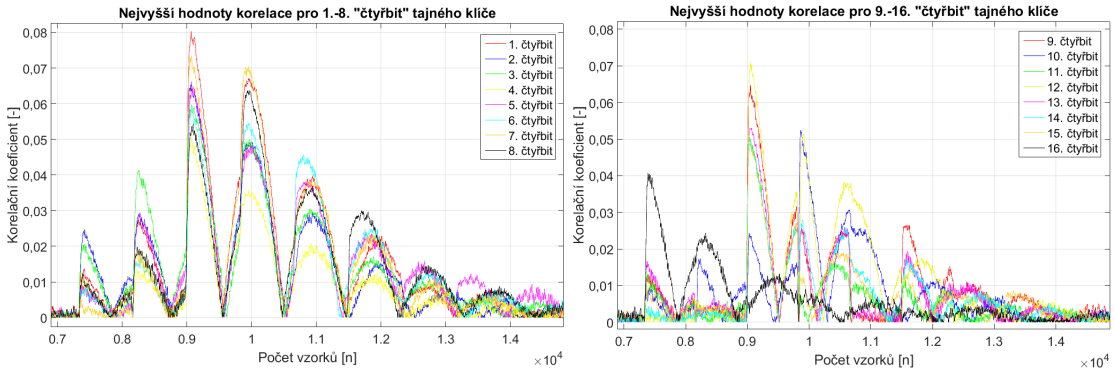
Po prozkoumání jednoho proudového průběhu je možné přes sebe vykreslit několik dalších a pokusit se vypozorovat Hammingovu váhu tajného klíče [20]. Každý náhodný vstupní text použitý pro šifrování obsahuje hodnoty v rozmezí 0-15. Pokud z každého otevřeného textu vybereme řádky, které obsahují například hodnotu 0, spojíme je s naměřenými proudovými průběhy a nalezené průběhy zprůměrujeme, získáme proudový průběh, který je z větší části očištěný od šumu. Pro toto zprůměrování bylo použito 100 proudových průběhů. Získaný průběh představuje proudovou spotřebu FPGA desky při práci, například s hodnotou 0. Tímto způsobem bylo vytvořeno 16 proudových průběhů, tedy pro všechny možné hodnoty. Jednotlivé grafy byly následně vykresleny přes sebe, což představuje obrázek 5.3. Graf je zaměřený na výše zmíněný časový úsek, který dle výsledků DPA lze považovat za významný. Cílem tedy je odhalit Hammingovu váhu tajného klíče. Avšak v tomto případě není možné určit, zda z dosaženého grafu lze Hammingovu váhu tajného klíče určit. Pravděpodobně se jedná o špatnou synchronizaci, která může

být způsobená chybou v měření. Pokud by se jednalo o Hammingovu váhu tajného klíče, průběhy by nebyly takto posunuté, ale zarovnané na sebe a rozšiřovaly by se až v místě, kde probíhá přičtení tajného klíče. Nebylo tedy možné odhalit žádné užitečné informace o Hammingově váze tajného klíče.



Obr. 5.3: Přiblížené proudové průběhy šifry PRESENT zaměřeny na odhalení Hammingovi váhy tajného klíče.

Pokud provedeme diferenční proudovou analýzu například cílenou na operaci *AddRoundKey*, můžeme z dosažených výsledků získat přehled „užitečných“ bodů. Tento bod získáme uložením pozice v čase, kde dosahuje korelace nejvyšší hodnoty pro jednotlivé čtyř bity tajného klíče, viz obr. 5.4. Například 1.-8. čtyř bit tajného



Obr. 5.4: Přiblížené výsledné grafy z DPA cílené na operaci AddRoundKey.

klíče dosahuje nejvyšších hodnot v čase asi 0.906. Kdežto pro 16. čtyř bit je nejvyšší hodnota v časovém úseku 0.738. Tento časový úsek byl již výše zmiňován. Takto získané jednotlivé body lze využít pro další provedení diferenční proudové analýzy

a zkrátit tak výpočetní a časové nároky. Není tedy třeba zaměřit DPA na celý proudový průběh, ale jen na „užitečný“ časový úsek.

Výsledkem jednoduché proudové analýzy byly zajímavé body, které budou použiti k realizaci DPA. Tímto způsobem bude výpočet DPA cílen jen na zajímavé body nesoucí informaci o tajném klíči a bude ušetřen výpočetní čas při realizaci útoku. Jednoduchá analýza vizuální inspekcí nepotvrdila přímou závislost proudové spotřeby na tajném klíči.

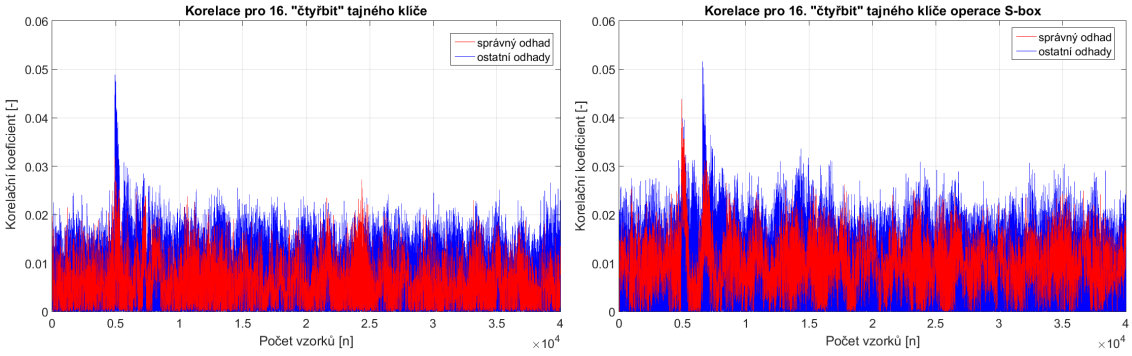
5.2 Diferenční proudová analýza

Pro realizaci DPA byl využit program MATLAB, ve kterém jsou potřebné matematické operace již předprogramovány a nabízí široké možnosti. První využitý skript byl převzatý od autorů knihy *Power Analysis Attacks – Revealing the Secrets of Smartcards* a volně dostupný ke stažení na jejich stránce *www.dpabook.org*. Skript byl určený jako příklad provedení DPA na jeden byte kryptografického algoritmu AES. Využíval pro svoji práci instalovaný balíček metod, které bylo velmi obtížné zásadně upravovat, bez jakékoliv dokumentace. Drobné úpravy byly možné a tak bylo dosaženo změny cíle, tedy z algoritmu AES na šifru PRESENT. Pracoval s jediným souborem, který obsahoval všechny naměřené průběhy a druhý soubor s použitými vstupními texty. Naměřených proudových průběhů bylo 15 100. Ukládány byly po jednotlivých složkách obsahujících vstupní text a k nim shodné proudové průběhy. Složky obsahovaly 100 vstupních textů, tedy výsledný počet byl 151 složek. Skládání všech vstupních textů do jednoho souboru a naměřených proudových průběhů do souboru druhého bylo výpočetně náročné, následná práce s takto obsáhlými soubory byla jak časově, tak i výpočetně velmi náročná. Z tohoto důvodu byl skript nevhodný pro další využití.

Po odborné konzultaci s vedoucím bakalářské práce, mi byl navrhnut odlišný skript, který pracoval s jednotlivými složkami a byl několikanásobně efektivnější. Využívá volně dostupný *framework* OpenSCA s otevřeným zdrojovým kódem, který je vytvořený pro realizaci útoku postranními kanály uvnitř prostředí MATLAB. Skript pracuje ve dvou smyčkách a postupuje dle popisu DPA v teoretické části 2.2. Nadřazená první smyčka je pro velikost klíče, která je pro šifru PRESENT 8 bytů nebo 16 „čtyř bitů“. Využita byla velikost čtyř bitů, jelikož je následná práce s nimi snazší z důvodu vstupu do nelineární funkce S-box, který je čtyřbitový. Vnořená druhá smyčka je pro průchod všemi složkami s naměřenými proudovými průběhy. Uvnitř této smyčky se načítají vstupní data spolu s příslušnými průběhy. Vytvoří se odhady pro jednotlivé hodnoty tajného klíče, které jsou 0-15. Následně je simulován průběh operace *AddRoundKey* a substituce dle substituční tabulky šifry PRESENT. Pokud se útok má zaměřit pouze na lineární operaci *AddRoundKey*, krok substituce

se nesimuluje. Výsledné odhadované hodnoty jsou přiřazeny k naměřeným hodnotám proudové spotřeby za pomoci některého z modelů. Základními modely využitými v této práci jsou modely Hammingovi váhy (v příloze A.1) a Zero value. Konečné odhadované mezivýsledky jsou porovnány se skutečnými naměřenými hodnotami. Porovnání je prováděné pomocí korelačního koeficientu, jehož hodnota pro jednotlivé odhady je vykreslena do grafu. Takto vytvořené grafy budou použity pro znázornění získaných výsledků. Jednotlivé odhadované hodnoty tajného klíče jsou ukládány do tabulky a řazeny dle korelačního koeficientu.

Nyní s naměřenými proudovými průběhy a otestovaným skriptem lze přikročit k samostatné analýze. Jako první byl využit model Hammingovy váhy. Tento model je často využívaný v oblasti útoků postranními kanály. Představuje počet nenulových bitů pro jednotlivé hodnoty. Tento model byl popsaný a použitý v práci [47], kde jej využili k určení tajného klíče algoritmu AES pomocí DPA. Model nabývá hodnot 0-255. Šifra PRESENT má však jen 0-15 hodnot. Pro správnou aplikaci modelu to však nepředstavuje problém a aplikuje se pouze část shodná se šifrou PRESENT. Útok na šifru PRESENT je schopný odhalit pouze 8 bytů, tedy 16 čtyř bitů. Důvodem je změna klíče po prvním rundě šifrování. Výsledky s použitým modelem Hammingovy váhy byly však nedostatečné. Nebylo možné správně odhalit hodnoty všech šestnácti čtyř bitů tajného klíče při cílení útoku na operaci *AddRoundKey*. Odhaleno bylo pouze sedm ze šestnácti hodnot. Výsledný graf znázorňuje obrázek 5.5 jako příklad špatně určené odhadované hodnoty tajného klíče. Následujícím krokem pro přesnější výsledky byla změna cíle útoku.

