UNIVERISTY OF PIRAEUS - DEPARTMENT OF INFORMATICS

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

MSc «Cybersecurity and Data Science»

ΠΜΣ «Κυβερνοασφάλεια και Επιστήμη Δεδομένων»

MSc Thesis

Μεταπτυχιακή Διατριβή

Thesis Title: Τίτλος Διατριβής:	**Development of Electromagnetic (EM) Attack Platform and Evaluation of Secure Embedded Implementations.** Ανάπτυξη Πλατφόρμας Ηλεκτρομαγνητικών (ΕΜ) Επιθέσεων και Αξιολόγηση Ασφαλών Ενσωματωμένων Υλοποιήσεων
Student's name-surname: Ονοματεπώνυμο φοιτητή:	**KONSTANTINOS SPYRIDON MOKOS** ΚΩΝΣΤΑΝΤΙΝΟΣ ΣΠΥΡΙΔΩΝ ΜΩΚΟΣ
Father's name: Πατρώνυμο:	**CHRISTOS** ΧΡΗΣΤΟΣ
Student's ID No: Αριθμός Μητρώου:	ΜΠΚΕΔ21037
Supervisor: Επιβλέπων:	**ATHANASIOS PAPADIMITRIOU, ASSISTANT PROFESSOR** ΑΘΑΝΑΣΙΟΣ ΠΑΠΑΔΗΜΗΤΡΙΟΥ, ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ

November 2024/ Νοέμβριος 2024

3-Member Examination Committee

Τριμελής Εξεταστική Επιτροπή

Michael Psarakis
Associate Professor
Μιχαήλ Ψαράκης
Αναπληρωτής Καθηγητής

Athanasios Papadimitriou
Assistant Professor
Αθανάσιος Παπαδημητρίου
Επίκουρος Καθηγητής

Panagiotis Kotzanikolaou

Professor
Παναγιώτης Κοτζανικολάου
Καθηγητής

Table of Contents

Acknowledgments

The primary motivation for enrolling in this master's program was the opportunity to conduct research on hardware security using the advanced equipment available in the university's Embedded Systems Lab (ESLab). Upon completing two semesters of study and beginning my thesis, I had the chance to work in the ESLab, gaining invaluable experience that would have been impossible to achieve on my own.

I would like to express my deepest gratitude to my supervisor, Dr. Athanasios Papadimitriou, whose expertise, understanding, and passion not only enabled me to conduct this research but also broadened my horizons in the field of hardware security. I am also immensely grateful to Associate Professor Michael Psarakis for granting me comprehensive access to the lab's equipment and enabling me to conduct in-depth research.

A special thank you goes to my peer, PhD student Amalia Koufopoulou, for her invaluable support during the lab hours of my thesis, providing guidance and assistance in configuring and operating the lab's equipment.

On a personal note, I extend my heartfelt appreciation to my parents, Eleni Sapriki and Christos Mokos, and grandparents, Spyridon and Chrysoula Sapriki, for their unwavering love and support throughout my academic and life journey. Their sacrifices have not gone unnoticed, and it is their encouragement that has kept me motivated and focused during challenging times. To that extent, I would like to dedicate this thesis to my grandparents and my parents.

Περίληψη

Με την ραγδαία ανάπτυξη του Διαδικτύου των Πραγμάτων (IoT), τα τελευταία χρόνια, έχουμε παρατηρήσει μια αυξημένη χρήση ενσωματωμένων συσκευών σε μια πληθώρα από διαφορετικούς τομείς, από έξυπνα σπίτια, ιατρικές συσκευές μέχρι εφαρμογές σε αυτοκίνητα και βιομηχανικές εφαρμογές. Οι ενσωματωμένες συσκευές ήταν πάντα μέρος της ζωής μας, από τις έξυπνες κάρτες, όπως οι πιστωτικές κάρτες, Αυτόματα ταμειακά μηχανήματα (ATM) και άλλες παρόμοιες συσκευές που βοηθούν στην αποθήκευση πληροφοριών ή την αυτοματοποίηση απλών διαδικασιών.

Σήμερα, το οικοσύστημα των ενσωματωμένων συσκευών έχει επεκταθεί σημαντικά καλύπτοντας ένα μεγάλο εύρος από βιομηχανίες και εφαρμογές. Καθώς αυτές οι συσκευές εξελίσσονται μπορούν να εκτελέσουν ολοένα και πιο προηγμένες λειτουργίες, το οποίο αυξάνει τη χρησιμότητά τους. Αυτή η εξέλιξη έχει ως αποτέλεσμα την αυξανόμενη ενσωμάτωση τέτοιων συσκευών στην καθημερινότητα μας, όπου σε πολλές περιπτώσεις διαχειρίζονται κρίσιμες διαδικασίες, όπως αυτόνομα αυτοκίνητα και ιατρικές συσκευές. Επιπλέον, ο όγκος των δεδομένων που συλλέγονται από αυτές έχει εκθετική αύξηση και σε πολλές περιπτώσεις συλλέγουν ευαίσθητες πληροφορίες.

Καθώς βασιζόμαστε όλο και περισσότερο σε αυτές τις τεχνολογικές εξελίξεις, είναι ζωτικής σημασίας να κατανοήσουμε και τους κινδύνους ασφάλειας που τους συνοδεύουν. Αυτές οι τεχνολογίες, που βασίζονται σε ενσωματωμένες συσκευές, ανοίγουν μια νέα επιφάνεια επίθεσης, η οποία συνήθως δεν υφίσταται σε εφαρμογές λογισμικού, που σχετίζεται άμεσα με τα υλικά εξαρτήματα των συσκευών, όπως οι επεξεργαστές τους. Η ασφάλεια των πληροφοριών, που συλλέγονται από αυτές τις συσκευές, μαζί με τις κρίσιμες λειτουργίες που ελέγχουν, είναι άμεσα συνδεδεμένη με την ασφάλεια του υλικού τους.

Στην παρούσα διπλωματική εργασία, θα παρουσιάσουμε την ανάπτυξη και υλοποίηση μιας Ηλεκτρομαγνητικής (EM) Πλατφόρμας Επίθεσης, σχεδιασμένης για να αξιολογήσει την ανθεκτικότητα ασφαλών ενσωματωμένων συστημάτων απέναντι σε EM παρεμβολές. Χρησιμοποιώντας μια μεθοδική προσέγγιση, θα εξετάσουμε ολόκληρη την επιφάνεια ενός μικροελεγκτή με σκοπό να χαρτογραφήσουμε την επιφάνεια του και να εντοπίσουμε περιοχές που παρουσιάζουν ευαισθησία όταν υποκινηθούν σε ηλεκτρομαγνητικούς παλμούς.

Θα ξεκινήσουμε παρέχοντας μια επισκόπηση του εξοπλισμού που χρησιμοποιήσαμε για να δημιουργήσουμε το εργαστηριακό περιβάλλον, περιλαμβάνοντας τη διαδικασία που ακολουθήσαμε προκειμένου να αυτοματοποιήσουμε τη χρήση τους ώστε να κάνουμε την πλατφόρμα όσο το δυνατόν πιο αποδοτική και αυτόνομη. Στη συνέχεια, θα παρουσιάσουμε την μελέτη μας, η οποία περιλαμβάνει την μεθοδολογία που προτείνουμε για την προσέγγιση αυτού του προβλήματος. Τέλος θα παρουσιάσουμε τα αποτελέσματα των πειραμάτων που διεξήγαμε με τη χρήση αυτής της πλατφόρμας και μεθοδολογίας και θα καταλήξουμε στα τελικά μας συμπεράσματα.

Abstract

With the rapid growth of the Internet of Things (IoT) over the past years we have seen an increased use of embedded devices in a range of different fields from smart homes and medical devices to automotive and industrial applications. Embedded devices have always been part of our lives, from smart cards, like credit cards, Automated Teller Machines (ATMs), and other similar devices that aid in storing information or automating simple processes.

Today, the ecosystem of embedded devices has expanded significantly, covering a wide range of industries and applications. As these devices evolve, they can perform more advanced features, which increases their utilization. This evolution has as a result the increasing incorporation of such devices in our daily lives, in many cases managing critical operations like self-driving cars and healthcare devices. Moreover, the amount of data these devices collect, and their sensitivity, have seen exponential growth.

As we rely more on these technological advances it is of paramount importance to understand the security risks they pose. These new technologies, relying on embedded devices, open a new attack surface, not commonly seen in software applications, related to their hardware components, like their processors. The security of the information these devices collect and store, along with the control over critical functions, is directly dependent on the security of the underlying hardware.

In this thesis, we will present the development and implementation of an Electromagnetic (EM) Attack Platform designed to evaluate the resilience of secure embedded systems against EM interference. Utilizing a methodical approach, we will examine the entire surface of a target microcontroller unit (MCU) to identify and map areas on its surface that are susceptible to EM-induced faults.

We will begin by providing an overview of the hardware equipment we use to create our lab setup, including all the automation procedures we establish to make the platform as efficient and autonomous as possible. Then we will present our case study, including the proposed methodology for this problem. Finally, we will present the results of the experiments we performed using this lab setup and proposed methodology and draw our final conclusions.

1 Introduction

In recent years, embedded devices have seen exponential growth, extending from Internet of Things (IoT) devices [16] to medical [14], automotive, and industrial applications [17], [18], [19]. These devices are now an integral part of our daily lives, controlling everything from home automation systems and wearable health monitors to advanced driver-assistance and self-driving systems in vehicles. Even though this rapid integration enhanced our daily lives, it also raised significant security concerns [1]. Not only do these devices manage critical functions, but they also collect a significant amount of sensitive information. Securing against potential threats that might affect the integrity of operations performed by these devices or the confidentially of the information collected by them has become more urgent now than ever [2]. This is especially true in environments where safety and privacy are paramount, such as healthcare devices and connected vehicles, as mentioned earlier.

The increased complexity and variety of these systems pose a unique challenge in cybersecurity [15], demanding specialized strategies to protect against a broad spectrum of vulnerabilities, including the hardware attack surface. Attacks on the hardware layer can compromise the security and reliability of embedded devices by altering the behaviour of the device or leaking sensitive information. Most of these devices rely on the use of cryptographic algorithms, which run locally, to secure the collected data and other sensitive information. While running the cryptographic algorithms locally offers many advantages like security, since there are no plaintext data transmissions that can be intercepted, increased performance, and high availability due to reduced reliance on real-time data transition, it also opens an attack surface to local threats. These threats can range from physical attacks to more sophisticated methods like side-channel [2], [11] and fault injection [3], [13] attacks. These attacks rely on the inherent vulnerabilities of the device's hardware and are extremely hard to mitigate, especially for devices that are already in the field, or even worse if they reached their end of life, resulting in no support from the vendors [20].

Among these threats, fault injection attacks pose a significant risk due to their ability to bypass conventional security measures and disrupt the operations of embedded devices [12]. This is due to the nature of the attack, which is an active attack, since it interferes with the target's operation, unlike side-channel attacks that are passive and rely on the target's emissions during operation. Fault injection attacks manifest in a variety of forms ranging from tampering with external signal sources to the target to influencing its internal components directly. In the next few paragraphs, we will briefly review some of the most known types of fault injection in hardware, outlining their unique characteristics.

Clock glitch attacks involve the manipulation of the processor's clock signal [7], to slow down, speed up its speed, introduce small breaks, or introduce very short and precise malfunctions (glitches) to alter the normal operation of the device temporarily. This can cause the processor to perform instructions incorrectly or skip them.

Power glitch attacks [3][4], a subset of voltage glitching attacks, work in a similar manner to clock glitch attacks, by manipulating the power supply of the device. By altering the voltage supplied to it to either undervolt or overvolt transient fault can be induced during the operation of the device. This can lead to altering the software behaviour [9], bypassing security measures altogether, by skipping critical commands, like security checks, to inducing faults in the process of cryptographic algorithms. Corrupting the operation of a cryptographic algorithm can lead to errors that can be propagated to the final output of the operation, which can lead to information leakage [10].

Temperature Variation Attacks [5] rely on extreme temperature variations to force the devices into an erroneous state. By operating the device outside its specified safe operational characteristics, by either lowering or increasing its temperature, it is possible to induce errors in its memory. These types of attacks can also be used in combination with other types of fault attacks to increase target susceptibility to faults [8].

Optical fault injection attacks [6] utilize the use of intense light sources, like lasers, to induce faults in a device. These types of attacks can be very precise and influence specific areas of the device's processor and even target individual transistors. This technique can alter the contents of memory on influent the execution path of the processor. The high accuracy required for these attacks results in increased complexity when mounting them since it requires precise instruments and direct access to the target component. In comparison, the other types of attacks mentioned so far have a global scope and are easier to implement.

Electromagnetic Fault Injection (EMFI) attacks use electromagnetic interference to induce errors in specific parts of the device, like optical attacks, but without the same level of precision. Despite this, the method is still highly localized and can target specific components on a chip, such as memory cells, the ALU, or other specific parts of the CPU. Moreover, depending on the type of EM, we can also perform precise attacks in the time domain, by injecting EM pulses at specific timings in the program execution, instead of having continuous EM interference.

There are several types of fault injection attacks, each with its own unique characteristics, but we can categorize them all into two different groups based on their scope of impact which is either global or local. Global fault injection attacks typically affect the entire system, potentially creating widespread disruption or failure. Clock glitching, power glitching, and temperature variations attacks have a global scale since they impact the entire process of the target. On the other hand, local fault injection attacks target specific components or regions within the target. These types of attacks can achieve precise results by manipulating specific elements like memory cells or sections of the chip.

In this thesis, we will focus on Electromagnetic Fault Injection (EMFI) attacks, as they provide an optimal balance between precision and ease of implementation. When compared to attacks with a global scope they offer higher localization, while they are easier to implement when compared to attacks with higher precision like optical fault injection attacks. Due to their localized nature, there is a significant overhead for setting up a successful testbench, identifying vulnerable areas on the surface of the target, and then mounting the attack. Our research aims to provide a detailed methodology that aims to map the surface of the target to identify sensitive areas that generate a high number of faults when subjected to EMFI attacks. Mapping the target surface will result in more accurate attacks with lower complexity in time. Additionally, this can outline specific weaknesses in the internal design of the IC, that could be addressed by the manufacturer.

2 State of art

Before beginning to design our methodology, it is imperative to review the current state of research in this field. During our research, we only managed to identify a small number of research papers referring to this exact problem statement. Even though the field of fault attacks has existed for decades [21], [22], [23] it appears the is no significant research performed in creating methodologies that aid in mapping the surface of a target in terms of its susceptibility to fault injection attacks. The research field is even more limited with regard to EMFI attacks since these types of attacks have only been explored in more recent studies. In the following paragraphs, we will review some of the most interesting research papers we identified, pertinent to the objectives outlined in this thesis.

In the study (Madau, 2018) [24], the authors developed a susceptibility criterion for EMFI attacks to aid in mapping different points on the chip surface depending on the sensitivity they exhibited when subjected to fault injections. The criterion is based on additional measurements performed on the target, including measuring the emitted power at the clock frequency and the connection between the emitted signal and the data being processed. Using their criterion, they scan the surface of the target and rank the points based on their susceptibility. Even though this method significantly reduces the attack surface, by more than 50%, they consider as a fault any disturbance in the normal operation of the device, and its program execution. This results in the identification of sensitive points but without any immediate value in performing efficient fault injection attacks, since the identified points might not generate useful faults.

(Wu et al., 2020) [25] propose a novel methodology for the rapid characterization of optical fault injection attacks. The methodology starts by generating a sensitivity curve to identify the most effective search space parameters. This approach also integrates deep learning, specifically a multilayer perceptron (MLP), to estimate the efficiency of various configurations of the search space parameters using a limited set of experimental data. This method improves efficiency as an exhaustive search is not required, though it requires careful configuration both in the initial dataset and the deep learning model used. The initial dataset needs to be representative of the target's potential vulnerabilities, while the use of machine learning introduces potential challenges like overfitting and bias that must be carefully considered.

Another interesting study by (Carta, 2023) [26], examines the exact problem under evaluation in our thesis, titled “Efficient attack-surface exploration for electromagnetic fault injection”. The methodology proposed in the thesis focuses on the efficient exploration of the search parameters to identify subregions that maximize the occurrence of information faults when subjected to EMFI attacks. The methodology consists of two parts, the surface search and the coordinate search. The first part involves the definition of a grid of coordinates on the target surface and the evaluation of each point using the maximum possible values for the pulse intensity and duration. If a point is susceptible to faults then the next part is to perform parameter optimization, using a multi-dimensional bisection algorithm, aiming to increase the generation of meaningful faults. This methodology provides a significant advantage by performing optimization of the search space parameters; to produce meaningful faults but this results in a significant increase in the time required to perform a comprehensive scan of the surface of the target.

It is evident that the process of mapping a target surface sensitive to EMFI attacks is a time-consuming matter, that needs to be better researched. Not only do we require a systematic approach how to identify faults, but we also need to be able and fine-tune the search space parameters to achieve meaningful results, that can aid in the implementation of attacks. In our methodology, we propose a modular testing procedure, consisting of many independent steps, which depending on the time allocated for research can be adjusted to provide valuable feedback even in a limited amount of time. Increasing the time allocated to perform these experiments will result in higher spatial accuracy and higher efficiency in identifying sensitive areas.

The rest of this thesis will be structured as follows. We will begin by providing an overview of the framework we have created to facilitate this research using MATLAB and a set of hardware equipment. First, we will review the lab setup we used to perform these types of attacks, including any automation procedures we set into place to create an automated testbench. Afterward, we will present our case study, which will outline the methodology used to perform this research. This includes the testing methodology followed to perform each scan (subjecting the target to EM pulses and monitoring its behaviour) as well as the overall testing procedure of how to systemically approach this problem in a timely efficient manner.

3 Lab Setup

3.1 Overview

The laboratory (lab) setup is utilizing a set of ready to use equipment to facilitate the easy and accurate evaluation of embedded systems against electromagnetic (EM) attacks. This setup enables the generation of EM pulses to test the susceptibility of the target against active EMFI attacks. The primary goal is to simulate real-life attack scenarios in a controlled environment, which will enable us to map the attack surface of the target device. In the following subsection, we will provide a brief overview of the specific equipment used for this Lab and the main features of each device we plan on utilizing for our experiments. The physical equipment used to perform real-life experiments is usually one of the limiting factors that define the level of detail in the results obtained from it. The lab setup consists of professional high-end precision equipment that allows us to perform tests with high accuracy and achieve consistent results across multiple test runs.

3.2 EM Attack Generation Equipment

To test the resilience of the target device against EM attacks, such as fault injections, we will need EM generation equipment. To achieve this, we will utilize a set of ready to use devices.

3.2.1 Langer EMV - Burst Power Station (BPS) 202

The Burst Power Station (BPS) 202 of Langer EMV, seen in Figure 1, will be used as the power supply and control unit for the EM probes that will used for the active EM attacks. The device can be connected to the PC via a USB interface that will enable control of the device from the BPS 202-Client software. Additionally, a DLL file is provided that can be used as a control API for the device that enables automation. Table 1 summarizes the main functionalities and technical characteristics that are of interest to our specific use case.

