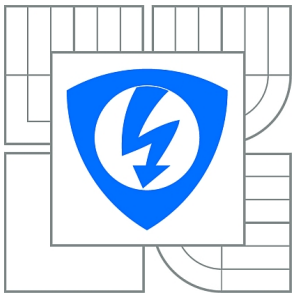
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

ÚTOKY POSTRANNÍMI KANÁLY NA ČIPOVÉ KARTY

SIDE CHANNEL ATTACKS ON CHIP CARDS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

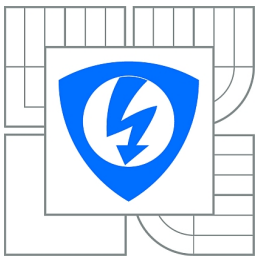
AUTOR PRÁCE — Bc. JIŘÍ MATĚJKA

AUTHOR

VEDOUCÍ PRÁCE — Ing. ZDENĚK MARTINÁSEK

SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Jiří Matějka	***ID:***	83481
Ročník:	2	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Útoky postranními kanály na čipové karty

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s implementací kryptografických algoritmů prostřednictvím čipových karet. Prostudujte a popište známé metody útoků na čipové karty postranními kanály. Seznámení se s různými formami jednoduché (SPA) a diferenční (DPA) výkonové analýzy. Z nastudovaných znalostí navrhněte a realizujte laboratorní úlohu demonstrující útok výkonovým postranním kanálem na čipovou kartu.

DOPORUČENÁ LITERATURA:

[1] RANKL, W, EFFING, W. Smart Card Handbook : Third Edition. [s.l.] : WILEY, c2003. 1088 s. ISBN 0-470-85668-8.
[2] RANKL, W. Overview about attacks on smart cards, 2003. http://www.wrankl.de/SCH/Attacks.pdf.

Termín zadání:	29.1.2010	***Termín odevzdání:***	26.5.2010

Vedoucí práce: Ing. Zdeněk Martinásek

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Práce se zabývá problematikou postranních kanálů čipových karet. Postranní kanály jsou novým směrem kryptoanalýzy. Na rozdíl od klasické kryptoanalýzy, která hledá nedostatky v matematické struktuře algoritmů, využívá analýza postranních kanálů nedokonalosti fyzické implementace algoritmů.

Úvodní část práce popisuje čipové karty, jejich užití a zabezpečení. Podrobněji je zde zmíněna problematika kryptografických algoritmů a jejich implementace na čipové karty. Dále jsou zde popsány nejznámější postranní kanály včetně příkladů jejich zneužití.

Praktická část práce je věnována návrhu laboratorní úlohy demonstrující útok výkonovým postranním kanálem na čipovou kartu. Pro tuto úlohu je navrženo kompletní laboratorní pracoviště, software pro měření a analýzu je implementován ve vývojovém prostředí LabVIEW. V závěru práce je tento útok zrealizován, jako jeho cíl je zvolena běžná SIM karta.

Klíčová slova

čipová karta, postranní kanál, proudová analýza, SIM karta, LabVIEW, laboratorní úloha

Abstract

This master thesis deals about the issues of chip cards' side channels. Side channels are new method in cryptanalysis. Unlike classical cryptanalysis, which looks for weaknesses of mathematical structure of algorithms, side channel's analysis use weaknesses of physical implementation of these algorithms.

First part describes chip cards, their usage a security. There is mentioned the issue of cryptographic algorithms and their implementation on smart cards. There are described well-known side channels and examples of their abuse.

Practical part of this thesis is focused on a proposal of a laboratory task, which demonstrates power-based side channel attack on smart card. There is designed laboratory workplace for this task, software for measurement and analysis is implemented in development tool LabVIEW. In the last part is this attack realized, the target of this attack is a standard SIM card.

Keywords

chip card, side channel, power analysis, SIM card, LabVIEW, laboratory task

Bibliografická citace

MATĚJKA, J. *Útoky postranními kanály na čipové karty* . Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 88 s. Vedoucí diplomové práce Ing. Zdeněk Martinásek.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Útoky postranními kanály na čipové karty jsem vypracoval samostatně pod vedením vedoucího diplomové práce s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne . . . .

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Úvod

S informačními a komunikační technologiemi se setkáváme ve všech oborech lidské činnosti, život bez nich si lze už jen těžko představit. Zprostředkovávají nám velké množství informací, ulehčují nám každodenní povinnosti, umožňují komunikaci s ostatními uživateli. S masovým rozšířením informačních technologií se stává stále aktuálnější otázka bezpečnosti, spolehlivosti, ochrany osobních i jiných citlivých údajů.

Jedním z prostředků, který nás každodenně doprovází a ulehčuje nám běžné činnosti, je čipová karta. Čipová karta je malé přenosné zařízení, umožňující celou škálu aplikací, od jednoduché identifikace uživatele po složité přístupy k bankovním systémům.

Čipová karta nenabízí vyšší bezpečnost pro citlivé údaje sama o sobě, pro tyto účely se využívají kryptografické algoritmy na ní implementované. Prolomit moderní kryptografické algoritmy standardní kryptoanalýzou je v současné době velmi obtížné. V druhé polovině 90. let minulého století byl ale objeven nový směr v kryptologii, bylo zjištěno, že na základě vlastností jako je délka výpočtu, spotřeba energie a dalších je možné získat z běhu algoritmu mnoho důležitých informací, ze kterých je možno odvodit tajné informace. Tyto útoky využívají tzv. postranní kanály.

Součástí této práce je studium postranních kanálů čipových karet. Jsou zde vysvětleny základní pojmy a souvislosti, týkající se čipových karet, dále je zde vysvětlena problematika bezpečnosti čipových karet. Závěr teoretické části tvoří popis jednotlivých postranních kanálů, včetně příkladů již dříve realizovaných útoků.

Hlavním cílem práce je komplexní návrh laboratorní úlohy zaměřené na ověření možnosti útoku výkonovým postraním kanálem na čipovou kartu a ověření její funkčnosti útokem na zvolenou čipovou kartu.

1. Čipové karty

1.1. Historie čipových karet

Předchůdcem karet sloužících k ukládání dat je pravděpodobně vizitka. První plastiková karta byla použita společností Diner Club v roce 1950. Na konci padesátých let se k používání této novinky přidaly další dvě firmy, Amarican Express a Carte Blance. První kreditní karta byla vydána bankou Bank of America, tento okamžik je počátkem VISA karet. Společnost Interbank přišla s vlastním systémem nazvaným Mastercard. Každopádně, tyto předchůdci dnešních karet byly schopné uchovávat pouze vyražené údaje (jména, čísla, kódy atd.)

První karta s magnetickým páskem byla vyvinuta společností International Air Transportation Association (IATA) v sedmdesátých letech 20. století. Magnetický proužek pojme přibližně 82 bitů/cm (210 bitů/palec), skládá se ze tří stop. Každá stopa je rozdělena na domény, které jsou při zápisu zmagnetizovány. Pokud se v rámci jedné domény polarizace nemění, pak tato doména reprezentuje 0. Pokud se polarizace mění, pak doména reprezentuje 1. Hlavní problémem magnetických karet je, že pokud máme k dispozici patřičné vybavení, lze jednoduše číst i měnit informace zapsané na magnetickém proužku. Proto na příklad u magnetických karet používaných pro bezhotovostní platby není PIN uložen přímo na kartě.

Mnohem větší objem dat může být uložen na optických kartách. V tomto případě, čtení i zápis dat probíhá optickou metodou. Obvykle je celý povrch karty použit pro uložení dat, kapacita dosahuje několika megabajtů. Nevýhodou těchto karet je vysoká pořizovací cena. Optické karty jsou hlavně využívány ve zdravotnictví, slouží jako úložiště medicínských dat pacientů.

Dalším významným krokem v evoluci karet bylo objevení čipových karet, tedy karet založených na použití mikroelektronických okruhů na počátku 70. let 20. století. V roce 1979 vyprodukovala firma Bull první kartu osazenou mikroprocesorem. Na této kartě byl procesor v odděleném čipu, což se ukázalo být nevhodným řešením. Až zlepšením technologie v 80. letech 20. století bylo možné integrovat všechny obvody do jednoho čipu. Tím se znatelně rozšířilo používání čipových karet, například ve Francii jsou čipové karty používané od roku 1986 ve veřejných telefonech.

Současné čipové karty disponují mnohem větší výpočetní silou. To umožňuje kartám nejenom vykonávat příkazy „z venku“, ale také spouštět oddělené programy. Tyto karty jsou označovány jako programovatelné smart karty. Karty jsou vytvořeny k použití víceuživatelského operačního systému umožňujícího multitasking. V případě smart karet je přístup k datům kontrolován samotnou kartou. [1] [2]

1.2. Čipové karty v současnosti

Čipová karta (bývá označována i jako smart karta) je zařízení obsahující vestavěný integrovaný obvod (ICC - integrated circuit chip), který může být tvořen jak procesorem s interní pamětí nebo jen samotným paměťovým čipem (tyto karty se označují jako paměťové karty). Integrovaný procesor si lze představit jako miniaturní počítač obsahující vstupní a výstupní port, operační systém a pevný disk. Umožňuje čipovým kartám nejenom uchovávat velké množství dat, ale zejména provádět vlastní funkce (např. šifrování, vzájemnou autentizaci a komunikovat s čtečkou čipových karet). Karta se připojuje ke čtečce pomocí přímého fyzického kontaktu nebo bezdrátovým radiovým rozhraním. Jelikož čipové karty neobsahují žádný zdroj energie, dodává jim ji právě čtečka. [2]

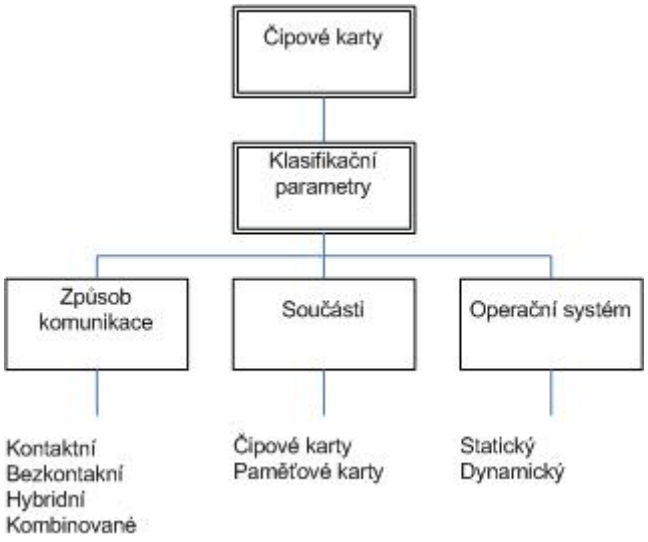
Technologické a technické parametry čipových karet určují mezinárodní standarty. Čipové karty mohou mít mnoho podob, kromě klasického tvaru platební karty se používají v mobilních telefonech tzv. SIM karty, Tyto karty jsou jen nepatrně větší než samotné kontaktní rozhraní. Další možností, jak může být čip zapouzdřen, je token USB. [2]

Čipové karty mají celosvětově velice široké uplatnění, využívají se např. v těchto oblastech:

- bezpečnost, identifikace uživatelů – zaměstnanecké přístupové karty, elektronické cestovní pasy, řidičské průkazy,
- zdravotnictví – přenosné lékařské záznamové karty,
- finančnictví – kontaktní a bezkontaktní kreditní/debetní karty,
- telekomunikace – GSM SIM karty, předplacené telefonní karty.

1.2.1. Rozdělení čipových karet

Čipové karty lze dělit podle několika kritérií: podle způsobu, jakým komunikuje se čtečkou karet, podle komponent, ze kterých se skládá a podle operačního systému, který karta obsahuje. Schéma takto zvoleného dělení je na Obr 1.1



Obr 1.1 Klasifikace čipových karet

- **Rozdělení podle způsobu komunikace**

Kontaktní čipové karty musí být vloženy do čtečky čipových karet, čtečka musí mít přímé spojení s vodivým kontaktem na povrchu karty. Předávání příkazů, dat a stavů karty se odehrává přes tyto kontaktní body.

Bezkontaktní čipová karta vyžaduje pouze blízké přiblížení ke čtečce. Karta i čtečka obsahují anténu a komunikují na radiových frekvencích. Rychlé bezkontaktní čipové karty pracují na frekvenci 13,56 MHz, podle standartu ISO14443 a umožňují čtení i zápis. Jsou vhodné na masovou identifikaci fyzického přístupu. Většina bezkontaktních čipových karet čerpá z elektromagnetického záření energii potřebnou pro svůj provoz. Vzdálenost, na jakou je nutné kartu přiblížit, se pohybuje od řádově jednotek centimetrů až po 50 cm. [2]

Hybridní karty obsahují dva čipy, jeden s kontaktním rozhraním a druhý s bezkontaktním. Tyto dva čipy nejsou navzájem propojeny.

Karty s kombinovaným rozhraním mají jediný čip obsahující obě rozhraní, kontaktní i bezkontaktní.

- **Rozdělení podle obsažených komponent**

Jak již bylo řečeno, karty mohou být vybaveny pouze paměťovým modulem nebo navíc i procesorem. V prvním případě pak hovoříme o paměťových kartách, v druhém o čipových kartách.

Paměťové karty obsahují dva typy pamětí:

ROM – read-only memory. Do této paměti jsou data uložena výrobcem a později se již nedají změnit. Jsou v ní tedy uložena data, která se v průběhu života karty nemění. Může obsahovat například číslo karty, jméno držitele karty, atd.

EEPROM – electrically erasable programmable read-only memory. Obsah této paměti lze měnit i dodatečně. Ukládají se do ní tedy aplikační data, která se mění s časem. U telefonních karet může EEPROM obsahovat například zbývající čas hovoru. Přistup do této paměti bývá často chráněn PINem (Personál Identity Number).

Bezpečnostní logika kontroluje přístupy k paměti karty a dovoluje čtení a zápis dat. Paměť je přístupná pouze po zadání bezpečnostního čísla. Toto číslo může poskytnout jak držitel karty, tak čtečka karet. Schéma paměťové karty je na obrázku

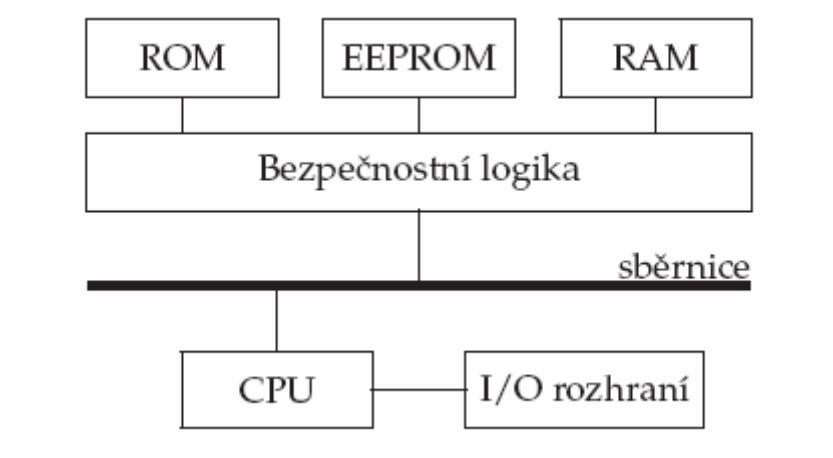
Karty s procesorem mají v sobě kromě pamětí zabudovaní i mikroprocesor. Důležité části těchto karet jsou:

ROM – v této paměti je uložený operační systém karty. Ten se nahraje obvykle při produkci karty a je neměnný.

EEPROM – v paměti EEPROM jsou uloženy aplikace a aplikační data. Tato data nejsou permanentní a lze je vymazávat a přepisovat. Velikost paměti EEPROM se pohybuje od 2kB až po 74 kB.

RAM – random access memory. Tuto paměť používá procesor pro samotný běh funkcí. Paměť RAM je energeticky závislá, takže se vymaže pokaždé, když je odpojena od čtečky. Typickou velikostí paměti RAM je 256 bajtů.

CPU – central processing unit. Jedná se o vlastní mozek karty. Obvykle je to 8-bitový mikroprocesor s rychlostí kolem 5MHz. Proces je zodpovědný za vykonávání jednotlivých instrukcí. Schéma čipových karet je na Obr 1.2



Obr 1.2 Schéma čipové karty

- **Rozdělení dle operačního systému**

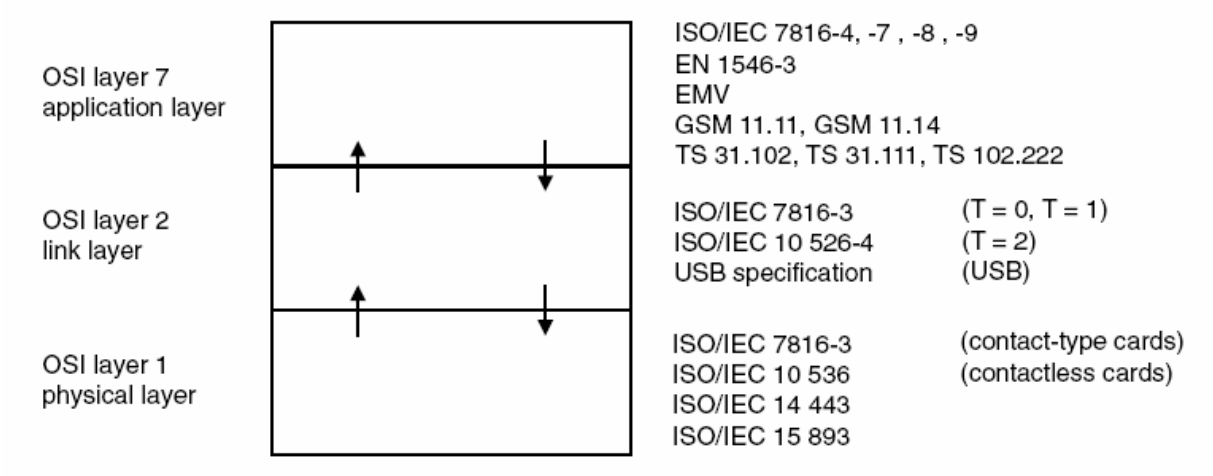
V závislostech na vlastnostech operačního systému lze čipové karty rozdělit na:

- statické čipové karty – umožňují číst data, zapisovat data a pracovat s daty (sem patří velka většina kryptografických karet, které jsou dnes na trhu – ActivCard, Gemplus, Datakey, Schlumberger Cryptoflex, StarCos, SIM karty).
- Dynamické čipové karty – jsou schopny do karty nahrát programový kód a prostřednictvím tohoto kódu měnit chování karty. Tyto karty je dále možno dělit podle nahraného programového kódu. Mezi nejrozšířenější v současné době patří JavaCards nebo MULTOS.

Operační systém je uložen v paměti ROM a zabírá přibližně 16 kB. Stará se o manipulaci se soubory, správu paměti a protokol pro komunikaci se čtečkou.

1.3. Komunikační protokoly

Komunikační proces mezi čtečkou karet a čipovou kartou může být znázorněn za užití vrstvového modelu OSI. Tento model od sebe odděluje dění na I/O lince, logické procesy aktuálního přenosového protokolu a chování aplikací užívajících tento proces. Chování a vzájemná interakce mezi těmito vrstvami jsou specifikovány několika mezinárodními standarty. Vztahy mezi vrstvami jsou znázorněny na Obr 1.3.

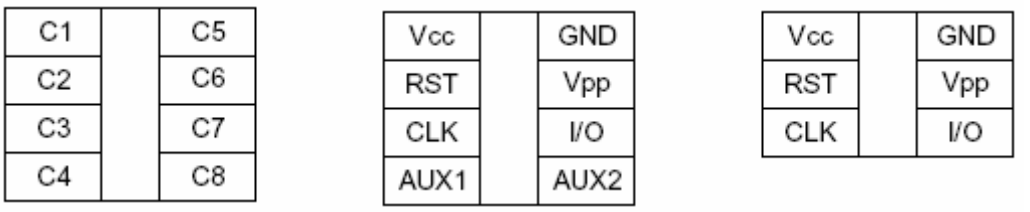


Obr 1.3 Schéma komunikačního procesu[4]

Nejrozšířenějšími komunikačními protokoly jsou protokoly popsané standarty ISO/IEC 7816-3 (pro kontaktní karty), ISO/IEC 14 443 (bezkontaktní karty) a ISO/IEC 7816-4. Tyto protokoly podporuje i čipová karta použitá pro experimentální měření, proto se tato kapitola věnuje pouze těmto standardům. [4]

1.3.1. Elektrická specifikace,fyzická vrstva

Fyzická vrstva čipových karet je definována standardem ISO 7816-2 (pozice kontaktů na kartě) a ISO 7816-3. Tato část standardu ISO 7816 kromě elektrických vlastností čipových karet definuje i transportní (linkovou) vrstvu. [4] [13]. Fyzická stavba čipové karty je znázorněna na Obr 1.4.



Obr 1.4 Konektory čipové karty

Význam jednotlivých konektorů (neboli pinů) je uveden v Tab 1.1.

Kontakt	**Označení**	**Popis**
C1	VCC	napájecí napětí čipu, +5V
C2	RST	reset
C3	CLK	hodinový signál
C4	AUX1	vyhrazen pro budoucí užití
C5	GND	země (referenční napětí), 0V
C6	VPP	programovatelný vstup
C7	I/O	vstup a výstup pro data
C8	AUX2	vyhrazen pro budoucí užití

Tab 1.1 Definice konktaktů čipové karty

Pro další práci jsou důležité elektrické vlastnosti kontaktů C1 a C7. Tyto parametry jsou popsány v tabulkách Tab 1.2, Tab 1.3[13].

Symbol	Minimum	Maximum	Jednotka
VCC	4,75	5,25 200 (10 pro SIM karty)	V mA

Tab 1.2 Elektrické vlastnosti kontaktu VCC za normálních podmínek

Symbol	Podmínky	Minimum	Maximum	Jednotka
V_{ih}	I_{ih} max = +/- 500 μA	2	VCC	V
	I_{il} max = +/- 50 μA	0,7 VCC	VCC	V
V_{il}	I_{il} max = 1mA	0	0,8	V
V_{oh}	I_{ol} max = +/- 100 μA	2,4	VCC	V
	I_{ol} max = +/- 20 μA	3,8	VCC	V
V_{ol}	I_{ol} max = 1mA	0	0,4	V

Tab 1.3 Elektrické vlastnosti kontaktu I/O za normálních podmínek

1.3.2. Linková vrstva

Přenosové protokoly, pracující na linkové se označují jako „T=x“, T symbolizuje pojem „Transmission protocol“, znax „x“ udává verzi transportního protokolu. Tyto protokoly definují formáty bloků dat pro fyzický přenos. Verzí protokolu T je několik, nejčastěji užívanými jsou protokoly T=0, T=1 (pro kontaktní rozhraní), dále pak T=CL (CL=contactless, pro bezdrátové rozhraní), T=USB a další.
Protokol T=0 je znakově orientovaný, odděluje požadavky a odpovědi na požadavky. Protokol T=1 je orientovaný blokově, požadavky a odpovědi nejsou striktně oddělovány, proto je efektivnější. Názorné srovnání protokolů T=0 a T=1 je uvedeno v Tab 1.4 [3] [4]

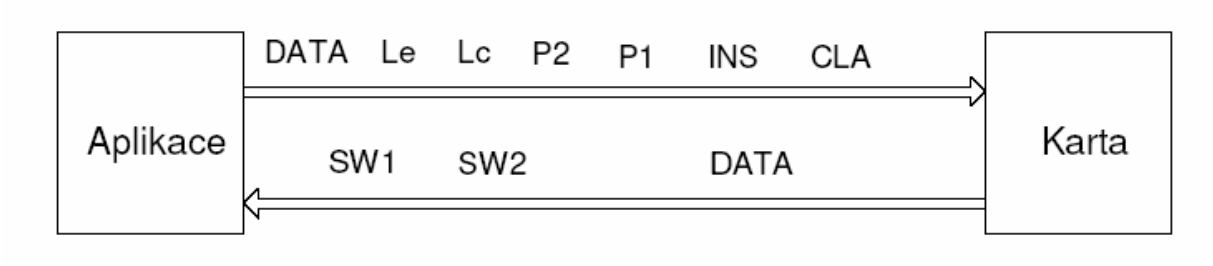
Kriterium	T=0	T=1
Transport dat	Asynchronní Half-duplex Bajtově orientovaný	Asynchronní Half-duplex Blokově orientovaný
Standard	ISO/IEC 7816-3, GSM 11.11, EMV	ISO/IEC 7816-3, EMV
Řetězení bloků	Nepodporováno	Podporováno
Detekce chyb	Paritní bit	Paritní bit, Chybový detekční kód
Paměť požadovaná pro implementaci	300 bajtů	1100 bajtů

Tab 1.4 Srovnání protokolu T=0 a T=1

1.3.3. Aplikační vrstva

Protokol aplikační vrstvy APDU (applications protocols data units) se používá k výměně všech dat mezi čipovou kartou a čtecím terminálem. Protokol APDU je mezinárodně standardizován jako datová jednotka aplikační vrstvy, v modelu OSI odpovídá 7. vrstvě. V případě čipových vrstev je tato vrstva umístěna přímo nad linkovou vrstvou, které odpovídají protokoly T=, protokol APDU je na jejich volbě nezávislý.