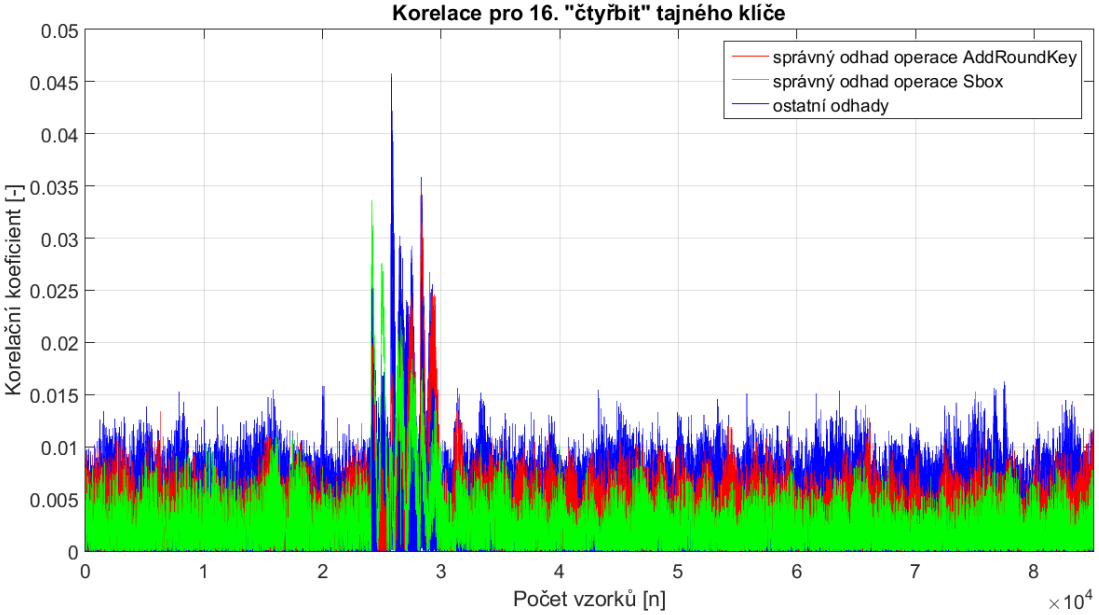


Obr. 5.5: Korelace při operaci addRoundKey (vlevo) a S-box (vpravo) na 16. čtyř bit tajného klíče s použitím modelu Hammingovi váhy.

Operace *AddRoundKey* je lineárního charakteru kvůli operaci XOR a proto může vytvářet *false-positives*, tedy špatné výsledky považované za správné. Naopak operace *S-box* je nelineárního charakteru a nedochází ke vzniku *false-positives*. Tato skutečnost je popisována v práci [48]. Při cílení útoku na operaci *S-box* však nedošlo k předpokládanému zlepšení. Výsledek při cílení útoku na *S-box* představuje obr. 5.5.

Při bližším prozkoumání výsledných grafů pro operace *AddRoundKey* a *S-box* byly spatřeny velké odlišnosti, které by nastat neměly. Drobný posun korelace v čase mohl nastat z důvodu posunu operace *S-box*, která je vykonávána vždy po operaci *AddRoundKey*. Ovšem zde je operace *S-box* před operací *AddRoundKey*. Také tvar a průběh grafů by měl být velmi podobný, zde se liší hodnoty korelace.

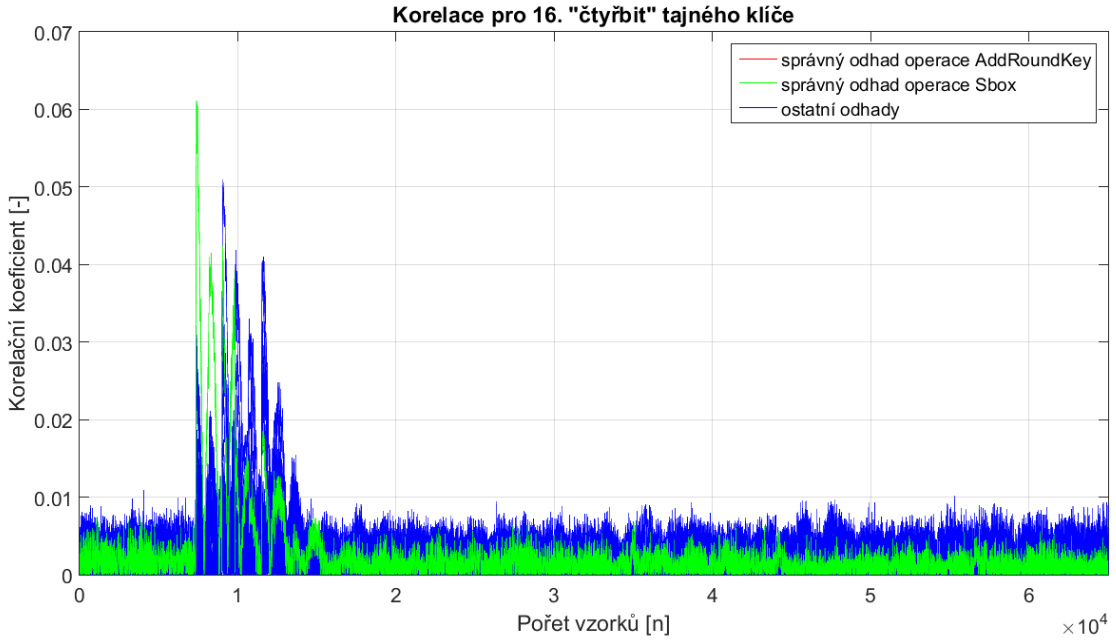
Během konzultace s vedoucím práce byly grafy diskutovány a byla navrhnuta změna použitého modelu. Druhou navrhovanou možností bylo zvýšit počet naměřených proudových průběhů. DPA je velmi závislá na počtu naměřených proudových průběhů, a tedy čím je vybraný model přesnější pro naměřený proudový průběh zařízení, tím větší je hodnota korelace mezi skutečnou a odhadovanou proudovou spotřebou. Jak bylo uvedeno v kapitole 2.2, velké množství naměřených průběhů kladně ovlivňuje správné odhadnutí hodnot tajného klíče. S touto myšlenkou byly naměřeny další proudové průběhy, přesněji 57 200 proudových průběhů. Zvýšení počtu proudových průběhů nezpůsobilo kladnou změnu odhadnutých výsledků, spíše naopak. Graf porovnávající výsledky pro 16. čtyř bit tajného klíče operací *AddRoundKey* a *S-box* obr. 5.6. Při prvním pohledu jsou zřejmé výše popsané rozdíly



Obr. 5.6: Korelace při použití 57 200 proudových průběhů, operace addRoundKey a Sbox na 16. čtyř bit tajného klíče s použitím modelu Hammingovi váhy.

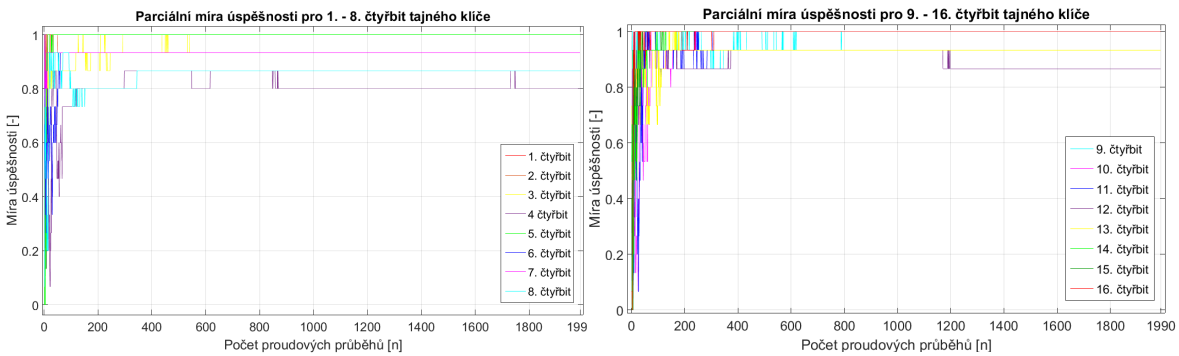
a operace *S-box* předcházející operaci *AddRoundKey*. Počet proudových průběhů zůstal stejný, namísto modelu Hammingovy váhy byl použit model Zero Value. Tento model je realizovaný tak, že první hodnota je hluboko pod hodnotou 0, například -200 a další hodnoty jsou konstantní, například na hodnotě 1. Analýza byla provedena znovu, nyní však s lepšími výsledky a již neprokazovala odlišností mezi výslednými grafy operací *AddRoundKey* a *S-box*. Teoreticky bylo vše v pořádku, a tak

byl ještě jednou zvýšen počet naměřených proudových průběhů za účelem odhalení celého tajného klíče. Počet proudových průběhů byl zvýšen mnohem výrazněji, a to na hodnotu 199 000. Výsledný graf, viz obr. 5.7, zobrazuje operace *AddRoundKey* a *S-box* ve stejném časovém úseku, což je v pořádku. Hodnota korelace při cílení útoku na operaci *AddRoundKey* je totožná s korelací pro operaci *S-box* a jednotlivé grafy se překrývají. Proudová analýza realizovaná s tímto počtem proudových průběhů spolu s modelem Zero Value je schopná odhalit více hodnot tajného klíče, než při použití modelu Hammingovi váhy. Výsledkem je zatím nejvyšší hodnota korelace pro odhady jednotlivých čtyř bitů. Příkladem je zmiňovaný 16. čtyř bit, viz obr. 5.7. V tomto případě zelený grafe operace *S-box* překrývá červený graf operace *AddRoundKey*.



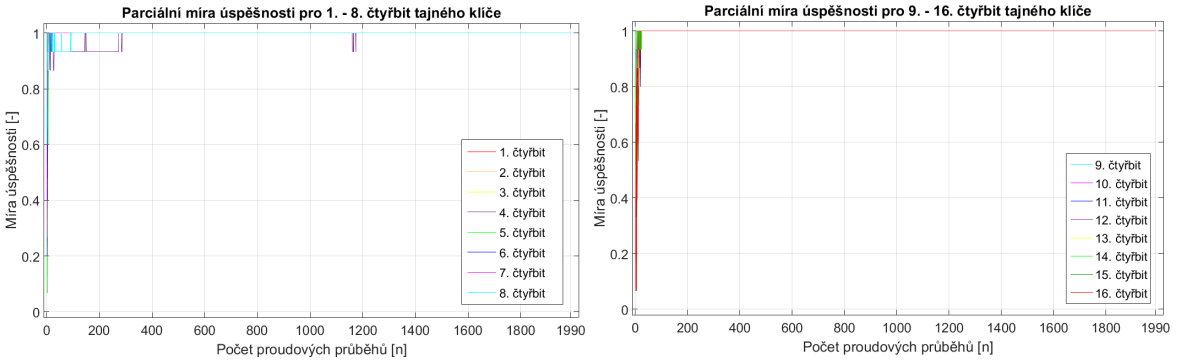
Obr. 5.7: Korelace při použití 199 000 proudových průběhů, operace addRoundKey a Sbox na 16. čtyř bit tajného klíče s použitím modelu Zero Value.