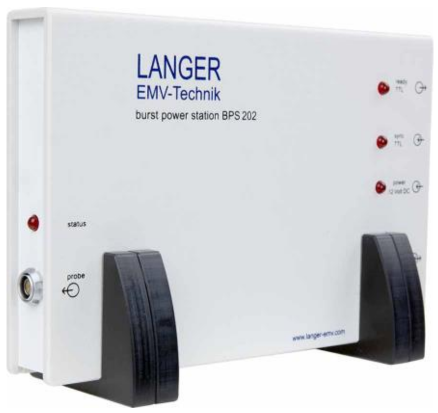

Figure 1: The Langer EMV - Burst Power Station (BPS) 202 control device

Technical Specification	Value	Notes
Output voltage	± (50 ... 500) V	The voltage range of the

		generated EM pulses (limits might differ based on the probe that is used in conjunction with BPS 202)
External trigger input	TTL / BNC	An external trigger can be supplied by the target to BPS202
Trigger-pulse delay (Timer Mode)		
min. Trigger-pulse delay (typ.)	130 ns	The minimum trigger delay the BPS202 can support (actual value will be probe-dependent)
max. Trigger-pulse delay (typ.)	100 ms	The maximum trigger delay the BPS202 can support (actual value will be probe-dependent)
max. Jitter (typ.)	± 15 ns	The variation in timing we can expect for each trigger. The Jitter defines the level of timing accuracy achievable by BPS202
Trigger delay, min. increment	10 ns	
Software	BPS 202-Client / DLL (32 Bit / 64 Bit)	

Table 1: Technical characteristics of the Burst Power Station (BPS) 202 relevant to the use case of the equipment for our laboratory setup.

3.2.2 Langer EMV - ICI HH500-15 L-EFT Pulse Magnetic Field Source

This probe, also seen in figure 2, is one out of three provided on the Langer EMV ICI 03 L-EFT set, which is part of our lab equipment, that can be used on its respective area sources emitting electric, magnetic field, and current pulses. Since our experimental setup requires a magnetic field probe we will be only utilizing one of the probes. The probe's high-resolution 500 μm probe tip in conjunction with the BPS2 control device's low jitter mode, which is supported by the probe, will provide very high precision. This will enable accurate experiments both in space (IC surface area) and time (program execution time). Table 2 outlines the most important technical information about the probe.

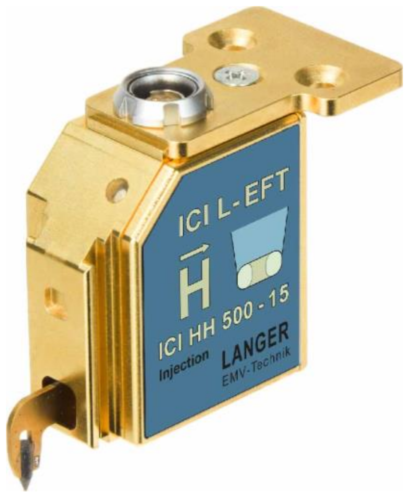

Figure 2: The LangerEMV - ICI HH500-15 L-EFT Pulse Magnetic Field Source probe

Technical Specification	Value	Notes
Probe head dimensions	500 µm (0.5 mm)	High-resolution probe tip enabling scans with step size.
Pulse rise time	< 2ns	Low overhead on the time factor (should be taken into consideration alongside the trigger jitter)
Pulse repetition frequency	0.1 Hz - 20 kHz	It determines how quickly consecutive pulses can be emitted by the probe. High-frequency results in faster and more accurate experiments.

Table 2: Technical characteristics of the ICI HH500-15 probe relevant to the use case of the equipment for our laboratory setup.

3.3 Measurement, Positioning, and Monitoring Equipment

Now that we have defined the hardware equipment responsible for simulating the fault injection attacks, it is time to break down the measurement and monitoring equipment required to observe the experiments and ensure their accuracy and safety. We will also use the same equipment during the calibration and testing phases of setting up the lab.

3.3.1 KEYSIGHT EXR254A Oscilloscope

One of the most critical devices in our lab setup is the Oscilloscope. It has an integral role in our lab setup since it allows us to perform debugging during the development phase of our framework to ensure that the rest of the equipment is working as expected. This includes the target and any external triggers we might utilize the pulse level, and timing in relation to the external trigger fed into the BPS202 station. To capture the EM pulses emitted by the probe we utilized a H-field (Magnetic Field) Probe. Moreover, with the use of the oscilloscope, we could potentially extend our framework to also include side-channel attacks or even a combination of both types of attacks.

3.3.2 Langer EMV - ICS 105 set IC Scanner 4-Axis Positioning System (3D Positioning Scanner)

Finally, the IC scanner, seen in Figure 3, is an integral tool of our lab setup. It is an advanced four-axis positioning system that enables us to perform precise IC scanning. By placing the target on its interface and securing the EM probe on it we can traverse the IC surface area using a small step and also repeat experiments with high spatial accuracy. The scanner interface size is (50 x 50 x 50) mm with a minimum step size of (10 x 10 x 10) µm. Its repetition precision is +/- 1 µm which should be sufficient for our use case. Table 3 showcases some of the most important characteristics of the device, which are relevant to our experimental setup.

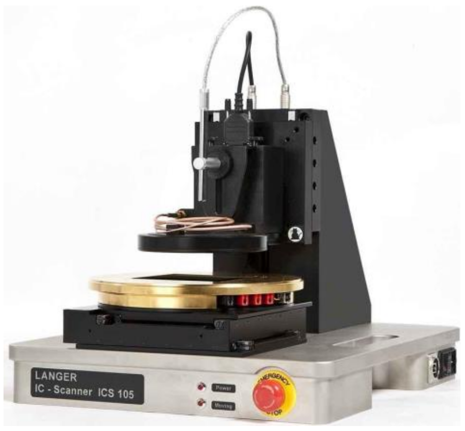

Figure 3: The Langer EMV - ICS 105 set IC Scanner 4-Axis Positioning System

Technical Specification	Value	Notes
Max. traverse range	(50 x 50 x 50) mm; α-Rotation ±180°	It also defines the surface area where we can place the target
Min. step size	(10 x 10 x 10) µm; α-Rotation 1	It defines the minimum step we can perform during our analysis
Positioning speed	(10 x 10 x 5) mm/s; α-Rotation 90°/s	The speed at which the scanner can move the plain. This must the taken into account as it might create potential bottlenecks during

		our assessment.
Repetition precision	+/- 1 µm (+/- 1°)	The precision the scanner can achieve when repeating a specific action. This is critical if we want to have high accuracy when performing tests.
Interface	Client / DLL (32 Bit / 64 Bit)	The scanner also contains a DLL that can be used as an API to enable further communication besides the provided software client.
Microscope Camera	N/A	The scanner also contains a camera that can be used to monitor the target device.

Table 3: Technical characteristics of the Langer EMV - ICS 105 set IC Scanner 4-Axis Positioning System relevant to the use case of the equipment for our laboratory setup.

3.3.3 Logitech C270 Web Camera

A simple web camera allowed remote monitoring of the lab equipment which extended the experiment hours as no physical presence would be required to perform the experiments. Simply connecting the camera to the PC we have set up the remote access and set up the camera in a place that provides a clear view of the equipment as well as all the indicator LEDs of the hardware equipment and the target device. We did not utilize the remote setup during the development phase of the framework to ensure that we would be able to react in a prompt manner should any issues arise. Once the framework was formalized remote access was used.

3.4 Target Embedded Device

3.4.1 ARM-based32-bit MCU

As our target device, we have selected an ARM-based 32-bit MCU. The choice of target was heavily influenced by the widespread adoption of ARM architecture-based MCUs and processors in the past years. ARM-based devices have a diverse array of applications and are commonly found in many embedded devices.

3.5 Lab Configuration

With all of the lab equipment defined above we can now expand on the Lab layout. This section will provide details on the configuration of the equipment and how they are all connected to enable seamless interactions and create a system that operates as a side-channel security evaluation framework. As previously outlined, in the description of each device, we will be using the following steps to set up the attack platform:

3.5.1 Configure Equipment

The first step is to configure each device individually and verify its functionality. To achieve that we will set up all of the equipment on the lab bench. Then we will connect each of the tools (Scanner, BPS202, Web camera) on the main PC, and using their clients we will verify that they function properly. An exception to this rule is the Keysight Oscilloscope which contains an Operating System (windows) on its own.

3.5.2 Target and Probe Positioning

The next step is to position and secure the target in place on the surface area of the scanner. The target must remain locked in the same place so we can accurately replicate experiments. The scanner's surface area is perforated allowing us to secure the target in place using the provided hooks. Once the target is in place and the probe is secured, we can measure the coordinates of the edges which will allow us to define the surface area the target MCU covers in relation to the scanner bed.

Once we have setup everything in the lab, we can utilize the provided GUI clients for each tool and perform some sample operations to ensure that everything is working properly. Figure 4 displays the lab equipment setup.

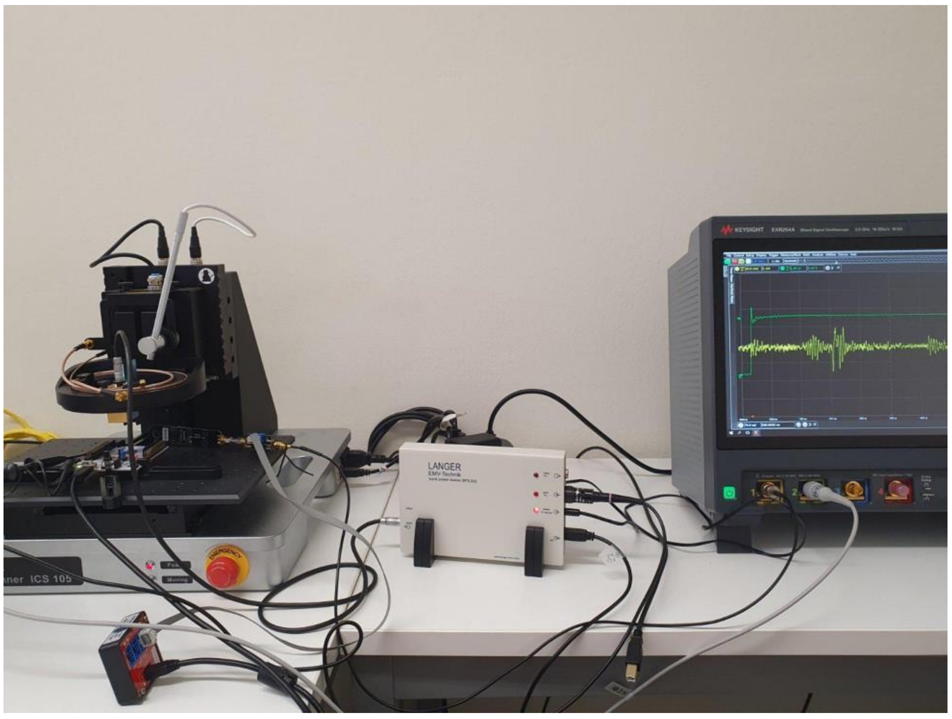

Figure 4: Lab equipment setup

4 Lab Automation

With the lab equipment set up and properly configured, our next step is to conduct a feasibility analysis to determine which components can be automated. This will not only enable remote access to the lab equipment but also optimize the process of conducting efficient and accurate experiments by removing the human factor for repetitive tasks. The following list contains all the possible automation we can perform:

- Scanner: Use the provided DLL API library to use code to automate the movement of the scanner
- EM Probe: Use the provided DLL API library of the BPS 202, the control device of the probe, to automate the configuration of the probe. Additionally, with the external trigger connected directly to the target device we can have control of the point in the program's execution time we will inject an EM pulse.
- Oscilloscope: Use the serial interface of the Keysight oscilloscope to automate the process of capturing and analyzing traces.
- Target Automation: Use the target's USB interface to automate the process of flashing firmware into the MCU. Additionally, we might need to automate the process of plugging and unplugging the target in case a hard reset of the target is required.

The rest of the section will be segmented into subsections, each providing additional information on the automation tasks mentioned above.

4.1 Scanner Automation

To automate the scanner's movement, we can use the provided API interface using the provided DLL library. The API does not contain any documentation, but the header file should be sufficient to understand all the functions and their use case. With a brief overview of the API's header file, we can categorize the provided functions into the following categories:

- **Initialize the scanner:** The functions used to initialize the connection to the scanner, calibrate it so it can accept commands, and finally close the serial connection to it.

- **Monitor the scanner's state:** This category encompasses functions that retrieve information about the state of the scanner, such as its version, position, and the range of movement it can facilitate.

- **Control the scanner:** This category includes functions dedicated to moving or rotating the scanner along its axes or to specific positions.

Additionally, we can utilize the Graphical User Interface tool to interact with the scanner and identify the proper way to control it. After some experimentation, it is evident that we must first calibrate the scanner, which will set all coordinates to an initial state before we can issue any successful commands. The diagram in Figure 5 explains the workflow.

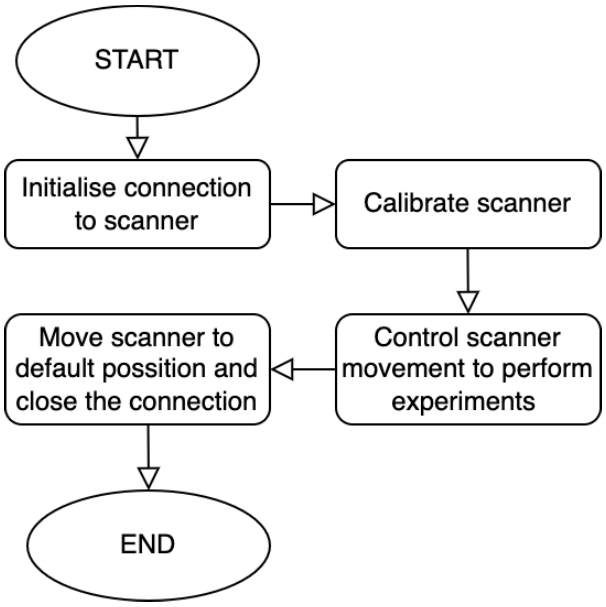
START
Initialise connection to scanner
Calibrate scanner
Control scanner movement to perform experiments
Move scanner to default possition and close the connection
END

Figure 5: Scanner automation flow chart

For the scope of this research, we are interested in controlling the scanner X, Y, and Z coordinates. The rotation of the probe head will be performed only once at the beginning of the experiments and will remain static for the rest of the duration.

The scanner's control functions can be separated into two categories depending on the type of movement. We can either move the scanner to an absolute position, by giving it specific coordinates, or we can move it relative to its current position. Finally, we will utilize the functions used to get information from the scanner to ensure that we will move the scanner within acceptable ranges by sanitizing the user input prior to executing any movements. Additionally, we can utilize these functions to verify the position of the scanner during experimentation.

We can now expand the flow chart provided above, specifically the “Perform Operations” block. The experiment process can be separated into two parts. The first step will be the movement of the scanner to a predefined point, given exact coordinates or a step relative to the current placement. The flowchart of Figure 6 illustrates this process.

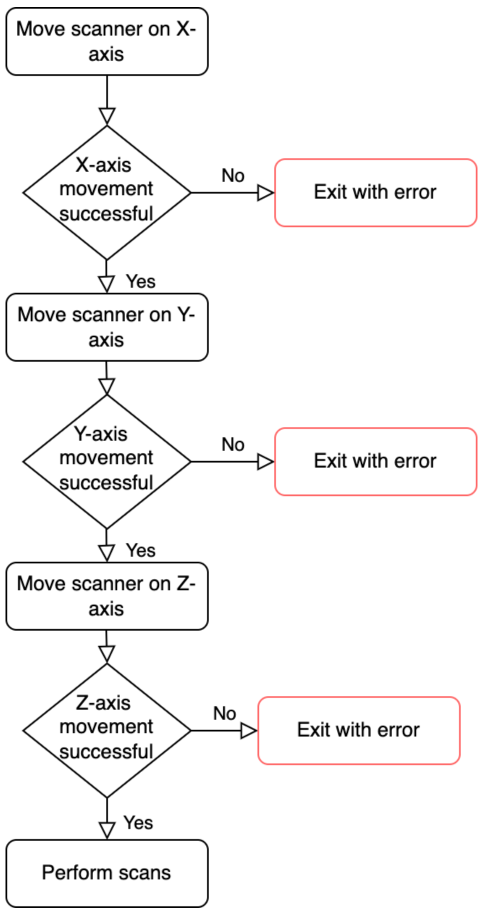

Figure 6: Scanner movement automation flow chart

The second step consists of an iterative process where we will continuously move the scanner body, and thus the position of the probe relative to the target's MCU. This will enable us to perform tests in different areas on the surface of the MCU. The flowchart of Figure 7 illustrates this process.

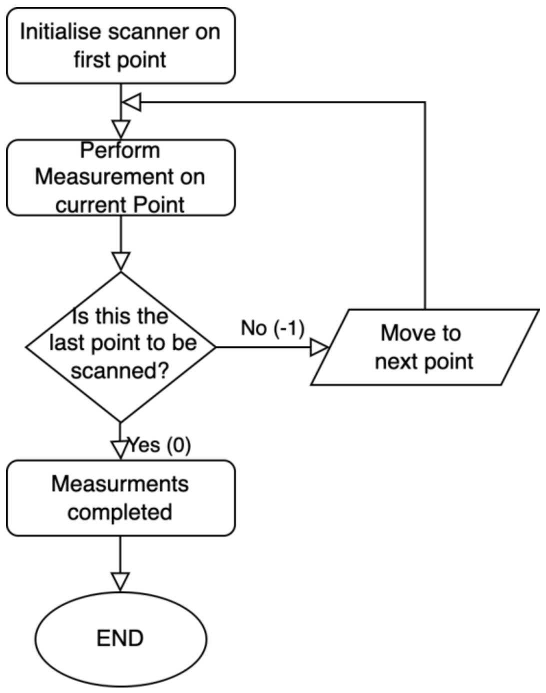
Initialise scanner on first point
Perform Measurement on current Point
Is this the last point to be scanned?
No (-1)
Move to next point
Yes (0)
Measurments completed
END

Figure 7: Flowchart of the scanner control process during the experiments.

We have now concluded the definition of the general workflow of the code that will enable us to automate the scanner's control. In a later stage of this thesis, we will expand more on the specific algorithm selected to traverse the surface of the target for our experiments.

4.2 Probe Automation

The probe controller, BP202, offers the same software capabilities as the scanner. It features a GUI that enables control of the probe and also includes an API accessible through a .dll library. For the scope of the project, we will require automation in the following areas:

- External Trigger: Configure an external trigger that will be supplied to BPS202.
- Pulse level: The voltage level of the EM pulse to be emitted.
- Pulse delay: The amount of time the probe will delay the pulse after the external trigger signal.

By utilizing various combinations of pulse levels and delays, we can test the resilience of the target under different conditions and at various stages of program execution. To ensure the probe is configured correctly, it is imperative to initialize it to the desired operational state prior to beginning the experiment. The flowchart of Figure 8 illustrates the initial code that we have created for our framework.

