UWB-Project documentatie

Gemaakt door: Donny Verhof

Bedrijf: Dent IoT.

# Samenvatting

Dit document bestaat uit een uitgebreide uitleg van de software (in C) die is gebruikt voor de UWB-hardware chips en de pythoncode die is gebruikt om de hardware te testen. Alle resultaten van de hardware kunnen worden opgeslagen in Excelsheets mits er een goede internet connectie is. (In de aanbevelingen wordt er dieper op ingegaan).

De hardware bestaat uit geïntegreerde chips met een ESP32 als microcontroller. De ESP32 verwerkt de data die binnenkomt uit de DW1000 chip en verwerkt deze zo tot een gemeten afstand. Een gemeten afstand is altijd tussen een Anchor en Tag. Afstand meten tussen 2 Anchors zonder tag is dus niet mogelijk.

Na het meten van (minimaal) 3 afstanden kan er een x- en y-coördinaat worden bepaald van de plaats waarin de Tag zich op dat moment bevindt.

[De C-code](../src) bestaat uit een speciale library gemaakt voor het UWB-component wat op de hardware chips van de Anchors en Tag zitten. De hardware van Tag en anchor zijn hetzelfde alleen de Software wat erop geflasht moet worden verschild. Voor een uitgebreide uitleg over het flashen en gebruiken van dit project check de README van deze directory.

De pythoncode bestaat uit de verwerking van de gemeten afstanden. Of de x-y berekening op de chip gebeurt of via de PC kan ingesteld worden in dit bestand (Settings.py). Check deze paragraaf voor een uitgebreide uitleg van de gehele pythoncode.

Het resultaat van deze 2 codes geïntegreerd is een tag die zijn afstanden of coördinaten verstuurd en een pythonscript die alles bijhoudt in excelsheets. Door alles op te slaan in excelsheets kan er duidelijk bewezen worden wat de accuratie en precisie is van de afstanden gemeten door de tag. Bekijk de paragraaf aanbevelingen als er interesse is deze pythoncode verder te gebruiken voor het opslaan van de testresultaten.

**Geen inhoudsopgavengegevens gevonden.**

# C-code

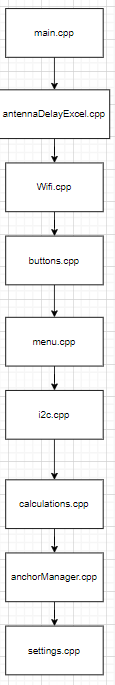
Deze paragraaf bestaat uit een uitleg van de gebruikte C/C++ code en de opbouw van de main met zijn toegevoegde libraries. In de code zelf staan ook nog comments voor een beter begrip van de werking.

## 1.1 Opbouw

De code is opgebouwd uit een main die een zelfgemaakte library aanroept deze library file roept weer een andere library aan en dit gaat zo door tot de laatste aangeroepen library Settings.cpp

De libraries zijn zo ingedeeld dat de integratie goed verloopt. Er zijn echter positieve aanpassingen te maken in de opbouw van dit document. Kijk hier om deze aanpassingen te overzien.

In de onderstaande afbeelding staat de volgorde van de libraries (verander deze volgorde niet als je geen ervaring hebt in C-programmeren!).



Afb. 1: Opbouw libraries C-code.

De code roept ook de DW1000.h library aan. Deze library zorgt voor de communicatie tussen de ESP32 en DW1000 chip. In de main.cpp staan de instellingen van deze communicatie al goed ingesteld. Dit betreft de SPI-interface. Het is ook niet aan te raden deze code te wijzigen als er geen ervaring is in de programmeertaal C/C++.

## 1.2 [Main.cpp](../src/main.cpp)

De main verwerkt alle outputs van de toegevoegde libraries. Het verkrijgen van informatie van de DW1000 wordt in functies afgehandelt en de verwerking van deze verkregen informatie.

Als eerst wordt de setup functie aangeroepen.

Deze functie stelt de onderstaande opsomming goed in:

* Alle pins van de ESP32 zijn goed ingesteld.
* Opent de communicatie tussen de ESP en DW1000 via SPI.
* Maakt een connectie via WiFi (zorg dat de ID en wachtwoord goed zijn ingesteld).
* Een afstandsmeet methode is ingesteld (voor afstandsmeting via de DW1000).
* Antenna Delay is ingesteld. Voor uitgebreide info over deze variabele check(link maken naar hoofdstuk).

Het verkrijgen van info wordt verkregen in de functie newRange() en is een loop-functie.

Deze functie wordt aangeroepen wanneer er een connectie is gemaakt met minimaal 1 Anchor. Er is een filter toegepast die pas data gaat processen wanneer er minimaal 3 anchors zijn gevonden.

Als de bovenstaande onderstreepte voorwaarde waar is worden functies in de toegevoegde libraries aangeroepen. De volgorde van functies komt overeen met de volgorde van libraries in afb. 1.

In de volgende subparagrafen wordt er uitgebreider uitgelegd hoe alle setup- en loop functies worden behandeld.

