Foucault Pendulum Electronics Kit.

D11_Initial Tests and Known Problems

www.foucaultpendulum.nl

Document version	2026-07-02
Related Documents	Electronic Schema's D05_Description of Electronics. Description of the Software

In brief:

Test Equipment:
Oscilloscope 10+ MHz, with at least 2 channels, 10x attenuator probes.
The scope can be analog or digital.
Multimeter with at least the functions Volts, Ohms, Diode Test.

BobControl Board.
After assembly and before placing the IC's, Arduino and Ethernet Shield and DDS module check the supply voltages.
Check the 24V power supply that it delivers 24V +/- 0.5 V.
Connect the 24V power supply to the board.
Check that there is + 5V +/- 0.5 V on the Arduino 5V connection (above pins 22/23)
Check that there is - 5V +/- 0.5 V on pin 11 of the IC1 and IC2 sockets.
Check that there is +24V +/- 0.5 V on pin 4 of the IC1 and IC2 sockets.
The green leds +24, +5 and -5 should be ON.

Now place the Arduino MEGA, the Ethernet Shield and the DDS module.
Upload the firmware into the Arduino (if not done before). Connect the ethernet cable, and start the GUI_Foucault on the work PC. Check the checkbox Talk and verify that the communication works.

Start up info.
On startup the Arduino produces an amount of information on the Arduino Monitor, when the USB cable is connected to a PC with the Arduino IDE running.
Press: *ctr-shift-m.* (this also causes an Arduino Reset)
A.o. Version info, several parameter settings read from EEPROM, and the sucess of initializing several functions.
Check that no errors are reported during the startup.
During *setup()* also a led test is done for the 8 function leds. Currently Option Jumper 1 (the top one) is assigned to skip this test.
At the very first start no info is available in the EEPROM of the Arduino. The values default to 65536 (hex FFFF). The information is written when (new) parameters are entered from the GUI with the button "Change Parameters" in the General field.

Monitor Commands.
The Arduino reacts on certain letters and digits entered in the top field of the monitor window. Enter "h" or "t" for an overview. (case sensitive)

Working DDS.
With the oscilloscope check the 465 kHz signal to be present on Arduino pin 47. There should be a 5V squarewave of 465 kHz, (2.1 usec period time)

465 kHz Wire Driver.

With the oscilloscope on TP-Wire check the presence of the 465 kHz signal. There should be a squarewave signal with slow slopes. Check that the amplitude can be adjusted with R19, up to clipping at 24V peak to peak. Adjust it to ca. 8 Vollt peak-peak.

PWM for drive strength.
With the oscilloscope on MEGA pin 6 check that there is a squarewave signal of 5V with a frequency of around 15 kHz. The duty cycle of this signal should change with checking the checkboxes Force Max and Force Min in the Drive Parameter pane, or by changing the values of Max and Min, followed by pressing the Change Parms button in the pane General. The higher the value chosen the higher the average level of the PWM signal. With the other channel of the oscilloscope on pin 3 of IC1 or hooked to the lower side of R16 one can see a DC voltage varying with the value chosen, from 0 to +5 Volt.

Check Drive pulses.
Place IC 1. Connect your drivecoil, or a resistor of a few Ohms to the DRV terminal block. (I used a 15 cm 8 Ohm loudspeaker for the initial test, then you can hear it too)
When no signals from the receiver board or a coil set are connected the firmware generates drive pulses at the rate determined by the Missed Center parameter and the choise made for Sync Center_Mag or Center_Cap.
Connect one channel of the oscilloscope to the tab of Q3 and another cannel to TP_I-DRV. On Q3 expect a pulse going from 24V down several volts and on TP_I_DRV a pulse with an amplitude the same as the DC level at the lower side of R16. A similar pulse can be seen on TP_DRV. You should make some calculations about the voltages to expect.
The current delivered to the drive coil is the voltage on TP_DRV / the combination of R32 and R35. The voltage drop you may expect on the Q3 tab is that current * the resistance of the drive coil. There may be the situation that the voltage drop is larger than 24Volt minus the voltage on TP_I_DRV, and the transistor Q3 gets saturated. As long as that is the case we have no current control, the circuil behaves as a voltage source. This situation is not detected by the firmware.

Fig. 1 Scope image of this test:
Yellow: The /DRV signal at Mega pin 41.
Blue: The signal at TP_I_DRV, across the current sensing resistors R32,35
Green : The signal at diagnose pin A15 showing the 20 kHz signal. We can see that the puls lasted 10 ticks, which corresponds to the setting at the time of taking this picture.
Red : The signal at the tab of transistor Q3. We can see that it is going down ca. 9 Volt during the pulse. The overshoot at the start and ending of the pulse are due to the inductance of the coil. This is a normal phenomenon.

Known problems.

Long Pendulums.
Not a real problem but it may need attention.
If your pendulum is over 10 meters long the period time becomes so long that integer overflow may occur in many 16-bit unsigned integers.
The remedy is either to reduce the sample frequency from 20 kHz to e.g. 10 kHz, or to replace many unsigned 16-bit integers by unsigned 32-bit integers. The latter is a lot of work, and has the risk that the increased processing time in the interrupt handler becomes so long that interrupts will be missed.

PCB BobControl, bad contacts
I have noticed a few times that the contact between some pins on the BobControl PCB and the Arduino Mega socket was bad. The pins could not get deep enough into the sockets.
In one case I could blame the 6 pin SPI Passthrough socket for this and I could repair it by carefully lifting the seat of these pins from the Arduino board with a hooked tweezers. Visible on the photo in Assembly Instructions. After this all pins could go a bit deeper into the sockets.
Most likely the problem was related to my source for Arduino's: Cheap Chinese Clones.

Clock Time in Pascal Programs.
If the system time changes, e.g. due to a switch in summer/wintertime, the pascal executable needed to be restarted before the new time was taken over.
This problem has been solved by placing *ReReadLocalTime* in the handler of *TimerOneSecond*. This should work for unix based systems. MSW systems may need another fix.

Clicking on the Main Form, outside any control clears the graphs.
This is a phenomenon which is now solved. In the older versions the plotting was done on the form.canvas. In the latest version plotting is done on a tImage.Picture.Bitmap.canvas, which is not affected.

After some 13 days of operation the GUI stops working and disappears.
After such a stop it can be restarted without any problem, but some data are missing in the logfiles.
For this I made a days-of-operation counter in the Communication pane on the GUI. When the critical day comes one can stop-and-restart the GUI at a convenient moment.
This behavior is platform dependent. After a change to a 64-bit laptop I have not seen it anymore. The "Critical Day" counter ihas been removed from the GUI.

Incorrect dimensions of groupboxes and graphs on the GUI, when the program runs on certain laptops or PC's.
This problem is related to the Pixels-Per-Inch (PPI) property of the screen. There can be a conflict due to the original scaling.
Lazarus 4.8 (most recent) handles this differently from older versions.
The program was developed on a desktop PC with Linux Mint 21 and a PPI of 96. On computers which have a different PPI the OS tries to scale the dimensions of controls to new values. This may not always work well. I encountered this problem on a laptop with a PPI of 115. For the dimensioning of the Groupboxes the problem could be solved by not explicitly programming these values, only set them by mouse actions during design time. For the graphics, Time display and Compass display, there might still be a problem. In that case experimenting with the screen resolution setting may help. In the units u_scope and u_compass certain scaling is explicitely done on the basis of the property *Screen.PixelsPerInch*.
In the module GUI_Pendulum.pas the property Application.Scaled may be turned off or on (default). There may be need to experiment with these values.