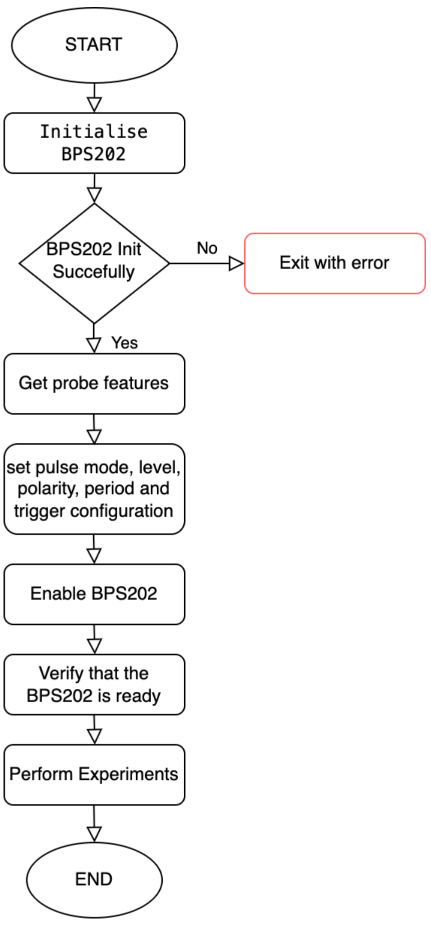

Figure 8: Initial flowchart of the BPS202 control process.

During the experimentation phase, we noticed that we also need to automate the transition between different experiments to facilitate a seamless operation, when testing for different set of search space parameters.

The flowchart of Figure 9 illustrates the final workflow utilized for the probe to ensure seamless operation with a set of different search space parameters.

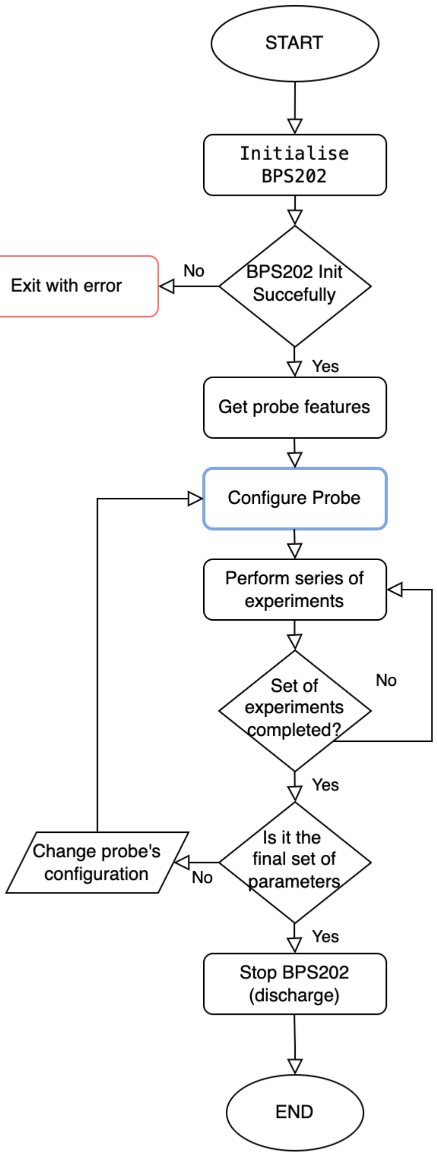

Figure 9: The final flowchart of the BPS202 control process.

4.3 Oscilloscope Automation

For the specific scope of our experiments automating the oscilloscope functions is not required. Configuring manually the oscilloscope one time to monitor the setup and configuration process as well as the experiments is sufficient.

4.4 Target Automation

For the target automation, the main procedure that needs to be automated is the loading of the firmware. Since the select board contains a programmer-on-board we can simply utilize a USB cable connected to our PC and upload the firmware via its serial interface. The same interface will also allow us to communicate with the target board to send and receive data to validate our experiments. The following flow chart outlines the procedure we followed during our experiments.

During the first phase of the experiment, we notice some additional needs for automation to streamline the process and eliminate the need for human input during each phase. By introducing faults via EM pulses into the target we observed the following responses, where some of them required interference before we could continue with the experiment sequence. Table 4 provides a list of the different reactions observed and the necessary mitigations for each for the target to return to a working state.

Response Categories	Details	Mitigation
Normal Operation	The target operated as expected	N/A (The result fell within the expected operational parameters.)
Erroneous Output	The target executed the loaded program, but the data returned were altered	N/A (The result fell within the expected operational parameters.)
System Hang (Firmware re-programming needed)	The target failed to respond. The firmware stops working.	Re-programming the firmware due to corruption of the image on the MCU
System Hang (Power cycle and re-programming needed)	The target responds with random data and executes and unknown subroutine.	Power cycle the board and then re-program the firmware. (Power cycling was mandatory, otherwise, the board after the normal firmware re-programming procedure would not operated as expected)

Table 4: Different responses (and the mitigation strategies, when needed) observed by the target during the experimentation process when subjected to EM pulses.

The mitigation of the System Halt response category only required the use of the established USB connection to resume normal operation. The Fallback Response category on the other hand required additional hardware to be added to the lab setup to automate the process of power cycling the board (a simple reset via the button or the board GPIOs did not have any effect). To achieve this we used the setup seen in Figure 10, which utilised an Arduino board and a modified USB-A male to USB-A female, that contains a MOSFET between the power (VCC) line. With that setup and a simple firmware loaded into the Arduino board, which is connected to our PC, we can send a command via the Arduino serial interface to control the MOSFET's gate and cut the power on the target board. This triggers a hard power reset of the target when necessary.

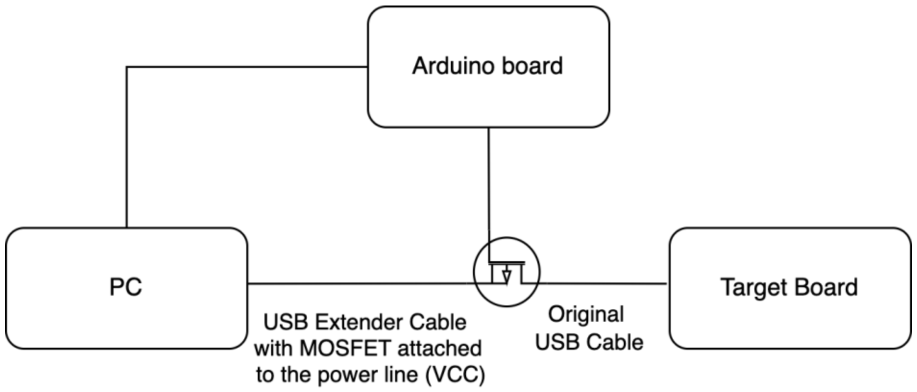
Arduino board
PC
Target Board
USB Extender Cable with MOSFET attached to the power line (VCC)
Original USB Cable

Figure 10: Diagram outlining power control on the target board using an additional Arduino board and a MOSFET

5 Case Study

In this case study, we examine the resilience of the targeted microcontroller unit (MCU) when subjected to electromagnetic (EM) pulses, or EM fault attacks. Given the increasing use of smart devices that rely on MCUs to perform critical applications, understanding their vulnerability to EM interference is of paramount importance. This analysis aids in defining a methodology and testing procedure to aid in the identification of potential weak points within the MCU's architecture that can be exploited via localized EM fault attacks. To achieve this, we will subject the MCU through a series of controlled lab experiments, to highlight areas prone to errors, when subjected to EM interference, providing a comprehensive overview of its EM susceptibility.

5.1 Methodology

5.1.1 Hypothesis

The hypothesis on which these experiments are based is that certain regions of the MCU, particularly those involved in memory storage and data processing, are more susceptible to EM interference and will potentially lead to higher error rates, that will lead to either data corruption or operational failures. By scanning the MCU surface area and conducting a series of deliberate fault injections, using an EM probe that generates high voltage pulses, we can create a heatmap that maps areas of interest in the surface of the MCU package.

5.1.2 Search Space

Several factors influence the likelihood of performing a successful EM fault injection attack. As a successful attack, we define it as any outcome that induces unexpected behaviour in the target MCU, as this is indicative of the area's sensitivity. The equipment in our laboratory setup allows us to control various parameters that impact the results of these experiments. Each parameter combination uniquely influences the outcome of each experiment. As a result, we must be mindful of how altering one parameter might affect the overall effectiveness of the attack. The list below provides a comprehensive overview of each parameter we considered important and the level of control we have over it.

- **Probe tip:** The size of the EM probe tip affects the level of detail we can achieve during our experiments. We cannot alter the tip size, which has a fixed size of 500 µm (0.5 mm).
- **Probe to target distance:** The distance of the probe's tip from the MCU package is also critical as it will affect the overall interest of the EM pulse reaching the MCU's internal components. Since the scanners offer control on the Z-axis, we can configure this parameter.
- **The angle of the probe:** The angle of the probe might also affect the outcome of the experiments. Our equipment does not allow us to have any control over this parameter. As a result, this will have a static perpendicular position.
- **The pulse's intensity:** This is a critical parameter that can greatly affect the experiment's outcome. It will be one of the main tuning parameters during our lab tests.
- **The pulse duration** is also a significant parameter. Our lab equipment does not allow the configuration of this parameter; thus, it will be considered static for the duration of the experiments.

- **Trigger delay:** The offset (delay) between the trigger and the EM pulse plays a significant role as this will affect the position in which the program is executed in the MCU will be hit.
- **Pulse polarity:** This is another significant factor in EM fault injection attacks, as it can substantially alter the outcomes by affecting how the EM pulse interacts with the MCU. Our equipment allows control of this variable but for our experiments, we will consider the polarity to be static. The selected polarity for all the experiments is positive.
- **Target code:** The code that the targets run, whether this is a cryptographic function, or a simple operation to test specific elements of the MCU.
- **Target Input:** Any input data given to the target MCU that will affect how different areas of the MCU are utilized during the analysis of these data. For example, in a cryptographic function, the plaintext input to the MCU determines which areas of the memory and the Arithmetic Logic Unit (ALU) are utilized during the mathematical operations performed for the encryption.
- **Environmental conditions**: The conditions of the target environment. Since we perform these experiments in a controlled environment, we can consider these as static throughout the duration of the experiments.

As illustrated by the extensive number of parameters and factors listed above, there are several elements to be considered in our experimental framework. Moreover, when viewing these factors as an interconnected system, where a change in one of the parameters can influence the rest, the complexity of the search space expands exponentially. As a result, an exhaustive search of the search space is impossible. To limit the search space, we selected only a few of the parameters we wanted to be dynamic during the experimentation phase. Table 5, based on the list above, provides an overview of the selection of parameters.

Search space parameter	Use case	Information
Probe tip	Static	Can not be changed
Probe position	Dynamic	We aim to scan the entire surface of the MCU
Probe to target distance	Static (After initial configuration)	We placed the probe as close to the MCU as possible, while also maintaining a sufficient offset to ensure safe operation.
Probe angle	Static	Can not be changed
Pulse intensity	Dynamic	The variation in the pulse intensity will enable us to test the level of sensitivity of the target.

Pulse duration	Static	Can not be changed
Pulse polarity	Static (positive)	We will not be changing the pulse's polarity as this would double the search space of our experiments.
Target code	Static (tinyAES)	The target will run a specific program throughout the experiments.
Target Input	Dynamic	To ensure uniform utilization of the target's internal components, we will employ randomly generated data sent to the MCU

Table 5: An overview of the search space parameters relevant to our experimentation process, detailing which were static and which were dynamic.

5.1.3 Testing Procedure

Experiments can be time-consuming, and many times yield no results. This is why we need to have a well-defined systematic methodology for approaching the target to perform our analysis. It is also crucial to gather comprehensive data during the experimentation phase to facilitate detailed analysis in subsequent stages. There are several methodologies to address the challenge of mapping the surface of an MCU concerning its sensitivity to EM emissions. Our main objective was to identify a time-efficient approach to mapping the surface of the target. Utilizing the automation scripts for the lab equipment we could easily control the parameters of the search space and create specific conditions that can be replicated.

Figure 11 presents a flowchart outlining our comprehensive testing methodology, focusing on the process performed for each scan. The flowchart demonstrates how the target interacts with the testing equipment through our software framework, developed to integrate both the target and testing equipment in a fully automated evaluation suite. Let's examine each step of the testing procedure and all the interactions between the target and the equipment.

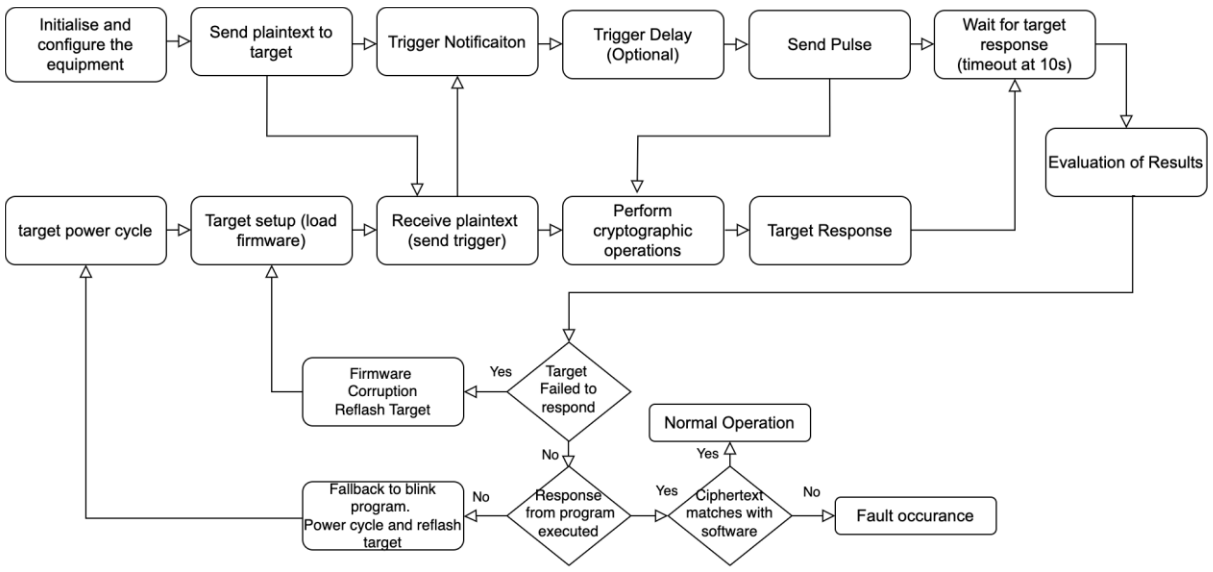
Initialise and configure the equipment
Send plaintext to target
Trigger Notificaiton
Trigger Delay (Optional)
Send Pulse
Wait for target response (timeout at 10s)
Evaluation of Results
target power cycle
Target setup (load firmware)
Receive plaintext (send trigger)
Perform cryptographic operations
Target Response
Firmware Corruption Reflash Target
Yes
Target Failed to respond
Normal Operation
No
Yes
Fallback to blink program. Power cycle and reflash target
No
Response from program executed
Yes
Ciphertext matches with software
No
Fault occurance

Figure 11: Flowchart of the proposed testing procedure for each scan.

Initialization and configuration: The first step involves the initialization and configuration of all the different tools like the scanner, EM probe, and its controller (BP202), including the target board. In this step, we also configure the search space parameters that we are going to use for each experiment, which is the most crucial step of the experiment. After we configure the search space and start the process the framework will begin the automated evaluation.

Perform encryption and send pulse: The next step involves a back-and-forth communication between our framework and the target. We initially send the plaintext, which is to be encrypted via AES, to the target. The target then responds with a trigger, at the desired point during the program execution, in our case the last round of AES. The trigger is fed into the BPS202 controller of the probe as an external trigger, which then initiates the trigger delay counter if the current scan requires a delay. Upon completion of the trigger delay a pulse will be sent via the probe directly into the surface of the MCU's package, during the execution of the encryption process.

Evaluation of target behaviour and Corrective Actions: The next logical block of our testing procedure involves the evaluation of the target behaviour. As we explained in the earlier section, under the target automation, the target exhibited three different behaviours when subjected to EM pulses. Depending on how severe the corruption is the framework will perform the necessary corrective actions, if needed, to recover the target to a working state before it proceeds to the next scan.

Log data for further analysis: The last step involves the logging of the generated data, including the current point under investigation, the search space parameters, and the target's response to the application of the pulse. This information will be used at a later stage to analyze the effectiveness of the EM pulses, showcase the effectiveness of our methodology, and outline potential shortcomings.

Now that we have defined the testing procedure for a single scan performed on a single point, it is time to expand on the rest of the methodology which involves the comprehensive scan of the entire surface of the MCU. Our objective is to map the surface area of the target MCU and identify which areas are more prone to EM attacks. To achieve that we need to define a way on how we will traverse throughout the surface of the MCU. Figure 12 illustrates the moving of the probe relative to the surface of the MCU using the snake pattern. This pattern ensures the full coverage of the surface area with minimal movements between the scan points, since it moves in a sequential order, thus enhancing the speed and efficiency while moving between points.

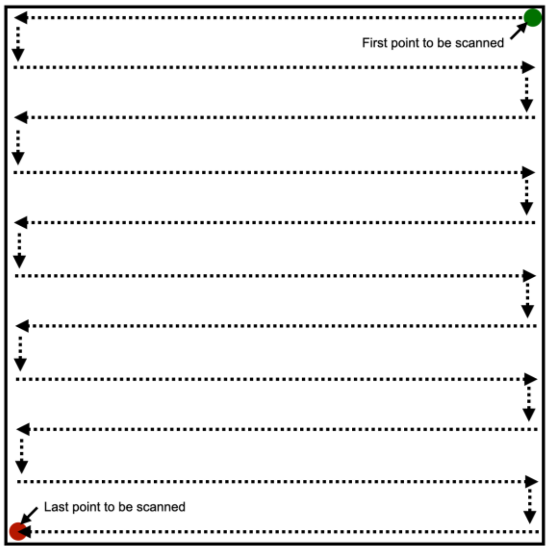

Figure 12: Snake pattern, used to scan the surface of the MCU. The green dot indicates the start point, while the red is the last point to be scanned. The arrows indicate the movement of the probe.

5.1.4 Testing Methodology

Having defined the testing procedure for each point and described how traversal per point is performed, we can present the overall testing methodology. Figure 13 depicts the testing procedure as it was defined and refined during the experimentation phase of this research. It is structured into 5, phases, as annotated in the flowchart. It is important to mention that phases two through four all utilize the core process defined in the first phase. The main difference is the adjustment of the search space parameters, based on the data evaluation of the subsequent faces. We have separated each stage into a different phase because they serve different purposes.

