VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

REALIZACE DIFERENCIÁLNÍ PROUDOVÉ ANALÝZY

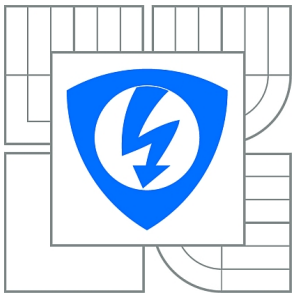
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE PAVEL MAREK
AUTHOR

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

REALIZACE DIFERENCIÁLNÍ PROUDOVÉ ANALÝZY

REALIZATION OF DIFFERENTIAL POWER ANALYSIS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

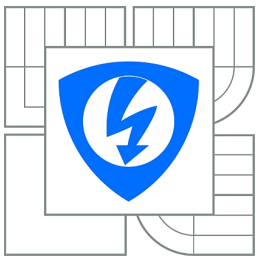
AUTOR PRÁCE — PAVEL MAREK

AUTHOR

VEDOUCÍ PRÁCE — Ing. ZDENĚK MARTINÁSEK, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Pavel Marek	***ID:***	136558
Ročník:	3	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Realizace diferenciální proudové analýzy

POKYNY PRO VYPRACOVÁNÍ:

V rámci bakalářské práce prostudujte problematiku proudového a elektromagnetického postranního kanálu. Pozorně prostudujte jednoduchou a diferenční analýzu například na šifrovacích algoritmech RSA a AES. Realizujte experimentální pracoviště určené k analýze proudovým popřípadě elektromagnetickým postranním kanálem. Pracoviště bude vybaveno vhodným vybavením umožňující připojení sondy a mělo by být uzpůsobeno k automatizovanému měření (tzn. realizace požadovaných 1000 a více měření bez nutnosti obsluhy). K realizaci využijte vývojové sady pro procesory Atmel nebo PIC. Na pracovišti realizujte jednoduchou a diferenční analýzu pro zvolené kryptografické moduly a výsledky zhodnoťte.

DOPORUČENÁ LITERATURA:

[1] Mangard, S.; Oswald, E.; Popp, T.: Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security). Secaucus, NJ, USA:Springer-Verlag New York, Inc., 2007, ISBN 0387308571.

[2] Kocher, P. C.; Jaffe, J.; Jun, B.: Differential Power Analysis. In CRYPTO ’99: Proceedings of the 19th Annual International Cryptology Conference onAdvances in Cryptology, London, UK: Springer-Verlag, 1999, ISBN 3-540-66347-9, s. 388–397.

Termín zadání:	10.2.2014	***Termín odevzdání:***	4.6.2014

Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

ABSTRAKT

V dnešní době nestačí věnovat pozornost bezpečnosti šifrovacího algoritmu pouze z matematického hlediska. Je také nutné věnovat pozornost implementaci šifrovacího algoritmu, protože šifrovací zařízení může o implementovaném šifrování prozrazovat nežádoucí komunikací mnoho informací. Tato bakalářská práce se zabývá problematikou postranních kanálů (především proudového postranního kanálu) a jejich využitím k získání tajného klíče šifrovacího algoritmu AES. K tomuto účelu jsou v této práci realizovány tři úkony. Nejprve je navrženo experimentální pracoviště, které zajišťuje automatické zaznamenávání průběhů šifrování. Následně je provedena jednoduchá i diferenciální proudová analýza těchto průběhů. Všechny části jsou v jednotlivých kapitolách popsány teoreticky. Po teoretickém popisu následuje praktická část, ve které je popsána vlastní realizace všech úkolů.

KLÍČOVÁ SLOVA

AES, ATmega8, ATmega16, automatické měření průběhů, diferenciální proudová analýza, kryptografický modul, postranní kanály, proudový postranní kanál

ABSTRACT

Nowadays, it is not enough to pay attention to encryption algorithm security from the mathematical aspect only. It is also necessary to pay attention to the implementation of encryption algorithm, because encryption devices can show plenty of information about implemented encryption via undesirable communication. This bachelor thesis deals with side channels issues (especially power side channel) and their use to obtain the secret key of AES encryption algorithm. For this purpose there are three operations realized in this thesis. At first, there is experimental workplace designed, which provides automatic saving of waveforms of encryption. Then there is a simple and differential power analysis of these waveforms performed. All parts are theoretically described in individual chapters. After theoretical description there is the practical part, which describes a proper realization of all tasks.

KEYWORDS

AES, ATmega8, ATmega16, automatic waveform measurements, differential power analysis, cryptographic module, side channels, power side channel

MAREK, Pavel *Realizace diferenciální proudové analýzy*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014. 67 s. Vedoucí práce byl Ing. Zdeněk Martinásek, PhD.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Realizace diferenciální proudové analýzy“ 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)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Kryptografické zařízení (například čipové karty) jsou stále častěji využívány pro ověřování pravosti nebo jako bezpečná úložiště tajných informací. Algoritmy implementované v kryptografických zařízeních jsou na takové matematické úrovni, že jejich prolomení a zjištění tajného klíče je prakticky nemožné, protože vyžaduje značný výpočetní čas a výpočetní výkon. Proto útočník hledá způsoby, jak snížit výpočetní čas a výpočetní výkon do přijatelných mezí.

Jednou z alternativ, jak dosáhnout značného snížení výpočetního času a výpočetního výkonu, je útok s využitím nežádoucí komunikace kryptografického zařízení. Jedná se o způsob útoku, při kterém se neútočí na matematickou podstatu šifrovacího algoritmu, ale útočí se na jeho implementaci. Nežádoucí komunikace je způsobena fyzikálními zákony a konstrukcí kryptografického zařízení. Nežádoucí komunikace kryptografického zařízení proto představuje velké bezpečnostní riziko. Proto se kryptoanalýze těchto modulů věnuje v dnešní době stále větší pozornost. Kryptoanalýza se zde může rozdělit na kryptoanalýzu implementovaného šifrovacího algoritmu a na kryptoanalýzu postranních kanálů použitého hardwaru.

V dnešní době je známo již mnoho technik analýzy, pomocí kterých lze z nežádoucí komunikace kryptografického zařízení získat tajné informace jako například šifrovací klíč. Mezi tyto techniky patří jednoduchá a diferenciální proudová analýza.

1 ÚTOKY POSTRANNÍMI KANÁLY

Cílem této kapitoly je základní seznámení s kryptografickým modulem a uvedení do problematiky postranních kanálů a využití těchto postranních kanálů k útoku na kryptografické moduly. Je zde popsáno, co to vlastně postranní kanály jsou. Dále je také popsáno, jaké typy postranních kanálů lze pro útoky využít a jakým způsobem.

1.1 Definice kryptografického modulu

Kryptografický modul je fyzickou interpretací konkrétního kryptografického algoritmu. K zajištění bezpečnosti poskytuje kryptografický modul následující služby:

- důvěryhodnost - utajení informace před neoprávněnou osobou,
- autentičnost - schopnost ověření autora,
- integritu - zabezpečení proti modifikaci zprávy,
- nepopíratelnost - odesílatel nemůže odeslání dané zprávy popřít.

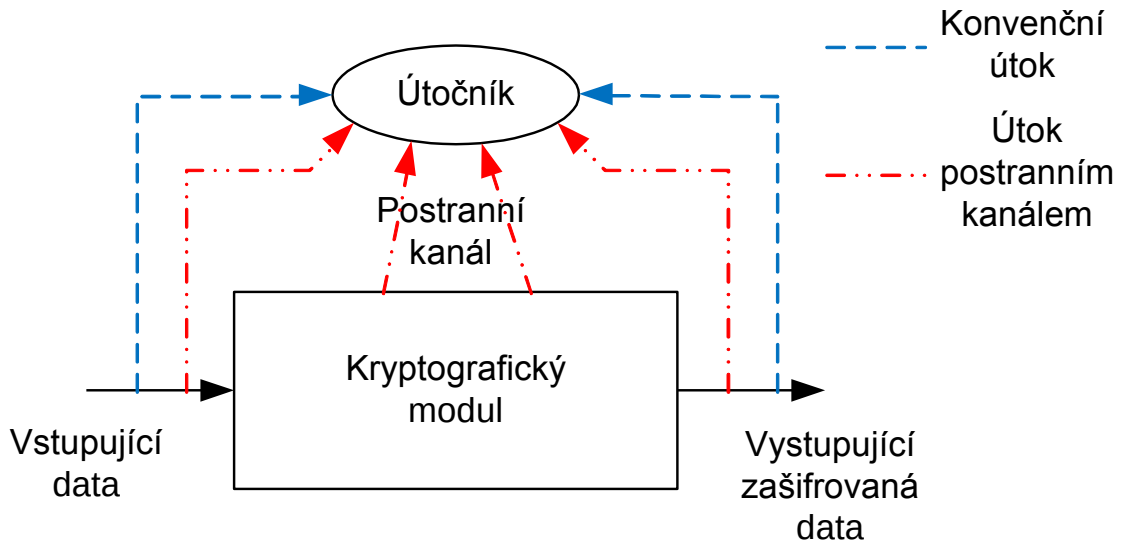
Kryptografický modul je konstruován tak, aby komunikoval s okolím pouze definovanými vstupy a výstupy. Uvnitř modulu probíhají matematické operace (například šifrování). To představuje jeho běžnou činnost. Avšak při této činnosti kryptografický modul vyzařuje například teplo a elektromagnetické pole, spotřebuje určitý čas nebo také odebírá proud z napájecího zdroje. Tato nežádoucí komunikace bývá označována za postranní kanál.

1.2 Definice útoku postrannímí kanály

Postranní kanál představuje nežádoucí komunikaci kryptografického modulu s okolním prostředím. Je vždy prezentovaný nějakou fyzikální veličinou, kterou útočník může změřit a vhodnými způsoby využít pro získání požadované informace [10].

Kryptografický modul může pomocí postranních kanálů vyzařovat do okolí informace, které jsou často závislé na implementovaném algoritmu. Případný útočník může takto zjistit různé informace týkající se šifrovacího algoritmu, který je do kryptografického modulu implementovaný. Mezi tyto informace patří například typ šifrovacího algoritmu, doba trvání kompletního šifrování nebo určité části šifrování. Využitím postranních kanálů lze však zjistit například i tajný klíč, který je pro šifrování využívaný. Tento fakt představuje velké riziko z hlediska bezpečnosti kryptografického modulu.

Na obrázku č. 1.1 jsou znázorněny modely 2 typů útoků na kryptografický modul. Jedná se nejprve o útok konvenční. Do této kategorie patří například útok hrubou



Obr. 1.1: Útoky na kryptografický modul.

silou. Útočník při tomto typu útoku využívá jen standardně definované vstupy a výstupy. Dále se jedná právě o útok s využitím postranních kanálů. Vhodné využití těchto postranních kanálů, společně s daty, která vstupují do kryptografického modulu a zašifrovanými daty, která z kryptografického modulu vystupují, může vézt k nežádoucímu úniku tajného klíče.

1.3 Typy postranních kanálů

Pro realizaci útoku na kryptografický modul je možné snadně využít několik typů postranních kanálů. V této práci je pro realizaci útoku využit elektromagnetický a především proudový postranní kanál. V této podkapitole je popsáno 5 využitelných postranních kanálů, a t elektromagnetický, časový, chybový, optický a akustický. Proudovému postrannímu kanálu je pak věnovaná samostatná kapitola č. 2.

Elektromagnetický postranní kanál
Při útoku elektromagnetickým postranním kanálem se využívá skutečnosti, že kryptografický modul vyzařuje při své činnosti proměnné elektromagnetické pole. Vnitřní strukturu kryptografického modulu tvoří několik milionů tranzistorů a spojů, kterými protékají proudy, které jsou závislé na právě zpracovávaných datech. Právě tyto proudy vytvářejí elektromagnetické pole, které lze následně zaznamenat a analyzovat. [8, 9, 10]

Časový postranní kanál

Při realizaci útoku časovým postranním kanálem se předpokládá, že každá operace trvá určitý čas. Operací se rozumí například vykonání určité instrukce nebo zápis do paměti. Tyto vykonávané operace mají různou dobu trvání, která je závislá na zpracovávaných datech a na šifrovacím klíči. Toho lze při analýze využít. [8, 9, 10]

Chybový postranní kanál

Při útoku chybovým postranním kanálem se analyzuje komunikace kryptografického modulu s okolím při vzniku chyby. Útočník může uvézt kryptografický modul do chybového stavu například změnou napájecího napětí nebo změnou okolní teploty. [8, 9, 10]

Optický postranní kanál

Kryptografický modul může při své činnosti vyzařovat energii v podobě fotonů. Toto vyzařování fotonů lze vhodným zařízením zachytit a následná analýza může vézt k odhalení šifrovacího klíče. [8, 9, 10]

Akustický postranní kanál

Při útoku s využitím akustického postranního kanálu se sledují například zvuky jednotlivých tlačítek klávesnice kryptografického modulu. Další možností je realizace útoku akustickým postranním kanálem na mikroprocesor tak, že se zaznamenává spektrum různých operací, které mikroprocesor vykonává. [8, 9, 10]

2 PROUDOVÝ POSTRANNÍ KANÁL

Kryptografický modul během šifrování spotřebuje určité množství energie. To znamená, že odebírá proud ze zdroje. Tento odebíraný proud není konstantní. Má závislost na datech, které kryptografický modul právě zpracovává.

Kryptografický modul je složen z integrovaných obvodů, které jsou nejčastěji založeny na technologii CMOS (*Complementary Metal-Oxide-Semiconductor*) [8]. V této kapitole je popsáno, jakým způsobem ovlivňují zpracovávaná data proudový odběr kryptografických modulů založených na technologii CMOS.

Proudová spotřeba obvodů CMOS

Celý obvod CMOS se skládá z několika elementárních logických buněk. Tuto elementární buňku může tvořit například invertor [8]. Celková proudová spotřeba obvodu CMOS je pak dána součtem proudových spotřeb jednotlivých buněk. Z tohoto důvodu je celková proudová spotřeba závislá na počtu logických buněk, na spojení mezi nimi a také na tom, jak jsou buňky tvořeny [8].

Kryptografický modul bývá napájen zdrojem konstantního napětí U_{DD}. Celkový okamžitý odebíraný proud se označuje $i_{DD}(t)$ a okamžitá výkonový spotřeba se označuje $p_{obv}(t)$. Průměrná výkonová spotřeba P_{obv} za čas T lze vypočítat ze vztahu:

$$P_{obv} = \frac{1}{T}\int_0^T p_{obv}(t)\mathrm{d}t = \frac{U_{DD}}{T}\int_0^T i_{DD}(t)\mathrm{d}t. \tag{2.1}$$

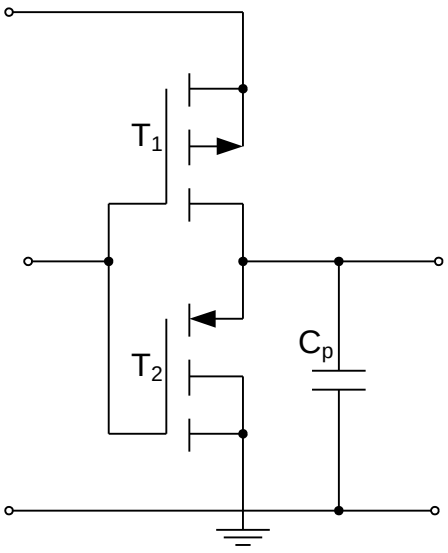
Proudová spotřeba lze v podstatě rozdělit na dvě části. První část lze označit jako statická spotřeba P_{stat} [8]. Druhá část lze označit jako dynamická spotřeba P_{dyn} [8]. Celková spotřeba je pak dána součtem statické a dynamické spotřeby (P_{stat}+ P_{dyn}). V elementární buňce (například invertoru na obrázku 2.1) pak může v určitém časovém úseku dojít na výstupu ke čtyřem přechodům mezi stavy (tabulka č. 2.1). Tyto přechody jsou definovány určitou proudovou spotřebou.

Tab. 2.1: Přechody výstupu elementární buňky.

Přechod	**Označení spotřeby**	**Spotřeba**
$0 \to 0$	P_{00}	statická
$0 \to 1$	P_{01}	statická + dnymaická
$1 \to 0$	P_{10}	statická + dnymaická
$1 \to 1$	P_{11}	statická

Pro přechody $0 \to 0$ a $1 \to 1$ je celková spotřeba dána pouze spotřebou statickou. Pro přechody $0 \to 1$ a $1 \to 0$ je celková spotřeba dána součtem spotřeby statické

a dynamické. Pro jednotlivé spotřeby pak platí, že $P_{00} \approx P_{11} \ll P_{01}, P_{10}$. Z toho lze určit, že dynamická spotřeba závisí na právě zpracovávaných datech.



Obr. 2.1: Příklad elementární buňky CMOS, schéma invertoru.

Statická spotřeba obvodu CMOS

Statická výkonová spotřeba P_{stat} je odebírána elementární buňkou v klidových stavech [8]. Lze ji vypočítat ze vztahu:

$$P_{stat} = I_{zbytk} \cdot U_{DD}, \tag{2.2}$$

kde I_{zbyt} je velmi malý (zbytkový) proud, který buňkou protéká.

Dynamická spotřeba obvodu CMOS

Dynamická výkonová spotřeba P_{dyn} je odebírána elementární buňkou ve stavech, kdy se mění hodnota na výstupu [8]. Existují dva typy dynamické výkonové spotřeby, které se dají označit jako svodová dynamická výkonová spotřeba a nabíjecí dynamická výkonová spotřeba.

Svodová výkonová spotřeba nastává při přepnutí stavu buňky, kdy je napájení zkratováno například přes zároveň otevřené tranzistory T1 a T2 invertoru (obrázek 2.1) zemi, protéká svodový proud. Lze ji vypočítat ze vztahu:

$$P_{dyn} = \frac{1}{T} \int_0^T p_{svod}(t) \mathrm{d}t = P_{0\to1} \cdot f \cdot U_{DD} \cdot I_{svod} \cdot t_{svod}, \tag{2.3}$$

kde $p_{svod}(t)$ je okamžitá výkonová spotřeba buňky během krátkého časového okamžiku T, $P_{0\to1}$ je pravděpodobnost přechodu $0 \to 1$, f je spínací kmitočet, U_{DD} je

napájecí napětí, I_{svod} je svodový proud a t_{svod} je doba, po kterou jsou tranzistory otevřeny.

Nabíjecí výkonová spotřeba je způsobena nabíjením parazitní kapacity invertoru proudem I_C. Ten představuje dominantní zdroj výkonové spotřeby [8]. Velikost parazitní kapacity závisí na fyzikálních vlastnostech použitého materiálu, výrobní technologii atd. Dynamická spotřeba způsobená nabíjením parazitní kapacity lze vyjádřit vztahem:

$$P_{dyn} = \frac{1}{T} \int_0^T p_{nab}(t) \mathrm{d}t = P_{0\to1} \cdot f \cdot C_p \cdot U_{DD}^2, \tag{2.4}$$

kde $p_{nab}(t)$ je okamžitá výkonová spotřeba buňky během nabíjení parazitní kapacity za čas T, $P_{0\to1}$ je pravděpodobnost přechodu $0 \to 1$, f je spínací kmitočet, C_p je parazitní kapacita a U_{DD} je napájecí napětí.

2.1 Jednoduchá proudová analýza

V této kapitole je na základě zadání prostudována jednoduchá proudová analýza (SPA - *Simple Power Analysis*) a také jsou uvedeny některé typy jednoduché proudové analýzy (přímá interpretace proudových průběhů, analýza s využitím šablon a kolizní útok).

Jednoduchá proudová analýza je technika útoku na šifrovací algoritmus, který je implementovaný do kryptografického modulu. Základním předpokladem pro úspěšné vykonání jednoduché proudové analýzy je, že vykonávaný algoritmus (především pak jeho šifrovací klíč), ať už přímo nebo nepřímo, nějakým způsobem ovlivňuje proudovou spotřebu kryptografického modulu [10]. Jednoduchost této analýzy spočívá ve skutečnosti, že pro její realizaci stačí útočníkovi změřit pouze pár průběhů proudové spotřeby při šifrování nebo za určitých podmínek v některých případech dokonce stačí zaznamenat jen jediný proudový průběh šifrování [8]. Tato analýza však vyžaduje poměrně detailní znalost šifrovacího algoritmu, na který je útok prováděn.

V případě analýzy jediného změřeného průběhu se jednoduchá proudová analýza označuje jako *single-shot* SPA [8]. V případě analýzy několika změřených průběhů se jednoduchá proudová analýza označuje jako *multiple-shot* SPA [8]. V případě *multiple-shot* je možné průběhy měřit dvěma způsoby. V prvním případě můžeme měřit průběhy pro šifrování několika různých dat. V případě druhém je možné změřit několik průběhů pro stejná data. Druhý případ má výhodu v tom, že se změřené průběhy mohou zprůměrovat. Zprůměrováním pak dojde k potlačení šumu.

