Differential Power Analysis on ARM32 Based Microcontrollers

Joshua Daniel Yewman

Supervisor: Prof. Konstantinos Markantonakis

Department of Computer Science / Department of Information Security
Royal Holloway, University of London

A report submitted in part fulfilment of the degree of
Masters in Science (Hons) Computer Science

July 2023

This project is dedicated to my parents, Karen and Neil Yewman, for without their support, this project and degree would never have been possible.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Joshua Daniel Yewman
July 2023

Abstract

This project will explore hardware attacks targeted at ARM microprocessors to extract cryptographic keys used within the encryption of data using side-channel power-based analysis. With the discovery of two high-class vulnerabilities, “Spectre” and “Meltdown”, for Intel’s x86 architecture brings these types of attacks into mainstream media. These attacks expose critical flaws in the design of microprocessor architecture and pose a significant challenge for the mitigation and resolution of attacks due to the nature of the vulnerabilities [30]. The purpose of such attacks is to bring the hardware into an undefined state whereby it either surrenders data or allows the deployment of an attack payload. These types of attacks have the possibility of being extremely damaging, as the combination of exploiting vulnerabilities without direct access to the target poses an even more significant threat to modern microprocessor security [8].

Table of contents

Nomenclature

Acronyms / Abbreviations

BGA Ball-Grid Array

CBC Cipher Block Chaining Mode

CDIP Ceramic Dual In-line Package

CFB Cipher Feedback Mode

CNN Convolutional neural network

CPU Central Processing Unit

CQFP Ceramic Quad Flat Package

CTR Counter Mode

DDoS Distributed Denial of Service

DFN Dual Flat No-Lead

DIP Dual Inline Package

DPA Differential Power Analysis

DSV Delimiter-separated values

ECB Electronic Codebook Mode

GB Gigabyte

GHz Gigahertz

GPIO General Purpose Input Output

GPU Graphics Processing Unit

IO Input/Output

IoT Internet of Things

LAN Local Area Network

LCC Leadless Chip Carrier

MCs Microcontrollers

MCU Module Control Unit

MHz Megahertz

ML Machine Learning

ML Machine Learning

OFB Output Feedback Mode

PCCs Power Consumption Curves

PLCC Plastic Leaded Chip Carrier

QFN Quad Flat No-leads

QFP Quad Flat Package

RPI Raspberry Pi

SOIC Small Outline Integrated Circuit

SPA Simple Power Analysis

TSOP Thin Small Outline Package

Chapter 1

Introduction

IoT or the Internet of Things is a term used for everyday devices that can be controlled or monitored via a network (for example the Internet). For example, a doorbell such as the "Ring Video Doorbell", connects to the consumer's home wireless network, enabling the device to send a notification to the consumer's phone, from anywhere in the world.

This advancement in affordable and connected devices unlocks the potential for home automation, remote security and remote control for consumers. Just as the landline telephone changed the way people viewed technology, not as a purely scientific tool, but as a household item, IoT devices are doing just that again.

IoT devices use low-powered RISC microprocessors, the majority of which are developed by ARM or Atmel. This is due to the fact that most IoT devices are designed to operate at low power, such as only having a battery supply. Despite this, due to the increase in performance, these microprocessors can now encrypt and decrypt data in real time. This allows cryptographic algorithms such as AES to run directly on the microprocessor. The advantage of this is to protect communications between devices and protect the device's internal data processing as well.

The primary methods of encrypting data is limited to a select number of algorithms, these being DES, Triple DES and AES.

Side-Channel attacks have been successfully deployed on consumer IoT and embedded devices [43]. This poses a significant threat not only to the end users but potentially to external entities, this is a result of the increase of compute power within IoT devices, enabling the possibility of these devices being used to mount numerous attacks such as Distributed Denial of Service (DDoS) attacks on external entities [35].

The use of side-channel analysis to discover hardware and software-based vulnerabilities has been an active field of research since the late 1990s. This type of analysis was primarily focused on smart-card based cryptographic functions, but in recent years increasing research,

and development of specialist devices and tools, side-channel analysis has become a powerful tool to both ensure hardware security and to prove otherwise.

1.1 Project objectives

The objectives of this research project is to:

- Identify a suitable target device to analyse and attack which contains a widely used microprocessor, such as those used in IoT devices.
- Identify vulnerabilities using within a suitable target using Side-Channel analysis
- Utilise the data collected to extract the secret keys of the encryption algorithm being executed on the microprocessor.

1.2 Project Structure

This project is split into 2 primary sections, the theoretical aspects of side-channel analysis and attacks and the practical implementation of side-channel analysis and attacks on a target device. The theoretical aspects of this project can be seen from the Second to the Fourth Chapters. The practical aspects of this project can be found in Chapter 5 onwards.

1.2.1 Chapter Outlines

Chapter 2 provides an overview of the most common and rapidly growing application of low-powered ARM microprocessors, the Internet of Things. This chapter also explores the history of the topic and how this project is relevant to most IoT-enabled devices.

Chapter 3 analyses the microcontrollers currently available on the market and explores the differences in architecture and use cases for the varied microcontrollers that are currently available.

Chapter 4 focuses on side-channel analysis, beginning with an overview of common encryption algorithms such as DES, Triple DES and AES. Further detail on AES is provided, including an overview of the operation and implementation of AES. This is then followed by an explanation of the different types of side-channel analysis and how they are performed, along with the history of side-channel analysis. A detailed analysis of power analysis is then performed, with explanations of the different types of power analysis, including Simple Power Analysis and Differential Power Analysis, finally followed by countermeasures for power analysis.

Chapter 5 explores the equipment needed to perform power analysis, along with extra tools and toolkits which supplement both the collection and analysis of the data.

Chapter 6 describes the physical data collection from the target device and the methodologies used to collect the data given the equipment described within Chapter 5.

Chapter 7 provides the security analysis of the collected data, more specifically, performing Differential Power Analysis on the collected power traces from Chapter 6. This chapter also investigates the limitations of this type of analysis and also its wider implications.

Chapter 2

IoT Devices

2.1 Definition of IoT

IoT or Internet of Things devices are physical devices with sensors, outputs, processing ability and software that connect to a network to exchange data with other devices or systems [10]. IoT devices can range from anything, such as connected microwaves to light bulbs. These devices have also transitioned into security in recent years through IoT-connected door locks, alarm systems and CCTV. Other types of IoT devices include speaker systems, fridges, HVAC (Heating and Ventilation Control) systems and many more.

Some of these IoT-enable devices would classify as mission-critical in terms of their security, for example an IoT enabled access control system. Mission-critical hardware and software is not a new concept. For example, the Apollo Guidance Computer, built in 1966, put the first humans onto the moon. By today's standards, a computationally basic computer was designed to be fault-tolerant, self-correcting, and "connected". This device was not connected to the internet, as there was no such thing at the time; however, it did send and receive a vast amount of data from control on earth [42]. The standards and methods of developing such software are still in use to this day and are still relevant to IoT devices currently deployed and in development. Considering the redundancy and resiliency required of such devices. The requirement to develop software and hardware that is secure, robust and reliable for physical-security IoT devices is one of paramount importance.

2.2 The IoT market

We have seen an exponential increase in the amount of IoT-enabled devices entering the market. IoT-enabled devices have reached approximately 14.4 billion worldwide [53], with this figure expected to continue growth at a continually increasing rate.

With the predicted number of devices to rise precipitately over the coming years, it brings an important question to light, how secure are these devices from attack?

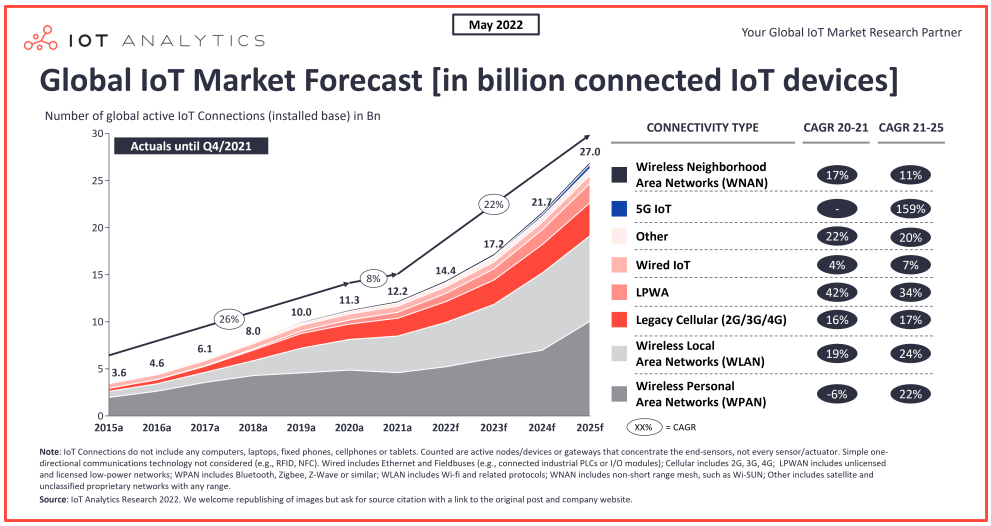

Fig. 2.1 IoT Market Increase [53]

The question of hardware security has been researched in depth, for example, research which is based specifically on IoT security [52] concluded that there are numerous attacks that can be mounted based on the hardware of IoT devices and referenced a term called "Hardware Trojan" which acts very much like a traditional (software implemented) virus, whereby the hardware is vulnerable to attack, this could be through a zero-day vulnerability or physical modification of the hardware, in which the end-user and developer do not know of its existence or potential threat until exploited.

2.3 Types of IoT Devices

As previously discussed, the IoT market is consistently growing in size, with more devices becoming connected. IoT-enabled devices can be split into subsections.

2.3.1 Home Automation Devices

IoT-enabled home automation devices, such as light bulbs, blind and curtain operators, smart speakers and heating/cooling control systems have become more prevalent over the past 5 years. This is primarily due to the decrease in the price of these devices and their ease of use for end-users.

Most of these devices do not directly communicate with the Internet, however, there are some notable exceptions such as the "Nest Thermostat" manufactured by Google [37]. Most consumer-grade IoT-enabled devices communicate with a central "hub", which is then connected to the LAN.

The communication between individual IoT-enabled devices uses protocols such as "Zigbee". This protocol is a low-power wireless communication protocol, and based on the IEEE 802.15.4 specification for high-level communication protocols. The advantage of using such a protocol for communications directly from IoT-enabled devices is it's low power consumption as compared to implementing WLAN 802.11, this is especially useful when the devices are solely battery powered [62]. Another advantage of using such protocols is the additional security, more specifically, protocols such as Zigbee requires specialist equipment such as signal analyser to intercept the data transmitted by the devices.

The ease of use of these devices, and their ability to integrate with other IoT-enabled devices, as well as mobile devices. The ability to easily control these devices via a mobile device has further increased the popularity of such devices within the consumer sector [61].

2.3.2 Access Control, Security and Mission-Critical Devices

Security devices, such as doorbells with integrated cameras, intruder alarm systems, physical access control systems (such as door locks) and CCTV systems are examples of these types of IoT-enabled devices.