Pro bližší porovnáni jsou zde dva grafy viz obr. 5.8, které porovnávají míru úspěšnosti při odhadování jednotlivých čtyř bitů tajného klíče. Počet potřebných proudových průběhů nutných pro správné určení hodnoty tajného klíče je znázorněn na ose x. Hodnoty jsou uvedeny jako počty použitých složek, každá složka obsahovala 100 proudových průběhů. Výsledný počet proudových průběhů lze tedy získat jako: počet složek·100. Osa y představuje úspěšnost při odhalování správné hodnoty. Některé odhady nedosahují hodnoty 1, což značí, že je nebylo možné správně odhadnout. Po těchto změnách stále nebylo možné tajný klíč odhalit celý, a proto mi byl navrhnut postup s využitím nového modelu. Tento model byl derivovaný



Obr. 5.8: Míra úspěšnosti odhadu tajného klíče pro 1.-8. čtyř bit a 9.-16. čtyř bit.

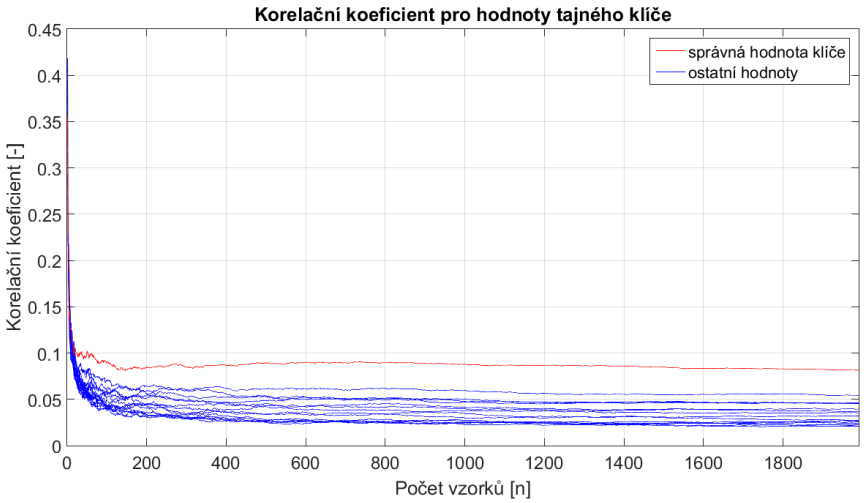
z proudových průběhů, přímo pro šifru PRESENT. S takto odvozeným modelem byly dosažené výsledky nejpřesnější a již bylo odhaleno všech 8 bytů, tedy 16 čtyř bitů, tajného klíče. Při jeho použití bylo potřeba pouze 30 000 proudových průběhů, dle grafu 5.9. Odhad pro 4. čtyř bit značně kolísá v oblasti 120 000 průběhů, což mohlo způsobit zvýšené množství šumu či drobná chyba v průběu měření. Zmíněný graf také dokazuje tvrzení, že byly odhaleny všech 16 čtyř bitů tajného klíče. Pro



Obr. 5.9: Míra úspěšnosti odhadu tajného klíče pro 1.-8. čtyř bit a 9.-16. čtyř bit.

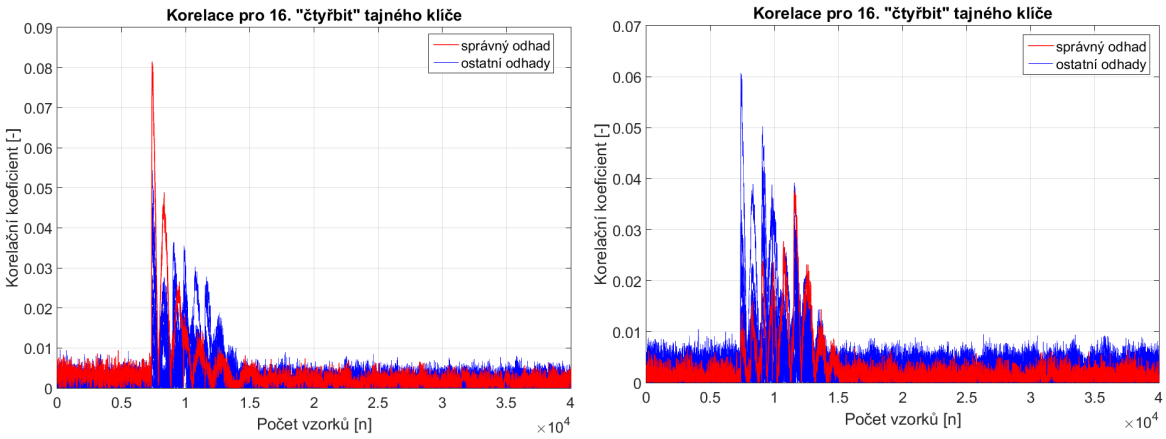
detailnější znázornění míry korelace pro jednotlivé odhady tajného klíče byl vytvořen graf 5.10. Tento graf je zaměřený na šestnáctý čtyř bit tajného klíče. Graf dokazuje nejvyšší hodnoty korelace pro správný odhad tajného klíče. Tyto hodnoty korelace jednotlivých odhadů tajného klíče jsou ukládány do tabulky pro jednotlivé použité průběhy. Nesprávné odhady tajného klíče nedosahují tak vysokých hodnot korelace a všechny jsou v úzké blízkosti (modré). Korelace správného odhadu je mnohem vyšší, než ostatní hodnoty a velmi přehledně vyčnívá nad ostatními (červený). Graf 5.10 představuje výše zmíněný děj. Nesprávné odhady klesají, zatímco správný odhad si udržuje svoji pozici či mírně narůstá.

Realizace DPA byla tedy úspěšná, výsledné hodnoty korelace byly nejvyšší ze všech předchozích měření. Nynější derivovaný model vykazuje kýžené výsledky při cílení útoku na operaci *AddRoundKey* a plní všechny předpoklady. Navzdory tomu



Obr. 5.10: Korelace pro jednotlivé hodnoty tajného klíče.

generuje odlišné výsledky pro operaci*S-box* viz obr. 5.11. I přes zmíněné nesrovnalosti mezi výsledky jednotlivých operací bylo možné odhalit celý tajný klíč, při cílení útoku na operaci *AddRoundKey*. Neodhalení tajného klíče při cílení útoku na operaci *S-box* bylo pravděpodobně způsobeno nedostatečným množstvím naměřených proudových průběhů pro použitý model. Pokud 199 000 proudových průběhů bylo více než dostatečné množství při cílení útoku na operaci *AddRoundKey*, u *S-box* tomu tak být nemusí. Nicméně tajný klíč byl odhalen a cíl realizované diferenční proudové analýzy je splněný.



Obr. 5.11: Korelace při operaci addRoundKey (vlevo) a S-box (vpravo) na 16. čtyř bit tajného klíče s použitím derivovaného modelu PRESENT.

5.3 Poslední dva byty klíče

Proudovou analýzou je možné odhalit jen osm z deseti bytů. Důvodem této skutečnosti je velikost přičítaného rundovního klíče, který je 64 bitový, tedy 8 bytů. Zbylé 2 byty jsou použity pro vytvoření následujících rundovních klíčů, viz kapitola 4.1, a není možné získat tyto byty nezměněné. Nicméně není nemožné tyto byty odhalit jiným způsobem. Možností, zřejmě nejsnazší, je využít metody *Brute Force*. Každý byte může nabývat hodnot 0-16 a kombinace dvou bytů přináší 256^2 možností. Čísly 65 536 možností, což je velmi snadno realizovatelné, jak bylo navrhnuto a dokázáno v práci [48]. Naproti tomu potřebujeme algoritmus, který bude porovnávat výstupní šifrovaný text pro jednotlivé odhady, s šifrovaným textem vytvořeným kryptografickým modulem. Jakmile vznikne shoda, celý tajný klíč je nyní k dispozici a bez dalších problémů můžeme dešifrovat jakékoliv zprávy z kryptografického modulu.

6 Implementované protiopatření

Po úspěšně realizované diferenční proudové analýze, pomocí které bylo možné odhalit tajný klíč kryptografického modulu, je nutné zabránit jejímu úspěšnému provedení. Protiopatření je realizované na základě Dynamické rekonfigurace FPGA desky. Předlohou pro realizované protiopatření je již zmiňovaný článek [33], ve kterém jsou hlavním typem protiopatření tyto metody:

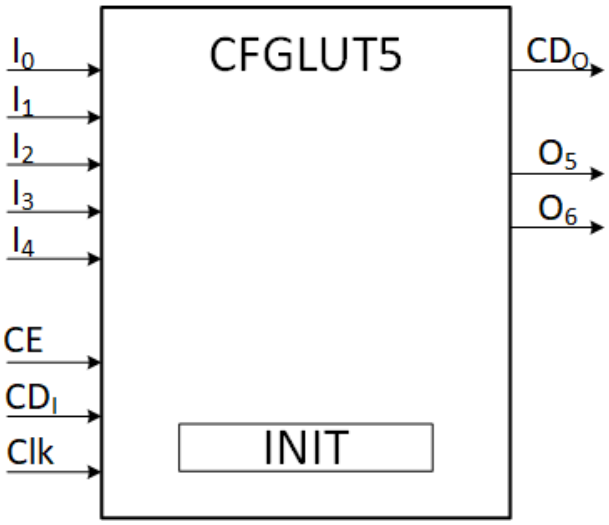
- dekompozice operace S-box (S-box decomposition),
- booleovské maskování (boolean masking),
- Předvyplnění registru (register precharge).

Samotná implementace je z diplomové práce[49]. Dynamické rekonfiguraci je věnována sekce 6.1, sekce 6.2 pak zmíněným metodám. Metody protiopatření budou aplikovány všechny tři, jelikož článek [33] předvedl výsledky pro jednotlivé metody, které však samostatně byly nedostačující. Pouze kombinací metod byla dosažena požadovaná úroveň protiopatření. Stěžejní komponentou pro protiopatření je generátor náhodných čísel, který určuje, jakým způsobem je provedena rekonfigurace a použitý blok dat pro metodu *register precharge*.

6.1 Dynamická rekonfigurace

FPGA deska poskytuje funkci dynamické rekonfigurace, což nám dovoluje desku úplně, nebo jen částečně, přeprogramovat[50]. Takové přeprogramování je prováděno během vykonávání některé činnosti, v tomto případě během šifrování, a to bez jakéhokoliv zásahu z venčí. Rekonfigurace je vypočítána uvnitř zařízení. Dynamické rekonfigurace může být dosaženo pomocí blokové RAM, distribuované RAM, nebo využitím komponenty CFGLUT5. Pro práci byla využita komponenta CFGLUT5, viz obr. 6.1, přístupná u Xilinx FPGA čipů, konkrétně u rodiny Spartan-6, kde byla poprvé představena.

