BESTELLUNG Semesterapparat DigiSem

MTP-000000268 **Bestelldatum: 2014-11-07 15:10:40**

Benutzernummer	
Name	Tempelmeier, Michael
Straße	Technische Universität München
Postleitzahl	80797
Ort/Stadt	München
E-Mail-Adresse	michael.tempelmeier@tum.de

Unter Anerkennung des Urheberrechtsgesetzes wird bestellt:

ISBN	0-387-30857-1
Haupt-Titel	Power analysis attacks : revealing the secrets of smart cards : Advances in inforn
Autor	Mangard, StefanOswald
Titel	Differential Power Analysis
Band/Heft	()
Jahrgang	2007
Seiten	119-129

Signatur 0102/DAT 179f 2010 A 4848

Vermerk der Bibliothek

- O Jahrgang nicht vorhanden
- O verliehen
- O nicht am Standort
- O beim Buchbinder
- O vermißt
- O Sonstiges

Chapter 6

DIFFERENTIAL POWER ANALYSIS

Differential power analysis (DPA) attacks are the most popular type of power analysis attacks. This is due to the fact that DPA attacks do not require detailed knowledge about the attacked device. Furthermore, they can reveal the secret key of a device even if the recorded power traces are extremely noisy.

In contrast to SPA attacks, DPA attacks require a large number of power traces. It is therefore usually necessary to physically possess a cryptographic device for some time in order to mount a DPA attack on it. Consider for example an owner of an electronic purse. This person can record a large number of power traces by transferring small amounts of money to and from the purse. These traces could then be used to reveal the cryptographic key that is used by the purse.

In this chapter, we provide a comprehensive introduction to DPA attacks. We discuss and compare different kinds of DPA attacks and we also illustrate them based on several examples. For this purpose, we use the software and the hardware implementation of AES that are described in Appendix B. We also elaborate on issues like the simulation of DPA attacks and the calculation of the number of traces that are needed to perform successful DPA attacks.

6.1 General Description

The goal of DPA attacks is to reveal secret keys of cryptographic devices based on a large number of power traces that have been recorded while the devices encrypt or decrypt different data blocks. The main advantage of DPA attacks compared to SPA attacks is that no detailed knowledge about the cryptographic device is necessary. In fact, it is usually sufficient to know the cryptographic algorithm that is executed by the device.

Another important difference between the two kinds of attacks is that the recorded traces are analyzed in a different way. In SPA attacks, the power consumption of a device is mainly analyzed along the time axis. The attacker tries to find patterns or tries to match templates in a single trace. In case of DPA attacks, the shape of the traces along the time axis is not so important. DPA attacks analyze how the power consumption at fixed moments of time depends on the processed data. Hence, DPA attacks focus exclusively on the data dependency of the power traces.

> DPA attacks exploit the data dependency of the power consumption of cryptographic devices. They use a large number of power traces to analyze the power consumption at a fixed moment of time as a function of the processed data.

We now discuss in detail how such an analysis reveals secret keys of cryptographic devices. In contrast to SPA attacks, there exists a general attack strategy that is used by all DPA attacks. This strategy consists of five steps.

Step 1: Choosing an Intermediate Result of the Executed Algorithm. The first step of a DPA attack is to choose an intermediate result of the cryptographic algorithm that is executed by the attacked device. This intermediate result needs to be a function $f(d, k)$, where d is a known non-constant data value and k is a small part of the key. Intermediate results that fulfill this condition can be used to reveal k. In most attack scenarios, d is either the plaintext or the ciphertext.

Step 2: Measuring the Power Consumption. The second step of a DPA attack is to measure the power consumption of the cryptographic device while it encrypts or decrypts D different data blocks. For each of these encryption or decryption runs, the attacker needs to know the corresponding data value d that is involved in the calculation of the intermediate result chosen in step 1. We write these known data values as vector $\mathbf{d} = (d_1, \ldots, d_D)'$, where d_i denotes the data value in the i^{th} encryption or decryption run.