2.1.1 Přímá interpretace proudových průběhů

Přímá interpretace proudových průběhů využívá toho, že každý šifrovací algoritmus je v kryptografickém modulu vykonáván v určitém pořadí. Algoritmus implementovaný do kryptografického modulu je přeložen do strojových instrukcí. Jako příklad může sloužit šifra AES (*Advanced Encryption Standard*), která je vykonávána v několika krocích, které se nazývají rundy [5]. Každá runda pak obsahuje operace s šifrovanými daty (substituce bajtů, rotace řádků, násobení maticí a přičtení rundovního klíče) [5]. Tato data jsou pak zpracovávána strojovými instrukcemi po jednotlivých bajtech.

Při zpracování jednotlivých bajtů se využívají různé typy strojových instrukcí a různé hardwarové části kryptografického modulu. Příklad strojových instrukcí může být instrukce pro přesun dat a instrukce pro aritmetické nebo logické operace. Mezi využívané hardwarové části kryptografického modulu patří například aritmeticko-logická jednotka a paměti typu ROM (*Read-Only Memory*) nebo RAM (*Random-Access Memory*). Právě vykonávané instrukce a používané hardwarové části mají specifickou proudovou spotřebu, která je může charakterizovat. [10]

Jelikož sekvence vykonávaných instrukcí má určitou závislost na hodnotě šifrovacího klíče, tak šifrovací klíč má proto i vliv na proudové průběhy. Tohoto poznatku lze využít pro zjišťování tajného klíče nebo pro také vyšetřování toho, zda proudový postranní kanál vyzařuje informace o implementovaném šifrování.

Je prokázáno, že implementovaný algoritmus obsahující modulární umocňování (využívá operace druhé mocniny a násobení), je vhodný pro realizaci jednoduché proudové analýzy, pokud je případný útočník právě operace druhé mocniny a násobení schopný určit ze změřených proudových průběhů šifrování [8]. Modulární umocňování využívají některé asymetrické šifrovací algoritmy. Například dešifrování RSA(*Rivest-Shamir-Adleman*) nebo generování podpisu RSA, kdy obě funkce vyžadují vykonávání modulárního umocňování s tajným klíčem.

Na počátku šifrování RSA si kryptografický modul volí 2 velká prvočísla p a q, které udržuje v tajnosti. Dále ve většině implementací RSA často dochází k tomu, že kryptografický modul nejprve zkontroluje, zda mají vstupní data větší velikost než právě pročíslo p nebo q. Pokud jsou vstupní data větší, provádí se jejich redukce. Právě tuto redukci vstupních dat může útočník určit ve změřeném proudovém průběhu, který na zpracovávaných vstupních datech závisí [8]. Útočník může dále volit různé velikosti vstupních dat a v proudových průbězích pozorovat to, zda dochází k redukci nebo ne až do doby, kdy se dostane k těsné blízkosti hodnoty neznámého prvočísla. Hodnotu prvočísla může tímto způsobem postupně celou odhalit.

2.1.2 Analýza s využitím šablon

Analýza proudových průběhů s využitím šablon se rozděluje do dvou fází. V první fázi se vykoná charakterizace zařízení pomocí šablon, ve druhé fázi se pak tato charakterizace využije pro útok. Tento typ jednoduché proudové analýzy je založen na předpokladu, že proudová spotřeba závisí také na datech, která se zpracovávají (šifrují) [8]. Proudové průběhy jsou charakterizovány vícerozměrným normálním rozdělením, které je definováno vektorem středních hodnot a kovarianční maticí $(\mathbf{m}, \mathbf{C})$ [10].

Při analýze s využitím šablon útočník předpokládá, že lze kryptografický modul a posloupnost instrukcí kryptografickým modulem vykonávaných dat charakterizovat právě šablonou $(\mathbf{m}, \mathbf{C})$. Pro charakterizaci této posloupnosti instrukcí a pro charakterizaci zařízení je pro útočníka nejjednodušší využít vlastní kryptografický modul stejného typu jako je modul, který je cílem útoku. Změřením proudových průběhů pro různá vstupní data d_i a pro různé šifrovací klíče k_j si útočník seskupí odpovídající průběhy (d_i, k_j) a vypočte si vektor středních hodnot a kovarianční matici vícerozměrného normálního rozdělení. Výsledkem je šablona pro všechny dvojice dat a klíčů $(d_i, k_j) : h_{d_i,k_j} = (\mathbf{m}, \mathbf{C})_{d_i,k_j}$.

Útočník pak může využít vytvořené šablony společně s průběhy změřenými na analyzovaném zařízení pro zjištění tajného klíče. K tomuto účelu se počítá pravděpodobnost vícerozměrného normálního rozdělení $(\mathbf{m}, \mathbf{C})_{d_i,k_j}$ a změřených proudových průběhů pomocí vztahu:

$$p(t; (\mathbf{m},\ \mathbf{C})_{d_i,k_j}) = \frac{exp(-\frac{1}{2} \cdot (\mathbf{t} - \mathbf{m})' \cdot \mathbf{C}^{-1} \cdot (\mathbf{t} - \mathbf{m}))}{\sqrt{(2 \cdot \pi)^T \cdot det(\mathbf{C})}}. \tag{2.5}$$

Vyhodnocením této pravděpodobnosti pak útočník odhaduje klíč použitý v analyzovaném zařízení, protože pravděpodobnosti ukazují jak dobře šablona odpovídá změřenému průběhu. Klíč obsažen v šabloně s nejvyšší pravděpodobností pak odpovídá klíči, který se útočník snaží zjistit [10].

2.1.3 Kolizní útok

Kolizní útok je založen na změření dvou proudových průběhů při šifrování různých vstupních dat neznámým klíčem. Z těchto dvou změřených proudových průběhů se pak určí vhodným zpracováním takzvaná hodnota, která odpovídá funkci vstupní proměnné a neznámého klíče. Pokud je tato určená hodnota stejná pro obě vstupní data, lze tato data a jejich průběhy označit za kolidující [8]. Tato kolize však nenastává pro všechny kombinace tajného klíče, ale nastává pouze v případech, že obě zkoumaná data byla šifrována určitou podskupinou klíčů.

Tímto způsobem je dosaženo značného snížení počtu kombinací hledaného tajného klíče. Dalším opakováním kolizního útoku lze pak identifikovat i přesnou hodnotu tajného klíče. [8]

Příklad kolizního útoku je uveden opět na algoritmu AES. Základem tohoto útoku jsou následující předpoklady. Zaprvé, útočník si může určit data, která nechá šifrovat. Zadruhé, útočník je schopen ve změřeném proudovém průběhu určit první rundu šifry AES. Zatřetí, útočník je schopen z proudových průběhů určit Hammingovy váhy jednotlivých bajtů prvního rundovního klíče.

V první rundě šifry AES může kolize nastat po operaci kombinování sloupců. Zde je určená hodnota stejná pro různé dva bloky otevřeného textu vstupujícího do šifry, které jsou například označeny jako d a d':

$$b_0 = 02 \cdot S(d_0 \oplus k_0) \oplus 03 \cdot S(d_1 \oplus k_1) \oplus S(d_2 \oplus k_2) \oplus S(d_3 \oplus k_3) \tag{2.6}$$

$$b_0^x = 02 \cdot S(d'_0 \oplus k_0) \oplus 03 \cdot S(d'_1 \oplus k_1) \oplus S(d'_2 \oplus k_2) \oplus S(d'_3 \oplus k_3) \tag{2.7}$$

K dosažení tohoto cíle je možné vybrat dva bloky otevřeného textu pro zašifrování d a d^x tak, aby $d_0 = d'_0 = 0$, $d_1 = d'_1 = 0$, $d_2 = d_3 = a$ a $d'_2 = d'_3 = b$. Potom dojde mezi d a d' ke kolizi:

$$b_0 \oplus b'_0 = S(a \oplus k_2) \oplus S(a \oplus k_3) \oplus S(b \oplus k_2) \oplus S(b \oplus k_3) = 0 \tag{2.8}$$

Parametry a a b jsou známy a dále je možno prozkoumat všechny možné hodnoty k_2 a k_3 a rozhodnout, které vyhovují. Tímto se zredukuje počet možných klíčů. Vhodným výběrem dat pro šifrování je dokonce možno určit tajný klíč přesně.

2.2 Diferenciální proudová analýza

Cílem této kapitoly je studie obecných principů a popis realizace diferenciální proudové analýzy. V rámci její prezentace je uveden opět příklad analýzy na šifrovacím algoritmu AES.

Diferenciální proudová analýze je označována jako DPA (*Differencial Power Analysis*). Na rozdíl od jednoduché proudové analýzy, diferenciální proudová analýza nevyžaduje detailní znalost šifrovacího algoritmu, na který ji chce útočník aplikovat. Dále lze pomocí této analýzy odhalit tajný klíč ze změřených průběhů, které jsou zašuměné. Na druhou stranu je však pro její realizaci nutné změřit podstatně více proudových průběhů a to v řádu stovek až tisíc [10].

Hlavním cílem diferenciální proudové analýzy je odhalení tajného klíče šifry, která je implementovaná do kryptografického modulu. K odhalení klíče se využívá velké množství proudových průběhů, které se měří při šifrování (nebo také při dešifrování) neustále se měnících dat. Jak již bylo výše uvedeno, pro realizaci této analýzy

není nutná detailní znalost algoritmu implementovaného do kryptografického modulu, který je cílem útoku. Dostatečné je vědět pouze to, o jaký šifrovací algoritmus se jedná. [8, 10]

DPA se zaměřuje na to, jak průběhy proudové spotřeba závisí v určitých časových okamžicích na zpracovávaných datech. Útok s vyžitím DPA bude nyní popsán v následujících pěti krocích a je znázorněn blokovým schématem na obrázku 2.2.

První krok: Stanovení mezivýsledku zpracovávaného algoritmu
V tomto kroku se provádí výpočet mezivýsledku s využitím nekonstantních vstupních šifrovaných dat označených například d a z malé části šifrovacího klíče označeného k. Tuto malou část klíče může tvořit jeden bajt. Vypočtený mezivýsledek je funkcí $f(d, k)$. [8]

Druhý krok: Měření proudové spotřeby
Ve druhém kroku se měří proudová spotřeba (proudové průběhy) šifrování nebo naopak dešifrování pro různá vstupní data, jejichž počet je označen D. Vektor s těmito daty lze označit jako $\mathbf{d} = (d_1, \ldots, d_D)$. Pro jedna vygenerovaná vstupní data je změřen jeden průběh šifrování. Ke každému změřenému průběhu pak musí být zároveň tato vstupní data přiřazena a zaznamenána, protože se také používají v prvním kroku pro stanovení mezivýsledku. Výsledkem měření je tedy matice proudových průběhů **T** o velikosti DxT, kde D je počet průběhů a T je počet zaznamenaných bodů na jeden průběh. [8]

Důležitým aspektem měření proudových průběhů je doplnit synchronizaci tak, aby bylo zajištěno, že se pro každá vstupní data zaznamená stejná část průběhu [8].

Třetí krok: Výpočet předpokládaných výsledků
Ve třetím kroku se provádí výpočet předpokládaných výsledků pro každou možnost vybrané části klíče k. To lze zapsat jako vektor $\mathbf{k} = (k_1, \ldots, k_K)$, kde K představuje počet všech možností části klíče k. Útočník může v tomto okamžiku z vektoru dat **d** a z vektoru předpokládaných klíčů **k** vypočítat předpokládané výsledky $f(d, k)$ pro všechny průběhy šifrování D a všechny předpoklady klíče K. Výsledkem tohoto výpočtu je matice **V** o velikosti DxK. Výpočet matice **V** je znázorněn rovnicí:

$$v_{i,j} = f(d_i, k_j);\ i = 1, \ldots, D;\ j = 1, \ldots, K. \tag{2.9}$$

Matice **V** nyní proto obsahuje jeden sloupec hodnot, které se shodují s hodnotami vypočítanými kryptografickým modulem během D šifrování. Sloupec j matice **V** obsahuje výsledky vypočítané z předpokladu klíče k_j. Jelikož vektor **k** obsahuje všechny možnosti části klíče k, obsahuje také jednu část, která odpovídá části klíče použitého v kryptografickém modulu. Tato část je označena k_{ck}. DPA má za úkol

najít, který sloupec v matici **V** odpovídá procesu šifrování v kryptografickém modulu. Jakmile bude tento sloupec matice **V** odhalen, je nalezena i část klíče k_{ck}. [8]

Čtvrtý krok: Mapování výsledků do hodnot proudové spotřeby
V dalším kroku DPA se mapují předpokládané výsledky **V** do matice předpokládaných hodnot proudové spotřeby **H**. K tomuto účelu se využívají techniky simulací (například Hammingova váha nebo Hammingova vzdálenost). Využitím těchto simulačních technik je proudová spotřeba kryptografického modulu pro každý předpokládaný výsledek $v_{i,j}$ transformována do předpokládané hodnoty proudové spotřeby $h_{i,j}$. [8]

Přesnost výsledků tohoto kroku závisí na útočníkově znalosti analyzovaného kryptografickém modulu. Čím lépe simulace charakterizuje dané zařízení, tím efektivnější bude DPA útok [8].

Pátý krok: Srovnávání předpokládaných hodnot proudové spotřeby se změřenými proudovými průběhy
V posledním kroku je prováděno srovnávání každého sloupce h_i matice předpokládané proudové spotřeby **H** s každým sloupcem t_j matice s naměřenými proudovými průběhy **T**. Výsledkem tohoto srovnávání je matice **R** o velikosti KxT, ve které pole $r_{i,j}$ obsahuje výsledek srovnání sloupce h_i a t_j. Srovnávání předpokládaných hodnot proudové spotřeby se změřenými proudovými průběhy je založeno na dále popsaném algoritmu.

Hodnota $r_{i,j}$ je tím vyšší, čím větší je shoda sloupců h_i a t_j. Součástí algoritmu je také stanovení mezivýsledku z prvního kroku DPA útoku. Proto kryptografický modul musí počítat střední hodnoty v_{ck} v několika různých vykonáváních algoritmu. V důsledku toho také určité části zaznamenaných průběhů závisí na těchto středních hodnotách. Tyto pozice proudových průběhů se označují jako ct. Například sloupec t_{ct} obsahuje hodnoty proudové spotřeby, která je závislí na hodnotých v_{ck}.

Z hodnot v_{ck} útočník určuje hodnoty h_{ck}. Z tohoto důvodu sloupce h_{ck} a t_{ct} spolu souvisí. Tyto 2 sloupce určují nejvyšší hodnotu v matici **R**, označovanou jako $r_{ck,ct}$. Všechny ostatní hodnoty v matici **R** nejsou tak vysoké, protože příslušné sloupce matic **H** a **T** spolu nesouvisí. Útočník nyní může z hodnoty $r_{ck,ct}$ určit správný klíč. [8]

2.2.1 Útok založený na korelačním koeficientu

Nejčastější způsob určení lineární závislosti dat je výpočet korelačního koeficientu. Jedná se o nejvyužívanější metodu vykonávání DPA útoku [8].

Při realizace DPA útoku se korelační koeficient využívá k určení lineární závislosti sloupců h_i a t_j, kde $i = 1, \ldots, K$ a $j = 1, \ldots, T$. Probíhá odhad každé hodnoty $r_{i,j}$ zakládající se na prvcích D sloupců h_i a t_j. To lze zapsat pomocí rovnice 2.10. V této rovnici prvky $\overline{h}_i$ a $\overline{t}_j$ vyjadřují hodnoty řádků h_i a t_j.

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h}_i) \cdot (t_{d,j} - \overline{t}_j)}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \overline{h}_i)^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \overline{t}_j)^2}} \tag{2.10}$$

2.2.2 Další statistické metody používané v DPA

V této části jsou pro přehled stručně popsány další možné metody vykonávání DPA útoků. Mezi tyto metody patří stanovení rozdílu středních hodnot, stanovení vzdálenost středních hodnot a použití šablon.

Rozdíl středních hodnot

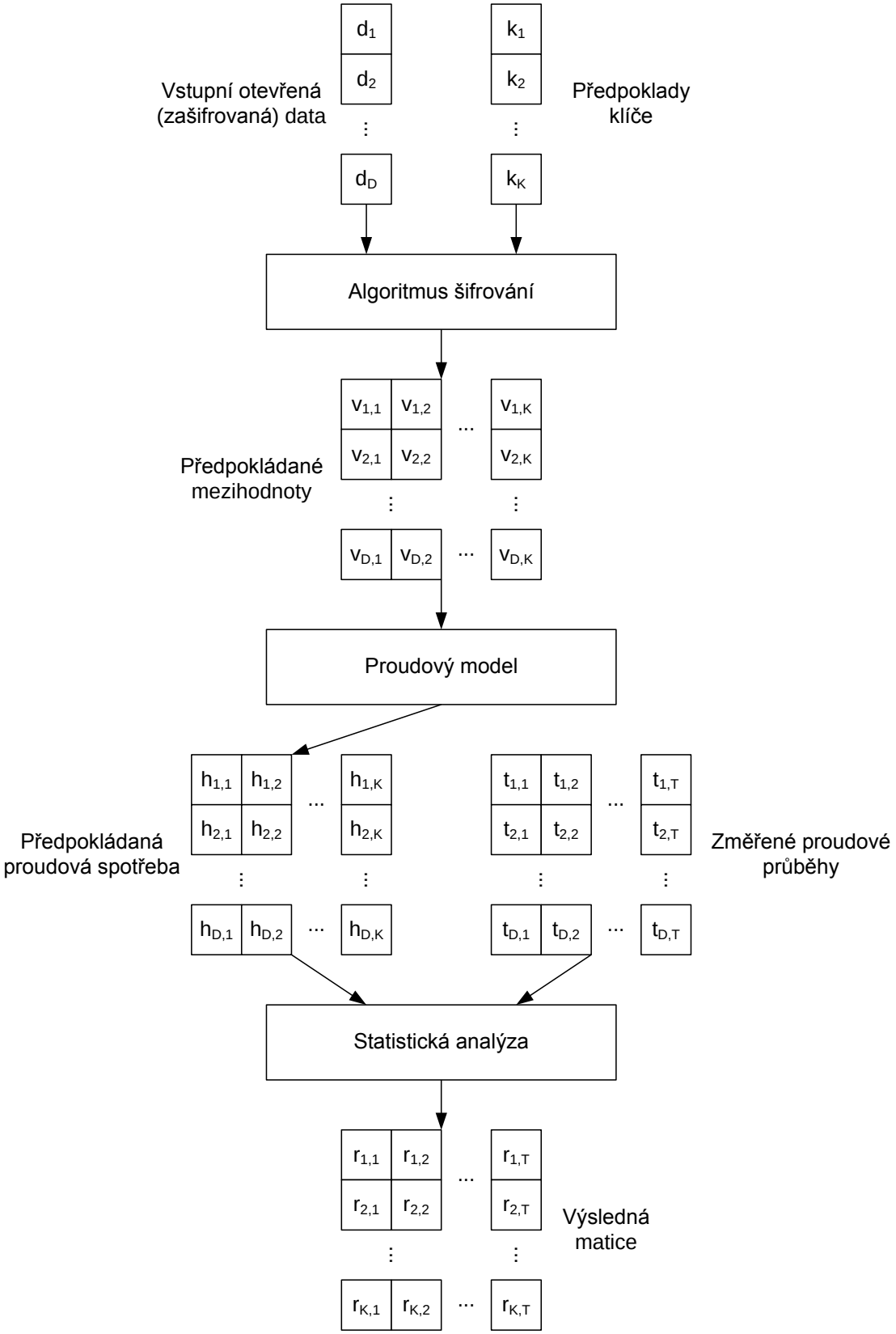
Statistická metoda rozdílu středních hodnot je založena na srovnávání dvou skupin naměřených hodnot s výpočtem rozdílu středních hodnot těchto dvou skupin. [8]

Vzdálenost středních hodnot

Metoda založená na vzdálenosti středních hodnot vychází z metody rozdílu středních hodnot a vylepšuje ji. Vylepšení spočívá v tom, že tato metoda bere v úvahu i směrodatné odchylky. Využívá testu k porovnání středních hodnot dvou různých rozdělení. [8]

Použití šablon

Při útoku DPA s využitím šablon využívá útočník modely, které popisují proudovou spotřebu kryptografického modulu, na který je prováděn útok. Vytváří tzv. šablony. Princip je prakticky stejný jako při SPA, pouze šablony jsou rozšířeny. [8]

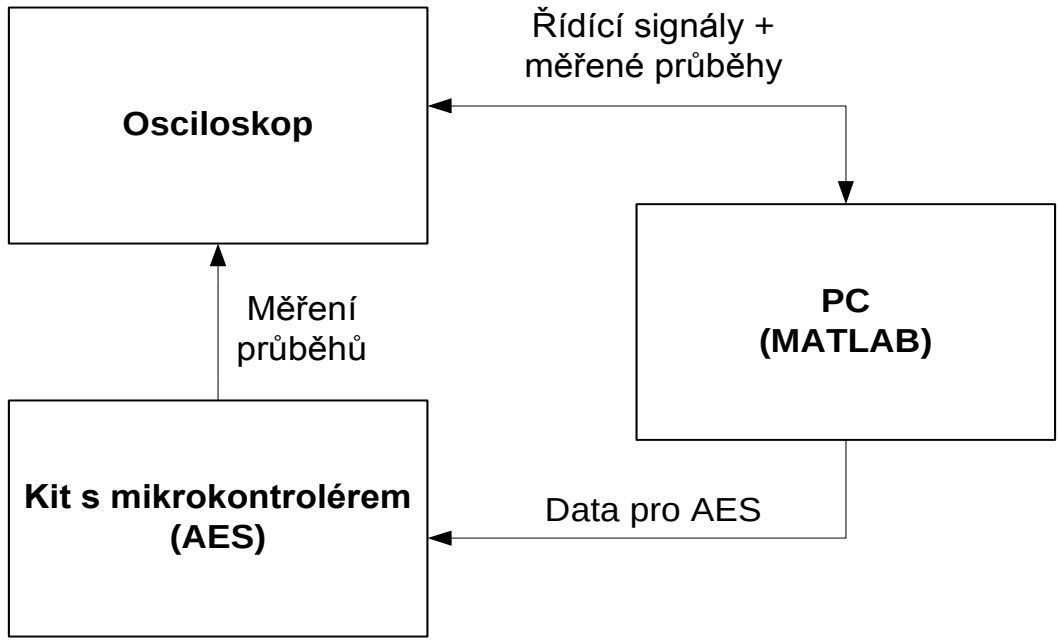


Obr. 2.2: Blokové schéma DPA útoku, převzato z [8].

