"Real-time" control system for wing twist and flapping frequency

C.H. (Camiel) Jongerius

BSc Report

Committee:
Dr.ir. G.A. Folkertsma
Dr.ir. J.F. Broenink
Prof.dr.ir. A. de Boer

July 2017

027RAM2017
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

This thesis presents the design of a control system for the experimental setup of a wing from the Robird. The Robird is a robotic bird mainly used to scare off birds where they pose serious threats. It mimics the flapping motion of real birds through the use of 2 connecting rods mechanisms. They translate the rotational motion of the servo-motors to the flapping motion of the wing. The twist angle of the wing can be created by having phase shift between the motors. The efficiency of the flapping depends on the frequency and twist angle. To increase the efficiency, experiments have to be done with different values for these parameters.
The current experimental setup needs to stop the system in order to change the parameters. Designing a new "real time" control system to change the parameters while it is running would make it easier and faster to do the measurements. The control system takes in the frequency and the phase shift, and applies currents to the motors attached to the mechanism. The response should be under 4 seconds, the frequency should go up to 7Hz and the phase shift up to 10°. It is designed using 20-sim and runs using 20-sim 4C. New components are chosen to better suit the new control system, they include the RaMstix to run the system and 2 motor controllers.

Acknowledgments

I would like to thank my supervisor Geert Folkerstma for his very useful guidance throughout the assignment and Steven Gies, who is also working with the experimental setup, for his thoughts and discussion. Special thanks to Marcel Schwirtz for his technical advice to help me properly connect all the components of the setup. Thanks to Gerben te Riet for helping me quickly fix the problems that came up from the components. Finally, thanks to my family and friends who supported me during my Bachelor years in the Netherlands.

Contents

1 Introduction

Birds can pose a serious threat to people and their assets. They have attacked crop lands on farms and now a days they can collide with airplanes. As these damages can quickly become dangerous, humanity has developed different methods to scare away these birds. Most of these solutions are limited as some only work for a certain amount of time while others have notable drawbacks. One of these solutions which has fewer limits is the use of a robotic bird, Robird for short, which mimics raptors in shape and movement, such as the Peregrine Falcon. It can be used to scare away those birds as they cannot distinguish it from a real raptor. Some may even use these lifelike Robird for espionage as it is difficult even for us to distinguish them from afar. [2]

Figure 1: The Peregrine Falcon Robird with a real Peregrine Falcon [6]

1.1 Fundamentals

In order for the Robird to mimic the movements of the real bird, its wing movements play an important role. As this movement is very difficult to implement, a lot of study, specifically in aerodynamics, has been done to understand it. To simplify, the downward motion during flapping, called the plunge, pushes the air downwards which produces a pressure difference. The lower pressure above and the higher pressure below pushes the body upwards. This is called the lift. As the wing also takes an angle, called the pitch, during a plunge, the air is also pushed backwards which in turn pushes the body forward. This is called thrust.

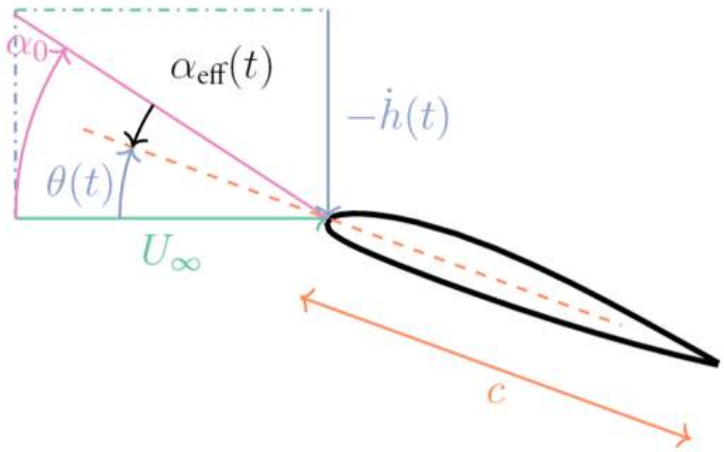

Figure 2: The pitch: θ, the plunging velocity: $\dot{h}$, the forward velocity: U_{inf}, the angle of attack: α_0, the effective angle of attack: α_{eff} and the chord wing length: c during the upward motion. [3]

The shape of the wing plays a very important part in the aerodynamics. Similar to a birds wing, the Robirds wing is curved downward on the front edge. This helps produce more force with downstrokes than with upstrokes to increase lift. During flapping, the back edge creates a street of vortices known as the von Karman vortex which can lead to an increase in energy lost. The Strouhal number, see equation 1, is a dimensionless number which relates the vortex shedding frequency (von Karman vortex), f, the flow velocity U and in this case of flapping, the amplitude of oscillation L. When this number is between 0.2 and 0.4, the thrust efficiency is the highest. The flexibility of the wing also plays an important role as a moderate flexibility was shown to have the highest thrust efficiency. [4] [5]

$$St = \frac{fL}{U} \tag{1}$$

The wing mechanism in the Robird utilizes 2 spars, each attached on one edge of the wing. These spars are moved by a con rod mechanism following a sine wave pattern. They move with a fix amplitude with the same frequency. A lag between the two is used to create a phase shift which causes the plunging and pitching of the wing. As the wing is made of foam and resin, it is very flexible and with the movement of the spars, it create the twisting motion at the end of the wing when changing directions. [3]

In order to increase the overall flight efficiency of the Robird, measurements have to be done in order to optimize the flapping frequency and twist. So far, the current system in place to take those measurements is cumbersome as it requires some manual changes of the phase shift system. This increases the time it takes to change each parameter for each measurement which can add up if many measurements need to be done.

1.2 The current developments

The current experiment setup for doing measurements for the Robird was developed according to the work of Cyrano Vaseur. He implemented a mechanical setup as well as a controller system. The mechanical setup follows figure 3 with 2 motors connected to the gears, which are connected to the connecting rod mechanism that moves the spars. A friction disc is added between the gears to create a secondary phase shift between the spars. The setup physically limited the phase shift between the spar to a value slightly larger that $\pm 10°$. In this report, the motor at the front of the wing will be called motor 1 or leading motor and the motor at the end of the wing will be called motor 2 or lagging motor. The current controller is a cascaded position (P) velocity (PI) controller, which regulates the phase shift between the spars by controlling the motor from the feedback it receives from their encoders. The currently used motor controller meets the power requirements but limits the controller to the PIP type. The current measurement tool is a 6 DoF force sensor. [7]

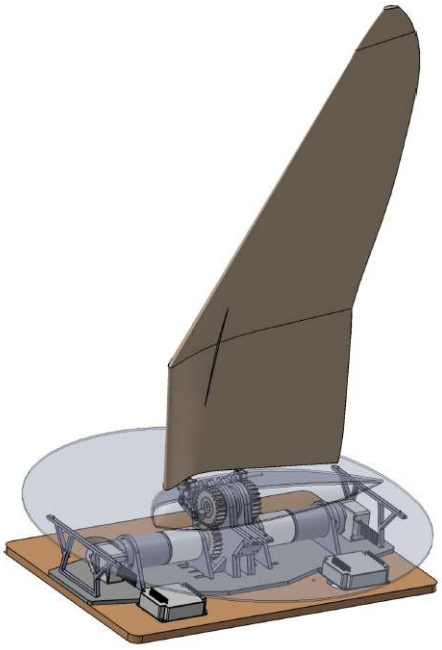

Figure 3: The current Robird measurement setup. [7]

1.3 Aims and approach

With the current test setup for the Robird, measurements can be done but with relatively little efficiency. With the continuous development of the Robirds, faster and more efficient measurements is necessary to apply quick changes. This project aims to enhance the setup by designing a new, more efficient controller to ease the whole measurement process. When using it, the flapping of the wing should have its parameters reach the inputs within 4 seconds. The phase shift range limit is chosen to fit the setup's physical limits. This controller will have the following requirements:

- 1: Two inputs: Flapping Frequency and Twist Angle
- 2: Two outputs: Power Motor 1 and Power Motor 2
- 3: Setup update time: 4 seconds
- 4: Flapping Frequency Range: 0-7Hz
- 5: Phase Shift Range: 0-10° with an accuracy of $e_{max} = \pm 0.1°$

A computational platform will have to be added to upload the controller to the setup. The motor controller will be replaced to remove the limitations. This system may be aided by implementing extra sensors.

While installing the new hardware and repairing the setup, a first controller will be designed using the current 20-sim model to give a preview of the controller needed, as explained in Chapter 2. Chapter 3 covers the actual controller design with direct changes applied to it when testing it through the setup until completion. The results will be presented in chapter 4 and the discussions in chapter 5. In Chapter 6 the conclusions will be given.

2 Model Controller Design

2.1 The Model

There currently exists a model of the Robird which was created to do simulations for different parameters[1]. A first controller was designed using this model but due to its complexity, it wasn't accurate in representing the experimental setup. Therefor a simpler model was created.

The new model is symmetrical where both of its parts represents one of the motor and the spar. The movement of the spar is modeled using the mechanism part of the full Robird model. The wing is modeled as two point masses, each being half of the total mass. Both point masses include drag and are connected at the end of each spar. The point masses are also connected together through a spring-damper combination which represents the crosswise flexibility of the wing. Without this, it would be easier for the model to change the phase shift as if there would be nothing blocking the spars.

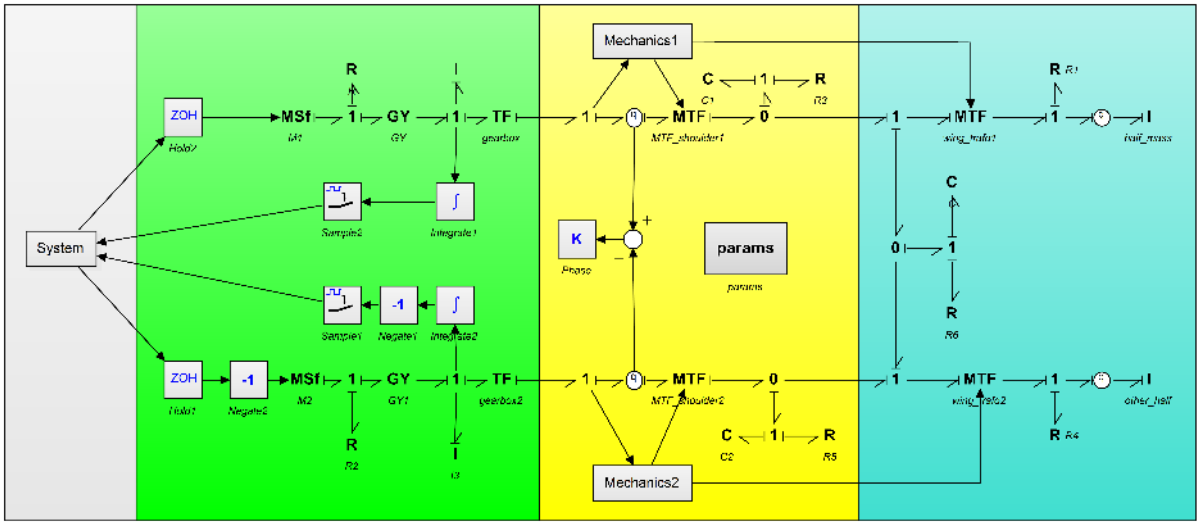

Figure 4: The new model used to create the controller. The grey section is the controller, the green section models the motor and the gearbox, the yellow section models the connecting rod mechanism and the blue section models the wing.

The drag, r, has parameters taken from Frank's model.

$$r = \frac{CdA\rho}{2} = 0.063112 \tag{2}$$

with the drag coefficient $Cd = 1.28$, the area $A = 0.0805m^2$ and the air density $\rho_{air} = 1.225kg/m^3$.[1]

As the wing was simplified to a point mass with drag and the gravity was not taken into account, the longitudinal flexibility of the wing which is a relatively important and complex parameter. These approximations will affect the final result as this means the model expects the wing to change direction faster than what really will happen.

2.2 The Controller

The controller, which can be seen in figure 5, necessitates 2 input, the frequency and the phase shift, and 2 outputs, both motors. In order to do so, an angle measurement device will be used on both motors for the feedback loop. The controller will be separated into 2 distinct parts.

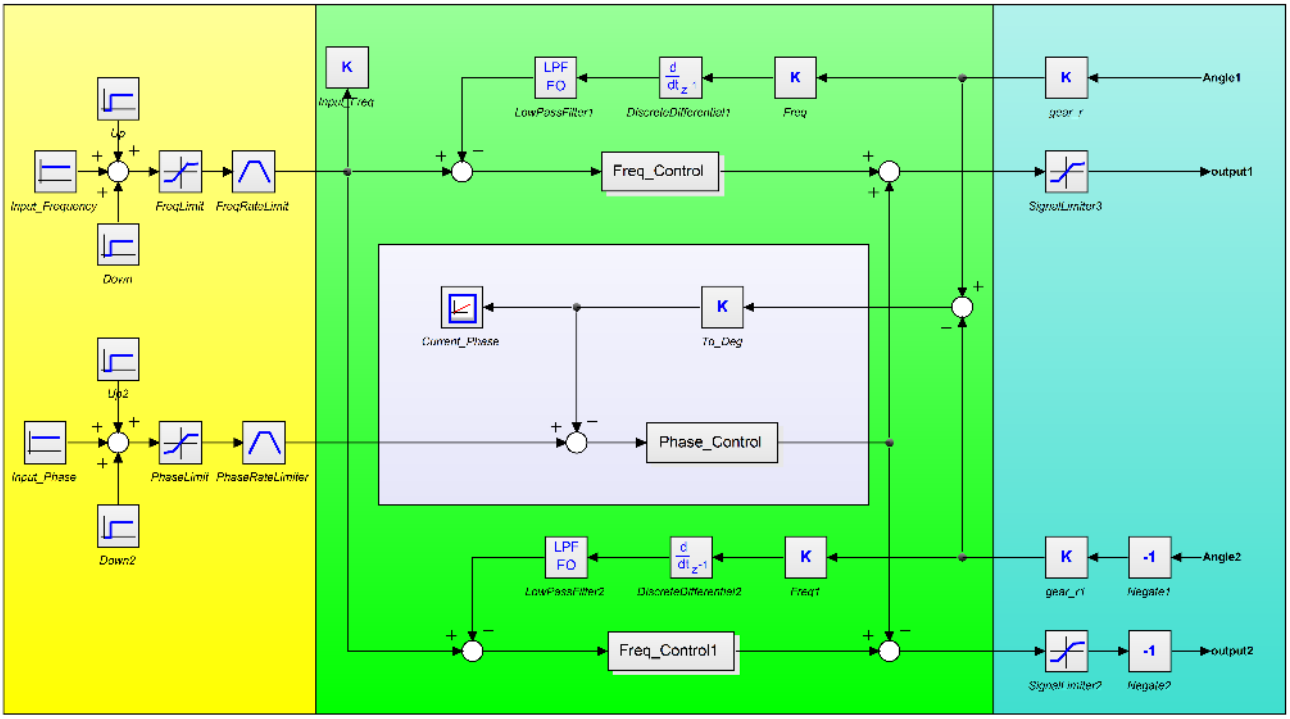

Figure 5: The controller used in the model. The yellow section represents the user inputs, the green section the frequency controller, the grey section the phase shift controller and the blue section the inputs and outputs of the system.

The first part controls the flapping frequency through controlling both motors separately but with an equal input. By measuring the angle of each motor, the motor's rotational frequency can be calculated which ultimately translates to the current flapping frequency. The PID controller will be the same for both motors as they will receive the same input, both motors should have the same value and they will affect the same system.
The PID parameter values were determined by first finding a working value for each gain. starting with a proportional gain, an increasing value, with large steps is applied and simulated. This is done until the result worsens instead of improving. With this, a range and a approximate value is obtained. The same method is done with the integral gain followed by the derivative gain. When applying even a low derivative gain, the results were always worse and would even destabilize. The value of the derivative gain is therefore 0.
After obtaining the necessary approximations, the optimization tool is used. It will automatically run the simulation using different values between the range given by the user. The nominal value can be filled in as the best approximation the user obtained. The tool will optimize the values for a minimum or maximum result. Here, the frequency error

needs to be minimized. The number of steps will determine how precise the values will be after the optimization but will largely increase the processing time. The number of steps will also increase depending on how many variables are being used. As the derivative gain is already 0, using the tool for the proportional and the integral gain is the most efficient. The optimization can be reused to obtain a more precise value for the gains by decreasing the range after every use. The values obtained are as follows:

- Proportional gain: 1
- Integral gain: 1.67
- Derivative gain: 0

As the frequency needs the velocity of the system instead of the position, a differentiator is added before calculating the error. This causes noise which needs to be reduced using a low pass filter. Having a too low bandwidth can also hurt the signal as it will reduce the value of the flapping. A bandwidth of 150Hz produced the best results.

