Networking Enhancement for a GM1358 Sound Level Meter (the swiss army knife)

Digest

There is a large choice of sound pressure level meters on the market, from pretty cheap to awfully expensive ones.
The cheap ones have frequently a sufficient accuracy for many purposes (albeit not being suitable for a legal enforcement). Most of them have however either no, or extremely primitive reporting abilities.

The purpose of this development is to provide networking and reporting abilities as close to IEC 61672-1:2013 specifications as possible to an extremely cheap sound pressure level meter GM1358, by adding a ESP8266 WiFi microcontroller to it.

In the first variant sound pressure level meter + WiFi adapter + Online Dashboard the total value of the bill of material will be below 30€ !seitenzeh

The ESP8266 microcontroller will be small enough to fit into the original case of the GM1358 and the requested soldering will be limited to three wires.
With soldering skills, you can do the job in less than 10 minutes.

Your modified GM1358 will then provide USB and WiFi connectivity and be programmable to do the coolest things that only high-end devices will provide:

Evaluation of the noise level according to following time response standards (simultaneously):

- Fast (Attack t=125mS, Decay t=125mS)
- Slow (Attack t=1S, Decay 4,3dB /sec)
- Impulse (Attack t=125mS, Decay 2,9dB /sec)
- Real peak value by the minute (125mS resolution, not the maximum of readings)
- Background level (t=2000s, excluding NAT)

Statistics according to residential aircraft noise standards:

(steady noise equivalents)

- Leq 1 minute
- Leq for each hour of the day
- Leq for 24h
- Leq daytime 06:00 to 22:00
- Leq nighttime 22:00 to 6:00
- Leq 22:00 to 24:00
- Lden

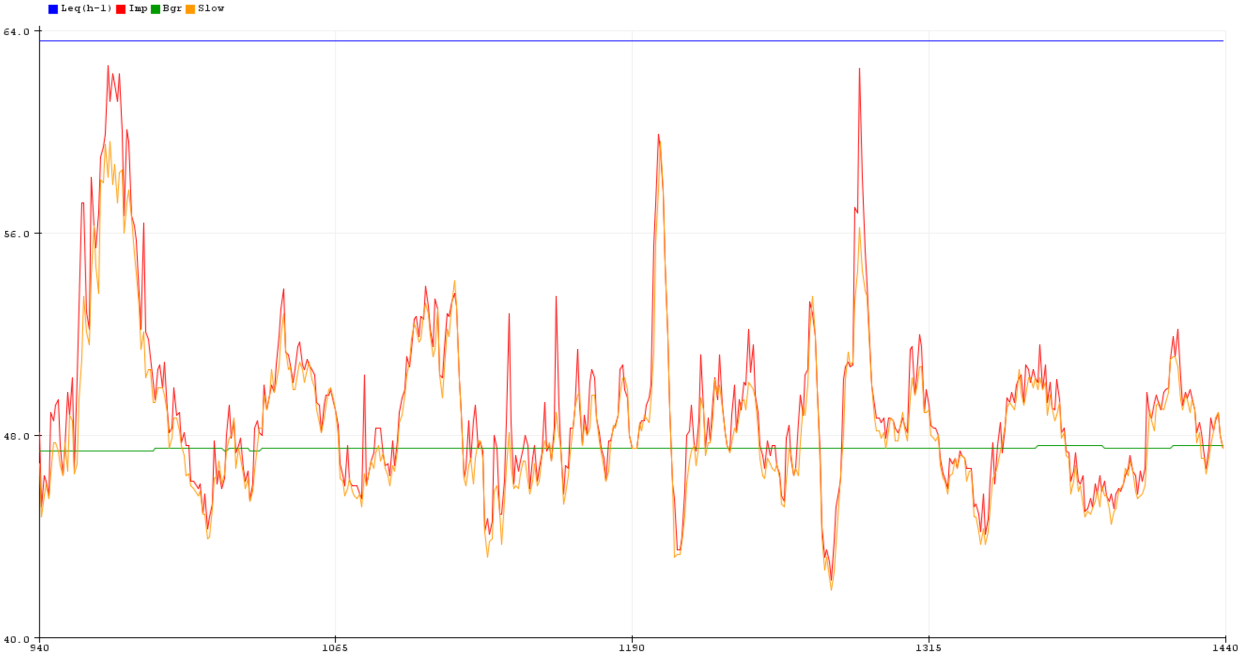
(number above threshold)

- NAT for each hour of the day
- NAT for 24h
- NAT daytime 06:00 to 22:00
- NAT nighttime 22:00 to 6:00
- NAT 22:00 to 24:00

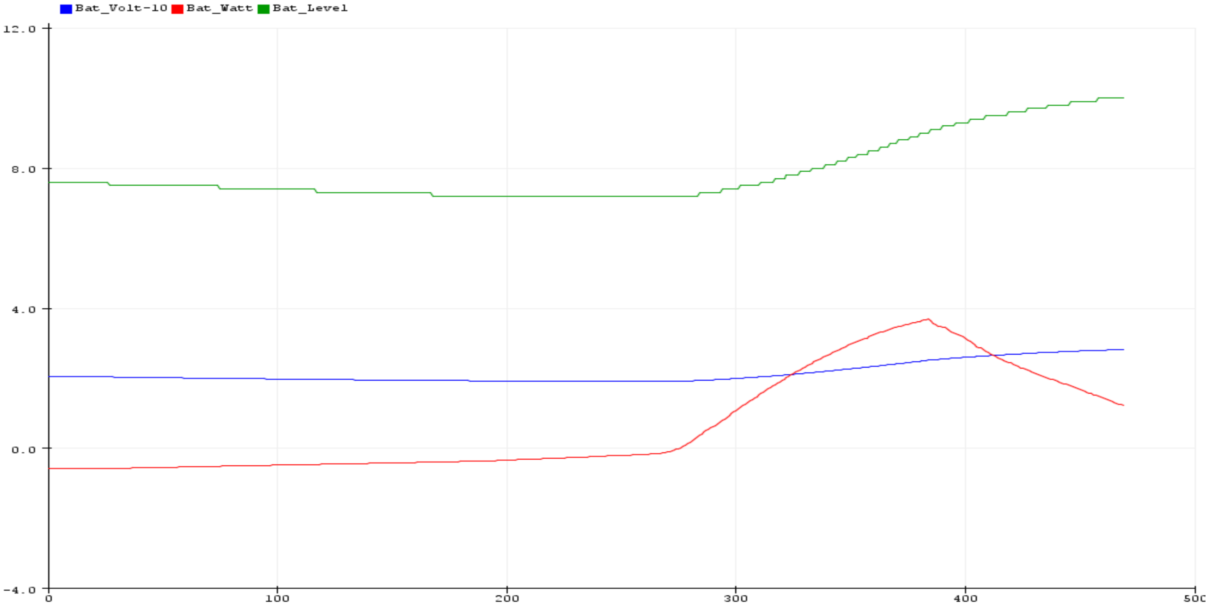
Additionally the program can grab weather information from openweathermap.org and provide the corresponding meteorological conditions.

a) You can report all this information over the USB port using a terminal program or the Serial Monitor of the Arduino IDE.
Over the Serial Plotter you can get a graphical output of the noise or battery evolution history:

USB Sound Plotter from the Arduino IDE:



USB Battery Plotter from the Arduino IDE:

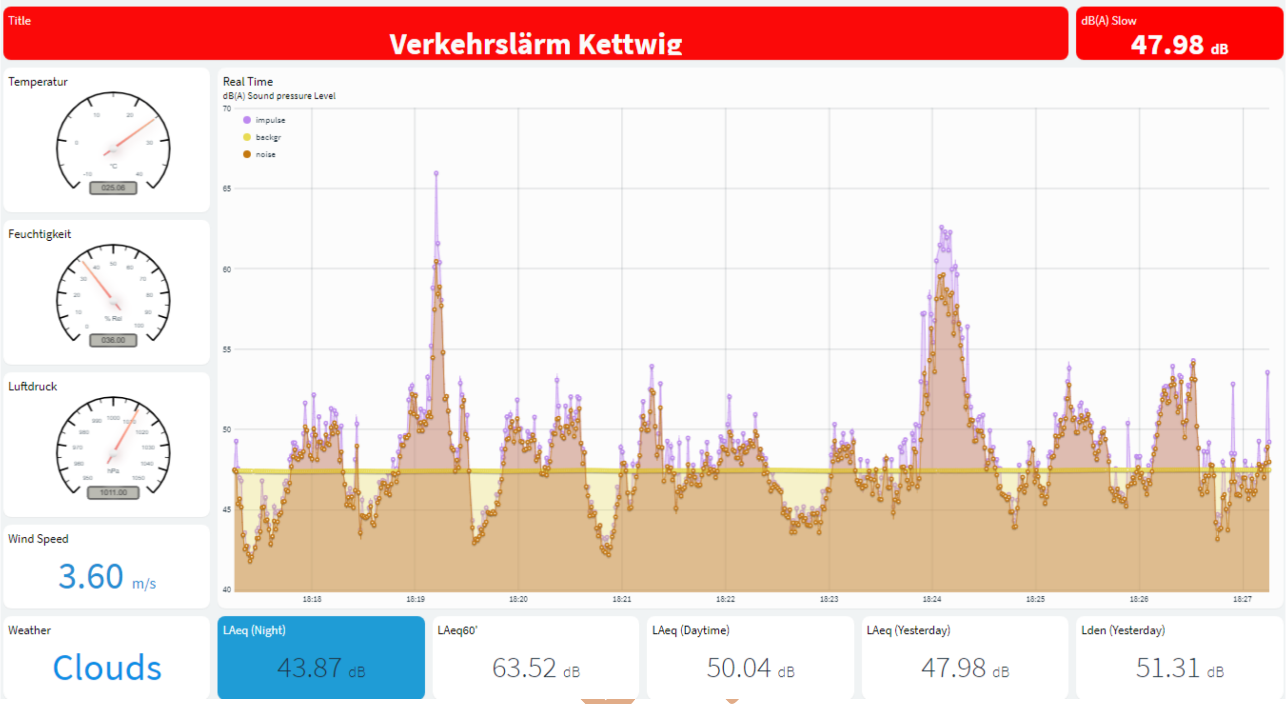


The plotter and reporting abilities of the Arduino IDE are however limited and you can only have one output at a time.
With a free cloud service as e.g Thinger.io, much more features can be used.

