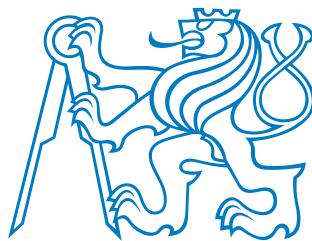


**Czech Technical University in Prague**

Department of Electromagnetic Field



**BACHELOR THESIS**

**Surveillance FMCW Radar**

Supervisor of the bachelor thesis: Ing. Viktor Adler, Ph.D

Study programme: Elektronika a komunikace

Prague TODO 2024

---

I hereby declare that I have independently written the submitted work and that I have cited all sources of information used in accordance with the Methodological Guidelines on Adherence to Ethical Principles in the Preparation of University Theses and the Framework Rules for the Use of Artificial Intelligence at CTU for Study and Educational Purposes in Bachelor's and Master's Studies.  
In Prague, January 29, 2025

---

**Title:** Surveillance FMCW Radar

**Author:** Havránek Kryštof

**Department:** Department of Electromagnetic Field

**Supervisor:** Ing. Viktor Adler, Ph.D, Department of Electromagnetic Field

**Abstract:** TODO

**Keywords:** TODO

**Název práce:** Přehledový FMCW radar

**Autor:** Havránek Kryštof

**Katedra:** Katedra elektromagnetického pole

**Vedoucí práce:** Ing. Viktor Adler, Ph.D, Katedra elektromagnetického pole

**Abstrakt:** TODO

**Klíčová slova:** TODO

---

# Contents

<b>Abbreviations</b>	<b>1</b>
<b>Introduction</b>	<b>2</b>
<b>1 FMCW Radar Fundamentals</b>	<b>3</b>
1.1 Comparison FMCW Radar to Pulse Radar . . . . .	3
1.2 Basic principles of ideal FMCW radar . . . . .	4
1.2.1 Limits of Range Measurement . . . . .	5
1.2.2 Speed Measurement . . . . .	6
<b>2 SiRad Easy<sup>®</sup></b>	<b>7</b>
2.1 Outline of Chosen Configuration . . . . .	8
2.2 24 GHz Header . . . . .	9
2.3 122 GHz Header . . . . .	11
<b>3 Rotary Platform</b>	<b>13</b>
3.1 Platform Design Parameters . . . . .	13
3.1.1 Physical Capabilities . . . . .	13
3.1.2 Software Requirements . . . . .	13
3.2 Platform Construction . . . . .	14
3.2.1 Platform Electronics . . . . .	15
3.3 Platform Software Realization . . . . .	15
3.3.1 Communication layer . . . . .	15
3.3.2 Application layer . . . . .	16
3.3.3 HAL Layer . . . . .	17
3.4 MATLAB Control Interface . . . . .	18
<b>4 Radar Data Processing</b>	<b>20</b>
<b>Conclusion</b>	<b>21</b>

---

# Abbreviations

Abbreviation	Meaning
AGC	Automatic Gain Control
CW	Continuous Wave
CFAR	Constant False Alarm Rate
FMCW	Frequency Modulated Continuous Wave
devkit	Development Kit
SNR	Signal to Noise Ratio
SISO	Single Input Single Output
TSV	Tab Separated Values

---

# Introduction

TODO

# 1. FMCW Radar Fundamentals

**TODO: Consider some semi-introductory chapter before this one. Also the name "Basics of FMCW Radar" is not very good.**

As opposed to classical continuous wave (CW) radars Frequency Modulated Continuous Wave (FMCW) radars broadcast a signal not on a single frequency but with linear sweep across a range of frequencies. Such approach allows computing a range estimation without requiring a pulsing signal. This is while the capability to measure speed of the target using doppler shift is still maintained. However the calculation of speed more complex than in case of broadcast on a single frequency.

The MW suffix in FMCW radar is used to denote that the radar operates in microwave range of frequencies. Such frequencies enable the antenna array to be small, enabling even on chip internaration. Also MM part of radio spectrum is generally license free [1] and offers large bandwiths – limiting possible interference.

## 1.1 Comparison FMCW Radar to Pulse Radar

Distance measurement with radar predates an invent of FMCW radar by a few decades. Traditional approaches relied predominantly on sending pulsing the signal electromagnetic radiation and measuring time it takes for the signal to return. On such radars speed can be calculated traditionally using Doppler effect as

$$v = \frac{f_{\text{dop}} c_o}{2 f_{\text{rad}}}, \quad (1.1)$$

where  $f_{\text{dop}}$  is doppler frequency,  $c_o$  is speed of light and  $f_{\text{rad}}$  is frequency of the radar signal. And distance was tied to time of flight  $t$  of the signal as

$$d = \frac{c_o \cdot t}{2}. \quad (1.2)$$

Such approach is on paper simpler and more intuitive however it has several drawbacks that seriously limit its precision and usability especially in close range applications. In order to achieve a good resolution in distance measurement the pulse must be very short. However in order to to impede on radars capabilities (need to maintain strong SNR over long distances) the power of the pulses must stay the same even when the duty cycle is dramatically increased [2].

Transmitting with high average power is problematic legally and technically. It can result in interfering with other devices not to mention circuitry to drive high power radars is bulky and expensive often requiring use of high voltages and vacuum tubes. Thus pulsed radars are predominantly used in application where high resolution in range is not needed such as long range detection.

One major advantage of pulsed radar is the ease of processing data. On FMCW radar the there is a entanglement of distance and speed readings as there is both frequency change due to Doppler effect and the frequency sweep itself.

## 1.2 Basic principles of ideal FMCW radar

Let us picture an ideal FMCW radar system sending a periodic chirp with frequency frequency sweep from  $f_c$  to  $f_c + BW$  so called sawtooth waveform. This waveform is also used on the devkit used in this thesis [3].

Other FMCW systems may use a different modulations such a linear triangular modulation or segmented linear frequency Modulation. These offer some advantages but the nature of the beat signal (which forms a sine wave with sawtooth modulation) is more complex in them especially when multiple targets are present [2].

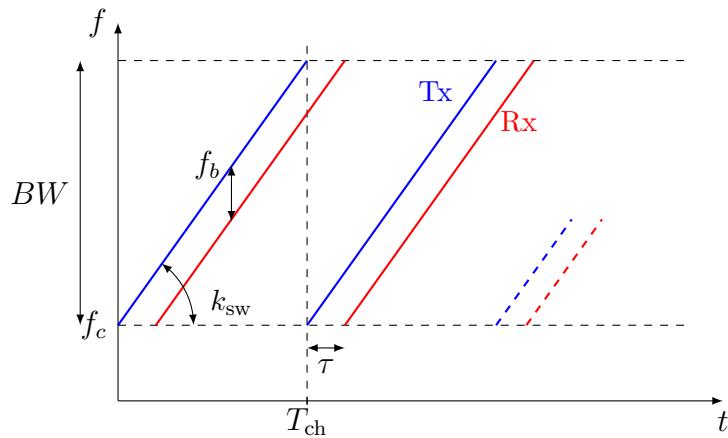


Figure 1.1: Ideal relation of frequency and time for received and sent signal

We can clearly see that in given time  $t$  the frequency spread from sent signal to received signal is proportional to the time delay  $\tau$ . However doing some simple subtraction in spectrogram of the both signals isn't really feasible – the calculation need to take a smarter approach.

