UNIVERSITY OF TURKU

CPU Activity Analysis using RTL-SDR

Cyber Security

Master's Degree Programme in Information and Communication Technology

Department of Computing, Faculty of Technology

Master of Science in Technology Thesis

Author:

Janne Metsänperä

Supervisors:

D.Sc.(Tech) Antti Hakkala

Petri Sainio

April 2025

Master of Science in Technology Thesis
Department of Computing, Faculty of Technology
University of Turku

Subject: Cyber Security
Programme: Master's Degree Programme in Information and Communication Technology
Author: Janne Metsänperä
Title: CPU Activity Analysis using RTL-SDR
Number of pages: 70 pages, 7 appendix pages
Date: April 2025

CPU activity analysis using RTL-SDR

Every device emits signals into the electromagnetic spectrum while operating. This data can contain useful or harmful data about the device or about the information that it is processing. These emissions can be observed with appropriate tools. In this thesis the examination radiated from a processor of a computer are monitored to gain useful information about the state of the processor. The focus is not to spy, rather than plan a device that could be used in detecting malicious code running in the computer by analysing emissions received from the machine. This whole process is done using RTL-SDR receiver. Main purpose of this study is to provide proof of concept if this kind of device can be used in this kind of rather extensive data gathering task.

This study consists of two parts. In firs part that is literary review the basic principle of electromagnetic radiation is discussed in the scope of the following empirical study.

For the literary review wavelength, frequency, hertz and sampling are considered for the spectrum data acquisition point of view. The principles on how electric field is formed and what constitutes to the electric density and distribution of that field is considered, so that reader can form an understanding on how this electromagnetic field is formed. In second part of this review computer components are considered in the scope of how they produce emanations described in the first part. Mostly a processor, memory and registers are considered, they hold most of the information content and they involve in the data processing directly, so emanations send by these components should hold most of the information content that is available from the computer.

The second part of this thesis is empirical study on how RTL-SDR manages to gather useful information from the target computer and can this data be utilized in some way. First goal is to prove that emissions caused by the computer are observable. If this can be proven, then this data is collected and analysed further to gain useful information from it.

The theoretical background clarified the nature of electric and magnetic field and how these can be observed. Density of these field was also discussed as well as useful properties of the magnetic field: the skin effect and the proximity effect which should bring the magnetic field at the surface of the conductor where it should be easiest to detect.

During the empirical part of this study, electric fields related to the processor was detected and it was measured and analysed both in MATLAB and Pandas. These results gave some information about the computer, but the constraints of the measuring device prevented the data gathering from higher frequencies. These obstacles can be overcome with higher quality gauges, so that whole necessary spectrum can be analysed.

Keywords: RTL-SDR, Side channel radiation, electromagnetic spectrum, magnetic field antenna, electric field antenna.

Table of contents

1 Introduction 1

1.1 Motivation and background Error! Bookmark not defined.

1.2 Scope of this thesis 2

1.3 Structure 4

1.4 Research questions 4

2 Electromagnetism in medium 6

2.1 Electromagnetic radiation 7

2.1.1 Wavelength, Frequency, Hertz and sampling 8

2.1.2 Energy density 10

2.1.3 Coulombs field law 11

2.1.4 Antenna theory 12

2.2 Emissions caused by hardware 14

2.2.1 Hardware effects on radiation 16

2.2.2 Transferring the data: the bus 17

2.2.3 Memory 19

2.2.4 Memory efficiency 22

2.2.5 Cache memory 23

2.2.6 Registers 24

2.2.7 The processor 25

2.2.8 Instructions 26

2.2.9 Processor cycles 27

2.2.10 Processor workflow 29

2.3 Computer processes 31

2.3.1 Basic processes 31

2.3.2 Linux kernel processes 32

2.3.3 Process management 33

2.4 Summary of theoretical background of the electromagnetism 35

2.4.1 Observations on electromagnetic radiations 35

2.4.2 Observations on the radiation of the components 36

3 Measurement setup 39

3.1 Devices and software to be used 39

3.1.1 Software defined radio 40

3.1.2 Antennas 43

3.1.3 MATLAB 45

3.2 Measurement setup 46
4 Executing measurements 54
4.1 The perfect antenna 54
4.2 Usable and interesting frequencies 56
4.3 More detailed analysis of the interesting frequencies 60
4.4 Analysis of the findings 65
5 Conclusions 67
6 References 71
Appendices 76
Appendix 1 Modified MATLAB frequency scan code 76
Appendix 2 Modified MATLAB code for analysis 81

1 Introduction

Emissions caused by computers have been increasingly in the focus of computer security for some time now. This is a valid point to consider when considering that these emissions reveal information about the machine and about the tasks that the computer is processing. For example, Tempest procedures are means to mitigate unintentional leakage of electromagnetic radiation that can reveal crucial information to outsiders. These procedures can also mean protection against attacks made in EM spectrum to gain information about the device without physically handling the device.[1]

Altogether many of the parts in the computer can leak information about it like the display, keyboard, memory or processor of the computer. This is clear in the case when the component actively transmits the data like a wireless mouse or a wireless keyboard. When considering any electrical component, some radiation is formed and radiated to the environment, so every powered device transmits radiation that can be analyzed, and this information can be exploited.[2]

Another way of approaching this subject is so-called side channel attacks (SCA), which have been relevant over the quarter of the century and are still viable ways of gaining information, especially from NO OS or RTOS IoT devices.[3] There are many refined methods of interpreting power consumption to reveal processes running inside of these relatively simple and popular devices. There is a lot of research done about how these devices are hacked just by analyzing their power consumption.[4]

Although previous paragraphs present side channel radiation or altogether analysis of the emanations caused by the computer components in negative light, this phenomenon might have some beneficial implementations: Can radiation emitted by the computer be used in detecting possible harmful programs or applications? And can this be done by using unexpensive and widely available RTL-SDR dongle? Analysing whole spectrum of programs and distinguishing harmful from beneficial is quite on ordeal, even if the resolution of RTL-SDR is enough to distinguish program signatures in EM spectrum.

If we want to analyse and comprehend radiation, we need to use devices to capture emanations and device to display captured emanations for us. Capturing the signal is done with the use of antenna, where wavelength and frequency of the radiation comes into play. Certain frequency has fixed size of wave form and for capturing this we need to have certain

sized antenna. Luckily for this research, we are dealing with relatively high frequencies: this means that wavelengths are relatively small and thus small sized antennas are adequate to capture the radiation. There are quite of few nuances in antennas that affect what we can and cannot capture with them, but altogether antennas used in capturing MHz to several GHz frequencies tend to be reasonable sized devices.[5] After capturing the radiation constant analogous signal needs to be sampled before it can be accessed via digital means. Sampling theorem and sample rate is important aspect that affect how well a signal translates to digital processing and how precise the data gathered is.[6]

Although electromagnetic radiation is present constantly in our everyday life, detecting it or even understanding the nature of this radiation is difficult. It is possible to approach these via equations that concern the fundamental nature of electromagnetism. These are the Maxwell's equations as well as Ampere's, Faradays and Gauss's laws that makes it possible to understand electromagnetics. In the case of emanations from the computer components it is crucial to understand how electric field is formed, how it spreads around and how it appears to the observer.[7] Understanding these helps us to understand the phenomenon at hand; How computer components generate radiation? What can be observed from these emanations? How charge changes inside the processor or the memory can be detected from afar? Even though these phenomena are important to be considered, in this thesis these theories are not expanded upon further, and the reader is referred to relevant literature for an in-depth explanation.

1.1 Scope of this thesis

Relevant point of interest is that how a computer operates, how does it form this radiation mentioned? Although theory of computer is well explained in the literature, here the focus has been put to the aspect of emanations of the component and processes. A computer universally has certain parts, that have tendency to function in certain ways, which contributes to the radiation pattern that this component generates. For example, a processor operates at certain way to process commands included in the program. Understanding this process can be crucial in interpreting the computer state from the radiation that is generated during the process. Also, other components contribute to cumulative emanation amount in the computer: these needs to be considered when analysing the collected pattern.[8]

Development of RTL-SDR device begun in 2012 and the idea of the receiver is based on DVB-T TV turner. Group of technology enthusiast found out that this TV receiver outputs frequency data. This makes RTL-SDR an inexpensive but rather useful software defined

radio, the cost of a unit is around ranges from 10 € up to 50€ depending on the quality of the device. This makes the possibilities with RTL-SDR quite wide: with cheap price it can be deployed easily. Community around this device has developed a lot of useful applications ranging from tuner programs all wide variety of scanner applications. While RTL-SDR is a versatile tool, it is not a transmitter, so one can only receive signals with it.[9] Strengths and weaknesses of this tool are discussed with more detail later in this work.

While side channel radiation is widely researched topic, there seems to be more to find out in this non-invasive and potential way of interpreting about the processes inside the computer. While many of the published publications focus on the privacy and data protection view, there might be useful means to employ analysis of the side channel radiation to more beneficial use. The initial plan was to construct a malware detecting tool that utilizes RTL-SDR dongle for identifying malware or viruses from the traces of side channel radiation produced by the processor when it is processing the harmful code.

Developing a rather inexpensive device that can sniff out harmful content from the radiation that it produces could be a useful tool for machines that run a rather uniform tasks, such as certain servers and industrial scale routers. Resources of the utility device could be all directed toward actual work and sniffing device that is attached to the processor of the computer could take care of the harmful content. This is at least the ambitious goal. Rather useful application for these types of computers would be the overall ability to identify if changes on their working patterns: this kind of change could indicate changes in the processing cycle and reasons for these changes should be investigated. Is the reason a software update, malfunction or malicious attack, the owner of the device probably would like to know and react to this.

There a several limitations for this work. For the theoretical part of this work where principles are explained, equations are usually presented, when this is necessary to explain the phenomenon. Further proofing or calculations with these equations with real values are usually excluded, they are not in the scope of this work. Only the most significant equations are presented here, just to clarify the phenomenon. Descriptions of the components focus on their impact on their fingerprint on the electromagnetic spectrum that they generate and on the hypothesis on how they reveal the information about process at hand. While general description about the function of each component is provided focus is mostly on this

information yielding and radiating components with the focus on about the information they can reveal.

The limitations on the field study part are that only RLT-SDR sensing device is used. This can be considered as a proof of concept on RTL-SDR capabilities in this particular case. In some cases, a commercial oscilloscope would have solved some issues better, but the decision was made to use only RTL-SDR device for this research. Linux machine was selected as target machine and for this research only one Linux target machine was used. Components are not isolated so mixed emanations will be preset and mostly radiation cannot be distinguished based on the source. Hypothesis is that most of the radiation captured comes from the processor: The antenna is kept in proximity of the processor during tests. Code for processing the data is not the focus on this research, so mostly existing code will be utilised. If changes are made to the code, it is mentioned in the text.

1.2 Research questions

Chapter 5 has the conclusions and answers to the research questions, excluding those presented in chapter 2.4. The research questions are:

1. How electrical radiation manifests in high frequency ranges and how it can be measured?
2. What kinds of mechanisms cause computer to leak information and what can be interpreted from this information?
3. Can RTL-SDR collect useful data from the computer: Is the resolution enough to gain usable information?
4. Can this data to be used in detecting different kinds of activity in the computer, can it detect processes of the computer?

1.3 Structure

This research consists of two parts: At first part rather short literature review is executed on the phenomenon itself and on the parts of the computer and on how they operate. This will be presented in chapter 2 of this work. Subchapter 2.1 will concentrate on the electromagnetic medium and on the basics of the radiation. Subchapter 2.2 will concentrate on the hardware of the computer and how they produce useful radiation in the scope of this work. Subchapter 2.3

concentrates on the core processes of the Linux machine. Findings from this part will be summarised at the end of the theoretical review in Subchapter 2.4.

The later part will consist of the setup for the empirical study and about the devices used, measurement setup and from the results of this study. In Chapter 3 the setup of the study is described including the devices and software that is used. Subchapter 3.1 describes the devices and software that is used during the measurements. Subchapter 3.2 describes the actual setup of the measurement, and the initial results of test scans are introduced.

In chapter 4 the actual measurement is conducted. First at 4.1 antennas are compared and their properties are analysed. In 4.2 Whole spectrum is scanned and points of interest from chapter 3 and findings from chapter 4 are selected for further analysis. In subchapter 4.3 these interesting findings are analysed further, and their nature is briefly discussed. In 4.4 these findings are further analysed and presented.

Chapter 5 consists of the conclusion of the empirical study and about some improvements of the study as well as further research suggestions.

2 Electromagnetism in medium

In this chapter Electromagnetic radiation is considered and discussed in a scope of this thesis: How radiation forms, how does it appear, how it can be measured? Measuring EM spectrum is considered; what factors should be taken in the account during measurements? Tools that are needed to measure the radiation emitted by the processor and other related components are described in later chapters. Focus of this chapter is to introduce unintentional emanation leaks from computers components, which should be relevant in obtaining information about processes inside the computer. With the ideas presented in this chapter one should be able to answer the question: how this side channel radiation should be measured so that we can gain useful information from analysis of this leakage.

Origins of EM emanations from a computer can be intentional emissions, like WIFI connectors, Bluetooth connectors or similar data transmitters. While these emanations can be exploited in data gathering, this work does not concentrate on these intentional and useful signals. Scope of this work is to concentrate on unintentional transmissions formed inside the computer during normal operation of the machine. These unintentional and hard to avoid emanations are generated when current flows inside the device, and the device conducts, in the case of computers, data processing procedures. While these procedures are executed, currents are formed on all contacted unprotected surfaces. These surfaces function as antennas to project emanations further as electromagnetic radiation or emanations.[10][11]

Emanations produced by processor, bus's or memory circuit are the most interesting when considering the task of gathering of the data from ongoing computing processes.[11] When considering EM emanations, a computer operates on two basic principles: data is stored in flip flops as 0: s and 1: s and logic gates mandate how this data is processed. Both operations require current for them to happen. While required current is miniscule it can be detected with precise enough tools. In simple terms memory operations are changes in the state of the memory circuit which is done with alternating current of said circuit. Logic state changes are controlled by a clock, so changes are tied to the clock frequency of the device. Each component produces specific kind of emanation based on the process it conducts, so combining these enables us to interpret the actual process running inside the computer.[10]

Detecting SCR is at the core of this work. There have been clear indications in previous articles that SCA is measurable and that there are clear implications between SCA and the

processes going on inside the computer. SCA has been a point of interest among many researchers during recent years, Google scholar yields about 13 500 results with “side channel radiation” search phrase. Interest in this field begun when Kocher released his conference proceedings “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems.” in 1996.[3] During the last 25 years side channel radiation has been studied in detail. Many of the studies seems to concentrate on breaking the cryptographic procedures especially in small IoT devices, where computational complexity is limited.[12]

Measuring electromagnetic radiation caused by a computer accumulates a lot of raw data. Analysing this data without proper tools is a time consuming and often fruitless task. With recent advances in machine learning and especially machine learning tools makes it possible to handle all of the collected data. These tools can be used to find anomalies or other deviations in the standard which might indicate problems in the process. Or as in the case of this study, it might indicate that someone is trying to execute malicious operations in the machine.

2.1 Electromagnetic radiation

Although mathematical approach is not at the core of this thesis, it is useful to understand the basic principles governing phenomena that cause radiation. Understanding these basics helps us to understand results obtained from devices used to measure this radiation enable us to execute observations made in this thesis. Electromagnetic radiation is studied widely and thus it is a relatively well-known phenomenon. In this chapter fundamentals of physicks behind radiation are discussed briefly without going too much into the details.

In this subchapter laws and principles related to this study are briefly discussed in context of this thesis: Gauss’s law which can be used assess the electric field on the surface, Coulomb’s law which can be used to assess force between two points and related electric fields, Superposition principle to further understand that electric field is product of multiple point charges. Finally, the principle of conversation of charge tells us that in isolated systems electric charge never changes. With these theories most of the fundamentals of emanations should be explained in the scope of this work, so that we can understand what kind of radiation can be observed.

2.1.1 Wavelength, Frequency, Hertz and sampling

Electromagnetic radiation is a phenomenon that causes a disturbance to medium, that consists of electric and magnetic components. When travelling through space this phenomenon causes disturbance which can be measured. Electromagnetic radiation travels on the waveform with constant speed, which is the speed of light. It is worthwhile to mention that the speed of light depends on the medium in which the light is traveling. In electromagnetic radiation wavelength is in relation to velocity and frequency, which can be expressed with formula:

$$c = \lambda v$$

Where c=velocity, λ=wavelength and v=frequency. Because velocity is constant, wavelength and frequency are inversely proportional. So, higher the frequency, smaller the wavelength is and vice versa. With this information we can assume, that we need suitable equipment to be able to listen frequencies that functions with desired frequency areas and thus are able to collect wavelengths of these areas.[7] I will go more in detail about how processors operate and how these principles effect on measurements. Computer frequencies are described with more detail in chapter 3, where study's target computer is introduced.

Correlation between wavelength and frequency is straightforward phenomenon, which sets some basic requirements for our test setup. As seen with these calculations, antennae's that are capable of operating around these wavelengths are needed and a radio that can handle the dataflow of several gigahertz per second. I will describe equipment used in experiment more in detail in later parts of this work, including how antennae are set up.

The use of 3 m long antenna to capture ~100 MHz emanations is rather impractical and not necessarily mandatory. Lower frequencies can be captured with shorter antennas, but the gain value might suffer. Still with these values all the emanations from the measurement range should be capturable.

A few words about Hertz mentioned above. Hertz is a SI approved unit for describing frequency. Hertz describes how many cycles or changes phenomenon has. With computers these are often with mega (10^6) or giga (10^9) prefixes, because of the high frequencies modern computers operate on. Frequencies come into play in this work especially because we want to store and operate on signals. For us to be able to process collected data we must make sure, that we sample the data with enough samples so that further analysis is possible. Possible operations might be for example Fourier analysis to interpret frequency components

present in signal. For this purpose, we can use Nyqvist theorem of maximum frequency to calculate the minimum sampling rate of the signal. To achieve this, we need to decide the maximum frequency f_{max} that still has useful data and then sample twice the frequency of this target frequency:

$$f_{sample} \geq 2f_{max}$$

With this frequency we do not lose any relevant data, and thus we are able to process the data as we desire.[6]

For this work there are additional benefit from quadrature sampling while executing experiments. For this research, the device used can only measure up to 4 MHz at the time, but still, it can sample 4 MHz range. This is done by down sampling the received signal and then dividing the signal based on Eulers formula to its real and imaginary components which are the sine and cosine component of the signal. These can be sampled individually so that we can exploit the whole captured frequency range, not just the half of the sample suggested by Nyqvist. Downside of this technology is, that the capture of the whole frequency range needs to be done in less than 4 MHz slices.[13]

In scope of related experiment, it is crucial to be able to collect data with adequate details so that the data contains enough information for successful analysis. We need to get to certain level of detail to interpret useful information from the data. Many of the operations that are relevant to this study happens at high frequencies with low amplitudes: they happen fast, and they cause weak signal. There are limits on what we can detect: Precise tools with wide scope are expensive and they produce a lot of data, which in turn need a lot of fast storing space to store. And a lot of computing power to analyse. With limited resources available to this study, it is necessary to find relevant frequencies which are to be monitored with relevant bandwidth and relevant sampling rate.

When a computer is turned on, certain operations happen at certain frequency rates. The frequencies employed by the computer vary from higher end frequencies as the highest frequency of the processor which in this case might be around frequency range of 3,5 GHz. Still there are several other frequencies that can be detected, like the frequency generated by the clock of the memory blocks which might be around frequency range from 1 GHz up to 2 GHz, or the base frequency of the processor, which can be around frequency range of 100 MHz. Some of these should be easily detectable with the equipment at hand, others might

vanish in the higher frequency range, that is not observable with the selected equipment. Altogether only sum of all emotions can be collected, as explained later in this chapter.

2.1.2 Energy density

Energy formed in electromagnetic radiation is carried by particle called photon. These particles are massless, and they can convey only limited amount of energy, which is proportional to frequency of the radiation. Energy transmitted is dependent on the frequency of the radiation. Electromagnetic radiation is described as sine wave, where electric and magnetic components correspond to each other. Energy density of the wave can be calculated from a formula: [7], [14]

$$\omega_E = \frac{1}{2}\varepsilon_0 E^2$$

Where ω_E is the energy density, ε_0 is common electricity constant and E is strength of the electric field. Magnetic field can be calculated with formula:[7], [14]

$$\omega_B = \frac{1}{2}\frac{B^2}{\mu_0}$$

Where ω_B is strength of the fields magnetic flux and μ_0 is the magnetic constant. Densities of the magnetic and electric fields can be measured as sine wave, which periodically go to zero, and combined field can be calculated from the formula:[7], [14]

$$\omega_E = \mathrm{c}\varepsilon_0 E$$

From these calculations we can conclude that electromagnetic radiation carries energy and from energy it is possible to interpret frequencies that has caused this radiation. So, the captured radiation includes all the information necessary to recover frequencies and amplitudes that are formed when target device has been active for further analysis.

It is noteworthy to notice a small difference in electric field and magnetic field. While these are manifestation of the same phenomenon, they differ slightly, and they can be observed separately with specific instruments. More of this will be explained when antennas are considered. When considering the energy density, we can notice that strength of electric field is squared. We can also presume, that electric fields that form from processor circuits are weak. This topic will be expanded upon later in this thesis in the chapter 2.2. With this information we can conclude that we need to be able to detect weak magnetic fields, which

forces us to make special attention to the antennae and the sensitivity of the sensing equipment, which is software defined radio with RTL-SDR dongle.

2.1.3 Coulombs field law

When considering strengths of charges across medium, we can use Coulombs field law to conclude that individual charges are present with a distance and thus we can conclude that they are measurable from certain distance of the source of radiation. Coulombs field law can be expressed as follows:[7]

$$E_{(r)} = \sum_i \frac{q(r)'_i}{\left|r - r'_i\right|^2} e_{r-r'_i}$$

In this equation r' denotes the source point and r denotes the observation point. In this theoretical review these points can be regarded as point charges, so they lack spatial extensions, thus being dimensionless even in reality both will have real dimensions. We will also mostly be interested in scalar values of currents in observation point so directions of vector field are not concern of this work. We will mostly concentrate on magnitudes of charges at observation point.

