Jochem van Kranenburg, 1741991, 3-7-2019

Jochem.vankranenburg@student.hu.nl

[](https://github.com/JVKran/Portable-Radio)Afbeelding met elektronica, binnen

Automatisch gegenereerde beschrijving

plan van aanpak

IPASS 2018-2019

Inhoud

In dit verslag zal besproken worden wat de plannen zijn en waren, en hou deze zijn uitgevoerd. Daarnaast zullen keuzes worden toegelicht en zal de software en hardware worden besproken.

Inhoud

[Hardware 2](#_Toc13158366)

[Library 2](#_Toc13158367)

[Applicatie 3](#_Toc13158368)

[Risicobeheersing 3](#_Toc13158369)

[Code 4](#_Toc13158370)

[Testrapport 4](#_Toc13158371)

[Schakelschema 5](#_Toc13158372)

# Hardware

In de basis zou ik gebruik gaan maken van een Arduino Due, TEA5767 (<https://bit.ly/2vt4BRC>), 3W 4 Ohm speakers en PAM8403 (<https://bit.ly/2PfxWKx>) om een FM-radio te maken. Dit deel omvat dan ook de Library en is naar meer dan tevredenheid gelukt. Daarnaast zou ik, indien de tijd het toelaat, de radio uit gaan breiden met een 0,96 inch SSD1306 (<https://bit.ly/2gYB8r0>) (of later 1,77inch ST7735 Display (<https://bit.ly/2LbwXum>)), KY040 Rotary Encoder, 24LC256I2C EEPROM IC (<https://bit.ly/2YKlYtH>), ST1089 IR receiver (<https://bit.ly/2DCbBA4>), Bluetooth Audio Module (gesloopt uit USB dongle), 18650 met TP4056 (<https://bit.ly/2fv19Od>) in combinatie met zonnepaneel (5V, 250mA) en step up converter en daarnaast nog een 3.5mm audio jack output of input voor externe speaker.

Uiteindelijk is dit deels gelukt. Ik heb de OLED met HWLIB gebruikt om een interface te maken voor de applicatie waarin een deel van de ondersteunde functionaliteiten getoond worden. Ik heb echter geen gebruik gemaakt van de ST7735 aangezien ik graag dicht bij de overige code, en natuurlijk ook HWLIB, wil blijven. Daarnaast heb ik ook de Rotary Encoder (zowel KY-040 als EC11) en 24LC256 EEPROM chip geïmplementeerd. De ST1089 heb ik ook achterwege gelaten met de reden die ook bij de ST7735 van toepassing is.

In plaats van de ST7735 en ST1089 heb ik gekozen voor de RDA58XX Series en DS3231 Real Time Clock. De RDA5807M is een nieuwere FM-chip die beter werkt dan de TEA5767. Daarnaast ondersteund deze het Radio Data System; een korte zoekopdracht toonde aan dat er niet veel zijn die alle ontvangen gegevens zo netjes uitpluizen als de door mij geschreven library; hier ben ik dan ook best trots op.

Tot slot heb ik ook nog de stroomvoorziening kunnen regelen; TP4056, MT3608, Zonnepaneel in combinatie met LM2596 zodat de speakers op vol vermogen kunnen draaien (~2A piek). Helaas zorgt het feit dat twee I2C bussen op één Arduino Due niet mogelijk is ervoor dat de speakers ruis gaan uitspugen (gegenereerd door I2C communicatie).

Wat betreft demo projecten is [dit](https://bit.ly/2vtk1Fo) de meest uitgebreide.

# Library

De library is bedoeld voor de TEA5767. Dit is een FM-ontvangstmodule die wordt aangestuurd via de I2C bus. De module bestaat uit vijf 8-bits registers die de volgende zaken besturen:

* Wel of geen output
* Wel of geen zoekmode
* Mono of stereo output
* Links wel of geen output
* Rechts wel of geen output
* Standby mode
* Noise cancelling
* Een aantal synthesizer opties.

Wat betreft methoden en inhoud van klassen vermoed ik één klasse te hebben met een aantal private attributen betreffende stereo/mono, frequentieband beperkingen en huidige frequentie. Verder verwacht ik voor alle hierboven genoemde instellingen een functie te hebben waarmee deze instelling kan worden uitgezet of toegepast. Daarnaast zullen er functies beschikbaar moeten zijn die de statussen van deze instellingen op kunnen vragen.

