Analiza kodu programu mikroprocesora na podstawie rejestracji zmian napięcia zasilającego

Andrzej Kwiecień, Michał Maćkowski

Streszczenie: Ochrona systemów informatycznych przed dostępem do informacji niejawnych przez osoby niepowołane jest zagadnieniem niezwykle ważnym. Praca traktuje o pewnym aspekcie tego problemu wynikającym stąd, iż twórca na przykład oprogramowania systemu wbudowanego nie zdaje sobie sprawy z tego, iż możliwe jest, jeśli nie w całości to w pewnym zakresie na pewno, poznanie kodu programu bez bezpośredniej ingerencji w pamięć programu. Prezentowana praca ma charakter eksperymentalny. Analiza zmian napięcia na bazie prezentowanych założeń badawczych pozwoliła na dość znaczną skuteczność rozkodowania programu bez ingerencji w strukturę wewnętrzną. Uzyskane rezultaty zwracają uwagę na możliwości niekontrolowanego dostępu do kodów programów. Uzyskane rezultaty uzasadniają konieczność szukania i opracowywania odpowiednich metod zabezpieczania oprogramowania.

Słowa kluczowe: emisja ujawniająca, inżynieria wsteczna, kod programu, mikrokontroler, zaburzenia elektromagnetyczne.

Abstract: *Protection of computer systems from an unauthorized access to the classified information is a very essential issue. The research deals with some aspect of this problem resulting from the fact, that for instance, an author of a software for embedded system is not aware that it is possible to identify partly or entirely, program code. This paper has an experimental character. The research focuses on analysis of microprocessor voltage supply changes. Such analysis based on the presented research assumptions allowed for a rather high efficiency of decoding program without interference in the internal structure of microprocessor. The obtained results show, that there is a possibility of uncontrolled access to program codes. Thus, it is necessary to search for and develop appropriate methods used for protecting program.*

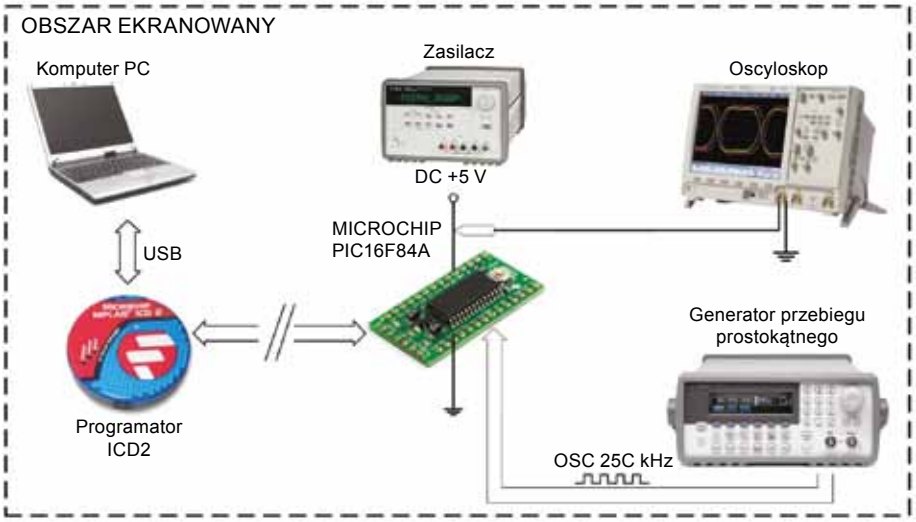
Key words: *compromising emanation, reverse engineering, program code, microcontroller, electromagnetic disturbances.*