As mentioned, we are not interested to measure the electric field generated by a single circuit to function. We cannot predict which circuits inside the processor are involved in certain operations, so measuring only certain circuits would provide us a little if any useful information. So, we are interested measuring radiation from all the circuits in processor while a computer is carrying out its tasks and from this data we try to form a clear picture about what is happening inside the computer. For this task Coulombs field law is a great news; single emanations generated by multiple single circuits form a larger field which we can observe.

Another important aspect from Coulombs field law is the relevance of the distance in measurements. We should get as close as possible to be able to collect most of the information provided radiation generated by the function of circuits inside the processor. To get the antennae as close as possible should give us the best results. At this moment this claim is the goal of this work. Though detecting patterns of radiation from the processor might be a relevant study for further analysing this phenomenon as whole.

2.1.4 Antenna theory

In this subchapter receiving antennas are discussed, we are not interested in transmitting antennas in the scope of this work. It is worthwhile to keep in mind, that antenna's main function is to receive disturbances in electromagnetic medium and concert them into a voltage, that can be further analysed with measuring devices. These are introduced with more detail in later chapters of this work.

As stated in previously, electromagnetic radiation can be divided into the electric and magnetic components. While most of the practical solutions exploits electric component of the radiation in both transmitting and receiving, magnetic component of the radiation can be useful in certain situations. For example, when one wishes to measure precisely radiation caused by certain smallish object inside the computer. Short introduction of theory in both cases is presented here, so later choices with equipment can be justified.

For a proof for differences in magnetic and electric fields, we can consider electric fields around static charges. In actual experiment electric field is dynamic, but differences in electric and magnetic fields remains the same and these can be observed from the theory considering static charges:

First, Faraday's and Gauss's laws for static charges in vacuum are:[15]

$\nabla \mathrm{x} \bar{E} = 0$

$\nabla \cdot \varepsilon_0 \bar{E} = \rho$

These can be rewritten by noting $\bar{E}$ with Φ meaning scalar electric potential measured in volts. Φ_p is charge (Coulombs) and by using Gauss's divergence theorem and we can form an equation describing electric potential generated by arbitrary charge distribution:[15]

$$\Phi_p = \oiiint [\rho_q / (4\pi\varepsilon_0 r_{pq})] dv$$

Where $\rho_q$is arbitrary charge distribution, r_{pq} describing distance of charge.

Same formulation while considering magnetic field generated by static charge, we need static Ampere's law and Gauss's law:

$\nabla \mathrm{x} \bar{H} = \bar{J}$

$\nabla \cdot \mu_o \bar{H} = 0$

Substituting Gauss's law with Amperes law we get similar expression as with electric charge:

$$\bar{A}_p = \oiiint [\mu_0 \bar{J}_q / (4\pi r_{pq})] dv$$

Here $\bar{A}_p$ can be analogous with Φ_p. As we can observe from these equations, while there are a lot of similarities considering electric and magnetic component, there are small differences between these two. In both equations the aim is to find the potential of each component, both magnetic and electric. Distances are also considered similarly as the attenuation of the signal. Most notable difference is the constant term, which resides in the numerator when considering magnetic component and the value of the vacuum permeability is about $1{,}257 \text{x} 10^{-6}$. In the electric component this constant resides in the denominator. Also, the value of the constant differs, electric constant being approximately $8{,}854 \text{x} 10^{-12}$. So there seems to be clear difference in mathematics between calculations when considering magnetic and electric component of the signal.[15]

Magnetic antenna is sensitive to magnetic field, not to electric field. Magnetic antenna is usually made by winding copper wire around a frame. This frame can be either nonconductive material for air-core loops or some ferromagnetic material for ferrite loops. Thios can be seen in the relative permeability term of the antenna; in air-core loops the value is 1, in ferrite core loops it depends on the ferromagnetic properties but can be at the value can be up to thousands meaning that the effect is much stronger. In As a result, when magnetic field affects this antenna, the antenna produces voltage which is proportional to this field. The received amplitude of the field depends also on the angle of the magnetic field and the normal of the frame of the antenna, closer than these two are, or smaller the difference is, the stronger the amplitude of the electric current is. Permeability of the antenna material affects the outcome also, if other material than copper is used.

While magnetic loop antennas function as good voltage generators, there are some additional benefits when using magnetic component rather than electric component. These are skin effect and proximity effect which can be very useful when one tries to measure exact piece of equipment in environment that holds several radiating objects.

Skin effect results in higher frequency emissions that suffer from higher resistance of the medium causing the emanation to be non-homogenous. This results the current density to be greater at the surface of the medium rather than closer of the core of the medium. This makes it possible to capture larger number of emissions from the proximity of the object, so even weaker emanations should be easier to capture. This should prove to be useful in the scope of this research: the goal is to capture radiation emitted by the processor, without shielding other sources, such as memory or motherboard outside the measuring area.[15]

Proximity effect is another interesting phenomenon and quite useful in the scope of this work. Basic effect of the proximity effect is, that it causes non uniform distribution of current in the medium. In proximity effect causes the current to flow away from nearby conductors and the magnetic fields that they generate. For example, if cable wiring is done tightly, magnetic field tends to spread outside, in the direction where there are no magnetic fields. This should apply also in the case of the computer processor: The radiation should flow more easily out of the surface of the processor rather than inside of it. Again, this is beneficial because it causes stronger currents outside of the processor which should be detectable from outside in more easy manner.

2.2 Emissions caused by hardware

In this chapter previously mentioned principles about theories of electromagnetisms are considered as part of the components of the computer. First here are a few notations about basic transistors and circuits. after this hardware that forms the computer is introduced, based on their relevance in interpreting processor tasks. Most of the peripherals are left out, only essential components are described here. In the second part other components excluding the processor are introduced briefly. While these are not the focus of this study, they still affect on the measured radiation levels. A short estimate on these components effects on radiations are given. At last part a processor is introduced with more detail about its structure and functionalities.

Computers use electronics to operate, data is stores and programs are written using ones and zeros to convey commands or information. In the physical world this operation is done using electric current to operate transistors, which can either hold a value or perform an operation based on desired values. When the state of the transistor is changed by altering the current of the transistor, either current is turned on or off. These small changes in currents cause changes to the electromagnetic field around them. Though the change is diminutive considering only

one transistor, the number of transistors is large, so combined effect causes measurable changes in electric and magnetic field, which can be observed from a distance as described in previous chapter when considering electric current and its appearance. [16]

When processor processes different tasks, power consumption varies based on complexity of the task and based on amount of data to be processed and different amounts of transistors to be utilized at each moment. If we consider about Ohm's law, where I is current, V is voltage and R is resistance:

$$I = \frac{V}{R}$$

And, if we assume that resistors are relatively stable, meaning that their resistance properties do not chance much during operation, we can calculate current flowing inside the device. With resistance being stable, voltage is directly proportional to the current flowing inside the device. Changes in voltages over time causes variations inside the device.[17] These variations in current flow causes different surfaces to reflect radiation around, which can be observed. With precise measurement tool we can capture this radiation and analyse changes in these radiations. When processor is processing the data and utilizes different kinds of commands small but detectable changes in surrounding electromagnetic spectrum are formed.

When considering power dynamics of the computer, up to date power architectures utilize dynamic voltage a frequency scaling, which enables the computer to operate on optimum power consumption range to avoid excessive power consumption when working on idle or simple tasks. Basic dynamic power equation is $P=\frac{1}{2}CV^2f$, so in power consumption in mind, a small decrease in voltage results in square savings in power. Reducing power often slows transistors, so usually we end up slowing clock frequencies also, to ensure robust operation of the computer. This causes detectable frequencies to shift, when computer adjusts dynamically voltages and related frequencies depending on workload so these changes in emanations enables us to predict processes inside the computer.[18]

If we consider integrated circuits which uses operational amplifiers as primary active devices we can form different kinds of amplifiers, diodes and bipolar junction transistors as building blocks we can manufacture devices that can perform wide variety of tasks as: signal processors, communication circuits, analog/digital converters and circuits that perform mathematical operations. Currents running inside these small devices can be calculated with

relative high precision, so we could calculate voltages radiated by these devices and thus get precise effect that they cause to electromagnetic spectrum.[17] These operations are done in different kinds of circuits. Most common of these circuits are flip flops, latches, multiplexers, RAM cells, CAM cells and logic gates.

Flip flops and latches can be used to store and to states. Inputting a signal can either read or alter the state of these circuits. Multiplexers are used to forward two or more input signals into one output line. These are often used with demultiplexer do decode information back to earlier form. RAM cells are used to store information. Storing information requires the use of high or low voltages to set the memory cell to store correct information; 1 or 0. CAM cell memory or associative memory search input signal from all the stored data and returns the address of matching data. Logic gates perform logical operations, Boolean functions which are the basic operators for all computing.[19]

Similar calculations used with operational amplifiers can be done with transistors. Current changes inside transistors can be derived from certain equations and thus current changes can be calculated. This becomes increasingly difficult when complexity of the microprocessor increases.[16]

Both transistors and circuits utilize current to perform they tasks. With each process these components perform, they also release a small emanation, which can be observed. It is a doomed task to try to survey a single component, but observing total radiation emanated by processor might yield us data about performance and processes of the processor.[16]

Rather than trying to interpret radiation caused by a single component it might be more feasible to concentrate on total radiation of the device rather than trying to interpret a function of a single component from total radiation. Even separation of the components might be possible it might not yield us much information. A better approach would be to collect total radiation and to analyse how this relates on processes inside the computer.

2.2.1 Hardware effects on radiation

To function a computer needs several mandatory components: A mass memory to store data, a bus to transfer data, a processing unit to process given data, registers for short term data storage and a printer of sort to give away obtained results. Usually input device is also required to control the computer. With these components a simple and operational computer can be constructed.

Power supply converts alternating current from 220 V input to the direct current which is used by the computer. This is also first source of the radiation; alteration causes radiation to spread around the device. Modern computers use 3,3 V, 5 V and 12 V currents which are generated by power supply and transmitted via wiring. 3,3 V and 5 V is usually for digital circuits, 12 V is usually for fans and motors. These both generate radiation when operational.[20]

The motherboard is a key circuit where many of the components are fastened. It also distributes the power provided by power supply to each component depending on the correct voltage level that the component needs. A motherboard has several connectors and related bus's running in it, that transfer signals across computer. A motherboard generates several levels of radiation and because CPU is usually fastened directly to motherboard, these can intervene heavily when we try to take precise measurements from CPU.[20]

Expansion cards are typically connected directly to motherboard, to ensure rapid communication rates and adequate power supply. These are usually graphic cards, sound cards and network cards. In newer computers sound- and network cards might be integrated directly to motherboard. Also separate graphic card is not necessary, if heavy graphical processing is not required, which is often the case in office computers or server computers. Still high-end graphic card has own processors and memory and has high power consumption, which generates significant radiation source. Wireless network adapter naturally radiates on set frequencies. Otherwise, generated radiation depends on purpose of the expansion card.[20]

Estimating motherboards yield on the general emanation levels is a rather difficult task: on the one hand this emanation level depends on the devices connected to motherboard and on the level of load they are experiencing at the moment of measurement. Emanations caused by bus's can be estimated more clearly, they are considered with more detail later in this work. Overall effects of the motherboard emanations remain to be estimated when considering the results of measurements.

2.2.2 Transferring the data: the bus

For a computer to operate it needs several components to perform any useful tasks. These components cannot be stacked together, mostly because of the heat dispatching reasons. For the components to operate, a data transport method is needed: This method is called the bus. Bus is required to transfer enough data with high speed from and to the processor and to adjacent devices in the computer. The bus is used in executing input/output (I/O) operation of

the processor and adjacent or even external devices. Detailed description about the traffic management and overall bus architecture is left out from this work, only basic details are explained that are necessary to estimate emanations caused by the bus.[21]

The processor or other devices are not in direct contact with bus, rather they are connected to specific interface chips that has the logic to handle the actual transaction between the host and recipient. For example, a processor just provides the necessary data and hands it to interface chip, that handles the actual exchange. The real speed of the transfer depends on the purpose of the connected device. System bus is very fast line, that operates between cache memory, actual memory and the processor. Local bus is between peripheral devices and between RAM or storage memory, with slower speed. These can be connected via bridges, so a bus system usually consists of all the data paths inside the computer. Some of these are implemented directly to the motherboard, others are plugs that can be used to connect other devices, like hard drives or optical drives etc. to the computer.[21]

Structure of the bus is a bit more complex than just a single connector going between component interfaces. Actual bus consists of three data transfer lines and two control lines. These data transfer lines are the address line, the control line and the data line. Control lines are interrupt line and arbitration line. A data transfer line is used to carry the actual payload from the memory to the processor or other device of the computer that is exploiting the data. Address line is used to carry exact information of data in the memory or exact address of the device. Control line is used to control what action is being done, for example: read or write from the location. Interrupt line is used to ensure the system doesn’t end up idling on a failed request. Finally, arbitration line is used to control who has got the right to employ the bus line at any given moment.[8] Read and write lines can be expected to behave in rather consistent manner, interrupts should be unintended and thus hard to predict.

The bus is interesting part of the computer in the scope of this work: It needs electricity to operate and signals travel in the pace of the processor. Radiation emitted by bus operation can be observed by the means of this research. Information gathered by this mean can provide useful information in the research part of this work. While bus frequencies can be analysed, it might be difficult to distinguish bus emanations from certain from other sources.

2.2.3 Memory in general

While processor is the main workhorse of the computer executing tasks appointed in processing data fed to it, this data is stored in the memory and transferred via buses. This data is stored in the memory of the computer, and it carries a lot of significance when considering about the actual task that the computer caries over. A processor and a memory are hard to distinguish; in the processor there are several memory components that operate with same pace as the processor does. These registers and caches most likely will contribute to overall emanations that the processor causes when operating and are most probably detectable when the computer is running.[8]

There are several kinds of memory types that are needed for the computer to operate. While mass storage of the data can be rather slow, it needs to have a lot of storage space and adequate transfer speed. The closer we get to the processor the faster the memory needs to be. Processor and its operating principles are introduced in next subchapter with more detail, but if we consider a modern computer with processing speed of 3,2GHz, it executes 3,2 billion cycles per second. This means that latency of the closest memory unit must be around a few nanoseconds so that a processor doesn't need to idle extensive periods while instructions or data is being fetched from the memory.[22]

Manufacturing a very fast memory (memory with very small latency) is expensive and fast memory might lack robustness needed for long term memory, so only certain registers and caches are equipped with this low latency memory.[23] While there are several ways of grouping memory types, in the scope of this work can be considered either immediate access memory and secondary storage memory. Secondary access memory is usually RAM memory or hard drive memory. Primary access memory is usually cache memory and processor register memory.[24][25]

The backbone of modern computers are large mass storages for the data. Because of the fast advances in semiconductor technology during recent decades, most modern computers these bulk data storages are Solid-State disks (SSD). SSD: s are non-volatile memory, which means that they do not need power to retain stored data. This have been achieved with floating gate transistors, that can keep their state stored for decade or even longer. In the scope of this work, SSD: s are not very interesting. First, it might be difficult to obtain any useful data from them with the tools used. Secondly SSD: s are usually placed further away in computer casing, they are rarely right on motherboard or processor.[25] [26]

Because SDD: s does not need constant power to maintain their state, SCA performed on them might not yield much useful information. Also, the vast size of SDD, it might be useless to try to interpret the data content of the drive with analysing the emanations that it emits. Only usable information form SSD might be that it is active, perhaps the activity level of the device and with precise enough tool the location of the drive that the information is being written on, but regarding data content, there seems to be no way to gain useful information with tools used in this research.[26] [25]

Memory caches are important part for computer to perform optimally. In the scope of this research caches and registers are relevant and interesting, because they reside close to processor and emit radiation while in use. Registers are placed within the processor and both registers and caches are volatile memory, so they need current to remain operational. They will emit signals that can be detected via electromagnetic detection tools. As mentioned earlier, separation of a single event is not interesting in this research, rather focus is to resolve certain processes based on generally collected radiation of the computer, while trying to focus on the processor.[24]

Memory speeds are usually given in frequencies or in clock cycles. which are either in MHz range, or they are marked as how many cycles does it take to complete a task. In later case this is relatively easily converted into the MHz range.

Random Access Memory (RAM) is on small cards that are inserted on reserved slots on motherboard. RAM is used as temporal storing location for data that is being processed, usually the data of the program that is being run is stored on RAM memory. RAM needs electricity to operate, in other words RAM memory is volatile memory, that erases if power is cut off. Operating speeds of RAM are around the same as what processor has, so it is possible that RAM causes interference on the frequencies that the processor operates on. This is also a worthwhile notification; In this work I will concentrate on processor, but RAM might be also a fine location to collect SCR information about the computer and processes it is executing.[8]

With the increase in memory speed, there comes a few terms or concepts that are worth mention for us to better understand the function of memory. First, memory runs at certain speed, which depends on the properties of the memory and properties of the computer, especially number of channels that are employed. For example, DDR3 2133 MHz RAM memory standard operating speed is 1333MHz, so standard single channel memory chip

would run at the speed of 1333 MHz. It is also possible to overclock these numbers, so exact speed depends on the system at hand.[27]

This means that RAM memory operates at approximately similar frequencies ranges and these the functions that RAM undergoes are closely related to operations of the processor.[1] Both of these operate on electric currents and frequency rates that are relatively close to each other. It seems to be safe to assume that we cannot distinguish these radiations from each other, rather they will be combined as an electric field that can be observed from a single point as described earlier, especially when discussing the coulombs field law.

Memory inside the chip is connected to the CPU via BUS described earlier, so limitations caused by the bus also apply and address information is conveyed by address line to the memory chip. With this information, interface chip can find exact location of the data that is to be retrieved. To better understand following definition, on should remember that memory chip is divided in matrices, data has row and column information, with can be used to retrieve the data. Notable features in the memory are:[28]

1. tCL or CAS (column access strobe) latency is the time for a request for data to leave from the processor, travel to the memory chip and for the memory chip to do its internal processes to fetch the data all the way until it is ready to return requested data.

2. tRCD (Row address to column address delay): RAS to CAS delay is the time that the memory chip needs to activate the line row access strobe (RAS) and column access strobe (CAS) to the slot where data is stored.

3. tRP (Row Precharge time): tRCD recharge is the delay on how long it takes to disable precious tRCD access line and establish a new one.

4. tRAS (Row Active Time) active to precharge delay: How long it takes altogether for a new access to the memory can be initiated

5. CMD command rate: How long it takes altogether for a memory chip to be ready for a new command once it have been activated.

[1] As mentioned in the text, the speed of the memory might be around 1.33GHz and the speed of the processor is around 2.5 faster than the speed of said memory being around 3.5 GHz, still the range is around the same in GHz area.

When considering features described and the radiation pattern that they might yields, here are a few viewpoints: tRCD tells us about how quickly certain row or column ca be accessed. This should emanate a signal; this is an electromagnetic current that causes the data to be accessed. Furthermore, there needs to be a silent stage when tRP occurs, a moment before next fetch command can be accessed. Also, tRAS should be detectable, when one command is finished. Even CMD might give a noticeable break in RAM activity, which should be notable in radiation pattern. These should be notable at least on high precision device. If these can be detected, the results will be announced with more detail during the test phase of this work.[28]

Based on these observations that in turn are based on basic functionalities of RAM, it seems that surveying RAM might give us information about data that is fetched for the processor to process further. These patterns might provide useful information about process at hand. RAM might give useful information about computer functionalities via emanations it produces.

2.2.4 Memory efficiency

These times can be enhanced with the use of dual or multichannel techniques, which increase memory timings and thus increase the overall performance of the computer. While there are limitations to dual- or multichaneling memory, this practice gives a significant performance increase: bus's can combined for example two 64-bit bus to form a 128-bit bus, dual memory can be accessed twice per clock cycle and with multilevel architecture related data can be stored in several chips, so only small amount of data needs to be fetched from one place and this can be done simultaneously, so larger amount of data can be fetched quickly from several locations. This naturally influence radiation pattern of the processor and related memory.[29]

If we consider processor speeds related to the time certain actions consume, we can see that memory timings and processor cycles are not that far away from each other: Timings of the memory might take some nanoseconds as described in earlier chapter. 3,5 GHz processor cycle time can be calculated with formula: 10^9ns / (3,5 x 10^9) which gives us approximate result: 0,29ns. So as mentioned, processors and RAM memory operate approximately on same frequencies, but when considering optimal workflow of the computer, even fast RAM memory means that a computer needs to idle. To obtain optimum results, still faster memory is needed to prevent as much processor idle time as possible.[29]

It seems that RAM memory timings get close enough to be useful source of data to be gathered via the means of this research. As mentioned earlier excess data generated by bus's

and memory chips close to the processor might enrich the overall data that can be collected via electromagnetic field sensors and thus it might provide additional data about processes happening in the computer. As seen in previous paragraph, these timings seem to be about 10-15 times slower than what processor working frequencies are (3,5GHz computer processor clock cycle was about 0,3 ns, total CAS of RAM memory seems to be around 5ns - 10ns). These probably fit in measured frequency areas and probably contribute to the information value of gathered results.