Cloud service Thinger

You can register free to the Cloud service Thinger.io to plot information in a very versatile way. You then can get fast real-time dashboards (that build up over time on screen) and also send information to data buckets from which you get historical data (immediately available)

Thinger real time sound plotter and weather information (example1)



Title

Verkehrslärm Kettwig

dB(A) Slow

47.98 dB

Temperatur

025.06

Feuchtigkeit

036.00

Luftdruck

1011.00

Wind Speed

3.60 m/s

Weather

Clouds

LAeq (Night)

43.87 dB

LAeq60'

63.52 dB

LAeq (Daytime)

50.04 dB

LAeq (Yesterday)

47.98 dB

Lden (Yesterday)

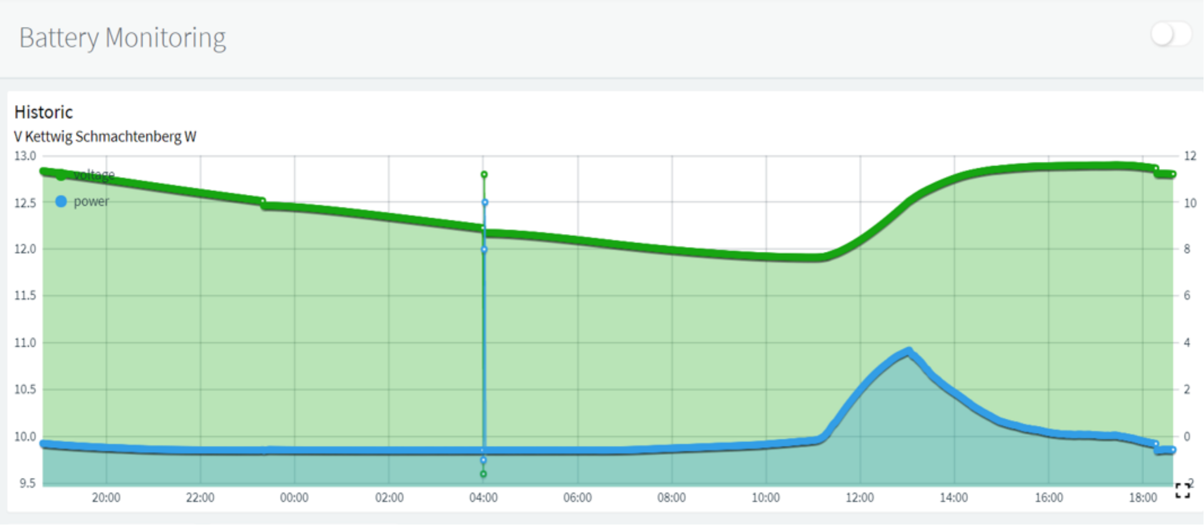
51.31 dB

With solar hardware and a separate casing, you may also build the system with the ability to be solar powered including a solar power monitor to report all information about the battery condition and the power fed by the solar panel.

You will then operate the system without electrical connection to your computer, using the Cloud service Thinger.io.

Full extension with weather report, solar power report, noise level (needs solar hardware):

Thinger historical battery (example2)



Battery Monitoring

Historic

V Kettwig Schmachtenberg W

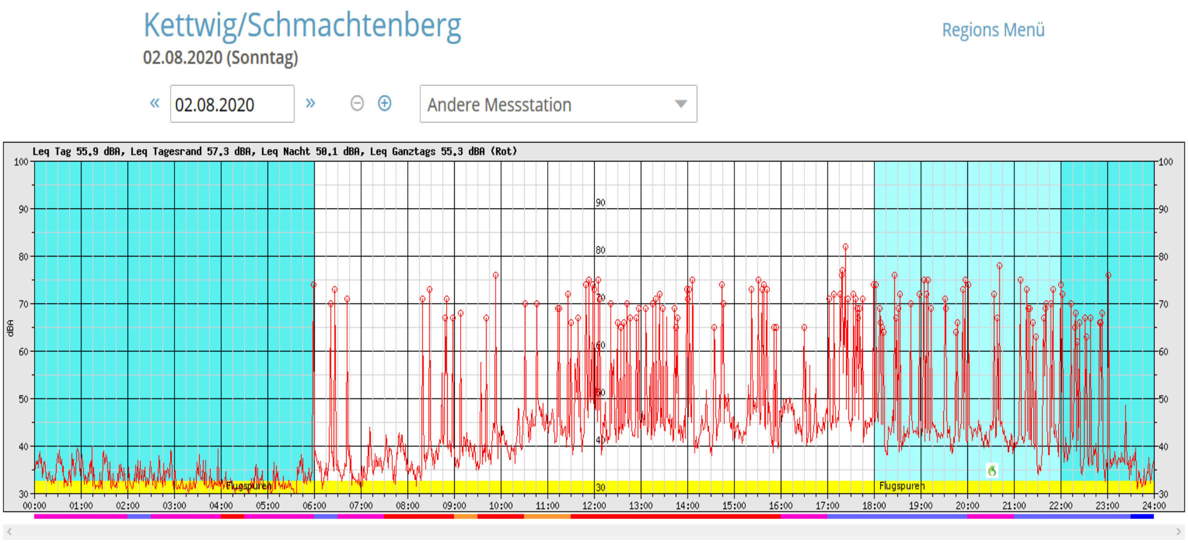
You can also build a split system with the SPL meter (+ the solar power circuitry) being located outside gathering and transmitting the sound (and battery) values over WiFi/ UDP

(Long Range LoRa is planned)
to another bare ESP8266 located inside that will provide the statistics over USB/ Thinger.

Last but not least, and back to the historical roots of the whole concept, the system is able to additionally to Thinger transmit over USB a single byte per second according to a proprietary "AK-Modulbus protocol" to a feeder program running e.g on a Raspberry Pi forwarding hourly reports to the European aircraft noise network . http://www.eans.net/EANSindex.php

This network is providing a very long time lobby-independent storage of aircraft/railway noise information managed by residentials, currently totaling about 700 privately and communal operated noise stations throughout Europa.

Noise record from DFLD/EANS. (Example3)


Kettwig/Schmachtenberg

02.08.2020 (Sonntag)

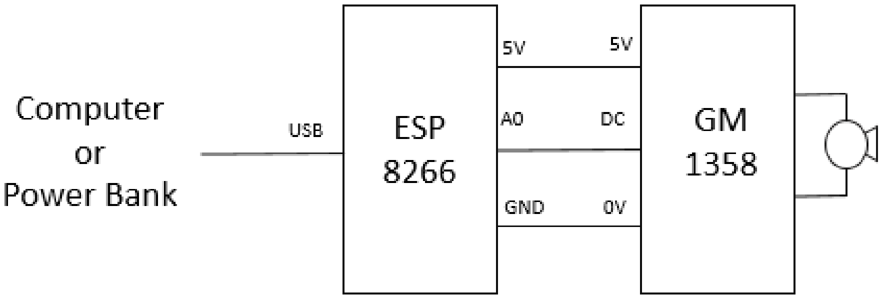
Leq Tag 55,9 dBA, Leq Tagesrand 57,3 dBA, Leq Nacht 50,1 dBA, Leq Ganztags 55,3 dBA (Rot)

Energy considerations

The built-in 9V alkaline battery block, provides about 24 hours of operation (@ 11mA).
With the retrofitted ESP8266 the total consumption will be about 48 mA, which will drain the battery block within about 8 hours only.
Fortunately, you can also power the system over the USB socket of the ESP8266 and provide over two days of continuous operation from a 3000mA lithium 18360 battery.