Hier verwacht ik dan ook dat de complexiteit van de library zit; tamelijk groot en niet altijd even triviaal. De datasheet zal een zeer groot aantal vragen moeten beantwoorden. Daarnaast wilde ik aan het begin van IPASS het project graag kunnen overhevelen naar een ESP32 (of ander bord met WiFi capabiliteit) voor eventuele extra wifi-functionaliteit (en het totaalplaatje) met webserver op de tweede core. Daarom zou ik het, indien de tijd het toelaat, leuk vinden de library naast dat het compatibel is met hwlib/bmptk ook compatibel te maken met Arduino. Uiteindelijk heb ik dat ook geprobeerd; het bleek te eenvoudig om überhaupt leuk te zijn. Zodoende heb ik dit plan uiteindelijk achterwege gelaten.

Indien het zo uit komt dat ik ook gebruik maak van de KY-040 Rotary Encoder en 24LC256, was ik van plan ook voor deze onderdelen een library te schrijven. Dit is dan uiteindelijk ook gebeurd.

Aangezien er voor vier chips een library is geschreven verwijs ik voor gedetailleerde overzichten van classes, methoden en attributen graag door naar de (uitgebreide) ingeleverde [Doxygen Documentatie](https://github.com/JVKran/Portable-Radio/tree/master/Documentation) en [voorbeelden](https://github.com/JVKran/Portable-Radio/tree/master/Examples). Daarnaast bevatten alle libraries duidelijke uitleg over gebruikte principes; niet iedereen zal bekend zijn met PLL, High side injection, RDS en Page-boundaries. De comments en Doxygen Documentatie zouden hier heel erg mee moeten helpen.

# Applicatie

Als eindresultaat staat er een radio met interface op een OLED display. Door middel van de Rotary Encoder kan er worden genavigeerd door de menu’s en kunnen instellingen worden aangepast. Zo kan men bijvoorbeeld wisselen tussen bluetooth en FM-radio, Bass-Boost in- en uitschakelen, decodering van Radio Data in- en uitschakelen en kiezen tussen presets, automatisch zoeken of handmatig zoeken. De presets kunnen worden opgeslagen in het EEPROM geheugen zodat de presets met naam (verkregen via het Radio Data System) en frequentie (verkregen via de RDA5807) worden behouden tussen power-cycles door. Daarnaast kan men automatisch zoeken via de ingebouwde zoekfunctie; per gevonden zender wordt dan eenmalig de naam via het Radio Data System opgevraagd. Dit gebeurt eenmalig omdat de gegevens erg storingsgevoelig zijn en het decoderen erg veel tijd en pakketten kost. Rechtsboven in het scherm zijn de signaalsterkte en het batterijpercentage zichtbaar. Linksboven is de tijd zichtbaar en onderin of er sprake is van Stereo ontvangst of Mono ontvangst.

Het zonnepaneel levert in de zon maximaal 250mA; opladen zou +- 10 uur duren. Dit is verre van ideaal, maar toch ook erg leuk. De ~3,7V wordt door de MT3608 omgevormd tot 12V wat via de Vin pin op de Arduino Due wordt aangeleverd. De versterker is via de TP4056 rechtstreeks aangesloten op de 18650 om de maximale stroom van 2A mogelijk te maken (de TP4056 is voor langdurig gebruik gewaardeerd op max 2,5A).

De LM2596 verzwakt de spanning van het zonnepaneel (13V – 0V) tot 5V. De spanning zou gemeten moeten worden door middel van een spanningsdeler zodat deze spanning via een mosfet geschakeld zou kunnen worden om de TP4056 te kunnen beschermen. Helaas heeft mijn mislukte poging alles te solderen (door een defect geleverd onderdeel) ervoor gezorgd dat ik dit niet meer heb kunnen regelen.

# Risicobeheersing

Het project is in de volgende stappen doorlopen:

1. Na afloop van het lezen van de datasheet het met behulp van de datasheet schrijven van de library op de manier waarvan wordt vermoed dat deze werkt.
2. Library fine-tunen zodat de minimale functionaliteit (wisselen tussen en zoeken van zenders) betrouwbaar en goed werkt.
3. Schakeling maken voor de speakers met versterker zodat de library voor de TEA5767 eenvoudig en goed kan worden getest (2x3W 4Ohm speakers en PAM8403).
4. Overige functionaliteiten in de library implementeren (Stereo/Mono, Links/Rechts, Noise Cancelling).
5. Library debuggen, testen, edge cases afvangen en van documentatie voorzien.
6. SSD1306 OLED met rotary encoder toevoegen zodat de radio kan worden gedemonstreerd en instellingen aan kunnen worden gepast.
7. Totale werking testen.

