**BM2 - Project Embedded Systems**

**TempusLudicus**

A round wooden board with colorful lights

Description automatically generatedA small electronic device with a screen

Description automatically generated*Productrapport*

Embedded Systems Engineering  
Academie Engineering en Automotive  
Hogeschool van Arnhem en Nijmegen

**Auteurs**

|  |  |
| --- | --- |
| 2218888 | Kevin van Hoeijen |
| 633992  2107391 | Maarten van Riel  Roel van Eeten |
| 1659237 | William Hak |

Begeleider

|  |
| --- |
| **Hugo Arends** |

**Datum 17 Januari 2024**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Documenthistorie**   |  |  |  |  | | --- | --- | --- | --- | | **Datum** | **Versie** | **Wie** | **Veranderingen** | | 19-11-23 | v0.1 | Maarten | Invulling functioneel ontwerp | | 20-11-23 | v0.2 | Kevin | Hoofdstuk interface toegevoegd met UART, US en schakelaars | | 22-11-24 | V | Kevin | Toevoeging hoofdstuk software en software diagram | | 28-11-23 | v0.3 | Maarten | Invulling Inleiding | | 23-11-23 | v0.4 | Maarten | Invulling temp in technisch ontwerp | | 23-11-23 | v0.5 | Roel | Invulling Technisch ontwerp | | 01-01-24 | v0.6 | Roel | Invulling Technische specificaties | | 06-01-24 | v0.7 | Kevin | Hoofdstuk software aangevuld met instel/debug software | | 13-01-24 | v0.8 | Maarten | Invulling voorwoord/samenvatting | | 13-01-24 | v0.9 | Roel | Invulling software | | 14-01-24 | v1 | Roel | Invulling software/invulling zelfreflectie | | 14-01-24 | V1.1 | Kevin | Zelfreflectie en uitwerken testrapport | | 14-01-24 | V1.2 | Maarten | Invulling Conclusie/zelfreflectie | | 14-01-24 | V1.3 | William | 7.1.5 LED strip en 7.1.8 voeding.  Bijlagen hardware schema en pcb design, 3d view.  Updated pinout schematic.  Updated pinout picture of temp sensor lm35 | | 15-01-24 | V2.0 | William  Kevin  Maarten  Roel | Gezamenlijke algemene revisie alle paragrafen. | | 16-01-24 | V2.1 | William | Specs US geüpdatet  led strip technisch ontwerp en realisatie update  Standaard tekst rood gemarkeerd  Fixed heading style  Technische eisen aangepast  Figuren toegevoegd plus figuurnummer/omschrijving | | 16-01-24 | V2.2 | Kevin | Laatste versie acceptatietesten | | 17-01-24 | V2.2 | Kevin | Update softwarediagram | |

Voorwoord

*Geachte lezer,*

Voor u ligt het productrapport “TempusLudicus” in het kader van ons project.

Dit rapport is het resultaat van ons project "Embedded Systems", een uitdagende opdracht gericht op het creëren van een unieke klok. Deze opdracht, onderdeel van onze studie, werd mogelijk gemaakt door de HAN en stond onder leiding van onze docent Hugo Arends. Deze ondersteuning en begeleiding waren cruciaal voor de ontwikkeling van ons project.

Terugkijkend op de gehele periode, hebben wij als team zowel uitdagingen als overwinningen ervaren. De samenwerking binnen onze groep was een leerzame ervaring, waarbij communicatie en het verdelen van de werkzaamheden/taken centraal stonden. Er waren keren dat we moesten schakelen en slim moesten denken, wat ons liet inzien hoe belangrijk het is om flexibel besluitvaardig te blijven bij het leiden van een project. Daarnaast werd ons ook duidelijk dat het realiseren van een project in teamverband, valt of staat met het borgen van afspraken.

Dit project leerde ons veel over samenwerken, technische skills en hoe we problemen kunnen oplossen. We ontdekten dat we ons beheer van de code moeten verbeteren en meer up-to-date moeten houden. Dit gaan we in de toekomst zeker aanpakken. Alles wat we hebben geleerd en ervaren, gaat ons zeker helpen in onze verdere carrière.

We willen even de tijd nemen om een dikke pluim te geven aan iedereen die een steentje heeft bijgedragen aan dit project. Aan alle teamleden en de begeleider, jullie zijn geweldig! Zonder jullie inzet, kennis, en soms eindeloos geduld (met name het stoeien met de Git), was dit project echt niet zo cool geworden als het nu is. Bedankt dat jullie dit avontuur met ons aangingen!

Dit rapport is geschreven voor veel verschillende lezers, van medestudenten en leraren tot experts op het gebeid van “Embedded systemen”. We hebben ons best gedaan om het makkelijk leesbaar te maken, maar ook vol met technische informatie. Zo bieden we inzicht in onze strategie en geven we technische details.

We hopen dat dit rapport laat zien hoe hard we hebben gewerkt en dat het anderen inspireert en informeert.

*Gorinchem, januari 2024*

*Maarten van Riel*

*Roel van Eeten*

*Kevin van Hoeijen*

*William Hak*

Samenvatting

*Tijdens het eerste contactmoment van de les “project Embedded Systems” werd ons kenbaar gemaakt dat wij, voor onze opdracht, een klok diende te realiseren binnen een bepaalde periode. Tevens bepaalde de opdrachtgever (in dit geval de docent) de verplichte uitgangspunten voor deze klok en welke vrij in te vullen waren.*

*De verplichte uitgangspunten betroffen onder andere dat de tijdwaarneming plaatsvond aan de hand van de Unix timestamp[[1]](#footnote-2). Daarnaast moet de klok de datum en de tijd op twee manieren weergeven. Eén van de manieren was via een display en de tweede manier was op een niet voor de hand liggende manier en gebruik maken van tenminste twee actuatoren.*

*Wij hebben enkele uitgangspunten zelf bepaald. Natuurlijk waren de ambities veel groter dan de beschikbare tijd en diende uiteindelijk wij genoegen te nemen met een haalbaar ontwerp. Wij hebben als doel gesteld om genoegen te nemen met een minder uitdagend en goedkoper product, zodat wij het project binnen de gestelde tijd kunnen opleveren.*

*Een van de uitgangspunten was dat de klok de dag/datum en tijd gaat weergeven. Dit zou afgebeeld worden aan de hand van verticale LED-strips. Aan beide zijden van de LED-strips komen LED-strips in een curve die de dag van de week en de datum weergeven. Onderaan de klok komt een LCD waarop eveneens de dag/datum en tijd worden weergegeven. Naast het LCD komen twee drukknoppen, een ultrasoon sensor en een temperatuursensor.*

*Hierna werd een verdeling gemaakt van de programmeertaken. Hierbij heeft iedere student ten minste één programmeer onderdeel toegewezen gekregen. Elk stukje code werd met behulp van het programma “GIT” samengevoegd tot één geheel. In het begin was één lid van het team verantwoordelijk voor het samenvoegen van de code op de “GIT”. Het gevolg hiervan was dat de “GIT” niet actueel was en dat de leden op de samenvoeging wachten en niet door konden met programmeren. Dit was niet de fout van de beheerder van de “GIT” maar van ons als team, na constatering werden de taken beter verdeeld.*

*Elke donderdagavond werden tijdens de les knelpunten besproken en het project geëvalueerd. Indien nodig maakten wij nieuwe afspraken gemaakt of verfijnde deze. Na enkele weken kwamen wij tot de conclusie dat het onze voorkeur had om een actiepuntenlijst te maken. Hierin werden afspraken vastgelegd en nog veel belangrijker, het was een goede geheugensteun omdat sommige leden de taken één dag weer vergeten waren... Daarnaast werkt een actiepuntenlijst als een soort van deadline voor de volgende les.*

*Uiteindelijk hebben wij een deadline gesteld wanneer de software gereed diende te zijn. Dit werd de laatste lesdag in december. Hierna werd een aanvang gemaakt om de klok in elkaar te zetten zodat een aanvang gemaakt kon worden om de soft- en hardware te testen.*

*Terugkijkend op het project zijn alle uitgangspunten behaald. Het ontbrak bij enkele leden van de werkgroep aan actuele kennis van zaken, om als team aan een “Embedded Project” te werken. Met behulp van de docent werden wij geregeld aan het denken gezet, waardoor tijdig problemen werden voorkomen.*

Inhoudsopgave

[Voorwoord 3](#_Toc156428691)

[Samenvatting 4](#_Toc156428692)

[1 Inleiding 8](#_Toc156428693)

[1.1 Achtergrond 8](#_Toc156428694)

[1.2 Opbouw 8](#_Toc156428695)

[2 Functioneel ontwerp 9](#_Toc156428696)

[2.1 Functionele eisen 9](#_Toc156428697)

[2.2 Technische eisen 14](#_Toc156428698)

[2.3 User interface 15](#_Toc156428699)

[3 Technisch ontwerp 16](#_Toc156428700)

[3.1 Architectuur 16](#_Toc156428701)

[3.2 Deelsystemen en interfaces 17](#_Toc156428702)

[3.2.1 Microcontroller 17](#_Toc156428703)

[3.2.2 Ultrasoon sensor 17](#_Toc156428704)

[3.2.3 Schakelaars 20](#_Toc156428705)

[3.2.4 LCD 21](#_Toc156428706)

[3.2.5 Led strip 24](#_Toc156428707)

[3.2.6 UART-Update en debug interface 26](#_Toc156428708)

[3.2.7 Temperatuursensor 27](#_Toc156428709)

[3.3 Software 29](#_Toc156428710)

[4 Realisatie 30](#_Toc156428711)

[4.1 Hardware 31](#_Toc156428712)

[4.1.1 Ultrasoon sensor 35](#_Toc156428713)

[4.1.2 Schakelaars 38](#_Toc156428714)

[4.1.3 LCD 44](#_Toc156428715)

[4.1.4 LED-strip 47](#_Toc156428716)

[4.1.5 Temperatuursensor 50](#_Toc156428717)

[4.1.6 Voeding 51](#_Toc156428718)

[4.1.7 UART-Update en debug interface 52](#_Toc156428719)

[4.2 Software Implementatie 53](#_Toc156428720)

[4.2.1 Keil μVision IDE 53](#_Toc156428721)

[4.2.2 QT Creator 54](#_Toc156428722)

[4.2.3 Ophalen van timestamp van server 55](#_Toc156428723)

[4.2.4 Debug interface 58](#_Toc156428724)

[4.2.5 Ultrasoon sensor 60](#_Toc156428725)

[4.2.6 Buttons 61](#_Toc156428726)

[4.2.7 Main functie 62](#_Toc156428727)

[5 Testen 65](#_Toc156428728)

[5.1 Acceptatietesten 65](#_Toc156428729)

[6 Conclusies en aanbevelingen 66](#_Toc156428730)

[7 Bronvermelding 67](#_Toc156428731)

[9 Bijlage 1 – Testscenario’s 68](#_Toc156428732)

[9.1 Testscenario 1 – Weergeven van de tijd 68](#_Toc156428733)

[9.2 Testscenario 2 – Extra functies testen 69](#_Toc156428734)

[9.3 Testscenario 3 – Instellen en tonen van debug informatie. 71](#_Toc156428735)

[9.4 Testscenario 4 – Meten van stroomverbruik. 73](#_Toc156428736)

[10 Bijlage 3 Pinout 74](#_Toc156428737)

[11 Bijlage 4 Zelfreflectie 79](#_Toc156428738)

[11.1 Zelfreflectie Roel 79](#_Toc156428739)

[11.2 Zelfreflectie Kevin 79](#_Toc156428740)

[11.3 Zelfreflectie Maarten 80](#_Toc156428741)

[11.4 Zelfreflectie William 81](#_Toc156428742)

[12 Bijlage 5 – Hardware Schema’s 82](#_Toc156428743)

[13 Bijlage 6 – PCB-design 84](#_Toc156428744)

[14 Bijlage 7 – 3d View 85](#_Toc156428745)

# Inleiding

Het project " TempusLudicus " omvat het programmeren van een ARM microcontroller in de taal C, met als hoofddoel het weergeven van de dag, datum en tijd op twee unieke manieren. Dit hoofdstuk introduceert de basisconcepten en de technische aspecten van ons project.

## Achtergrond

Dit rapport gaat dieper in op het programmeerproces van de microcontroller, een kernonderdeel van onze klok. We hebben verschillende componenten geïntegreerd om een functionele en visueel aantrekkelijke tijdweergave te creëren. De hoofdelementen van ons ontwerp omvatten:

**Microcontroller:** Het brein van onze klok, geprogrammeerd om alle onderdelen te coördineren en de tijd nauwkeurig weer te geven.

**RGB strips:** Gebruikt voor een dynamische en kleurrijke weergave van de tijd en datum.

**Ultrasone sensor:** Een innovatieve toevoeging die interactieve mogelijkheden biedt.

Temperatuursensor: Biedt extra functionaliteit door de actuele temperatuur weer te geven.

**LCD:** Een traditionele, maar heldere weergave van de tijd en datum.

**Houten Plaat:** Dient als de stevige basis waarop alle componenten zijn gemonteerd, wat bijdraagt aan zowel de esthetiek als de stabiliteit van de klok.

Elk van deze componenten speelt een cruciale rol in het functioneren en de uitstraling van de klok. In de volgende paragrafen zullen we de selectie, integratie en programmering van deze onderdelen in detail bespreken.

## Opbouw

De opbouw van dit rapport bestaat uit de volgende hoofdstukken:

* In hoofdstuk 2 is het functionele ontwerp te lezen. Het functioneel ontwerp beschrijft de technische en de functionele eisen, hierin is beschreven welke producten zijn gebruikt en hoe de klok werkt.
* In hoofdstuk 3 is het technisch ontwerp beschreven. Hierin is beschreven hoe de technische oplossingen zijn ontworpen. Aan de hand van een diagram is te zien hoe de architectuur van de klok is opgebouwd. Daarnaast is beschreven hoe de deelsystemen en de software-architectuur zijn vormgegeven.
* In hoofdstuk 4 is de realisatiefase en de testfase beschreven. In de realisatiefase is beschreven hoe de hardware en software ontwerpen tot realisatie zijn gebracht. Aan de hand van voorbeelden en schema's wordt de realisatie in details uitgelegd.
* In hoofdstuk 5 worden de testresultaten beschreven.
* In hoofdstuk 6 wordt afgesloten met het eindresultaat en enkele aanbevelingen. Er wordt teruggekeken naar het project en daarmee wordt er een conclusie beschreven.
* In hoofdstuk 7 worden de bronvermeldingen weergegeven.
* Aan het einde van het rapport bevindt zich de bijlagen, de bijlagen bevat onder andere de gebruikershandleiding van de Klok en de pin-bezetting van de microcontroller.

## 2. Functioneel Ontwerp

Dit document beschrijft hoe de Datalogger wordt ontworpen en welke functionele en technische eisen eraan gesteld worden. Het ontwerp heeft als doel om tijdwaarnemingen met GPS-coördinaten te registreren en de verzamelde gegevens overzichtelijk weer te geven. De datalogger zal real-time informatie tonen en opslaan, met een gebruiksvriendelijke interface voor de bediening.

