Reverse Engineering of Microprocessor Program Code

Andrzej Kwiecień, Michał Maćkowski, and Krzysztof Skoroniak

Silesian University of Technology, Institute of Computer Science,
Akademicka 16, 44-100 Gliwice, Poland
{akwiecien,michal.mackowski,krzysztof.skoroniak}@polsl.pl
http://www.polsl.pl/

Abstract. This paper has an experimental character. Theoretical backgrounds presented here allow creating a research method. The research focus 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.

Keywords: reverse engineering, program code, microcontroller, conducted emission, electromagnetic disturbances, Hamming distance, embedded system.

1 Introduction

The current continual trend towards more and more miniaturization and integration has led to existence a very large scale integration circuits. In recent years, SOC (*System On Chip*) which merge into one chip analog, digital, mixed-signal, and often radio-frequency functions, has become very popular solutions – especially in embedded systems. This may indicate a present level of advancement of production technology of integrated units. The high integration scale and continuous increase of microprocessor circuit frequency cause the current peaks to be generated with higher amplitudes and shorter rise times on power supply and I/O lines of the electronic circuits. Such impulses are generated by thousands/millions of transistors inside the integrated unit, which are switching simultaneously. Propagation of such currents through wires and paths on a PCB (*Printed Circuit Board*) to other electronic systems may cause the problems with their normal functioning. On the other hand, the total current drawn by all the gates during execution of a single instruction, may indicate what instruction is currently executed.

During the analysis of microprocessor program code based on the measurement of power supply changes, the system is treated as "a black box" executing a sequence of instructions stored in a program memory. In idealistic approach it can be assumed that the operation of the implemented algorithm consists of processing some input data and returning the result without any interaction between the device and environment. In fact, any device powered by electric energy and processing digital signals affects the environment and other devices through the emission of electromagnetic disturbances in the way of conducted and radiated emission [1,2,3,4]. Changes in power consumption of the device and emitting the electromagnetic field can be described as side effects of realization of the algorithm, which make up affiliate channel providing additional knowledge about the algorithm.

The paper deals with an aspect of this problem resulting from the fact, that for example an author of software of embedded system is not aware of the possibility to recognize, to a certain extent, a program code without direct interference into microcontroller program memory. Microprocessor units currently used in network devices as network controllers can be also considered as advanced chips being responsible for data processing and reconstruction of transmitting frames. Such units can also be source of electromagnetic disturbances. The authors in previous papers [5,6] presented the research referring to analysis of the influence of data bus, instruction operand, its result and address in memory to the supply power line during subsequence instruction cycles and recognition of instruction that are realized with the arguments with the value of zero.

In this paper, the authors analyze 8-bit microprocessor program code based on the power supply changes, and focus on recognition of instructions that operate on arguments with the value of any kind.

2 Test Bench and the Research Procedure

Test bench consisted of Microchip microprocessor with PIC16F84A signature, connected to the power supply and an external square-wave generator with a frequency of 250 kHz. 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, have been presented in the previous papers of the authors [6,7].

In papers [5,8] the authors developed a method for recognizing instructions that operate on arguments with the value of zero, based on the analysis of voltage disturbances. 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. Based on the obtained results, it was proved that the method developed to recognize instructions operating on arguments with the value of zero, is effective in 91%.

In this paper the authors extended the scope of the research already discussed in [5] by taking into consideration both instruction and the instruction operand. In case of compiling the database of samples used for recognizing instructions that operate on arguments with the value of any kind, it is required to take into consideration not only the instruction code but also instruction argument and its result. Hence, the procedure of compiling the samples is a very time consuming process and requires a careful synthesis of each instruction, which is based not only on technical specification, but also on changes in voltage supply. Therefore, in the process of compiling the database of samples the study focuses on ten instructions of the microprocessor instruction list: ADDLW, ANDWF, BSF, CLRF, COMF, INCF, MOVF, MOVLW, NOP, and XORLW.

In previous research the authors showed that the voltage waveform during realization of the first and third machine cycle is not directly affected by instruction argument and the result of operation. It appears that the voltage waveform during realization of instruction cycle, is affected not only by the operation code, but also by the differences between the state of data bus and instruction argument (*machine cycle Q1*), and between instruction argument and the result of operation (*machine cycle Q3*). These differences can be measured by using Hamming distance parameter. In this case a schema based on Hamming distance calculated for instruction arguments, can be used to create database of samples. This information allows for a significant simplification of the construction of database used in the process of instruction recognition.

