Data Security in Microprocessor Units

Andrzej Kwiecień[1], Michał Maćkowski[1], and Marcin Sidzina[2]

[1] Silesian University of Technology, Institute of Computer Science
[2] University of Bielsko-Biała, Department of Mechanical Engineering Fundamentals
{akwiecien,michal.mackowski}@polsl.pl, msidzina@ath.bielsko.pl

Abstract. Protection of computer systems from an unauthorized access to the classified information is a very essential issue. Security of IT systems can concern various aspects, such as software security, connected with ensuring its confidentiality, as well as preventing its modification. Thanks to the developed methods, it is possible to analyze the program code based on the disturbances of voltage supply, which occur during the execution of the program. Moreover, it also enables to recognize the numbers of bits changes on microcontroller data bus, as a result of realized instruction. In such context, it can constitute a potential threat for data processed by microcontroller program or embedded systems. The presented method is very similar to simple power analysis method, which is very effective in relation to cryptographic algorithms, whose execution in many cases depends on processed data. The results indicate that presented method is an effective and low-cost attack, due to its simplicity in many real applications. Moreover, the research results inspire to study carefully the ways and methodology for developing software and hardware, which should reduce the possibility of software reverse engineering.

Keywords: reverse engineering, data security, program code, microcontroller, conducted emission, electromagnetic disturbances, electromagnetic interference, simple power analysis.

1 Introduction

The emission of electromagnetic field is a phenomenon that is contributed to electric current flow, which in turns is the base for working the whole electronic and electric devices. In many cases, based on the electromagnetic field changes, it is possible to deduce about the work of devices being its source. What is more, the properties of electromagnetic field allow for its remote registration and further analysis. The phenomenon of electromagnetic emission, which can provide the information about the work of electric and electronic devices, is called compromising emanation or compromising radiation [1, 2]. As the authors notice, the security of processing and storing information [3–5], and also safety of communication protocols [6, 7] is one of the most important problems of information technology.

Protection against remote, non-invasive reading of information based on detection of electromagnetic waves, constitute an important area, which first of

A. Kwiecień, P. Gaj, and P. Stera (Eds.): CN 2013, CCIS 370, pp. 495–506, 2013.
© Springer-Verlag Berlin Heidelberg 2013

all is connected with creating of software and designing computer devices. The use of electronic devices for processing the information, very often confidential, caused that compromising emanation has become of a special meaning.

For decades, issues connected with the reduction of electromagnetic emission of disturbances that can adversely affect the work of other devices, as well as ensuring the suitable immunity of devices to electromagnetic interference are the subject of intense study, described extensively in [8, 9]. It is obvious, that the level of signals which enable to reconstruct the information processing by a device, can be considerably lower than the level of disturbances causing an incorrect work of other devices. Contrary to the issues related to the electromagnetic compatibility, the information about compromising emanation based on the electromagnetic emission is not very often published or is confidential, and the access to it in many cases is very difficult.

Microprocessor units, currently used in network devices as network controllers may also be considered as advanced chips responsible for data processing and reconstruction of transmitting frames. Such units can also be source of electromagnetic disturbances and can, for example provide information about work of the network controller.

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 a program code (the following executed instruction), or to provide some information about the data processed by the system, without any direct interference into the program memory of microcontroller. The research results presented in the previous authors studies [10, 11] concern the analysis of the microprocessor program code based on the nature of disturbances in the voltage course. However, the research did not focus on details of data processing, but mostly on the possibility of reconstructing the program code.

This study presents the analysis of data processed by microprocessor during execution of program code, based on the voltage supply changes. The expected results of the research may lead to a broader look at the ways and methodologists of software development for microprocessor systems and more widely used of embedded systems.

2 Security of Microprocessor Systems

In recent years the issue connected with the security of microcontroller systems has become of special meaning. It is due to the fact that the number of electronic devices, which use microprocessor units of various manufacturers increase continuously. The systems responsible for processing the strictly confidential data, are particular noteworthy.

Obtaining information about the operation of a device through the influence on its work or monitoring the parameters of its activity, is called side-channel attack. The existence of the side-channel through which such information is obtained, is usually unintended and results from the construction of a device or technology, in which it was built.