Both the software and hardware of these devices are paramount. As previously mentioned the software requirements for these devices are very much mission-critical. The security of the hardware is also of paramount importance, the requirement to evade and protect against side-channel analysis and attacks is fundamental to the security of these devices.

2.3.3 Other Devices and Implications

It is important to note, that IoT devices aren't isolated to the consumer sector. These devices can also be found in industries such as healthcare. For example, smart pacemakers and other monitoring systems. Other industries include transport, with connected cars and other vehicles. Industrial IoT devices are also coming to market. For example, digital control systems, agriculture and manufacturing utilising IoT-enabled devices.

The rapid increase in the development and availability of these devices poses a serious question, especially for mission-critical applications such as medical, or security-based IoT devices. Are the devices we use day-to-day secure, both in their ability to process data, and to communicate that data to other devices. Modern cryptographic standards such as AES are classified as secure, and have been since their introduction at the start of the century. However, the implementation of such algorithms is may pose a threat to the integrity of both the data being encrypted with these algorithms, as well as the cryptographic algorithms themselves.

Chapter 3

Microcontrollers

Microcontrollers have become more prevalent in everyday life, and it is now easier than ever to use microcontrollers to develop complex and connected systems. For example, the Arduino project, which began in 2005, is an open-source microcontroller [57], making creating projects easy. The advantage of such projects is that they make creating advanced systems using a combination of both software and hardware easy to accomplish.

3.1 Legacy Microcontrollers (1970/1990)

Microcontrollers for use in hobbyist projects date back further than the Arduino Project. The PIC microcontroller developed by Microchip Technology dates back to the 1970s with the PIC1650 [13]. However, these microprocessors were complex to program and expensive at the time. These microprocessors also carried another constraint, that of the complexity of the software deployed on them, this was primarily due to memory limitations, however, the limited compute capabilities of these chips were also an important factor. The advantage of projects such as Arduino is that they can be programmed in easier-to-use languages such as C++ and deployed instantly via USB.

Fig. 3.1 PIC1655 [39]

3.2 Modern Microcontrollers(2000/Present Day)

Focusing on Arduino devices, additions have been made to allow them to communicate via a LAN, either with the use of Ethernet or WiFi. This addition allows the microcontrollers to send and receive data from any device at any time. This poses both the advantage and disadvantages of enhanced communication with the MCs. The disadvantage is that poorly implemented software can put the system at significantly more risk of attack because an attacker no longer needs to have direct contact with the device to perform an attack. The connectivity of these devices also means that the device's hardware may be at higher risk of attack; this could be through poorly designed hardware or microarchitectural vulnerabilities in the microcontroller itself.

The majority of Arduino devices use microcontrollers that are either made by Atmel or Atmel derivatives, specifically the AVR family of microcontrollers. These chips are 8-bit RISC microcontrollers.

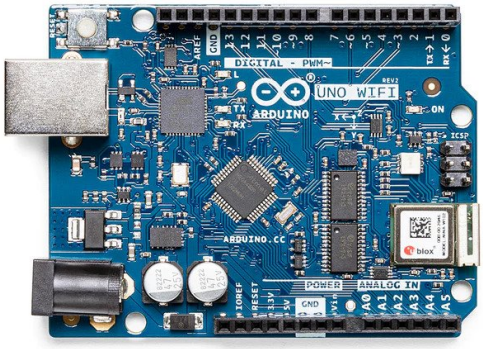

Fig. 3.2 Arduino Uno Wifi REV2 [5]

There are other notable examples of other widely used microcontrollers, those made or licensed by ARM, specifically the Cortex-M-based microcontrollers seen in devices such as the Raspberry Pi Pico.

While the ARM Cortex-M isn't directly classified as a microcontroller, as it is a microprocessor. The Cortex-M is often implemented in single-chip microcontroller packages. The implementation of the Cortex-M in microcontrollers gives numerous advantages, specifically in the computational power offered by the Cortex-M, the low power consumption, and enhanced GPIO (General Purpose Input/Output).

3.3 Advancements in capabilities

As the price of microcontrollers has decreased over the last 20 years [27], a logical transition from "lower-powered" (less complex) microcontrollers to more complex and feature-full microcontrollers has been seen. While such microcontrollers have become more complex, so has their efficiency, especially in terms of power efficiency. A well-known example of the increases in the power efficiency of microprocessors is ARM, due to its RISC (Reduced Instruction Set Computing) architecture, it enables the power consumption to average as low as 5 watts [58]. The primary advantage of this is the increase in computing power as well as the decreasing power consumption for microcontrollers, furthermore, the addition of more features, including wireless networking and cryptographic capabilities allows more complex devices to be created, for a continued lower cost over time.

A key example of this is the ESP range of microcontrollers created by Espressif Systems[19]. These devices allow low-cost and high-performance microcontrollers to be manufactured, with built-in wireless networking capabilities with the inclusion of an integrated TCP/IP stack within the microprocessor itself [19]. Not only do such devices allow advancing features such as wireless communication, but they also include significantly more I/O capabilities, making such microprocessors very attractive for mobile and embedded applications.

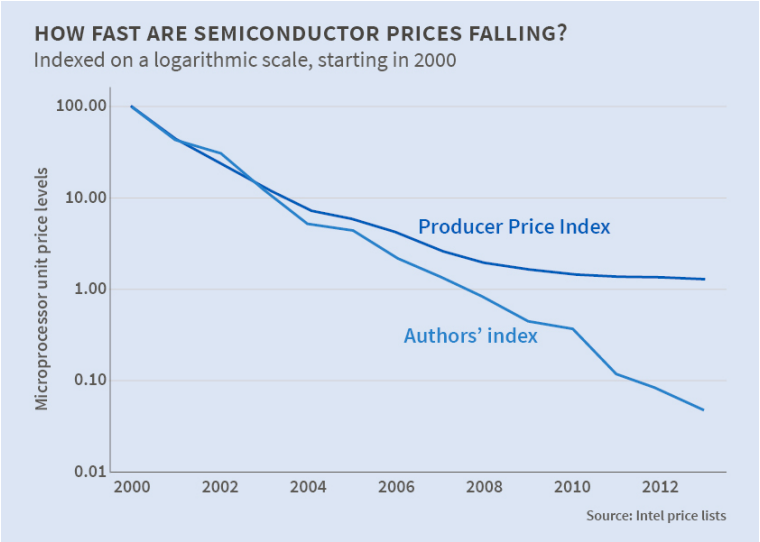

Fig. 3.3 Semiconductor Price [27]

3.3.1 Microcontroller Package types

When analysing potential attack vectors in side-channel analysis, it is important to consider the microcontroller packages themselves. The differences in package types can often make side-channel analysis more (or less) complex to perform.

There are multiple types of integrated circuit packages used in microcontrollers, but not all are commonly used. Examples of packages are:

- Dual In-line Package (DIP): DIP is a through-hole package that has two rows of pins, typically used for older digital logic and linear integrated circuits. The pins are spaced at a 0.1-inch pitch and are inserted into holes on a printed circuit board.
- Small Outline Integrated Circuit (SOIC): SOIC is a surface-mount package that has two parallel rows of pins with a 0.05-inch pitch. SOIC packages are commonly used for digital and analogue circuits, and they have a smaller footprint than DIP packages.
- Quad Flat Package (QFP): QFP is a surface-mount package that has four sides of pins, typically used for digital and analogue circuits. QFP packages have a square or rectangular shape, and the pins are spaced at a pitch of 0.5mm to 1.0mm.
- Ball Grid Array (BGA): BGA is a surface-mount package that has an array of metal balls on the bottom of the package instead of pins. BGA packages are used for high-density digital circuits and microprocessors, and they provide better electrical performance and thermal dissipation than other packages.
- Plastic Leaded Chip Carrier (PLCC): PLCC is a through-hole or surface-mount package that has J-shaped leads on all four sides of the package. PLCC packages are used for digital and analogue circuits, and they provide better mechanical and thermal performance than DIP packages.
- Quad Flat No-leads (QFN): QFN is a surface-mount package that has no leads on the sides of the package. Instead, the pins are located on the bottom of the package. QFN packages have a small footprint and are used for digital and analogue circuits.
- Ceramic Dual In-line Package (CDIP): CDIP is a through-hole package that has ceramic material instead of plastic. CDIP packages have good thermal performance and are used for high-reliability applications.
- Ceramic Quad Flat Package (CQFP): CQFP is a surface-mount package that has ceramic material instead of plastic. CQFP packages have better thermal and mechanical performance than QFP packages.

There are also other less common types of IC packages such as Thin Small Outline Package (TSOP), Leadless Chip Carrier (LCC), and Dual Flat No-Lead (DFN), among others.

Packages that contain pads on the underside of the package are especially difficult to analyse using power-based side-channel analysis, this is as the pins themselves are precluded by the package. One countermeasure to this issue would be to remove the chip from the PCB and analyse it on a specialised PCB which exposes each of the pins located on the underside of the package. However, this may have a negative effect on the results of any analysis, this is as the interaction between the target chip and other components on the original PCB may effect the results of the analysis.

Chapter 4

Side-Channel Analysis

4.1 Side-Channel Preliminaries

4.1.1 Physical Access

The majority of side-channel attacks require physical access to the target device. Some methods, such as optical analysis, require invasive and potentially destructive access to the silicon within an Integrated Circuit Package.

However, there are some methods of side-channel analysis which do not require "direct" access to the device. For example, Electromagnetic emission analysis does not necessarily require direct access to the device. For example, research published in Topics in Cryptology outlined a method in which secret keys could be remotely retrieved from a laptop using a directional antenna [21].

4.1.2 Power Consumption

Power consumption within the terms of side-channel analysis is the measurement of real-time current draw from a microprocessor. This is important to note because a side-channel analysis is focused on the difference of current draw of the device associated with operations, this could either be a single instruction, or a cryptographic algorithm. These measurements are usually performed with the use of an oscilloscope, however, there are specific tools available that will be discussed later, which are designed specifically for side-channel power analysis.

4.1.3 Fundamentals of Side-Channel Analysis

Side-Channel attacks follow the same principles as software attacks, to gain access to a system with the goal of collecting data, bypassing security mechanisms or creating other

vulnerabilities that can be later exploited. Side-Channel attacks differ from other types of attacks such as software attacks, this is because physical access to the device is required in order to perform the analysis. Side-Channel attacks also require expensive hardware to perform the analysis and attack.

A side-channel attack is classified as any attack based on the fundamental implementation of software or hardware. These attacks can be achieved by gathering information through power consumption and electromagnetic emissions and by introducing potential fault-inducing states, for example, timing or power attacks.

4.2 History of Side-Channel Analysis

Side-Channel analysis can be traced back to the 1950s, most notably the U.S National Security Agencies project TEMPEST. This project utilised electromagnetic emissions to recover the plain text of encrypted military communications [18].