Protokol APDU rozlišuje příkazy kartě (command, C-APDU) a odpovědi karty aplikaci (response, R-APDU). Strukturu APDU znázorňuje Obr 1.5.



Obr 1.5 Schéma protokolu APDU

Struktura command APDU je definována jeho prvním bajtem. Ve většině případů je uspořádán podle struktury znázorněné na Obr 1.5 Comand APDU je složen z povinné hlavičky a volitelného těla. Obsahuje:

CLA (1 bajt) Toto povinné pole slouží pro identifikaci instrukční třídy,
INS (1 bajt) Povinné pole, označuje konkrétní instrukci z instrukční třídy definovanou CLA,
P1 (1 bajt) Toto pole definuje parametr 1,
P2 (1 bajt) Toto pole definuje parametr 2,
LC (1 bajt) Nepovinné pole určuje počet bajtů v datovém poli,
Datové pole (proměnný počet, LC bajtů) Nepovinné pole příkazu, obsahuje data,
Le (1 bajt) Volitelné pole specifikuje maximální počet bajtů očekávané odpovědi,
SW1 (1 bajt) Povinné pole obsahující stavové informace (status word),
SW2 (1 bajt) Podobně jako SW1 [4].

Základní příkazy, definované polem CLA a INS jsou:

SELECT – otevírá soubor nebo adresář,
READ - čte soubor,
UPDATE - změní obsah souboru,
AUTHENTICATE - autentizuje kartu okolnímu prostředí,
VERIFY - kontroluje PIN zadaný uživatelem karty. [4]

Komunikaci se čtečkou karet lze tedy shrnou do těchto bodů:

- komunikaci inicializuje čtečka,

- čtečka dále figuruje jako prostředník mezi kartou a počítačem (aplikací),
- komunikace je založena na APDU formátu.

1.4. Java Card

Experimentální část této práce je zaměřena na experimenty s čipovou kartou SIM card a čipovou kartou specifikace Java Card. Tato kapitola se věnuje historii a popisu specifikace Java Card, srovnáním s programovacím jazykem Java, základnímu popisu implementace aplikace na samotnou čipovou kartu a běhovému cyklu takto implementované aplikace

První karta Java Card byla představena roku 1997, je postavena na specifikaci Java Card Platform, která byla vyvinuta firmou Sun Microsystems. Technologie Java Card je založena na otevřeném standartu, usnadňuje přenositelnost softwaru, odstraňuje bariéry vývoje software pro čipové karty. Ostatní čipové karty byly historicky vyvíjeny jako proprietární, instrukce psané pro jeden mikroprocesor nemohly pracovat na jiné platformě bez potřebných modifikací. [5]

Java Card zahrnuje standardní třídy a rozhraní, které umožňují bezpečný běh aplikací na paměťových kartách a dalších jim podobných zařízeních s omezenou pamětí a omezeným přísunem energie. Už z názvu je patrné, že platforma Java Card nějakým způsobem souvisí s programovacím jazykem Java, nabízí se provést srovnání obou technologií. Java Card podporuje pouze podmnožinu funkcí jazyka Java. Mezi nejzásadnější rozdíly platforem Java a Java Card patří:

- Java Card podporuje pouze primitivní datové typy. Dostupné jsou obvykle pouze 8 a 18ti bitové celá čísla (záleží na hostitelském hardware).
- Java Card umožňuje používat pouze jedno rozměrná datová pole. Řetězce a více rozměrná datová pole nejsou podporována.
- Java Card nepodporuje vlákna.

Aplikace vytvořená v prostředí Java Card se stane po kompilaci na bajt-kód (*.class soubor) standardním zdrojovým kódem jazyka Java, jako jakýkoliv jiný Java program. Poté Java Card konvertor, běžící na uživatelském PC, konvertuje tento class soubor na soubor formátu Converted Applet (CAP). CAP soubor obsahuje informace o

třídě, vykonavatelný bajt kód a další informace ve velice kompaktním tvaru. CAP soubor neobsahuje kód, použitelný přímo na procesorech čipových karet, interpretaci dat zajišťuje virtuální stroj umístěný na čipové kartě. Tímto způsobem je zajištěna přenositelnost kódu na čipové karty nezávisle na použitém hardware, princip je stejný jako u klasického jazyka Java. [3].

O běh programu na čipové kartě se starají Java Card Virtual Machine a Java Card Runtime Environment.

Java Card Virtual Machine (JCVM) interpretuje bajt kód, přiděluje pamět a vykonává instrukce. Tento virtuální stroj umožňuje napsat jednu Java Card aplikaci použitelnou na různých fyzických platformách

Java Card Runtime Environment (JCRE) může být obecně popsáno jako kompletní operační systém, obsahující virtuální stroj popsaný výše, rozhraní a systémové funkce.

Protože čipové karty jsou externě napájeny pouze když jsou používány, JCRE (a všechny aplikace běžící v něm) má unikátní běhový cyklus. JCRE je inicializováno jen jednou během životního cyklu karty. Poté je stav JCRE i stav všech aplikací na kartě uložen do permanentní paměti, typicky do paměti typu EEPROM. V okamžiku, kdy je napájení čipové karty odpojeno či přerušeno, běh virtuálního stroje je přerušen. Při obnovení napájení je virtuální stroj a stav všech aplikací obnoven z těchto uložených stavů.

Aplikace Java Card jsou nazývány jako aplety. Protože karty podporují více apletů, každá instance apletu je označena unikátním aplikačním identifikátorem (AID). Když chce terminál komunikovat s konkrétním apletem, odešle příkaz SELECT APDU obsahující vhodné AID. JCRE směruje příkaz SELECT a další příkazy k vybranému apletu, případně pokud je to nutné, aplet inicializuje. [2] [6]

1.5. SIM Card

SIM (Subscriber Identity Module = účastnický identifikační modul) karta slouží k identifikaci a autentizace uživatelů v mobilních sítích.

SIM karta je čipová karta s mikroprocesorem, obsahuje následující součásti:

- CPU,
- ROM paměť,

- RAM paměť,
- EEPROM paměť,
- síťové komunikační rozhraní. [12]

Vyskytují se tři rozdílné formy provedení SIM karet se stejnými funkcemi. První SIM karty měly velikost standardní kreditní karty (85.60 mm × 53.98 mm × 0.76 mm). Tyto karty byly postupně nahrazovány kartami Mini SIM (25 mm × 15 mm × 0.76 mm) a Micro SIM (15 mm × 12 mm × 0.76 mm). [12]

SIM karta provádí následující funkce:

- identifikace účastníka,
- autentizace účastníka,
- úložiště uživatelských dat,
- běh aplikací

Nejcitlivější informace související se SIM kartou jsou kryptografické algoritmy A3, A8, tajné proměnné Ki, PIN, PUK a Kc. Algoritmy A3 a A8 jsou na SIM kartu uloženy během procesu výroby, pro uživatele jsou nedostupné. PIN kód je nastaven uživatelem karty, PUK kód je uložen v databázi mobilního operátora. Klíč Kc je odvozen v procesu šifrování z Ki. Každá SIM karta je jednoznačně identifikována unikátním číslem IMSI (International Mobile Subscriber Identity). [12]

Souborový systém je uložen v interní EEPROM paměti a je chráněn bezpečnostními prvky na kartě. Souborový systém je organizován v hierarchické stromové struktuře, obsahuje tři typy prvků:

- Master File (MF) – kořen stromové struktury,
- Dedicated File (DF) – adresáře, mohou obsahovat další adresáře nebo základní soubory (EF), existují dva druhy – $DF_{TELECOM}$ obsahuje uživatelská data jako například telefonní seznam a je přístupný jiným aplikacím, DF_{GSM} obsahuje servisní údaje a tajné klíče,
- Elementary File (EF) – různé typy formátovaných dat [12].

Na SIM kartě jsou uloženy následující bezpečnostní a identifikační data: MSISDN (Mobile Subscriber Integrated Services Digital network Numer = mezinárodní

telefonní číslo), Ki, PIN, PUK, TMSI (Temporary Mobile Subscriber Identity = náhodně přidělené identifikační číslo), LAI (Location Area Identity = identifikátor oblasti), ICC-ID (Integrated Circuit Card ID = evidneční číslo SIM karet operátorů). [12]

Přístup na SIM kartu a její vlastní použití chrání 4 až 8 místný kód PIN (Personal Identification Number = osobní identifikační číslo). PIN je nastaven provozovatelem GSM sítě na výchozí hodnotu, která může být změněna prostřednictvím mobilního telefonu. Jestliže je ochrana PIN kódem aktivní, musí být vložen při každém zapnutí telefonu. Pokud je kód vložen třikrát po sobě špatně, SIM karta je zablokována a pro další provoz vyžaduje kód PUK (Personal Unblocking Key – kód pro odblokování) definovaný provozovatelem sítě.

Použití pokročilých funkcí mobilního telefonu chrání kód PIN2, jeho odblokování při několikanásobném špatném vložení umožňuje kód PUK2. [12]

Kromě těchto bezpečnostních a servisních dat, mohou být na sim kartě uloženy textové zprávy, telefonní čísla a dále aplikace napsané v jazyce Java Card, umožňující uživatelům např. obsluhovat bankovní účet.

2. Bezpečnost čipových karet

Podstatným rysem čipové karty je, že poskytuje bezpečné prostředí pro data a programy. Je samozřejmě prakticky nemožné nakonfigurovat jakýkoliv systém, a tedy i čipovou kartu, která by poskytovala dokonalé zabezpečení proti všem útočníkům a útokům. Je-li snaha o prolomení ochrany na dostatečně vysoké úrovni, je možné získat přístup ke každému systému a manipulovat s ním. Potenciální útočník musí provést analýzu, zda hodnota takto získaných dat je vyšší, než náklady vynaložené na dosažení svého cíle, nabourání systému.

Bezpečnost čipové karty lze rozdělit do tří oblastí. První je fyzická bezpečnost, chránící data před mechanickými útoky, druhou oblast tvoří softwarová bezpečnost, nedílnou součástí bezpečnosti čipových karet je i třetí složka, bezpečnost periferií, tedy čteček a terminálů. Pro dosažení vyššího stupně bezpečnosti je nezbytná přítomnost bezpečnosti všech těchto složek a jejich spolupráce.

Čipové karty jsou populárním cílem útočníků z několika důvodů:

- Úspěšné útoky umožňují zneužití prostředků nebo informací a jsou finančně vysoko oceněné.
- Čipové karty jsou laciné a přístupné, útočník si lehce obstará cvičné vzorky.
- Čipové karty jsou přenosné, útočník je může lehce přenést do atakovaného prostředí a použít.

Tato kapitola se věnuje popisu možných hrozeb pro čipové karty, popisem typů útoků a dále možností softwarového zabezpečení čipových karet – implementací kryptografických protokolů.

2.1. Hrozby pro čipové karty

Čipovým kartám hrozí tyto hrozby:

- **confidentality** – neautorizovaný přístup k informacím,
- **integrity** – neautorizovaná modifikace informací,

- **authenticity** – neautorizované používání služeb.

Existují 3 skupiny útoků

- fyzické,
- logické,
- postranní.

2.1.1. Fyzické útoky

Všechny funkce čipové karty jsou realizované na jednom čipu, který je možné zkoumat a provázet reverzní inženýrství. Na to je potřebné kvalitní laboratorní a výkonné vybavení. Při útocích se využívá analýza a modifikace hardware. Příkladem fyzického útoku je útok pomocí sond.

Prvním krokem tohoto útoku je odstranění plastového krytu tak, aby čipy karty nebyly poškozeny. K takto odkryté kartě lze napojit vodiče na datovou sběrnici nebo pozorovat mikroskopem paměťové buňky. Tuto práci lze snadno provést pomocí pozorovací stanice. Pozorovací stanice se skládá z mikroskopu s mikromanipulátory umožňujícími připojit n a povrch čipu detektory.

Aby si útočník ulehčil pozorování čipu, může se pokusit snížit taktovací frekvenci čipové karty. Takto lze snáze pozorovat postupné stavy, kterých čipová karta v průběhu výpočtu dosahuje.

Modernější čipové karty bývají chráněny proti invazím pasivní vrstvou. Často se jedná o nějaký kryt čipu, který brání snadnému přístupu k němu. Aby bylo možné čip pozorovat, je potřeba takovou vrstvu nejdříve odstranit. Odstranění pasivní vrstvy však nebývá příliš velký problém. Některé čipové karty jsou proto navíc vybaveny různými detektory. Typickým příkladem může být metalická mřížka, která je umístěna nad čipem.Poškození této mřížky způsobí ztrátu citlivých dat. Podobně může čip monitorovat taktovací frekvenci a odmítat provádět výpočty pokud je frekvence příliš vysoká nebo nízká, Pro ochranu čipové karty na ní mohou být umístěny i další druhy senzorů, jako jsou UV senzory, světelné senzory apod. [7]

2.1.2. Logické útoky

Čipové karty mají jeden komunikační kanál, přes který si vyměňují informace s terminálem (čtečkou). Jelikož čipové karty jsou podobné malým počítačům, podporují velké množství příkazů vložených různými výrobci. Kvůli takovéto komplexnosti se může stát, že se vyskytne chyba, který se neprojeví při normálním používání ani při bezpečnostních testech, Logické útoky zneužívají takovéto chyby, aby zmátly čivovou kartu a získali tajné informace, nebo aby je modifikovali. Společným znakem této skupiny útoků je nedestruktivní povaha, lehkost reprodukce a jednoduchost získání prostředků (čipová karta, čtečka, PC, manuály, standardy). [7]
Mezi používané techniky logických útoků patří:

- scan příkazů (skryté příkazy),
- scan souborového systému,
- neplatné a nevhodné požadavky, z toho plynoucí přetečení zásobníku,
- kryptografická analýza a zneužití protokolu.

2.1.3. Postranní kanály

Těmto útokům je věnována samostatná kapitola

2.2. Kryptografické algoritmy

Čipové karty neposkytují vyšší bezpečnost samy o sobě, ale využívají k tomu navržené kryptografické algoritmy, které jsou implementovány v jejich čipu. Jedním z nejrozšířenějších šifrovacích algoritmů současné doby je Advanced Encryption Standard (AES). Tento algoritmus zatím standardní kryptoanalýza neprolomila, takže se jeví jako dostatečně bezpečný. Jeho popisu se věnuje následující kapitola.

2.2.1. Historie šifrovacích algoritmů

Prvním široce rozšířeným šifrovacím algoritmem se stal DES (Data Encryption Standart), který vznikl z iniciativy Ministerstva obchodu USA. Hlavním důvodem vzniku DES byl přechod na elektronický bankovní systém v sedmdesátých letech

v USA. DES vstoupil v platnost v roce 1977. Tento algoritmus šifroval 64 bitové bloky dat s délkou klíče 56 bitů.

Koncem minulého století byly již známy efektivní útoky, které byly schopny s poměrně malými náklady DES prolomit. Bylo ukázáno, že DES lze luštit s nákladem pouhých 10 tisíc dolarů a že tedy není bezpečný ani před menšími organizacemi. Proto bylo nutné pro šifrovací algoritmus navrhnout nový standard, který by neobsahoval chyby DES, a byl tak mnohem bezpečnější.

Vítězem výběrového řízení americké organizace NIST (National Institute of Standarts and Technology) se stal algoritmus Rijndael navržený týmem belgických kryptologů. Na základě tohoto algoritmu byl v roce 2001 vyhlášen nový standard pro symetrické šifrování pod názvem AES (Advanced Encryption Standard), který je v následujících odstavcích popsán.[2]

2.2.2. Advance Encryption Standard

AES je symetrický blokový šifrovací algoritmus s šířkou bloku 128 bitů a možnými délkami klíčů 128, 192 nebo 256 bitů. Vstupem algoritmu je 128 bitový blod dat a klíč. Tento blok dat je v rundách modifikovám čtyřmi transformacemi:ByteSub, ShiftRow, MixColumn a AddRoundKey. Počet kol (dále značeny Nr) je závislý na délce klíče (Tab 2.1)

	$N_k = 4$	$N_k = 6$	$N_k = 8$
N_r	10	12	14

Tab 2.1 Hodnoty počtu kol pro různé délky klíčů

Vstupní blok dat se označuje jako stav. Ten lze zapsat jako matici 4 x 4 bajtů (Obr 2.1). Při implementaci se stav reprezentuje jako pole bajtů. Stav je pak mapován do pole po sloupcích v pořadí $a_{0,0}$, $a_{1,0}$, $a_{2,0}$, $a_{3,0}$, $a_{0,1}$, $a_{1,1}$, $a_{2,1}$, a_{31},…

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{0,3} \end{pmatrix} \qquad \begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{0,3} \end{pmatrix}$$

Obr 2.1 Reprezentace stavu a 128-bitového klíče pomocí matice bajtů

Stejně tak se i klíč mapuje do pole po sloupcích. Počet sloupců klíče se značí Nk, počet sloupců stavu je Nb = 4.

Runda šifry

Jak již bylo uvedeno, jedna runda se skládá ze čtyř operací. Vstupní blok dat je tedy v závislosti na délce klíče transformován vhodným počtem rund. V každé rundě je bloku přičten RoundKey, který je odvozen z klíče. V pseudokódu lze rundu zapsat takto:

```
Round (State, RoundKey) {
      ByteSub (State);
      ShiftRow (State);
      MixColumn (State);
      AddRoundKey (State, RoundKey)
}
```

V posledním kole je vynechána operace MixColumn.

```
Round (State, RoundKey) {
      ByteSub (State);
      ShiftRow (State);
      AddRoundKey (State, RoundKey)
}
```

V následujícím textu jsou zmíněny některé matematické operace, které algoritmus používá. Jejich popis je nad rámec této práce, jsou popsány v dokumentu [9], ze kterého je čerpán i následující popis.

Operace `ByteSub`

Tato transformace je nelineární substitucí bajtů vstupního bloku. Je to jediná nelineární operace v AES. Substituční tabulka (S-box) je konstruována kompozicí dvou operací:

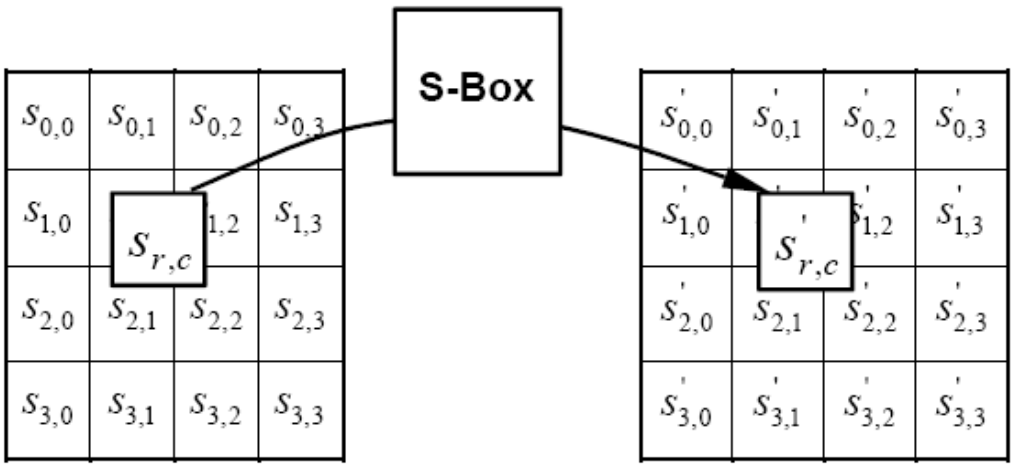
1. multiplikativní inverzí v GF(2^8),
2. afinní transformací definovanou maticí:

$$\begin{pmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

.

Inverzní operaci k ByteSub získáme tak, že nejdříve aplikujeme inverzní afinní transformaci a výsledek invertujeme v GF(2^8)

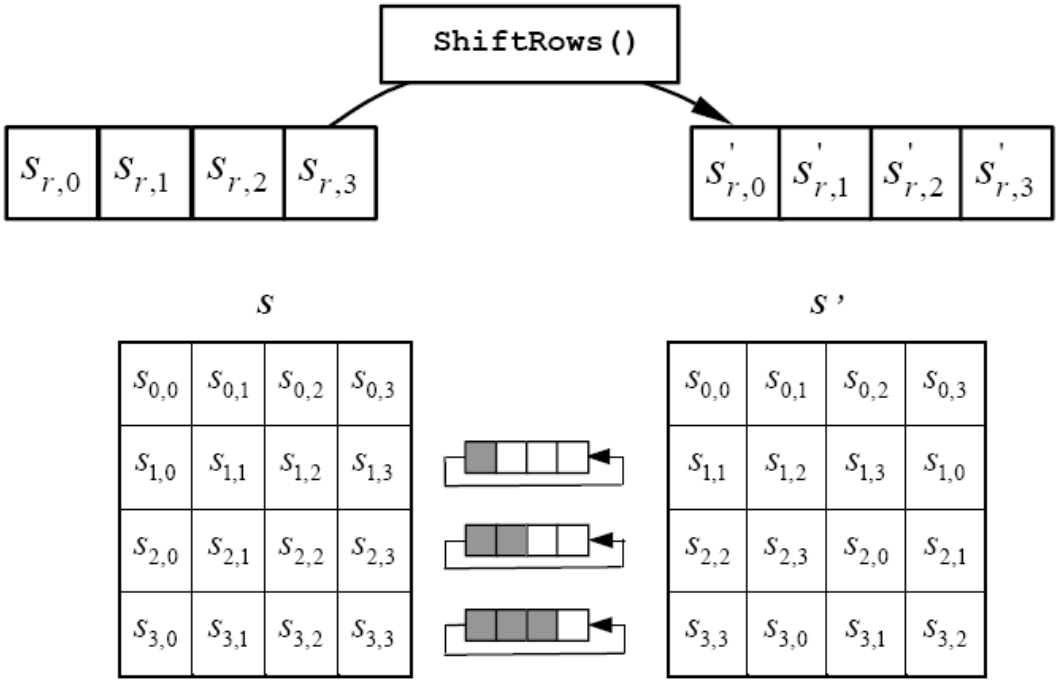
Výsledný efekt operace ByteSub je znázorněna na Obr 2.2.



