Building a spare parts LP12 clone

Bjorn Hugsted

August 1, 2021

Contents

1 Abstract

We describe the design and implementation of a Linn Sondek [Linn Sondek] LP12 copy built from leftover parts after upgrades of the original. The build consist of the turntable itself as well as a new principles synthesized power supply.

2 Introduction

The LP12 is, according to Linn, the pinnacle of music reproduction. In its fully upgrade state - that is. Following this road leaves one with spare parts that is fully operational, and could be reused for a spare part turntable. After several upgrades we had a subchassis, the main bearing, the platter and one AC motor avaliable.

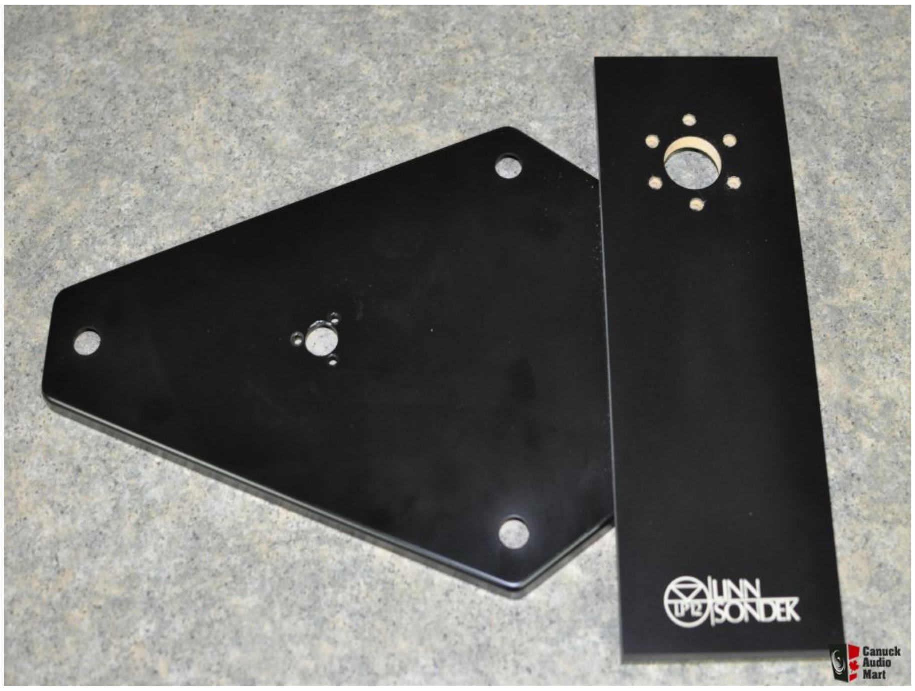


Figure 1: Linn Sondek subchassis and armboard. Curtesy Canuk Audio Mart

3 Rebuilding the subchassis

We always thought that the balance of the subchassis was disturbed by the asymmetric mounting of the tonearm. The original layout of the subchassis and armboard assembly is shown in Figure 1. With the armbase in the uppper left any upwards or downwards motion will make the subchassis wiggle and possibly also introducing some flexing of the wooden armboard. Later development of the LP12 made the subchassis and armboard into a single part. Our solution is to move the armbase to a position on the line that runs through the leftmost support spring position and the platter bearing position. The armbase will then have to be moved rightwards and this may actually create a better balance of the platter and the armbase around the upper and lower spring positions. Still any sidways motion will introduce wiggling as the whole subchassis and armboard assembly is top heavy. Nevertheless we do not think any further rebuild will be practical. Figure 2 shows the Spart Part LP12 from above. We see that with this configuration the left spring position, the main bearing, and the armbase are on one line. Also the two rightmost spring positions are between the main bearing and the armbase.

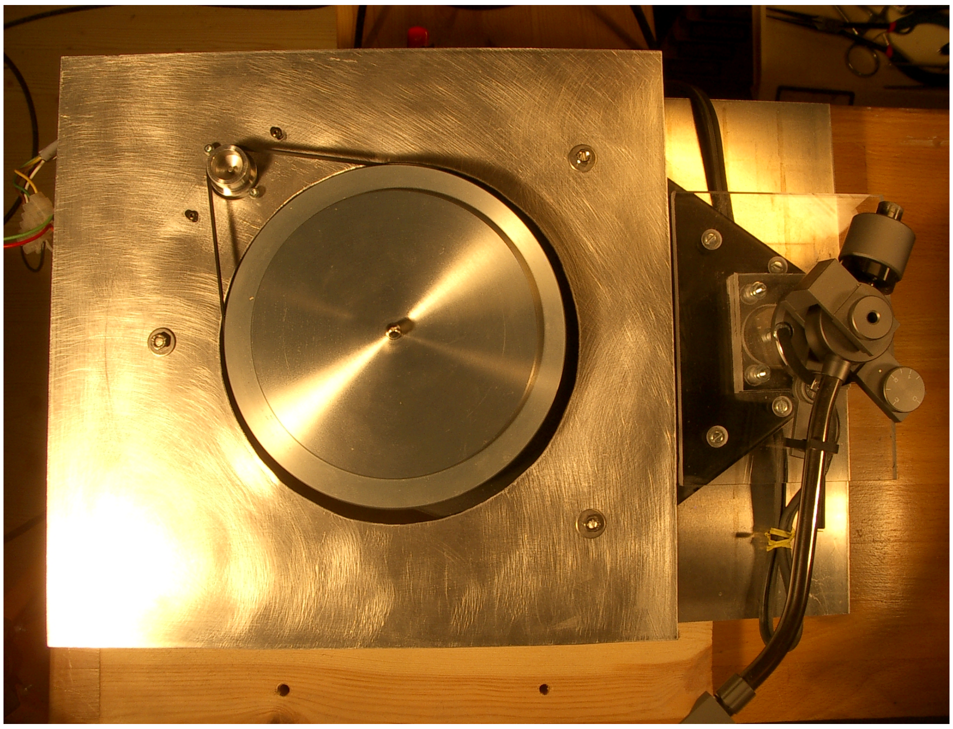


Figure 2: Spare Part LP12 from above with platter removed

3.1 Mounting height of armbase

When we now mount the armbase directly to the subchassis we will need some distance spacer to somwhat increase the height. As the original tonearm mounting board, produced by Linn, is 10 mm this would be the most natural choice of the thickness of the spacer. We could consider a thicker spacer to create an even sturdier mount, but what maximum height we can allow is dictated by the need to lower the arm to accomondate low pickups. Here the height of the pickup itself is the vertical distance from the stylus tip to the upper surface.