Tato komponenta představuje libovolnou logickou funkci obsahující pět vstupů, I_0 - I_4, a jeden výstup O_5. O_6 je druhým výstupem, ale na rozdíl od O_5 nebere v úvahu hodnotu vstupu I_4, ale považuje ji vždy za nulovou. Porty CD_I, CD_O, CE slouží pro řízení rekonfigurace. Port CD_I, je jednobitovým vstupem pro rekonfiguraci, CD_O, je jednobitovým výstupem a CE port, pokud obsahuje hodnotu 1, povoluje přepis hodnoty INIT. INIT je počátečním obsahem CFGLUT5 komponenty. Pro zjednodušení lze na CFGLUT5 pohlížet jako na posuvný registr. CLK reprezentuje hodinový signál a řídí posun hodnot uvnitř registru.



Obr. 6.1: Schématická značka CFGLUT5

6.2 Metody protiopatření

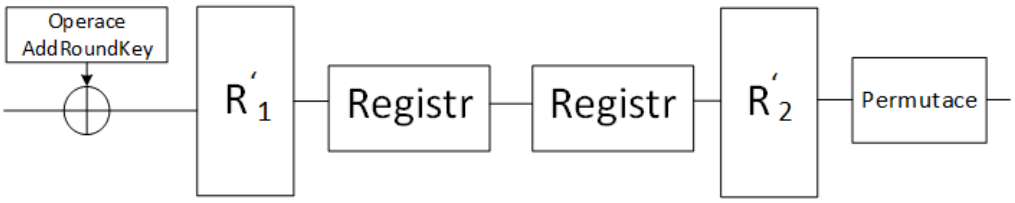
Metoda s dekompozicí S-box je způsob protiopatření, kdy je nelineární operace S-box rozdělena do dvou odlišných bijekcí. Vytvořené bijekce, nazvané R_1 a R_2, dohromady představují originální S-box, což je důvodem pro jejich jedinou podmínku, kterou je rovnice $R_2(R_1(x)) = Sbox(x)$. Možností, jak vytvořit tyto bijekce, je opravdu velké množství, avšak pouze jedna kombinace splňuje podmínku rekonstrukce původního S-boxu. Díky dynamické rekonfiguraci, která vytvoří vždy jinou bijekci použitou při šifrování, je pro útočníka takřka nemožné odhadnout správnou kombinaci. Mezi bijekcemi je umístěný registr, který ukládá pouze výstupní hodnotu první bijekce, a tedy vstupní hodnotu do bijekce druhé. Jednotlivé bijekce mohou být vytvářeny vždy úplně celé, nebo jen náhodně obměněné z bijekce předchozí. Podle práce [33], není nutné vytvářet vždy novou, a tak je použitá bijekce jen měněna. Po resetování kryptografického modulu je první bijekce vytvářena tak, že první z bijekcí je náhodná a dle ní je dopočítávána bijekce druhá tak, aby platil požadovaný vztah mezi nimi. Před každým šifrováním je první bijekce upravena záměnou osmi párů řádků a stejným způsobem jsou upraveny řádky v druhé bijekci. Stále musí platit vztah, že kombinací obou bijekcí vzniká originální S-box. Jednou vytvořená bijekce není sdílená všemi rundami šifry PRESENT, ale je uvnitř každé rundy vypočítána bijekce nová, což má za cíl zvýšit dosaženou ochranu.

Druhým použitým protiopatřením je booleovské maskování. Tato metoda přičítá náhodná data (maska), k výsledkům z operací prováděných šifrou PRESENT. Přičítání je prováděno operací XOR. Útočník nezná přičítanou hodnotu a tak není schopný předpovědět výsledky operací. Pokud by maska zůstala přičtena k výsledku,

mělo by to také vliv na šifrování a výsledný šifrovaný text by nebylo možné dešifrovat. V průběhu šifrování musí být druhá operace XOR, která přidanou masku „odečte“ a zajistí, že je výsledek správný. Použitá maska je náhodný binární vektor o velikosti 64 bitů, je to velikost se kterou pracuje šifra PRESENT. Maska (m_1) je přičtena za druhou bijekcí (R_2) operace S-box. Její odečtení je před první bijekcí (R_1) operace S-box uvnitř následující rundy. Mezi přičtením a odečtením masky (m_1) se nachází operace Permutace, proto je nutné odečítat permutovanou masku $P(m_1)$. Ovšem uvnitř první rundy se nachází nemaskovaná hodnota a tedy odečtení masky by vneslo chybu do zpracovávaných dat, proto je nutné před první odečtení masky vložit operaci XOR s permutovanou maskou $P(m_1)$. Tímto způsobem je vliv odečítání masky (m_1) odstraněný. Do rund je také vložena druhá maska (m_2). Jejím vložením za bijekci (R_1) a následným odečtením před bijekcí (R_2) bude vytvořeno maskování vstupních a výstupních hodnot z registru, který je umístěný mezi dekompozicemi operace S-box. Operace s maskami m_1 a m_2 jsou umístěné vždy před nebo za bijekce R_1 a R_2, a proto je možné průběh jednotlivých rund zjednodušit takovým způsobem, že operace s maskami budou součástí vytvářených bijekcí. Výsledné bijekce budou tedy realizovány rovnicemi: $R'_1(x) = R_1(x \oplus P(m_1)) \oplus m_2$ a $R'_2(x) = R_2(x \oplus m_2) \oplus m_1$. Průběh poslední rundy je také nutné upravit, protože poslední runda šifry PRESENT neobsahuje operace S-box a permutace. Tato runda obsahuje pouze přičtení poslední verze rundovního klíče, proto je nutné masku m_1 odečíst ještě před přičtením klíče.

Posledním implementovaným tipem protiopatření je předvyplnění registru. Tato metoda ovlivňuje hodinový cyklus šifrování a útočník není schopný odhadnout, v kterém čase byla šifrovaná data skutečná, tedy ta, která útočník do zařízení poslal. Mezi jednotlivé průběhy šifrování je z registru vložen náhodný blok dat, který je šifrován. Vytváří se tak šifrování prázdných operací, jak bylo představeno v sekci 3.1.1 uvnitř kapitoly Skrývání. Tímto šifrováním prázdných operací nejsou zvýšeny nároky na FPGA desku, ale zvýší se počet hodinových taktů dvojnásobně. Také je nutné ukládat hodnotu, která je nahrazena náhodnou hodnotou z registru, proto je zde nutný ještě jeden registr.

Kombinace jednotlivých protiopatření jsou znázorněna na obrázku 6.2. Pozice jednotlivých masek nejsou nakresleny, ale jsou součástí R'_1 a R'_2.



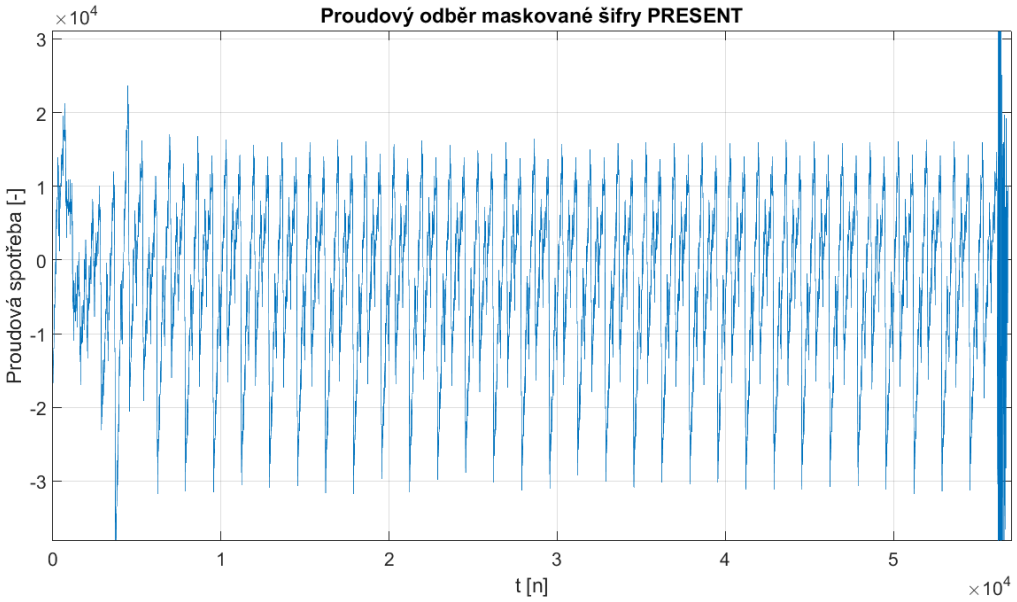
Obr. 6.2: Umístění jednotlivých ochran uvnitř jedné rundy šifry PRESENT

6.3 Výsledky protiopatření

Po představeném protiopatření, které je implementované na FPGA vývojové desce SAKURA-G, bylo nutné naměřit proudové průběhy a dokázat, či vyvrátit, účinnost protiopatření. Proudové průběhy byly naměřeny stejným způsobem a na stejném experimentálním pracovišti, jak již bylo popsáno v kapitole 4. První byla provedena jednoduchá proudová analýza, s návazností na ni pak i diferenční proudová analýza.