1. Wstęp

Panujący nieustanny trend w kierunku jak największej miniaturyzacji i integracji doprowadził do powstania układów scalonych wielkiej skali integracji. W ostatnich latach zaczęto nawet mówić o systemach określanych mianem SOC (ang. *System On Chip*), scalających w jednej obudowie elektroniczne układy cyfrowe, analogowe (także radiowe) oraz cyfrowo-analogowe, co może świadczyć o wysokim obecnym poziomie zaawansowania technologii produkcji układów scalonych. Wysoka skala integracji oraz ciągłe zwiększanie częstotliwości pracy układów mikroprocesorowych powoduje generowanie impulsów prądowych o coraz większych amplitudach i coraz krótszych czasach narastania na liniach zasilających i wejściach/wyjściach układów elektronicznych. Impulsy takie generowane są poprzez równocześnie przełączające się tysiące/miliony tranzystorów wewnątrz struktury układu scalonego. Propagacja takich prądów poprzez przewody i ścieżki na płycie PCB (ang. *Printed Circuit Board*) do innych układów elektronicznych może powodować problemy z prawidłowym ich funkcjonowaniem. Z drugiej jednak strony sumaryczny prąd pobierany przez wszystkie bramki w trakcie wykonywania pojedynczego rozkazu może wskazywać, jaki rozkaz w danym momencie jest wykonywany.

Podczas dokonywania analizy kodu programu mikroprocesora na podstawie rejestracji zmian napięcia zasilającego układ traktowany jest jak „czarna skrzynka" wykonująca pewien ciąg instrukcji zapisanych w pamięci programu. W idealistycznym podejściu można założyć, że działanie zaimplementowanego algorytmu polega na przetworzeniu pewnych danych wejściowych i zwróceniu wyniku bez jakiegokolwiek oddziaływania urządzenia z otoczeniem. W rzeczywistości każde urządzenie zasilane energią elektryczną i przetwarzające sygnały cyfrowe wpływa na otoczenie oraz inne urządzenia poprzez emisję zaburzeń elektromagnetycznych na drodze emisji przewodzonej i promieniowanej [1, 2, 6]. Zmiany poboru mocy przez urządzenie oraz emitowanie pola elektromagnetycznego można nazwać ubocznymi efektami wykonania algorytmu, które tworzą stowarzyszony kanał informacyjny, dostarczający dodatkowej wiedzy na temat jego działania.

Pozyskiwanie informacji na temat działania urządzenia poprzez wpływ na jego pracę lub monitorowanie parametrów jego działania nazywane jest atakiem typu *side-channel*. Istnienie kanału „bocznego", którym pozyskano takie informacje, jest zazwyczaj niezamierzone i wynika z budowy urządzenia lub technologii, w jakiej je zbudowano. Przykładem tego może być każde urządzenie zasilane energią elektryczną – w tym przypadku emitowane sygnały elektromagnetyczne są rezultatem przepływu prądów i istniejących napięć. Kiedy sygnały używane są do przesyłania informacji z jednego punktu do kolejnego drogą przewodzoną lub promieniowaną w postaci fal elektromagnetycznych w sposób celowy, mówimy, że tworzą kanał transmisyjny. W przypadku, kiedy sygnały powstają



Rys. 1. Schemat stanowiska badawczego

w sposób niezamierzony i ponadto niosą one informacje na temat stanu pracy urządzenia, mamy do czynienia z kanałem „bocznym" i ulotem informacji. Rozchodzenie się takich sygnałów nazywane jest często emisją ujawniającą (ang. *compromising emanation*). Za emisję ujawniającą możemy więc uznawać wszystkie niezamierzone sygnały, które w przypadku ich przechwycenia i przeanalizowania powodują ujawnienie przetwarzanej informacji. Źródłem tych sygnałów może być dowolne urządzenie elektryczne służące do wysyłania, odbierania, przechowywania lub przetwarzania informacji. Niniejszy artykuł traktuje o pewnym aspekcie tego problemu, wynikającym stąd, iż twórca na przykład oprogramowania systemu wbudowanego (ang. *embedded system*) nie zdaje sobie sprawy z faktu, iż możliwe jest w pewnym stopniu poznanie np. kodu programu, bez bezpośredniej ingerencji w pamięć programu mikrokontrolera.