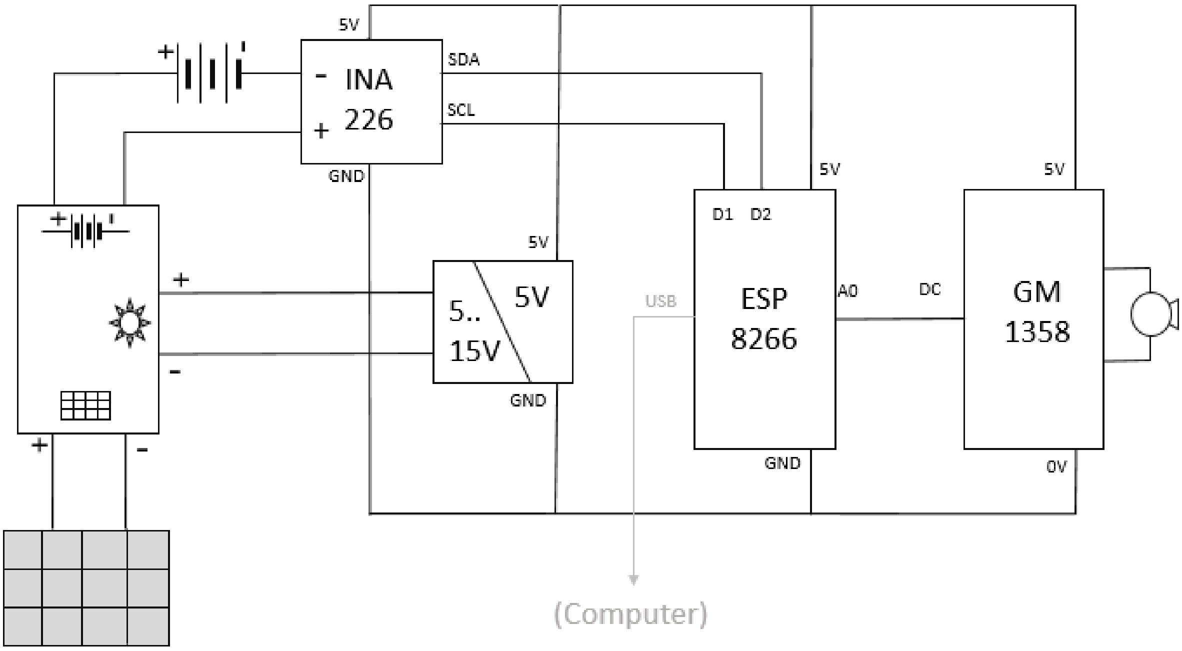
Hardware schematics

Simple version



Solar Powered Version with battery reporting

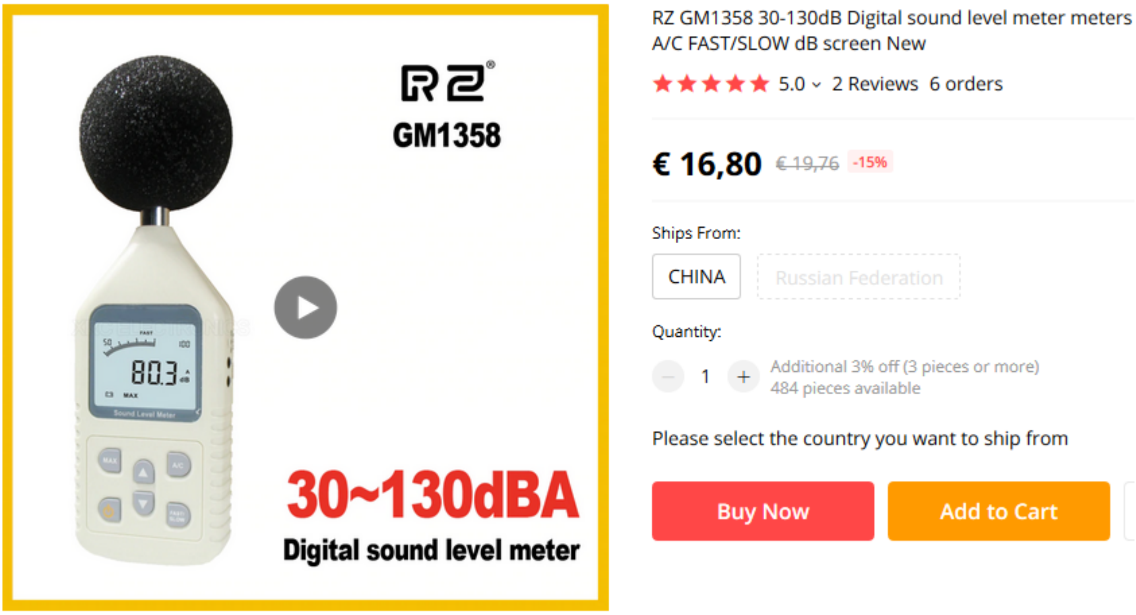
If you want a 24/24/365 off-grid operation on solar power, depending on your location (e.g. at a 45° latitude) you will need a 12V 30Ah battery and (at least) a 20W solar panel, you can hardly imagine how little energy such as solar panel still delivers on a rainy winter day!



Bill of material

Simple version

You will need exactly that model of Sound level meter: GM 1358
Do not order another model!
That one has the unique feature of providing a linear 0..1V DC signal output that is tied to GND and the ability to autostart when powered with 5V.
It can interface with an ESP 8266 with only 3 wires.
You can find some devices on eBay.com, but you will find the best offers on AliExpress to largely varying prices. Anything below 30€ is OK…



RZ GM1358 30-130dB Digital sound level meter meters A/C FAST/SLOW dB screen New

★★★★★ 5.0 ⌄ 2 Reviews 6 orders

€ 16,80 ~~€ 19,76~~ -15%

Ships From:

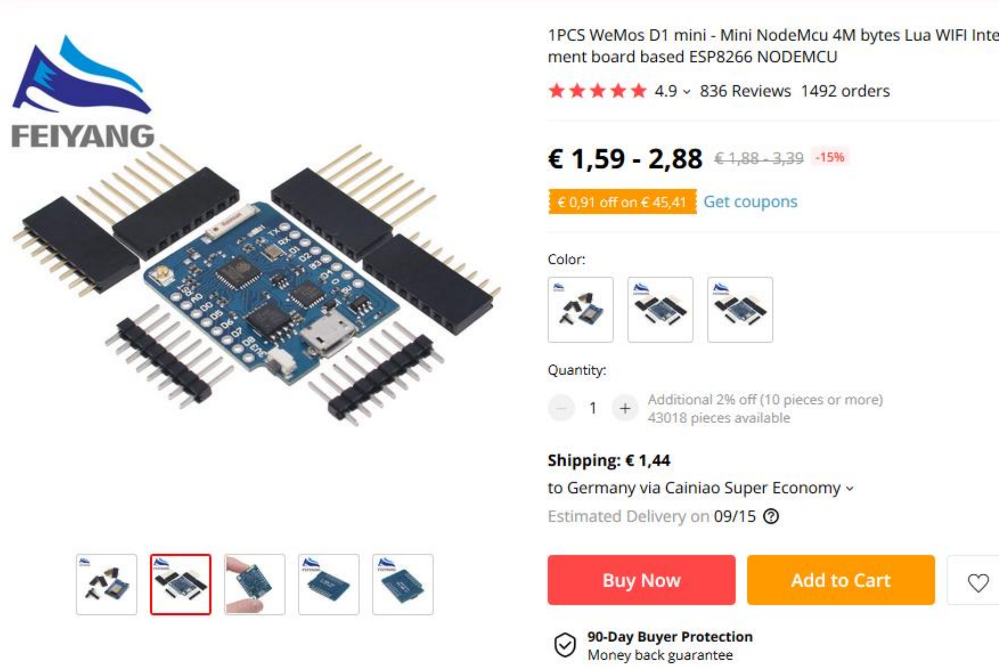
CHINA Russian Federation

Quantity:

− 1 + Additional 3% off (3 pieces or more)
484 pieces available

Please select the country you want to ship from

https://www.aliexpress.com/item/1647511133.html



1PCS WeMos D1 mini - Mini NodeMcu 4M bytes Lua WIFI Inte ment board based ESP8266 NODEMCU

★★★★★ 4.9 ⌄ 836 Reviews 1492 orders

€ 1,59 - 2,88 ~~€ 1,88 - 3,39~~ -15%

€ 0,91 off on € 45,41 Get coupons

Color:

Quantity:

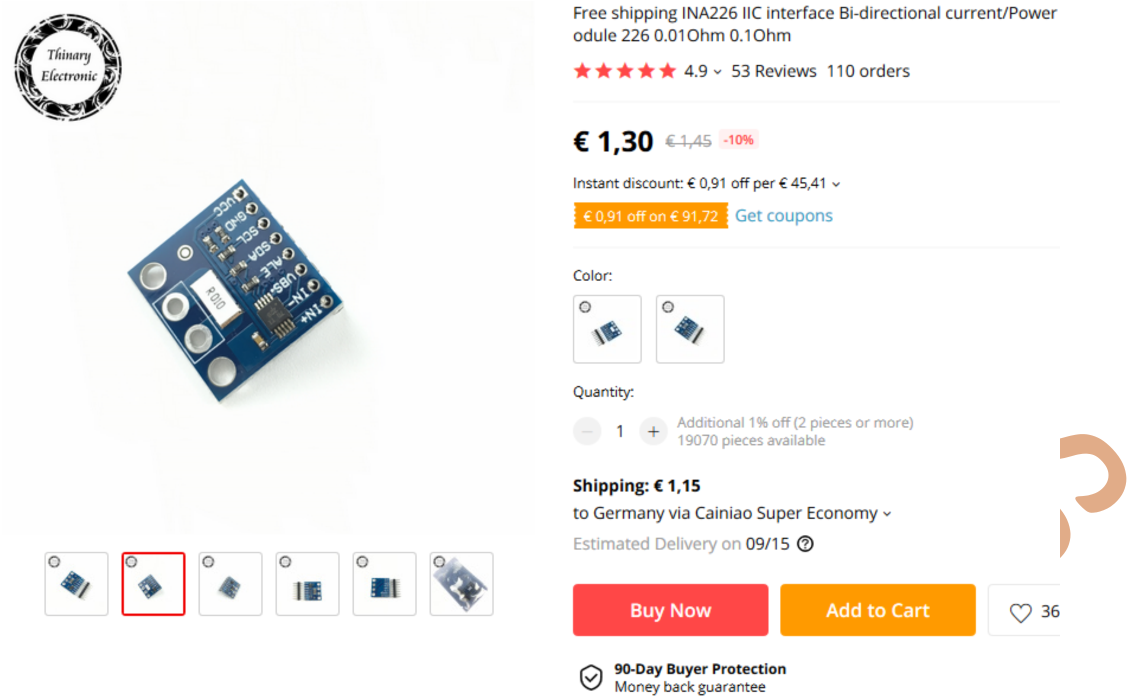
− 1 + Additional 2% off (10 pieces or more)
43018 pieces available