### 2.1 Functionele eisen

Hieronder staan de functionele eisen geformuleerd volgens het MoSCoW-principe.

| **#** | **Prioriteit (MoSCoW)** | **Specificatie** | **Details** | **Opmerkingen** |
| --- | --- | --- | --- | --- |
| F1.1 | Must (M) | Tijdregistratie | De hoofdfunctie van de datalogger is het registreren van de tijd van gemeten gebeurtenissen. | Nauwkeurigheid is cruciaal. |
| F1.2 | Must (M) | Meetfrequentie | De datalogger moet een meetfrequentie van 10.000 Hz ondersteunen en de gegevens binnen 5 ms na detectie verwerken en opslaan. | Zorgt voor een nauwkeurige tijdregistratie. |
| F1.5 | Must (M) | Opslagcapaciteit | De datalogger moet minimaal 1000 metingen kunnen opslaan in intern geheugen. | Voldoende opslagruimte voor veldmetingen. |
| F1.6 | Must (M) | Detectie | De datalogger moet signalen van een infraroodafstandsdetector kunnen detecteren en verwerken. De detectie moet, Objecten tot een bereik van 200 cm kunnen waarnemen. Binnen 10 ms na een detectie een meetwaarde registreren en opslaan. Een tijdstempel genereren wanneer een object wordt waargenomen en deze opslaan in het geheugen. Een visuele indicatie tonen op het OLED-scherm zodra een object wordt gedetecteerd. Compatibel zijn met standaard IR-sensoren met een analoge of digitale output. | Moet een reflector kunnen detecteren die detectie van een detectielus signaleert. |
| F1.6 | Must (M) | Status opgeslagen metingen | Het scherm toont het aantal opgeslagen metingen (bijvoorbeeld "Samples: 5"). | Helpt de gebruiker bij het beheren van metingen. |
| F1.7 | Must (M) | Opslagbehoud | De opgeslagen data moet behouden blijven, zelfs bij stroomuitval. | Belangrijk voor data-integriteit. |
| F1.8 | Must (M) | Gebruikersinterface (OLED-scherm) | De meetwaarden worden weergegeven op een monochroom OLED-scherm. | Duidelijke en efficiënte weergave. |
| F1.9 | Must (M) | Live tijdsweergave | Rechtsonder wordt de actuele tijdwaarneming weergegeven in milliseconden (HH:MM:SS.mmm) en moet binnen 10 ms geactualiseerd worden. | Real-time visualisatie van metingen. |
| F1.10 | Must (M) | Reflectiepaaltjes registratie | Links worden de nummers 1-8 weergegeven, met een vinkje achter de nummers waarvan de tijd is opgeslagen. | Helpt bij het monitoren van voltooide metingen. |
| F1.11 | Must (M) | Bediening via knoppen | Onderin het scherm worden knoppen weergegeven: STOP, ERASE, NEXT, START. | Fysieke knoppen zorgen voor snelle bediening. |
| F1.12 | Must (M) | Datalog exporteren | De meetgegevens moeten via USB kunnen worden uitgelezen op een PC. | Handig voor verdere analyse. |
| F1.3 | Should (S) | GPS-ondersteuning | De datalogger moet real-time GPS-coördinaten opslaan bij elke meting. | Belangrijk voor locatiegebonden data. |
| F1.4 | Should (S) | Aantal Satellieten | Rechtsboven op het display moet het aantal verbonden satellieten worden weergegeven. | Biedt inzicht in signaalsterkte. |
| F1.13 | Should (S) | Energiebeheer | De datalogger moet een energiezuinige modus hebben die automatisch wordt geactiveerd na X minuten inactiviteit. | Verlengt de batterijduur. |
| F1.14 | Could (C) | Draadloze communicatie | Optioneel kan Bluetooth of WiFi worden toegevoegd voor draadloze data-export. | Handig voor realtime monitoring zonder kabels. |

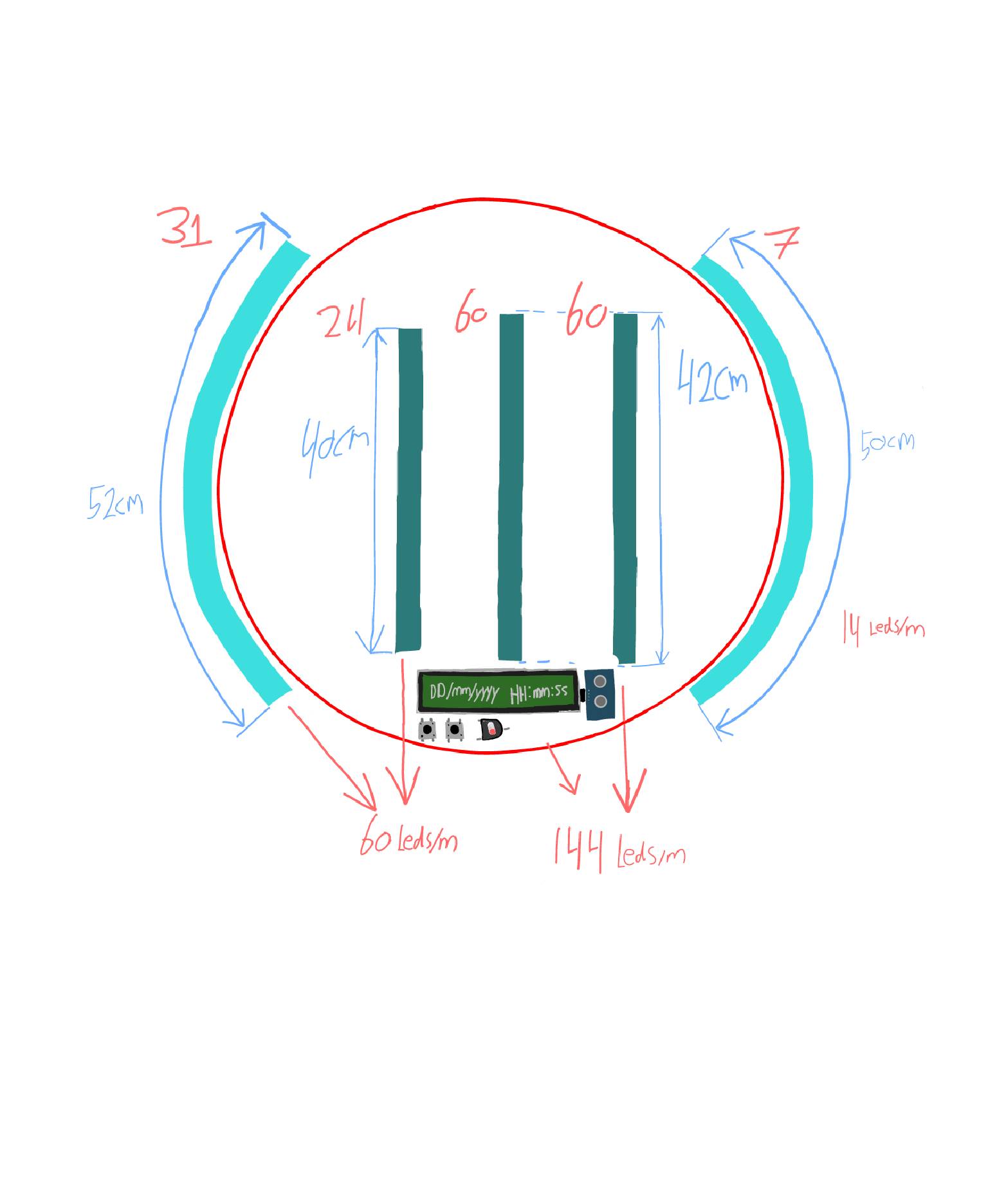
### 2.2 Technische eisen

Hieronder volgen de technische eisen die nodig zijn voor de implementatie van de datalogger.

| **#** | **Prioriteit (MoSCoW)** | **Omschrijving** |
| --- | --- | --- |
| T1 | Must (M) | De datalogger moet worden gebaseerd op de FRDM-KL25Z microcontroller. |
| T2 | Must (M) | De hardware wordt ontworpen als een uitbreidingsmodule (shield) voor de FRDM-KL25Z. |
| T3 | Must (M) | De firmware wordt geschreven in C (C11-standaard). |
| T4 | Must (M) | Het systeem moet via UART of USB data kunnen exporteren. |
| T5 | Must (M) | De datalogger moet worden gevoed door een 5V USB-voeding of batterijpack. |
| T6 | Must (M) | De microcontroller moet een RTC (Real-Time Clock) ondersteunen om tijdregistraties te synchroniseren. |
| T7 | Should (S) | Het OLED-scherm moet een minimale resolutie hebben van 128x64 pixels. |
| T8 | Should (S) | De datalogger moet GPS-gegevens via een NMEA 0183-compatibele module kunnen ontvangen. |
| T9 | Could (C) | Het systeem kan worden uitgebreid met een SD-kaartmodule voor grotere opslagcapaciteit. |

## User interface

De user interface komt er als volgt uit te zien:



Figuur 1 User interface schets

De drie strips in het midden zullen de uren minuten en seconden weergeven respectievelijk. De strip links zal op de platte ronde zijkant komen en naar links schijnen, deze zal de dagen van de maand weergeven. Verder zal de strip rechts eveneens naar rechts schijnen en de weekdagen weergeven.

Onderaan is er een display die de tijd in een wat gangbaarder formaat weergeeft. En 2 knoppen om de klok te bedienen.

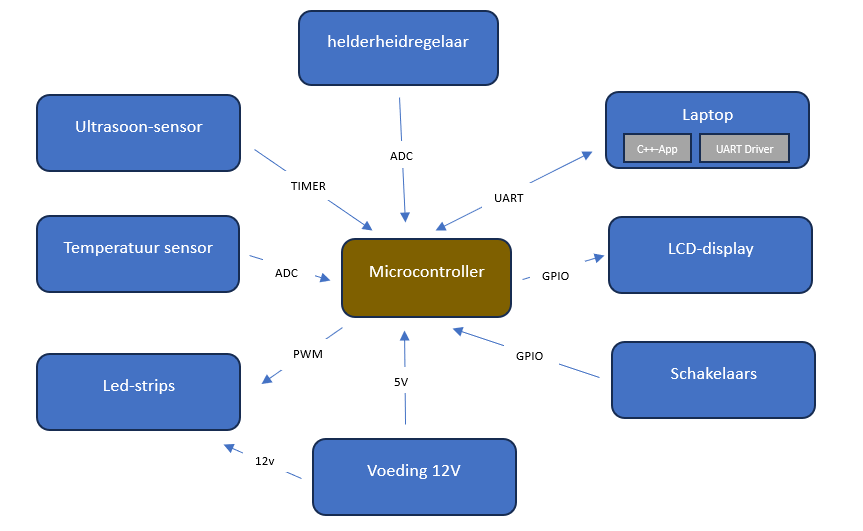
Verder is er naast het LCD een ultrasone sensor te zien, welke gebruikt kan worden voor enkele ‘easter eggs’.

Voor de lichtintensiteit instelling van de strips is er naast de knoppen ook nog een potentiometer de zien. Aan deze knop kan je draaien om de lichtintensiteit te variëren.

# Technisch ontwerp

## Architectuur

De TempusLudicus is een kloksysteem met als hart een microcontroller. Het systeem kan worden onderverdeeld in deelsystemen. Hieronder staat een illustratie van de deelsystemen. De microcontroller communiceert via bepaalde interfaces met deelsystemen. Deze communicatie kan eenzijdig zijn, maar kan ook dubbelzijdig zijn. De deelsystemen zijn aangegeven in de blauwe kaders. De interfaces ofwel communicatiemethoden zijn aangegeven op de pijlen.



Figuur 2 Architectuur diagram

## Deelsystemen en interfaces

Zoals hierboven te zien zijn er verschillende communicatiemethoden gebruikt om de communiceren met de deelsystemen. In dit hoofdstuk worden de verschillende deelsystemen en interfaces beschreven.

### Microcontroller

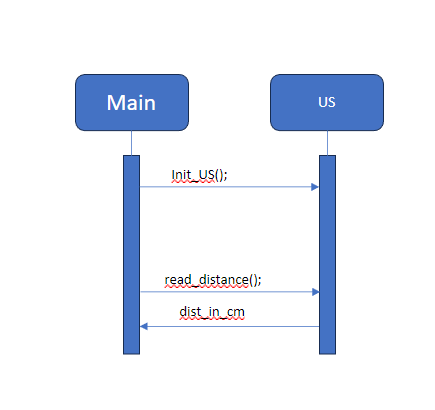
Voor de Microcontroller wordt er een FRDM-KL25Z board gebruikt welke gebruikt maakt van een 3.3v voeding. De microcontroller bevat een breed scala aan peripherals inclusief een MUX, de MUX is er handig bij het kiezen van IO pinnen. De reden hiervoor is omdat 1 functie kan worden geprogrammeerd op verschillende IO pinnen, wat ervoor zorgt dat de hardware een stuk meer flexibel is in het kiezen van pinnen.

De microcontroller heeft 80 pinnen, waarvan er 64 beschikbaar zijn op de headers van het FRDM-KL25Z board. De processor heeft een maximale klok frequentie van 48Mhz.

### Ultrasoon sensor

Voor de ultrasoon sensor wordt een HC-SR04 gebruikt. Deze sensor werkt met digitale signalen en zal de communicatie via GPIO zijn.

De sensor werkt doormiddel van een geluidsgolf. Door bij een frequentie van 10hz een puls van 10us naar de trigger te sturen wordt er een geluidsgolf gegenereerd die niet hoorbaar is voor het menselijk oor. Bij weerkaatsing van het geluid wordt dit opgevangen door de sensor en wordt er een puls terug gestuurd. Door de tijd van die inkomende puls te meten kan de afstand met een formule worden bepaald. De maximale afstand die de ultrasoon sensor kan meten is 400 cm, met een resolutie van 0.3cm. Voor meer info over de ultrasoon sensor zie de datasheet van de HC-SR04.



Figuur 3 Software Interface schema ultrasoon sensor

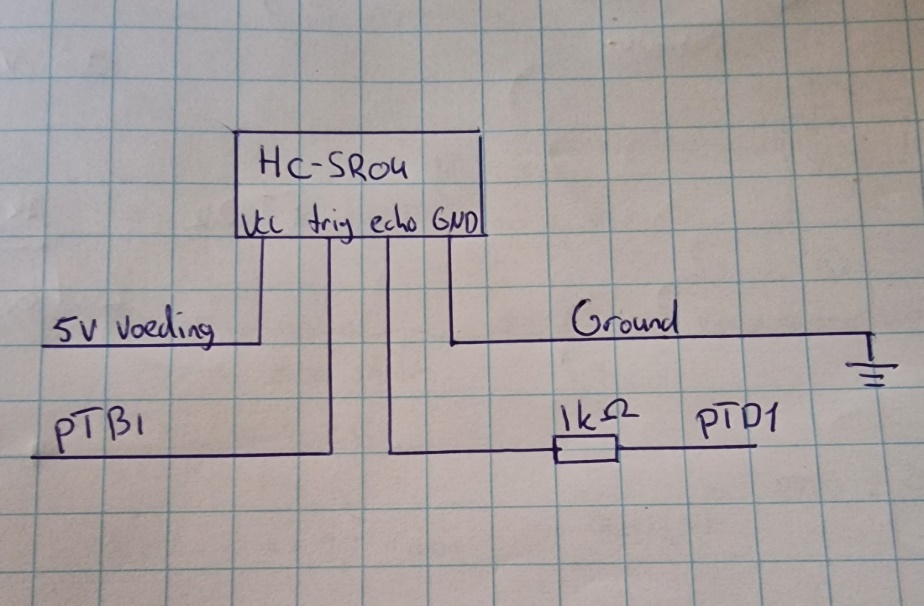
**Specificatie** Er wordt 10 keer per seconde gemeten of er een object voor de klok is.

**Specificatie** De uitgangspanning van de sensor is 5VDC en moet worden gereduceerd voor de ingang naar 3.3V met een weerstand van 1KOhm

**Specificatie** Als er een object voor de klok is wordt de afstand bepaald en verstuurd een functie verstuurt dit als een **int** door aan andere functies.

**Specificatie** In de software wordt de afstand afgerond op 1cm.

Hier volgt het aansluitschema en de pin outs



Figuur 4 Schets aansluitschema ultrasoon sensor

|  |  |  |
| --- | --- | --- |
| Pin sensor | Pin microcontroller | Omschrijving |
| VCC | NC | Externe 5V voeding |
| GND | NC | Ground/aarde |
| Echo | PTD5 | Echo - Geluid ontvangen vanuit sensor |
| Trig | PTA13 | Trigger - Geluid verzenden vanuit sensor |

### Schakelaars

Volgens functionele specificatie F6 zijn er 2 druktoetsen aanwezig. De twee schakelaars aangesloten op de microcontroller op twee input pinnen. Deze inputpinnen worden geïnitialiseerd doormiddel van sw\_init() in de main functie van de microcontroller.

**Specificatie** Beide schakelaars zijn hoog actief:

• Wel ingedrukt, dan is de digitale input van de microcontroller logisch 1 (3.3V).

• Niet ingedrukt, dan is de digitale input van de microcontroller logisch 0 (GND).

Volgens functionele specificatie F7, F8 en F9 moeten er 3 functies beschikbaar zijn bij het indrukken van de schakelaar. Onderstaande functie zorgt voor een reactie bij het indrukken van de schakelaar(s).

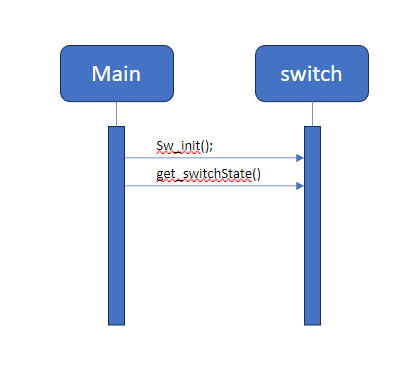
Get\_switchState() geeft de volgende waarden terug:

1. Geen toets ingedrukt.
2. Toets 1 ingedrukt.
3. Toets 2 ingedrukt.
4. Beidde toetsen ingedrukt.

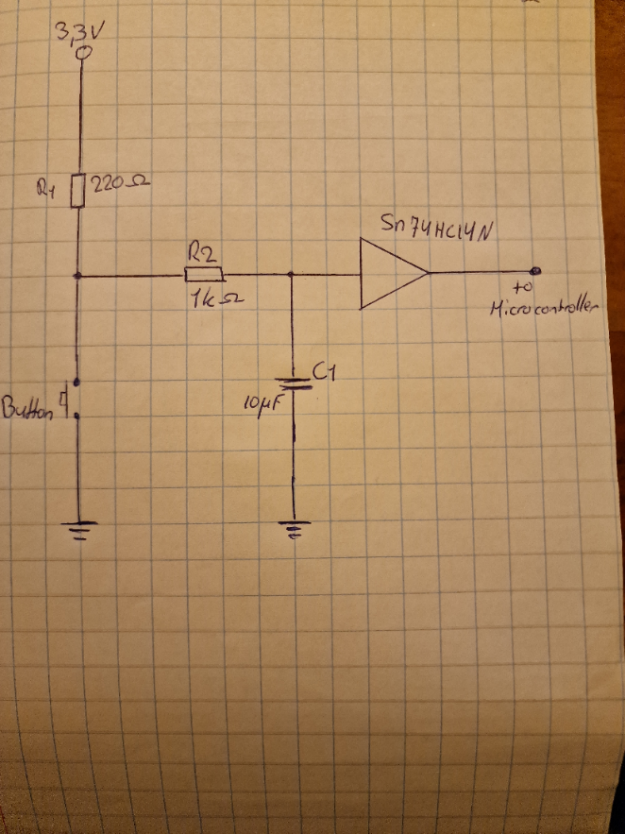
**Specificatie** De schakelaar wordt aangesloten doormiddel van een hardware matige debounce. Zie paragraaf 7.1.2.1 voor informatie over de debounce.