From information at Vinylengine [Vinyl E.] and own measurements the heigh of pickups vary from 15 mm to 20.5 mm, but this is from a selection of only nine pickups and pickup families. The pickup we use at the Spare-Part-LP12 is an AT95E with a height of 17 mm. Actually the 15 mm was an old Rega Bias and all the rest is over 17 mm. Currently we use two perspex plates, each of thickness 8 mm between the subchassis and the armbase.

We use this armboard thickness with the AT95E that has a height of 17 mm. We also use the original Linn rubber mat and an ordinary Long Playing (LP) record. To get the armwand parallel to the record surface we need to raise the arm piller about 5 mm above the lowest position. We should therefore

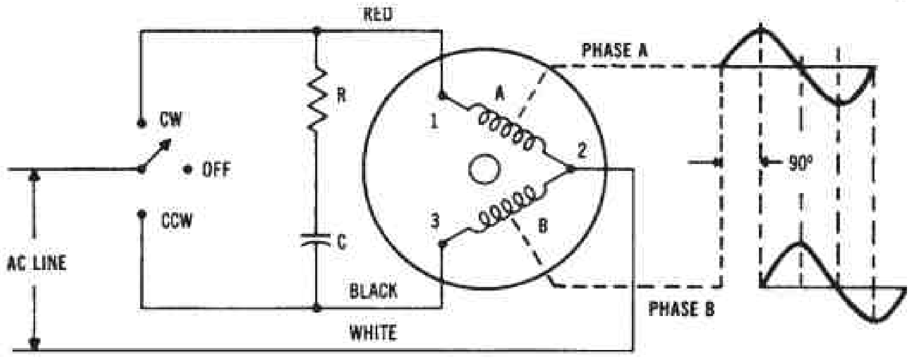


Figure 3: Principle of two winding PMSM

be able to mount pickups with height as low as 12 mm. There is also about 5 mm distance between the arm cable connector and the lower part of the subchassis. We may therefore still be able to raise the arm piller about 5 mm to allow for pickups of height up to 22 mm.

4 The Alternating Current synchronous motor

This is the original Linn motor that is a Permanent Magnet Synchronous Motor (PMSM) with two different stator windings giving a total pole count of 24. These motors are made to run synchronously with the frequency of the mains power grid. The windings are fed with AC current from same frequency source, but with the phase displaced 90°. The motor control and connection to the two windings are shown in Figure 3. Here we also see how the direction of rotation can be changed. Direction of rotation is fixed in the LP12. The speed of rotation is 250 r.p.m when feed with 50 Hz voltage. On 60 Hz mains the rotational speed is 300 r.p.m.

The Linn motor has four leads red, blue, and two grey. The four wires are pairwise connected to the motor windings. The red and one of the grey connects to one of the windings as do the blue and the other grey to the other winding. There is no electrical connection between the pairs. The DC resistance of one winding is about 4.1 kΩ For more measurements on this motor see Section 5.2 below.

To my best knowledge the motor is manufactured by Premotec [Premotec]. The motor I have at hand is the 9904 111 04802 while I belive Linn currently supplies the 9904 111 31813 that is manufactured to tighter tolerances that again minimizes vibrations. The cost of a 31813 is about $ 70 from an electronics dealer, but this is without the pulley. On eBay a used Linn motor with pulley was offered at about $ 80, but also as high as $ 180, both

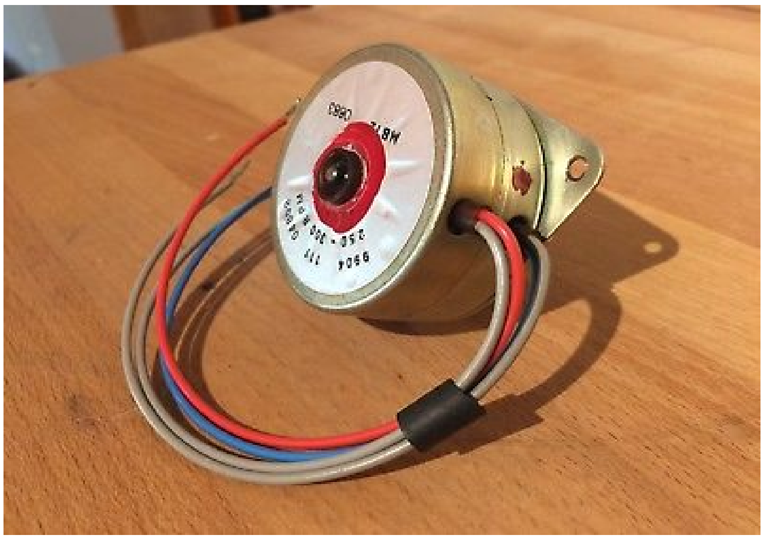


Figure 4: Linn Sondek LP12 AC motor

from delalers in the UK. Even when the 31813 is a better motor I will not take that cost.

5 Two phase synchronous motor power supplies

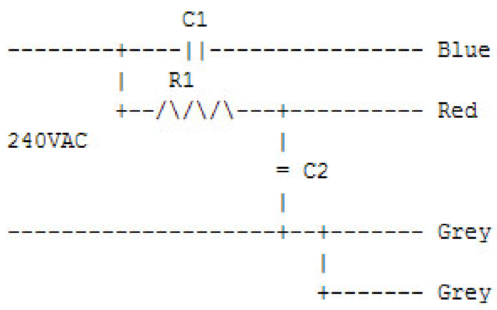
The motor I have at hand is rated 110 VAC and the power supply must deliver two alternating voltage feeds with same frequency, but with a phase displacement of 90°. Several possible mechanisms to accompish this exists and several has alredy been implemented in readily avaliable power supplies from more manufacturers.

5.1 The original Linn power supply

The original supply found in very old LP12s is a pretty simple construction. The schematics are shown in Figure 5. The supply delivers the mains voltage trough a capacitor to the blue motor wire and through a 20 kΩ resistor to the red wire. We also see that there is another capacitor to the gray wires that are connected together.

