# Using the ANOVA F-Statistic to Isolate Information-Revealing Near-Field Measurement Configurations for Embedded Systems

Vishnuvardhan V. Iyer
*Department of Electrical and Computer Engineering*
*The University of Texas at Austin*
Austin, TX, USA
vishnuv.iyer@utexas.edu

Ali E. Yilmaz
*Department of Electrical and Computer Engineering*
*The University of Texas at Austin*
Austin, TX, USA
ayilmaz@mail.utexas.edu

***Abstract*—The analysis of variance (ANOVA) F-statistic is proposed as a tool to isolate near-field measurement configurations that are sensitive to targeted chip processes in embedded systems. It is hypothesized that the desired measurement configurations have high F-values, i.e., the variation in a target process is a major contributor whereas obfuscating background processes and measurement uncertainty are minor contributors to the variance of measured signals. The concept is demonstrated by isolating data-dependent measurement configurations for a commercially available variant of the 8051 microcontroller: First, a multi-stage measurement protocol using F-values is developed to rapidly isolate optimal measurement configurations within the 4-D search space of 2-D probe location over chip area, probe orientation, and time. Then, signals captured using configurations with high F-values are analyzed to identify the Hamming weights of the output data computed by a randomized test code running on the 8051. It is shown that configurations with higher F-values generally result in more accurate classification of the output data; the configuration with the highest F-value results in 100% accuracy.**

***Keywords*—electromagnetic measurements, analysis of variance, side-channel attacks, measurement uncertainty, electromagnetic interference**

## I. Introduction

Fields unintentionally emanated by integrated circuits offer a viable path for recovering critical information [1]-[7]; e.g., EM side-channel analysis (SCA) attacks extract information about a target chip-process by statistically interrogating the fields emanated by exploitable on- and off-chip sources. Especially potent are EM SCA attacks that use near-field scanning systems (Fig. 1) to collect numerous signals from a device under test (DUT). Such fine-grained EM SCA attacks were used recently to recover secret keys from cryptographic chips during encryption operations [5], [6] and to localize instruction-dependent sources during program execution on processors [7].

Fine-grained EM SCA attacks can significantly reduce the marginal cost of future attacks on an implementation by first isolating optimal measurement configurations for information recovery. Finding effective information-revealing measurement configurations involves, among other factors, evaluating the performance of various probe types, bandwidths, sizes, locations, and orientations; this can require an extensive, potentially infeasible number of measurements. The number of measurements also depends on the complexity of the DUT and the information and noise content in captured signals—sum of fields emanated by exploitable and other sources.

The attempt to extract information about a target chip-process via fine-grained EM SCA attacks is confounded by various uncertainties in near-field measurements, e.g., due to equipment sensitivity, environment, drift, etc. [4], [8] and by concurrent processes in the system. Obfuscation of the signals of interest due to both measurement noise and fields emanated by extraneous/background processes (henceforth referred to as "algorithmic noise") can be quantified by statistically characterizing the measurements. In particular, the analysis of variance (ANOVA) of near-field measurements is a promising approach for such characterizations; e.g., the ANOVA F-statistic (referred to as signal-to-noise ratio in the context of EM side-channel security) was shown to successfully isolate optimal configurations for recovering one of the key-bytes used for encryption from a 128-bit implementation of the advanced encryption standard (AES), while fields emanated from unrelated processes involving other key-bytes obfuscate the signals [4], [5].

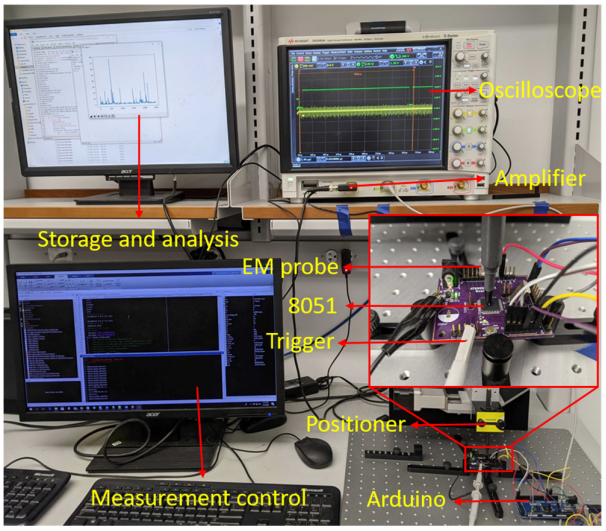



Fig. 1. The near-field measurement setup used for EM SCA attacks. Experiments were performed with an 8051 microcontroller, a general-purpose embedded system. Near-fields were sensed using an H-field probe, scanning the chip at a height of 0.2 mm.

This article presents a methodology that characterizes the obfuscating factors in near-field measurements of embedded systems via ANOVA F-tests and isolates effective measurement configurations. The methodology assumes that configurations most sensitive to a target process will be least affected by measurement and algorithmic noise. First, obfuscation due to measurement noise is quantified and configurations that have both relatively small measurement noise and high sensitivity to changes in the target process are identified via F-tests. Next, the configurations least impacted by measurement noise are used to quantify the obfuscation due to algorithmic noise. Multiple stages of F-tests are required to quantify the impact of algorithmic noise, where ineffective configurations identified in one stage are discarded

from analysis in the subsequent stage. At the end of the protocol, only configurations most sensitive to the target process remain, which are further tested for potential information recovery. This approach reduces the cost of isolating optimal near-field measurement configurations by subsuming process variations within smaller groups based on information-leakage models [5]-[6].

