



HamGroup

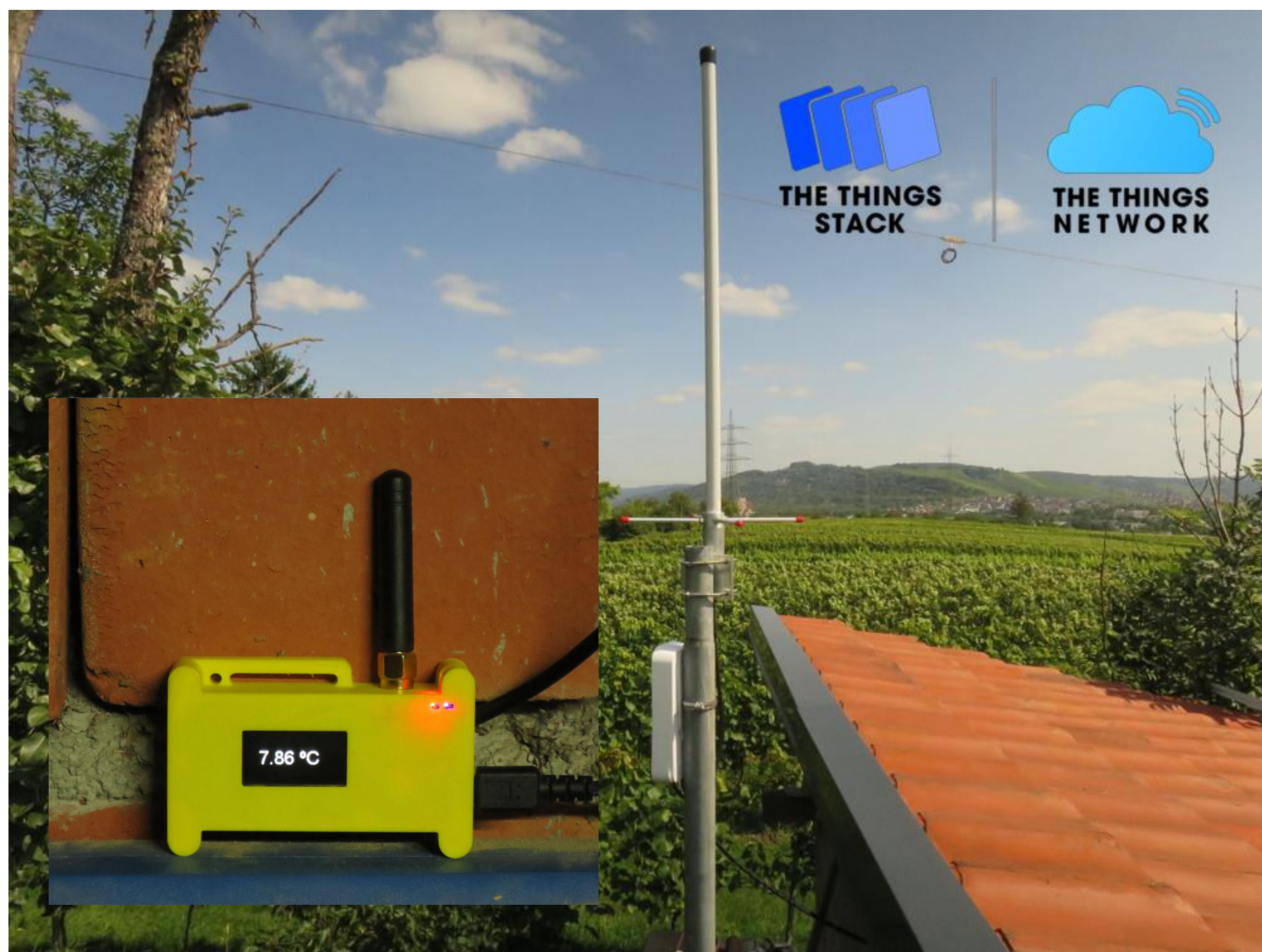
LoRa / LoRaWAN / IoT

Internet der Dinge

Online-Workshop

*LoRaWAN-
Temperatursensor*

DS 18B20



LORAWAN-WORKSHOP

- » Ziel:
 - » Programmierung des LoRaWAN-Temperatur-Sensors



LORAWAN-WORKSHOP