Shipping: € 1,44
to Germany via Cainiao Super Economy ⌄
Estimated Delivery on 09/15

90-Day Buyer Protection
Money back guarantee

https://www.aliexpress.com/item/32831353752.html

Option Solar power & monitoring



Free shipping INA226 IIC interface Bi-directional current/Power odule 226 0.01Ohm 0.1Ohm

★★★★★ 4.9 ⌄ 53 Reviews 110 orders

€ 1,30 ~~€ 1,45~~ -10%

Instant discount: € 0,91 off per € 45,41 ⌄

€ 0,91 off on € 91,72 Get coupons

Color:

Quantity:

– 1 + Additional 1% off (2 pieces or more)
19070 pieces available

Shipping: € 1,15

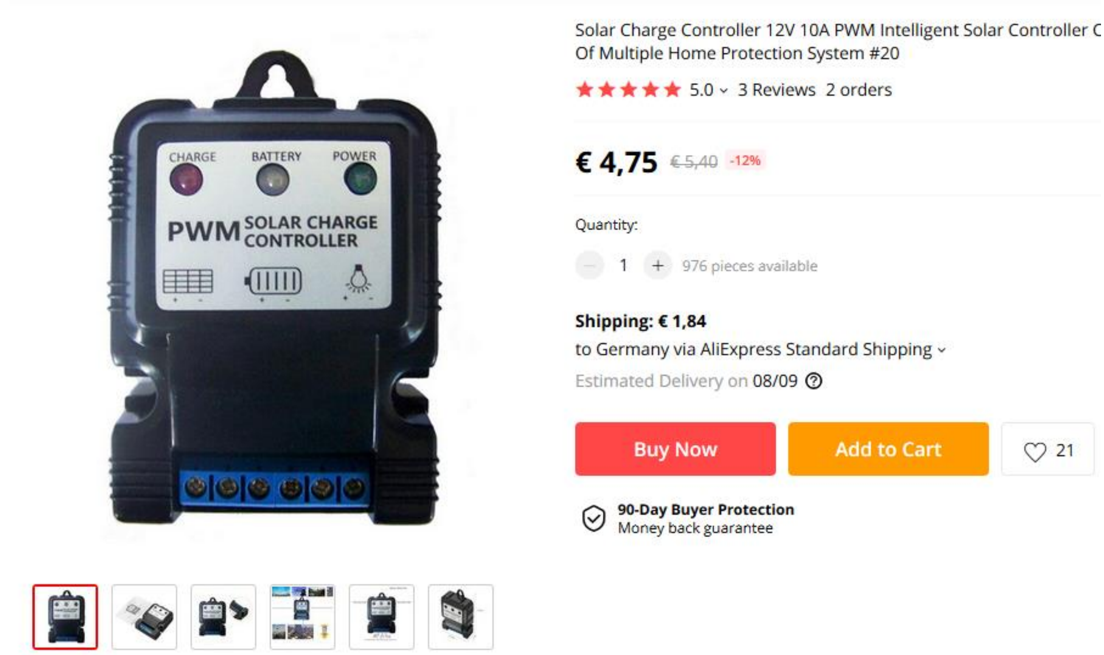
to Germany via Cainiao Super Economy ⌄

Estimated Delivery on 09/15 ?

Buy Now | Add to Cart | ♡ 36

90-Day Buyer Protection
Money back guarantee

https://www.aliexpress.com/item/32830512534.html



Solar Charge Controller 12V 10A PWM Intelligent Solar Controller C Of Multiple Home Protection System #20

★★★★★ 5.0 ⌄ 3 Reviews 2 orders

€ 4,75 ~~€ 5,40~~ -12%

Quantity:

– 1 + 976 pieces available

Shipping: € 1,84

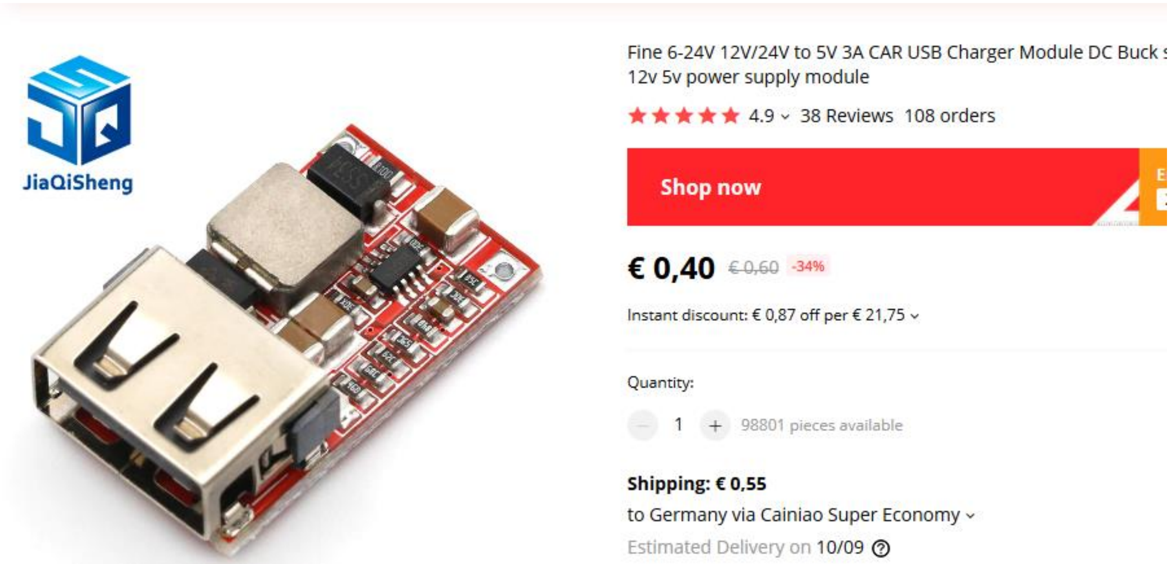
to Germany via AliExpress Standard Shipping ⌄

Estimated Delivery on 08/09 ?

Buy Now | Add to Cart | ♡ 21

90-Day Buyer Protection
Money back guarantee

https://www.aliexpress.com/item/4000083888167.html



Fine 6-24V 12V/24V to 5V 3A CAR USB Charger Module DC Buck s
12v 5v power supply module

★★★★★ 4.9 ⌄ 38 Reviews 108 orders

€ 0,40 ~~€ 0,60~~ -34%

Instant discount: € 0,87 off per € 21,75 ⌄

Quantity:

− 1 + 98801 pieces available

Shipping: € 0,55

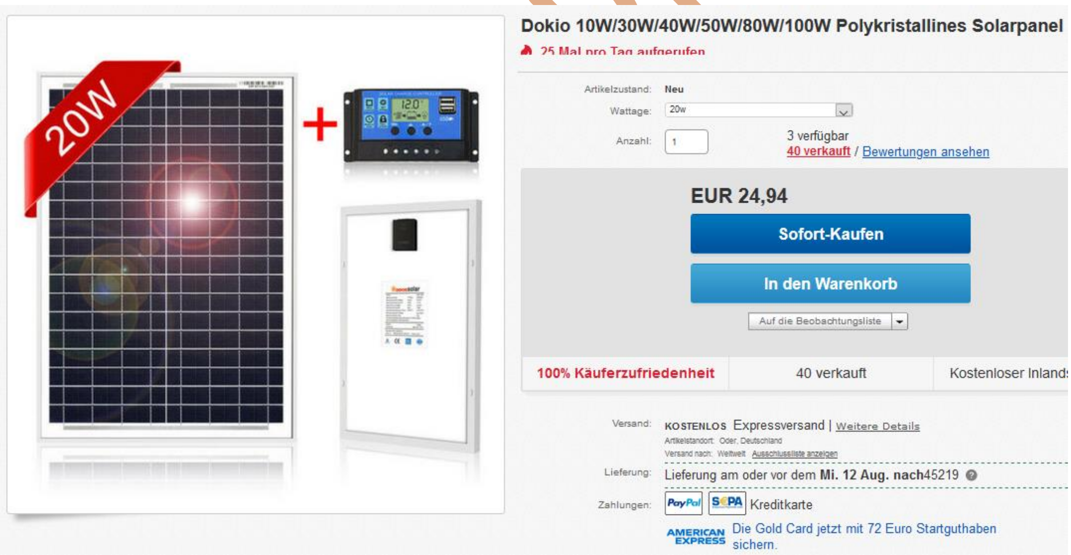
to Germany via Cainiao Super Economy ⌄

Estimated Delivery on 10/09

Order a **12V 12Ah or more lead acid battery** and a **20W or more solar panel** locally.