5.2 Armageddon and clones supplies

As a reaction to the frequence synthesized Linn Valhalla Naim [Naim] came up with a very simple, but sturdy power supply, viz. the Armageddon. This supply does the same as the basic Linn supply, but uses a very large power



R1 = 20k ohms C1 = 0.1uF, C2= 0.1uF, Vout is about 75V

Figure 5: Linn Basic motor power supply

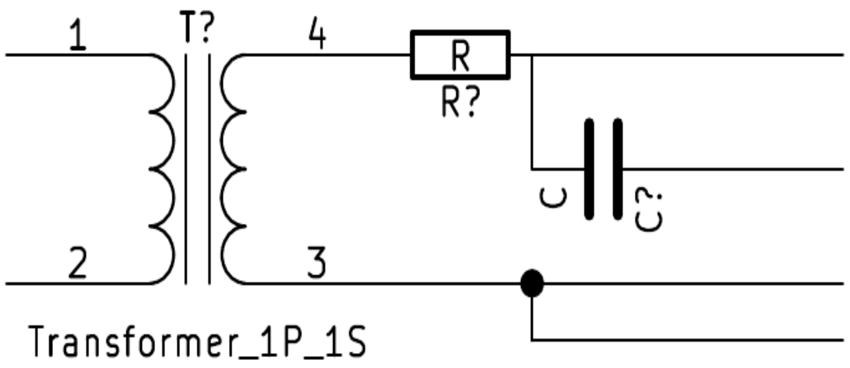


Figure 6: Mini Armageddon schematic

transformer in front of the phase shift network. The supposed action of the transformer is to act as a filter and remove spikes and distortion from the mains supply. Naim uses a 500 W transformer to power the network and the motor. Based on this power supply we built the Mini Armageddon with a somewhat smaller transformer. Instead of the 500 W transformer used by Naim we selected a 50 W Noratel [Noratel] transformer. The original Armageddon connects to the Linn AC motor using a Harting connector with 5 pins. On this connector only 3 pins are actually used for power supply. This will be discussed in section 6.

The schematic of this Mini Armageddon is shown in Figure 6 Measurements on our own Mini Armageddon show the voltage at the red lead beeing about 6 ms after the blue lead. The amplitude seems to be almost the same with

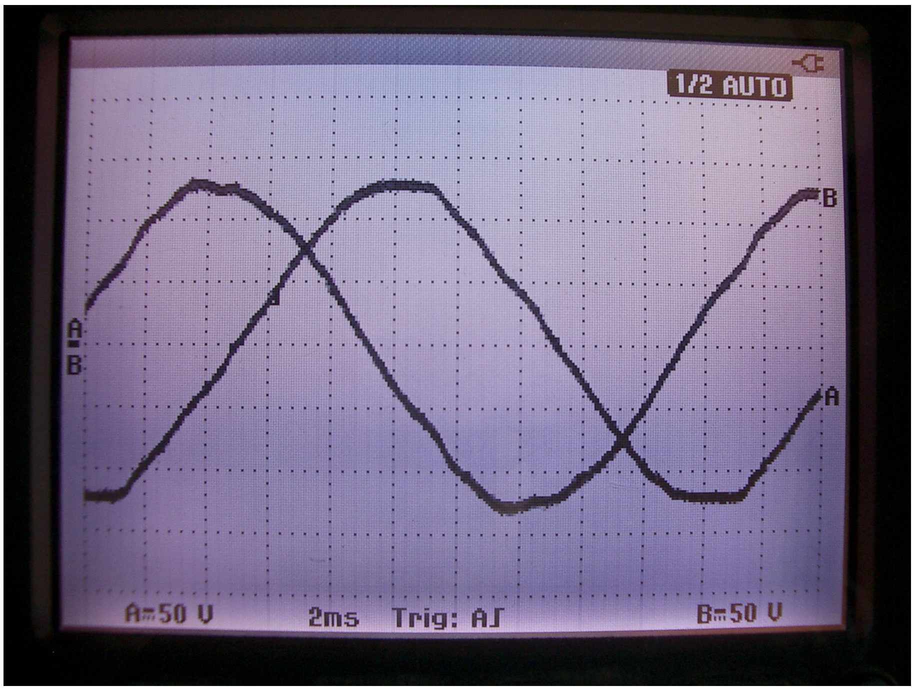


Figure 7: Motor winding voltages on red (ch A) and blue (ch B) wire

peaks of 125 V, i.e. 88 VAC. The scope screen is shown in Figure 7. While we were at it we connected an AC amperemeter in series with the blue lead. The reading was 8 mA. If the voltage and current is in phase the power consumed by this motor while turning the platter at 33 1/3 r.p.m. is about 1.4 W. Based on this the "resistance" for 50 hz AC current is 11 kΩ when running normally.

5.3 Frequency synthesized power supply

The beauty and the drawback of the basic, the Armageddon, and my own Mini Armageddon is the lack of frequency control. These supplies feeds the mains frequency straight through so that the platter always rotate at 33 1/3 rotations per minute. Linn themselves came up with the Valhalla and later the Lingo that are frequency synthesized power supplies where the frequency can be adjusted to let the platter rotate at 45 r.p.m as well. It is frequently reported that the Lingo in some way introduces disturbances on the mains supply so that the sound reproduction is detoriated, and this takes place as long as the Lingo is conncted to the mains supply.

6 LLingo Nouveau synthesized power supply

About the summer 2018 we came up with a possible mechanism for producing two voltage signals that may be suitable for driving a PMSM. Simply said we use a computer with a soundcard to produce two sinewaves of same frequency, but 90° phase difference. For a turntable designed to obtain 33 1/3 r.p.m. from 50 hz power the 45 r.p.m. speed will be obtained at $45 * 50/33.33333$, i.e. 67.5 Hz. At 50 Hz the two phases shall be displaced 5 ms, while at 67.5 Hz the displacement must be 3.7 ms.

The name LLingo is in no way connected to the Linn Products Limited, any Linn trademarks, or Linn copyrighted products. The name is actuall based on the LLingo bird that is widely known for its ability to hold a steady pitch in its song while still being able to change the pitch very fast and consistently.

The output from an ordinary soundcard has an amplitude of about 1 V as a maximum while the AC motor need up to 100 V to function. We may obtain this by using a two channel audio amplifier followed by two mains transformers connected with their secondaries to the amplifier and the primaries to the motor windings. From the measurements done in section 5.2 the amplifier will only have to deliver about 1.4 W total for steady running, but probably more during startup.

We will consider using an amplifier that can deliver 10 V peak. The RMS value when producing a sinewave will then be 7 V. We will have to use a transformer that delivers 100 V on the secondary for 7 V input on the primary. We assume that a mains transformer that produces 15 V for an input of 220 V may be used in reverse. For the amplifier itself it will have to deliver 0.7 W per channel at a voltage of 7 V, i.e. it will have to deliver 0.1 A on each channel. Said another way the load resistance seen by the amplifier is 70 Ω. Even when the resistance is much lower during startup just about any audio amplifer will suffice.