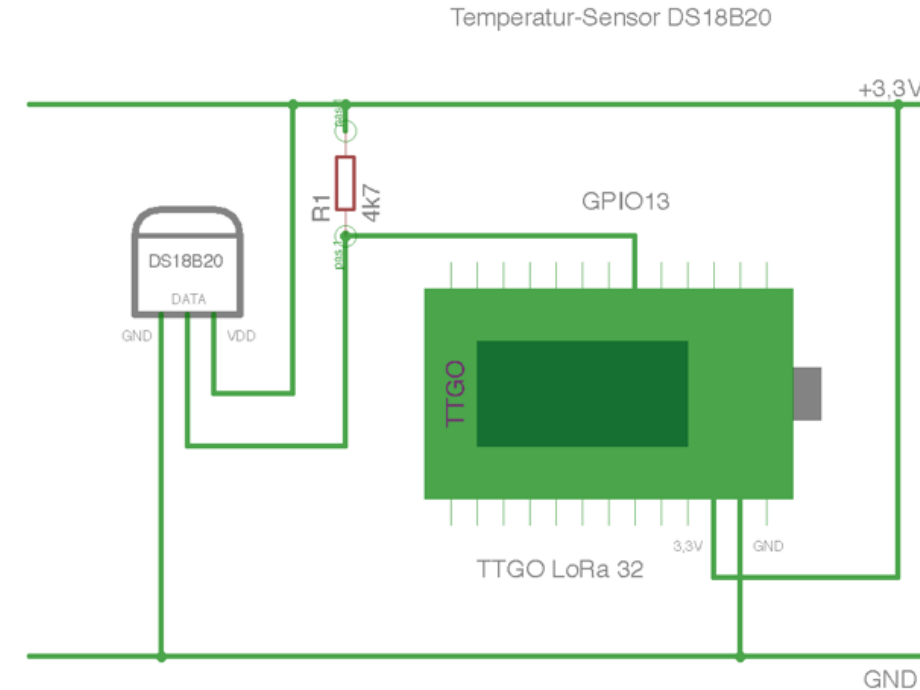
- » Der LoRaWAN-Temperatur-Node misst die Temperatur mit einem DS 18B20, generiert eine Payload und versendet diese mit LoRaWAN

Zusätzlich wird die Temperatur auch auf dem Display angezeigt und über die serielle Schnittstelle übertragen.