3 EXPERIMENTÁLNÍ PRACOVIŠTĚ

Před vlastní realizací diferenciální proudové analýzy bylo nejprve nutné dle zadání realizovat kompletní experimentální pracoviště, které bude schopno automaticky zaznamenávat průběhy z potřebných postranních kanálů. Realizace experimentálního pracoviště spočívala v implementaci šifrovacího algoritmu AES na vhodný mikrokontrolér, propojení osciloskopu s počítačem a na závěr bylo za úkol měření průběhů automatizovat. Experimentální pracoviště je v této kapitole podrobně popsáno.

Experimentální pracoviště se skládá celkem ze tří bloků, a to z počítače s nainstalovaným vývojovým prostředím MATLAB, vývojové sady ATMEL STK500 s mikrokontrolérem řady AVR, do kterého je implementovaná šifrovací algoritmus AES, a osciloskopu Tektronix MSO4034B. MATLAB, popřípadě počítač, je dále rozšířen o další ovladače. Na obrázku č. 3.1 je znázorněno blokové schéma experimentálního pracoviště.



Obr. 3.1: Blokové schéma experimentálního pracoviště.

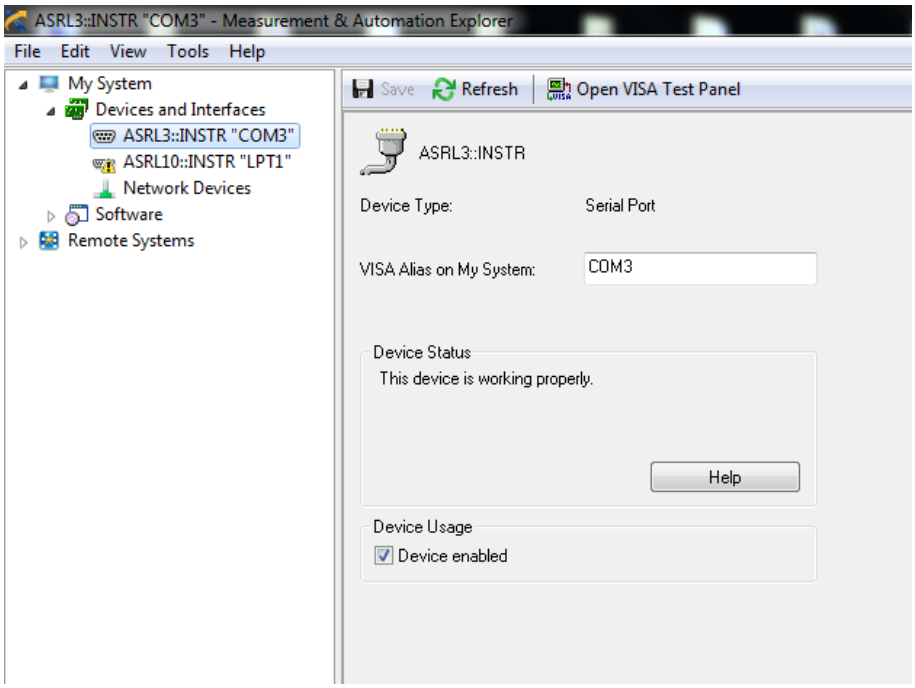
Experimentální pracoviště je řízeno z prostředí MATLAB. V tomto textu je dále počítač s vývojovým prostředím MATLAB uváděn jako řídící blok. Pro řízení je napsán skript, který má za úkol nejprve navázat komunikaci s osciloskopem přes USB (*Universal Serial Bus*) port, ovládat funkce osciloskopu a zaznamenávat průběhy. Zároveň pomocí tohoto skriptu řídící blok komunikuje přes sériovou linku (RS-232)

s mikrokontrolérem a posílá mu všechna data potřebná pro šifrování AES (tajný klíč, blok otevřeného textu určeného k zašifrování a potvrzující zprávy).

Osciloskop je spojen s počítačem pomocí USB kabelu. Typ tohoto kabelu nese označení USB A-B. Pro tento typ propojení je určena zdířka na zadní straně osciloskopu, která je označena jako *Device*. Na osciloskopu je poté nutné provézt příslušné nastavení propojení s počítačem přes rozhraní USB. Toto nastavení je podrobně popsáno v uživatelském manuálu osciloskopu [11]. Vlastní komunikace počítače s osciloskopem pak probíhá pomocí prostředí MATLAB, který je doplněn toolboxem Instrument Control Toolbox. Jedná se o toolbox určený právě pro ovládání osciloskopů. Instrument Control Toolbox je dále doplněn o příslušný ovladač osciloskopu, v tomto případě ovladač osciloskopu Tektronix MSO4034B (soubor MSO4034.mdd). Tento ovladač je nutné zkopírovat do příslušné složky s ovladači. Standardně má tato složka tuto cestu: *Složka s nainstalovanými programy* → *MATLAB* → *toolbox* → *instrument* → *instrument* → *drivers*. Ovladač je stejně jako velké množství dalších ovladačů pro různé typy osciloskopů, k dispozici na webových stránkách MathWorks [14]. Pro potřeby této práce však musel být ovladač osciloskopu mírně modifikován. Modifikace ovladače spočívá však pouze ve zvýšení počtu zaznamenaných bodů na jeden průběh.

Pro samotné navázání komunikace s osciloskopem je nutné vytvořit v prostředí MATLAB virtuální rozhraní USB, do kterého je osciloskop připojen. Jedná se v podstatě o přiřazení adresy k příslušnému USB portu. Tuto adresu poté MATLAB využívá při obousměrné komunikaci s osciloskopem. Pro vytvoření tohoto virtuálního rozhraní slouží ovladač typu VISA (*Virtual Instrument Software Architecture*), který se nazývá Measurement & Automation Explorer od společnosti National Instruments (zkráceně NI MAX). Tento ovladač se musí nainstalovat do počítače a MATLAB, popřípadě jeho rozšíření Instrument Control Toolbox, si jej načte a začne ho využívat pro komunikaci s osciloskopem. Virtuální adresu použitého USB portu lze získat po otevření Measurement & Automation Explorer, rozkliknutím *My System* → *Devices and Interfaces*, kde jsou vypsané všechny používané porty (obrázek č. 3.2). Mezi nimi se nachází i USB port, na který je osciloskop Tektronix MSO4034B připojen i s jeho virtuální adresou.

Spojení počítače s vývojovou sadou ATMEL STK500 (s mikrokontrolérem) je provedeno tak, že je k počítači do USB portu připojen převodník USB → RS-232 a k převodníku je sériovým kabelem připojen vývojová sada. Připojení převodníku USB → RS-232 k počítači vytvoří v počítači virtuální sériový port, kterému přiřadí adresu. Tuto adresu lze získat opět z Measurement & Automation Explorer, rozkliknutím *My System* → *Devices and Interfaces* (obrázek č. 3.2). Pomocí této adresy sériového portu je potom možné navazovat spojení přes sériovou linku a komunikovat s mikrokontrolérem na kitu.



Obr. 3.2: Zjištění adresy rozhraní z Measurement & Automation Explorer.

Snímání příslušných průběhů pak provádí osciloskop pomocí příslušných sond. V této práci se jedná o 3 typy měřených průběhů. Snímá se elektromagnetické vyzařování u pinu mikrokontroléru, na který je připojeno napájecí napětí. Toto snímání zajišťuje elektromagnetická sonda. Dále probíhá měření proudu přiváděného do pinu, na který je připojeno napájecí napětí. Toto měření zajišťuje proudová sonda [13]. Nakonec se snímá synchronizační signál na příslušném portu mikrokontroléru pomocí pasivní napěťové sondy.

3.1 Popis protokolu šifrování a komunikace mikrokontroléru

V této části je popsán protokol implementovaného šifrování. Protože je diferenciální proudová analýza je zaměřena na šifrovací algoritmus AES, je nutné právě AES na vybraný mikrokontrolér implementovat. Implementaci popřípadě dále rozšířit do více mikrokontrolérů.

Implementace šifrovacího algoritmu AES na mikrokontrolér je součástí komplexního protokolu, který kromě šifrování zajišťuje také komunikaci mikrokontroléru s okolními objekty pracoviště, proto je do implementace také dále nutné zapracovat i ovládání některých periferií. Jedná především o ovládání USART (*Universal Syn-*

chronous and Asynchronous Serial Receiver and Transmitter) pro komunikaci s řídícím blokem (MATLAB) a ovládání LED (*Light-Emitting Diode*), které se využijí pro signalizaci. LED signalizují celkem dva stavy implementovaného šifrování. První stav je stav, kdy mikrokontrolér čeká na všechna data (tajný klíč a blok otevřeného textu, který je určen k zašifrování), která potřebuje k tomu, aby mohl zahájit šifrování. Druhý stav odpovídá vlastnímu šifrování bloku otevřeného textu šifrovacím klíčem. Pro kontrolní účely je dále také do implementace doplněna obsluha tlačítek.

Před vlastním programováním je však nutné na vývojové sadě ATMEL STK500 propojit příslušné konektory. Tímto propojením vývojovou sadu připravit na programování mikrokontroléru a dále na to, aby mikrokontrolér mohl ovládat periferie vývojové sadě. Jedná se následující propojení:

- propojení soketu mikrokontroléru s ISP (*In-System Programming*) pro programování,
- propojení příslušného portu mikrokontroléru s LED,
- propojení příslušného portu s tlačítky,
- propojení rozhraní RS-232 s piny přijímání a vysílání USART mikrokontroléru.

Všechna potřebná propojení konektorů na kitu ATMEL STK500 jsou popsána v jeho uživatelském manuálu [12]. Celá implementace lze pak rozdělit na 2 hlavní části. První část lze označit jako inicializační a druhá část je vlastní šifrování AES.

3.1.1 Popis inicializační části implementace

V inicializační části je provedena inicializace LED, tlačítek, USART a přerušení. Použitá skupina mikrokontrolérů AVR má tu výhodu, že prakticky pro všechny mikrokontroléry této skupiny lze použít totožná nastavení. Toho je využito i v inicializační části.

Pro inicializaci LED a tlačítek stačí pouze v kódu označit příslušné porty jako výstupní, respektive vstupní. LED se ovládají tedy pomocí pinů na portech C a D, které jsou označeny v příslušném registru jako výstupní. Tlačítka zase vyvolávají události, které se vyhodnocují na pinech portu D. Jelikož jsou využita pouze dvě tlačítka, jsou tedy v příslušném registru portu D označeny jako vstupní pouze piny PD2 a PD3.

Inicializace USART je obsáhlejší. Kromě povolení a nastavení přijímání a vysílání dat na příslušný port mikrokontroléru je také nutné nastavit některé parametry komunikace. Přijímání a vysílání dat je nastaveno na port D. Na tomto portu se nachází piny určeny k tomuto účelu. Jedná se o piny označeny jako PD0 a PD1. Dále je také nutné nastavit parametry jako baudrate, velikost přenášených dat, parita, režim komunikace a povolení přerušení od USART. Nastavení těchto parametrů je uvedeno v tabulce č. 3.1.

Tab. 3.1: Nastavené parametry USART.

Parametr	**Nastaveno**
Baudrate	9600 bps
Velikost dat	8 bitů
Parita	není
Režim	simplex
Přerušení	od příjmu

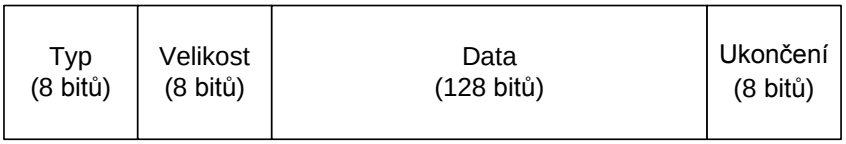
Pro komunikaci s řídícím blokem experimentálního pracoviště je také v inicializační části provedena definice datových jednotek, které přenášejí do mikrokontroléru data potřebná pro chod implementovaného šifrování. Celkem jsou definovány 3 datové jednotky, které mikrokontrolér dokáže rozpoznat a zpracovat. Všechny tři datová jednotky jsou vysílány z řídícího bloku. První datová jednotka je určena k přenosu klíče pro šifru AES, druhá jednotka je určena pro přenos dat určených k zašifrování a třetí jednotka je využívána řídícím blokem pro potvrzování stavů (například potvrzení stavu, že průběhy byly zaznamenány). Všechny tři struktury jsou graficky znázorněny a popsány v následujícím textu.

Datová jednotka pro přenos klíče

Datová jednotka pro přenos tajného klíče se skládá ze 4 polí. První pole, označené jako Typ, vyjadřuje typ datové jednotky. Má velikost 1 bajt a v tomto případě má hodnotu 0, která značí přenos klíče. Pole Velikost vyjadřuje velikost vlastních dat v datové jednotce. Data (klíč) v datové jednotce pro přenos klíče mají pevně danou hodnotu 32. Tato hodnota vyjadřuje velikost klíče v bajtech. Poté následuje pole Klíč o velikosti 32 bajtů, které obsahuje klíč, kterým se využije pro šifrování. Poslední buňka Ukončení má za úkol signalizovat konec datové jednotky, obsahuje hodnotu 255. Struktura této datové jednotky je zobrazena na obrázku č. 3.3.

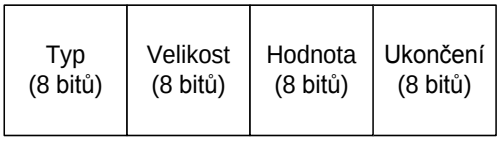
Obr. 3.3: Struktura datové jednotky pro přenos klíče.

Datová jednotka pro přenos dat
U této datové jednotky mají první dvě pole stejný význam jako u datová jednotka pro přenos klíče. Avšak pole Typ má hodnotu 1, což signalizuje datovou jednotku, která obsahuje data určená z zašifrování. Pole Velikost má vždy hodnotu 16, protože datová jednotka obsahuje 16ti bajtová data. Následuje pole Data, kde jsou obsažena vlastní data určená k šifrování. Datovou jednotku opět uzavírá pole Ukončení s hodnotou 255. Struktura této datové jednotky je zobrazena na obrázku č. 3.4.



Obr. 3.4: Struktura datové jednotky pro přenos dat.

Datová jednotka pro přenos potvrzení
Zde mají první dvě buňky stejný význam, jako u předchozích datových jednotek. Pole Typ má vždy hodnotu 2, která označuje potvrzovací datovou jednotku. Pole Velikost má hodnotu 1, která oznamuje, že následující pole Hodnota má velikost 1 bajt. Pole hodnota má vždy hodnotu 255, která nemá žádný zvláštní význam, ale patří k signalizaci typu datové jednotky. Datovou jednotku pro potvrzování opět uzavírá pole Ukončení s hodnotou 255.



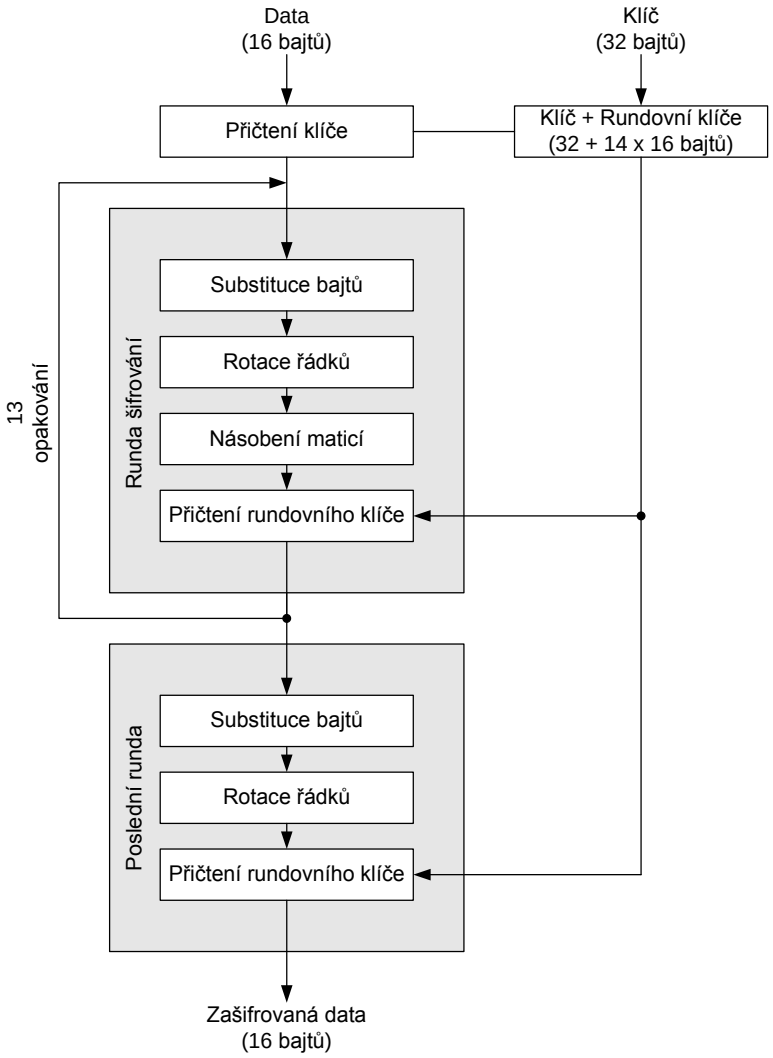
Obr. 3.5: Struktura datové jednotky pro přenos potvrzení.

Tuto datovou jednotku lze označovat jako ACK (*Acknowledgement*). Mikrokontrolér ji přijímá od řídící buňky. Řídící buňka mu touto zprávou signalizuje, že může přerušit šifrování (signalizuje především stav, že proběhla všechna potřebná měření). Struktura této datové jednotky je zobrazena na obrázku č. 3.5.

3.1.2 Popis implementace AES

Pro implementaci vlastního šifrování je vybrána knihovna AVR-Crypto-Lib [4]. Tato knihovna obsahuje sadu mnoha různých kryptografických algoritmů pro skupinu 8mi

bitových mikrokontrolérů AVR. Z této knihovny byla pro implementaci vybrána pouze její část, konkrétně proces šifrování AES s délkou šifrovacího klíče 256 bitů, která je napsána v programovacím jazyce C.



Obr. 3.6: Blokové schéma implementovaného šifrování.