**Specificatie** Schakelaar 1 wordt aangesloten op PTD0 die wordt geconfigureerd als ingang.

**Specificatie** Schakelaar 2 wordt aangesloten op PTD2 die wordt geconfigureerd als ingang.



Figuur 5 Software Interface schema Buttons



Figuur 6 Schets aansluitschema debounce schakeling

### LCD

Volgens de functionele specificatie F1.6 wordt er gebruik gemaakt van een display die de volgende data kan laten zien:

* Laten zien van de tijd.
* Laten zien van de datum.
* Laten zien van tijd tot pensioenleeftijden.
* Laten zien wat de omgevingstemperatuur is.
* Laten zien van mood instelling.
* Laten zien dat de klok in debug modus staat.
* De tijd wordt weergegeven in HH:MM:SS
* De datum wordt weergegeven in YYYY/MM/DD

Hiervoor wordt een 16x2 LCD gebruikt. Tijdens de lessen aan de HAN is er gebruik gemaakt van dit display. Dit zorgde voor de benodigde ervaring en is daarom ook de keuze op dit display gevallen.

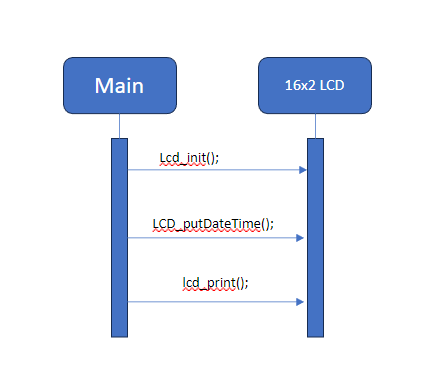
Het geselecteerde display wordt aangestuurd volgens het HD44780U protocol. Via dit protocol kan de microcontroller leesbare tekens laten zien op het display.

Voor de toepassing hiervan worden gebruik gemaakt van 8 digitale output signalen.

Voor meer informatie, zie datasheet van de HD4478U die is meegeleverd met het project.

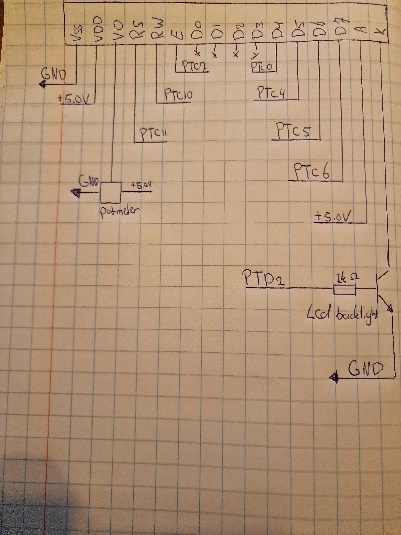
**Specificatie:**

Het display wordt via de functie lcd-init() geïnitialiseerd. Daarnaast is een functie beschikbaar genaamd LCD\_putDateTime(), deze functie zorgt ervoor dat de tijd en datum correct wordt. Weergegeven op het scherm. De lcd\_print() functie kan elke string of char waarde afdrukken op het scherm en is dus inzetbaar op meerdere fronten.



Figuur 7 Software Interface schema LCD

Hier volgt het aansluitschema van de lcd:



Figuur 8 Schets aansluitschema LCD

### Led strip

De tijd wordt onder meer weergegeven door de led strip. Het type LED waar we voor gekozen hebben is de WS2815B. De WS2815B strip, biedt betere fouttolerantie dan bijvoorbeeld de WS2811. Dit komt door de extra back-up datalijn die de WS2815B bevat. Deze extra back-up datalijn zorgt ervoor dat de rest van de LEDs blijven werken wanneer er een LED defect raakt. De manier waarop dit is geïmplementeerd geeft enkele redundantie. Ofwel wanneer 2 pixels direct achter elkaar defect raken dan zullen alle pixel die volgen geen signaal meer krijgen en dus niet functioneren.

De strip is per led instelbaar door middel van het Non-Return to Zero (NRZ) protocol: “In digital communication, NRZ is a method of mapping a binary signal to a physical signal. Unlike other signaling schemes, in NRZ, the signal level does not return to zero (the baseline voltage) between bits. It stays at a high or low level for a specified amount of time, and switches from level at least once per bit, depending on the bit value (1 or 0). (Non-return-to-zero, 2024)

Om dit protocol zo snel en efficiënt mogelijk te laten verlopen is er gekozen om de implementatie te realiseren met de DMA en PWM peripheral van de FRDM-KL25Z. De keuze voor deze 2 peripherals zijn gedaan zodat met minimale CPU-interventie de strips aangestuurd worden. Om de strip aan te sturen moeten de volgende stappen worden ondernomen:

1. DMA schrijft van een buffer met een tijdswaarde, naar een tussen variabele.
2. DMA 2 schrijft van de tussenvariabele naar de timer CnV register.
3. De timer creëert de benodigde puls aan de hand van de waarde in het CnV register

De reden dat er 2 DMA transfers nodig zijn om maar 1 waarde te kopiëren heeft te maken met de architectuur en de mogelijkheden van de peripherals. En de grootste limiterende factor die meespeelt, is het feit dat de DMA peripheral niet een 8 bits waarde kan schrijven in een 16-bits register zonder de count waarde met 2 op te hogen. Wat ongewenst is want dat betekent dat voor elke 8-bits waarde een 16-bits geheugenlocatie benodigd is.

Om dit dus te optimaliseren en alleen geheugen te gebruiken die strikt nodig is hebben we ervoor gekozen om 2 DMA operaties aan elkaar te linken. De DMA peripheral kan namelijk wel een enkele transfer doen van een 8-bits waarde naar een 8-bit deel van het geheugen die daadwerkelijk 32-bits groot is omdat bij 1 transfer geen rekening gehouden hoeft te worden met die Byte count waarde, hebben we hier niet het probleem dat de Byte count waarde incorrect wordt opgehoogd.

Als deze transfer klaar is wordt de 2e transfer geïnitieerd. Deze transfer schrijft de waarde die net in het 32-bits geheugenlocatie is geschreven naar het CnV register.

Deze cruciale optimalisatie zorgt ervoor dat we de minimale geheugengrootte hebben bereikt, en geen geheugenruimte onnodig wordt verspild.

Verder om alle tijdseenheden goed weer te geven, hebben we minimaal 193 led pixels nodig.

**Specificatie** Het formaat is de SMD5050 chip, wat staat voor 5\*5 mm;

**Specificaties per LED-pixel:**

Quiescent stroom: 2.1 mA

Max Stroom verbruik: 15mA

Aantal Bytes aan RAM benodigd: 24 Bytes

**Specificaties voor 193 pixels:**

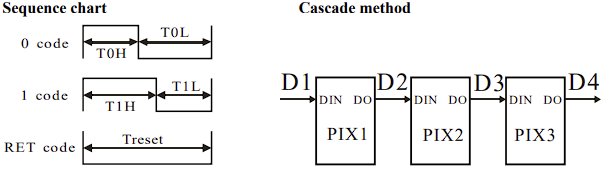
Output pin benodigd: 1 PWM output

Voeding: 9.5 – 13.5 VDC (12v nominaal)

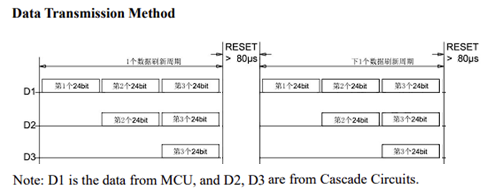
Quiescent stroom 193 pixels: 405.3 mA

Max stroom 193 pixels: 2.89 A

Aantal Bytes aan RAM benodigd: 4632 Bytes



Figuur 9 Schema van de pulse width naar binary encoding



Figuur 10 Schema van het serial cascading protocol van de WS2815B

### UART-Update en debug interface

Volgens functionele specificatie F1.3 moet er gecommuniceerd worden met een PC-applicatie die:

* De UNIX-tijd kan ophalen en kan verzenden.
* Debug informatie kan ontvangen vanuit de microcontroller.

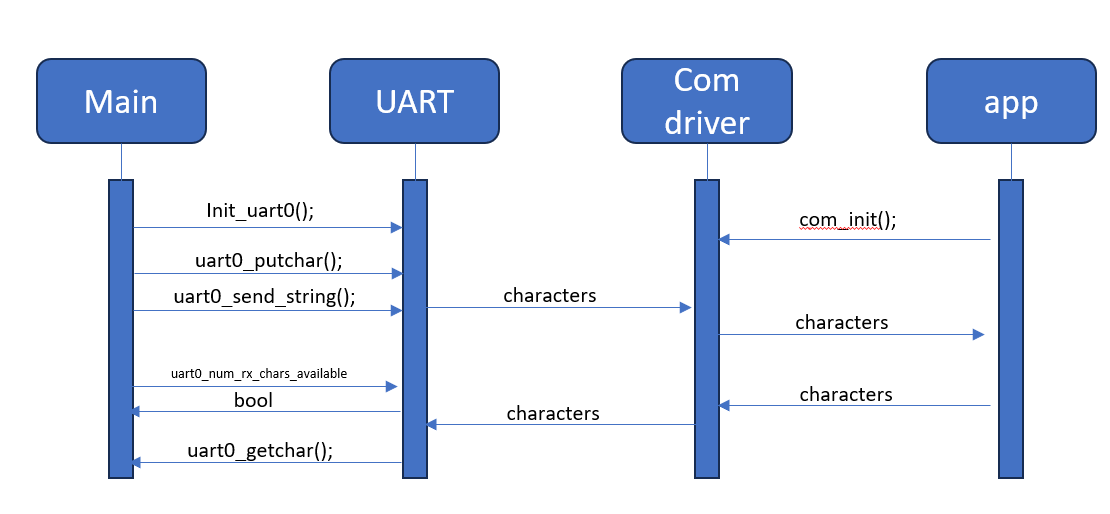
Deze applicatie wordt geschreven in C++.

Er is voor een seriële verbinding gekozen waarbij de communicatie wordt gerealiseerd met USB. De microcontroller communiceert met de PC volgens het UART-protocol.

**Specificatie** Er wordt serieel gecommuniceerd met 9600 bps, 8 data bits, 1 stop bit en geen pariteit via een USB-COM poort.

De communicatie tussen de C++ applicatie en de microcontroller gebeurt met vooraf afgesproken gecodeerde ASCII-strings die door de ontvangende kant opgeslagen worden in het afgesproken data type.

|  |  |  |  |
| --- | --- | --- | --- |
| C++ -> Microcontroller | | | |
| Type | Waarde | Omschrijving | Grootte |
| Prefix | U: | Unix timestamp | Uint64 |
| Prefix | M: | Mood setting | UInt8 |
| Postfix | S | Einde data | Uint8 |
| Microcontroller -> C++ | | | |
| Type | Waarde | Omschrijving | Grootte |
| Prefix | DU: | Debug informatie over UNIX timestamp | Uint64 |
| Prefix | DM: | Debug informatie over Mood setting | UInt8 |
| Prefix | DD: | Debug informatie over US sensor | UInt16 |
| Prefix | DT: | Debug informatie over temperatuur sensor | Float |
| Postfix | S | Einde data | Uint8 |



Figuur 11 Software Interface schema uart

### Temperatuursensor

**Technisch Ontwerp van de Temperatuursensor LM35**

De temperatuursensor LM35 wordt gebruikt voor het meten van de temperatuur in ons project. Deze sensor staat bekend om zijn lineaire output van 10mV per graad Celsius, met een meetbereik van -55 tot 150 graden Celsius. Voor dit project kan de temperatuur sensor meten binnen een meetbereik tussen de 2 en 150 graden Clusius.

**Sensor Specificaties en Configuratie:**

* **Type Sensor:** LM35
* **Voedingsspanning**: Minimaal 4 VDC, waardoor deze direct aangesloten kan worden op de 5V voedingsspanning van de microcontroller.
* **Output Spanning:**

-55°C: -550mV

25°C: 250mV

150°C: 1500mV

De output van de LM35 wordt omgezet in een digitale waarde door de Analog Digital Converter (ADC) die aanwezig is op de microcontroller. Deze waarde wordt vervolgens gebruikt om de temperatuur weer te geven op het LCD.

**Pinconnecties:**

* VCC (Sensor) naar PT5V (Microcontroller) voor 5V voeding.
* GND (Sensor) naar PTGND (Microcontroller) voor aarding.
* VOUT (Sensor) naar PTE20 (Microcontroller) voor het ontvangen van de spanning.

**Technische Implementatie en Kalibratie:**

**Kalibratie van de sensor:** De sensor is gekalibreerd voor nauwkeurige temperatuurmetingen binnen het gespecificeerde bereik. De nauwkeurigheid is geverifieerd tegen een standaard thermometer.

**Kalibratie ADC:** Bj het opstarten van de code zal de ADC worden gekalibreerd, waarna de foutmarge wordt gebruikt bij het verrekenen naar de temperatuur.

**Resolutie:** De sensor kan temperatuurveranderingen detecteren tot op een nauwkeurigheid van 0.5°C.

**Software en Foutafhandeling:**

**Software Implementatie:** De ADC-converter stuurt een floating-point waarde naar de main-functie van het programma, waaruit de temperatuur wordt berekend en weergegeven.

**Foutafhandeling:** In het geval van sensorfouten of buiten bereik metingen, genereert de software een foutmelding en neemt passende maatregelen, zoals dat een temp beneden de 2 en boven de +120 als onwaarschijnlijk zijn en een “error” opleveren.

**Testen en Omgevingsfactoren:**

**Testprocedure:** De sensor is uitgebreid getest in verschillende omgevingsomstandigheden om de consistentie van de metingen te waarborgen. Hierbij is vast komen te staan dat de resolutie erg hoog was en dat daardoor de temperatuur erg sterk schommelde. Dit is opgelost door een gemiddelde temperatuur te bepaalde over een bepaalde periode.

**Omgevingsinvloeden:** Factoren zoals luchtvochtigheid en blootstelling aan direct zonlicht zijn onderzocht om hun invloed op de nauwkeurigheid van de sensor te begrijpen. De afwijking was minimaal en had nauwelijks invloed op de omgevingstemperatuur.

**Gebruikte formule voor het berekenen van de temperatuur:**

## Software

Voor de software implementatie is gekozen voor een “***cyclic executive with interrupt scheduler”*** architectuur***.*** Met deze architectuur is de opbouw modulair en kan elk lid van het team zijn bijdrage eenvoudig implementeren.

De main loop zal met een frequentie afhankelijk van de kloksnelheid uit worden gevoerd waarin wordt gekeken of er acties moeten worden uitgevoerd aan de hand van flags die gezet zijn door interrupts.

Indien een waarde van een sensor, schakelaar of communicatie bus is veranderd zal er een functie worden aangeroepen zoals omschreven in onderstaande flowchart

A diagram of a flowchart

Description automatically generated

Figuur 12 Software flow chart

# Realisatie

De hardware- en softwarematige ontwerpen zijn hier tot realisatie gebracht. Aan de hand van voorbeelden en schema's wordt de realisatie in details uitgelegd.