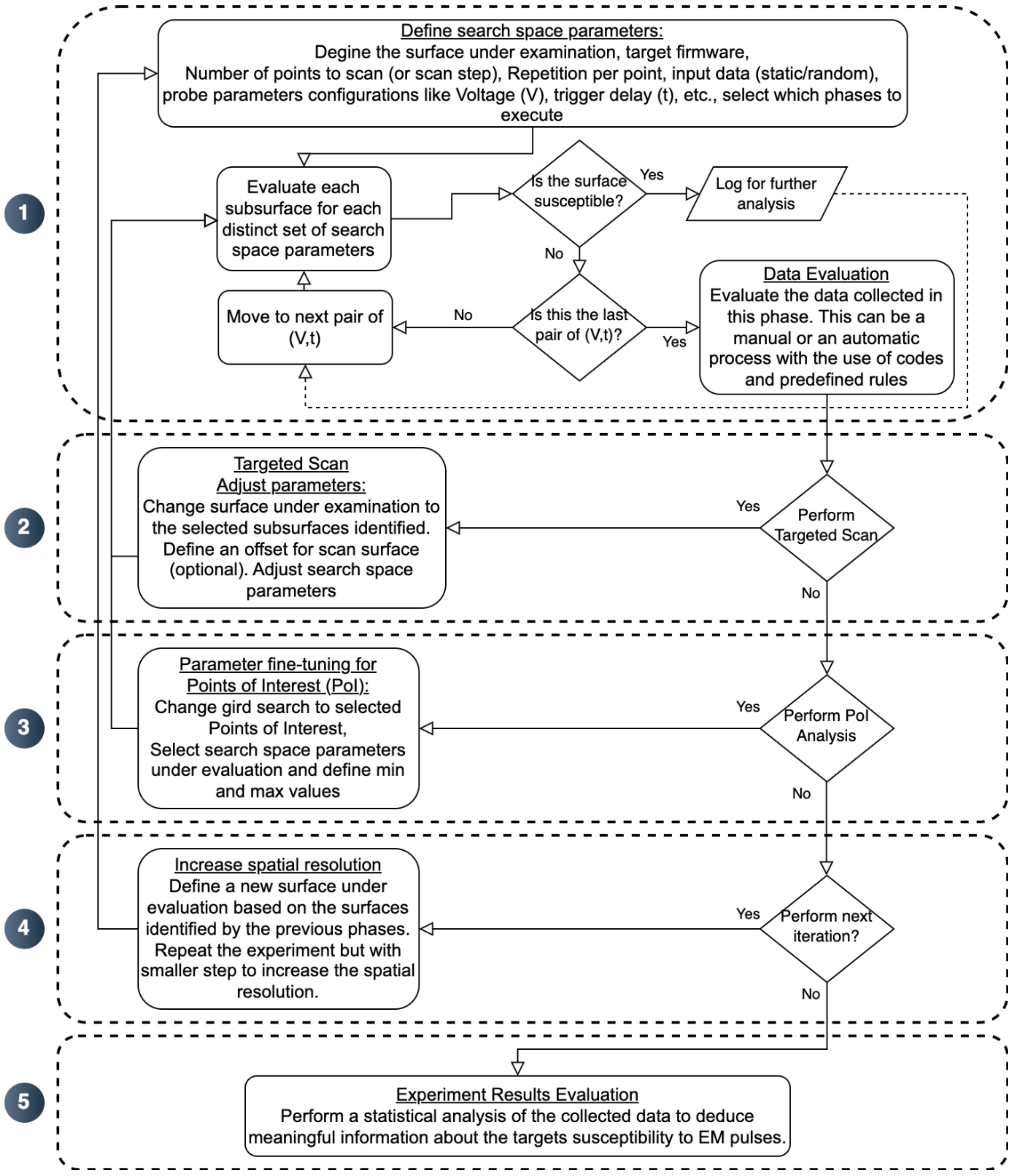
Define search space parameters:
Degine the surface under examination, target firmware, Number of points to scan (or scan step), Repetition per point, input data (static/random), probe parameters configurations like Voltage (V), trigger delay (t), etc., select which phases to execute
Evaluate each subsurface for each distinct set of search space parameters
Is the surface susceptible?
Yes
Log for further analysis
No
Move to next pair of (V,t)
No
Is this the last pair of (V,t)?
Yes
Data Evaluation
Evaluate the data collected in this phase. This can be a manual or an automatic process with the use of codes and predefined rules
1
Targeted Scan
Adjust parameters:
Change surface under examination to the selected subsurfaces identified. Define an offset for scan surface (optional). Adjust search space parameters
Yes
Perform Targeted Scan
No
2
Parameter fine-tuning for Points of Interest (PoI):
Change gird search to selected Points of Interest, Select search space parameters under evaluation and define min and max values
Yes
Perform PoI Analysis
No
3
Increase spatial resolution
Define a new surface under evaluation based on the surfaces identified by the previous phases. Repeat the experiment but with smaller step to increase the spatial resolution.
Yes
Perform next iteration?
No
4
Experiment Results Evaluation
Perform a statistical analysis of the collected data to deduce meaningful information about the targets susceptibility to EM pulses.
5

Figure 13: High-level flowchart of the testing procedure used during the experiments presented in this paper.

The **first phase** of our procedure consists of an initial scan of the entire surface of the target. The first step is the initial configuration of the search space parameters and other necessary configurations. This stage is paramount for the success of the experiment, as it sets the foundational criteria that will directly influence the scope and depth of our research. By carefully selecting and calibrating these parameters, we ensure that we cover a sufficient portion of the search area in a targeted manner.

- **Define the surface under investigation** The first parameter we must define is the surface area we want to analyze, in our case the MCU's package surface area. Our target MCU package is an LQFP package with dimensions of 9mm x 9mm. To incorporate that into the framework we first secure the board on the bed of the scanner and measure using the GUI to get the coordinates of the upper right corner. Then based on the package shape and dimension we can calculate the rest of the coordinates of the MCU edges and define the surface area under evaluation.
- **Target firmware:** The firmware running on the target is a crucial parameter that will affect the results of our experiments. For our experiments, we selected the AES algorithm to be examined. The specific version we selected was tinyAES [27], in which we performed slight modifications in order to add a trigger into the last round to eliminate the need to identify it with the use of an oscilloscope. This also ensured consistency among different iterations of our experiments.
- **Number of points to scan (scanner step):** It refers to the step we will use to move the board to reposition the static probe, relative to the MCU package. The step can be either provided by the user with an exact number or calculated dynamically based on the total number of points we want to segment the surface to. In our experiments we decided to separate the surface into a grid of 11x11, resulting in 121 distinct points evenly spaced. With an 11x11 grid, the calculated step corresponds to 0.9mm. The number of points was selected based on the search's time constraints to enable a full scan of the surface in a moderate amount of time.
- **Repetition per point:** The number of consecutive experiments to run using the same search space parameters to ensure we identify points with a low probability of generating faults. Even under ideal conditions, performing a successful EM fault injection attack may yield a low success rate. To ensure we properly identify points of interest, we must do multiple repetitions of the experiment for each different configuration. For the initial scan, given the large number of points and the time constraints we had, we limited the repetitions to 10 encryptions per point. After the initial identification of the points of interest, we can increase that number to 100, to collect more data to have a more comprehensive view on each point.
- **Input data:** The data sent to the MCU, in our case the plaintext to be encrypted by AES, also plays an important role as it affects the specific areas of the MCU being utilized during the encryption process. For the initial scan, we will limit the number of different plaintexts to match the number of unique repetitions, that will remain static among different repetitions.
- **Trigger:** For the scope of these experiments, we will use an external trigger to the probe provided by the target device. The trigger will be sent just before the final cycle of AES. Exploring the full program execution of AES would be inefficient, given the extensive time required to execute the algorithm. Furthermore, targeting the last round of AES the likelihood that a fault error will be propagated into the output ciphertext, which is usually the objective when targeting the algorithm with fault injection attacks. It is worth noting that in real-life scenarios, where access to the firmware may not be possible to add a

trigger, we could use the plaintext sent with an additional delay as our external trigger to target the last round of AES. With the use of an oscilloscope, it is easy to differentiate between each round of AES and estimate the required delay, even though this will not be quantity consistency among different executions of the experiment, as mentioned before.

- **Voltage level:** The voltage level of the pulse generated by the probe, will be set to 400V for the initial scan. This decision assumes that higher levels increase the likelihood of generating a fault, even though it might have led to the corruption of the target.
- **Trigger Delay:** As the program execution progresses different parts of the MCU are utilized. By utilizing different delay times, we effectively scan the program's execution over time. The delay step employed is 200ns, which allows for a comprehensive scan across the entire range of the last round of AES. Even though the smallest step possible with our equipment is 10ns, to perform enough repetitions and also test a relatively large area of the defined search space we are required to substantially increase the scan step in the time domain to scan the full range in a moderate amount of time.

After we configure the targets and search parameters we can begin the first phase of the testing procedure. During the execution of the phase, no other input is required by us, as the process is fully automated. The framework we designed can perform the experiments, logging the necessary data, and recovering the target back to a working state in case of systemic failure. At this stage, we consider each point as a square sub-surface with dimensions of 0.9mm x 0.9mm, corresponding to the step selected for the scanner. This means that if a point in a given sub-surface returns any type of fault, this area will be considered susceptible to faults and logged for further analysis. Once the initial scan of the MCU's entity surface is completed, the first phase is concluded, and we can move into the next phase. All the subsequent phases will include decision-making based on the data evaluation of the previous phases.

The **second phase** involves a targeted scan of the target. During this phase, we will exclude all the sub-surfaces that produced no faults and focus on the areas that appear to be susceptible. We can include an offset that will cover an area sourcing the areas of interest, which is optional. In our experiments we decided to also include the sounding areas, thus expanding the surface under examination around the identified regions. Finally, we can adjust any of the search space parameters based on the results of the data evaluation step of the previous phase. For the scope of our experiments, we decided to increase the number of repetitions from 10 to 100, as this scan covers only a portion of the original surface. This will allow us to gather more detailed information for the areas under evaluation, which will be substantial for statistical analysis. Once the parameterization of the search space is completed, we can repeat the testing procedure to gather new data about the target. Once all scans are complete, we can evaluate the collected data to obtain useful information for the next round.

The **third phase** of our testing methodology involved the parameter fine-tuning for the points of interest identified by the previous phase. The objective is to finetune the search space parameters for the points that exhibited sensitivity to EM pulses. Depending on the available time allocated for this phase we can either analyze all the points or select based on the metrics, from the data evaluation phase, for each point. Furthermore, fine-tuning all the search space parameters is very time-consuming, so careful reasoning is necessary to select the parameters that will have the greatest impact. For our experiments, we decided to focus on the points exhibiting many faults, regardless of their type, and examine different values for the voltage level. Our objective was to increase the total number of faults generated while also shifting their distribution to produce more meaningful faults.

The **fourth phase** aims to increase the spatial resolution of the identified sensitive areas. During the first two phases a large step to rapidly scan the entire surface of the target. A large step should be sufficient to identify the placement of the IC within the target's package, but it lacks enough detail to pinpoint specific regions within the MCU. To achieve this, we can repeat the entire procedure, or some phases of it, but this time defined as the initial target surface only the areas that exhibit sensitivity to faults. This in combination with a significantly smaller step will allow to obtain more detailed information as to which areas of the IC are vulnerable. This could also increase the efficiency of attacks by raising the total number of fault occurrences. Due to the limited time available to perform all the experiments, we were unable to complete this phase and gather substantial data for analysis, so it will be excluded from our experiment results.

The **fifth phase** is the final stage of the testing procedure and exclusively involves the analysis of the collected data. By performing an extensive statistical analysis of the collected data from all the phases, we can draw conclusions and create a clear understanding of the target. Depending on the approach this be a distinct phase at the end of the experiments or performed in parallel with each phase.

The proposed testing procedure is designed to allow for iterative cycles through each phase with adjustable parameters. Moreover, a nonlinear approach can be used, by selecting which phases to execute in each cycle. This enables us to configure the details and scope of the experiment, whether to identify large areas that are susceptible or to achieve detailed mapping of the target's surface. As a result, we can gather useful information about the target even when we have limited time to perform an analysis. If there are no constraints in the time, we can then perform a detailed analysis of the target which will result in more accurate and efficient results. In the next section, we will present the results of the experiments we conducted, focusing on the first three phases of the proposed methodology.

6 Experiment Results

In this section, we will present the results of the experiments we conducted in the scope of this thesis. Each subsection will cover each of the three phases of the proposed methodology, including information about the search space parameters used, a high-level overview of the number of scans performed, and the required time. Finally, each sub-section will contain a detailed analysis of the collected data using a series of graphs to provide a visual representation of the observations made.

6.1 First Scan: Full surface of the MCU

6.1.1 Introduction

The primary objective of this scan was to systematically map the entire microcontroller's surface to establish a basic understanding of its susceptibility to electromagnetic (EM) disturbances to the surface. To achieve that we used a predefined grid spanning 121 discrete points on the surface of the IC. To traverse through the points, we utilized the snake pattern described in the previous chapter, allowing us to scan efficiently the full surface. If a point yielded any meaningful results, then the sub-surface surrounding this point will be eligible for further analysis. Table 6 contains all the information regarding the search space variables used for this scan.

Search Space Parameters	Value	Type
Voltage Level	400 volts	Static
Trigger delay	130ns to 19530ns	Dynamic (200ns step)
Z-axis clearance	0.2mm	Static
X-axis step	0.9mm	Static (Defined based on the selected numbers of points on the grid)
Y-axis step	0.9mm	Static (Defined based on the selected numbers of points on the grid)
Plaintext (Target input)	Randomly generated	Dynamic (10 different plaintexts that were used for the repetitions of each scan using the same parameters)

Table 6: Search space parameters and their corresponding values used for this phase of the experiment.

The main search space parameter that was variable on this scan was the trigger delay, which contained 98 different values per point. Furthermore, we decided to repeat each experiment with the same configurations 10 times to increase the chances of identifying points with a low probability of yielding faults. In each of the 10 repetitions, a random plaintext was used, that was generated with the use of a random number generator. Upon completion of each set of repetitions, the seed of the random number generator was reinitialized into its original value, thus providing the same set of plaintexts for each scan configuration. Some initial metrics are shown in the following table:

Metric	Value
Total number of scans	118580
Total Scans per point	980
Repetitions per point	10
Total hours required (approximate, including only the scan time)	4
Average time per scan (approximate)	0.121 seconds/scan

Table 7: Metrics for the first phase of the experiments, detailing the total number of scans, scans per point, repetitions per point, and the estimated total hours required. The average time per scan is also provided.

It is worth noting that the average time per scan encompasses subprocesses required beyond the actual scan time, such as configuring the probe settings, moving the scanner bed, and most importantly, recovering the target when encountering type 2 or type 3 faults. Table 8 further divides the time required for one scan based on the fault type produced.

Fault Number	Faut Type	Average time per scan in seconds
0	No fault	0.05
1	Erroneous Output	0.05
2	System Hang (Firmware re-programming needed)	56
3	System Hang (Power cycle and re-programming needed)	75

Table 8: Average time, in seconds, required to perform one scan that resulted in a specific type of fault, as observed in the first phase of the experiment.

Out of all the unique scans performed only 139 produced meaningful results. The table below categorizes each fault based on the three types described earlier in this research.

Fault Number	Faut Type	Number of faults
1	Erroneous Output	24
2	System Hang (Firmware re-programming needed)	2
3	System Hang (Power cycle and re-programming needed)	113

Table 9: The number of unique faults observed in the first phase of the experiments categorized by type.

6.1.2 Presentation of Data and Analysis

To understand the information collected, we will utilize a series of charts and diagrams to help us visualize the observations made during the analysis of the data. Starting with the scatter plot shown in Figure 14, we can observe the points that produced a fault and their corresponding areas to the surface of the MCU. The black square corresponds to the outline of the MCU surface, with physical dimensions of 9mm for each side, whereas the smaller blue and red squares (sides of 0.9mm) correspond to the subsurface of each of the 121 points used for the scan and are defined based on the X-axis and Y-axis steps. The Y and X coordinates are irrelevant to our analysis, as they are directly correlated to the surface of the scanner and are only used to facilitate the mapping of each point into the surface of the MCU.

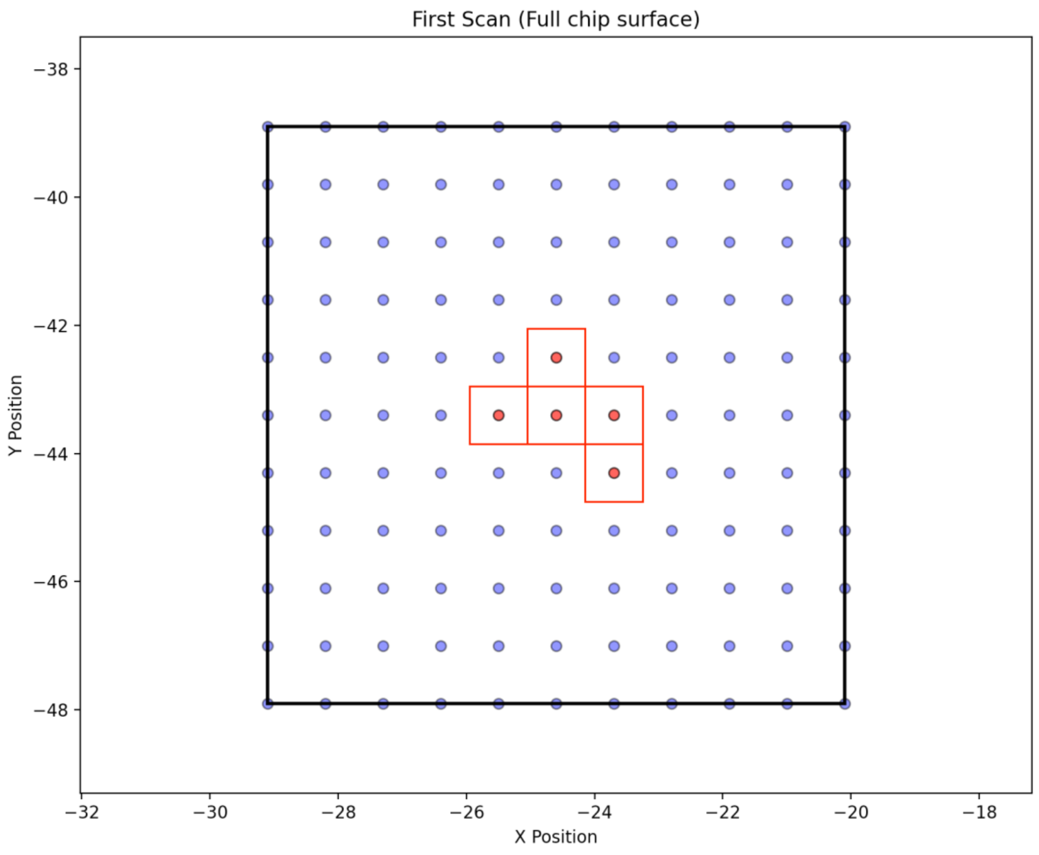

Figure 14: Scatter plot of scan points in relation to the MCUs surface (outlined in black). The red points correspond to points that yielded faults, whereas the blue points did not produce any result.

All of the observed faults were concentrated in the center of the MCU surface and were annotated with the use of the red points. Out of the 121 points scanned only 5 produced any meaningful findings. To help us visualize the sensitivity of each point, we will use a heatmap. As seen in Figure 15 we have annotated each point with the number of faults produced in it.