6.1 Dual 100 V transformer coupled amplifier

Our first attempt is built around a two channel amplifier obtained from eBay [eBay], more specific the dealer: "xiumeche-0". It is an "AC/DC 12V TDA7297 2x15W Digital Audio Amplifier" at a cost of \$ 2.75. As indicated this amplifier is built around the ST microelectronics TDA7297. The TDA7297 is a bridged amplifier with a floating output so neither of the output leads must ever be connected to power supply ground or to each other. We also used two transformers that could take 230 V to 12 V at 0.5 A. We connected these transformers in reverse and adjust the gain of the

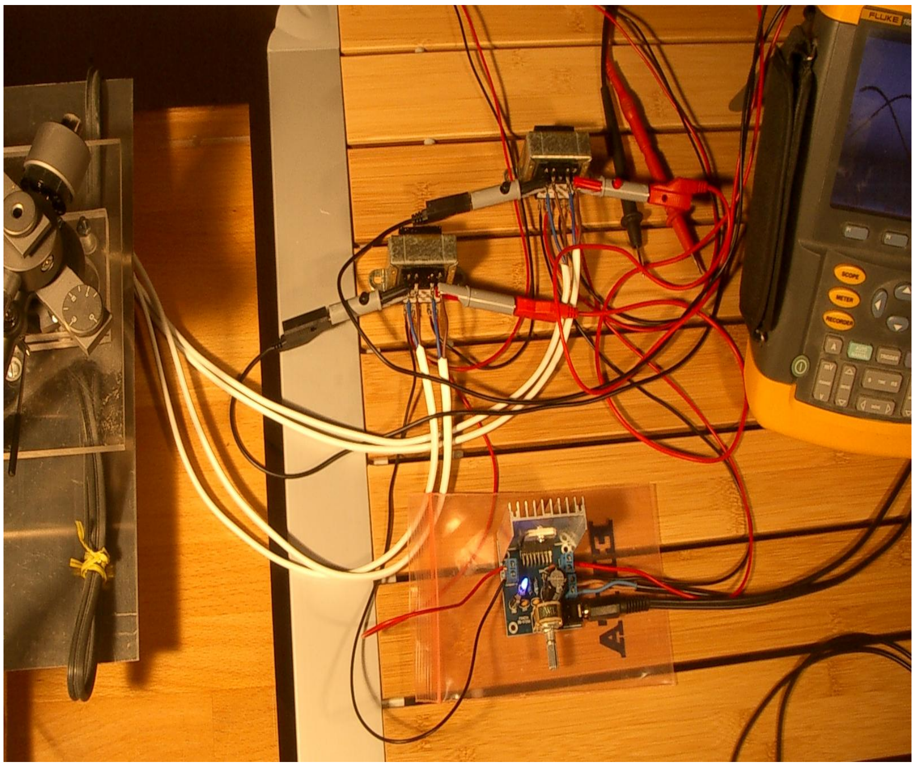


Figure 8: Two phase experimental setup

amplifier to obtain an output of around 100 V peak (70 V RMS). Actually there are more windings, but this is the configuration we used.

For experiments we used a laboratory power supply and set it for 12 V to power the amplifier. At that voltage the current draw was about 0.2 A. The amplifier gets quite hot when the turntable is running. It is still possible to hold, but an extra heat sink would not be out of the way. The experimental setup is shown in Figure 8. The el cheapo amplifier is in the lower middle and above are the two transformers. Part of the turntable is visible on the left as is also a glimpse of the Fluke 192B Scopemeter on the right. There are quite a bit of wiring, but this is a test setup. The waveform while running is shown on the oscilloscope screen in Figure 9. We may observe that the phase difference is spot on while the amplitude still may need some adjustment. Such adjustment is done in the software. The sinewave also appears to be somwhat cleaner than what was measured with the Mini Armageddon.

With this setup the audio left channel is amplified and connected to the left transformer (on the right side in Figure 8) that again is connected to the blue/grey motor winding pair. Same for the right audio channel that goes to right transformer and further to the red/grey motor winding pair. The oscilloscope measurement verifies this as the channel A is connected to the

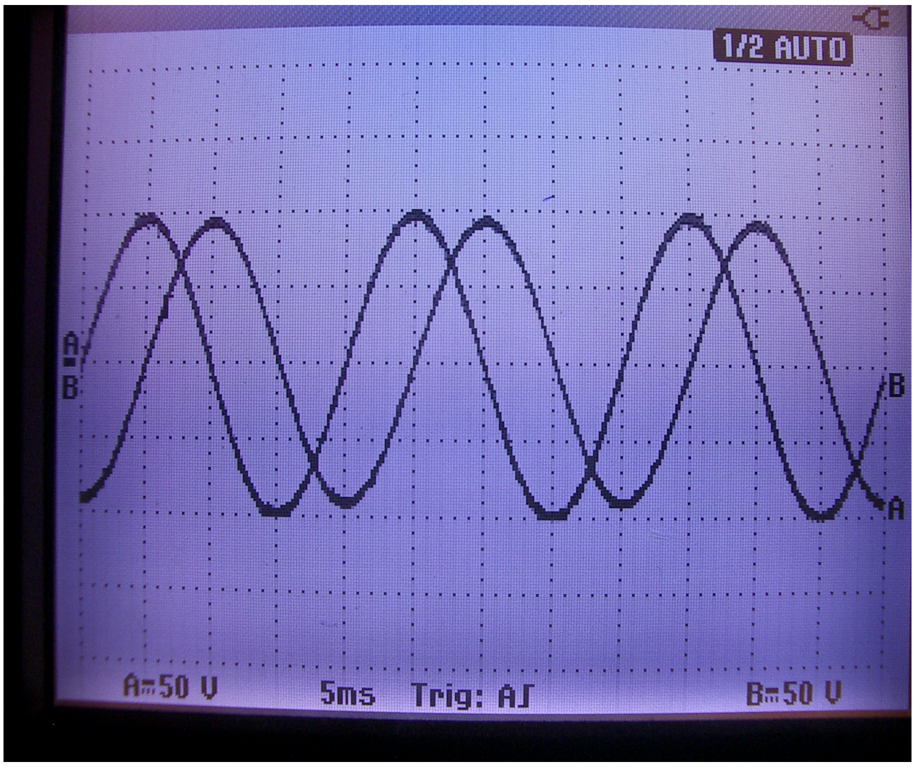


Figure 9: Oscilloscope screen during run

red/grey motor leads. The polarity is maintained by all the time connecting the live wire (brown, red) to the transformer connection marked 6 V on the low voltage side and to the terminal marked 115 V on the high voltage side. On each side the two windings are connected in series with the terminal marked 0 V directly to the terminal marked 115 V/6 V. The positive input of the scope is connected in the same way.