Let us define the chirp slope  $k_{sw}$  we can describe the change in frequency of the received signal as

$$\Delta f_s(t) = k_{sw}t = \frac{BW}{T_{ch}}t, \quad (1.3)$$

where  $t$  is the time goes from 0 to chirp length  $T_{ch}$ . Standard equation of FM signal can be written as

$$s_t(t) = A \cos \left( \omega_c t + 2\pi \int_0^t f_s(t) dt \right), \quad (1.4)$$

where  $A$  is amplitude of the signal,  $\omega_c$  is carrier frequency and  $f(s)$  is frequency of the signal. Substituting (1.3) into (1.4) we get the signal borrowed from the radar

$$s_t(t) = A \cos(\omega_c t + \pi k_{sw} t^2). \quad (1.5)$$

Signal bounced back from the target will have the same equation with the only difference being the time delay  $\tau$ ,

$$s_r(t) = A \cos(\omega_c(t - \tau) + \pi k_{sw}(t - \tau)^2). \quad (1.6)$$

Now we can calculate the product of the two signals, this can be done easily in the real world using a frequency mixer. This is done with the standard formula

## 1. FMCW Radar Fundamentals

---

$\cos(\alpha) \cos \beta = \frac{1}{2}(\cos(\alpha - \beta) + \cos(\alpha + \beta))$  The result of the multiplication is

$$s(t) = s_r(t) \cdot s_t(t) = \frac{A^2}{2} \cos(2(\omega_c - 2\pi k_{sw} T_\tau)t + 2\pi k_{sw} t^2 + (\pi k_{sw} \tau^2 - \omega_c \tau)) + \frac{A^2}{2} \cos(2\pi k_{sw} \tau t + (\omega_c \tau - \pi k_{sw} \tau^2)). \quad (1.7)$$

First additive term will lead to a signal with very high frequency, well above  $2\omega_c$ , this term doesn't carry any useful information and is usually filtered out – either by low pass filter or the frequency mixer itself [4]. Second term is so called beat signal whose frequency is directly proportional to the time delay  $\tau$ . Applying a first time derivative to the cosine argument we get the frequency of the beat signal.

$$f_b = \frac{1}{2\pi} \frac{\partial}{\partial t} (2\pi k_{sw} \tau t + (\omega_c \tau - \pi k_{sw} \tau^2)) = k_{sw} \tau. \quad (1.8)$$

Calculating the distance to the target is now trivial, delay  $\tau$  is equal to the time it takes for the signal to travel to the target and back

$$R = \frac{c_0 \tau}{2}, \quad (1.9)$$

By substituting (1.8) into (1.9) we get an equation for distance

$$R = \frac{c_0 f_b}{2k_{sw}} = \frac{c_o f_b T_{ch}}{2BW}. \quad (1.10)$$

### 1.2.1 Limits of Range Measurement

Absolute limit for maximal distance is given by the time it takes for the signal to travel from the radar to the target and back. Would the distance be greater than a time of single chirp the signal would be interpreted as coming from a closer target. That gives us a maximal limit on beat frequency  $f_b = BW$ .

However in most cases the limit will be imposed not by  $T_{ch}$  respectively  $BW$  but by our sampling frequency  $f_s$ . In order to avoid aliasing the Nyquist-Shannon theorem must be satisfied thus limiting the maximal beat frequency to  $f_s/2$  and giving us maximal distance of

$$R = \frac{c_o f_s}{4k_{sw}}. \quad (1.11)$$

While sampling with frequency  $f_s$  we get  $N = f_s T_{ch}$  samples. Applying a Discrete Fourier Transform (DFT) to the signal we get  $N$  samples in spectrum with frequency resolution of

$$\Delta f_b = \frac{f_s}{N} = \frac{1}{T_{ch}}, \quad (1.12)$$

we can see that the resolution of spectrum is only inversely proportional to the chirp length and doesn't have any relation to sampling frequency [2]. Now we can enter  $\Delta f_b$  into (1.10) to get the minimal distance that can be measured as

$$\Delta R = \frac{c_o}{2BW} \quad (1.13)$$

Thus in order to increase resolution in range a wider bandwidth is needed.

## 1. FMCW Radar Fundamentals

---

There are of course other effect impeding the resolution of the radar system – such a phase noise around targets or sweep nonlinearity. Sweep Linearity can be both in the ramp itself – leading to decreasing resolution with range (with both linear and quadratic errors present) [4] (Appendix D and E). Or in sweep recovery (time to return to the start of the sweep) which leads to a fix decrease in resolution [5]. Both are however largely compensated in modern radar systems by using a closed feedback loop [4].

### 1.2.2 Speed Measurement

In order to demonstrate the effect of moving target on the beat frequency we can redefine the time delay  $\tau$  as

$$\tau = \frac{2(R_0 + vt)}{c_0} \quad (1.14)$$

where  $R_0$  is the initial distance to the target and  $v$  is the radial speed of the target. Within a single chirp there is no way to distinguish between the effects distance and speed of the target – thus multiple chirps are needed. Rewriting as

$$\tau = \frac{2(R_0 v(nT_{\text{ch}} + t_s))}{c_0} \quad (1.15)$$

where  $n$  is the number of chirps,  $T_{\text{ch}}$  is the chirp length and  $t_s$  denotes time within a single chirp ( $0 \leq t_s \leq T_{\text{ch}}$ ). Substituting (1.15) into low frequency part (1.7) leads to very complex equation, however according to [6] most of the terms can be neglected leading us to

$$s(t_s, n) = \frac{A^2}{2} \cos \left( \frac{4\pi k_{\text{sw}} R_0}{c_0} t_s + \frac{2\omega_c v n}{c_0} T_{\text{ch}} + \varphi_0 \right), \quad (1.16)$$

where  $\varphi_0$  is a phase shift given by the initial distance to the target. Its clear that first element describes predominantly the distance to the target and the second one the speed of the target. We can also see that speed will not affect a beat frequency in a single spectrum but will lead to a phase shift across multiple spectrums.

In order to calculate doppler shift frequency

$$f_d = \frac{2f_c v}{c_0}, \quad (1.17)$$

we can use 2D Fourier transform – first in time within a single chirp and then in time across multiple chirps. This will lead to a so called range-Doppler map which on one axis contains information about speed and on second distance of the target [6].

Speed resolution is derived from a number of chirps  $N$  we are analyzing and their length  $T_{\text{ch}}$  as

$$\Delta v = \frac{c_0}{2f_c} \frac{1}{NT_{\text{ch}}}. \quad (1.18)$$

On the other hand the increase in number of chirps will affect the distance measurement if the target movement speed is sufficiently high.

## 2. SiRad Easy<sup>®</sup>

---

# 2. SiRad Easy<sup>®</sup>

Indie Semiconductor's SiRad Easy<sup>®</sup> is an FMCW radar system development kit designed primarily for automotive applications. Out of the box, it offers two headers a 24GHz and a 122GHz module both based on ICs from Indie Semiconductor (TRX-024-007 and TRA-120-001, respectively). Both are strictly SISO IC with two antennas – one for receiving another for transmitting so azimuth estimation is not possible by default.