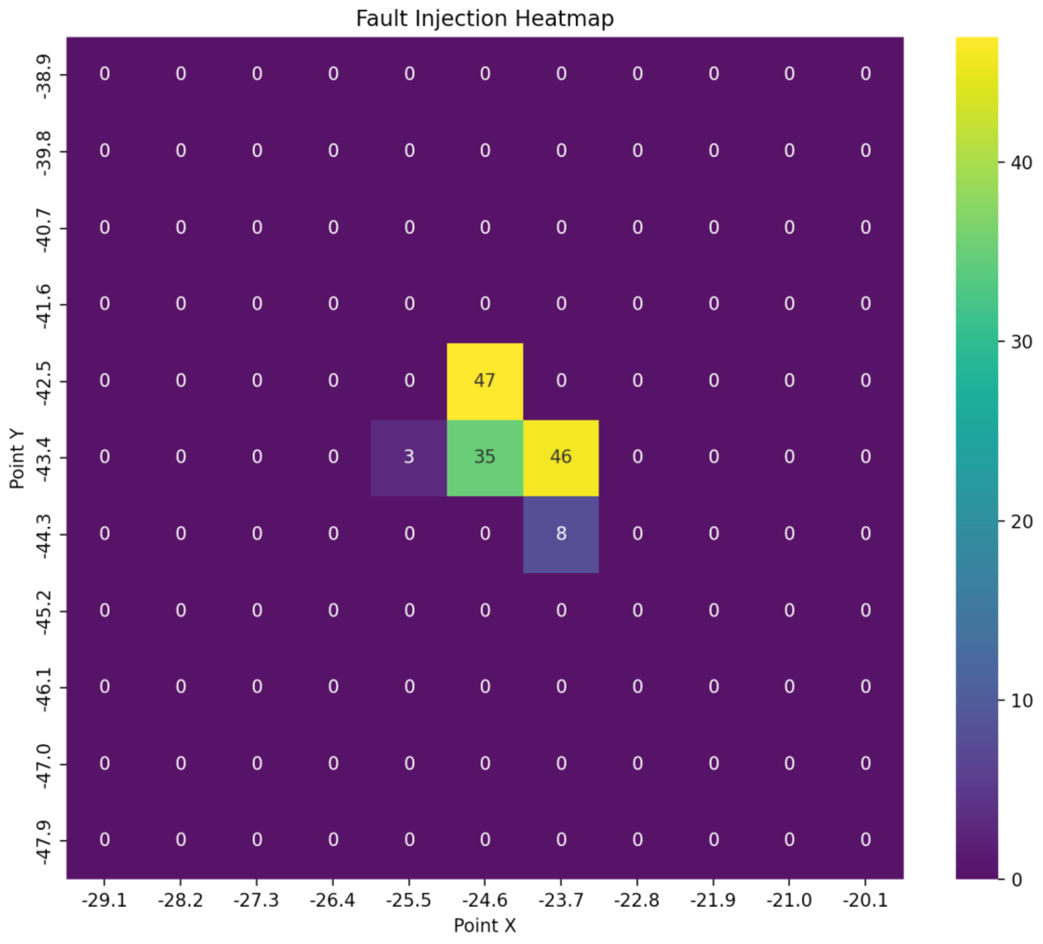

Figure 15: Heatmap of the points that exhibited anomalies when subjected to EM pulses and their corresponding fault rates

The number of faults produced is relatively small, compared to the total scans performed for each point. This is likely attributable to the insufficient configuration of the search space parameters. The highest percentage is 4.795% as seen in the following table:

No.	Point_X	Point_Y	Faults Count	Scans Count	Faults Percentage
0	-25.5	-43.4	3	980	0.306122
1	-24.6	-43.4	35	980	3.571429
2	**-24.6**	**-42.5**	**47**	**980**	**4.795918**

3	-23.7	-44.3	8	980	0.816327
4	-23.7	-43.4	46	980	4.693878

Table 10: Number of faults, total scans per point that exhibited sensitivity, and the fault occurrence percentage relative to the total scans.

When categorizing the faults by type, it is evident that most lead to severe outcomes such as breaking the program being executed or bricking the device. These types of faults do not produce any meaningful information that can be utilized to perform any attack in cryptographic algorithms like AES. Figure 16 illustrates a bar chart with the different types of faults generated per point.

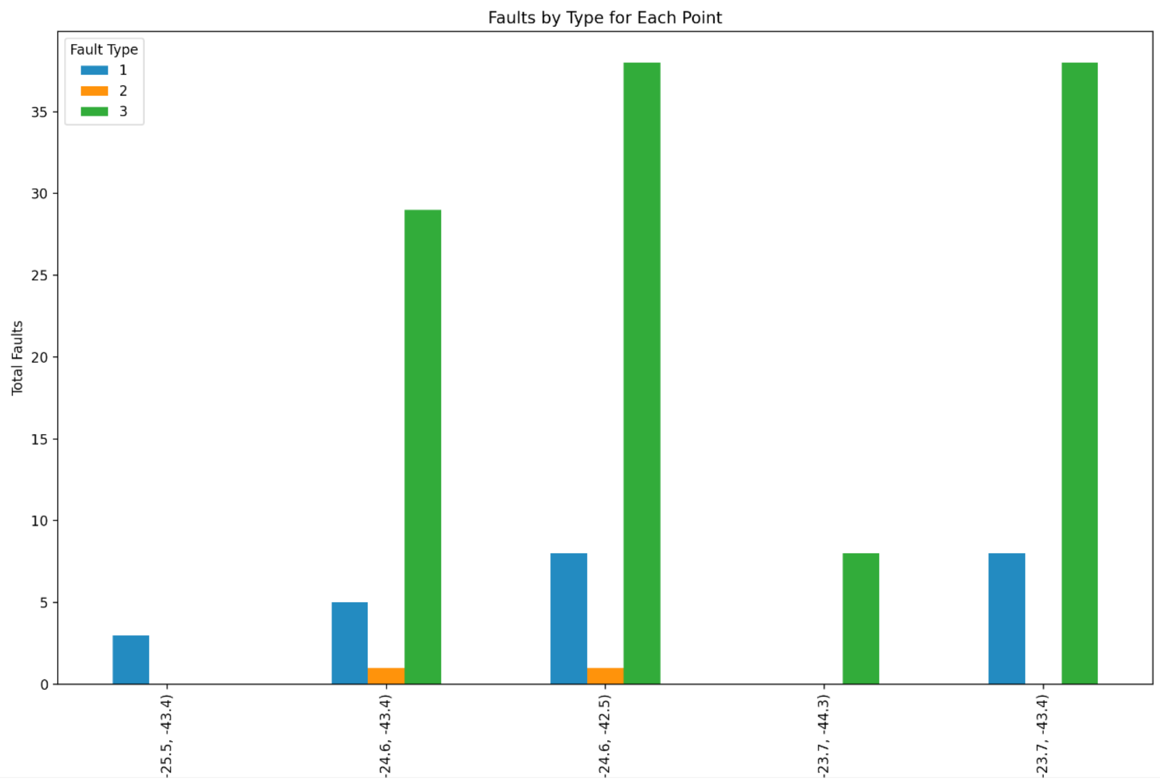

Figure 16: Bar chart of the points that return faults and their corresponding fault rates per fault type

The following observation can be made by analyzing Figure 16:

1. The point (-25.5, -43.4), even though it has a low number of faults generated showcases a 100% probability of generating type 1 faults, which is the objective of our research.
2. The rest of the points exhibit a high probability of generating type 3 faults which result in the corruption of the MCU firmware. These errors require corrective actions to revert the target back to its original state which dramatically increases the time complexity of our research and a potential attack, thus reducing its efficiency.

The exact numbers can be seen in the table below, with the above observations highlighted in bold text.

Points (X, Y)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-25.5, -43.4	**3**	**0**	**0**
-24.6, -43.4	5	1	**29**
-24.6, -42.5	8	1	**38**
-23.7, -44.3	0	0	**8**
-24.6, -42.5	8	1	**38**

Table 11: Number of faults by type for the points that exhibited sensitivity, with areas highlighted in bold as discussed in the text.

In Figure 17 we can observe the distribution of each fault type across each point, with the Y-axis displaying the percentages of each fault type. This visualization allows us to easily compare the presence of each type on each point and identify any potential patterns. As we move in the subsequent stages of our experiments, we will reference this chart to compare the distribution among the different phases.

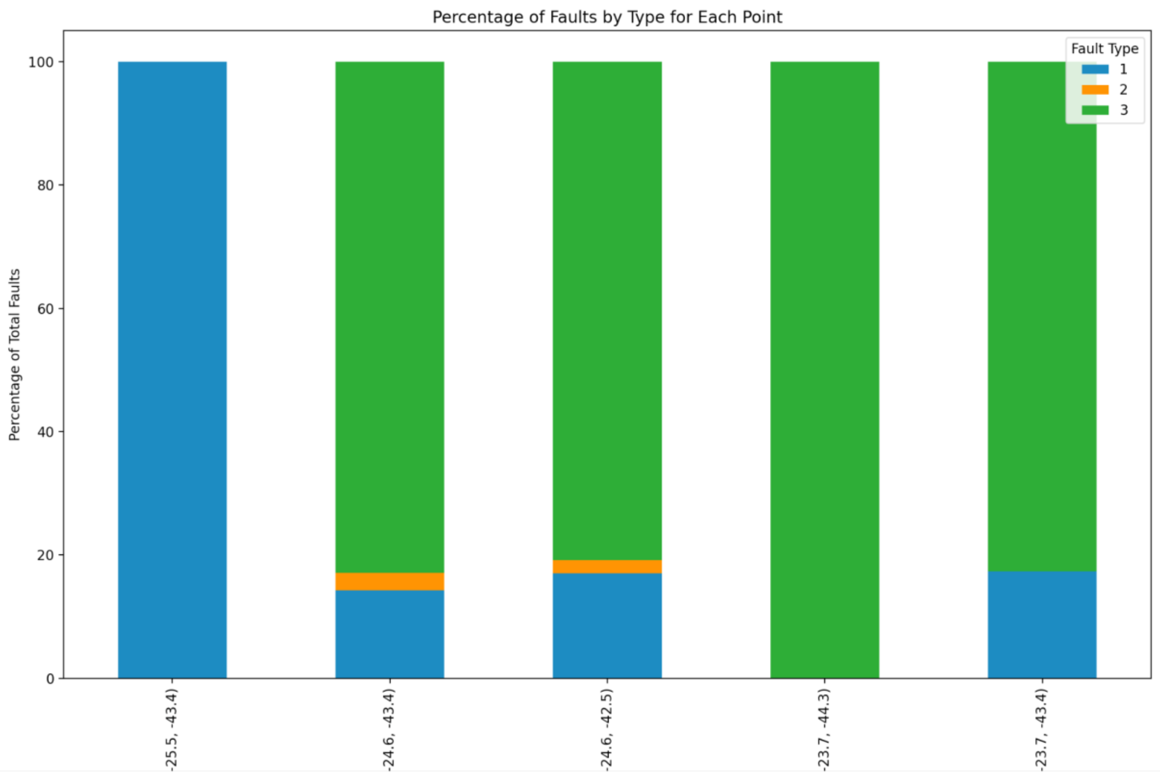

Figure 17: Bar chart displaying the points that returned faults, with the Y-axis representing the percentage of each fault type.

Table 12 shows the exact numbers, of the distribution of faults per point. The highest percentage per point is highlighted in bold.

Points (X, Y)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-25.5, -43.4	**100%**	0%	0%
-24.6, -43.4	14.286%	2.857%	**82.857%**
-24.6, -42.5	17.021%	2.127%	**80.851%**
-23.7, -44.3	0%	0%	**100%**
-23.7, -43.4	17.391%	0%	**82.608%**

Table 12: Distribution of type of faults per point, with the highest percentage per point highlighted in bold.

6.1.3 Conclusion of Initial Findings

It is worth noting that performing only 10 scans per search space configuration is a small number to gather substantial statistical information per point. Additionally having only a high voltage level of 400v might have been a critical factor that lead to faults with inconclusive results. These considerations were acknowledged before the initiation of the first phase of the experiments. However, the time required to perform scans with a higher repetition per point, like 100 or 1000 scans, would increase the time required substantially. With the time consideration in mind, we decided to analyze a small portion of the search space and use a high voltage level to increase the probability of creating any type of faults, to identify points of interest. Although we did not produce meaningful faults with a high probability the faults generated are indicative of the sensitivity of the sub-surface where they occurred. Further investigation of these areas will help us fine-tune the search parameters to enable us to generate meaningful faults with higher probability.

6.2 Targeted scan

Due to the low number of repetitions performed in the first scan, we decided to extend the search area to include the 5 points that produced faults and their surrounding points to account for any potential points that we missed on the first scan. This is predicated on the assumption that the MCU's sensitive components are concentrated in the central region of the package. These findings effectively narrow down the attack surface of the MCU to 25 points from the total 121 points of the initial scan. This translates to an approximate 79.3% reduction in the attack surface area.

This reduction in the attack surface will enable us to perform scans with higher repetitions per point to retrieve more information about the points of interest identified so far, and potentially identify more points with low probability that went undetected on the initial scan. The table below contains the statistics for this scan in terms of several scans and the total time required, essentially showcasing that we increased the total number of scans by approximately a factor of 2 while focusing on only 20% of the initial surface area. As in the first scan, each set of repetitions consists of a set of randomly generated plaintexts (100 plaintexts matching the number of repetitions), which remains the same among repetitions.

Metric	Value
Total number of fault injections	245000
Total fault injections per point	9800
Repetitions per point	100
Total hours required (approximate)	8
Average time per scan (approximate)	0.12 seconds/scan

Table 13: Metrics for the second phase of the experiments, detailing the total number of scans, scans per point, repetitions per point, and the estimated total hours required. The average time per scan is also provided.

Out of all the unique scans performed only 1286 generated meaningful results. The table below categorizes each fault based on the three types described earlier in this research.

Fault Number	Faut Type	Number of faults
1	Erroneous Output	224
2	System Hang (Firmware re-programming needed)	21
3	System Hang (Power cycle and re-programming needed)	1041

Table 14: The number of unique faults observed in the second phase (targeted scan) of the experiments categorized by type.

6.2.1 Presentation of Data and Analysis

After completing the second phase of the testing procedure, we identified another point that initially evaded our detection. Figure 18 showcases all the points that yielded a fault response in the second scan, following a similar layout as the scatter plot from the previous section.

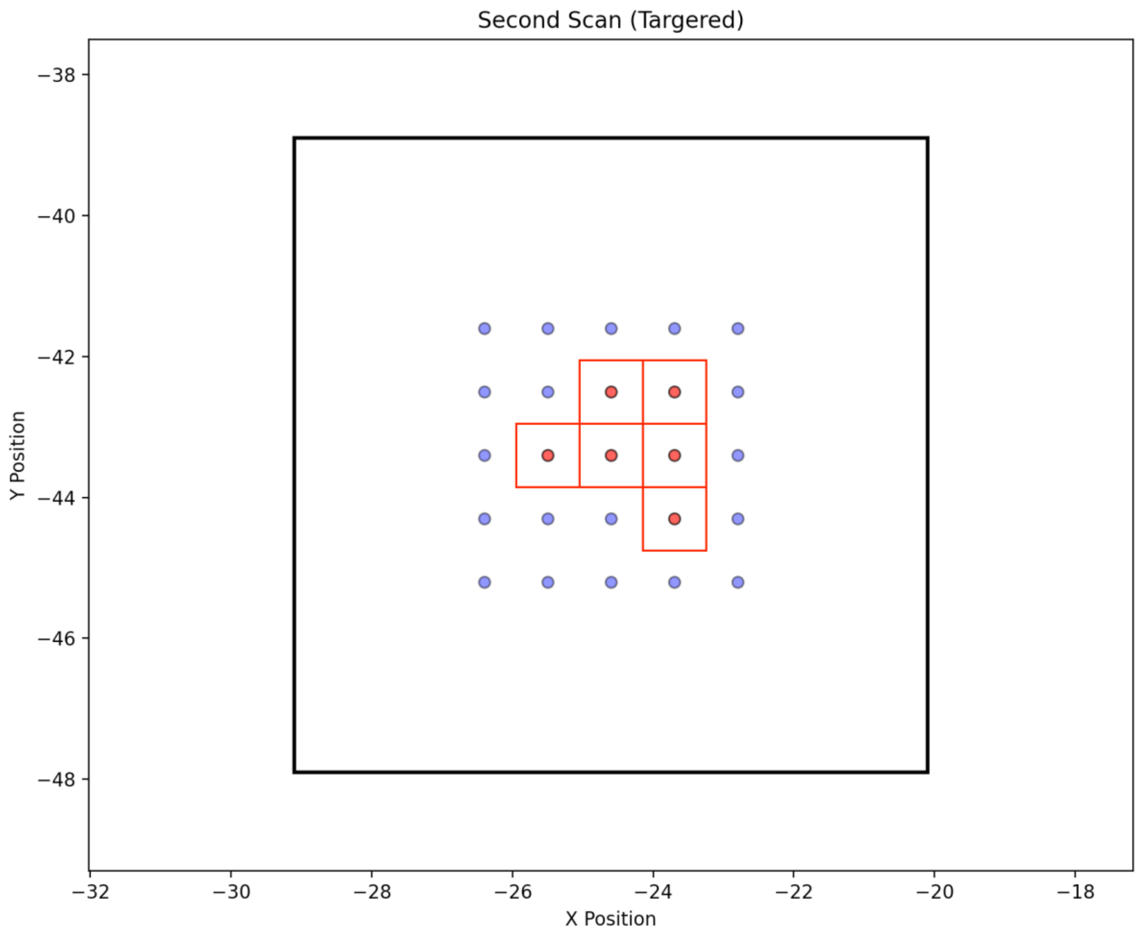

Figure 18: Scatter plot of the points analyzed in the second scan about the MCUs surface area (outlined in black). The red regions correspond to points that yielded faults, whereas the blue areas did not produce any result.

Following the same approach for visualizing the data Figure 19 showcases a heatmap of the points. We can observe that the newly identified point has a lower number of points compared to the rest.

