Mid-Project Report

Software and Hardware Improvements for The Low-Power Microwave Breast Cancer Radar-Based Screening Prototype

Group Members:
Haoran Du, haoran.du@mail.mcgill.ca, 260776911, ECSE 458N1
Jacob Silcoff, jacob.silcoff@mail.mcgill.ca, 260767897, ECSE 458N1

Group Number: 2

Project Supervisor:
Prof. Milica Popović, milica.Popovićh@mcgill.ca, Tel: 514 398 3417

Contents

Software and Hardware Improvements for The Low-Power Microwave Breast Cancer Radar-Based Screening Prototype*

*Mid-project report of Group 2 for ECSE 458N1 (W2020) at McGill University.

Haoran Du
Department of Electrical and Computer Engineering
McGill University
Montreal, Canada
haoran.du@mail.mcgill.ca

Jacob Silcoff
Department of Electrical and Computer Engineering
McGill University
Montreal, Canada
jacob.silcoff@mail.mcgill.ca

***Abstract*—In our design project, we worked with Dr. Milica Popović's lab to automate the process of testing antennae hardware that is part of an ongoing research project focused on developing low-power microwave breast cancer detection. Specifically, we were designing an interface to set the registers within the Linear Technology LTC6946 Board, a frequency synthesizer that is being used to drive the microwave antennae that are being used to image the breast tissue. When we began our work, the research team was using PLL Wizard, a GUI that is the standard interface for the LTC6946, and automating it with the pywinauto and pyautogui libraries. These libraries enable the programmatic control of mouse clicks and keyboard inputs, a solution that was computationally inefficient and time consuming. Our goal was to expedite this process by circumventing the GUI, enabling the registers to be set entirely through python, and thus making it faster and easier to test various waveforms and frequencies on the antennae. This semester, we researched the hardware being used and developed a lower level python solution that should be substantially more efficient. However, due to the time conflicts with the research team being at a conference abroad in early March, followed immediately by the campus shutdown caused by the COVID-19 pandemic, we have not been able to sufficiently test our solution physically in the lab this semester. Additionally, we began independently researching other aspects of the research, including the imaging techniques and data processing algorithms that will enable us to contribute to processing and visualizing data collected by the research team once we finish our automation task.**

***Index Terms*—low-power microwave breast cancer detection, breast cancer radar-based screening prototype**

I. List of Abbreviations and Notation

In this report, we will be referring to a number of terms and technologies that we worked with over the course of this project, which are listed below.

- **LTC6946**. The LTC6946 is a frequency synthesizer made by Linear Technologies. It is used to drive the antennae that scan breast tissue in hopes of detecting tumors [1].
- **DC590**. The DC590 is a usb-controlled SPI/I2C converter that is used to connect the LTC6946 to a computer via USB. It is also created by Linear Technologies [2].
- **Pylibftdi**. Pylibftdi is a python library designed to interface with devices using FTDI chips that wraps FTDI's own interface in a comparatively simpler one, to enable streamlined programming of FTDI hardware [3].
- **Phantoms**. The phantoms are synthetic models of breast tissue that are designed to simulate the electromagnetic properties of various types of human tissue, including healthy breast tissue, glands, fat, and tumors. They are used to test breast imaging prototypes in a cheap, repeatable way that does not require human test subjects [4].

II. Introduction

Breast cancer is a disease that kills more than 40,000 people every year [5]. One of the key problems in combating breast cancer is early detection, which is hindered by the fact that current testing technologies like x-ray mammography have negative side effects on patients, meaning their use must be limited, and frequent testing is infeasible [6].

A proposed imaging technique would use low powered microwaves to image the breast, exposing the tissue to radiation on the same order of magnitude as receiving a cell phone call and enabling frequent testing that could detect tumors earlier than conventional imaging technologies, meaning more timely and effective treatments for breast cancer patients [6]. This research is based on the idea that the refractive indices of different types of tissue should lead to different refraction patterns when the tissue is irradiated. By sending a microwave pulse into the breast and observing how that pulse propagates, the refractive properties of the tissue can be reasoned about, and by proxy, their likelihood of abnormal growths. It is still unknown whether this technology will be able to overcome barriers such as low SNR, inconsistency in measurement setup, and other factors, but there are currently several research teams attempting to find out.

In particular, a team led by Dr. Milica Popović and Prof. Mark Coates at McGill university is experimenting with a fixed antenna array placed around a bra shaped device that is currently undergoing testing with both synthetic phantoms and human volunteer subjects [7]. In order to properly test prototypes, Dr. Popović's research team needs access to data from a large number of test scans, conducted at McGill University Health Centre, across a wide range of frequencies, and currently getting this data represents a bottleneck in their research process.