During each of these runs the attacker records a power trace. We refer to the power trace that corresponds to data block d_i as $\mathbf{t}'_i = (t_{i,1}, \ldots, t_{i,T})$, where T denotes the length of the trace. The attacker measures a trace for each of the D data blocks, and hence, the traces can be written as matrix $\mathbf{T}$ of size $D \times T$. It is important for DPA attacks that the measured traces are correctly aligned. This means that the power consumption values of each column $\mathbf{t}_j$ of the matrix $\mathbf{T}$ need to be caused by the same operation. In order to obtain aligned power traces, the trigger signal for the oscilloscope needs to be generated in such a way that the oscilloscope records the power consumption of exactly the same sequence of operations during each encryption or decryption run. In case such a trigger signal is not available, the power traces need to be aligned using the techniques described in Section 8.2.2.

Step 3: Calculating Hypothetical Intermediate Values. The next step of the attack is to calculate a *hypothetical intermediate value* for every possible choice of k. We write these possible choices as vector $\mathbf{k} = (k_1, \ldots, k_K)$, where K denotes the total number of possible choices for k. In the context of DPA attacks, we usually refer to the elements of this vector as key hypotheses. Given the data vector $\mathbf{d}$ and the key hypotheses $\mathbf{k}$, an attacker can easily calculate hypothetical intermediate values $f(d, k)$ for all D encryption runs and for all K key hypotheses. This calculation (6.1) results in a matrix $\mathbf{V}$ of size $D \times K$. The first part of Figure 6.1 illustrates this calculation step.

$$v_{i,j} = f(d_i, k_j) \quad i = 1, \ldots, D \quad j = 1, \ldots, K \tag{6.1}$$

Column j of $\mathbf{V}$ contains the intermediate results that have been calculated based on the key hypothesis k_j. It is clear that one column of $\mathbf{V}$ contains those intermediate values that have been calculated in the device during the D encryption or decryption runs. Remember, $\mathbf{k}$ contains all possible choices for k. Hence, the value that is used in the device is an element of $\mathbf{k}$. We refer to the index of this element as ck. Hence, k_{ck} refers to the key of the device. The goal of DPA attacks is to find out which column of $\mathbf{V}$ has been processed during the D encryption or decryption runs. As soon as we know which column of $\mathbf{V}$ has been processed in the attacked device, we immediately also know k_{ck}.

Step 4: Mapping Intermediate Values to Power Consumption Values. The next step of a DPA attack is to map the hypothetical intermediate values $\mathbf{V}$ to a matrix $\mathbf{H}$ of *hypothetical power consumption values*, see Figure 6.1. For this purpose, the attacker uses the simulation techniques we have discussed in Section 3.3. Using one of these techniques, the power consumption of the device for each hypothetical intermediate value $v_{i,j}$ is simulated in order to obtain a hypothetical power consumption value $h_{i,j}$.

The quality of the simulation strongly depends on the knowledge of the attacker about the analyzed device. The better the simulation of the attacker matches the actual power consumption characteristics of the device, the more effective is the DPA attack. The most commonly used power models to map $\mathbf{V}$ to $\mathbf{H}$ are the Hamming-distance and the Hamming-weight model. However, as pointed out in Section 3.3, there are also many other ways to map data values to power consumption values.