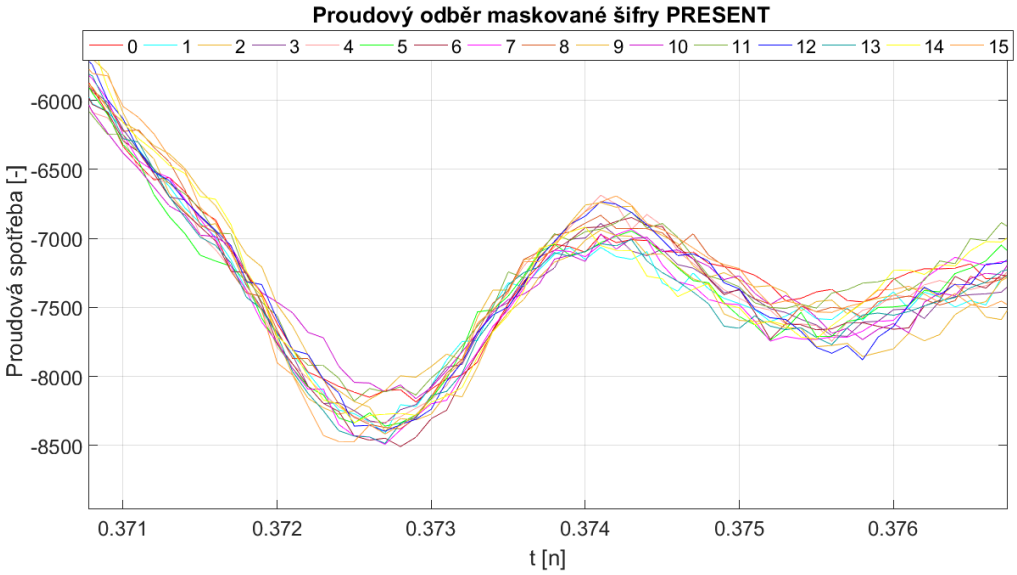
Jednoduchá proudová analýza

Naměřené proudové průběhy jsou vloženy do grafu pro vizuální inspekci 6.3. Vykreslený graf programem MATLAB vyobrazuje proudový odběr maskované implementace šifry PRESENT. Na první pohled je viditelný počet rund, které šifra provádí, což je stejné jako u nemaskované implementace. Rozdílem jsou výsledné hodnoty, které jsou v tom případě menší, což může být způsobeno mírně odlišným nastavením osciloskopu od předchozího provedeného měření. Viditelné je i zvýšení naměřených hodnot první rundy během šifrování, a je tedy možné předpokládat, že tato oblast nese jakkoliv užitečnou informaci. Závěrem připadajícím v úvahu by tedy mohlo být, že je možné realizací DPA odhalit bližší informace o tajném klíči, či snad jeho přesné hodnoty. Druhou možností je závěr, že protiopatření není viditelné pouhým okem a útočník, který by se zaměřil na tuto implementaci ji není schopný zpozorovat. Který ze závěrů je pravdivý dokáží výsledky realizované diferenční proudové analýzy. Pokud bude postup stejný, jako u nemaskované implementace šifry PRESENT,



Obr. 6.3: Proudový odběr šifry PRESENT

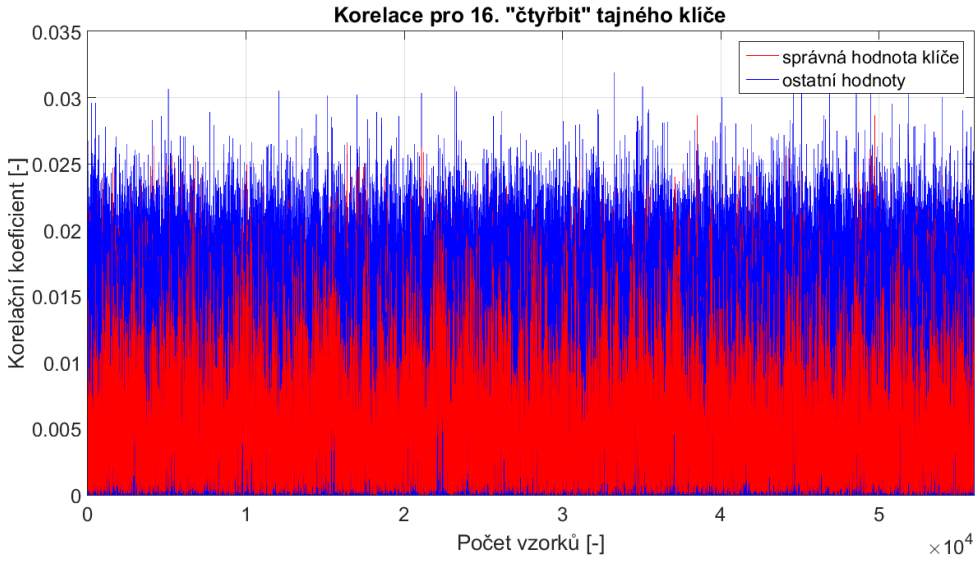
vezmeme zprůměrované proudové průběhy a přiložíme je na sebe. Výsledný graf 6.4 je podobný grafu pro nemaskovanou implementaci. Ovšem posunutí jednotlivých proudových průběhů není tak velké, i přes to není možné jednoznačně říci, že by se mohlo jednat o Hammingovu váhu tajného klíče. S největší pravděpodobností se opět jedná o problémovou synchronizaci. Tudíž ani v tomto případě se nepodařilo zjistit nové informace pomocí jednoduché proudové analýzy.



Obr. 6.4: Proudový odběr šifry PRESENT

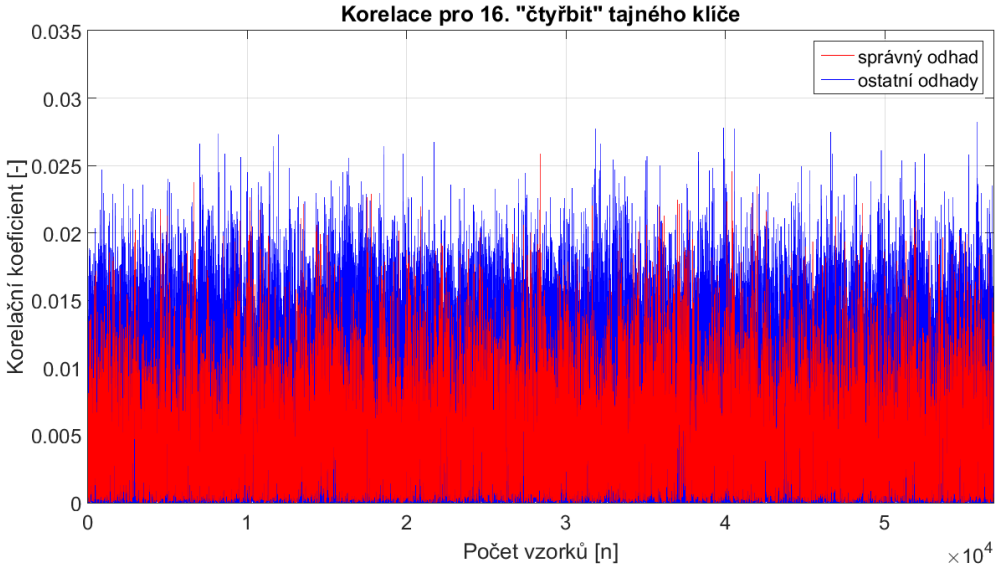
Diferenční proudová analýza

Po vizuální inspekci, a snaze odhalit užitečné informace z malého množství průběhů, lze navázat realizací Diferenční proudové analýzy. Naměřeno bylo celkem 28 000 proudových průběhů, což bylo dostatečné množství pro odhalení tajného klíče za pomocí nově derivovaného modelu na nemaskovanou implementaci šifry PRESENT. Ovšem v tomto případě není možné zjistit žádné užitečné informace. Skript pro realizaci DPA zůstal nezměněný. Použitý model vykazovat velmi dobré výsledky na nemaskovanou implementaci PRESENT. Cílem byla pouze lineární operace *AddRoundKey*, z důvodu neúspěšnosti útoku při cílení na operaci *S-box*. I přes to však výsledný graf 6.5 nenese žádnou užitečnou informaci. Žádný z odhadů tajného klíče nedosahuje vysoké míry korelace a graf působí dojmem, že se jedná pouze o velké množství šumu. Správný odhad tajného klíče, znázorněný červeně, je skrytý mezi korelací pro ostatní odhady. Nyní nebylo možné správně určit hodnotu tajného klíče. Ve skutečnosti zde nebylo možné označit žádný odhad tajného klíče za „správný“, protože není jediný, který by dosahoval mnohem vyšší hodnoty korelace, v porovnání s ostatními.



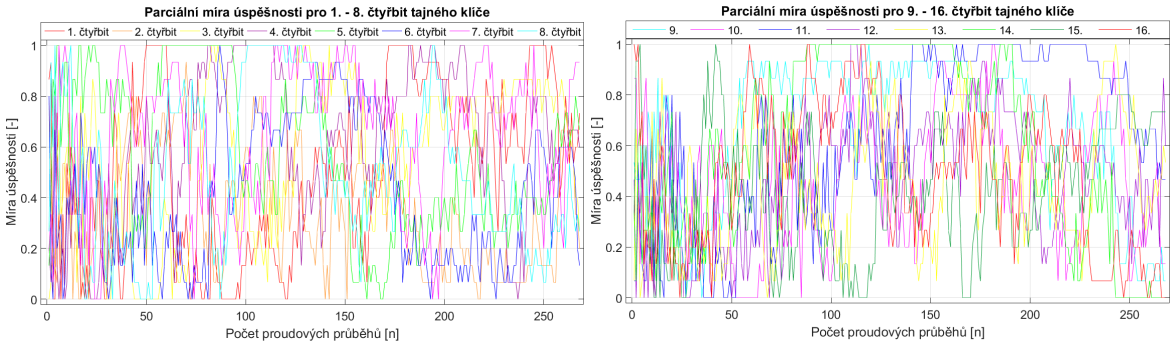
Obr. 6.5: Korelace při operaci addRoundKey na 16. čtyř bit tajného klíče s použitím derivovaného modelu PRESENT.

Jak již první realizace DPA prokázala, velmi záleží na použitém modelu. Z toho důvodu byl použit model Zero Value, který v minulém případě dokázal správně odhalit větší část tajného klíče. Pokud by realizovaná DPA, s tímto modelem, prokázala odlišné výsledky, byla by pozornost zaměřena na změnu použitého modelu. Výsledný graf 6.6 ovšem nebyl schopný cokoliv prokázat. V tomto případě ani změna modelu neposkytla kýžené výsledky.



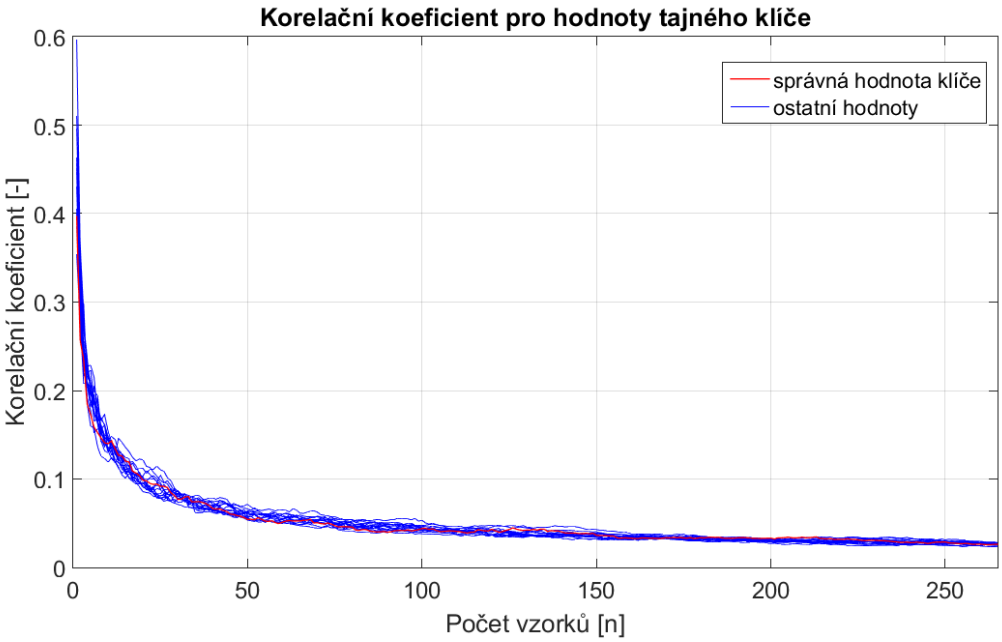
Obr. 6.6: Korelace při operaci addRoundKey na 16. čtyř bit tajného klíče, model Zero Value.