Figure 1 presents the schema of database construction and the process of recognizing instructions that operate on arguments with the value of any kind. The database consists of samples which describe 10 instructions, where samples from 1 to N describe instruction 1, next M samples describe instruction 2, etc. In this way, it was possible to create a database for the previously mentioned instructions, consisting of 1935 samples and used then in the process of recognition of microprocessor program code.

Having compiled the database, three test programs consisting of 100 instructions were generated. They were used to determine the effectiveness of method for instruction recognition. The programs were created by using the same 10 instructions, which were previously used in the process of compiling database. Both, the order of instructions in the test programs, as well as instruction arguments were selected at random.

3 The Research Results

According to research procedure, each instruction included in test program was compared to 1935 samples compiled in database.

Figure 2 presents the numbers of correctly recognized instructions in subsequent instructions proposition, which were received as a result of a method

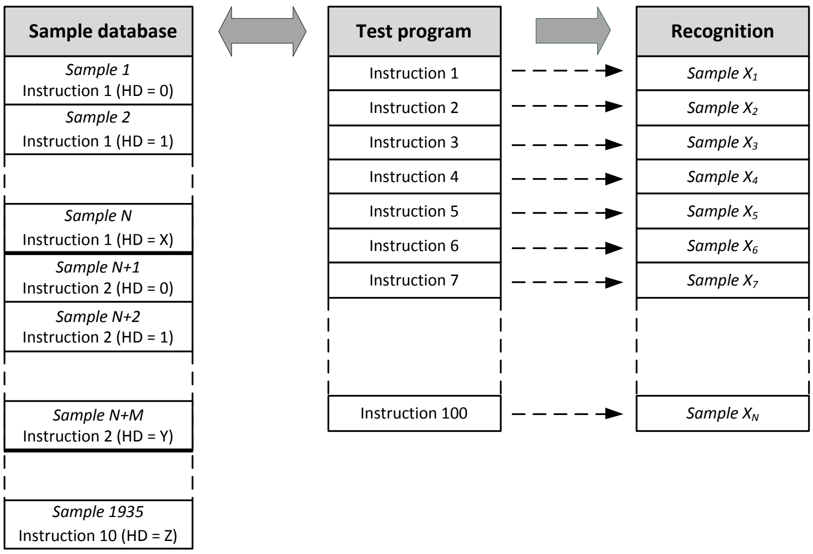

Fig. 1. The process of recognizing instructions

used for all three test programs. Further analysis of the bar chart reveals that the largest number of instructions was correctly recognized in the first three propositions. Table 1 presents the number of correctly recognized instructions in following instructions suggestions and their average values for test programs 1, 2 and 3. Based on the obtained research results it can be stated that for the first three propositions it was possible to recognize on average 33.67%, 22.00%, and 16.67% of instructions in the test programs. For the next columns the average number of correctly recognized instructions was less than 10%.

Table 2 presents the effectiveness of the method used for instruction recognition. Efficiency is the ratio of the numbers of instructions correctly recognized in one of the first three instructions propositions, to the total number of samples in test program. For the test program 1, in 72% of cases the correctly recognized instruction was among the suggestions of the first three the most similar instructions, returned as a result of the method used in the research. For the other test programs the achieved effectiveness of instruction recognition is about 74 and 71%.

Table 1. Number of correctly recognized instructions in the subsequent instructions propositions and their average values for test programs 1, 2 and 3

	Number of correctly recognized instructions						
	Instr. 1	Instr. 2	Instr. 3	Instr. 4	Instr. 5	Instr. 6	Instr. 7
Test program 1	35	22	15	9	9	10	0
Test program 2	37	20	17	9	11	6	0
Test program 3	29	24	18	11	8	9	1
Average value	**33.67**	**22.00**	**16.67**	9.67	9.34	8.34	0.34