Step 5: Comparing the Hypothetical Power Consumption Values with the Power Traces. After having mapped $\mathbf{V}$ to $\mathbf{H}$, the final step of a DPA attack can be performed. In this step, each column $\mathbf{h}_i$ of the matrix $\mathbf{H}$ is compared with each column $\mathbf{t}_j$ of the matrix $\mathbf{T}$. This means that the attacker compares the hypothetical power consumption values of each key hypothesis with the recorded traces at every position. The result of this comparison is a matrix $\mathbf{R}$

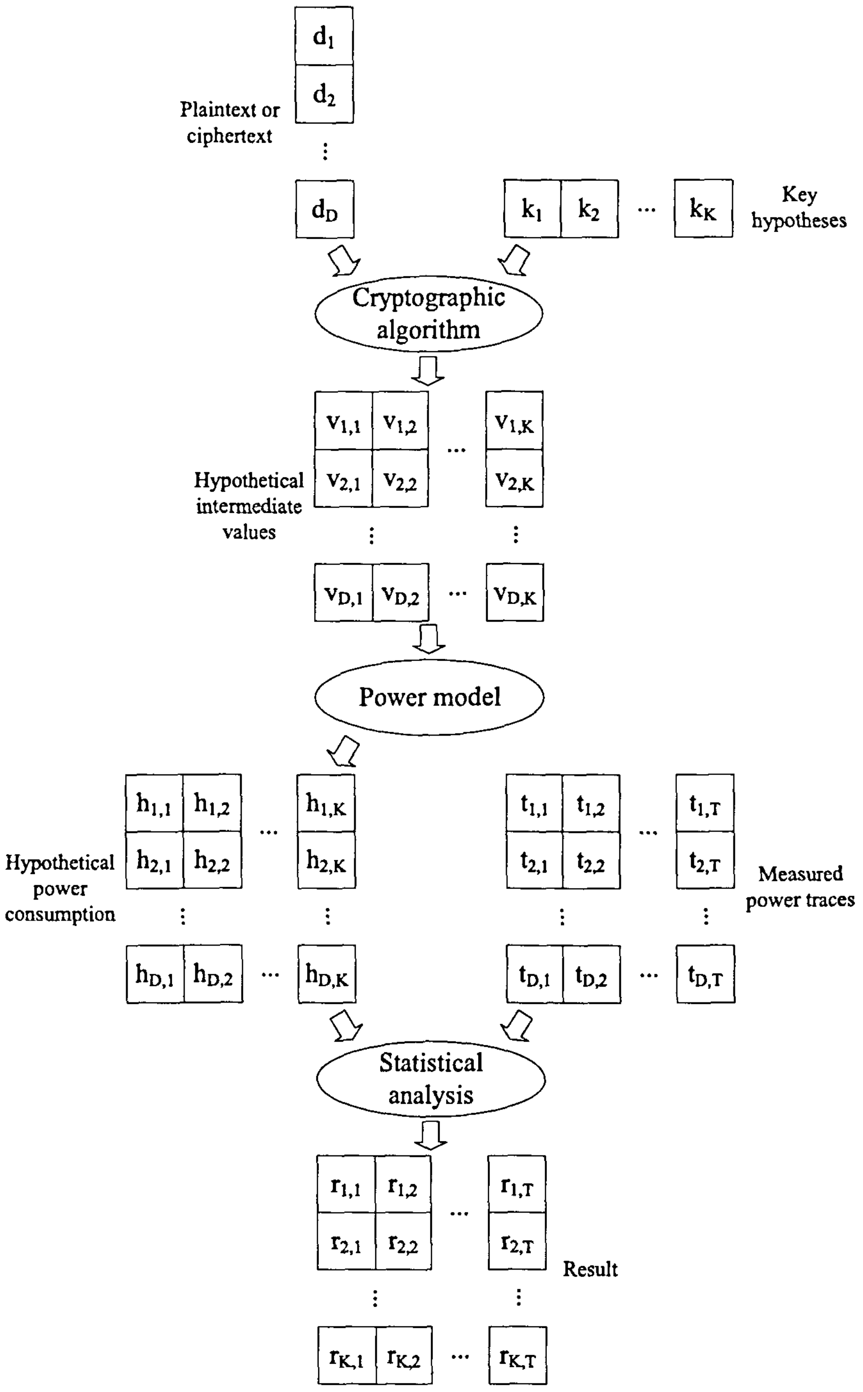

Figure 6.1. Block diagram illustrating the steps 3 to 5 of a DPA attack.

of size $K \times T$, where each element $r_{i,j}$ contains the result of the comparison between the columns $\mathbf{h}_i$ and $\mathbf{t}_j$. The comparison is done based on algorithms we discuss later in this chapter. All algorithms have the property that the value $r_{i,j}$ is the higher, the better the columns $\mathbf{h}_i$ and $\mathbf{t}_j$ match. The key of the attacked device can hence be revealed based on the following observation.