Het minimale eindresultaat (Arduino Due met PAM8403, Speakers, TEA5767, Rotary Encoder en OLED) zou hier behaald moeten zijn. Voordat ik verder ga, zal ik eerst alles dubbel en dwars langslopen om te kijken of alles aan de eisen voldoet, kwalitatief goed is en er geen betere oplossing voor opgedane problemen zijn. Indien dat het geval is, ga ik verder met het volgende:

1. 24LC256 toevoegen zodat zenders en instellingen tussen stroomonderbrekingen door aanwezig kunnen blijven en er niet telkens opnieuw gezocht hoeft te worden naar zenders.
2. Bluetoothmodule (enkel hardware-matig) toevoegen. Er kan tussen audio-inputs worden gewisseld door middel van een transistor, mosfet of andere schakelaar (softwarematig).
3. De schakeling afmaken met TP4056, 18650 en zonnepaneel. Indien alles werkt wordt alles gesoldeerd op één of meerdere protoboards.

Het maximale eindresultaat voor de Arduino Due (Arduino Due met PAM8403, Speakers, TEA5767, Rotary Encoder, OLED, 24LC256, Bleutoothmodule, TP4056, 18650 en Zonnepaneel) zal hier behaald zijn. Vervolgens is hetvolgende nog gedaan:

1. DS3213 implementeren en testen.

Aangezien er heel stapsgewijs wordt gewerkt en alles voortbouwt op het eerder gedane werk, zal er hoe dan ook een eindresultaat zijn wat bestaat uit een radio, FM-ontvanger en speaker. Aangezien vorige projecten ook met ESP’s, displays, rotary encoders en accu’s is gewerkt, verwacht ik daar geen problemen mee. Ik kan me wel verkeken hebben op de library. Indien de library lastiger is dan ik vermoed, zal de library in de basis betrouwbaar maar niet uitgebreid zijn. Ik denk dan ook dat daar de prioriteit ligt. Al met al vermoed ik dat er met het hierboven beschreven stappenplan niet veel fout kan gaan. Indien er tijd over is, en de kwaliteit van het geleverde werk goed is, kan ik op tal van manieren verder uitbreiden. Indien het maximale eindresultaat wordt bereikt, hoop ik alle onderdelen op een protoboard te kunnen solderen. De ESP32 zou dan op female header-pins geklikt kunnen worden zodat met zowel de ESP32 als de Arduino Due dezelfde hardware kan worden gebruikt tijdens de demonstratie.

# [Code](https://github.com/JVKran/Portable-Radio)

De code is door middel van de hier bovenstaande link te bereiken. Hier is ook de definitieve schakeling te vinden.