The research on side-channel attacks were introduced in the second half of 90's and were related mainly to the attacks on cryptographic systems. The assumptions of those days as to the security of such systems, were based mainly on the correct execution of an algorithm implemented in the cryptographic system, and the lack of data distortion during its work. As it was proved, those assumptions were wrong [12, 13], and the possibility to impact on the device work and forcing mistakes in its work may simplify the cryptanalysis in a significant way. Side channel attacks can be divided according to: methods of influence, the way of interference into a device, and also the way of analysis the received results. Considering the way of interference into device the attacks can be divided into 2 groups:

1. Active attacks involving the attempt to take control over the system and observation of its response by:
   - voltage supply changes (voltage dips, short interruptions and voltage variations),
   - generating fault states, for example, due to the disturbances of clock signal timing microcontroller,
   - interacting with electromagnetic field of determined frequency and intensity.
2. Passive attacks involving the observation and measurement the microcontroller parameters during its normal work. The most frequent parameters to be monitored are as follows:
   - power consumption by a device,
   - signal levels on interface lines,
   - time for executing a particular part of algorithm,
   - electromagnetic disturbances spreading via conducted and radiated emission.

The presented method of analysis of data processed by microprocessor, which is based on voltage supply changes can be included into group of non-invasive attacks, which means with no direct physical access to the internal components included in the microcontroller. Although there is an access to a device (system), but any attempts to reconstruct the information processed by the central unit cannot break the structure of integrated circuit and leave the traces of activity. This type of attack is reduced to measure working parameters and observing inputs and outputs of the unit.

Moreover, the presented method is very similar to simple power analysis method (SPA) [14–17]. Such analysis requires very often the exact knowledge of the algorithm implemented into the device – which is regarded, at the same time, as its main disadvantages. As mentioned above, the method is very effective in relation to cryptographic algorithms, whose execution in many cases depends on processed data (e.g. in smartcards). A good example can be an algorithm, in which some instructions are executed only when a particular condition is performed. In this case, if the following instructions are characterized by various power consumption, then SPA analysis can enable to determine for example the value of the encryption key.

In papers [16, 17] it was noticed that the level of disturbances (noise) has a great impact on the effectiveness of extracting sensitive information, for example the encryption key. In current work, this problem was solved by placing the whole test bench in shielded area which helped to minimize the influence of electromagnetic disturbances on parameters to be measured. The test bench is presented in Sect. 3. Such solution increases the effectiveness of data analysis processed by the microcontroller.

What is more, some authors [15, 17] often use the Hamming weight parameter, which provides the information about the number of bits in the string being different from zero. This paper, in contrast, uses Hamming distance parameter. In information theory, the Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. Put another way, it measures the minimum number of substitutions required to change one string into the other. The advantages of using it are described in the further part of the paper.

This paper presents the analysis of data processed by the microcontroller, based on the voltage supply changes, which is free of these defects. In this case, change of the character of power consumption by the system is not a result of executing the other instructions (e.g. during program branching in IF conditional instruction), but is due to the processing the other data by the system. At the same time, the number of attempts required to restore the data processed by the processor is less than in the method of differential power analysis (DPA) [18, 19].

3 Test Bench and Research Procedure

Test bench consisted of Microchip microcontroller with PIC16F84A signature. This processor represents very numerous group of Mid-Range processors. There are about 85 models in this serie of processors. The main differences among microcontrollers included in Mid-Range class concern: the size of data memory (RAM/EEPROM) and Flash program memory, the maximum work frequency of circuits, and available numbers of communication modules. The access to the program memory, which can store 1024 instructions is via 14 bits program bus. To supply the microprocessor Agilent stabilized power supply was used. Oscilloscope probe was connected to microcontroller supply lines to monitor voltage drop during realization of following instructions. The test bench, for the period of research was placed in shielded cell – GTEM (*Gigahertz Transverse ElectroMagnetic*), which provided total separation of measuring area from external electromagnetic influences. The exact description of the test bench, methods used to measure voltage disturbances and ways to analyze the obtained results in the time and frequency domain, were presented in previous paper of the authors [20].