2.2.5 Cache memory

There are different sizes of caches and cache blocks in modern computers to ensure the most effective use of processor time. Caches are the way to store data that is to be processed by the processor. There are several levels of cache to ensure a good performance. If a processor needs to idle while waiting for data, it slows the overall performance of the computer, so cache usage is well optimised to ensure optimal performance of the computer. Cache prediction and optimization is important part of smooth functions, though they remain outside of the scope of this work. In next paragraphs cache system is presented, focus staying at the effects that cause possible emanations related to this research.[24]

The idea of the cache memory is to provide steady stream of data for the processor to consume. Cache memory is located as close to the processor as possible; it can reside either inside the processor or at the motherboard at close vicinity of the processor and connected to fast bus for rapid transfer. Cache data can be divided in different levels or layers depending on the properties and demands of each layer. Layer 1 Is fastest and smallest, layer 2 is larger, but slower, than layer 1 and finally layer 3 is largest, but slowest. The faster the memory is, the more expensive it is to manufacture, it also consumes a lot of electricity to operate. Even though large electricity consumption is a good thing in this research, because it causes stronger signals, but for manufacturers this is a drawback; heat generations is larger and this excess heat needs to be removed.[30], [31]

Cache sizes are connected to the processor data sizes, which usually means 32- bit or 64-bit word sizes. These accumulated words can be selected by using data addressing schemes, which are in significant role in computer performance. With large size of programs, it is not possible to locate all data in the memory blocks that are used by the processor; some predictions are made about what data is needed next. These predictions are not always true: Some data might get discarded, if the prediction fails. This happens more regularly on a

higher-level cache, where more data is collected. Level 1 cache is a little different, it has direct connected to the processor's registers and thus it almost always contains the data that is going to be used next. This data is discarded only in the situations, where a process is interrupted by processor interrupt command.[31], [32]

Cache blocs are probably noticeable by their side channel emanations. In the scope of this work, lower-level cache might be noticeable and useful resource to cumulate data for further analysis. The fact that it has greater energy consumption gives a reason to assume that it can be seen more clearer in the spectrum than what other higher-level caches can be seen. Also, data in level 1 cache is usually used by the processor, so there should be only a few erase cycles, and cache exclusively used according to processor needs. Radiations caused by level 1 cache seems to be mostly beneficial for this research, because it contains information about current process and the data is digested by the processor during its next cycles.[30], [31], [32]

Before registers are introduced with more details, here are some core differences between registers and caches. Registers are even smaller in storage sizes than what caches are, and they are in the processor. Caches can be seen as data storages, while registers are only used to store the next piece of data used by the processor or equivalent data with immediate need. While cache is used to reduce the time for searching the data from RAM, register is needed to reduce the time searching the data from cache: while cache search times can be several cycles, register access time is one cycle.[18], [31]

2.2.6 Registers

Registers reside inside the processor, and they functions are closely related to the functions of the processor. Each processor model and each processor manufacturer might deploy registers differently, so here is a general view on how registers operate in general viewpoint. For the scope of this work, it is enough to have general understanding about registers and their functions, concentrating on the radiation aspect of their operations.[33], [34]

There are several different kinds of registers in the processor to ensure its effective operation. Processor registries are general purpose registers, where data can be loaded from the memory, or which can hold operands waiting to be processed. Number of registers varies as mentioned. For example, 64- bit RISC-V processor type has 32 registers, ARM processor has 31 registers and Intel x86 architecture has only 16 registers.[35]

Memory Buffer Registers (MBR) are used to store words received or being transferred from/to the memory or I/O devices. Memory Address Register (MAR) contains the exact memory location that contains the data needed or where already processed data is to be transmitted. Instruction Register (IR) holds microprogrammed opcode that is currently being processes. Instruction Buffer Register (IBR) stores following instruction for it to be available when needed. IBR might also be called as Curren Instruction Register (CIR), which fills the same role as IBR. Program Counter (PC) stores the address for the next instruction that will be fetched. Accumulator (AC) is used to temporarily store operands of the processes and results before those are being carried to the memory.[33], [34]

From emanations viewpoint we can conclude that there are couple of different types of register: there are data and general-purpose register, such as MBR and AC. There are memory registers such as MAR. And finally, there are internal and architectural registers such as IR and PC. Data and general-purpose registers seem to be the ones which are mostly active during a processor operation: most of the times a processor functions require some of these registers to be active. Memory registers need to be active either when a new data is required from the RAM, when data needs to be written there or when new instruction location is needed. Internal registers are needed when an operation is finished, and a processor needs to move forward to next stage: a new instruction is fetched, and instruction register can direct other components to perform required tasks.[33]

With accurate enough measuring tools, it might be possible to distinguish signals emitted by each of these registers. Although we could not see what data is being processed this might yield very accurate picture about what is happening inside the processor and what kinds of tasks it performs at any given moment. This would require very precise tools and a lot of memory, if signals from each clock cycle is to be recorded. Still with tools on hand for this research it is assumed that general picture of the processes can be recorded and analysed.

2.2.7 The processor

Central Processing Unit (CPU) is a key component in computers operations. A processor processes actions executed by the computer, so it is at the core of computers operations. While operating it also emits emanations tied to its clock cycles, so it is a valid target for emanations analysis. These are the reason why a processor is chosen to be the main interest of this research: understanding what the processor enables us to understand the whole computer. If every command that the processor operates on could be interpreted, it could be possible

replicate every action that is performed on that computer. To put this in the scope of this research: It could be possible to detect programs running in the computer.[8], [36]

Every program that a computer runs includes the instruction for processor to execute the program. These programs are written in variety of languages. Instruction to every programming language is not stored in the processor. Programming languages can be translated to an assembly language, which contains instruction set for the CPU to execute the code. In most cases a program written in some programming language is transformed to assembly language, that is further transformed to machine language that the processor can execute.[36] [37]

There are several different kinds of assembly languages, and these handle instructions differently. There are two main paradigms on how to utilize processor: fixed length and variable length instruction sets. Fixed length operation sets are as stated: fixed length. This means that there are only certain number of bits in each instruction that they perform, and these are easy to decode due to simplicity of set. Drawback is, that there are only limited number of actions that can be performed with these instructions. Variable length instruction sets vary in length and actions can be decoded in different ways. This gives a much wider range of opportunities to perform actions with processor. Drawback is that decoding these instructions is a very complex task. For example, RISC-V utilizes fixed length instruction sets, which there are about 100 instructions altogether. The x86 utilizes variable length instructions. Although exact number is not given, there is estimated to be couple of thousand instructions in x86 architecture.[37][38] [39]

2.2.8 Instructions

Instructions executed in the processor emanate a signal based on the circuit level activity described earlier. With precise enough devices these emanations can be detected and analysed. If we consider fixed length assembly languages, it might be possible to exactly capture the instructions performed by the processor and replicate these on other processor or even straight forward interpret their meaning to gain insight what processes the computer is doing and thus gain knowledge about exact operations that are done. This same is much more difficult with variable length instructions, but it might still be possible to interpret the meaning of the emanations from the circuit level activity inside the processor while the processor is using the variable length instruction sets. After all there are only limited number

of instructions that the processor executes: arithmetic instructions, branch instructions and memory instructions.[32]

When we know that processor runs a certain set of instructions, which are uniform in specific architecture, it might be possible to interpret these operations from radiation signals formed when certain operation is performed. Although there are several types of instructions, number of different combinations is limited, and thus it might be possible to find them. Although this trail of thought is worrisome in context of espionage, it might also provide us a way of detecting malicious programs in computer with external device which is totally disconnected from the target computer. This task would be too vast for the scope of this work, but with enough time and precise measuring tools, it might be possible to achieve.

A processor stores operands and commands in register. For these there are two possibilities: either 32-bit or 64-bit data sizes. If all other elements are let similar, a 64-bit computer is twice as fast as a 32-bit computer. Most of the new computers are made on 64-bit architecture, but there are plenty of computers in use which are based on 32-bit architecture.[32] [40]

These differences must be considered when trying to interpret what the kind of instruction sets the processor is processing. Instruction is the smallest execution packet, that a processor can handle, and it might consist of several operations.[32] More of these in later of this chapter. Instruction differs from 32-bit architecture to 64-bit architecture and sizes of data blocks is differ. In simple words: bit amount effects on how and how many circuits are active in each step. This naturally causes differences in radiation patterns of 32- bit and 64- bit processors: in both cases there are different amounts of components active, so radiation pattern differs between these two possibilities.[40]

2.2.9 Processor cycles

The real magic that makes the processor to function is a clock. A clock gives timing to every operation that a processor executes, and a clock cycle dictates the processes that a processor can altogether do and these also assign the execution order of these processes.[40] A modern processor is a very complex device, and the exact functions are closely guarded business secret of the manufacturer. Because of this lack of exact information, this presentation here is a proximation and simplification at the best. Still, it provides us useful understanding on how the processor executing orders affects the radiation emitted by the device.

Applying a clock that provides exact time to the processor is not a simple task. Even when electricity travels with the speed of light, this small delay can affect the processor greatly. This means that processor must be constructed so that time differences caused by limited speed of light does not affect the performance of the processor. In the context of this work this thrive for simultaneous operations has beneficial outcome: When operation is executed simultaneously (or as precisely as it is possible), radiation caused by this is emitted at the same time and thus it creates one clear emanation event, that can be observed.[8], [24], [40]

In efficient processor the cycle times are minimized, and cycle counts are reduced so that the number of empty cycles would be as low as possible. Still, it seems impossible to manufacture a perfect processor, so some idleness and wated cycles are to be accepted so that a processor would perform reliably. Most missed cycles are usually caused by register misread, or similar faults. Cycles perform instructions made in machine language; the goal for this language is to break more complex instruction simple enough, so that they can be executed during cycle time.[41]

A cycle length is determined by maximum time that a processor takes to accomplish certain task. It is very likely, that one task takes longer time than another task. It might be possible to divide that certain task in shorter components, but still there will be certain time limit for the lengthiest task. It might be ALU calculation or fetching information from memory, and this certain task determines the longest cycle time. A processor can be synchronized either to be a synchronous system or asynchronous system. In synchronous system, simply the longest possible task time defines the overall cycle time. If certain task is completed quicker, it only means, that there is idle time, before next cycle begins. In asynchronous systems, when a task is completed, a competition signal is sent and a new cycle begins after this signal.[8], [36], [40]

In synchronous system this can be described as follows: A cycle begins, when instruction decoder decodes received opcode and specifies the values for the system registers based on this order. After this, data is forwarded to other registers or other components of the processor. Finally, registers are sampled by a pulse from the clock and cycle is finished. It is possible that task is completed earlier, this causes the processor to idle for certain amount of time. In asynchronous system instruction decoder decodes the opcode and specifies the values for system registers. Then as earlier, the data is forwarded onwards, and specified tasks are

completed. When result is written to final register, signal is transmitted, and next cycle begins. This makes the system more complex but saves idle time.[36]

When considering on how a processor can complete complex or long tasks at most efficient level, there are several possible solutions: executable code can be written in a manner that supports parallelism when executing that code. Also, processor can be built in a way that supports multiple operations simultaneously, so called superscalar processors, or a more straightforward approach, which is called pipelined processor. Manufacturers tend to choose optimal solution for favour cost efficiency and performance to get most output from the processors they produce. It seems, that modern processors try to exploit most of these approaches, and they might even combine these to get the best results. Gaining information on how a processor really work is basically impossible, this information is not distributed outside of the company because it is their core business secret. Still, once a processor is designed, made choices are permanent, when a processor enters manufacturing phase it cannot be altered anymore.[32]

2.2.10 Processor workflow

Exact information on how measured processor functions exactly cannot be obtained due the commercial secret of the producing details. Understanding the basics of the processor underlaying design philosophy is beneficial, when trying to interpret the emanation readings that the processor produces when it is running and executing its work. In next few paragraphs scalar, superscalar and pipelined approaches are introduced, which seems to be the main design philosophies of the processor.

A scalar processor is simplest solution for a processor. With simple approach it is the easiest to manufacture, but it lacks the speed that other processor types possess. Scalar processor processes instruction from instruction stream in order and it completes full cycle before accepting next instruction. This causes the processor to idle longer periods and thus causing slower performance rates. Actions that a scalar processor performs are: Fetching the instruction to instruction register, decoding the instruction, generating address to related memory location, fetching the operands into registers, executing instructed operation and writing back the result.[42][32]

Superscalar processors can be considered as scalar processors, but they can perform several operations simultaneously, thus increasing the efficiency of the processor. This can be done

either with parallel pipelines or diversified pipelines. Parallel pipelines can handle certain number of operations simultaneously. For example, parallel and spatial pipeline capable of performing three operations simultaneously can be considered pipeline with width of three and it is approximately three times more efficient than scalar pipeline. The downside of gained performance is that the system tends to be three times more complex than what scalar pipeline is. In diversified pipelines are designed in the way that the pipeline is divided in multiple functional units, which can perform different operations. Results of these operations are finally unified in a single result, which can be addressed forward. This approach makes it possible for processor control logic to divide operations to suitable branches. When operation types differ, there is no risk from accessing the same data when a processor performs its operations.[42] [43]

In a pipelined processor, operation cycle or revolution of the work cycle of the processor is independent. During each phase operations can be overlapped and if a conflict is detected, downstream processes are halted to force dependency and to guarantee robust operation. In pipelined processor only one of the following phases is active at any given moment. Pipelined processor operation consists of six phases: Fetching the operation, decoding the operation, generating address, accessing operands, executing the operation and finally storing the results. [42][32]

Pipelined approach can be further divided to static and dynamic approaches: In static approach a processor goes through all the stages a phase, regardless of if they require action or not. In dynamic approach, a processor can skip one or more phases, if requirements of the operation do not require them. In dynamic approach a processor can perform some actions out of order or even initiate out of order. Maintaining integrity in out of order processor is a demanding task and require a lot of effort to provide robust operation of the orders.[32]

Differences in these execution methods could be seen in the emanation scope when analysing the radiation processors emanate: In more effective solutions there should be less downtimes, and a processor should operate mostly at the higher frequencies and higher workload. With less sophisticated approaches, like scalar processes there should be more downtimes that should be seen as smaller power consumption and lower average working frequencies.

2.3 Computer processes

Certain processes are active in the computer regardless of the actual work that it is executing. These processes use process time, and they should be seen in EM spectrum even when a computer is on idle. Understanding these constant processes may help to understand emanation spikes when analysing the results obtained from the spectrum.

At the fundamental level computer processes are a simple thing: User, other program or similar input is given, a program is started and executed and finally result of execution is stored somewhere in the memory or printed out. This would be ideal condition for this thesis; it could only be needed to detect simple process and analyse this as it is running. Unfortunately, reality is often much more complex. For a computer to be functional device, it usually needs an operating system. These can vary from a simple real time operating systems to complex operating systems. More complexity is added with different layers for each devices connected; all the connected devices come with software; these drivers operate simultaneously when computer is being used. This complexity makes detecting traces of emissions of a simple program difficult task: we should be able to distinguish noise generated from normal tasks from signals emitted by task we are interested in.[29]

In this subchapter a closer look is taken into the processes inside the computer, which effect on our ability to detect processes inside the computer without going too much in to details of these tasks. Entire books could be written about simplest drivers or kernel level programs, but here focus is on basic understanding of these processes considering the computer related to emissions they generate. Complex processes inside the computer are abstracted down so that they can be discussed in reasonable manner. A few critical details have brought us, which can be seen to have significant effect on emanations generated by the machine. This includes a short overview on programs generally, then a few paragraphs about drivers and they purpose on computer and finally fundamental idea about system or kernel level programs. Target operating system is Linux, which is at the centre of these observations, but most of these notations apply on windows or other operating systems.[29]

2.3.1 Basic processes

In the previous chapter functions of the CPU cycles were described as well as fetching data from a memory block. These still form a core of the execution of a program, but as mentioned, running a usable program is way more complex task. A program is written on a

certain programming language, these languages differ from each other on how they are meant to be executed. Programs generally have a runtime environment where they are run. These environments help to allocate memory resources, variable access, parameter passing, interfacing with OS and they usually are language specific.[40]

To be able to launch a program, a suitable runtime environment needs to be launched. This causes operations across the computer, when runtime environment establishes itself. When a program is launched, first data of that program is located inside the mass storage. This data is written usually either in entry point or on a separate file in the executable program itself. After this necessary data is loaded in memory, where it is fed to the processor alongside with required variables, parameters and end addresses. Finally, when program is executed, it needs to be removed from main memory, unless it is likely to be used again. To perform these tasks, a bunch of small but vital programs are needed: such as loaders, linkers, schedulers and exception handlers.[40]

These described processes make it possible to utilize computer efficiently by distributing resources where they are needed. On hindsight these operations cause a lot of unrelated noise to be generated from the computer, which obscures exact processes that a computer is executing. On the other hand, these related processes can factor to a certain fingerprint, which can be used in identifying processes that are to be executed.

2.3.2 Linux kernel processes

In this thesis Linux machine is used as target device, which processing data is analysed. That's why Linux processes are inspected here with more detail. Though these are considered here with more detail, this will merely be a scratch about what complexity modern computer and operating system consists of. Hardware level processes were considered in previous chapter and user level programs are considered in next chapters, so here our focus is on kernel level programs, which are ones that user have limited or even no control over.

A computer running a Linux operating system, can be divided in three layers to gain better understanding on how components relate to each other and how this whole system can be arranged to a logical order: We have hardware, Kernell processes and user processes. [44]

Hardware is discussed more in detail in previous chapter, as reminder the crucial components are processing unit, memory at different hierarchies and bus:es. These form a core of the computer with related software so they can function as intended. In previous chapter

pipelining and multicore functions were briefly discussed. In addition to these, Linux utilises interrupts to perform necessary tasks. These interrupts make it possible to run the actual payload and necessary system usage utilities without too much interruption for the computer itself or the processes it is supposed to run. If we consider the CPU from this waypoint, it is evident, that while executing certain process, a CPU performs slices of system management code every now and then, which might disturb the actual pattern formed of the process that is being monitored.[45] These will be discussed more in detail, later in this chapter.

Kernel consists of programs that are specially designed and developed for specific tasks that they perform. Most important ones are the ones which are responsible for distributing resources for different tasks, and those ones which act as interface between hardware and the user. Other notable functionalities are schedulers, which are responsible for CPU resource scheduling, memory managers including loaders, which load processed content into the memory blocks. Inter process communicators, which are responsible for conveying information between processes and river system for both physical drivers and for logical driver. Physical drivers are drives for physical devices which are connected to the computer, logical drivers are more abstracted programs that fill certain functionality in disc management or connection protocol space.[46]

2.3.3 Process management

Whatever kind of program user is using, Linux uses only certain types of functions to access memory storage and schedule tasks for CPU. Here is a brief description of this system. In application or user layer there are four main methods of operations: read()/write(), mmap, asynchronous I/O and O_direct. Traditional read() write() are the standard file system, which works in synchronous manner to perform system calls involving read and write functions. This is synchronous method, which can be inefficient in modern computers. Memory-mapped file I/O mmap, can map files directly to memory so that specific read or write operations are not needed. It has its own issues in complex scenarios but is relatively fast method otherwise. Asynchronous I/O is read write system, where other processing is permitted even when initial task is not finished. This method is especially useful with use of multicore processors. The final possibility is directing I/O, which can be utilized to for a very efficient way of transferring data directly to memory. These can be seen in Figure 1: Linux workflow. The figure is from the source cited here.[47]

When an application chooses one of these access functions, it contributes files or memory slots to computers virtual file system, which directs these to the buffer cache, which finally directs operation to be handled in the block layer, where task is first submitted, then staged and tagged. Then task is scheduled and finally directed to CPU cores as they are scheduled. After this processing, obtained results are directed to either a device driver, distributed storage or to local storage.[48]

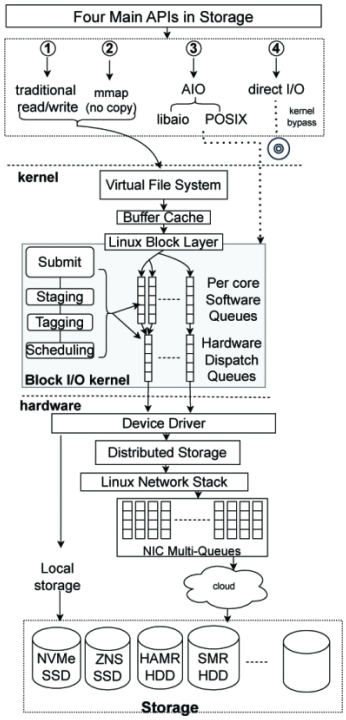

Figure 1:Linux workflow

While previously presented task scheduling is complex system, it gets even more complex when we take into consideration system interrupts and they effect on operating system noise, which is well presented in cited study “operating system noise in the Linux kernel” by Oliveira et all. While their focus is on improving the profitability of the processor, the observations can be utilized in the scope of side channel emissions of the processor.

When a processor is running, and executing operations scheduled by a scheduler, it has computational, communications and synchronization phases, which helps it to run on phase with other components and thus makes it possible to a processor to run efficiently. Between these phases there seems to happen system noise phases, which can be seen as inefficiencies in the process. These can be defined as: “The OS noise is defined as all the time spent by a

CPU executing instructions not belonging to a given application task assigned to that CPU while the task is ready to run.”.

Tracking certain program with process emanations seems to be a very difficult task Even if we could monitor schedulers efficiently, we could not monitor all the processes because of this unexpected noise, which makes building a precise prediction tool, almost impossible task. But still with modern AI-applications it might be possible to identify certain programs running in the processor and react to these, if necessary.

2.4 Summary of theoretical background of the electromagnetism

Here is summation of the notable observation from the first chapter, which is also a literature review on the matter at hand. Research questions, which were to clarify in this chapter are:

1. How electromagnetic radiation manifests in high frequency ranges and how it can be measured?

2. What kinds of mechanisms cause computer to leak information and what can be interpreted from this information?

For the first question the answer is given in subchapter 2.1 and for the second questions answers are mostly given in the subchapters 2.2 and 2.3. Notable observations from these subchapters are brought up in the following paragraphs in the order of the subchapters.

2.4.1 Observations on electromagnetic radiations

First notable observation is the inversely proportional relations between wavelength and frequency: For higher the frequency the smaller the wavelength. This might cause a problem with antennas with lower frequencies, but it seems that observable frequencies tend to lay around 1GHz, so wavelengths are relatively small and thus required antennas are also reasonably sized. This means that the radio waves up to 1,7GHz can be easily detected with antenna set described later in this work and with the selected SDR device, but higher frequencies that the processor uses cannot be observed with this setup. This leaves some improvements to be done in later research with more precise and wide ranged equipment.

Nyqvist sampling theory proposes that if certain sample frequency needs to be sampled, samples from at least double of this frequency needs to be collected. This holds true in many cases, but for this research there is a small twist based on the equipment used: The RTL-SDR

radio collects data from 4 MHz window at the time and this device does not make it possible to scan the whole range instantly. To overcome this challenge, in the parts where data from whole frequency range is needed, the range is scanned in less than 4 MHz slices, this data is collected and then assembled to present the whole spectrum. The process of this data collection takes around 4 to 5 minutes, so this does not make it possible to collect instant results from the whole spectrum.