The proposed approach is demonstrated on an 8051 micro-controller unit (MCU) with a 2-stage pipeline and shared bus architecture [12]. A 3-stage measurement protocol is proposed to isolate the data-dependent probe locations on the chip, probe orientations, and observation time windows. At each stage, ineffective measurement configurations are discarded via null hypothesis testing for F-values. The protocol aims to isolate configurations that can effectively extract the output for each instruction execution (the target process); changes in remaining architectural components (instructions, memory locations, program counter, etc.) are categorized as algorithmic noise. To verify that these isolated configurations actually exhibit high sensitivity to the target process, signals measured with these configurations are used to recover the Hamming weights (HW) of binary output data—the number of bits with value 1 in the 8-bit output generated as a result of executing an instruction—for a test code running on the MCU.

## II. Statistical Analyses of Near-Field Measurements

### A. *Dependence of Signals on Chip-Processes*

Consider a signal $V_{pr,r}^{pc,t}$ measured by a near-field probe located above a chip. The measured signal depends on the probe configuration $pc$; in this article, this corresponds to 4 independent variables: the probe's 2-D location $(x, y)$ on the chip, height above the chip surface $h$, and orientation $o$. It also depends on the observation time $t$ and all the processes $pr$ performed on the chip at that time instant. The signal is also designated as a function of measurement repetition index $r$ to account for measurement-to-measurement variations. In the following, it is assumed that the probe is located at one of the $N_\mathrm{x} \times N_\mathrm{y}$ regular grid points on a chip of size $l_\mathrm{x} \times l_\mathrm{y}$, i.e., at points

$$\left(x_{n_\mathrm{x}}, y_{n_\mathrm{y}}\right) = \left(\frac{n_\mathrm{x} l_\mathrm{x}}{N_\mathrm{x}-1}, \frac{n_\mathrm{y} l_\mathrm{y}}{N_\mathrm{y}-1}\right) \tag{1}$$

where $n_\mathrm{x} = 0,1,\cdots,N_\mathrm{x}-1$ and $n_\mathrm{y} = 0,1,\cdots,N_\mathrm{y}-1$, and $N_\mathrm{t}$ samples are recorded in each clock cycle with a sampling rate of $1/\Delta t$.

The measured signal's dependency on chip-processes $pr$ stems from data-dependent switching in CMOS logic [5],[6]. Signals emanated by the 8051 MCU can depend on the instructions in its execution pipeline, the memory locations they are referencing, program counter changes, and data transfers within each execution. Consider two instructions that move data to register A: "MOV A, #00h" and "MOV A, R0". If the register R0 is loaded with #00h (hexadecimal notation for the MCU), then executing either instruction will move the same value (00h) to register A. Even when all other chip processes (e.g., pipelined instructions) are identical, because they access different memory locations, the two executions will emanate different fields observable near the chip (Fig. 2).

Let chip-processes $pr$ be expressed as a combination of a "target process" $\mathrm{Tpr}_i$ and one or more irrelevant "background processes" $\mathrm{Bpr}_j^k$, where the subscript indexes $i$ and $j$ represent versions within each process and the

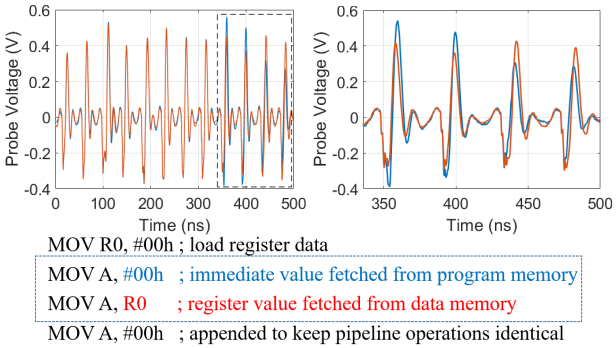



Fig. 2. Instruction dependence of observed signals as a MOV instruction with the same operand value 0x00 was executed. The data was fetched from either the program memory (blue) or data memory (red) as shown in the snippet. The signals were observed by an $x$-oriented H-field probe at the centre ($x = 5\ mm, y = 5\ mm$) of an 8051 chip operated at a clock frequency of 2 MHz.

superscript index $k$ represents different types of background processes. For example, if output data is designated the target process (see Section III.A), $\mathrm{Tpr}_1$ to $\mathrm{Tpr}_{256}$ will represent the data values from 00h to FFh in the 8051 MCU. In this case, increment of the program counter is a type of background process, varying from $\mathrm{Bpr}_1^1$ to $\mathrm{Bpr}_{4096}^1$ for each byte of code fetched from memory, for a maximum code size of 4 KB. Other background processes that have dependencies among them are represented as different versions of one type of process. In particular, because all instruction opcodes are decoded to determine each instruction and memory locations of all of its operands, a single type of background process $k$ represents all of them and a specific instruction and its memory locations are represented by the subscript; e.g., $\mathrm{Bpr}_1^2$ can represent MOV using program memory, $\mathrm{Bpr}_2^2$ ADD using program memory, and $\mathrm{Bpr}_3^2$ ADD using data memory.