The power traces correspond to the power consumption of the device while it executes a cryptographic algorithm using different data inputs. The intermediate result that has been chosen in step 1 is a part of this algorithm. Hence, the device needs to calculate the intermediate values $\mathbf{v}_{ck}$ during the different executions of the algorithm. Consequently, also the recorded traces depend on these intermediate values at some position. We refer to this position of the power traces as *ct*, *i.e.* the column $\mathbf{t}_{ct}$ contains the power consumption values that depend on the intermediate values $\mathbf{v}_{ck}$.

The hypothetical power consumption values $\mathbf{h}_{ck}$ have been simulated by the attacker based on the values $\mathbf{v}_{ck}$. Therefore, the columns $\mathbf{h}_{ck}$ and $\mathbf{t}_{ct}$ are strongly related. In fact, these two columns lead to the highest value in **R**, *i.e.* the highest value of the matrix **R** is the value $r_{ck,ct}$. All other values of **R** are low because the other columns of **H** and **T** are not strongly related. An attacker can hence reveal the index for the correct key *ck* and the moment of time *ct* by simply looking for the highest value in the matrix **R**. The indices of this value are then the result of the DPA attack.

The indices of the highest values of the matrix **R** reveal the positions at which the chosen intermediate result has been processed and the key that is used by the device.

It is important to point out though that it can also happen in practice that all values of **R** are approximately the same. In this case, the attacker has usually not measured enough power traces to estimate the relationship between the columns of **H** and **T**. The more traces an attacker measures, the more elements are in the columns of **H** and **T**, and the more precisely the attacker can determine the relationship between the columns. This also implies that the more measurements are made, the smaller relationships between the columns can be determined.

6.2 Attacks Based on the Correlation Coefficient

The correlation coefficient is the most common way to determine linear relationships between data. Therefore, it is also an excellent choice when it comes to performing DPA attacks. There exists a well-established theory for the correlation coefficient that can be used to model statistical properties of DPA attacks. Furthermore, this theory also makes comparisons of different attacks quite easy.

We have already discussed the basics of the correlation coefficient in Chapter 4. Section 4.4.1 has provided a definition of the correlation coefficient (4.14) as well as a formula to estimate its value (4.15). The sampling distribution of the estimator has then been discussed in Section 4.6.5.

In DPA attacks, the correlation coefficient is used to determine the linear relationship between the columns $\mathbf{h}_i$ and $\mathbf{t}_j$ for $i = 1, \ldots, K$ and $j = 1, \ldots, T$. This results in a matrix $\mathbf{R}$ of estimated correlation coefficients. We estimate each value $r_{i,j}$ based on the D elements of the columns $\mathbf{h}_i$ and $\mathbf{t}_j$. Using the notation of the previous section, we can therefore rewrite (4.15) as (6.2). In (6.2), the values $\bar{h}_i$ and $\bar{t}_j$ denote the mean values of the columns $\mathbf{h}_i$ and $\mathbf{t}_j$.

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \bar{h}_i) \cdot (t_{d,j} - \bar{t}_j)}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \bar{h}_i)^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \bar{t}_j)^2}} \tag{6.2}$$

In order to illustrate how DPA attacks based on the correlation coefficient work, we now discuss several examples of such attacks. We first show attacks on a software implementation of AES, and subsequently we look at a hardware implementation.