Obr 2.2 Operace ByteSub

Operace `ShiftRow`

V této operaci jsou bajty v řádku stavu cyklicky posouvány o různé hodnoty. Nultý řádek stavu není posouván, první je posouván o 1, druhý o 2 a třetí o 3. Schéma je zobrazeno na Obr 2.3



Obr 2.3 Operace ShiftRows

Operace `MixColumn`

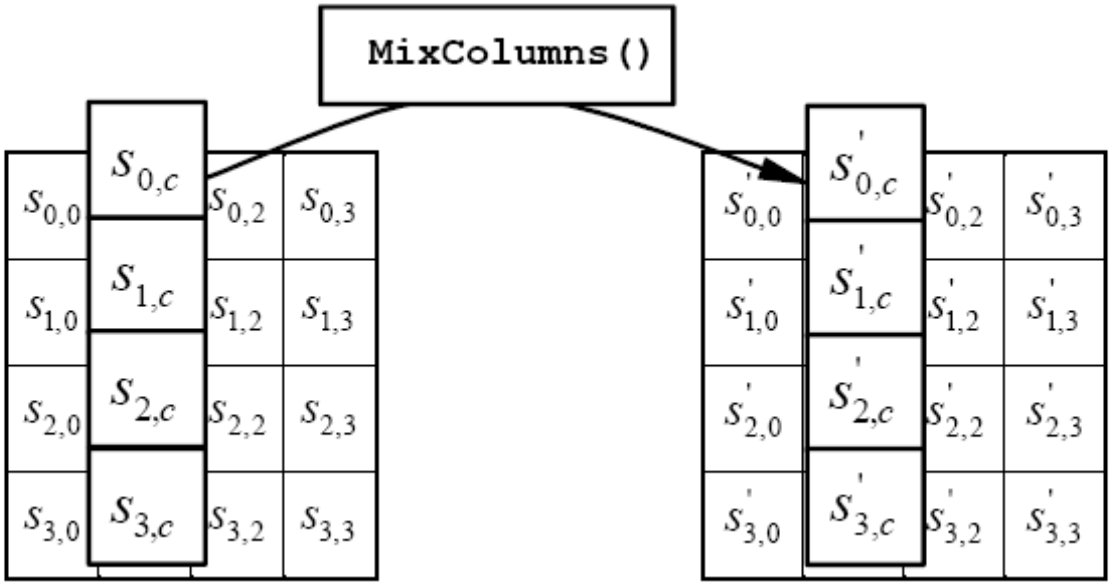
Transformace MixColumn uvažuje sloupce stavu jako polynomy nad GF(2^8) a násobí je modulo $x^4 + 1$ s fixním polynomem a(x) definovaným

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \qquad (2.1)$$

Tento polynom je nesoudělný s $x^4 + 1$, a proto ho lze invertovat. Násobení fixním polynomem a(x) lze zapsat jako násobení maticí $s'(x) = a(x) \otimes s(x)$

$$\begin{pmatrix} S'_{0,c} \\ S'_{1.c} \\ S'_{2.c} \\ S'_{3.c} \end{pmatrix} = \begin{pmatrix} 02 & 06 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} S_{0,c} \\ S_{1.c} \\ S_{2.c} \\ S_{3.c} \end{pmatrix} \qquad (2.2)$$

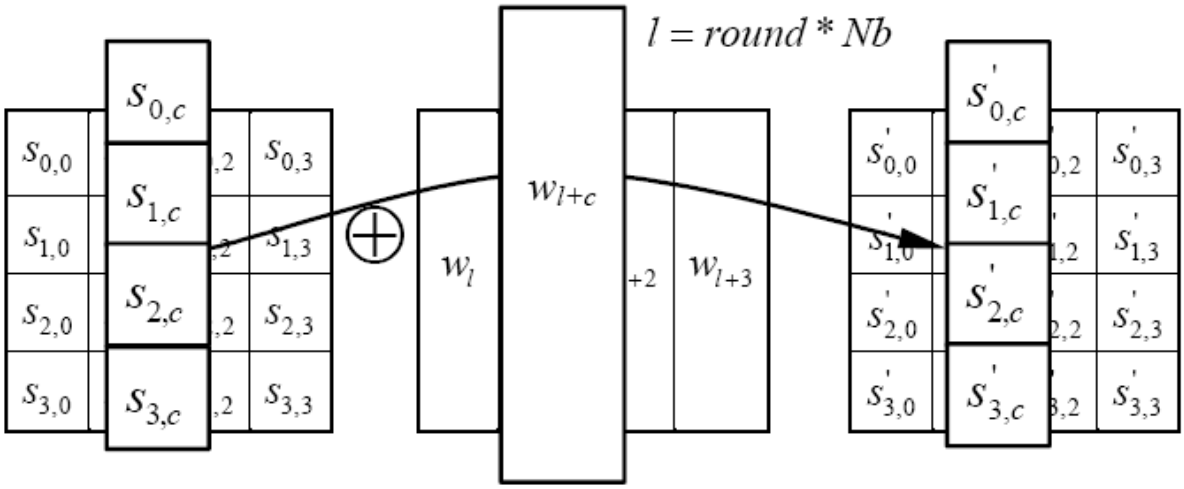
Schéma operace MixColumns je znázorněno na Obr 2.4



Obr 2.4 Operace MixColumns

Operace `AddRoundKey`

V této operaci je ke stavu přičten rundovní klíč. Přičtením se myslí jednoduchou XOR operaci. Rundovní klíč je odvozen z šifrovacího klíče a jeho délka je rovna délce bloku.



Obr 2.5 Operace AddRoundKey

Expanze klíče

Rundovní klíče se odvozují z šifrovacího klíče pomocí operace expanze klíče. Expandovaný klíč je tvořen čtyřbajtovými slovy a je označován *W[Nb*(Nr + 1)]*. Prvních *Nk* slov obsahuje šifrovací klíč. Zbytek slov je definován rekurzivně ze slov s menším indexem. Rundovní klíč *i* je dán slovy *W[Nb*i]*až *W[Nb*(i+1.]*.

2.2.3. Implementace AES na čipových kartách Java Card

Java Card API od verze 2.2 obsahuje efektivní implementaci vybraných kryptografických algoritmů, včetně algoritmu AES. Pro implementaci AES algoritmu využijeme za Java Card API balíčky javacard.framework, javacard.security a javacardx.crypto. [5]

Javacard.framework poskytuje důležité rozhraní a třídy pro vytváření, komunikaci a práci s aplety Java Card. Důležitá je třída APDU obsahující metody pro komunikaci s terminálem či čtečkou.

Javacard.security poskytuje třídy a rozhraní, které obsahují funkce pro implementaci bezpečného prostředí na platformě JavaCard. Obsahuje rozhraní pro několik kryptografických algoritmů, pro naši potřebu je důležité rozhraní AESkey obsahující 16/24/32 bajtové klíče pro výpočty na základě algoritmu Rijndael.

Javacardx.crypto obsahuje veřejně dostupnou metodiku pro implementaci bezpečnostního a šifrovacího systému. Obsahuje jediné rozhraní KeyEncryption se dvěma metodami getKeyCipher a setKeyCipher a jedinou třídu Cipher (šifra).
Třída Cipher poskytuje metody pro zakódování a dekódování zpráv, k tomuto účelu obsahuje statické proměnné reprezentující šifrovací klíče jako například ALG_AES_BLOCK_128_CBC_NOPAD, konstruktor Cipher(), metody doFinal (generuje zašifrovaný/dešifrovaný výstup ze všech/posledních vložených dat), getAlgorithm (vrátí šifrovací algoritmus), getInstance (vytvoří instanci objektu Cipher vybraného algoritmu), init (inicializuje objekt Cipher s daným klíčem). [5]

Jednoduchý kód algoritmu AES implementovaný pro Java Card je znázorněn v následujících ukázkách zdrojového kódu.

```
//globální proměnné
AESKey aesKey;
      Cipher cipherAES;
      RandomData random;
      static byte a[];
      final short dataOffset = (short) ISO7816.OFFSET_CDATA;

//konstruktor
private CryptoAES (byte bArray[], short bOffset, byte bLength)
      {
            aesKey = (AESKey) KeyBuilder.buildKey(KeyBuilder.TYPE_AES,
KeyBuilder.LENGTH_AES_128, false);
            cipherAES                                                    =
Cipher.getInstance(Cipher.ALG_AES_BLOCK_128_CBC_NOPAD, false);
            a = new byte[ (short) 128];
            random.generateData(a, (short)0, (short)128);
            aesKey.setKey(a, (short) 0);
            register(bArray, (short) (bOffset + 1), bArray[bOffset]);
      }

//hlavní metoda třídy, přijímá data prostřednictví protokolu
//APDU, šifruje, výstup opět předává protokolu APDU
private void doAES(APDU apdu)
      {

            byte b[] = apdu.getBuffer();

            short          incomingLength          =          (short)
(apdu.setIncomingAndReceive());
            if            (incomingLength            !=            24)
ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);

a[] automatically

            cipherAES.init(aesKey, Cipher.MODE_ENCRYPT);
            cipherAES.doFinal(b, (short) dataOffset, incomingLength, a,
(short) (dataOffset + 24));
            cipherAES.init(aesKey, Cipher.MODE_DECRYPT);
```

```
            cipherAES.doFinal(b,    (short)    (dataOffset    +    24),
incomingLength, a, (short) (dataOffset + 48));


            apdu.setOutgoing();
            apdu.setOutgoingLength((short) 128);
            apdu.sendBytesLong(b, (short) dataOffset, (short) 128);
      }
```

Funkčnost apletu je možno vyzkoušet na simulátorech Java Card CREF a JCWDE, které jsou součástí balíčku Java Card Development Kit. Výše uvedený zdrojový kód byl v simulátorech úspěšně testován.

Zdrojový kód appletu je nutné nejprve zkompilovat na .class soubory. Zdrojový kód Appltu je překládán pomocí překladače *javac* dodávaného s JDK firmy Sun. Kompilace apletu CryptoAES:

```
javac -g -target 1.1 -source 1.2 -cp %JC21_HOME%\lib\api21.jar
krypto_aes\*.java
```

Parametr -g určuje, že soubor má být přeložen se všemi informacemi pro debug. Parametry -target a -source instruují překladač, aby zachovával kompatibilitu překladu se staršími verzemi jazyka, což je nutné pro nástroj konverze. -cp určuje classpath, tj. soubory a knihovny, vůči nimž má být kód překládán. Zde je nutné uvést cestu k souboru api21.jar či api.jar, který je součástí Java Card Development Kitu (v adresáři bin či lib). Ten obsahuje Java Card API, které applet na kartě využívá, a je nutné, aby vůči němu byl soubor přeložen.

Kompilací získaný .class soubor (popř. soubory) je nutné zkonvertovat. Konverze je proces, při němž se ještě předtím, než je applet nahrán na kartu, vykoná preprocessing, který virtuální stroj na pc při zavádění normální javovské třídy vykonává. To se týká i kontrol, zda přeložený soubor splňuje požadavky Java Card platformy a zda volá pouze podmnožinu příkazů jazyka Java na Java Card platformě dostupných.

Výsledkem konverze je .cap soubor, který se svou podstatou podobá jar souboru – také jde o kontejner tříd. Soubor .cap je již applet připravený k nahrání na kartu. Konverzi zajišťuje nástroj Converter, který je součástí Java Card Development Kitu.

V konfiguračním souboru je možná nástroji Converter předat parametry konverze. K nim patří například přiřazení AID instanci appletu a jeho balíčku.

Následně lze .cap soubor nahrát na kartu k tomu určeným postupem. Na kartě vznikne Load soubor. Z něho poté lze na kartě vytvořit instanci appletu, kterou lze vybrat příkazem SELECT. [14]

3. Útoky postranními kanály

V této kapitole se budeme zabývat útoky na kryptografické algoritmy, které jsou vedeny postranními kanály. Objasníme, na jakém principu tyto útoky fungují, v čem spočívá jejich síla a jakých nedostatků kryptografických zařízení využívají. Dále pak rozebereme jednotlivé druhy těchto útoků.

3.1. Úvod

Útoky postranními kanály jsou založeny na informacích z postranních kanálů, které může útočník získat z kryptografického zařízení v průběhu jeho činnosti.

Na kryptografické algoritmy můžeme nahlížet dvěma způsoby. První způsob nahlíží na kryptografický algoritmus jako na abstraktní matematických objekt. Jedná se tedy o pouhou matematickou transformaci parametrizovanou klíčem, která daný vstup přepracuje na určitý výstup. Druhý pohled nahlíží na vlastní implementaci algoritmu. I kryptografické algoritmy totiž musí být v konečném důsledku implementovány programem, který běží na konkrétním procesoru a v nějakém konkrétním prostředí. Tato implementace pak bude mít samozřejmě svoje charakteristiky, které nejsou při matematickém pohledu vůbec uvažovány.

Klasická kryptoanalýza pracuje s matematickým modelem kryptografického algoritmu. Útoky touto metodou jsou proto vedeny na základě jedné z následujících znalostí:

- znalosti zašifrované zprávy (ciphertext-only attacks),
- znalosti otevřené i zašifrované zprávy (know plaintext attacks),
- možnosti volit zprávy pro zašifrování a pak analyzovat takto zašifrovanou zprávu (chosen plaintext attacks).

Oproti tomu kryptoanalýza postranních kanálů využívá implementačních charakteristik pro získání tajných parametrů, které jsou součástí počítání. Kryptografické zařízení poskytuje například lehce měřitelnou informaci o čase, jak dlouho trvala každá jednotlivé operace, kolik energie ta která operace spotřebovala a jiné. Kryptografická zařízení mívají často také dodatečné neúmyslné vstupy jako například napětí, které může být modifikováno, aby způsobilo předvídané změny

v zařízení. Útoky postranními kanály využívají některé nebo všechny tyto informace pro získání klíče z kryptografického zařízení.
Analýza založena na postranních kanálech je sice méně obecná, jelikož je specifická pro každou implementaci, ale na druhou stranu je mnohem mocnějším nástrojem než klasická kryptoanalýza.

Velký zájem o techniky analýzy postranních kanálů způsobuje fakt, že útok tímto způsobem může být implementovaný pomocí snadno dostupného hardwaru, jehož cena se pohybuje od stovek do tisíců dolarů. Čas potřebný pro útok pak závisí na použitých technikách. [2]

3.2. Chybové postranní kanály

Útok vedený tímto způsobem je založený na generování chyb v zařízení, které vlastní útočník nebo na standardních chybách, které se běžně mohou vyskytnout. Výsledky z těchto chybných výpočtů poskytují další informace o tajných parametrech výpočtu. Vynutit chybu na zařízení lze například změnou napětí, změnou taktu procesoru nebo aplikováním nějakého druhu ozařování.

3.2.1. Typy chyb

Chyby můžeme dělit z několika následujících hledisek.

Permanentní vs. Dočasná: permanentní chyba ovlivní (poškodí) kryptografické zařízení trvale tak, že se v dalších výpočtech nebude chovat správně. Typickým příkladem takové chyby může být zablokování paměťové buňky na konstantní hodnotu. Dočasnou chybou se myslí, že je zařízení ručeno takovým způsobem, kdy nepracuje korektně během výpočtu. Tímto rušením se však zařízení nepoškodí. Zařízení můžeme rušit radioaktivním paprskem, abnormálně velikým napětím, nebo mu můžeme měnit taktovací frekvenci na abnormální hodnoty.

Místo výskytu: Některé útoky vyžadují, aby se chyba vyskytla na konkrétním místě

Čas výskytu: Podobně některé útoky vyžadují, aby chyba nastala v určitém okamžiku výpočtu.

Je zjevné, že chybový model, který se uvažuje během kryptoanalýzy, hraje důležitou roli týkající se proveditelnosti útoku. Jsou dva způsoby, jak vést kryptoanalýzy založené na chybových modelech. První způsob se snaží vynutit konkrétní typ chyby v zařízení. Druhý předpokládá (méně či více realisticky) chybový model a snaží se využít tento model k prolomení systému. Tento model se však nezatěžuje tím, jak tento typ chyby v praxi na zařízení vynutit. [2]

3.2.2. Techniky vynucení chyby

Chybu na zařízení můžeme vynutit ovlivňováním jeho prostředí. Je známo mnoho chyb, které můžeme s patřičným vybavením na kryptografickém zařízení vynutit. Zde je stručný seznam technik, které lze použít:

Napětí: čipové karty musí podle standardu ISO tolerovat výkyvy napětí v rozsahu 10% (4,5V – 5,5V, standard pro napětí je 5V). V tomto rozsahu by měla čipová karta pracovat bezchybně. Překročení této hranice může způsobit chyby ve výpočtu. Karta přitom nebude zničena.

Taktovací frekvence: podobně jako v předchozím bodě definuje standard i referenční takt karty s nějakou tolerancí, ve které ještě karta bude pracovat bezchybně. Snížení nebo zvýšením frekvence nad tyto hodnoty opět způsobuje vynucené chyby, aniž by se karta zničila.

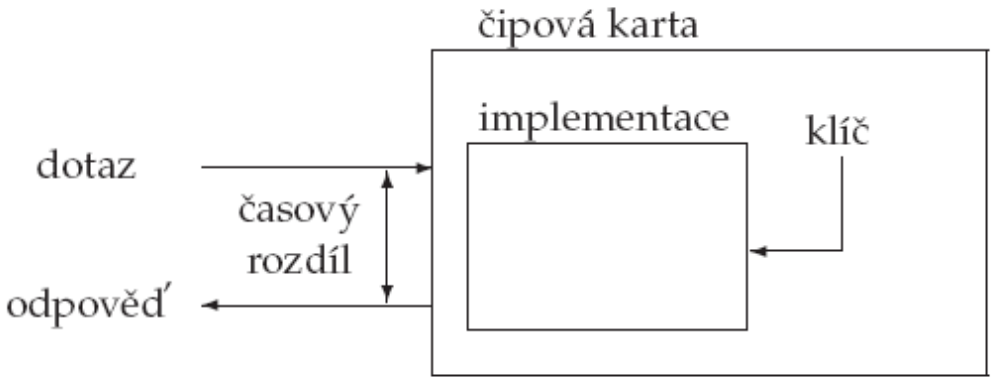
Teplota: čipové karty také reagují chybou na prostředí s extrémními teplotami. Tato technika se ovšem moc nepoužívá.

Záření: tradičně jsou také uváděny útoky pomocí mikrovln. Útočník vloží kartu do mikrovlnné trouby, ta pak během výpočtu vykazuje chybovost. Je prokázané, že správně nasměrované zření ovlivňuje výpočet karty.

Světlo: použitím intenzivního zdroje světla lze změnit hodnoty jednotlivých bitů v SRAM. Stejnou technikou lze zasahovat do instrukcí. Tak lze například změnit instrukce skoku. [2]

3.3. Časové postranní kanály

Čas běhu programu je obvykle zvažován jen jako omezení, které je nutno minimalizovat. Zkoumáním času běhu kryptografického zařízení lze získat informace, které mohou vést k odhalení tajného klíče. Proto se implementace kryptografických algoritmů nesnaží za každou cenu čas minimalizovat, ale jedním z hlavních cílů je, aby čas běhu algoritmu nebyl závislý na vstupu (klíči a zprávě).



Obr 3.1 Schéma časové analýzy

Při časovém útoku má útočník k dispozici množinu zpráv, která mají být zpracovány kryptografickým zařízením. Pro každou z nich má útočník také odpovídající čas běhu. Úkolem útočníka je získat tajné parametry výpočtu. Jelikož taktovací frekvence je čipové kartě poskytnuta prostřednictvím terminálu, lze snadno provádět precizní měření času, jaký karta potřebuje k provedení kryptografické operace. [2]

3.4. Výkonové postranní kanály

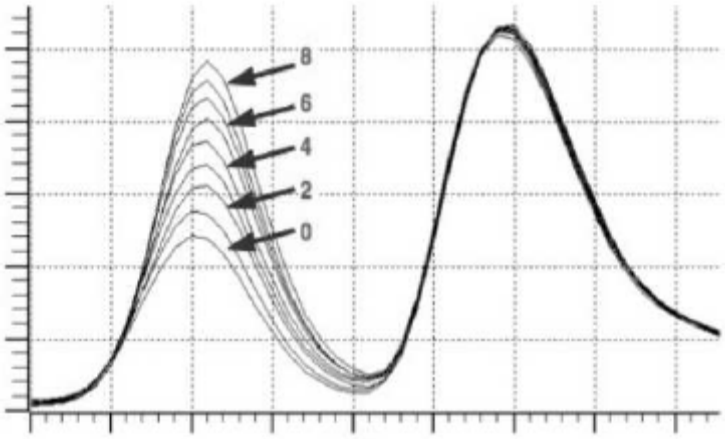
Energie, kterou kryptografické zařízení spotřebuje v průběhu výpočtu, poskytuje mnoho informací nejen o operacích, jaké byly použity, ale také o parametrech, které se účastnily výpočtu. Tato idea je společná pro jednoduchou i diferenční výkonovou analýzu.

Stejně jako taktovací frekvence je poskytována čipové kartě i energie. Proto není příliš obtížné měřit spotřebovanou energii, kterou karta využije při výpočtu. Pro jednoduché měření postačí sériově zapojený malý rezistor s napěťovým nebo zemním vstupním kontaktem. [2]

3.4.1. Jednoduchá výkonová analýza

Jednoduchá výkonová analýza (simple power analysis – SPA) je technika zahrnující přímou interpretaci měřené spotřeby energie zaznamenané během kryptografických operací. Z jednoduché výkonové analýzy můžeme odvodit informace o operacích nebo o klíči, který zařízení používá během výpočtu.

Útočník pozoruje přímou spotřebu kryptografického systému v čase. Množství spotřebované energie závisí na instrukcích, které jsou prováděny. Díky výkonové analýze lze identifikovat větší část kryptografických algoritmů, které mají podobné vlastnosti jako například rundy u AES. Výkonová analýza může být použita k prolomení RSA implementace nalezením rozdílů mezi operacemi násobení a umocňování. Podobně lze v mnoha implementacích blokových šifer (včetně AES) vysledovat viditelné rozdíly mezi operacemi (permutace, bitové posunutí), což může útočníkovi poskytnout dostatek informací k prolomení implementace šifry.



Obr 3.2 Průběhy instrukce pro různé Hammingovy váhy operandu [2]

V článku [8] je prezentován úspěšný SPA útok na AES. Útok je zaměřen na expanzi rundovních klíčů. Předpokladem je velice dobrá znalost výkonnostních charakteristik kryptografického zařízení – je totiž nezbytné, aby byl útočník na základě výkonnostní analýzy schopný zredukovat počet možných hodnot některých mezivýsledků expanze klíče.

Pokud je například útočník schopen podle křivky výkonu zjistit (Obr 3.2), že je Hammingova váha konkrétního mezivýsledku rovna třem, pak se počet možných hodnot mezivýsledku redukuje z 2^8=256 na 56.

Jelikož dokáže SPA odhalit sekvenci vykonávaných instrukcí, může prolomit kryptografickou implementaci, ve které pořadí vykonávaných instrukcí závisí na datech, které jsou ve zpracování. Například:

Porovnání: porovnání řetězců nebo paměťových bufferů běžně obsahuje podmíněnou větev, která se vykoná při nalezení první neshody. Tato podmíněná větev se obvykle projeví při výkonnostní (někdy také časové) analýze.

Násobení: standardní násobící obvody prozrazují na základě výkonnostní analýzy velmi mnoho o parametrech násobení. Množství informací, které poskytuje analýza výkonu, záleží na návrhu násobiče. Téměř vždy však existuje velké provázání hodnotami a Hammingovou váhou operandů.

Umocňování: standardní umocňovací funkce neprochází v cyklu exponent a provádí umocnění na druhou v každém průchodu spolu s dodatečným násobením, jestli je bit exponentu roven1. Exponent může být odhalen za předpokladu že umocňování a násobení mají jiné výkonnostní charakteristiky, trvají různě dlouho nebo jsou prováděny

jinými bloky kódu. Umocňovací funkce, které operují na více bitech exponentu v jedno průchodu, mohou mít ještě výraznější nedostatky. [2]

3.4.2. Diferenciální výkonová analýza

Útok pomocí diferenciální výkonové analýzy (different power analysis – DPA) neanalyzuje jen jednu křivku (měření) výkonu kryptografického systému, jak je tomu u SPA. Diferenciální výkonová analýza používá statistiku jako nástroj pro zkoumání mnoha křivek měření výkonu, a tím dosahuje odfiltrování nežádoucího šumu, který vzniká měřením. Útok vedený touto metodou se skládá ze dvou částí: sběru dat a z analýzy těchto dat. Při analýze většího vzorku dat se úspěšně používají právě statistické funkce pro korekci chyb (šumu). Takto lze získat detailnější informace o procesech, které probíhají uvnitř kryptografického systému.