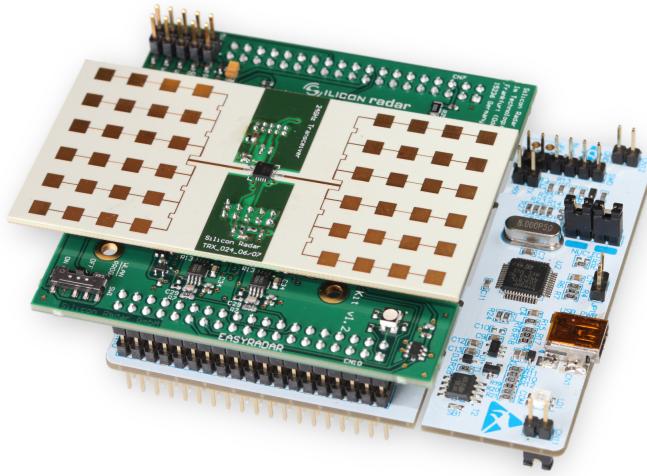


Figure 2.1: SiRad Easy<sup>®</sup> 24 GHz configuration

Direct communication with the radar board itself is not possible, or at least, the communication interface is undocumented. Instead there is always an intermediary in the form of STM32 Nucleo series microcontroller. To this microcontroller does the user connect either directly with UART over USB or with WiFi over on board ESP32. Both are relatively low bandwidth communication – the serial maxing at baudrate of 1 000 0000.

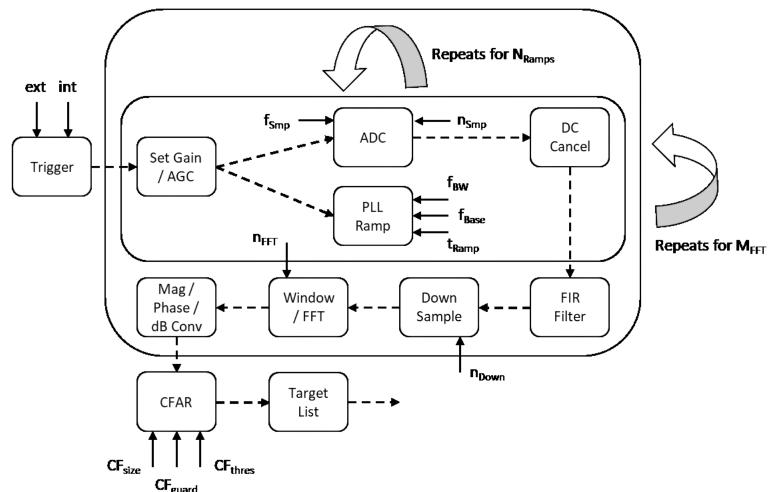


Figure 2.2: Flow of Radar Measurement on SiRad Easy<sup>®</sup>

The devkit is designed for easy integration into existing projects; however, this

## 2. SiRad Easy<sup>®</sup>

---

unfortunately poses limitations for this thesis. In normal operation, the radar system functions as a black box (processing schema shown in 2.2), implementing its own filtering, FFT, and CFAR algorithms while only reporting a target list to the user.

This entire sequence can be triggered either internally (driven by a configurable oscillator) or externally (via a GPIO pin or UART input). Once trigger is received the device carries out a user configured number of chirps. The chirp length is determined by the number of sampling steps and the ADC sampling time as follows:

$$t_{\text{ramp}} = \frac{t_{\text{ADC}} \cdot (N_{\text{samples}} + 85)}{36 \text{ MHz}} \mu\text{s}, \quad (2.1)$$

with the manufacturer recommending an optimal time of 1 ms for good SNR [7]. What happens to the sampled ramps remains unclear, as the manufacturer has not provided details. It is likely that the data are somehow averaged to improve the SNR of the reported signal. As only  $N_{\text{samples}}$  are reported by the radar but according to manufacturer provided equation (2.1) each ramp should be sampled with that number of samples. Additionally, due to the devkit's limited computational resources, the radar does not emit a continuous sawtooth waveform but rather a segmented one [7].

### 2.1 Outline of Chosen Configuration

For the purpose of this project, the devkit was configured to output raw data from the 12-bit ADC in the form of in-phase and quadrature components of the signal. This allows for a more detailed analysis of the radar system and enables the implementation of custom signal processing algorithms. For these reasons, the default windowing and undocumented filtering was also disabled.

After configuration of the output regimes, it is necessary to find the correct balance between the number of ramps, samples, and sampling frequency. Since the radar will be in motion, frequent updates are crucial to ensure accurate reporting. Faster updates allow for higher platform speeds or make it possible to use neighboring samples to aid in evaluation of current one (As the angular distance between them remains relatively small.).

Reducing the number of ramps slightly improves the update rate, but the effect is minimal—decreasing the number of ramps from 16 to 1 results in only an 11 % increase in reporting speed. Similarly, changes in sampling frequency yield only minor improvements. The only parameter that significantly impacts the update rate is the number of samples—halving the number of samples leads to an approximately 50 % increase in speed. This is with the sampling frequency adjusted for optimal SNR (close to 1 ms). Even then, reports are received roughly every 60ms (for 256 samples), and the maximum detection range is more than halved.

Another parameter that requires consideration in rotary applications is AGC. Using it adds two additional ramps that are used solely to set gain value but it also can lead to inconsistent weights of values between two neighboring data sets. Tough former only introduces a small slowdown but the latter could be a problem. Especially since radar doesn't really report change in AGC value. Thus its safest to turn AGC off and set the gain manually depending on how the readings appear in post processing.

## 2. SiRad Easy<sup>®</sup>

---

It is also important to note that the radar system is not well-suited for on-the-fly configuration changes. Applying a new configuration takes a considerable amount of time, and the radar does not provide any feedback to indicate when a new configuration can be safely applied. As a result, commonly used techniques such as alternating chirp slope or frequency modulation are not feasible.

Regarding the output format, the radar system supports two options: binary and TSV. Since the output speed does not differ significantly between the two, the primary deciding factor is the ease and speed of parsing. Even though MATLAB is not particularly optimized for parsing binary data, it still processes binary output approximately 40

### 2.2 24 GHz Header

Center of 24 GHz header is a SISO TRX-024-007 transceiver which integrates low noise amplifier, frequency mixer, filters and VCO into a single chip. It is primarily designed to operate in the ISM band (24.0–24.25GHz), with an additional ultra-wideband mode supporting 23–26 GHz [8]. On the SiRad Easy<sup>®</sup> there is no distinction made between those two modes and the user is free to set any bandwidth [7] . The transmitter output power ranges from 2.5 dBm to 6 dBm, depending on the configuration [8]. A maximum range of 400 m is advertised [9], though this is likely under ideal conditions when observing a large target. Also as previously stated the radar will operate on much lower sampling frequency than needed to achieve such results.

As shown in Figure 2.1, the chip is connected to two microstrip patch antennas. The patches are arranged in a relatively standard configuration, forming a  $6 \times 4$  array with spacing approximately equal to half the wavelength at 24 GHz.