Celá implementace šifrování AES se skládá ze 2 funkcí. První funkce je definována jako:

```
aes256_init(key, &ctx);
```

Tato funkce se vlastně může ještě zařadit do inicializační části. Zajišťuje výpočet všech rundovních klíčů, tedy klíčů, které se používají v jednotlivých rundách šifry AES. Parametr *key* vyjadřuje 256ti bitový klíč a parametr *ctx* je struktura (matice), kam jsou rundovní klíče uloženy, včetně vlastního šifrovacího klíče, ze kterého jsou vypočteny. Doba trvání této funkce je 6 161 hodinových cyklů mikrokontroléru [4].

Druhá funkce je definovaná jako:

```
aes256_enc(data, &ctx);
```

Tato funkce zajišťuje vlastní šifrování dat. V této funkci parametr *data* vyjadřuje proměnnou s daty, která se šifrují. Parametr *ctx* zde již obsahuje všechny potřebné klíče uspořádané v matici, které tam byly vloženy vykonáním funkce aes256_init(key, &ctx). Implementovaná šifra AES je podrobně popsána v [7] a lze ji vyjádřit blokovým schématem na obrázku č. 3.6. V dokumentaci ke knihovně AVR-Crypto-Lib je uvedeno, že vybrané šifrování (tedy AES s délkou klíče 256 bitů) je vykonáno v 30 174 hodinových cyklech mikrokontroléru [4].

Celý proces lze zjednodušeně popsat, tak že nejprve se vypočítají rundovní klíče. Následně se k šifrovaným datům přičte šifrovací klíč, poté následuje 14 rund šifrování. V každé rundě se postupně vykovávají operace substituce bajtů, rotace řádků, násobení maticí a přičtení rundovního klíče. V poslední 14. rundě se vynechává operace kombinování sloupců. Výsledkem je zašifrovaný text stejné velikosti jako šifrovaná data. Počet rund odpovídá délce šifrovacího klíče, tedy pro délku klíče 256 bitů se data šifrují tedy ve 14ti rundách.

Synchronizace na měřenou část

Aby bylo možné na osciloskopu přesně rozlišit vybranou část šifrování, je do kódu zapracovaná i synchronizace na tuto vybranou část.

Tato synchronizace spočívá nejprve v rozsvícení příslušných LED na vývojové sadě s mikrokontrolérem na začátku měřené části a zhasnutím na konci měřené části. Příslušné LED nesou na kitu označení LED0 a LED1. Tyto LED mikrokontroléru rozsvěcuje a zhasíná vysíláním signálu na piny PB0 a PB1. Jsou použity dvě LED, protože na jeden z jejich pinů (PB0 nebo PB1) lze připojit pasivní napěťovou sondu a na jednom kanálu osciloskopu obdélníkový průběh signálu zobrazit. Druhá LED je pouze signalizační. Vlastní průběh šifrování je zobrazen na jiném kanálu osciloskopu. Zobrazením obou kanálů lze pak najít tu část šifrování, která je určená k analýze.

Pro realizaci diferenciální proudové analýzy je zasynchronizovaná ta část šifrování, ve které se vykonává operace přičtení klíče a operace záměny bajtů v první rundě. Pro analýzu se tato část (především operace přičtení klíče) dá považovat za nejdůležitější, protože se zde pracuje přímo s tajným klíčem.

Ověření správnosti šifrování

Kód je dále doplněn o testování toho, zda implementované šifrování probíhá v pořádku. K tomuto účelu jsou využita dvě tlačítka, která jsou na kitu označena jako SW0 a SW1.

Postup ověření je popsán v následujícím textu. Nejprve se do mikroprocesoru pošle šifrovací klíč. Následuje vyslání bloku otevřených dat a mikrokontrolér začne tato data neustále šifrovat ve smyčce. Tato smyčka se přeruší až následným tlačítkem SW1. Mikrokontrolér ještě dokončí právě rozpracovanou smyčku a následně přestane šifrovat. V tomto okamžiku se nachází ve stavu, kdy čeká na další data a zároveň již má k dispozici pro odeslání zašifrovaná data. Stiskem tlačítka SW0 mikrokontrolér odešle tato zašifrovaná data do počítače, kde je lze zobrazit v prostředí MATLAB, kde je připravena funkce pro příjem těchto dat. Vyslaná data z mikrokontroléru mají 16 bajtů. Mikrokontrolér je odesílá po 1 bajtu jako datový typ uint8.

V příloze A je uveden sled příkazů zadávaným v prostředí MATLAB, podle kterých lze kontrolu správnosti šifrování realizovat. Zároveň jsou v této příloze uvedeny kontrolní příklady pro ověření správnosti implementované šifry AES.

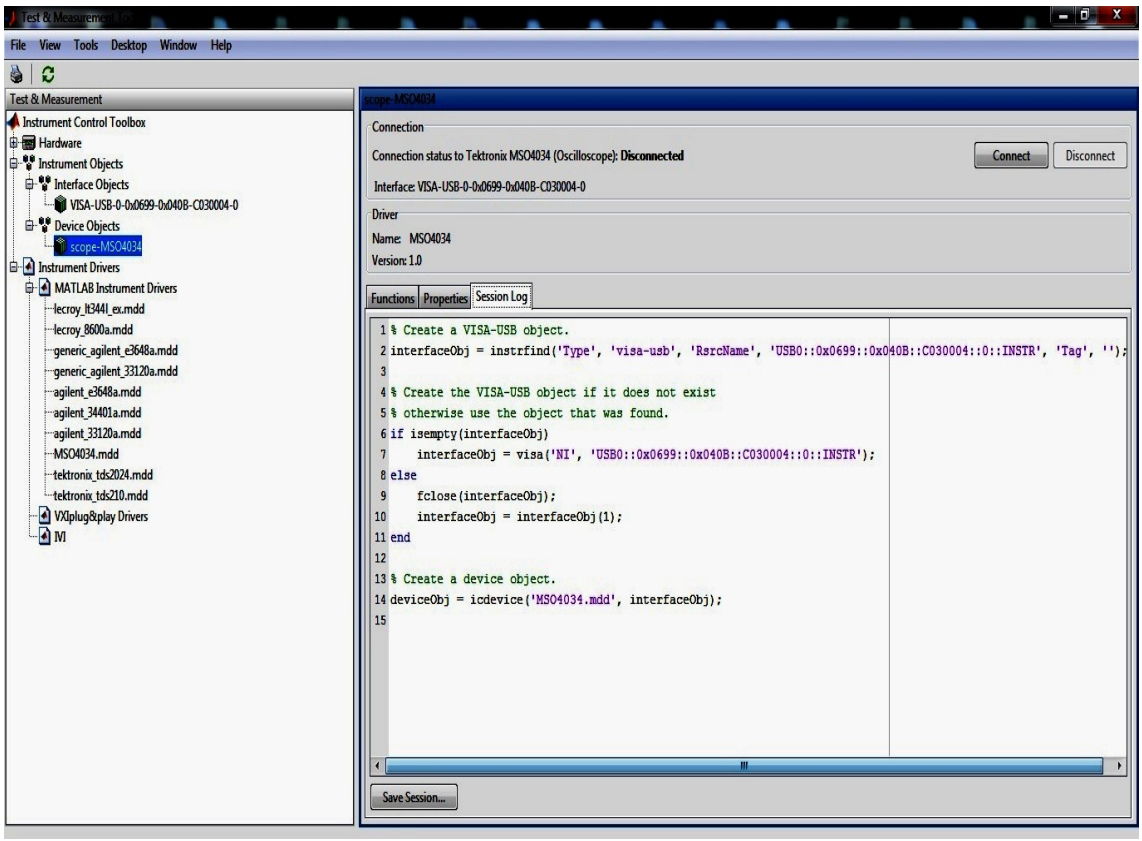
3.2 Skript pro zaznamenávání průběhů

Na základě zadání je pro zaznamenávání měřených průběhů vytvořen skript, který řídí celé pracoviště. Pomocí tohoto skriptu je možné automaticky zaznamenat přímo do matice v prostředí MATLAB prakticky libovolný počet průběhů. Práce s programem (Instrument Control Toolbox), ze kterého skript vychází, je stejně jako vlastní skript popsán v následujícím textu.

Práce s Instrument Control Toolbox
Instrument Control Toolbox lze v prostředí MATLAB spustit příkazem *tmtool*. Po spuštění tohoto příkazu se otevře grafické uživatelské rozhraní Test & Measurement Tool, ve kterém lze vytvořit spojení s osciloskopem následujícím způsobem: v levém menu postupným volením *Instrument Drivers* → *MATLAB Instrument Drivers*, dále kliknutím pravým tlačítkem myši na *MSO4034.mdd* a otevřením možnosti *Create Device Object Using Driver...* se otevře nové okno s názvem *Create Device Object*. V tomto okně se pokračuje kliknutím na tlačítko *Create*, čímž se otevře další okno *Device Object Creation*. V tomto okně v záložce *Interface Object Type* se vybere možnost *VISA*, v záložce *Vendor* se vybere *ni* a nakonec v záložce *Resource Name* se vybere adresa USB portu, do kterého je osciloskop připojen. Pokud by tato adresa nebyla nalezena, je nutné ji do tohoto pole zkopírovat z Measurement & Automation Explorer. Následným kliknutím na tlačítko *OK* se vrátí zpět do okna *Create Device Object*, kde se vše opět potvrdí kiknemutím na tlačítko *OK*. Vykonáním tohoto postupu je provedena inicializace připojení k osciloskopu.

V levém menu grafického uživatelského rozhraní Test & Measurement Tool se nyní zvolí *Instrument Objects* → *Device Objects* a kliknutím levým tlačítkem myši

na *scope-MSO4034*. Tímto se zobrazí rozhraní, ve kterém je nyní možné osciloskop plně ovládat a zaznamenávat průběhy pomocí příslušných prvků s příkazy. Nachází se zde také záložka *Session Log*, kde jsou zaznamenány všechny výše popsané úkony pomocí příkazů, které MATLAB zpracovává. Právě těchto příkazů využívá skript pro automatické zaznamenávání průběhů. Uživatelské rozhraní Test & Measurement Tool je zobrazeno na obrázku č. 3.7.



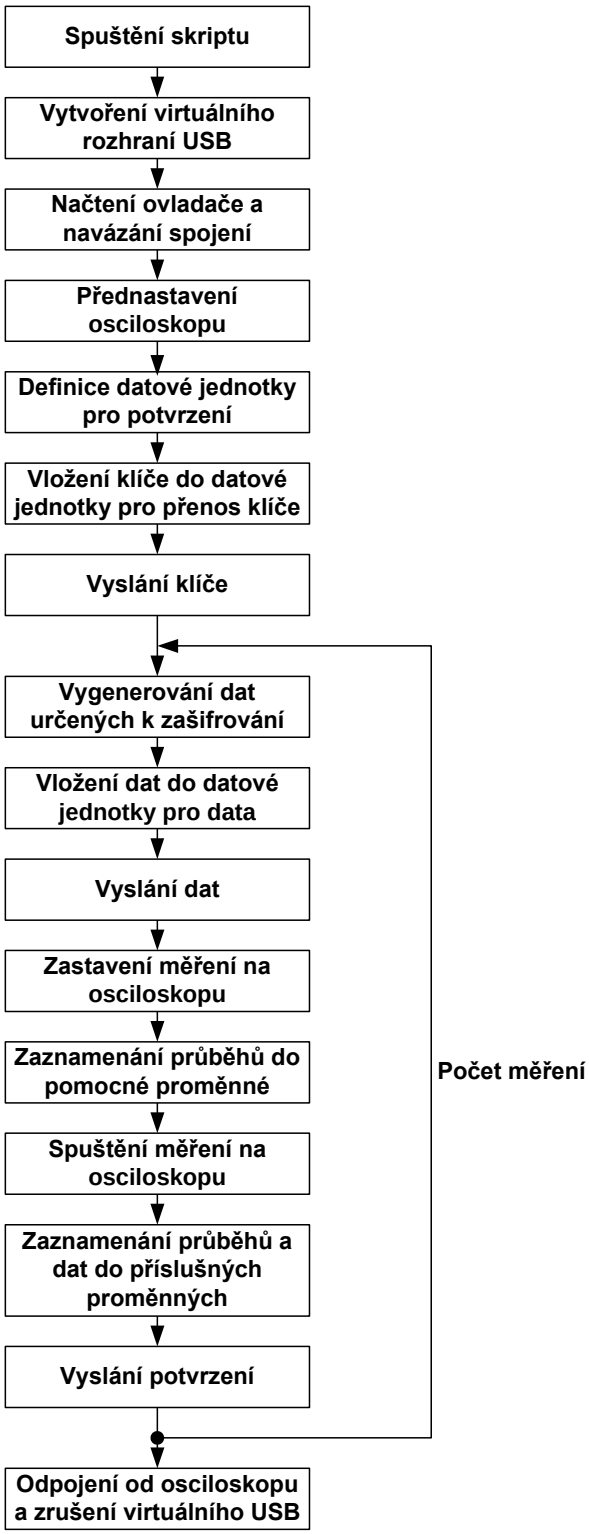
Obr. 3.7: Uživatelské rozhraní Test & Measurement Tool.

Popis skriptu

Skript pro automatické zaznamenávání průběhů lze v prostředí MATLAB spustit zadáním příkazu:

```
save_waveforms(usb, keys, n_measures);
```

Význam parametrů v příkazu je následující: parametr *usb* vyjadřuje virtuální adresu USB portu, do kterého je osciloskop připojen. Tato adresa lze získat z VISA ovladače Measurement & Automation Explorer. Parametr *keys* vyjadřuje proměnnou (matici), do které se zadává šifrovací klíč pro šifrovací algoritmus AES o velikosti



Obr. 3.8: Blokové schéma skriptu pro automatické ukládání průběhů.

32 bajtů. Poslední proměnná *n_measures* je číslo, které vyjadřuje počet měření. Příklad příkazu zadaného zadaného v prostředí MATLAB může vypadat takto:

```
key = [ 208 231  33 233 162  25  72 140 245 247
         41 248 245 124 205  37 108 234 203 245
        168  10 217 239 174 194 190 101 168  44
        181   9 ];

save_waveforms('USB0::0x0699::0x040B::C030004::INSTR',
                    key, 1000);
```

V tomto příkladu je vložen klíč do matice *usb*. Tato matice má velikost $32x1$ polí. Je zde vidět tvar adresy USB portu, tvar klíče a poslední parametr 1000 vyjadřuje, že bude naměřeno 1000 průběhů. Příkazový řádek bude během vykonávání skriptu vypisovat číslo měření, které právě provádí.

Po spuštění skriptu jsou tedy nejprve provedeny všechny potřebné úkony pro navázání komunikace prostředí MATLAB s osciloskopem. K tomuto se právě využívá výše popsaný Instrument Control Toolbox. Mezi tyto vykonané úkony patří vytvoření virtuálního rozhraní USB, načtení ovladače osciloskopu a otevření vlastní komunikace. Na obrázku č. 3.7 jsou ve zobrazené části *Session Log* naznačeny příkazy, které tyto úkony vykonávají.

Po navázání spojení s osciloskopem začíná inicializační část skriptu. V inicializační části je nejprve provedeno základní přednastavení osciloskopu, konkrétně počet bodů zaznamenaných na jeden průběh. Pro potřeby diferenciální proudové analýzy je zvoleno 100 tisíc bodů. Tento počet bodů by měl zachytit průběh s dostačující podrobností a zároveň by měl zatěžovat buffer vývojového prostředí MATLAB v rozumné míře tak, aby nedošlo ke zpomalení chodu měření. Další nastavení osciloskopu (především rozlišení napěťové osy, rozlišení a posunutí časové osy, zapnutí potřebných kanálů a nastavení triggeru) je nutné provézt manuálně na osciloskopu. Následuje definice datové jednotky pro potvrzování, která se bude později posílat mikrokontroléru. Dále je také odeslán klíč do mikrokontroléru, který je ještě před vlastním odesláním vložen do příslušné datové jednotky.

Po inicializační části následuje část skriptu, která zajišťuje vlastní měření. Tato část lze označit jako měřicí. Nejprve jsou náhodně s normálním rozdělením vygenerována data, který jsou určená k šifrování. Následně jsou tato data vložena do příslušné datové jednotky a odeslána do mikrokontroléru. V tomto okamžiku mikrokontrolér začíná šifrovat a může začít vlastní proces zaznamenání průběhů. Nejprve je osciloskopu odeslán příkaz STOP, který zastaví měření a skript může zaznamenat průběhy. Pro jedna odeslaná data jsou zaznamenány do pomocné proměnné celkem dva průběhy, a to vlastní průběh šifrování z kanálu osciloskopu č. 1 a synchronizační

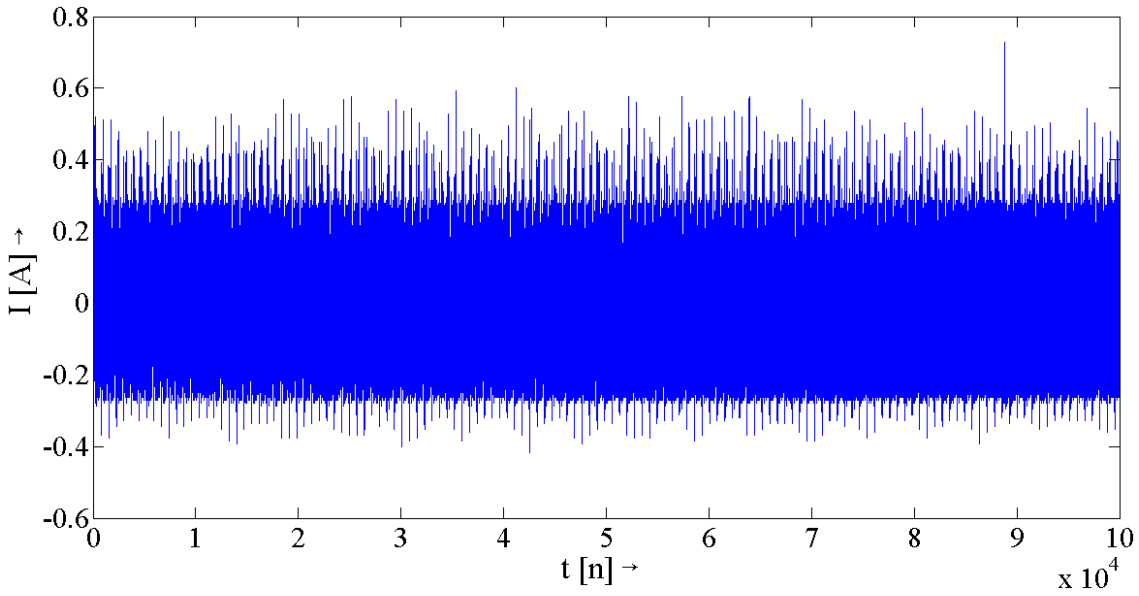
průběh z kanálu č. 2. Osciloskopu je dále odeslán příkaz START. V tomto okamžiku jsou zachycena všechna potřebná data právě měřeného průběhu. Tato data jsou rozdělena do 3 příslušných matic, které jsou poději výstupem skriptu. První matice obsahuje náhodně vygenerovaná 16ti bajtová data, druhá matice obsahuje 10 tisíc bodů průběhu šifrování a třetí matice obsahuje 10 tisíc bodů synchronizačního průběhu. Následuje je odeslání potvrzovací datové jednotky, pomocí které se oznámí mikrokontroléru, že potřebná data byla zaznamenána a mikrokontrolér přestává dočasně šifrovat do doby, kdy mu přijdou další data určená k zašifrování. Celá měřicí část se opakuje ve smyčce dokud není zaznamenán požadovaný počet měření, který je zadán do vstupujících parametrů skriptu.

Z důvodu omezené velikosti vnitřní paměti prostředí MATLAB, jsou všechny 3 matice obsahující data o jednotlivých průbězích uloženy po změření každých 200 průběhů. Například tedy pro celkem 400 změřených průběhů bude výstupem skriptu celkem 6 matic s označením: data200, data400, traces200, traces400 a syn200, syn400. Matice data200 je matice o velikosti 200x16 a obsahuje 16ti bajtová data s pořadím 1 až 200. V maticích traces200 (200x100 000 bodů) a syn200(200x100 000 bodů) jsou uloženy průběhy s pořadím 1 až 200m které odpovídají datům v matici data200. Matice s označením data400, traces400 a syn400 obsahují stejná data, nyní už pro průběhy s pořadím 201 až 400.