To analyze the signals, let the array $\mathbf{V}_r^{pc,t}$ list all the measured signals corresponding to all possible combinations of processes. Each observed signal in the array, $V_{\mathrm{Tpr}_i,\mathrm{Bpr}_j^k,r}^{pc,t}$, can be decomposed into three independent, abstract signals $T_{\mathrm{Tpr}_i}^{pc,t}$, $N_r^{pc,t}$, and $B_{\mathrm{Bpr}_j^k}^{k,pc,t}$. Here, $T_{\mathrm{Tpr}_i}^{pc,t}$ and $B_{\mathrm{Bpr}_j^k}^{k,pc,t}$ represent the contribution of the target and background processes $\mathrm{Tpr}_i$ and $\mathrm{Bpr}_j^k$ to the observed signal, whereas $N_r^{pc,t}$ represents the effect of measurement-to-measurement variations. In information-revealing measurement configurations, the observed signal will depend strongly on $T_{\mathrm{Tpr}_i}^{pc,t}$ and will be insensitive to $N_r^{pc,t}$ and $B_{\mathrm{Bpr}_j^k}^{pc,t}$. If the quantities $T_{\mathrm{Tpr}_i}^{pc,t}$, $N_r^{pc,t}$, and $B_{\mathrm{Bpr}_j^k}^{pc,t}$ are listed in the arrays $\mathbf{T}^{pc,t}$, $\mathbf{N}^{pc,t}$, and $\mathbf{B}^{k,pc,t}$, their variances are related as

$$\frac{\mathrm{Var}\left(\mathbf{V}_r^{pc,t}\right)}{\mathrm{Var}(\mathbf{T}^{pc,t})} = 1 + \underbrace{\frac{\mathrm{Var}(\mathbf{N}^{pc,t})}{\mathrm{Var}(\mathbf{T}^{pc,t})}}_{1/F_N^{pc,t}} + \underbrace{\frac{\mathrm{Var}(\mathbf{B}^{k,pc,t})}{\mathrm{Var}(\mathbf{T}^{pc,t})}}_{1/F_B^{k,pc,t}} \tag{2}$$

While the abstract signals in $\mathbf{T}^{pc,t}$, $\mathbf{N}^{pc,t}$, and $\mathbf{B}^{k,pc,t}$ cannot be measured separately, the ratio of their variances can be computed by using ANOVA F-statistics on observed signals. Specifically, F-values for one-way ANOVA are computed by separating datasets into multiple groups, each group dependent on one version of a test parameter, while quantities within each group depend on several variations in other parameters [4], [11]. High F-values indicate that

variance between groups is significantly larger than the variance within the groups, implying that the given dataset is highly sensitive to the test parameter. Here, the F-values are computed by separating observed signals $V^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_j,r}$ into groups for each version $i$ in $\mathrm{Tpr}_i$, with each group consisting of variations in repeated measurements, and $\mathrm{Bpr}^k_j$ within them. For signals in $\mathbf{V}^{pc,t}_r$ to have minimal contributions from measurement uncertainty and background processes, the sum of terms $1/F^{pc,t}_N$ and $1/F^{k,pc,t}_B$ in (2) must be minimized across probe configurations and time instances (operations randomized in space [11], where the effective configurations vary for the same operation, are not considered in this work).

### B. Computation of F-statistics

The $F^{pc,t}_N$ value quantifies the variation in a target process with respect to measurement uncertainty and does not depend on background processes. To compute it, various programs are run on the processor such that the target process varies as $\mathrm{Tpr}_1, \mathrm{Tpr}_2, \cdots, \mathrm{Tpr}_{N_{\mathrm{Tpr}}}$, while the background processes are kept constant as $\mathrm{Bpr}^k_{\mathrm{j}_1}$, for $k = 1,2, \cdots, N_k$; here, $N_{\mathrm{Tpr}}$ is the number of target-process changes, $N_k$ is the number of background process types, and $j_1$ is one version of a background process. Measurements for the configurations of interest are repeated $N_{\mathrm{r}}$ times. For each target process $\mathrm{Tpr}_i$, the sample mean $\bar{x}^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_j}$ and sample variance $s^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_j}$ of the measured signals $V^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_{\mathrm{j}_1},1}, \cdots, V^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_{\mathrm{j}_1},N_{\mathrm{r}}}$ are computed. Then, the F-value is computed as:

$$F^{pc,t}_N = N_{\mathrm{r}} \frac{\mathrm{Var}(\bar{x}^{pc,t}_{\mathrm{Tpr}_1,\mathrm{Bpr}^k_{\mathrm{j}_1}}, \bar{x}^{pc,t}_{\mathrm{Tpr}_2,\mathrm{Bpr}^k_{\mathrm{j}_1}}, \cdots, \bar{x}^{pc,t}_{\mathrm{Tpr}_{N_{\mathrm{Tpr}}},\mathrm{Bpr}^k_{\mathrm{j}_1}})}{\mathrm{Mean}(s^{pc,t}_{\mathrm{Tpr}_1,\mathrm{Bpr}^k_{\mathrm{j}_1}}, s^{pc,t}_{\mathrm{Tpr}_2,\mathrm{Bpr}^k_{\mathrm{j}_1}}, \cdots, s^{pc,t}_{\mathrm{Tpr}_{N_{\mathrm{Tpr}}},\mathrm{Bpr}^k_{\mathrm{j}_1}})} \quad (3)$$

The $F^{pc,t}_N$ value is large when the measured signals exhibit large changes as the target process varies and small changes as the measurements are repeated.