Side-Channel Analysis and attacks gained further momentum with the introduction of smart cards, specifically chip and pin cards used for payments. It was discovered that early smart cards were vulnerable to a variety of side-channel attacks. For example, it was discovered that electromagnetic attacks such as exposing the chip to radiation, x-rays and light could, in some cases induce fault states allowing an attacker to bypass security measures on the chip [41]. Smart card based side-channel research continued for the majority of the 2010s. These attacks included power attacks based on power consumption analysis and electromagnetic emission analysis [36].

One of the primary catalysts for side-channel research within the smart card domain was the introduction of SIM cards, Pay-TV and EMV (Electron, MasterCard & Visa) chips used within bank cards. All of which continue to be used to date.

4.3 Types of Side-Channel Attacks

4.3.1 Timing Attacks

Timing attacks primarily analyze the time taken to perform cryptographic functions within an SoC (System on Chip) or microprocessor. By analyzing the time taken to perform computations, it is possible to compute the input of the cryptographic algorithm in which the encrypted text was generated.

If the plain text input is known, it is more statistically probable that the keys used to encrypt the plain text can be found.

There are multiple known instances where timing attacks have been deployed. For example, the Spectre and Meltdown attacks, which were side-channel attacks for the x86 architecture, relied on timing attacks.

4.3.2 Electromagnetic Attacks

Electromagnetic attacks focus on the electromagnetic radiation emitted from hardware. By analyzing this radiation, an attacker can capture data about a microprocessor's current computation. This could be in the form of cryptographic key extraction or the identification of computations or instructions.

There are multiple known attacks based on this principle. For example, within smartphones, it is possible to extract the ECDSA secret keys from SSL (Secure Socket Layer) on some Apple iPhone devices by monitoring the electromagnetic radiation emitted from the device.

4.3.3 Power-monitoring attacks

Power-monitoring attacks focus on the power consumption of a device while performing instructions or cryptographic algorithms. These attacks rely on having physical access to the device in order to perform profiling. These power samples, or traces, are usually collected by an oscilloscope. There are multiple types of power analysis attacks. Simple Power Analysis (SPA) involves interpreting the power traces over time. Differential Power Analysis (DPA) involves analyzing numerous samples, usually of a cryptographic algorithm, to obtain the cryptographic keys used in the encryption of the plain text.

4.3.4 Differential Fault Analysis

Differential fault analysis focuses on introducing fault states into a system. A fault is classified as an unexpected condition. This is then used to reveal the states of operations, such as cryptographic functions.

This type of attack has successfully been performed on embedded systems, such as smartcards; by exposing the device to excessive temperature, voltage, or radiation, the processor may begin to output incorrect data, which can then be used to identify the instruction the processor is currently executing or the identification of the internal data state.

4.3.5 Optical Analysis

Optical analysis focuses on the physical analysis of a decapsulated chip. This can be useful for analyzing solid-state memory such as Electronically Erasable Programable Memory (EEPROM) or another solid-state storage medium. By analyzing high-definition photography of the physical silicon, it may be possible to read data stored within the memory. If this technique is used on a chip that handles cryptographic operations, it may also be possible to retrieve hardware-defined secret cryptographic keys.

The disadvantage to this approach is this method is highly time-consuming and requires laboratory equipment to decapsulate the chip and take a high-resolution silicon image. This process is becoming increasingly difficult as the size of transistors on silicon wafers are decreasing in side down to 5nm, and decapsulating chips may also damage the silicon.

4.4 Power Analysis attacks

Power analysis has been used as a method of extracting information and data from embedded systems since the late 1990s [33]. This method utilises the study of the power consumption of a device while performing instructions or cryptographic operations. These attacks rely on the simple premise of when a semiconductor changes state the amount of current used will use will change proportionally.

Using this method it is possible to gain information about the IC's state or the data being processed. This can be particularly useful for analysing cryptographic algorithms used in encryption and decryption, as it may be possible to extract the secret keys used in cryptographic algorithms using these methods.

The collection of power consumption data within a given time frame is called a trace. A power trace begins at a trigger point, this is usually controlled by an external device, such as a microcontroller or the oscilloscope itself. The power trace finishes at the end of the target execution. For example, when capturing a power trace of AES encryption, the trigger would be at the start of execution and finish at the end of execution.

Data that is captured from a device using side-channel power analysis is regarded as leakage.

4.5 Simple Power Analysis

SPA or Simple Power Analysis involves collecting power traces from a device running an instruction or program over time. Using this method it is possible to identify specific

instructions running on a microprocessor, by visually or statistically observing the power trace.

Each instruction performed by a microprocessor will result in a variation of the current draw of the microprocessor, by measuring these variations it is possible to either identify a single instruction or reconstruct an entire program. The analysis can be performed visually and has successfully been accomplished [14].

4.6 Differential Power Analysis

DPA or Differential Power Analysis is one of the most common forms of side-channel analysis and attacks. To understand DPA, it is most logical to understand the power analysis method in a generalised form, such as SPA. Power analysis is a side-channel attack in which we monitor the power consumption of a microprocessor during cryptographic functions. By measuring the fluctuations in power current usage, we can determine information about the data currently being manipulated by the microprocessor. SPA involves the interpretation of power traces (usually collected with an oscilloscope) over time to determine the state or data being processed. DPA utilises multiple power traces to analyse correlations in power traces statistically [34]. Within DPA, we use the term leakage to define the method of extracting data from a microprocessor during the execution of cryptographic functions. For example, DES encryption gives a plain text input, and a pseudo-random key is generated for use within the cryptographic function. Leakage occurs when the microprocessor inadvertently exposes data. In this example, this would be the secret key used in DES.

As explained in [34] the differential traces are calculated as:

$$\Delta_D[j] = \frac{\sum_{i=1}^{N} D(P_i, K_s) T_i[j]}{\sum_{i=1}^{N} D(P_i, K_s)} - \frac{\sum_{i=1}^{N} (1 - D(P_i, K_s)) T_i[j]}{\sum_{i=1}^{N} (1 - D(P_i, K_s))} = \varepsilon_1 - \varepsilon_0$$

where K_s are the unknown key bits, P_i the i-th know plain text, $D(P_i, K_s)$ the selection function, $T_i[j]$ the j-th sample of the PCC and $\Delta_D[j]$ the j-th element of the differential trace [11].

Power analysis can also be used to determine the microprocessor's current instruction. The determination of instructions can be vital in breaking cryptographic functions and any program being executed by a microprocessor. For example, determining the execution path can be used to determine further when and what data is being processed [32].

4.7 Power Analysis Countermeasures

Many countermeasures have been developed to prevent the use of power analysis. These countermeasures are deployed on both a hardware and software level. These countermeasures aim to either prevent leakage or obfuscate the execution of processes. However, in low-power ICs, these techniques are either not implemented, or are not effective due to the low current draw of the IC making leakage more prominent during power trace collection.

Noise

This type of countermeasure, commonly referred to as "noise countermeasures", is widely used and is a form of software-level countermeasures. This countermeasure employs inserting "dummy" operations into the code to obfuscate the currently executed function. [54]. This "noise" makes an analysis of a function more complicated, as operations that serve no purpose can interrupt the execution flow of the function, therefore, creating a profile of the function using DPA more complex.

Masking

This type of countermeasure, called "Boolean Masking", is a method whereby the use of actual values for calculations is split into multiple randomly sized sections, thereby making extraction of actual data using power analysis more complex [12]. This method, however, is not commonly used today as the use of DPA makes extracting the actual values easier to accomplish [54].

Randomization

Randomization is a countermeasure that involves introducing random values or operations into the cryptographic algorithm to prevent an attacker from deducing information about the secret key. Randomization can be used to protect against SPA, DPA, and CPA attacks.

Power Smoothing

Power smoothing is a countermeasure that involves filtering or smoothing the power consumption of a device to make it more difficult for an attacker to extract sensitive information. Power smoothing can be used to protect against SPA and CPA attacks.

Higher-order Masking

Higher-order masking is a countermeasure that involves adding additional random values to the input or output of a cryptographic operation to increase the level of protection against SPA, DPA, and CPA attacks.

Secret Sharing

Secret sharing is a countermeasure that involves dividing the secret key into multiple shares and distributing them across different devices or components to make it more difficult for an attacker to extract the entire key. Secret sharing can be used to protect against DPA and CPA attacks.

Threshold Implementations

Threshold implementations are a countermeasure that involves dividing the cryptographic algorithm into multiple components, each of which requires a subset of the secret key to perform the encryption or decryption operation. Threshold implementations can be used to protect against DPA and CPA attacks.

4.8 Encryption algorithms

There is a multitude of potential encryption algorithms that could be used for analysis, DES, Triple DES and AES are all viable options for analysis and attack.

4.8.1 DES & Triple DES

There are multiple advantages and disadvantages of each algorithm. The DES encryption algorithm was developed in 1975 by IBM. DES is a symmetric-key block cipher encryption algorithm that uses a fixed block size of 64 bits, and a fixed key size of 56 bits. The algorithm encrypts and decrypts data using a series of 16 rounds of substitution, permutation, and XOR operations. The key schedule generates 16 round keys, each 48 bits long, from the 56-bit key, which is used in each round of the algorithm [6, 40]. DES in this form is no longer commonly used as the limited key size is susceptible to key exhaustion and brute-force attacks.

DES can be split into sections as outlined below:

1. The 56-bit is divided into 16 48-bit sub keys
2. The plain text input is divided into 64-bit blocks

3. The input block is permuted, which rearranges the bits according to a predefined pattern
4. The round function, which is in the form of a Feistel structure, of which contains 16 rounds
   (a) The 32-bit input is expanded to 48 bits using an expansion function
   (b) The 48-bit input is divided into eight 6-bit blocks, which are then substituted using the predefined S-Box. Each S-box takes a 6-bit input and produces a 4-bit output.
   (c) The 32-bit output of the S-Boxes is then permuted
   (d) The output of the permutation is then XOR-ed with the 32-bit half of the input block.
5. After the final round has been completed, the 64-bit output is permuted, which is the inverse of the initial permutation function used at the beginning of the algorithm

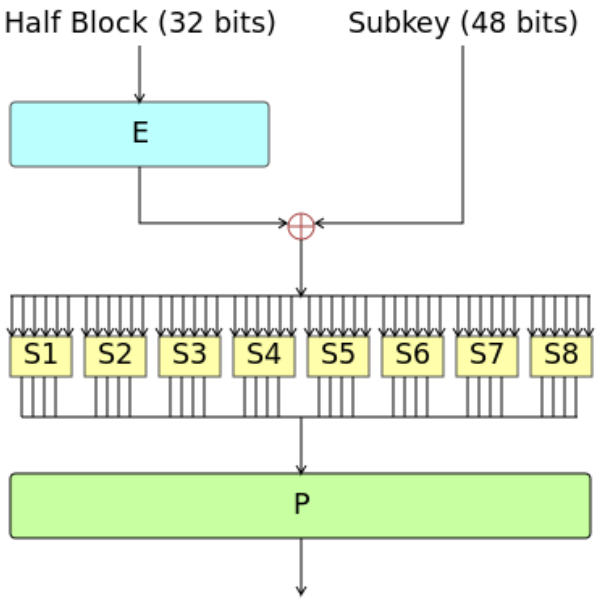

Fig. 4.1 DES Diagram [24]

TripleDES was designed to increase the security of DES by using three rounds of the DES algorithm and is designed to be secure and flexible. TripleDES has been used for secure communication, data storage, and other applications that require strong encryption.