Realizovaná DPA na maskovanou implementaci PRESENT nebyla v žádném případě úspěšná. Vyobrazovat míru úspěšnosti, jak tomu bylo v sekci 5.2, nyní nenese žádnou výpovědní hodnotu, viz obr. 6.7. Výsledná míra úspěšnosti je kompletně ná-



Obr. 6.7: Míra úspěšnosti odhadu tajného klíče pro 1.-8. a 9.-16. čtyř bit maskované implementace.

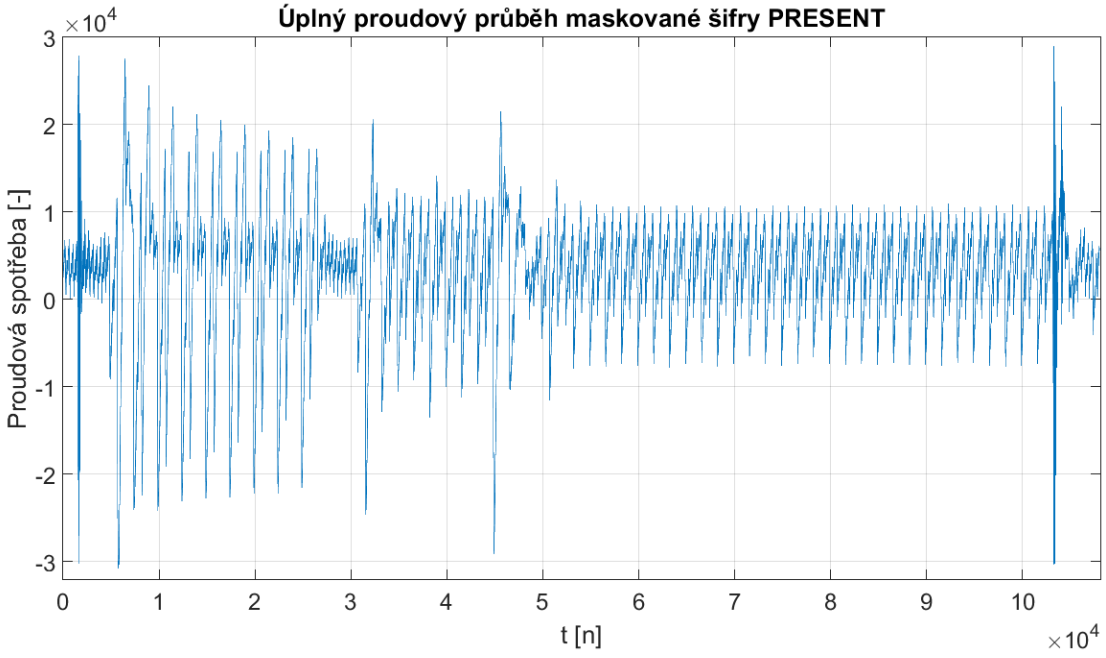
hodná, neudržuje žádný tvar a dokazuje správnost implementovaného protiopatření pro všechny čtyř bity tajného klíče. Při snaze pohlédnout na samotné hodnoty korelace byl vytvořen graf, který zobrazuje míru korelace pro jednotlivé odhady tajného klíče. Graf je zaměřený na šestnáctý čtyř bit tajného klíče. V ideálním případě, jako tomu bylo u nemaskované implementace šifry PRESENT 5.10, by jeden odhad dosahoval mnohem vyšších výsledků a oddělil by se od ostatních. Takto separovaný odhad by s rostoucím množství použitých proudových průběhů znatelně rostl, zatímco ostatní odhady klesají. Graf 6.8 neobsahuje žádný rostoucí odhad, nebo alespoň odhad, který by znatelně převyšoval ostatní. Správný odhad, označený červeně,



Obr. 6.8: Korelace pro jednotlivé hodnoty tajného klíče.

klesá spolu s ostatními odhady tajného klíče. Výsledky DPA dokazují její neúspěšnost a nebylo možné odhalit žádné užitečné informace vedoucí k odhalení tajného klíče. Což je požadovaný výsledek a bylo dokázáno, že implementované protiopatření skutečně zamezí proudové analýze prvního řádu.

Pokud budeme dále pokračovat v diskuzi, je možné naměřit ještě více proudových průběhů, které by mohly výsledky DPA kladně ovlivnit. Avšak, dle práce [33], ani 10 miliónů proudových průběhů neneslo užitečnou informaci o tajném klíči. Implementované protiopatření z této práce vychází a lze tedy předpokládat, že ani 10 miliónu proudových průběhů by výsledek DPA nezměnilo. Ovšem přichází v úvahu rozšířit pohled na naměřený proudový odběr a zaměřit se na celý, nejen na část nesoucí informace o rundách šifry PRESENT. Celý proudový odběr znázorňuje obrázek 6.9. Od hodnoty 5, na ose x, se nachází známý proudový průběh jednotlivých



Obr. 6.9: Naměřený maskovaný proudový průběh s částí masky.

rund. Jemu předchází vytvoření masky, která bude dále implementována. Takto tvořená maska je pro každé šifrování nových dat odlišná. Avšak, pokud by útočník měl toto zařízení k dispozici na neomezený čas, mohl by se pokusit o odhalení hodnoty masky. Masku by do zařízení mohl vkládat ručně a dle výsledných naměřených hodnot určit, které se rovnají. Pokud by odhalil správnou hodnotu masky použitou pro průběh šifrování, pokračoval by v útoku jako na implementaci bez aplikované masky. Což, dle práce [33], vede k možnému odhalení tajného klíče. V této implementaci protiopatření musí fungovat všechny tři zmíněné metody současně. Pokud je jedna eliminována, vede to k bezpečnostnímu riziku a DPA může být úspěšná. Útočník by musel zjistit použitou masku každého průběhu, což je velké množství.

Vezmeme-li v úvahu, že stále fungují i druhé dvě metody protiopatření, útočník bude potřebovat velké množství naměřených proudových průběhů. Každému průběhu musí najít shodu s použitou maskou a poté může přistoupit k samotné DPA. Útočník by musel rozhodnout, zda jeho práce a motivace jsou adekvátní k výsledkům, kterých by dosáhl, pokud by zjistil tajný klíč kryptografického modulu. Tento postup k získání tajného klíče je velmi časově nároční, avšak ne nemožný.

7 Závěr

Množství kryptografických zařízení roste a spolu s nimi i nároky na jejich zabezpečení. Útočník se nemusí zaměřit pouze na konvenční způsoby útoku, ale může využít proudový postranní kanál, pro získání požadovaných informací o použitém tajném klíči, nebo jeho konkrétní hodnoty. Proudový postranní kanál představuje závislost proudové spotřeby na právě zpracovávaných datech. Proudové průběhy jsou zpracovávány proudovou analýzou (PA), která se dále dělí na jednoduchou proudovou analýzu (SPA) a diferenční proudovou analýzu (DPA).

Cílem bakalářské práce bylo odhalit použitý tajný klíč kryptografického zařízení, které bylo realizováno FPGA deskou SAKURA-G s implementovanou odlehčenou šifrou PRESENT pracující s 80 bitovou délkou tajného klíče. Využívané proudové průběhy byly měřeny na experimentálním pracovišti za pomocí osciloskopu. Měření proudové spotřeby probíhalo přes bočník, který je již součástí desky SAKURA-G a synchronizace měření byla zajištěna přes uživatelské vstupní piny pomocí pasivní napěťové sondy.

Menší množství naměřených proudových průběhů bylo vstupem pro SPA a s využitím metody vizuální inspekce bylo cílem získat informace vedoucí k odhalení tajného klíče. Z proudových průběhů se podařilo určit všech 32 rund šifry PRESENT, ale žádné informace vedoucí například k odhalení informací o Hammingově váze tajného klíče. Do SPA také spadá nalezení a prozkoumání „zajímavých“ bodů, které byly výstupem prvního provedení DPA. Tyto body snižují hardwarové nároky na zařízení a časovou náročnost dalších provedení DPA, protože již není potřeba pracovat s celým proudových průběhem, ale jen s úsekem obsahujícím „zajímavé“ body. Během realizace samotné DPA, využívající korelační koeficient, byly použity modely Zero Value a Hammingova váha. Model Zero Value dosahoval lepších výsledků, avšak stále nedostatečných. Jeho využitím bylo odhaleno pouze 11 čtyř bitů, namísto 16. To bylo důvodem derivace vlastního modelu spotřeby, s jehož použitím již bylo odhaleno všech 16 čtyř bitů, neboli 8 bytů tajného klíče při cílení útoku na operaci *AddRoundKey*. Tím byl splněný dílčí cíl bakalářské práce, tedy úspěšně realizovat DPA. Odhalení posledních 2 bytů tajného klíče není možné pomocí PA, avšak lze využít metodu *Brute Force*. Její provedení není příliš časově náročné, z důvodu 256^2 možných kombinací, což ani s dosavadním výpočetním výkonem nepředstavuje náročný cíl.

Po úspěšné realizaci útoku DPA na nemaskovanou šifru PRESENT, následovala implementace protiopatření proti proudové analýze a opětovné provedení útoku na již maskovanou implementaci šifry. Protiopatření využívalo hardwarové rekonfigurace FPGA desky a sestávalo z dekompozice operace S-box, booleovského maskování a *register precharge*. FPGA rekonfigurace vytváří útočníkem nepředvídatelné změny

dekompozice S-box. Provedení SPA dosáhlo stejných výsledků jako u nemaskované implementace šifry PRESENT. Viditelný byl počet rund, ale bez možnosti zjistit informaci o Hammingově váze tajného klíče. Avšak v tomto případě, ani za pomocí DPA, nebylo možné odhalit hodnoty tajného klíče, nehledě na použitý model spotřeby. Tímto je účinnost implementovaného protiopatření otestována a představuje více než dostatečné zabezpečení proti proudové analýze, čímž byly všechny cíle bakalářské práce naplněny.