Jak wiadomo, mikroprocesor realizujący program wykonuje pewne powtarzające się czynności, polegające na cyklicznym pobieraniu kodów rozkazów z pamięci i wczytywaniu ich do układu sterowania mikroprocesora, a następnie realizacji rozkazu, którego kod został pobrany. Wszystkie te operacje, realizowane przez mikroprocesor, synchronizowane są przez sygnał zegarowy, czego wynikiem jest wzmożona aktywność elementów wchodzących w skład jednostki centralnej w każdym cyklu maszynowym. To właśnie między innymi aktywność magistrali danych i adresowej, dekodera rozkazów, jednostki arytmetyczno-logicznej, powoduje zmiany stanów wyjść bramek wchodzących w skład tych elementów, a tym samym przyczynia się do dynamicznego poboru energii ze źródła zasilania. Przedstawione w poprzednich pracach autorów [3, 4] wyniki badań dotyczyły analizy wpływu stanu magistrali danych, argumentu rozkazu, wyniku operacji, adresu rozkazu w pamięci na przebieg napięcia zasilającego w trakcie trwania kolejnych cykli rozkazowych oraz rozpoznawania rozkazów operujących na argumentach o wartości zero.

W niniejszym artykule dokonano natomiast analizy kodu programu mikroprocesora 8-bitowego na podstawie rejestracji zmian napięcia zasilającego, skupiając się na rozpoznawaniu rozkazów operujących na dowolnych argumentach. Otrzymane rezultaty badań mogą zostać uogólnione również na inne układy mikroprocesorowe realizujące zapisany w pamięci kod programu.

2. Stanowisko i procedura badawcza

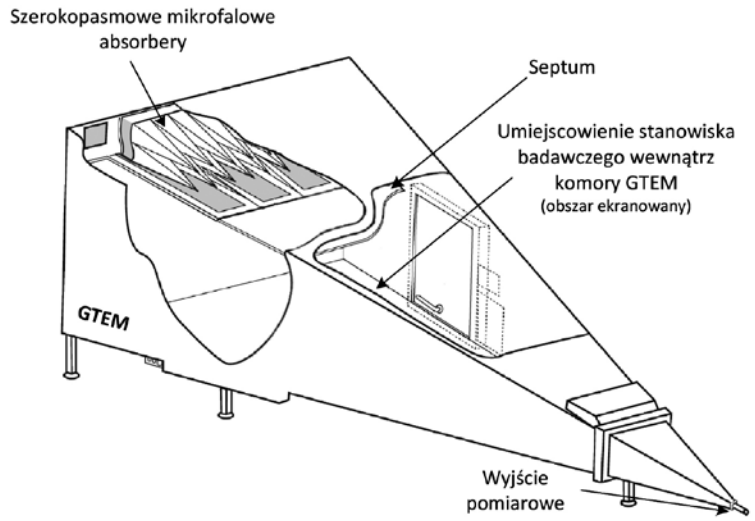
Stanowisko badawcze przedstawione na rysunku 1 składało się z mikrokontrolera firmy Microchip o oznaczeniu PIC16F84A, do którego podłączono zasilanie oraz zewnętrzny generator przebiegu prostokątnego o częstotliwości 250 kHz. Do zasilania mikrokontrolera wykorzystano zasilacz stabilizowany firmy Agilent. Sonda oscyloskopu została podłączona do linii zasilających mikrokontroler, w celu monitorowania na nich spadków napięcia podczas realizacji kolejnych rozkazów programu. Szczegółowe sposoby pomiaru zaburzeń w liniach zasilających oraz sposoby analizy otrzymanych wyników w dziedzinie czasu i częstotliwości zostały przedstawione w poprzednich pracach autorów [4, 5].

W trakcie badań całe stanowisko badawcze umieszczone zostało w komorze ekranowanej (rys. 2) – komorze GTEM (ang. *Gigahertz Transverse ElectroMagnetic*), która zapewniła całkowitą separację obszaru pomiarowego od wpływu zewnętrznych pól elektromagnetycznych.