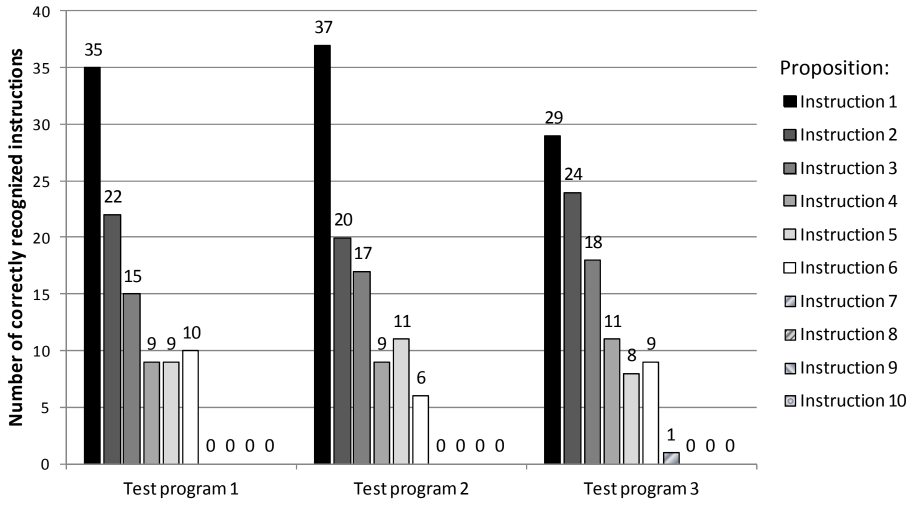

Fig. 2. Number of correctly recognized instructions in the subsequent instructions propositions, received as a result of method used for test programs 1, 2 and 3

Table 2. Comparison of the method effectiveness used for instructions recognition, that operate on arguments with the value of any kind, for test programs 1, 2 and 3

	Number of instructions in the first three instructions suggestions		The effectiveness of method used for recognizing instructions
	correctly recognized	incorrectly recognized	
Test program 1	72	28	72%
Test program 2	74	26	74%
Test program 3	71	29	71%

Decrease of effectiveness of instructions recognition, presented in this paper, compared to the effectiveness of instructions recognition that operate on arguments with the value of zero, is caused in this case by the influence of data processing on the voltage supply waveform when a particular instruction is executed. The result is that various instructions of microprocessor unit for specific values, such as: state of data bus, instruction argument and result of operation, can have the same or very similar voltage waveforms measured in microprocessor power circuit. Based on the database of samples, the research method returns propositions of instructions which were recognized correctly together with the value of similarity for each instruction in the test program. The study assumes that the effectiveness of method for recognizing microprocessor program code will be determined based on the numbers of recognized instructions in the first three instructions suggestions.

4 Conclusion

The paper concerns the problem of microprocessor systems security, and in particular the threats to the programs stored in memory of such systems, and information they process.

The authors proved the possibility of partial recognition of the instruction currently executed based on the changes of microprocessor voltage supply. Moreover, the authors indicate that it is possible to determine to a certain extent Hamming distance between the state of data bus and instruction argument, and between instruction argument and result of operation. Such possibility in this case refers to the recognition of numbers of bits changes on instruction argument, in consequence of instruction realization.

As a result of conducted research, a database consisted of 1935 samples was compiled, and then was compared to three test programs. Each test program consisted of 100 instructions and arguments which both were selected at random. It was assumed that if the instruction after having been compared to the samples database is in one of the first three places, then it was correctly recognized. With such assumptions, the effectiveness of the presented method for the following test programs is: 72%, 74% and 71%. Results of research can be generalized to other microprocessor systems executing program code stored in the memory.

Results of research can be also generalized to other microprocessor units with very similar internal architecture which execute program code stored in the memory. In case of microcontrollers with completely different datapaths, such as multiple cycle and pipeline, to fetch and execute instructions it is necessary to conduct further research.

This study and the results presented here should be considered also as an attempt to draw attention to the threats resulting from the phenomena of emanation emission, or any kind of unintended signals, which in case of being captured and analyzed reveal the information processed by the device. Moreover, several questions arise. The first is, whether obtained results have only to draw attention to the risk of decoding programs with the use of presented method? Another refers to the necessity of improving the method described above (*or develop a new one*) to get more effectiveness, and thereby indicate threats in a more precise way, and at the same time to build protection against unauthorized access to the source version (*following assembler instructions*) of a software. Finding answers to these questions is so far an open issue.

References

1. Bao, F., Deng, R.H., Han, Y., Jeng, A., Narasimhalu, A.D., Ngair, T.: Breaking Public Key Cryptosystems on Tamper Resistant Devices in the Presence of Transient Faults. In: Christianson, B., Lomas, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 115–124. Springer, Heidelberg (1998)
2. 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)

3. Maćkowski, M.: The Influence of Electromagnetic Disturbances on Data Transmission in USB Standard. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009. CCIS, vol. 39, pp. 95–102. Springer, Heidelberg (2009)
4. Mangrad, S., Oswald, E., Popp, T.: Power Analysis Attacks – Revaling the Secrets of Smart Cards. Springer (2007)
5. 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)
6. 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)
7. Maćkowski, M., Skoroniak, K.: Instruction Prediction in Microprocessor Unit Based on Power Supply Line. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 173–182. Springer, Heidelberg (2010)
8. Maćkowski, M., Skoroniak, K.: Electromagnetic Emission Measurement of Microprocessor Units. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009. CCIS, vol. 39, pp. 103–110. Springer, Heidelberg (2009)