6.2 Connecting the 100 V amplifier to the LP12 motor

We selected to use the same connectors as used for the original Armageddon and also close to the same wiring scheme. This power supply connects to the LP12 AC motor through a cable fitted with a Harting han-kit 4AWK connector set, part 10 20 004 0004. This kit contains both the female panel mounted part as well as the cable connector and can be obtained at Elfa, Art 143-47-241, at about $ 30. The connections used by Naim for the Armageddon is to connect the blue motor wire to pin numer 1 of the connector and the red motor wire to pin number 3. The two grey wires are both connected to pin number 2. Wether or not any shields are connected is not known to us.

We mean that we can utilize the full width of the connectors by using all pins and our connection scheme is to connect the blue motor wire to pin 1, the red motor wire to pin 3, the grey wire of the blue/grey pair to pin 2, and the grey wire of the red/grey pair to pin 4, We will also connect the shield pin and let the turntable decide wether it shall be used for anything. Some earth connection schemes do introduce mains hum.

On the amplifier end we will then connect the free 115 V terminal of the left transformer to pin 1 on the Harting connector, the left transformer 0 V terminal connects to pin 2, the free 115 V terminal of the right transformer connects to pin pin 3 on the Harting, and finally the 0 V terminal of the right transformer connects to Harting pin 4. We also soldered together a connection cable with a female connector in one end and a male connector at the other end. This cable can be used for extending the component leads or in case the turntable is fitted with a male connector. We use a cable with brown, green, yellow, and white leads. The connections are shown in Table 1. In this table the notion "gray (blue)" means the grey lead of the blue/grey pair and so forth.

7 Software synthesizer

We have implemented the synthesizer itself as a C++ class in the files synth.h and synth.cpp that contains declarations and definitions of the Synth

100 V amp.	Harting 4AWK pin	Ext. cable color	LP12 motor
Left 115 V	1	yellow	Blue
Left 0 V	2	white	grey (blue)
Right 115 V	3	green	red
Right 0 V	4	brown	grey (red)

Table 1: Connections between amp. connctor plug and LP12 motor

class and a couple of helper classes and structs. Upon the Synth class we have written two control programs. The first, ttysynth.cpp allows for terminal control, while the other wbssynth.cpp is a web server that may be connected to by using the Websocket mechanism from Javascript running in the Web browser.

7.1 Building the synthesizer

We use the Advanced Packaging Tool (APT) to assert that all parts needed are installed

```
# apt update
# apt upgrade
# apt install libasound2-dev
# apt install libssl-dev
```

If the system is fairly new one may skip the upgrade. One also needs the cJSON [cJSON], but this may be installed simply by copying the cJSON.h and cJSON.c into the Synthesizer directory and compile with the ANSI C compiler (gcc).

The synthesizer programs also need building blocks from the src.dev collection, viz. the Websocket/wbssingle.h and .cpp and also the ReportInterface/report.h. These include files must be available during compilation, but need not be present for running the synthesizer programs. One also need gcc and g++ as well as make. When all is ready enter the Synthesizer directory and build ttysynth and wbssynth with ”make”.

7.2 Initial and preserved values

The synthesizer need some values for correct startup and function. In addition one may be interested in the total playing time. The latter may be used to keep track of the playing time of the cartridge needle. Such values are stored in two files, i.e. synth_values.txt and synth_times.txt. Both stored in the synthesizers current ditrectory. The values store settings for signal generation while the times store the run time.

7.2.1 Synth_values.txt

The include file synth.h contains initial values that are our best guesses to make the synthesizer work from start. The sound card to use is "default" with a sampling rate of 48000 Hz. For more explanation of how to select the card name see section A.1 on use of The Advanced Linux Sound Architecture (ALSA). After these initial values the synth will look for and read the file synth_values.txt in its current directory. This file has the form

```
default
48000
50 0.75 90 0.75 0
67.5 0.75 90 0.75 0
50 0.75 90 0.75 0
```

where the first line is the card name and the next line is the sampling rate. The three following lines relates to the speeds 33, 45, XX respectively and give the waveform frequency, the amplitude and phase of the left channel, and amplitude and phase for right channel.

In the file above the default sound output will be used at a rate of 48000 Hz. The frequency for playing at 33 r.p.m. is 50 Hz. The amplitude for the left channel is 0.75 of the maximum and the waveform is shifted 90°. Observe that the selected sampling rate may not be supported by the hardware. In such a case the ALSA selects the closest rate that is compatible with the other settings, i.e. 16 bit little endian, interleaved, stereo format. The synth will report the selections during startup - this report should be watched when testing on new hardware. If 48000 Hz is requested the report should be "Info: Exact rate = 48000/1 Hz".

7.2.2 Synth_times.txt

For keeping track of the time the synthesizer has been running the file synth_times.txt is updated each time the turntable is halted, and is read back when the synthisizer is started. This number is intended as a log of the playing time of the stylus. **The file must be found in the current directory**, but if this file cannot be found the total run time is set to zero and the synth_time is created and written whent he turntable is halted or synthesizer is terminated. During testing omne will probably run the syntheziser from the development directory. Then there will be several synth_times.txt around. The format of the synth_times.txt is very simple. It only contains one single line giving the total run time in hours, minutes, and seconds. As an example, if the synth_times.txt contains

```
123 44 55
```

The synthesizer has been active for 123 hours, 44 minutes, and 55 seconds total. OBSERVE that this is the time the synthesizer has been active, we have no way to tell wether a turntable is actually present or wether the cartridge needle is actually on the record.

7.3 Controlling the synthesizer

Using an ordinary computer as synthesizer does allow more mechanisms to be easily implemented. Viz. control through the serial port or through the computer network. The latter requires the synthesizer SBC to have either Ethernet or WiFi interface, but this is almost mandatory these days. Terminal access through the network is avaliable by using the Secure SHell (SSH), PuTTY, or similar mechanisms.

7.3.1 ttysynth

Is a simple program that starts the synthesizer with a command interface on the terminal. This must therefore be used either from a keyboard and screen direcetly on the synthesizer machine or throug a networked terminal emulator. For the latter we have been using SSH, but if a serial line is available one could consider PuTTY [PuTTY] The set of commands used during normal run are: ., 33, 45, XX. The single dot "." stops the synthesizer. The 33, 45, XX sets the speed. The SV command display the current values for signal generation, while the RT command display the total playing time of the synthesizer. The .! halts the turntable and exits the program.

In addition to the above mentioned commands there are a number of adjustment commands that are: +ra, -ra, +rp, -rp, +la, -la, +lp, -lp, and the WV command. + and minus means increase and decrease the amplitude "a" or phase "p". Working on "l" for left or "r" for right channel. So the command +ra will increase the amplitude of the right channel while the -lp will reduce the phase delay of the left channel. For adjustments one will most likely need a two channel oscilloscope. The "WV" command makes the synthesizer save the current setting to the file "synth_values.txt" in the synthesizers current directory. The "WV" command may only be used when the turntable is halted.