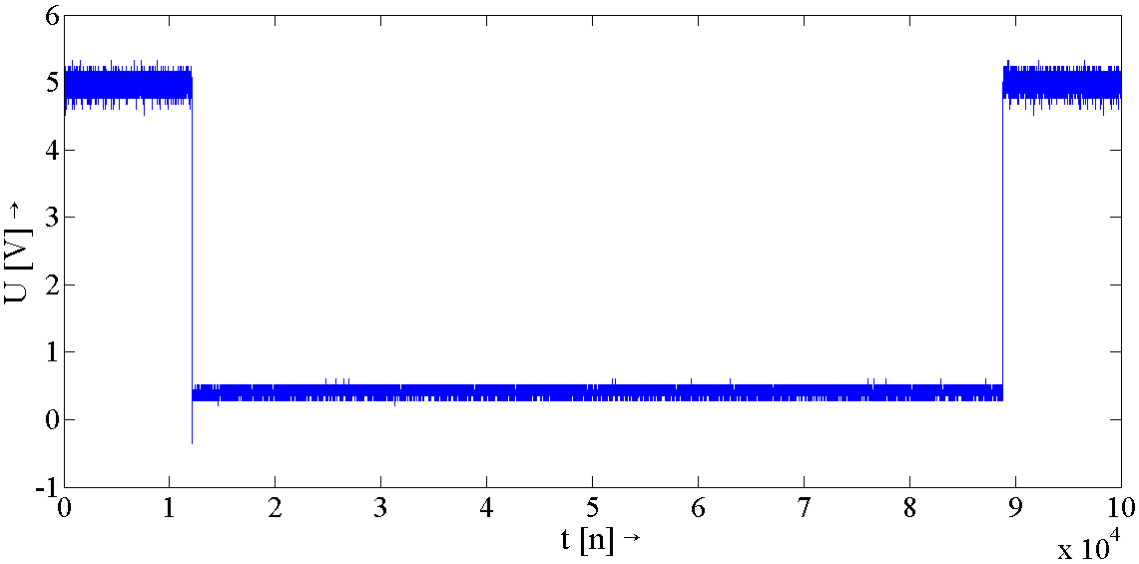
Po změření zadaného počtu průběhů následuje poslední část skriptu. V této části je provedeno ukončení spojení s osciloskopem a zrušení virtuálního rozhraní USB. Celá funkce skriptu je znázorněna blokovým schématem na obrázku č. 3.8. Jak již bylo zmíněno v popisu implementace AES, je synchronizace zavedena pouze na operaci přičtení klíče a operaci záměny bajtů, proto je i celý skript automatického ukládání průběhů přizpůsoben pouze těmto dvěma operacím.

Pro příklad výstupu ze skriptu je na obrázku č. 3.9 graficky znázorněn jeden náhodně vybraný průběh šifrování náhodných dat zaznamenaný skriptem pro automatické zaznamenávání průběhů a na dalším obrázku č. 3.10 v grafu jemu odpovídající průběh synchronizačního signálu.

Takto zaznamenané průběhy se před vlastní realizací diferenciální proudové analýzy ještě dále zpracovávají pomocí dalších skriptů napsaných v prostředí MATLAB. Tyto skripty jsou popsány v příloze B.



Obr. 3.9: Příklad změřeného proudového průběhu šifrování.



Obr. 3.10: Příklad synchronizačního signálu.

4 REALIZACE ANALÝZY MĚŘENÝCH PRŮBĚHŮ

V této kapitole je popsána kompletní analýza měřených průběhů. Nejprve je provedena počáteční celková, tzv. jednoduchá, analýza. V této jednoduché analýze se zkoumá vyzařování informace elektromagnetickým a proudovým postranním kanálem u vybraných mikrokontrolérů. Ve změřených průbězích šifrování se proto hledají určité charakteristické znaky, které potvrzují fakt, že měřené průběhy vykazují závislost na šifrování, které je mikrokontroléry vykonávané. Dále je zkoumáno i to, jakým způsobem jsou proudové průběhy šifrování ovlivňovány právě zpracovávanými daty.

Po jednoduché analýze následuje vlastní realizace diferenciální proudové analýzy. V této části analýzy je již prováděn vlastní útok na implementované šifrování AES. Cílem tohoto útoku je zjištění šifrovacího klíče ze změřených průběhů, který AES implementovaný do mikrokontroléru pro šifrování používá.

4.1 Ověření experimentálního pracoviště

Na základě konzultace s vedoucím práce není provedena jednoduchá proudová analýza za účelem zjištění tajného klíče šifry AES. Je provedena pouze analýza, jejímž cílem je zkoumání měřených průběhů. Pro tuto jednoduchou analýzu jsou provedena měření tzv. pomocných průběhů. Z těchto pomocných průběhů se tedy nezjišťuje hodnota tajného klíče, ale s jejich využitím je zkoumáno především to, u kterého mikrokontroléru elektromagnetický a proudový postranní kanál nejlépe vyzařují informace a zda jsou informace o implementované šifře AES postranními kanály vůbec vyzařovány. Jedná se především o vizuální vyšetřování změřených průběhů a vyšetřování toho, zda zařízení vyzařuje informaci o Hammingově váze.

Pro realizaci analýzy jsou k dispozici celkem 3 mikrokontroléry, které odpovídali všem potřebám pro realizaci experimentálního pracoviště i pro realizaci vlastní analýzy. Jedná se o mikrokontroléry Atmel Atmega8 [1], Atmega16 [2] a ATmega162 [3].

Mikrokontrolér Atmega162 však musel být později pro analýzu vyloučen, protože implementovaný šifrovací algoritmus AES neprošel kontrolou správnosti šifrování. Šifrování na tomto mikrokontroléru sice probíhalo správně, ale pouze pokud nebyl napájecí pin připojen k napájecímu napětí. Proč mikrokontrolér pracuje bez připojeného napájení se i po důkladném ověření správnosti připojení podle datového listu a také po ověření správnosti všech nastavení nepodařilo zjistit. Po připojení napájení šifrování už neprobíhalo správně. Zašifrovaný text přijatý z mikrokontroléru neodpovídal textu očekávanému. Ovšem bez připojení napájení není možné provézt

měření potřebná pro analýzu. Mikrokontrolér byl označen jako nevyhovující pro další realizace analýzy.

Analýza dále proto probíhá pouze na šifrování implementovaném do mikrokontrolérů ATmega8 a ATmega16. Zhodnocení vyzařování postranních kanálů těchto dvou mikrokontrolérů je provedeno dále v podkapitole 4.1.1.

Dále je také vhodné dodat, že kmitočet interního oscilátoru je u všech použitých mikrokontrolérů nastaven na 2 MHz. Byl zvolen nejnižší možný kmitočet, při kterém mikrokontrolér vykonává všechny operace správně (především u USART docházelo při nižších kmitočtech k problémům s přenosem dat). Nejnižší možná hodnota je zvolena proto, aby osciloskop stíhal spolehlivě reagovat na změny při měření průběhů.

Pro kompletní ověření nastavení experimentálního pracoviště byla provedena jednoduchá proudová analýza za cílem naměření proudového průběhu implementovaného šifrovacího algoritmu AES. To znamená, že je změřen proudový průběhy obsahující operaci přičtení klíče a následujících čtrnáct rund. Teoreticky by na změřené proudové spotřebě měla být operaci přičtení klíče a 14 prováděných rund šifrování jasně vidět. Změřený průběh se nachází na obrázku č. 4.1.

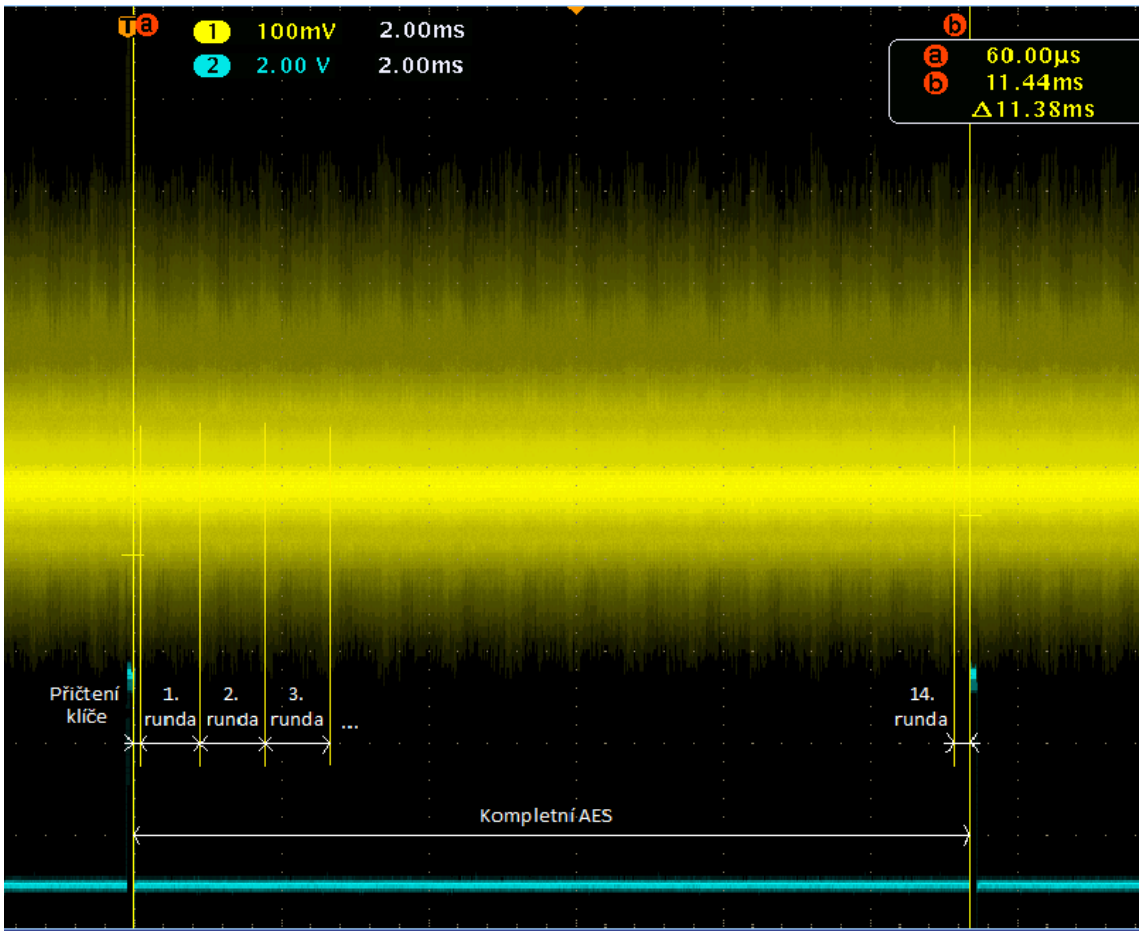
Tab. 4.1: Časy jednotlivých operací šifry AES.

Operace	**Nastaveno**
Přičtení klíče	100 μs
Klasická runda (1. až 13. runda)	900 μs
Poslední runda (zkrácená)	400 μs

Z obrázku průběhu kompletního šifrování AES (obrázek č. 4.1) je všech 14 rund šifry AES možné napočítat. Pro kontrolu je však pomocí osciloskopu změřena doba trvání operace přičtení klíče, doba trvání první rundy (dalších 13 rund trvá stejnou dobu) a poslední zkrácené čtrnácté rundy. Přehled všech časů jednotlivých operací je uveden v tabulce č. 4.1. Součtem všech měřených časů jednotlivých částí lze získat celkovou dobu trvání implementované šifry AES:

$$t_{AES} = t_{AddKey} + 13 * t_{Round} + t_{LastRound} = 0,1 + 13 * 0,9 + 0,4 = 12,2\,ms \quad (4.1)$$

Z obrázku č. 4.1 lze dále vyčíst, že kompletní šifrování AES trvá 11,38 ms. Tuto hodnotu můžeme ověřit, protože je známý počet cyklů implementovaného šifrování, který je uveden v podkapitole 3.1.2. Zde je uvedeno, že počet hodinových cyklů je 30 174. Dále v datovém listu mikrokontroléru ATmega16 [2] (také mikrokontroléru ATmega8 [1]) je uvedeno, že mikrokontrolér vykoná milion instrukcí za sekundu



Obr. 4.1: Průběh kompletního šifrování AES.

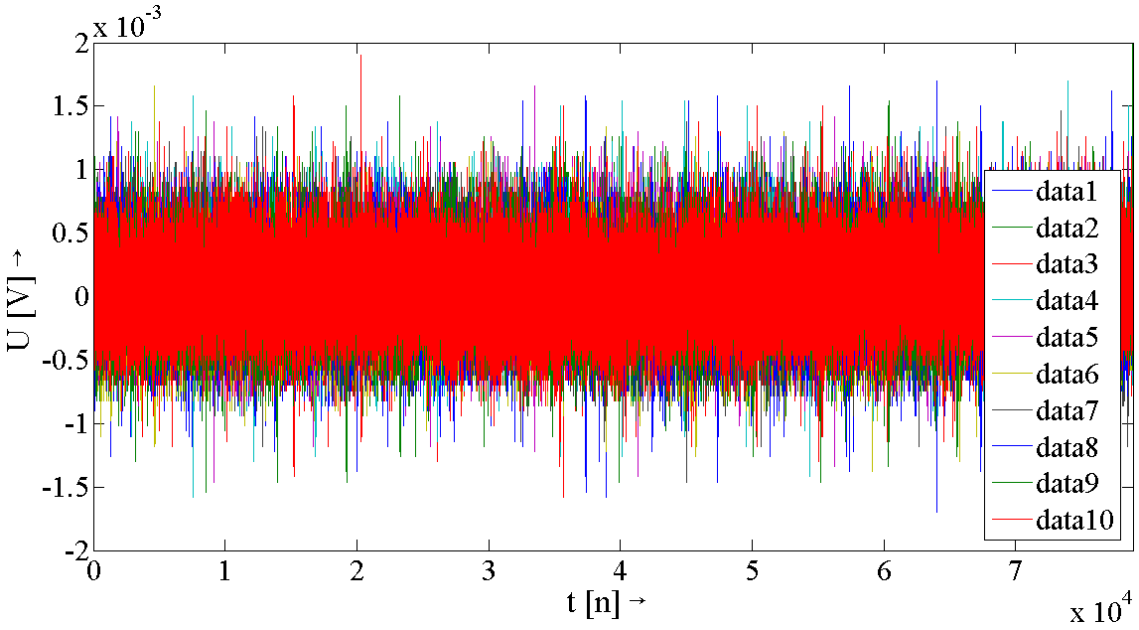
při kmitočtu 1 MHz. Z toho vyplývá, protože je kmitočet mikrokontroléru nastaven na 2 MHz, že za 1 sekundu vykoná 2 miliony instrukcí. Z těchto hodnot lze například pomocí trojčlenky vypočítat, že 30 174 hodinovým cyklům odpovídá čas přibližně 15 ms. Tato hodnota se tedy oproti změřeným 11,38 ms liší, ale vzhledem k přesnosti měření ji lze považovat za odpovídající. Tomu v podstatě odpovídá i čas vypočtený v rovnici 4.1, a to čas 12,2 ms. Z těchto zjištěných hodnot časů a především ze změřeného průběhu lze proto určit, že implementovaná šifra AES na mikrokontroléru ATmega16 určitou informaci vyzařuje. U mikrokontroléru ATmega8 bylo dosaženo téměř shodných výsledků.

4.1.1 Realizace jednoduché analýzy

V této podkapitole je provedeno srovnání průběhů naměřených elektromagnetickou sondou s průběhy naměřenými proudovou sondou pro implementace šifrování na obou mikrokontrolérech. Cílem této podkapitoly je vybrat vhodnější sondu pro mě-

ření průběhů a následnou analýzu. Postupně jsou analyzovány průběhy změřené oběma sondami na obou mikrokontrolérech, na ATmega16 i na ATmega8. Měření je provedeno pomocí skriptu pro automatické zaznamenávání průběhů, který je pro tento účel mírně upraven. K mikrokontroléru jsou posílána stále stejná data a ten tato data neustále šifruje. Pro tato data je změřeno celkem 200 průběhů pro každý mikrokontrolér.

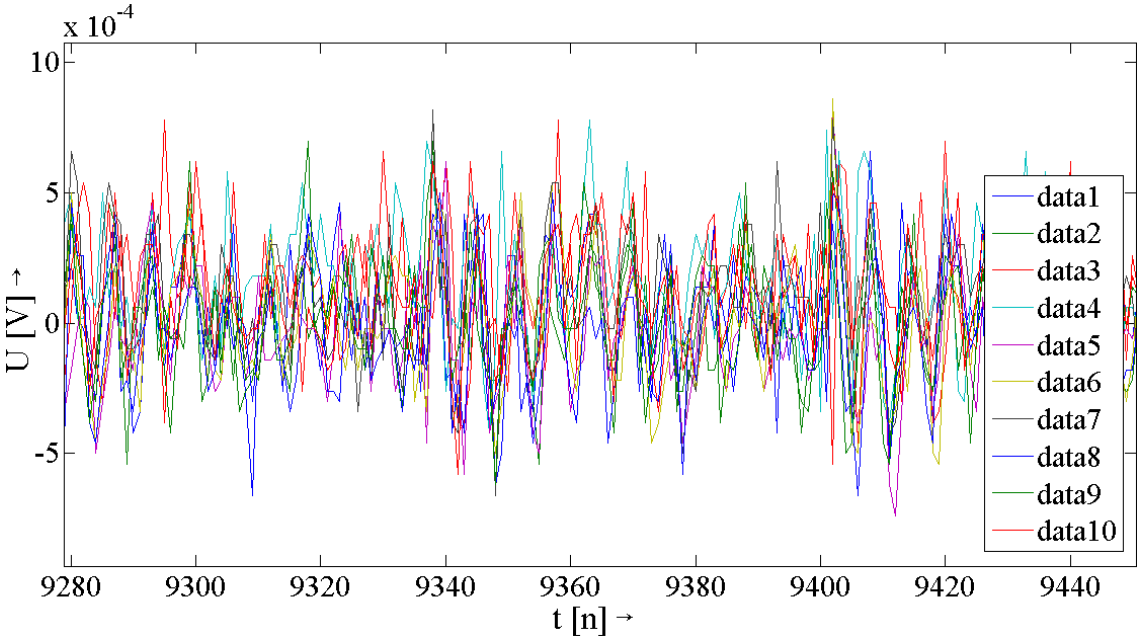
Nejprve je elektromagnetickou sondou změřeno 100 průběhů a následně dalších 100 průběhů je změřeno sondou proudovou. Jsou zaznamenány průběhy při vykonávání pouze části implementovaného šifrování, a to části operace přičtení klíče a záměny bajtů v první rundě. Jelikož jsou všechny průběhy měřeny při šifrování stále stejných dat, měly by se podle teorie shodovat svým tvarem. Pokud dojde pouze k mírným odlišnostem v jednotlivých tvarech, lze tyto odlišnosti přisoudit přítomnému šumu.



Obr. 4.2: Průběhy šifrování změřené elektromagnetickou sondou pro ATmega16.

Měření elektromagnetickou sondou

Nejprve je provedena analýza průběhů změřených elektromagnetickou sondou. Na obrázku č. 4.2 je pro interpretaci uvedeno 10 náhodně vybraných průběhů šifrování stále stejných dat na mikrokontroléru ATmega16. Z těchto průběhů je i přes hůře vnímatelné rozlišení grafu patrné to, že dochází k mnoha odlišnostem v jednotlivých tvarech průběhů. Část těchto průběhů je dále v grafu na obrázku č. 4.3 několikrát přiblížena. Z toho přiblížení je patrné, že se průběhy změřené elektromagnetickou sondou vůbec neshodují a jejich tvary připomínají náhodný šum.



Obr. 4.3: Detail průběhů šifrování změřených elektromagnetickou sondou pro mikrokontrolér ATmega16.