Take care to order <u>solar suited batteries and solar panels with a glass front and an aluminum frame</u>. Avoid cheap solar stuff with resin front, they will decay within a few months in bright sun.

Here are some examples from Germany:



Dokio 10W/30W/40W/50W/80W/100W Polykristallines Solarpanel

25 Mal pro Tag aufgerufen

Artikelzustand: **Neu**

Wattage: 20w

Anzahl: 1 — 3 verfügbar
40 verkauft / Bewertungen ansehen

EUR 24,94

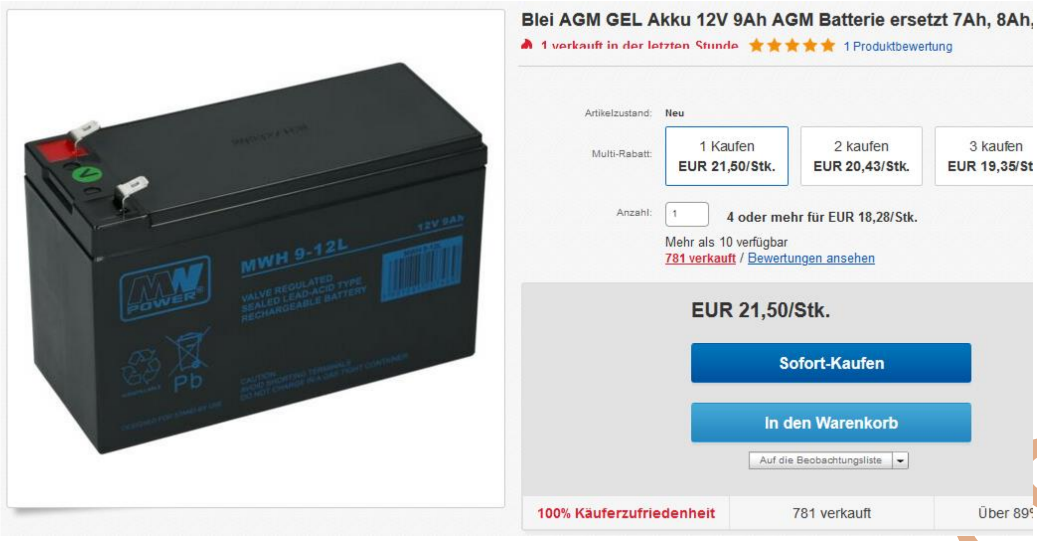
100% Käuferzufriedenheit | 40 verkauft | Kostenloser Inlands

Versand: KOSTENLOS Expressversand | Weitere Details
Artikelstandort: Oder, Deutschland
Versand nach: Weltweit Ausschlussliste anzeigen

Lieferung: Lieferung am oder vor dem **Mi. 12 Aug. nach**45219

Zahlungen: PayPal SEPA Kreditkarte
AMERICAN EXPRESS Die Gold Card jetzt mit 72 Euro Startguthaben sichern.

(this combo-offer already includes the solar controller and even the USB 5V power supply)



Blei AGM GEL Akku 12V 9Ah AGM Batterie ersetzt 7Ah, 8Ah,

1 verkauft in der letzten Stunde ★★★★★ 1 Produktbewertung

Artikelzustand: Neu

Multi-Rabatt: 1 Kaufen EUR 21,50/Stk. | 2 kaufen EUR 20,43/Stk. | 3 kaufen EUR 19,35/St

Anzahl: 1 4 oder mehr für EUR 18,28/Stk.

Mehr als 10 verfügbar

781 verkauft / Bewertungen ansehen

EUR 21,50/Stk.

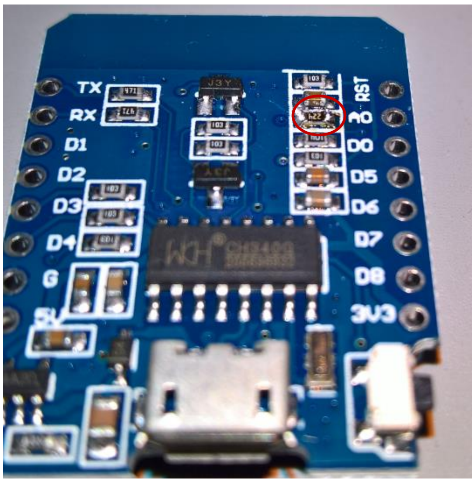
100% Käuferzufriedenheit | 781 verkauft | Über 89

You may use an existing car battery if you have one in acceptable condition, but do not buy one: car batteries are not optimized for longer periods of operation with a voltage below 13,8v and will decay rapidly.

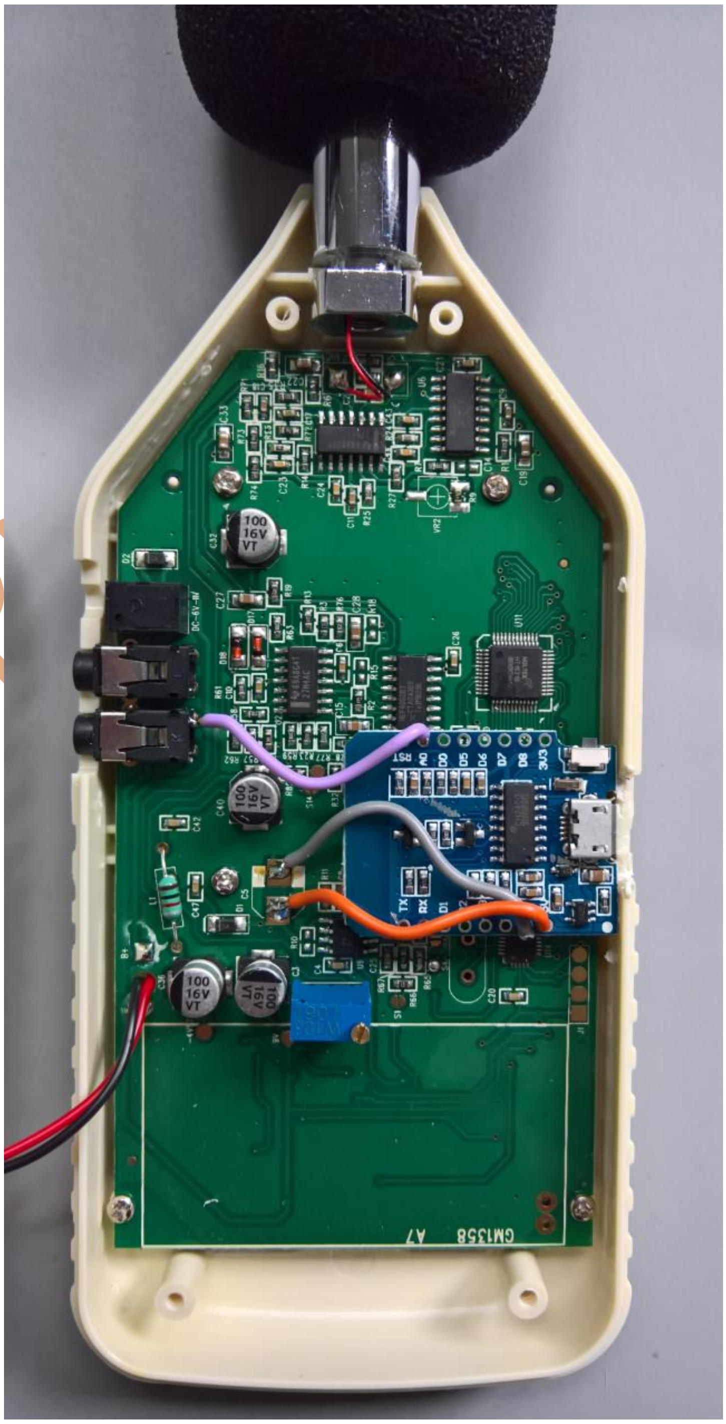
Modification of the GM1358

Simple Version

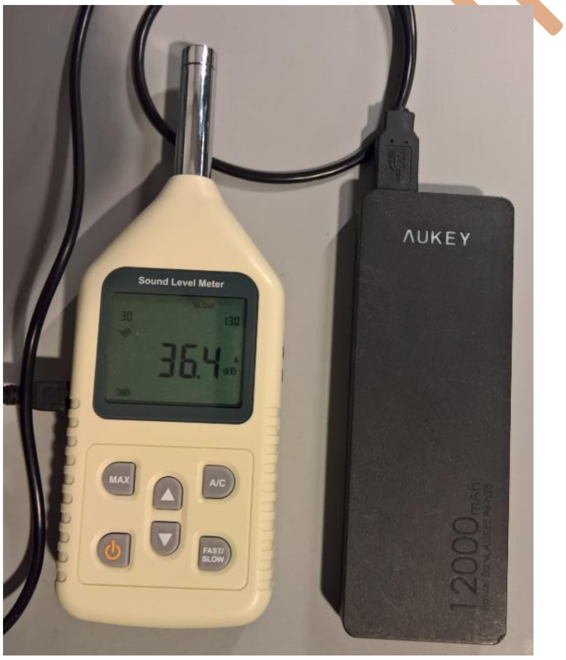
The GM1358 is easy to dismantle, remove 4 screws: 2 at the top and 2 in the battery compartment and you can open the casing.
The first thing might sound scaring but it is easier as it looks: we need to bridge one very tiny resistor on the ESP8266 to change the input range from 0..3.3V to 0..1V.