I přes dosažené výsledky a zvýšenou bezpečnost kryptografického zařízení zde vzniká možné riziko. Pokud by útočník byl schopný získat kryptografické zařízení, mohl by naměřit proudový průběh, který obsahuje informace o použité masce. Poté by naměřil proudové průběhy pro jednotlivé hodnoty masky (0-15), které by porovnal s originálním proudovým průběhem. Odpovídající průběh by určil použitou hodnotu masky a útočník by postupoval dále, jako při nemaskované implementaci. Naproti tomu by útočník musel použít tento postup pro každý proudový průběh a stále by zbývaly dvě implementované metody, které by zvyšovaly potřebné množství proudových průběhů pro úspěšné odhalení tajného klíče. Vzhledem k časové náročnosti je tento postup velmi nepravděpodobný a téměř nemožný. Na základě dosažených výsledků lze konstatovat, že veškeré definované cíle bakalářské práce byly splněny.

Literatura

[1] SPREITZER, Raphael, Veelasha MOONSAMY, Thomas KORAK and Stefan MANGARD. 2018. Systematic Classification of Side-Channel Attacks: A Case Study for Mobile Devices. IEEE Communications Surveys & Tutorials [online]. 20(1), 465-488. Dostupné z: http://ieeexplore.ieee.org/document/8141882/

[2] LORENC, V a V MATYÁŠ. Autentizační HW a možná vylepšení [online]. 2007 [cit. 28. 4. 2019].
Dostupné z: http://www.ics.muni.cz/bulletin/articles/563.html.

[3] GENKIN, Daniel, Adi SHAMIR a Eran TROMER. RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis. GARAY, Juan A. a Rosario GENNARO, ed. Advances in Cryptology – CRYPTO 2014 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, 2014, s. 444-461 [cit. 2018-11-30]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-662-44371-2_25. ISBN 978-3-662-44370-5.

[4] BONEH, Dan, Richard A. DEMILLO a Richard J. LIPTON. On the Importance of Checking Cryptographic Protocols for Faults. FUMY, Walter, ed. Advances in Cryptology — EUROCRYPT '97 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, 1997-7-13, s. 37-51 [cit. 2018-11-30]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-69053-0_4. ISBN 978-3-540-62975-7.

[5] BLEICHENBACHER, Daniel. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. KRAWCZYK, Hugo, ed. Advances in Cryptology — CRYPTO '98 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, 1998-5-28, s. 1-12 [cit. 2018-11-30]. Lecture Notes in Computer Science. DOI: 10.1007/BFb0055716. ISBN 978-3-540-64892-5.

[6] MARTINASEK, Zdenek, Vaclav ZEMAN a Krisztina TRASY. Simple Electromagnetic Analysis in Cryptography. International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems [online]. 2012, 1(1), 13-19 [cit. 2018-11-30]. DOI: 10.11601/ijates.v1i1.6. ISSN 1805-5443.

[7] DING, Guo-liang, Zhi-xiang LI, Xiao-long CHANG a Qiang ZHAO. Differential Electromagnetic Analysis on AES Cryptographic System. In: 2009 Second Pacific-Asia Conference on Web Mining and Web-based Application [online]. IEEE, 2009, 2009, s. 120-123 [cit. 2018-11-30]. DOI: 10.1109/WMWA.2009.46. ISBN 978-0-7695-3646-0.

[8] SANLYDE, D., S. SKOROBOGATOV, R. ANDERSON a J.-J. QUISQUATER. On a new way to read data from memory. In: First International IEEE Security in Storage Workshop, 2002. Proceedings [online]. IEEE Comput. Soc, 2003, s. 65-69 [cit. 2018-11-30]. DOI: 10.1109/SISW.2002.1183512. ISBN 0-7695-1888-5. http://ieeexplore.ieee.org/document/1183512/

[9] VAN ECK, Wim. Electromagnetic radiation from video display units: An eavesdropping risk?. Computers & Security [online]. 1985, 4(4), 269-286 [cit. 2018-11-30]. DOI: 10.1016/0167-4048(85)90046-X. ISSN 01674048.

[10] ROHATGI, Pankaj. Electromagnetic Attacks and Countermeasures. KOÇ, Çetin Kaya, ed. Cryptographic Engineering [online]. Boston, MA: Springer US, 2009, 2009, s. 407-430 [cit. 2018-11-30]. DOI: 10.1007/978-0-387-71817-0_15. ISBN 978-0-387-71816-3. Dostupné z: http://link.springer.com/10.1007/978-0-387-71817-0_15

[11] KOCHER, Paul C. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. KOBLITZ, Neal, ed. Advances in Cryptology — CRYPTO '96 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, 1996-7-13, s. 104-113 [cit. 2018-11-30]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-68697-5_9. ISBN 978-3-540-61512-5. Dostupné z: http://link.springer.com/10.1007/3-540-68697-5_9

[12] ARJUNAN, Amuthan, Praveena NARAYANAN a Kaviarasan RAMU. Securing RSA algorithm against timing attack. In: The International Arab Journal of Information Technology. 13. Indie, 2016, s. 471-476. Dostupné z: https://www.semanticscholar.org/paper/Securing-RSA-algorithm-against-timing-attack-Arjunan-Narayanan/89bab40626c419acb820400e3c38a0098b22ab7f

[13] NEVE, Michael a Kris TIRI. On the complexity of side-channel attacks on AES-256 – methodology and quantitative results on cache attacks. In: IACR Cryptology ePrint Archive [online]. Report 2007/318. 2007, s. 13 [cit. 2018-11-28]. Dostupné z: http://eprint.iacr.org/2007/318

[14] BERNSTEIN, J. Daniel. Cache-timing attacks on AES. Chicago, IL 60607–7045, 2005. The University of Illinois at Chicago.

[15] BONNEAU, Joseph a Ilya MIRONOV. Cache-Collision Timing Attacks Against AES. GOUBIN, Louis a Mitsuru MATSUI, ed. Cryptographic Hardware and Embedded Systems - CHES 2006 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, 2006, s. 201-215 [cit. 2018-11-29]. Lecture Notes in Computer Science. DOI: 10.1007/11894063_16. ISBN 978-3-540-46559-1.

[16] TAHA, H. Yaseen, ABDULH M. Settana, SADALLA A. Naila a ELSHOUSH Huwaida. Cache-timing attack against aes crypto system – countermeasures review. Edith Cowan University, Research Online, Australian Information Security Management, Conferences, Symposia and Campus Events. Dostupné z: https://ro.ecu.edu.au/cgi/viewcontent.cgi?referer=https://www.google.cz/&httpsredir=1&article=1167&context=ism

[17] KOCHER, Paul, Joshua JAFFE a Benjamin JUN. Differential Power Analysis. WIENER, Michael, ed. Advances in Cryptology — CRYPTO' 99 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, 1999-12-16, s. 388-397 [cit. 2018-11-18]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-48405-1_25. ISBN 978-3-540 6634 -8. Dostupné z: https://link.springer.com/chapter/10.1007/3-540-48405-1_25

[18] BAKER, R. Jacob. c2008. CMOS circuit design, layout, and simulation. Rev. 2nd ed. Hoboken, NJ: Wiley-Interscience.

[19] PEETERS, Eric, François-Xavier STANDAERT a Jean-Jacques QUISQUATER. Power and electromagnetic analysis: Improved model, consequences and comparisons. Integration [online]. 2007, 40(1), 52-60 [cit. 2018-11-18]. DOI: 10.1016/j.vlsi.2005.12.013. ISSN 01679260. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0167926005000647

[20] MARTINÁSEK, Zdeněk. Kryptoanalýza postranními kanály. Technická 3058/10, 616 00 Brno, 2013. Disertace. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací.

[21] MAYER-SOMMER, Rita. Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards. KOÇ, Çetin K. a Christof PAAR, ed. Cryptographic Hardware and Embedded Systems — CHES 2000 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, 2000-1-29, s. 78-92 [cit. 2018-11-18]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-44499-8_6. ISBN 978-3-540-41455-1.

[22] Choudary, Omar. Breaking Smartcards Using Power Analysis [online]. University of Cambridge [cit. 2018-11-18]. Dostupné z URL: https://www.cl.cam.ac.uk/ osc22/docs/smartcards.pdf.

[23] Courtois, Nicolas T. All About Side Channel Attacks [online]. [cit. 2018-11-18]. Applied Crypto COMPGA12. Dostupné z URL: http://www.nicolascourtois.com/papers/sc/sidech_attacks.pdf.

[24] MEDWED, Marcel and Elisabeth OSWALD. 2009. Template Attacks on ECDSA. CHUNG, Kyo-Il, Kiwook SOHN and Moti YUNG (eds.). Information Security Applications [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, p. 14-27. Lecture Notes in Computer Science. Dostupné z: http://link.springer.com/10.1007/978-3-642-00306-6_2

[25] MEDWED, Marcel. Template-Based SPA Attacks on 32-bit ECDSA Implementations. Inffeldgasse 16a Graz, Rakousko, 2007. Diplomová práce. Graz University of Technology, Faculty of Computer Science and Biomedical Engineering.

[26] KOCHER, Paul, Joshua JAFFE, Benjamin JUN a Pankaj ROHATGI. Introduction to differential power analysis. Journal of Cryptographic Engineering [online]. 2011, 1(1), 5-27 [cit. 2018-11-18]. DOI: 10.1007/s13389-011-0006-y. ISSN 2190-8508.

[27] GOUBIN, Louis a Jacques PATARIN. DES and Differential Power Analysis The "Duplication" Method. KOÇ, Çetin K. a Christof PAAR, ed. Cryptographic Hardware and Embedded Systems [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, 1999-2-8, s. 158-172 [cit. 2018-11-23]. Lecture Notes in Computer Science. DOI: 10.1007/3-540-48059-5_15. ISBN 978-3-540-66646-2.

[28] KOCHER, Paul, Joshua JAFFE, Benjamin JUN a Pankaj ROHATGI. Introduction to differential power analysis. Journal of Cryptographic Engineering [online]. 2011, 1(1), 5-27 [cit. 2018-11-29]. DOI: 10.1007/s13389-011-0006-y. ISSN 2190-8508. Dostupné z: http://link.springer.com/10.1007/s13389-011-0006-y

[29] RECHBERGER, Christian a Elisabeth OSWALD. Practical Template Attacks. LIM, Chae Hoon a Moti YUNG, ed. Information Security Applications [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, 2005, s. 440-456 [cit. 2018-11-26]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-540-31815-6_35. ISBN 978-3-540-24015-0.