4.8.2 Triple DES Vulnerabilities

With the creation of AES, Triple DES is also becoming obsolete in favour of AES [31, 46, 55]. This was compounded by a serious vulnerability discovered in both DES and Triple DES, namely CVE-2016-2183 [15]. This vulnerability implemented a birthday attack, in which DES and Triple DES have a birthday bound of four billion blocks, which allows an attacker to perform a birthday attack with limited computational resources.

4.8.3 AES

AES is ubiquitously for multiple applications and the principles of AES were first outlined in 1998 and adopted by NIST (The National Institute of Science and Technology) in 2001 [20]. The decision to choose this cipher is because of its continued relevance to modern-day information security.

There are multiple advantages to using AES, the most influential reason for choosing AES is there are implementations of AES designed to run on low-powered micro-controllers. The specific implementation that is employed in a lightweight version of AES called "Tiny-AES".

AES is a block cipher that can be implemented in multiple configurations, AES[block size], is used to denote the size of the fixed block in bits. The most common are AES128, AES192, AES256 and so on.

The number of rounds corresponding to each block size is 10 rounds for AES128, 12 rounds for AES192 and 14 rounds for AES256. While a full description of AES is not required for the scope of this project, a brief explanation of the key steps within the algorithm is required to understand the vector of attack on AES using DPA [28].

AES can be split into sections as outlined below:

1. Expansion of the round keys which are derived from the cipher key within the pre-defined AES key schedule.

2. (For the initial round) key addition. Each byte of the current state is combined with the round key using an XOR function.

3. During rounds 9, 11 or 13 (Depending on bit size) there are four steps executed.

   (a) SubBytes - This is a substitution where each byte is replaced with another in accordance with the S-Box

   (b) ShiftRows - The last three rows of the state are shifted in a number of steps.

   (c) MixColumns - A mixing operation that is applied to the columns of the state, where four bytes of each column are combined.

(d) AddRoundKey - The subkey is combined with the current state.

4. The final round

   (a) SubBytes

   (b) ShiftRows

   (c) AddRoundKey

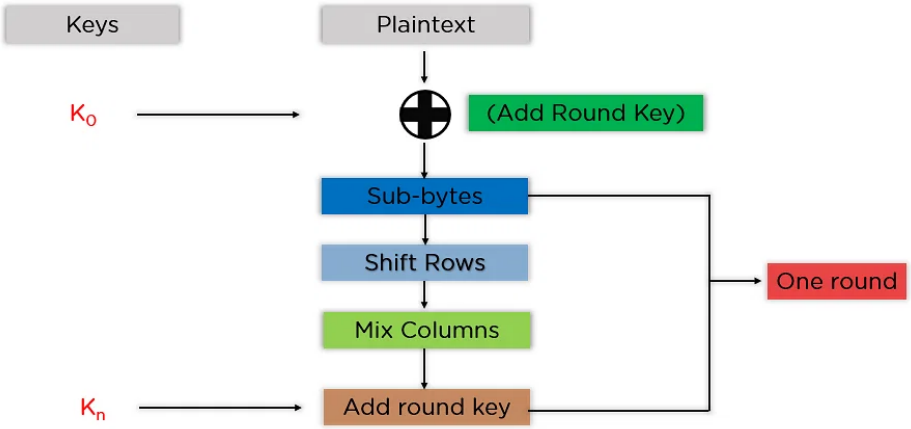

Fig. 4.2 AES Diagram [28]

The S-Box is predefined and is expressed in a matrix. This is especially useful for analysis, this is as the S-Box can be easily defined as a matrix within Python. Below is a sample of the forward S-Box used in AES. There is also an inverse S-Box which is simply the inverse of the forward S-Box.

```
sbox = [
    0     1     2     3     4     5     6     7
    0x63,0x7c,0x77,0x7b,0xf2,0x6b,0x6f,0xc5, ... 0
    0xca,0x82,0xc9,0x7d,0xfa,0x59,0x47,0xf0, ... 1
    0xb7,0xfd,0x93,0x26,0x36,0x3f,0xf7,0xcc, ... 2
    0x04,0xc7,0x23,0xc3,0x18,0x96,0x05,0x9a, ... 3
    0x09,0x83,0x2c,0x1a,0x1b,0x6e,0x5a,0xa0, ... 4
    0x53,0xd1,0x00,0xed,0x20,0xfc,0xb1,0x5b, ... 5
    0xd0,0xef,0xaa,0xfb,0x43,0x4d,0x33,0x85, ... 6
    0x51,0xa3,0x40,0x8f,0x92,0x9d,0x38,0xf5, ... 7
]
```

As can be seen, in this sample, the data S-Box is organised into rows and columns within a matrix starting with 0x63 at location 0,0. The AES S-Box is a fixed-size square matrix that spans 16 rows and columns. The S-Box maps an 8-bit input to a corresponding 8-bit output. Both the input and the output are interpreted as polynomials over a finite field.

4.8.4 Cryptographic Modes of Operation

Both the AES and DES (including Triple DES) algorithms support various modes of operation, each possessing distinct characteristics [16, 44, 29].

ECB

The Electronic Codebook (ECB) mode is the simplest, wherein each block of plain text undergoes encryption independently with the same key. However, this mode has a vulnerability, whereby identical plain text blocks produce identical cipher text blocks, making it exploitable by attackers.

CBC

The Cipher Block Chaining (CBC) mode operates by XORing each plain text block with the previous cipher text block before encryption, ensuring that identical plain text blocks result in different cipher text blocks. Furthermore, CBC mode provides integrity, as any modifications made to the cipher text affect the decryption of subsequent blocks.

CFB

The Cipher Feedback (CFB) mode utilizes the output of the preceding encryption operation to encrypt the next block of plain text, allowing for the encryption of individual bits and is suitable for streaming data.

OFB

The Output Feedback (OFB) mode is analogous to the CFB mode, except for the XOR operation of the key stream with the plain text, thereby producing the cipher text. Consequently, any alterations made to the cipher text do not influence the decryption of subsequent blocks.

CTR

The Counter (CTR) mode enables the transformation of a block cipher into a stream cipher, with the generation of a key stream from a counter. The XORing of the key stream with the

plain text produces the cipher text. CTR mode facilitates parallel encryption and decryption and is ideal for streaming data.

4.8.5 Choice of Cryptographic Mode

It is important to note that for the purposes of this research, AES128 in Cipher Block Chaining (CBC) mode will be used. This is because AES CBC Mode is the most commonly used implementation for the algorithm. CBC is also highly documented, with multiple sources available for its implementation with most architectures.

Chapter 5

Experiment Setup

5.1 Required equipment

5.1.1 Oscilloscope

To collect a power trace, an Oscilloscope with either passive or active probes is needed, as well as physical access to the target device. For this configuration, I am using two probes, one used as a trigger, and another for current measurement.

To gather accurate power traces, the probe needs to be connected as close as possible to the target package, this is to reduce noise or the effects of power smoothing capacitors.

When using passive oscilloscope probes 5.1, the use of a shunt resistor is required to measure the current draw of the target package. Ohms's law can then be used to calculate the required resistance value without "shorting" the power rail (by creating a direct path to ground). This slightly increases the complexity of the setup as this resistor will need to have a connection to the ground voltage rail as well as the positive voltage rail on the PCB.

Using an active Oscilloscope probe is also a viable option to gather power traces, the advantage of using an active probe is that you can directly measure current draw without the need for extra hardware (resistors) which may alter the accuracy of the power trace. However, the disadvantage of using active probes is that of price and bandwidth limitations. These probes are often significantly more expensive than their passive counterparts.

There are several potential devices to choose from that are suitable for side-channel analysis, devices with higher bandwidth are preferable, as you can capture more data from a single trace, especially when the clock speed of the chosen device is high.

Most Oscilloscopes are expensive devices, however, they are a crucial tool in hardware analysis, and can not only be used for the capturing of the power trace data but can later

Fig. 5.1 Passive Picoscope Probe

be used to analyse the operation of a target device without necessarily capturing any traces. There can be multiple advantages to this, monitoring a device which has already been profiled can be helpful in ensuring the device is executing instructions as anticipated.

Another important factor is the ability to easily retrieve several hundred acquired samples from the device, this factor naturally favours PC (or remotely) controlled devices, such as the Picosope 5.2.

Fig. 5.2 Picoscope 6400

5.1.2 Arbitrary Waveform Generator

This tool is extremely useful when performing side-channel attacks. It is seldom used in the profiling or analysis stage, however the use of such a device allows fault injection into a SoC with the goal of inducing a fault state within the SoC. This can be especially useful when used in conjunction with DPA, this is as waveforms can be directly injected into the SoC while performing cryptographic functions. The primary advantage to this is to either force the SoC into an unknown state which may allow an additional vector of attack when performing key extraction, or, the waveform generator can provide a square wave clock signal, allowing either a clock glitch attack or modification of the SoC operating speed.

For the purpose of this project, a Rohde & Schwarz AM300 waveform generator can be used to perform the operations outlined above.

5.1.3 ChipWisperer

Another tool that is extremely useful is the NewAE ChipWisperer, this tool is specifically designed for performing side-channel analysis on devices [9].

5.2 Target Devices

There are a multitude of devices available for profiling and attacking. This project aims to profile one device in order to recover the keys used within AES, with an ARM-based microprocessor. These requirements assist in reducing the number of possible devices available to test. This selection is based on the hardware requirements. For example, the device needs to have the hardware capabilities to be programmable with little effort, more specifically, direct USB programming from a PC would be preferable. The device also needs to be commonly used as a bespoke device, or microprocessor implementation may negatively impact the outcome of the project. This negative impact would be attributed to the fact that only a specific board or chip may be susceptible to the specific type of attack.

There is a further requirement of the cost that has to be considered. The chosen device has to be of an affordable price point to limit the project's capital overhead.

For the purpose of choosing an appropriate device, a set of requirements have been produced. These requirements enable the comparison of available devices and the selection of the most appropriate device. The device's requirements are as follows, Does the device have networking capabilities? What is the clock speed of the device? What is the architecture of the device? What is the package type of the microcontroller on the device? What is the overall price of the device?

These requirements each have significant importance. The device's clock speed is a significant contributing factor, as the clock speed has to be within an appropriate range to gather power trace samples effectively. The Picoscope 6000E Series has a maximum sample frequency of 1 GHz [48]. This limits the potential devices to a clock speed of the device to under 1 GHz. The architecture of the device is also essential. For ease of sampling, the device should not require an Operating System or underlying kernel, as it is difficult to predict and control other processes running "in the background" whilst trying to gather power traces. The package type of the microcontroller is another critical consideration; for example, a BGA (Ball Grid Array) microcontroller would not be practical as it is impossible to access the chip's pins directly as they are located underneath the package [3].