In paper [11] the authors presented and described the method, which enable to analyse the program code based on the voltage supply changes. Depending on a program code and instruction argument, the different disturbances on power supply lines in a particular machine cycles were noticed.

The first step to do this is to measure the microprocessor voltage supply waveform while running the entire program. The next step is to cut the part of time waveform referring to the instruction being tested. Then the minimum and maximum value of the voltage for the first three machine cycles is saved – a total of six values are saved. In this way the sample database was created, in which each microprocessor instruction is characterized by 6 points – three maximum and three minimum values of voltage, measured in particular machine cycles *Q1*, *Q2*, and *Q3*. Moreover, the fourth machine cycle was not considered due to the fact that in the last cycle (*Q4*) another (next) instruction is also fetched from the program memory, thus not only currently realized instruction but also next instruction has the influence on the current flow (shape).

Presented method based on measuring the voltage changes during the realization of each microprocessor instruction, and then on creating a database of samples. The database of voltage disturbances samples was next used to recognize the instructions for any program executed by the microprocessor (reverse engineering). The detailed description of a manner for instruction recognition was presented in papers [10, 11]. The authors achieved the effectiveness of instruction recognition on any arguments at the level of 72 %.

During the research, it became clear that voltage waveform during instruction execution depends not only on the instruction code, but also on instruction argument, especially when the first (fetching the argument) and the third (executing instruction by arithmetic-logic unit) machine cycle is realized.

Suppose that the result of some operation is zero (such state is maintained on data bus till receiving another argument). Then, while monitoring the voltage change during executing another instruction, i.e. `MOVLW` (move constant value to *W* register), regardless whether it is instruction `MOVLW 1`, `MOVLW 2`, `MOVLW 4`, `MOVLW 8`...`MOVLW 128`, it can be seen that the voltage waveform is the same – in each of these cases only one bit is changed on data bus.

In order to explain the changes that occur in the voltage waveform during instruction realizations, it is helpful to use the parameter called Hamming distance (HD). Figures 1a–c presents the voltage waveforms on the microprocessor supply lines in the middle of executing `MOVLW 0`, `MOVLW 15`, `MOVLW 255` instructions (the instructions arguments were selected to present the values of HD, which equal 0, 4 and 8). Comparing the voltage waveforms in the figures it can be seen that mainly the differences during the first machine cycle (first 4th μs) occur. The voltage shapes for 2, 3 and 4 cycle are the same for all considered instructions. In this case `MOVLW` instruction does not involve ALU, hence the third machine cycle is the same in all cases. The similar analysis can be conducted for other instructions with various arguments. These issues were presented in more details in paper [20].

In this case, in order to create a database of samples a schema based on Hamming Distance was used. This knowledge enabled to simplify significantly the creation of samples database, which was used afterwards in the process of instruction recognition. Hence, it is not necessary to create a sample for an instruction with each possible argument, but only include Hamming Distance.