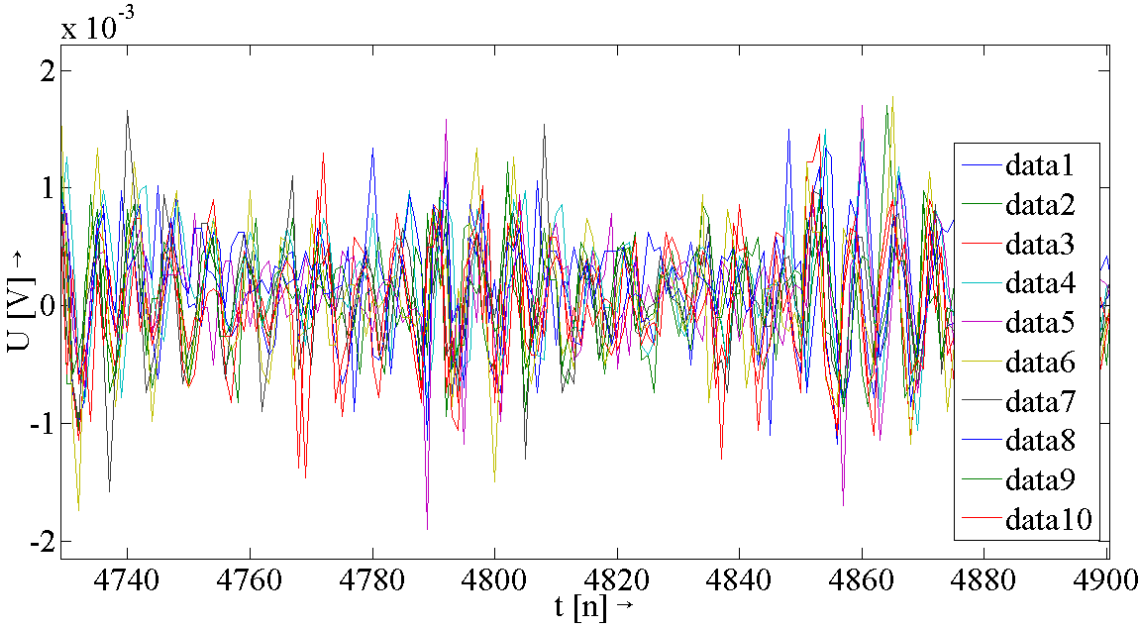
Stejných výsledků bylo dosaženo i při měření průběhů elektromagnetickou sondou na ATmega8. Na obrázku č. 4.4 je zobrazen detail průběhů změřených na ATmega8. I v tomto případě jednotlivé průběhy připomínají náhodný šum.

Z měření průběhů elektromagnetickou sondou lze dosažení negativních výsledků vysvětlit dvěma způsoby. Zaprvé se měření nezdařilo, protože oba mikrokontroléry, ATmega16 i ATmega8, jsou konstruovány tak, že jejich elektromagnetický postranní kanál nevyzařuje žádné informace. Druhým (velmi pravděpodobným) důvodem neúspěšného měření může být fakt, že elektromagnetická sonda, která byla k dispozici, je nějakým způsobem poškozena a proto nemá správnou funkci.

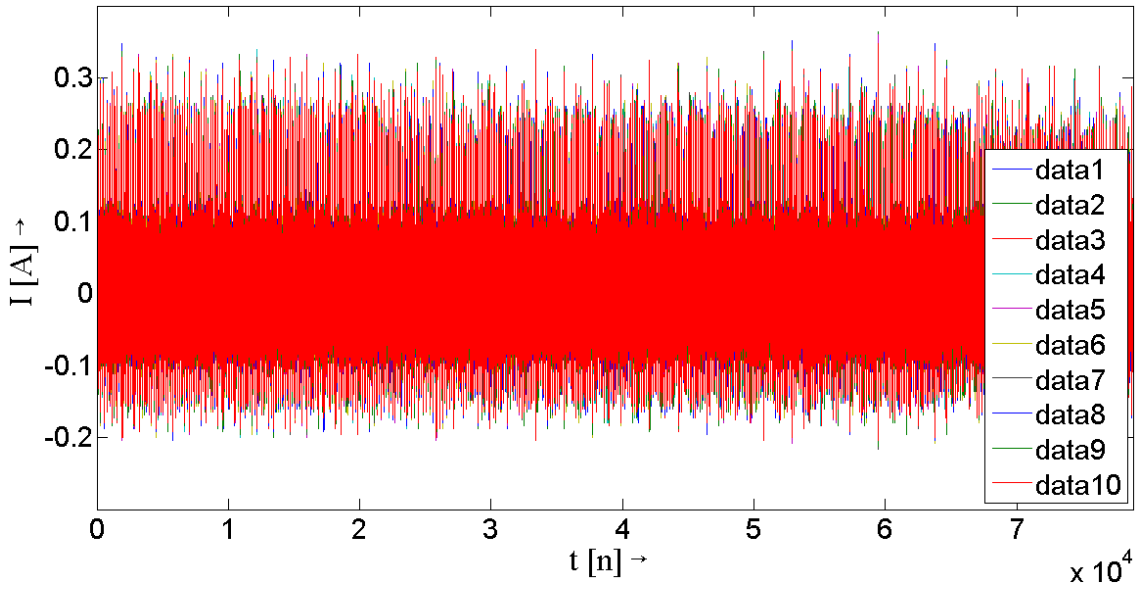
Měření proudovou sondou

V druhém případě je provedena jednoduchá analýza průběhů změřených proudovou sondou. V grafu na obrázku č. 4.5 je opět zobrazeno 10 náhodně vybraných průběhů šifrování neměnných dat na mikrokontroléru ATmega16. Zde je na první pohled patrné, že jsou jednotlivé průběhy prakticky shodné. Mírné odlišnosti lze v tomto případě vysvětlit šumem, který je při měření přítomný.

Část průběhů změřených proudovou sondou je opět několikrát přiblížena a zobrazena v grafu na obrázku č. 4.6. Zde je možné už z tohoto přiblížení jednoznačně určit proudové průběhy jednotlivých instrukcí, které mikrokontrolér ATmega16 vykonává. Zároveň je patrné, že se jednotlivé průběhy přesně kopírují. Z tohoto měření

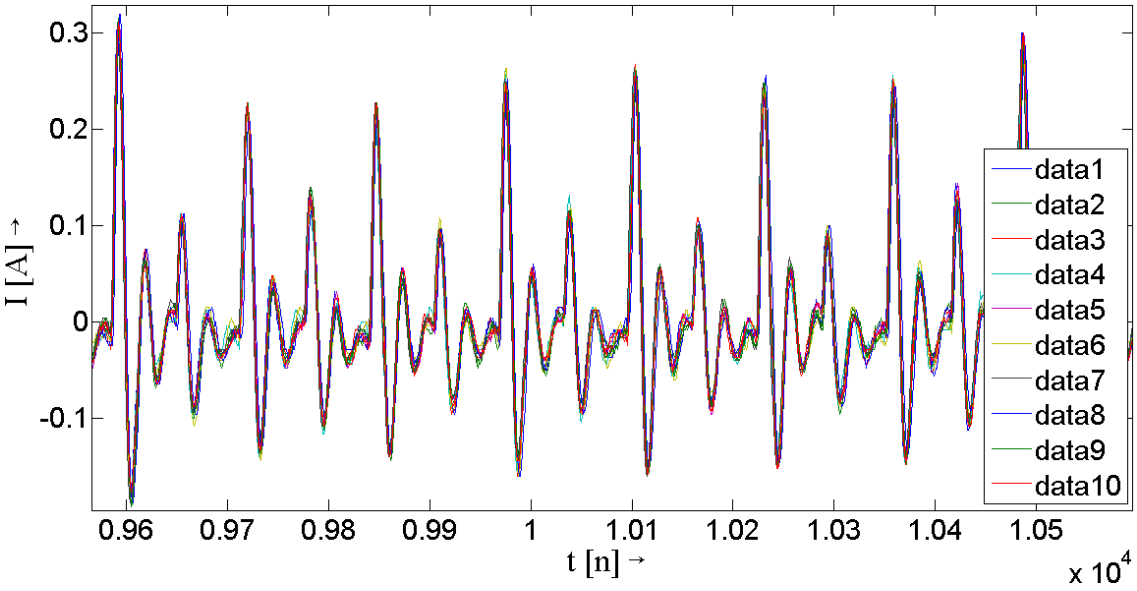


Obr. 4.4: Detail průběhů šifrování změřených elektromagnetickou sondou pro mikrokontrolér ATmega8.

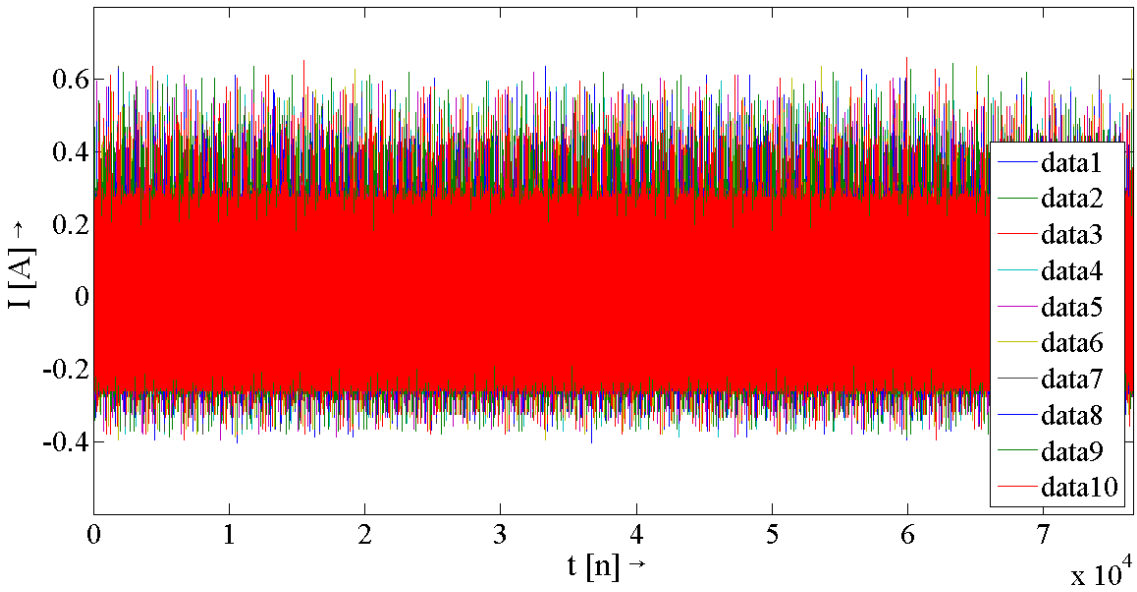


Obr. 4.5: Průběhy šifrování změřené proudovou sondou pro ATmega16.

je patrné, že proudový postranní kanál mikrokontroléru ATmega16 vyzařuje informace o implementovaném šifrování.

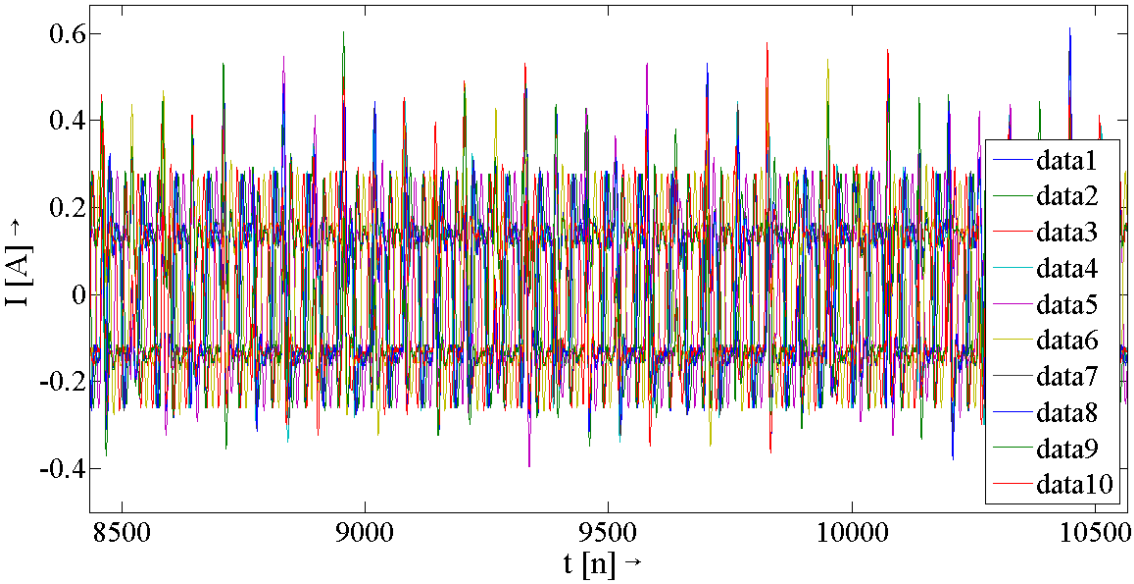


Obr. 4.6: Detail průběhů šifrování změřených proudovou sondou pro mikrokontrolér ATmega16.

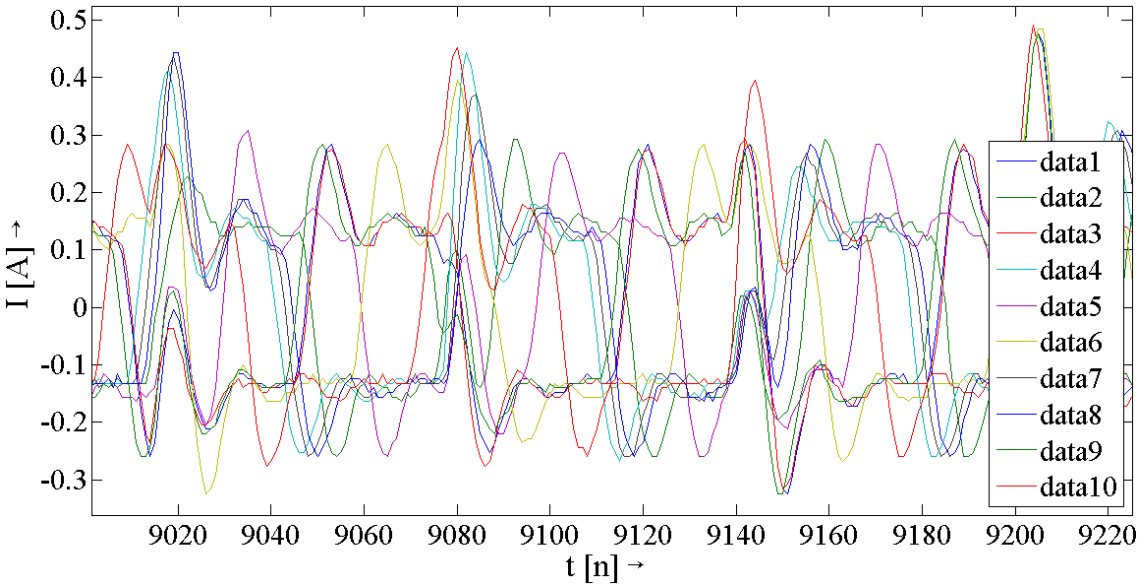


Obr. 4.7: Průběhy šifrování změřené proudovou sondou pro ATmega8.

V další části bylo provedeno měření průběhů proudovou sondou i na šifrování vykonávaném mikrokontrolérem ATmega8. V tomto případě již výsledky měření nejsou tak příznivé. Na obrázku č. 4.7 je zobrazen graf s 10 náhodně vybranými průběhy pro šifrování neměnných dat. Z tohoto grafu zatím jsou patrné zatím velmi malé rozdíly v jednotlivých průbězích. Několikanásobným přiblížením těchto prů-



Obr. 4.8: Detail průběhů šifrování změřených proudovou sondou pro mikrokontrolér ATmega8.



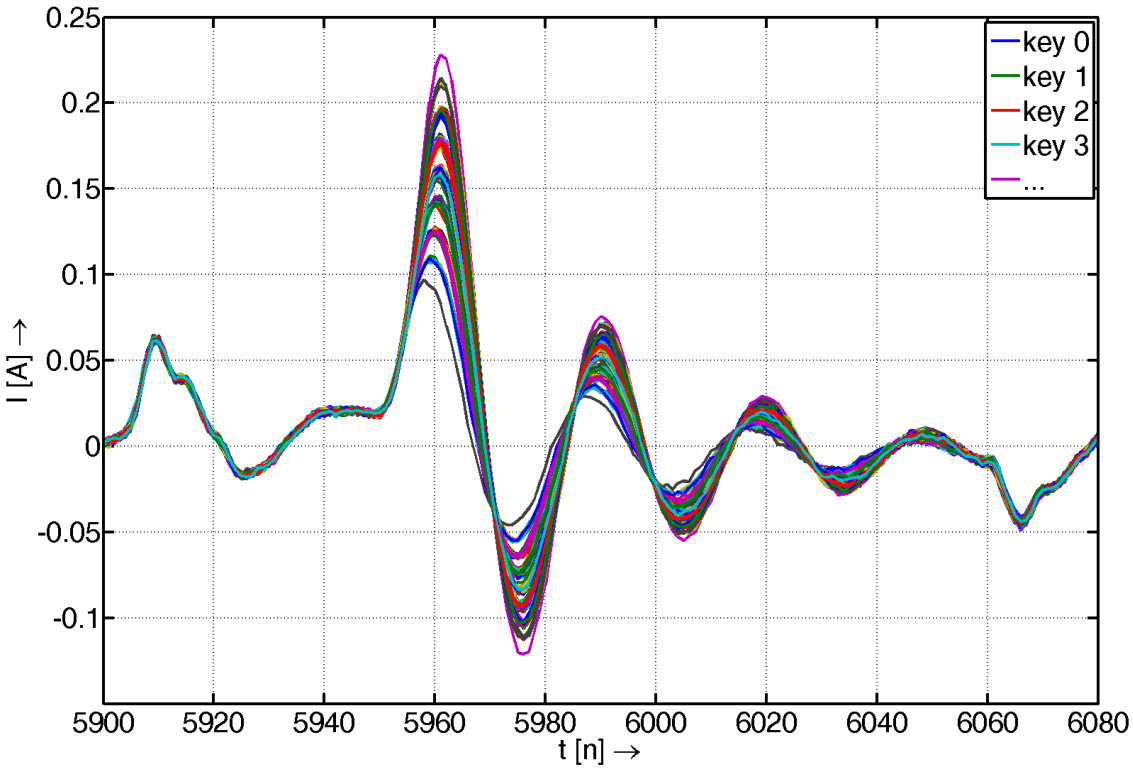
Obr. 4.9: Podrobnější detail průběhů šifrování změřených proudovou sondou pro mikrokontrolér ATmega8.

běhů (obrázek č. 4.8) lze dojít k závěru, že je změřen opět pouze náhodný šum. Pokud je však provedeno ještě větší přiblížení (obrázek č. 4.9), je patrné, že lze z těchto průběhů určit vykonávání jednotlivých instrukcí mikrokontrolérem. Některé průběhy se v tomto grafu dokonce kopírují. Při komplexním pohledu na všechny zob-

razené průběhy lze zjistit, že se všechny tvary shodují. Naměřené průběhy jsou však značně navzájem posunuty a mohou na první pohled připomínat náhodný šum. Důvod posunutí se při analýze nepodařil zjistit (je pravděpodobné, že je v ATmega8 zabudována ochrana ovlivňující časovou oblast). Celkově však lze říci, že proudový kanál vyzařuje informace o implementovaném šifrování.

4.1.2 Analýza modelu proudové spotřeby

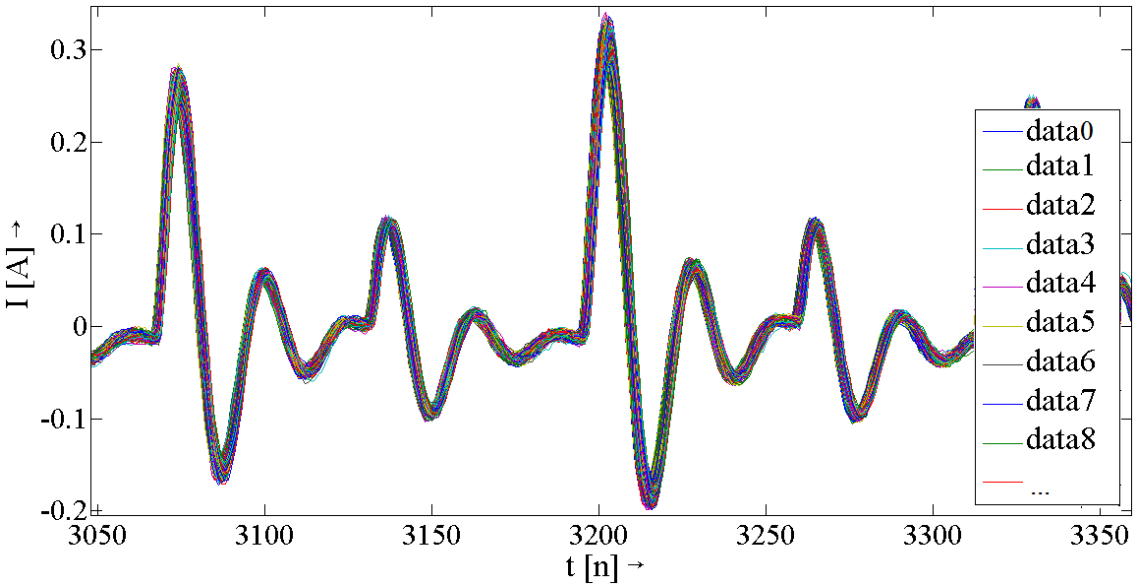
Dalším cílem jednoduché analýzy je ověření skutečnosti, zda lze ze změřených průběhů určit jejich závislost na právě zpracovávaných datech. K tomuto účelu proběhlo další pomocné měření. Pro toto měření je skript pro automatické zaznamenávání upraven tak, aby v datech určených k šifrování generoval pouze změnu prvního bajtu těchto dat. První bajt tedy nabývá hodnot od 0 do 255 a pro každou změnu hodnoty prvního bajtu je zaznamenáno 5 průběhů. Těchto 5 průběhů je poté zprůměrováno pro redukci případného šumu. Výsledkem je tedy 256 změřených průběhů.