Interesting bandwidths are analysed on these 5 MHz bands, results can be observed basically in real time. Naturally it takes a miniscule amount of time for signal to travel, again a small fraction of second is lost when signal is processed and presented, but changes in the signal can be detected almost instantly. This 5 MHz window should be adequate to investigate interesting signals that they hold.

Based on the energy density, we can conclude that electricity inside the components leak out and from the components and this leaked current is proportional on the strength of the field in the component. Based on the Coulombs field law we can conclude that all the small currents emitted from the components form an observable field which can be measured, but distinguishing emission sources from this field is impossible based on their signature in the electromagnetic field. When considering antenna theory and partly energy density, electric and magnetic fields differ from each other, though this difference is quite small, it is still observable. Also, with right kind of sensing equipment, it is possible to concentrate either on magnetic component or on the electric component of the radiation. This can yield different kinds of results when observing emissions either with magnetic field or electric field antenna.

Altogether electromagnetic radiation observed is the sum of all the sources around the measurement instrument. These cannot be distinguished in any simple manner, so collected sample consists of useful signals seen on different frequencies, as well as useless ones that cannot be avoided. It is up to receivers and antennas properties as well as analysing software to present the collected sample space for analysis. While analysing these findings, some insight can be used in identifying certain frequency ranges based on the prior knowledge about what the signal might be.

2.4.2 Observations on the radiation of the components

Functions of the components in the computer are based on the electric current changing states of transistors and other micro components. These currents form an electromagnetic field when

they flow through the parts. Surfaces of these components and wirings makes a transmitting antenna, which relays these current changes into the medium as electromagnetic radiation. While modern components are miniscule so measuring one single transistor at the time seems to be impossible task, monitoring EM field generated by several can yield useful information about operations going on in the computer. Usefulness of this data depends on the quality of the sensing equipment.

Computers are complex systems with several interlinked components. It is difficult to distinguish most of the components from each other or to define them exactly: For example, a processor connected to a motherboard form an integral component. Other components are also closely interconnected via motherboard. In motherboard data is transmitted via a bus. The bus is integrated to both the destiny device and to the processor with their own microcontroller. Separating these could be possible at some situations, in this research this is not possible and even not recommendable. All these components contribute to the total electromagnetic field with their own emanations.

While a processor is the focus of this study, the memory holds the information, which is the interesting aspect that we need to find out. Normally this information can be accessed via the normal usage of the computer, but in this research, this is done by analysing the radiation patterns emitted by the processor. Memory, especially the cache and register memory holds valuable information about the process and state of the computer, it is possible to detect read and write processes of the memory if a proper frequency is found: As stated in the text, memory operates on certain frequencies and especially fast but volatile memory operates basically on current changes.

Reading or writing a memory seems to be rather simple task, but as often with computers, there are a plenty of technologies which enhance the use of the basic operations thus making them complicated. With the case of memory there are different kinds of timings to be considered as well as dual and multichannelling techniques to be concerned of. These makes it more difficult to interpret the emanations caused by the read and write cycles. These variations and uncertainties should be considered when analysing the emanated spectrum and this seems to require quite hefty AI component and a lot of samples for analysis.

Analysing emanations caused by cache memory might prove an effective way in estimating overall effectiveness of the computer and programs. With precise enough tools, interrupts caused by incorrect estimations in data content might be detectable and with careful planning

the number of these interrupts might be lowered. These interrupts hamper the effective use of the computer and minimizing them would increase the overall performance of the computer.

Computer operates on predefined instruction sets. These instructions are simple commands that tells the processor what to do and on with data. Could it be possible to record the emanation pattern of these instructions sets for every instruction set. If this kind of data store could be formed, could it be possible to interpret the state of the computer just by comparing emanation patters to these stored patterns? Thes could make it possible to read the computer just by analysing its emanations.

Idle times and uneven workflow are a clear disadvantage for an optimum performance of the computer. Detecting these can be difficult with selected equipment but it would be beneficial to gain information about these at least on industrial scale server and machines. If operating volumes are high, even a slightest improvement could yield a lot of benefits. Also, on more toned-down scale of small servers, it might be useful to acquire simple, robust and unexpensive mean to gain information about a processor workload.

It seems that without any shielding, almost all information can be gathered via the emanations caused by the components of the computer. This information can be shielded via different kinds of protections. Also obtaining useful information might be a difficult task due the complex nature of the data processing and various techniques used to improve the efficiency.

Based on the analysis of the computer basics there are certain frequencies and events that should be detectable either in this research or later with a bit more precise equipment:

Frequencies to look for: CPU base clock, graphic card frequency and bus activity.

Events to consider: Ram timings, delay between processor command and activation of the memory block, processor cycles.

While the theoretical background on how electromagnetic radiation is formed in the computer and how it manifests, it would have been beneficial to calculate with real numbers and real estimated values on what kinds of electromagnetic fields and what kinds of amplitudes certain currencies in the devices might produce. Computer components are also discussed at relatively general level, more details would be beneficial but writing these would require a lot more pages.

3 Measurement setup

First goal is to find changes in spectrum when computer is turned off and then on. This proofs us that the properties of the RTL-SDR is enough to detect these kinds of chances in spectrum. This goal also helps us to find frequencies that are affected by computer hardware. This is not as trivial as it might sound, as stated in previous subchapter. Achieving this goal makes it possible to detect the computer via selected device and method. If no traces are found, then whole experiment setup need to be devised differently.

Second goal is to detect changes in pattern when a program is run on the target computer. This requires more detailed results from the device; it is not enough to detect functioning computer; more subtle changes need to be detected if a starting or running program is to be detected.

Final and most detailed results are needed if different programs are to be distinguished from each other. If this goal is met, then it is possible to detect wide variety of instances happening in the computer with totally unintrusive manner with cheap devices. This proof of concept opens a wide variety of options combined with statistical analysis to inspect a computer from afar.

3.1 Devices and software to be used

The goal of this work is to proof, that a cheaper RTL- dongle is suitable tool for measuring emanations caused by a computer, so measurements are done with RTL-SDR media converter, linked to laptop running on windows. RTLSDR dongle and antennas are described with more detail later in this chapter. They were chosen because of their affordability and relatively easy usage. Costs of this setup is reasonable, only around 50 €, excluding the laptop that is used to run analysis software.

For laptop a windows laptop was chosen. Most of the SDR software is compatible for Linux and windows, so with these either one could have sufficed. Final analysis was done using MATLAB- program, which is also compatible for both Windows and Linux, so there are no restrictions in here either. With this setup experiments were executed as described later in this work.

Overall measurement setup can be seen on Figure 2: Overall measurement setup: first signal arrives to antenna, where it is carried to RTL-SDR dongle for RF processing and Digital

processing. Then this digitalized processed signal is carried to windows laptop via USB port, where it is analysed either with MATLAB with communication toolbox and Simulink, or signal is processed with dedicated SDR program.

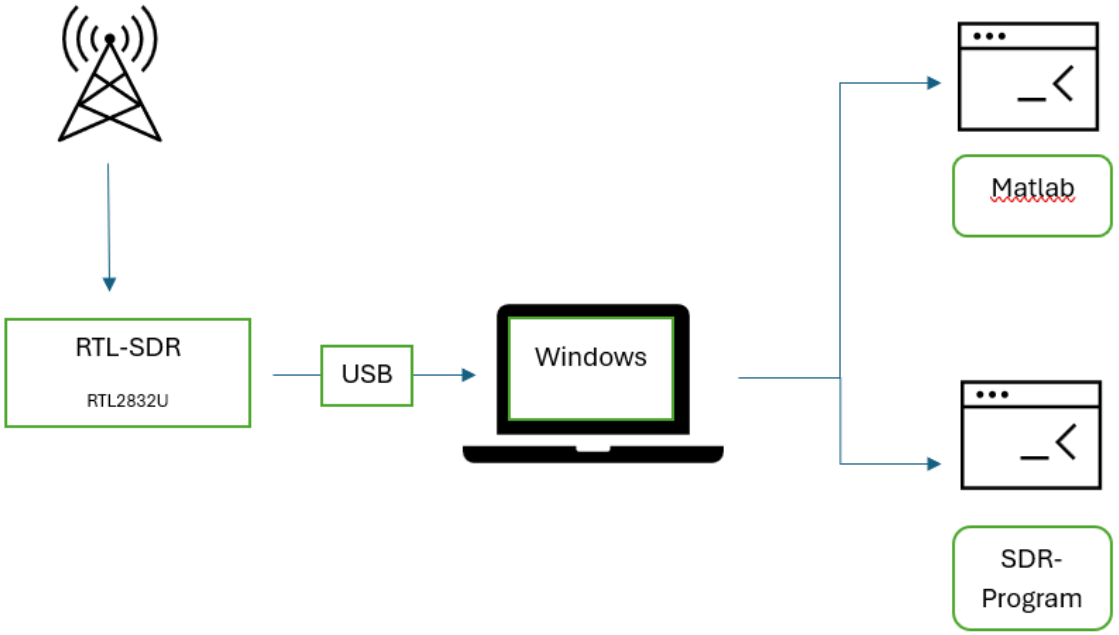

Figure 2: Overall measurement setup

Most of the data is gathered from MATLAB, but some data is analysed with JUPYTER LAB with the use of Pandas and Matplotlib extensions. These makes it possible to present the data more accurately in some cases. Data in these cases are gathered with MATLAB with the exception that collected data is saved to CSV- file which is read with JUPYTER LAB for further analysis.

3.1.1 Software defined radio

Traditional analog radios have several rather expensive components, including crystals to receive certain frequencies, capacitors, inductors, tubes, transistor and other kind of electronic components, which are rather expensive, space consuming and due to their analog nature, rather slow to react to changes. To counter these issues a development of software tuneable radio begun with the concept of SDR radios in 1992 at MIT with the work of J. Mitola at the "Spectrum Ware software radio project". With the idea of combining the analog functions into the digital system.[49]

The idea of software defined radio is that most of the components are configurable zero-IF receptors. In this manner many of the physical components can be replaced with software. This makes it possible to build whole digital systems such as transmitters, receivers, oscilloscopes and many more with the use of only digital components. Only mandatory physical components are Radio Frequency stage (RF-stage) and Intermediate Frequency stage (IF-stage), which are in direct contact of real-life phenomenon, and thus they cannot be a digital component. The idea can be seen at figure 3: Data flow through stages, where antenna, RF-stage and IF stage are real components, the rest are digital components, which reside at previous Figure X inside RTL-SDR box.[50]

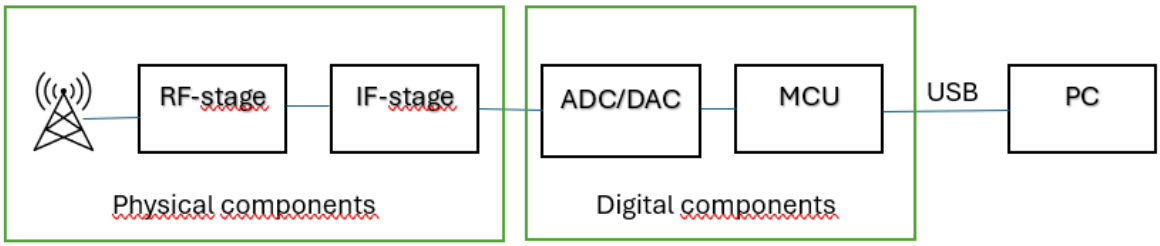

Figure 3:Data flow through stages

After the antenna catches the signal the RF-stage processes the signal at received frequency. This usually include limiting the bandwidth and it also adds low-noise amplifier, which strengthens the weak signals received by the device so that they are recognized by next stages of the processing. Antenna and FR stage determines the frequency range on where the device can operate.[51]

The IF stage shifts the signal to usually lower intermediate frequency. Signal processing at higher frequencies is usually more resource consuming and it requires more expensive equipment because of increased precision demands on the equipment. With the fixed signal frequency gained at the IF stage, further processing can be done in more beneficial frequency range.[51]

Analog to digital converter (ADC/DAC) is the device which handles the sampling of the signal transforming it from analog signal to discrete digital signal. In most cases, according to Nyqvist-Shannon sampling theorem, the input bandwidth needs to be twice the output rate of the signal. Still there are ways to gain equal sampling rate related to bandwidth with quadrature sampling. This will be introduced later in this chapter. Properties of ADC/DAC

determines the resolution of the samples and thus the overall quality of the digital signal[2]. The MCU processes the signal further to produce suitable output to the actual computer. It also has the logic implemented for it to be able to do quadrature processing of the signal. Finally digitalized and processed signal is sent via USB port to the computer for further analysis or other usage.[51]

RTL-SDR

During 2012 Finnish engineering student Antti Palosaari found out that RTL based Digital TV (DVB) devices could capture FM and DAB radios. The output stream of the device was demodulated and processed on a computer. With the help of enthusiasts, they were able to demodulate this capture manually, which led to further investigation of the RTL-SDR: s USB protocol and its output. Original device developers from Osmocom involved in figuring out Realtek drivers provided by Realtek figured out the way to program the tuner. With these development stages, it made possible to use this rather simple DVB receiver to receive signals from the RF frequency from 25 MHz to 1,7GHz and sample the frequency spectrum at a rate of up to 2,8MHz.[52]

RTL-SDR dongles are basically DVB-T TV tuners, which have been repurposed to function as cheap SDR device. This has made RTL-SDR to be a very popular device: they are both cheap and easily obtainable. This have caused these devices to be very popular. There are a lot of free software to be utilized with RTL-SDR and there are plenty of development ideas which to try. Alos it is possible to tap into the raw data stream to build own solutions. In this work mostly existing solutions have been used to gain necessary information from the experiments executed.

Extracting exact technical data from the device seems to be a little complicated task: first, this device was originally meant to be TV-tuner, so exact values related to signal processing was probably not the focus of technical documentation. Although there are technical datasheets available, they seem to be confidential by Rafael microelectronics. Rafel microelectronics does not have these older used at RTL-SDR dongles versions displayed at their product catalogue although they should be used in RTL-SDR dongles. Also, some specifications seem to be extracted by enthusiasts, although this data they have collected is probably valid, it’s not

[2] For example, a device with 12 DAC bits, like Airspy mini can yield the resolution of 2^{12}which means that it can represent analog input at 4096 different levels. With RTL-SDR this would be $2^{8}$and would result at 256 different levels of detail describing the phenomenon at hand.

in the scope of this work to validate it. Exact name of the product used is: V3 R860 RTL2832U 1PPM TCXO SMA Software Defined Radio.

Technical data of this RTL-SDR dongle for its main properties are:

- maximum sample rate: 3,2 mega samples per second
- optimal sample rate: 2,56 mega samples per second
- resolution: 8 bits
- input impedance: 75 Ohms
- Frequency range: 24-1766 MHz
- bandwidth: ~4 MHz

Many of these values depend on related hardware or software. For example, input impedance may vary to 50 Ohms, with certain antennas or resolution can be upscaled with various floating-point tools. Computers USB interface can affect sample rate, with more recent solutions being faster. [53], [54]

3.1.2 Antennas

As mentioned in the second chapter, electromagnetic radiation is divided in magnetic and electric component which are caused by electric charges. In the scope of this work, both manifestations of electric current in the medium can be used, there are small differences for magnetic field antennas and electric field antennas.

Iin this research ESCO technologies Model 7405 Near-Field Probe Set will be used. This set consists of three magnetic field (H) and two electric field (E) passive near field probes. These are initially designed to detect emission problems in electric equipment, and due to their nature, these are ideal for task at hand in this research. As spectrum analyser RTL-SDR will be used and for oscilloscope MATLAB will be used. This set can be seen on Figure 4: Probes.

Figure 4: Probes

On Figure 4: Probes First item in the image is SMA male-male 1m cable (SMA1010-1M), which is used to extend the probe closer to measurable object and to provide some distance between measuring computer and the target computer to prevent unnecessary emanations. Small object item the cable is SMA-BNC adapter to connect probes to the measuring device. Next three items are 6cm-, 3cm-, and 1 cm magnetic loop antennas described more in detail in later. Last two rightmost items are electric field stub probe and ball probe which are also described in more detail later.[55]

Magnetic field probes are labelled as model 901 for 6- cm loop, model 902 for 3- cm, loop and model 903 for 1- cm loop. These vary on sensitivity for magnetic fields but are all designed to be highly selective for magnetic field and they should be relative immune to electric field. They have single turn of 50-ohm shorted semi rigid coax loop inside balanced electric field shield. This loop is then connected to shield at the shaft. At the high point there is a notch cut in to the loop, which creates this balanced electric field shield for the probe. Model 901 has 41 dB rejection for electric field and has upper resonant frequency of 790 MHz. Model 902 has 29 dB rejection of electric field and has upper resonant frequency of 1.5 GHz. Model 903 has 11 dB rejection for electric field and has upper resonant frequency of 2,3 GHz.[55]

Electric field probes are the stud probe model 904 and the ball probe model 905. Stud probe is relative ineffective due to small sensing element. This is advantageous, when one tries to

pinpoint exact location of the examination. The impedance of the stub probe is same as non-terminated length of the 50-ohm coaxial cable which is 6 mm of the centre conductor exposed at the tip. Without loop construction this probe is unsensitive to magnetic fields. These properties are very useful at this research at hand. Ball point probe is much more sensitive than stud probe, which allows detecting weaker signals with the cost of precision in locating the origin of the signals. The ball probe has length 50-ohm coax which is terminated with 50-ohm resistor to present a conjugate termination to 50-ohm line. the centre conductor is extended beyond the 50-ohm termination and attached to 3,6-cm diameter metal ball, which serves a s electric field pick-up. the absence of lop structure provides resistance to the magnetic field. Both of model 904 and model 905 have 30 dB rejection of magnetic field. model 904 have upper resonant frequency of below 1 GHz and model 905 have upper resonant frequency of below 3 GHz.[55]

3.1.3 MATLAB

MATLAB is a computing platform designed specially to engineers and scientists. MATLAB is a programmable platform, so making own programs is possible and often encouraged. MATLAB approaches data via matrices and this makes it effective tool in collecting large quantities of data. There are many useful features in the program, for this research very useful ability is the possibility to collect and analyse data collected from radio spectrum via SDR radio and displaying this data in frequency/amplitude pairs is possible.[56]

MATLAB uses its own coding language, which can be used to code own functions or programs. This own programming language is based on Fortran. While this own programming language makes it possible to write code in MATLAB, the strongest point in MATLAB is its ability to use other programming languages with the MATLAB program. There are APIs available for C, C++, Fortran, Java and Python. Also, some Visual based languages launched by Microsoft. Whit this ability, one can write program in familiar language and run it in MATLAB without fear of breaking the program in interpretation process.[56]

MATLAB has also a tool called Simulink, which is a block based graphic programming tool. This tool is mostly designed for prototyping new products, but it can also be used in evaluating effects in different situations by feeding parameters to a model and monitoring the outcome. In Simulink there are a lot of different kinds of blocks in its library and these are customizable to meet different requitements.[56]

MATLAB enables also different kinds of toolboxes to be implemented. These can be packages that enables hardware to be connected to AMTLAB program. One of these toolboxes is TRL-SDR support from communication toolbox. This toolbox can be used to connect RTL-SDR radio directly to MATLAB. The ability to get direct stream from the radio dongle is very useful in analysing the data collected by the dongle. This data can be used also in Simulink environment if needed. Also, different kinds of graphical presentation of the data are possible.[57]

In this research MATLAB is mostly used as an endpoint for the data stream and for further analysis of the data. With different kinds of tools collected data can be combined or otherwise presented in graphical form for visualization and easier understanding of the data that is collected.

One quality of the RTL-SDR dongle is that it scans only a 4 MHz slice at the time. While this slice can be monitored in almost real time, collecting data from wider range requires combination of data. This was a small chokepoint at time being and the best compromise was to collect data from the whole spectrum in these slices and then combining this data and presenting it in one graph. While this process takes approximately 300 seconds, it is the most convenient way of analysing the whole spectrum ranging from 40MHz all the way up to 1.7GHz. Once the interesting frequencies are pinpointed, it is possible to deploy real time surveilance and analysis to these up to 4MHz sections.

3.2 Measurement setup

Measurements are meant to be executed in ordinary building without any external radiation dampening or any other kind of protection against external radiation. Results are to be gathered on the Acer Aspire 7 laptop, which is also the device used to collect data from RTL-SDR device, which is described with more detail earlier. Details of computer for data gathering are irrelevant in scope of this research, if it holds enough resources to run MATLAB and RTL-SDR software.

For the target computer, there are several qualities that are relevant and should be described. This can be seen in Figure 5: Target computer properties. Information is obtained via CPU-X program, which is maintained by a developer called The Tumultuous Unicorn Of Darkness.

The software is grabbed from GITHUB, where it is as free software. The details are as follows:

Cache size is 4 x 32 kB, 4 x 256kB and one 6MB, all in 64-byte lines.

The motherboard is Hewett-Packard model 1497 with intel chipset.

Main memory is dual Samsung DIMM DDR3 4 GB memory. Graphic card is an intel integrated graphic card with base clock speed at 850 MHz.

Operating system is Linux Mint 22.1, with Linux 6.8.0-51-generic kernel. Operating system is used from USB-stick.

Processor is Intel Sandy Bridge core i5 with 32nm technology with maximum speed of 3.1 GHz and core speed of 1591 MHz.

The bus speed is 99,77 MHz.