6.2.1 Examples for Software

The target of the first DPA attack is the AES software implementation that is described in Appendix B.2. We have executed this implementation on our microcontroller. The power consumption of the microcontroller has been measured using the setup presented in Section 3.4.4. In this setup, the microcontroller receives plaintexts via an RS-232 interface, encrypts them, and returns the corresponding ciphertexts. As an attacker of the device, we therefore have access to the plaintext and the ciphertext. Consequently, we are quite flexible in step 1 of the DPA attack, where we need to choose an intermediate result of AES. We can choose any intermediate result that is a function of the plaintext or the ciphertext and a few key bits.

In the concrete attack, we have decided to choose the output byte of the first AES S-box in round one. This intermediate result is a function of the first byte of plaintext and the first byte of the secret key. After having chosen this intermediate result, we have recorded the power consumption of the microcontroller during the first round of AES while it has encrypted 1 000 different plaintexts. This second step of the DPA attack has led to a matrix $\mathbf{T}$ of power consumption values. The third step has then been to calculate hypothetical intermediate values based on the 1 000 known plaintexts. This means that we have calculated the values $v_{i,j} = S(d_i \oplus k_j)$, where $d_1, \ldots, d_{1\,000}$ are the first byte of each of the 1 000 plaintexts and $k_j = j - 1$ with $j = 1, \ldots, 256$. The matrix $\mathbf{V}$ has hence a size of $1\,000 \times 256$ values.

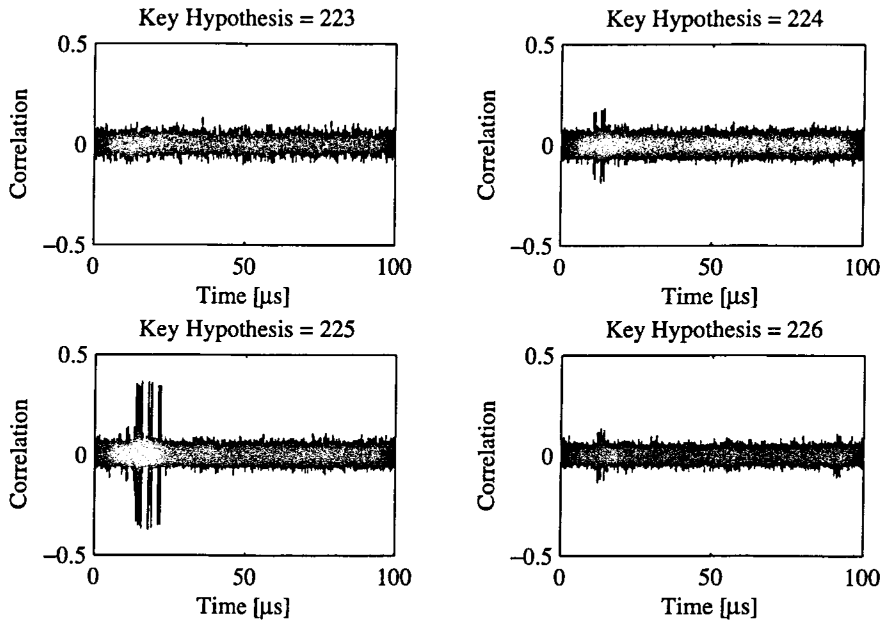

Figure 6.2. The rows of the matrix **R** that correspond to the key hypotheses 223, 224, 225, and 226.

The fourth step of the DPA attack has been to map **V** to a matrix **H** of hypothetical power consumption values. In the current attack, we have decided to use a very simple power model for this mapping. We have just considered the LSB of the values in **V**. Hence, we have used $h_{i,j} = LSB(v_{i,j})$ as our power model. Based on **H**, we have then performed the last step of the DPA attack. We have calculated the correlation coefficients between all columns of **H** and all columns of the recorded power consumption values **T**. The result of this calculation is a matrix **R** of correlation coefficients.