## 1.3. [antennaDelayExcel](../src/antennaDelayExcel.cpp)

Deze library reageert op de gevonden afstanden en stopt deze waarden in de juiste variabelen van de anchorstruct. Deze variabelen in de struct worden blijven opgeslagen en worden in andere libraries weer verwerkt. Als de is wifi aangesloten zal er in deze library na volledige meting worden gewacht op de request die komt vanuit de python-code.

Na deze request worden de nieuwe afstanden gemeten en worden de bovenstaande functies herhaald.

## 1.4 [Wifi](../src/Wifi.cpp)

Dit bestand bestaat uit meerdere delen:

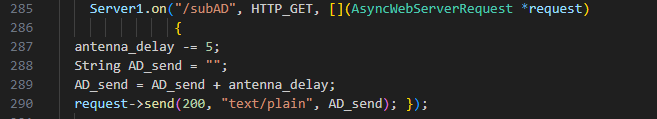
* Verbinding maken met wifi.
* Servers om instellingen van de UWB-tag draadloos te veranderen.
* Server om de data die gemeten is door te sturen naar een PC.

### Verbinding maken met wifi

In dit deel wordt de wifi-communicatie opgesteld (mits deze mogelijk is). De code blijft doorgaan of er nu wel of niet een verbinding is gemaakt met de router. De juiste SSID en wachtwoord moeten worden ingevuld in de bovenste regels van het bestand. Er kan ervoor gekozen laten worden om de ESP als Accespoint te gebruiken, maar wordt niet aanbevolen in deze toepassing.

### Servers om instellingen van de UWB-tag draadloos te veranderen:

Er zijn Aparte Server interrupts gemaakt om de instellingen die in de C-code worden gebruikt te kunnen veranderen. Deze instellingen kunnen ook handmatig worden veranderd via het gebruik van de knoppen. In de code staan met comments duidelijk wat elke instelling doet. Om een voorbeeld te geven van zo’n Server functie staat er in de onderstaande afbeelding een voorbeeld van zo’n functie.



Deze server kan worden gebruikt om de Antenna delay van een van de anchors te veranderen.

Om de juiste anchor te kiezen moet het juiste IP adres worden ingevuld voor de naam van de server.

Als voorbeeld:

* Het IP van Anchor#1 = 192.164.2.42.
* De server die aangeroepen moet worden = “/subAD”

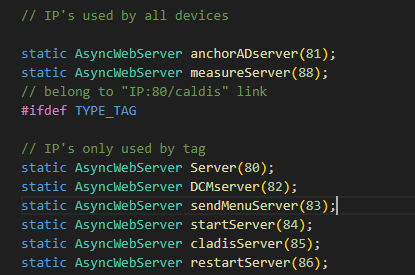
De totale link wordt dan (Let op http geen https!).

<http://192.164.2.42/subAD>.

Er moet dan een getal uitkomen wat de nieuwe antenna delay waarde weergeeft.

### Server om de data die gemeten is door te sturen naar een PC:

Er zijn ook Servers die alleen worden gebruikt om datat te versturen. Het is daarom ook noodzakelijk dat deze servers niet worden aangeroepen door iets anders dan de python-code.



Van al deze servers mogen alleen anchorADserver en measureServer worden gebruikt.

Wat deze servers doen staan in de comments bij de functies.

## 1.5 [Buttons](../src/Buttons.cpp)

Dit programma zorgt ervoor dat de knoppen goed reageren om de menu op de tag te gebruikten.

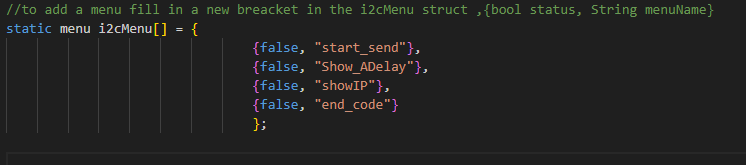
Deze code bestaat uit de interrupt functies die worden aangeroepen van het indrukken van 1 van de 3 knoppen en hiermee kan door het menu gescrollt worden. Door het klikken op de enter-knop kan een menu-functie aangeroepen worden.

Het menu bestaat uit 4 functies en door het aanpassen van [menu.cpp](../src/menu.cpp) kunnen er extra menu’s worden toegevoegd.

## 1.6 [Menu](../src/menu.cpp)

Dit programma wordt gebruikt om menu’s toe te voegen die kunnen reageren op de knoppen.

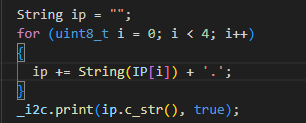
Om een menu toe te voegen kan er een extra array element worden toegevoegd.



## 1.7 [i2c](../src/i2c.cpp)

Dit programma regelt de library voor het kleine schermpje op de UWB-microcontrollers.

In de main moet eerst de settings functie worden aangeroepen en erna kan er simpel geprint worden door gebruik te maken van:



Voobeeld van het printen van een string.

Er moet altijd een const char\* worden ingestopt bij print. Dit kan worden gedaan door een String aan te roepen en deze String om te zetten met behulp van de .c\_str() functie.

## 1.8 [calculations](../src/calculations.cpp)

In dit programma worden alle berekeningen uitgevoerd om van de afstanden naar x-y coordinaten te gaan.