As the motors work separately in the frequency part, they are currently independent which gives a random phase shift. However, this also strongly reduces the phase shift between the motors at the start as both motors will begin to rotate simultaneously to match the given frequency. With this, the second part of the controller needs to be created to obtain the desired phase shift.
By measuring the angle of both motor's, the phase lag can be calculated following equation 3. The phase shift controller can use this information and the input phase shift to change the input of the motor. As the phase shift, ϕ, is defined from equation 3, the phase shift controller's output must be added from the leading motor's input and subtracted to the lagging motor's input.
The PID controller parameters were determined using the same method as used for the frequency control. For this controller, the derivative gain affects the system positively when it is near but not equal to 0. It will have to be included in the optimization and as the value is very small, its value can be quickly determined using the tool. The values obtained are as follows:

- Proportional gain: 2
- Integral gain: 6
- Derivative gain: 0.04

$$\Phi = \theta_1 - \theta_2 \tag{3}$$

After obtaining good values for the gains of the controllers, the proportional gains had to be adjusted to reduce the outputs to the motors. If the inputs becomes too large, controlling will become difficult and will increase the error.

A limiter is added to the input of the system following the requirements of the controller. This simplifies the problem as there are now four extrema for which, in those range, the controller needs to work as intended, which can be tested for. Furthermore, a signal rate limiter is added to the input to diminish its rate of change, which reduces the risk of

breaking the model and the system. As the user wants the system to update as fast as possible, the change rate was chosen to fit both requirements. The maximum rate change was chosen as a third of the maximum input of the system such that when the input requires the system to go from one extrema to the other, it will take 3 seconds to achieve it. For lower changes, it will take less time.

The system is non-linear due to its strong and sudden change in movements which unables us to use tools such as the Nyquist plot or Bode diagram to create the controller. However, using the tools given in 20-sim and instincts with prior knowledge of control engineering, a controller was created which seems to work correctly following the model. It can be seen which gain should be more significant from what is controlled. The frequency, meaning the velocity plays a major in the controller. However position is measured. This suggests a higher integral part to balance the controller. Using a high derivative part would suggest controlling acceleration instead. During measurements, the true results will be obtained.

3 Setup Controller Design

3.1 The Setup

The final setup follows the construction seen in figure 6. The controller is uploaded in the RaMstix using 20-sim 4C. It receives information from the encoders and sends the required input to the motor controllers. The motor controllers are programmed, using Escon Studio, to send a current to the motors following a PWM input of the RaMstix. Detailed connections of each components can be found in the appendix.

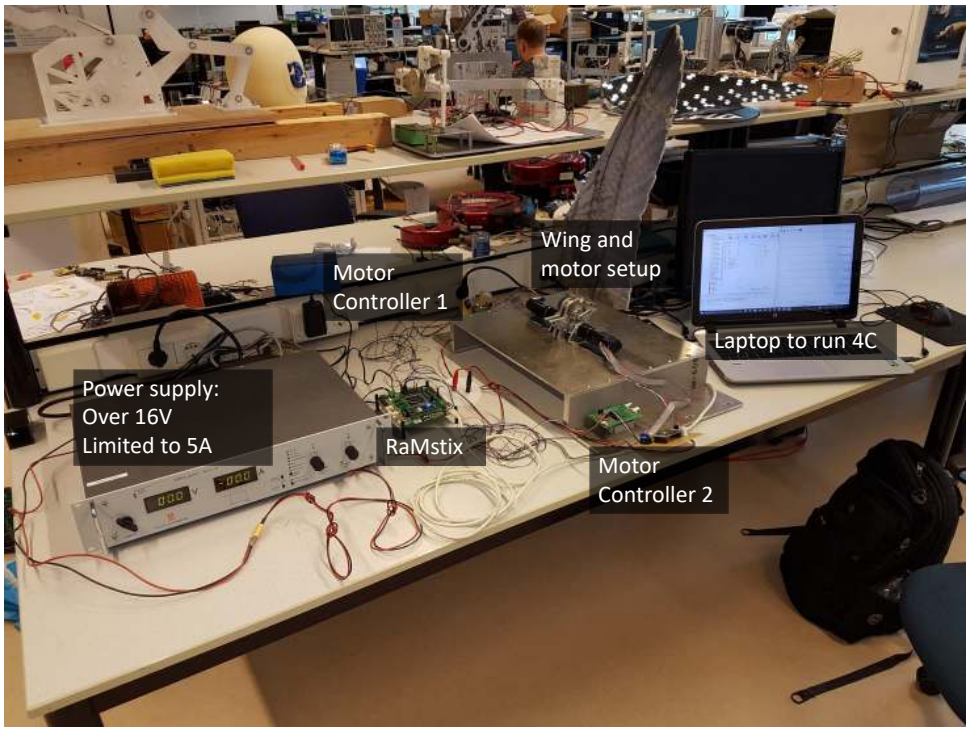

Figure 6: The final setup for the wing control measurement.

The programming of the motor controllers only need to be done once, but both motors need to be programmed separately. The steps to do so can be seen in the appendix. 20-sim 4C has to be used continuously for the measurements as it is this program which helps us run the RaMstix and obtain the data for the measurements. Every step to use it can also be seen in the appendix. For each measurement, there are some safety steps that need to be applied such that the motors do not activate when not asked to. If these steps are not followed, the motors may run from an impulse at the start or the end of running the measurements. This is caused by having the motor controllers enabled when running or stopping the measurements. They will send an unwanted current to the motors at that time. At the start, the motor controllers are automatically enabled, however they are not automatically disabled.

- Step 1: Check the variables: InputFrequency (C) and InputPhase (C) should be 0, Enable (Amplitude) should be 1.
- Step 2: After selecting the values you want to measure, run the program.
- Step 3: Change the frequency and phase as wanted, the system should reach the output within 3 seconds excluding connection time.
- Step 4: When done, Enable, InputFrequency and InputPhase must be changed to 0 starting with the InputPhase and finishing with Enable.
- Step 5: Stop the program and save the measurements.

3.2 The Controller

The structure:

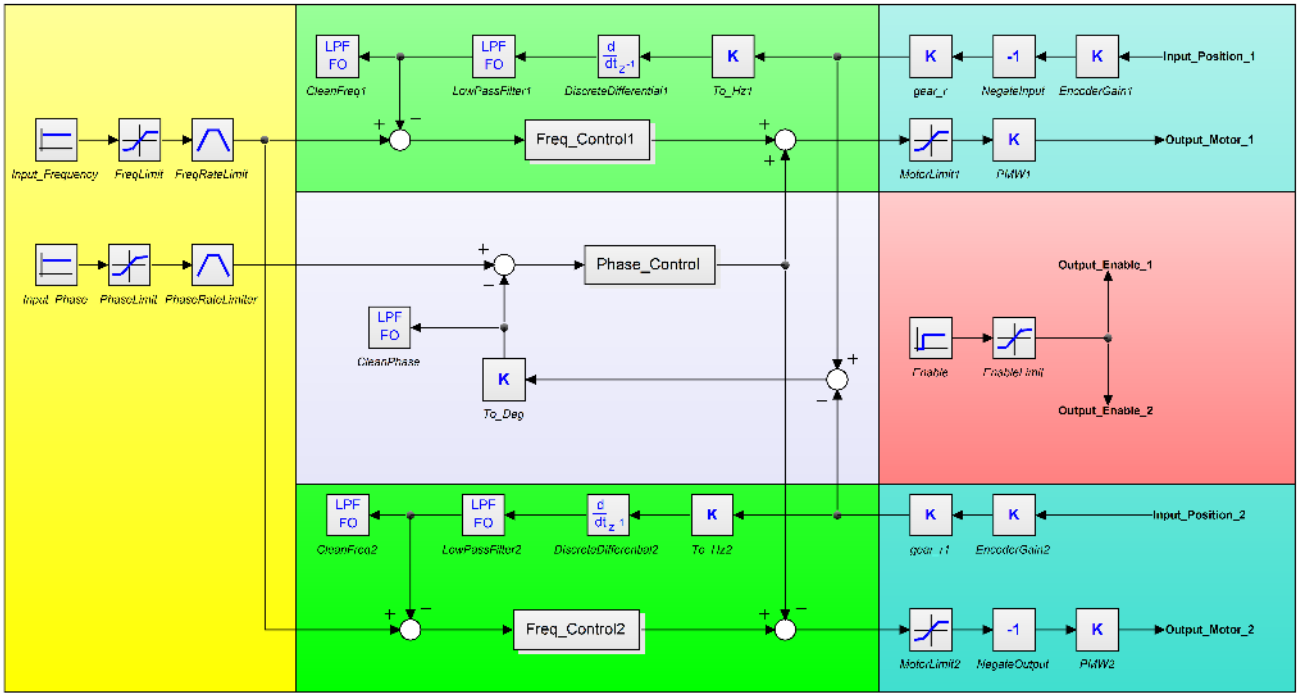

Figure 7: The 20-sim controller for the setup. The yellow section represents the user inputs, the green section the phase shift controller, the grey section the frequency controller, the red section the enable function and the blue section the inputs and outputs of the system.

The controller used for the setup will be substantially different than the one used in the model, which can be seen in figure 7, as now the physical inputs have to be taken into account. The angle is not measured directly but rather through the pulses from 2 channels of the encoder. The gain will translate these pulses into the angle we are looking for according to the following equation:

$$output = \frac{\pi}{180} * 0.18 * input \tag{4}$$

$$\frac{360}{4 * 500} = 0.18 \tag{5}$$

"$input$" is the total number of pulses counted, $0.18°$ is the resolution of the encoder and the whole is then translated into radians. The resolution was calculated using equation 5 as it is 500 counts per revolution and each channel produces 2 pulses per count. The "$Input\_Frequency$" and "$Input\_Phase$" remain unchanged as they will be used to change their value in real-time during the measurements. For the output to the motor, a PWM signal will be used, which means that we cannot define the output as -5 to 5 Amps. The value will have to be translated where -5 is represented with 0.1 and 5 with 0.9 using the following equation:

$$output = \frac{2}{25} * (\frac{25}{4} + input) \tag{6}$$

Unlike the controller used in the model, the setup has an "$enable$" function which enables or disables the motor controller's functions to reduce the risk of accidentally running. This will help keep the motor turned off when everything is shut down. If the motor controller would be connected and it wouldn't receive any signal in the PWM input port, it would translate to sending -5A to the motors. This is done by having a simple constant output of value 1, which enables the motor controller when everything is turned on. This is what prevents the motors from running unwillingly at the start and the end of the run. By having the enable output come from a step function, it will automatically enable the system after running the program such that the user doesn't have to do it. Making this automatic reduces another risk that comes from the controller. The integral portion of the controller helps overcome effects like friction as it increases until the value of the parameter reaches the users input. If the system would be set to a high frequency while being disabled, the integral output of the controller would increase. If it would be enabled after some time, the motors will use all the stored power. Another safety system is added to prevent this problem which is to limit the possible output of the integral portion of the controller. After doing some measurements, the output was usually below 0.5 which is the value of the limiter.

In the physical system, the motors are mirrored which means that a positive current in both would turn one motor in the opposite direction compared to the other. This also means that turning the system in one direction would record a positive value in one encoder and a negative value in the other. In the final controller, this problem is solved by multiplying one of the input and output by -1. The input and output has to be determined through an experiment as simply choosing one motor/encoder combination may lead to "switching" the leading and lagging motor, which doesn't work with the current setup use of connecting rod mechanism. A special controller was build to make these experiments which didn't contain any PID controllers and had simple motor inputs. This is to make sure the necessary variables were closely controlled. The controller can be seen in figure 8.

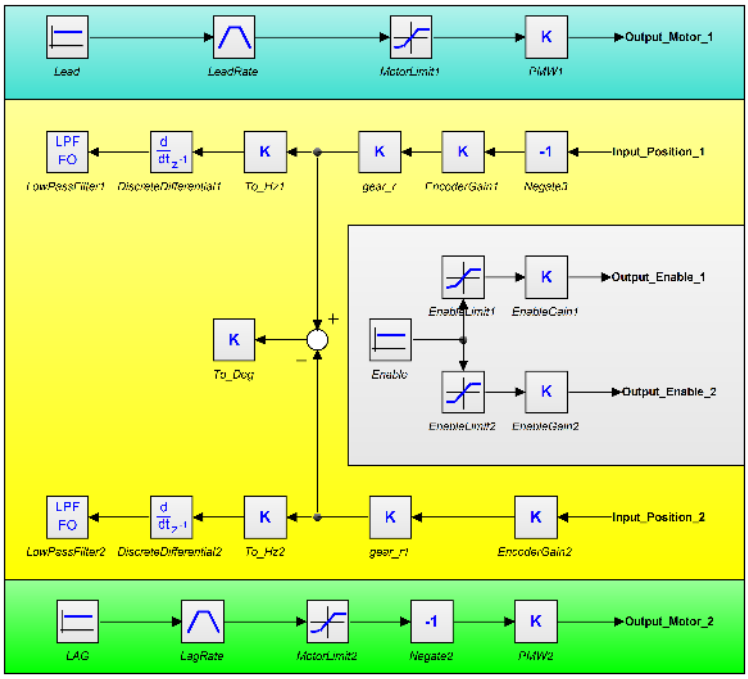

Figure 8: The 20-sim controller for making tests on the setup. The blue section represents the output to the first motor, the green section the output to the second motor, the yellow section the inputs of the encoders and the grey section represents the enable function.

This experiment was done by putting a positive then negative current in one motor at a time and read out the encoder. A positive encoder readout is taken as the positive direction. After doing a couple of times, the multiplication by -1 could be placed in the controller, which can be seen in figure 8. One final test was done to make sure the directions were correct. The result can be seen in figure 9. A positive input followed by a negative input is applied to the lead motor at 12 seconds. This motor's input was not multiplied by -1. Both outputs resulted in the positive direction followed by the negative. Note that the encoder gain 1 would already be multiplied by -1. The same input was then applied to the lagging motor at 51 seconds. Before sending it to the motor, this input is multiplied by -1, meaning a negative input first followed by the positive input. The same result was obtained which shows that the placement of the -1 multiplier at the leading encoder's output and the lagging motor's input is correct.

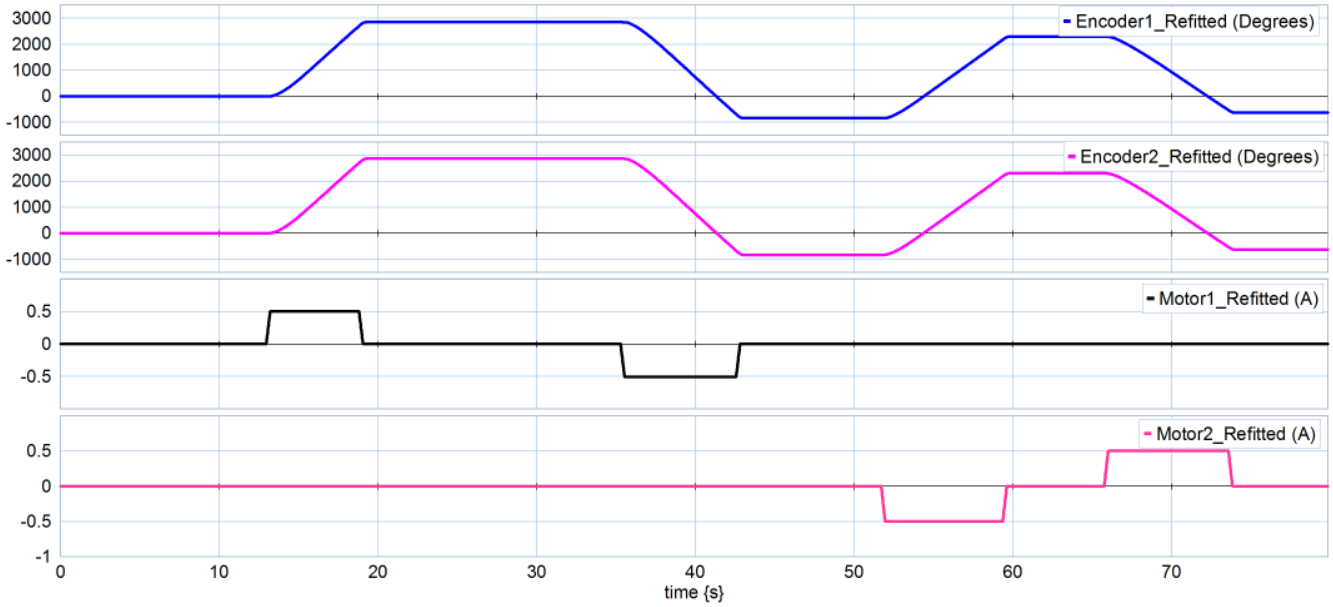

Figure 9: The results for testing the direction of the motor and the encoder.