Because the arrays $\mathbf{T}^{pc,t}$ and $\mathbf{B}^{k,pc,t}$ are by definition independent of measurement noise, averaged signals are used to compute the $F^{k,pc,t}_{\mathrm{B}}$ value. This computation can be performed for each type of background process $k$ separately, keeping other background processes constant. For each version $i$ of the target process $\mathrm{Tpr}_i$, first a background process $\mathrm{Bpr}^k$ is varied as $\mathrm{Bpr}^k_1, \mathrm{Bpr}^k_2, \cdots, \mathrm{Bpr}^k_{N^k_{\mathrm{Bpr}}}$, where $N^k_{\mathrm{Bpr}}$ is the number of possible versions in that process, and the mean $\bar{\bar{x}}^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k}$ and the variance $\bar{s}^{k,pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k}$ of the sample mean $\bar{x}^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}^k_j}$ are computed. Then the F-value is computed as:

$$F^{k,pc,t}_B = N^k_{\mathrm{Bpr}} \frac{\mathrm{Var}(\bar{\bar{x}}^{pc,t}_{\mathrm{Tpr}_1,\mathrm{Bpr}^k}, \bar{\bar{x}}^{pc,t}_{\mathrm{Tpr}_2,\mathrm{Bpr}^k}, \cdots, \bar{\bar{x}}^{pc,t}_{\mathrm{Tpr}_{N_{\mathrm{Tpr}}},\mathrm{Bpr}^k})}{\mathrm{Mean}(\bar{s}^{pc,t}_{\mathrm{Tpr}_1,\mathrm{Bpr}^k}, \bar{s}^{pc,t}_{\mathrm{Tpr}_2,\mathrm{Bpr}^k}, \cdots, \bar{s}^{pc,t}_{\mathrm{Tpr}_{N_{\mathrm{Tpr}}},\mathrm{Bpr}^k})} \quad (4)$$

### C. Practical Considerations

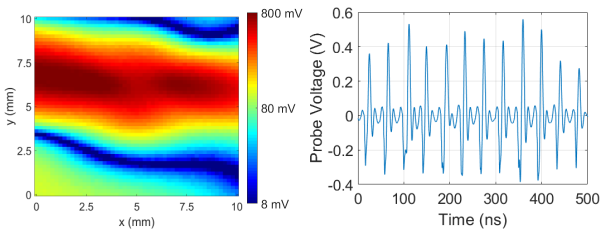
From (3) and (4), computing the F-statistics requires measurements for all possible variations in processes, which may be infeasible for large $N_{\mathrm{Tpr}}$ and $N^k_{\mathrm{Bpr}}$ values. F-statistics can be evaluated using fewer measurements by adopting leakage models such as Hamming weight (HW) model or Hamming distance (HD) model for target and background processes in the DUT. Such leakage models are commonly used to correlate observed signals to system outputs in EM SCA attacks. Because HWs represent the number of bits with value 1 in binary representation, to correlate signals with HWs, the information-leaking block must reset to a known state (e.g., a pre-charged bus) after every operation, i.e., signals will not depend on the previous state of the block but only on its current state. For example, if a block is reset to 00h before an operation, signals will be similar for block values F0h and 0Fh, both having the same HW 4, and signals will vary for values 0Fh (HW 4) and 1Fh (HW 5). HD represents the number of $0 \rightarrow 1$ and $1 \rightarrow 0$ bit transitions between 2 binary numbers. The HD model is used for information-leaking blocks that hold their previous states before updating their values (e.g., registers).

The proposed methodology assumes complete control over the processes. While the DUT may in general be assumed to not be a black box, i.e., observers can control inputs and outputs, all chip processes may not be controllable; e.g., the program counter will increment irrespective of fetched instruction or data. Therefore, it may not be possible to ensure background processes are constant for the computation of $F^{pc,t}_N$ or $F^{k,pc,t}_B$. Furthermore, the target process and all background processes can be studied independently, with no dependencies between them, only in some special cases; e.g., in the AES algorithm, byte-substitution operations are performed independently, allowing byte-wise EM SCA attacks [6]. For complex embedded systems, however, the target and certain background processes may not be completely independent.

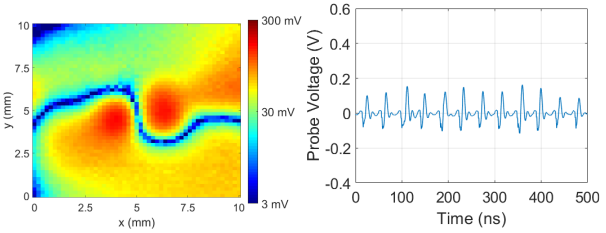
To address these confounding issues, a non-independent background process is split into two processes—an independent background process and an intermediary "linking process" that is correlated with both the background and the target processes; e.g., an instruction can be split into output-independent opcode fetch/decode and the correlated input operand values. Here, the dependence between the linking process and the target process varies for changes in a background process; e.g., input values will be strongly correlated with the output values for a MOV instruction compared to an ADD instruction. To reduce the sensitivity of the F-value estimates in (4) towards linking processes, test cases are specifically chosen such that the variations in the linking process are included during the characterization of a background process; e.g., by varying the input operands in the arithmetic and logical instructions to achieve the same results, the $F^{k,pc,t}_B$ metric can be made less sensitive to operand values.