Our role was to hasten the data collection process by creating software that automatically reprograms the prototype between tests, speeding up the current set up that relies on a slow GUI interface. Although our specific work represents a small part of a much larger project, it should still speed up the research process as a whole, and once finished the software we design could be used by anyone needing to quickly prototype and test devices using a LTC6946 frequency synthesizer, which has applications ranging from communications technology to digital circuitry.

III. Background

First and foremost, a holistic understanding of the current prototype is necessary for understanding the work being done by the lab. The design being used features a number of fixed microwave antennas attached around the cup of a prosthetic bra, which is chosen for the fact that it fits around a breast while maintaining a rigid shape for reliable scanning [7]. One at a time, a single antenna will emit a microwave frequency electromagnetic pulse, while every other other antenna will act as a receiver, recording the power received from the emitter as a function of time [7]. From the data received at each antenna, processing can be done to reason about the path that the original pulse took through the tissue, with the entire array being able to make predictions about the overall electromagnetic properties of the tissue being imaged.

This is a technique more similar to radar than it is to other forms of imaging such as x-rays or MRI, in that the goal is not to create a crisp, coherent image of the tissue, but rather to determine whether there is an abnormal growth or not, which can then be used to inform medical professionals to order more invasive, higher resolution forms of imaging. Previous research has been used to determine the best known optimal arrangement of the antennae around the breast, and although different research teams use different approaches to the arrangement of antennae, Dr. Popović's team is currently using a fixed antenna arrangement [6, 7].

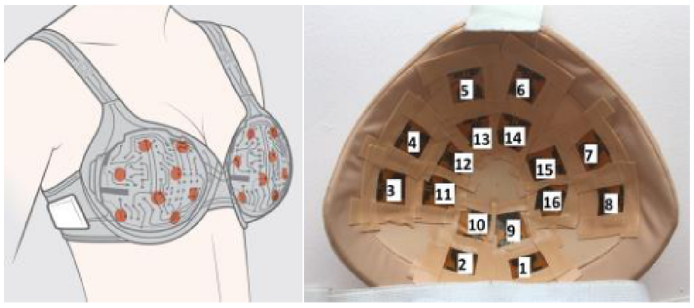

Fig. 1. The bra prototype with antenna array inside for microwave screening.

To test the prototype, models of breasts called "phantoms" are used, which have identical electromagnetic properties to human breasts, while enabling repeatable testing in a controlled setting [7]. The phantoms are designed with various levels of complexity, from a design with exclusively healthy breast tissue analog with a plug where a simulated tumor can be inserted, to a version with analogs for large glands that would introduce noise in a real world setting [6, 7].

With these phantoms, the research team is able to isolate problems to small, repeatable changes in the model. With human patients, natural changes in tissue including hormonal cycles within a patient and variation between breast tissue across a population make controlling for confounding variables much more difficult, and it is impossible to know with certainty whether or not a test subject has a growth in the first place, let alone test the prototype. Hence, models of breasts called “phantoms” are used, which have identical electromagnetic properties to human breasts, while enabling repeatable testing in a controlled setting. The phantoms are designed with various levels of complexity, from a design with exclusively healthy breast tissue analog with a plug where a simulated tumor can be inserted, to a version with analogs for large glands that would introduce noise in a real world setting [7].

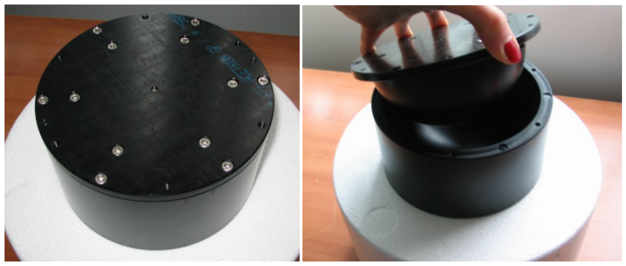
Fig. 2. The design of a breast-like phantom.

The phantoms are a way to improve test quality while reducing the need for testing on humans in the early phases of development data that compares the impact of a tumor on a patient’s scan while controlling for all other variables in the early phases of development [7]. At the current phase of testing, running a large number of experiments on the phantoms and gathering data to process is a bottleneck to moving forward, and one of the issues in testing is the relatively slow process by which the prototype is reprogrammed between tests.