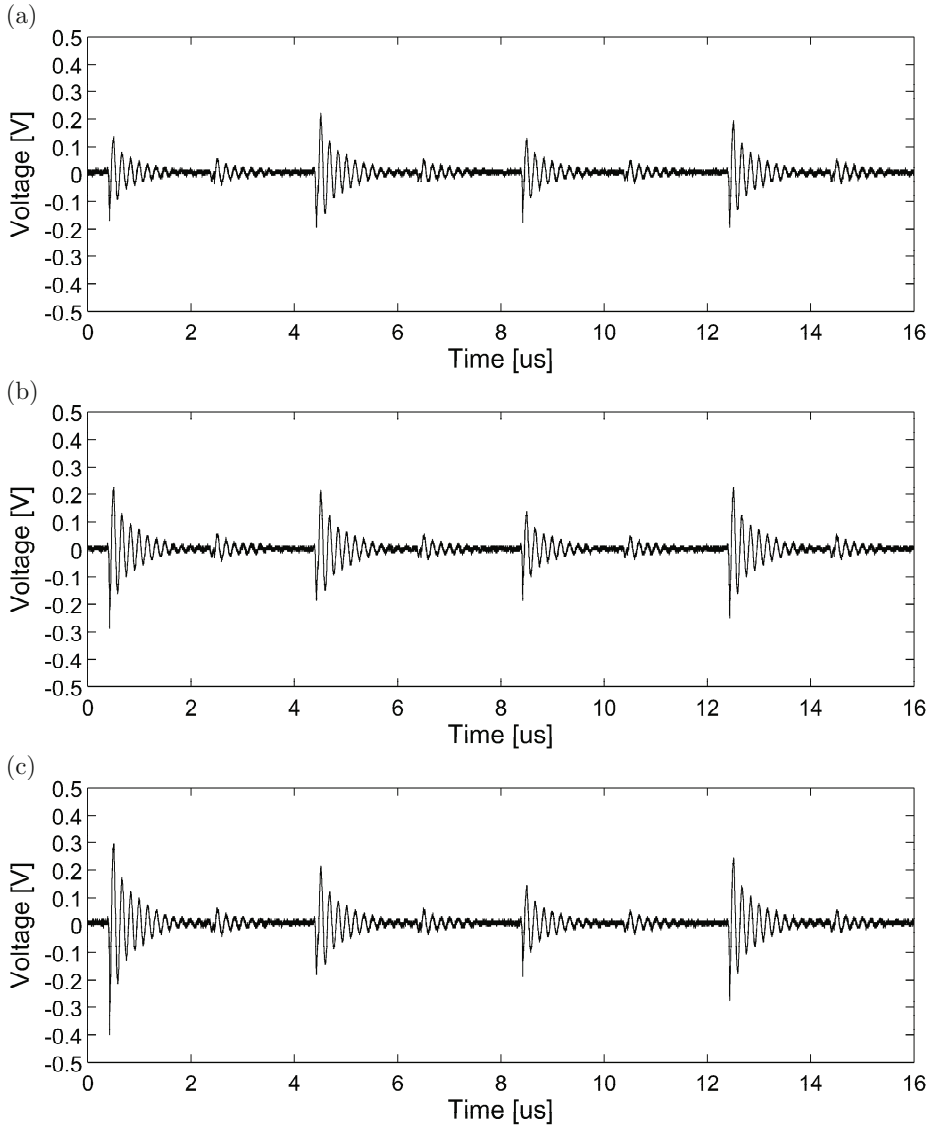

Fig. 1. The voltage waveforms on the microprocessor power supply lines during the execution: of (a) `MOVLW 0` – {HD = 0}, (b) `MOVLW 15` – {HD = 4}, (c) `MOVLW 255` – {HD = 8} instructions

All programs used for creating a database (excising a part of time waveform of a tested instruction, searching for a minimum and maximum value of voltage, estimating the voltage average value, etc.) were implemented in Matlab [20]. A serious problem was the acquisition of individual waveforms for each instructions, including all Hamming distance. At the current stage of the research this process was performed manually. However, in the future the authors intend to automate this process, which may reduce significantly time of creating the database of samples.

Figure 2 presents the schema of compiling database of samples and the process of instructions recognition executing the arguments with the value of any kind. The database consists of samples describing 10 instructions, where samples from 1 to N describe instruction 1, then M samples describe instruction 2, etc. According to the above schema, it was possible to create a database for the previously mentioned instructions, consisting of 1935 samples and used next in the process of microprocessor program code recognition. Then, each of the instructions of a test program (in the form of 6 minimum and maximum voltage values) was compared to all samples in database using the method of the least squares and correlation [10]. It enabled to reveal that a sample from database, which was the most similar to the tested instruction, was then typed as a recognized instruction.