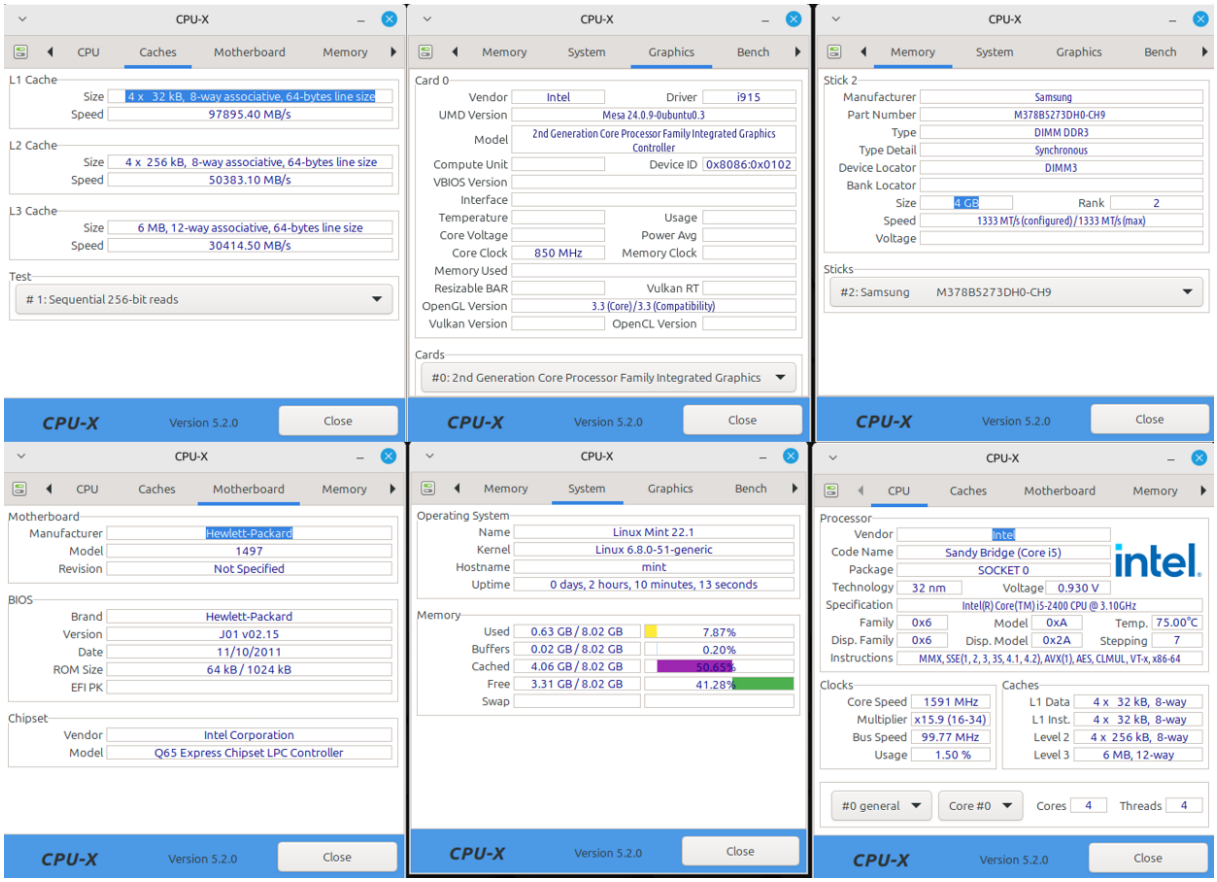

Figure 5: The target computer properties

As mentioned, the problem with RTL-SDR device is that it can only monitor 4MHz bandwidth at the time. So, if real time monitoring is needed, the phenomenon must be inside

the 4 MHz range. This requires us to narrow down the frequency range or frequency ranges so that the bandwidth of the device is sufficient to monitor desired range.

Another problem arises with these limitations: It is possible to scan the whole range from 24 MHz to 1766 MHz using <4MHz slices, but this does not happen in real time: scanning the whole range can take up to 100 seconds to complete, so real time observations are impossible with this arrangement. If an interesting phenomenon happens at the higher end of the whole frequency range, it should last for several minutes to be detected by a scan for the whole range.

There are a few good solutions to tackle these problems. The first one would be that the initial scan would be executed with a device capable of scanning the whole range instantly. This would yield the more specific frequencies that could be monitored with limited frequency range of RTL-SDR. This solution would go against the initial idea of this work: can this device be used in this task.

Second solution and the one chosen in this research is that first a scan is made through the entire frequency range using code that scans the whole frequency in less than 4 MHz chunks, performs Fourier transform on this data and yields the power ratio and relative power of the signal at frequency. Thes a second scan is made in proximity to the target computer while it is running. This should reveal the areas where frequency shifts because of the radiation caused by the computer running close by. It is also possible to perform limited scans with more specific frequency ranges, if a certain estimate is made where changes could be detectable. This would run much faster than scanning the whole range but reveal more information than scanning the whole range. Final goal is to pinpoint frequencies of interest that fit to the are less than 4 MHz bandwidth are to be inspected continuously, while further experiments are executed.

For the first part of the experiment the setup can be seen in Figure 6: Initial scan. The only difference from this figure is that connecting cable is spread fully, so that the probe is at maximum distance from possible disturbance sources. Electric field stub antenna (model 905) is used so that the least possible number of interfering signals are detected.

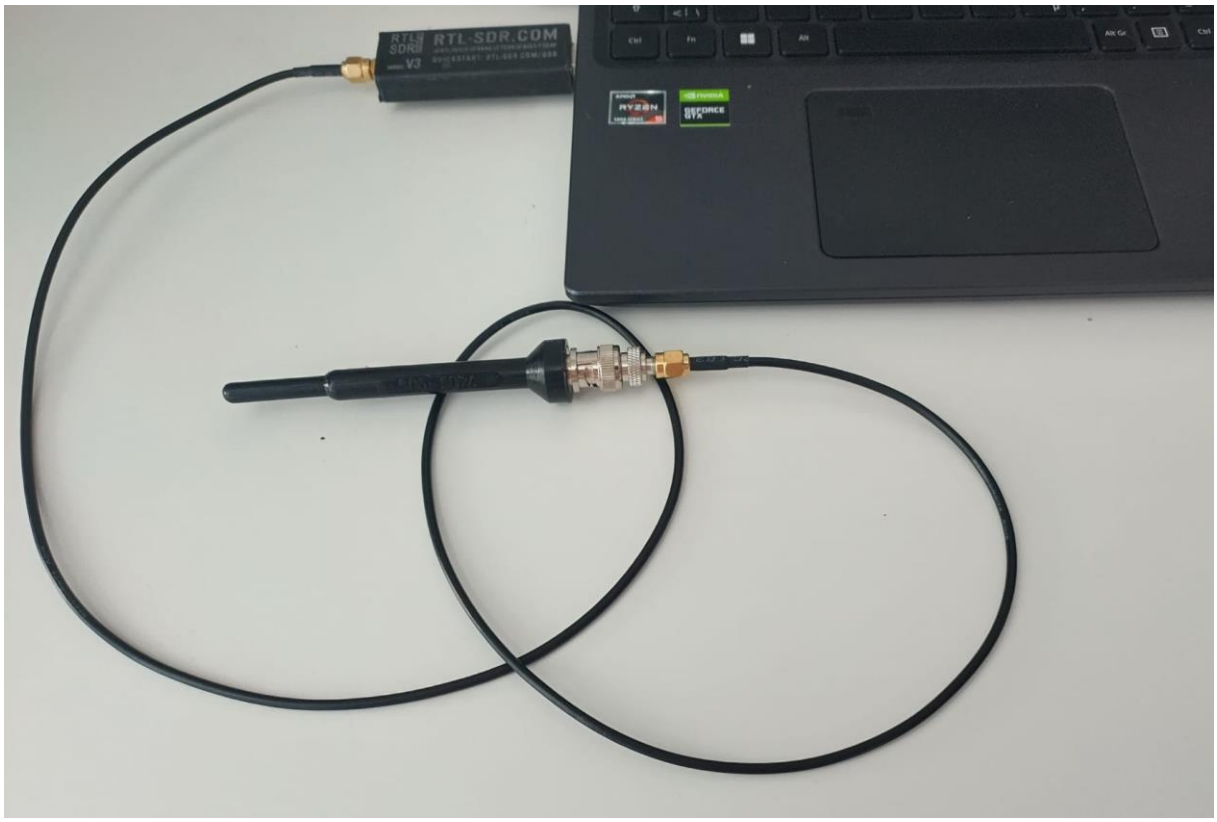

Figure 6:Initial setup

As mentioned earlier, the full scan of the spectrum takes almost a couple of minutes to run, so the resulting graph is a collection of 2,8 MHz slices of spectrum taken to form the full scan. What can be seen on the graph is Power ratio relative to 50-ohm load at upper graph painted in blue color and relative power in in Watts written in orange color at lower graph. These tell the amplitude of emanations preset at each Frequency. This can be seen in figure 7: Initial scan. In this initial scan we can see that there is lot of activity in the spectrum at lower frequencies ranging from ~80MHz all the way to ~110 MHz. Smaller spikes can be seen from ~500MHz to ~700MHz and couple of larger spikes around 800 MHz and 950 MHz. The rest of the spectrum seems to be rather calm and without clear spikes in emanation.

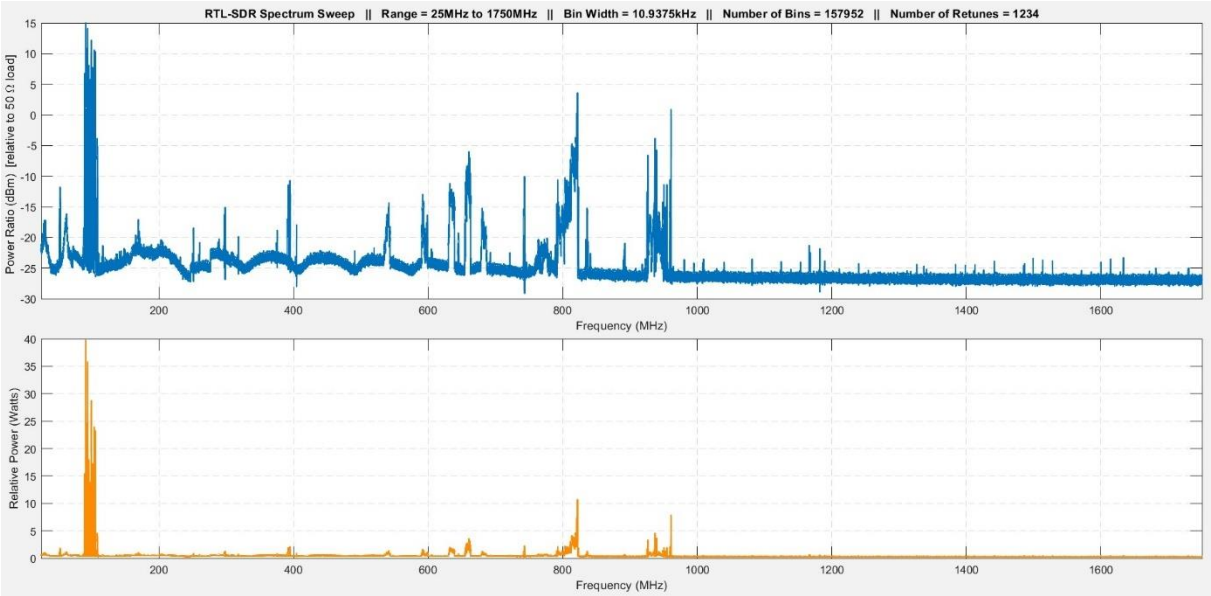

Figure 7:Initial scan

From the initial scan it is possible to detect certain frequencies flooded with activity. First spikes around ranging from approximately 85 MHz to 110 MHz are radio broadcast frequencies.[58] Around 400 MHz, 703 - 733 MHz and 758 - 788MHz and 791 - 821 MHz and 832 - 862 MHz are frequencies for datalinks utilized by cell phone operators. These are mostly LTE and LTE 5g NR frequencies. Frequencies at 880 - 915 MHz and 925 - 960 MHz are also cell phone operator frequencies, but these are reserved for GSM ULTRA LTE.[59] These are the frequencies that we are not interested in, because the heavy interference and thus acquiring useful data from these frequencies is probably not worth the effort, because the useful data will most likely to be lost in interfering signals.

Before we can start scanning the target computer some preparations are needed. First, we need an operating system for our target computer. For this purpose, a bootable memory stick is used to minimize unnecessary emissions caused by the hard drive of the target computer. For the bootable disk we need an ISO file of Linux distro and a suitable memory stick. For this experiment Linux Mint 22.1 xfce is selected. Bootable disk is then created with Balena Etcher program which is used to print the ISO file to the bootable disk. After this a few adjustments are needed for the target computer: the case needs to be opened, and the heat sink and fan need to be removed so that the processor is clearly accessible for further experiments. These should not yield any further problems, the processor is not heavily burdened during these initial tests, so overheating should not be a problem. These stages described can be seen in Figure 8: A processor with heatsink and fan on and Figure 9: heatsink removed. At Figure 8, where heatsink and fan are still on, it is evident that monitoring the processor at proximity

is not possible because the heatsink fan combo covers the whole processor setup. As seen in Figure 9, where heatsink and fan are removed, the processor's surface is easily accessible so that measurements can be taken right from the surface of the processor. If the heatsink and fan are left on, it is hard to access the processor and the structure of heatsink can distort the measurement, so it needs to be removed.

Figure 8: Processor with heatsink and fan

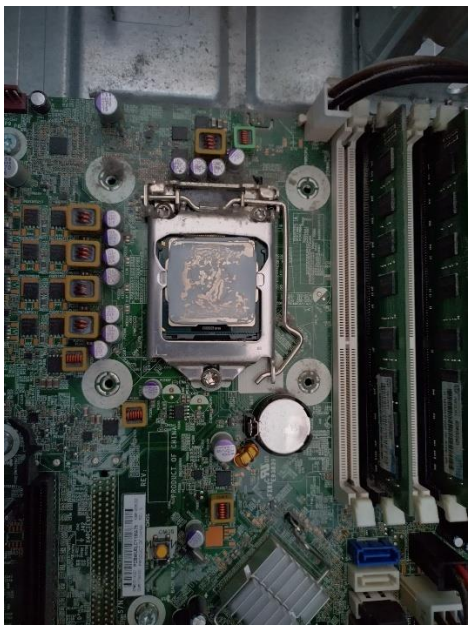

Figure 9:Processor with heatsink and fan removed

After this we can boot up our target computer and run the first scan of the computer to see if we are able to detect changes in the spectrum presented earlier. These changes should be clearly visible so that they are detectable with the naked eye and no further analysis of the

whole scanning graphs are needed. The first scan is executed with model 904 electric field antenna so that all the necessary emanations are detected, especially the weak ones running the computer. Measurement setup can be seen in Figure 10: Setup for the first scan. As seen in the image, the probe is hanged above the processor. Distance between the processor and the head of the probe is approximately 4 cm during the time of measurement.

Figure 10: Setup for the first scan

During this scan the target computer was turned on, but the machine was idle without no further task to execute. Results in spectrum can be seen in Figure 11: Results from the first scan. As for this scan and for other whole spectrum scans, the upper blue area indicates the results of power ratio that is relative to 50-ohm load and the lower orange represents power ratio in Watts.

As seen in the Figure 11 the whole spectrum is clearly more active. In relative to 50-ohm load graph, the whole spectrum is at approximately -10 dB when it was approximately at -24 during the initial scan. Also, the difference in the spikes in the relative power and relative to 50-ohm load can be seen clearly above 1 GHz frequency. The initial scan was relatively flat in this area, but during this scan there are clear spikes at 1,5 GHz and 1,64GHz.

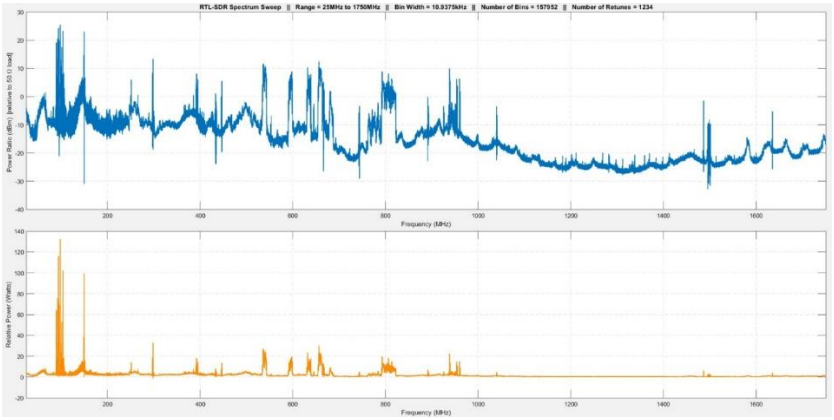

Figure 11: Results from the first scan

When proper frequencies are found certain set of experiments are executed: First experiment is done, when change in spectrum is detected, this proofs us that RTL-SDR can detect emanations sent by functioning computer. Next step is to find changes in the spectrum when a program is started. If this stage is successful, then differences between spectrums in different programs are searched. If this stage is successful, then it seems to be evident that RTL-SDR can be used to detect differences in various programs running in the computer.

Now with the initial setup is completed and the setup working, we can proceed to the next stage and begin the actual data collection. This will be reported in the next chapter.

4 Executing measurements

With the initial setup completed, the next step is the collection of actual data and a few adjustments to ensure the effectiveness of the measurements. In this chapter every phase of the experiment is reported as well as results being analyzed, either at the end of the chapter or in some cases during the ongoing case. For these measurements a modified MATLAB code was used. This code can be seen at appendix 1: modified MATLAB frequency scan code. There is a small addition to the code that saves the measurement to CSV file, which can be later used to access and analyze the datapoints via other programs, like exel or pandas. Otherwise, code is based on the code grabbed from the book Software defined radio using MATLAB & Simulink and the RTL-SDR.

Work order is to first find out the best antenna from the four candidates. Either one of the magnetic loop antennas or the ball antenna. With the manufacturers' description about the stud antenna, it seems to be most inefficient. When this is done and the most accurate antenna is found, then it is time to pinpoint the frequencies, where most of the activity related to the computer is happening. These are all done with the whole frequency range scans. When proper frequencies are found, then it is time to move to the point frequencies that fit inside the 4 MHz constant scan radius and then it is possible execute more precise measurements in these limited areas.

4.1 The perfect antenna

The second step is to find the best antenna for the following experiments. Antenna is the only hardware device that can be affected, so that's why it is important to form an understanding of these: which is the most suitable for the coming measurements.

Setup for this test was relatively simple: Target machine was turned on and it was running idle. Each of the antennas were placed similarly above the processor. Then the spectrum was recorded from these and stored for comparison. Obtained scans can be seen in Figure 12: Comparison scans for the antennas. In the figure sweeps over the spectrum are visible in the order: The upper row are antennas 901 and 902, on the lower row are antennas 903 and 904.

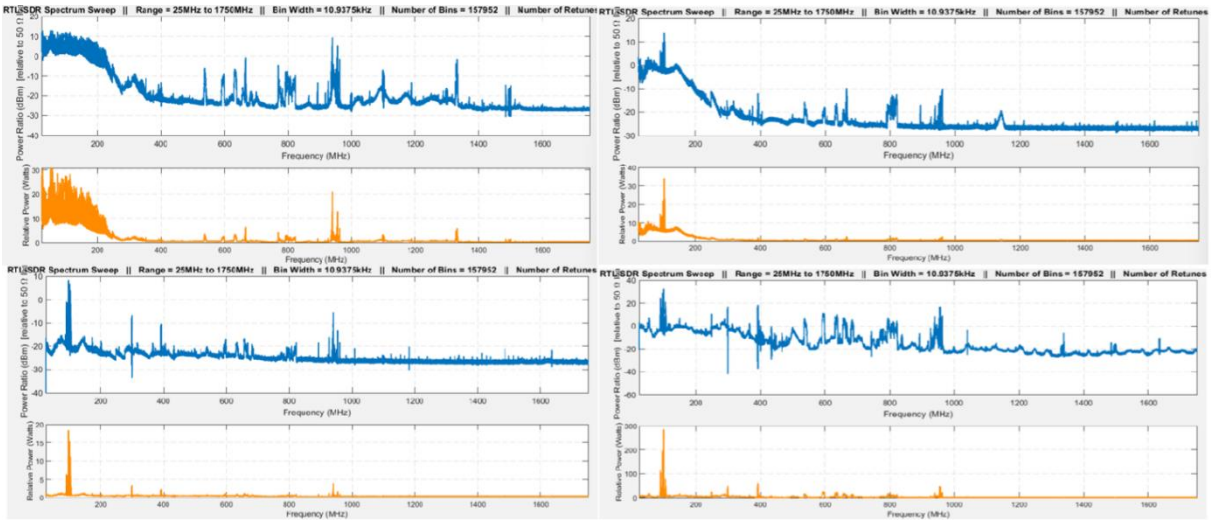

Figure 12:Comparison scans for the antennas

Observations from the Figure 12 Comparison scans for the antennas can be interpreted the following details. When considering the relative power scope (lower in each image) can be seen that the lower frequency range is relatively congested for antennas 901 and 902 up to about 200MHz-250MHz. The mid-range frequencies from 250MHz up to 1 GHz seem relatively similar for each antenna. For some reason 901 has detected signals stronger in around 9050 MHz than other antennas. Antenna 901 excels also in higher frequency area from 1GHz up to 1,7GHz, with more activity in relative power scope. Resolution of the Figure fails here, but there are small spikes in the scope for antenna 903 and 904 also, but these are much more modest than what 901 has.

When considering the scope of the 50-ohm power ratio view it seems that all the observations are around -20dBm range excluding antenna 901 and 902 at lower end frequencies up to around 250MHz. And antenna 904, which seems to detect a lot of signals or noise at up to 950 MHz range. Antenna 903 seems to have the most stable curve, still yielding spikes at certain frequencies. Overall, it seems that 50-ohm relative power scope is much more active than the relative power scope, and for higher frequencies from 1 GHz up to 1,7GHz this scope might yield results more clearly.

Power ratio scope seems to work better with clear signal spikes and with a more toned-down presentation of the scope. With 901 antenna it covers well the higher end of the spectrum. Antennas 903 and 904 seem to pick up signals in power ratio spectrum even in higher frequencies, so these can be used as additional tools here. With Power ratio relative to 50-ohm scope signals are easier to detect, spikes tend to jump very clearly. This is a beneficial property in higher frequency end, where signals tend to be weaker. On lower frequency range

this is more of a hindrance, the scope fills up very fast with spikes. 902 and 903 antennas seem to have a relatively stable profile throughout the whole frequency range. 904 has the strongest activity at below 1 GHz range, but this frequency range seems to have a lot of interference, and the results are difficult to read.