In practice, there exist several different ways to visualize this matrix **R**. One of these ways is to show each row of the matrix in a separate plot. In this case, each plot corresponds to one key hypothesis. Figure 6.2, for example, shows the plots for the key hypotheses 223 to 226 of the current attack. It can be observed that there are very high peaks in the plot for key hypothesis 225. In fact, these peaks are the highest ones of the entire matrix **R**. All other values of **R** are significantly smaller. The fact that these high peaks occur for key hypothesis 225 and the fact that the first peak occurs at 13.8 μs provides a lot of information to an attacker.

First of all, the peaks indicate that the first byte of the secret key of the microcontroller is 225. Second, the microcontroller computes the output of the first AES S-box at position 13.8 μs of the recorded traces. Furthermore, we can

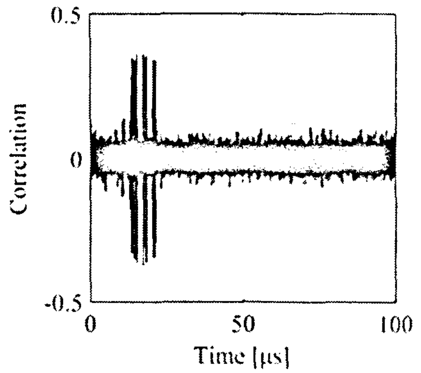

Figure 6.3. All rows of **R**. Key hypothesis 225 is plotted in black, while all other key hypotheses are plotted in gray.

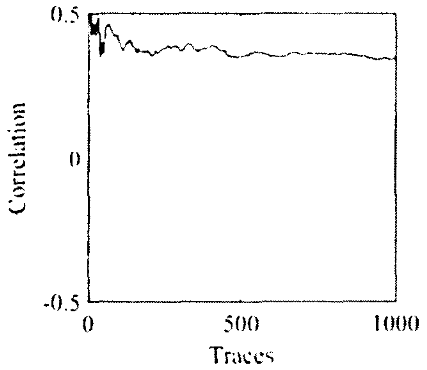

Figure 6.4. The column of **R** at 13.8 μs for different numbers of traces. Key hypotheses 225 is plotted in black.

also learn from the number of the peaks in the plot that this intermediate result is used in several instructions. This is very typical for software implementations. After the attacked intermediate result has been calculated, it is usually moved from a register to memory and then later on loaded back into a register as operand for subsequent operations of the algorithm. Each time the microcontroller performs an operation that involves the attacked intermediate result, there occurs at least one peak in **R**. This is why there are so many peaks in the plot for key 225.

The peaks for key 225 are the most significant ones. However, when looking closely at the other plots of Figure 6.2, we can also observe some smaller peaks in these plots. These peaks occur because not all columns of **H** are independent. This means whenever a column of **H** leads to a high correlation coefficient, also some other columns lead to some correlation. However, this correlation is typically significantly smaller, and hence, the key of the device can be identified easily. In fact, the plots for all keys, except for key 225, look very similar in the current attack. This can also be observed in Figure 6.3. In this figure, the plots for all keys are shown. Key 225 is plotted in black, while all other keys are plotted in gray. There are no significant peaks in gray—only the plot for key 225 contains high peaks.

An important question when performing DPA attacks is how many traces are needed in order to obtain such high peaks in the matrix **R**. Figure 6.4 provides a preliminary answer to this question. This figure shows how the first peak in the plot for key 225 (*i.e.* the peak located at 13.8 μs in Figure 6.3) evolves as a function of the used number of traces. Key 225 is again plotted in black, while the correlation of all other key hypotheses at position 13.8 μs is plotted in gray.