### D. Measurement Protocol

To demonstrate the methodology, a protocol is proposed to isolate measurement configurations sensitive to output data. The protocol is implemented in $N_{\mathrm{b}} + 1$ stages, where $N_{\mathrm{b}}$ is the number of background processes ($N_{\mathrm{b}} = 2$ for the chosen test case in Section III.A). Stages are numbered from 0 onwards, where the $0^{\mathrm{th}}$ stage characterizes obfuscation due to measurement noise. It is assumed that the configuration with the minimum sum in (2), will belong to an intersection of optimal configurations identified in each stage of the protocol, i.e., the F-values computed in each stage are maximized separately (minimizing the 1/F-values) and configurations with F-values maximized for all stages will include the optimal ones. To this end, ineffective




(a) *x*-orientation




(b) *y*-orientation

Fig. 3. Space-time distribution of the signals measured using two orthogonal probe orientations at 51×51 observer locations for the instruction MOV A, #00h. Spatial map was plotted at 200 ns and time variation was plotted at the centre of the chip.

configurations identified in one stage are discarded from the analysis in the subsequent stage. Critical F- values $F_{\mathrm{c}}$ from F-distributions are used as thresholds for discarding such configurations with null hypothesis testing. These values are computed at a confidence ratio of 99.99%, for appropriate $N_{\mathrm{Tpr}}$, $N_{\mathrm{Bpr}}^{k}$, and $N_{\mathrm{r}}$ values at each stage of the protocol. In cases where signals are obfuscated significantly, the confidence with which configurations can be isolated will reduce.

Stage 0 of the protocol estimates $F_N^{pc,t}$ using (3), which is followed by the creation of a mask, to be used for the next stage,

$$Mask_0^{pc,t} = \begin{cases} 0 & \text{if } F_N^{pc,t} < F_{\mathrm{c},0} \\ 1 & \text{if } F_N^{pc,t} \geq F_{\mathrm{c},0} \end{cases} \quad (5)$$

Every subsequent stage $k$ characterizes algorithmic noise from different background processes by computing $F_B^{k,pc,t}$ only for the optimal configurations identified in the previous stage $k-1$ using the generated masks as

$$Mask_k^{pc,t} = \begin{cases} 0 & \text{if } Mask_{k-1}^{pc,t} \times F_B^{k,pc,t} < F_{\mathrm{c},k} \\ 1 & \text{if } Mask_{k-1}^{pc,t} \times F_B^{k,pc,t} \geq F_{\mathrm{c},k} \end{cases}, \quad (6)$$

for $k = 1,2,\cdots,N_{\mathrm{b}}$. The F-statistics are first computed for background processes that do not depend on inputs supplied to the system. Optimal configurations identified after the last stage (with $Mask_{N_{\mathrm{b}}}^{pc,1} = 1$) can be tested for potential information recovery.

## III. Measurement Results

### A. Measurement Setup

The setup shown in Fig. 1 used Atmel's variant of 8051, AT89S51 as the DUT. The chip was programmed with HEX files generated from test codes compiled using Keil's 8051 emulator. The files were loaded to the memory of the chip using SPI transfer protocol, with an Arduino board used as the intermediary. The chip operated at a clock frequency of 2 MHz. A Keysight Infiniium oscilloscope was used as the signal capture device with signals sampled at $\Delta t = 0.2$ ns. For the clock period of 500 ns, $N_{\mathrm{t}} = 2500$ points were recorded.

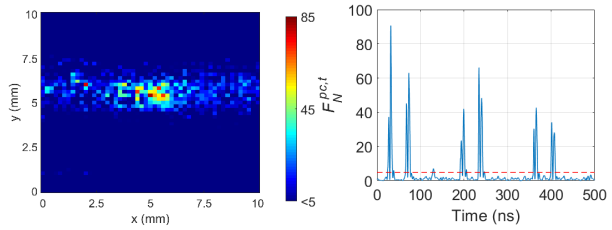
```
ORG 0000H
SJMP TEST
TEST:  CPL P1.3     ; measurement marker 1
       CPL P1.3     ; measurement marker 2
       MOV A,#00h   ; output HW 0
       MOV A,#01h   ; output HW 1
       MOV A,#03h   ; output HW 2
       MOV A,#07h   ; output HW 3
       MOV A,#0Fh   ; output HW 4
       MOV A,#1Fh   ; output HW 5
       MOV A,#3Fh   ; output HW 6
       MOV A,#7Fh   ; output HW 7
       MOV A,#0FFh  ; output HW 8
       MOV A,#0FFh  ; Same fetch as previous cycle
       SJMP TEST    ; loop statement
```

Fig. 4. Assembly code used to collect signals for computing $F_N^{pc,t}$. Background processes were constant – MOV instruction fetching data from program memory. An additional MOV instruction was appended at the end to ensure the background processes are consistent for all captured signals.