[30] LO, Owen, William J. BUCHANAN a Douglas CARSON. Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA). Journal of Cyber Security Technology [online]. 2017, 1(2), 88-107 [cit. 2018-11-25]. DOI: 10.1080/23742917.2016.1231523. ISSN 2374-2917.

[31] Hoheisel, Andreas. Side-Channel Analysis Resistant Implementation of AES on Automotive Processors [online]. Diplomová práce, Ruhr-University Bochum, Německo, 2009, [cit. 2018-11-28].

[32] MENTENS, Nele. Hiding side-channel leakage through hardware randomization: A comprehensive overview. In: 2017 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS) [online]. IEEE, 2017, 2017, s. 269-272 [cit. 2018-12-04]. DOI: 10.1109/SAMOS.2017.8344639. ISBN 978-1-5386-3437-0.

[33] SASDRICH, Pascal, Amir MORADI, Oliver MISCHKE a Tim GUNEYSU. Achieving side-channel protection with dynamic logic reconfiguration on modern FPGAs. In: 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST) [online]. IEEE, 2015, 2015, s. 130-136 [cit. 2018-12-06]. DOI: 10.1109/HST.2015.7140251. ISBN 978-1-4673-7421-7.

[34] Ptáček, Luboš. Power analysis of AES [online]. Bakalářská práce, Masarykova univerzita Brno, Fakulta informačních technologií, 2008, [cit. 2018-11-28].

[35] VEYRAT-CHARVILLON, Nicolas, Marcel MEDWED, Stéphanie KERCKHOF a François-Xavier STANDAERT. Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note. WANG, Xiaoyun a Kazue SAKO, ed. Advances in Cryptology – ASIACRYPT 2012 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, 2012, s. 740-757 [cit. 2018-12-05]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-642-34961-4_44. ISBN 978-3-642-34960-7.

[36] FEI, Yunsi, et al. A Statistics-based Fundamental Model for Side-channel Attack Analysis. IACR Cryptology ePrint Archive, 2014, 2014: 152.

[37] KAMOUN, Najeh, Lilian BOSSUET and Adel GHAZEL. 2009. Correlated power noise generator as a low cost DPA countermeasures to secure hardware AES cipher. In: 2009 3rd International Conference on Signals, Circuits and Systems (SCS) [online]. IEEE, p. 1-6.
Dostupné z: http://ieeexplore.ieee.org/document/5412604/

[38] Reparaz, Oscar. Analysis and Design of Masking Schemes for Secure Cryptographic Implementations [online]. Dizertační práce, KU Leuven, Faculty of Engineering Science, Belgie, 2016, [cit. 2018-11-27].

[39] Duan, X., Cui, Q., Wang, S., Fang, H., & She, G. (2016). Differential power analysis attack and efficient countermeasures on PRESENT [Online]. In 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN) (pp. 8-12).
Dostupné z: https://doi.org/10.1109/ICCSN.2016.7586627

[40] RATANPAL, G.B., R.D. WILLIAMS a T.N. BLALOCK. An on-chip signal suppression countermeasure to power analysis attacks. IEEE Transactions on Dependable and Secure Computing [online]. 2004, 1(3), 179-189 [cit. 2018-11-29]. DOI: 10.1109/TDSC.2004.25. ISSN 1545-5971.

[41] CORON, Jean-Sebastien, David NACCACHE a Paul KOCHER. Statistics and secret leakage. ACM Transactions on Embedded Computing Systems [online]. 2004, 3(3), 492-508 [cit. 2018-11-29]. DOI: 10.1145/1015047.1015050. ISSN 15399087.

[42] MANGARD, Stefan. Hardware Countermeasures against DPA – A Statistical Analysis of Their Effectiveness. OKAMOTO, Tatsuaki, ed. Topics in Cryptology – CT-RSA 2004 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, 2004, s. 222-235 [cit. 2018-11-29]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-540-24660-2_18. ISBN 978-3-540-20996-6.

[43] CHEN, Shenghua, Wei GE, Jinjiang YANG, Bo LIU a Jun YANG. A Power Analysis Attack Countermeasure Based on Random Execution. In: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE) [online]. IEEE, 2018, 2018, s. 1474-1479 [cit. 2018-11-29]. DOI: 10.1109/TrustCom/BigDataSE.2018.00206. ISBN 978-1-5386-4388-4.

[44] GUNTUR, Hendra, Jun ISHII a Akashi SATOH. Side-channel AttacK User Reference Architecture board SAKURA-G. In: 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE) [online]. IEEE, 2014, 2014, s. 271-274 [cit. 2018-12-07]. DOI: 10.1109/GCCE.2014.7031104. ISBN 978-1-4799-5145-1.

[45] BOGDANOV, A., L.R. KNUDSEN, G. LEANDER, C. PAAR, A. POSCHMANN, M. J. B. ROBSHAW, Y. SEURIN a C. VIKKELSOE. PRESENT: An Ultra-Lightweight Block Cipher. PAILLIER, Pascal a Ingrid VERBAUWHEDE, ed. Cryptographic Hardware and Embedded Systems - CHES 2007 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, s. 450-466 [cit. 2018-12-09]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-540-74735-2_31. ISBN 978-3-540-74734-5.

[46] LARA-NINO, Carlos Andres, Arturo DIAZ-PEREZ and Miguel MORALES-SANDOVAL. 2017. Lightweight Hardware Architectures for the Present Cipher in FPGA. IEEE Transactions on Circuits and Systems I: Regular Papers [online]. 64(9), 2544-2555.
Dostupné z: http://ieeexplore.ieee.org/document/7911280/

[47] LO, Owen, William J. BUCHANAN and Douglas CARSON. 2017. Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA). Journal of Cyber Security Technology [online]. 1(2), 88-107.
Dostupné z: https://tandfonline.com/doi/full/10.1080/23742917.2016.1231523

[48] LO, Owen, William J. BUCHANAN and Douglas CARSON. 2018. Correlation Power Analysis on the PRESENT Block Cipher on an Embedded Device. In: Proceedings of the 13th International Conference on Availability, Reliability and Security - ARES 2018 [online]. New York, USA: ACM Press, p. 1-6.
Dostupné z: http://dl.acm.org/citation.cfm?doid=3230833.3232801

[49] BREJNÍK, Jan. 2019. OBRANY PROTI ÚTOKŮM POSTRANNÍMI KANÁLY ZALOŽENÉ NA DYNAMICKÉ REKONFIGURACI FPGA. Praha. Magisterská práce. České vysoké učení technické v Praze. Vedoucí práce Stanislav Jeřábek.

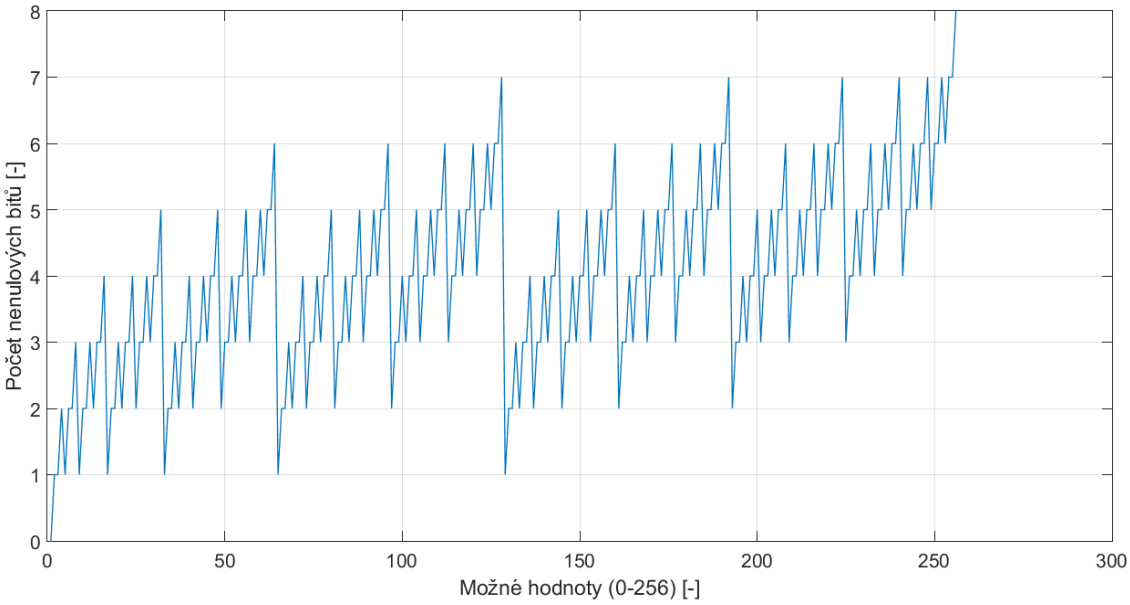
[50] ROY, Debapriya Basu, Shivam BHASIN, Sylvain GUILLEY, Jean-Luc DANGER, Debdeep MUKHOPADHYAY, Xuan Thuy NGO and Zakaria NAJM. 2015. Reconfigurable LUT: A Double Edged Sword for Security-Critical Applications. CHAKRABORTY, Rajat Subhra, Peter SCHWABE and Jon SOLWORTH (eds.). Security, Privacy, and Applied Cryptography Engineering [online]. Cham: Springer International Publishing, p. 248-268. Lecture Notes in Computer Science.
Dostupné z: http://link.springer.com/10.1007/978-3-319-24126-5_15

Seznam symbolů, veličin a zkratek

AES	Advanced Encryption Standard – Standard symetrického šifrování
RSA	R. Rivest A. Shamir a L. Adleman – Jména autorů asymetrické šifry
DES	Data Encryption Standard – Standardizovaná symetrická šifra ze sedmdesátých let 20. století
CPU	Central Processing Unit – Centrální procesorová jednotka
OAEP	Optimal Asymmetric Encryption Padding
CMOS	Complementary Metal-Oxide-Semiconductor
PA	Power Analysis – Proudová analýza
SPA	Simple Power Analysis – Jednoduchá proudová analýza
DPA	Differential Power Analysis – Diferenční proudová analýza
ECDSA	Elliptic Curve Digital Signature Algorithm – Algoritmus digitálního podpisu využívající eliptické křivky
SNR	Signal-to-Noise Ratio – Odstup signálu od šumu
RFID	Radio Frequency Identification – Identifikace na rádiové frekvenci
FPGA	Field Programmable Gate Array – Programovatelná hradlová pole

Seznam příloh

A Graf Hammingovy váhy



Obr. A.1: Model Hammingovy váhy

B Obsah přiloženého CD

- Elektronická verze této bakalářské práce ve formátu pdf.
- Složka obsahující všechny použité obrázky, spolu s dosaženými výsledky ve formě grafů.