7.3.2 wbssynth

We decided to use Java Script Object Notation (JSON) for sending messages between the Browser and the server. The main reason is that JSON is very well implemented in JavaScript. For parsing and writing JSON from C++ code we uses the cJSON [cJSON] library. The cJSON is described as: "Ultralightweight JSON parser in ANSI C". The simplest way to embed cJSON in a project is to copy in the files cJSON.h and cJSON.c and compile with an ANSI C compiler. We are using the cJSON version 1.7.7 - the latest we found. The cJSON is used to parse JSON textual messages received from the controlling JaveScript running in the Web browser. When the wbssynth returns the reply for a command it will be with something like the JSON element:

```
{
    "send" : "LP12synth",
    "recv" : "LP12cntrl",
    "time" : "2018-08-30T21:01:01.172Z",
    "type" : "rply",
    "rply" : "OK"
}
```

We will see below that the form of this reply is quite similar to the received command message, but of type "rply". The synthesizer may also send reports that will be of type "rprt". The exact form of the messages is not determined yet - this is very much a work in progress.

7.3.3 LP12speed.html and LP12control.html

Based on the Websocket mechanism and the JSON formatting we have written two JavaScript programs and embedded them into hypetext code. The first is the LP12speed.html that allows control of the very basic functions viz. halt, speed 33, and speed 45. We have also written a more elaborate program that give access to all the functionality of the synthesizer - especially functions for calibrating the amplitude and phase of the output waveform. The latter program is included in the LP12control.html.

In JavaScript the JSON parse and stringify are included in the language. As an example the JavaScript object below represents the command to be sent to wbssynth for setting the speed 33.

```
var js_object= {
    send : "LP12ctrl",
    recv : "LP12synth",
    time : "2018-08-30T21:01:01.172Z",
    type : "cmnd",
    cmnd : "33"
}
```

When the JavaScript variable js_object contains such an object it may be converted to a string with the statement

```
var json_string= JSON.stringify(js_object);
```

When messages in the form of JSON strings are received they may be converted to JavaScript object with the JSON.parse. The json_string created above can be converted into a JavaScript object with

```
var js_object= JSON.parse(json_string);
```

The above JSON element will when parsed produce a JavaScript object that is equivalent to the initial js_object.

Observer that the JSON is not a JaveScript object, but is one single string. Also wether the text is written as a single line or in the form above give the same result. Still the resemblance is striking.

In both of these programs the network address of the machine running wbssynth are written directly into the JavaSCript code. For a particular network configuration one will have to edit the .html files manually: Use a normal text editor and open the file LP12speed.html (or LP12control.html) and find the line "var wbsUri = "ws://localhost:8090";". The last part is the address for the wbssynth. The localhost should be changed into the name or IP address of the actual machine that wbssynth runs on. The port number 8090 is built into our implementation of Websockets so this cannot be changed unless you are capable of editing and compiling C++ code.

A Appendices

A.1 Advanced Linux Sound Architecture (ALSA)

The Advanced Linux Sound Architecture (ALSA) [ALSA] provides audio and MIDI functionality to the Linux operating system. The project was started ...

A.1.1 Find your card

For testing out a soundcard one may use the ALSA provided program aplay. For playing a sound file on the default sound card simply do

```
aplay LRMonoPhase4.wav
```

The LRMonoPhase4.wav [LRMP.wav] is a very convenient soundfile we found on the Internet. In cases wher there are more cards in the computer, as it will be when an USB sound card has been installed, the problem of selecting the correct output device arises. According to aplay -h the option -L or –list-pcms will list the names of the available devices. Then the option –device=NAME will select that device for output. With an USB sound card inserted in develop the –list-pcms lists about 30 possible output devices. Parts of the list is shown below

```
sysdefault:CARD=PCH
    HDA Intel PCH, ALC662 rev3 Analog
    Default Audio Device
hdmi:CARD=PCH,DEV=1
    HDA Intel PCH, HDMI 1
    HDMI Audio Output
sysdefault:CARD=Device
    USB Audio Device, USB Audio
    Default Audio Device
hw:CARD=Device,DEV=0
    USB Audio Device, USB Audio
    Direct hardware device without any conversions
```

We tried out a number of the devices with commands of the sort

```
$ aplay --device=sysdefault:CARD=Device LRMonoPhase4.wav
Playing WAVE 'lrphase.wav' :
    Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
aplay --device=sysdefault:CARD=Device hsong1.wav
Playing WAVE 'hsong1.wav' :
    Unsigned 8 bit, Rate 11025 Hz, Mono
aplay --device=sysdefault:CARD=Device english.au
Playing Sparc Audio 'english.au' :
    Mu-Law, Rate 8000 Hz, Mono
```

Long lines have been broken. At this particular test all the files seemed to play correctly. The english.wav is actually Linus Torvalds himself telling us how to pronounce the word Linux. Selecting another device may lead to different results

```
$ aplay --device=hw:CARD=Device,DEV=0 english.au
Playing Sparc Audio 'english.au' : Mu-Law, Rate 8000 Hz, Mono
aplay: set_params:1299: Sample format non available
Available formats:
- S16_LE
```

It may seem like the different devices have different capabilities when it comes to reformatting and possibly resamping the digitized sound. Our best advice is to actually try out the listed cards.

A.2 Computers and sound cards

We want to use a cheap and simple machine with some sort of Puls Code Modulator (PCM) to produce the two phase shifted sinwaves. We would prefer to use some brand of SBC and have tested a few, more to come. The test is finally done with the el cheapo amplifier and a oscilloscope, but for a first test we use a sound file and headphones. We have found a suitable soundfile (LRMonoPhase4.wav) that contains speach from right, left, middle, and middle with phase reversal. We use the aplay with this file and if it is successful we go on to connect the amplifier and LP12 motor.

In the list below we would have loved to be able to use the cheapest machines, but these also seemd to be the most troublesome. Of the NanoPIs the only one we have been able to get working is the NaoPI M1 - that by the way is not available anymore.

Althoug we cannot rule out the possibility that we configured the machines wrong, or connected the headphones or scope incorrectly, the results below leed us to the conclusion that the built in sound of the NanoPIs are not to be trusted. For this use we think the Raspberry PI is a bit to expensive. Also we believe that he Raspberry PIs do not have proper ADC, rather they emulate sound through the PWM mechanism. Actually this may be the case for the NanoPIs as well. All in all we will try some really cheap USB sound cards and hope for better luck there.