The values:
When using the PID values obtained from the controller used in the model, the system is very unstable and loses control very quickly even when having an input of 0Hz and $0°$. This is most likely due to the model not accurately depicting the setup, otherwise the system would have been more stable. The non-linear friction between the gear connections may have a large influence. It has a fairly important impact when moving the system, but it was not modeled. Other differences may be from the values used for the different spring-damper combinations between different parts of the system.

New values had to be determined through the use of the setup in order to find suitable PID values for both controllers. This was first done with the setup without the wing for safety purposes. Both controllers were determined using low frequencies and then tested for higher frequencies. It was then tested and enhanced using the wing on the setup by using the same method as the wingless setup. The method to find those values are very similar to the one used in the model, where values are applied and tested to see which works better. Unlike the model, the setup will directly be affected by the values, meaning it will have to be stopped for every destabilizing value and there is no optimization tool so the computer cannot run multiple values quickly. It wouldn't be useful to use the tool in any case due to the reason explained before as it will be more difficult to stop the system.

The steps to obtain the values also differed in order to put forward safety. Firstly the bandwidth was diminished due to the seemingly large effect of the bandwidth on the system, as there was a lot of noise. The change was great when changing it from 150Hz to 50Hz, but the changes were not noticeable when changing the value around 50Hz. Then the attention was put on the phase controller first as this needed the least movement from the setup. As the phase shift input's stayed to $0°$, the proportional gain value's was determined. When it became to large, the system would destabilize, otherwise the setup would be difficult to move by hand due to the input. The integral gain was obtained through the same means. By alternately increasing the proportional and integral gain until destabiliza-

tion, then increasing one or the other, the destabilizing threshold was obtained. Afterwards, a differentiator gain was added as damping to stabilize the system by increasing the settling time, to give more time for the controller to react. This helped increase the other values of proportional and integrator gain. After applying the same methods to redetermine the proportional gain and integral gain, the values where it destabilized started around 0.2 for both. The differentiator gain could be redetermined as well in order to increase the other gains but using 0.004 was already too large and produced unwanted results such as the destabilization of the motors outputs, different than the general setup destabilization.

When the phase shift controller functioned properly, the frequency controller could be added. The same steps were followed by starting with the proportional gain, followed by the integral gain. Here, the integral gain was added relatively early due to the high friction point. As discussed in the model, the differentiator gain remains 0. The destabilizing threshold were obtained around 0.9 for the proportional gain and 1.9 for the integral gain. The new values used are:

Frequency Control

- Proportional gain: 0.8
- Integral gain: 1.85
- Derivative gain: 0

Phase Control

- Proportional gain: 0.1
- Integral gain: 0.1
- Derivative gain: 0.003

The PID values for the frequency control are very similar discounting the decrease of the proportional gain in order to keep the system stable. The PID values for the phase control are very different as it is now much smaller than the model counterpart. It destabilizes quickly with anything larger than the values chosen.

4 Result

4.1 The Model

Model PID values:
The results obtained from the model will give insight for the setup and a better understanding of the setups result. The model controller responds well to the users inputs and follows said input with precision. Despite that, it is not very accurate at higher frequencies. The results can be seen in figure 10.

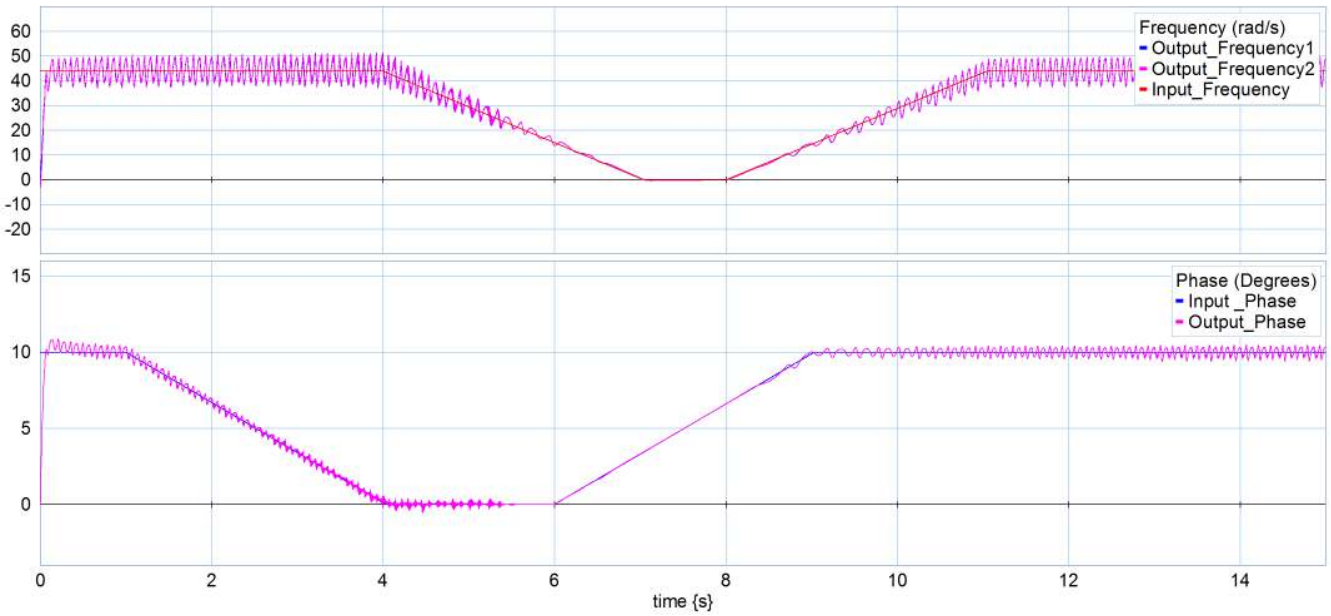

Figure 10: The results for testing the model.

When running the simulation at 7Hz and $10°$ from the start, it takes less than 0.1 seconds to reach the input. This is merely an evaluation tool in order to visualize the expected reactivity of the system as in the setup, the system will always start with an input of 0Hz and $0°$. When changing the input, the system follows the change with high precision and nod deviation is observed. The highest errors obtained are as follow:

- High Frequency:$\pm 5.97 rad/s$ or $\pm 0.95 HZ$
- Low Frequency:$\pm 7.72 rad/s$ or $\pm 1.23 HZ$
- High Phase: $\pm 1.31°$
- Low Phase: $\pm 0.69°$

The errors are very large, larger than the requirements given, but as it fluctuates around the set point, it does not represent the error which would be obtained. The average will be the final expected error.

- Frequency:$e = \pm 0.1412 Hz$
- Phase: $e = \pm 0.31°$

Matlab is used to calculate the Root Mean Squared error of the model, whose data was exported as a csv file from 20-sim. Equation 7 is used and the following is obtained for the maximum input which has the maximum error:

- Frequency: $RMSE = 3.6191 Hz$
- Phase: $RMSE = 0.2662°$

The same can be done for different inputs to see how the input effects the accuracy of the system. The following shows the RMSE for an input frequency of 2Hz and for an input phase shift of $5°$:

- Frequency: $RMSE = 0.6600Hz$
- Phase: $RMSE = 0.0811°$

$$RMSE = sqrt(mean((y_{output} - y_{setpoint}.^2)) \tag{7}$$

With these different error measurements, it becomes clear that for the frequency, the average error is small but it varies a lot. For the phase shift, the average error and variation in error is very small, but it is still superior to the maximum error. For high inputs, the system is much less accurate than for low inputs, as can be seen when comparing the RMSE values for the maximum input and that of the inputs of 2Hz and $5°$.

Setup PID values:
As the controller for the model did not work in the setup and new values for the PID had to be determined, applying those values in the model may give some insight on the controller, which can be seen in figure 11. The result for the frequency does not change much but that for the phase changes drastically. This is do to the little difference in the PID values of the frequency controller and the large values in the phase shift controller. These changes greatly diminishes the accuracy of the controller at high frequencies and increases the settling time a little.

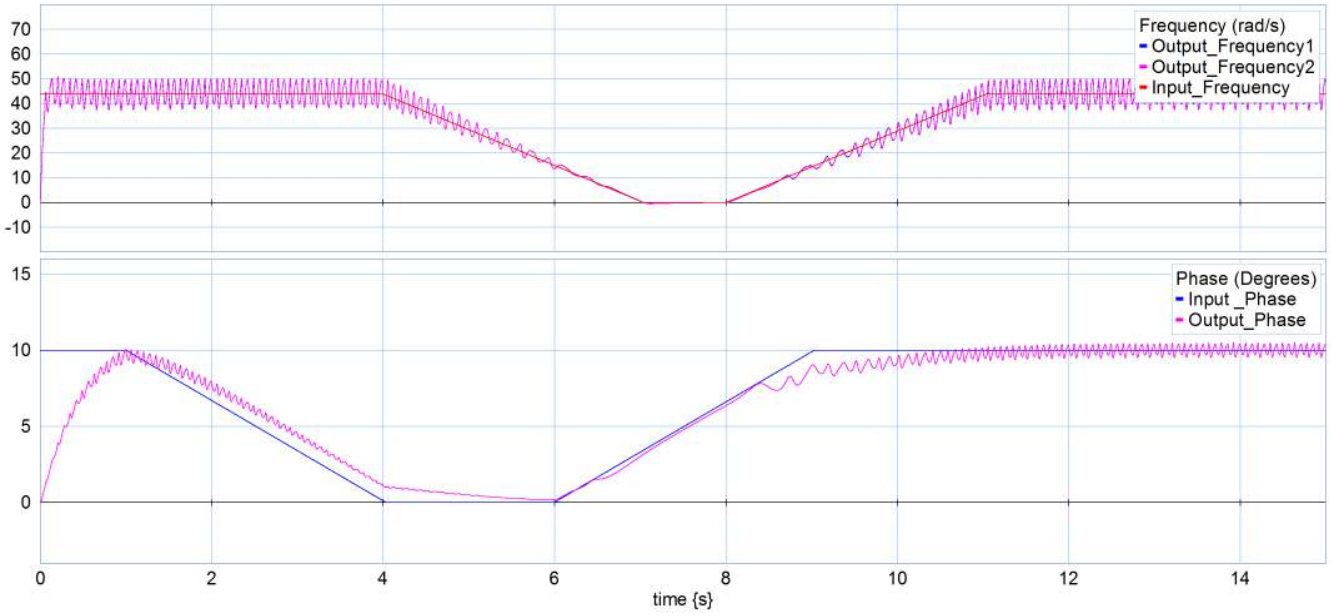

Figure 11: The results for testing the model with the values of the controller of the setup.

4.2 The Setup

The results obtained from the setup will finally show if the designed controller can make the wing flap at the frequency and the twist the user had inputted. Due to the differentiator, some noise is visible which makes it harder to determine the exact values of the setup. A low pass filter, different than the in the controller, is used to make the result more readable.

Result of the frequency control:
The results for testing the frequency can be seen in figure 12. The phase shift was kept at $0°$. There does seem to be a large problem with the frequency as there seems to be a large offset depending on the input. At low frequencies, the output seems to follow the input but the noise makes it difficult to determine for certain. At high frequencies, it is certain that there is an offset as an input of 7Hz does not even reach 6HZ of output. Another problem becomes very apparent when going from a higher frequencies to 0Hz. The system stabilizes near 0.5Hz instead of stopping. This can be seen in the frequency output in figure 12 at around 95 seconds. By using 1Hz for some time before applying 0Hz, the system is able to stop. Here, low frequencies were applied before going to 0Hz which explains why it did not stay on 0.5Hz for too long. This is most likely due to the integral portion of the controller as it retains the influence of the earlier frequencies.

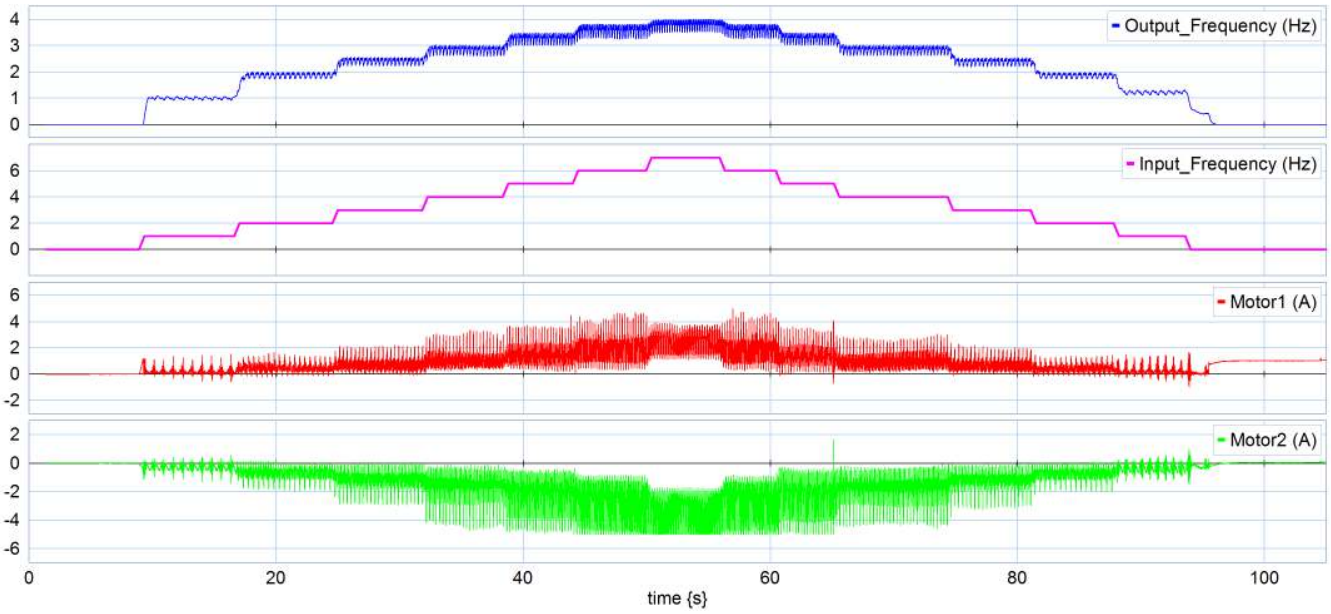

Figure 12: The results for testing the setup at different frequencies with a phase shift of $0°$.

Result of the phase shift control:
The results for testing the phase can be seen in figure 13. The frequency was kept around 3.5Hz. Unlike the frequency output, the phase shift seems to follow the user input with more precision. This is less so for high phase shift such as at $10°$ where it seems unable to reach it, see figure 13. This is most likely due to the motor current limit which can also be seen in the figure, as that limit is reached. This result is supported by the increase of wind created from the wing when increasing the phase shift. When changing the frequency, a small change in the phase can be seen, see figure 13 at time 10 and 100 seconds. Sometimes, when changing the input, the phase shift may be locked at an unwanted phase as can be seen at 55 seconds. This is however temporary and will reach the desired input within some time, not more than a few seconds.

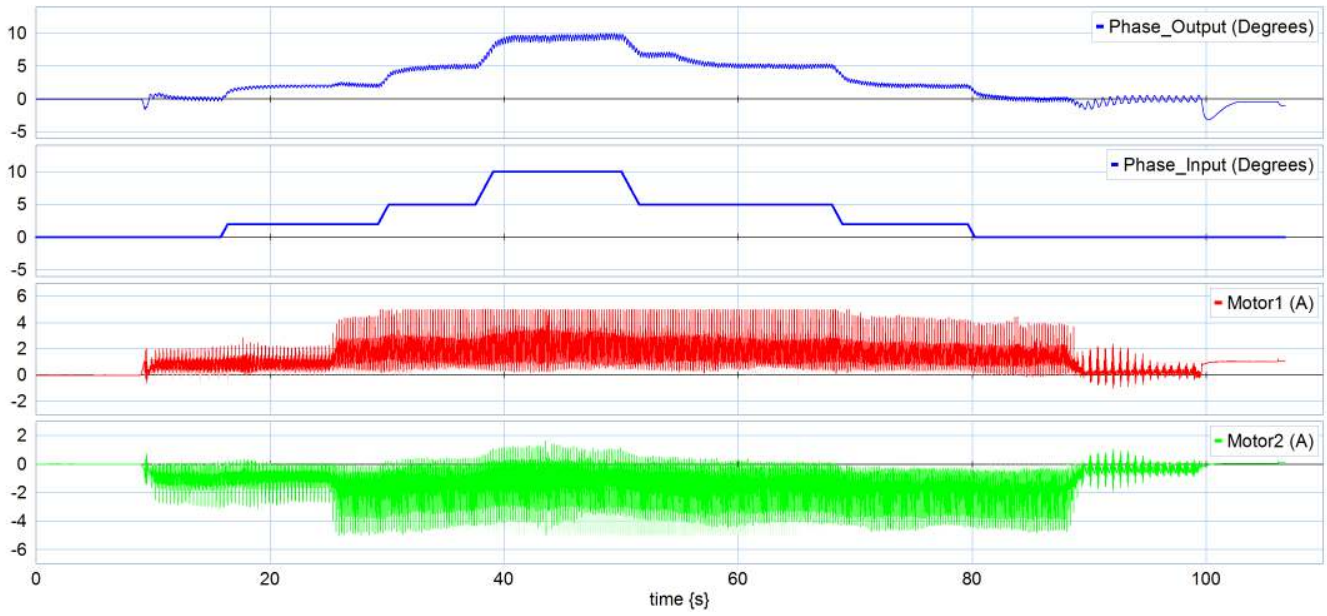

Figure 13: The results for testing the setup at different phases with a frequency of 3.5Hz.