For our role on this project, the most immediately relevant theory is the operation of the DC590 and LTC6946, the pieces of hardware that we are working with. Broadly, the LTC6946 is a low noise frequency synthesizer whose output is controlled by 12 byte-wide registers that can be read and written to over an SPI-compatible serial port [1]. The documentation for the LTC6946 defines waveforms needed to write single bytes, or perform multibyte data transfers over the SPI connection.

Given that most computers do not have an SPI port, the LTC6946 board is programmed with the help of the DC590, which receives USB input from a computer and outputs SPI to the LTC6946 [2]. In typical use, the LTC6946 is controlled using custom software called PLL Wizard, a GUI created by Analog Devices that enables the LTC6946 to be programmed from a PC over a USB connection through the DC590. PLL Wizard was the basis for the form of automated testing that Dr. Popović’s research team was using prior to our participation in the project, by using python GUI automation libraries to click through the menus and upload data to send to the board. However, the DC590 also provides an interface of its own, that enables generic communication with any device it is connected to using its own language of ascii codes that correspond to different I/O actions. This interface is appealing in that it is substantially more lightweight than PLL Wizard, and would enable nearly instantaneous programming of the LTC6946 compared to the relatively time consuming process of clicking through the PLL Wizard GUI.

IV. Problem Description & Requirements

A. Problem Description

The general problem we needed to solve was to quickly program the antennae to perform a desired test, but realistically this was limited to using pre existing hardware, with any additions running off of a PC, meaning we were constrained to a purely software approach.

When we began the process of designing our solution, we were given a specific interface with which we would need to comply, based on the preexisting code used by the team on GitHub. Because test automation software already existed, we planned on using the same function headers and to simply reimplement the same functionality in a more efficient way, such that our code could be seamlessly integrated with pre existing software. This implied that we would need to write our code in python, the language currently being used by the research team, and would need to implement a function `freq_set(freq)`, that took in a real valued frequency, and programmed the LTC6946 to operate at that frequency. To this end, we were provided with a register file that mapped frequencies to the register values needed to achieve those frequencies. The design would need to be substantially faster than the previous solution, while having the same end result of setting the registers within the LTC6946 to the proper values.

B. Requirements

Although the exact speed requirements were left open, meaning that any speed lower than the GUI approach would be tolerable, an optimal solution would be as fast as possible, on the order of milliseconds. This led us to the following requirements for what our solution would need to look like:

1) The design must be invoked by the `freq_set(freq)` function call used in the current codebase.

2) The design must be faster than the current implementation of `freq_set(freq)`, ideally on the order of milliseconds.
3) The design must only use software that can run on a personal computer, interfacing with the LTC6946 through a DC590 board.
4) The design must correctly set registers on the LTC6946 based on the register mapping provided in the register file.
5) The design should not require the use of a GUI.
6) The design should rely exclusively on freely available software.
7) The design should be fully automated, and should not rely on human intervention between or during function calls.

V. Design & Results

A. Design

We considered a number of possible solutions at various levels, starting with the notion that any signal sent through USB from the PLL Wizard GUI could be detected and repeated using a packet sniffer, such as WireShark, in conjunction with a library like pyusb, which sends arbitrary USB signals using python. This would enable us to trivially repeat the behavior of the PLL Wizard without concerning ourselves with the actual underlying functionality of the board, by simply listening to the USB communication process once for each desired set of frequencies to program, and storing those communications processes to be repeated for every subsequent test.

This was the first solution we suggested to the graduate students we were working with, and were told that it was too low level a solution; dealing with packet sniffers would mean a relatively large amount of work for each of the frequencies we wanted to be able to program, and the research team required the ability to test a relatively large number of frequencies, meaning this solution would be quite labor intensive for us to manage, while being less flexible to future changes in the case where we blindly copying whatever PLL Wizard would output, or requiring us to decode a communication process that was not designed for users to implement themselves. For this reason, we ultimately decided against the packet-level approach.

We also considered other solutions involving examining the PLL Wizard program itself, either by looking at its source code to find how it communicated with the board and copying that ourselves, or by trying to see if it used a more general API that we could take advantage of. However, after thorough research we were unable to track down either an accessible API or readable copy of the source code, ruling out these solutions.

B. Approach and Result

The design process then moved towards thoroughly reading the documentation of the LTC6946 and DC590 boards, given that we had never worked with them before and needed to understand their function at a fairly low level. Within the documentation for the DC590, we found instructions to interface with the DC590 directly over USB, using a protocol made up of ASCII characters that defined communication with the board.