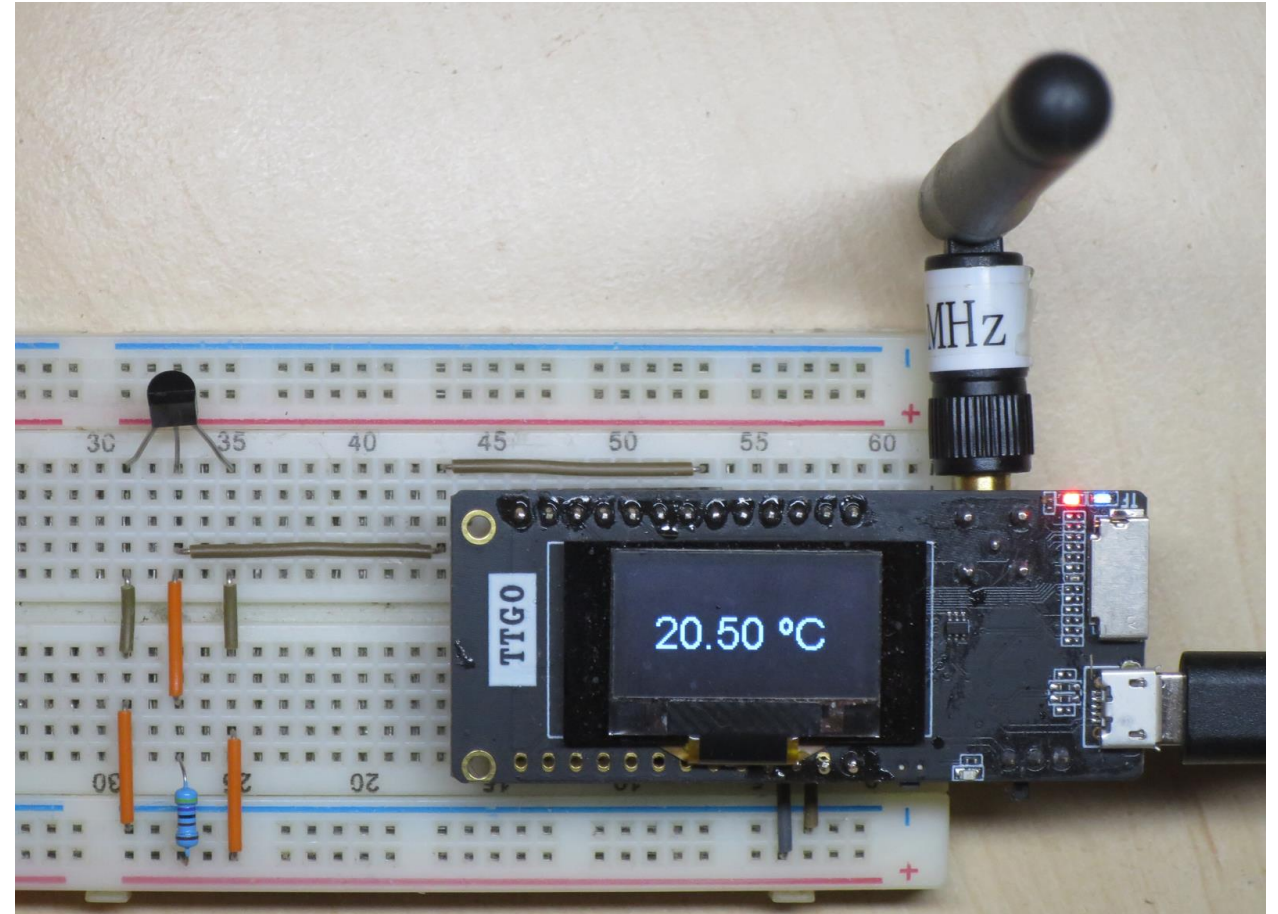
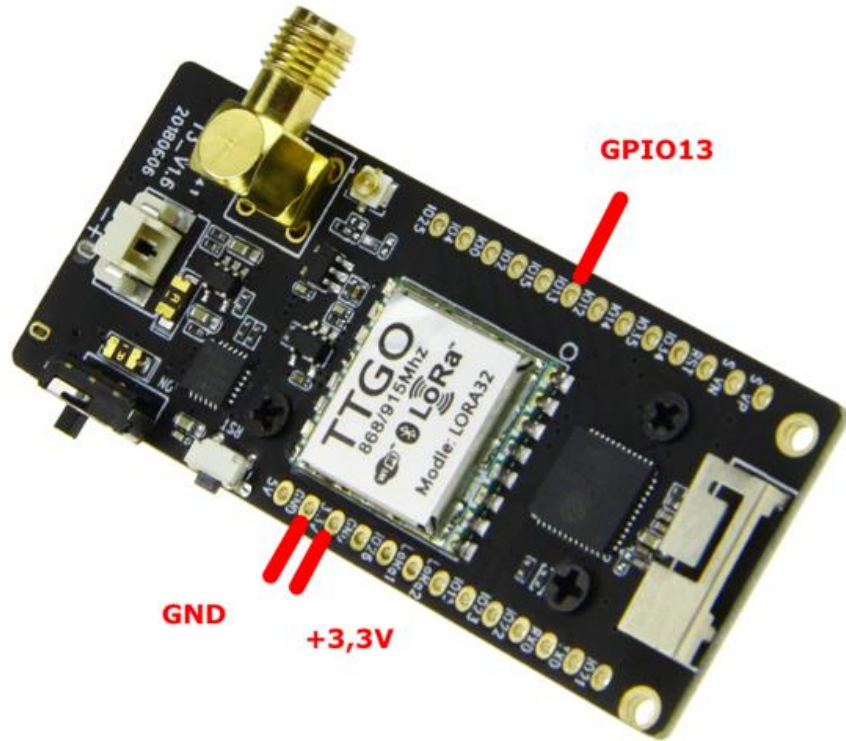