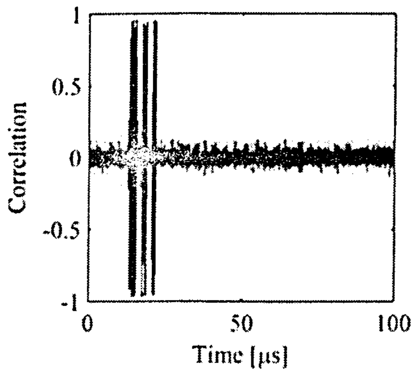

Figure 6.5. All rows of **R**. Key hypothesis 225 is plotted in black, while all other key hypotheses are plotted in gray.

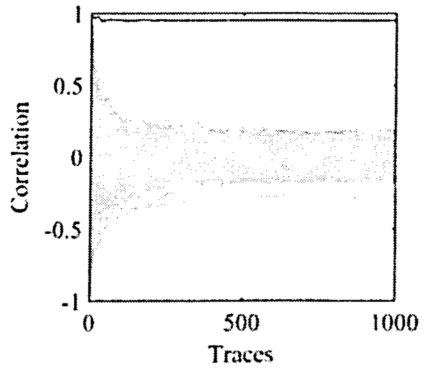

Figure 6.6. The column of **R** at 13.8 μs for different numbers of traces. Key hypothesis 225 is plotted in black.

Figure 6.4 nicely illustrates how the estimated correlation coefficients $r_{k,13.8}$ converge towards $\rho_{k,13.8}$ for $k = 1, \ldots, K$. The more traces are used, the better is the estimation of the correlation, see Section 4.6.5. The value $r_{225,13.8}$ converges to about 0.35, while all other correlation coefficients converge to values below 0.2. Key 225 leads to the highest correlation coefficient if about 160 or more traces are used. Hence, as a first estimate we can say that about 160 traces are required for a successful attack. A more precise number will be derived later in Section 6.4.

DPA Attack Using the Hamming-Weight Model

In the previous DPA attack on the microcontroller we have used the bit model $h_{i,j} = LSB(v_{i,j})$ to estimate the power consumption. However, from Chapter 4 we already know that the power consumption of the microcontroller is inversely proportional to the Hamming weight of the processed data. Hence, we can use this knowledge to improve the DPA attack on the S-box output. We now show results of a DPA attack that uses the same traces as before. However, in step 4 we have used the Hamming-weight model instead of the bit model, *i.e.* we have set $h_{i,j} = HW(v_{i,j})$.

Figure 6.5 shows the result of this attack. The plot for key 225 again contains the highest peaks. However, in comparison to the result before, the peaks are much higher. In the attack based on the Hamming-weight model, the value $r_{225,13.8}$ converges to about 0.95 instead of 0.35. Therefore, this peak can be detected with a much lower number of traces. Figure 6.6 shows that already after 20 traces, the value $r_{225,13.8}$ is bigger than the other correlation coefficients at 13.8 μs. Hence, DPA attacks on the microcontroller require significantly less