Device List					
Device Name	Networking	Clock Speed	Architecture / Chip	Package Type	Price
HiFive1 RevB[25]	WiFi	320MHz	RISC-V	TQFP	£70
HALJIA UNO[22]	WiFi	160-240MHz	ESP32	MCU	£7.44
Arduino Uno[4]	WiFi / Ethernet Addon	16MHz	ATMega328p	DIP	£15
Raspberry Pi Zero 2W[47]	WiFi	1GHz	ARM	BGA	£17
Raspberry Pi Pico W[1]	WiFi	133MHz	ARM Cortex-M0+	TQFP	£5.40

The limitation of the chip package may not be a sufficient reason to reject a device, this is because, with most SoCs, they can be removed from their PCB and analysed by means of direct-chip analysis. However, this may have a negative effect on the output of the analysis, this is as other components on the PCB may interact with the SoC directly resulting in a variance in the observed power trace.

5.3 Chosen Device

For this project, I have chosen to use a Raspberry Pi (RPI) Pico W for performing both the analysis and the attack. This decision has been made for a number of factors. Firstly, the device's low cost and other related hardware are required to perform the analysis and attack—secondly, the ease of programming the device. The ease of profiling the Raspberry

Pi Pico W is another key factor, all of the pins are easily accessible, which makes probing the device quite simple. Another important factor in this decision is the compatibility of the ARM Cortex chip with the ChipWisperer toolkit, allowing for much easier profiling of the microcontroller. Another advantage to using the RPI Pico W is the microprocessor itself, being an ARMv8 Cortex allows me to use other devices which use the same chip implementation to enable a comparative study of the findings.

Further to justify the decision of this device is its direct choice of the ARM SoC, as these SoCs are prolific, in that they can be found in devices of all types, such as mobile phones, embedded systems and personal computers, it would be advantageous to choose an architecture and SoC that is widely used in a multitude of applications [2].

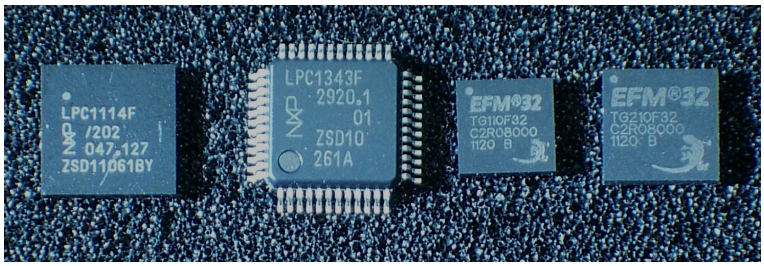

Fig. 5.3 ARM Cortex M0 SoC [59]

The RPI Pico W utilises a two-core ARM Cortex M0, as the architecture can be licensed to different manufacturers, the microprocessor isn't limited to the specific implementation restrictions that would limit the scope of this project. The Cortex M0 is manufactured by several manufacturers which all implement the underlying ARM architecture.

Chapter 6

Data Acquisition

6.1 Data Sampling

For the profiling of the device itself, a simple program is used to encrypt the data using an AES library, and approximately 5000 samples of each plain text input will be gathered to train the ML model. The device sets a GPIO pin high to trigger the sampling of trace data on the Picoscope, runs the encryption, pulls the pin low and then resets to start the process again.

This method of data sampling uses a shunt resistor to measure the current. I connected the probe to the negative end of the resistor to obtain sample data. The Picoscope, like many other oscilloscopes, can store individual samples. Because of this, it makes the collection of samples very time efficient.

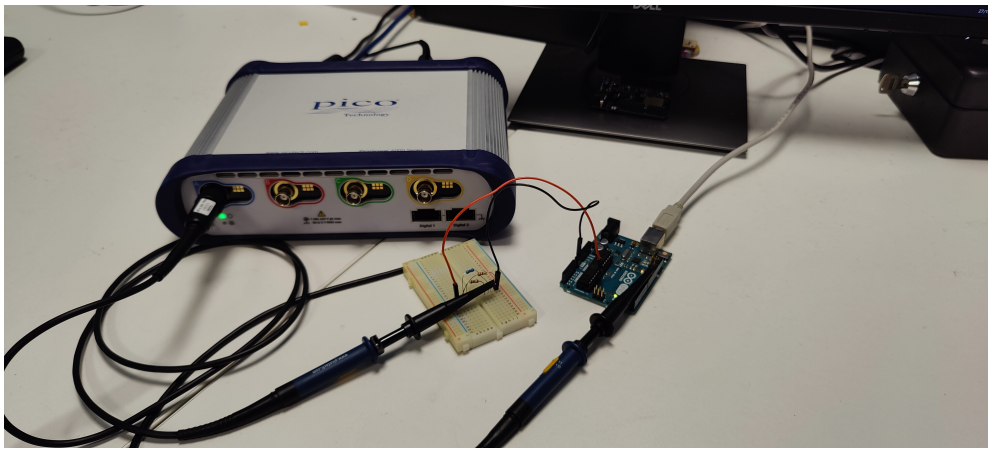

Fig. 6.1 DPA Sampling Setup

To measure the current without shorting the circuit, I am using a 100Ω resistor. As I did not have a 100Ω resistor, I used multiple resistors in parallel to gain the required resistance. The three individual resistors are 1x 2KΩ 2x 220Ω. The layout of the resistors is found in Figure 6.3. The formula used to calculate the parallel resistance is [56]:

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} \ldots$$

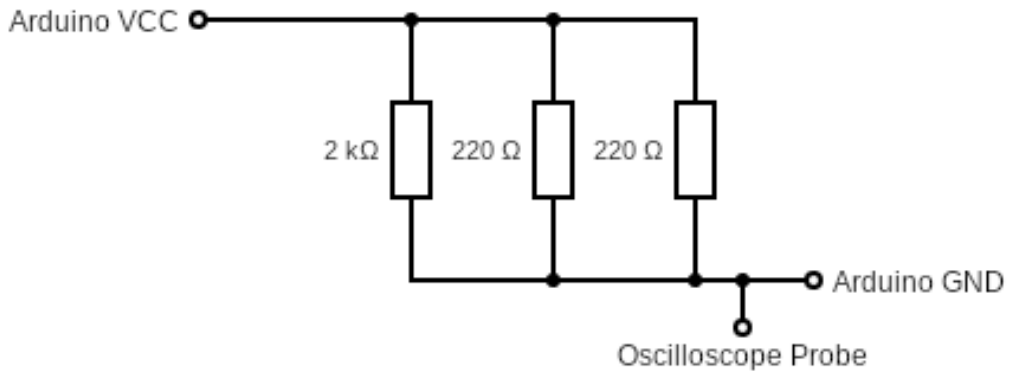

Fig. 6.2 Parallel Resistor Setup

I sampled 5000 PCCs at first, automatically using the Picoscope. This sampling figure is somewhat arbitrary. I chose 5000 samples per plain text as this will reduce the probability of ghost spikes within DPA, if, for example 1000 PCCs were use, there would be a higher probability of ghost spiking, making the results of the DPA less accurate. To reduce the effects of ghost spiking, where false positives occur within the guessing of the key, it is important that a high number of samples are taken. The disadvantage of using this many samples is the increased computational complexity during training. The profile of 5000 PCCs took approximately 5 minutes. The file size of the samples is approximately 1GB.

The code required to gather the power samples is programmed in python and compiled and uploaded directly to the device. The software implements TinyAES in order to compute the cypher. One digital IO pin is then configured in output mode, and this pin is used to trigger the Picoscope. The plain text to be encrypted is then defined, and the known key is used. The pin is pulled high (+3.3 Volts), and the AES encryption begins. Once the encryption has been completed, the IO pin is pulled low (0 Volts). The program then resets the device, which triggers the code to be run continuously. The decision was made to reset the device after each encryption cycle to automate the sampling process.

As seen from Figure 6.3, you can see the start of the algorithm at the beginning of the power trace, which is the beginning of the leakage. At approximately two picoseconds, the waveform PCC changes; this is leakage from the microcontroller during the computation of encrypting the plain text with AES. Note the PCC spike during execution, this is further leakage from the microcontroller, this leakage is the change of power consumption while executing AES. The end of the power trace marks the end of the AES computation, as sampling is halted automatically after the algorithm has finished its execution cycle.

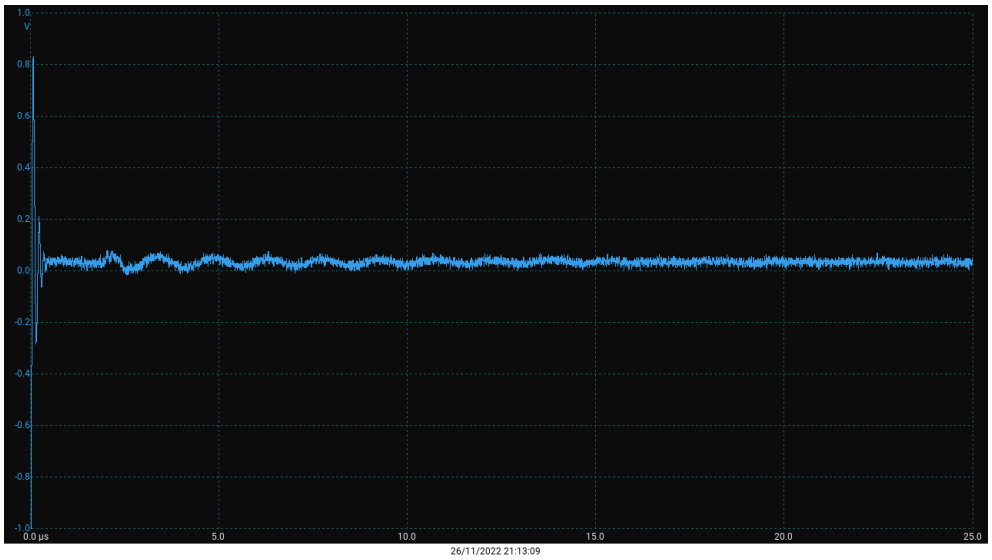

Fig. 6.3 Power Trace Sample

Each of these power samples is exported as delimited text documents. This enables importing this data into a usable form such as a numpy array. Each value within the DSV has an associated timestamp with the corresponding measured voltage.

6.2 ChipWisperer

As prior mentioned, the ChipWisperer is a unique tool enabling the collection and analysis of SoCs for side-channel analysis. The advantage of using such a tool for the collection of data and analysis of such data is the reduction in time spent. The disadvantage to using such a device is the hardware complexity of performing analysis is slightly increased, as can be seen from figure 6.4.

The J-Link depicted in this figure is not necessarily required, as this device is primarily used to monitor the SoC's state during execution, the rationale of using such a device is to monitor the state of the SoC in real-time, this can be advantageous to ensure the correct flow

of execution. This could be an important factor during analysis as during compilation, the flow of instructions may be changed in order to achieve code optimisation. A countermeasure to this issue could be to disable optimisation during code optimisation.