Result for controlling both parameters:
The frequency and phase shift control works correctly at low frequencies but it may be necessary to have a relatively large frequency while changing the phase shift. When doing so, the output current to the motors reach the limit and the required phase has not been reached as seen in figure 14.

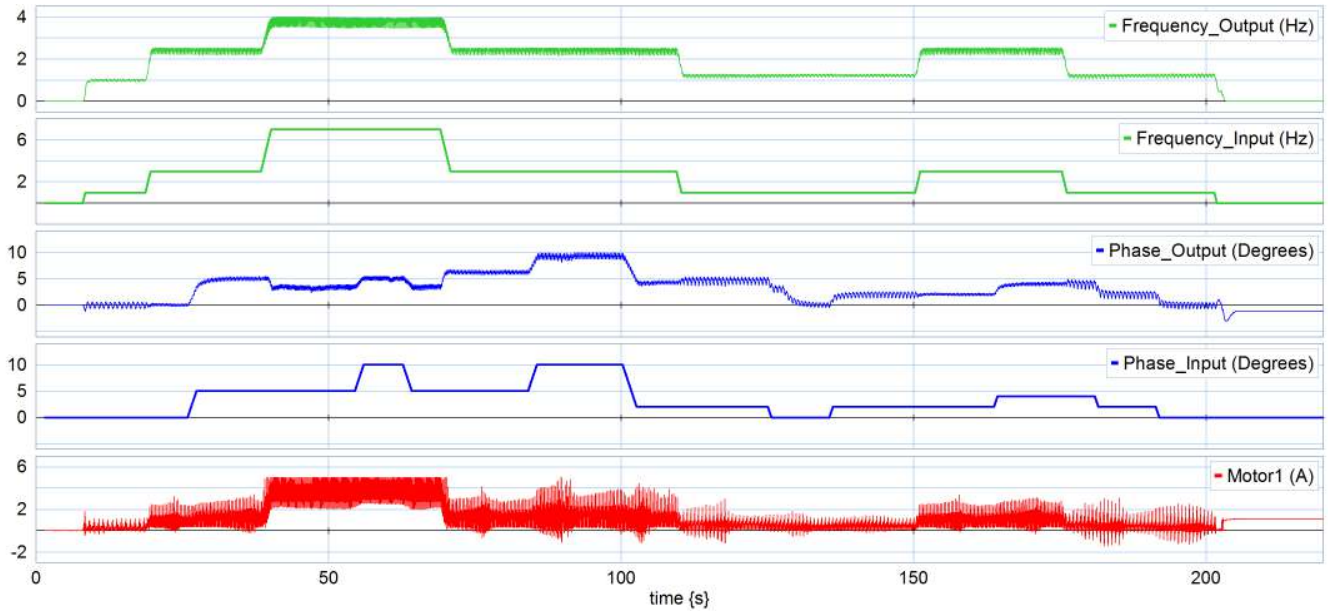

Figure 14: The results for testing the setup at different phases and different frequencies.

Result for the reactivity of the system:
The system is generally reactive to the users inputs as the values of the frequency and phase shift change accordingly unless there is a connection problem. When viewing the response

of the frequency when going from 0Hz to 7Hz as an input, see figure 25, the system reaches the value within 3 seconds with an extra 1 second response time and settling time. When viewing the response of the phase shift, the same can be said. The system reaches the value within3 seconds excluding the extra 1second response time for going from 0° to 10°, see figure 15 in the appendix.
From these results, the values reach the input within 4 seconds as required. But in both cases, the system may be temporarily locked at a specific value as can be seen in figure 14 at 200 seconds for the frequency and figure 13 at 50 seconds. This will decrease the reactivity by a couple of seconds, even if this happens unpredictably, it does not happen constantly.

Result on the non-linear friction:
The results of doing measurements to view the effects of the non-linear friction can be found in figure 17 in the appendix. There are a couple of spots in the setup where the friction is higher. This spot is where the setup will usually stop when it runs at 0.5Hz after applying a 0Hz input. It will also usually block the setup from moving when a frequency of 1Hz is applied. It is because of this friction that a high integral gain is applied. At these low frequencies, it is difficult to tell if the controller does work due to that high non-linear friction. The controller will have to be adapted after fixing that friction.

Result of the system through non-digital measurements:
Due to the difficult visualization of the output due to noise in 20-sim 4C, a more accurate measurement method was needed to be used. A slow-motion camera is used to measure the output frequency of the system. This directly gives the value of the output instead of obtaining digital measurements. The frequency can be calculated using equation 8. The slow-motion has a time factor of $1/8$. The results seems to be consistent as some of these measurements were redone and very similar result were obtained.

- 1 Hz: 20 cycles were counted in the span of $124/8 = 15.5$ seconds which approximates to 1.3Hz.
- 2 Hz: 30 cycles were counted in the span of $100/8 = 12.5$ seconds which approximates to 2.4Hz.
- 3 Hz: 30 cycles were counted in the span of $77/8 = 9.625$ seconds which approximates to 3.1Hz.
- 4 Hz: 30 cycles were counted in the span of $65/8 = 8.125$ seconds which approximates to 3.7Hz.
- 5 Hz: 40 cycles were counted in the span of $76/8 = 9.5$ seconds which approximates to 4.2Hz.
- 6 Hz: 40 cycles were counted in the span of $70/8 = 8.75$ seconds which approximates to 4.6Hz.
- 7 Hz: 40 cycles were counted in the span of $67/8 = 8.375$ seconds which approximates to 4.8Hz.

$$frequency = \frac{cycles}{time} \tag{8}$$

In figure 19 to 25 in the appendix, we can see the digital measurements of the non-digital measurements done above. The results from the camera do not fully agree with that of the digital measurements. They both show the trend of the frequency decreasing at higher input frequencies but they do not agree with the values of the frequency. The frequency obtained from the camera is closer to the input then the digital measurements. Where the camera measured 3.1Hz for an input of 3Hz, the system measured roughly 2.4Hz, see figure 21 or even figure 12.

Other notable outcomes:
As the controller was already proven to be imprecise, it will be difficult to determine the accuracy error without including the imprecision error.
The integral portion of the controller plays a large role as it enables to surpass the non-linear friction in order to reach the desired input. It may also come with some disadvantages as it will continuously increase until the value is achieved. If the system is blocked while there is an input, when the blockage is released, the system will send a power contained when it during the block. This may be dangerous so a limit can be added to prevent that. Here a limit of 0.5 was chosen. This limit may also affect the performance as it may restrain the system from reaching the inputs. Figure 18 shows the value of the output from integral part, which is greater than the limit of 0.5. This high value may be explained as the system has not reached the frequency demanded most likely due to the limits of the setup as the maximum of 5A is achieved.
At larger frequencies, starting near 4 Hz, the force created by the flapping become very large. The system will have to be immobilized somehow in order to diminish the large vibrations and the displacement the whole system makes. The same especially applies for the windshield.

5 Discussion

The controller reacts to the users inputs and changes the frequency and phase shift accordingly. The reaction time of the system is usually as expected. The phase shift is precise when it is not affected by the motor or the frequency change. At low frequencies, the system performs at its best.

Oppositely the controller is imprecise when controlling the frequency, especially at higher frequencies. The effects of previous inputs is too large which hinders the performance for short amounts of time. At high valued parameters, the controller is not accurate. The controller is not consistent as the system may be temporarily blocked at undesired input values.

Different improvements can planned in order to increase the performance of the controller. Better tuning of the controller is necessary to diminish the effect observed from previous inputs. This can also be expanded to reduce or nullify the temporary blockages that were observed and increase the precision and accuracy of the system. A feed forward controller can be added in order to strongly diminish the effect the phase shift receives when changing the frequency.

The setup will also have to be improved. The non-linear friction will have to be fixed, this will also diminish the need for the integral part of the controller, which will have to be

re-tuned. As was seen in the results, the maximum of 5A was reached when approaching high frequencies and phase shift. Changing the motors and motor controllers to increase the limit will increase the range of frequency and phase shift the system can achieve. The use of a differentiator in the system creates noise, adding a sensor which can measure the frequency, directly or indirectly without the use of a differentiator can greatly reduce the noise. One possibility would be adding a gyroscope at the start of the wing, whose signal can be integrated in order to obtain a velocity related to the frequency.

6 Conclusion

Conclusively, the controller works to a certain extent. It fulfills most of the requirements such as working in "real time" but does not manage to fulfill a few of them completely.

- 1: The system has two parameters, the flapping frequency and the twist angle, which the user can change by inputting the values desired in the running controller through 20-sim 4C and the RaMstix.
- 2: The system sends 2 outputs, which are the motor currents necessary to achieve the desired inputs. They are sent through a PWM signal going from -5A to 5A.
- 3: The response time of the system usually fits the required 4 seconds. The new desired input is reached within 3 seconds excluding the connection time. The system may occasionally have a longer settling time, mostly notable during the adjustment of the phase shift.
- 4: At low frequencies, the output is considerably precise. For higher frequencies, the output is completely off. It is not able to reach the 7Hz it was designed for. Its accuracy also depends on the input, where higher input values leads to lower accuracy. The model agrees with these conclusions as the RMSE is about 3.6Hz for an input of 7Hz and drops to 0.6Hz for an input of 2Hz.
- 5: The phase shift is able to reach the whole range of values it was designed for with the exclusion of inputs nearing 10°. At high frequencies, that range strongly diminishes due to the current limits of the setup. When the frequency changes, the phase shift also temporarily varies. The error does not fulfill the requirements for higher input values. The model agrees with these conclusions as the RMSE is about 0.3° for higher inputs and drops to 0.08° for lower inputs.

7 Appendix

A Other results

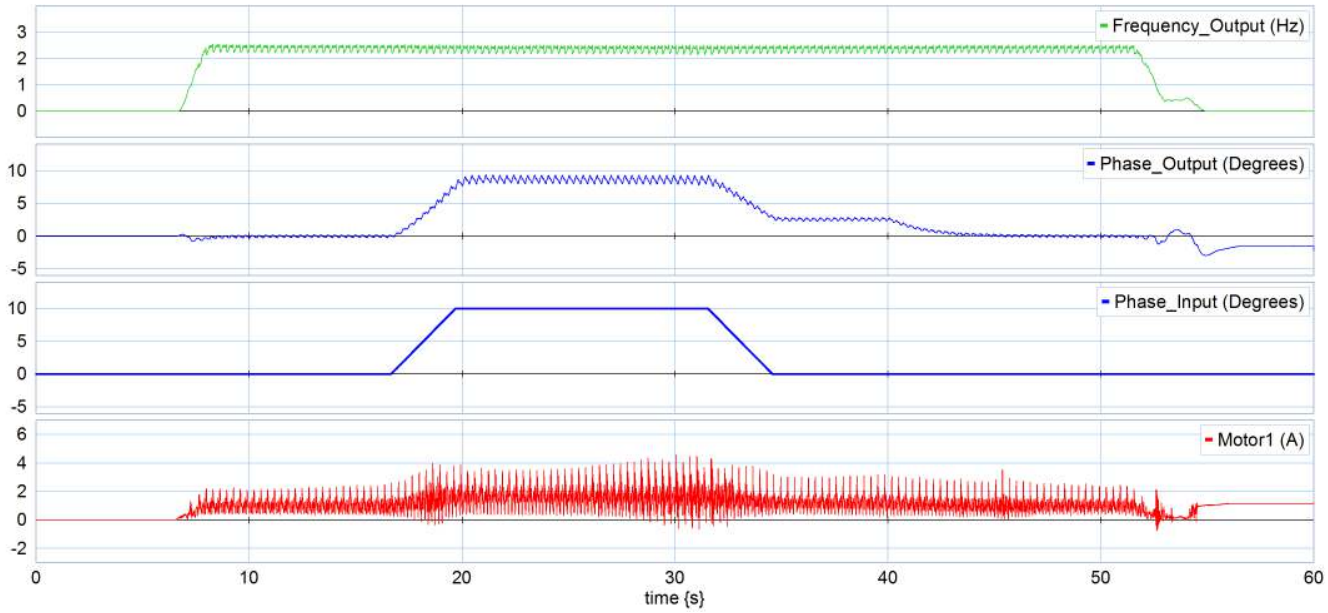

Figure 15: Measurement done at a frequency of 3Hz and a phase shift of 10° to view the reactivity of the phase controller.

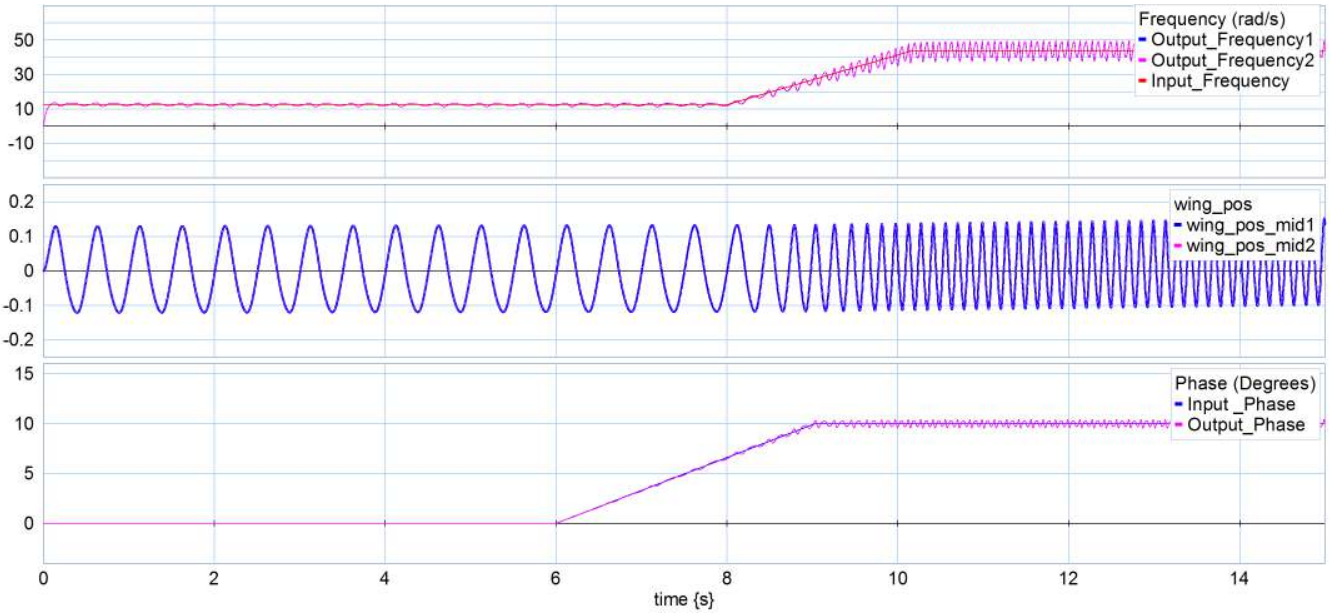

Figure 16: The movement of the wing created by the model is as expected.

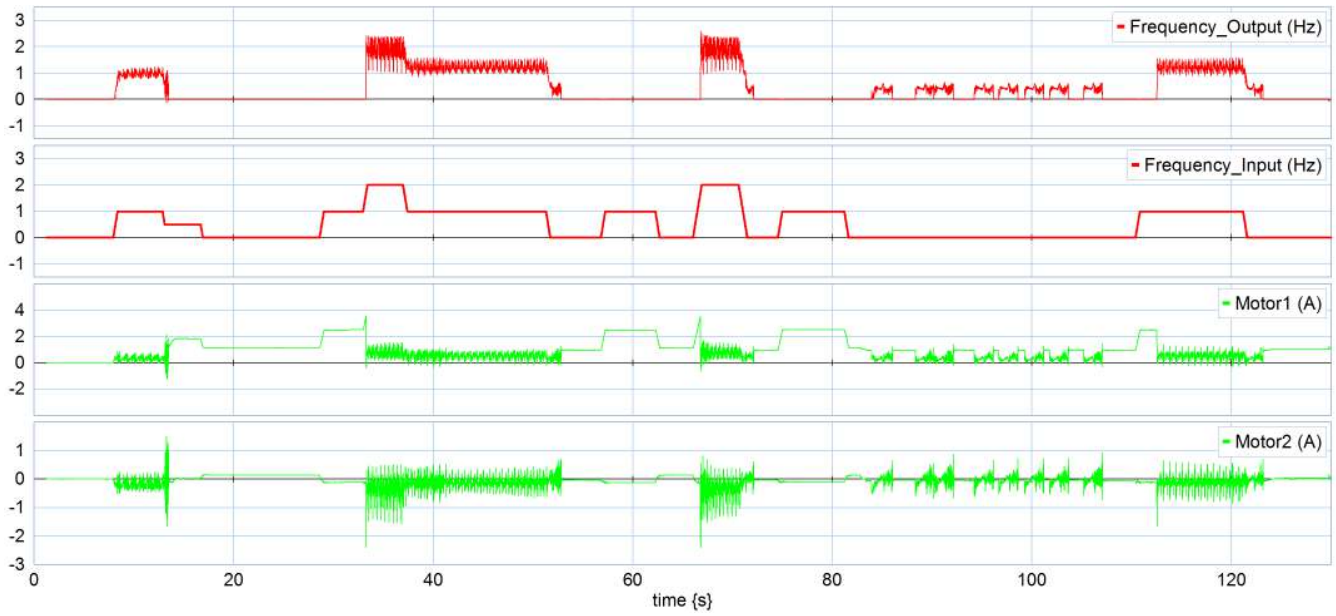

Figure 17: Measurements done at low frequency and a phase shift of $0°$ to view the effect of the non-linear friction.