A.2.1 Desktop computer - GOOD, USB - GOOD

We started out developing the frequency synthesizer program on an ordinary desktop with the Debian Stretch distribution installed. There were no problems whatsoever when using the card selection "hw:CARD=PCH,DEV=0" at a sampling rate of 48000. The machine was a Lenovo IdeaCentre 510s running Debian 9.4 a.k.a Stretch.

We also acquired a Deltaco UCA-03 USB sound card. Using the aplay –list-pcms we found the entry

```
hw:CARD=Device,DEV=0
    USB Audio Device, USB Audio
    Direct hardware device without any conversions
```

We tried this as the correct device for thwe USB card and:

```
aplay --device=hw:CARD=Device,DEV=0 lrphase.wav
```

seemed to work as expected.

A.2.2 Laptop computer - NOT TESTED

A.2.3 NanoPI NEO SBC - BAD, USB OK

We could never make this one work. There were signal out, but the two sinewaves were always just 180° shifted. One may suspect that there is some problem with the driver or indeed the hardware. Later we tried this once more with the FriendlyCore 4.14.52 20180628 and a stereo test file, LRMonoPhase4.wav, that confirmed the signal is mono with the polarity of the channels inverted.

Even later we tried this with an USB connected Creative X-Fi sound card. That one worked perfectly. We also tested the same USB soundcard as on the desktop computer (Deltaco UAC-03). The command was

```
aplay --device=hw:CARD=Device,DEV=0 lrphase.wav
```

and the card performed flawlessly. It seems like we may use NanoPIs if we can find an USB sound card that is cheap enough to be used permanently in this combination. The Deltaco is about NOK 100 - that may be acceptable even in combination with the about NOK 150 NanoPI NEO.

A.2.4 NanoPI M1 SBC - GOOD, USB - GOOD

This machine works without problems. The image we installed was the nanopi M1 friendlycore xenial 4.14.52 20180628. Testing with aplay and LRMonoPhase4.wav reveals the channels are correct, the sound is in the middle when both channels, the change of polarity of the channels gove aconfused impression. In the alsamixer one can mute line in, all mics, and the DAC reversal should be "MM" although we do not known what that means. Set the line out and DAC to maximum. The sound buffer is about 8 s long so changing speed delays that much.

On this machine we also tested the same Deltaco UAC-03 USB sound card. This time we actually did it with the syntesizer. The the two lines in the file synth.txt was set to:

```
hw:CARD=Device,DEV=0
48000
```

and we listened with headphones. At startup the following lines were printed on the tty

```
Synthesizer version 1.0.0 started
Info: Opened sound device: hw:CARD=Device,DEV=0
Info: Exact rate = 48000/1 Hz
Command (".!" to exit): .!
```

This indicates that the 48000 Hz sampling rate is supported in hardware.

While the wbssynth was running we look at the output from top and the wbssynth is near the top, but not at the top:

```
%Cpu(s):  0.3 us,  8.0 sy,  0.0 ni, 91.7 id

PID  USER  VIRT    RES    SHR S  %CPU  %MEM  COMMAND
1442 root     0      0      0 I  25.0  0.0   kworker/0:0
1448 bhu  14536   3428   3104 S  11.8  0.7   wbssynth
```

We have removed several coloumns to fit the page. The kworker/0:0 is a kernel worker process that probably does driver related work on behalf of the Linux kernel. It seems to be closely related to the wbssynth. When the wbssynth is terminated the kworker also falls down the list. The total system load is nevertheless not alarming as this machine is not supposed to have any other tasks.

Even later we obtained two equal USB soundcards from eBay (seller meiguo1501). They were $ 1.99 a piece. Both worked and seemd to have hardware samplingrates of 44100 Hz and 48000 Hz. With these parts at that price it is ecomonical to use USB sound to build the synthesizer. These cards have cable with USB connector at the end and we intend to cut of the connector and solder the leads directly to the NanoPI.

A.2.5 NanoPI M1 Plus SBC - BAD

This is a current model that has both Ethernet, WiFi, and eMMC. The cost is somewhat higher than that of the M1, but as the M1 Plus has built in mass storage one does not need a SD card other than for installing. Sadly this machine showed same behaviour as the NEO. That was with nanopi m1 plus debian jessie 4.16.0 20180410 that we can no more find. We should try this with the FriendlyCore of same version that works on the M1.

A.2.6 NanoPI Duo SBC - BAD

Same result as most of the NanoPIs. The test was done with the Friendlycore xenial 4.14.52 20180628. For this test we also checked the aplay -l that among others returned

```
card 2: Codec [H3 Audio Codec],
    device 0: CDC PCM Codec-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

This agrees with the /etc/asound.conf where the card 2 is default both for pcm and ctl. The setup is probably correct, it simply does not work.

A.2.7 NanoPI NEO Core LTS - Unknown, USB good

This SBC has same capabilities, and runs on same Ubuntu distribution, as the plain NanoPI NEO. There benefit of the Core LTS is the built in eMMC. This makes the SBC suited to be embedded as part of the two channel synthesizer. When delivered without PIN headers soldered we may solder the USB cable of the soundcard directly to the PCB. We believe that this machine in combination with the $ 1.99 USB soundcard will be the most economical way to realize the two channel synthesizer. Connecting the eBay USB soundcard and entering aplay –list-pcms gave among others the:

```
hw:CARD=Device,DEV=0
    USB Audio Device, USB Audio
    Direct hardware device without any conversions
```

This is our best candidate for getting the synthesizer going and a test with the aplay and the LRMonoPhase4.wav returned