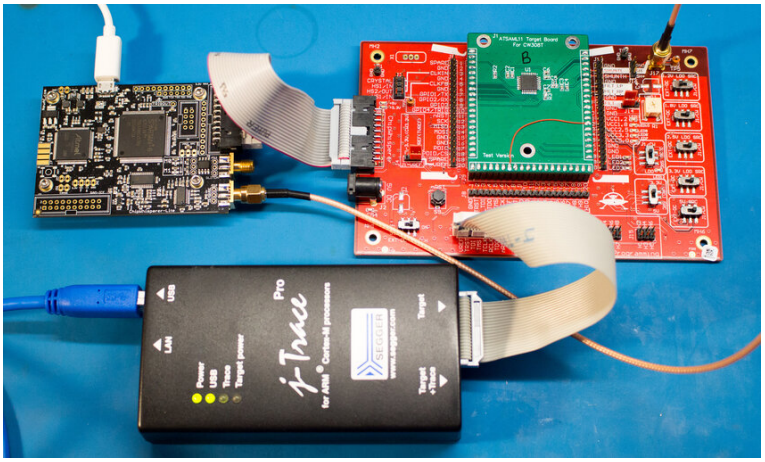

Fig. 6.4 ChipWhisperer Setup [45]

Another issue with the ChipWisperer is the reduction of software complexity as the ChipWisperer software library reduces the overall complexity of performing attacks such as SPA, DPA and CPA. However, the reduction in software complexity allows for increased data collection and processing.

As a result of access to such a device, I have also collected identical samples using the ChipWisperer, the advantage to collecting samples using this method is the ease of analysis based on the collection of the samples. These samples are stored much in the same way, using a numpy array. For the analysis of AES power traces, the same S-Box is used. A sample of a single trace collected using the ChipWisperer can be seen in figure 6.5.

This trace varies considerably from those observed within Oscilloscope sampling. This could be attributed to incorrect configuring of the oscilloscope, however, when compared statistically, the traces are both similar in modulation and variance. As outlined previously, the power traces collected are of AES128.

Using the ChipWisperer with an oscilloscope to monitor the current of the microprocessor caused an unforeseen issue, in that the acquisition of the sub-key values would be incorrect, or the results would be completely unattainable. This is most likely attributed to interference caused by either the probe or the oscilloscope itself.

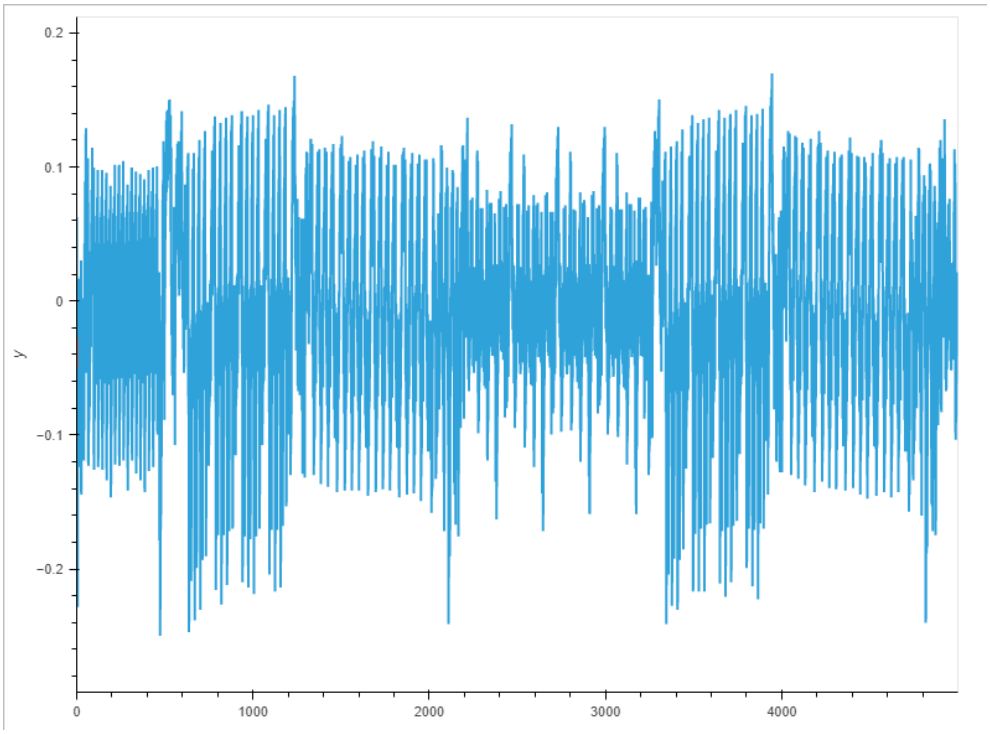

Fig. 6.5 Power Trace of AES captured by the ChipWisperer

6.3 ML-based Side-Channel Analysis

The method of using Machine Learning for developing or deploying attacks is not novel. ML has been used to develop side-channel attacks over the past decade, such as in the application of differential power analysis of microprocessors [26]. The advantage of such methods is the vast reduction of time needed to analyse vast amounts of profiled data. ML has also been used for the practical implementation of side-channel attacks, whereby a neural network can extract secret keys used in data encryption, even when masking techniques are implemented to protect against DPA [50].

More recently, unsupervised machine learning models have been used to perform differential power analysis side-channel attacks [51]. The method mentioned above used a single neural network to retrieve the secret keys from a microprocessor. A leakage model with sensitivity analysis is created based on the collection of raw power traces. This analysis allows the training of a multi-layer model for each key candidate to map the power features to the individual bits of the key. In the context of machine learning, unsupervised machine learning differs from previous techniques using supervised machine learning to profile the device under attack; an identical device is required with known secret keys. By using supervised machine learning, attackers are limited to specific devices and hardware iterations and are

not transferable out of a controlled environment, limiting the practical application of such an attack. Therefore, an unsupervised machine learning model is more favourable.

ML technique usage within side-channel analysis has gained significant research in recent years [17, 38]. One key benefit of using ML is the ability to automate various aspects of the analysis process, such as data processing, feature extraction, and classification. For example, CNNs are effective in automatically extracting features from raw side-channel data, thereby reducing the need for manual feature engineering. Supervised learning algorithms, such as SVMs and decision trees, can be trained to classify side-channel data, identifying patterns and anomalies that may indicate a side-channel attack [49, 23, 60].

This advancement of side-channel analysis assisted with the use of machine learning drastically increases the probability of reliable results within the analysis.

SCAMML is a Python library designed and developed by Google Research [7] that provides sampling methods for multi-label and multi-class classification problems found within side-channel analysis. This library builds on an existing library called scikit-learn and contains various procedures that enhance data augmentation, imbalance correction and model training.

By applying sampling approaches that consider the issue's multi-classification and multi-labelling character, the SCAMML library enables a set of sampling techniques that may improve the precision of side-channel attacks.

It is sometimes more efficient to try to recover keys without the use of such a toolkit, however, if the sampled power traces are "noisy", the advantages of using a toolkit such as SCAMML are clear, as the model is highly resilient against noise or other external factors that may affect traditional analysis.

While SCAMML is extremely useful for certain side-channel applications, scikit-learn can be use directly for use within Differential Power Analysis, this library can predict the key data using a Random Forest Classifier.

Chapter 7

Security Analysis

In order to analyse the data in the most efficient method, the data analysis is implemented alongside the collection. The primary rationale for combining both data acquisition and analysis is to reduce the time between collection and analysis. As DPA requires a large number of traces, the time to import the data once collected is substantially time-consuming.

Each power trace can be stored directly within an array, this makes the analysis more efficient than storing each power trace within a CSV or text file for example.

7.1 Initial Analysis

As can be seen from the sample in figure 6.5, pronounced spikes along the PCC can be seen when the SoC is executing AES128.

The first differential power analysis was computed with the use of the ChipWisperer, the library assists in the analysis somewhat, however, the core principles of DPA require outlining. The analysis consists of a nested for loop, which iterates over each key byte (from 0 to 255) and defines two lists. The nested for loop iterates over each trace and splits the trace into input bytes (for the plaintext input), the leakage data is then retrieved using the library's aes-internal function. This function takes the guess (which could be any value from 0 to 255) and the input byte. This function will output the leakage bit, which can either be a zero or a one. This is then appended to the lists defined at the beginning of the algorithm. An average of the ones and zeros lists are then taken and the differential value is computed as a maximum of the average of the one average list minus the zero average list.

Using this algorithm allows the iteration over each power trace collected, and the values of each trace are all appended to the lists and an overall average is returned, along with the average differential values. As can be seen from figure 7.1, the differential peak is pronounced when plotted.

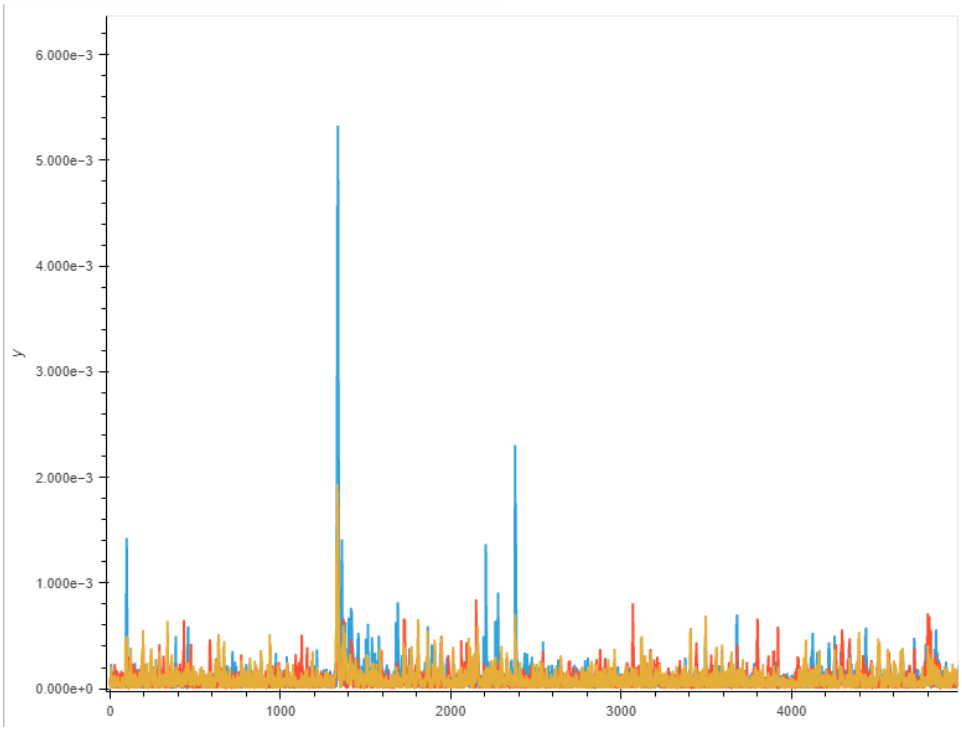

Fig. 7.1 AES128 Differential Peak

These average differential values are called guesses, as the name suggests this is the guessed value of the key. The algorithm will output multiple guesses each with an individual rank, these guesses can then be ranked in order of maximum differences. The higher the difference, the more probable the guess is to be the true key value.