4.2 Usable and interesting frequencies

Next task is to choose frequencies that are monitored more closely with that 5 MHz bandwidth that are available for constant monitoring. This selection is done by comparing scans done above the processor; in first scan the target computer is turned off and in second scan it is turned on. Differences in these results are first compared visually, and then smaller sub frequencies are selected for more detailed analysis. This analysis is done with jupyter lab tool. After suitable subsets are found they will be monitored more closely in the next subchapter.

Interesting points are searched from the scans based on the observations made about the antennas. The 901-antenna seemed to gather the best results from higher frequencies, so data gathered by this antenna was used on the higher frequencies. At lower frequencies other antennas were also used to find interesting spots, though there seemed to be quite a lot of interfering signals at below 750 MHz area.

Based on the information obtained from certain components, there were a few frequency ranges that could be assumed to hold interesting information. The first of these components is the graphic card of the computer, which is integrated on the motherboard of the computer. Based on observations made with the CPU-X program this device should operate on 850 MHz core clock frequency. This frequency falls well within the boundaries of the measuring equipment.

Figures seen here are constructed in jupyter lab from the data that was taken during the scan of the whole range. In the following Figures the blue colour indicates the situation when computer power was off, and the orange colour indicates the situation when the power was turned on as also described in the information box. Here the difference caused by the electricity flowing through components can be observed: The difference in the graphs tells us the effect of the computer to the electromagnetic field. Also, relative power (Watts) reading was used for the figures. This seemed to provide more detailed presentation of the

phenomenon, the relative to 50-ohm load was usually more indistinct and could not provide enough clarity for the more detailed analysis.

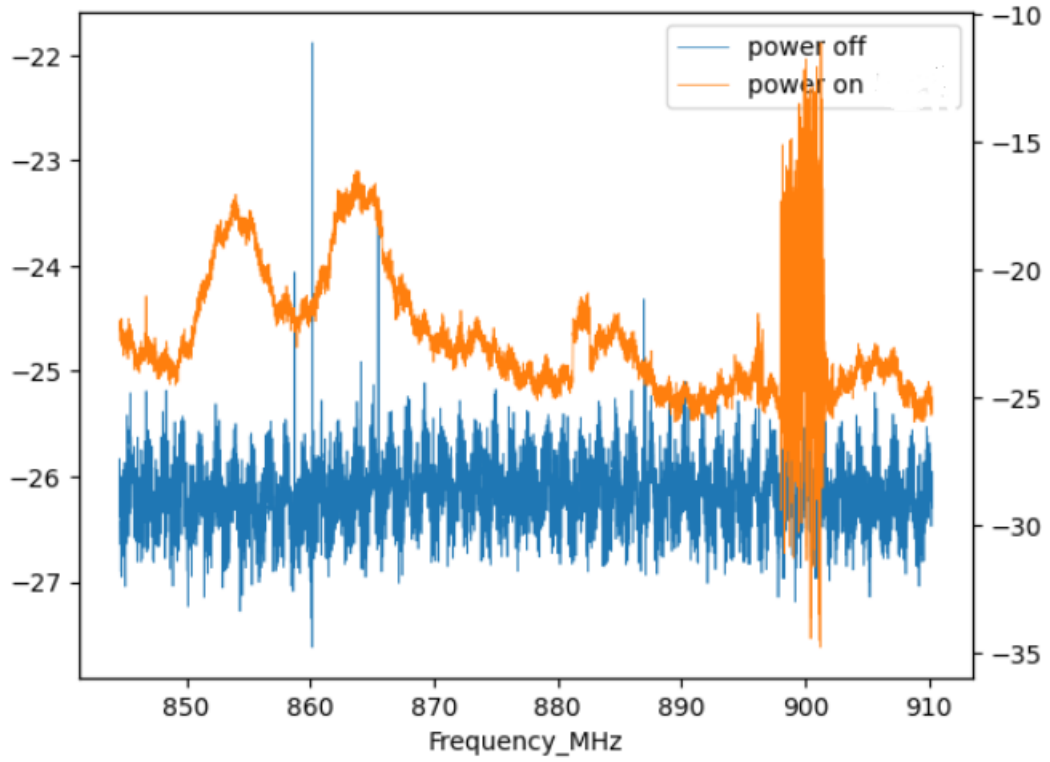

Figure 13:Display card

Prior knowledge was, that graphic card operates at 850 MHz core frequency. Measurement of the scanner can be seen in Figure 13: Display card. As can be seen the spike in 855 MHz could be the spike caused by the display card, as well as the following spike at 865MHz. It is unknown why this spike is a little off, on a higher frequency what it should be, but it might have something to do with the measuring accuracy of the device.

Also, the large spike at around 900 MHz remains a little of the mystery. It does not appear on the other antennas so it might just be an error, or it might be something to investigate. The source of the spike remains also unknown, it does not match on any known frequency. As said before, this is not a surprise: based on the properties of the reflecting surface and on the current running through, frequencies caused by different devices may vary.

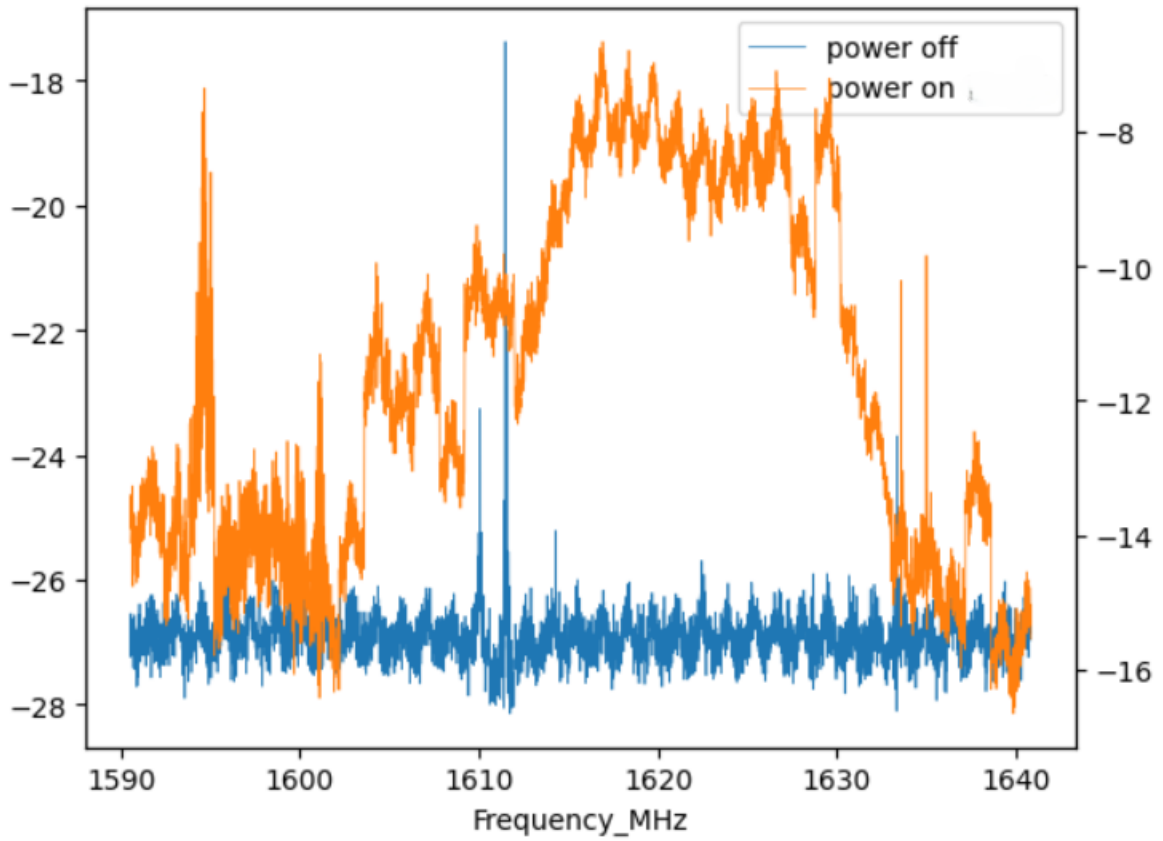

Figure 14:Processor core frequency

Next interesting frequency and the most important one on this research is the base frequency of the processor, that is based on prior knowledge 1591 MHz. As seen in Figure 14: Processor core frequency the first spike correlates well on the prior knowledge on this frequency. It is assumed that the clear spike seen clearly is the core frequency of the processor.

It is assumed that the larger area of elevated received power level is also in relation with the processor: A processor operates at its core speed when idle, when more processing power is needed it can increase the speed up to the maximum capacity. In this case this means that the elevated frequencies range from 1600MHz up to 3400 MHz. This 1605 MHz -1635 MHz area fits well into the lower portion of the elevated area of this processor type. This is the frequency are that will be monitored closer later in this study.

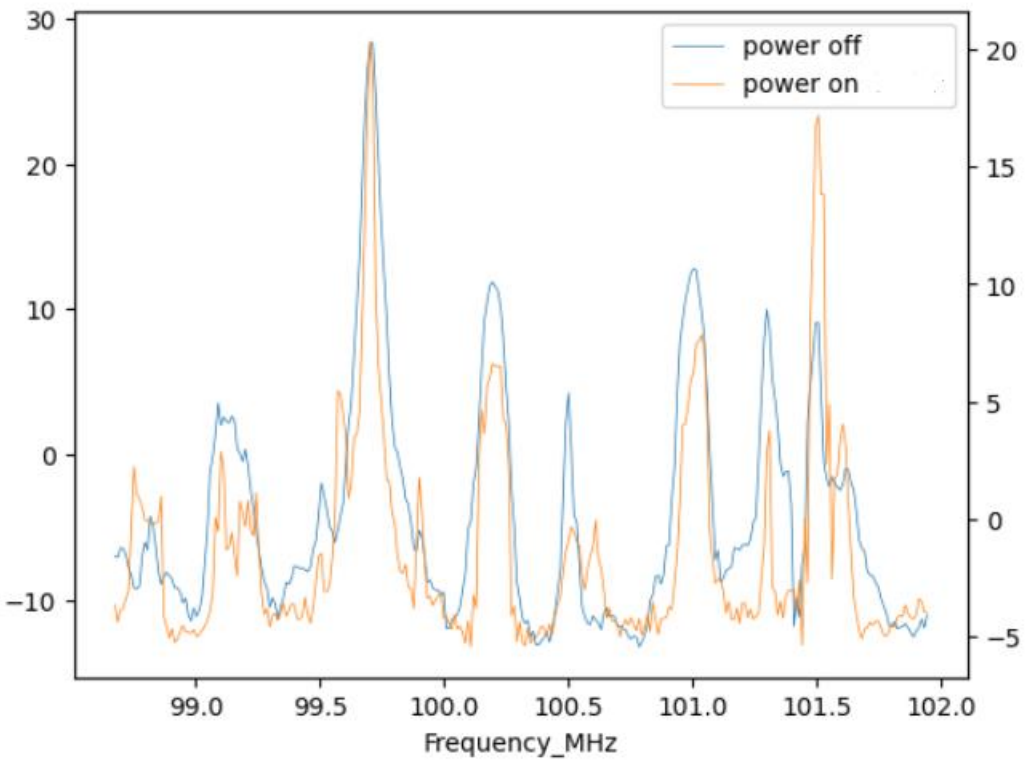

Figure 15:Bus frequency

The last of the pre-defined interesting frequency was the bus operating frequency at 99,77 MHz, that can be seen on Figure 15: Bus frequency. While searching through all antenna data and trying to find a scope where the bus speed activity would show, it was impossible to find any data with any real difference between shut and running computers. There is a single spike at 101,5 MHz, so this might be something related to bus frequencies. At least when considering that all other frequencies presented here were a little upshifted. Still the evidence of this is relatively weak and it is assumable that the spike seen here is just an anomality.

There seems to be three reasons for the difficulties detecting bus frequencies: First of tall, assumably the current flowing in bus lanes I relatively weak and thus it does not yield a strong signature. Secondly measurements were taken so that the antenna was above the processor, while bus connections run under the processor at mainboard. In this manner there would be a large obstacle between the source and the antenna. Thirdly many of the radio broadcasts are delivered in this frequency range. This floods the bandwidth, so occurrences in weak currents does not stand out.

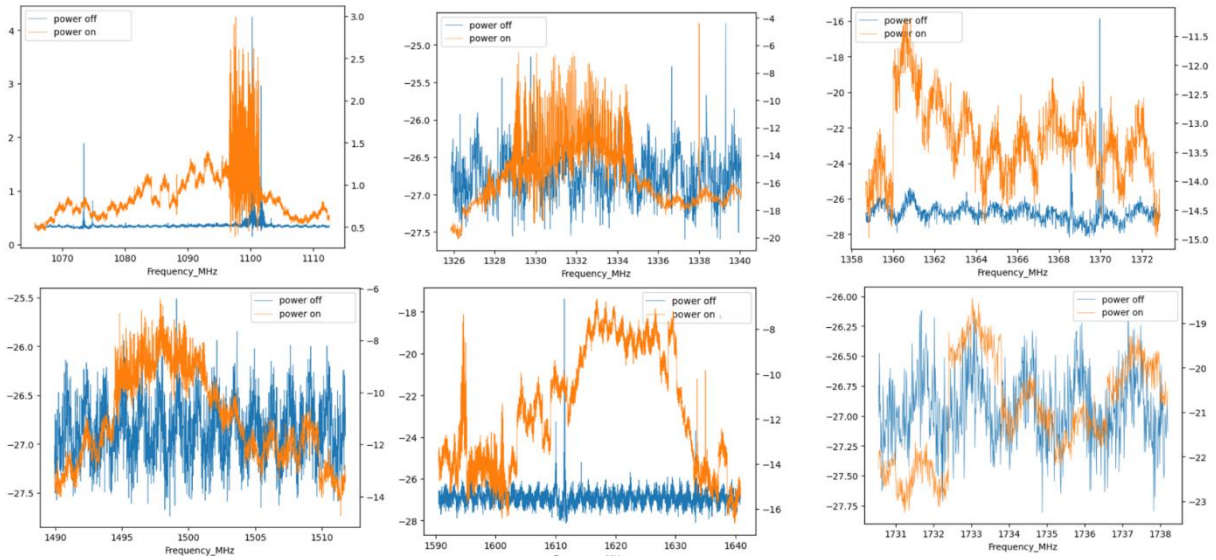

Figure 16: Frequencies that stand out

In addition to mentioned known interesting frequencies, there were planet of frequencies which stud out while analysing the spectrum. At least in the higher end of the spectrum, over 1,5 GHz range, there were significantly increased power levels when comparing running computer to shut down computer. It seems that one could detect running computer from shut down one in this frequency are just by looking the average received amplitude.

There are some of these clearly visible differences presented in the Figure 16: Frequencies that stand out. Just to clarify; there were plenty of frequencies where the difference between frequencies were present. Here a just some of those to demonstrate the differences between running and shut computer. There seem to be clear spiker around 1100 MHz, 1320 MHz, 165MHz and 1620 MHz. These might be harmonic frequencies to some lower frequencies, or they might be indications of occurrences in the processor.

As mentioned earlier in this work, not all the phenomena follow the strict frequency in which they appear, some might jump from frequency to another based on the current and properties of the reflecting surface. There are also more obscure observations around 1500 MHz and 1734 MHz. These might just be strengthened external signals, that for some reason happen to stand out. They might also be something related to processor or other parts of the computer.

4.3 More detailed analysis of the interesting frequencies

In this part of the study, the properties of the RTL-SDR device can be utilized in full potential. In this part of the study the 3 MHz frequency that the device can monitor is exploited to get constant and almost live monitoring on the phenomenon in the bandwidth.

The monitoring frequency is limited due to the high stress the full 4 MHz frequency sets to the analyzing computer. Also, samples are limited to 6 seconds because the file size for longer scans grows very large. This makes it possible to make more accurate observations about phenomenon on current bandwidth. These can be analyzed in more detail, and it might be possible to gain additional details.

There were several interesting frequencies discovered in the previous subchapter. The CPU core frequency and the graphic card core frequency were known already, though they got confirmed. They were Other interesting frequencies that were found are also tested here. Some of the interesting frequencies that are inspected here are the following: 1100 MHz, 1320 MHz, 1581 MHz, 1605 MHz, 1620 MHz and 1676 MHz. Most of the frequencies tested are around 1600 MHz. The reason for this is, that the target computer core processor speed is at 1600 MHz, so this should provide most interesting results. That odd frequency spike at 900 MHz will also be investigated, if something interesting comes up in there.

The procedure for samples is relatively simple: When the scan is initiated around desired frequency, the antenna is first kept away from the target computer and after 3 seconds, it is put on the processor. This yields spectrum where change of the frequencies close to the processor are seen. After this the scan is done again, this time the antenna is kept on the processor constantly, so that whole frequency is captured. Received spectrum is captured and shifted to match real frequencies. This capture is useful asset in future: these captures can be used to verify that certain activities or areas have similarities and thus they can be identified based on their unique signal composition. MATLAB code for these is based on the code in Software Defined Radio using MATLAB & Simulink and the RTL_SDR with a small addition. This code is stored in appendix 2 Modified MATLAB code for analysis. Finally, a half second slice is taken, from these possible timings are visible to the viewer.

First part of the analysis is to confirm that the emanations captured are really from the processor. This should be seen as clear cut-off in the time/amplitude spectrum of the spectra. In power spectrum the real frequency and related signal should be visible. This graph can be used to analyse the power versus frequency properties of the signals and later it can be used to identify the phenomenon in the computer. These are the Fourier transformed signals received around the centre frequency.

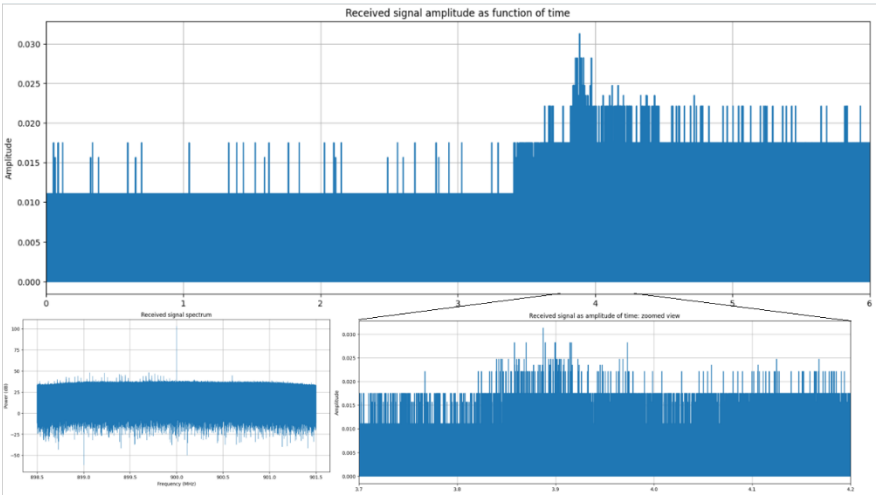

Figure 17:900 MHz

In Figure 17: 900 MHz is a view on 900 MHz spectrum, with 3 MHz bandwidth. This frequency was chosen because of the interesting spike detected there during the initial analysis of the whole spectrum. That spike is still detectable, so it was not a passing anomaly. It is clearly visible also on the zoomed-out view at the bottom right of the Figure and its signature on the spectrum is evident. Other interesting feature in this capture are the relatively constant spikes throughout the upper Figure, which lay about 0,3 seconds apart regardless of if the antenna is on the processor or not. They seem to be caused by background radiation.

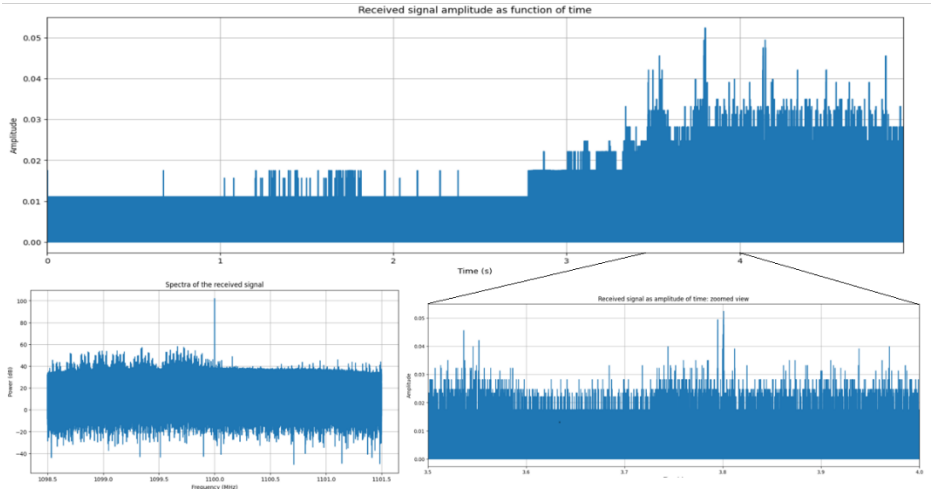

Figure 18:1100 MHz

In Figure 18: 1100 MHz can clearly be seen that the amplitude shifts around 3 second mark, meaning that the captured signal from 3 second up to 6 second are from the processor. In the lower left corner can be seen the spectra of the captured signal shifted to correspond the right frequency. Here can be seen, that most of the activity captured is between 1098 MHz and 1100 MHz. At lower right corner there is a detailed view on the spectrum. There are clearly a

few spikes at the upper part of the Figure that are clearly visible. At least three of these spikes seem to have some sort of uniform form: with wider lower part and clear spike at the end. This might indicate that these frequencies are related to the target computer. The exact source remains hidden: there should not be any known emissions at 1100 MHz.