Data zpracovávaná instrukcemi mají také vliv na změnu výkonu zařízení. Tyto změny nejsou moc zřetelné (na rozdíl od změn, které lze pozorovat u sekvence instrukcí) a jsou někdy zastíněny chybami měření nebo šumem. I z takto malých změn pomocí statistických funkcí přizpůsobených konkrétnímu algoritmu je možné odvodit informace, které pomohou prolomit systém. Celý proces lze navíc automatizovat a není zapotřebí velká znalost implementace.

Abychom mohli provést DPA útok, je nejdříve potřeba mít nějaký model spotřeby proudu konkrétním kryptografickým zařízením. Spotřeba energie u čipových karet závisí (kromě jiného) také na datech, která jsou přenášená po sběrnici. Proto můžeme uvažovat jednoduchý model spotřeby proudu, ve kterém hodnota Hammingovy váhy přenášeného bytu po sběrnici určuje hodnotu odběru proudu. [2]

3.5. Elektromagnetické postranní kanály

Pohyb nabitých částic je doprovázen elektromagnetickým polem. Proudění procházející procesorem může být jeho charakteristikou. Elektromagnetické útoky využívají elektromagnetického pole, která vzniká kolem kryptografického zařízení. Studováním elektromagnetických změn, které zaznamenávají cívky umístěné kolem kryptografického zařízení, lze podobně jako v případě výkonové analýzy odhalit tajné parametry výpočtu.

Na časovou, výkonovou a elektromagnetickou analýzu můžeme nahlížet také z hlediska vzrůstající dimenze (postranních) informací, které lze analýzou získat. Časová analýza poskytuje skalární informaci (dobu trvání) pro každé měření. Výkonová analýza poskytuje informaci o výkonu kryptografického zařízení v konkrétním okamžiku výpočtu. Elektromagnetická analýza poskytuje navíc ještě prostorovou informaci. S pomocí elektromagnetické analýzy lze vybudovat trojrozměrnou mapu změn elektromagnetického pole v blízkosti kryptografického zařízení, které nastanou během výpočtu. To umožňuje například oddělené zkoumání částí čipu.

Informace poskytnuty elektromagnetickými postranními kanály mohou být zpracovány stejnými metodami jako informace z výkonových postranních kanálů. Mluví se pak o jednoduché (SEMA) a diferenciální elektromagnetické analýze (DEMA). Tyto analýzy mohou navíc poskytnou mnoho dalších cenných informací. Z tohoto důvodu se tato metoda používá, přestože lze provést výkonovou analýzu. Navíc existují situace, kdy útočník může provést měření elektromagnetického pole. [2]

4. Měření výkonového kanálu

Následující kapitoly jsou zaměřeny na experimentální ověření možnosti útoku na čipovou kartu prostřednictvím výkonového postranního kanálu. V této kapitole jsou popsány různé metody měření výkonového kanálu, z nich je vybrána jedna, která bude dále užita. Další kapitola se věnuje návrhu a popisu experimentálního pracoviště. V rámci vlastní úlohy je provedena výkonová analýza procesu zpracování několika instrukcí SIM kartou a výkonová analýza algoritmu AES popsaném a navrženém v předchozích kapitolách běžícím na čipové kartě specifikace Java Card.

4.1. Vznik proudového kanálu

Útoky výkonovým postraním kanálem jsou popsány v předchozí kapitole, pro potřeby jeho analýzy je nyní vhodné zmínit vlastní fyzikální podstatu jeho vzniku.Výkonový nebo též proudový postranní kanál je vidět u všech dnešních procesorů, nejen u těch obsažených v čipových kartách. Procesor se více zahřívá při náročném výpočtu prvočísel, kdy se každým taktem hodin překlápí velké množství tranzistorů. Méně se zahřívá při psaní v textovém editoru, kdy většinu času mikroprocesor stráví čekáním na vstup a výstup. Z toho je patrné, že proudová spotřeba, respektive výkonová není u procesorů s časem konstantní. Pokud budeme měřit proud odebíraný čipovou kartou, dojdeme k průběhu, který bude svým zvlněním na první pohled podobný nahodilému šumu se stejnosměrnou složkou. Tato nahodilost je však pouze zdánlivá, neboť průběh proudové spotřeby je závislý na změnách stavu elektronických součástek uvnitř procesoru a ostatních součástí čipové karty. Vzhledem k velkému množství těchto součástek je pak výsledný průběh proudové spotřeby zdánlivě nahodilý šum. Změny proudové spotřeby vznikají na úrovni elementárních elektronických součástek, u procesorů je touto elementární součástkou převážně tranzistor. V současné době je většina procesorů založena na technologii CMOS, základním stavebním prvkem těchto obvodů je invertor. Invertor obsahuje dva tranzistory T1 a T2 řízené napětím s opačným typem vodivosti. Tyto dva tranzistory vůči sobě v závislosti na logické úrovni vstupního napětí zaujímají dva stavy (logická úroveň 1 a logická úroveň 0). Výkonová špička nastává při přechodu mezi těmito stavy,

tento jev se nazývá dynamickou spotřebou. Vznik proudové špičky je závislý na tom, kolik tranzistorů přepíná. Proudová spotřeba elektronických obvodů je přímo závislá na operacích v nich probíhajících, tedy na počtu překlápěných tranzistorů. [15]

4.2. Způsoby měření odběru proudu

Výkonová analýza je založena na sledování proudové spotřeby elektrického zařízení. Spotřeba proudu je měřena za předpokladu, že mikroprocesor je napájen konstantním napětím z ideálního zdroje napětí. Výkonová spotřeba je přímo úměrná spotřebě proudu dle vztahu:

$$p(t) = U \times i(t) \tag{4.1}$$

kde *p(t)* je příkon v čase t, *U* je napětí zdroje a *i(t)* je proud odebíraní mikroprocesorem v čase t. Napětí U zdroje zůstává konstantní, proud *i(t)* se mění. Pro změření změn odběru proudu v závislosti na provádění instrukci je zapotřebí řetězec, který je schopen tyto změny změřit a zaznamenat. Řetězec se skládá z analogově digitálního převodníku, který odebírá vzorky a sondy, kterou je převodník připojen do obvodu.

V úvahu přichází tři typy sond:

1. Proudová sonda,
2. Převodník I → U,
3. Odporový bočník.

Tyto sondy jsou připojeny na vstup analogově digitálního převodníku, který odebírá vzorky v konstantním čase. [15]

4.2.1. Proudová sonda

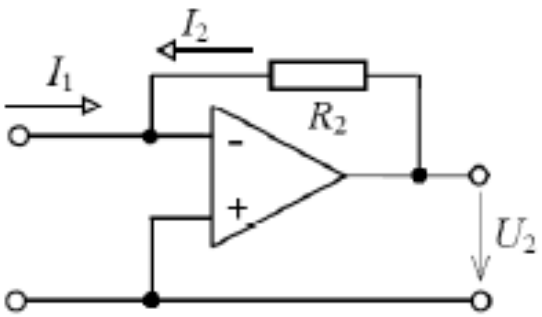
Proudová sonda umožňuje měřit proud ve vodiči bez nutnosti obvod přerušit.Pracuje většinou na induktivním principu pro střídavé proudy a stejnosměrné proudy měří pomocí hallovy sondy. Základem měřícího řetězce je transformátor, kde primární vinutí tvoří měřený vodič a v sekundárním vinutí se indukuje napětí odpovídající proudu procházejícímu primárním vinutím. Stejnosměrný proud je měřen pomocí hallovy sondy, na které se mění hallovo napětí. Napětí odpovídá magnetickému

poli, v němž je sonda vložena. Výsledný průběh proudu je dán složením výsledků měření těchto dvou metod. [15]

Výhodou proudové sondy je, že neovlivní rozložení napětí v měřeném obvodu čipové karty. Nevýhodou je její nedostačující citlivost, není schopna rozlišit malé změny v odběru proudu. Jelikož zachycení těchto malých hodnot proudů je pro úspěšnou analýzu klíčové, je použití proudové sondy nevhodné.

4.2.2. Převodník I → U

Základ převodníku I → U tvoří operační zesilovač, který se mezi svými vstupy snaží udržet nulový potenciál. Tohoto procesu je využito v převodníku I → U zapojeného dle Obr 4.1.



Obr 4.1 Převodník I → U

Proud I_1 je proudem měřeným. Neinvertující vstup je připojen na zem, proto na invertujícím vstupu musí být také nulový potenciál. Nulový potenciál bude na invertujícím vstupu, bude-li platit

$$I_1 = -I_2 = -\frac{U_2}{U_1}. \tag{4.2}$$

Vstupní odpor tohoto zapojení je

$$R_{VST} = \frac{U_1}{I_1} = 0. \tag{4.3}$$

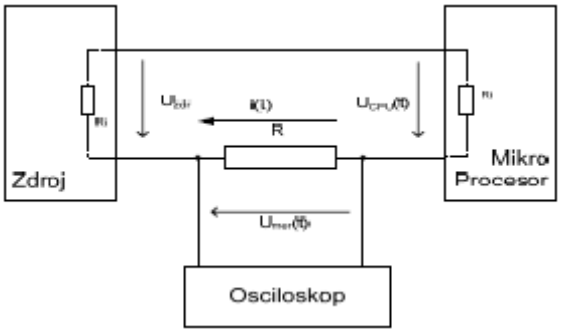
Napětí U_1 je vždy nulové, proto i vstupní odpor tohoto zapojení je nulový. Napětí U_2 odpovídá proudu I_1 dle vztahu:

$$I_1 \times R_2 = -U_2 . \tag{4.4}$$

Zapojení sice svým nulovým vstupním odporem neovlivní rozložení napětí v měřeném obvodu, ale frekvenční závislost operačního zesilovače vylučuje jeho použití. Proto, aby mohlo být zapojení použito, by byl potřeba operační zesilovač s minimálně dvakrát větší šířkou přenášeného pásma, než jaká je frekvence hodinových pulzů procesoru. Tato metoda pro naši analýzu tedy také není vhodná. [15]

4.2.3. Odporový bočník

Odporový bočník je malý odpor vřazený mezi zdroj a napájenou čipovou kartu. Výhodou bočníku je jeho frekvenční nezávislost. Vhodnou volbou velikosti odporu lze dosáhnout vhodné úrovně $U_{mer}(t)$, která odpovídá odebíranému proudu $i(t)$.



Obr 4.2 Odporový bočník

Na bočníku je okamžitá hodnota proměnného proudu $i(t)$ převedena na okamžitou hodnotu proměnného napětí $u(t)$ dle Ohmova zákona:

$$u(t) = R \times i(t) , \tag{4.5}$$

kde R je odpor bočníku a i(t) je proud v čase t.

Okamžitá hodnota získaného napětí $u(t)$ je přivedena na vstup A/D převodníku a je dále zaznamenána a zpracována. [15]

Pro správnost měření je důležitá volba bočníku. Jeho hodnotu je možné určit výpočtem, nebo experimentálně zvolit. Tato metoda je pro měření výkonového postranního kanálu nejvhodnější, proto bude v laboratorním měření použita.

4.3. Labview

Jako prostředí pro vývoj softwaru je zvoleno programovací prostředí Labview. Bylo zvoleno z důvodu kvalitní podpory námi dostupných měřících karet.

Programovací a vývojové prostředí LabVIEW (Laboratory Virtual Instruments Engineering Workbench = laboratorní pracoviště virtuálních přístrojů), někdy též označované LV, je produktem americké firmy Nationall Instrument. Hlavní výhodou tohoto vývojového prostředí oproti ostatním je, že programování neprobíhá pomocí textových příkazů, jak je tomu například u jazyka C, ale je čistě graficky orientované tzv. G jazyk (Graphical language). Výsledná aplikace vytvořená v LabVIEW se nazývá virtuální přístroj (VI). Jednou z nejdůležitějších vlastností programu je jeho univerzálnost, která spočívá ve schopnosti obstarat všechny fáze měřicího procesu. Tyto fáze jsou sběr, analýza a prezentace naměřených dat.

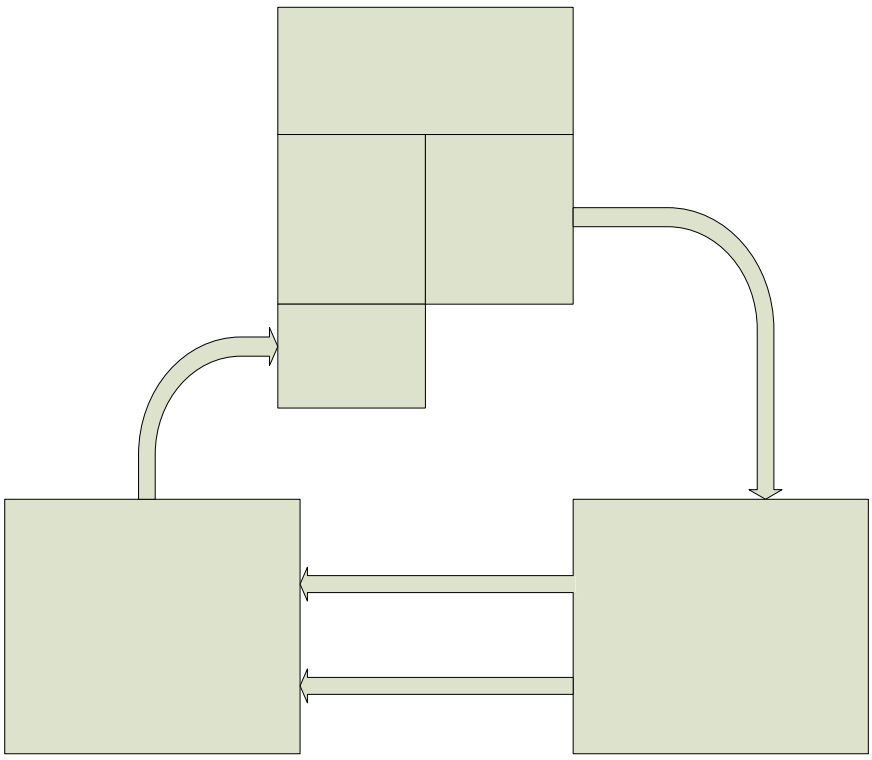
Struktura vývojového prostředí LabVIEW bývá často označována jako modulární a hierarchická. Označení modulární struktura znamená, že virtuální přístroj, který v LabVIEW vytvoříme, vznikne spojením více podřízených virtuálních přístrojů tzv. sub-VI. Z hlediska programové struktury nejsou tyto sub-VI ničím jiným než podprogramy. Tyto podprogramy mohou být vykonávány buď jednotlivě a nebo společně jako jeden program (hierarchická struktura). [19]

Pro potřeby měření je využito Labview verze 8.6.

5. Laboratorní pracoviště

Tato část je věnována návrhu laboratorního pracoviště pro měření výkonového postranního kanálu čipové karty, popisuje měřící přípravky, měřící hardware i navržený software.

Pro měření odběru proudu čipové karty je zapotřebí čtečka karet standardu ISO 7816 s vloženým bočníkem, rozhraní snímající proudový odběr (resp. okamžité změny napětí) a umožňující A/D převod, PC s prostředím Labview a vhodně navrženým software sloužícím k finálnímu zpracování signálu a jeho zobrazení a podpůrný software určený ke komunikaci se čtečkou karet a s analyzovanou čipovou kartou. Blokové schéma je zobrazeno na Obr 5.1 Blokové schéma měřícího pracoviště.

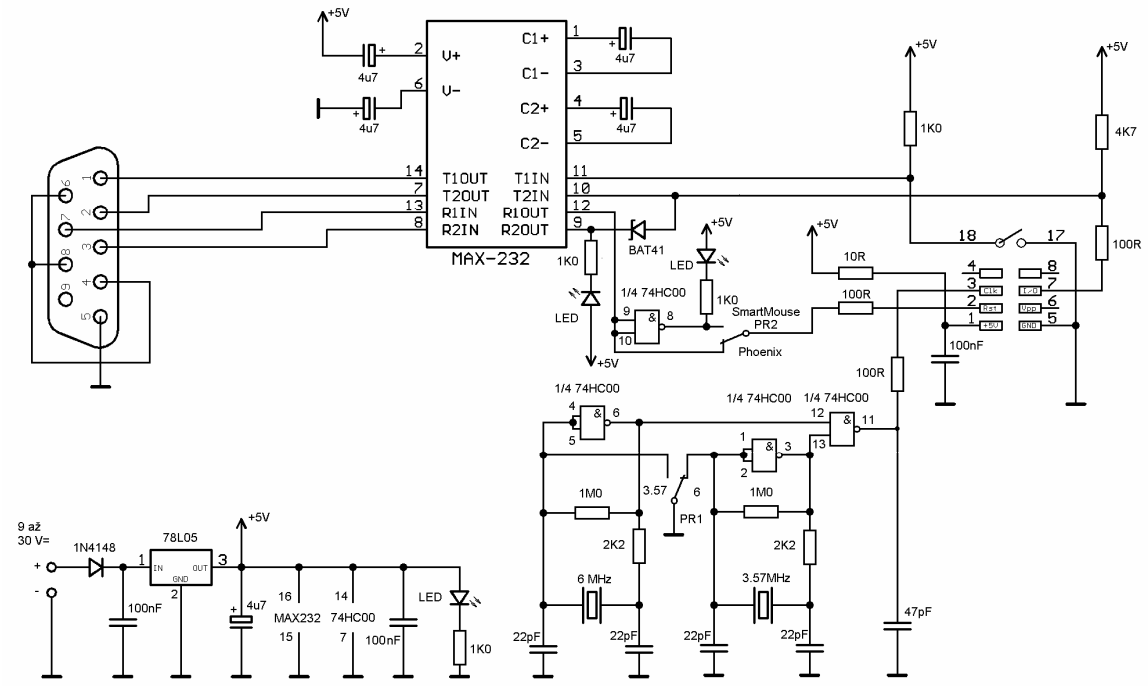


Obr 5.1 Blokové schéma měřícího pracoviště

5.1. Měřící přístroje

5.1.1. Čtečka karet

Jako rozhraní pro čtení čipových karet byla zvolena čtečka karet standardu Phoenix/Smartmouse interface. Tato čtečka umožňuje transformaci signálu ze seriového rozhraní na úroveň požadovanou čipovou kartou a obráceně. Čtečka byla zkonstruována na základě volně dostupného schéma zapojení. Její výhodou je možnost modifikace pro potřeby měření výkonového kanálu. Čtečka Smartmouse/Phoenix nabízí široké možnosti pro potřeby čtení a programování karet standardu ISO 7816, její vyžití je závislé jen na použitém obslužném software. Schéma čtečky je na Obr 5.2. [16]



Obr 5.2 Schéma zapojení Phoenix/Smartmouse rozhraní

Zařízení pracuje s jedním ze zvolených taktovacích kmitočtů: 3,579 MHz a 6 MHz. Tyto frekvence odpovídají přenosovým rychlostem 9600 Baud a 16457 Baud. K nastavení oscilátoru slouží switch PR1. Základním nastavením je 3,579 MHz, většina používaných čipových karet umí komunikovat při nastavení 6 MHz. Dle této zvolené

rychlosti je třeba nastavit parametry přenosu na PC (přenosová rychlost seriového rozhraní).

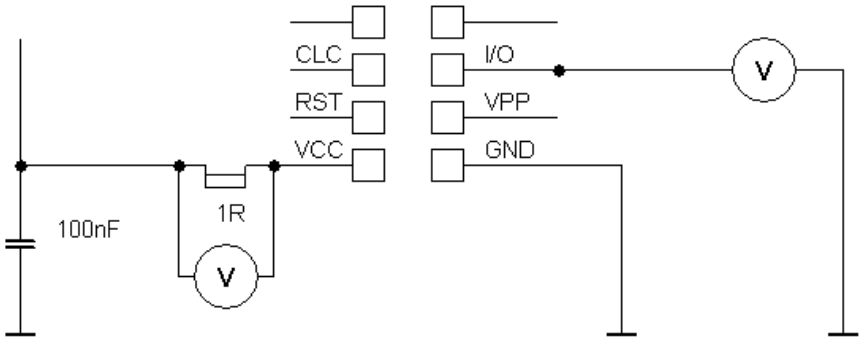
Druhý switch PR2 nastavuje režim práce rozhraní. V pozici „Phoenix“ je resetovaní signál veden beze změny na kartu, v pozici „Smartmouse“ je invertován hradlem obvodu 74HC00. Využívá se k nastavení kompatibility s různými programy.

Zařízení je napájeno stejnosměrným napětím o velikosti U = <9, 24> V, při měření bylo je vstupní napětí nastaveno na 12 V.

Stav zařízení popisují tři LED. LED1 indikuje napájení, LED2 přenos dat a LED3 stav signálu RESET. [16]

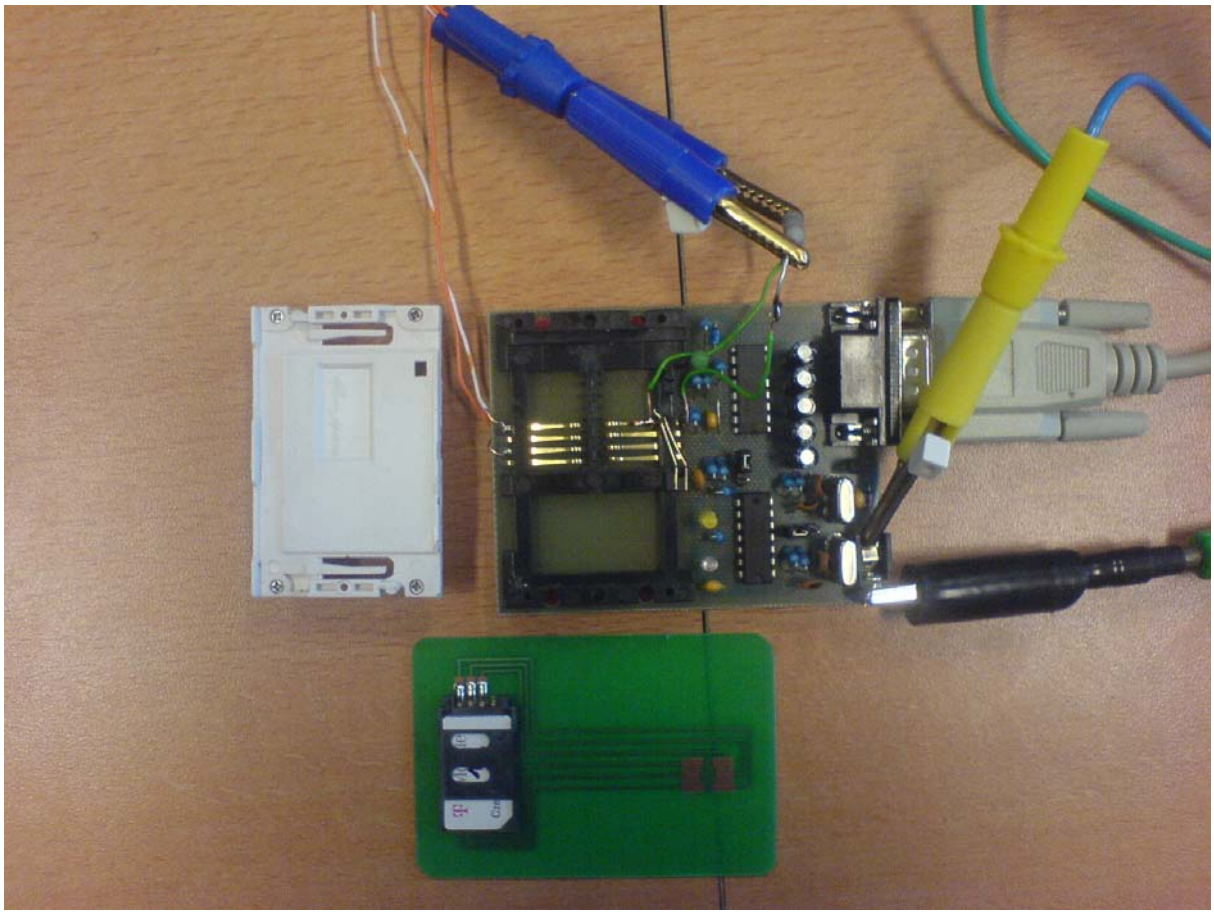
Odporový bočník, potřebný pro měření proudového odběru (okamžitých změn napětí), je vložen před kontakt PIN 1, tedy kontakt VCC. Velikost odporu je zvolena 1Ω. Tímto je zaručen minimální úbytek napětí na kontaktu VCC a zjednodušení transformace U → I, velikost okamžité změny napětí v tomto případě odpovídá velikosti proudového odběru analyzované čipové karty.