LORAWAN-WORKSHOP

- » 1-Wire bzw. One-Wire oder Eindraht-Bus beschreibt eine serielle Schnittstelle der Firma Dallas Semiconductor Corp. (heute Maxim Integrated)
- » in unserer Schaltung verwenden wir drei Leitungen:
 - VDD
 - DQ mit Pullup-Widerstand 4k7
 - GND
- » VDD wird an +3,3V angeschlossen. DQ nutzt den GPIO13 als Datenbusleitung.
- » weitere Infos zum 1-Wire-Bus: <https://de.wikipedia.org/wiki/1-Wire>
- » Datenblatt des Temperatursensors: <https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf>



LORAWAN-WORKSHOP





LORAWAN-WORKSHOP

» Bibliotheken:

» OneWire Library

https://www.pjrc.com/teensy/td_libs_OneWire.html

» DallasTemperature Library

<https://github.com/milesburton/Arduino-Temperature-Control-Library>

OneWire von Jim Studt, Tom Pollard, Robin James, ...
Glenn Trewitt, Jason Dangel, Guillermo Lovato, Paul...

2.3.8 installiert

Access 1-wire temperature sensors, memory and other chips.

[Mehr Information](#)

2.3.8 ▼

ENTFERNEN

DallasTemperature von Miles Burton ...
<miles@mnetcs.com>, Tim Newsome...

3.9.0 installiert

Arduino Library for Dallas Temperature ICs Supports

DS18B20, DS18S20, DS1822, DS1820

[Mehr Information](#)

3.9.0 ▼

ENTFERNEN



LORAWAN-WORKSHOP

» Bibliotheken:

- » Die OneWire Library ist für die Datenkommunikation über den 1-Wire-Bus zuständig.
- » Die DallasTemperature Library übernimmt die Kommunikation mit dem Temperatursensor und liefert die aufbereitete Messwerte

OneWire von Jim Studt, Tom Pollard, Robin James, Glenn Trewitt, Jason Dangel, Guillermo Lovato, Paul... ...

2.3.8 installiert

Access 1-wire temperature sensors, memory and other chips.

[Mehr Information](#)

2.3.8 ▼

ENTFERNEN

DallasTemperature von Miles Burton <miles@mnetcs.com>, Tim Newsome... ...

3.9.0 installiert

Arduino Library for Dallas Temperature ICs Supports DS18B20, DS18S20, DS1822, DS1820

[Mehr Information](#)

3.9.0 ▼

ENTFERNEN



LORAWAN-WORKSHOP

» Temperaturnode

» Quellcode:

- » herunterladen
- » entpacken
- » Keys eintragen
- » compilieren
- » hochladen



```
*****
DARC-HamGroup LoRaWAN

Temperatur-Node DS18B20

Based on the LoRaWAN example from Thomas Telkamp and Matthijs

16.01.2022 Jürgen, DL8MA
18.12.2024 Jürgen, DL8MA

http://www.amateurfunk.de/lora

passender Payload-Formatter befindet sich ganz unten in einem Kommentarfeld

*/
#include <lmic.h> // LMIC library from the library-manager
#include <hal/hal.h> // included in the lmic library
#include <SPI.h>
#include "SSD1306Wire.h" // ThingPulse OLED SSD1306 - Library ht
#include <OneWire.h>
#include <DallasTemperature.h>

// This EUI must be in little-endian format, so least-significant-byte
// first. When copying an EUI from ttnctl output, this means to reverse
// the bytes. For TTN issued EUIS the last bytes should be 0xD5, 0xB3,
// 0x70.
static const u1_t PROGMEM APPEUI[8]={ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
void os_getArtEui (u1_t* buf) { memcpy_P(buf, APPEUI, 8);}
```

<https://amateurfunk.de/lora/vortraege/241218/temp-node.zip>



LORAWAN-WORKSHOP

» Payload

» es werden zwei Bytes übertragen.