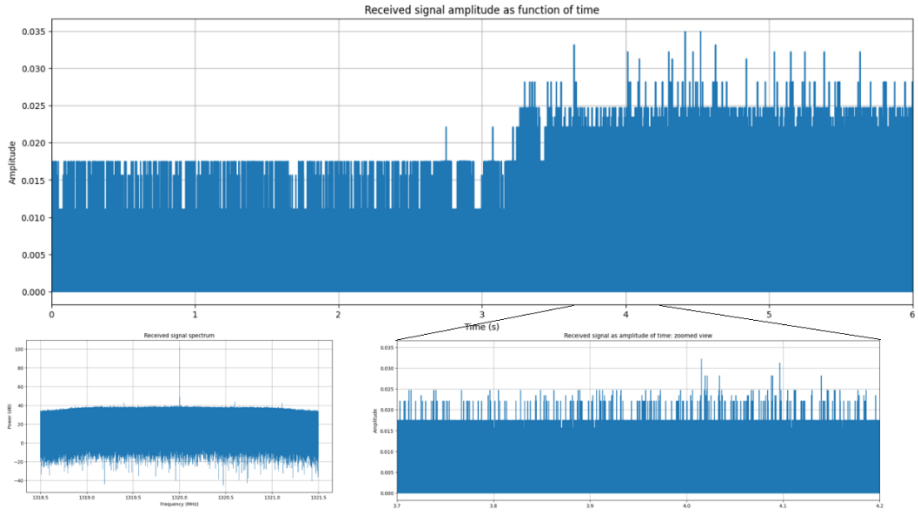

Figure 19:1320MHz

In Figure 19: 1320 in the zoomed down image there seems to be clear and distinguishable timings that seem to stand out from Figures of the smaller frequencies. Also, the transition in the frequency field is very clear: Before 3 sec, there are hardly any spikes visible, after the 3 sec transition the spikes are clear, indicating that they belong to the processor spectrum. This same effect is even more clear in Figure 20: 1581 MHz strengthening the assumption that the processor frequencies are present in these spectrums. In Figure 20 the spikes between 3,5 sec up to 6 sec seem to have some sort of formal structure as well as formal composition Again this might indicate that these spikes are caused by the processor operations.

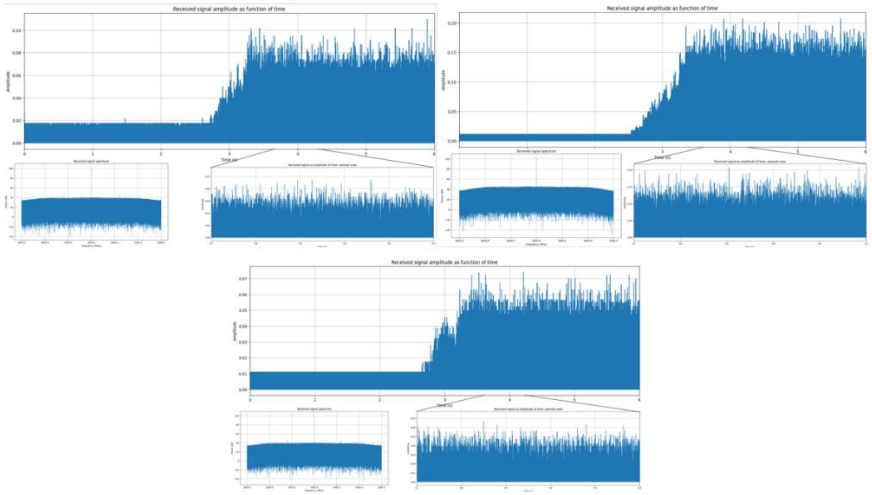

Figure 20:Combined spectrum of 1605MHz 1620MHz and 1676MHz

On the final image of the spectrums are the spectrums of 1605 MHz at upper left corner, 1620 MHz at upper right corner and 1676 MHz at lower part of the Figure. These Figures were combined because the formation of the spectrum of all these frequencies is relatively similar. The sharp transition when antenna is brought down to the processor is evident, as well as the rapidly increased activity during the last 3 seconds of the whole 6 sec scan. Also, the zoomed in Figure seem to be relatively similar, there is so much activity present that patterns are hard to recognise from these Figures. The uniformity of these Figures is not a huge surprise: They are all taken from the frequencies close to processor core frequency and during these measurements, the processor was on idle, so it seems clear, that the core processor speed is mostly present, not the accelerated frequencies, which end up to 3,4 GHz.

Last task is to predict ongoing processes in the target computer to be able to analyse the tasks that the machine is executing. This was done while reading the processor emanations and simultaneously increasing the workload of the target computer The workload was raised from 1 % idle speed up to 25% workload during monitored 6 sec timeframes. Workload was monitored on CPU-X tool, that provides current stress on the processor. This workload should be seen in the scope in increased activity. Results can be seen in Figure 21: Increased workload.

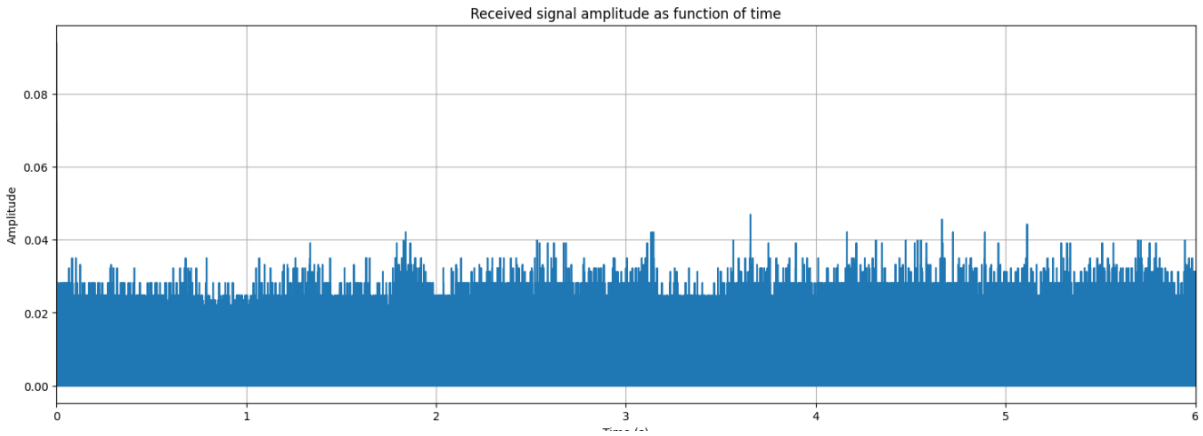

Figure 21: Increased workload

In Figure 21 the initial acceleration of the processor can be seen in around 1,8 second mark. This phenomenon was persistent through several scans: The workload increases and simultaneously the frequency that the processor operates on. The demand raises the processor frequency rapidly close to 3,2 GHz, that is outside the scope of RTL-SDR device. The initial activation can be seen as a small increase in activity, but this increase does not continue, while the target computers processor accelerates.

4.4 Analysis of the findings

Altogether the measurements went as planned and there were no big surprises. The measurement phase was successfully executed. Antennas and the RTL-SDR worked as described. The antennas provided varying results from different frequency ranges as they are designed. The collected frequency scan seen at the bottom left of the Figures makes it possible to store and use the spectrum detected. This data can be used for example for machine learning models to recognize these areas from the overall spectrum.

Different properties of the antennas proved to be fruitful: different areas could be investigated with different antennas. The 901 magnetic loop antenna was reliable on higher frequencies, while the 903 small magnetic loop antenna and 904 electric field antennae were more effective at the lower end of the spectrum. While it was evident, that the antenna type will affect the results of the scans, it was interesting to notice how noticeable the differences were from each other. It is also good to remember that the antennas were already like each other belonging to the same set.

A small surprise was on the hardware side during the more detailed measurements, how large the data files were and how burdensome it was for the computer to run the tests on the JUPYTER lab and Pandas. There were a few occasions when the analyzing computer ran out of memory while executing the tests. For further research, a more capable computer might be recommended: With 3 MHz bandwidth and 6 sec measurement time, the CSV files where the data of the scans were stored were the size of about 650 Mb, so it is no wonder that the analyzing laptop was under heavy stress and thus slowing down.

Full spectrum scans provided plenty of interesting points to study further. These scans provided also a good overview of the spectrum and on the other congested frequencies, that were hard to analyze any further. It is a shame that the bus frequency on 99,77 MHz was overcrowded with radio broadcast frequencies and thus could not be investigated any further. It also seemed that on the higher end of the spectrum, the clearer the bandwidth was: There was only a little congestion there and useful frequencies stood out clearly. At the lower end of the spectrum, there was much more noise and interference.

The interesting frequencies that were picked out for further analysis were on specific points of interest or around the core processor frequency of about 1600MHz. Interesting frequencies that were investigated were clear and strong spike at 900 MHz and clear and strong signals at

1100 MHz, 1320 MHz and 1581 MHz. These were detected and analyzed though the source of these emanations remained uncertain. It is evident that they relate to the computer and most likely to the processor because they were clearly stronger close to the running processor. As mentioned earlier, some emanations may appear on arbitrary frequencies based on the cause of the emanation and on the properties of the reflecting surface.

Detailed results from the 1600MHz range showed the relatively uniform profile of the processor emanations to be detected through these wider range of frequencies. It is evident that these scans are similar among each other, and they differ from other scans done here. The similarities are explained mainly because of the strong signa strength of the running processor at these frequencies.

It was unfortunate that the increased workload could not be monitored with the current setup. Although initial acceleration was detectable in several scans, the processor frequency quickly outpaced capacity to monitor frequencies. Increased activity could be monitored if the frequency range of the processor would stay close to idle, but this would require possible limitations done to the processor frequency.

5 Conclusions

Findings for theoretical questions were given in chapter 2.4, so this chapter concentrates on the practical side of the research about the RTL-SDR dongle in collecting data from the target computer. The research questions that are answered here were:

1. Can RTL-SDR collect useful data from the computer: Is the resolution enough to gain usable information?

2. Can this data to be used in detecting different kinds of activity in the computer, can it detect processes of the computer?

For the first question presented here the answer is yes: RTL-SDR setup used in this research can provide useful data from the target computer. Interesting frequencies like processor core frequency could be found and monitored. For example, graphic card frequency was within the range of the gauge and could be most likely monitored as well. Bus frequencies were also within the range of the device, but the interference of external frequencies prevented execution of further analysis. With further analysis there should be more obtainable information about the timings and cycles inside the computer in the frequency range of the gauge.

For the second question the answer is partially: While the increasement in the processor cycles were detectable with the measurement setup, the full scale of the processor increasement was left undetectable. The operating frequency of the processor quickly arose above the maximum detection range of the RTL-SDR and thus the full spectrum caused by the processor was left unnoticed. Some traces of the increased activity however remained observable, and these could be utilized to construct a detection device for certain type of activity.

Performance of the RTL-SDR during this research gives an encouraging example on how relatively cheap SDR device can be utilized in scanning and analysing wide and complex fields of signals and outcomes. A probe capable to detect higher frequencies, and with methods introduced in this research it could be probable to build a device that could monitor the whole frequency range needed to analyse the functions of the processor in detail.

RTL-SDR combined with the MATLAB is a versatile tool, thanks to the MATLAB: s own communication toolbox as well as the community support in applications and advice that

were available online. The free to use software and modest price of the RTL-SDR dongle gives a plenty of opportunities to build affordable but effective devices, even on a larger scale. This could make possible to build several single purpose tools for various purposes without extravagant expense. This makes engineering these approaches rather simple: Without the need to combine complex code for several applications, one dongle can handle a simple task with ease and the overall complexity of the solution remains low.

The antenna selection provided useful room for different kinds of situations depending on the overall congestion of the frequency: The magnetic field antennas seemed to handle better in the higher end of the frequency spectrum, electric field antennas seemed to yield a little more readable result at lower end of the spectrum. Selected antenna setup was sufficient for this kind of limited frequency research. While the antenna selection for this research was adequate, if measurements are aimed at wider frequency scale, a different selection of antennas are needed for fully utilizing the wider frequency spectrum.

The properties of the RTL-SDR gave an interesting obstacle: The limited 4 MHz frequency range required rather complicated code so that the whole spectrum could be scanned at one go. This was a necessity for the study to succeed: probing blind with 4MHz bandwidth would have taken a long time to scan the whole range of 1.7GHz through. Even if the whole range would be manually checked, there might be signals that go unnoticed. The solution to scan the whole range in 2.8MHz slices and write them to a file gives the possibility to return to old scan file and analyse it again.

JUPYTER lab and Pandas provided to be a useful tool when analysing the CSV-files that were produced alongside the measurements. These made it possible to store the data and further analyse it. In Pandas there are plenty of machine learning algorithms that can be utilized for further analysis of obtained results. This would be a beneficial approach in further studies of this field. The size of obtainable datafiles was a small surprise. It could be wise to implement ways to reduce the size of the capture file. This could make it possible to run longer analysis and to use wider bandwidth to obtain more information per run.

There might be more useful information in the details of the already captured datafiles, that should be investigated even further. At least the captured frequency spectrum of findings might yield useful data about the phenomenon at hand. It might be feasible to develop a code that captures data within intervals so that long lasting measurements might be possible even

with the limited resources. With current data acquisition rate, ten seconds is maximum length of the capture and even this would yield a 1 Gb datafile.

As discussed earlier in the introduction, this kind of research is already executed relatively widely in both tempest environment and as side channel analysis for IoT environment. The tempest approach tries to minimize emanations to outside, which is a reasonable protection method in the light of this research: A lot of information can be captured from emanations. Side channel analysis focuses more on breaking the encryption of the IoT devices. While this tool seems to have a lot of useful implications, analysing the real time operating system or no operating system of the IoT device provides a little different environment than what is in this research.

This study tried to implement a low-cost and helpful analysis foundation for tabletop computers. As mentioned, the frequency range did not allow full implementation of desired data analytics tool, but the possibility to build one still exists, with a receiver capable of wider frequency range. The difference of this research from those mentioned is that the goal of this work is to build beneficial tool for computer owner to monitor the activity of one's own machine. This might be especially useful for servers, routers and other active devices on the internet, that has a relatively stable operating model so that variations from the norm would be easily distinguishable.

The key takeaway of this work would be the proven possibility to analyse processor frequencies with RTL-SDR setup as described here as well as the idea of implementing these to provide protection for the devices under threat from the internet. For this work to gain its full potential, it requires a more suitable gauge and more processing power for vast datafiles that holds the data of the measurements.

The main things to improve in this research would be to acquire second computer to verify the results obtained from the initial machine. This would provide more robust results, when some phenomenon detected might be unique to target computer. It is also evident, that the use of wider range gauge would have been beneficial, so that higher frequencies and wider bandwidth could be analysed. This might set rather tough demands for the analysing computer: even now the datafiles are large and computer ran out of memory on several occasions, with larger datafiles even more computing power would be required.

Implemented analysis tools might also be improved, and even more detailed samples could be provided. Now the analysis consisted of frequency analysis and amplitude analysis. Analysing spikes and their distance, mean or average amplitude strengths or highest frequency spikes might yield further understanding to the information content of the findings. RTL-SDR might enable other analysing methods that are not considered here, but which might yield useful information about the information content revealed by the emanations.

For further research and to advance the findings done here it would be necessary to use gauge that could detect higher frequencies to obtain better understanding about the processor and its current state. A spectrum analyser would also be advisable to use: Searching interesting frequencies might be sone in more convenient way with proper analyser. Then the interesting frequencies could just be monitored with SDR device. Also analysing means could be improved: There are more refined ways of extracting data from collected samples. Also, a clever format of storing the data should be developed. Now datafiles were very large.

When enough samples are collected from sufficient frequency range, these results could be fed to the machine learning algorithm to classify them to form a classifier for different kinds of signals emitted by the processor or computer. It could be useful to identify malicious encryption algorithm or perhaps bitcoin mining algorithm from an industrial computer. Classifying malicious signals and monitoring the devices might lead to robust form of securing industrial computers and active devices on the internet.

6 References

[1] V. Antić *et al.*, 'Protecting Data at Risk of Unintentional Electromagnetic Emanation: TEMPEST Profiling', *Applied Sciences 2024, Vol. 14, Page 4830*, vol. 14, no. 11, p. 4830, Jun. 2024, doi: 10.3390/APP14114830.

[2] M. Martin, F. Sunmola, and D. Lauder, 'A TEMPEST vulnerability prediction method for cyber security practitioners', *Alexandria Engineering Journal*, vol. 78, pp. 561–575, Sep. 2023, doi: 10.1016/J.AEJ.2023.07.059.

[3] J. Danial, D. Das, S. Ghosh, A. Raychowdhury, and S. Sen, 'SCNIFFER: Low-Cost, Automated, Efficient Electromagnetic Side-Channel Sniffing', *IEEE Access*, vol. 8, pp. 173414–173427, 2020, doi: 10.1109/ACCESS.2020.3025022.

[4] Randolph, M.; Diehl, W. Power Side-Channel Attack Analysis: A Review of 20 Years of Study for the Layman. *Cryptography* **2020**, *4*, 15. https://doi.org/10.3390/cryptography4020015

[5] M. Kryjevskaia, M. R. Stetzer, and P. R. L. Heron, 'Student understanding of wave behavior at a boundary: The relationships among wavelength, propagation speed, and frequency', *American Journal of Physics*, vol. 80, no. 4, pp. 339–347, Apr. 2012, doi: 10.1119/1.3688220.

[6] Por, E.; van Kooten, M.; Sarkovic, V. *Nyquist–Shannon Sampling Theorem*; Leiden University: Leiden, The Netherlands, 2019; Volume 1

[7] M. A. Heald and J. B. Marion, *Classical electromagnetic radiation*. Courier Corporation, 2012.

[8] A. Clements, 'Principles of computer hardware', p. 656, 2006, Accessed: Jul. 09, 2024. [Online]. Available: https://global.oup.com/academic/product/principles-of-computer-hardware-9780199273133

[9] RTL-SDR.COM 'About RTL-SDR'. Accessed: Apr. 09, 2025. [Online]. Available: https://www.rtl-sdr.com/about-rtl-sdr/

[10] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi, 'The EM Side—Channel(s)', in *Cryptographic Hardware and Embedded Systems - CHES 2002*, B. S. Kaliski, çetin K. Koç, and C. Paar, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 29–45.

[11] M. Prvulovic, A. Zajic, R. L. Callan, and C. J. Wang, 'A Method for Finding Frequency-Modulated and Amplitude-Modulated Electromagnetic Emanations in Computer Systems', *IEEE Transactions on Electromagnetic Compatibility,* vol. 59, no. 1, pp. 34–42, Feb. 2017, doi: 10.1109/TEMC.2016.2603847.

[12] M. Devi and A. Majumder, 'Side-Channel Attack in Internet of Things: A Survey', in *Applications of Internet of Things*, J. K. Mandal, S. Mukhopadhyay, and A. Roy, Eds., Singapore: Springer Singapore, 2021, pp. 213–222.

[13] M. Valkama, 'Advanced I/Q signal processing for wideband receivers models and algorithms', 2001, Accessed: Apr. 16, 2025. [Online]. Available: https://trepo.tuni.fi/handle/10024/115010

[14] I. M. Sefton, ‘Understanding electricity and circuits: What the text books don’t tell you’, in *Science Teachers’ Workshop*, Citeseer, 2002.

[15] Huang, Y. (2021). *Antennas: From Theory to Practice*. United Kingdom: Wiley.

[16] A. Bindal, ‘Fundamentals of Computer Architecture and Design’, *Fundamentals of Computer Architecture and Design*, pp. 1–533, Aug. 2017, doi: 10.1007/978-3-319-25811-9.

[17] T. Schubert and Kim Ernest, *Fundamentals of Electronics Book 1 Electronic Devices and Circuit Applications*. Morgan & Claypool, 2015. doi: 10.2200/S00598ED1V01Y201409DCS045.

[18] S. Kaxiras, M. Martonosi - Google-kirjat’, ‘*Computer Architecture Techniques for Power-efficiency*, Springer Nature Switzerland AG 2008, doi: https://doi.org/10.1007/978-3-031-01721-6

[19] M. C. Pereira, P. V. Viera, A. L. A. Raabe, and C. A. Zeferino, ‘A basic processor for teaching digital circuits and systems design with FPGA’, *SPL 2012 - 8th Southern Programmable Logic Conference*, 2012, doi: 10.1109/SPL.2012.6211804.

[20] ‘Motherboard and user experience | IEEE Conference Publication | IEEE Xplore’. Accessed: Apr. 17, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6596346

[21] K. N. Vikram and V. Vasudevan, ‘Hardware–software co-simulation of bus-based reconfigurable systems’, *Microprocess Microsystems*, vol. 29, no. 4, pp. 133–144, May 2005, doi: 10.1016/J.MICPRO.2004.07.004.

[22] ‘CPU Speed: What Is CPU Clock Speed? | Intel’. Accessed: Jul. 17, 2024. [Online]. Available: https://www.intel.com/content/www/us/en/gaming/resources/cpu-clock-speed.html

[23] N. Kim and K. Choi, ‘Exploration of trade-offs in the design of volatile STT–RAM cache’, *Journal of Systems Architecture*, vol. 71, pp. 23–31, Nov. 2016, doi: 10.1016/J.SYSARC.2016.06.005.

[24] R. C. Martin, *The Hardware/Software Interface , ARM ® Edition*. 2017. Accessed: Jul. 17, 2024. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:The+Clean+Coder+a+code+of+conduct+for+professional+programmers#0

[25] J. S. Meena, S. M. Sze, U. Chand, and T. Y. Tseng, ‘Overview of emerging nonvolatile memory technologies’, *Nanoscale Res Lett*, vol. 9, no. 1, pp. 1–33, Sep. 2014, doi: 10.1186/1556-276X-9-526/FIGURES/29.

[26] M. Jung and M. Kandemir, ‘Revisiting widely held SSD expectations and rethinking system-level implications’, *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1 SPEC. ISS., pp. 203–216, Jun. 2013, doi: 10.1145/2494232.2465548.

[27] D. Heath, ‘Parallel RAM from Cyclic Circuits’, Sep. 2023, Accessed: Apr. 27, 2025. [Online]. Available: https://arxiv.org/pdf/2309.05133

[28] S. J. Kim *et al.*, 'The bottom of the memory hierarchy: Semiconductor and DNA data storage', *MRS Bull*, vol. 48, no. 5, pp. 547–559, May 2023, doi: 10.1557/S43577-023-00510-X/FIGURES/7.