Pro potřeby synchronizace jsou měřeny změny napětí na kontaktu I/O.
Výřez ze schématu zapojení čtečky karet znázorňující tyto úpravy je na Obr 5.3.



Obr 5.3 Zapojení bočníku ve čtečce karet

Takto modifikovaná čtečka karet je zobrazena na Obr 5.4.



Obr 5.4 Modifikovaná čtečka karet

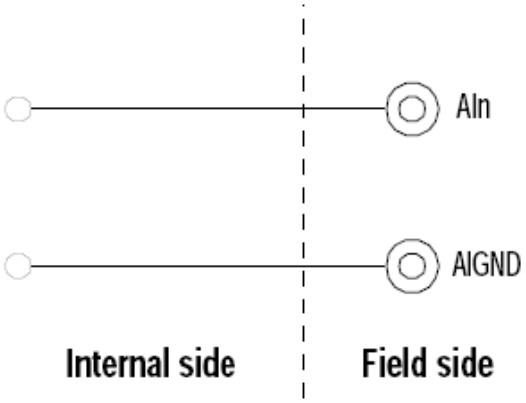
5.1.2. Měřící karty

Měřící karty tvoří mezičlánek mezi zdrojem signálu a vývojovým prostředím vyhodnocujícím zdroje z těchto externích zdrojů. Kombinace měřící karta + vývojové prostředí byla oproti kombinaci diferenciální napěťová sonda + osciloskop upřednostněna kvůli násobně vyšší citlivosti na očekávané velmi malé změny napětí. Dostupné diferenciální sondy disponují rozsahem měřitelného diferenciální napětí začínajícím na stovkách milivoltů, což je pro měření napěťových změn v řádech jednotek milivoltů, ke kterým dochází při aktivitě např. SIM karet, nedostačující.

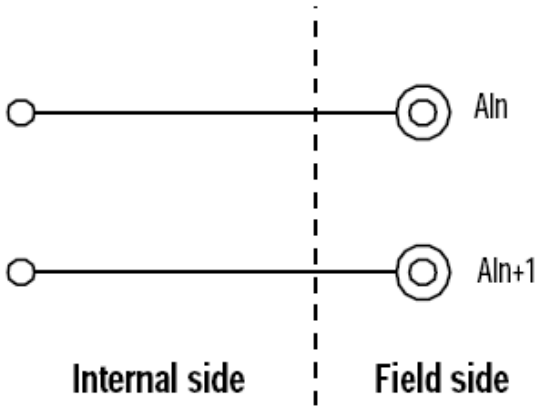
Pro měření výkonového kanálu čipových karet je použita dvojice karet Advantech PCLD-8710 (externí měřící terminál) a Adventech PCI-1716 (interní multifunkční karta). Karta PCLD-8710 obsahuje svorkovnici pro připojení analogových vstupů, konektor pro připojení digitálního zdroje dat, konektor pro digitální výstup a

obvod umožňující měření teploty. Karta je určena k připojení k multifunkčním kartám disponujících konektorem 68-pin SCSI-II, jako je například právě karta PCI-1716. Karta PCI-1716 je multifunkční karta určená pro PCI sběrnici. Karta obsahuje 16-bitový A/D (analogově/digitální) převodník s vzorkovacím kmitočtem 250kS/s a 1024 vzorkový zásobník typu FIFO (First In First Out). Karta poskytuje 16 jednoduchých nebo 8 diferenciálních A/D vstupních kanálů, případně odpovídající počet jejich kombinací, 2 16ti bitové D/A výstupní kanály a 16 digitálních vstupních/výstupních kanálů. [17][18].

Schéma zapojení kontaktů na kartě PCLD-8710 pro měření diferenciálního napětí na odporovém bočníku je znázorněno na Obr 5.6, schéma zapojení pro měření napětí kontaktu I/O čipové karty znázorňuje Obr 5.5.



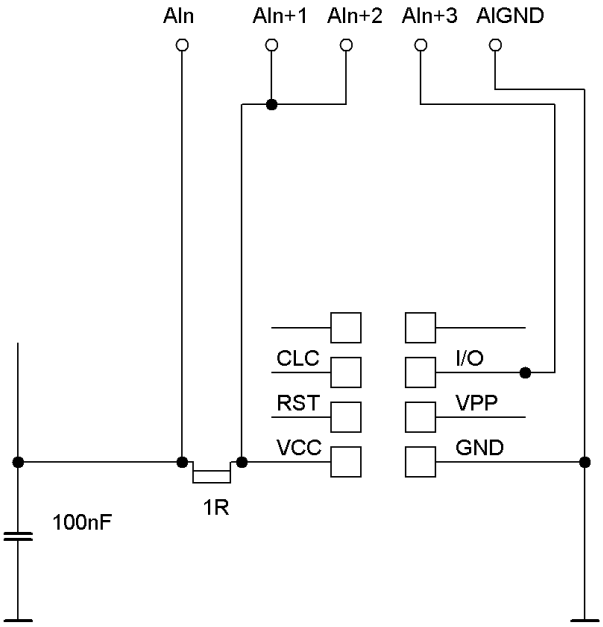
Obr 5.5 Zapojení konektorů karty PCLD-8710 pro měření napětí [18]



Obr 5.6 Zapojení konektorů karty PCLD-8710 pro měření diferenciálního napětí [18]

Parametr *n* značí číslo kanálu, v případě prvního schématu je v rozsahu <0, 15>, v případě měření diferenciálního napětí je v intervalu <0, 14>.

Čtečka karet Smartmouse/Phoenix je s kartou PCLD 8710 zapojena dle schématu na Obr 5.7. Čísla kontaktů měřící karty odpovídají skutečnosti.



Obr 5.7 Připojení čtečky karet na měřící rozhraní PCLD 8710

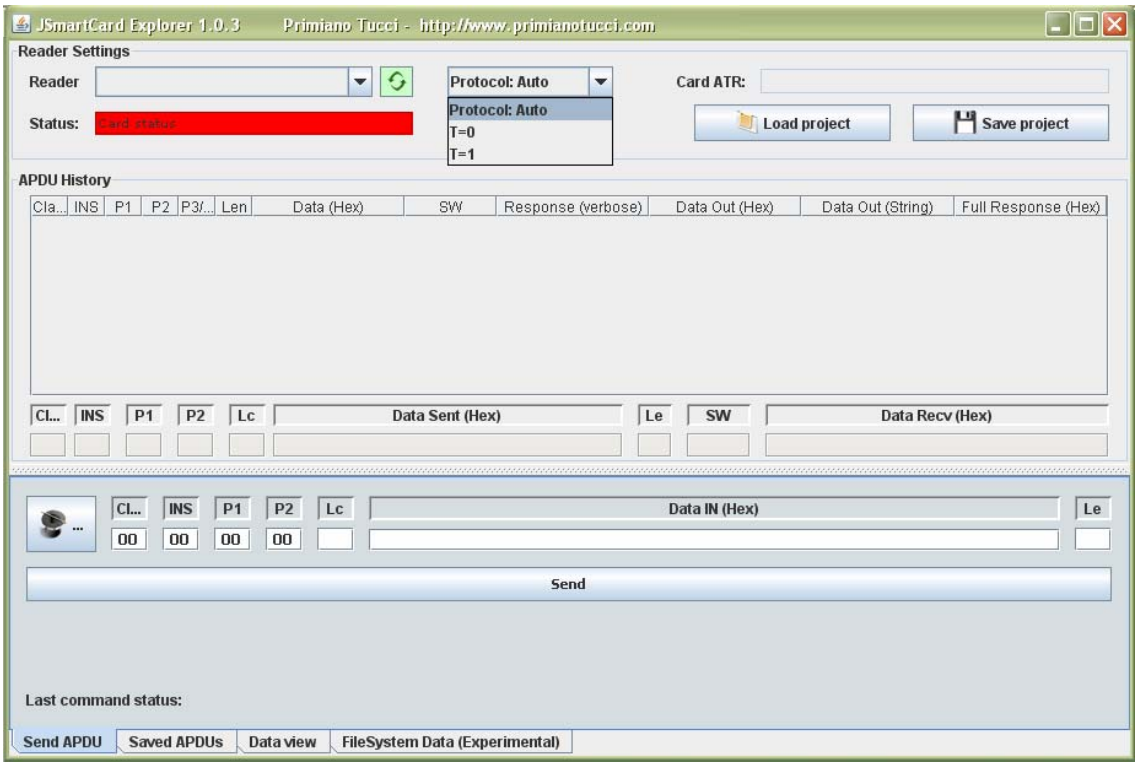
Kontakty AIn a AIn+1 karty PCLD 8710 budou zapojeny v diferenciálním módu, kontaky AIn+2 a AIn+3 v normální módu. Toto nastavení lze provést v konfiguračním rozhraní karty PCI 1716. Tímto nastavením způsobíme, že k dalšímu užití nám budou k dispozici 3 logické kanály. Na prvním kanále budeme sledovat vlastní proudový odběr čipové karty, na kanále 2 napájecí napětí čipové karty, prostřednictvím kanálu 3 můžeme sledovat napěťové změny na kontaktu I/O. Údaje získané z kanálů 2 a 3 jsou užity ve fázi zpracování výsledků jako pomocné hodnoty nezbytné k vyfiltrování žádoucích informací o proudovém odběru karty získaných sběrem dat z kanálu 1.

5.2. Software

Pro potřeby komunikace a obsluhy analyzované čipové karty je použit volně dostupný software, pro analýzu proudového kanálu je navržen specializovaný balíček programů.

5.2.1. Obslužný software

S čipovými kartami standardu ISO 7816 je možno komunikovat prostřednictvím příkazů komunikačního protokolu APDU (Kap. 1.3.3). Pro tuto činnost je možno využít velké množství volně dostupných i komerčních programů, lišících se jak cenou tak škálou nabízených možností. Základním nástrojem pro komunikaci s čipovými kartami je utilita APDUtool obsažená ve vývojovém balíčku pro vývoj aplikací specifikace Java Card[5]. Aplikace umožňuje z prostředí příkazového řádku komunikovat s čipovou kartu na základě uživatelem vytvořených skriptů. Výhodou použití tohoto skriptu je naprostá kontrola nad činností čipové karty, nevýhodou časová náročnost potřebná pro návrh skriptů, vysoká pravděpodobnost vložení chybných údajů jako například nepřesný převod hodnot do hexadecimálního tvaru, chybný výpočet parametru délky APDU příkazu apod. Kompromisem je tedy volba obslužného software, který tyto nutné výpočty a převody vykoná automaticky místo uživatele a který uživateli zachová nezbytnou kontrolu nad činností čipové karty. Vhodným prostředkem pro tuto obslužnou činnost se ukázala být volně dostupná aplikace JSmart Card Explorer [20]. Aplikace nabízí možnost volby čtečky karet (pokud jich je na PC nainstalováno více), transportního protokolu (T0 nebo T1, volíme bajtově orientovaný T0). Práci ulehčuje možnost použití předdefinovaných příkazů, které je možno libovolně měnit, převod dat z desítkové soustavy do šestnáctkové soustavy, automatické spočítání parametru P3 délky dat. Uživatelské rozhraní programu JSmart Card Explorer je znázorněno na Obr 5.8.



Obr 5.8 Uživatelské rozhraní JSmart Card Explorer

5.2.2. Smart Card Power Analyzer

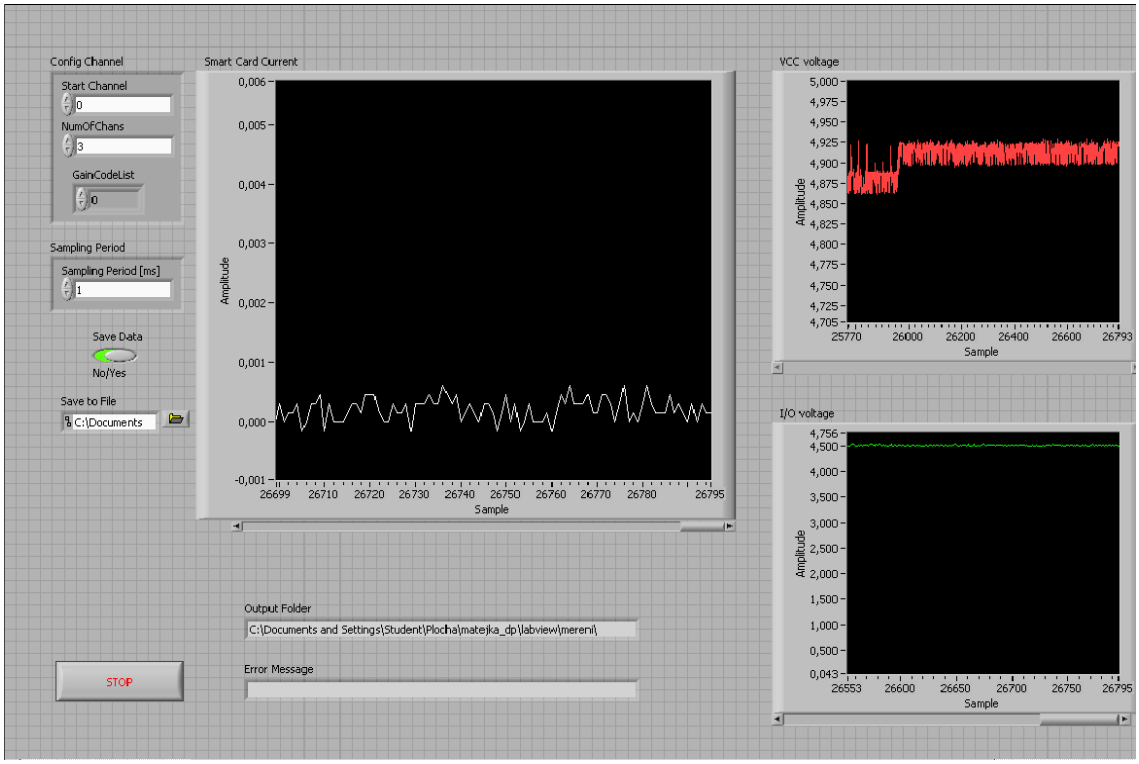
Balíček aplikací Smart Card Power Analyzer je vytvořen pro komplexní analýzu proudového kanálu čipové karty, spolupracuje s měřícími kartami firmy Adventech. Je vytvořen ve vývojovém prostředí LabVIEW verze 8.6, pro jeho činnost je kromě tohoto prostředí potřeba instalace ovladačů měřících karet firmy Adventech pro tento systém. Aplikace poskytuje prostředky pro zpracování dat ve všech fázích laboratorního měření, tedy pro fází měření a záznamu dat, fázi filtrace naměřených dat a fázi vyhodnocení výsledků.

Balíček se skládá z několika modulů (podprogramů), které slouží k jednotlivým úkonům měření a zpracování. Jedná se o moduly Oscilloscope, Plotter, Filter, Correlation. Význam jednotlivých modulů vychází z jejich názvu. Oscilloscope nahrazuje funkci běžného osciloskopu, Plotter slouží k zobrazení naměřených průběhů, Filter redukuje velikost naměřených dat o nadbytečné údaje, Correlation slouží k srovnání naměřených dat s daty referenčními.

Základním pravidlem pro užití této aplikace je rovnost jeden měřený průběh = jeden výsledný datový soubor. Spuštění všech modulů je možné standardně prostřednictvím tlačítka Run VI na hlavním panelu okna aplikace.

Oscilloscope

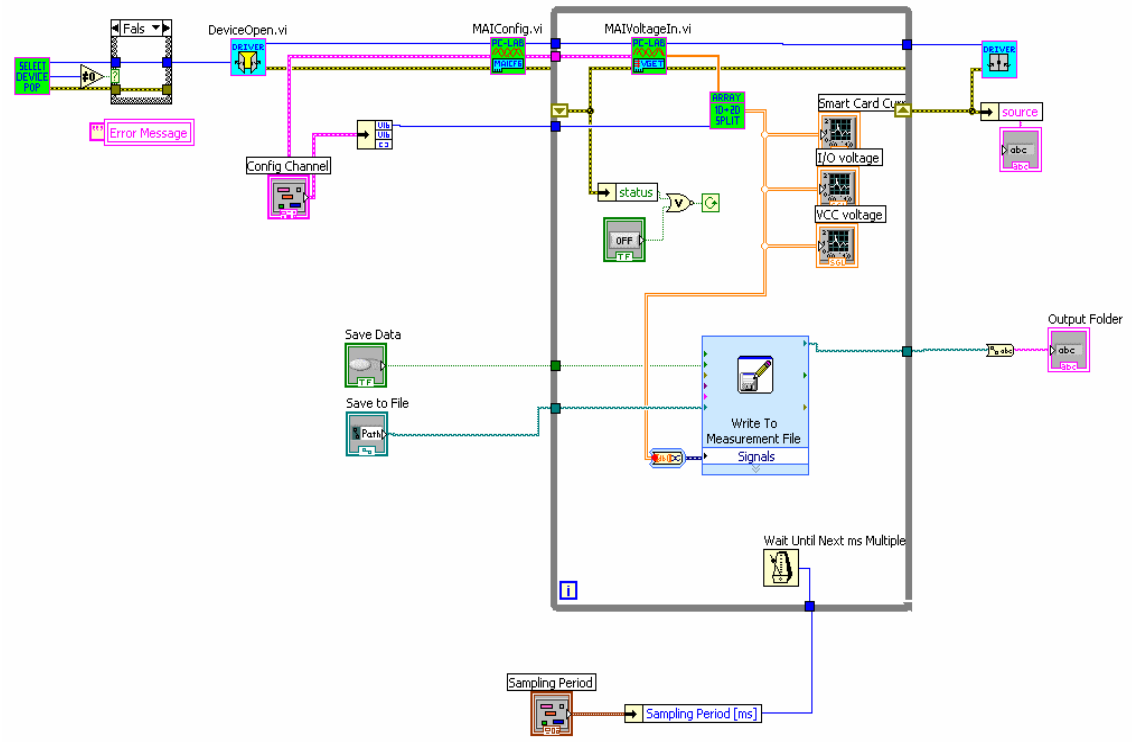
Modul oscilloscope nahrazuje funkci běžného osciloskopu. Umožňuje zobrazení průběhu signálu z jednoho až tří kanálů a jejich záznam do souboru formátu lvm (LabVIEW Measurement File). Pracovní postup s tímto modulem probíhá v řetězci spuštění aplikace -> odeslání apdu příkazu a následná činnost čipové karty -> zastavení průběhu. Uživatelské rozhraní modulu Oscilloscope je znázorněna na Obr 5.9.



Obr 5.9 Uživatelské rozhraní modulu Oscilloscope

Před spuštěním programu je třeba zadat několik důležitých parametrů. Položka Start channel musí odpovídat číslu kanálu označeném na Obr 5.7 jak AIn. Hodnota tohoto pole je omezena na hodnotu 0 – 12, jako výchozí je nabízen kanál 0. Hodnota zadaná v poli NumOfChan nastavuje počet zobrazovaných kanálů, maximální hodnota je 3, nastavením jiných hodnot nezískáme data z kanálů měřících napětí VCC a I/O důležitá pro další zpracování. Hodnota z nabídky GainCodeList je vyžadována ovladačem A/D převodníku firmy Adventech, označuje měřící rozsah. Maximální

hodnota měřeného napětí je dle specifikace čipové karty 5,5V, proto použijeme nejbližší možný rozsah, který je 10V a dle manuálu měřící karty PCI 1716 mu odpovídá hodnota GainCodeList 0. Do pole Sampling Period vložíme požadovanou vzorkovací periodu. Vhodná volba vzorkovací periody je důležitá pro kvalitu a vypovídající hodnotu naměřených dat. Hodnota 0 je vzorkovací perioda řízena systémem LabVIEW na základě dostupného výpočetního výkonu. Pomocí přepínače Save data zvolíme, zda chceme průběhy pouze zobrazit nebo i ukládat do souboru typu lvm. V případě kladné volby zvolíme v poli Save to File umístění a název výstupního souboru, pokud mezi jednotlivými měřeními nebudeme název souboru měnit, přidá aplikace za název souboru číslovku kterou postupně dle počtu opakovaného měření zvyšuje.



Obr 5.10 Blokové schéma modulu Oscilloscope

Na Obr 5.10 je znázorněno blokové schéma modulu Oscilloscope, ze kterého je patrný vlastní běh programu. Po spuštění programu je uživatel vyzván prostřednictvím bloku Select device pop k volbě měřící karty. Tato volba je důležitá pro případ, že je na PC nainstalováno více těchto karet. Po zvolení karty je povolen a otevřen přístup k rozhraní a nastává fáze nastavení parametrů zadaných v uživatelském prostředí v poli Config Channel. Tím je ukončen proces nastavení a inicializace a nastává fáze čtení dat. Tento proces běží v nekonečné smyčce, perioda této smyčky a tedy vzorkovací perioda

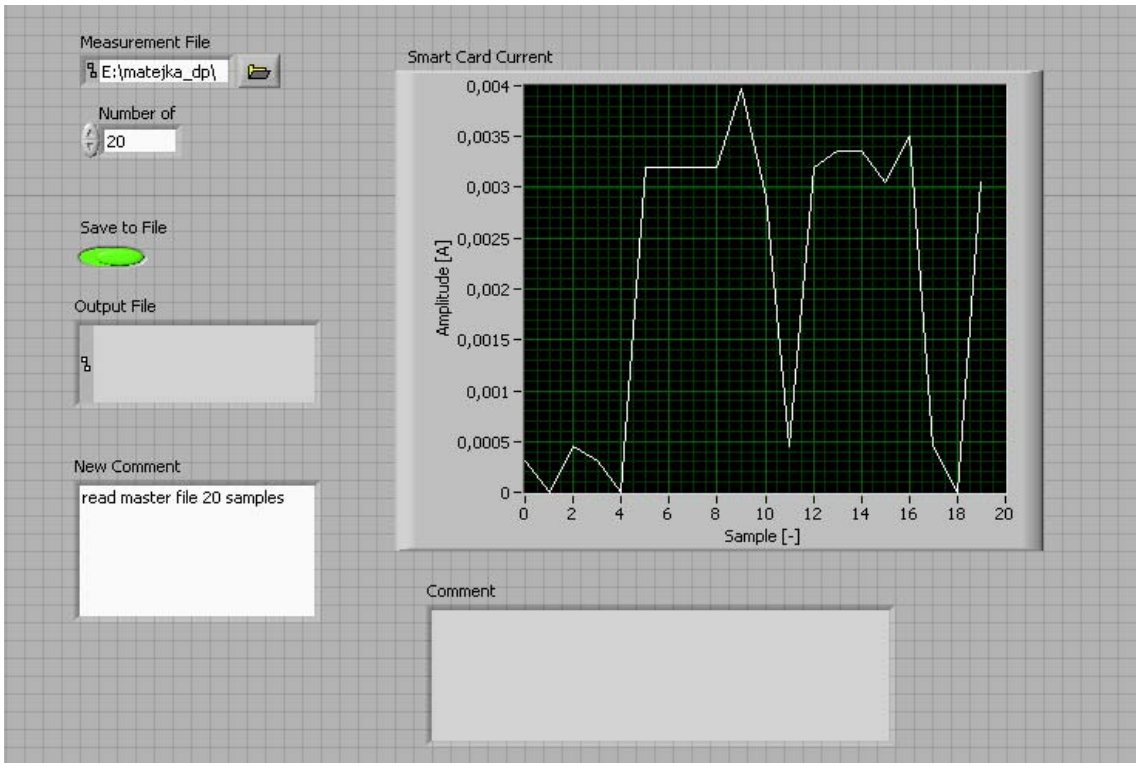
procesu je nastavena hodnotou Sampling period a zajištěna blokem Watt Until Next ms Multiple, smyčka je ukončena stiskem tlačítka Stop v uživatelském rozhraní. Vlastní čtení dat z měřícího rozhraní zajišťuje blok MAIVoltageIn, který slouží k paralelnímu sběru dat z více kanálů. Výstupem je pole dat (datový typ array), jeho rozdělení na jednotlivé kanály zajišťuje blok Array Split. Tyto jednotlivé pole – kanály jsou graficky zobrazeny pomocí bloků Smart Card Current, I/O voltage a VCC voltage typu Waveform Graph a pokud je to uživatelem zvoleno ukládána pomocí bloku Write To Measurement File. Tento blok byl k ukládání dat zvolen kvůli široké nabídce parametrů. Během následných experimentů bylo zjištěno, že činnost tohoto bloku.je velice náročná na výpočetní výkon. Při použití aplikace Oscilloscope na méně výkonném počítači je následkem snížení dosažené vzorkovací periody (nastavená hodnota je ignorována, smyčka pracuje na frekvenci které je schopen počítač dosáhnout) a tím pádem snížení vypovídací hodnoty naměřených hodnot. Na pracovišti s PC s procesorem Pentium 4 o frekvenci 3.0 GHz, operační paměti 512 MB byl experimentálně zjištěn nárůst ve vzorkovací periodě až 150 %, což je velice podstatný rozdíl a značně degraduje vypovídací hodnotu zaznamenaných hodnot.