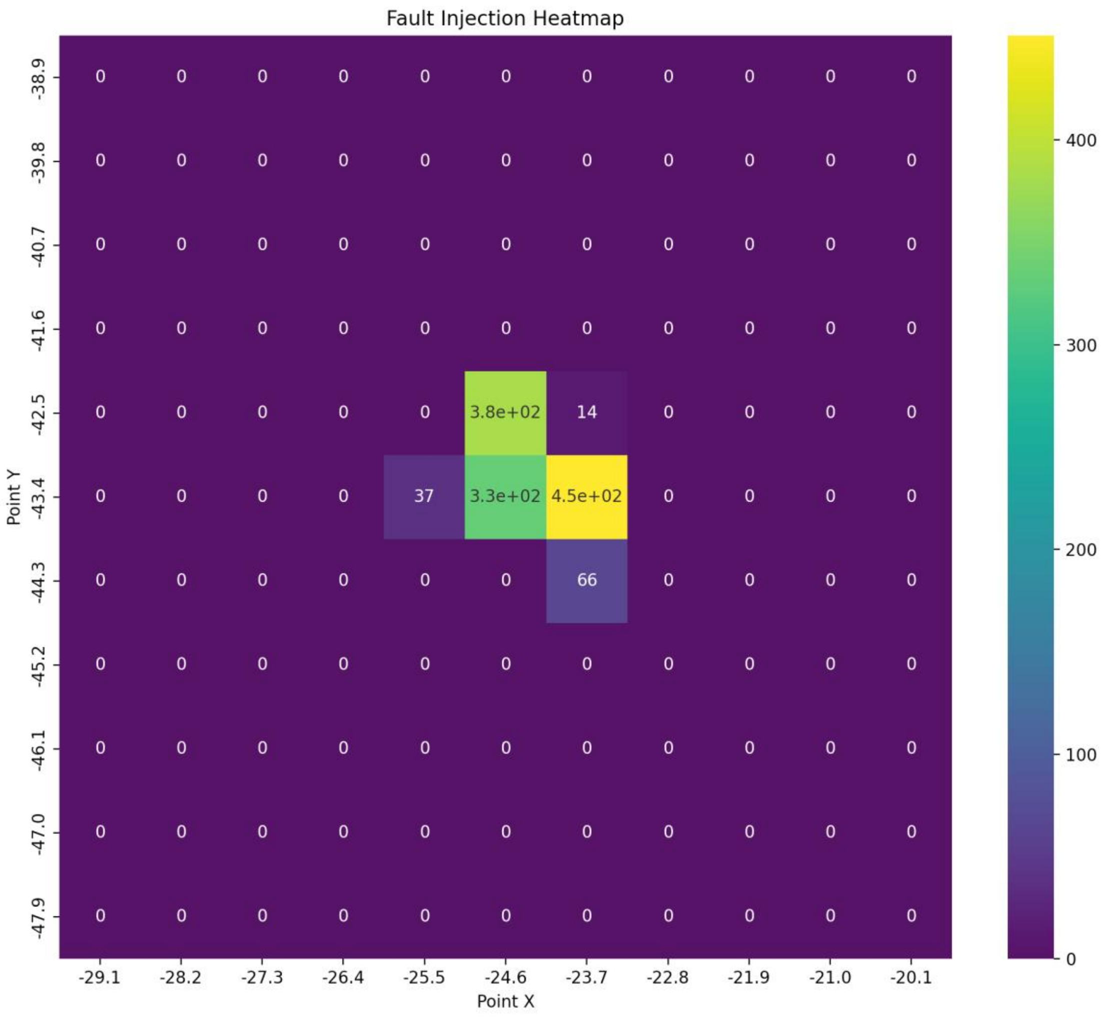

Figure 19: Heatmap of the points that produced faults and their corresponding fault rates

If we calculate the percentage of fault appearance per point, we will observe a similar distribution to the first scan. The exact percentage per point has slight deviations, with the highest one being around 23.34% increase in faults, on point (-25.5, -43.4). Conversely, the points (-24.6, -44.3) displayed the highest decline at -18.32%. The new point exhibited only 0.142% faults, which is indicative of why we did not successfully identify the point during our first scan. On the first scan we performed 10 scans per trigger delay totalling up to a total of 980 scans, a fault percentage of 0.142% corresponds to approximately 1.4 faults in this case.

No.	Point_X	Point_Y	Faults Count	Scans Count	Faults Percentag e (%)	Difference from the

						first scan (%)
0	-25.5	-43.4	37	9800	0.377551	**23.34%**
1	-24.6	-43.4	334	9800	3.4081632	-4.57%
2	-24.6	-42.5	384	9800	3.918367	**-18.32%**
3	-23.7	-44.3	66	9800	0.673469	-17.50%
4	-23.7	-43.4	451	9800	4.6020408	-1.96%
5	-23.7	-42.5	14	9800	0.142857	N/A

Table 15: Number of faults, total scans per point that exhibited sensitivity, and the fault occurrence percentage relative to the total scans, including the difference in percentage from the first scan.

In Figure 20 we can see that the distribution per fault type remains at similar levels as with the initial scan, which is to be expected since we only increased the number of reparations and did not change any of the other search parameters.

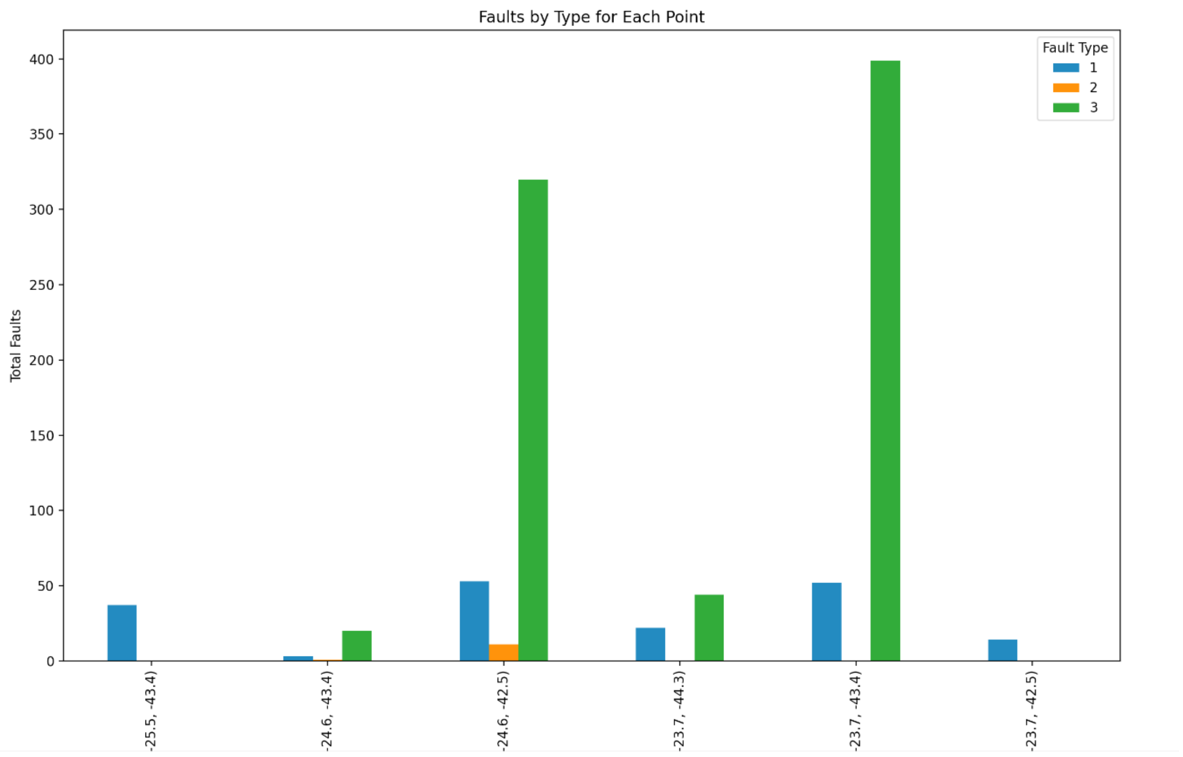

Figure 20: Bar chart of the points that return faults and their corresponding fault rates per fault type

The only noticeable difference is at point (-23.7, -44.3), which exhibits a low number of type 1 faults, whereas the first scan exclusively contained type 3 faults. Table 16 contains the exact numbers for each type of fault per point, displayed in Figure 20.

Points (X, Y)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-25.5, -43.4	37	0	0
-24.6, -43.4	3	1	20
-24.6, -42.5	53	11	320
-23.7, -44.3	22	0	44
-23.7, -43.4	52	0	399
-23.7, 42.5	14	0	0

Table 16: Number of faults by type for the points that exhibited sensitivity.

If we plot the types of faults per point with the Y-axis representing the total percentage, as seen in Figure 21, it is evident that point (-23.7, -44.3) exhibits significant change compared to the first scan, with one-third of the generated faults being type 1.

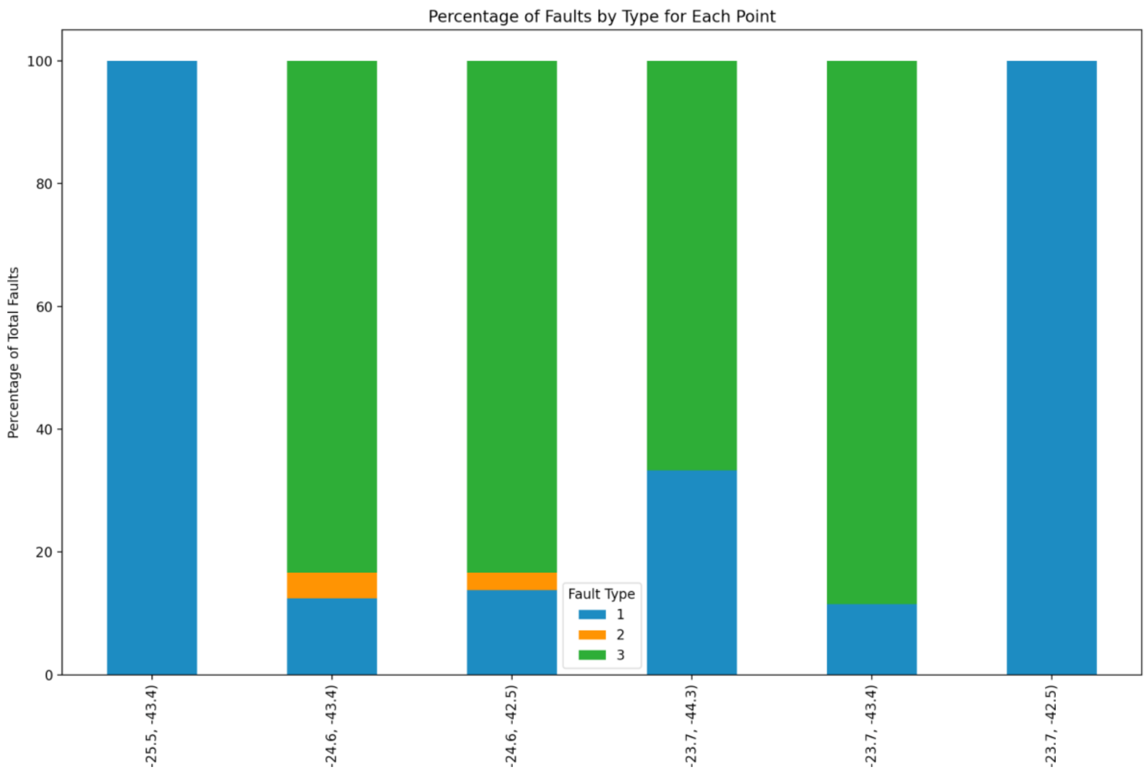

Figure 21: Bar chart of the points that return faults and their corresponding fault rates per fault type with the Y-axis representing the percentage of the occurrences.

The exact numbers, of the distribution of faults per point, can be seen in table 17. The areas that exhibit significant differences compared to the first scan are highlighted in bold. It is visible that there are some shifts in the distribution of faults but the prominent fault types remain the same.

Points (X, Y)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-25.5, -43.4	100%	0%	0%
-24.6, -43.4	12.500%	4.166%	83.333%
-24.6, -42.5	13.802%	2.864%	83.333%
-23.7, -44.3	**33.333%**	**0%**	**66.666%**
-23.7, -43.4	11.529%	0%	88.470%

-23.7, 42.5	**100%**	**0%**	**0%**

Table 17: Shift in the distribution of faults per point, expressed as a percentage, compared to the first scan. Areas showing significant differences from the first scan are highlighted in bold.

6.2.2 Conclusion of targeted scan

The target scan showcased the importance of having a higher repetition of each condition tested to identify points that yield low percentages of faults. Moreover, with more repetitions, we can get more in-depth results to analyze each point. This also outlines the need for a broad search space since each point might be susceptible to different types of emissions, like higher or lower voltage levels.

6.3 Point of interest sensitivity analysis

After identifying the points of interest on the surface of the MCU we continued with performing a more in-depth analysis by conducting a series of scans with an expanded search space. One of the driving factors for producing faults is the voltage level that the EM pulse is producing. On the first two scans, we used a relatively large value of 400 volts to ensure that we would yield results even in areas with low susceptibility.

After we identified a few points of interest we expanded the search space in terms of the voltage levels tested. We decided to expand the space also to include smaller voltage levels, as can be seen in the table below, outlining this scan's search space. We started with a very low voltage, 100 volts, and a very high step of 50 volts. Once the target exhibited sensitivity, we then changed the search space and utilized a smaller step of 10 volts to closely scan the range that exhibited sensitivity to EM pulses.

Due to the high time complexity, we were not able to analyze all of the 5 identified points. We decided to further explore the points that produced a high number of faults, with a higher probability of type 3 faults, and attempt to increase the generation of meaningful, type 1, faults. Table 18 shows the search space parameters used for this scan.

Search Space Details	Value	Type
Voltage Level	100v, 150v, 200v + The range from 250v to 360v with a step of 10v.	Dynamic
Trigger delay	130ns to 19530ns	Dynamic (200ns step)
Z-axis clearance	0.2mm	Static
Points of Interest (X, Y)	-24.6, -43.4 24.6, -42.5 -23.7, -43.4	Static (3 predefined points)

Plaintext (Target input)	Randomly generated	Dynamic (100 different plaintexts that were used for the repetitions of each scan using the same parameters)

Table 18: The search space parameters used for the third phase of the experiments.

Table 19 presents some general information regarding this phase of the experiments.

Metric	Value
Total number of fault injections	441000
Total fault injectionsper point	9800
Repetitions per point	100
Total hours required (approximate)	15
Average time per scan (approximate)	0.12 seconds/scan

Table 19: Metrics for the third phase of the experiments, detailing the total number of scans, scans per point, repetitions per point, and the estimated total hours required. The average time per scan is also provided.

The total number of faults produced was 1596, which can be further categorized into one of the three types as shown in Table 20.

Fault NumbTer	Faut Type	Number of faults
1	Erroneous Output	1208
2	System Hang (Firmware re-programming needed)	14
3	System Hang (Power cycle and re-programming needed)	374

Table 20: The number of unique faults observed in the third phase of the experiments categorized by type.

The distribution of faults appears to have shifted, with a higher number now being observed in type 1 faults. By further analyzing and visualizing the data we will be able to understand what the driving factor to this change was. Beginning with a heatmap outline the number of faults on each of the 3 points as seen in Figure 22.

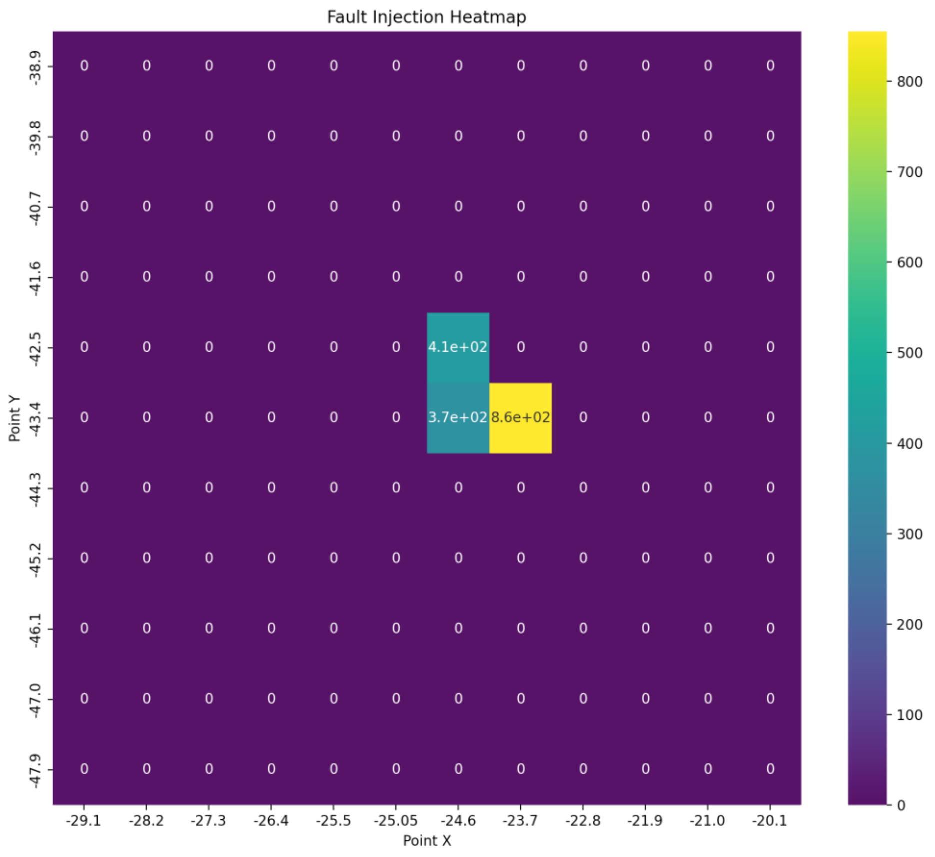

Figure 22: Heatmap of the points that produced faults and their corresponding fault rates

The distribution of the total number of faults per point is similar when compared to the other scans. For this phase of our experiment, we did not focus on the percentage of the fault occurrences in contrast to the total number of scans performed, as the scope was to fine-tune the search space parameter of the voltage level, which is expected to result in a lot of scans that did not produce any meaningful results. When plotting the different types of faults per point on the bar chart in Figure 23 it is evident that there is a significant shift in the distribution of faults.

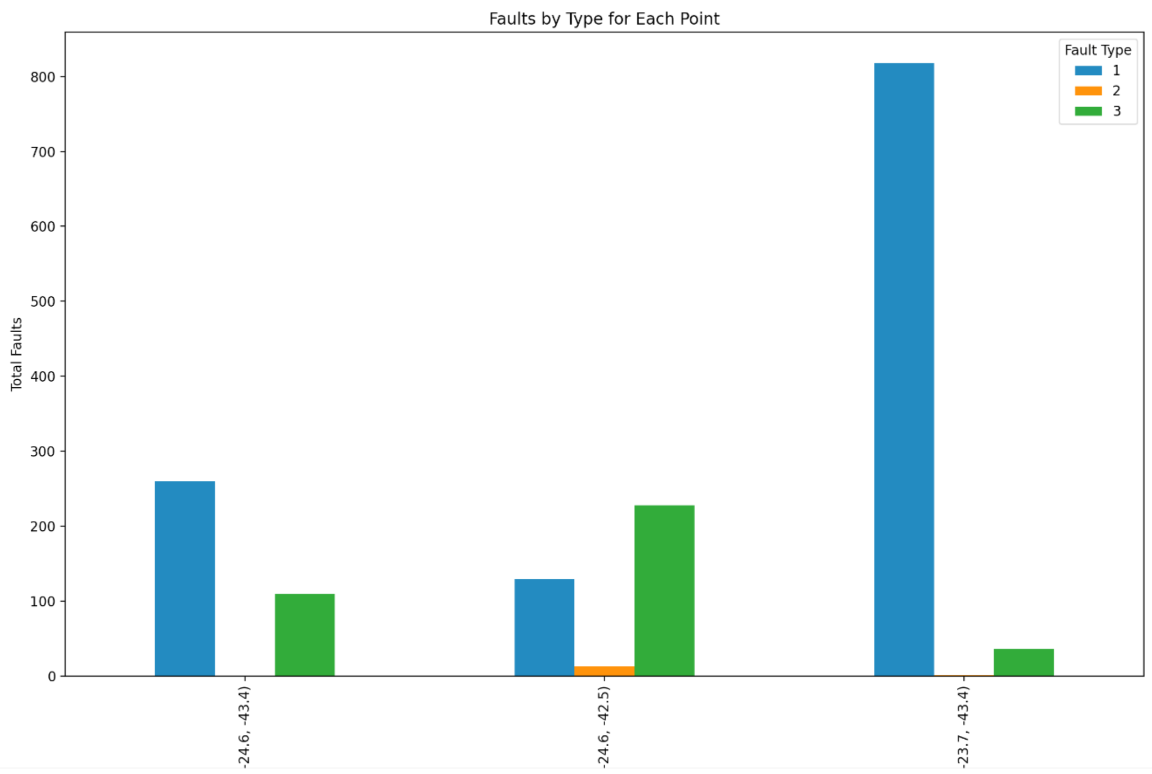

Figure 23**. Bar chart that shows the number of faults per type for each point of interest.**

In two out of the three points, the majority of faults are now type 1 faults. Point (-24.6, -42.5) exhibits an increase in the number of type 1 faults as well. However, most of the generated faults are still type 3. Table 21 displayed the exact number of faults based on their type for Figure 23.