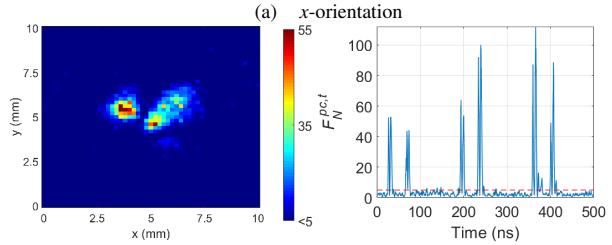
Bandwidth for the setup was limited to 500 MHz by the oscilloscope. The chip's surface area $l_{\mathrm{x}} \times l_{\mathrm{y}}$ is 10 mm × 10 mm. Probe locations were chosen over $N_{\mathrm{x}} \times N_{\mathrm{y}} = 51 \times 51$ grid points over the chip area, spaced at 0.2 mm in both axes. A 30 dB pre-amplifier stage was used to boost signals from a 1-mm H-field probe from Langer. The probe was positioned using the Riscure EM probe station which has a step-size of 2.5 $\mu$m. The setup allows measurements with *x* and *y* probe orientations, where the probe can move up to 40 mm in all directions. The probe was fixed at a height of 0.2 mm above the chip. Measurement and analysis costs were reduced by storing and processing data locally on the oscilloscope. Acquisition time associated with each stage ranged from 2 to 3 hrs. Space-time maps of received signals for both orientations are shown for one instruction in Fig. 3. The figure shows significant dependence of the measured signal on the probe configuration.

Processes sensitive to the output of two-operand arithmetic and logical instructions were chosen as the target for experiments. Since these instructions use the same architectural blocks, for this target process, two background processes were considered – increment of the program counter and changes in combinations of memory location and instruction functions. In pipelined embedded systems, processes within each pipeline stage must be accounted for, within background processes. Pipeline effects do not need to be considered, however, in these experiments because fetch- and execute-related operations use the same architectural blocks for the given set of instructions in the 8051 MCU [12].

Complete computation of F-statistics for the 8051 MCU requires measurements for $N_{\mathrm{Tpr}} = 256$ data values. For each of these data values, a total of $N_{\mathrm{Bpr}}^{1} = 4096$ variations in program counter (maximum code size) and $N_{\mathrm{Bpr}}^{2} = 16$ variations in background processes–8 arithmetic and logical instructions and 2 memory locations of operands–are to be measured. Measuring all process combinations using all probe configurations is infeasible; thus, leakage models were used for estimation of the F-statistics. The MCU implements a shared 8-bit bus with all bus bits pre-charged to logic 1. This allows the use of the HW model for correlating signals to data transfers on the bus. Using HW model of leakage, $N_{\mathrm{Tpr}}$ was reduced from 256 possible values of output data to 9 values corresponding to a HW from 0 to 8. HD model was used to represent variation in program counter-dependent leakage.

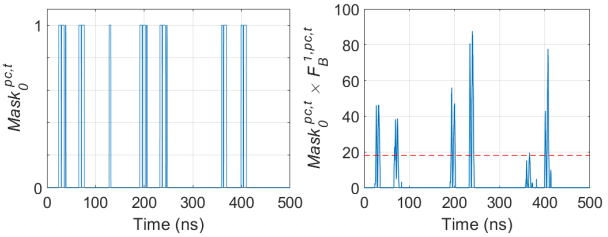



(a) $x$-orientation




(b) $y$-orientation

Fig. 5. Space-time maps of $F_N^{pc,t}$ for two probe orientations. Space map was plotted at 23 ns and time variations were plotted at the centre of the chip. Configurations with F-values below the threshold $F_{c,0} = 4.8$ (dark blue regions for space maps and below the dashed red lines for time plots) were assigned $Mask_0^{pc,t} = 0$ and ignored when computing $F_B^{1,pc,t}$ .



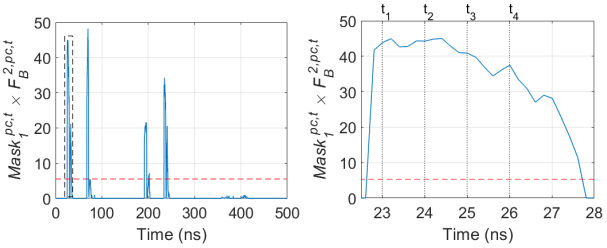

(a) (b)

Fig. 6. (a) Time instances where $F_N^{pc,t}$ values are above the critical threshold $F_{c,0} = 4.8$ (Fig. 5), i.e., $Mask_0^{pc,t} = 1$ are identified. (b) The $F_B^{1,pc,t}$-value is computed at those time intervals and the time instances where $Mask_0^{pc,t} = 1$. Similarly, $F_B^{2,pc,t}$-value is computed where $F_B^{1,pc,t} > F_{c,1}$~18 (dashed red line).

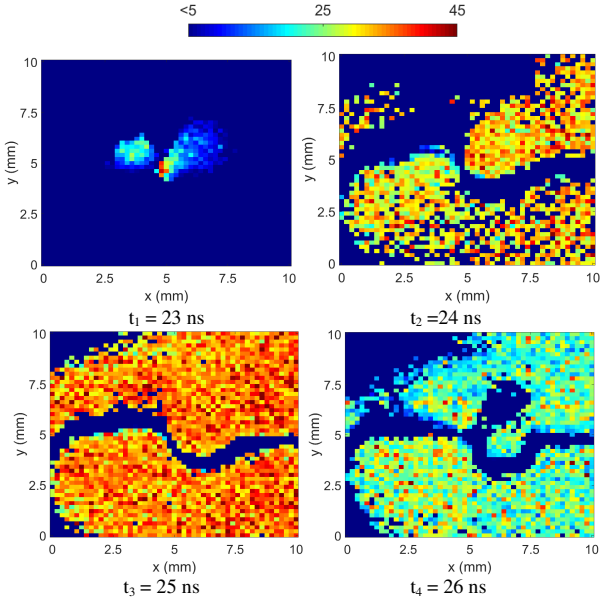
The program counter increments by 1 for every fetch from program memory. To estimate $F_B^{1,pc,t}$, $N_{\mathrm{Bpr}}^1$ =4 variations in program counter switching, from HD 1 to HD 4, were considered, and target processes were varied for each of these variations. To minimize the cost of computing $F_B^{2,pc,t}$, one instruction was selected from each functional group – MOV for data transfer, ADD from arithmetic instructions, and ORL from logical instructions. For each of these instructions, the source of operands may vary between 2 locations – data memory or program memory. The total number of variations in this background process was $N_{\mathrm{Bpr}}^2 = 6$.