Výsledkem použití modulu Oscilloscope je soubor s daty o průběhu jedné operace či jednoho měřícího cyklu. Soubor je v textové formě, data z jednotlivých kanálů jsou oddělena tabelátorem, takže je možno tento soubor zobrazit nebo případně editovat v mnoha textových či tabulkových editorech. Ve výstupním souboru jsou čtyři sloupce dat: časová hodnota a jí odpovídající hodnoty proudového odběru karty, napětí na kontaktu VCC a napětí na kontaktu I/O. V této fázi není výstup dat nijak ošetřen, takže soubor kromě užitečných hodnot obsahuje i redundantní informace zaznamenané před začátkem vlastního měřeného cyklu a po jeho ukončení.

Filter

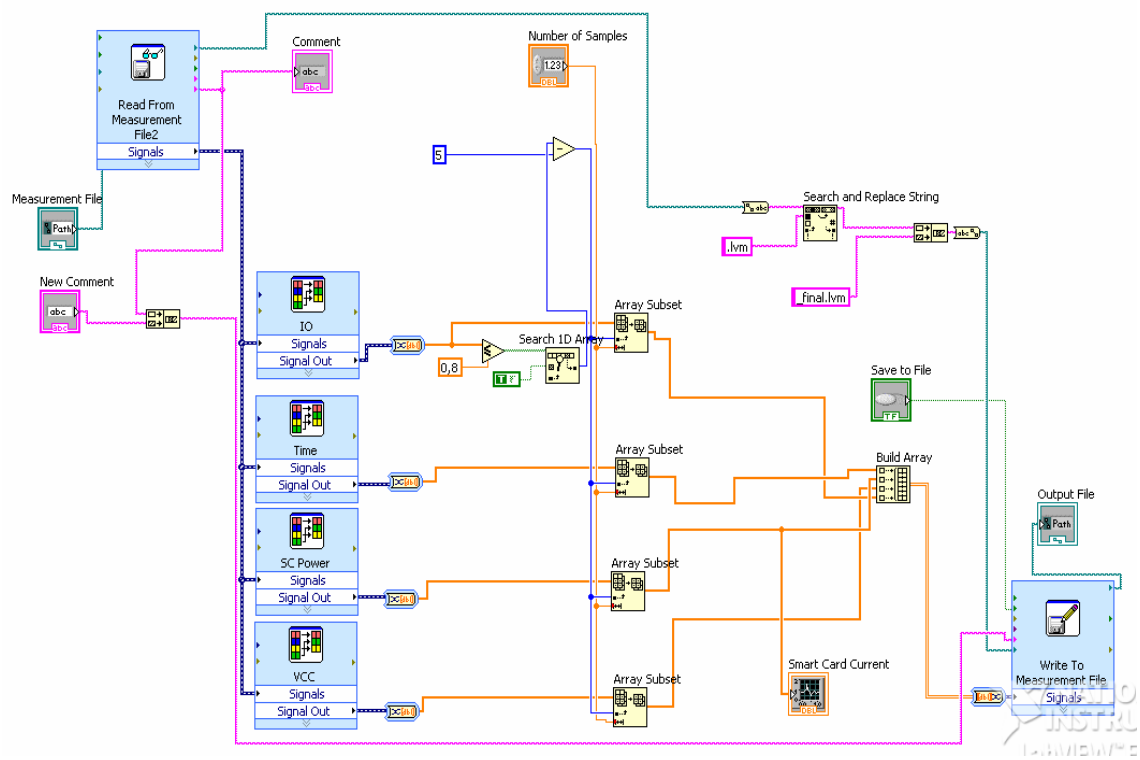
Modul Filter slouží k filtraci souborů získaných z modulu Oscilloscope. Nadbytečné hodnoty vznikají v prodlevě mezi spuštěním modulu Oscilloscope a odesláním příkazu a mezi ukončením výkonu instrukce a ukončením běhu aplikace. Počátek měřených dat je určen okamžikem přijetí instrukce vyslané uživatelem. Této synchronizace je dosaženo sledováním hodnoty napětí na kontaktu I/O. Hodnoty napětí na tomto kontaktu jsou obsaženy v Tab. 1.3. V klidovém režimu se velikost napětí nachází na hodnotě stavu V_{ih}. Při přijetí příkazu přejde kontakt I/O do stavu V_{il}, dle

výše zmíněné tabulky tato hodnota dosahuje maximální velikosti 0,8 V. Tato hodnota je nastavena jako prahová, pokles napětí pod tuto hodnotu značí počátek činnosti čipové karty a tedy počátek filtrovaných dat. Ukončení činnosti procesoru čipové karty nelze přesně s napětím kontaktu I/O synchronizovat, velikost napětí toho kontaktu ve stavu vysílacím se překrývá s hodnotami v klidovém režimu. Z tohoto důvodu je stanovení konce analyzované činnosti zadáno uživatelsky počtem vzorků. Uživatel může vybrané průběhy zobrazovat, měnit počet vzorků a na základě své vlastní zkušenosti stanovit optimální počet. Uživatelské prostředí modulu Filter je na Obr 5.11.



Obr 5.11 Uživatelské prostředí modulu Filter

Uživatel zvolí vstupní soubor a počet vzorků. Uložený soubor získá stejné jméno, jako původní soubor, doplněný o text „_final“. K souboru je možno připojit komentář.



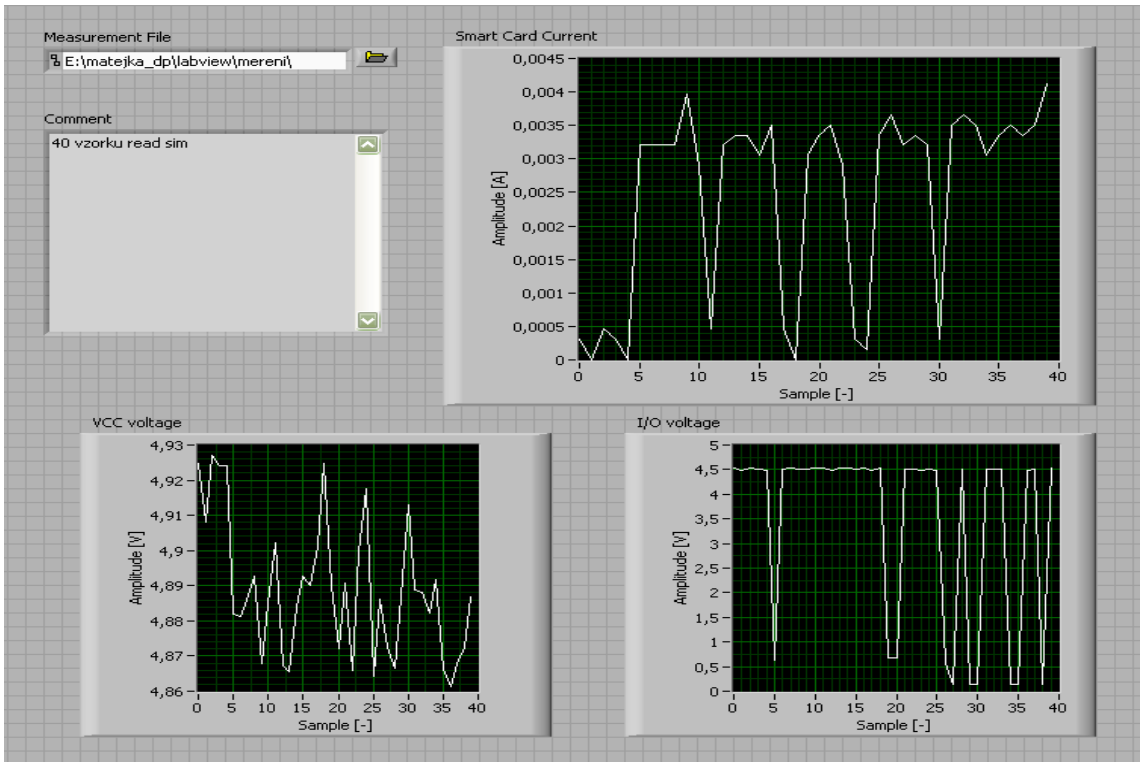
Obr 5.12 Blokové schéma modulu Filter

Funkce modulu Filter je patrná z blokového schématu na Obr 5.12. Po načtení souboru je signál rozložen pomocí čtyř bloků typu Select Signal na jednotlivé kanály. Ty jsou následně převedeny na datový typ pole. Hodnoty pole kanálu I/O jsou následně srovnávány s prahovou hodnotou 0,8 V, pokud je nalezena rovnající se či menší hodnota, je zjištěn její index, ten určuje počátek užitečných dat v původním signálu. Pro lepší názornost je tato hodnota snížena o číslo 5, tím je možno ve výsledném signálu zřetelně poznat přechod z klidového režimu čipové karty do fáze vykonání příslušné instrukce. Délka výsledného signálu je dána uživatelsky zadanou hodnotou Number of Samples. Hodnota indexu, délky dat a pole jednotlivých kanálů jsou vstupy čtyř bloků Array Subret, které z původního pole odstraní nepotřebné údaje. Následuje spojení jednotlivých polí, výsledné pole strukturou odpovídá poli původnímu. Následně je převedeno na signál a pokud je to uživatelem zvoleno, je uloženo do souboru s názvem původního souboru doplněného o koncovku „_final“, soubor je doplněn o uživatelský komentář. Pro budoucí zpracování je důležité, aby soubory zachycující data stejných instrukcí, měly stejný počet vzorků.

Takto vytvořený soubor je připraven ke zpracování v modulech Plotter a Correlation

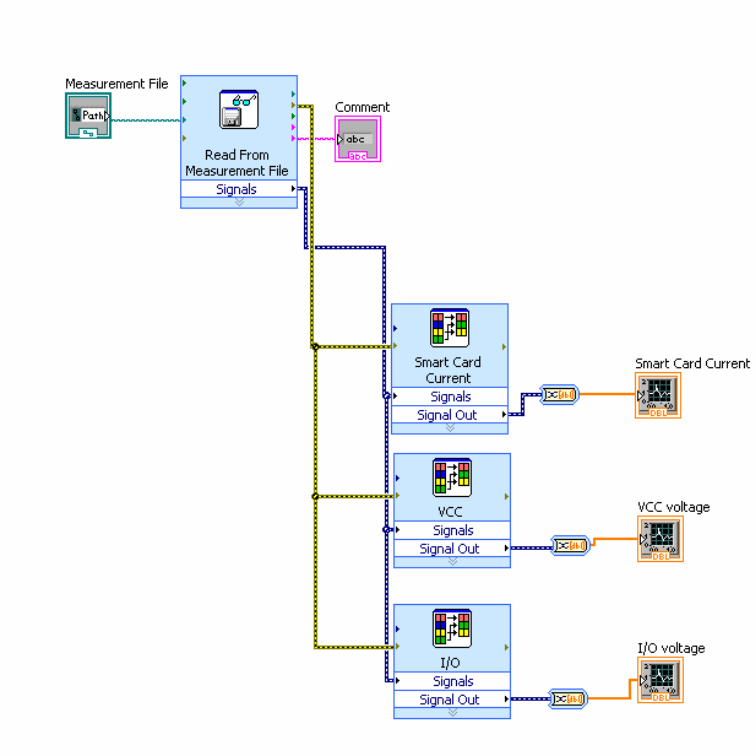
Plotter

Modul Plotter slouží k zobrazení signálů ze souborů vytvořených modulem Filter (tedy soubory s dovětkem _filter ve svém názvu). Uživatelské rozhraní je zobrazeno na Obr 5.13.



Obr 5.13 Uživatelské rozhraní modulu Plotter

Uživatel zvolí cestu k souboru, v jednotlivých grafech jsou poté zobrazeny průběhy signálů proudového odběru, napětí kontaktu VCC a napětí kontaktu I/O.
Funkci programu znázorňuje blokové schéma na Obr 5.13.



Obr 5.14 Blokové schéma modulu Plotter

Signál načtený z datového souboru je bloky Select Signal rozdělen na jednotlivé kanály, ty jsou poté jako datové pole přivedeny do jednotlivých grafů.

Correlation

Posledním krokem laboratorního měření je vyhodnocení jednotlivých signálů, souvislost mezi jednotlivými vzorky. Na srovnávání neznámých naměřených vzorků a vzorků, u nichž je nám znám detailní průběh vykonávané instrukce, je založena diferenciální proudová analýza. Používaným operátorem, pro zjištění podobnosti tvaru signálů, je korelace. Tato metoda je použita právě v modulu Correlation.

Korelace dvou signálů je hodnota z intervalu <-1, 1>. Hodnota 1 znamená identický průběh, hodnota -1 inverzní průběh. Korelace 0 značí, že hodnoty spolu nesouvisí.

Pro výpočet korelace je třeba nejdříve určit rozptyl (variance) a kovarianci. Rozptyl je definován funkcí

$$\mathrm{var}(x) = \frac{\left(\sum x_i - \bar{x}\right)^2}{n-1}, \tag{5.1}$$

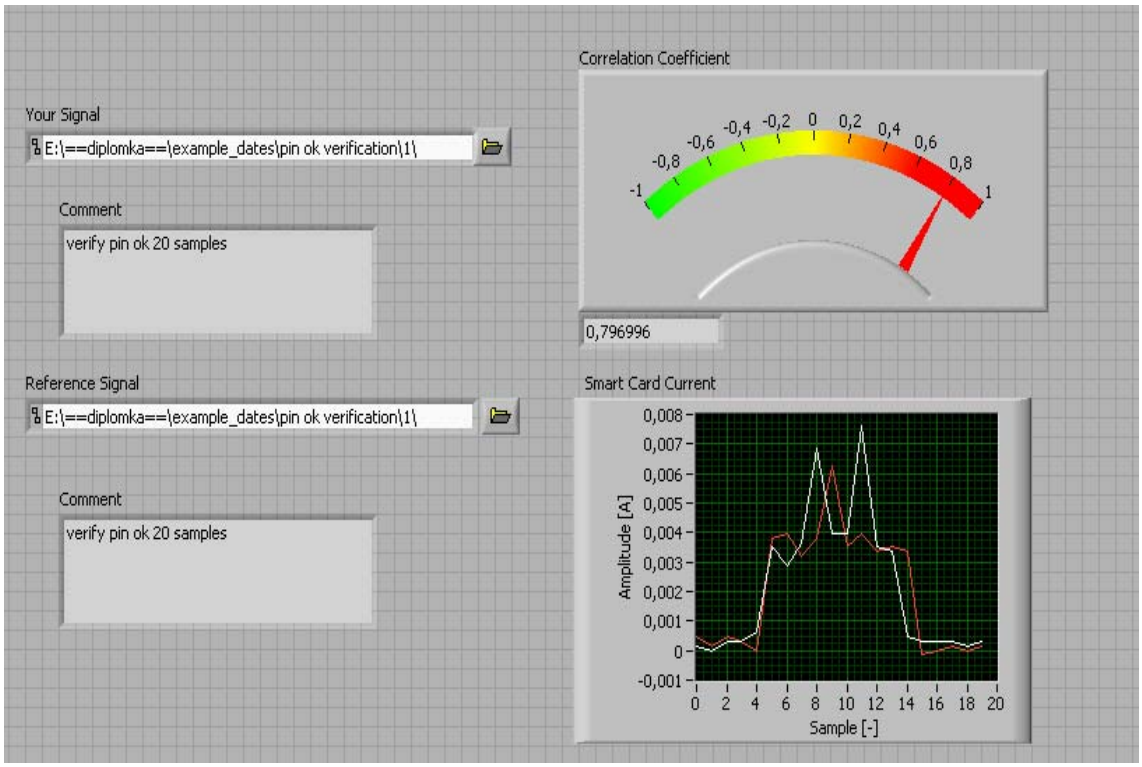
kde x reprezentuje i-tý člen x, $\bar{x}$ aritmetický průměr hodnot x a n počet prvků. Kovariance je definována vztahem

$$\mathrm{cov}(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \overline{y})}{n-1}. \tag{5.2}$$

Korelaci dvou signálů definuje vztah

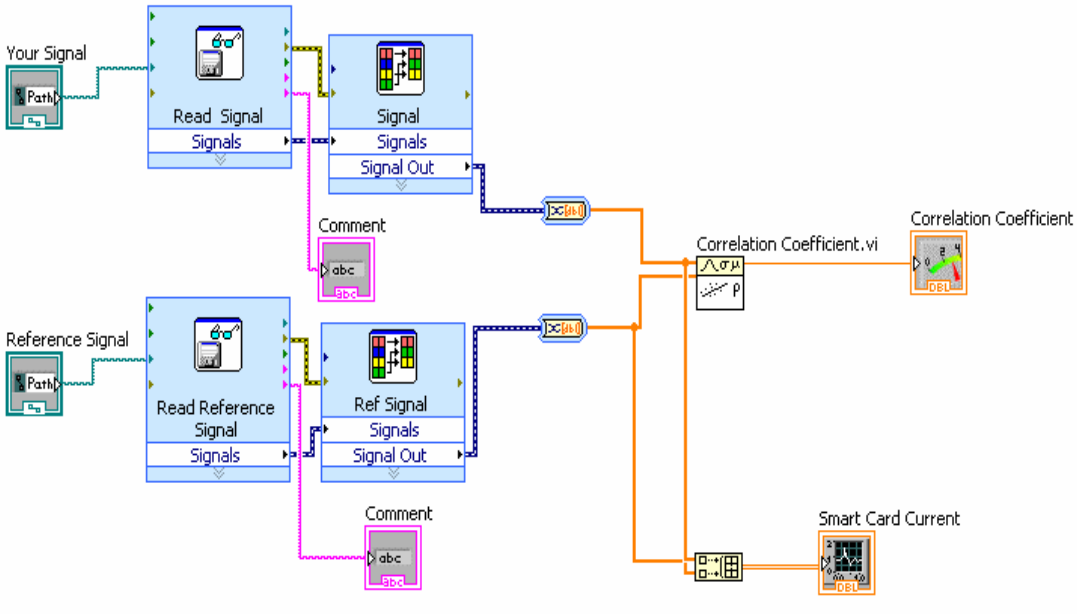
$$corr(x,y) = \frac{\mathrm{cov}(x,y)}{\sqrt{\mathrm{var}(x)\,\mathrm{var}(x)}}.[21] \tag{5.3}$$

Uživatelské rozhraní je znázorněno na Obr 5.15.



Obr 5.15 Uživatelské rozhraní modulu Correlation

Uživatel zvolí naměřený neznámý vzorek a referenční vzorek, vzniklý aritmetickým průměrem mnoha měření známé operace. Na základě velikosti korelačního koeficientu je schopen určit, které operaci daný vzorek pravděpodobně odpovídá.



Obr 5.16 Blokové schéma modulu Correlation

Na Obr 5.16 je znázorněno blokové schéma modulu Correlation. Neznámý i referenční vzorek jsou načteny ze souboru, pomocí bloku Select Signal jsou z nich vybrány hodnoty proudového odběru čipové karty. Hodnoty jsou převedeny na pole a pomocí bloku Correlation Coefficient je určena jejich korelace.

6. Útok výkonovým kanálem na čipovou kartu

Na základě laboratorní úlohy navržené a popsané v předchozích kapitolách je realizován útok výkonovým postranním kanálem na čipovou kartu. Jako cíl tohoto útoku jsou zvoleny algoritmy běží na čipové kartě SIM. Útok na algoritmus AES implementovaný pro čipovou kartu Java Card se nepodařilo uskutečnit z důvodu nekompatibility dostupné karty Java Card s metodami potřebnými pro funkci algoritmu AES. Tento problém je popsán v následující kapitole.

Veškeré měření bylo prováděno na pracovišti vybaveném čtečkou karet Smartmouse/Phoenix, měřícími kartami PCI 1716 a PCLD 8710, PC s procesorem Pentium 4 o frekvenci 3,0GHz a operační paměti 512 MB, programem JSmartCardExplorer, balíčkem programů Smart Card Power Analyzer a analyzovanými sim kartami vydaných různými telekomunikačními operátory.

6.1. Útok na algoritmus AES

Pro útok na algoritmus AES byla k dispozici implementace tohoto algoritmu pro platformu Java Card uvedená v kapitole 2.2.3 a čipová karta specifikace Java Card s operačním systémem JCOP verze 31 a pamětí EEPROM 72kB (tato konfigurace se označuje jako JCOP 31 v22 72K - S/C I/F). Dostupné zdroje se v tvrzí o kompatibilitě karty této specifikace a tříd potřebných pro výkon algoritmu AES liší, dle [22] je tato karta s AES kompatibilní, dle [23] a názorů na odborných fórech kompatibilita není. V mém případě nebylo možné aplet na kartu úspěšně nahrát a nainstalovat, což je nezbytné pro další využití apletu. Řešení problému nepomohly ani pokusy s kompilací zdrojového kódu různými verzemi Javy a Java Card Development Kitu a jejich kombinacemi. Na základě těchto zjištění se přikláním k tvrzení, že Java Card JCOP 31 v22 72K – S/C I/F algoritmus AES nepodporuje.

