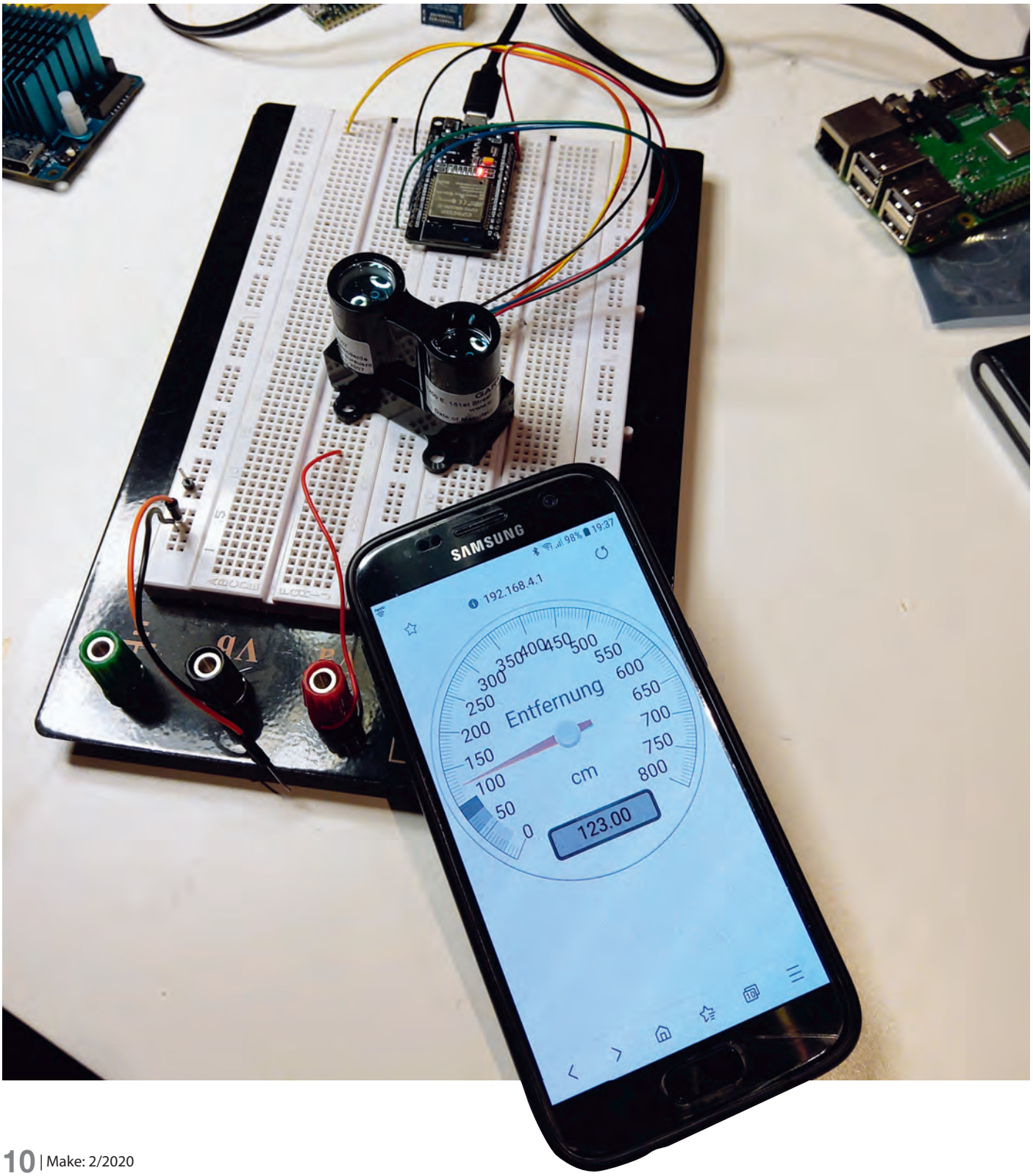
Laserafstandsmeter

Of het nu gaat om het concreet meten van een afstand of om bewaken van een afstand: met onze eenvoudig te na te bouwen webgebaseerde laserafstandsmeter kun je afstanden op de centimeter nauwkeurig bepalen en weergeven op pc's, smartphones of tablets.

door Daniel Bachfeld



Afstandssensoren op basis van infrarood of ultrasoon geluid zijn goedkoop, maar meestal vrij onnauwkeurig, relatief langzaam en hun bereik is minder dan 4 meter. Als je verder, sneller en nauwkeuriger wilt meten, kun je gebruik maken van een LIDAR (laser imaging detection and ranging), die je in een handomdraai in je eigen projecten kunt integreren dankzij kant-en-klare bibliotheken en interfaces die door iedere microcontroller worden ondersteund. In ons voorbeeld gebruiken we een ESP32 met on board wifi om de lidar-meetwaarden grafisch beschikbaar te maken via een zelfgeprogrammeerde Simpel-webserver, zodat elke browser de gegevens kan ophalen en weergeven – ongeacht het apparaat. Een korte introductie in de werking van een lidar is te vinden in het kader "LIDAR-technologie".