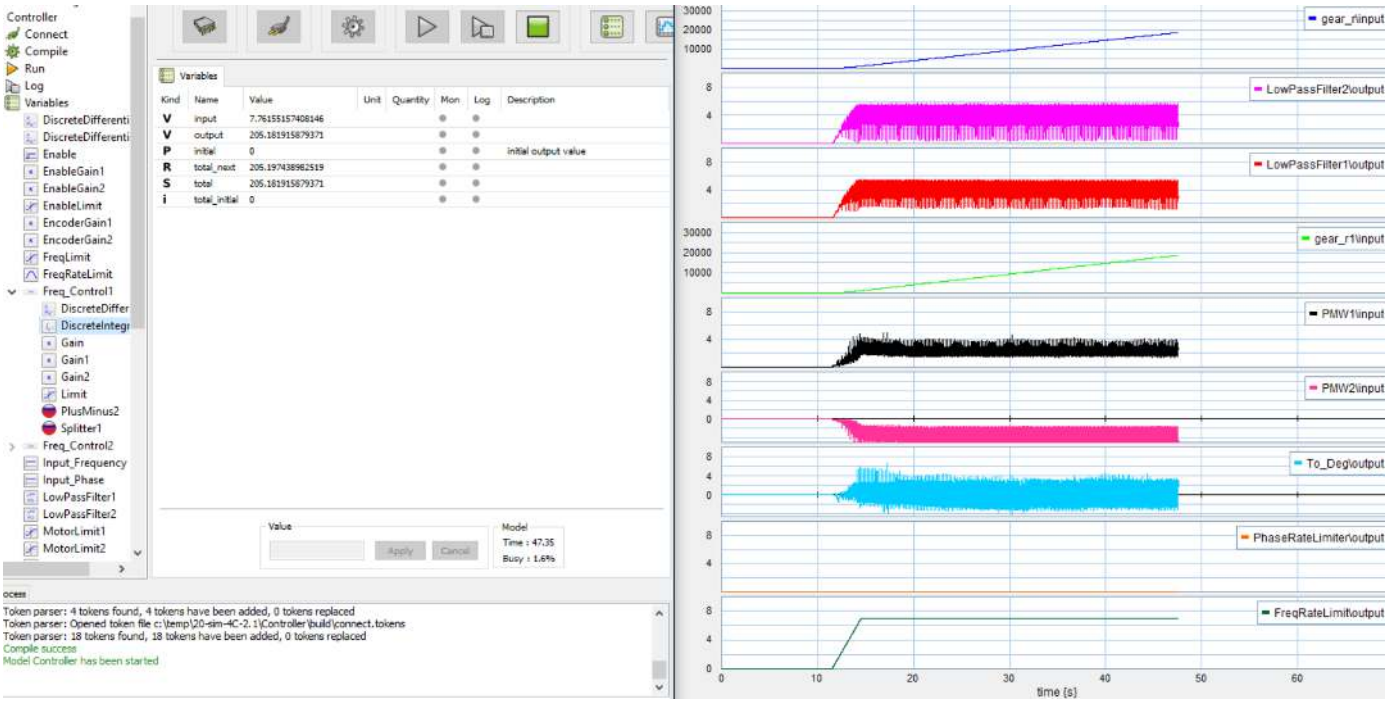

Figure 18: Measurements done at 7Hz and $0°$ before the application of noise reduction. The system does not achieve 7Hz and the output of the integral part of the controller reaches 205.

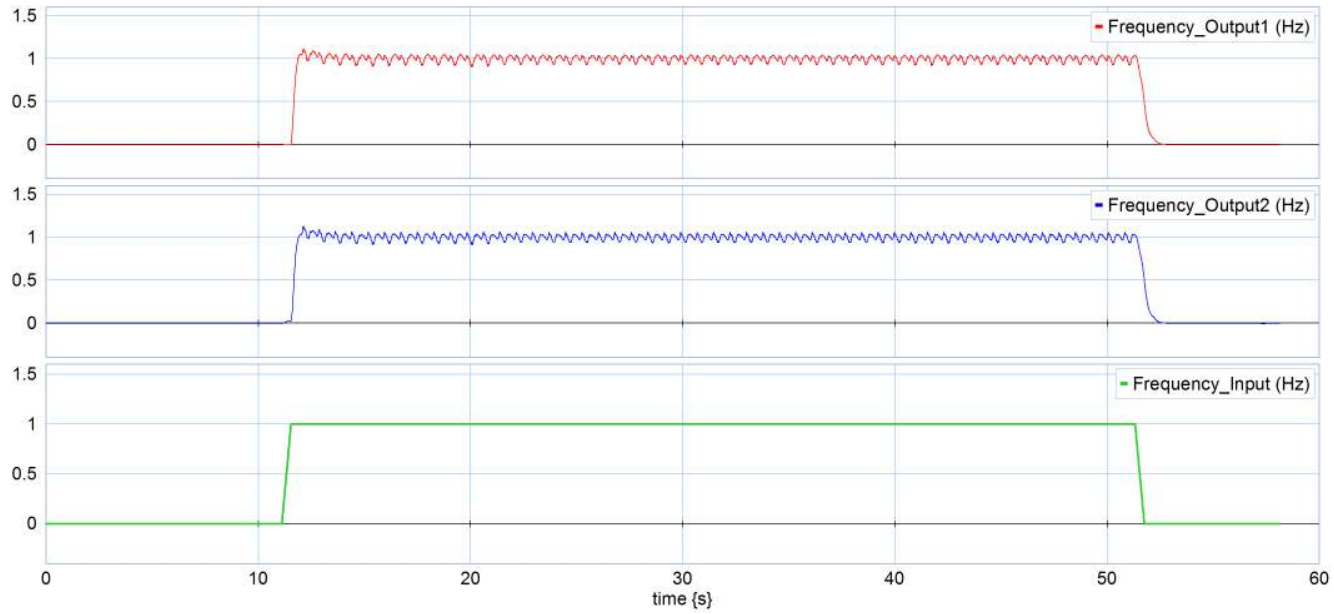

Figure 19: Measuring the system at 1Hz and 0° for non-digital measurement.

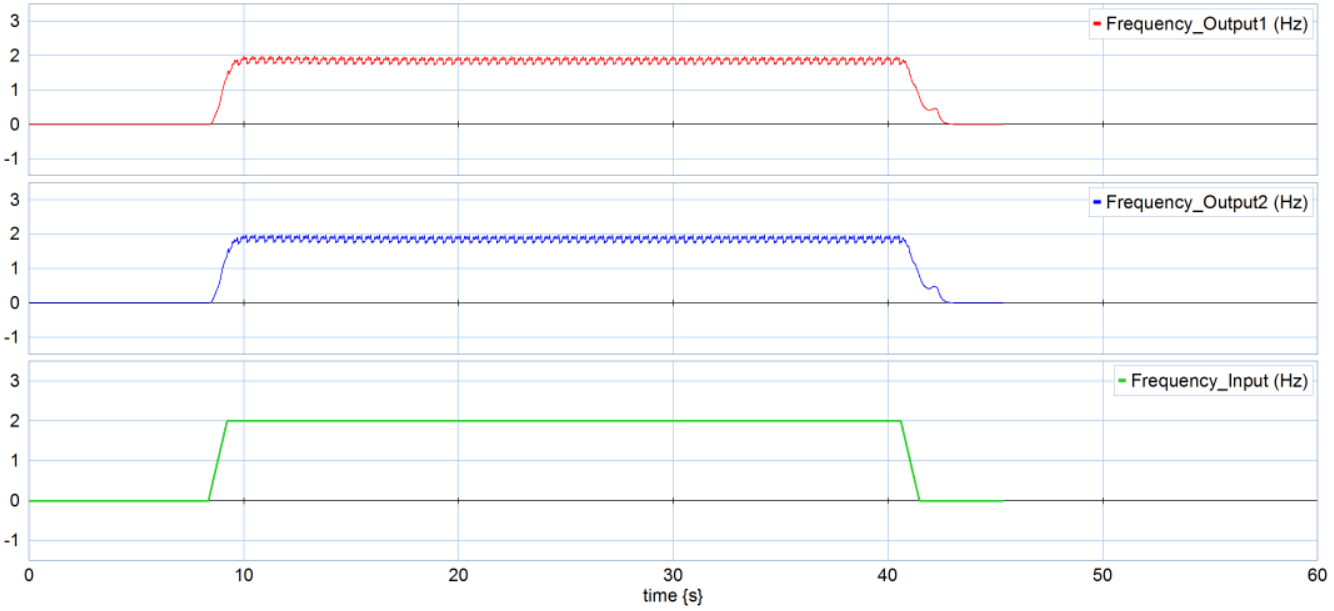

Figure 20: Measuring the system at 2Hz and 0° for non-digital measurement.

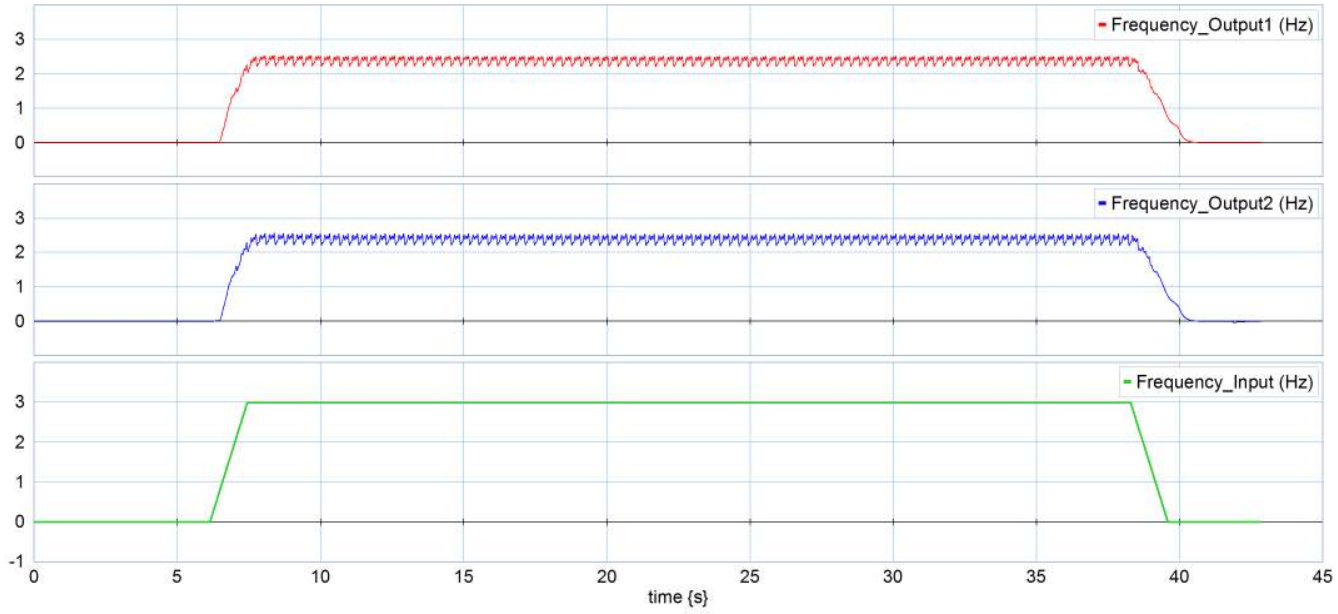

Figure 21: Measuring the system at 3Hz and 0° for non-digital measurement.

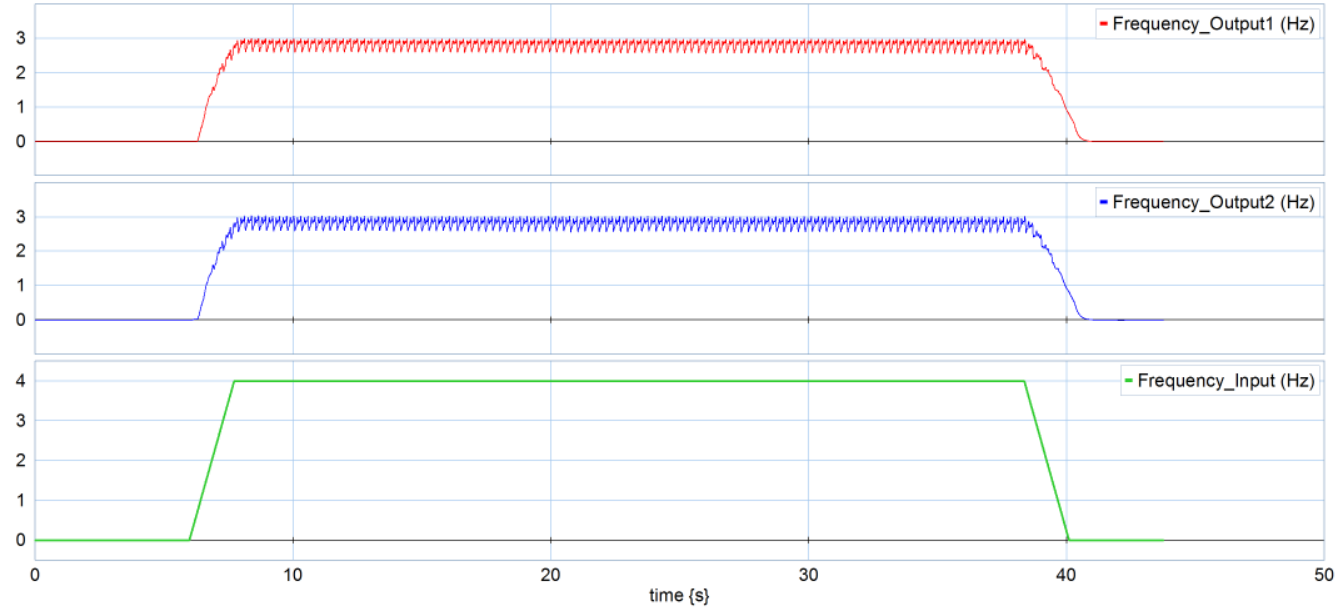

Figure 22: Measuring the system at 4Hz and 0° for non-digital measurement.

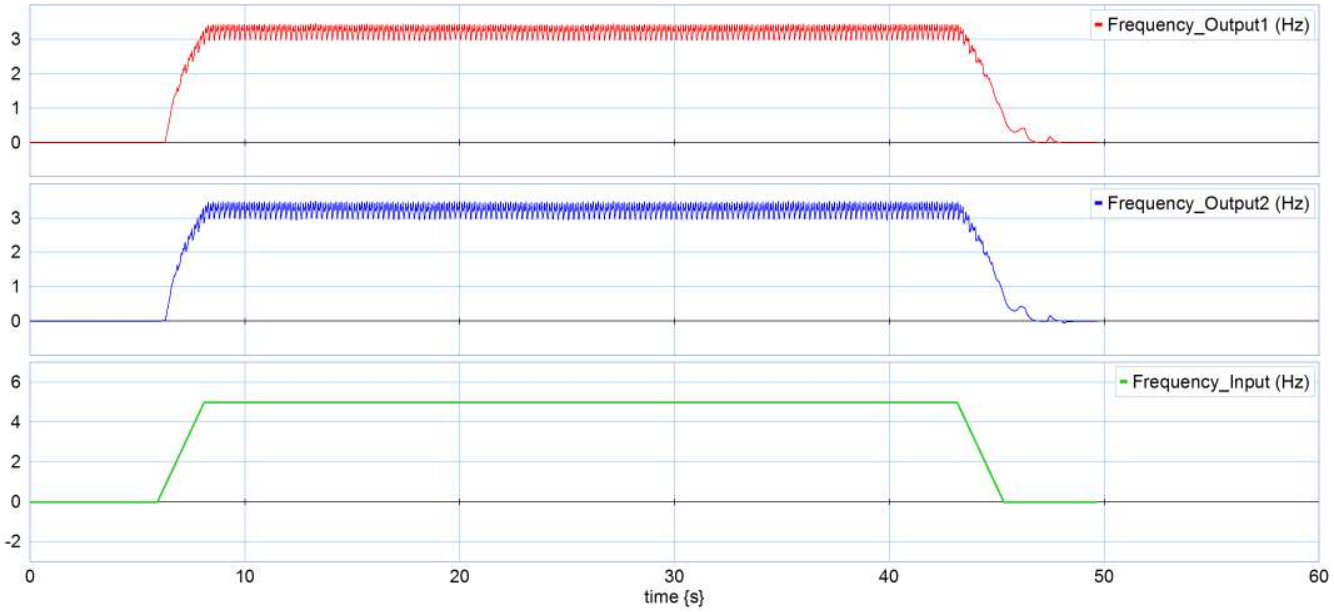

Figure 23: Measuring the system at 5Hz and 0° for non-digital measurement.