Since the manufacturer did not provide any information about the radiation pattern of the array, a simulation was conducted using the CST simulation suite. Additionally, the manufacturer did not disclose the substrate parameters for the radar board specifically. Fortunately, the TRX-024-007 datasheet includes a board stack-up for the chip's evaluation board. It was assumed that the same stack-up would be used—18  $\mu\text{m}$  copper for the traces and ground plane, with a 250  $\mu\text{m}$  thick Rogers RO4350B substrate [8].

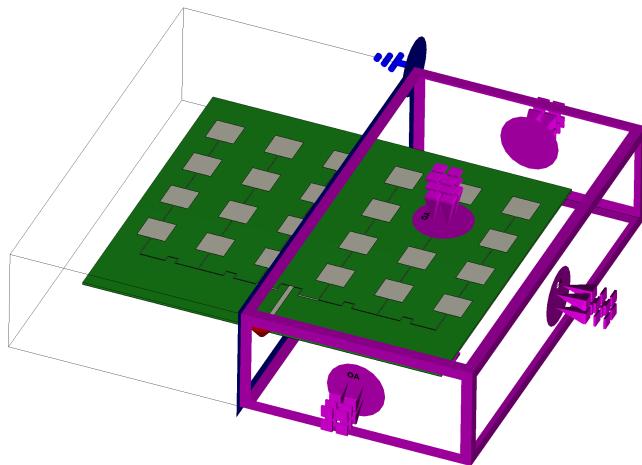


Figure 2.3: Simulated 24 GHz header with boundary conditions showed

## 2. SiRad Easy<sup>®</sup>

---

After measuring the dimensions of the array using an optical microscope, the entire array was redrawn in CST Studio. Figure 2.3 shows the applied boundary conditions: the antenna was placed in open space, and an  $H = 0$  condition was set along the symmetry plane to speed up the simulation.

After performing a standard time-domain simulation with an excitation signal ranging from 0 to 26GHz, the antenna array exhibited a minimum reflection coefficient  $s_{11} \doteq -26.4$  dB at 23.478 GHz with second minimum at 24.518 GHz (Figure 2.4). Lack of minimum at 24 GHz may be attributed to measurement difficulties (The etching quality of the copper traces was suboptimal.) and the neglecting variations of substrate parameter due to interactions of different layers.

At both frequencies with minimal reflection, a far-field radiation pattern was calculated. For clarity, only the 24.518 GHz pattern is shown in Figure 2.5. The main lobe width was measured at approximately 16 degrees (Figure 2.6) along the 180-degree norm (for orientation refer to the red cone on the PCB), with a peak gain of 18.6 dBi and side-lobe suppression of -13 dB. For the 90-degree norm, the main lobe width (Figure 2.7) was 30 degrees, with side-lobe suppression of -10 dB.