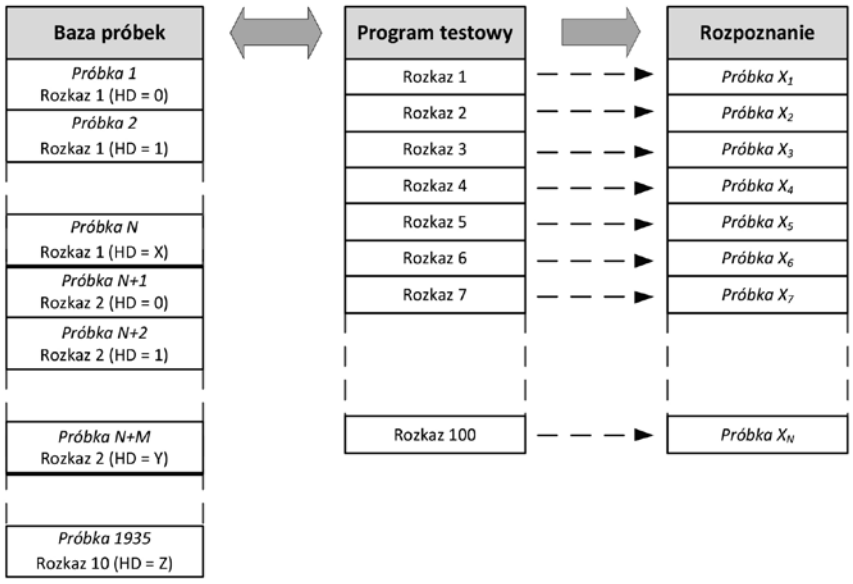
W pracy [3] opracowano metodę pozwalającą na rozpoznawanie rozkazów operujących na argumentach o wartości zero na podstawie analizy zaburzeń napięcia zasilającego. W tym celu w pierwszej kolejności należy zmierzyć przebieg napięcia zasilającego procesor w trakcie działania całego programu. Kolejnym etapem jest wycięcie części przebiegu czasowego odnoszącego się do badanego rozkazu. Następnie zapisywana jest wartość minimalna i maksymalna napięcia dla trzech pierwszych cykli maszynowych – łącznie zapisywanych jest sześć wartości. W ten sposób została utworzona baza próbek, w której każdy rozkaz mikrokontrolera scharakteryzowany jest przez 6 punktów – trzy wartości maksymalne i trzy wartości minimalne napięcia, zmierzonego w poszczególnych cyklach maszynowych Q1, Q2 oraz Q3. Wykazano wówczas, że skuteczność tak opracowanej metody do rozpoznawania rozkazów operujących na argumentach o wartości zero wynosi 91%.

W niniejszym artykule autorzy rozszerzyli zakres badań omawiany we wspomnianej publikacji, uwzględniając zarówno badany rozkaz, jak i argument tego rozkazu. W przypadku tworzenia bazy próbek wykorzystanej następnie w procesie rozpoznawania rozkazów operujących na dowolnych argumentach, poza kodem rozkazu w bazie, należy uwzględnić zarówno argument rozkazu, jak również wynik jego działania. Proces tworzenia próbek w takim przypadku jest procesem bardzo czasochłonnym i wymagającym wnikliwej syntezy każdego rozkazu nie tylko na podstawie specyfikacji technicznej, ale również na podstawie zmian mierzonego napięcia zasilającego. Z tego też powodu w procesie tworzenia bazy próbek w trakcie badań skupiono się na dziesięciu rozkazach z listy instrukcji procesora: ADDLW, ANDWF, BSF, CLRF, COMF, INCF, MOVF, MOVLW, NOP, XORLW.