## Hardware

## In deze paragraaf staat de realisatie van de hardware beschreven. De aansturing van de hardware komt vanuit de FRDM-KL25Z ontwikkelbord, dit gaat via de beschikbare headers die op het bord aanwezig zijn. Voor de hardware is een PCA gemaakt, hier komen alle deelsystemen bij een. De schema's van de hardware zijn terug te vinden in Zelfreflectie William

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

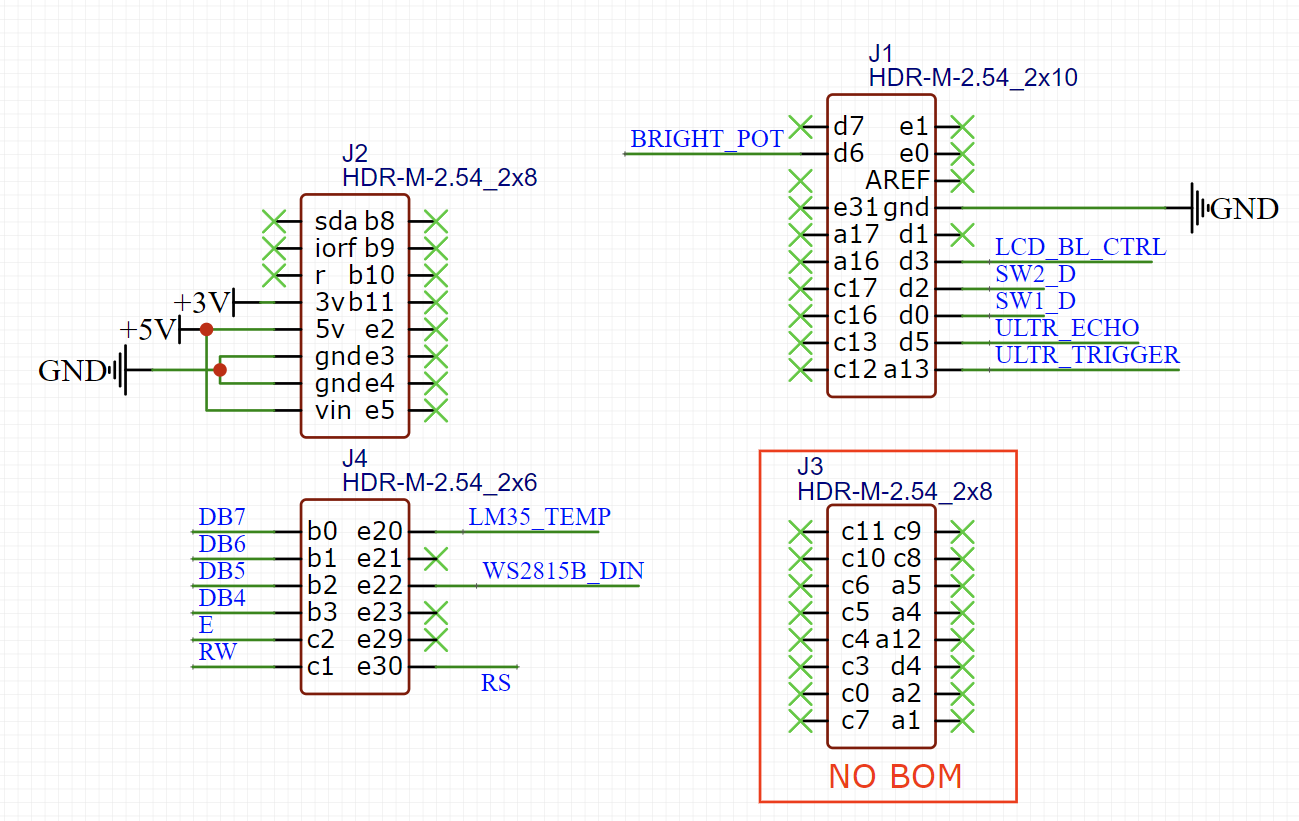
Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

Bijlage 5 – Hardware Schema’s. Voor het PCB-design zijn de illustraties terug te vinden is Bijlage 6 – PCB-design. Tevens is er een 3d view beschikbaar van het PCA en deze is terug te vinden in Bijlage 7 – 3d View.

Voor de pin selectie is er zorgvuldig gekeken om header 1 (in het schema J3) te vermijden. De reden hiervoor is omdat deze afwijkt van plek en dus niet in een through hole prototyping pcb past.

Mocht deze echt nodig geweest zijn dan is het noodzakelijk om elke pin een halve plek om te buigen, wat lijdt tot meer kans op fouten en vergrote assembleertijd.



Figuur 13 Pinout van de Shield headers voor de FRDM-MLK25Z

### Ultrasoon sensor

De ultrasoon sensor HC-SR04 wordt direct aangestuurd vanuit de micro controller. Via de functie process\_ultrasonic\_sensor() wordt de ultrasoon sensor getriggerd. Via het trigger kanaal krijgt de sensor het signaal om de geluidsgolf uit te sturen.

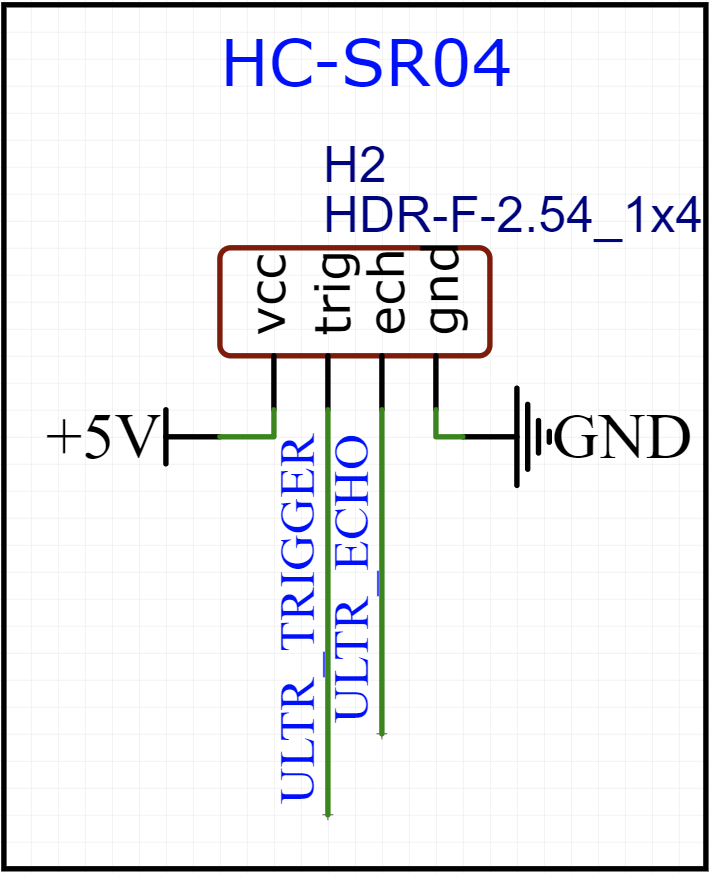
Standaard is de spanning op dit kanaal laag (0v), eens per 100 ms wordt er een puls van 3.3v voor een tijd van 10us uitgestuurd. Hiermee wordt een korte geluidsgolf uitgestuurd.

Door weerkaatsing bij een object komt de golf terug en vangt de sensor dit op. Als de golf is opgevangen stuurt de sensor een puls terug naar de microcontroller waarvan de lengte van de puls afhankelijk is van de afstand die de sensor heeft gemeten.

De afstand wordt dan berekend doormiddel van de volgende formule:

Volgens de datasheet moet de ultrasoon sensor wordt aangesloten op een 5v voeding. De trigger is aangesloten op PTA13.

## Hier is het aansluitschema te zien, voor het volledige schema zie hoofdstuk 11.4 Zelfreflectie William



Figuur 14 Aansluitschema ultrasoon sensor

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

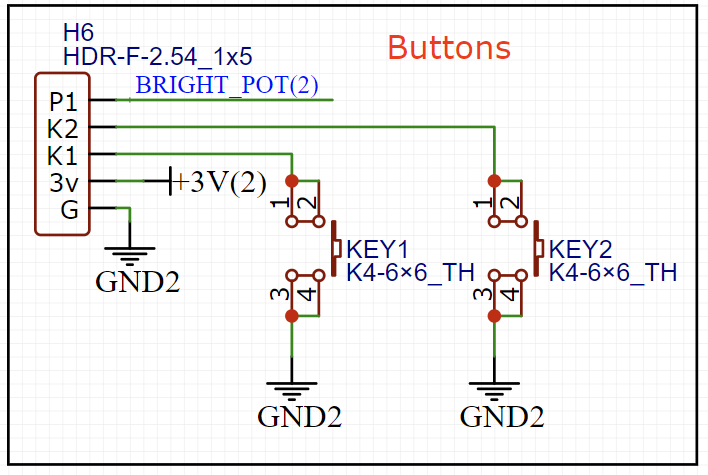
Bijlage 5 – Hardware Schema’s.

### Schakelaars

De schakelaars zijn aangesloten op PTD0 en PTD2. Er wordt geen gebruik gemaakt van de interne pull-up weerstand van de microcontroller. Er is een externe pull-up weerstand aanwezig die samen werkt met een hardware matige debounce. In 7.1.2.1 wordt deze debounce verder toegelicht.

De knoppen werken op een voedingsspanning van 3.3V wat geleverd kan worden door de 3v3 aansluiting op de microcontroller.

## Zie “hoofdstuk 11.4 Zelfreflectie William



Figuur 15 Hardware schema switches

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

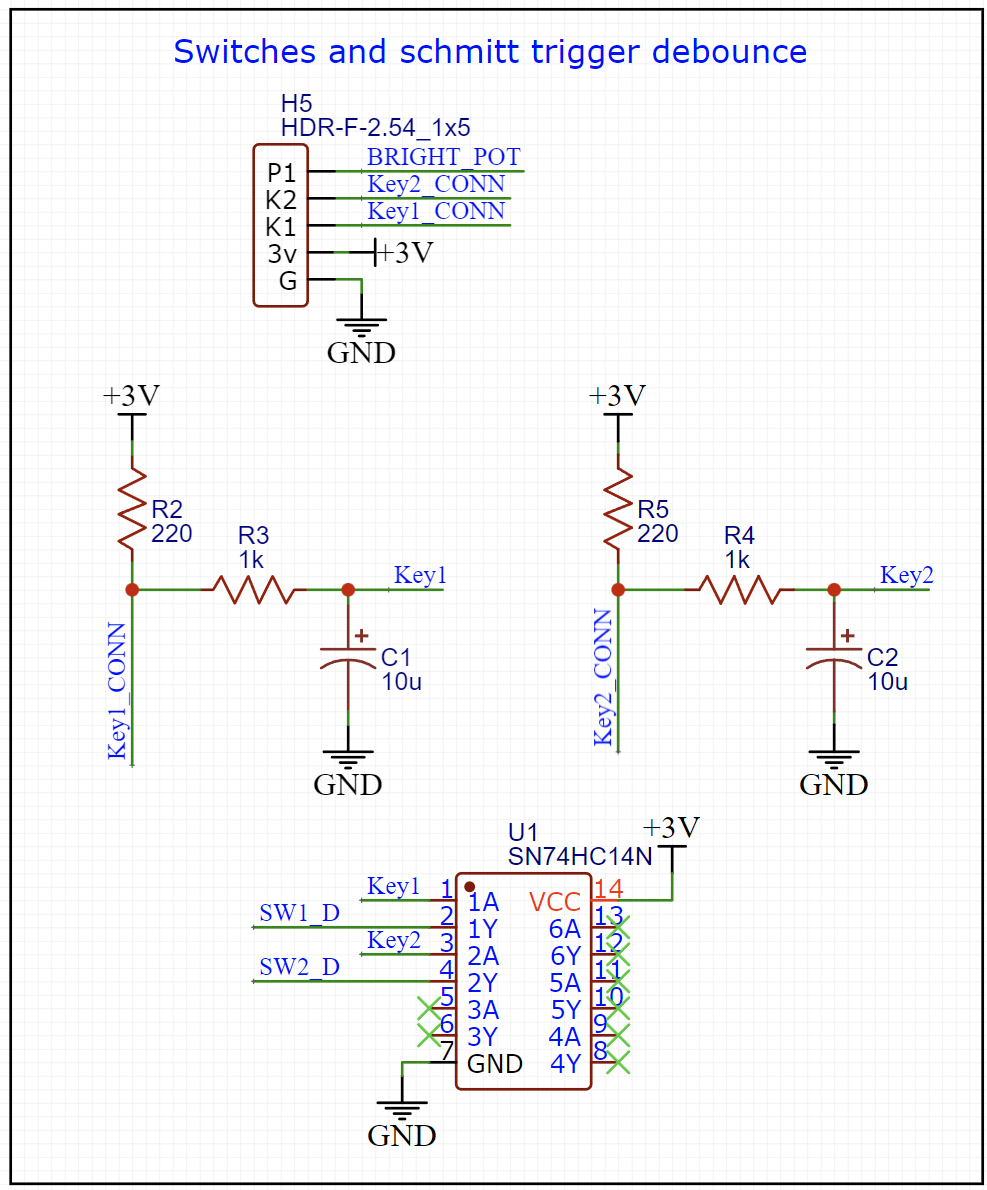
Bijlage 5 – Hardware Schema’s” voor het volledige aansluitschema.

#### Hardware debounce

We gebruiken een hardware matig debounce schakeling die is gemaakt om de spanning op de ingangspin van de microcontroller de stabiliseren. Dit voorkomt onnodige opgaande en neergaande flanken aan de ingang in de microcontroller wanneer een knop zogezegd stuitert.

Dit wordt gerealiseerd door een zes kanaal signaal inverter (SN74HC14N) met Schmitt-triggers te gebruiken.

Zie hieronder een aansluitschema van het debounce schakeling. Dit circuit wordt bij beide schakelaars toegepast.



Figuur 16 Hardware schema switches

De schakelaar heeft in ruststand een open verbinding. De voedingspanning van 3.3v laadt de C1 condensator op via weerstanden R1 en R2. Indien opgeladen zal via de Schmitt-trigger de spanning op de uitgang hoog zijn (3.3v).

Als de schakelaar wordt ingedrukt zal de 3.3v aan de ground getrokken worden en wordt de condensator ontladen. Als de spanning onder de minimale spanning van de schmitt-trigger valt zal de spanning op de uitgang laag worden(0v).

## Voor het volledige schema zie hoofdstuk 11.4 Zelfreflectie William

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

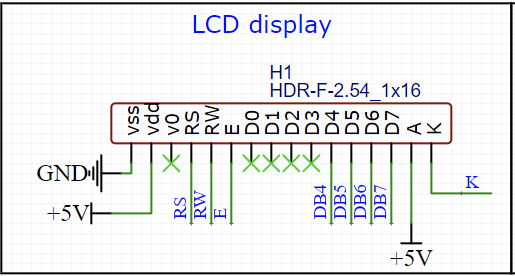
Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

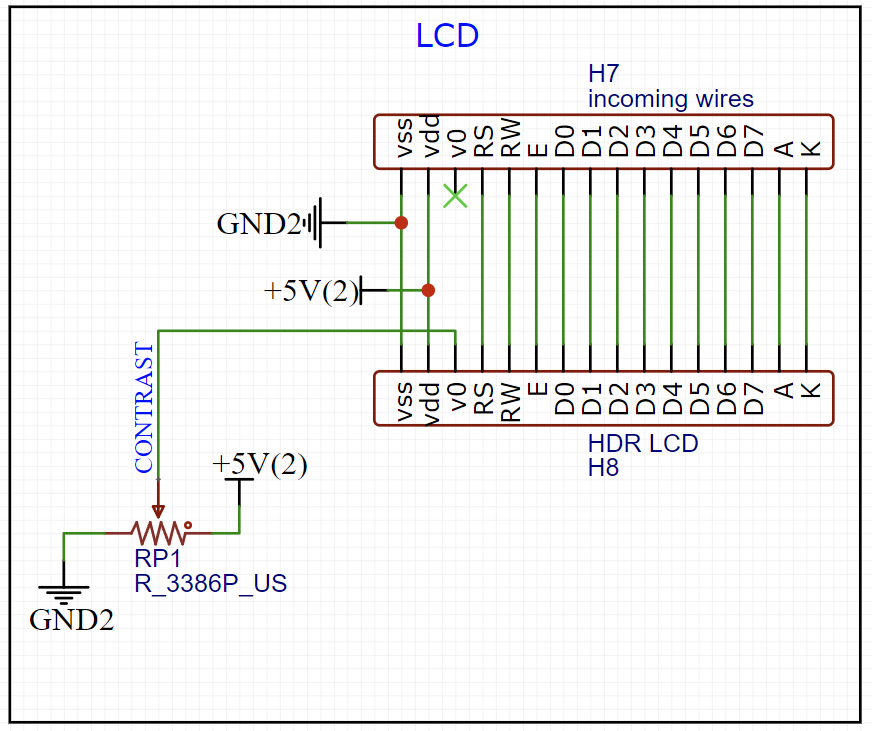
Bijlage 5 – Hardware Schema’s.

### LCD

Het LCD heeft een header met 16 pinnen. Van deze 16 pinnen worden er in totaal 12 gebruikt. Het contrast van het display wordt geregeld door middel van een instelbare 10k Ohm potentiometer. Zie hieronder het aansluitschema:

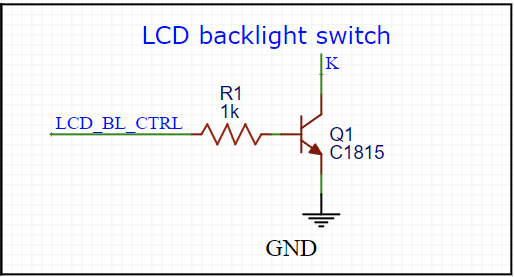


Figuur 17 Hardware schema LCD main board connector



*Figuur 18 Hardware schema LCD*

De achtergrond verlichting wordt aangestuurd doormiddel van een PWM-signaal via TPM0 en wordt uitgezonden via poort PTD3 (zie bijlage 3). Er wordt in serie een 1k Ohm weerstand geplaatst en het signaal wordt geschakeld doormiddel van een generieke C1815 NPN transistor.