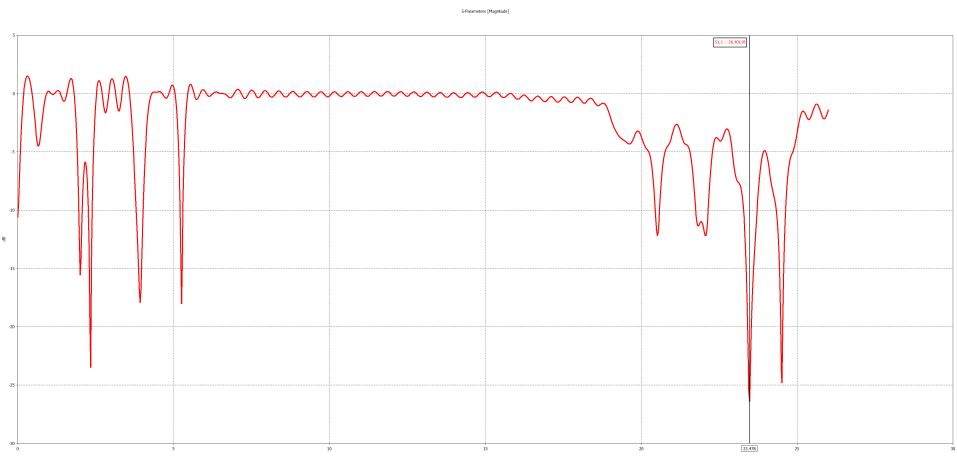


Figure 2.4:  $s_{11}$  parameter of the 24 GHz header

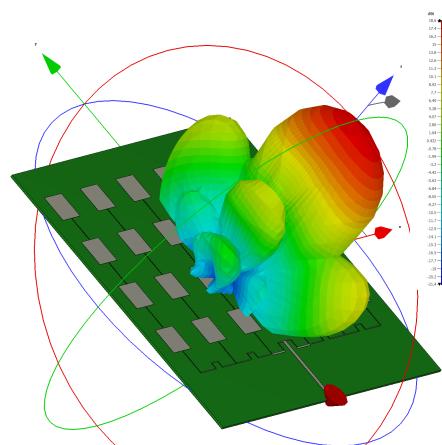


Figure 2.5: Radiation pattern of 24 GHz header – 3D view

## 2. SiRad Easy<sup>®</sup>

---

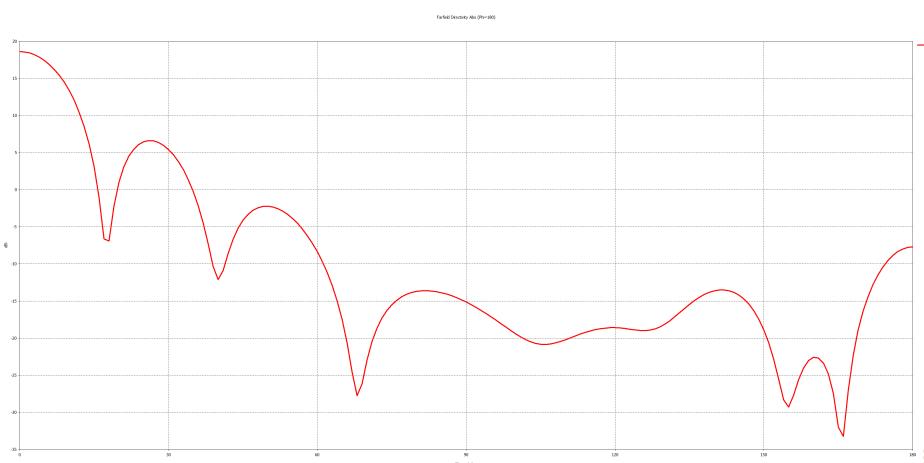


Figure 2.6: Radiation pattern of 24 GHz header – 180° norm

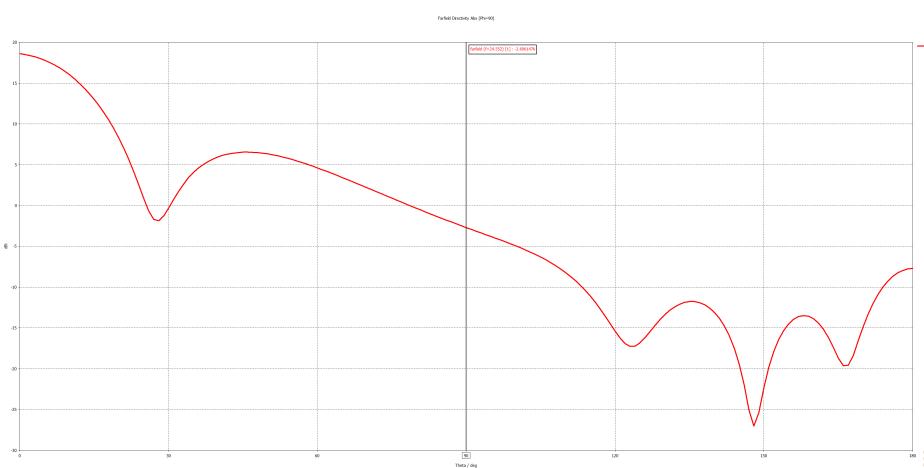


Figure 2.7: Radiation pattern of 24 GHz header – 90° norm

### 2.3 122 GHz Header

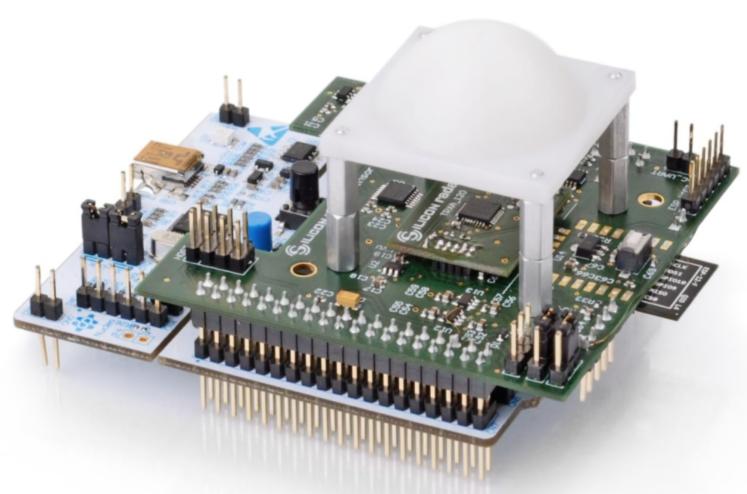


Figure 2.8: SiRad Easy<sup>®</sup> with 122 GHz header

## 2. SiRad Easy<sup>®</sup>

---

The 122 GHz header is based on the TRX-120-001 transceiver, which, in addition to the essential components required for RF transmission and reception, also incorporates two on-chip antennas. It's designed to operate in the 122-123 GHz band, with output power ranging from -7 dBm to 1 dBm [10]. The chip is capable of detecting large targets at distances of up to 40 m [9].

Out of the box performance of the system is quite bad (Figure 2.9) with width of the main lobe being roughly  $\pm 40^\circ$  in both E-plane and H-plane [10]. However, this can be significantly improved using the supplied collimator lens, reducing the main lobe width to  $\pm 4^\circ$  [11] (see Figure 2.9).

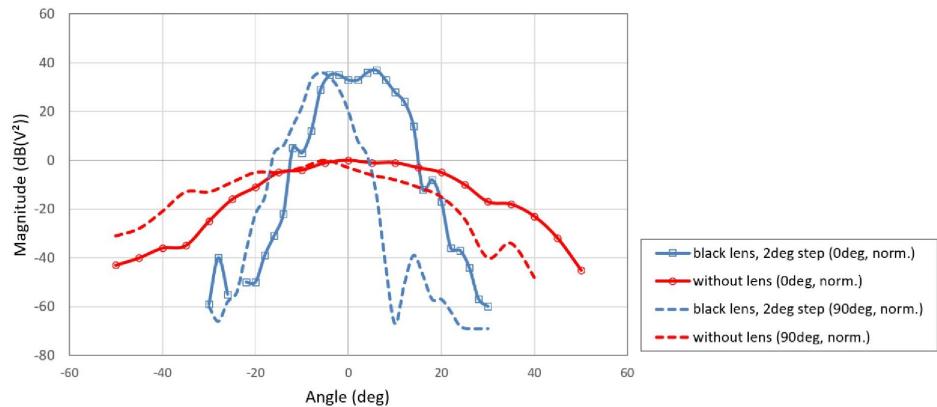


Figure 2.9: Radiation pattern of 122 GHz header comparison

# 3. Rotary Platform

Following chapter outlines design process and operation of a rotary platform specifically designed for SiRad Easy<sup>®</sup> radar system.

## 3.1 Platform Design Parameters

To begin, it is essential to outline the fundamental requirements for the platform. These stem from the physical capabilities of the SiRad Easy<sup>®</sup> and need to provide easy to use interface to control the platform.

### 3.1.1 Physical Capabilities

The primary constraints on the physical design arise from the radar's radiation pattern (Both 24 and 122 GHz headers are accounted for.). That gives us information about the clearance needed in front of the radar and more importantly how precise the platform needs to be. As stated previously the 24 GHz radar has a main lobe width of some  $\pm 7^\circ$  and the 122 GHz radar has a main lobe width of  $\pm 4^\circ$  [10]. Thus a conservative clearance of  $\pm 45^\circ$  in front of the radar was set in order to prevent any strong reflections be it from the main lobe or side lobes.

With a relatively low angular resolution of the radar a high precision of the platform is not needed. Basic 200 step stepper motor with step size of  $1.8^\circ$  is sufficient however smoother motion will be easier to compensate for in the software. Still due to low radar weight, measured at 120 g with the mounting bracket, the smoothens of the motion can be achieved relying solely on microstepping of the motor.

In terms of speed high-speed movement is unnecessary. The manufacturer specifies a maximum update frequency of 50 Hz, equating to a new update every 20 ms [3]. And in this application the update frequency will be closer to 10 to 20 Hz. Using the following equation:

$$t_{\text{angle}} = \frac{60}{360 \cdot N_{\text{RMP}}} \cdot \alpha, \quad (3.1)$$

where  $t_{\text{angle}}$  is time between spend on traveling angle of  $\alpha$  in seconds and  $N_{\text{RMP}}$  is number of rotations per minute, we can calculate that even for low RPM of 60 an angle of 8 degrees the platform travels (angular width of main lobe for 122 GHz radar) in 10 ms – too fast to properly interpret the data.

### 3.1.2 Software Requirements

Given its widespread adoption as an industry standard for controlling multi-axis machines, G-code over serial is a natural choice for the platform's communication format. Beyond the basic functionality typically offered by G-code interpreters, the platform must support additional features to reduce the user's manual control burden. These features include the ability to define movement limits and preprogram sequences of movements for autonomous execution by the platform.

For uplink communication, the platform must provide real-time information about its current position and speed. This data allows the user to make mathematical corrections to the radar's gathered data.

### 3. Rotary Platform

## 3.2 Platform Construction

As the platform needs to transmit data from the rotating part to the stationary a slip ring is needed. Due to relatively low transmission speed of the radar and lack of any special requirements, like waterproofing, an affordable model UH3899-01-0810 from manufacturer Senring was chosen classical contact slip ring was chosen. It boasts dedicated USB 2.0 connection and 8 wires for additional signal with advertised insertion loss of sub 2 dB [12].

Unfortunately the manufacturer went with non standard male-male USB 2.0 connection which requires a female-female adapter in order to connect the radar to the slip ring. Still especially when used with a poor quality cable the whole system starts to have signal integrity problems. These issues could most likely be mitigated by integrating a signal conditioner into the transmission line.