Wcześniejsze badania autorów wykazały, że na przebieg napięcia w trakcie realizacji pierwszego oraz trzeciego cy-



Rys. 2. Budowa komory GTEM oraz umiejscowienie stanowiska badawczego



Rys. 3. Proces rozpoznawania rozkazów operujących na dowolnych argumentach

klu maszynowego nie mają wpływu bezpośrednio argument rozkazu oraz wynik operacji. Okazuje się, że na przebieg napięcia w trakcie realizacji cyklu rozkazowego poza kodem operacji wpływ mają również odległości Hamminga pomiędzy stanem magistrali danych a argumentem rozkazu (cykl maszynowy Q1) oraz odległość Hamminga pomiędzy argumentem rozkazu a wynikiem operacji (cykl maszynowy Q3). W takim przypadku do stworzenia bazy próbek może zostać wykorzystany schemat oparty na odległości Hamminga. Informacja ta pozwala na znaczne uproszczenie budowy bazy próbek wykorzystanej następnie w procesie rozpoznawania rozkazów.

Na rys. 3 przedstawiono schemat budowy bazy próbek oraz proces rozpoznawania rozkazów operujących na dowolnych argumentach. Baza zawiera próbki opisujące 10 rozkazów, gdzie próbki od 1 do *N* opisują rozkaz 1, następnie *M* próbek opisuje rozkaz 2 itd. W ten sposób dla wymienionych wyżej rozkazów zbudowano bazę składającą się z 1935 próbek, wykorzystaną następnie w procesie rozpoznawania kodu programu mikroprocesora.

Po utworzeniu bazy próbek wygenerowano trzy programy testowe zbudowane ze 100 rozkazów, które służyły do określenia skuteczności metody rozpoznawania rozkazów. Programy zostały zbudowane z wykorzystaniem tych samych dziesięciu rozkazów, które zostały wykorzystane przy tworzeniu bazy próbek. Zarówno kolejność występowania rozkazów w programach testowych, jak również argumenty rozkazów zostały wybrane w sposób losowy.

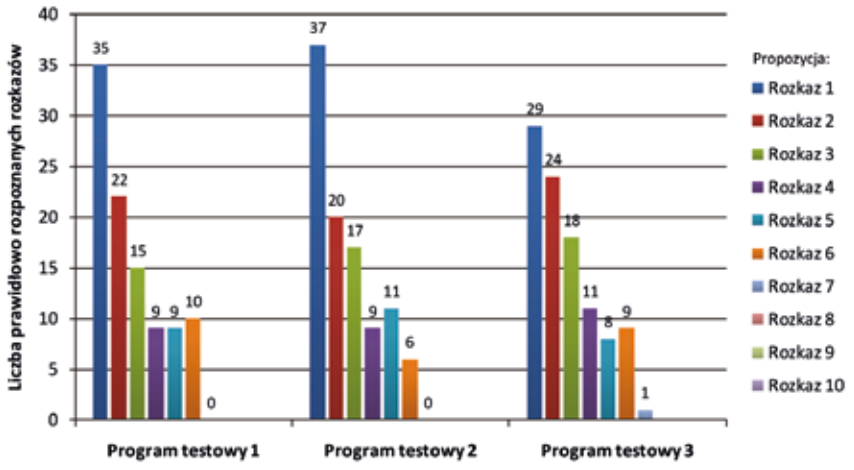
Wyniki badań

Zgodnie z procedurą badawczą każdy rozkaz programu testowego został porównany ze wszystkimi próbkami w utworzonej wcześniej bazie 1935 próbek.

Na rysunku 4 zaprezentowano wykres liczby prawidłowo rozpoznanych rozkazów w kolejnych propozycjach rozkazów otrzymanych w wyniku zastosowanej metody, dla wszystkich trzech programów testowych. Dalsza analiza wykresu pozwala zauważyć, że największa liczba rozkazów została rozpoznana prawidłowo w trzech pierwszych próbach. W tabeli 1 przedstawiono liczbę prawidłowo rozpoznanych rozkazów w kolejnych propozycjach rozkazów oraz wyznaczono ich wartości średnie dla programów testowych 1, 2 oraz 3. Na podstawie otrzymanych wyników stwierdzić można, iż dla kolejnych trzech pierwszych prób średnio udało się prawidłowo rozpoznać kolejno 33,67%, 22,00% i 16,67% rozkazów w rozpatrzonych programach testowych. Dla kolejnych kolumn średnia liczba prawidłowo rozpoznanych rozkazów była mniejsza od 10%.