# Testrapport

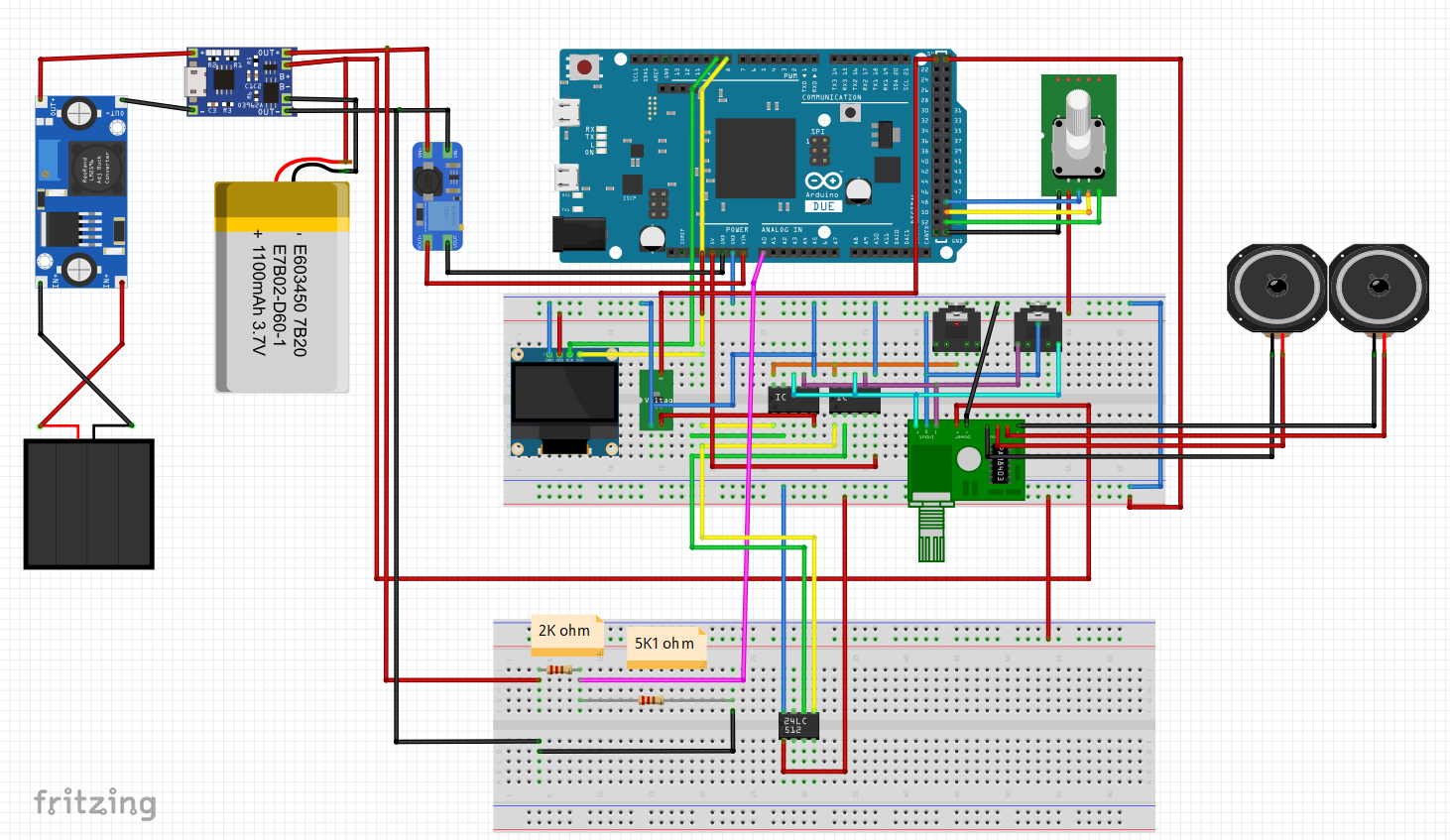
Alle Libraries zijn uitbundig getest. Tijdens het testen bleek dat veel functies niet waterdicht waren. Tijdens het testen zijn deze herschreven zodat ze voorkomen dat er dingen gebeuren die niet horen te gebeuren. Zodoende kan de kwaliteit worden gewaarborgd.

Daarnaast zijn alle individuele tests ook gecombineerd tot één test zodat alle Libraries eenvoudig kunnen worden getest als alles met elkaar verbonden is. Alle libraries zijn zodoende getest dat wat men verwacht ook datgeen is wat gebeurt. Zo is het niet mogelijk een tijdobject te initialiseren wat meer dan 23 uur, 59 seconden en 59 minuten heeft. Ook is het niet mogelijk een EEPROM object aan te maken wat een geheugen heeft dat geen macht van 2 is. Daarnaast kan men ook niet schrijven naar een stuk geheugen wat er niet is en worden page-boundaries keurig afgehandeld zodat data niet wordt overschreven. Daarnaast is er rekening gehouden met interchangability tussen de RDA5807M en TEA5767; ze hebben een abstracte basisklasse. Er is geprobeerd alle functies zo triviaal mogelijk, maar toch ook flexibel te houden. Hier zijn ze dan ook op getest.

Voor het gemak en de compleetheid hebben alle libraries ook een enkele functie waarmee kan worden getest of de chips correct werken en reageren of commando’s verstuurd via I2C.

In de applicatie is een debug modus aanwezig waarmee in de terminal kan worden afgedrukt welke acties ondernomen worden op de achtergrond en wat de gebruiker doet.

# Schakelschema

Onderstaand is de schakeling van het eindresultaat weergegeven. De gebruikte hardware is reeds besproken en uiteengezet.

Schakelschema Eindresultaat