Figuur 18 Hardware schema LCD backlight control

Het display werkt met een voeding van 5v.

Aansluitingen D0 t/m D3 worden niet gebruikt.

## Op de voeding en ground na wordt alles direct op de microcontroller aangesloten. Voor het volledige schema zie hoofdstuk 11.4 Zelfreflectie William

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

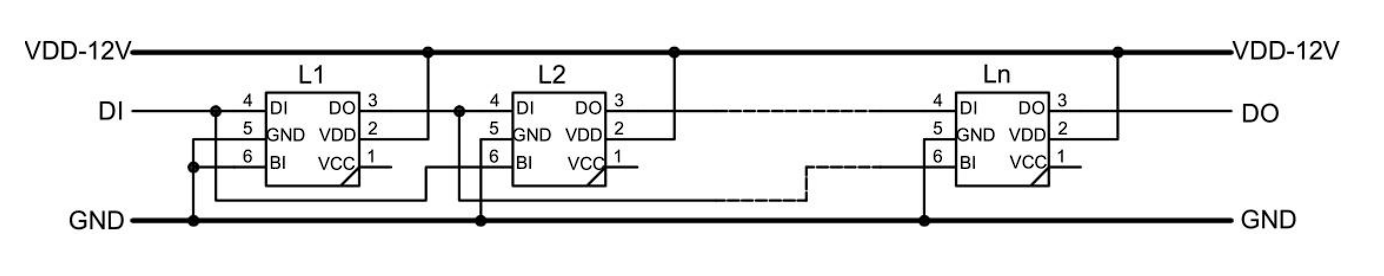
Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

Bijlage 5 – Hardware Schema’s.

### LED-strip

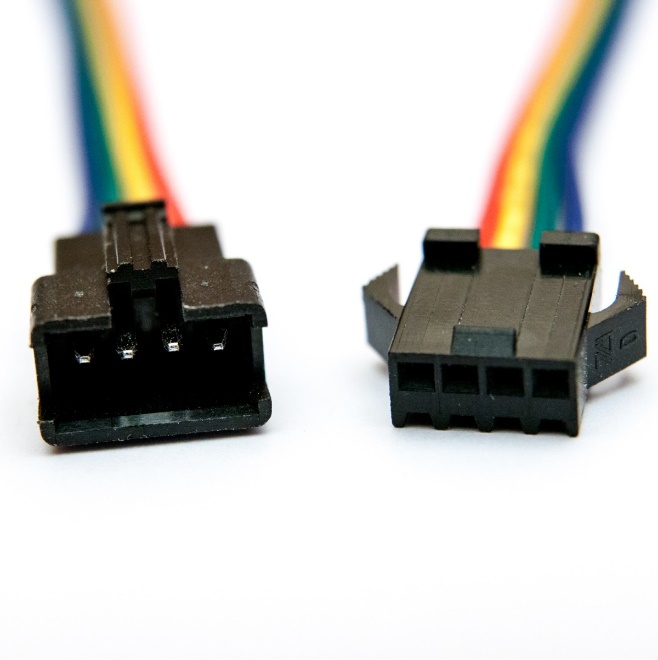
De LED-strips bestaan uit een lange flexibele strip die data en power vervoeren. De strips hebben RGB-LED pixels van het formaat SMD5050. In het volgende figuur is het connectie schema dat herhaald wordt in de strip.



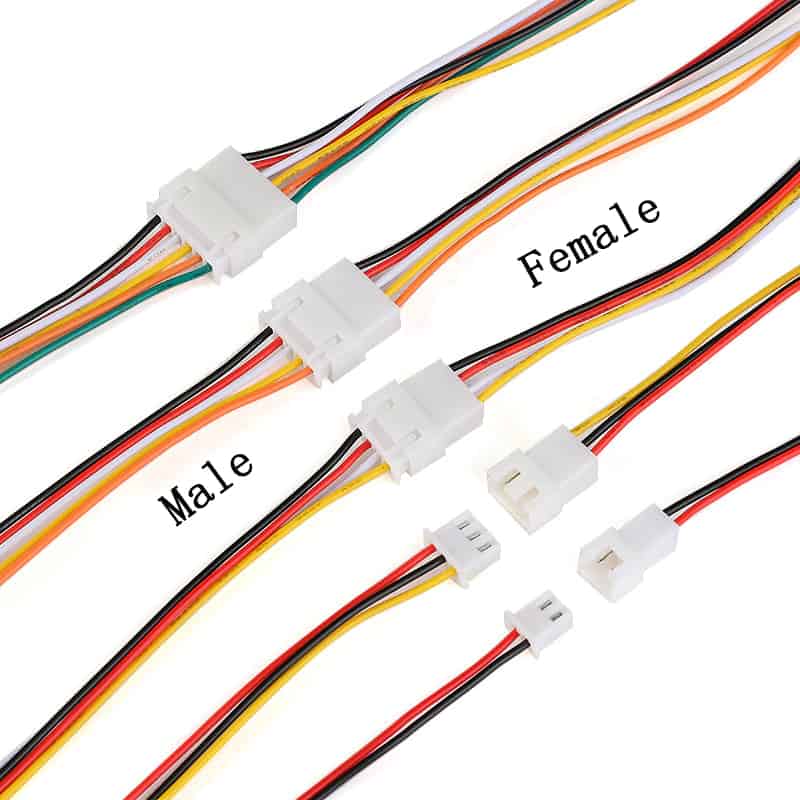
Figuur 19 Typical Application circuit WS2815B

De LED-strips zijn onderling verbonden met JST SM connectoren. De strips zijn allemaal in serie geschakeld. En op enkele plaatsen is de voeding parallel doorgeschakeld om de stroomtoevoer te bevorderen, omdat de strips zelf een redelijk hoge weerstand hebben.

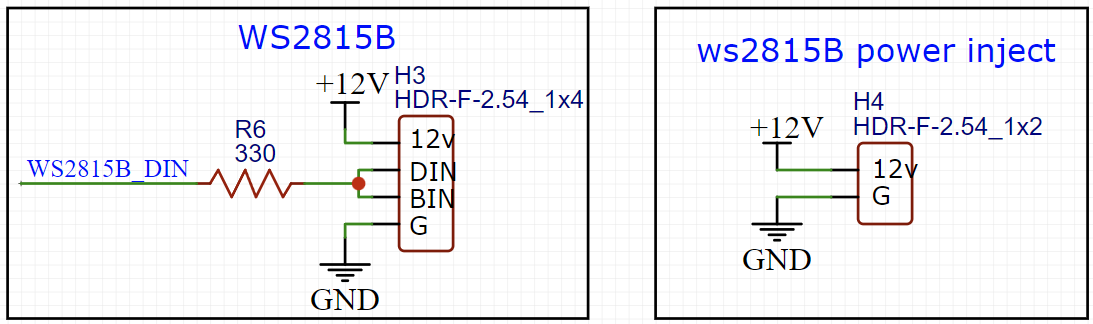
Verder zijn er twee JST SM naar JST XH adapters die de led stip verbinden met de 12 volt voeding en de data signalen op de Hoofd PCB.



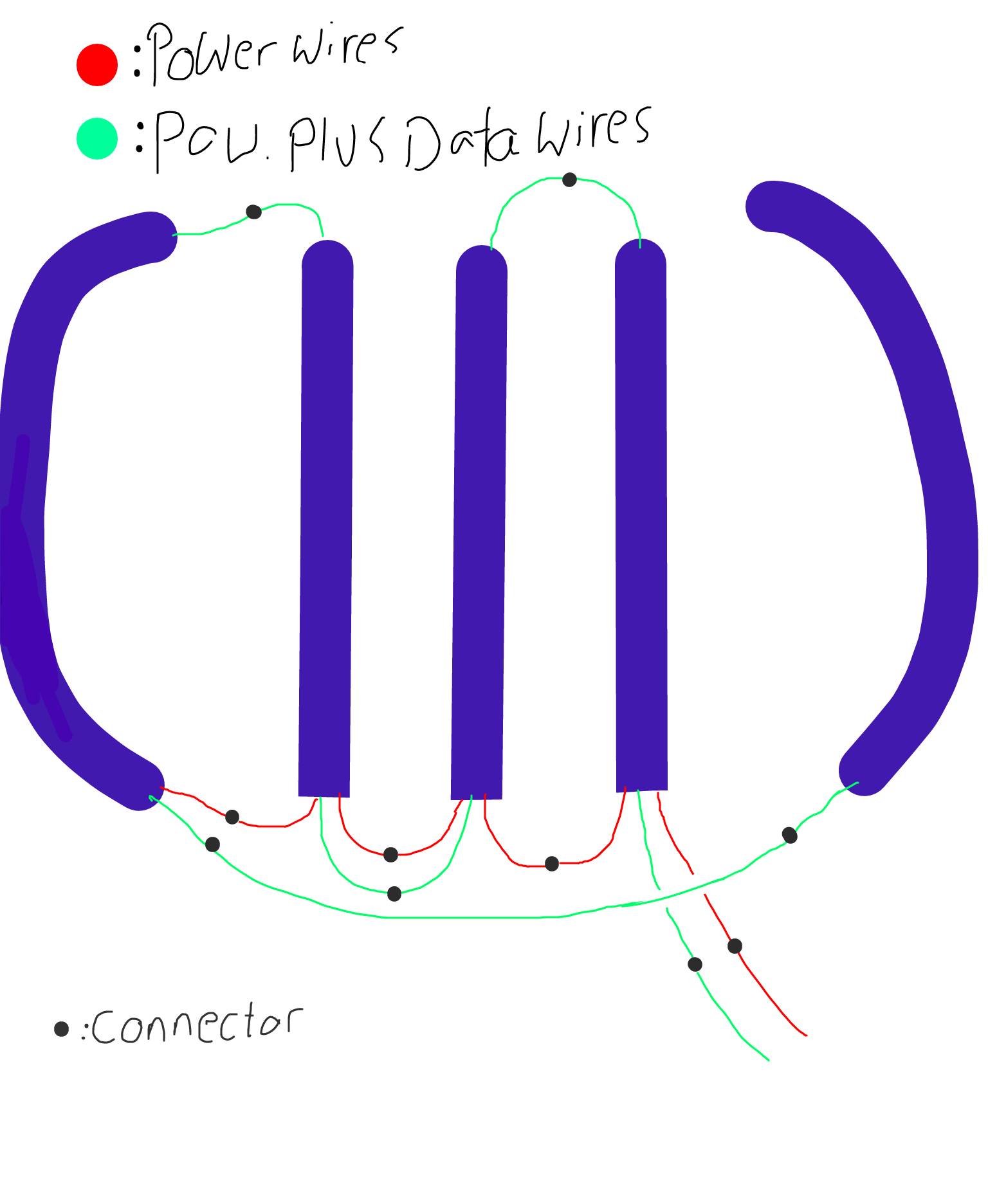
Figuur 20 JST SM connectoren



Figuur 21 JST XH connectoren



Figuur 22 Hardware schema rgb strip connectors

In het volgende figuur zijn de connecties die tussen de led strips geschetst.

### Temperatuursensor

De temperatuur sensor LM35 wordt gemeten door de micro controller. De LM35 dient te worden aangesloten op een voeding van minimaal 4V tot maximaal 30V.

De temperatuursensor geeft via de Vout pin een waarde van 10mV per graden terug aan de microcontroller. Door de temperatuursensor aan 5V en GND aan te sluiten wordt er een spanning uitgestuurd door de temperatuursensor. Het uitgangsignaal van de sensor komt binnen op pin PTE20.

Omdat de temperatuursensor een analoog voltage teruggeeft, dient het Vout-signaal te worden omgezet naar een digitaal signaal met de ADC-converter. Deze omgezette waarde wordt het adc\_result genoemd.

Vervolgens wordt deze waarde gebruikt om de float measured\_voltage te berekenen door

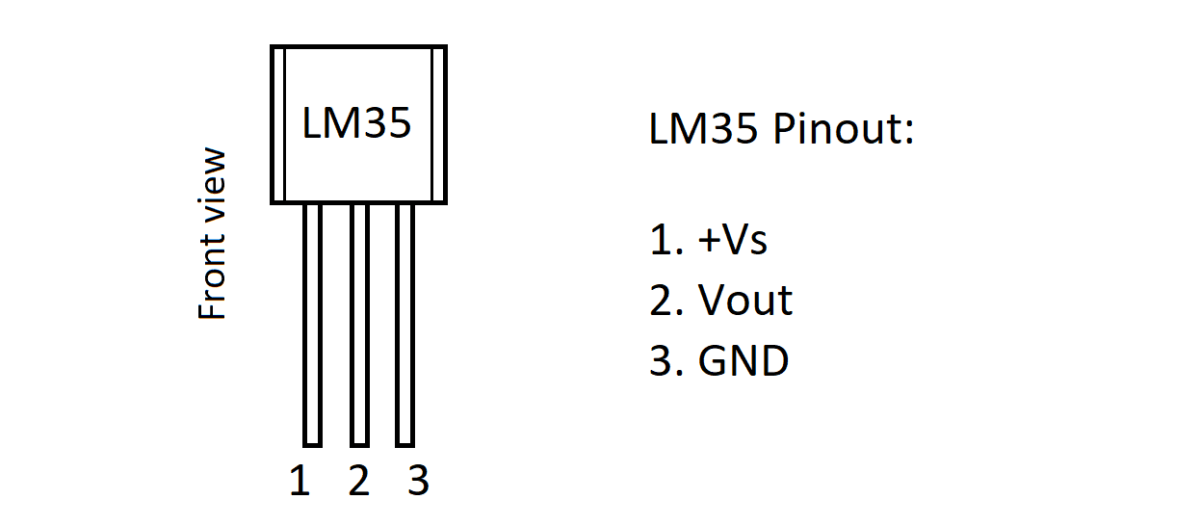
.

Hiermee kan de float temperature worden berekend aan de hand van de volgende formule:

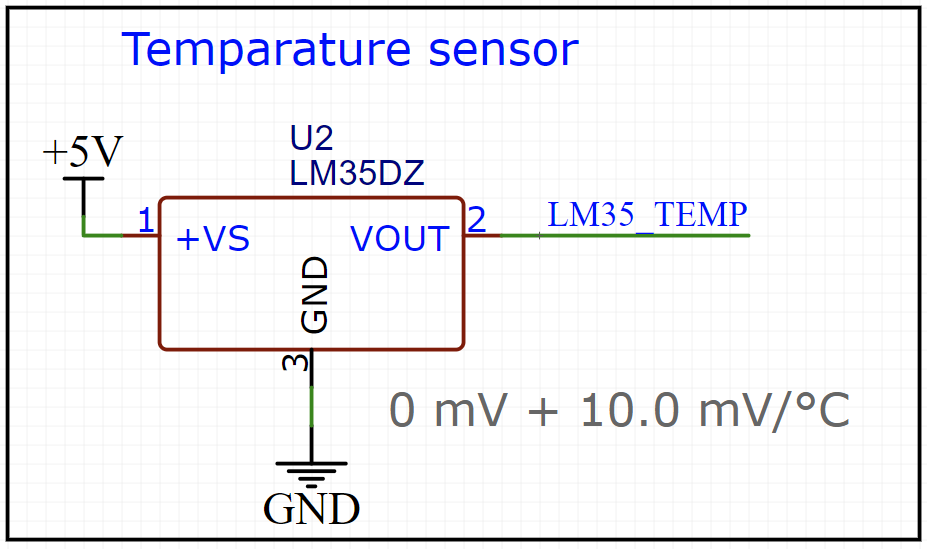
Waarin 0.01 de resolutie is van de temperatuur sensor.

Om de temp stabiel te krijgen wordt het gemiddelde van 1000 temperatuurmetingen weergegeven als waarde, zodat dit grote schommelingen tegengaat. De temperatuur geeft een waarde aan met een nauwkeurigheid van 0,5 graden en rond de waarde af.

Volgens de datasheet moet de temperatuur sensor worden aangesloten op een voeding van minimaal 5V. Hieronder is het hardware schema te zien van de LM35.



Figuur 23 Pinout diagram LM35



Figuur 24 Hardware schema temperatuur sensor

### Voeding

Voor de stroomtoevoer wordt een 2A, 230VAC naar 12VDC voeding gebruikt. De gebruikte voeding is ASUS AD2055M20 010-3LF. Via deze voeding worden alle componenten voorzien van spanning. Een aantal componenten werken met 5v en een aantal werken met 3.3v. Er is een linear regulator gebruikt voor step down van 12v naar 5 volt, en een linear regulator op het prototype board voor de 3.3v.