	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-24.6, -43.4	**260**	0	110
-24.6, -42.5	130	13	228
-23.7, -43.4	**818**	1	36

Table 21: Number of faults by type for the points that exhibited sensitivity, with areas highlighted in bold as discussed in the text.

In Figure 24 we can observe the percentage of each type of fault for the three points.

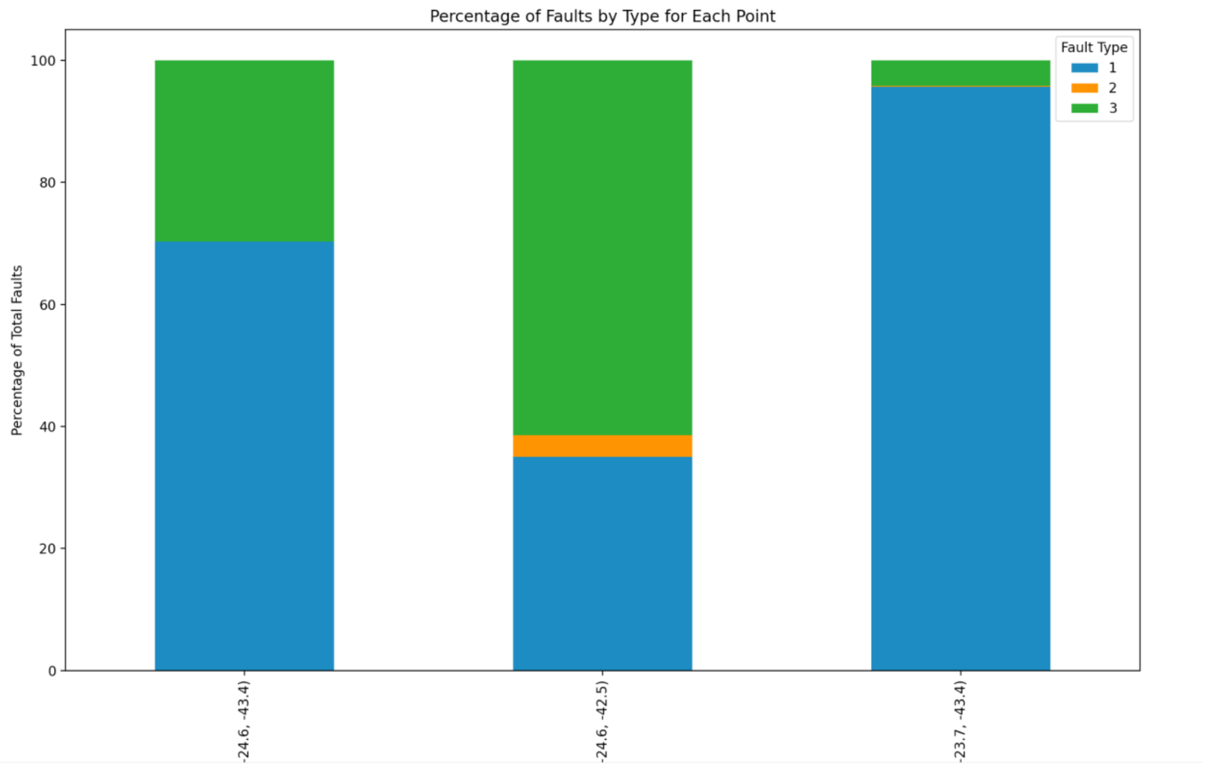

Figure 24**. A stacked bar chart that shows the number of faults per type for each point of interest with the Y-axis representing the percentage of the occurrences.**

Getting the exact numbers of the above plot reveals the success of this phase of the experiment since two of the three points, (-24.6, -43.4) and (-23.7, -43.4), exhibit 70.2% and 95.6% type 1 faults respectively. Table 22 displays the exact numbers for all three points and the different fault types.

Points (X, Y)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
-24.6, -43.4	70.270%	0%	29.729%
-24.6, -42.5	35.040%	3.504%	61.455%
-23.7, -43.4	95.672%	0.116%	4.210%

Table 22: Distribution of type of faults per point, with the highest percentage per point highlighted in bold.

When compared to the percentages acquired from the targeted scan, we can see a significant increase in the efficiency of the attack. The shift can be observed in Table 23, which displays the percentage difference for each fault type at each point.

Points (X, Y)	Type 1	Type 2	Type 3
-24.6, -43.4	+57.770%	-4.166%	-53.604%
-24.6, -42.5	+21.238%	+0.64%	-21.878%
-23.7, -43.4	+84.143%	+0.116%	-84.26%

Table 23: Difference, in percentage, for each fault type per point in comparison to the second scan (targeted scan)

The third point in figure 24 (-23.7, -43.4) yielded the largest number of faults with the majority of them being type 1 faults significantly outperforming the other two points. On the first two scans, the distribution of the type of faults and the total occurrences was similar for these points. This significant change demonstrates that each point can have its particularities, which leads to different results depending on the search space parameters. Visualizing the type of faults for all the voltage levels tested, Figure 25 illustrates the levels at which the most faults occurred.

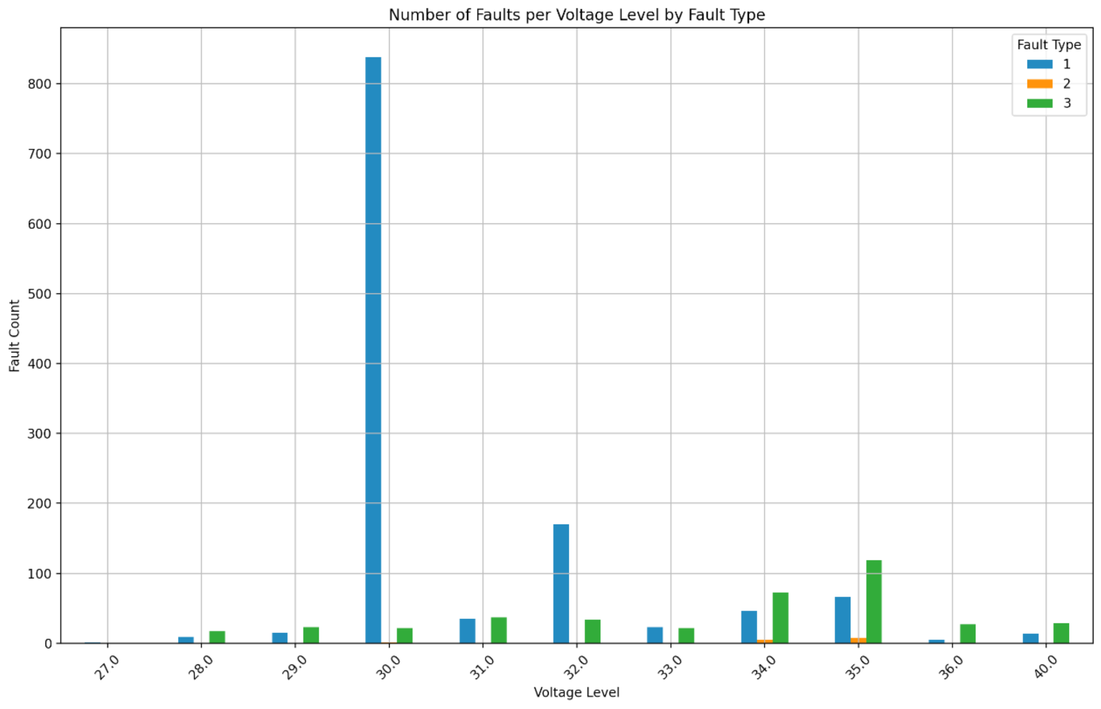

Figure 25**. Bar chart with the number of faults per point per voltage level for each fault type. The X-axis represents the voltage level parameter given to the probe on each test. The actual voltage level is ten times the referenced value.**

Figure 25 provides significant insight into the distribution of faults depending on the voltage level. It appears that lower voltage levels can still produce faults without resulting in the failure of the program being executed (type 2 and 3 faults). Moving on we will examine the sensitivity of each point in correlation to the voltage level. Figure 26 displays a heatmap that correlates the occurrences of faults with each voltage level for each point.

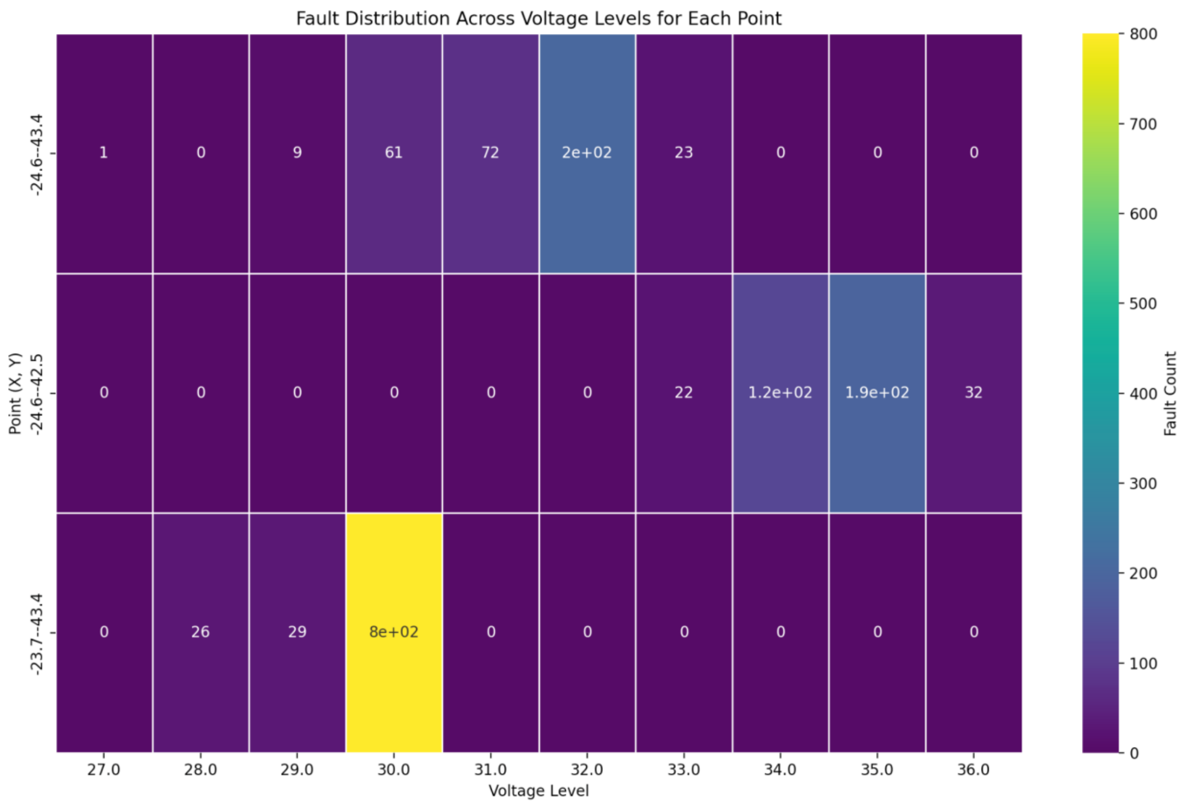

Figure 26**. Heatmap showcasing the faults per point per voltage level for point of interest scan. The X-axis represents the voltage level parameter given to the probe on each test. The actual voltage level is ten times the referenced value.**

It appears that each point exhibits sensitivity on a different voltage range with the first point on the heatmap (-24.6, -43.4) producing most faults around 320 volts, the second point (-24.6, -42.5) around 340 to 350 volts while the last point (-23.7, -43.4) produced the majority of faults in the 300 voltage range. If we further break down the information presented on the heatmap of Figure 26 we can observe the distribution of the different fault types for each voltage level across each point. Starting with the first point, (-24.6, -43.5), in figure 27 appears to exhibit a higher sensitivity at 320 volts. Furthermore, we can see how the number of faults increases as it reaches that level from 290 volts to 310 volts.

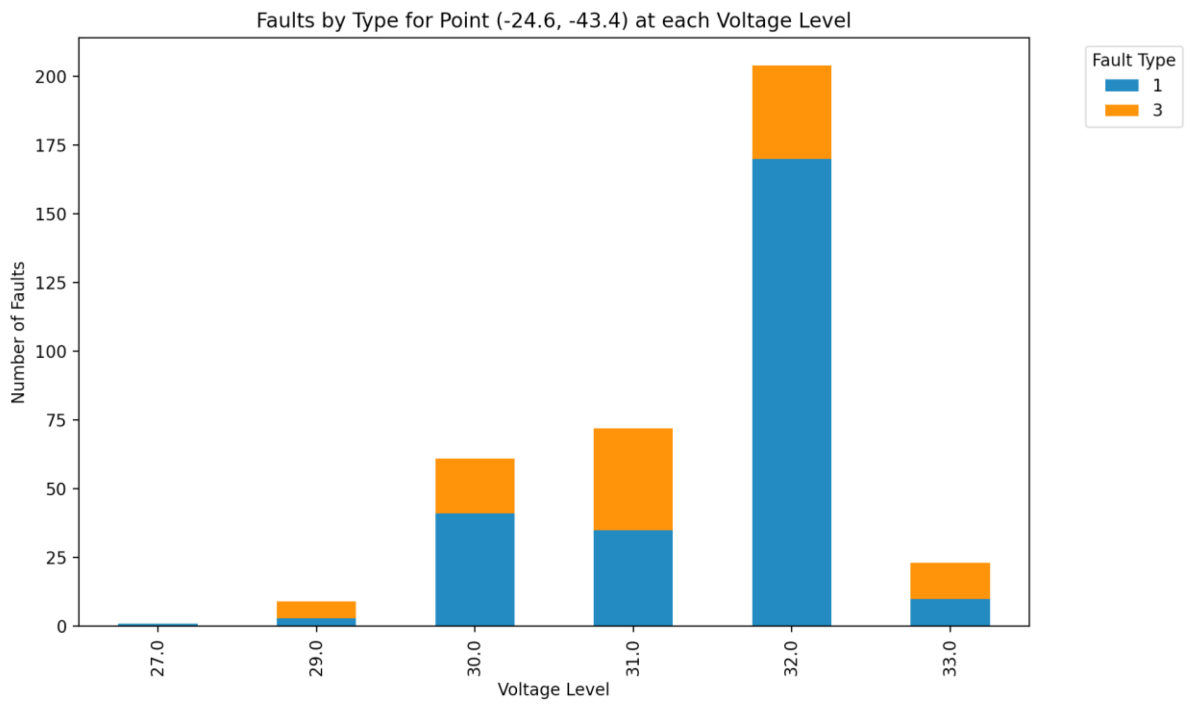

Figure 27**. Stacked bar chart showcasing the number of faults generated, separated by type, for each voltage level parameter that the point (-24.6, -43.4) exhibited sensitivity. The X-axis represents the voltage level parameter given to the probe on each test. The actual voltage level is ten times the referenced value.**

Table 24 contains the exact number of fault types per voltage level. For this point, we will select the voltage level corresponding to 320 volts. This level exhibits the largest number of total faults generated while maintaining a significant difference between the type 1 faults and the other two types. This substantially increases the chances of a successful attack while minimizing the time required.

Voltage level (*10 for actual value)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
27	1	0	0
29	3	0	6
30	41	0	20

31	35	0	37
32	**170**	**0**	**34**
33	10	0	13

Table 24: The number of generated faults per type for each of the voltage levels that exhibited sensitivity for the point (-24.6, -43.4)

Moving on to the second point, Figure 28 contains all the voltage levels that the point exhibited sensitivity. In this point's case, it is not very clear which is the optional level to perform an attack as they are all exhibiting many type 3 faults.

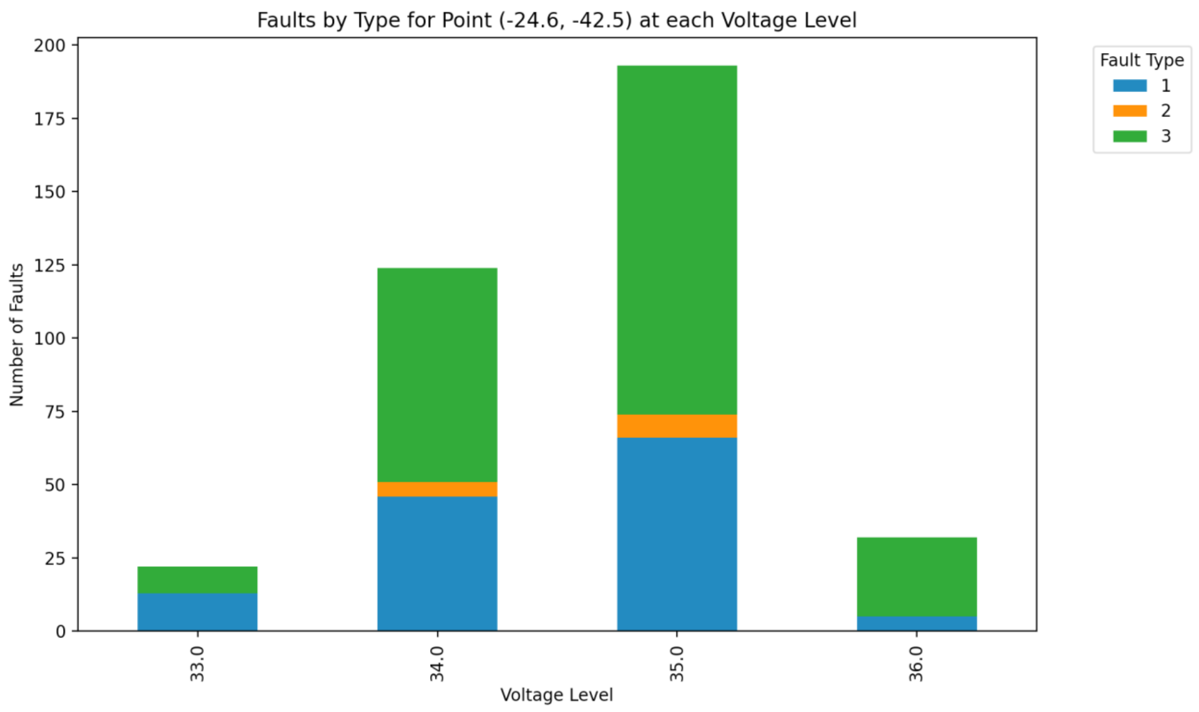

Figure 28**. Stacked bar chart showcasing the number of faults generated, separated by type, for each voltage level parameter that the point (-24.6, -42.5) exhibited sensitivity. The X-axis represents the voltage level parameter given to the probe on each test. The actual voltage level is ten times the referenced value.**

The extract numbers of Figure 28, as seen in Table 25, do not provide a clear picture. It appears that the search parameter analysis for this point where not sufficient to identify a set of parameters with a high success rate by analyzing the number of faults generated. At first glance, it appears as if the voltage level 35 is the best performing due to the high number of type 1 faults. However, it also has a high number of type 3 faults, which are time-consuming. The most efficient level is highly likely to be 34. In the next section, efficiency analysis, we will expand more on how to identify the most optimal configuration.