Wyniki skuteczności zastosowanej metody do rozpoznawania rozkazów dla trzech programów testowych przedstawiono w tabeli 2. Jako skuteczność rozumiany jest tutaj stosunek liczby rozkazów, które zostały prawidłowo rozpoznane w jednej z trzech pierwszych propozycji rozkazów, do całkowitej liczby próbek w programie testowym. Dla programu testowego 1 w 72% przypadków rozkaz prawidłowo rozpoznany znalazł się wśród propozycji trzech pierwszych rozkazów najbardziej podobnych, zwróconych w wyniku zastosowanej metody. Dla pozostałych programów testowych osiągnięta skuteczność rozpoznawania rozkazów wynosi odpowiednio 74 i 71%.

Spadek skuteczności rozpoznawania rozkazów w porównaniu ze skutecznością rozpoznawania rozkazów operujących na argumentach o wartości zero jest spowodowany w tym przypadku wpływem przetwarzanych danych na przebieg napięcia zasilającego w trakcie realizacji cyklu rozkazowego konkretnego rozkazu. Powoduje to, że różne rozkazy procesora dla konkretnych wartości: stanu magistrali danych, argumentów rozkazu oraz wyniku operacji, mogą charakteryzować się takimi samymi lub bardzo zbliżonymi przebiegami napięć mierzonymi w torze zasilającym mikroprocesor. Opracowana metoda badawcza, dla każdego rozkazu w programie testowym, na podstawie wcześniej przygotowanej bazy próbek zwraca propozycje rozkazów rozpoznanych wraz z podaniem wartości podobieństwa. W badaniach przyjęto, że skuteczność metody do rozpoznawania kodu programu mikroprocesora będzie określona na podstawie liczby rozpoznanych rozkazów w trzech pierwszych uzyskanych propozycjach rozkazów.



Rys. 4. Liczba prawidłowo rozpoznanych rozkazów w kolejnych propozycjach rozkazów, otrzymanych w wyniku zastosowanej metody dla programów testowych 1, 2 oraz 3

Tabela 1. Liczba prawidłowo rozpoznanych rozkazów w kolejnych propozycjach rozkazów oraz ich wartości średnie dla programów testowych 1, 2 i 3

	Liczba prawidłowo rozpoznanych rozkazów						
	Rozkaz 1	Rozkaz 2	Rozkaz 3	Rozkaz 4	Rozkaz 5	Rozkaz 6	Rozkaz 7
Program testowy 1	35	22	15	9	9	10	0
Program testowy 2	37	20	17	9	11	6	0
Program testowy 3	29	24	18	11	8	9	1
Wartość średnia	33,67	22,00	16,67	9,67	9,34	8,34	0,34

Wnioski

W niniejszym artykule autorzy wykazali możliwość częściowego określenia aktualnie wykonywanego przez mikroprocesor rozkazu na podstawie rejestracji zmian napięcia zasilającego mikroprocesor. Wykazano także możliwość częściowego określenia odległości Hamminga pomiędzy stanem magistrali danych i argumentem rozkazu oraz argumentem rozkazu i wynikiem operacji. Możliwość określenia odległości Hamminga w tym przypadku sprowadza się do rozpoznania liczby zmian bitów wykonanych na argumencie rozkazu w następstwie wykonania operacji. W artykule poruszono problem bezpieczeństwa układów mikroprocesorowych, a w szczególności istnienia zagrożenia dla programów zapisanych w pamięci tych układów oraz przetwarzanych przez nie informacji.