De voedingsconnector is aangepast, en bevat nu een female JST XH connector.

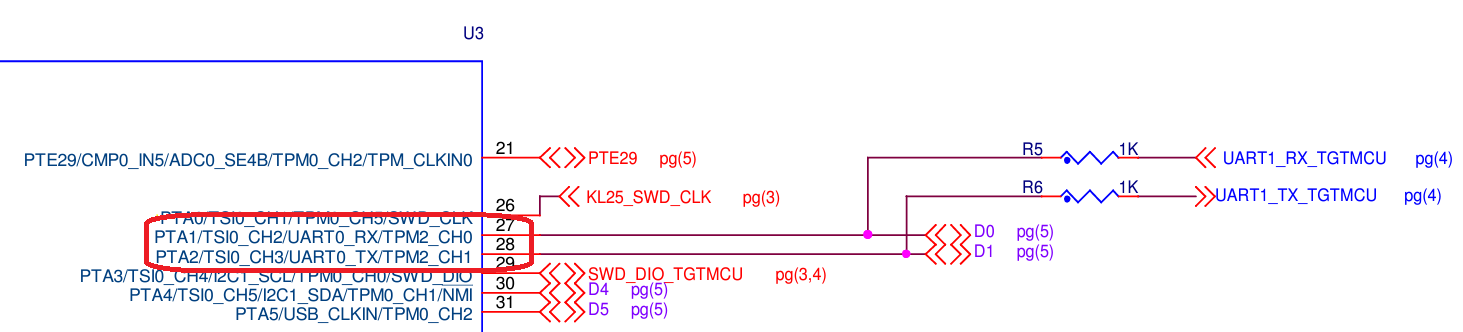
Er is gekozen voor deze voeding omdat deze beschikbaar was en een fijn formaat heeft. Zie volgende figuur voor de voeding.



Figuur 25 12V 2A Voeding

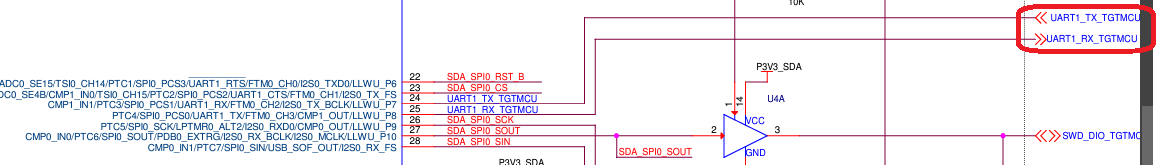
### UART-Update en debug interface

Er is gebruik gemaakt van de on-board UART-interface “UART0” op PTA1 en PTA2.



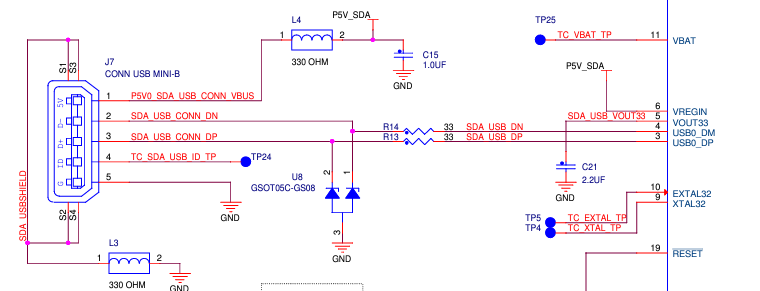
Figuur 26 Aansluitschema uart pins op het protyping board

Deze is weer verbonden op de development kit aan de SDA op pin 24 en 25.



Figuur 27 Aansluitschema uart naar sda chip op het protyping board

Deze SDA is weer verbonden aan de “OpenSDA” USB-poort op de development kit:



Figuur 28 Aansluit schema USB data met de sda chip

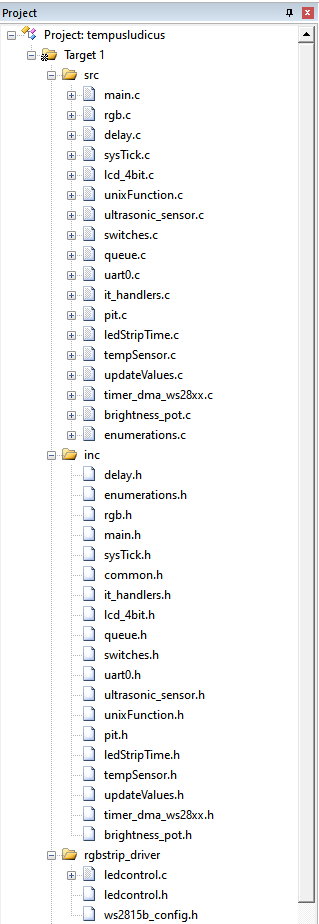
Op deze manier kan er eenvoudig met gestandaardiseerde Micro-USB kabel en door de SDA-chip gerealiseerde virtuele COM poort worden gecommuniceerd met de microcontroller om het apparaat te kunnen configureren en de te debuggen.

## Software Implementatie

In deze paragraaf staat de code beschreven. De complete code wordt naast dit rapport los meegeleverd. Hier worden een aantal snippets van de code getoond en beschreven.

### Keil μVision IDE

Voor het ontwikkelen van de software is de Keil µVision IDE (ARM, 2022) gebruikt. In het volgende figuur zie je een overzicht van de source files voor het project.

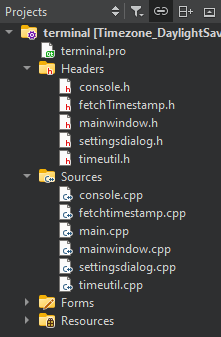


Figuur 29 Source file overview embedded software

### QT Creator

Voor het ontwikkelen van de C++ Debug applicatie is er gebruikt gemaakt van QT Creator 10.0.1.

De functionaliteiten zijn ondergebracht in source en headerfiles om leesbaarheid, onderhoud baarheid en teamgericht werken te kunnen verbeteren in de toekomst.



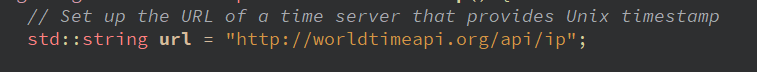
Figuur 30 Source file overview terminal applicatie PC

### Ophalen van timestamp van server

In het C++ programma is er een knop die de functie “buttonClicked” aanroept.

In deze functie na het indrukken van de knop wordt de functie FetchTimestamp aangeroepen die de timestamp van een internetadres ophaalt.

Herin wordt deze URL gezet waarnaar de fetcher moet kijken:



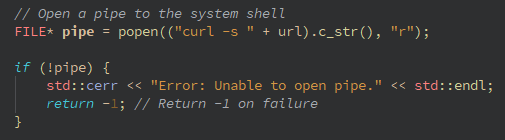
Figuur 31 snippet terminal program “fetchtimestamp.cpp”

Deze URL is een API die een bij een verzoek een antwoord geeft in JSON tekst format. JSON is een gestandaardiseerd gegevensformaat waarin sleutelwoorden en de waarden daarvan makkelijk leesbaar en onderscheidbaar worden gemaakt (json, 2024). In het geval van de gebruikte API gebeurt dat als volgt:



Figuur 32 curl response

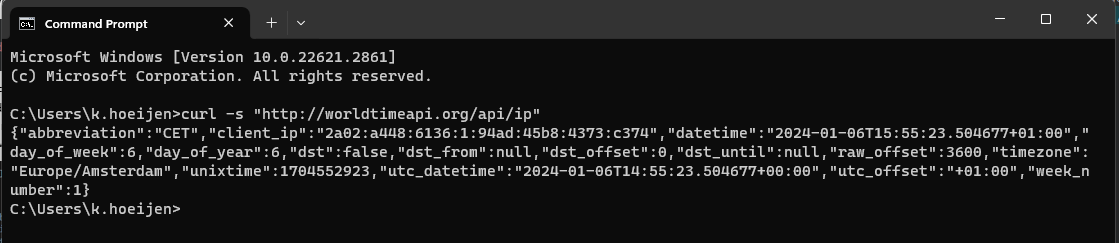
Er wordt een pipe geopend, wat een gedeelte gedeelde geheugen is tussen processen. Het proces dat de pipe opent kan hierdoor informatie delen met andere processen. In het geval van deze software is er een pipe tussen de shell van Windows en het C++ programma. (Microsoft, 2024)



Figuur 33 snippet terminal program “fetchtimestamp.cpp”

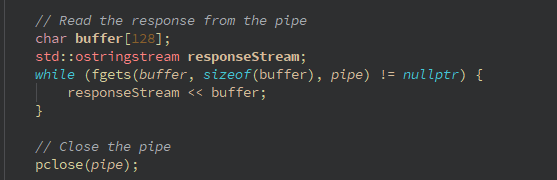
Deze pipe stuurt het volgende commando naar de systemshell. In onderstaand figuur is te zien hoe het commando eruit ziet op Windows command prompt.

curl -s <http://worldtimeapi.org/api/ip>



Figuur 34 snippet terminal program

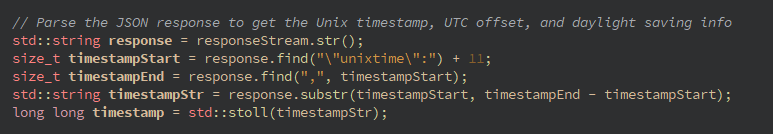
De response van de curl wordt dan in responseStream geplaatst. Hierna wordt de pipe weer gesloten.



Figuur 35 snippet terminal program “fetchtimestamp.cpp”

#### Unix timestamp

In de code wordt dan in het geval van unix timestamp:



Figuur 36 snippet terminal program “fetchtimestamp.cpp”

De response is het complete antwoord van de API. De timestamp is in de JSON geplaatst als “unixtime”: met daarachter de daadwerkelijke timestamp.

De code kijkt in de lijst naar waar het ““**unixtime”:** “kan vinden en telt vervolgens 11 karakters verder om het begin van de timestamp te markeren. Deze 11 karakters komt van het feit dat ““**unixtime”:** “11 karakters lang is.

Vervolgens wordt er gezocht naar het eind van de timestamp met het karakter “**,**”.

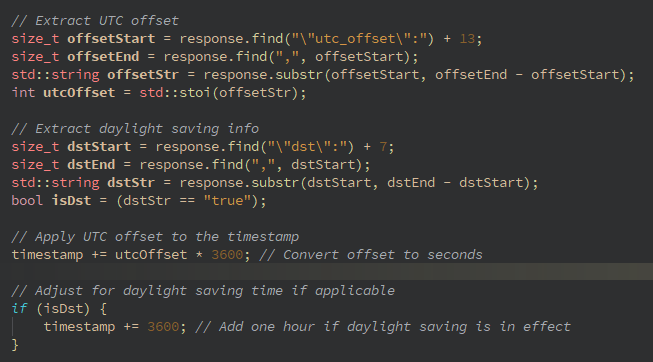
Als laatste wordt de bruikbare data uit de response gehaald met functie “**sting.substr**“waarmee je een nieuwe string maakt op een positie van de oude string en met een aangegeven lengte. (cplusplus.com, 2024)

De positie is timestampStart, de lengte is timestampEnd – timestampStart.

Hiermee wordt de gehele unxitimestamp in getallen uit de tekst gehaald en meegegeven aan de variabele “timestamp”.

#### Toevoegen timezone en winter/zomertijd.

Op dezelfde manier als de unixtimestamp worden ook de tijdzone en winter/zomertijd opgehaald uit de JSON geformatteerde response van de API.



Figuur 37 snippet terminal program

De UTC offset wordt teruggegeven in uren. Deze worden geconverteerd naar seconden door te vermenigvuldigen met 3600 en bij de unix timestamp opgeteld.

Als de “Daylight saving time“wordt toegepast en deze waarde =true is in de JSON response wordt er een extra 3600 seconden bij de unix timestamp opgeteld.

Met al deze correcties wordt ongeacht van locatie en tijd van het jaar de juiste tijd en datum naar de microcontroller gestuurd.

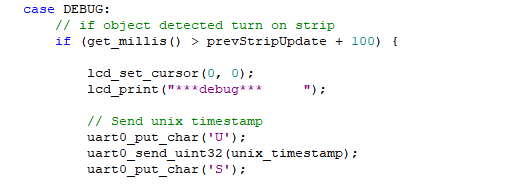
### Debug interface

#### Microcontroller Code

Wanneer de microcontroller terecht komt in debug mode worden de tijd, temperatuur, afstand en mood setting elke cyclus van main doorgestuurd via UART.

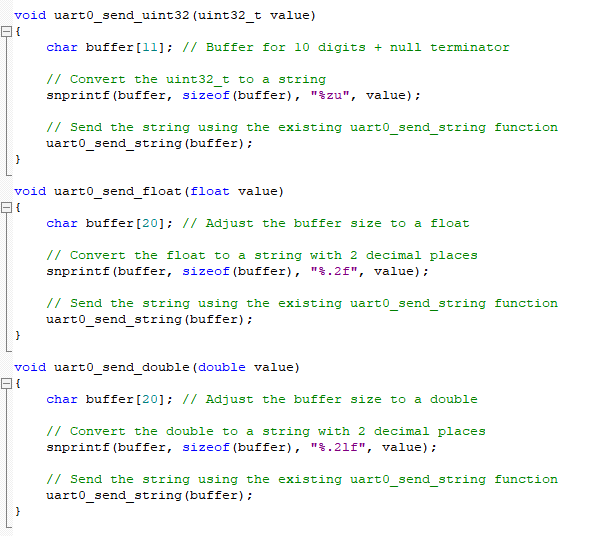
|  |  |
| --- | --- |
| Identifier | Gegeven |
| U | Unix timestamp |
| D | Distance |
| M | Mood |
| T | Temperature |

Onderstaand een voorbeeld hoe de debug informatie wordt verstuurd in de case “DEBUG” elk type informatie wordt afgesloten met een S.



Figuur 38 Software snippet voorbeeld van Informatie sluiting met 'S'

Er zijn verschillende functies gemaakt voor het verzenden van verschillende data typen:



Figuur 39 Software snippet uart op de microcontroller

Sprintf wordt gebruikt om van een datatype een string te maken (cplusplus.com, 2024), zodat deze in de al bestaande uart0\_send\_string(); gebruikt kon worden.

#### C++ pc-code

Het onderscheiden van informatie vanuit de microcontroller naar de terminal applicatie werkt met onderstaande code:

A computer screen shot of a program

Description automatically generated

De code kijkt naar de ingekomen karakters via de seriële bus en zoekt in de ingekomen data naar de volgende patronen:

|  |  |
| --- | --- |
| Identifier | Gegeven |
| U | Unix timestamp |
| D | Distance |
| M | Mood |
| T | Temperature |

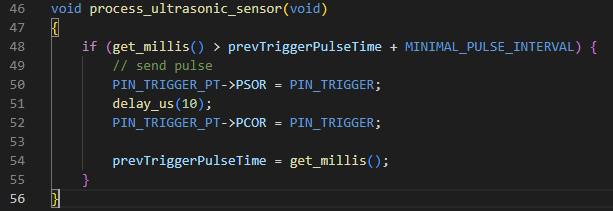
Daarna wordt de ingekomen data afgesloten met een S. Hiermee weet de applicatie dat de inkomende data niet meer bij een van de identifiers hoort en zal de applicatie de informatie tonen in het main window.

A screen shot of a computer code

Description automatically generated

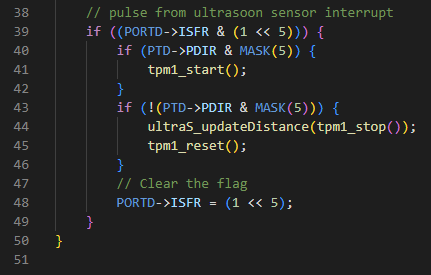
### Ultrasoon sensor

De gebruikte ultrasoon sensor is ingesteld volgens datasheet HC-SR04. Het uitgang signaal is 10µs hoog om een meting te triggeren.