Voltage level (*10 for actual value)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
33	13	0	9
34	**46**	**5**	**73**
35	66	8	119
36	5	0	27

Table 25: The number of generated faults per type for each of the voltage levels that exhibited sensitivity for the point (-24.6, -42.5)

Moving on to the last point of interest, in Figure 29 we can see the three voltage levels that yielded results for this point. The voltage level parameter corresponding to 300 volts has a significant advantage over the other two.

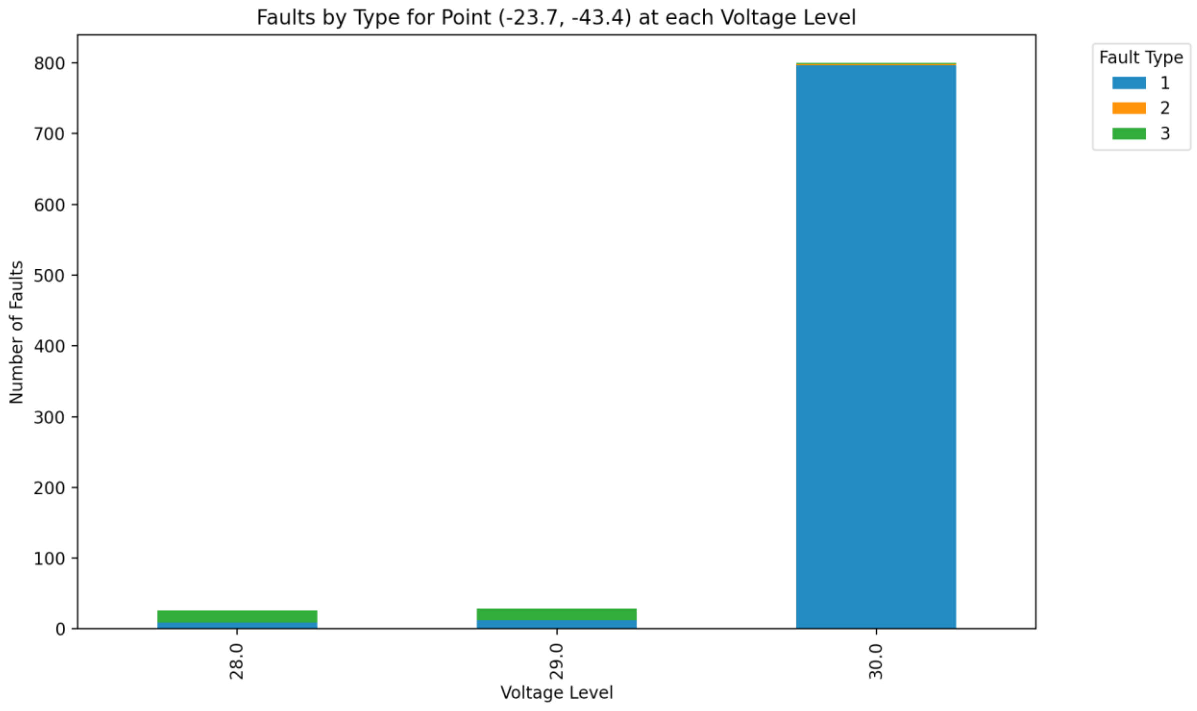

Figure 29**. Stacked bar chart showcasing the number of faults generated, separated by type, for each voltage level parameter that the point (-23.7, -43.4) exhibited sensitivity. The X-axis represents**

the voltage level parameter given to the probe on each test. The actual voltage level is ten times the referenced value.

The exact numbers, shown in Table 26, show that almost all of the faults, when the point is subjected to 300 volts, are type 1 faults.

Voltage level (*10 for actual value)	Type 1: Erroneous Output	Type 2: System Hang (Firmware re-programming needed)	Type 3: System Hang (Power cycle and re-programming needed)
28	9	0	17
29	12	0	17
30	**797**	**1**	**2**

Table 26: The number of generated faults per type for each of the voltage levels that exhibited sensitivity for the point (-23.7, -43.4)

6.3.1 Efficiency analysis

During this phase of our experiment, we concluded that each point could exhibit sensitivity to different voltage levels, which might also be the case for other parameters of the search space. By performing a sensitivity analysis for points that exhibit a high number of fault occurrences it is possible to finetune the parameters to produce meaningful results when performing EM fault attacks.

Out of the three points of interest analyzed we identified voltage level parameters for two of them that significantly increase the attack efficiency. If we compare again the distribution of faults, but this time for each specific voltage level selected for the points, in contrast to the overall number of faults that we did at the start of this section, we can see an even greater increase in the generation of faults. Table 27 shows the percentage of each type of fault, based on the total number of faults generated for each of the points and the selected voltage levels.

Voltage level (*10 for actual value)	Points (X, Y)	Type 1	Type 2	Type 3
32	-24.6, -43.4	83.333%	0%	16.667%
34	-24.6, -42.5	37.096%	4.032%	58.870%
30	-23.7, -43.4	99.625%	0.125%	0.250%

Table 27: The distribution of faults, expressed as percentages, for each type of fault relative to the total number of faults generated at each point, for a specific voltage level.

If we compare the fault distribution of the scans in Table 27 with the statistics obtained during the targeted scan for the same points, using the voltage level of 40, we can see a significant increase Table 28 shows the shifts in the percentages for each type of fault. The number of total scans for both scans where the same. The increase in type 1 faults indicates a higher probability of encountering a type 1 fault each time the target exhibits faulty behaviour when subjected to an EM pulse.

Points (X, Y)	Type 1	Type 2	Type 3
-24.6, -43.4	+70.833%	-4.166%	-66.666%
-24.6, -42.5	+23.294	+1.168%	-24.463%
-23.7, -43.4	+88.096%	+0.125%	-88.220%

Table 28: The change in the distribution of each type of fault for the current phase, expressed as a percentage of the total number of faults generated, using the specified voltage level parameters, compared to the targeted scan.

Besides the change in the distribution of faults, we can also observe an overall higher occurrence of type 1 faults, for the same amount of total scans, with the exception of point (-24.6, -42.5), which shows a decrease in all types of faults. The table illustrates this change, showing the percentage increase in the number of faults generated at each point. The rise in the number of faults suggests a higher probability of encountering a fault when the target is subjected to EM pulses.

Voltage level (*10 for actual value)	Points (X, Y)	Type 1	Type 2	Type 3
32	-24.6, -43.4	+5566.66%	-100.0%	+70.000%
34	-24.6, -42.5	-13.207%	-54.545	-77.187%
30	-23.7, -43.4	+1432.692%	N/A	-99.498%

Table 29: The increase in the number of each type of fault for the current phase, expressed as a percentage, using the specified voltage level parameters, compared to the targeted scan.

When compared to the total number of scans (9800 per point), the number of generated faults remained low but again there was a significant increase from the targeted scan when compared to the adjusted voltage levels. Table 30 shows the percentage of each fault type occurrence compared to the total number of scans for each point for the targeted scan.

Points (X, Y)	Type 1	Type 2	Type 3
-24.6, -43.4	0.030%	0.010%	0.204%
-24.6, -42.5	0.540%	0.112%	3.265%
-23.7, -43.4	0.530%	0%	4.071%

Table 30: Percentage of type 1 faults in comparison to the total number of scans performed for the targeted scan.

Table 31 similarly shows the same information from the data obtained from the points of interest analysis. Additionally includes the increase in the percentage from the targeted scan for reference. The increase in the percentage of the number of faults relative to the total number of scans, performed for each point, further demonstrates that not only is the target more likely to experience faults when exposed to EM pulses, but also that the rate of faults per scan is escalating. This trend underscores the heightened vulnerability of the target under continued electromagnetic stress with the adjustments in the search space parameters.

Voltage level (*10 for actual value)	Points (X, Y)	Type 1 (increase)	Type 2 (increase)	Type 3 (increase)
32	-24.6, -43.4	1.734% (+1.704)	0% (-0.0102)	0.346% (+0.142)
34	-24.6, -42.5	0.469% (-0.071)	0.051% (-0.061)	0.744% (-2.521)
30	-23.7, -43.4	8.132% (+7.602)	0.010% (+0.010)	0.020% (-4.051)

Table 31: Percentage of type 1 faults in comparison to the total number of scans performed for the points of interest analysis. Inside the parentheses, the percentage difference from the targeted scan is visible

As we have seen, using the number of type 1 faults generated in comparison to the combination of all types of faults generated and the total number of scans we can calculate metrics in order to quantify the efficiency of each parameter configuration. Figure 30 and Figure 31 illustrate the two mathematical types used to calculate these two metrics.

$$\text{fault\_generation\_efficiency} = \left(\frac{\text{type\_1\_faults}}{\text{total\_faults}}\right) \times 100$$

Figure 30**: Fault generation efficiency metric**

$$\text{scan\_efficiency} = \left(\frac{\text{type\_1\_faults}}{\text{total\_scans}}\right) \times 100$$

Figure 31**: scan efficiency metric**

6.3.2 Conclusion of Point of Interest Sensitivity Analysis

Our analysis has demonstrated an increased susceptibility to faults when the target is subjected to EM pulses, with a notable rise in type 1 faults. This suggests that even minor modifications in the search space parameters, like adjusting the voltage level of the generated EM pulses, can significantly impact the efficiency of a fault attack. This highlights the need for a broader search space that will contain varying parameters, to test the target against a range of different configurations. This should also include other parameters besides the voltage level, like the trigger delay, the distance of the probe from the target, and other relevant parameters that could influence the outcome of the experiments. This will provide a more comprehensive understanding of the target susceptibility to EM pulses under different testing conditions, which could lead to more effective and efficient EM fault attacks.

7 Conclusions

Drawing this thesis to a close, we have demonstrated the creation of an automated attack platform, using MATLAB, for EMFI attacks, that cannot only facilitate targeted fault injections but also enable the mapping of a target's surface. Considering the high time complexity in performing these attacks, having a fully automated attack platform that can perform a series of different scans, monitor and log the target, and restore the target to a functional state in case of systemic failure, significantly enhances the efficiency of these attacks. This automation eliminates the need for constant manual observation.

Moreover, we defined a testing procedure, that clearly outlines the necessary steps required to configure the equipment, and target, and how to traverse through its surface using a snake pattern movement to minimize movements when performing a surface scan. Additionally, the procedure outlines all the possible states the target might end up in when performing EMFI attacks, and mitigation tactics to recover the target to a working state in an automated manner.

Besides the testing procedure, which focuses on the process of the technical control of the equipment, we also proposed a testing methodology that addresses the problem of mapping a target's surface susceptibility to EMFI attacks. Considering the high time complexity we aimed to design a modular methodology that allows for adjustable control over the level of spatial accuracy and search space parameter optimization based on the requirements of each experiment.

We also presented the results of our experiments, which we performed using the designed attack platform and proposed testing methodology. We showcased the effectiveness of the methodology by performing a series of scans, that helped us map the surface of the target MCU and identify its susceptible areas. Furthermore, we performed parameter optimization, focusing on the voltage level of the EM pulses, for selected points. This showcases that by optimizing the search space parameters it is possible to increase the efficiency of the attack and gain a higher number of meaningful faults.

Due to the constrained timeframe, we had to design, implement, and use the attack platform we were unable to exhaustively test all steps of the proposed testing procedure, and draw meaningful conclusions for each step, particularly in increasing spatial details. Furthermore, the development of complex algorithms that would help us minimize the required time in the parameter optimization phase and produce more accurate results was beyond our reach within this project's timeline. These enhancements could significantly improve the platform's effectiveness and contribute to a more robust methodology for performing EMFI attacks.

8 References

[1] R. W. Anwar, A. Zainal, T. Abdullah and S. Iqbal, "Security Threats and Challenges to IoT and its Applications: A Review," 2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC), Paris, France, 2020, pp. 301-305, doi: 10.1109/FMEC49853.2020.9144832.

[2] Luvaha, E. (2023). Data privacy, conceptual framework for iot based devices in healthcare: a systematic review. East African Journal of Information Technology, 6(1), 119-134. https://doi.org/10.37284/eajit.6.1.1333

[3] K. Gomina, J. -B. Rigaud, P. Gendrier, P. Candelier and A. Tria, "Power supply glitch attacks: Design and evaluation of detection circuits," 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), Arlington, VA, USA, 2014, pp. 136-141, doi: 10.1109/HST.2014.6855584.

[4] Saß, M., Mitev, R., & Sadeghi, A.-R. (2023). *Oops..! I Glitched It Again! How to Multi-Glitch the Glitching-Protections on ARM TrustZone-M*. arXiv:2302.06932. Available at: https://arxiv.org/abs/2302.06932

[5] Hutter, M., & Schmidt, J. (2013). The Temperature Side Channel and Heating Fault Attacks. IACR Cryptol. ePrint Arch., 2014, 190.

[6] Skorobogatov, S.P., Anderson, R.J. (2003). Optical Fault Induction Attacks. In: Kaliski, B.S., Koç, ç.K., Paar, C. (eds) Cryptographic Hardware and Embedded Systems - CHES 2002. CHES 2002. Lecture Notes in Computer Science, vol 2523. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-36400-5_2

[7] J. Obermaier, R. Specht and G. Sigl, "Fuzzy-glitch: A practical ring oscillator based clock glitch attack," 2017 International Conference on Applied Electronics (AE), Pilsen, Czech Republic, 2017, pp. 1-6, doi: 10.23919/AE.2017.8053601. keywords: {Inverters;Clocks;Field programmable gate arrays;Ring oscillators;Logic gates;Hardware},

[8] T. Korak, M. Hutter, B. Ege and L. Batina, "Clock Glitch Attacks in the Presence of Heating," 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, Busan, Korea (South), 2014, pp. 104-114, doi: 10.1109/FDTC.2014.20.

[9] Lu, Y. (2019). *Injecting Software Vulnerabilities with Voltage Glitching*. arXiv:1903.08102v1. Available at: https://arxiv.org/abs/1903.08102

[10] Lu, Y. (2023). *Attacking Hardware AES with DFA*. Available at: [link to the document if available online].

[11] Standaert, F.-X. (2008). *Introduction to Side-Channel Attacks*. UCL Crypto Group, Place du Levant 3, B-1348 Louvain-la-Neuve, Belgium.

[12] Breier, J., & Hou, X. (2022). *How Practical are Fault Injection Attacks, Really?* Silicon Austria Labs, Graz, Austria; Slovak University of Technology, Bratislava, Slovakia. Available at: https://eprint.iacr.org/2022/301

[13] Michel Agoyan, Jean-Max Dutertre, Amir-Pasha Mirbaha, David Naccache, Anne-Lise Ribotta, et al.. How to Flip a Bit?. On-Line Testing Symposium (IOLTS), 2010 IEEE 16th International, Jul 2010, Corfu, Greece. ff10.1109/IOLTS.2010.5560194ff. Ffemse-01130826f

[14] -, A. (2024). internet of things for healthcare. International Journal for Multidisciplinary Research, 6(2). https://doi.org/10.36948/ijfmr.2024.v06i02.16474

[15] Chukwuere, J. (2022). internet of things (iot) cybersecurity challenges and mitigation mechanisms. Khazanah Sosial, 4(2), 235-240. https://doi.org/10.15575/ks.v4i2.17638

[16] Ashwini, T. (2017). A small review on internet of things (iot). Ijarcce, 6(6), 275-282. https://doi.org/10.17148/ijarcce.2017.6648

[17] Ashtikar, P. (2019). Water resource management using internet of things (iot): literature survey. International Journal for Research in Applied Science and Engineering Technology, 7(9), 80-83. https://doi.org/10.22214/ijraset.2019.9012

[18] Bynagari, N. (2016). industrial application of internet of things. Asia Pacific Journal of Energy and Environment, 3(2), 75-82. https://doi.org/10.18034/apjee.v3i2.576

[19] Li, Y., Yang, F., Zhang, X., Ren, F., & Chen, C. (2022). Intelligent manufacturing execution design of gear industry based on internet of things technology.. https://doi.org/10.21203/rs.3.rs-2090613/v1

[20] Wang, D. (2021). A measurement study on the (in)security of end-of-life (eol) embedded devices.. https://doi.org/10.48550/arxiv.2105.14298

[21] Randolph, M. and Diehl, W. (2020). Power side-channel attack analysis: a review of 20 years of study for the layman. Cryptography, 4(2), 15. https://doi.org/10.3390/cryptography4020015

[22] K¨ommerling, O., Kuhn, M.G.: Design principles for tamper-resistant smartcard processors. In: Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology, p. 2. Berkeley, CA, USA, USENIX Association (1999)

[23] Boneh, D., DeMillo, R.A., Lipton, R.J. (1997). On the Importance of Checking Cryptographic Protocols for Faults. In: Fumy, W. (eds) Advances in Cryptology — EUROCRYPT '97. EUROCRYPT 1997. Lecture Notes in Computer Science, vol 1233. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-69053-0_4

[24] Madau, M., Agoyan, M., Maurine, P. (2018). An EM Fault Injection Susceptibility Criterion and Its Application to the Localization of Hotspots. In: Eisenbarth, T., Teglia, Y. (eds) Smart Card Research and Advanced Applications. CARDIS 2017. Lecture Notes in Computer Science(), vol 10728. Springer, Cham. https://doi.org/10.1007/978-3-319-75208-2_11

[25] Wu, L., Ribera, G., Beringuier-Boher, N., & Picek, S. (2020). A fast characterization method for semi-invasive fault injection attacks., 146-170. https://doi.org/10.1007/978-3-030-40186-3_8

[26] Emanuele, C., Zaccaria, V., Gabriele, Q., & Molteni, M. (2023). efficient attack-surface exploration for electromagnetic fault injection., 23-41. https://doi.org/10.1007/978-3-031-29497-6_2

[27] https://github.com/bricke/tiny-AES-C