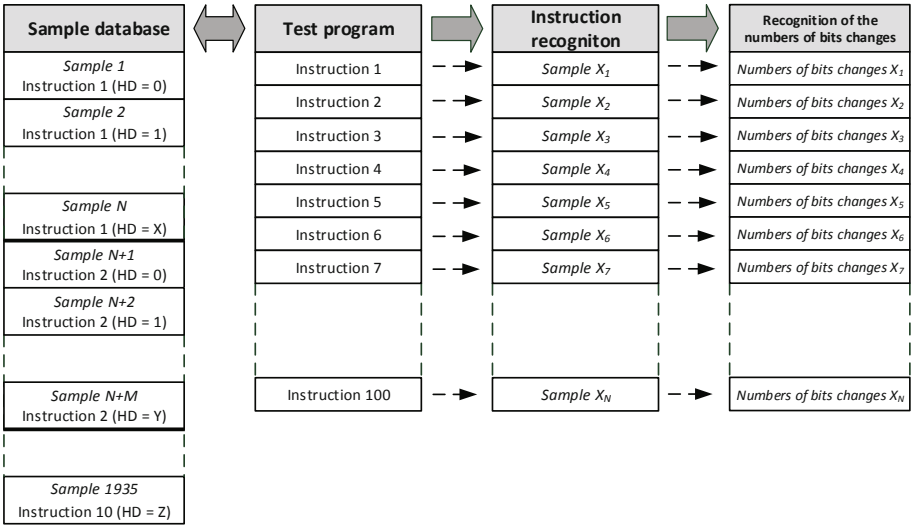

Fig. 2. The process of instruction recognition and the numbers of bits changes on data bus as a result of fetching and executing instructions

Next, as the result of fetching the argument and executing operation, it was possible to recognize the numbers of bits changes on data bus for each instruction. In order to correlate the database trace with the voltage trace in real program, the procedure in Matlab environment was written. This procedure returns such

samples and the Hamming distance values from database, which are the most similar to the instruction in the test program.

The usefulness of this information is presented in the further part of the paper.

4 The Research Results

Having compiled the database three test programs consisting of 100 instructions were generated, and were used for determining the effectiveness of the method of recognition the numbers of bits changes on data bus, as the result of fetching and executing the instruction. Both, the order of instructions in the test programs and instruction argument were selected at random.

The database of samples besides using it for recognising following instructions of a program written into the microcontroller memory, can be also used for receiving some information on data processed by the microprocessor. As mentioned before, in the process of compiling the database one of the useful information was that Hamming Distance had the influence on the power supply course during execution of instruction cycle, between the state of data bus and instruction argument (machine cycle *Q1*) and between operation result and instruction argument (machine cycle *Q3*). In such case, the correct instruction recognition in test program, except the instruction name, provides also information about operations realized by the instructions on data bus.

Table 1 presents the fragment (first 10 out of 100) of recognition the numbers of bits changes on data bus, as the result of fetching the argument and executing the instruction for test program 1. The following columns presents:

- Columns 1 and 2: instructions order and the following instructions in the test program.
- Columns 3 and 4: states of registers *W* and *F*, after executing each operation. The register whose contents was modified, as a result of a particular instruction execution, was marked with bold font. At the same time, the contents of this register is parallel to the state of data bus after executing the whole instruction cycle.
- Columns 5 and 6: the real numbers of bits changes on data bus after receiving an argument in the machine cycle *Q1*, and executing the operation in the machine cycle *Q3*.
- Columns 7 and 8: the recognized numbers of bits changes on data bus after receiving an argument in the machine cycle *Q1*, and executing the operation in the machine cycle *Q3*.

The numbers of bits changes on data bus during receiving an argument and executing operation is analysed on the example of the three first instructions of test program 1 (Table 1):

- Before the instruction `MOVLW .196` is executed, the state on data bus has the value 0. In the first machine cycle occurs fetching of argument 196 (binary 11000100) and setting it on data bus, where 3 bits change their state on

Table 1. Results of recognition the numbers of bits changes on data bus, as the result of fetching the argument and executing instructions for test program 1

Following instructions of test program 1		State of register		The real numbers of bits changes		The recognized numbers of bits changes	
		W	*F*	The numbers of bits changed on data bus after fetching the instruction argument	The numbers of bits changed in data bus after executing the instruction	The numbers of bits changed on data bus after fetching the instruction argument	The numbers of bits changed in data bus after executing the instruction
1	MOVLW .196	**196**	0	3	0	3	0
2	BSF 15,1	196	**2**	3	1	3	1
3	ADDLW .224	**164**	2	4	2	4	3
4	BSF 15,2	164	**6**	4	1	3	1
5	ADDLW .107	**15**	6	5	3	5	3
6	COMF 15,1	15	**249**	2	8	2	8
7	NOP	**15**	249	6	4	6	5
8	MOVLW .247	**247**	249	5	0	5	0
9	MOVF 15,1	247	**249**	3	0	4	0
10	MOVF 15,1	247	**249**	0	0	0	0
11	...	...	...	...	...	...	...

the value of 1 (3 bits changes), in the third machine cycle ALU sets the operation result (value 196) on data bus, which maintains the same state (0 bits changes).