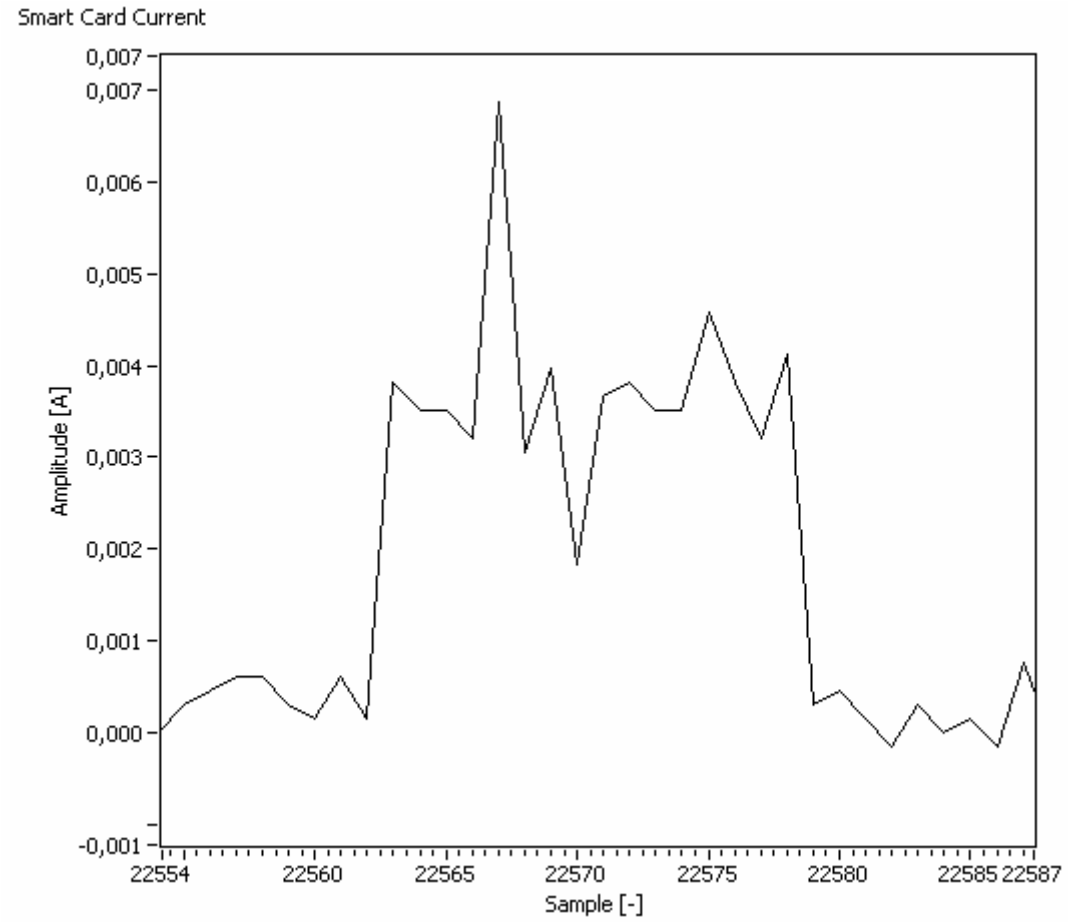
6.2. Útok na SIM karty

Cílem analýzy proudového kanálu SIM karty je zjistit, zda je možno z naměřených hodnot určit informace o procesu, který v daném okamžiku SIM karta vykonává. Pro analýzu byly zvoleny operace s PINem karty – ověřování úspěšné i neúspěšné, vypnutí a zapnutí ověřování PINem, dále čtení ze zapisovatelné paměti $DF_{TELECOM}$. Pro operace s PINem byla vytvořena databáze údajů, jednotlivé průběhy byly navzájem korelovány, aby se potvrdila či vyvrátila jejich souvislost. Každá databáze obsahuje 30 naměřených průběhů.

Následující zobrazené průběhy byly vzorkovány, pokud není u nich uvedeno jinak, v periodě 2 ms, což byla nejnižší možná dosažitelná perioda na dané konfiguraci pracoviště. Průběhy uložené v měřících souborech jsou vzorkovány v periodě 5,5 ms. Nižší vzorkovací periody nebylo možné dosáhnout, problém je popsán v kapitole 5.2.2.

6.2.1. Útok na bezpečnostní mechanismy

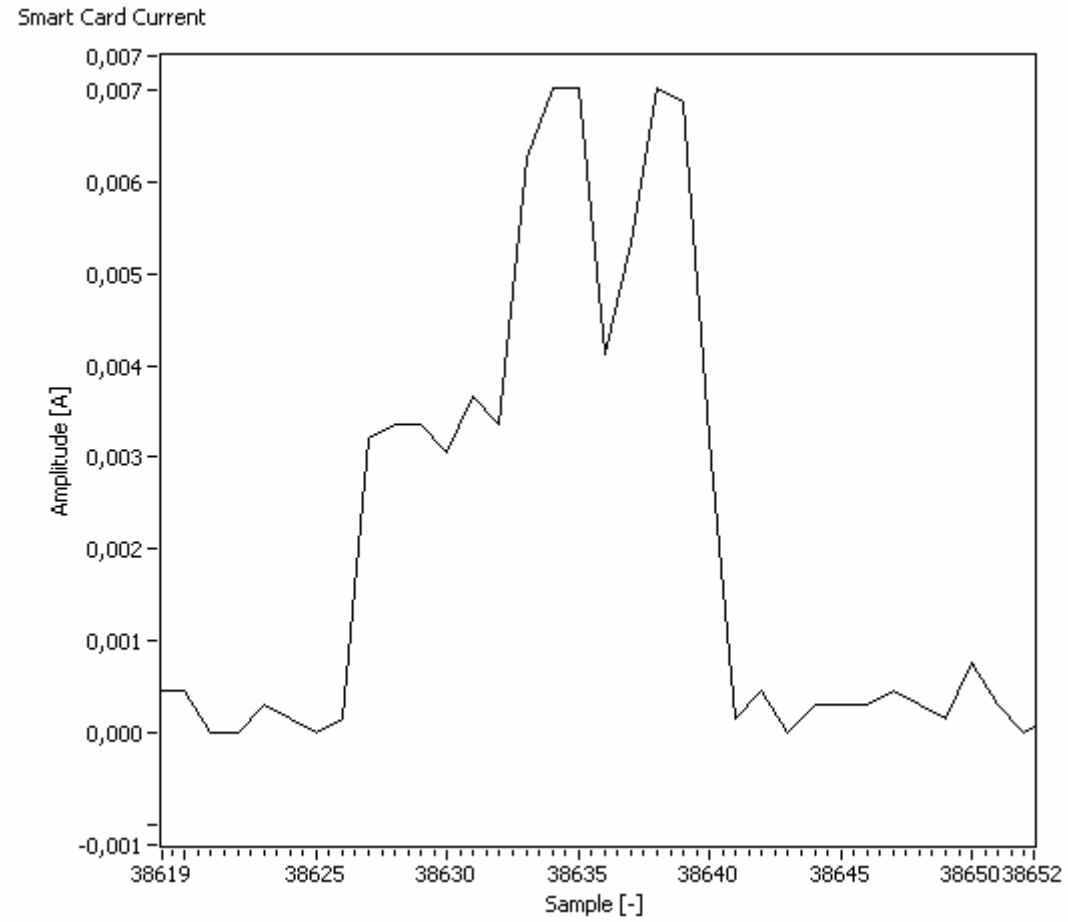
Obr 6.1 znázorňuje aktivitu čipové karty během vykonání instrukce VERIFY PIN, tedy ověření PIN kódu. Jedná se o odezvu karty na příkaz APDU o struktuře CLA = A0, INS = 20, P1 = 00, P2 = 01, LC = 08, pole DATA obsahuje správný PIN čipové karty.



Obr 6.1 Aktivita karty při zpracování instrukce VERIFY PIN

Průměrná délka této operace činí 31 ms.

Obr 6.2 znázorňuje aktivitu čipové karty při vykonání instrukce VERIFY PIN, v tomto případě při kódu lišícím se od správného v jedné hodnotě. Struktura APDU příkazu má tvar CLA = A0, INS = 20, P1 = 00, P2 = 01, LC = 08, DATA = nesprávný kód.

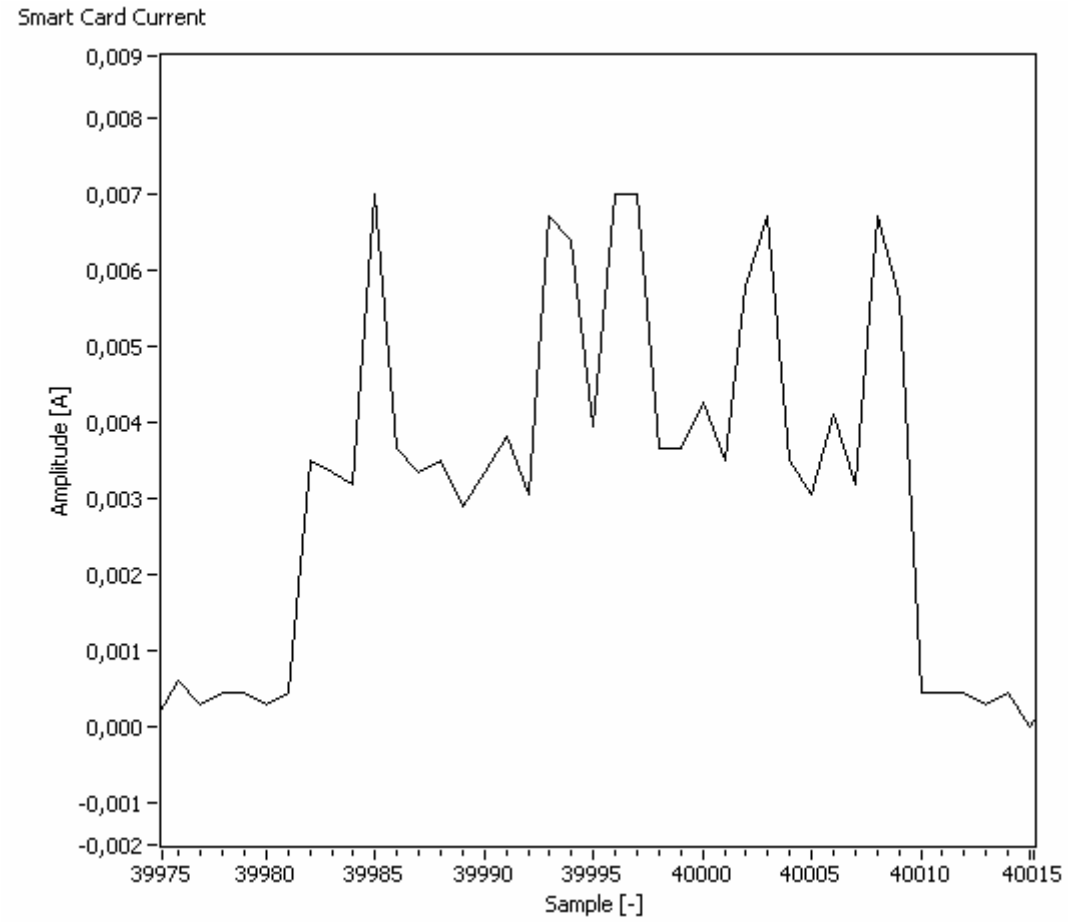


Obr 6.2 Aktivita karty při zpracování instrukce VERIFY PIN – špatný kód

Doba zpracování této instrukce činila taktéž 31ms.

Zvýšený proudový odběr v závěru výkonu instrukce, na Obr 6.2 vzorky 38635 až 38640 značí dekrementaci počtu zbývajících pokusů zadání PINu. V této části se průběh zásadně liší od verifikace správným PINem. Pokud by se v tomto okamžiku přerušilo napájení, tato dekrementace by nenastala. Na základě této techniky by bylo možné provést útok typu brutal force na PIN kód SIM karty.

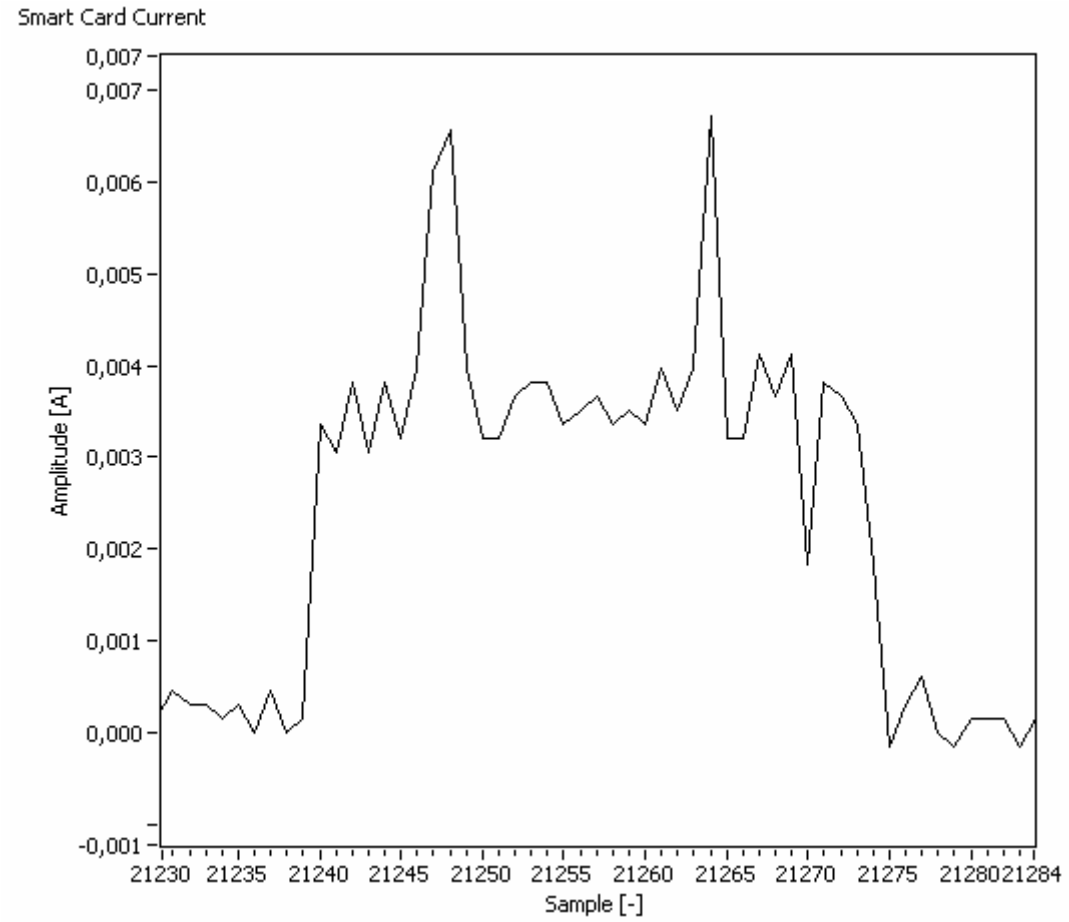
Na Obr 6.3 je znázorněna aktivita SIM karty při zpracování instrukce DISABLE PIN – zrušení požadavku na ověření PINem. Jedná se o APDU příkaz CLA = A0, INS = 26, P1 = 00, P2 = 01, LC = 08, DATA = PIN kód.



Obr 6.3 Aktivita karty při zpracování instrukce DISABLE PIN

Délka zpracování této instrukce činila průměrně 73ms. Prvních 16 vzorků (39981 – 39992) je identických s průběhem signálu instrukce VERIFY.

Obr 6.4 znázorňuje aktivitu procesoru při zpracování instrukce ENABLE PIN – aktivaci verifikace PINem. APDU příkaz má v tomto případě strukturu CLA = A0, INS = 28, P1 = 00, P2 = 01, LC = 08, DATA = nový PIN kód.



Obr 6.4 Aktivita karty při zpracování instrukce ENABLE PIN

Délka výkonu této instrukce činí 78ms. První napěťová špička značí nastavení PINu, vzorky 21260 – 21275 opět odpovídají průběhu proudového odběru instrukce VERIFY PIN.

Pro každou výše uvedenou a popsanou instrukci bylo provedeno 30 měření. Po nezbytném ošetření (filtrace dat) byly vzorky korelovány. Korelace proběhla mezi vzorky stejné instrukce a poté mezi vzorky instrukce opačné. Vzorky byly označeny *n* dle čísla měření, byly vzájemně korelovány dle vzorce

$$\begin{matrix} n * n+1 \\ n \in \{1,2,3...30\} \end{matrix} \quad . \tag{6.1}$$

Pro každý z těchto případů je tak zjištěno 30 korelačních koeficientů, z těchto hodnot je dále spočítán aritmetický průměr. Tyto hodnoty pro příkaz VERIFY PIN jsou uvedeny v Tab 6.1, pro příkaz ENABLE PIN a DISABLE PIN v Tab 6.2.

	VERIFY PIN (správný PIN)	VERIFY PIN (špatný PIN)
VERIFY PIN (správný PIN)	0,783391	0,570097
VERIFY PIN (špatný PIN)	0,570097	0,778653

Tab 6.1 Korelační koeficienty instrukce VERIFY PIN

	DISBLE PIN	ENABLE PIN
DISABLE PIN	0,745929	0,661195
ENABLE PIN	0,661195	0,795354

Tab 6.2 Korelační koeficienty instrukcí DISABLE PIN a ENABLE PIN

Na základě těchto výsledků je provedena jednoduchá výkonová analýza neznámé instrukce SIM karty. Neznámý naměřený průběh proudového odběru karty se koreluje s referenčními známými průběhy, nejvyšší získané hodnoty korelačního koeficientu budou s největší pravděpodobností patřit odpovídající známé instrukci. Pro tento účel je naměřen nový průběh 4 výše zmíněných instrukci, parametry jsou nastaveny stejně, jako u referenčních vzorků, není nám známo, který vzorek odpovídá dané instrukci (vzorky byly naměřeny studentem, který jejich přiřazení k jednotlivým instrukcím zná). Průběhy jsou korelovány se všemi známými vzorky, ze získaných hodnot je získán aritmetický průměr. Výsledné hodnoty zachycuje Tab 6.3.

naměřený průběh	referenční vzorek instrukce	VERIFY PIN correct	VERIFY PIN wrong	DISABLE PIN	ENABLE PIN
vzorek č.1		0,545698	0,765481	0,632147	0,487954
vzorek č.2		0,777895	0,516874	0,564712	0,668947
vzorek č.3		0,598213	0,545719	0,487541	0,814789
vzorek č.4		0,645792	0,458702	0,716543	0,648741

Tab 6.3 Korelační koeficienty neznámých a referenčních průběhů

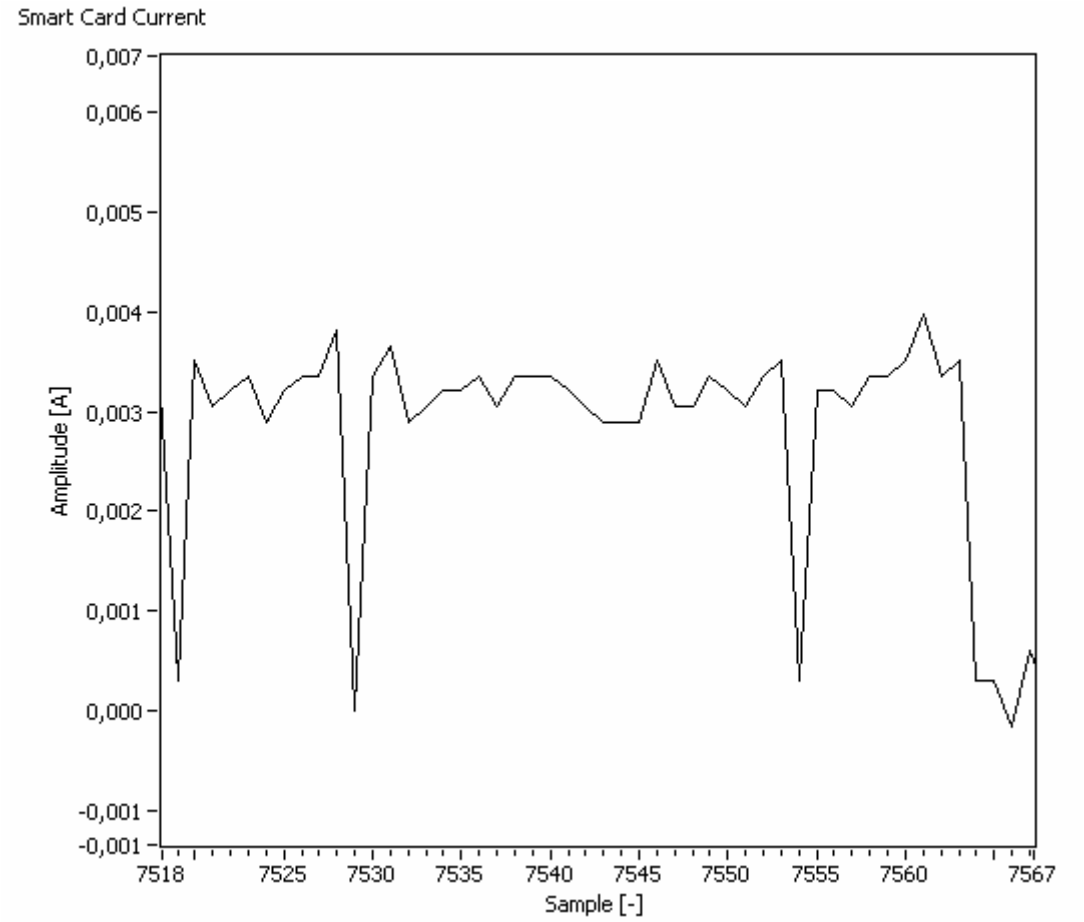
Dle předpokladů by měly nejvyšší korelační koeficienty patřit ke správným referenčním průběhům. Z tohoto tvrzení nám vyplývá, že vzorek č. 1 odpovídá instrukci VERIFY PIN wrong, vzorek č. 2 instrukci VERIFY PIN correct, vzorek č. 3 instrukci ENABLE PIN a poslední vzorek č. 4 instrukci DISABLE PIN. Po kontrole je dokázáno, že toto zjištění je správné.

V některých situacích (např. kombinace VERIFY PIN wrong / DISABLE PIN, VERIFY PIN correct / ENABLE PIN) je korelační koeficient na takové úrovni, která by mohla souhlasit s odpovídající kombinací vzorků. Z tohoto důvodu je potřebné k vyloučení těchto situací provést srovnání se všemi průběhy. Lze předpokládat, že v případě kvalitnějších referenčních vzorků (získané s vyšší vzorkovací periodou) by tyto případy nenastávaly a výsledky získané proudovou analýzou by měly vyšší informační hodnotu.

6.2.2. Útok na přepisovatelnou paměť

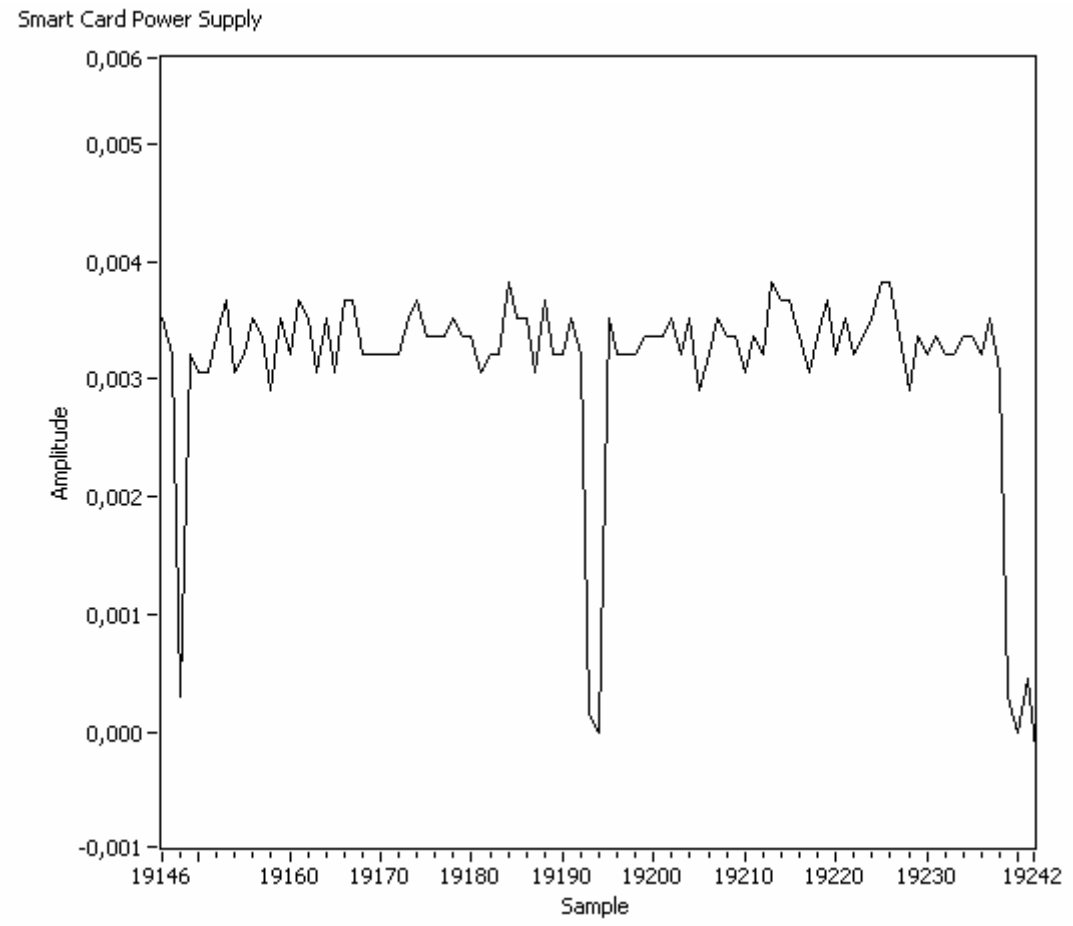
Další část měření se zabývá analýze aktivity čipové karty při čtení údajů z datových souborů. Z paměti $DF_{TELECOM}$ byla provedena proudová analýza vyčítání telefonních kontaktů a textových zpráv.