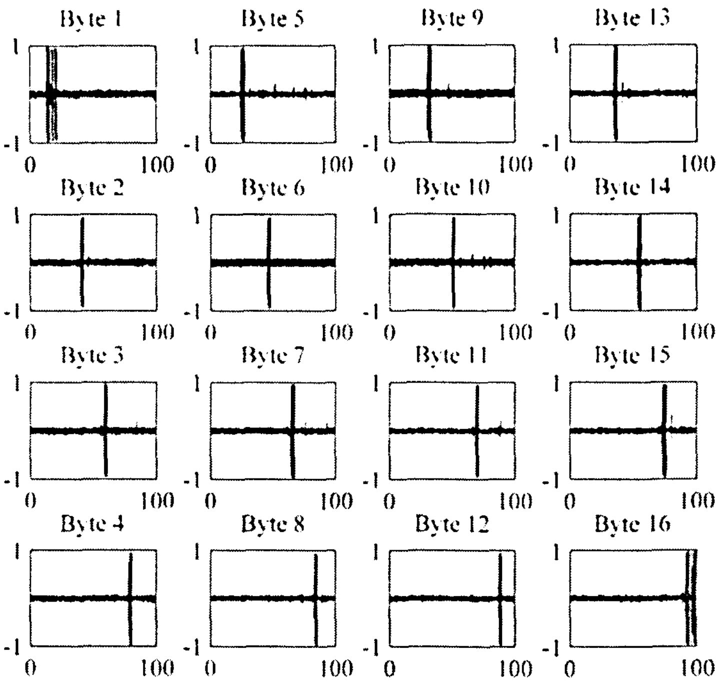

Figure 6.7. Plots for the correct key hypotheses in the DPA attacks on the 16 S-box outputs in the first round. The y-axes show the correlation and the x-axes show the time in μs.

traces, if the Hamming-weight model is used. This is because the Hamming-weight model describes the power consumption of the microcontroller much better than the bit model.

The better the used power model describes the attacked device, the less traces are needed in DPA attacks.

The DPA attacks that we have presented so far have revealed the first byte of the secret key that is used by the microcontroller. We now also show results of DPA attacks on the remaining key bytes. In order to reveal all these key bytes, we have performed the same attack as described before on all 16 key bytes. This means that we have performed 16 DPA attacks in total. Each of these DPA attacks has targeted a different S-box output in the first round of AES based on the Hamming-weight model. For all attacks, the same power traces have been used.

Figure 6.7 shows the results of the attacks. Each of the 16 plots in this figure corresponds to the plot for the correct key hypothesis in the corresponding attack. Hence, the first plot is the same as in Figure 6.5. Based on the 16 plots in Figure 6.7, we can make some important observations. First, all 16 DPA attacks have been successful and lead to similar correlation values for the correct key hypothesis. This is due to the fact that the same instruction is used for all S-box look-ups. Therefore, also the same leakage occurs for all S-box outputs. The second important observation is that the S-box look-ups are performed sequentially. The software implementation performs one S-box look-up after the other, and hence, also one key byte after the other is used. Based on the traces shown in Figure 6.7 it is easy to see that the microcontroller performs the S-box look-ups in the following sequence: $1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15, 4, 8, 12, 16$. The microcontroller hence processes one row of the AES state after the other, see Appendix B. Note that in the Appendix the bytes of the state are counted from 0 to 15.

A final interesting observation is that for some key bytes more peaks occur than for others. In particular, there are more peaks in the plots for the bytes 1 and 16. This is due to the fact that the software implementation performs some additional operations with the corresponding S-box outputs. Every peak in the plots has its source and an attacker can learn many things about the device by analyzing these peaks. In fact, it is often possible to perform a kind of reverse engineering of an implementation based on the observed peaks.

6.2.2 Examples for Hardware

The DPA attacks on the software implementation that we have presented in the previous section have been very effective. Using the Hamming-weight model, we have been able to reveal the entire key of the AES implementation with about 20 traces. We now investigate DPA attacks on a more challenging target. The target of the attacks in the current section is the AES ASIC that is described in Appendix B.3. We have performed several DPA attacks on this chip using the measurement setup described in Section 3.4.4.

In this setup, we have access to the plaintexts and the ciphertexts. Hence, we have again been quite flexible when choosing a suitable intermediate result for our DPA attacks. This time, we have decided to choose the inputs of the AES S-boxes in the last round. Each input byte of these S-boxes depends on one byte of the last round key of AES. Hence, 16 DPA attacks are necessary to reveal the entire key. Just like in the previous attacks, we have started with the first byte.

After having selected an intermediate result for our attack, we have recorded the power consumption of our AES ASIC while it has encrypted 100 000 random plaintexts. Based on the 100 000 resulting ciphertexts, we have then calculated the hypothetical intermediate values $v_{i,j} = S^{-1}(d_i \oplus k_j)$. The size of the resulting matrix **V** is $100\,000 \times 256$. The next step of the DPA attack has been to map **V** to a matrix **H** of hypothetical power consumption values. In case of the attack on the hardware implementation, this step is much more critical than