W wyniku przeprowadzonych badań wykonano bazę 1935 próbek, którą na-

Tabela 2. Porównanie skuteczności zastosowanej metody do rozpoznawania rozkazów operujących na dowolnych argumentach, dla programów testowych 1, 2 i 3

	Liczba rozkazów rozpoznanych [1]	Liczba rozkazów nierozpoznanych [2]	Skuteczność metody rozpoznawania rozkazów wyrażona w %
Program testowy 1	72	28	72
Program testowy 2	74	26	74
Program testowy 3	71	29	71

1. Liczba rozkazów rozpoznanych prawidłowo w trzech pierwszych propozycjach rozkazów
2. Liczba rozkazów, które nie zostały rozpoznane prawidłowo w trzech pierwszych propozycjach rozkazów

stępnie porównywano z trzema programami testowymi. Każdy program testowy składał się ze 100 losowo wybranych instrukcji oraz losowo wybranych argumentów. Założono, że jeżeli po porównaniu badanego rozkazu z bazą próbek rozkaz ten znajduje się na jednym z trzech pierwszych miejsc, to dany rozkaz został rozpoznany prawidłowo. Przy takich założeniach skuteczność zaprezentowanej metody dla kolejnych programów testowych jest następująca: 72%, 74% i 71%. Otrzymane rezultaty badań mogą zostać uogólnione również na inne układy mikroprocesorowe realizujące zapisany w pamięci kod programu.

Niniejszy artykuł oraz zaprezentowane wyniki badań należy traktować również jako próbę zwrócenia uwagi na zagrożenia będące wynikiem zjawiska emisji ujawniającej, czyli wszelkiego rodzaju niezamierzonych sygnałów, których przechwycenie i przeanalizowanie powoduje ujawnienie przetwarzanej przez urządzenie informacji. Z drugiej zaś strony powstaje pytanie, czy uzyskane rezultaty mają jedynie zwrócić uwagę twórców oprogramowania na zagrożenia dotyczące możliwości zastosowania inżynierii wstecznej, celem rozkodowania programów, czy też należy doskonalić opisaną metodę (lub opracować nową), aby uzyskać większą skuteczność i dzięki temu w pełniejszy sposób wskazywać niebezpieczeństwa i jednocześnie budować zabezpieczenia przed niepowołanym dostępem do wersji źródłowej oprogramowania? Szukanie odpowiedzi na tak postawione pytanie jest jak na razie kwestią otwartą.

Literatura

[1] Bao F.: *Breaking Public Key Cryptosystems on Tamper Resistant Devices In the Presence of Transient Faults*. Proceedings of the 5th International Workshop on Security Protocols, LNCS 1361, Springer-Verlag (1997).

[2] Biham E., Shamir A.: *Differential fault analysis of secret key cryptosystems*. LNCS 1294, Springer-Verlag (1997).

[3] Kwiecień A., Maćkowski M., Skoroniak K.: *Instruction prediction in microprocessor unit*. In: A. Kwiecień, P. Gaj, P. Stera (eds.): CN 2011, CCIS 160, pp. 427–433, Springer-Verlag, Berlin Heidelberg (2011).

[4] Kwiecień A., Maćkowski M., Skoroniak K.: *The analysis of microprocessor instruction cycle*. In: A. Kwiecień, P. Gaj, P. Stera (eds.): CN 2011, CCIS 160, pp. 417–426, Springer-Verlag, Berlin Heidelberg (2011).

[5] Maćkowski M., Skoroniak K.: *Instruction prediction in microprocessor unit based on power supply line*. In: A. Kwiecień, P. Gaj, P. Stera (eds.): CN 2010, CCIS 79, pp. 173–182, Springer-Verlag, Berlin Heidelberg (2010).

[6] Mangrad S., Oswald E.,Popp T.: *Power Analysis Attacks – Revealing the Secrets of Smart Cards*. ISBN 978-0-387-30857-9, p. 338. Springer (2007).

Praca była współfinansowana ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego (nr umowy o dofinansowanie projektu: UDA-POKL.04.01.01-00-106/09).

prof. nzw. w Pol. Śl. dr hab. inż. Andrzej Kwiecień; dr inż. Michał Maćkowski – Politechnika Śląska, Instytut Informatyki, andrzej.kwiecien@polsl.pl, michal.mackowski@polsl.pl

artykuł recenzowany