### *B. Computation of F-statistics*

Variation in output data $\mathrm{Tpr}_i$ is represented using 9 values 0b00000000, 0b00000001, 0b00000011,⋯, 0b11111111 in binary notation, such that all HWs from 0 to 8 are covered. Test program to compute $F_N^{pc,t}$is shown in Fig. 4. The number of measurement repetitions was fixed as $N_r = 10$. Space-time maps of $F_N^{pc,t}$ are shown in Fig. 5. Critical F-value $F_{c,0}$, for computing $F_N^{pc,t}$ was found to be ~4.8. To compute $F_B^{1,pc,t}$, 4 HD variations in the program counter were used as background processes, for 4 HW variations of output data, due to code length constraints. The background process $\mathrm{Bpr}_j^2$, which represents the pair of instruction function and memory location, was kept constant as MOV from program memory.




(a) Time variation at (5, 5) mm




(b) Spatial maps of $Mask_1^{pc,t} \times F_B^{2,pc,t}$

Fig. 7. (a) Time instances were selected from the plotted curves to analyze spatial variation of the $F_B^{2,pc,t}$ statistic, along with threshold $F_{c,2} = 5.3$ (dashed red line). (b) Distribution of $F_B^{2,pc,t}$ varied significantly across space. Results shown for $y$-oriented probe.

The threshold $F_{c,1}$ was found to be ~18. Configurations with $F_N^{pc,t}$ lower than the critical value were ignored when computing $F_B^{1,pc,t}$ , by only considering configurations with $Mask_0^{pc,t} = 1$ as shown in Fig. 6. Comparing F-values at the center of the chip in Figs. 5 and 6, it can be inferred that F-values close to ~360 ns show some dependence on outputs which is obfuscated by program counter processes. Computation of $F_B^{2,pc,t}$ used 9 variations in target processes with 6 variations in background processes. Ineffective configurations were ignored by only computing the F-statistic for configurations with $Mask_1^{pc,t} = 1$. The threshold $F_{c,2}$ was found to be ~5.3. Space-time maps for the $F_B^{2,pc,t}$statistic is shown in Fig. 7. Comparing the $F_B^{2,pc,t}$ statistic with $F_N^{pc,t}$ and $F_B^{1,pc,t}$ values at the center of the chip, it was observed that all F-values between 300 ns to 450 ns were below the threshold and are ineffective configurations for information recovery, while the values remain consistent for other time intervals. From spatial maps, it was observed that the $F_B^{2,pc,t}$ statistic, masked with $Mask_1^{pc,t}$, was highly localized during the initial loading of the bus. As time progressed, localization reduced, and the statistic showed very high dependence on the target for multiple on-chip configurations. Following this, the F-values reduced till they dropped below $F_{c,2}$, indicating they were ineffective configurations. Spatio-temporal variations of $Mask_1^{pc,t} \times F_B^{2,pc,t}$depend on chip layout as well as variations

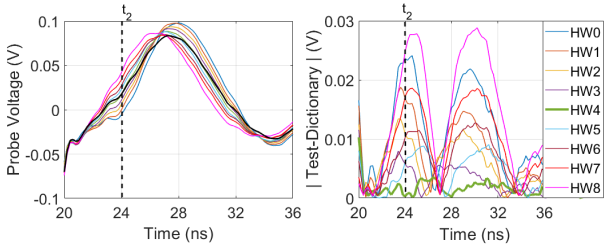



Fig. 8. Signals for different HWs along with a test signal (black) with HW 4 (left) at the centre of the chip, with clear separation at certain time instances. Difference between test signal and reference signals (right) identified HW 4 dictionary correctly as the best fit at 24 ns.

in the data-dependent current. The most optimal configurations were found to be close to the center of the chip at ~24 ns ($t_2$ in Fig. 7) in both orientations.

### C. Information Recovery from Isolated Configurations

The potential of optimal configurations to recover information was tested by identifying output data HWs for a test code with several randomly generated background process variations, which in turn randomizes output data. The experiment described in this section mimics a profiled attack [11], where profiled EM models are constructed a priori, for an implementation with fully controlled access. Test signals, from similar implementations, corresponding to unknown system inputs are correlated with constructed profiles to recover information. For the 8051 MCU, output data-dependent signals corresponding to each output HW were compiled into a reference dictionary. Signals for an optimal probe configuration and time interval are shown in Fig. 8. Comparisons can be performed at optimal configurations by computing the difference between test and dictionary signals and identifying the dictionary with least difference as best fit.