The rest of the design is rather simple. Fixed section mounts a sliring with stepper motor underneath directly driving a shaft connected to the rotating platform. Connection is done using long set M4 crews that go trough the sliring and fix the shaft in place. 3D printed housing than acts only as a centering guide and isn't load bearing.

Rotating section than has a simple A frame design that elevates the radar which is mounted with bearing enabling the radar to tilt freely. In order to control the tilt a second stepper is mounted on the rotating platform a linked with 2:1 down gearing ration using a standard 8 mm belt to the radar. On second support start there is an optical endstop mounted that is used to home the platform.

Since mechanical stresses are quite low most parts could be 3D printed using standard PLA filament.

The rest of the structure is 3D printed from PLA, since mechanical stresses on the platform are minimal. Only non 3D printed parts are the screws, bearings, and the stepper motors. Final assembly 3.1 measures some 33 cm and has footprint of 20x20 cm.



(a) 3D render



(b) Photo

Figure 3.1: Form of the final assembly

### 3. Rotary Platform

---

#### 3.2.1 Platform Electronics

Electronic side of the project is rather simple – there are only two things to handle: driving of stepper motors and enabling some homing mechanism.

Given the low load on stepper motors and the platform’s inability to accumulate significant momentum, a simple stepper driver without feedback control is adequate. For this purpose, the A4988 stepper driver was selected, due to its low cost, microstepping capabilities and basic current control [13]. Minor disadvantages is the lack of any feedback from the driver to microcontroller including some sort of stall detection. In order to simplify the design A4988 devkits were used soldered onto a prototyping board without creating a custom PCB design.

To implement homing, two potential solutions were considered: Hall effect sensors and optical gates. While Hall effect sensors offer the advantage of angle sensing, allowing correction of any positional drift during operation, they require precise alignment. If the orthogonal Hall effect sensor is not perfectly placed in the axis of rotation, calibration becomes a necessity [14]. While the feedback would be nice, given the fact that microcontroller sends signal for each step taken by the motor in normal operating conditions the position of the platform can be easily calculated using software.

Thus for simplicity and ease of integration, optical gates were selected. This decision eliminates the need for complex calibration while providing reliable functionality.

Driving the whole system is an ESP32 microcontroller. ESP32C6 version was chosen due to author’s extensive experience with this particular version. System doesn’t require any special peripherals or high processing power thus any ESP32 version would be sufficient.

### 3.3 Platform Software Realization

To maximize efficiency in processing commands and ensure accurate stepper motor control, the program workflow is divided into three distinct layers, as illustrated by figure 3.2.

The commonly used two-component architecture—where one component handles communication/command parsing and the other manages execution—was deemed unsuitable for this use case. Such an approach would complicate integration of programming interface and require just-in-time processing of commands, which could lead to performance issues.

In the chosen architecture, the degree of abstraction decreases with each successive layer, simplifying processing at each step. This design allows the final layer to operate with maximum efficiency, where transition from one command to the next is primarily limited by the inertia of stepper motors and not by the software.

#### 3.3.1 Communication layer

The communication layer manages incoming data over the serial line, with efficient handling facilitated with the aid of RTOS queues. Upon receiving data the text string is parsed and either pushed to a queue or added to programm declaration, in case we are currently declaring program.

### 3. Rotary Platform

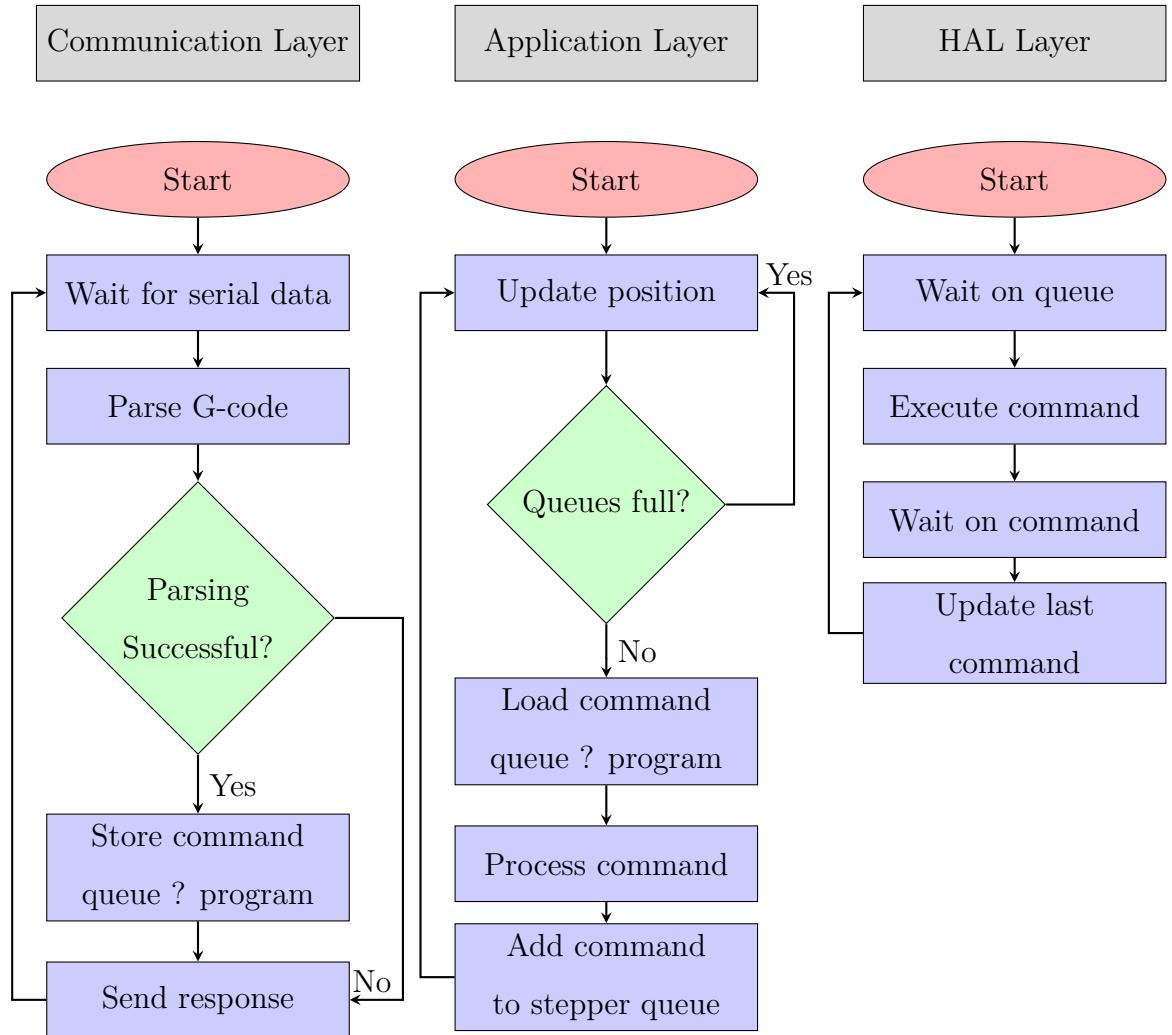


Figure 3.2: Programm diagram

Immediately after parsing, a response is send to the user confirming whether the command was parsed correctly or not. However, as the communication layer does not a can not check command within context of all previous commands, it is possible that command will be parsed correctly but its execution will fail in the application layer.