Obr. 4.10: Teoretický průběh znázorňující Hammingovu váhu, převzato z [10].

V grafickém zobrazení těchto průběhů se tedy vyhledává Hammingova váha, která by se měla projevit při vykonávání instrukcí, které pracují právě s prvním bajtem šifrovaných dat. Výsledek měření by se měl podobat průběhům naznačeném

na obrázku č. 4.10. V grafu na tomto obrázku je vidět rozdělení průběhů do 9 skupin různých úrovní. To odpovídá Hammingově váze.



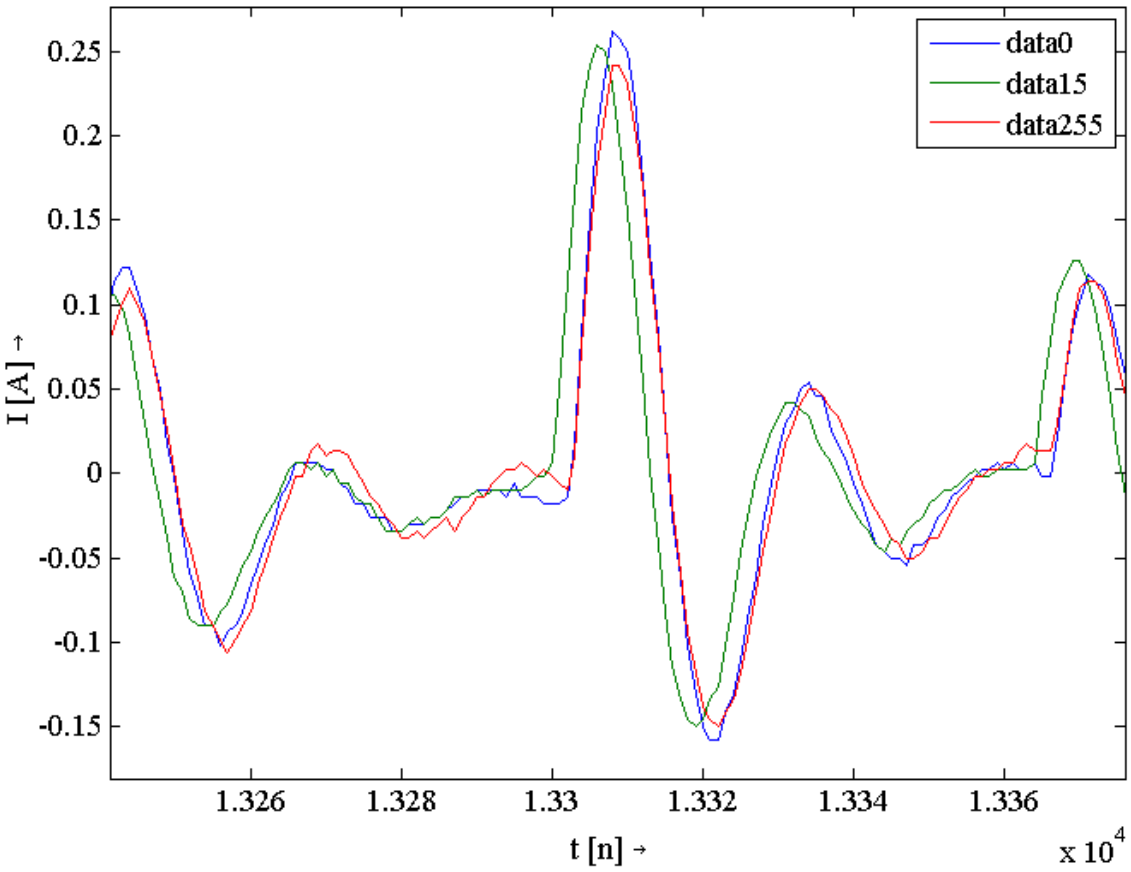
Obr. 4.11: Zjištění Hammingovy váhy z naměřených průběhů.

Výsledkem měření je graf, jehož část je zobrazena na obrázku č. 4.11. Při zkoumání tohoto průběhu však nebyla nalezena žádná část, ze které by bylo možné Hammingovu váhu určit. Celý průběh je stejný, jako právě část zobrazená na obrázku č. 4.11 Neschopnost určit Hammingovu váhu z tohoto grafu je pravděpodobně způsobena tím, že jednotlivé průběhy jsou od sebe mírně posunuty.

Posunutí lze vidět při dostatečném přiblížení průběhů, což je zobrazeno v grafu na obrázku č. 4.12, kde jsou zobrazeny 3 průběhy pro hodnoty prvního bajtu šifrovaných dat 0, 15 a 255. Těmto hodnotám odpovídají Hammingovy váhy s hodnotami 0, 4 a 8. V grafu na obrázku č. 4.12 již lze i přes posunutí Hammingovu váhu vidět, ale v obráceném pořadí. To znamená, že pokud je Hammingova váha rovna 0, má průběh nejvyšší úroveň. Naopak pokud je Hammingova váha rovna 255, má průběh nejnižší úroveň. Jedná se tedy o model obrácené Hammingovy váhy.

4.1.3 Vyhodnocení jednoduché analýzy

Z jednoduché analýzy naměřených průběhů vyplývá několik skutečností. Zaprvé, pro měření průběhů šifrování je vhodnější proudová sonda. To z důvodu, že z průběhů změřených proudovou sondou lze určit informace o implementovaném šifrování. To pro průběhy změřené elektromagnetickou sondou neplatí. Elektromagnetická sonda nepracovala správně.



Obr. 4.12: Detail posunutí průběhů naměřených na ATmega16.

Zadruhé, z průběhů změřených proudovou sondou se pravděpodobně podařila nalézt obrácená Hammingova váha, která znázorňuje závislost těchto měřených průběhů na právě zpracovávaných datech. Při měření těchto průběhů bylo zjištěno, že jsou jednotlivé průběhy od sebe navzájem mírně posunuty. Posunutí pravděpodobně vychází z kmitání časové osy průběhů, což lze vidět i při zobrazení na osciloskopu. Z tohoto důvodu pravděpodobně při měření dochází k problému se synchronizací měřené části. Tento problém se nepodařil úspěšně vyřešit.

Problém se synchronizací měřené části byl řešen několika způsoby. Byl změněn způsob měření, tak aby nedocházelo k přerušení šifrování. To znamená, že skript byl upraven tak, aby mikrokontroléru neposílal potvrzovací datovou jednotku. Mikrokontrolér tak nepřerušoval šifrování, ale šifroval neustále aktuální data, které přijal a uložil do paměti. Při této změně však došlo k ještě většímu vzájemnému posunutí jednotlivých průběhů.

Dále byly prováděny pokusy s kmitočtem interního oscilátoru mikrokontroléru. Tento kmitočet byl zvyšován i snižován, ale ke zlepšení synchronizace měřených průběhů opět nedošlo. Na závěr byl interní oscilátor mikrokontroléru nahrazen externím

krystalovým oscilátorem, ale naměřené výsledky měly stejný charakter. I při pokusech s různými kmitočty a oscilátory byly průběhy vzájemně posunuty.

Z těchto důvodů je problém se synchronizací vlivem kmitání měřených průběhů přisouzen vlastnostem mikrokontrolérů ATmega.

4.2 Realizace diferenciální proudové analýzy

V této části textu je popsáno, jakým způsobem probíhala vlastní realizace diferenciální proudové analýzy. Nejprve je zde popsáno vlastní měření proudových průběhů změřených pomocí skriptu pro automatické zaznamenávání průběhů a dále jsou tyto průběhy popsány. Nakonec je popsána i vlastní realizace diferenciální proudové analýzy a diskutovány její výsledky. Cílem diferenciální proudové analýzy je tedy odhalení šifrovacího klíče šifry AES implementované na mikrokontroléru ATmega16.

4.2.1 Příprava na realizaci

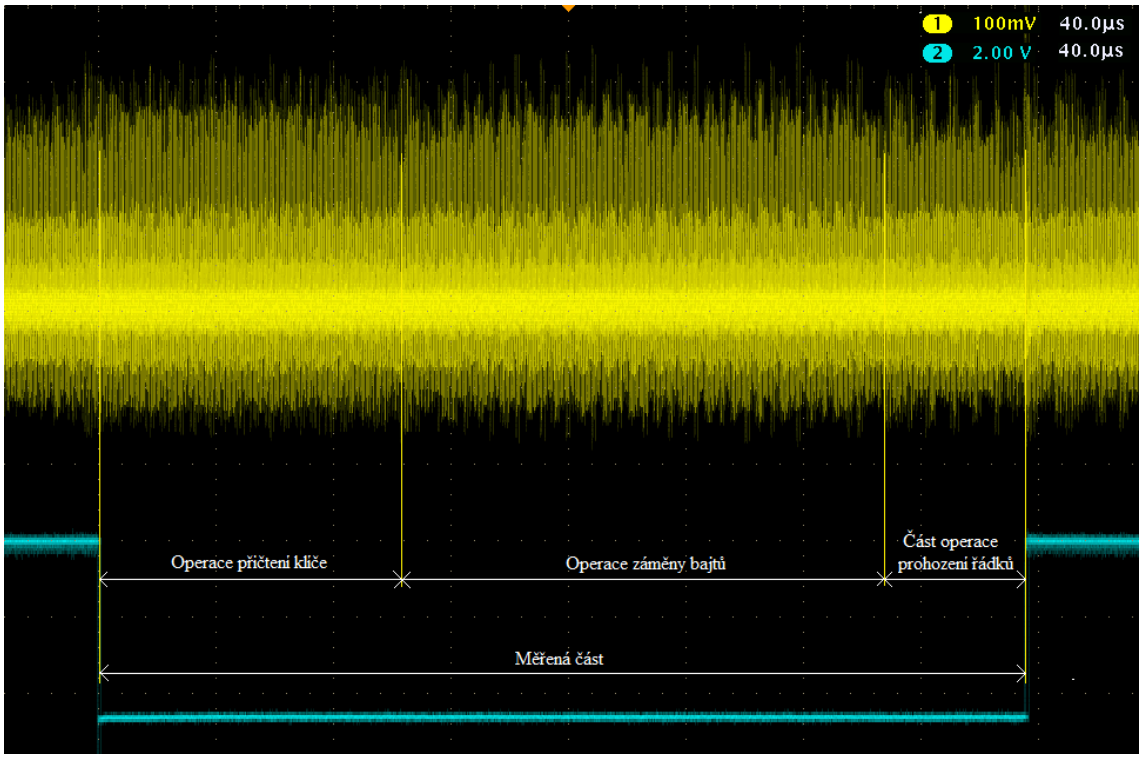
Z analýzy popsané v podkapitole 4.1.1 vyplynulo, že průběhy změřené elektromagnetickou sondou neodpovídají vykonávaným operacím, ale připomínají náhodný šum. Proto je diferenciální proudová analýza realizována pouze z průběhů změřených proudovou sondou, ze kterých je patrné, že obsahují informace o implementované šifře. Realizace proběhne pro oba použité mikrokontroléry, tedy pro ATmega8 i pro ATmega16. Analýza je provedena s využitím standardního i obráceného modelu Hammingovy váhy.

Pro realizaci diferenciální proudové analýzy je využitý skript, který je napsaný v prostředí MATLAB. Tento skript byl podrobně prostudován a proběhlo ověření správnosti jeho výpočtů. Skript byl vytvořen autory knihy *Power Analysis Attacks - Revealing the Secrets of Smartcards* [8]. Pracuje takovým způsobem, že realizuje útok na výstup operace záměny bajtů šifry AES v první rundě. Pracuje proto s operací přičtení klíče a s operací substituce bajtů. Může realizovat diferenciální proudovou analýzu dvěma metodami, a to buďto pomocí metody využívající korelační koeficient nebo pomocí Kocherovy metody. Útok se provádí vždy na 1 bajt klíče. Pro klíč o délce 32 bajtů musí být pro kompletní odhalení klíče útok proveden 32krát. Skript pro realizaci diferenciální proudové analýzy je k dispozici ke stažení na webových stránkách www.dpabook.org [6].

4.2.2 Měření proudových průběhů a jejich rozbor

Pro měření průběhů určených k analýze byl využitý skript pro automatické ukládání průběhů popsaný v kapitole 3.2. Pomocí tohoto skriptu bylo pro realizaci diferen-

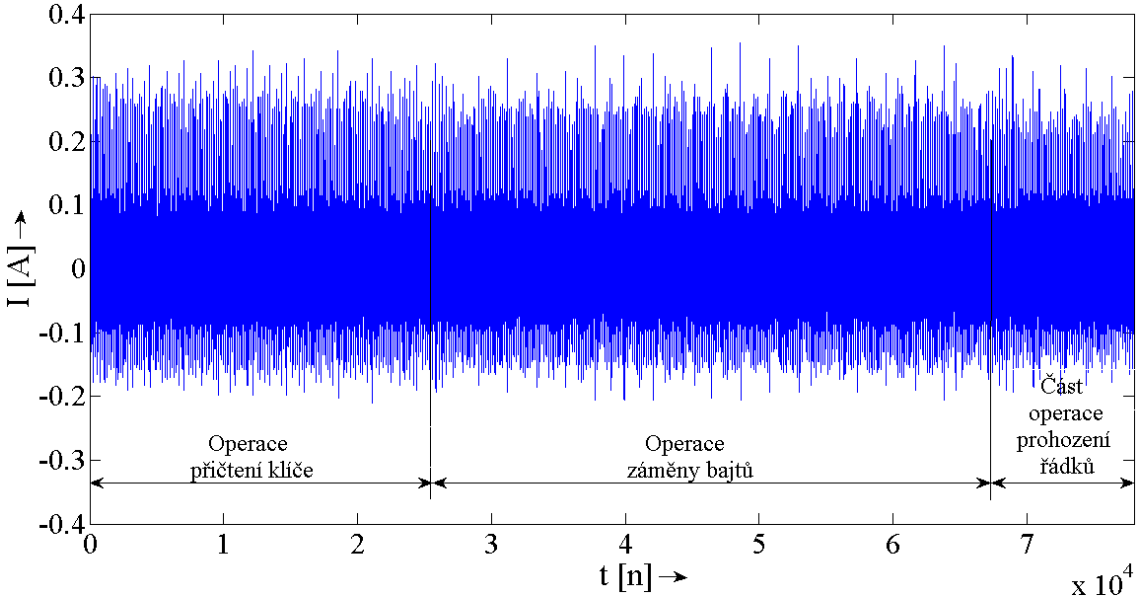
ciální proudové analýzy změřeno 3 tisíce průběhů šifrování na jednom mikrokontroléru. Toto měření proběhlo pomocí proudové sondy pro oba mikrokontroléry (tedy pro ATmega16 i pro ATmega8) 3krát. Průběhy byly následně zpracovány a ořezány a tímto převedeny do finální podoby pro analýzu. Celkem bylo diferenciální proudové analýze podrobeno 18 tisíc průběhů šifrování. Pro analýzu byl vyzkoušen standardní i obrácený model Hammingovy váhy. V této podkapitole je v následujícím textu proveden rozbor těchto průběhů.



Obr. 4.13: Popis měřené části průběhů.

Na obrázku č. 4.13 je z obrazovky osciloskopu zachycen průběh šifrování (žlutá barva) implementovaného do mikrokontroléru ATmega16 společně se synchronizačním signálem (modrá barva). Dále jsou v tomto obrázku zaznamenány jednotlivé zachytávané operace. Jelikož je skript pro realizaci zaměřený na operaci přičtení klíče a operaci záměny bajtů z první rundy, jsou právě i průběhy těchto dvou operací zaznamenávány. Protože program pro ořezávání průběhů šifrování podle synchronizačního signálu ořezává také malé množství bodů z konců měřených průběhů, je jako rezerva měřena i část následující operace, a to operace prohození řádků z první rundy.

Na dalším obrázku (č. 4.14) je pak zobrazen stejný změřený průběh šifrování na ATmega16, který je už připraven pro realizaci analýzy. To především znamená,



Obr. 4.14: Popis průběhu připraveného k analýze.

že průběh je ořezán podle synchronizačního signálu. Také v tomto průběhu jsou vyznačeny všechny 3 části šifry AES.

Z těchto dvou uvedených obrázků lze vyčíst počet zpracovávaných bajtů jednotlivými operacemi. Především ve tvaru proudového průběhu operace záměny bajtů lze napočítat 16ti bajtový blok zpracovávaných dat. Také v tomto případě se stejně jako v jednoduché analýze potvrzuje skutečnost, že proudový postranní kanál vyzařuje informace.

Stejně jako pro mikrokontrolér ATmega16 byl tento rozbor proveden i pro mikrokontrolér ATmega8. Protože bylo dosaženo prakticky stejných výsledků u obou těchto rozborů, jsou tyto výsledky prezentovány pouze na průběhu změřeném mikrokontrolérem ATmega16.

4.2.3 Realizace analýzy ze změřených průběhů

V této části je popsána vlastní realizace diferenciální proudové analýzy a její výsledky. Protože u obou mikrokontroléry, které byly analýze vystaveny, bylo dosaženo shodných výsledků, budou tyto výsledky prezentovány pouze na mikrokontroléru ATmega16. Pro prezentaci výsledků je vybrán útok na 2. a na 16. bajt bajt šifrovacího klíče. Tento příklad útoku je realizován pomocí metody korelačního koeficientu s využitím obráceného modelu Hammingovy váhy.

Do skriptu, který diferenciální proudovou analýzu realizuje, vstupují tyto parametry: matice s vlastními měřenými proudovými průběhy (velikost 3 000x78 000

Tab. 4.2: Hodnoty jednotlivých bajtů hledaného klíče šifry AES implementované do mikrokontroléru ATmega16.

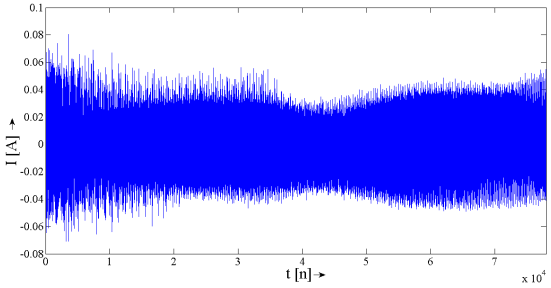
Pořadí bajtu	**Hodnota**	**Pořadí bajtu**	**Hodnota**
1. bajt	10	17. bajt	228
2. bajt	25	18. bajt	245
3. bajt	71	19. bajt	140
4. bajt	174	20. bajt	36
5. bajt	168	21. bajt	39
6. bajt	42	22. bajt	66
7. bajt	31	23. bajt	215
8. bajt	128	24. bajt	65
9. bajt	245	25. bajt	208
10. bajt	87	26. bajt	63
11. bajt	150	27. bajt	237
12. bajt	58	28. bajt	90
13. bajt	192	29. bajt	51
14. bajt	66	30. bajt	65
15. bajt	130	31. bajt	158
16. bajt	179	32. bajt	121

bodů, tento počet bodů zůstane po ořezání průběhů), matice s daty, pro jejichž zašifrování byly proudové průběhy změřeny (velikost 3 000x16 bajtů), matice s Hammingovými vahami (1x256 bajtů) a matice s S-boxem, pomocí kterého se provádí operace záměny bajtů (1x256 bajtů).

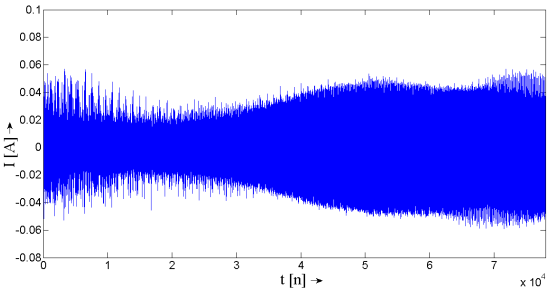
Výsledkem vykonání skriptu je matice průběhů (256x78 000 bodů). Jeden průběh odpovídá jedné hodnotě bajtu, na který je prováděn útok. Bajt může nabývat hodnot 0 až 255, z toho vyplývá 256 průběhů. Z těchto průběhů se má podle teoretických předpokladů svým tvarem jeden z průběhů lišit. Právě tento rozdílný průběh odpovídá správné hodnotě bajtu hledaného šifrovacího klíče implementované šifry AES. V tabulce č. 4.2 jsou uvedeny hodnoty všech 32 bajtů šifrovacího klíče vyslaného do mikrokontroléru ATmega16, kterým se šifrovala data při měření proudových průběhů. Tento klíč je pomocí diferenciální proudové analýzy odhalován.