DevAddr:	26 0B 78 1B	Payload:	{ bytes: [5,230], temperatur: 51 }	05 E6	FPort: 1
DevAddr:	26 0B 78 1B				
DevAddr:	26 0B 78 1B	Payload:	{ bytes: [5,229], temperatur: 50.9 }	05 E5	FPort: 1
DevAddr:	26 0B 78 1B				
DevAddr:	26 0B 78 1B	Payload:	{ bytes: [5,228], temperatur: 50.8 }	05 E4	FPort: 1



LORAWAN-WORKSHOP

» Temperaturmessung

```
for( i = 0; i < 10; i++ ) {                                     // 10 Messungen
  sensors.requestTemperatures();
  temperatur = sensors.getTempCByIndex(0);
  summe = summe + temperatur;                                   // aufsummieren
}
temperatur = summe / i;                                       // Mittelwert bilden

/* optional */
ringPuffer[ 2 ] = ringPuffer[ 1 ];                             // neue Messung in Ringpuffer schreiben
ringPuffer[ 1 ] = ringPuffer[ 0 ];
ringPuffer[ 0 ] = temperatur;
if( ringPuffer[ 1 ] < 9999 and ringPuffer[ 2 ] < 9999 ) {
  temperatur = 0.25 * ringPuffer[ 0 ] + 0.5 * ringPuffer[ 1 ] + 0.25 * ringPuffer[ 2 ]; // gleitender Mittelwert aus den letzten
}                                                            // bilden
/* optional */

Serial.println( temperatur );
```



LORAWAN-WORKSHOP

» Webhook

zur Weiterleitung der Daten auf einen
Applikationsserver

<https://www.p37.de/LoRaWAN/ttn/liste.php?call=DL8MA>

<https://www.p37.de/LoRaWAN/ttn/anzeige.php?call=DL8MA>

DL8MA durch eigenes Call / Kennung ersetzen



Webhook format *

JSON

Base URL *

https://www.p37.de

Downlink API key

The API key will be provided to the endpoint using the "X-Downlink-Apikey" header

Request authentication ?

Use basic access authentication (basic auth)

Additional headers

+ Add header entry

Filter event data ?

+ Add filter path

Enabled event types

For each enabled event type an optional path can be defined which will be appended

Uplink message

An uplink message is received by the application



LORAWAN-WORKSHOP

» Payload

Beispiel: 49,95 °C

* 10 499,5

round() 499

+ 1000 1499

Komma verschieben (für eine Nachkommastelle)

runden

1000 addieren -> in positiven Bereich verschieben

Durch die Addition von 1000 verschiebt sich der Wertebereich ins Positive.

Damit können Temperaturwert von -100 °C aufwärts als positive Zahl übertragen werden.

z.B.: - 12,6 °C => - 126 => 1126

```
DevAddr: 26 0B 78 1B Payload: { bytes: [5,230], temperatur: 51 } 05 E6 FPort: 1  
DevAddr: 26 0B 78 1B  
DevAddr: 26 0B 78 1B Payload: { bytes: [5,229], temperatur: 50.9 } 05 E5 FPort: 1  
DevAddr: 26 0B 78 1B  
DevAddr: 26 0B 78 1B Payload: { bytes: [5,228], temperatur: 50.8 } 05 E4 FPort: 1
```



LORAWAN-WORKSHOP

» Payload

```
int temp = 1000 + round( temperatur * 10 );  
mydata[0] = temp >> 8;  
mydata[1] = temp & 0xFF;
```

Der konvertierte Zahlenwert 1194 wird dann in zwei Bytes aufgeteilt:

0000 0101 1101 1011	1499 als Binärzahl
>> 8	
0000 0000 0000 0101	=> obere 8 Bit in mydata[0]
0000 0101 1101 1011	1499 als Binärzahl
& 0xFF	
0000 0000 1101 1011	=> untere 8 Bit in mydata[1]
mydata[]: 0x05 0xDB	

LORAWAN-WORKSHOP

» Payload-Decoder

Formatter code*

```
1 function Decoder(bytes, port) {
2
3   |   var decode = {};
4
5   |   decode.temperatur = ( ((bytes[0] << 8) | bytes[1]) / 10 ) - 100;
6   |   decode.temperatur = round( decode.temperatur );
7
8   |   decode.bytes = bytes;
9
10  |   return decode;
11  | }
12
13  function round(num) {
14  |   var m = Number((Math.abs(num) * 100).toFixed(15));
15  |   return Math.round(m) / 100 * Math.sign(num);
16  | }
```