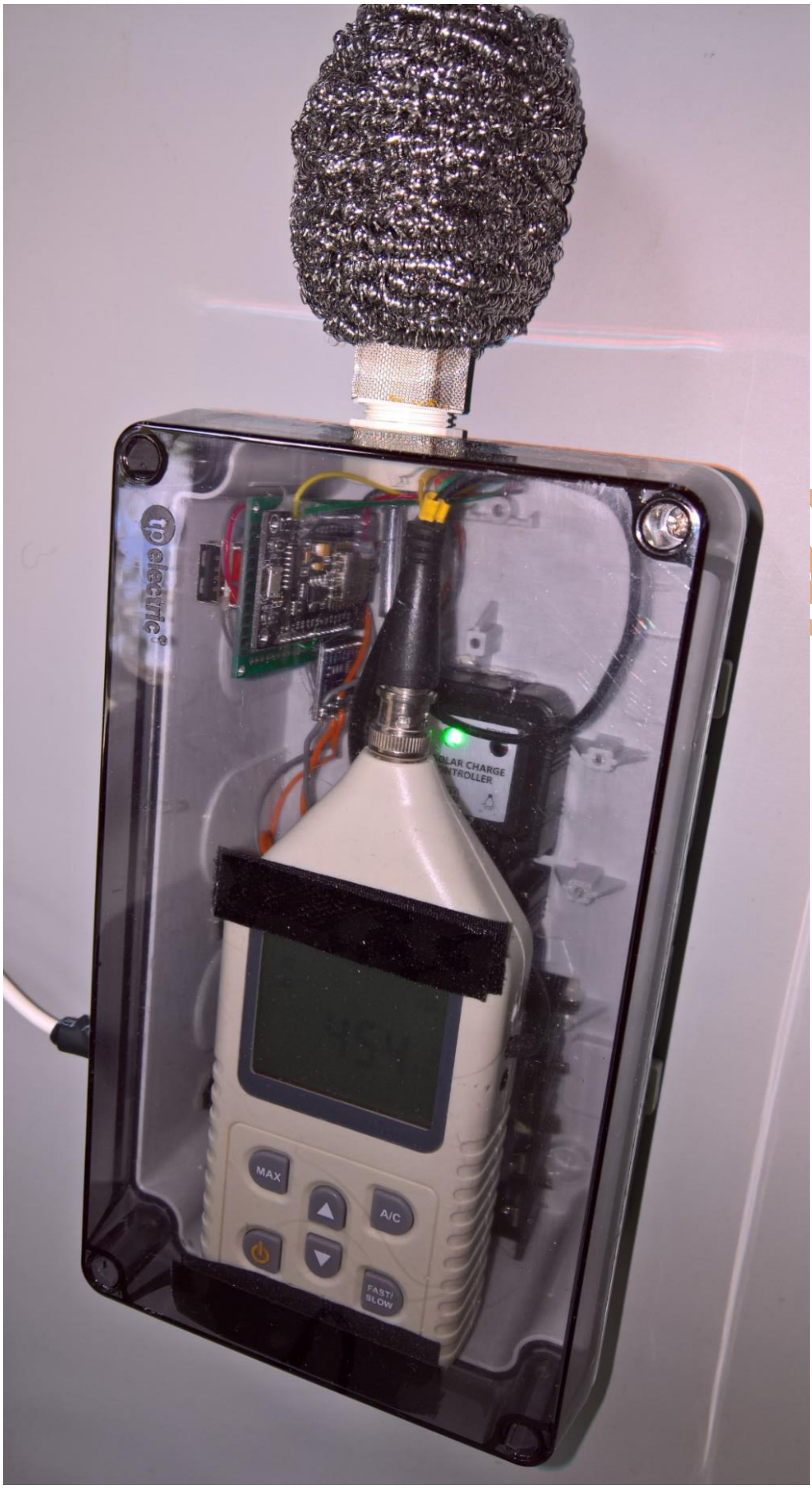
Take a single strand of electrical wire and solder it to bridge the resistor close to A0 as shown:



Solder the grey wire to GND and the pad on the PCB board, solder the orange wire to GND and the pad on the PCB board, solder the pink wire to A0 and the inside contact of the DC jack
Grind the plastic case to free room for the USB plug.
Glue the ESP8266 as shown.
Screw the case back. Done!



Outdoor / Solar Version



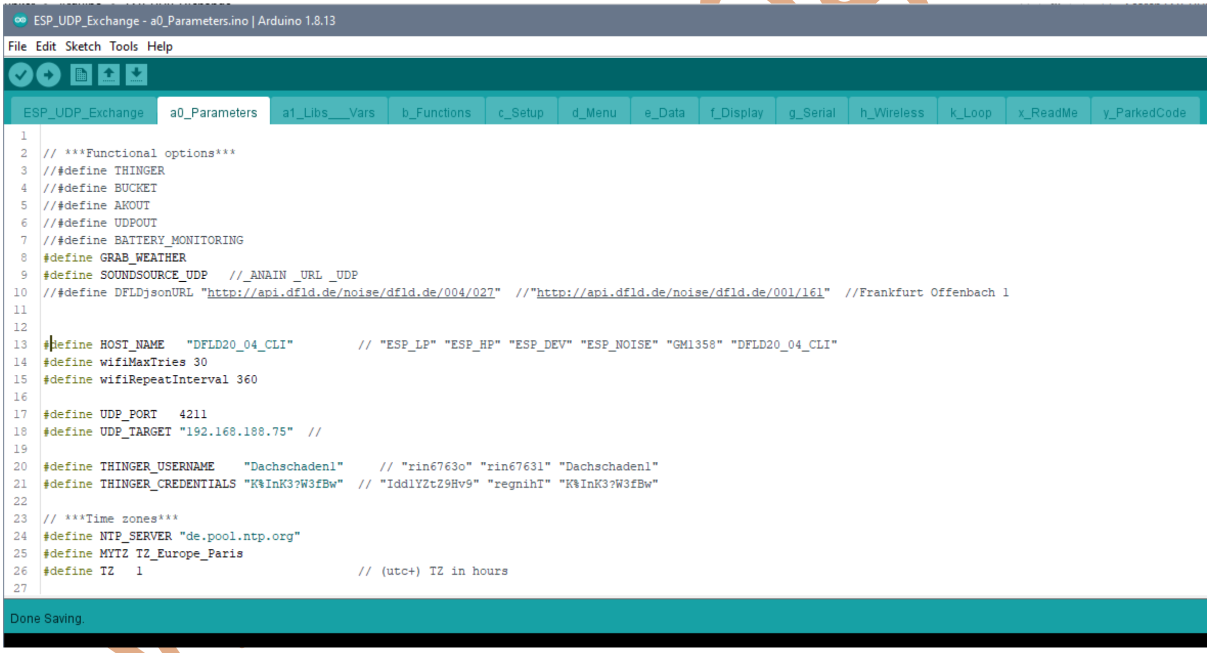
to be described in detail later…

Software

To operate the noise station you will need a WLAN connection and internet connectivity.
You will need to upload the program to the ESP 8266.
For that, if you don't have it already, you should install the Arduino programming Interface (IDE) from www.arduino.cc <http://www.arduino.cc>, on your favourite PC, and then install the ESP framework.
Cf these instructions: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/

Optionally you could register to two free services:
The first step will be to create a free account at www.thinger.io <http://www.thinger.io>
(you can manage two devices free of charge) and
optionally create a free account at www.openweathermap.org <http://www.openweathermap.org>
if you want to get weather information for your location.

The current files of my program are hosted on GitHub:
https://github.com/rin67630/Swiss-Army-Knife-for-GM1380-Sound-Pressure-Meter
You may download all .ino files in a folder of your choice and start the Arduino IDE:



```

// ***Functional options***
//#define THINGER
//#define BUCKET
//#define AKOUT
//#define UDPOUT
//#define BATTERY_MONITORING
#define GRAB_WEATHER
#define SOUNDSOURCE_UDP   //_ANAIN _URL _UDP
//#define DFLDjsonURL "http://api.dfld.de/noise/dfld.de/004/027"  //"http://api.dfld.de/noise/dfld.de/001/161"  //Frankfurt Offenbach 1


#define HOST_NAME   "DFLD20_04_CLI"        // "ESP_LP" "ESP_HP" "ESP_DEV" "ESP_NOISE" "GM1358" "DFLD20_04_CLI"
#define wifiMaxTries 30
#define wifiRepeatInterval 360

#define UDP_PORT   4211
#define UDP_TARGET "192.168.188.75"  //

#define THINGER_USERNAME    "Dachschaden1"     // "rin6763o" "rin67631" "Dachschaden1"
#define THINGER_CREDENTIALS "K%InK3?W3fBw"  // "Idd1YZtZ9Hv9" "regnihT" "K%InK3?W3fBw"

// ***Time zones***
#define NTP_SERVER "de.pool.ntp.org"
#define MYTZ TZ_Europe_Paris
#define TZ   1                             // (utc+) TZ in hours

```

I have built the program on the top of my framework ESP-Krarajan on GitHub:
https://github.com/rin67630/ESP-Karajan , which takes care of ancillary tasks like, booting to the network and providing scheduling and timing functionalities.