This algorithm only performs DPA on the first byte of the key. However, it is not at all difficult to scale this algorithm over all sub-keys, by implementing a further for loop, iteration over all sub-keys can be computed.

7.2 Complete Key Recovery

This initial analysis was conducted with a known key and random plain text data, as this initial analysis was performed with a known key, it gave a basis for comparison between the key retrieved by DPA and the known key. The purpose of such a test is to confirm that the analysis yields correct results. To ensure the initial analysis was accurate, and the key was identical in each complete execution of DPA, this initial analysis was run 10 times. The number of tests is purely arbitrary, as the results were the same throughout each test, it is concluded that the extraction of key data, can be accomplished by means of DPA.

The plain text input is not required for initial testing, as we are only concerned with the extraction of the key and not plain text extraction the input data is pseudo-randomly generated.

As can be seen from the data below, each value of the key, split into sub-keys, which are individual bytes of the key were successfully extracted using this methodology. For the purposes of correlation between the key extracted using DPA, and the prior knowledge of the key, a visual comparison can be made to determine the correct, or incorrect, value of the extracted key from leakage. The data is formatted by the individual sub-keys. This algorithm is relatively efficient and takes approximately 5 minutes to guess the entire AES128 key from the collected traces.

```
Subkey  0 - 2A (actual 2A)
Subkey  1 - 7E (actual 7E)
Subkey  2 - 15 (actual 15)
Subkey  3 - 16 (actual 16)
Subkey  4 - 28 (actual 28)
Subkey  5 - AE (actual AE)
Subkey  6 - D2 (actual D2)
Subkey  7 - A6 (actual A6)
Subkey  8 - AD (actual AD)
Subkey  9 - F3 (actual F3)
Subkey 10 - 15 (actual 15)
Subkey 11 - 84 (actual 84)
Subkey 12 - A3 (actual A3)
Subkey 13 - CF (actual CF)
Subkey 14 - 4F (actual 4F)
Subkey 15 - 3D (actual 3D)
```

As you can see from figure 7.1, when plotted, the differential peaks are pronounced for each sub-key value. As seen in this figure, there are a considerable amount of smaller peaks within the plot, these are lower-ranked guesses of each individual sub-key. As the guesses are ranked in order from highest to lowest, the larger peaks are more pronounced.

These smaller peaks may appear, by visual analysis, to be statistically significant, whereby they have a higher potential of being the correct sub-key guess. However, when statistically observed, the numerical values are significant enough to discount them as the potential sub-keys.

In usual cases, the key would be unknown, with no basis for comparison, however, as we know the true key, we can correlate the highest-ranked guess with each true sub-key. If the key

is not known, which is the purpose of DPA, the only method of verifying the key is to decrypt the input plain text. If the plain text produced during decryption resembles understandable results, for example, alphanumeric sequences or readable text, the key recovery is deemed to be accurate.

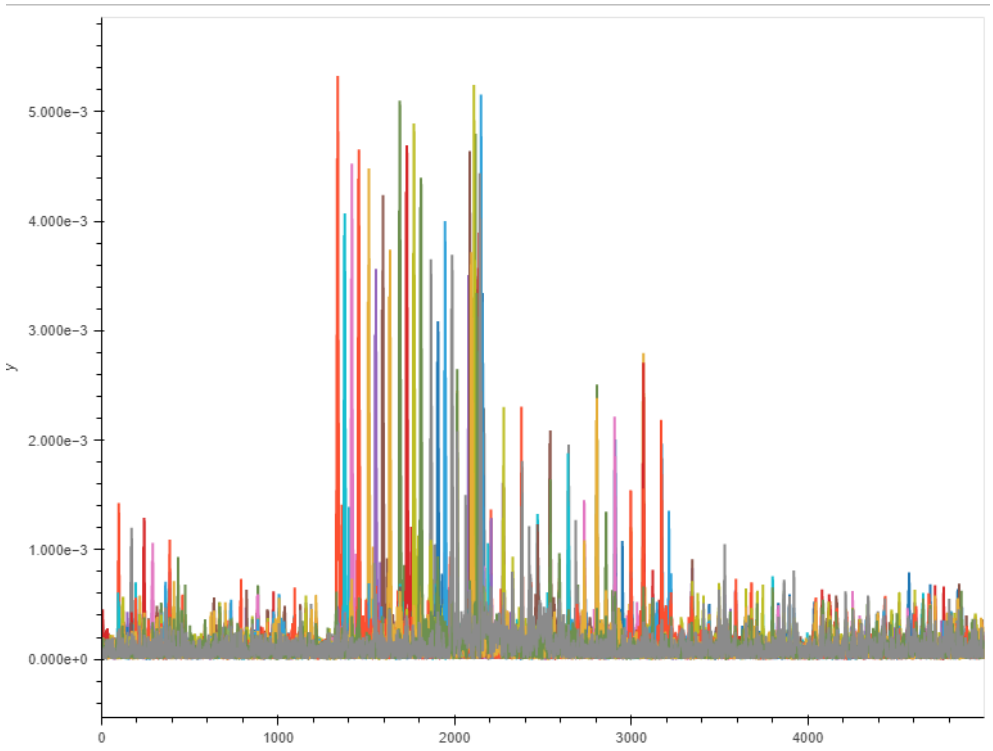

Fig. 7.2 DPA Analysis Results

7.3 Implications of Complete Key Recovery

The implications of extracting the secret keys from a SoC is significant. By having access to the secret keys, any data processed by the AES algorithm on the targeted SoC can be decrypted with little effort, simply by running the AES algorithm in reverse, with the key collected by means of DPA.

In some cases SoCs will include on-chip cryptographic sub-processors, the advantage of implementing such a feature is the reduction in software complexity required, this is as the SoC will process the encryption and decryption of data. The primary disadvantage of this choice of implementation is that if the keys are extracted from the cryptographic sub-processor on the SoC there is no ability to mitigate the vulnerability with software, this vulnerability would require SoC re-manufacture or replacement.

Chapter 8

Conclusion

Upon reflection, the results of this project would be classified as a success. The device chosen was an ARM Cortex SoC that Is widely used within the IoT sector and in other applications such as embedded systems and mobile devices. The identification of an attack vector, being differential power analysis on the implementation of AES, in this case Tiny-AES128 was successful. The extraction of the secret keys used within the AES encryption algorithm running on an ARM Cortex SoC was also successful. Gaining the keys used within AES was the limit in scope of this project, however gives the opportunity to research the further implications of gaining access to keys used in ubiquitous cryptographic algorithms such as AES.

Further research could include expanding the profiling and analysis of multiple SoCs within the ARM architecture, this would further substantiate that there are substantial hardware vulnerabilities within the implementation of ARM based chips. For example, further analysis could be performed on different variations of the Cortex line of chips.

When performing any analysis and attack on cryptographic algorithms, there is an obvious data privacy and data protection aspect to the research. While there are countermeasures in place, this project, and the additional research cited, proves that there is a continuing risk to the security of data, even when the data is encrypted with advanced cryptographic algorithms such as AES. While the algorithm itself has not been compromised, the implementation poses a significant threat to the integrity of such cryptographic algorithms.

There are however several limitations to the project. Physical access to the target device is required to both profile and attack the target, this may not always be possible if reproduced outside of a laboratory environment. The attack on performed on the SoC may also be limited to the specific chip, iteration or implementation. This would pose a challenge for further research as the methodology and specific implementation of the attack may not be successful if used on another chip, or, another architecture. Another limitation of this project is the

equipment required to perform such an analysis and attack. The equipment outlined is often expensive and requires specific conditions in order to operate effectively. For example, a field attack of a device using power analysis would most likely not be successful as variables such as the power supply stability, interference from other circuitry or countermeasures may be a limiting factor in the success of performing such an attack.

The difficulties and limitations of performing power analysis based side-channel attacks were made clear during the analysis of the target device, as simple interference from an oscilloscope monitoring the state of data output from the device caused difficulty in gaining significant differential peaks. Another difficulty in the analysis was identifying ghost peaks from accurate differential peaks, however this was reduced by increasing the number of traces collected.

References

[1] Dec. 2022. URL: https://datasheets.raspberrypi.com/picow/pico-w-datasheet.pdf.

[2] Thomas Alsop. *ARM-based chip unit shipments by quarter 2021*. Nov. 2022. URL: https://www.statista.com/statistics/1131983/arm-based-chip-unit-shipments-as-reported-by-licensees-worldwide/.

[3] *An introduction to BGA package*. Apr. 2021. URL: https://anysilicon.com/an-introduction-to-bga-package/.

[4] *Arduino Uno REV3*. URL: https://store.arduino.cc/products/arduino-uno-rev3.

[5] *Arduino Uno WIFI REV2*. URL: https://store.arduino.cc/products/arduino-uno-wifi-rev2.

[6] Eli Biham and Adi Shamir. "Differential cryptanalysis of DES-like cryptosystems". In: *Journal of Cryptology* 4.1 (1991), pp. 3–72.

[7] Elie Bursztein et al. *SCAAML: Side Channel Attacks Assisted with Machine Learning*. 2019. URL: https://github.com/google/scaaml.

[8] Shuo Chen et al. "Side-Channel Leaks in Web Applications: A Reality Today, a Challenge Tomorrow". In: *2010 IEEE Symposium on Security and Privacy*. 2010, pp. 191–206. DOI: 10.1109/SP.2010.20.

[9] *Chipwhisperer*. URL: https://www.newae.com/chipwhisperer.

[10] Jen Clark. *What is the internet of things, and how does it work?* Aug. 2020. URL: https://www.ibm.com/blogs/internet-of-things/what-is-the-iot/.

[11] Christophe Clavier, Jean-Sébastien Coron, and Nora Dabbous. "Differential Power Analysis in the Presence of Hardware Countermeasures". In: *Cryptographic Hardware and Embedded Systems — CHES 2000*. Ed. by Çetin K. Koç and Christof Paar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 252–263. ISBN: 978-3-540-44499-2.

[12] Jean-Sébastien Coron and Louis Goubin. "On Boolean and Arithmetic Masking against Differential Power Analysis". In: *Cryptographic Hardware and Embedded Systems — CHES 2000*. Ed. by Çetin K. Koç and Christof Paar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 231–237. ISBN: 978-3-540-44499-2.

[13] General Instruments Corp. 1977. URL: https://picaxe.com/docs/gi_cat_1977.pdf.

[14] Nicolas Courtois. *All About Side Channel Attacks*. URL: http://www.nicolascourtois.com/papers/sc/sidech_attacks.pdf.

[15] *CVE-2016-2183 Detail*. Aug. 2016. URL: https://nvd.nist.gov/vuln/detail/CVE-2016-2183.

[16] Joan Daemen and Vincent Rijmen. “Design of Rijndael: the Advanced Encryption Standard”. In: *Springer* (2002).

[17] Siyang Dai and Qing Guo. “Side-channel attack detection based on machine learning: A review”. In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 2604–2624.

[18] *Defense Technical Information Center*. Aug. 1966. URL: https://apps.dtic.mil/sti/pdfs/ADA365316.pdf.