Ook staan hier berekeningen om test afstanden te verkrijgen vanuit zelf gekozen y-coordinaten. Hierdoor kan er goed getest worden op accuratie en precisie.

## 1.9 [Settings](../src/Settings.cpp)

Hier staan alle instellingen van het hele programma in. Ook staan hier alle belangrijke constanten die worden gebruikt in alle hiervoor gebruikte programma’s

Er is duidelijk met comments aangegeven wat elke optie/constante doet in dit programma.

## 1.10 Conclusies

De code werkt of zonder wifi goed waarbij hij alleen serieel verstuurd. Of alleen met wifi waarbij hij niet serieel verstuurd. Met beide onderdelen erin is er een integratiefout.

De code werkt, maar is niet netjes verdeeld in header en source files. Integratie van de code zal beter verlopen wanneer de code op die manier is ingedeeld.

## 1.11 Aanbevelingen

Los de integratiefout van Wifi en serieel op. Hierdoor kan je met of zonder gebruik van de python code de afstanden via wifi gebruiken voor de user interface.

Een andere manier is om de info vanuit de boot op te vragen (die serieel binnenkomt van de tag) en van deze info de user interface op te zetten.

# Python-code

In dit onderdeel wordt alle python-code uitgelegd. De python-code wordt gebruikt voor het opslaan van meetwaarden die in de tag zijn berekend. Hierdoor kon er duidelijk getest worden hoe accuraat en precies de tag was op welk moment.

Deze code kan nog gebruikt worden in toekomstige iteraties als het gewenst is de accuratie en precisie te verhogen.

## 2.1 [Main.py](../Python-code/main.py)

De main roept alle library setups aan die worden gebruikt in de loop van de main.

De main roept de volgende onderdelen aan:

* Excelinstellingen.
* Seriële instellingen (optioneel).
* Wifi-instellingen (voer de juiste IP-adressen in).
* Instellingen om te tekenen.
* Maak alle objecten aan die gebruikt worden als instellingen.

Erna wordt er een loop in gegaan die wacht op een nieuwe waarde van de tag. Als dit het geval is wordt deze waarde verwerkt in excel en ingetekend via draw.

## 2.2 Request.py

Dit programma regelt alle Wifi-instellingen en hierdoor is een request makkelijk aan te roepen in de main. Elke request geeft een string terug die in de main verder verwerkt zal worden.

Ook komt er een hoop data binnen van de tag. Deze data wordt vooral gebruikt om de coördinaten te weten van de anchors aan de kanten van het zwembad.

De belangrijkste request is de ‘/anchors’ request. Deze geeft namelijk de afstanden van alle anchors.

In [triliteration.py](../Python-code/trilateration.py) worden de x-en y waarden berekend. Deze worden ook berekend in de tag zelf, maar het is beter deze niet te versturen, zodat de berichten die verstuurd worden over wifi zo klein mogelijk zijn.

## 2.3 xls.py

Dit programma handelt alle instellingen voor excel en maakt nieuwe excelsheets wanneer deze gewenst zijn in de main. Er kan een antenna delay test te doen. Of alle metingen kunnen worden bijgehouden.

### Antenna delay test

Deze test geeft de tag een startpunt met antenna delay. En blijft deze verhogen met een stap, totdat het maximum is bereikt. Hierdoor kan er goed worden gezien, wat de beste antenna delay is op een coordinaat in het zwembad. Dit kan handig zijn voor het kalibreren van de tag.

## 2.4 draw.py

Dit programma zorgt voor het tekenen van een 3d-as van de gevonden coördinaten. Helaas waren de 3d berekeningen niet accuraat op de z-as. Het wordt dus aangeraden draw uit te zetten via [Settings.py](../Python-code/settings.py)

## 2.5 Settings.py

Dit is het belangrijkste bestand voor het gebruiken van de Python-code. Hiermee kunnen alle onderdelen aan of uit worden gezet, lay-out van excel worden aangepast, en de IP-adressen worden ingevuld.

Er staat duidelijk gecommend wat alle onderdelen inhouden.

## 2.6 Conclusies

Er zitten integratie fouten in waar geen tijd meer voor was om op te lossen.

De python code werkt goed in een omgeving waar goede wifi aanwezig is en de afstanden tussen router en UWB-module niet te groot zijn.

In het zwembad kwamen veel requests laat of niet aan.

## 2.7 aanbevelingen

Naar mijn mening kan het zeer handig zijn deze code over te nemen of deels te gebruiken voor het ontwerpen van een mooie user interface.

Het is zeker handig aangezien het doel van de UWB is om de boot te testen dat het excel-gedeelte wordt gebruikt. Dan zullen alle toekomstige testen worden opgeslagen in bestanden en kan er na de test goed bekeken worden wat er precies is gebeurd.

Er waren ook problemen met de Wifi communicatie in het zwembad. Als oplossing zou er een nieuwe sterke router gebruikt kunnen worden of het python programma laten runnen via de PC van het aqualab (de PC met meerdere schermen). Dan zou er visueel getest kunnen worden, maar worden de waarden die uit te test komen ook automatisch opgeslagen.