#### 3.3.2 Application layer

The application layer performs two primary functions: tracking the current device position and scheduling commands to be sent to stepper motors. Aside from current position the program also keeps track of the end position of the last scheduled command. Thanks to this the application layer make necessary calculations to facilitate absolute positioning and enforce movement limits.

A key departure from standard G-code interpreters, like [15], is how the platform handles single-axis move commands. When a move command targets only one axis, the other axis remains free to read next command and begin its execution. If this behavior is undesirable, the user must issue commands for both axes. In relative positioning mode, a zero value results in no motion; in absolute positioning mode, the command must specify the current position to prevent

### 3. Rotary Platform

---

movement.

This behavior is a necessary side effect of the spindle regime, which typically cannot be toggled on or off dynamically. Another consequence is the requirement for separate positioning modes for each axis. Continuous rotation prevents calculations of a move's end position, making it impossible to make calculation for absolute positioning commands – thus necessitating relative positioning. However it would be rather restrictive to force user to relative positioning on second axis, therefore the independent positioning settings.

In order to support or possible operating regimes a manual override mode was also implementing. This enables the user to manually push a move command directly to stepper queues totally skipping the application layer. Primary usecase of this mode is to allow tracking of targets or other application that require real time control of the platform. However in this regime no limits are enforced and the platform operates strictly in relative positioning mode.

#### 3.3.3 HAL Layer

The final layer manages stepper motor control and provides the application layer with essential data for position calculations. In its loop, the program waits for the next command in the stepper queue. Upon receiving a command, it sets up execution, waits for one or both steppers to complete their movement, and then proceeds to the next command. Since limit and absolute positioning calculations are handled in the application layer whole routine remains highly efficient.

The main challenge lies in generating precise PWM signals (Used to control stepper motors drivers.) and stopping signal generation after a specific number of steps. Using the equation:

$$t_{\text{delay}}(s) = \frac{60}{2 \cdot N_{\text{steps}} \cdot s}, \quad (3.2)$$

where  $s$  is speed in RPM,  $N_{\text{steps}}$  is the number of steps (Anywhere from 200 to 1600 depending on microstepping.), and  $t_{\text{delay}}$  is the time between steps, we calculate that even at 30 RPM, the delay between output changes is 5 ms per step. With microstepping at a 2:1 ratio, this reduces to 2.5 ms – faster than lowest sleep interval on ESP32 and without sleeping the RTOS watchdog will trigger. Therefore, signal generation must leverage specialized microcontroller peripherals.

The ESP32 platform offers two options: Remote Controlled Transceiver (RMT) and Motor Control Pulse Width Modulation (MCPWM) combined with Pulse Counter (PCNT). While RMT allows smooth PWM frequency adjustments, it has several drawbacks. Such as the fact that generating a specific number of pulses is supported only on newer ESP32 models [16], synchronization is restricted to its proprietary API, and there is no straightforward way to track progress during a move [17].

For these reasons, MCPWM and PCNT were chosen. MCPWM handles pulse generation, while PCNT counts steps, enabling easy synchronization, continuous rotation, and a robust API for step tracking [18]. The only limitation is the PCNT's 15-bit counter, which caps the maximum steps per move at 32.767.

### 3. Rotary Platform

---

#### Performance of the HAL Layer

Table 3.1 illustrates the stability of PWM generation by the MCPWM module at various speeds. Measurements were conducted using a Saleae Logic Pro 16 logic analyzer, with no microstepping enabled.

The results show that frequency deviation is minimal, though the generated speed is consistently marginally faster than the target, and the error increases slightly with higher speeds. Nevertheless, when measuring time of 24,000 steps at 120 RPM, the relative error in time duration (or speed) was only  $\epsilon = -0.004\%$ , demonstrating excellent accuracy.

Table 3.1: Stability of PWM generation

RPM	$f_{\text{desired}}$ (Hz)	$f_{\text{low}}$ (Hz)	$f_{\text{high}}$ (Hz)	$f_{\text{avg}}$ (Hz)
10	33.334	33.334	33.334	33.334
30	100	100	100.003	100.002
60	200	200	200.01	200.004
120	400	400	400.02	400.007

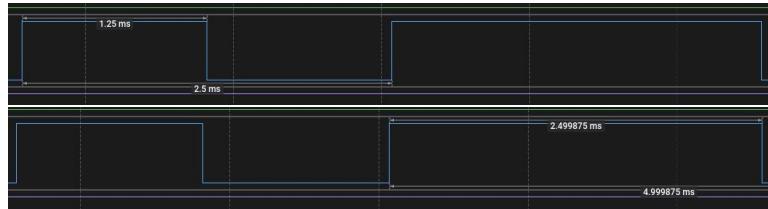


Figure 3.3: Moment of change between commands (120RPM  $\Rightarrow$  60RPM)

An attempt was made to also measure the delay between switching commands, displayed in figure 3.3. The results indicate that the delay between commands is imperceptible. Similar outcomes were also observed for other command combinations.

This demonstrates the efficiency of the HAL layer in managing stepper motor control and transitioning seamlessly between commands. As long as stepper queues are supplied with commands in advance, the platform can operate without noticeable interruptions. Most importantly, the platform's timely and predictable behavior ensures that mathematical corrections to the radar data can be applied accurately.

### 3.4 MATLAB Control Interface

In its basic operating mode the platform control is totally deattached from the radar system. Radar data processing might do some correction based on the platform speed, accelerations and position but it doesn't send any commands to the platform itself. Thus the control flow of the platform is rather simple – user uploads a program to the microcontroller and starts its execution. In order to manage programs and provide window to read platform diagnostics from a simple MATLAB application was created.

After connecting to the platform (Picking correct port and baudrate in preferences, clicking the connect button in the main window.) user can go into platform

### 3. Rotary Platform

---

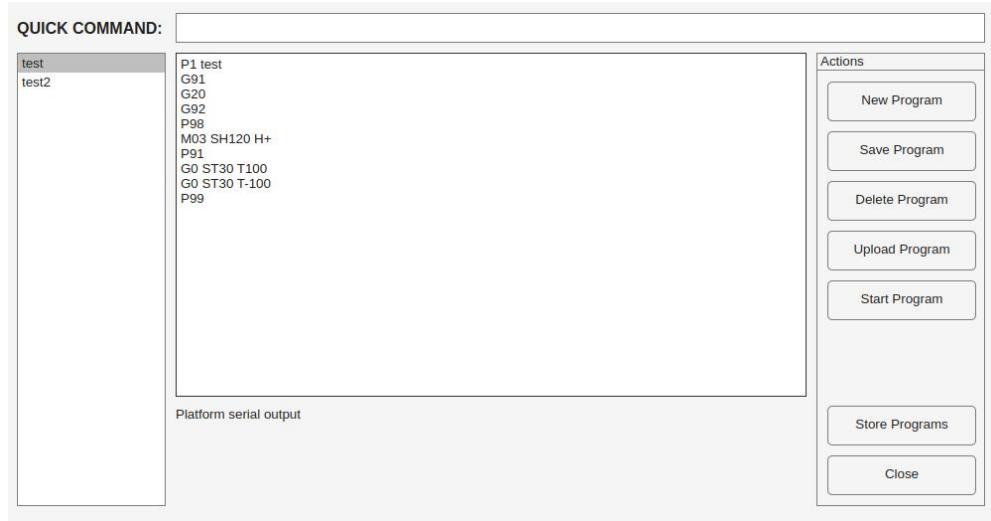


Figure 3.4: GUI of the platform control

control window 3.4. On the left there is a sidebar with a list of all stored programs. Clicking on any of them will load it into the editor windows and by presssing buttons on the right user can send the program to the platform.