Instructions such as SUBB and XOR, which were not considered for estimation of F-statistics, were included for tests, to verify the coverage of the protocol for the selected set arithmetic and logical instructions. The test code included 100 instructions. The test code and output HW identified for an optimal configuration are shown in Fig. 9. Test configurations included 5 randomly selected measurement configurations, where $Mask_2^{pc,t} = 1$. Results are shown in Table I.

TABLE I. ACCURACY OF INFORMATION RECOVERY

| **Probe Configuration** | | **Time instant (ns)** | $\boldsymbol{Mask_1^{pc,t} \times F_B^{2,pc,t}}$ | **Accuracy (%)** |
|---|---|---|---|---|
| ***Location*** $\left(x_{n_x}, y_{n_y}\right)$ ***mm*** | ***Orientation*** | | | |
| (5,5) | *x* | 24 | 75 | 100 |
| (6,4) | *x* | 235 | 58 | 93 |
| (5,5) | *y* | 70 | 45 | 99 |
| (4,6) | *y* | 75 | 28 | 92 |
| (7,5) | y | 195 | 20 | 84 |

Although configurations with low F-values had comparatively more misclassified outputs, high F-values alone cannot ensure maximum leakage, since information recovery will also depend on signal distributions at these configurations. For example, if one value in the target process varies significantly, compared to other values, the F-statistic will be skewed. While F-statistics are reliable indicators of dependence on processes, they may not be directly used as a metric to quantify information leakage, such as correlation coefficients for EM SCA attacks on cryptography [6].

| Randomized Test Code | | Output HW |
|---|---|---|
| MOV A, #3Fh ; Pre load A register | | |
| MOV R0, #13h ; Pre load R0 register | | |
| ADD A, #23h | Test | 3 |
| XRL A, R0 | | 4 |
| ⋮ | | ⋮ |
| SUBB A,#0Fh | | 6 |
| MOV A,#00h ; Appended instruction | | |

Fig. 9. A test code with randomly generated instruction, memory referencing, and input operands, with observed output HW at one of the most optimal configurations at (5,5) mm and 24 ns for an *x*-oriented probe. The same code was repeated for different probe configurations to allow comparisons of accuracy of information recovery. An instruction was appended to keep pipeline operations consistent.

## IV. CONCLUSION

This article presented a measurement protocol to rapidly isolate information-revealing measurement configurations for a general embedded system. The protocol was used to isolate configurations sensitive to the output of instruction execution in the 8051 MCU. Isolated configurations were tested to verify that they can effectively recover the HWs of the output data. The proposed methodology can be extended to isolate effective configurations for off-chip leakage sources by increasing scan area, subject to increasing measurement costs. Methods for identification of such configurations can be extended to source localization for EM interference testing [9] and fingerprinting techniques for Trojan detection [10]. Rapid isolation of such configurations also enables testing whether countermeasures designed to mask information leakage do actually rectify vulnerabilities as intended [11].

## REFERENCES


[1] M. Vuagnoux and S. Pasini, "An improved technique to discover compromising electromagnetic emanations," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, pp. 121-126, July 2010.

[2] A. Zajic and M. Prvulovic, "Experimental demonstration of electro-magnetic information leakage from modern processor-memory systems," *IEEE Trans. EMC*, vol. 56, no. 4, pp. 885-893, Aug. 2014.

[3] Y. I. Hayashi *et al.*, "Analysis of electromagnetic information leakage from cryptographic devices with different physical structures," *IEEE Trans. EMC*, vol. 55, no. 3, pp. 571–580, June 2013.

[4] V. V. Iyer and A. E. Yilmaz, "Using the ANOVA F-statistic to rapidly identify near-field vulnerabilities of cryptographic modules," in *Proc. IEEE Int. Microw. Symp.,* June 2021.

[5] F. Unterstein *et al.*, "Dissecting leakage resilient prfs with multivariate localized em attacks," in *Proc. COSADE*, July 2017.

[6] V. V. Iyer and A. E. Yilmaz, "An adaptive acquisition approach to localize electromagnetic information leakage from cryptographic modules," in *Proc. IEEE Texas Wireless Symp.,* Mar. 2019.

[7] F. Werner *et al.*, "A method for efficient localization of magnetic field sources excited by execution of instructions in a processor," *IEEE Trans. EMC*, vol. 60, no. 3, pp. 613-622, June 2018.

[8] B. F. Jamroz *et al.*, "Accurate monte carlo uncertainty analysis for multiple measurements of microwave systems," in *Proc. IEEE MTT-S Int. Microw. Symp.*, Jun. 2016.

[9] A. Gorbunova, A. Baev, M. Konovalyuk, and Y. Kuznetsov, "Localization of cyclostationary EMI sources based on near-field measurements," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, pp. 450-455, Aug. 2015.

[10] J. Balasch, B. Gierlichs, and I. Verbauwhede, "Electromagnetic circuit fingerprints for hardware Trojan detection," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, pp. 246-251, Aug. 2015.

[11] G. Li, V. Iyer, and M. Orshansky, "Securing AES against localized EM attacks through spatial randomization of dataflow," in *Proc. IEEE HOST,* May 2019.

[12] J. Wharton, "An Introduction to the Intel MCS-51 Single-Chip Microcomputer Family," Intel Corporation, Application Note AP-69, May 1980.