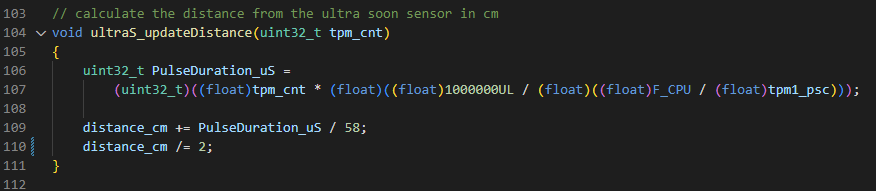


Figuur 40 Software snippet van de signaal trigger van de ultrasoon sensor

Het terugkomende signaal gegenereerd een interrupt op opgaande en neergaande flank, die een TPM-timer aanzet en weer stopt. Deze meet de tijd hoe lang het terugkomende signaal hoog is. Zie afbeelding hieronder:



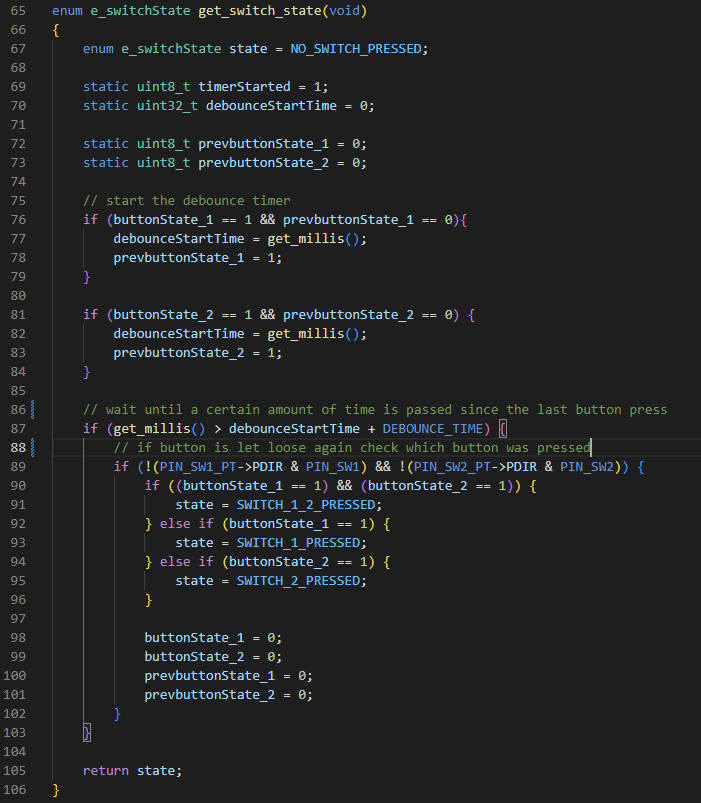
Figuur 41 Software snippet ultra soon sensor interrupt callback



Figuur 42 Software snippet ultrasoon sensor berekening afstand

### Buttons

De twee aanwezige schakelaars kunnen door de software drie verschillende waardes veroorzaken. Zie code hieronder;



Figuur 43 Software snippet switches.c

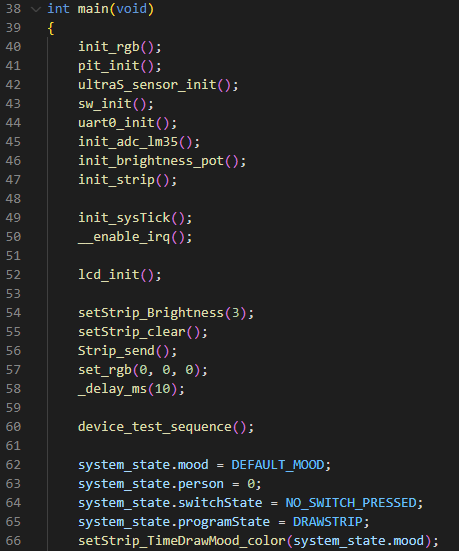
De code hierboven beschrijft dat het mogelijk is om een schakelaar in te drukken en om twee schakelaars tegelijk in te drukken. Om twee schakelaars tegelijk in te drukken en dit correct te registreren wordt er pas gecheckt op flags die gezet worden in de interrupts wanneer alle knoppen weer zijn losgelaten. Dit maakt het mogelijk om een knop in de drukken en pas seconden later de 2e knop in te drukken terwijl de eerste knop nog steeds is ingedrukt, en alsnog een correcte waarde te lezen.

Deze teruggegeven waarde wordt in de main.c gebruikt om bepaalde acties uit te voeren.

### Main functie

In de main functie worden alle deelsystemen met de init functie aangeroepen en geïnitieerd. Daarnaast wordt er aan het eind een deviceTestSequence() aangeroepen, die alle componenten op aanwezigheid en werking test.

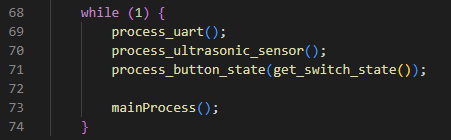
**Initialisatie:**



Figuur 44 Software snippet main.c initialisatie

**Main while loop:**

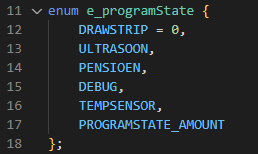
Daarna wordt er een eindeloze loop gestart. In deze loop wordt als eerst een aantal waardes gecontroleerd (waaronder process\_buttons\_state(get\_switch\_state()) om de keuze van het programma (programState) te bepalen en de bijhorende code uit te voeren.



Figuur 45 Software snippet main.c while loop

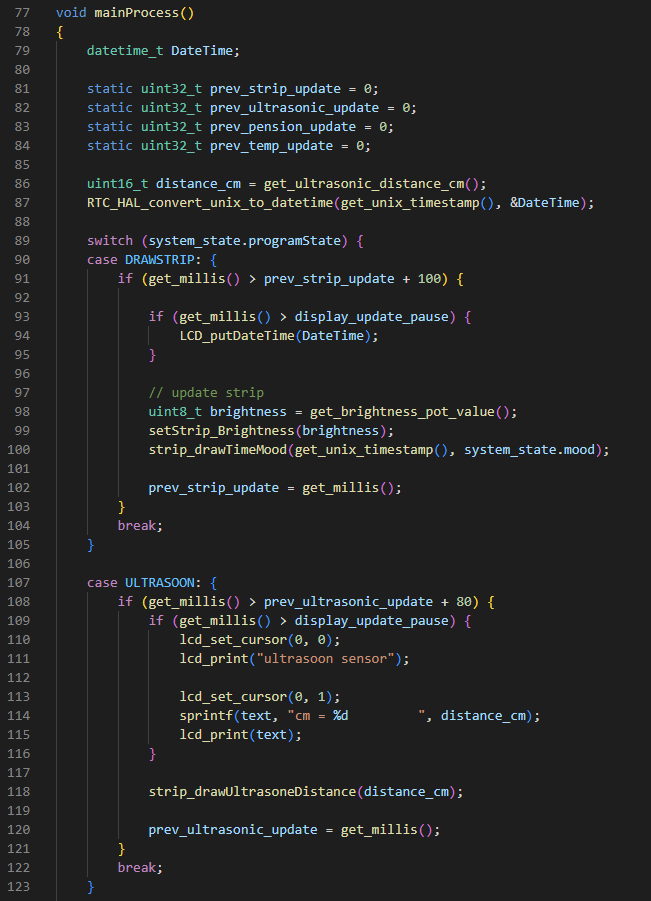
**MainProcess:**

In mainProcess is een switch case te vinden die de functies uitvoert die bij elke programma modus horen in het volgende figuur zie je de programma modus (states).



Figuur 46 Software snippet Program states

In het volgende figuur zie je een deel van de switch case implementatie, niet alles is bijgevoegd in verband met de hoeveelheid. Voor het volledige overzicht zie de source code die bijgelevert is bij het project.

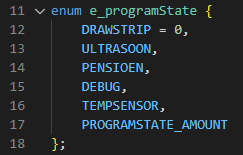


Figuur 47 Code snippet mainProcess Switch case.

**ButtonsProcess:**

De bediening van het programma's gaat doormiddel van de aanwezige schakelaars die zijn uitgelegd in paragraaf **Fout! Verwijzingsbron niet gevonden.**. Hier zit een extra functie bij die van veel belang is voor de gebruiksvriendelijkheid.

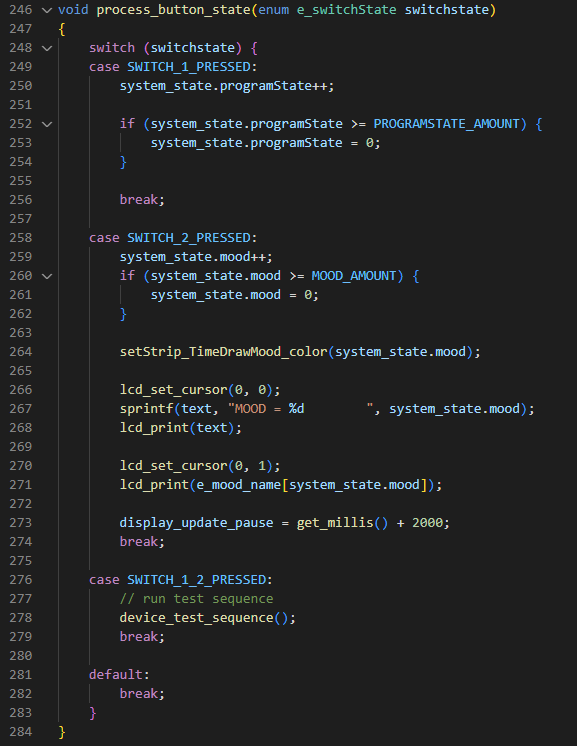
Met button 1 kan worden geschakeld tussen de systeem modussen (program states).



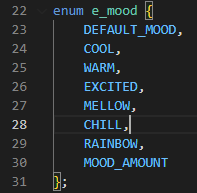
Figuur 48 Program states

Wanneer de mood knop is ingedrukt dan wordt er naar de volgende mood selectie geschakeld, en om de gebruiker direct te laten zien welke mood dit is, word het LCD geüpdatet met het nummer en de naam van de mood, ook wordt er een pauze waarde geschreven die ervoor zorgt dat over in de code de LCD voor de duur van deze pauze waarde niet wordt aangepast. Zo wordt gegarandeerd dat de mood naam en nummer voor 2 seconden wordt weergegeven.

Onder de moods worden verschillende kleuren gezien die de led strip kan aannemen. In het volgende figuur zie je een overzicht van de beschikbare moods.



Figuur 49 Code snippet button state process



Figuur 50 Code snippet Moods

# Testen

In dit hoofdstuk staan de resultaten van de acceptatie testen. Deze testen zijn gedaan om te controleren in welke mate de functionele eisen/ wensen zijn voldaan.

## Acceptatietesten

Voor het controleren van de werking van het prototype zijn vier acceptatietesten uitgevoerd. Bijlage 1 omvat alle uitgevoerde testen en wat er voor nodig is om de testen uit te voeren.

Een test kan wel, gedeeltelijk of niet slagen.

Wel (✓) – Alle teststappen kunnen uitgevoerd worden

Gedeeltelijk (O) – Maximaal twee stappen kunnen niet uitgevoerd worden

Niet (X) – Meer dan twee stappen kunnen niet uitgevoerd worden

|  |  |
| --- | --- |
| Testscenario | Uitkomst |
| Testscenario 1 | ✓ |
| Testscenario 2 | ✓ |
| Testscenario 3 | ✓ |
| Testscenario 4 | ✓ |

# Conclusies en aanbevelingen

Tijdens de aanvang van het project werd door ons een flinke brainstormsessie gehouden. Hierbij kwamen verschillende ideeën ter tafel, enkele werden al snel als onhaalbaar bestempeld. Eén van de ideeën was om de klok de tijd aan te laten geven door middel van RGB-led strips op een achtergrond.

Iedere les werd het project bijgeschaafd en werd het uiteindelijk het project dat succesvol is afgerond. Het resultaat is een klok, die de tijd op basis van de Unixtijdwaarneming weergeeft op een LCD en door middel van RGB-led strips. Als extra functies heeft de klok een temperatuursensor en een ultrasoon sensor, die op afstand de pensioengerechtigde leeftijd weergeeft van één van de vier bouwers/ontwerpers van de klok. Met het bedienen van de drukknoppen kan geschakeld worden tussen verschillende “moods”. Bij zonsopkomst en zonsondergang verschiet de RGB-led van kleur.

Wat door ons niet werd gehaald, was een klok realiseren waarbij het LCD en de RGB-LED strips in een lichte achtergrond werden geïntegreerd. Omwille van de tijd, materiaal en de kosten is er gekozen voor een achtergrond van hout, waarbij de hardware erop werd gemonteerd. Daarnaast werd ook het idee net niet gehaald om de klok te voorzien van een voeding op batterijen.

**Aanbevelingen voor Toekomstige Ontwikkeling:**

**1. Toekomstige Materiaalkeuze:**

Voor volgende iteraties van het project kan onderzoek worden gedaan naar alternatieve materialen voor de achtergrond. Het gebruik van een lichter materiaal kan een betere integratie met het LCD en de RGB-led strips bieden, wat de esthetische waarde van de klok kan verhogen.

**2. Verbetering van het Ontwerp:**

Verdere verfijning van het ontwerp zou een onderwerp kunnen zijn in toekomstige versies. Dit kan omvatten het verkleinen van de klok voor een compacter ontwerp of het toevoegen van aanraakgevoelige bedieningselementen.

**3. Gebruikersevaluatie:**

Het uitvoeren van een gebruikersevaluatie kan waardevolle feedback opleveren over de gebruiksvriendelijkheid en het ontwerp van de klok. Deze feedback kan worden gebruikt om toekomstige ontwerpaanpassingen leidend te kunnen laten zijn.

**4. Energie-efficiëntie en Duurzaamheid:**

Het onderzoeken van energie-efficiëntere componenten en het gebruik van gerecycled of hernieuwbare materialen kan helpen de duurzaamheid van het product te verhogen en het energieverbruik verder te verlagen.

**5. Innovatie:**

Het overwegen van functies zoals Wi-Fi-connectiviteit voor automatische tijdssynchronisatie of integratie met slimme huis-technologieën kan de functionaliteit en aantrekkelijkheid van de klok vergroten.

# Bronvermelding

*Vermeld gebruikte bronnen volgens de* [*APA richtlijnen*](http://www.han.nl/apa)*.*

cplusplus.com. (2024, 1 6). *std::string::substr*. Opgehaald van cplusplus.com: https://cplusplus.com/reference/string/string/substr/

json. (2024, 1 6). *https://www.json.org/json-en.html*. Opgehaald van https://www.json.org/json-en.html : https://www.json.org/json-en.html

Microsoft. (2024, 1 6). *Pipes (Interprocess Communications)*. Opgehaald van learn.microsoft.com: https://learn.microsoft.com/en-us/windows/win32/ipc/pipes

*Non-return-to-zero.* (2024, 1). Opgehaald van en.wikipedia.org: https://en.wikipedia.org/wiki/Non-return-to-zero

worldtimeapi. (2024, 1 6). *worldtimeapi*. Opgehaald van https://worldtimeapi.org/ : https://worldtimeapi.org/

# Bijlage 1 – Testscenario’s

Deze bijlage beschrijft in detail welke acceptatietesten zijn uitgevoerd om de functionele eisen te testen.

## Testscenario 1 – Weergeven van de tijd

**Benodigdheden:**

* Klok
* Hardware versie 1.0
* Software versie 1.0
* Meegeleverde voeding

**Geteste functionele eisen:**

F1, F1,2, F1.3, F1.6, F1.7

**Mogelijke uitkomsten:**

- Wel (✓)

- Gedeeltelijk (O)

- Niet (X)

**Test uitgevoerd**

**17/10/24 “William”**

|  |  |  |
| --- | --- | --- |
| **01** | **Sluit de klok aan op een spanningsbron met de micro USB kabel** |  |
| De volgende tijd wordt weergegeven op het LCD: TIME: 00:00:00 DATE: 01/01/1970  De tijd wordt boven aan gegeven en de datum daaronder  De led strip begint met tellen |  |
| *Waargenomen zoals beschreven* | ✓ |
| **02** | **Wacht 1 minuut** |  |
| Het aantal brandende LEDS voor seconden gaan terug naar 0 en er komt 1 led bij voor minuten |  |
| *Waargenomen zoals beschreven* | ✓ |
| **03** | **Wacht 1 uur** |  |
| Het aantal branden de LEDS voor minuten gaan terug naar 0 en er komt er 1 bij voor uren |  |
| *Waargenomen zoals beschreven* | ✓ |
| **04** | **Wacht tot de tijd 18:00 is** |  |
| De led is veranderd van kleur aan de hand van zonsondergang |  |
| *Waargenomen zoals beschreven* | X |

Conclusie:

De functies werken, behalve de kleurverandering aan de hand van de zonstand. Dit komt omdat er te weinig tijd over was om deze functie te implementeren.