In conjunction with the documentation from the LTC6946, which specified the SPI commands needed to write to each register, we could figure out the proper ASCII command that needed to be sent to the DC590 in order to properly program the LTC6946 at a desired frequency. However, the documentation for how to get this command from our computer to the DC590 was somewhat sparse. Two methods to interface with the host PC are given, both a "Virtual Com Port" and a set of direct drivers. The problem was that research into both of these options lead to relatively little productive results. The direct drivers were supposed to be described in the FTD2xx Programmer's guide, but when we tried to find this document, it was unclear if it even existed, and the link that was provided on the documentation itself took us to FTDI's homepage, with no indication whatsoever of where this "programmer's guide" was to be found.

Searching for this guide generated a document called the "D2xx Programmer's Guide", which was published by FTDI and had a very similar name, indicating that it may be the document referenced in the DC590's documentation, but it was entirely unclear why the names were different or why we were unable to find it from the link provided to us, making us skeptical of whether this genuinely was the document we hoped it would be. In terms of the "virtual com port" option, we found that although this provided a way to treat a USB device like SPI, it was unclear how exactly it needed to be set up for this particular hardware application, and there was no documentation we could find on using it with the DC590 from our extensive research. Given this, we ended up trying to use the D2xx Programmer's Guide, which we took to be the same thing as the FTD2xx Programmer's Guide.

Looking at the D2xx Programmer's Guide, we ran into a number of design problems. The first was that the code for the D2xx drivers was written entirely in ARM, which presented a challenge when interfacing with the python code that we needed to work with as one of our design requirements. The other issue was that although we were operating under the assumption that this library had been directly referenced by the DC590 documentation, there was no clear conclusion to be drawn about how exactly to use the library's functions to send the commands to the DC590, with noticeable gaps in specifications. We assumed the "FT_write" command that would write data to the device, but there were different versions of this function for writing to EEPROMs, files, and other devices, and the DC590 never specified which of these would apply to its interface.

There were also a number of unanswered configuration questions, as the D2xx documentation provided a considerable number of mechanisms to specify how USB communications should be configured, none of which were specified by the DC590's documentation. For instance, the D2xx driver required setting a baud rate for data transmission, and while we thoroughly looked through the documentation for both the DC590 and LTC6946, neither specified a baud rate for communication that fell within the range that the D2xx driver specified requiring. The LTC6946 did specify clock speeds to send data at, but they were higher than the baud rates we had seen used by the D2xx driver by a wide margin. It was also unclear what function was necessary to send a command.

To address the language differences, we found multiple wrappers of the D2xx library written in python that were designed to use all of the same functions from a python program. Since we intended to use the FT_write command and not much else, our emphasis was placed on simplicity and ease of use, meaning that when choosing between python libraries, we settled on the one with the best documentation and simplest structure, which ended up being pylibftdi. This enabled us to interface with the ARM drivers using python, meaning we could meet our design requirements of working within the current code base used by the research team. After familiarizing ourselves with the pylibftdi documentation and reading through some example applications, we reasoned about how to use it for our application, and eventually wrote a simple program that from our knowledge should be sufficient to write the registers to the LTC6946. This still did not answer our questions about configuration, or even about whether the library was the correct program to use in the first place, but it did give us a starting point for design. Our working assumption was that the default settings would be sufficient, which may perhaps explain why the documentation was so sparse, and that physical testing would enable us to determine if our configurations were correct.

The program we wrote used the bulk transfer protocol outlined in the LTC6946 documentation, and translated the register values for the given frequency to a command that met the interface defined by the DC590 into a string, that to the best of our knowledge would program the registers to the desired values. We then instantiated a connection to the device, and attempted to write this string to it. The code was very simple, fitting into only 14 lines, but represented weeks of research into various libraries and approaches.

```python
from pylibftdi import Device
# from register_mapping import register_values
from Transmitter_LTC6946_USB.register_mapping
    import register_values

"""
Written by: Jacob Silcoff & Haoran Du
Description:
1. Module for controlling the LTC6946 through
    the DC590B demo controller without the
    need for a GUI
2. Requires pylibftdi (if I understand
    correctly!), and its dependencies,
    including libFTDI and libusb/libconfuse
3. This code is currently untested, and very
    likely does not work as of this commit!
4. Functions::
     freq_set : sets a Tx frequency by setting
     registers from a register file
"""

def freq_set(freq):
    reg_vals = register_values(freq)
    if not reg_vals: return
    # construct command string:
    # start by sending a 0 byte to specify
    writing to register 0
    dc590_command = b'S00'
    for reg_val in reg_vals:
        if len(reg_val) is 1:
            reg_val = '0' + reg_val
        dc590_command += str.encode(''S'' +
    reg_val)
    print(dc590_command)
    with Device(mode='b') as dc590:
        # set baudrate?
        dc590.write(dc590_command)
```