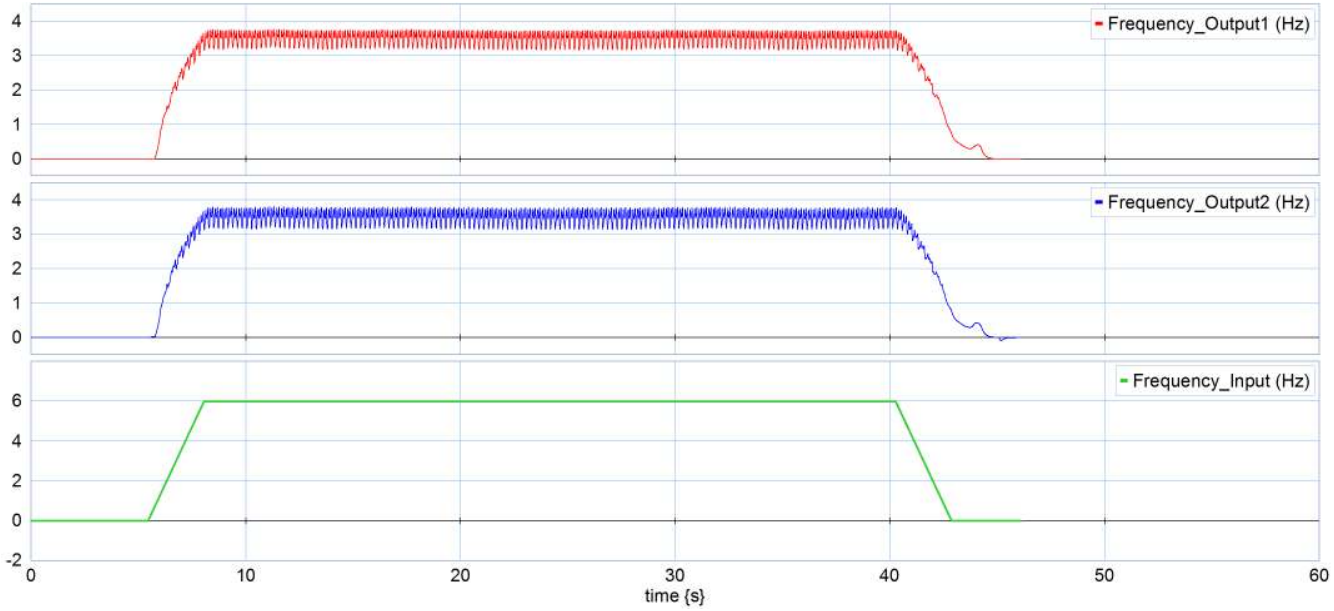

Figure 24: Measuring the system at 6Hz and 0° for non-digital measurement.

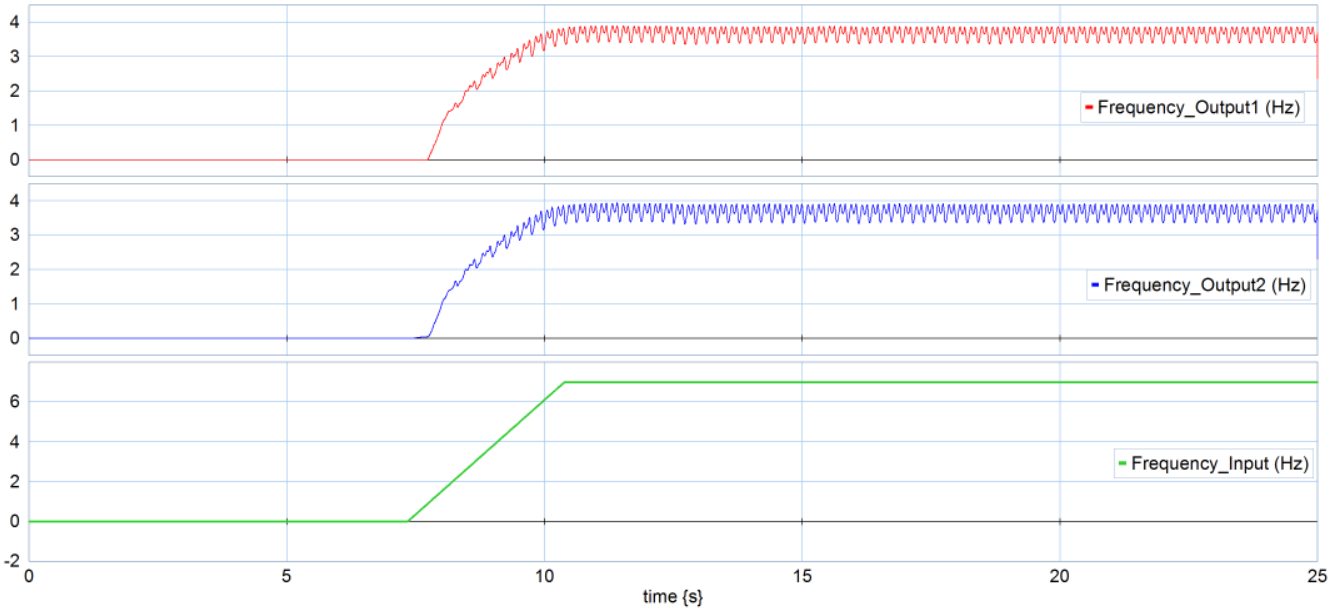

Figure 25: Measuring the system at 7Hz and 0° for non-digital measurement.

B Equipment Selection

Motor:
Maxon EC 32 diameter 32, brushless, 80 Watt, CE approved.
http://www.dioda.hu/pdf/d06046.pdf

Gearbox:
Maxon Planetary Gearhead GP 32 C diameter 32 (ceramic version).
http://www.maxonmotor.nl/medias/sys_master/root/8821069578270/16-342-343-EN.pdf

Motor Controller:
Maxon Escon Module 50/5 motor controller.
http://www.maxonmotor.nl/medias/sys_master/root/8818448859166/438725-ESCON-Module-50-5-Hardware-Reference-En.pdf

Encoder:
Encoder HEDL 5540 500 CPT, 3 Channels, with Line Driver RS 422.
http://www.maxonmotor.nl/medias/sys_master/root/8821074427934/16-401-402-403-404-EN.pdf

Base Controller:
The RaMstix is used as it can work with 20-sim. It also has Encoder, PWM and Digital I/O pins. It works on 50Hz but can be changed to the 500Hz the controller was designed for by changing its code. It needs to be setup to work as target for 20-sim 4C, the steps to do so can be found in its documentation:
RaMstix FPGA board documentation > Target setup for 20sim > Prepare 20sim 4C with RaMstix toolchain and target.
https://www.ram.ewi.utwente.nl/ECSSoftware/RaMstix/docs/index.html

C Setup Connections

The motors, the encoders, the motor controllers and the RaMstix are connected following the map shown below in figure 27. The wire colors are as represented, gray replaces white for visibility, orange represents "not connected", connections to ground were not drawn for readability purposes even though every ground from the encoder and the controller is connected to a ground on the RaMstix. See Appendix B for more information about the components and their connections.

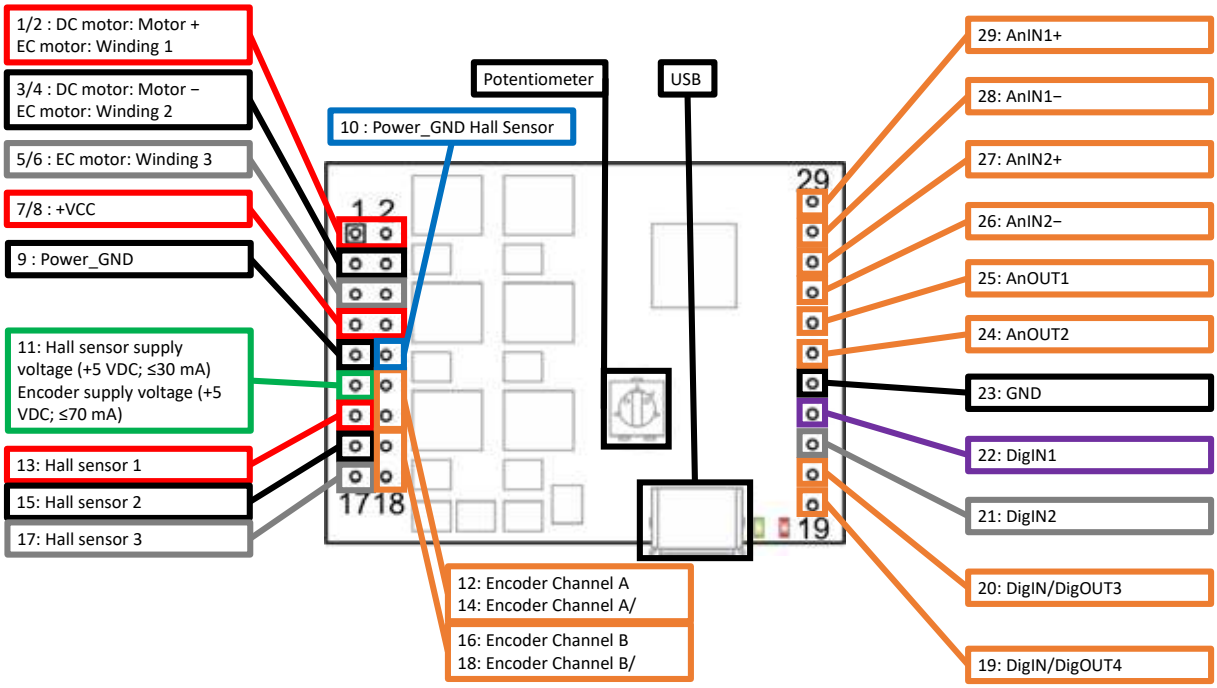

Figure 26: The map of the connections of the motor controller.

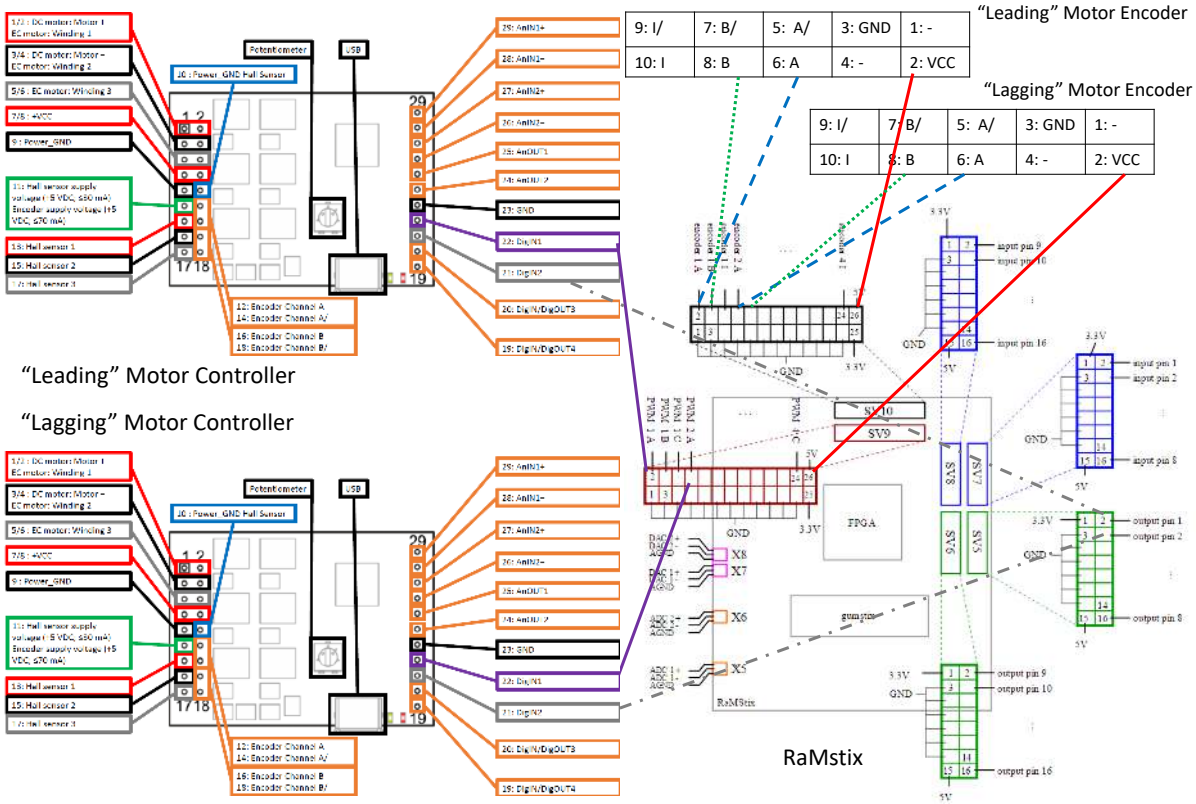

Figure 27: The map of the connections of the setup.

D Escon Studio: Programming the motor controllers

Escon studio is necessary to have the motor controllers working. They only need to be programmed once but each controllers has to be done separately. When programming it, it has to be connected to the motors, a power source and a laptop running the software. The steps are as follows:

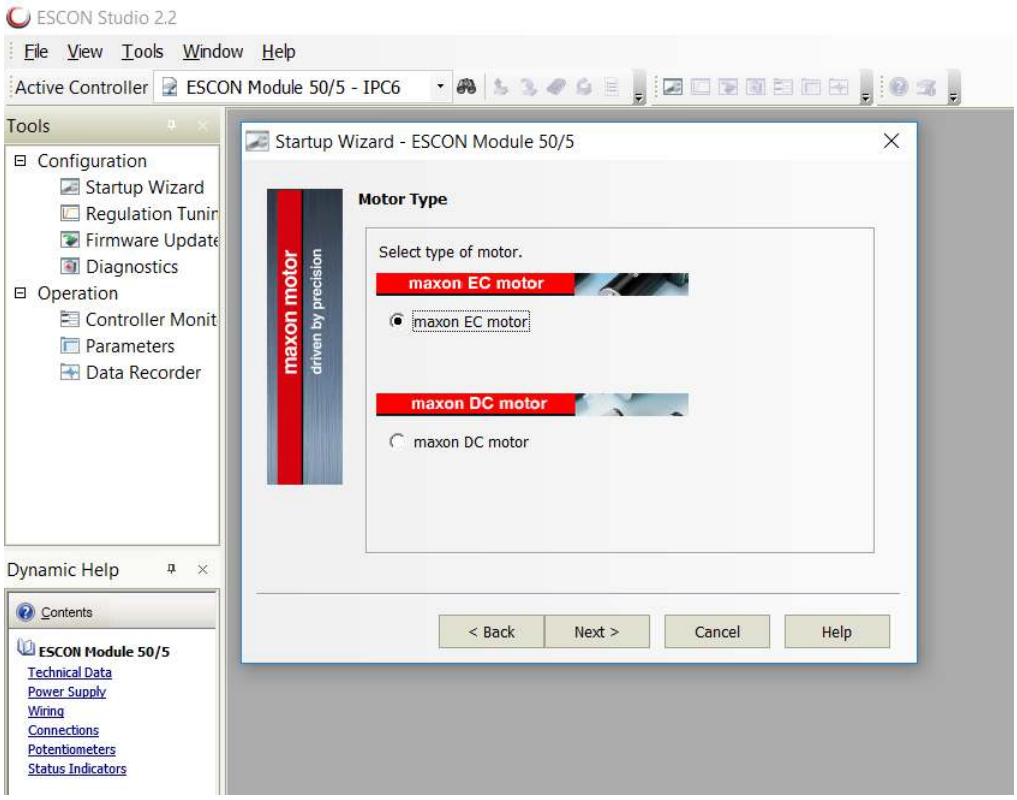

Figure 28: Step 1: When started, if the startup wizard isn't open yet, open manually. Choose for maxon EC motor.