```
aplay --device=hw:CARD=Device,DEV=0 LRMonoPhase4.wav
Playing WAVE 'LRMonoPhase4.wav'
: Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
```

Where the report line has been broken to fit the page. The sound in the headphone wqs correct so this machine and soundcard combination will most likely be suitable for the signal generator. The details of connecting the NanoPI NEO Core is given in appendix B.

A.2.8 USB Sound cards

We have tested two USB connected sound cards, the Deltaco UAC-03 at about \$ 12 and the noname eBay card at \$ 1.99. The cheap eBay card is advertised as "USB2.0 AUDIO SOUND CARD ADAPTER CABLE 3D VIRTUAL 7.1CH". Once connected to the computer we may obtain some information about manufacturer and model with the lsusb command (Linux) The UAC-03 is reporte to be **ID 1b3f:2008 Generalplus Technology Inc.** while the noname card is reported to be **ID 1b3f:2008 Generalplus Technology Inc.** It is most likely that the inner workings are the same. It should be noted that the Deltaco have much better casing and a number of buttons. These are of no use to us, but may justify the higher price.

B Assembling the synthesizer hardware

The 100 V amplifier, the transformers, and the connection to the Linn LP12 AC motor are described in section 6.1. In this appendix the power supplies as well as the more complex connections into the SBC are explained. After testing on the bench all parts are mounted inside a steel enclosure with dimensions of about 24 x 26 x 7 cm^3 (WxDxH). The enclosure has one single mains power switch in front and on the back we find the the Harting 4AWK connector, the Ethernet 8P8C connector, and the mains lead.

B.1 12 V and 5 V power

The power supply selected for this amplifier is a 12 V, 36 W mains supply obtained from eBay at a price of about \$ 4 apiece. The seller was winddeal. This supply is used for powering the audio amplifier. For 5 V power to the SBC we uses a 5 V, 3 A step down buck converter that are fed from the 12 V supply. The latter was again obtained from eBay at a price of \$ 0.99 each. the seller was elec-module58. The first 12 V supply is mounted under a cover at the back of the enclsure and fed from the mains power switch through a 1 A fuse.

B.2 Single board computer and sound card

Following our very extensive tests we selected the NanoPI NEO Core LTS from FriendlyElec as the signal generator and control computer. This machine is built on a 40 x 40 mm^2 circuit board with soldering holes for cabling all around. The pinout of the NanoPI NEO Core LTS is shown in Figure 10. This computer has small size, small price, and built in eMMC, making it very attractive for embedded projects. We obtained the model with 256 MB

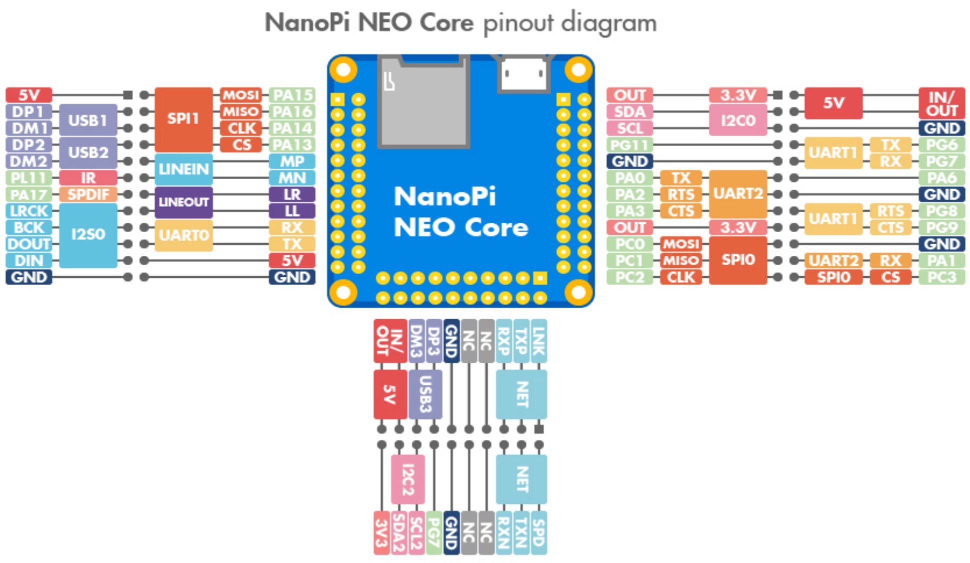


Figure 10: Connections to the NanoPI NEO Core LTS SBC

GPIO2 PIN #	Signal name	Cable color	USB
1	+5 V	red	USB power
3	USB1 Data+	green	Sound card green
5	USB1 Data-	white	sound card white
7	USB2 Data+	green	USB Data+
9	USB2 Data+	white	USB Data-
23	GND	black	USB Ground

Table 2: USB connections to the NanoPI

RAM and 4 GiB eMMC. The ordering forms does not presicely tell what eMMC capacity will be delilverd, but 4 GB seems to be sufficient. The CPU itself is an Allwinner H3, Quad-core Cortex-A7 Up to 1.2GHz. The NanoPI Core has three double rows of through board holes that may be used for soldering in PIN headers or, as we do, solder connection leads directly to the SBC PCB. The three rows are named GPIO1, GPIO2, and GPIO3. In Figure 10 the GPIO1 is on the right side, the GPIO2 on the left, and the row at the bottom is GPIO3. We did not bother to try out the sound system of the machine itself, rater connected the $ 1.99 USB sound card right away.

The connection to the USB system are all on the left side GPIO2 and the sound card conncetions are shown in Table 2. One free USB receptacle has been connected to the USB2 pins for a possible connection of a second USB device. On the same left side GPIO2 we also find the debug UART connections. These are to pins 18 (RxD), 20 (TxD), 22 (+5 V), and 24 (GND). PIN 22 can be used to power the SBC, but we choose not to connect

GPIO3 PIN #	Signal name	Cable color	Pair #	8P8C PIN #
3	TX+	white/green	3	1
4	TX-	green	3	2
5	RX+	white/orange	2	3
6	RX-	orange	2	6

Table 3: Connections between NanoPI and 8P8C connector

this lead as we supply power on GPIO1.

The connections to the Ethernet transceiver are located on the lower PIN row - GPIO3. We decided to use the EIA/TIA-568-A [TIA/EIA] color conventions. The connection between the SBC and the 8P8C Ethernet connector (often called RJ45) are shown in table 3. As the Ethernet transceiver of the NanoPI NEO is 10/100 Mbit/s only pairs 2 and 3 are used.

5 V power from the buck converter are fed in to GPIO1. We use both PINs 2 and 4 for 5 V and PINs 6 and 14 for the ground return.

References

[ALSA]	Advanced Linux Sound Architecture (ALSA) project homepage www.alsa-project.org
[cJSON]	cJSON Ultralightweight JSON parser in ANSI C http://www.json.org/
[eBay]	https://www.ebay.com/
[Kjell & Co.]	Kjell & Company https://www.kjell.com/no
[Linn Sondek]	Linn Products Limited Glasgow Road, Waterfoot, Eaglesham, Glasgow G76 0EQ
[LRMP.wav]	http://www.kozco.com/tech/soundtests.html
[Naim]	Naim Audio Ltd Southampton Road Salisbury SP1 2LN England
[Noratel]	Noratel AS Elektroveien 7

	3300 Hokksund NORWAY
[Premotec]	PREMOTEC Precision Motor Technology B.V. https://premotec.home.xs4all.nl/coreless.htm
[PuTTY]	PuTTY: a free SSH and Telnet client https://www.chiark.greenend.org.uk/ sgtatham/putty/
[Raspberry PI]	The Raspberry Pi Foundation www.raspberrypi.org
[TIA/EIA]	Wikipedia TIA/EIA-568 https://en.wikipedia.org/wiki/TIA/EIA-568
[Vinyl E.]	Vinylengine https://www.vinylengine.com/