Listing 1. Our approach on the freq_set(freq) function

This code was finished in late February, and our plan was to test it on the physical hardware as soon as we had access to the labs following March break. However, the graduate students were at a research conference in Barbados, and when they returned, campus had been shut down, meaning that we were never able to get physical access to the hardware, and any attempts to debug and test our code to figure out next steps were cut off. Although the solution represents the best initial design we could think of with the limited resources available, we did not expect it to work on the first attempt. Rather, this was to be a starting point for future experimentation that relied on us having access to the hardware to run tests and make improvements were needed. The problem for us was that we didn't have a stack trace to look into fixing, meaning that we didn't have any path to continue with this automation task.

Despite this, our goal is to continue helping the research team, and we are currently reading through research publications on data processing using this type of antennae setup, to get a headstart on what we will likely be working on once the automation task is finished.

VI. Plan for Next Semester

Our plans for next semester begin first and foremost with getting physical access to the lab to test our code. We intend to do this at the first opportunity where one of us is free, and the graduate students are able to give us access to the labs to test on the hardware. From there, it is unclear how exactly we will continue. There is a very small chance that the code works on the first try, which every programmer dreams about but is highly unlikely given our uncertainty in this design. If this happens, we will begin the next phase of our project.

Most likely, we predict we will encounter some sort of issues with the code that can be overcome within two weeks of access to the hardware. This process will involve reading stack traces of where our code went wrong, and reevaluating whether the selected library is the correct tool for the job. Because we have already thoroughly familiarized ourselves with the hardware, it is unlikely this will take a substantial period of time even in the worst case where we need to start again with a new approach, since the most difficult part for us was getting the initial understanding of the system we were working with. Our hope is that within three weeks, we will have the automation tool working, tested, and integrated into the research team's code base.

In the absolute worst case scenario, we are confident that a low level packet sniffing approach would work, even if it was substantially less elegant, meaning that regardless of the code we have now functioning during testing, we should be able to deliver functioning code by this deadline.

We will be able to know that our design meets specifications when it can run without errors, and when we are able to get the same experimental results in a side by side comparison as we can with the old code, which we know to be functioning correctly. We also intend to run timing analysis to ensure that we are successful in our goal of reducing latency, and we will ensure that our client, Leonardo Fortaleza who is the researcher that will be using our software, approves of it.

Once we conclude the automation software, the research team has informed us that we have other potential tasks that we will be working on to assist with the work that the team is doing. These include analyzing data that comes from the tests in order to better visualize it, or helping to create tools to better measure the data and reduce noise. At this point, we currently have not chosen a specific task given that we have yet to complete our current objective, which means that we will likely know what additional tasks will be included in our project after meeting with the research team to reevaluate what tasks we will be most useful in performing.

VII. Impact on Society & the Environment

In a big picture sense, the goal of this project is to have a large, positive societal impact. If successful, Dr. Popović's research team hopes to reduce the number of breast cancer deaths and increase the quality of care that patients with breast cancer can receive by enabling cheaper, safer, and more frequently repeatable forms of imaging that can catch tumors earlier in their development when they are more easily and effectively treated. This represents a group of some 276,480 people that are diagnosed with breast cancer each year [5], that if this technology were to make it to the clinical mainstream, could be positively impacted. That being said, the likely societal impact of our contributions to the project are likely negligible.

At this stage of research, it is currently unknown whether the technology will ever be able to overcome intrinsic barriers to its ability to succeed, and the possibility of this technology being used in a clinical setting is at best many years away, meaning there is a chance that our work will have no societal impact whatsoever, which is the unfortunate nature of experimental research. Our individual contribution is also likely not pivotal to the success of this project, and while we may certainly improve the ability of the research team to go forward with their trials through more efficient testing procedures, it is unlikely that our design project team can claim any of their benefits given our small role.

As with any experimental research, there is a possibility of societal harm; should the prototype report false positives or false negatives that inform harmful treatment options, people could be severely hurt. Studies published by the NIH report that breast cancer may be over-diagnosed to the detriment of patients, meaning that researchers into new testing schemes have an obligation to ensure that their developments do not lead to over-diagnosis and impose more harm to patients than they do protection.