[19] Espressif. *ESP32 series - espressif*. URL: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[20] *FIPS 197, Advanced Encryption Standard (AES) - NIST*. Nov. 2001. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

[21] Daniel Genkin et al. “ECDH Key-Extraction via Low-Bandwidth Electromagnetic Attacks on PCs”. In: *Topics in Cryptology - CT-RSA 2016*. Ed. by Kazue Sako. Cham: Springer International Publishing, 2016, pp. 219–235. ISBN: 978-3-319-29485-8.

[22] *Haljia Uno R3*. URL: https://www.haljia.com/products/haljia-uno-r3-d1-r32-esp32-esp-32-ch340g-development-board-dual-mode-wifi-bluetooth-4mb-flash-dc-5v-12v-with-micro-usb-compatible-with-arduino.

[23] Takahiro Hayashi et al. “Machine learning-based side-channel attack on cryptographic circuits”. In: *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE. 2018, pp. 44–49.

[24] Hellisp. *File:data ENCRIPTION standard flow diagram.svg*. June 2014. URL: https://commons.wikimedia.org/wiki/File:Data_Encription_Standard_Flow_Diagram.svg.

[25] *Hifive1 Rev B*. URL: https://www.sifive.com/boards/hifive1-rev-b.

[26] Gabriel Hospodar et al. “Machine learning in side-channel analysis: A first study”. In: *J. Cryptographic Engineering* 1 (Dec. 2011), pp. 293–302. DOI: 10.1007/s13389-011-0023-x.

[27] *How fast are semiconductor prices falling?* URL: https://www.nber.org/digest/jul15/how-fast-are-semiconductor-prices-falling.

[28] Baivab Kumar Jena. *What is AES encryption and how does it work? - simplilearn*. Feb. 2023. URL: https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption.

[29] Jakob Jonsson. “Modes of operation of symmetric key block ciphers”. In: *Journal of Cryptology* 15.3 (2002), pp. 213–246.

[30] W.J. Kee et al. “A Review on spectre attacks and meltdown with its mitigation techniques”. In: *International Journal of Engineering and Technology(UAE)* 7 (Jan. 2018), pp. 209–213.

[31] John Kelsey and Bruce Schneier. “Related-key cryptanalysis of 3-key triple-DES”. In: *Fast Software Encryption* (1996), pp. 233–246.

[32] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential Power Analysis”. In: *Advances in Cryptology — CRYPTO’ 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397. ISBN: 978-3-540-48405-9.

[33] Paul Kocher et al. *Introduction to differential power analysis - journal of cryptographic engineering*. Mar. 2011. URL: https://link.springer.com/article/10.1007/s13389-011-0006-y.

[34] Paul Kocher et al. "REGULAR PAPER Introduction to differential power analysis". In: *J. Cryptographic Engineering* 1 (Apr. 2011), pp. 5–27. DOI: 10.1007/s13389-011-0006-y.

[35] Daniel Kopp, Christoph Dietzel, and Oliver Hohlfeld. "DDoS Never Dies? An IXP Perspective on DDoS Amplification Attacks". In: *Passive and Active Measurement*. Ed. by Oliver Hohlfeld, Andra Lutu, and Dave Levin. Cham: Springer International Publishing, 2021, pp. 284–301. ISBN: 978-3-030-72582-2.

[36] Uwe Krieger. "Side Channel Attacks on Smart Cards: Threats & Countermeasures". In: *Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe 2003 Conference*. Wiesbaden: Vieweg+Teubner Verlag, 2004, pp. 73–81. ISBN: 978-3-322-84982-3. DOI: 10.1007/978-3-322-84982-3_8. URL: https://doi.org/10.1007/978-3-322-84982-3_8.

[37] Nest Labs. *Nest Thermostat*. https://nest.com/thermostats/. 2011.

[38] Prateek Malhotra and Rajat Subhra Chakraborty. "Machine learning based side-channel attacks: A survey". In: *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE. 2019, pp. 51–60.

[39] Alex Martin. *Electronics, firmware and Mechanical Design*. Nov. 2015. URL: https://martinelectronics.co.uk/wp/?cat=17.

[40] Mitsuru Matsui. "Linear cryptanalysis method for DES cipher". In: *Advances in Cryptology—EUROCRYPT'93*. Springer. 1993, pp. 386–397.

[41] Adam Matthews. "Side-channel attacks on smartcards". In: *Network Security* 2006.12 (2006), pp. 18–20. ISSN: 1353-4858. DOI: https://doi.org/10.1016/S1353-4858(06)70465-2. URL: https://www.sciencedirect.com/science/article/pii/S1353485806704652.

[42] Michael Mattioli. "The Apollo Guidance Computer". In: *IEEE Micro* 41.6 (2021), pp. 179–182. DOI: 10.1109/MM.2021.3121103.

[43] Francesca Meneghello et al. "IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices". In: *IEEE Internet of Things Journal* 6.5 (2019), pp. 8182–8201. DOI: 10.1109/JIOT.2019.2935189.

[44] Alfred Menezes, Paul C van Oorschot, and Scott A Vanstone. "Handbook of Applied Cryptography". In: *CRC Press*. 1996.

[45] Colin O'Flynn and Alex Dewar. "On-Device Power Analysis Across Hardware Security Domains.: Stop Hitting Yourself." In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (Aug. 2019), pp. 126–153. DOI: 10.46586/tches.v2019.i4.126-153.

[46] Christof Paar and Jan Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer, 2010.

[47] Raspberry Pi. *Raspberry Pi Zero 2 W*. URL: https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/.

[48] *Picoscope 6000E series oscilloscopes*. URL: https://www.picotech.com/oscilloscope/6000/picoscope-6000-overview.

[49] Partha Pramanik et al. “Exploring the use of machine learning for differential power analysis attack detection”. In: *IEEE Access* 8 (2020), pp. 75289–75306.

[50] Emmanuel Prouff and Matthieu Rivain. “Masking against Side-Channel Attacks: A Formal Security Proof”. In: vol. 7881. May 2013. ISBN: 978-3-642-38347-2. DOI: 10.1007/978-3-642-38348-9_9.

[51] Keyvan Ramezanpour, Paul Ampadu, and William Diehl. “SCAUL: Power Side-Channel Analysis with Unsupervised Learning”. In: *IEEE Transactions on Computers* PP (July 2020), pp. 1–1. DOI: 10.1109/TC.2020.3013196.

[52] Simranjeet Sidhu, Bassam J. Mohd, and Thaier Hayajneh. “Hardware Security in IoT Devices with Emphasis on Hardware Trojans”. In: *Journal of Sensor and Actuator Networks* 8.3 (2019). ISSN: 2224-2708. DOI: 10.3390/jsan8030042. URL: https://www.mdpi.com/2224-2708/8/3/42.

[53] *State of IOT 2022: Number of connected IoT devices growing 18 percent to 14.4 billion globally*. June 2022. URL: https://iot-analytics.com/number-connected-iot-devices/.

[54] Bart Stevens. *DPA countermeasures done right*. Mar. 2022. URL: https://semiengineering.com/dpa-countermeasures-done-right/.

[55] Douglas R Stinson. *Cryptography: theory and practice*. CRC Press, 2018.

[56] Wayne Storr. *Resistors in parallel - parallel connected resistors*. Aug. 2022. URL: https://www.electronics-tutorials.ws/resistor/res_4.html.

[57] The Arduino Team. *About Arduino*. 2022. URL: https://www.arduino.cc/en/about.

[58] *Understanding the differences between arm and x86 processing cores - news*. URL: https://www.allaboutcircuits.com/news/understanding-the-differences-between-arm-and-x86-cores/.

[59] Viswesr. $ARM_Cortex-M0_and_M3_ICs_in_SMD_Packages$. Wikimedia Commons, Dec. 2011. URL: https://commons.wikimedia.org/wiki/File:ARM_Cortex-M0_and_M3_ICs_in_SMD_Packages.jpg.

[60] Mingjie Zhang, Hongbing Wang, and Qi Li. “Machine learning assisted differential side-channel analysis attacks on symmetric cryptography”. In: *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 2321–2325.

[61] Tian Zhang et al. “IoT Devices in the Consumer Market: Trends and Challenges”. In: *IEEE Consumer Electronics Magazine* 9.1 (2020), pp. 27–32.

[62] Zigbee. *Zigbee specification*. URL: https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf.

Appendix A

Plan, Risk and Mitigations

A.0.1 Term 1

- Week 1: Study literature regarding ML, fault analysis and remote deployment
- Week 2: Analyse potential target devices for profiling and attacking
- Weeks 3-5: Decide on target device, write code for Side-Channel analysis using ML
- Weeks 6-7: Profile the device using ML model to identify weaknesses
- Week 8: Research into attack vectors for the device
- Week 9: Research into deployment methods over network
- Week 10-11: Prepare interim report and corresponding presentation

A.0.2 Term 2

- Week 1: Compare and analyse the approaches researched for deployment
- Weeks 2-5: Implementation of the vulnerability based on identified weaknesses
- Weeks 6-7: Implementation of the vulnerability using a network deployable payload
- Weeks 8-9: Evaluate the project results and the effectiveness of the implementation
- Weeks 10-11: Prepare final report, presentation and poster

A.1 Risks and Mitigations

All projects, of any nature, come with risk. However, the identification and preparation of potential risks will reduce the impact and mitigate the potential for risk within my project. These risks are not limited to the scope of my project, however, there are specific risks directly corresponding to my project.

A.1.1 Failure of Hardware

This project includes a large variety of hardware dependencies. Not only the device in which I am writing the report, but the hardware used to profile, deploy and evaluate. In order to mitigate the risk of hardware failure of the device used to write this report, regular backups will be made of the code and reported to an offsite location. This action will mitigate the risks of data loss due to hardware failure.

In relation to hardware failure of the target device or analysis equipment, the mitigation of having multiple identical devices to use will reduce the risk of this having a detrimental impact on the report.

A.1.2 Knowledge of Electronics and Hardware Implementation

There is some aspect of risk due to potential gaps in my knowledge of hardware implementation, electronics and hardware profiling. I understand the technical aspects in regard to programming, networking and payload delivery, however, there may be unforeseen shortcomings in my knowledge of the micro-architectural implementation. To mitigate this risk and prevent delays in my project I will undertake further research in order to be better prepared for any shortfall of knowledge.

A.1.3 Electrical Safety Risk

As with all projects including aspects of electronics, there is a risk of self-injury, or, damage to the devices themselves. In order to mitigate this risk I will ensure to comply with all health and safety legislation to protect myself from injury. I will also ensure that I am protected against damage to the target device by ensuring the device is protected against electrostatic discharge.

A.1.4 Over-running tasks and miscalculation of time requirements

In order for this project to run without unforeseen delays or overlaps in time, effective time management will be crucial. I will ensure that the time estimated to complete milestones and tasks is realistic. I will also undertake further analysis of each section to ensure that the predicted time is as close to reality as possible.

A.1.5 Computational Overhead

For any project that requires the use of an ML model, the computational overhead to run such an analysis is an important factor to consider. In order to mitigate the risk of outstretching the computational capacity to run such a model I will be using a server with sufficient resources to execute the project code.