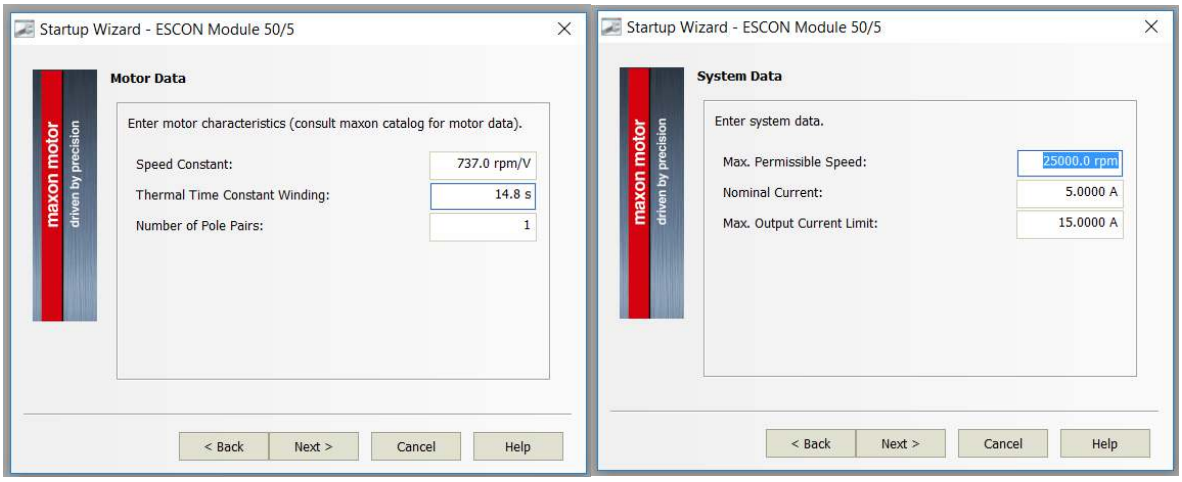

Figure 29: Step 2: Choose the motor characteristics. The ones used here can be found in the equipment appendix, the link of the motor, number 18888.

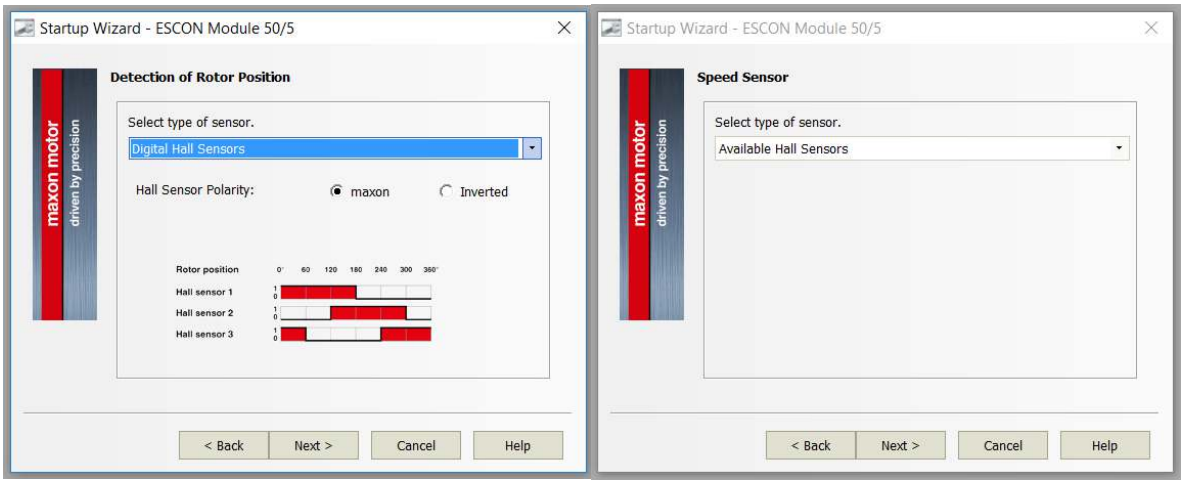

Figure 30: Step 3: The EC motor are equipped with a digital hall sensor which has to be selected.

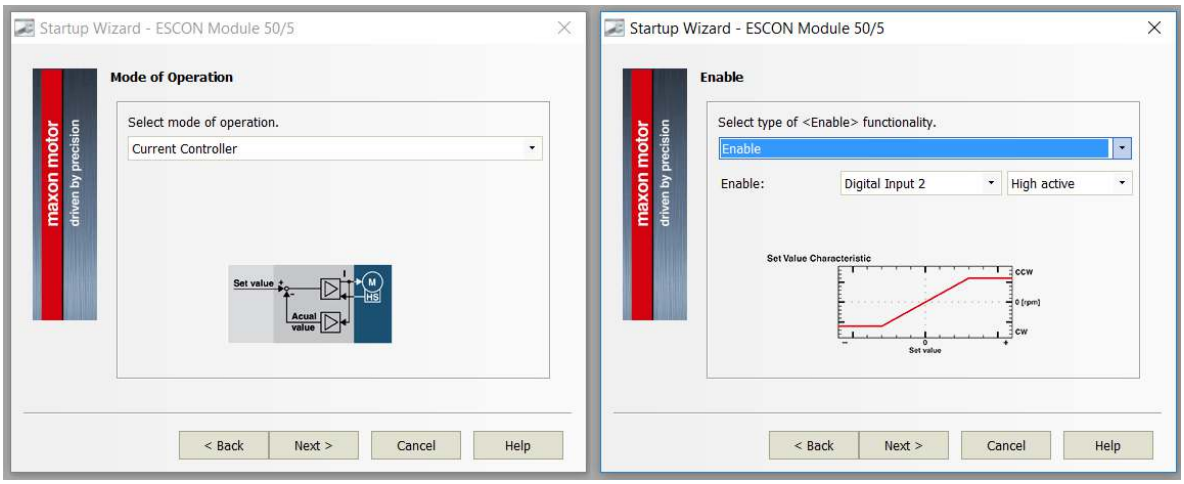

Figure 31: Step 4: A current controller is used here so it needs to be selected. The enable function must be used, it will be used through the digital input 2.

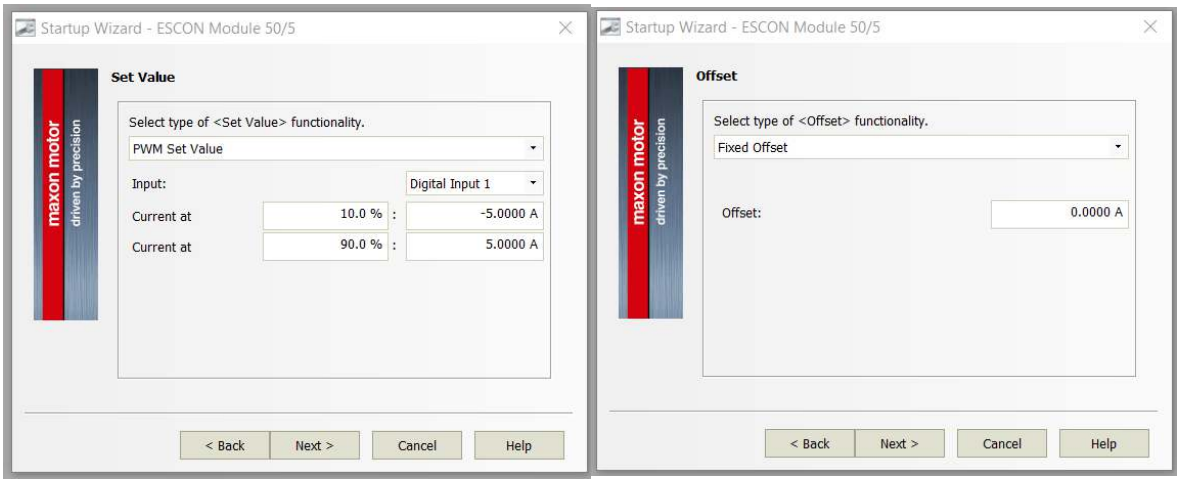

Figure 32: Step 5: A PWM with set value is used, as it is limited between 10% and 90%, the output of the RaMstix will match it. This will be connected to the digital input 1. Also no offset will be applied.

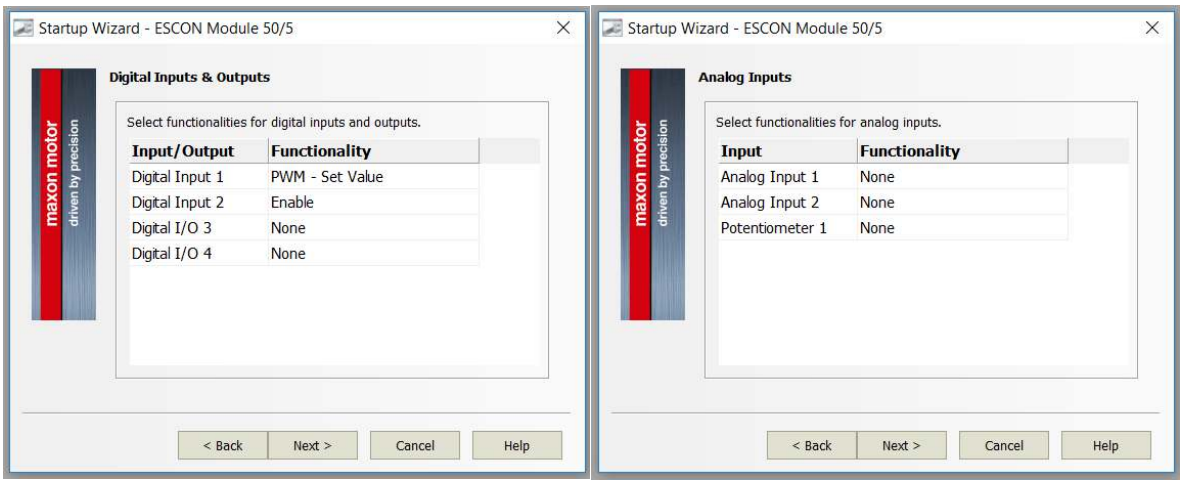

Figure 33: Step 6: Review the inputs and outputs of the motor controller, DI1 must be connected to PWM, DI2 to Enable, and all the rest of the connection must not be connected.

Figure 34: Step 7: Regulation tuning will have to be opened. Finish if everything is correct.

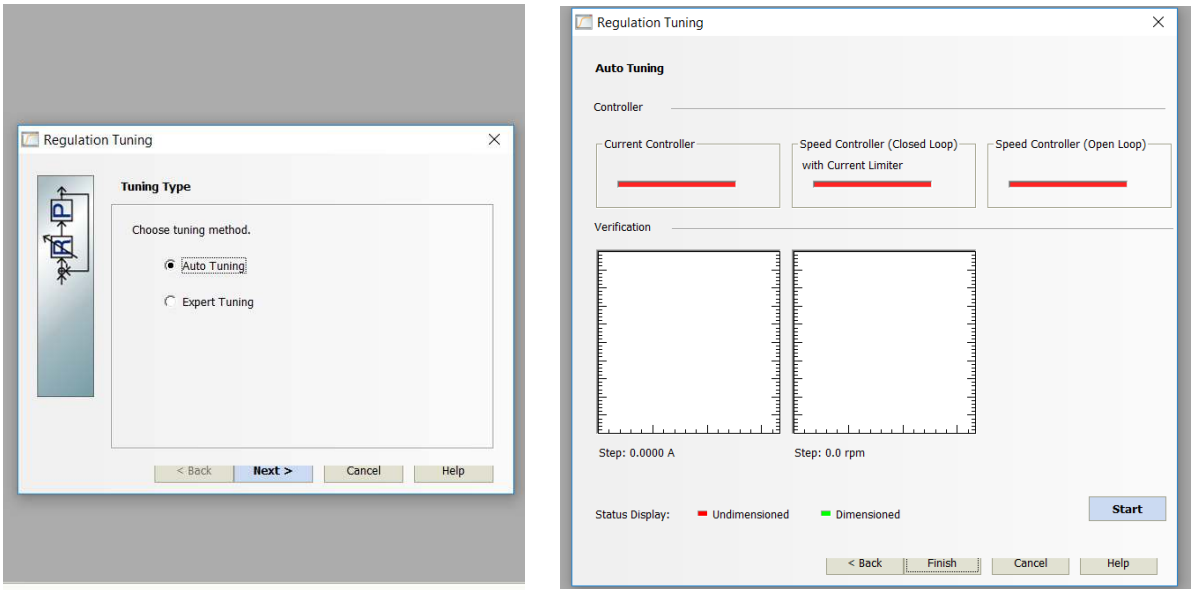

Figure 35: Step 8: Select and start the Auto Tuning when there are no disturbances at the motor, this includes the wing but excludes the gears connecting it to the setup.

E 20-sim 4C: Setup

20-sim 4C needs to run continuously in order to do the measurements. The steps in order to have the controller running in the RaMstix from 20-sim 4C can be seen below:

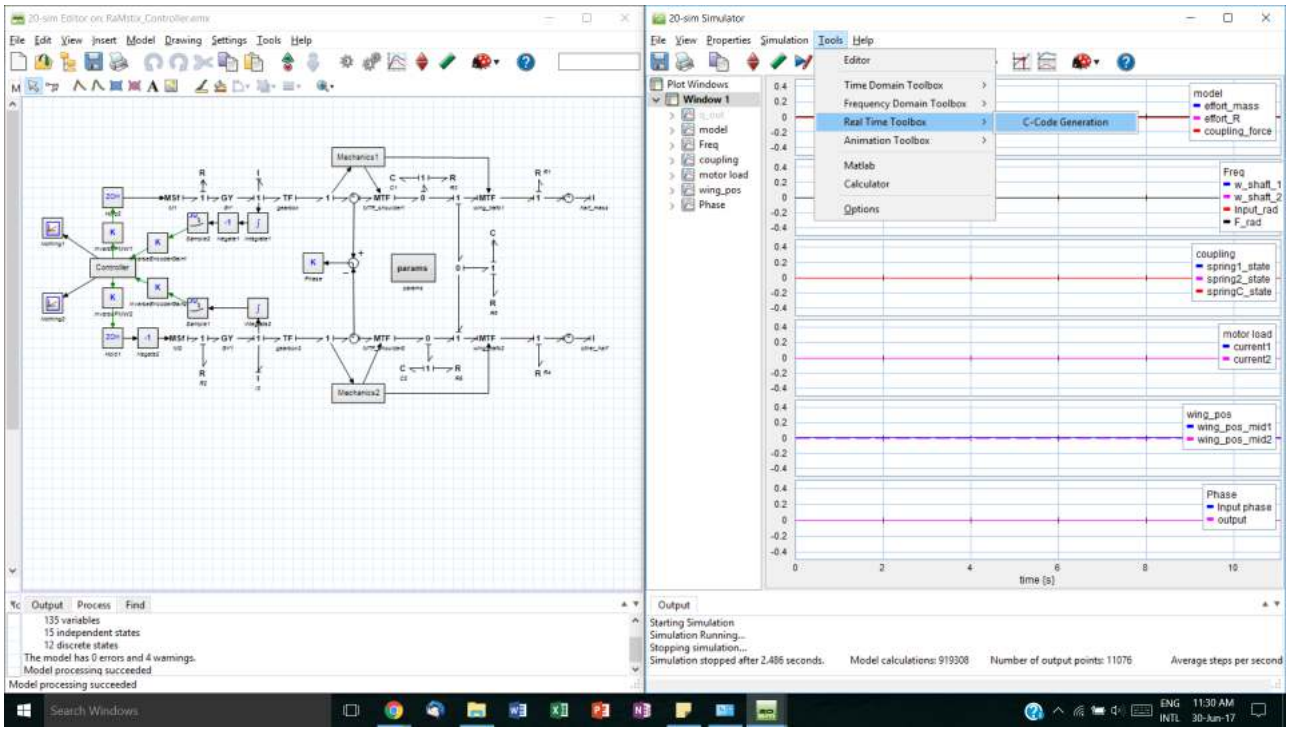

Figure 36: Step 1: 4C has to be opened through 20-sim. This is done through the simulation, by using the c-code generation in the real time toolbox.