- In the `BSF 15,1` instruction, occurs overwriting of the state on data bus 196 (binary 11000100), through the contents of *F* register, with the address 15 equals 0 – resetting the state of data bus (3 bits changes). Next, `BSF` instruction set a chosen bit (1) of register *F* in the machine cycle *Q3* and set the result (binary 00000010) on the data bus (1 bit change).
- Before the instruction `ADDLW .224` is executed, the state on data bus has the value of 2 (binary 00000010), then the argument 224 (binary 11100000) is received and the new value is overwritten on the data bus (4 bits changes). Next, argument 224 is added into the contents of *W* register with the value of 196, the result – 164 (binary 10100100) is set on data bus, and the argument fetched in the first machine cycle is overwritten (2 bits changes).

The analysis of all instructions in test programs 1, 2 and 3 was conducted in the similar way, and then the real numbers of bits changes, for each instruction on data bus, were compared to recognised numbers of changes (correctly recognition is marked in gray color in Table 1). The results of this comparison are presented in Table 2. For the following 100 instructions of test programs 1, 2 and 3 the effectiveness of proper recognition of bits changes on data bus after fetching instruction argument is as follows: 81 %, 85 % and 80 %. For the remaining instructions, the maximum mistake in the numbers of correctly recognised bits changes is one. The similar remarks refer to the numbers of bits changes achieved on data bus in the third machine cycle. For three test programs the numbers

of correctly recognised bits changes on data bus being the result of fetching the instruction argument and executing the operation are approximately 82 % and 71.67 %.

Table 2. Numbers of correctly recognized bits changes on data bus after fetching the instruction argument and executing the instructions for test program 1, 2 and 3

	Numbers of correctly recognized bits changes on data bus after fetching the instruction argument (out of 100 instructions)	Numbers of correctly recognized bits changes on data bus after executing the instruction (out of 100 instructions)
Program 1	81	69
Program 2	85	72
Program 3	80	74
Average value	**82**	**71.67**

5 Conclusion

Presented results show the high efficiency of the numbers of correctly recognised bits changes on data bus resulting from fetching the argument and executing the instruction. During the research, it was noticed that the numbers of bits changes on data bus after fetching the argument in the machine cycle 1, and after executing the instruction and setting the result on data bus in the machine cycle 3, caused the significant changes of voltage amplitude measured on microprocessor power supply, during the instruction cycle. Based on these observations, an attempt was made in order to render the information about the amount of bits changes on data bus, focusing only on the voltage waveforms measured for the following instructions in test programs. For three test programs the numbers of correctly recognised bits changes on data bus during the machine cycle *Q1* and *Q3* are approximately 82 % and 71.67 %. In this case, it is of course impossible to answer directly the question concerning the argument value, for which a particular instruction was executed. However, this method can be used if the implementation of algorithm is known (presented method recognises the algorithm in the first step), and if there is a possibility to select the input data, i.e. in cryptographic algorithm. Based on the several iteration for different values of input vector (plain text) and the observation of the result, there is a possibility to recognise the instruction argument, which in case of cryptanalysis may be an additional source of information about the encryption key. For example XOR operation, being the part of many developed cryptographic algorithms, especially many stream and block ciphers, is undoubtedly prone to such kind of attacks. The essential properties of this function is its commutativity. This feature causes that XOR instruction is very often used in cryptography.