De hier gebruikte lidar (1) is afkomstig van Garmin, die ook bekend staat als fabrikant van navigatiesystemen en smart watches. De module heeft in totaal 6 pinnen: Vcc (rode kabel), Power Enable (oranje), Mode (geel), SCL (groen), SDA (blauw) en GND (zwart). Vcc verwacht een spanning tussen 4,5 en 5,5 volt, die de ESP32 op zijn pin Vin levert. Normaal gesproken wordt Vin gebruikt om een spanning te leveren aan de spanningsregelaar van de ESP32 als deze niet via de usb-poort wordt gevoed. Hier gebruiken we de poort als output omdat deze (via een beveiligingsdiode) is aangesloten op de 5V-voeding van de usb-poort.

Pinnen

We kunnen de *Power Enable* en *Mode* pinnen negeren en hoeven ze niet aan te sluiten. Ze zijn al intern voorzien van pull-up-weerstanden zodat ze ons niet in de weg zitten. *SDA* en *SCL* zijn de lijnen van de I2C-bus, waarover de lidar met 400kHz zijn gegevens stuurt of commando's ontvangt. De aansluiting van de ESP32 DevKit met de lidar is weergegeven in (2).

Garmin biedt een bibliotheek voor de Arduino IDE (LIDARLite.h), die handige functies bevat voor het initialiseren, configureren en opvragen van de meetwaarden. Je kunt de bibliotheek downloaden via Github (zie link), en daarna makkelijk integreren met zip-import via Schets > Bibliotheek gebruiken > Voeg .ZIP bibliotheek toe. Zo kun je snel een eerste sketchvoor de lidar ontwerpen met slechts enkele regels, zoals te zien is in Listing lidarlite.ino. De sketch spreekt voor zich, maar we moeten erbij opmerken dat je met de functie `myLidarLite.configure()` verschillende modi kunt instellen, waarbij de lidar beter werkt op korte of lange afstand of langzamer en nauwkeuriger meet.

Korte info

- Wat is LIDAR
- Aansluiten op de ESP32
- Programmeren van de webserver

Checklist

Tijdsduur:
ongeveer 1 uur

Kosten:
ongeveer 150 euro

Programmeren:
Gebruik van de Arduino-IDE

Meer over dit onderwerp online
make-magazin.de/xznk

Materiaal

- Garmin Lidar Lite v3
- ESP32-Board Dev Kit
- evt. breadboard
- Jumperkabels
- Smartphone
- USB-powerbank

lidarlite.ino

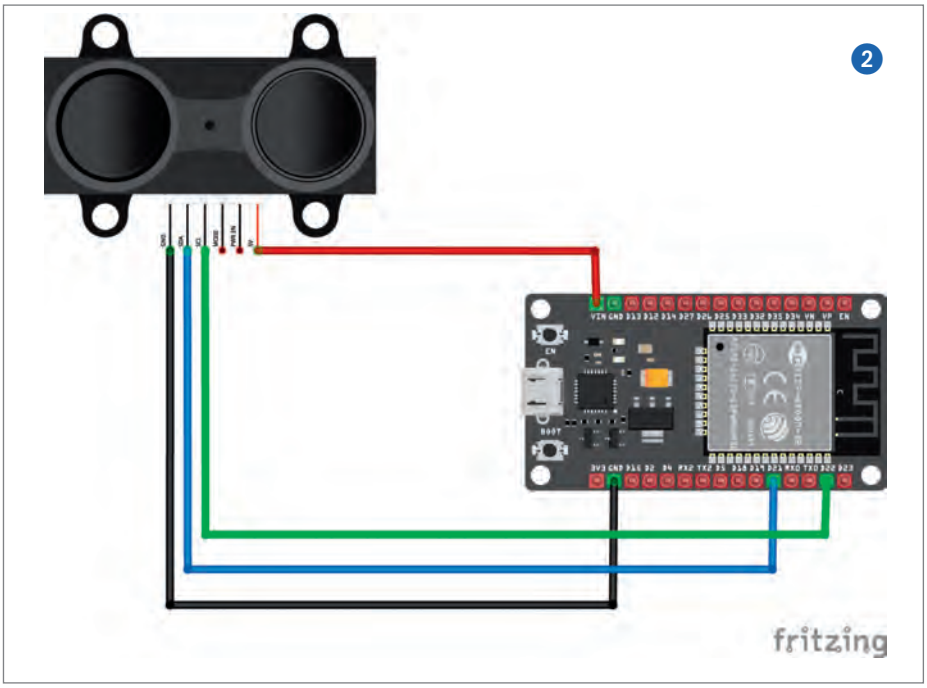
```
#include <Wire.h>
#include <LIDARLite.h>

LIDARLite myLidarLite;

void setup()
{
  Serial.begin(115200);
  myLidarLite.begin(0, true);
  myLidarLite.configure(0);
}

void loop()
{
   Serial.println(myLidarLite.distance());
}
```