The Arduino IDE provides tabs to split the program into well-structured subparts, so you can jump easily during development between every subpart:

a0) parameters and options to use the program *(the only part you really need to modify)*
a1) libraries and global variables used
b) different functions used in the program
c) setup process
d) menu
e) data processing
f) display (not used here...)
g) serial reports
h) wireless processes
k) finally the scheduler itself which will periodically start the routines listed from d) to h).

Additionally two tabs with comments only are added for convenience:
x_ReadMe and y_ParkedCode, where you can put reminders and code examples.

In my programming style, I deliberately refrain to use too abstract c++ concepts in order to make the program accessible to the majority of people with a minimum of Arduino experience.
I also did put a great attention to comment my code indicating the reason why I have done most steps in that way.

Basically, unless you know exactly what you are doing the only Tab, for you to make changes is a0 parameters and options.

The software is written to operate on different hardware configuration and can gather data from different sources.

```
#define HOST_NAME "GITHUB"

// ***Functional Configuration***
#define WEATHER_SOURCE_URL  //_URL _NONE (UDP planned)
#define BATTERY_SOURCE_UDP  //_INA _UDP _NONE
#define SOUND_SOURCE_UDP    //_ANAIN _URL _UDP _NONE
//#define DFLDjsonURL "http://api.dfld.de/noise/dfld.de/004/027"
//"http://api.dfld.de/noise/dfld.de/001/161"  //Frankfurt Offenbach 1

#define THINGER
#define WRITE_BUCKETS        //(Comment out, if second device @ Thinger)

#define Console0 Serial      // Port for user inputs
#define Console1 Serial      // Port for user output
#define Console2 Serial     // Port for midnight report
#define Console3 Serial     // Port for boot messages
#define Console4 Serial      // Port for AK-Outputs

//#define PUBLISH_DFLD        //If this is the DFLD master, comment out else
//#define PUBLISH_BATTERY     //If this is the battery master, comment out else
//#define PUBLISH_SOUND       //If this is the sound master, comment out else
#define UDP_TARGET "192.168.xxx.xxx"  // Client address for Sound or Battery if defined
#define UDP_PORT   4211

//  ***Credentials***
#define SMARTCONFIG  // (WiFi Credentials over GogglePlay/Apple App SmartConfig)
// alternatively to Smartconfig App, you can comment out Smartconfig and enter your credentials
//#define WIFI_SSID "Enter Your SSID"
//#define WIFI_PASS "Enter Yout Password"
#define wifiMaxTries 30
#define wifiRepeatInterval 1000

String OPEN_WEATHER_MAP_APP_ID =      "Application ID";
String OPEN_WEATHER_MAP_LOCATION_ID = "Location Id";
String OPEN_WEATHER_MAP_LANGUAGE =    "de";
boolean IS_METRIC =                   true;
```

```
#define THINGER_USERNAME    "Username"
#define THINGER_CREDENTIALS "Credentials"

// ***Time zones***
#define NTP_SERVER "de.pool.ntp.org"
#define MYTZ TZ_Europe_Paris
#define TZ   1                              // (utc+) TZ in hours

// ***Acoustical parameters***
#define Ao94 1024  // 747  for AK module with offset and 2,5v | 1050 for linear 0..1V
#define Ao47 530   // 458  for AK module with offset and 2,5v |  550 for linear 0..1V

#define  UPPER_LIMIT_DB              78 // Just defines the upper/lower limit of plots
#define  LOWER_LIMIT_DB              31
#define  EVENT_THRESHOLD_LEVEL       52 // Begin of Exceedance level
#define  MEASUREMENT_THRESHOLD_LEVEL 48 // Begin of measurement level
#define  MIN_EXCEEDANCE_TIME         15 // Minimum duration of an event
#define  MAX_EXCEEDANCE_TIME         60 // Maximum duration of an event
#define  LISTENING_TIME              50 // mimimum time between events

// ***Electrical parameters***
//#define DEVICES_FOUND INA.begin(3, 40000) //3A Max, 40mOhm Shunt
#define SHUNT    40000    // 16666 = 0,1 Ohm +// 0,02Ohm  or 40000
#define AMPERE   3        // 10 or 5
#define SERIAL_SPEED 9600 //9600  115200 230400
#define MIN_VOLT  9.6     // 11.8  9.6
#define MAX_VOLT 12.8     // 14.2  12.8
#define MIN_AMP  -0.8
#define MAX_AMP  +0.8
#define MIN_WATT -1
#define MAX_WATT +8
```

In the first chapter ***Functional configuration*** you will determine:

a) The host name (for your WiFi and e.g. for Thinger

b) Whether you will be using Weather information, mainly you will use
   _URL to get the information from openweathermap.org.
   _NONE if you don't need the weather information.

c) Configure the source of the battery information:
   _NONE, if you don't have any battery information.
   _INA if you have an INA226 module to measure current and voltage
   _UDP if you have in the same network another ESP8266
   running the same sketch as a master

d) Configure the source of the sound information
   _NONE, if you don't have any battery information
   _ANAIN if the sound level is coming from the analog input A0
   _UDP if you have in the same network another ESP8266
   running the same sketch as a master
   _URL if the sound level is coming from a JSON URL of www.dfld.de
   (then you need to uncomment the next line and provide the URL
   that will provide the sound information)

The line #define THINGER is determining, if you are using the cloud services of thinger.io. If you do not, please un-comment that line.

The lines with #define CONSOLEx determine where some parts of the sketch are issued. Normally to Serial, with additional UARTS ports like Serial1 are possible.

The line #define WRITE_BUCKETS must be un-commented if you are using a second device in one Thinger account. (Only one device is supposed to write the buckets defined in the common program (buckets are longtime storage at Thinger)

In the next chapter ***Credentials***, you will enter the credential information for different services.

The sketch is using SmartConfig a facility from Espressif that enables to configure your WiFi credentials we do not need to recompile the whole sketch.

You have got three possibilities:

a) If you ever have a run your ESP8266 with another sketch containing the credentials successfully, then the ESP has stored how to connect in its nonvolatile memory and you just can't go on without caring for credentials.
b) If you never have run your ESP8266 with credentials, you can comment out #define SMARTCONFIG and enter your credentials in the 2 next lines, after having removing the slashes. After successful connection, it is recommended to return to the initial active SMARTCONFIG and for security remove your credentials from the sketch
c) You can reconfigure your ESP8266 from an Android or Apple smartphone to every local WiFi to which you have access.
   To do that, please download the application ESP SMARTCONFIG from the respective app-store and follow the instructions there.

USB-Serial Menu

The Software provides a simple “single character” command-line menu over the USB serial line.
Commands are given by a single character and executed over [return]
Commands are stackable: you can give several characters then [return]; all commands will be executed in sequence.
Example: U+++ means: Apply 94&47dB Defaults and Increase Offset by 3dB.
Usually an upper case letter sets the function and the lower case reset the function.

Control actions

'Z': //Reset the ESP device
'C': //Apply 94dB Calibration
'c': //Apply 47dB Calibration
'U': //Apply 94&47dB Defaults
'+': //Increase Offset by 1dB
'-': //Reduce Offset by 1dB

Control Display

(abandoned, could be re-implemented, it would need a display, Thinger is more powerful and needs no hardware)

'0': //Display mode 0
…
'3': //Display mode 3

Periodical reports over the USB serial line

'A': //serialPage AK

(this is not a printable report, it issues one byte every second to feed the DFLD website)

'P': //Periodical Reports on	'p': //Periodical Reports off

Options for periodical reports:

'D': //Day Report	'd': //no Day Report	
'H': //Hour Report	'h': //no Hour Report	
'M': //Minute Report*	'm': //no Minute Report	(Battery)
'S': //Second Report*	's': //no Second Report	(Noise)
'E': //Event Report	'e': //no Event Report	

* these reports are designed to produce serial Plotter compatible results.

Example1 : PDHmsE	means: Print Daily, Hourly, no minute, no second, Events
Example 2: p	means: stop printing reports.
Example 3: P	means: resume printing reports with last options
Example 4: Sd	means: now with Second reports without Daily reports.

One shot reports

n.b. these reports stop periodical reports, resume with “P” to return to periodical printing.

'L':	// Leq Report	by 24h
'N':	// NAT Report	by 24h
'B':	// Battery Report by 24H	
'b':	// Battery Report (Actual measurements)	

'?': //List parameters
'~': //List WLAN / Radio settings.