After the program has been sent it can be either started manually by sending a start command from the **QUICK COMMAND** box or by pressing the start button. Using the start button will make sure that the platform queues are empty, the platform is homed and only then the program will start. To see feedback from the platform user can see the text window on the bottom of the screen.

## 4. Radar Data Processing

**TODO:** target detection at least using CFAR

---

# Conclusion

TODO

---

# Bibliography

- [1] *Plán spektra ČTU od 24 po 170 GHz* [online]. [visited on 2025-01-23]. Available from: <https://spektrum.ctu.cz/kmitocty?filter%5BfrequencyFrom%5D=24%5C&filter%5BfrequencyFromUnit%5D=GHz%5C&filter%5BfrequencyTo%5D=170%5C&filter%5BfrequencyToUnit%5D=GHz>.
- [2] JANKIRAMAN, Mohinder. *FMCW radar design*. Boston: Artech House, [2018]. ISBN 978-1-63081-567-7.
- [3] INDIE SEMICONDUCTORS GMBH. *SiRad Easy® r4, SiRad Easy® & SiRad Simple®: User Guide*. 2019. Version 2.1.
- [4] BROOKER, Graham. Long-Range Imaging Radar for Autonomous Navigation. 2005. Available also from: <https://ses.library.usyd.edu.au/handle/2123/658>.
- [5] PIPER, S.O. Homodyne FMCW radar range resolution effects with sinusoidal nonlinearities in the frequency sweep. In: *Proceedings International Radar Conference*. 1995, pp. 563–567. Available from DOI: 10.1109/RADAR.1995.522609.
- [6] SULEYMANOV, Suleyman. Design and Implementation of an FMCW Radar Signal Processing Module for Automotive Applications. 2016. Available also from: <http://essay.utwente.nl/70986/>.
- [7] INDIE SEMICONDUCTORS GMBH. *SiRad Easy® r4, SiRad Easy® & SiRad Simple®: System & Protocol Description* [online]. 2017. Version 2.1 [visited on 2025-01-25]. Available from: [https://downloads.ffc.indiesemi.com/datasheets/Protocol\\_Description\\_Easy\\_Simple\\_V2.1.pdf](https://downloads.ffc.indiesemi.com/datasheets/Protocol_Description_Easy_Simple_V2.1.pdf).
- [8] INDIE SEMICONDUCTORS GMBH. *TRX\_024\_007: Datasheet* [online]. 2019. Version 1.5 [visited on 2024-11-19]. Available from: [https://siliconradar.com/datasheets/Datasheet\\_TRX\\_024\\_007\\_V1.5.pdf](https://siliconradar.com/datasheets/Datasheet_TRX_024_007_V1.5.pdf).
- [9] INDIE SEMICONDUCTORS GMBH. *SiRad Easy® r4, SiRad Easy® & SiRad Simple®: User Guide*. 2019. Version 2.1.
- [10] INDIE SEMICONDUCTORS GMBH. *TRX\_120\_001: Datasheet* [online]. 2022. Version 1.6 [visited on 2024-11-19]. Available from: [https://siliconradar.com/datasheets/Datasheet\\_TRX\\_120\\_001\\_V1.6.pdf](https://siliconradar.com/datasheets/Datasheet_TRX_120_001_V1.6.pdf).
- [11] INDIE SEMICONDUCTORS GMBH. *Collimator Lens* [online]. 2021. Version 1.1 [visited on 2025-01-23]. Available from: [https://siliconradar.com/datasheets/Datasheet\\_Collimator\\_Lens\\_V1.1.pdf](https://siliconradar.com/datasheets/Datasheet_Collimator_Lens_V1.1.pdf).
- [12] *UH3899-01 Series USB2.0 Signal Slip Ring* [online]. [visited on 2024-11-10]. Available from: <https://www.senring.com/usb-conductive-slip-ring/single-channel/uh3899-01.html>.
- [13] *A4988 Datasheet* [online]. [visited on 2024-11-23]. Available from: <https://www.allegromicro.com/~/media/Files/Datasheets/A4988-Datasheet.ashx>.

## 4. BIBLIOGRAPHY

---

- [14] CHEN, Xiao; HE, Xiran; ZHANG, Ying; GAO, Bo. Angle Measurement Method Based on Cross-Orthogonal Hall Sensor. In: *2023 IEEE 16th International Conference on Electronic Measurement & Instruments (ICEMI)* [online]. IEEE, 2023-8-9, pp. 153–158 [visited on 2024-11-23]. ISBN 979-8-3503-2714-4. Available from DOI: 10.1109/ICEMI59194.2023.10270587.
- [15] *GCode dictionary* [online]. [visited on 2024-11-24]. Available from: [https://docs.duet3d.com/en/User\\_manual/Reference/Gcodes](https://docs.duet3d.com/en/User_manual/Reference/Gcodes).
- [16] *RMT Based Stepper Motor Smooth Controller* [online]. [visited on 2024-10-03]. Available from: [https://github.com/espressif/esp-idf/tree/master/examples/peripherals/rmt/stepper\\_motor](https://github.com/espressif/esp-idf/tree/master/examples/peripherals/rmt/stepper_motor).
- [17] *Remote Control Transceiver (RMT)* [online]. [visited on 2024-10-03]. Available from: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/rmt.html>.
- [18] *Pulse Counter (PCNT)* [online]. [visited on 2024-11-09]. Available from: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/pcnt.html>.

---

# List of Figures

1.1	Ideal relation of frequency and time for received and sent signal . . . . .	4
2.1	SiRad Easy <sup>®</sup> [9] . . . . .	7
2.2	Flow of Radar Measurement on SiRad Easy <sup>®</sup> [7] . . . . .	7
2.3	Simulated 24 GHz header with boundary conditions showed . . . . .	9
2.4	$s_{11}$ parameter of the 24 GHz header . . . . .	10
2.5	Radiation pattern of 24 GHz header – 3D view . . . . .	10
2.6	Radiation pattern of 24 GHz header – 180° norm . . . . .	11
2.7	Radiation pattern of 24 GHz header – 90° norm . . . . .	11
2.8	SiRad Easy <sup>®</sup> with 122 GHz header [7] . . . . .	11
2.9	Radiation pattern of 122 GHz header comparison [11] . . . . .	12
3.1	Form of the final assembly . . . . .	14
3.2	Programm diagram . . . . .	16
3.3	Moment of change between commands with 120RPM and 60RPM	18
3.4	GUI of the platform control . . . . .	19

---

# List of Tables

3.1 Stability of PWM generation . . . . .	18
---	----