(1)



Webserver

Het korte voorbeeld is natuurlijk niet voldoende voor een artikel in Make. Daarom gebruiken we de mogelijkheden van de ESP32 en voegen we een webserver en een access point toe aan de korte sketch, zodat de gegevens niet alleen via de seriële poort worden uitgevoerd. De listing lidar-webserver.ino toont de sketch. Ondanks het kleine aantal regels slaagt hij erin om een access point op te zetten, een webserver te starten en de lidar-gegevens grafisch weer te geven. De magie zit verborgen achter de bibliotheken Wifi.h, ESPAsyncWebServer.h en SPIFFS.h.

lidar-webserver.ino

```
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <SPIFFS.h>
const char* SSID = "MijnESP32";
const char* PASSWORD = "testpassword";
#include <LIDARLite.h>
LIDARLite myLidarLite;
AsyncWebServer server(80);

void setup(){
  Serial.begin(115200);
  if(!SPIFFS.begin(true)){
    Serial.println("Bestandssysteem kon niet geinitialiseerd worden.");
    return;
  }
  myLidarLite.begin(0, true);
  WiFi.softAP(SSID, PASSWORD);
  Serial.print("IP-adres: ");
  Serial.println(WiFi.softAPIP());
  server.on("/gauge.min.js", HTTP_GET, [](AsyncWebServerRequest* request) {
    request->send(SPIFFS, "/gauge.min.js");
  });
  server.on("/", HTTP_GET, [](AsyncWebServerRequest* request) {
    request->send(SPIFFS, "/index.html", String(), false, replaceVariable);
  });
  server.begin();
}
void loop() {}
String replaceVariable(const String& var) {
  if (var == "AFSTAND")
    return String(myLidarLite.distance(false));
  }
```

Wifi.h handelt alles af rond de wifi. Met ESPAsyncWebServer.h kun je snel een multi-session webserver bouwen. Daarvoor moet je behalve de al genoemde bibliotheken ook de AsyncTCP-bibliotheek toevoegen aan de Arduino IDE. Ze staan op Github (zie link) en worden ook als zip-bestand geïmporteerd – zoals de lidar-bibliotheek.

Bestandssysteem

Als je de sketch zorgvuldig bestudeert, kun je je afvragen waar de gauge.min.js en index.html bestanden die vanaf de server worden aangeroepen, zich bevinden. Behalve de tot nu toe gebruikte bibliotheken wordt aan het begin van het programma ook de bibliotheek SPIFFS (Serial Peripheral Interface Flash File System) geïmporteerd. Deze is na de installatie van de ESP32-ondersteuning beschikbaar en maakt het mogelijk om delen van het flashgeheugen van de ESP32 als bestandssysteem te gebruiken en daar bestanden aan te maken, te lezen en te wijzigen. De bibliotheek wordt geïnitialiseerd in de `setup`-functie met het aanroepen van `SPIFFS.start`. Maar hoe komen de bestanden in de ESP32 terecht?

Om het geheugen ook vanuit de Arduino IDE te kunnen gebruiken en beschrijven, gebruik je de tool "Arduino ESP32 Filesystem Uploader". Deze breidt de IDE uit met de mogelijkheid om willekeurige bestanden naar het flashgeheugen van de ESP32 te kopiëren. Om dit te doen, moet je de plug-in downloaden (zie link) en deze in de Arduino-map in de submap tools op je pc zetten (vergeet niet om de IDE opnieuw op te starten). Nu zie je een nieuw item in het menu Hulpmiddelen "ESP32 Sketch Data Upload".

Dit zal de bestanden index.html (zie listing index.html) en gauge.min.js van de pc naar de ESP32 uploaden. Beide bestanden moeten zich bevinden in de map van de sketch lidar-webserver.ino in de submap data.

In plaats van de bestanden met al hun rommel in de broncode in te bedden, leest de ESP32 ze nu gewoon uit het flashgeheugen met behulp van de SPIFFS-klasse. In dit geval bepaalt de AsyncWebServer-bibliotheek automatisch het type inhoud. Het bestand gauge.min.js bevat een JavaScript inclusief bibliotheek, waarmee grafische weergaven zoals thermometers of snelheidsmeters in de browser worden weergegeven. Het bestand index.html stelt de parameters voor de weergave in, zoals de elementgrootte, het waardebereik, de eenheden, de schaling en de labeling, en roept vervolgens het JavaScript op. De placeholder `%AFSTAND` wordt door het webserverprogramma (om precies te zijn door de klasse `AsyncWebServerRequest`) vervangen met de lidar-meetwaarden.