[29] Stefanos. Kaxiras and Margaret. Martonosi, 'Computer architecture techniques for power-efficiency', p. 207, 2008.

[30] S. Aga, S. Jeloka, A. Subramaniyan, S. Narayanasamy, D. Blaauw, and R. Das, 'Compute Caches', *Proceedings - International Symposium on High-Performance Computer Architecture*, pp. 481–492, May 2017, doi: 10.1109/HPCA.2017.21.

[31] B. Jacob, S. W. Ng, and D. T. Wang, *Memory Systems: Cache, DRAM, Disk*. Elsevier, 2007. doi: 10.1016/B978-0-12-379751-3.X5001-2.

[32] J. Michael, J. Flynn, W. Luk, *Computer system design : system-on-chip*. Wiley, 2011.

[33] B. Jacob, S. W. Ng, and D. T. Wang, *Memory Systems: Cache, DRAM, Disk*. Elsevier, 2007. doi: 10.1016/B978-0-12-379751-3.X5001-2.

[34] M. Kovalev, S. M. Müller, and W. J. Paul, *A pipelined multi-core mips machine: Hardware implementation and correctness proof*, vol. 9000. Springer Verlag, 2014. doi: 10.1007/978-3-319-13906-7/COVER.

[35] B. W. Mezger, D. A. Santos, L. Dilillo, C. A. Zeferino, and D. R. Melo, 'A Survey of the RISC-V Architecture Software Support', *IEEE Access*, vol. 10, pp. 51394–51411, 2022, doi: 10.1109/ACCESS.2022.3174125.

[36] G. Blanchet and Bertrand. Dupouy, *Computer Architecture.* Wiley, 2013.

[37] P. Koppe *et al.*, *Reverse Engineering x86 Processor Microcode*. 2017, 26th USENIX Security Symposium (USENIX Security 17), 1163-1180

[38] S. Ramprakash, D. P.- Memory, and undefined 2022, 'A REVIEW ON ANALYSIS OF 32-BIT AND 64-BIT RISC PROCESSORS', *academia.edu*, Accessed: Jul. 10, 2024. [Online]. Available: https://www.academia.edu/download/89565643/IRJET_V9I3303.pdf

[39] W. Mueller, '1st International QEMU Users' Forum', Grenoble, Mar. 2011.

[40] J. Ledin, *Modern Computer Architecture and Organization*. Packt Publishing, 2020.

[41] A. Khan, M. Vijayaraghavan, S. Boyd-Wickizer, and Arvind, 'Fast and cycle-accurate modeling of a multicore processor', *ISPASS 2012 - IEEE International Symposium on Performance Analysis of Systems and Software*, pp. 178–187, 2012, doi: 10.1109/ISPASS.2012.6189224.

[42] J. Shen and M. Lipasti, *Modern processor design: fundamentals of superscalar processors*. 2013, Waveland Press, Inc, pp. 39-94

[43] H.-S. Kim and J. E. Smith, 'An Instruction Set and Microarchitecture for Instruction Level Distributed Processing', 2002.

[44] Brian. Ward, 'How Linux Works : What Every Superuser Should Know, 3rd Edition', p. 464, 2021, Accessed: Feb. 21, 2025. [Online]. Available:

https://books.google.com/books/about/How_Linux_Works_3rd_Edition.html?id=73okEAAAQBAJ

[45] D. B. De Oliveira, D. Casini, and T. Cucinotta, 'Operating System Noise in the Linux Kernel', *IEEE Transactions on Computers*, vol. 72, no. 1, pp. 196–207, Jan. 2023, doi: 10.1109/TC.2022.3187351.

[46] H. Subhi Malallah *et al.*, 'Article no.AJRCOS.68517 Reviewers: (1) Ramjeet Singh Yadav, Ashoka Institute of Technology and Management, India. (2) Guruprakash CD, Sri Siddhartha Academy of Higher Education', *Asian Journal of Research in Computer Science*, vol. 8, no. 3, p. 68517, 2021, doi: 10.9734/AJRCOS/2021/v8i330201.

[47] B. Khan and A. Koch, 'DeLiBA-K: Speeding-up Hardware-Accelerated Distributed Storage Access by Tighter Linux Kernel Integration and Use of Modern API Tenth International Workshop on Heterogeneous High-performance Reconfigurable Computing H2RC'24 SC24: The International Conference for High Performance Computing, Networking, Storage and Analysis Background (primer on I/O storage) and Research Problem', vol. 1, p. 32, 2024.

[48] B. Khan and A. Koch, 'DeLiBA-K: Speeding-up Hardware-Accelerated Distributed Storage Access by Tighter Linux Kernel Integration and Use of a Modern API', *Proceedings of SC 2024-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 531–544, 2024, doi: 10.1109/SCW63240.2024.00075.

[49] J. M. Mitola, 'Software Radios Survey, Critical Evaluation and Future Directions', *IEEE Aerospace and Electronic Systems Magazine*, vol. 8, no. 4, pp. 25–36, 1993, doi: 10.1109/62.210638.

[50] A. A. Del Barrio *et al.*, 'HackRF + GNU Radio: A software-defined radio to teach communication theory', *International Journal of Electrical Engineering and Education*, vol. 60, no. 1, pp. 23–40, Jan. 2023, doi: 10.1177/0020720919868144

[51] B. A. Fette, 'History and Background of Cognitive Radio Technology', *Cognitive Radio Technology*, pp. 1–26, 2009, doi: 10.1016/B978-0-12-374535-4.00001-1.

[52] R. W. . Stewart, K. W. . Barlee, D. S. W. . Atkinson, and L. H. . Crockett, 'Software defined radio using MATLAB & Simulink and the RTL-SDR', p. 646, 2017.

[53] Rafaelmichro.com, 'TV Receiver - Global Leader in RFIC Technology'. Accessed: Mar. 05, 2025. [Online]. Available: https://www.rafaelmicro.com/tv-receiver/#

[54] 'High Performance Low Power Advanced Digital TV Silicon Tuner Datasheet', 2011. [Online]. Available: www.rafaelmicro.com

[55] ETS-Lindgren, 'Model 7405 Near Field Probe Set User Manual', Cedar Park, May 2013.

[56] The MathWorks, Inc, 'MATLAB'. Accessed: Apr. 08, 2025. [Online]. Available: https://www.mathworks.com/products/matlab.html

[57] RTL-SDR.com, ‘RTL-SDR Support from Communications Toolbox - Hardware Support - MATLAB & Simulink’. Accessed: Apr. 08, 2025. [Online]. Available: https://www.mathworks.com/hardware-support/rtl-sdr.html

[58] Traficom, ‘Radioasemat Suomessa | Traficom’. Accessed: Apr. 11, 2025. [Online]. Available: https://traficom.fi/fi/viestinta/tv-radio-ja-muut-mediapalvelut/radioasemat-suomessa

[59] Traficom, ‘Matkaviestinverkkojen taajuudet ja luvanhaltijat | Traficom’. Accessed: Apr. 11, 2025. [Online]. Available: https://www.traficom.fi/fi/viestinta/radioluvat-ja-taajuudet/taajuuksien-suunnittelu-ja-kaytto/matkaviestinverkkojen-taajuudet

Appendices

Appendix 1 Modified MATLAB frequency scan code

```
function rtlsdr_rx_specsweep

% PARAMETERS (can change)
location            = '7405_903_first scan_vertical';    % location used for
figure name
start_freq          = 25e6;         % sweep start frequency
stop_freq           = 1750e6;       % sweep stop frequency
rtlsdr_id           = '0';          % RTL-SDR stick ID
rtlsdr_fs           = 2.8e6;        % RTL-SDR sampling rate in Hz
rtlsdr_gain         = 40;           % RTL-SDR tuner gain in dB
rtlsdr_frmlen       = 4096;         % RTL-SDR output data frame size
rtlsdr_datatype     = 'single';     % RTL-SDR output data type
rtlsdr_ppm          = 0;            % RTL-SDR tuner parts per million correction
% PARAMETERS (can change, but may break code)
nfrmhold            = 20;           % number of frames to receive
fft_hold            = 'avg';        % hold function "max" or "avg"
nfft                = 4096;         % number of points in FFTs (2^something)
dec_factor          = 16;           % output plot downsample
overlap             = 0.5;          % FFT overlap to counter rolloff
nfrmdump            = 100;          % number of frames to dump after retuning (to
clear buffer)

% CALCULATIONS
rtlsdr_tunerfreq  = start_freq:rtlsdr_fs*overlap:stop_freq;     % range of tuner
frequency in Hz
if( max(rtlsdr_tunerfreq) < stop_freq )                         % check the whole
range is covered, if not, add an extra tuner freq
    rtlsdr_tunerfreq(length(rtlsdr_tunerfreq)+1) =
max(rtlsdr_tunerfreq)+rtlsdr_fs*overlap;
end
nretunes = length(rtlsdr_tunerfreq);                           % calculate number
of retunes required
freq_bin_width = (rtlsdr_fs/nfft);                              % create xaxis
freq_axis = (rtlsdr_tunerfreq(1)-rtlsdr_fs/2*overlap  :  freq_bin_width*dec_factor
:  (rtlsdr_tunerfreq(end)+rtlsdr_fs/2*overlap)-freq_bin_width)/1e6;

% create spectrum figure
h_spectrum = create_spectrum;

% run capture and plot
capture_and_plot;

% make spectrum visible
h_spectrum.fig.Visible = 'on';

% save data
filename =
['rtlsdr_rx_specsweep_',num2str(start_freq/1e6),'MHz_',num2str(stop_freq/1e6),'MHz
_',location,'.fig'];
savefig(filename);

%% FUNCTION to create spectrum window
    function h_spectrum = create_spectrum
```

```
        % colours
        h_spectrum.line_blue = [0.0000 0.4470 0.7410];      % spectrum analyzer
blue
        h_spectrum.line_orange = [1.0000 0.5490 0.0000];    % spectrum analyzer
orange
        h_spectrum.window_grey = [0.95 0.95 0.95];          % background light
grey
        h_spectrum.axes_grey = [0.1 0.1 0.1];               % dark grey for axes
titles etc
        h_spectrum.plot_white = [1 1 1];                    % white for plot
background

        % sizes
        fig_w = 1200;
        fig_h = 600;
        scnsize = get(0,'ScreenSize');                      % find monitor 1 size
        if scnsize(3) < fig_w                               % if monitor is not
fig_w wide
            fig_w = scnsize(3);                             % reduce fig_w
        end
        if scnsize(4) < fig_h                               % if monitor is not
fig_h tall
            fig_h = scnsize(h);                             % reduce fig_h
        end
        fig_pos = [(scnsize(3)-fig_w)/2 (scnsize(4)-fig_h)/2 fig_w fig_h];   % set
to open in middle of monitor 1

        % create new figure
        h_spectrum.fig = figure(...
            'Color',h_spectrum.window_grey,...
            'Position',fig_pos,...
            'SizeChangedFcn',@resize_spectrum,...
            'Name',['RTL-SDR Spectrum Sweep: ',location],...
            'Visible', 'off');
        h_spectrum.fig.Renderer = 'painters';

        % subplot 1
        h_spectrum.axes1 = axes(...
            'Parent',h_spectrum.fig,...
            'YGrid','on','YColor',h_spectrum.axes_grey,...
            'XGrid','on','XColor',h_spectrum.axes_grey,...
            'GridLineStyle','--',...
            'Color',h_spectrum.plot_white);
        box(h_spectrum.axes1,'on');
        hold(h_spectrum.axes1,'on');
        xlabel(h_spectrum.axes1,'Frequency (MHz)');
        ylabel(h_spectrum.axes1,'Power Ratio (dBm)  [relative to 50 \Omega load]
');
        xlim(h_spectrum.axes1,[start_freq/1e6,stop_freq/1e6]);

        % subplot 2
        h_spectrum.axes2 = axes(...
            'Parent',h_spectrum.fig,...
            'YGrid','on','YColor',h_spectrum.axes_grey,...
            'XGrid','on','XColor',h_spectrum.axes_grey,...
            'GridLineStyle','--',...
            'Color',h_spectrum.plot_white);
        box(h_spectrum.axes2,'on');
```

```
        hold(h_spectrum.axes2,'on');
        xlabel(h_spectrum.axes2,'Frequency (MHz)');
        ylabel(h_spectrum.axes2,'Relative Power (Watts)');
        xlim(h_spectrum.axes2,[start_freq/1e6,stop_freq/1e6]);

        % figure title
        title(h_spectrum.axes1,['RTL-SDR Spectrum Sweep   ||   Range =
',num2str(start_freq/1e6),'MHz to ',...
            num2str(stop_freq/1e6),'MHz   ||   Bin Width =
',num2str(freq_bin_width*dec_factor/1e3),...
            'kHz   ||   Number of Bins = ',num2str(length(freq_axis)),'   ||
Number of Retunes = ',...
            num2str(nretunes)]);

        % position axes
        axes_position(fig_w,fig_h);

        % link plots together for zooming
        linkaxes([h_spectrum.axes1,h_spectrum.axes2],'x');

    end

%% FUNCTION to calculate axes positions
    function axes_position(fig_w,fig_h)

        h_spectrum.axes1.Position = [...        % dBm axes
            70/fig_w,...                        % 70px from left
            (fig_h/2)/fig_h,...                 % at centre line
            (fig_w-100)/fig_w,...               % 100px from right
            (fig_h/2-30)/fig_h];                % 80px from top

        h_spectrum.axes2.Position = [...        % Watts axes
            70/fig_w,...                        % 70px from left
            50/fig_h,...                        % 50px from bottom
            (fig_w-100)/fig_w,...               % 100px from right
            (fig_h/2-100)/fig_h];               % 100px below centre line

    end


%% FUNCTION (callback) to resize axes in spectrum window
    function resize_spectrum(hObject,callbackdata)

        % find current sizes
        fig_w = h_spectrum.fig.Position(3);
        fig_h = h_spectrum.fig.Position(4);

        % update axes positions
        axes_position(fig_w,fig_h);

    end


%% FUNCTION to capture data from the RTL-SDR and plot it
    function capture_and_plot

    % START TIMER
    tic;
    disp(' ');
```

```
    % SYSTEM OBJECTS
    obj_rtlsdr = comm.SDRRTLReceiver(...
        rtlsdr_id,...
        'CenterFrequency',      rtlsdr_tunerfreq(1),...
        'EnableTunerAGC',        false,...
        'TunerGain',                rtlsdr_gain,...
        'SampleRate',           rtlsdr_fs, ...
        'SamplesPerFrame',              rtlsdr_frmlen,...
        'OutputDataType',        rtlsdr_datatype ,...
        'FrequencyCorrection',   rtlsdr_ppm );

    obj_decmtr = dsp.FIRDecimator(...
        'DecimationFactor',     dec_factor,...
        'Numerator',            fir1(300,1/dec_factor));

    rtlsdr_data_fft = zeros(1,nfft);
    fft_reorder = zeros(length(nfrmhold),nfft*overlap);
    fft_dec = zeros(nretunes,nfft*overlap/dec_factor);

    % Check RTL-SDR
    if isempty(sdrinfo(obj_rtlsdr.RadioAddress))
        error('RTL-SDR failure. Tarkista yhteys "sdrinfo"-komennolla.');
    end

    tune_progress = 0;

    for ntune = 1:1:nretunes

        obj_rtlsdr.CenterFrequency = rtlsdr_tunerfreq(ntune);

        for frm = 1:1:nfrmdump
            rtlsdr_data = step(obj_rtlsdr);
        end

        disp(['            fc = ',num2str(rtlsdr_tunerfreq(ntune)/1e6),'MHz']);

        for frm = 1:1:nfrmhold
            rtlsdr_data = step(obj_rtlsdr);
            rtlsdr_data = rtlsdr_data - mean(rtlsdr_data);
            rtlsdr_data_fft = abs(fft(rtlsdr_data,nfft))';

            fft_reorder(frm,1:(overlap*nfft/2)) =
rtlsdr_data_fft((overlap*nfft/2)+(nfft/2)+1:end);
            fft_reorder(frm,(overlap*nfft/2)+1:end) =
rtlsdr_data_fft(1:(overlap*nfft/2));
        end

        if strcmp(fft_hold,'avg')
            fft_reorder_proc = mean(fft_reorder);
        elseif strcmp(fft_hold,'max')
            fft_reorder_proc = max(fft_reorder);
        end

        fft_dec(ntune,:) = step(obj_decmtr,fft_reorder_proc')';

        if floor(ntune*10/nretunes) ~= tune_progress
            tune_progress = floor(ntune*10/nretunes);
            disp(['      progress = ',num2str(tune_progress*10),'%']);
```

```
        end

    end

    fft_masterreshape = reshape(fft_dec',1,ntune*nfft*overlap/dec_factor);
    y_data = fft_masterreshape;
    y_data_dbm = 10*log10((fft_masterreshape.^2)/50);


plot(h_spectrum.axes1,freq_axis,y_data_dbm,'Color',h_spectrum.line_blue,'linewidth
',1.25);

plot(h_spectrum.axes2,freq_axis,y_data,'Color',h_spectrum.line_orange,'linewidth',
1.25);

    % Save as CSV
    timestamp = datestr(now, 'yyyymmdd_HHMM');
    data_filename_csv = ['spectrum_data_', timestamp, '_',
num2str(start_freq/1e6), 'MHz_', num2str(stop_freq/1e6), 'MHz_', location,
'.csv'];
    data_filename_dbm_csv = ['spectrum_data_', timestamp, '_',
num2str(start_freq/1e6), 'MHz_', num2str(stop_freq/1e6), 'MHz_', location,
'_dbm.csv'];

    data_table = table(freq_axis(:), y_data(:), 'VariableNames', {'Frequency_MHz',
'Power_W'});
    data_table_dbm = table(freq_axis(:), y_data_dbm(:), 'VariableNames',
{'Frequency_MHz', 'Power_dBm'});

    writetable(data_table, data_filename_csv);
    writetable(data_table_dbm, data_filename_dbm_csv);

    % STOP TIMER
    disp(' ');
    disp(['      run time = ',num2str(toc),'s']);
    disp(' ');

end
end
```

Appendix 2 Modified MATLAB code for analysis

```
function rtlsdr_rx_startup_matlab_record_csv_timestamp_amplitude

%% PARAMETERS
rtlsdr_id           = '0';          % RTL-SDR ID
rtlsdr_tunerfreq    = 1100e6;        % RTL-SDR tuner frequency in Hz
rtlsdr_gain         = 25;           % RTL-SDR tuner gain in dB
rtlsdr_fs           = 3e6;         % RTL-SDR sampling rate in Hz
rtlsdr_frmlen       = 4096;          % RTL-SDR output data frame size
rtlsdr_datatype     = 'single';      % RTL-SDR output data type
rtlsdr_ppm          = 0;             % RTL-SDR tuner parts per million correction
sim_time            = 6;            % simulation time in seconds

save_filename_csv   = '901_6s_idle_run_2.csv';  % CSV-file name

%% SYSTEM OBJECTS
obj_rtlsdr = comm.SDRRTLReceiver(...
    rtlsdr_id,...
    'CenterFrequency', rtlsdr_tunerfreq,...
    'EnableTunerAGC', false,...
    'TunerGain', rtlsdr_gain,...
    'SampleRate', rtlsdr_fs,...
    'SamplesPerFrame', rtlsdr_frmlen,...
    'OutputDataType', rtlsdr_datatype,...
    'FrequencyCorrection', rtlsdr_ppm);

% Spectrum analyzer objects
obj_specfft = dsp.SpectrumAnalyzer(...
    'Name', 'Spectrum Analyzer FFT',...
    'Title', 'Spectrum Analyzer FFT',...
    'SpectrumType', 'Power density',...
    'FrequencySpan', 'Full',...
    'SampleRate', rtlsdr_fs);

obj_specwaterfall = dsp.SpectrumAnalyzer(...
    'Name', 'Spectrum Analyzer Waterfall',...
    'Title', 'Spectrum Analyzer Waterfall',...
    'SpectrumType', 'Spectrogram',...
    'FrequencySpan', 'Full',...
    'SampleRate', rtlsdr_fs);

%% CALCULATIONS
rtlsdr_frmtime = rtlsdr_frmlen / rtlsdr_fs;
total_frames = ceil(sim_time / rtlsdr_frmtime);

% For saved data
all_data = complex(zeros(rtlsdr_frmlen, total_frames, rtlsdr_datatype));

%% SIMULATION

% Check if RTL-SDR is active
if isempty(sdrinfo(obj_rtlsdr.RadioAddress))
    error(['RTL-SDR failure. Please check connection to ',...
        'MATLAB using the "sdrinfo" command.']);
end

% reset run_time to 0 (secs)
run_time = 0;
```

```
frame_idx = 1;

% Run while run_time is less than sim_time
while run_time < sim_time

    % Fetch a frame from the RTL-SDR
    rtlsdr_data = step(obj_rtlsdr);

    % Update spectrum analyzer windows with new data
    step(obj_specfft, rtlsdr_data);
    step(obj_specwaterfall, rtlsdr_data);

    % Save the frame
    all_data(:, frame_idx) = rtlsdr_data;

    % Update run_time and frame index
    run_time = run_time + rtlsdr_frmtime;
    frame_idx = frame_idx + 1;

end

%% SAVE DATA

% Flatten the data: as vector
data_vector = all_data(:);

% Distinguish real ja imag
I_data = real(data_vector);
Q_data = imag(data_vector);

% Calculate amplitude (sqrt(I^2 + Q^2))
Amplitude = sqrt(I_data.^2 + Q_data.^2);

% Timestamp for data samples
n_samples = numel(data_vector);
timestamps = (0:(n_samples-1))' / rtlsdr_fs;  % Sekunteina

% Convert as CSV table: [timestamp, I, Q, amplitude]
csv_data = [timestamps, I_data, Q_data, Amplitude];

% Write CSV file
writematrix(csv_data, save_filename_csv);

disp(['Data saved to ', save_filename_csv]);

end
```