Obr 6.5 znázorňuje aktivitu karty při čtení uloženého kontaktu. Jedná se odpověď na příkaz APDU CLA=A0 INS=B2 P1=01 LC=04 DAT=1C, této sekvenci předcházeli nutné příkazy SELECT MASTER FILE, SELECT DF TELECOM, SELECT DIALING NUMBERS.



Obr 6.5 Aktivita karty při čtení uloženého kontaktu

Na Obr 6.6 je znázorněna aktivita karty při čtení uložené textové zprávy. Příkaz této instrukce má tvar CLA=A0 INS=B2 P1=01 LE=04 DATA=B0



Obr 6.6 Aktivita karty při čtení uložené textové zprávy

7. Závěr

Cílem této práce je poukázat na možnost útoku různými postranními kanály na čipovou kartu. V teoretické části jsou jednotlivé kanály popsány, včetně jejich principu a popisu dříve realizovaných útoků.

Hlavní prioritou bylo navrhnout experimentální pracoviště a laboratorní úlohu demonstrující útok výkonovým postranním kanálem na čipovou kartu a tento útok realizovat. Pro účely měření je sestavena a modifikována čtečka čipových karet a ve vývojovém prostředí LabVIEW vytvořena sada aplikací pokrývající celý proces měření a vyhodnocení výsledků. Jako cíl útoku jsem zvolil bezpečnostní mechanismy čipové karty specifikace SIM card. Tato karta je zvolena zejména pro svou rozšířenost a možnost demonstrace analýzy postranního kanálu jejího zabezpečení PIN kódem. Prvním krokem tohoto útoku je vytvoření databáze průběhů odběru proudu známých instrukcí založených na autentizaci PIN kódem. Vzorky jednotlivých instrukcí jsou navzájem srovnány a tím je dokázána jejich vypovídací hodnota. V dalším kroku jsou tyto hodnoty srovnávány s neznámými naměřenými průběhy, a na tomto základě stanovena instrukce, během které byly tyto průběhy zjištěny.

Dosažené výsledky tohoto útoku ukazují na závažnost úniku citlivých informací výkonovým postranním kanálů čipové karty.

Literatura

[1] BERTA, István Zsolt, MANN, Zoltán Ádám. Smart Cards – Present and Future. *Híradástechnika, Journal on C* [online]. 2000 [cit. 2009-10-28]. Dostupný z WWW: <http://www.crysys.hu/publications/files/BertaM2000hir.pdf>.

[2] KLUZ, Jakub. *Advanced Encryption Standard (AES) a jeho bezpečná implementace pro kryptografickou čipovou kartu s podporou JavaCard*. [s.l.], 2007. 61 s. Vedoucí diplomové práce Mgr. et Mgr. Jan Krhovják, Ph.D.

[3] BERKES, Jem E. *Side-Channel Monitoring of Contactless Java Cards*. [s.l.], 2008. 79 s. Diplomová práce.

[4] RANKL, Wolfgang , EFFING , Wolfgang . *Smart Card Handbook*. 3rd edition. [s.l.] : [s.n.], 2002. 1088 s.

[5] *Java Card Technology* [online]. c1994-2009 [cit. 2009-10-28]. Dostupný z WWW: <http://java.sun.com/javacard/>.

[6] CHEN, Zhiqun . How to write a Java Card applet: A developer\'s guide. *JawaWorld* [online]. 1999 [cit. 2009-11-13]. Dostupný z WWW: <http://www.javaworld.com/javaworld/jw-07-1999/jw-07-javacard.html?page=1>.

[7] BAR-EL, Hagai. Known Attacks Against Smartcards. *Information Security Analyst* [online]. 2009 [cit. 2009-11-22]. Dostupný z WWW: <http://www.discretix.com/PDF/Known%20Attacks%20Against%20Smartcards.pdf>.

[8] StefanMangard. A Simple Power-Analysis (SPA) Attack on Implementations of the AES Key Expansion. In Pil Joong Lee and ChaeHoon Lim, editors, *Information Security and Cryptology - ICISC 2002, 5th Internatiotnal Conference Seoul, Korea, November 28-29, 2002, Revised Papers*, LNCS 2587, s. 343 – 358. 2003.

[9] Fips-197: Advanced encryption standart. 2001.

[10] ISO/IEC. ISO 14443: Identification cards { Contactless integrated circuit(s) cards { Proximity cards. Final committee draft, International Organization for Standardization, 1999.

[11] MATTHEWS, Adam. Low Cost Attacks on Smart Cards The Electromagnetic Side-Channel. *Next Generation Security Software Ltd.* [online]. 2006 [cit. 2009-11-17]. Dostupný z WWW: <http://www.ngssoftware.com/research/papers/EMA.pdf>.

[12] HE, Sheng. *SIM Card Security* [online]. Bochum : Ruhr-University, 12.07.2007. 18 s. Seminární práce. Ruhr-University of Bochum. Dostupné z WWW: <http://www.crypto.ruhr-uni-bochum.de/imperia/md/content/seminare/itsss07/sim_card_security.pdf>.

[13] *Cardwerk.com* [online]. 2009, 4.11:2009 [cit. 2010-04-19]. ISO 7816-3: Electronic Signals and Transmission Protocols. Dostupné z WWW: <http://www.cardwerk.com/smartcards/smartcard_standard_ISO7816-3.aspx>.

[14] VLÁŠEK, Petr. *Demonstrace šifrování na platformě Java Card*. [s.l.], 2008. 133 s. Diplomová práce. ČVUT v Praze. Vedoucí práce Ing. Jiří Buček.

[15] NOUZÁK, Josef. *Postranní kanály mikroprocesorů*. ČVUT v Praze, 2007. 48 s. Bakalářská práce. ČVUT v Praze, FEL.

[16] *Neoficiální karty* [online]. 10. dubna 2004 [cit. 2010-04-03]. Phoenix a SmartMouse. Dostupné z WWW: <http://www.eremy.onicom.sk/phoenix/popis_d.htm>.

[17] *PCI-1716 datasheet* [online]. 2010/1/27 [cit. 2010-03-23]. Dostupné z WWW: <http://origindownload.advantech.com//ProductFile/1-32ALJS/PCI-1716_DS.pdf>.

[18] *PCLD-8710 datasheet* [online]. 2008/2/26 [cit. 2010-03-23]. Dostupné z WWW: <http://origindownload.advantech.com//ProductFile/1-2KJIZO/PCLD-8710_DS.pdf>.

[19] VLACH, Jaroslav; HAVLÍČEK, Josef; VLACH, Martin . *Začínáme s LabVIEW*. 1.vydání. Praha : BEN - technická literatura, 2008. 247 s. ISBN 978-80-7300-245-9.

[20] TUCCI , Primiano . *Primiano Tucci* [online]. c2007 [cit. 2010-04-09]. JSmartCardExplorer : Cross-platform, low level APDU smart card development. Dostupné z WWW: <http://www.primianotucci.com/default.php?view=112>.

[21] VERMOEN, Dennis ; WITTEMAN , Marc ; GAYDADJIEV, Georgi N. Reverse Engineering Java Card Applets Using Power Analysis.. *LNCS 4462*. Německo : Springer-Verlag, 2007. s. 138 - 149. Dostupné z WWW: <http://www.riscure.com/fileadmin/images/Docs/1287_634_44620138.pdf>. ISSN 0302-9743.

[22] ŠVENDA, Petr. *Petr Svenda* [online]. 14.05.2009 [cit. 2010-04-11]. JavaCard's algorithms support test. Dostupné z WWW: <http://www.fi.muni.cz/~xsvenda/jcsupport.html>.

[23] NXP Identification product range. [online]. 24:10:2007, [cit. 2010-05-11]. Dostupný z WWW: <http://www.ebv.com/fileadmin/products/EBV.TV/NXP_Near_Field_Communication/Z-Card_Identification.pdf>.

Abecední přehled použitých zkratek, veličin a symbolů

AES	Symetrický blokový algoritmus (*Advanced Encryption Standard*)
AID	Identifikátor aplikace (*Application identifier*)
APDU	Aplikační komunikační protokol čipové karty a čtečky(*Applications protocols data units*)
API	Rozhraní pro programování aplikací (*Application Programming Interface*)
AUX	Pomocný vstup (*Auxilliary*)
CAP	Applet připravený pro Java Card (*Converted applet*)
C-APDU	Příkaz protokolu APDU (*Command APDU*)
CL	Bezkontaktní (*Contactless*)
CLA	Instrukční třída datové jednotky protokolu APDU
CLK	Vstup hodinového signálu
CPU	Procesor (*Central processing unit*)
CREF	Simulátor Java Card (*C reference implementation of a JAVA CARD environment*)
DEMA	Diferenciální elektromagnetická analýza (*Differential electromagnetic analysis*)
DES	Symetrický blokový algoritmus (*Data Encryption Standard)*
DPA	Diferenciální výkonová analýza (*Differential power analysis*)
EEPROM	Elektricky mazatelnou semipermanentní paměť (*Electrically erasable programmable read-only memory*)
EMV	Standard pro čipové karty (*Europay, Master Card, VISA*)
GND	Zemnící kontakt (*Ground*)
GSM	Globální Systém pro Mobilní komunikaci (*Global System for Mobile Communications*)
I/O	Datový vstup a výstup (*Input Output*)
IATA	Mezinárodní asociace leteckých dopravců (*International Air Transportation Association*)
ICC	Integrovaný obvod (*Integrated circuit chip*)

ICC-ID	Evidneční číslo SIM karet operátorů (*Integrated Circuit Card ID*)
I_{ih}	Vysoká úroveň proudu v přijímacím módu (*High level input current*)
I_{il}	Nízká úroveň proudu v přijímacím módu (*Low level input current*)
INS	Instrukce datové jednotky protokolu APDU
ISO	Mezinárodní organizace pro normalizaci (*International Organization for Standardization*)
JCOP	Platforma Java karty (*Java Card Open Platform*)
JCRE	Běhové prostředí Java Card (*Java Card Runtime Environment*)
JCVM	Virtuální stroj Java Card (*Java Card Virtual Machine*)
JCWDE	Simulátor Java Card (*Java Card Workstation Development Environment*)
LabVIEW	Laboratorní pracoviště virtuálních přístrojů (*Laboratory Virtual Instruments Engineering Workbench*)
LAI	Identifikátor oblasti (*Location Area Identity*)
LC	Datové pole označující délku jednotky protokolu APDU
LE	Volitelné pole jednotky protokolu APD
NIST	Národní institut standardů a technologie (*National Institute of Standarts and Technology*)
OSI	Propojení otevřených systémů (*Open Systems Interconnection)*
P1	Parametr jednotky protokolu APDU
P2	Parametr jednotky protokolu APDU
PIN	Osobní identifikační číslo (*Personal Identification Numer*)
PUK	Kód pro odblokování (*Personal Unblocking Key*)
RAM	Paměť s libovolným přístupem (*Random access memory*)
R-APDU	Odpověď komunikačního protokolu APDU (*Response APDU*)
ROM	Pevná paměť (*Read-only memory*)
RST	Kontakt reset
SEMA	Jednoduchá elektromagnetická analýza (*Simple electromagnetic analysis*)
SIM	Účastnický identifikační modul (*Subscriber Identity Module*)
SPA	Jednoduchá výkonová analýza (*Simple power analysis*

SW	Parametr odpovědi protokolu APDU (*Status word*)
TMSI	Náhodně přidělené identifikační číslo (*Temporary Mobile Subscriber Identity*)
USB	Univerzální seriové rozhraní (*Universal serial bus*)
VCC	Napájecí kontakt
VI	virtuální přístroj, formát výstupního souboru protředí LabVIEW (*Virtual intrument*)
V_{ih}	Vysoká úroveň napětí v přijímacím módu (*High level input voltage*)
V_{il}	Nízká úroveň napětí v přijímacím módu (*Low level input voltage*)
VISA	Kartová asociace
V_{oh}	Vysoká úroveň napětí v odesílacím módu (*High level output voltage*)
V_{ol}	Nízká úroveň napětí v odesílacím módu (*Low level output voltage*)
VPP	Programovatelný napěťový vstup

Přílohy

A.1 Obsah CD

DP.pdf

Smart Card Power Analyzer

- Oscilloscope.vi
- Filter.vi
- Plotter.vi
- Correlation.vi

Naměřené průběhy

- Referenční data
  - Verify PIN OK
  - Verify PIN KO
  - Disable PIN
  - Enable PIN
- Neznámé průběhy

JavaCardAES

- JCAES.java

A.2 Laboratorní úloha

VÝKONOVÝ KANÁL ČIPOVÝCH KARET

CÍLE

Cílem úlohy je seznámit studenta s novou možností kryptoanalýzy, kterou nabízí přítomnost postranních kanálů. Student provede jednoduchou výkonovou analýzu bezpečnostních mechanismů SIM karty.

ZADÁNÍ

1. Prostudujte problematiku postranních kanálů a čipových karet.
2. Proveďte proudovou analýzu zadaných instrukcí SIM karty.
3. Porovnejte naměřené výsledky s neznámými vzorky a rozhodněte, kterým instrukcím odpovídají.

TEORETICKÝ ÚVOD

Každý nežádoucí způsob výměny informací mezi kryptografickým modulem a okolím se nazývá postranní kanál. Útoky postranními kanály jsou jakékoli útoky, které se nesnaží najít teoretické slabiny v matematické struktuře algoritmu, ale pokouší se o zneužití informací, které unikají přímo z fyzické implementace systému během běhu kryptografického algoritmu. Pokud je totiž uniklá informace nějakým způsobem závislá na tajném klíči algoritmu, může tato informace pomoci útočníkovi klíč odhalit. Cílem útoku postranními kanály nemusí být pouze kryptografický klíč, ale například jen informace o tom, jaký algoritmus se pro šifrování používá, jak dlouho trvá vykonání algoritmu nebo jeho části, či odhalení PINu. Mezi nejznámější postranní kanály patří chybový kanál, časový kanál, elektromagnetický kanál a výkonový kanál

Chybový kanál

Útok vedený tímto způsobem je založený na generování chyb v zařízení, které vlastní útočník nebo na standardních chybách, které se běžně mohou vyskytnout. Výsledky z těchto chybných výpočtů poskytují další informace o tajných parametrech

výpočtu. Vynutit chybu na zařízení lze například změnou voltáže, změnou taktu procesoru nebo aplikováním nějakého druhu ozařování.

Časový kanál

Při tomto útoku využívá útočník faktu, že délky výpočtů prováděných s tajným klíčem jsou na tomto klíči závislé. Na vstup programu posílá útočník různá data a měří, jak dlouho trvá jejich zpracování.

Elektromagnetický kanál

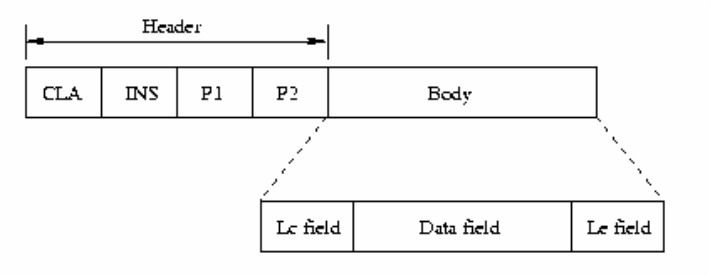
Změny proudů při činnosti zařízení generují střídavé magnetické pole, které, pokud je dostatečně silné, může být detekováno. Útočník umístí do blízkosti zařízení cívku a naměřené elektromagnetické pole posléze analyzuje.

Výkonový kanál

Spotřeba energie zařízení je závislá na vykonávané instrukci a na datech, se kterými instrukce manipuluje. Pokud je útočník schopen sledovat, jak se mění spotřeba zařízení během provádění kryptografických operací, může zjistit nejen jaké operace zařízení provádí, ale také mu tato informace může pomoci k získání tajného klíče, se kterým je kryptografický algoritmus prováděn. Odběrovou analýzu lze rozdělit do dvou základních skupin. Pomocí jednoduché odběrové analýzy je možno zjistit některé informace o systému, jako například jaký algoritmus je používán, identifikovat jeho větší části. Diferenciální výkonová analýza používá statistiku jako nástroj pro zkoumání mnoha křivek měření výkonu, a tím dosahuje odfiltrování nežádoucího šumu, který vzniká měřením. Útok vedený touto metodou se skládá ze dvou částí: sběru dat a z analýzy těchto dat.

Čipová karta nebo smart card je plastová karta kapesní velikosti s integrovaným obvodem (čipem), který je schopen zpracovávat data. To znamená, že zařízení je schopno přijmout data, zpracovat je a vrátit požadované informace. Používají se například tam, kde je potřeba spolehlivé a bezpečné autentizace. SIM karta (SIM je zkratka z anglického *subscriber identity module*) je účastnická identifikační karta která slouží pro identifikaci účastníka v mobilní síti. Základní ochranu SIM karty tvoří přístupový kód PIN.

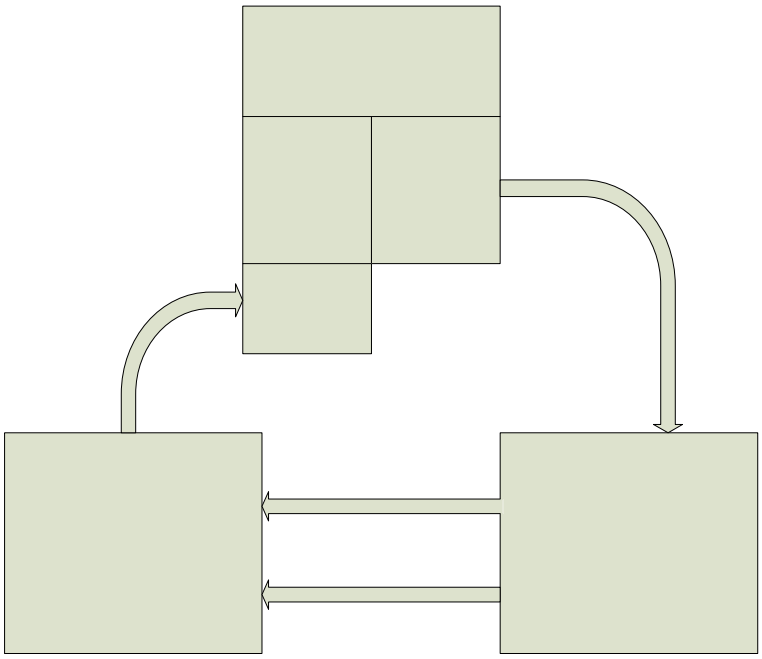
Komunikace čipové karty a čtečky karet či mobilního zařízení probíhá prostřednictvím aplikačního protokolu APDU. Existují dvě jednotky tohoto protokolu – příkaz a odpověď. Struktura příkazu APDU znázorňuje Obrázek 1:



Obrázek 1 Struktura příkazu APDU

LABORATORNÍ PRACOVIŠTĚ

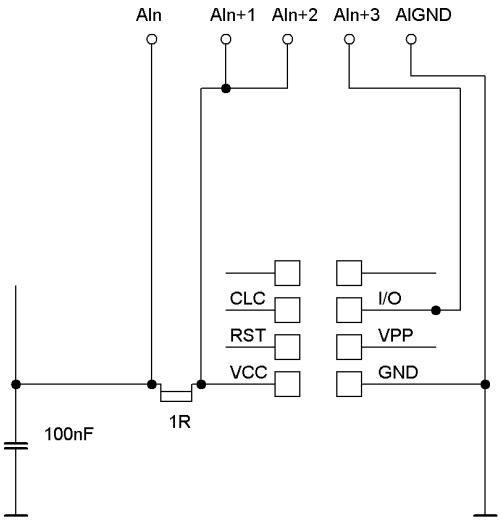
Laboratorní pracoviště se skládá z osobního počítače s programy JSmartCardExplorer, Smart Card Power Analyzer (obsahující moduly Oscilloscope, Filter, Plotter, Correlaiton), měřících karty PCI 1716, terminálu PCLD 8710, čtečky karet a zdroje napětí a SIM karty. Blokové schéma pracoviště znázorňuje Obrázek 2.



Obrázek 2 Blokové schéma pracoviště

Propojení čtečky karet a měřícího terminálu znázorňuje Obrázek 3.

Osc



Obrázek 3 Zapojení čtečky karet a měřícího terminálu

POSTUP MĚŘENÍ

1. Zapojte pracoviště dle výše uvedených schémat. Jako kanál AIn zvolte kanál 0. Vložte SIM kartu do čtečky, napájecí napětí čtečky karet je 12V. Spusťte program JSmartCardExplorer, nastavte komunikační protokol T=0 a připojte se k SIM kartě. Úspěšné připojení je indikováno zeleným polem Card Status.
2. Spusťte modul Oscilloscope, nastavte výchozí kanál, počet kanálů (3). Pole Gain a Sampling Period nastavte na hodnotu 0.
3. V programu JSmartCardExplorer nastavujte hodnoty příkazu APDU pro následující instrukce. Pin kód je „12341234“, neměňte ho! Nastavte první z příkazů a přejděte ke kroku 4, poté pokračujte pro další příkazy:
    a. VERIFY PIN CLA=A0 INS=20 P1=0 P2=01 LE=08 DATA=<nesprávný PIN kód>,
    b. totéž, ale zadejte správný PIN,
    c. DISABLE PIN CLA=A0 INS=26 P1=0 P2=01 LE=08 DATA=< PIN kód>,
    d. ENABLE PIN CLA=A0 INS=28 P1=0 P2=01 LE=08 DATA=< PIN kód>.

4. V modulu Oscilloscope zadejte název souboru dle použité instrukce. Spusťte modul tlačítkem Run, přepněte se do programu JSmartCardExplorer, odešlete jeden z výše uvedených příkazů a tlačítkem Stop zastavte běh modulu. Tímto

postupem získáte soubory s průběhy proudového odběru pro všechny 4 výše uvedené instrukce.

5. Modulem Filter vyfiltrujte ze získaných souborů užitečná data. Počet vzorků nastavte na 30.
6. Modulem Plotter zobrazte získané i neznámé průběhy, pokuste se dle průběhů určit, které instrukce si navzájem odpovídají.
7. Modulem Correlation vzájemně srovnejte všechny naměřené i neznámé vzorky. Podle velikosti korelačních koeficientů určete instrukce neznámých vzorků.