Ingebruikname

Nu alle bibliotheken en tools zijn geïnstalleerd, de bestanden via de uploader van het bestandssysteem naar de ESP32 zijn geüpload en de sketch zonder fouten is gecompileerd en eveneens is geüpload, kun je je smartphone of tablet oppakken en verbinding maken met het wifinetwerk "MyESP32" (wachtwoord testpassword). In de browser typ je het adres http://192.168.4.1 en dan zou je een pagina moeten zien zoals in 3. De inhoud wordt elke 3 seconden bijgewerkt en toont de gemeten afstand een keer als decimaal getal en een keer als kilometerstand. Als je geen pagina te zien krijgt, kan het helpen om de mobiele dataverbinding op de smartphone uit te schakelen. Om te controleren of de sketch naar behoren werkt en een wifinetwerk heeft opgezet, kun je de meldingen ook via de seriële monitor op de pc laten weergeven.

Over weergeven gesproken, de parameters voor de weergave van de snelheidsmeter kunnen eenvoudig worden aangepast in het bestand index.html. Als je een grotere weergave wilt hebben, verander dan gewoon `data-width='1000'` en `data-height='1000'`. Als je meetwaarden van boven de 800 centimeter verwacht, verander dan `data-max-value="800"` en pas de regel `data-major-ticks="0,50,100,150,200,250,300,350,400,450,500,550,600,650,700,750,800"` aan overeenkomstig de nieuwe indeling. De wijzigingen voer je uit in het bestand op de pc om het vervolgens met de uploadtool weer naar de ESP32 te uploaden. Meer ideeën voor gauges zijn te vinden op http://canvasgauges.com/documentation/examples/.

Als je geen mobiele versie van de afstandsmeter nodig hebt, kun je de ESP32 ook gewoon vanaf een vaste locatie in het lokale wifi inloggen. Hoe dat moet is te zien in de voorbeelden van de Arduino IDE. De rest van de code blijft hetzelfde. *–dab*

index.html

```
<!doctype html>
<html>
 <head>
  <title>ESP32 Afstandsmeter</title>
  <meta http-equiv="refresh" content="3" >
 </head>
 <body>
  <canvas id='lidar'
        data-width='1000'
         data-height='1000'
         data-type='radial-gauge'
         data-title='Afstand'
         data-units='cm'
         data-max-value="800"
         data-major-ticks="0,50,100,150,200,250,300,350,400,450,500,550,600,650,700,750,800"
         data-value="%AFSTAND%'>
  </canvas>
   <script src='gauge.min.js'>
  </script>
 </body>
</html>
```

LIDAR-Technologie

Om de afstand tot een object te meten, zendt de laser in het eenvoudigste geval een lichtpuls uit en meet een processor de tijd totdat hij de gereflecteerde puls weer ontvangt (Time of Flight, ToF). Op basis van de tijd en de bekende lichtsnelheid berekent de processor nu de afstand. Omdat het licht een snelheid heeft van 300.000 km/s, heb je goede meetelektronica nodig om de doorlooptijden in het pico- of nanoseconde bereik te meten, afhankelijk van de afstand. Tot een paar jaar geleden kostte zo'n module enkele honderden euro's en was gewoonweg te duur voor de meeste hobbyprojecten.

Ongeveer 4 jaar geleden kwamen de eerste betaalbare "LIDAR Lite" modules op de markt voor ongeveer 100 euro, met behulp van een vereenvoudigd concept dat in silicium kan worden gegoten op een goedkopere manier. Simpel gezegd zendt de laser continu gemoduleerd licht uit in de meervoudige pulsmodus en meet de faseverschuiving tussen de uitgezonden en ontvangen pulsen. Dit komt eveneens overeen met een tijd en is dus evenredig met de afstand. Sinds deze vereenvoudiging zijn lidars ook te vinden in veel hobby-projecten rond robotica en drones om obstakels op tijd te herkennen.

De hier gebruikte module Lidar Lite v3 van fabrikant Garmin kost bijna 130 euro en is door zijn lage gewicht (22g) en kleine afmetingen (20 x 48 x 40 mm) geschikt voor mobiele toepassingen. Het huidige verbruik is met 135mA niet bepaald laag, maar toch acceptabel voor gebruik onderweg. Het maximale bereik is 40m met een fout van +/-2.5cm. Per seconde kan de lidar tot 500 metingen verzamelen en via zijn I2C-bus naar een microcontroller sturen.