The paper presents the problem of security of microprocessor systems, and it discusses also the possibility of threats for programs written into the system

memory and the information processed. The developed method for analysing the microprocessor program code based on the voltage supply changes, enables also to recognise the numbers of bits changes on microcontroller data bus, as a result of realized instruction. In such context, it can constitute a potential threat for data processed by microcontroller program or embedded systems. What is more, such threat can be the issue for cryptology research, where the security of cryptographic algorithms is analysed not only from the mathematical point of view, but also as the real hardware and software implementations.

Acknowledgments. This work was supported by the European Union from the European Social Fund (grant agreement number: UDA-POKL.04.01.01-00-106/09).

References

1. Kuhn, M.G.: Security limits for compromising emanations. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 265–279. Springer, Heidelberg (2005)
2. Tanaka, H.: Information leakage via electromagnetic emanations and evaluation of tempest countermeasures. In: McDaniel, P., Gupta, S.K. (eds.) ICISS 2007. LNCS, vol. 4812, pp. 167–179. Springer, Heidelberg (2007)
3. Kwiecień, A., Stój, J.: The cost of redundancy in distributed real-time systems in steady state. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 106–120. Springer, Heidelberg (2010)
4. Pieprzyk, J., Hardjono, T., Seberry, J.: Fundamentals of computer security. Springer, Heidelberg (2003) ISBN 978-3-540-43101-5
5. Stera, P.: Company's data security – case study. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 290–296. Springer, Heidelberg (2010)
6. Gaj, P., Jasperneite, J., Felser, M.: Computer communication within industrial distributed environment – a survey. IEEE Transactions on Industrial Informatics 9(1), 182–189 (2013)
7. Sidzina, M., Kwiecień, B.z.: The algorithms of transmission failure detection in master-slave networks. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2012. CCIS, vol. 291, pp. 289–298. Springer, Heidelberg (2012)
8. Clayton, P.: Introduction to electromagnetic compatibility, 2nd edn. John Wiley and Sons, New Jersey (2006) ISBN: 978-0-471-75500-5
9. Montrose, M., Nakauchi, E.: Testing for EMC compliance: approaches and techniques. Wiley-IEEE Press, Canada (2004) ISBN: 978-0-471-43308-8
10. Kwiecień, A., Maćkowski, M., Skoroniak, K.: Instruction prediction in microprocessor unit. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 427–433. Springer, Heidelberg (2011)
11. Kwiecień, A., Maćkowski, M., Skoroniak, K.: Reverse engineering of microprocessor program code. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2012. CCIS, vol. 291, pp. 191–197. Springer, Heidelberg (2012)
12. Bao, F., Deng, H., et al.: Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults. In: Christianson, B., Crispo, B., Lomas, M., Roe, M. (eds.) IW 1997. LNCS, vol. 1361, pp. 115–124. Springer, Heidelberg (1998)

13. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
14. Ahn, M., Lee, H.-J.: Experiments and hardware countermeasures on power analysis attacks. In: Gavrilova, M.L., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3982, pp. 48–53. Springer, Heidelberg (2006)
15. Dabbish, E.A., Messerges, T.S., Sloan, R.H.: Investigations of power analysis attacks on smartcards. In: WOST 1999, Proceedings of the USENIX Workshop on Smartcard Technology, Chicago (1999)
16. Vermoen, D., Witteman, M., Gaydadjiev, G.N.: Reverse engineering java card applets using power analysis. In: Sauveron, D., Markantonakis, K., Bilas, A., Quisquater, J.-J. (eds.) WISTP 2007. LNCS, vol. 4462, pp. 138–149. Springer, Heidelberg (2007)
17. Mayer-Sommer, R.: Smartly analyzing the simplicity and the power of simple power analysis on smartcards. In: Paar, C., Koç, Ç.K. (eds.) CHES 2000. LNCS, vol. 1965, pp. 78–92. Springer, Heidelberg (2000)
18. Broujerdian, M., Doostari, M., Golabpour, A., et al.: Differential power analysis in the smart card by data simulation. In: Proceedings of the 2008 International Conference on MultiMedia and Information Technology, MMIT 2008, USA, pp. 817–821 (2008)
19. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
20. Kwiecień, A., Maćkowski, M., Skoroniak, K.: The analysis of microprocessor instruction cycle. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 417–426. Springer, Heidelberg (2011)