Útok na 2. bajt klíče

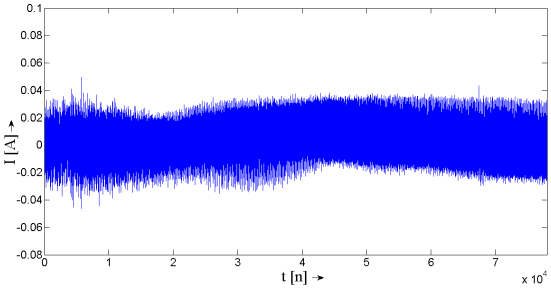
Hodnota šifrovacího klíče je známá. Klíč byl vygenerován v prostředí MATLAB, vyslán do mikrokontroléru a zaznamenám do matice. Proto je známá i hodnota 2. bajtu



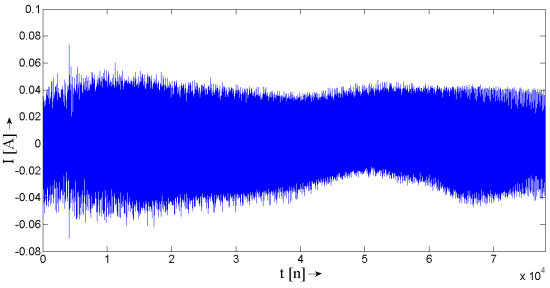
Obr. 4.15: Průběh klíče pro bajt s hodnotou 23.



Obr. 4.16: Průběh klíče pro bajt s hodnotou 24.



Obr. 4.17: Průběh klíče pro bajt s hodnotou 25.



Obr. 4.18: Průběh klíče pro bajt s hodnotou 26.

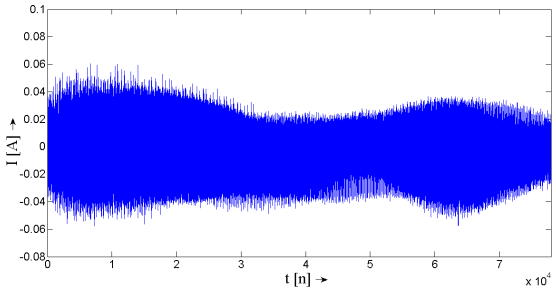
klíče, který se bude vyhledávat z průběhů klíče určených skriptem. Tento druhý bajt klíče má podle tabulky č. 4.2 hodnotu 25. Z průběhů určených skriptem by měl tedy v pořadí 26. průběh vyčnívat nad ostatními průběhy. Vyčnívat by měl až 26. průběh proto, protože první průběh odpovídá hodnotě bajtu rovné nule, druhý průběh odpovídá číslu jedna, až 26. průběh odpovídá hodnotě 25. Na obrázcích s čísly 4.15, 4.16, 4.17 a 4.18 jsou postupně vykresleny průběhy pro hodnoty 23, 24, 25 a 26 hledaného 2. bajtu klíče.

Z těchto průběhů nelze jednoznačně určit hodnotu druhého bajtu hledaného klíče. Všech 256 průběhů vyjadřující hodnoty druhého bajtu má odlišné tvary a žádný tvar průběhu nijak nevyčnívá.

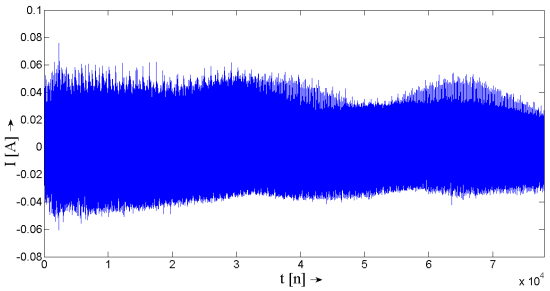
Útok na 16. bajt klíče

Šestnáctý bajt klíče má podle tabulky č. 4.2 hodnotu 179. Z průběhů určených skriptem by měl tedy v pořadí 180. průběh vyčnívat nad ostatními průběhy. Na obrázcích s čísly 4.19, 4.20, 4.21 a 4.22 jsou postupně vykresleny průběhy pro hodnoty 178, 179, 180 a 181 hledaného 16. bajtu klíče.

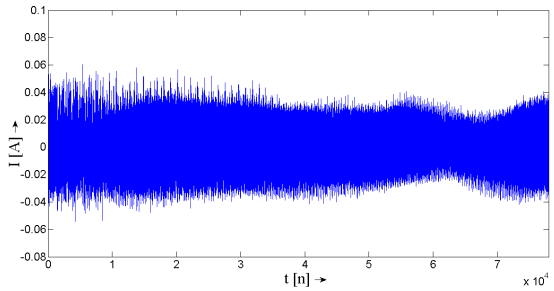
Ani u těchto průběhů nelze říci, že by nějaký z nich jednoznačně vyčníval nad ostatními. Proto správná hodnota 16. bajtu šifrovacího klíče nelze stejně jako při hledání 2. bajtu z těchto průběhů určit. Stejně neúspěšných výsledků jako při zde prezento-



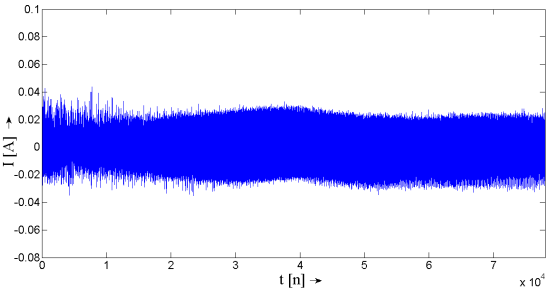
Obr. 4.19: Průběh klíče pro bajt s hodnotou 178.



Obr. 4.20: Průběh klíče pro bajt s hodnotou 179.



Obr. 4.21: Průběh klíče pro bajt s hodnotou 180.



Obr. 4.22: Průběh klíče pro bajt s hodnotou 181.

vaném hledání 2. a 16. bajtu šifrovacího klíče bylo dosaženo i pro zbylých 30 bajtů tohoto klíče.

4.2.4 Vyhodnocení diferenciální proudové analýzy

Realizace diferenciální proudové analýzy korelační metodou proběhla s negativními výsledky (s využitím obráceného i standardního modelu Hammingovy váhy). Z průběhů určených skriptem, který diferenciální proudovou analýzu vykonával, nelze určit žádný průběh, který by svým tvarem vyčníval a tímto by označoval hledanou hodnotu určitého bajtu šifrovacího klíče. Tohoto výsledku bylo dosaženo po vykonání všech analýz: žádná z vykonávaných analýz na mikrokontroléru ATmega16 neodhalila ani jeden bajt, stejně tak ani žádná z analýz vykonávaných na mikrokontroléru ATmega8. Výsledek analýzy nezměnila ani výměna modelu Hammingovy váhy za model obrácené Hammingovy váhu.

Z důvodu negativních výsledků byla vykonána diferenciální proudová analýza Kocherovou metodou. Také však bylo dosaženo stejných výsledků jako při vykonávání analýzy metodou korelačního koeficientu. Ani v tomto případě nebyl výsledkem analýzy žádný průběh, pomocí kterého by šla určit hodnota hledaného bajtu klíče. Tato skutečnost platí také pro ATmega16 i pro ATmega8.

Výsledky vyplývající ze všech provedených diferenciálních proudových analýz s největší pravděpodobností způsobila skutečnost, že měřené průběhy jsou při zpracování vzájemně mírně posunuty. Tento fakt vyplynul již z jednoduché analýzy. Posunutí pravděpodobně způsobí to, že skript, který diferenciální proudovou analýzu realizuje, nezaznamená správným způsobem ty body nebo části průběhů, ze kterých by mohl šifrovací klíč přesně určit.

5 ZÁVĚR

V této bakalářské práci bylo hlavními úkoly realizovat komplexní experimentální pracoviště pro automatické ukládání průběhů z osciloskopu, realizovat jednoduchou analýzu a diferenciální proudovou analýzu. Všechny úkoly byly také splněny.

Při vývoji experimentálního pracoviště se dospělo k závěru, že pro realizaci automatického měření průběhů je nejvhodnější využít MATLAB s rozšířením Instrument Control Toolbox. Pomocí tohoto rozšíření lze ovládat kromě osciloskopu použitého v této práci také velké množství dalších osciloskopů. Také lze s jeho pomocí splnit všechny požadavky automatického měření průběhů.

Na základě Instrument Control Toolboxu byl navržen a napsán program (skript v prostředí MATLAB), který zaznamenává všechna potřebná data o měřených průbězích, jako vlastní průběhy šifrování, synchronizační signály a šifrovaná data. Součástí tohoto skriptu je komplexní jednoduchý komunikační protokol, ve kterém je definována jednak veškerá potřebná komunikace s osciloskopem Tektronix a dále také veškerá komunikace s mikrokontroléry řady ATmega, na kterých je implementováno šifrování.

Výhodou tohoto programu je, že lze s drobnými úpravami využít také v budoucnosti pro měření průběhů šifrování na mikrokontrolérech jiných řad s využitím jiných osciloskopů. Další výhodou je skutečnost, že program ukládá data přímo do matic v prostředí MATLAB, kde lze s těmito daty dále přímo pracovat. Program byl dále v této práci využit pro měření průběhů určených k analýzám.

Z informací zjištěných při zkoumání proudového postranního těchto mikrokontrolérů lze implementované šifrování do určité míry popsat. Proto průběhů změřených proudovou sondou na proudovém postranním kanálu bylo využito pro realizaci diferenciální proudové analýzy. Cílů této analýzy, tedy zjištění šifrovacích klíčů, nebylo dosaženo. Ze změřených průběhů byl zjištěn fakt, že při měření dochází k problémům se synchronizací na měřenou část vlivem kmitání časové základny při vykonávání instrukcí mikrokontrolérem. Tyto problémy se nepodařilo odstranit a jejich přítomnost byla přisouzena vlastnostem mikrokontrolérů řady ATmega.

Problémy se synchronizací u naměřených proudových průběhů by mohla vyřešit aplikace nástroje, který se označuje jako principiální analýza komponent (PCA - *Principal Component Analysis*). Jedná se o nástroj, který by měl proudové průběhy naměřené pro diferenciální proudovou upravit analýzu potřebným způsobem.

LITERATURA

[1] 8-bit AVR Atmel with 8K Bytes In-System Programmable Flash. *ATmega8(L)*. Atmel, 331 s.

[2] 8-bit AVR Microcontroller with 16K Bytes In-System Programmable Flash. *ATmega16(L)*. Atmel, 357 s.

[3] 8-bit AVR Microcontroller with 16K Bytes In-System Programmable Flash. *ATmega162(V)*. Atmel, 234 s.

[4] AVR-Crypto-Lib. *The implementation of AES* [online]. [cit. 2014-04-27]. Dostupné z: <`http://avrcryptolib.das-labor.org/trac/wiki/AES`>.

[5] DAEMEN, Joan a Vincent RIJMEN. *The design of Rijndael: AES - the Advanced Encryption Standard*. Berlin: Springer, 2002, 238 s. ISBN 35-404-2580-2.

[6] DPAbook.org. *Power Analysis Attacks - Revealing the Secrets of Smartcards* [online]. [cit. 2014-04-27]. Dostupné z: <`http://http://www.dpabook.org`>.

[7] Federal Information Processing Standards Publication 197. *Advanced Encryption Standard (AES)*. 2001. Dostupné z: <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

[8] KOCHER, P. C.; JAFFE, J.; JUN, B.: Differential Power Analysis. In CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference onAdvances in Cryptology, London, UK: Springer-Verlag, 1999, ISBN 3-540-66347-9, s. 388–397.

[9] MANGARD, S.; OSWALD, E.; POPP, T.: Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security). Secaucus, NJ, USA:Springer-Verlag New York, Inc., 2007, ISBN 0387308571.

[10] MARTINÁSEK, Zdeněk. *Kryptoanalýza postranními kanály*: dizertační práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2013. 129 s. Vedoucí práce byl doc. Ing. Václav Zeman, Ph.D.

[11] MSO4000B and DPO4000B Series Digital Phosphor Oscilloscopes. *User Manual*. Tektronix, 217 s.

[12] STK500. *User Manual*. Atmel, 61 s.

[13] Tektronix. *CT-6 High Frequency AC Current Probe: Instruction manual* [online]. [cit. 2014-05-20]. Dostupné z: <http://www.tek.com/sites/tek.com/files/media/media/resources/60W_12572_2.pdf>.

[14] THE MATHWORKS, Inc. *Instrument Control Toolbox: Instrument Driver* [online]. [cit. 2014-04-27]. Dostupné z: <http://www.mathworks.com/products/instrument/supported/instrument-drivers-search.html>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

A	Ampér
ACK	Potvrzení – Acknowledgement
AES	Pokročilý standard šifrování – Advanced Encryption Standard
bps	bit za sekundu
CMOS	Doplňující se kov-oxid-polovodič – Complementary Metal-Oxide-Semiconductor
DPA	Diferenciální proudová analýza – Diferential Power Analysis
I	proud
ISP	Vnitřní programování – In-System Programming
LED	Dioda emitující světlo – Light-Emitting Diode
LSB	Bit s nejnižší váhou – Least Significant Bit
PCA	Principiální analýza komponent – Principal Component Analysis
n	díl času
RAM	Paměť s náhodným přístupem – Random-Access Memory
ROM	Paměť určená pouze ke čtení – Read-Only Memory
RSA	Rivest-Shamir-Adleman
SPA	Jednoduchá proudová analýza – Simple Power Analysis
MHz	megahertz
ms	milisekunda
t	čas
U	napětí
USART	Univerzální synchronní a asynchronní sériový přijímač a vysílač – Universal Synchronous and Asynchronous Serial Receiver and Transmitter

USB	Jednoduchá proudová analýza – Simple Power Analysis
V	Volt
VISA	Architektura virtuálního softwarového nástroje – Virtual Instrument Software Architecture

SEZNAM PŘÍLOH

A OVĚŘENÍ IMPLEMENTOVANÉHO AES

Správnost šifrování AES je ověřena na několika příkladech. V této příloze jsou uvedeny dva z vyzkoušených příkladů.

A.1 Příklad č. 1

Do mikrokontroléru je vyslán klíč. Posloupnost zadaných příkazů do prostředí MATLAB vypadá takto:

```
key = uint8([   0   32    1   35   69  103  137  171  205  239    1
               35   69  103  137  171  205  239    1   35   69  103
              137  171  205  239    1   35   69  103  137  171  205
              239  255]);
```

```
transmit(key);
```

Následuje vyslání dat zadáním posloupností příkazů:

```
data = uint8([   1   16    1    2    3    4    5    6    7    8
                 9   10   11   12   13   14   15    0  255]);
```

```
transmit(data);
```

Na kitu s mikrokontrolérem se stiskne tlačítko SW1. Následně se v prostředí MATLAB zavolá funkce:

```
recieve();
```

V tomto okamžiku se stiskne tlačítko SW0 na vývojové sadě s mikrokontrolérem. MATLAB obdrží těchto 16 čísel, která odpovídají zašifrovaným vyslaným datům:

148 118 97 230 5 42 141 237 246 233 126 61 131 108 61 145

A.2 Příklad č. 2

Stejným způsobem jako v příkladu č. 1 jsou vyslána respektive přijata data zadáním zde uvedené posloupnosti příkazů:

```
key = uint8([   0   32  181  193   71  174  168   42   31  128  245   87
              150   58  192   66  130  179  228  245  140   36   39   66
              215   65  208   63  237   90   51   65  158  121  255]);

transmit(key);

data = uint8([   1   16  208  231   33  233  162   25   72  140
               245  247   41  248  245  124  205   37  255]);

transmit(data);

% Stisk tlacitka SW1.

recieve();

% Stisk tlacitka SW0.
```

MATLAB by měl obdržet následujících 16 čísel (16ti bajtový text):

221 89 157 243 2 13 62 33 31 165 248 250 145 56 178 224

B POPIS DALŠÍCH SKRIPTŮ

V této příloze jsou popsány další skripty, které jsou určeny pro zpracování naměřených průběhů. Do této kategorie patří skript, který všechna naměřená data přizpůsobuje skriptu pro diferenciální analýzu. Dále do této kategorie patří skript určený pro ořezání vybrané části průběhu podle synchronizačního signálu.

Dále jsou v této příloze popsány skripty, které jsou součástí skriptu pro automatické zaznamenávání průběhů nebo je lze využít i jako samostatné skripty například pro ověření funkce šifrování AES, které je implementované na mikrokontroléru.

B.1 Skript pro spojení dat a průběhů do jedné matice

Výsledkem měření pomocí skriptu pro automatické zaznamenávání průběhů jsou 3 skupiny informací (data, proudové průběhy šifrování a průběhy synchronizačních signálů), které jsou rozděleny do dílčích matic po 200. Je proto nutné tato data přizpůsobit skriptu, který vykonává ořezávání průběhů a také skriptu, který vykonává diferenciální proudovou analýzu. Oba skripty pracují s komplexními daty o jednotlivých průbězích, proto přizpůsobení spočívá ve sloučení všech dílčích matic z jednotlivých skupin informací do matice jedné. Pro tento účel byl vytvořen skript, který toto sloučení zajišťuje. Výsledkem tohoto skriptu je tedy to, že všechny změřené dílčí matice jsou sloučeny do celkem 3 matic. Jedna matice obsahuje všechna data, která byla šifrována a pro které byly zaznamenávány průběhy. Druhá matice obsahuje vlastní průběhy šifrování a třetí matice obsahuje synchronizační průběhy k měřeným proudovým průběhům.

B.2 Skript pro ořezávání průběhů

Tento skript je určen pro ořezání průběhů šifrování podle příslušných průběhů synchronizačního signálu. V prostředí MATLAB ho lze zavolat příkazem:

```
traces = traces_cutting(c_traces, s_traces);
```

Vystupující proměnná *traces* je proměnná, do které se ukládají ořezané průběhy šifrování. Vstupující proměnná *c_traces* je proměnná, ve které jsou uloženy změřené průběhy šifrování a *s_traces* je proměnná se synchronizačními průběhy.

B.3 Skript pro vyslání dat

Skript pro vyslání dat z prostředí MATLAB do mikrokontroléru je volán příkazem:

```
transmit(var);
```

Vstupující proměnná *var* je proměnná s daty, která se odesílají. Lze odeslat pouze definované datové jednotky pro klíč, data i potvrzení. Tento skript je součástí skriptu pro zaznamenávání průběhů, ovšem lze ho v prostředí MATLAB zavolat i samostatně. Samostatné volání tohoto skriptu je určeno pro ověření pravosti šifrování.

Tento skript pracuje se sériovou linkou. Proto je při použití tohoto skriptu nutné správně zadat adresu sériové linky. Adresa je defaultně pevně nastavena na COM3, ale editací skriptu ji lze snadno změnit.

B.4 Skript pro příjem dat

Skript pro příjem dat z mikrokontroléru do prostředí MATLAB je volán příkazem:

```
var = recieve;
```

Do vystupující proměnné *var* jsou uložena přijatá data. Tento skript je určen pouze pro příjem bloku zašifrovaných dat o velikosti 16 bajtů. Tento skript je vytvořen za účelem příjmu zašifrovaných dat z mikrokontroléru pro ověření správnosti šifrování.

I zde je při použití tohoto skriptu nutné, aby byla správně zadána adresa sériové linky. Také je defaultně nastavena na COM3.

C OBSAH PŘILOŽENÉHO DVD

Složka Elektronická podoba bakalářské práce
Zde se nachází elektronické podoba této práce v .pdf souboru.

Složka Implementace AES
Zde se nachází implementace šifrovacího algoritmu AES na ATmega8, ATmega16 a Atmega162.

Složka MATLAB - Skripty
Zde se nachází skripty napsané v prostředí MATLAB (.m soubory). Jedná se o skripty pro realizaci diferenciální proudové analýzy, skript pro zaznamenávání průběhů a skripty pro zpracování naměřených průběhů.

Složka MATLAB Tektronix Driver
Zde se nachází ovladač k osciloskopu Tektronix MSO4034B.

Složka Naměřené průběhy
Zde se nachází průběhy změřené proudovou sondou, které byly použity pro analýzu (.mat soubory).

Složka NI MAX
Zde se nachází VISA ovladač (Measurement & Automation Explorer).

Složka Tektronix Software
Zde se nachází další ovladače osciloskopu Tektronix. Například ovladač pro zachytávání obrazovky osciloskopu do PC.