In terms of environmental implications, they are likely quite negligible. All of our contributions are based on low power software that is powered by the renewable energy of Hydro Quebec, and the prototype on which we are testing has no likelihood of mass production in the near future.

That being said, in the best case where this project was a massive success that ended up making it to market, the microwave circuitry is very low power and unlikely to generate substantial energy use, and the manufacturing process is, to the best of our knowledge, unlikely to cause any meaningful levels of environmental harm.

VIII. Report on Teamwork

Our collaboration has been fairly equitable and productive this past semester. Starting at the beginning of the project, we laid out weekly meeting times for us to meet together one-on-one to go over the plans for the coming week and check in on progress. Even when campus was shut down we managed to meet up long distance through video conferencing software, which was an integral part of dividing work and ensuring deadlines would be met.

For division of labor on reports, we tended to alternate based on each other's schedules, with one person writing the bulk of the report and the other editing and finishing the work. This worked well to accommodate our schedules, given that one of us typically had a mid-term or major project before nearly every submission deadline on a more or less alternating basis, meaning that we were able to both put in equal amounts of work without sacrificing the quality of our work in other classes.

In terms of research for the project, Haoran focused on a broader scale, reading research publications from the research team to get background on the lab's work, and liaising with the research team, while Jacob focused on specifics related to the hardware, and did research work focusing on different available libraries and technologies to interface with the hardware. This research would be shared at each weekly meeting, meaning both parties would be informed on the most relevant information acquired by the other party, including solutions researched, and ideas for future research paths.

We found that one of the key advantages of working in a small two person team was that a rigid division of tasks was substantially less necessary, and it was relatively easy to keep tabs on the other person's work ensuring that each member contributed equally to the project.

IX. Conclusion

In this semester, we worked for Dr. Milica Popović's research group and we had a deeper understanding of the fixed-point antenna array approach on low-power microwave breast cancer detection. We have worked on the software part of automating the process of testing the antennae hardware, which includes replacing a GUI with an CLI to set the registers within the Linear Technology LTC6946 Board in python with the pywinauto and pyautogui libraries. This enables the memory access speed of the registers to be set entirely through python to improve the data transfer performance. Due to the current COVID-19 pandemic, we have not been able to sufficiently test our solution physically in the lab this semester.

For the next semester, our first priority would be to test our code physically in the lab. Furthermore, in the summer, to make up with the time lost caused by the campus shutdown in this semester, we would like to conduct research on the imaging techniques and data processing algorithms which are vital to process and visualize the team's data in the future.

Acknowledgment

We would like to acknowledge Dr. Popović and her graduate students that worked with us this semester, including Lena Kranold, Leonardo Fortaleza, and Wesley Kendall, each of whom provided a great deal of insight into the research they are working on, and guided us through the process of developing our project. We would also like to express our appreciation to Prof. Mark Coates and his research group for showing us how the research lab works during our first meeting.

References

[1] "LTC6946," LTC6946 Datasheet and Product Info — Analog Devices. [Online]. Available: https://www.analog.com/en/products/ltc6946.html#product-overview. [Accessed: 08-Apr-2020].

[2] "DC590B," DC590B Evaluation Board — Analog Devices. [Online]. Available: https://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/dc590b.html#eb-overview. [Accessed: 08-Apr-2020].

[3] "pylibftdi," PyPI. [Online]. Available: https://pypi.org/project/pylibftdi/. [Accessed: 08-Apr-2020].

[4] E. Porter, A. Santorelli, M. Coates, and M. Popović, “Time-Domain Microwave Breast Cancer Detection: Extensive System Testing with Phantoms,” Technology in Cancer Research & Treatment, vol. 12, no. 2, pp. 131–143, Apr. 2013.

[5] “Breast Cancer - Statistics,” Cancer.Net, 18-Feb-2020. [Online]. Available: https://www.cancer.net/cancer-types/breast-cancer/statistics. [Accessed: 08-Apr-2020].

[6] L. Kranold, C. Quintyne, M. Coates, and M. Popovic, “Microwave Radar for Breast Screening: Initial Clinical Data with Suspicious-Lesion Patients,” 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Jul. 2019.

[7] A. Santorelli, Y. Li, E. Porter, M. Popović and M. Coates, ”Investigation of classification algorithms for a prototype microwave breast cancer monitor,” The 8th European Conference on Antennas and Propagation (EuCAP 2014), The Hague, 2014, pp. 320-324.