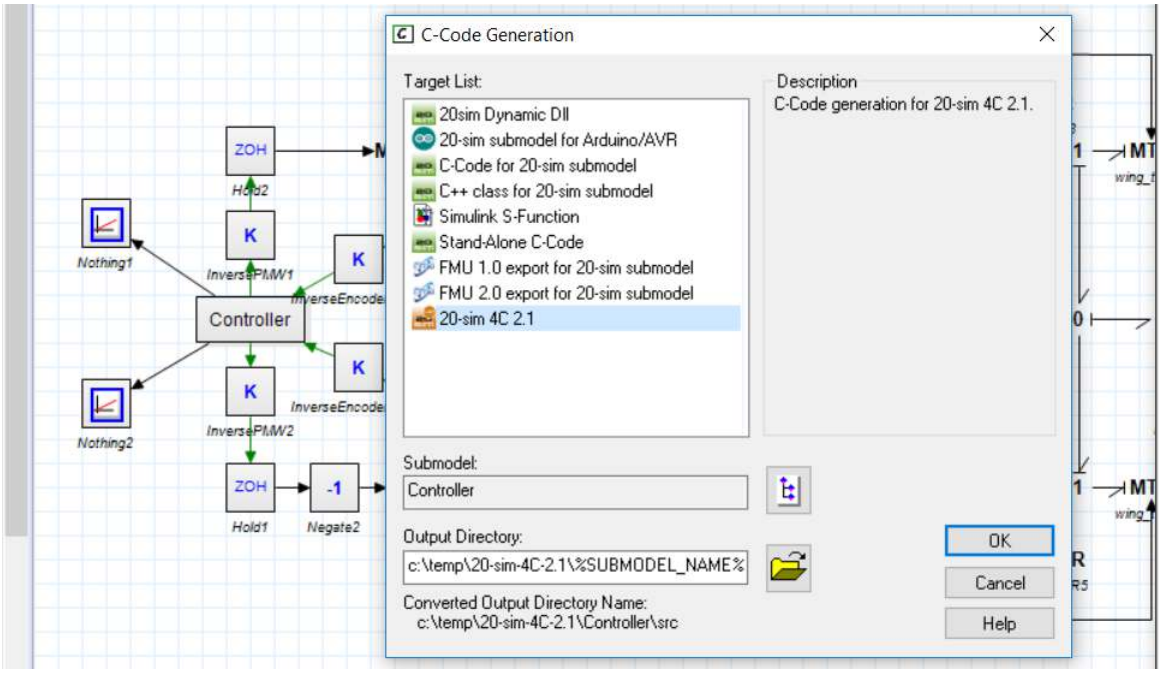

Figure 37: Step 2: Using 20-sim 4C as the target, the submodel containing everything for the controller has to be selected.

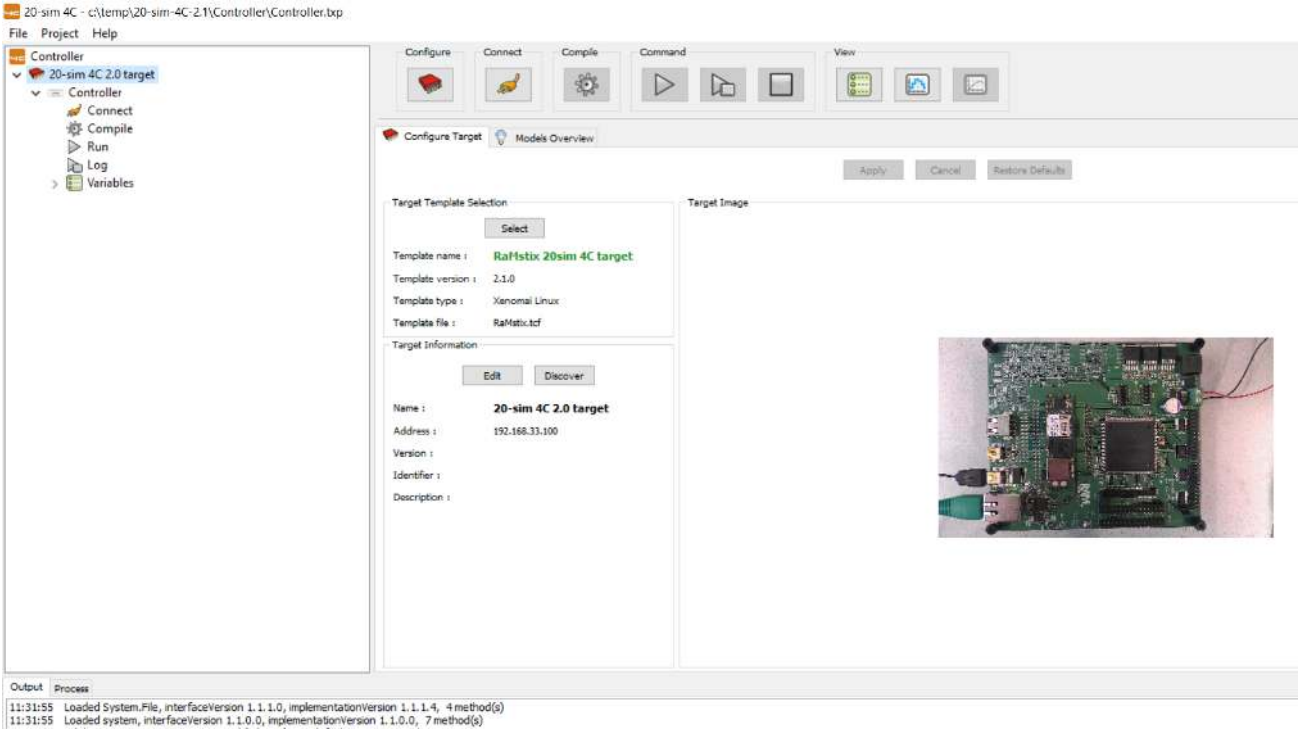

Figure 38: Step 3: The RaMstix with the correct IP address must be selected. For that the RaMstix must have already been configured in 20-sim 4C. This can be viewed in the equipment appendix.

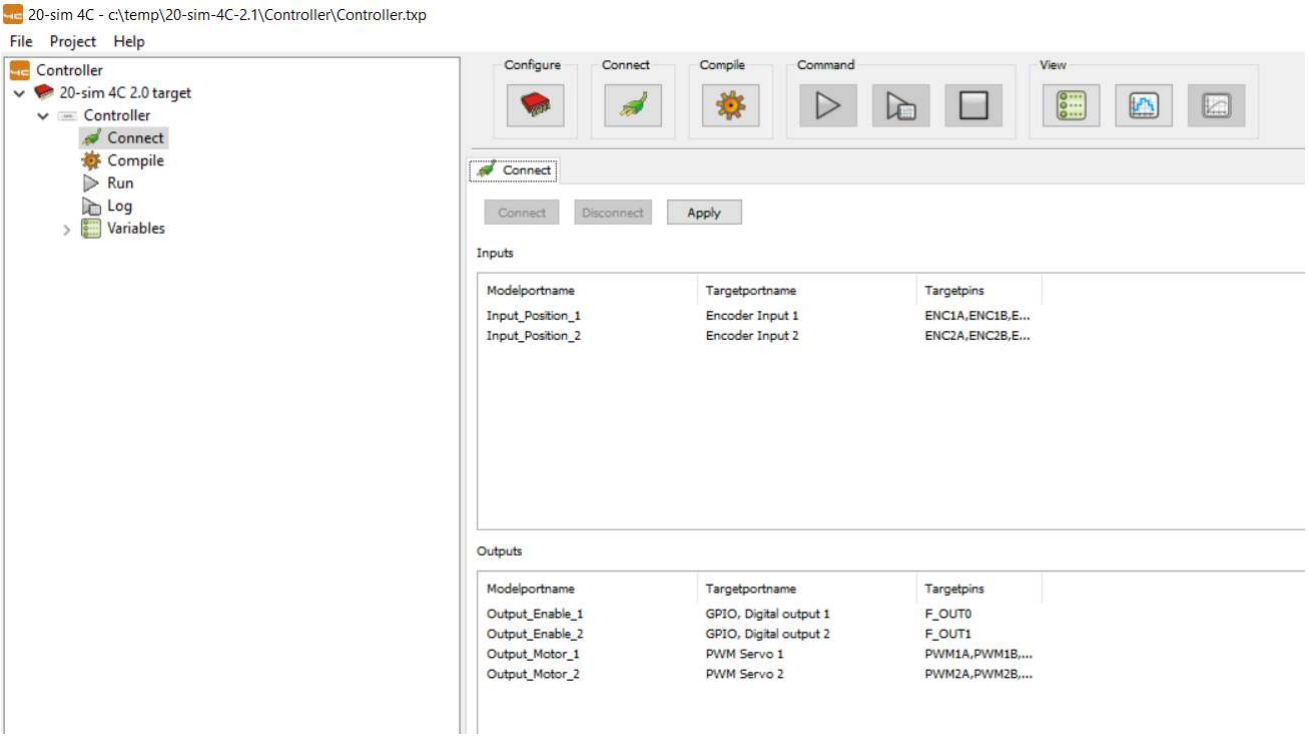

Figure 39: Step 4: The inputs and outputs must be connected. They follow their numbering, the encoder is a quadrature counter noindex, the enable is a GPIO digital output and the motor is a PWM servo.

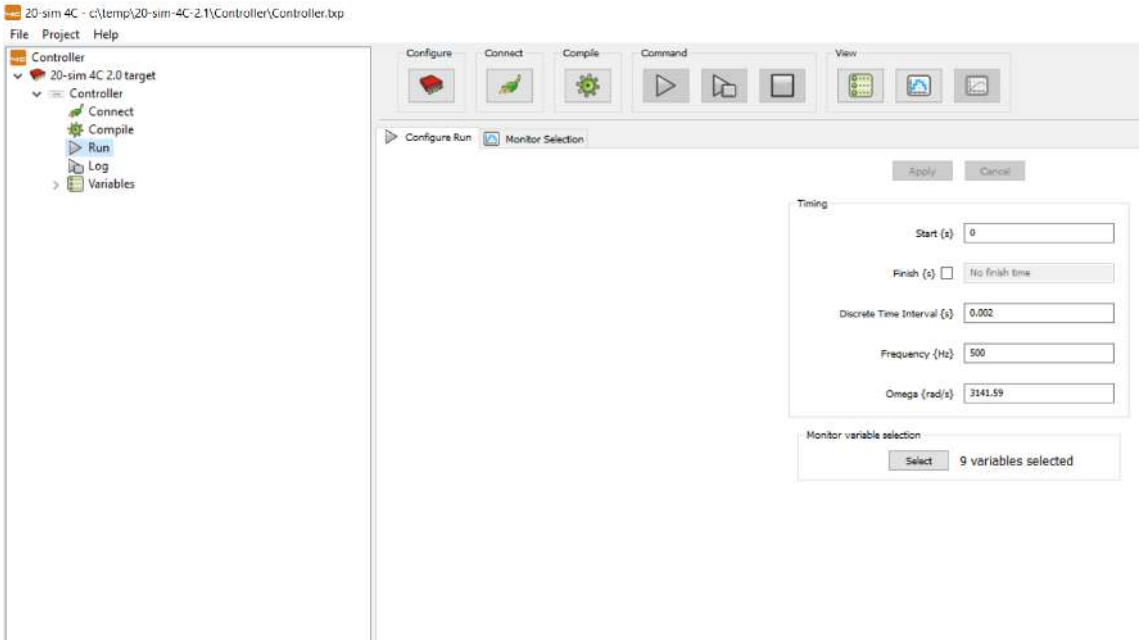

Figure 40: Step 5: Run must be configured without finish time as it can be stopped at anytime.The frequency must be at 500 Hz as this is the value at which the controller and setup were designed.

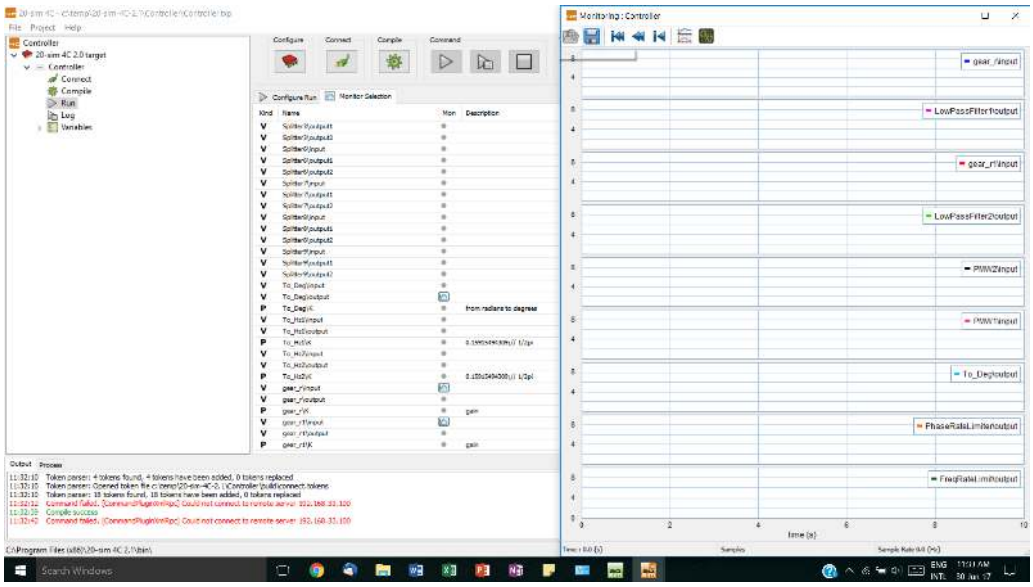

Figure 41: Step 6: That parameters that need to be measured can be selected at the monitor selection screen. After the run, they can be saved if no new run has been started or the values were erased from the monitor.

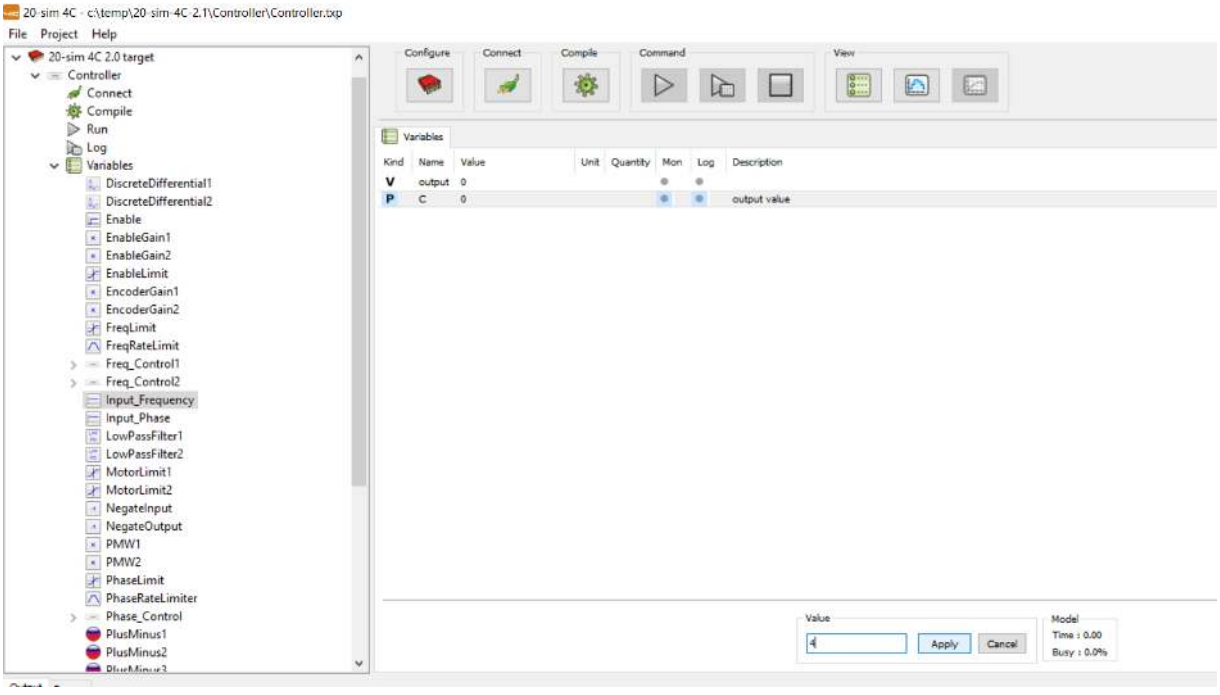

Figure 42: Step 7: While running, the values can be changed under variables. This is how the frequency and phase shift of the system can be changed and how it can be enable/disables.

References

[1] F.B.W. Berndsen. Development of a port-based modular simulation model of the robird using an iterative approach, November 2016.

[2] A. E.-A. S. S. Desoky. A review of bird control methods at airports. *Global Journal of Science Frontier Research*, 14(2), 2014.

[3] G. A. Folkertsma, W. Straatman, N. Nijenhuis, C. H. Venner, and S. Stramigioli. Robird: a robotic bird of prey, 2016.

[4] S. Heathcote, Z. Wang, and I. Gursul. Effect of spanwise flexibility on flapping wing propulsion. *Journal of Fluids and Structures*, 24(2):183 – 199, 2008.

[5] J.L. Mulder. Towards the understanding of flapping wing propulsion, December 2013.

[6] B. van Dijk. This robird is a 3d-printed drone that looks like a falcon, October 2015.

[7] C.S.E. Vaseur. Robird wind tunnel test setup design, November 2014.