Report examples

Minute Hour and Events

Menu command MHEP:

```
08:55:59.490 -> Bat_Volt-10:1.928 Bat_Watt:-0.593 Bat_Level:7.200
08:56:59.461 -> Bat_Volt-10:1.927 Bat_Watt:-0.593 Bat_Level:7.200
08:57:59.462 -> Bat_Volt-10:1.926 Bat_Watt:-0.593 Bat_Level:7.200
08:58:59.502 -> Bat_Volt-10:1.925 Bat_Watt:-0.593 Bat_Level:7.200
08:59:59.489 -> BatAhBat:0.000 A0dBLEQ:58.8 WindSpeed:0 Direction:1072693248
08:59:59.536 -> Bat_Volt-10:1.924 Bat_Watt:-0.593 Bat_Level:7.200
09:00:59.478 -> Bat_Volt-10:1.923 Bat_Watt:-0.593 Bat_Level:7.200
09:01:59.466 -> Bat_Volt-10:1.922 Bat_Watt:-0.593 Bat_Level:7.200
09:02:47.466 -> PKTm: 09:01:37 PKdB:52.8 ATdB: 50.2 ATsec:48 PK-10dB: 49.7 PK-10sec: 41 NAT:1
09:02:59.489 -> Bat_Volt-10:1.921 Bat_Watt:-0.593 Bat_Level:7.200
09:03:59.488 -> Bat_Volt-10:1.920 Bat_Watt:-0.593 Bat_Level:7.200
```

Menu command L:

```
Leq : for Saturday, 08 August 2020
 Hour   |  00  |  01  |  02  |  03  |  04  |  05  |  06  |  07  |  08  |  09  |  10  |  11  |
 Leq dB | 58.1 | 58.3 | 58.3 | 58.4 | 58.4 | 58.5 | 58.6 | 58.7 | 58.8 | 61.1 | 65.4 | 65.8 |
 Hour   |  12  |  13  |  14  |  15  |  16  |  17  |  18  |  19  |  20  |  21  |  22  |  23  |
 Leq dB | 66.0 | 66.2 | 66.3 | 62.9 | 63.2 | 63.6 | 63.8 | 54.0 | 56.3 | 56.6 | 57.3 | 57.8 |
```

Menu command N:

```
NAT : for Saturday, 08 August 2020
 Hour   |  00  |  01  |  02  |  03  |  04  |  05  |  06  |  07  |  08  |  09  |  10  |  11  |
 NAT    |  00  |  00  |  01  |  00  |  01  |  00  |  00  |  00  |  00  |  01  |  24  |  23  |
 Hour   |  12  |  13  |  14  |  15  |  16  |  17  |  18  |  19  |  20  |  21  |  22  |  23  |
 NAT    |  20  |  21  |  19  |  06  |  12  |  09  |  34  |  12  |  13  |  01  |  01  |  00  |
```

Menu command B:

```
Battery History :
 Hour   |   00   |   01   |   02   |   03   |   04   |   05   |   06   |   07   |   08   |   09   |   10   |   11   |
 Bat Ah | -0.045 | -0.047 | -0.048 | -0.048 | -0.048 | -0.048 | -0.048 | -0.049 | -0.049 | +0.000 | +0.000 | +0.000 |
 Hour   |   12   |   13   |   14   |   15   |   16   |   17   |   18   |   19   |   20   |   21   |   22   |   23   |
 Bat Ah | +0.000 | +0.000 | +0.000 | +0.000 | +0.000 | +0.000 | +0.000 | +0.000 | -0.044 | -0.044 | -0.044 | -0.044 |
```

Automated reporting on Serial1 (e.g. on thermic-printer) or on UDP:

This is an example of the ongoing event + midnight summary report

NAT: (Number of Above Threshold in hour;

PKTime: (Peak Time of the event)

Leq4:   (Level equivalent for the time defined by max-10dB to max-10dB on the other side)

t10:   (time defined by max-10dB to max-10dB on the other side)

Leq3::   (Level equivalent for the time above threshold)

t AT: (Time above threshold)

```
------------------------------------------------

NAT|PKTime   |PKdB|Leq4|t10|Leq3|tAT|
 09|18:46:07|72.0|67.6|28 |66.9|66 |
 01|19:03:48|72.9|68.5|27 |67.8|66 |
 02|19:16:37|69.8|65.6|23 |65.2|51 |
 03|19:18:56|65.2|62.3|25 |62.3|50 |
 04|19:21:32|66.5|63.6|33 |63.4|69 |
 05|19:31:20|64.6|60.0|17 |60.1|33 |
 06|19:49:01|62.8|59.2|21 |59.3|41 |
 07|19:55:36|69.9|65.1|24 |64.8|53 |
 08|19:57:44|70.7|67.8|25 |67.1|59 |
 01|20:00:22|64.7|61.6|30 |61.6|60 |
 02|20:12:42|63.9|61.2|32 |61.2|63 |
 03|20:36:49|69.5|66.4|29 |66.2|61 |
 04|20:45:26|64.5|61.3|25 |61.4|49 |
 05|20:58:18|68.5|65.2|22 |64.9|48 |
 01|21:10:13|68.2|65.2|34 |64.9|75 |
 02|21:21:39|70.7|66.8|28 |66.2|67 |
 03|21:33:48|72.2|68.6|30 |68.1|69 |
 04|21:38:24|71.7|67.7|24 |67.2|55 |
 05|21:41:08|74.4|71.2|27 |70.5|64 |
 06|21:43:09|70.7|68.0|31 |67.6|70 |
 07|21:45:01|70.1|66.4|23 |65.8|54 |
 08|21:46:41|72.9|69.1|32 |68.8|70 |
 09|21:48:48|71.7|68.9|30 |68.4|68 |
 01|22:02:02|69.2|66.6|25 |66.2|56 |
 02|22:05:02|72.2|68.4|22 |67.2|60 |
 03|22:07:24|68.3|65.3|28 |65.2|57 |
 04|22:14:10|74.1|68.6|19 |67.7|49 |
 05|22:19:29|69.1|66.5|27 |66.2|58 |
 06|22:24:02|67.0|64.1|28 |64.0|57 |
 07|22:25:55|69.8|65.6|26 |65.2|58 |
 08|22:28:26|69.0|65.5|26 |65.0|59 |
 09|22:33:42|65.6|62.7|26 |62.7|52 |
 01|23:10:48|57.3|54.0|22 |55.0|31 |
 02|23:16:23|59.9|54.6|23 |55.5|30 |
 03|23:23:34|58.8|54.5|51 |55.3|64 |
 04|23:41:10|56.0|51.2|44 |51.5|46 |

Daily Report for
Sunday, 23 August 2020
Hour| Leq  |  NAT  |  Ah    |
00  | 39.3 |  026  | -0.039 |
01  | 39.5 |  000  | -0.037 |
02  | 37.1 |  000  | -0.036 |
03  | 36.8 |  000  | -0.035 |
04  | 37.0 |  000  | -0.035 |
05  | 43.9 |  001  | -0.035 |
06  | 43.4 |  002  | -0.035 |
07  | 52.5 |  010  | -0.040 |
08  | 47.6 |  004  | -0.038 |
09  | 52.5 |  010  | -0.040 |
10  | 52.4 |  017  | -0.037 |
11  | 52.3 |  008  | -0.018 |
12  | 33.2 |  012  | +0.000 |
13  | 57.9 |  012  | +0.000 |
14  | 55.1 |  013  | +0.132 |
15  | 59.0 |  010  | +0.000 |
16  | 58.1 |  015  | +0.000 |
17  | 59.3 |  013  | +0.000 |
18  | 56.2 |  009  | +0.000 |
19  | 53.3 |  008  | +0.000 |
20  | 49.9 |  005  | +0.000 |
21  | 57.1 |  009  | +0.000 |
22  | 54.8 |  009  | +0.000 |
23  | 48.1 |  004  | +0.000 |
Extra hours cf. Man.
25  | 48.1 |  000  | +0.000 |
26  | 49.0 |  197  | -0.295 |
27  | 52.5 |  157  | -0.295 |
28  | 42.1 |  040  | +0.000 |
29  | 52.4 |  013  | +0.000 |

NAT|PKTime  |PKdB|Leq4|t10|Leq3|tAT|
 01|00:48:56|57.3|52.8|47 |53.5|55 |
 01|05:58:36|76.2|73.3|24 |72.4|59 |
 01|06:15:56|70.9|65.8|26 |65.3|60 |
 02|06:21:05|71.2|66.0|24 |65.5|55 |
 03|06:45:38|68.5|65.7|29 |65.5|61 |
 05|08:14:30|56.5|51.8|32 |52.5|36 |
 06|08:17:54|67.2|63.7|18 |63.2|42 |
 07|08:47:40|74.6|69.5|20 |68.2|58 |
 01|09:05:24|71.5|67.9|25 |67.0|64 |
 02|09:09:36|59.0|54.9|51 |55.6|76 |
 03|09:15:44|72.5|69.6|28 |69.1|64 |
 04|09:25:56|72.5|69.0|27 |68.2|66 |
 05|09:32:01|62.2|58.0|37 |58.0|72 |
 06|09:34:02|58.1|52.9|42 |53.6|49 |
 07|09:35:25|57.5|53.7|43 |54.5|57 |
 08|09:40:17|75.6|72.4|30 |71.1|84 |
 09|09:42:43|62.6|57.4|52 |57.9|86 |
 10|09:44:18|59.9|57.0|20 |57.4|35 |
 11|09:46:54|61.5|55.9|39 |56.6|55 |
 12|09:53:19|69.5|56.0|60 |56.5|95 |
 00|09:59:03|59.9|56.0|22 |56.8|30 |
...
```

To hold the solar panel: https://www.thomann.de/de/thomann_orchesterpult.htm ~15€
To hold the electronics and battery:
https://www.thomann.de/de/flyht_pro_uac_universal_alu_case_s.htm ~50€