LORAWAN-WORKSHOP

» Payload-Decoder

Beispiel: 0x05DB

0000 0101

<< 8

0000 0101 0000 0000

1101 1011

|

0000 0101 1101 1011

byte[0] = 0x05

byte[1] = 0xDB

=> Temperatur = (1499 - 1000) / 10 = 49,9 °C

Formatter code*

```

1  function Decoder(bytes, port) {
2
3      var decode = {};
4
5      decode.temperatur = ( ((bytes[0] << 8) | bytes[1]) - 1000 ) / 10;
6
7      decode.temperatur = round( decode.temperatur );
8
9      decode.bytes = bytes;
10
11     return decode;
12 }
13
14 function round(num) {
15     var m = Number((Math.abs(num) * 100).toFixed(15));
16     return Math.round(m) / 100 * Math.sign(num);
17 }
18

```



LORAWAN-WORKSHOP

» Applikations-Server



	Gateways	RSSI	Payload	
18.12.2024 19:39:24	1	-72	05 C2	
18.12.2024 19:38:16	1	-71	05 C2	
18.12.2024 19:37:09	1	-77	05 C3	
18.12.2024 19:36:02	1	-69	05 C4	
18.12.2024 19:34:55	1	-69	05 C4	
18.12.2024 19:33:48	1	-68	05 C4	
18.12.2024 19:32:40	1	-71	05 C5	
18.12.2024 19:31:33	1	-68	05 C5	
18.12.2024 19:30:26	1	-71	05 C6	
18.12.2024 19:29:19	1	-71	05 C6	
18.12.2024 19:28:11	1	-69	05 C7	
18.12.2024 19:27:04	1	-69	05 C7	
18.12.2024 19:25:57	1	-72	05 C8	
18.12.2024 19:24:50	1	-71	05 C8	
18.12.2024 19:23:42	1	-67	05 C9	
18.12.2024 19:22:35	1	-77	05 C9	
18.12.2024 19:21:28	1	-68	05 CA	



LORAWAN-WORKSHOP

» Applikations-Server

18.12.2024 15:33:52



entschlüsselte Payload:

0x05 0xDB

dekodierte Payload:

```
stdClass Object  
(  
    [bytes] => Array  
        (  
            [0] => 5  
            [1] => 219  
        )  
    [temperatur] => 49.9  
)
```

Datenrate:

Spreading Faktor	SF 7
Bandbreite	125 kHz
QRG	867.5 MHz
Sendezeit:	0.046336s

Device-EUI: eui-70b3d57ed004acab

Gateways:

Gateway-Id	RSSI	SNR	Locator	
mikrotik-ltab-ii	-71	7.25	48.820537672214	9.3818340973403 X

06 | Online-Workshop

Mitmachen:

- auf der Mailingliste anmelden
- Materialliste folgt noch auf www.amateurfunk.de/lora



```
ttgo-node-einfach-3 | Ardui
ttgo-node-einfach-3
*/
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>
#include "SSD1306Wire.h" // ThingPulse OLED
SSD1306Wire display(0x3c, SDA, SCL); // Display-Anschlü

// This EUI must be in little-endian format, so least-significant-byte
// first. When copying an EUI from ttncnl output, this means to revers
// the bytes. For TTN issued EUIs the last bytes should be 0xD5, 0xB3,
// 0x70.
static const u1_t PROGMEM APPEUI[8]={ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
void os_getArtEui (u1_t* buf) { memcpy_P(buf, APPEUI, 8);}

// This should also be in little endian format, see above.
static const u1_t PROGMEM DEVEUI[8]={ 0x8A, 0x5A, 0x04, 0xD0, 0x7E, 0x
void os_getDevEui (u1_t* buf) { memcpy_P(buf, DEVEUI, 8);}

// This key should be in big endian format (or, since it is not really
// number but a block of memory, endianness does not really apply). In
// practice, a key taken from ttncnl can be copied as-is.
// The key shown here is the semtech default key.
static const u1_t PROGMEM APPKEY[16] = { 0x76, 0x33, 0xB9, 0x9A, 0x19,
void os_getDevKey (u1_t* buf) { memcpy_P(buf, APPKEY, 16);}

static uint8_t mydata[] = " ";
static osjob_t sendjob;
```



Vielen Dank für Eure Aufmerksamkeit!

73 de Jürgen, DL8MA

Für Fragen stehe ich gerne zur Verfügung.

dl8ma@amateurfunk.de