## Testscenario 2 – Extra functies testen

**Benodigdheden:**

* Klok
* Hardware versie 1.0
* Software versie 1.0
* Meegeleverde voeding

**Mogelijke uitkomsten:**

- Wel (✓)

- Gedeeltelijk (O)

- Niet (X)

**Geteste functionele eisen:**

F5, F7, F8, F9, F10, F13, F14

**Test uitgevoerd**

**17/10/24 “William”**

|  |  |  |
| --- | --- | --- |
|  | **Sluit de klok aan op de meegeleverde voeding** |  |
| Klok gaat aan en toont de datum en tijd 00:00:00 01/01/2023 |  |
| *Waargenomen zoals beschreven* | O |
| **02** | **Druk op de linker drukknop** |  |
| Afstandssensor modus wordt getoond |  |
| *Waargenomen zoals beschreven* | ✓ |
| **03** | **Houd een object op 30cm afstand** |  |
| Scherm geeft 30cm aan |  |
| *Waargenomen zoals beschreven* | ✓ |
| **04** | **Houd een object op 150cm afstand** |  |
| Scherm geeft 150cm aan |  |
| *Waargenomen zoals beschreven* | ✓ |
| **05** | **Druk op de linker drukknop** |  |
| Klok gaat in “tijd tot pensioen” modus.  Test de volgende afstanden en naam combinaties:  - 50cm – Maarten  - 80cm – Roel  - 120cm – Kevin  - 150cm – William |  |
| *Waargenomen zoals beschreven* | ✓ |
| **06** | **Druk op de linker drukknop** |  |
| Debug modus wordt getoond |  |
| *Waargenomen zoals beschreven* | ✓ |
| **07** | **Druk op de linker drukknop** |  |
| Temperatuursensor modus wordt getoond. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **08** | **Leg de vinger op de temperatuursensor** |  |
| De temperatuur loopt op tot ongeveer lichaamstemperatuur (acceptabele range 30-40 graden) |  |
| *Waargenomen zoals beschreven* | ✓ |
|  | **Verbreek de VOUT van U2 (temperatuursensor)** |  |
| Het scherm toont de foutmelding “Error!” |  |
| *Waargenomen zoals beschreven* | ✓ |
| **09** | **Druk op de linker drukknop** |  |
| Tijd modus wordt getoond |  |
| *Waargenomen zoals beschreven* | ✓ |
| **10** | **Druk zeven keer op de rechter drukknop** |  |
| De kleurstelling van de klok veranderd per druk op de toets en komt terug bij de originele kleurstelling na de laatste druk op de toets. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **11** | **Druk op beide knoppen tegelijk** |  |
| Klok gaat in zelftest modus.  Alle LEDS gaan branden van beneden naar boven  Klok gaat verder waar deze gebleven was. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **11** | **Draai de potentiometer onder het LCD van helemaal links naar helemaal rechts** |  |
| De helderheid van de LEDS gaan van zwak naar fel. |  |
| *Waargenomen zoals beschreven* | ✓ |

Conclusie:

Alle functies werken naar behoren, behalve de startdatum. De startdatum is 1/1/1970, unix 0

## Testscenario 3 – Instellen en tonen van debug informatie.

**Benodigdheden:**

* Klok
* Hardware versie 1.0
* Software versie 1.0
* Meegeleverde voeding
* Micro USB kabel
* C++ Programma “terminal” uit de projectfolder

**Mogelijke uitkomsten:**

- Wel (✓)

- Gedeeltelijk (O)

- Niet (X)

**Geteste functionele eisen:**

F1.4, F1.5, F2

**Test uitgevoerd**

**17/10/24 “William”**

|  |  |  |
| --- | --- | --- |
|  | **Sluit de klok aan op een spanningsbron met de micro USB kabel** |  |
| De volgende tijd wordt weergegeven: TIME: 00:00:00 DATE: 01/01/1970  De tijd wordt boven aan gegeven en de datum daaronder  De led strip begint met tellen |  |
| *Waargenomen zoals beschreven* | ✓ |
| **02** | **Open het programma “terminal”** |  |
| Het programma toont de huidige tijd in het middelste venster. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **03** | **Verbind de microcontroller met de UART interface door op de verbind knop te drukken** |  |
| Onder in de statusbalk komt de gebruikte COM poort te zien |  |
| *Waargenomen zoals beschreven* | ✓ |
| **04** | **Druk op de “Fetch and send Unix knop** |  |
| De microcontroller toont de huidige datum en tijd |  |
| *Waargenomen zoals beschreven* | ✓ |
|  | **Zet de microcontroller in “Debug” mode door drie keer op SW1 te drukken** |  |
| Het LCD scherm van de klok toont “\*\*Debug\*\*”  De C++ applicatie toont de tijd  De C++ applicatie toont de afstand  De C++ applicatie toont de temperatuur  De C++ applicatie toont de mood instelling |  |
| *Waargenomen zoals beschreven* | ✓ |
| **7** | **Verbreek de verbinding door op de disconnect knop te drukken** |  |
| De COM poort in de statusbalk verdwijnt |  |
| *Waargenomen zoals beschreven* | ✓ |
| 8 | Probeer opnieuw de knop “Fetch and send Unix” |  |
| Er verschijnt een pop-up melding “Serial port not open!” |  |
| *Waargenomen zoals beschreven* | ✓ |
| 9 | Verbreek op de PC verbinding met het internet en probeer de timestamp op te halen |  |
| Er verschijnt een melding "Failed to fetch Unix Timestamp." |  |
| *Waargenomen zoals beschreven* | ✓ |

Conclusie:

Alle functies werken naar behoren.

## Testscenario 4 – Meten van stroomverbruik.

**Benodigdheden:**

* Klok
* Hardware versie 1.0
* Software versie 1.0
* Lab voeding

**Mogelijke uitkomsten:**

- Wel (✓)

- Gedeeltelijk (O)

- Niet (X)

**Geteste functionele eisen:**

F3

**Test uitgevoerd**

**17/10/24 “William”**

|  |  |  |
| --- | --- | --- |
| **01** | **Sluit de klok aan op de lab voeding en stel deze in op 12V 2A** |  |
| De voeding geeft 12V 2A aan. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **02** | **Zet de klok op de felste instelling door de potentiometer naar rechts te draaien** |  |
| De LEDS gaan op maximale brightness branden. |  |
| *Waargenomen zoals beschreven* | ✓ |
| **03** | **Zet de klok in test modus door beide schakelaars tegelijk in te drukken** |  |
| Alle LEDS gaan branden |  |
| *Waargenomen zoals beschreven* | ✓ |
| **04** | **Zet test modus nog een keer aan en controleer op de voeding of deze niet gaat knipperen** |  |
| De getallen op de lab voeding zijn niet gaan knipperen, wat betekend dat de maximale stroom van 2A niet is behaald. |  |
| *Waargenomen zoals beschreven* | ✓ |

# Bijlage 3 Pinout

| FRDM-KL25Z Pin name | FRDM-KL25Z On-board usage | Available on IO Header | I2C1 | UART0 | UART2 | TPM0 | TPM1 | TPM2 | GPIO | ANALOG | CLKOUT |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PTE0 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTE1 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTE2 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTE3 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTE4 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTE5 | - | J9 |  |  |  |  |  |  |  |  |  |
| VDD | Power | - |  |  |  |  |  |  |  |  |  |
| VSS | Power | - |  |  |  |  |  |  |  |  |  |
| USB0\_DP | USB | - |  |  |  |  |  |  |  |  |  |
| USB0\_DM | USB | - |  |  |  |  |  |  |  |  |  |
| VOUT33 | 2.2uF cap | - |  |  |  |  |  |  |  |  |  |
| VREGIN | USB VBUS (5V) | - |  |  |  |  |  |  |  |  |  |
| PTE20 | - | J10 |  |  |  |  |  |  |  |  |  |
| PTE21 | - | J10 |  |  |  |  |  |  |  |  |  |
| PTE22 | - | J10 |  |  | UART2\_TX |  |  |  |  |  |  |
| PTE23 | - | J10 |  |  | UART2\_RX |  |  |  |  |  |  |
| VDDA | Power | - |  |  |  |  |  |  |  |  |  |
| VREFH | Power | J2 |  |  |  |  |  |  |  |  |  |
| VREFL | Power | - |  |  |  |  |  |  |  |  |  |
| VSSA | Power | - |  |  |  |  |  |  |  |  |  |
| PTE29 | - | J10 |  |  |  |  |  |  |  |  |  |
| PTE30 | - | J10 |  |  |  |  |  |  |  |  |  |
| PTE31 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTE24 | Accelerometer I2C | - |  |  |  |  |  |  |  |  |  |
| PTE25 | Accelerometer I2C | - |  |  |  |  |  |  |  |  |  |
| PTA0 | Debug (SWD\_CLK) | - |  |  |  |  |  |  |  |  |  |
| PTA1 | - | J1 |  | UART0\_RX |  |  |  |  |  |  |  |
| PTA2 | - | J1 |  | UART0\_TX |  |  |  |  |  |  |  |
| PTA3 | Debug (SWD\_DIO) | - |  |  |  |  |  |  |  |  |  |
| PTA4 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTA5 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTA12 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTA13 | - | J2 |  |  |  |  |  |  | Trigger US |  |  |
| PTA14 | Accelerometer INT1 | - |  |  |  |  |  |  |  |  |  |
| PTA15 | Accelerometer INT2 | - |  |  |  |  |  |  |  |  |  |
| PTA16 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTA17 | - | J2 |  |  |  |  |  |  |  |  |  |
| VDD | Power | - |  |  |  |  |  |  |  |  |  |
| VSS | Power | - |  |  |  |  |  |  |  |  |  |
| PTA18 | 8MHz XTAL | - |  |  |  |  |  |  |  |  |  |
| PTA19 | 8MHz XTAL | - |  |  |  |  |  |  |  |  |  |
| PTA20 | Reset | J9 |  |  |  |  |  |  |  |  |  |
| PTB0 | - | J10 |  |  |  |  |  |  | LCD D7 |  |  |
| PTB1 | - | J10 |  |  |  |  |  |  | LCD D6 |  |  |
| PTB2 | - | J10 |  |  |  |  |  |  | LCD D5 |  |  |
| PTB3 | - | J10 |  |  |  |  |  |  | LCD D4 |  |  |
| PTB8 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTB9 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTB10 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTB11 | - | J9 |  |  |  |  |  |  |  |  |  |
| PTB16 | Touch Slider | - |  |  |  |  |  |  |  |  |  |
| PTB17 | Touch Slider | - |  |  |  |  |  |  |  |  |  |
| PTB18 | Red LED | - |  |  |  |  |  |  | Red LED channel 0 |  |  |
| PTB19 | Green LED | - |  |  |  |  |  |  | Green LED channel 1 |  |  |
| PTC0 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC1 | - | J10 |  |  |  |  |  |  | LCD RW |  |  |
| PTC2 | - | J10 |  |  |  |  |  |  | LCD Enable |  |  |
| PTC3 | - | J1 |  |  |  |  |  |  |  |  | RTC |
| VSS | Power | - |  |  |  |  |  |  |  |  |  |
| VDD | Power | - |  |  |  |  |  |  |  |  |  |
| PTC4 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC5 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC6 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC7 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC8 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC9 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC10 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC11 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTC12 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTC13 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTC16 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTC17 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTD0 | - | J2 |  |  |  |  |  |  | Switch 1 input |  |  |
| PTD1 | Blue LED | J2 |  |  |  |  |  |  | Blue LED channel 1 |  |  |
| PTD2 | - | J1 |  |  |  |  |  |  | Switch 2 input |  |  |
| PTD3 | - | J2 |  |  |  | Backlight lcd |  |  |  |  |  |
| PTD4 | - | J1 |  |  |  |  |  |  |  |  |  |
| PTD5 | - | J2 |  |  |  |  | (Used) |  | echo US |  |  |
| PTD6 | - | J2 |  |  |  |  |  |  |  | ADC0\_SE5b  Potentiometer |  |
| PTD7 | - | J2 |  |  |  |  |  |  |  |  |  |
| PTE20 | - | J10 |  |  |  |  |  |  |  | ADC0\_SE0  Tempsens |  |
| PTE22 | - | J10 |  |  |  |  |  | Dout  Led-strip |  |  |  |
| PTE30 | - | J10 |  |  |  |  |  |  | LCD RS |  |  |
| GND | Power | J2 |  |  |  |  |  |  |  |  |  |
| P3V3 | Power | J9 |  |  |  |  |  |  |  |  |  |
| P3V3 | Power | J9 |  |  |  |  |  |  |  | Voeding Switch |  |
| P5V\_USB | Power | J9 |  |  |  |  |  |  |  | Voeding LCD/US |  |
| GND | Power | J9 |  |  |  |  |  |  |  | Switch/ LCD/ US |  |
| GND | Power | J9 |  |  |  |  |  |  |  |  |  |
| P5-9V\_VIN | Power | J9 |  |  |  |  |  |  |  |  |  |

# Bijlage 4 Zelfreflectie

## Zelfreflectie Roel

Ik kijk positief terug op het project. Tijdens onze eindpresentatie kunnen we een indrukwekkend prototype laten zien, wat we hebben bereikt door effectieve onderlinge communicatie en samenwerking. Hoewel het tijdsbestek voor het project kort was, ben ik tevreden over wat we als team hebben weten te presteren. Desalniettemin zijn er enkele punten die ik in een toekomstig project anders zou willen benaderen. Een daarvan is het aanstellen van een lead engineer vanaf het begin van het project. Dit is nu niet gebeurd, en ik miste iemand die de leiding kon nemen. In een later stadium heeft Kevin deze rol op zich genomen, wat direct resulteerde in meer vaart in het project. Voor een volgend project zou ik dit dus als eerste stap willen implementeren.

Bovendien is dit project gestart zonder een duidelijk plan van aanpak en taakverdeling. In de toekomst zou ik graag zien dat we hier anders mee omgaan. Hoewel iedereen zijn eigen verantwoordelijkheden heeft opgepakt, kwam aan het einde van het project een groot deel van de hard- en software bij één persoon terecht. Met een goed doordacht plan van aanpak en een heldere taakverdeling had dit voorkomen kunnen worden, zodat het werk gelijkmatiger verdeeld zou zijn. Ondanks deze verbeterpunten ben ik tevreden over het behaalde resultaat, dat mogelijk werd gemaakt door een goed samenwerkend team.

## Zelfreflectie Kevin

In de laatste periode hebben we het project “TempusLudicus” tot ons genomen en als projectgroep samen tot een mooi ontwerp gekomen.

Ik had als leerdoel in dit project, na de eerste paar weken, kennismaken met C++ genoteerd. Al was dit niet het hoofddoel van het project, heb ik in dit project goed kunnen opstarten met deze vaardigheid heb ik hier geleerd een simpel software project op te kunnen leveren. Hiermee heb ik een kleine voorsprong kunnen maken op een latere module in de hoofdfase.

Als ik reflecteer hoe de rolverdeling was in de projectgroep vind ik het erg lastig om in een projectgroep te werken die elkaar voor het project 1 lesuur per week ziet en heb ik als doel voor volgende projecten dat ik een actievere en gestructureerde rol wil spelen om de kwaliteit van zowel software als de verslaglegging te waarborgen.

Dit ga ik de volgende keer aanpakken door meer momenten in te plannen met projectgroepen om samen te evalueren en te kijken waar we staan. Dit is te realiseren door meer harde deadlines af te spreken met elkaar.

Wanneer dit gecombineerd wordt met tussentijds feedback vragen van “opdrachtgever” (docent) kan je een netter geheel opleveren die voldoet aan de eisen die gesteld worden door de HAN.

Al met al vind ik dat we als eerste project, met de tijd die beschikbaar is naast een fulltime baan, naast het feit dat we nog veel kunnen leren we onze functie eisen hebben gehaald en een mooi product hebben neergezet.

## Zelfreflectie Maarten

Reflectie op het TempusLudicus Project

Dit verslag gaat over mijn bijdrage aan het ontwerpen en bouwen van een klok, specifiek mijn rol in het coderen van de software. Mijn taken waren onder andere het documenteren van afspraken, het implementeren van de PIT-timer, het opzetten van de temperatuursensor in de code en hardware, en het bijdragen aan het productrapport.

In het begin van het project was voor mij de taakverdeling niet helemaal duidelijk, deels omdat onze lessen 's avonds laat plaatsvonden en ik soms details vergat naarmate de avond vorderde. Anderzijds ontbrak er iemand in een sturende rol, hierdoor werd niet doorgepakt als het ging om finetunen en deadlines stellen. Ik had hier achteraf gezien een actievere rol in moeten spelen, vooral omdat ik veel ervaring heb met het werken in groepsverband en het sturen ervan. Alleen voelde ik mij niet de aangewezen persoon, omdat ik te weinig kennis en inzicht heb in het programmeren. Bij eventueel volgens project ga ik mij actiever opstellen als het gaat om het samenwerken en sturen, zonder daarbij al te nadrukkelijk op de voorgrond te treden.

Ik was niet de enige die moeite had om de taken te onthouden en stelde ik voor om de afspraken op papier vast te leggen. Dit zorgde voor meer helderheid en ik begon aantekeningen te maken van mijn taken en verantwoordelijkheden, waardoor het voor mij duidelijk werd er van mij werd verwacht.

Na de PIT-timer ging ik aan de slag met de temperatuursensor. Eerst probeerde ik de interne temperatuursensor uit, maar het was lastig om de juiste waarde op het display weer te geven. Uiteindelijk ontdekte ik met hulp van de docent dat ik de stack moest vergroten om dit op te lossen, een les die ik niet snel zal vergeten. Vervolgens werkte ik aan de LM35-temperatuursensor, maar ik had de hardware niet bij de hand om de code te testen. Nadat ik de benodigde materialen had aangeschaft, bleek dat de code niet meteen werkte. Ik worstelde met nauwkeurigheid en stabiliteit, maar na veel inspanning en hulp van online forums kreeg ik de temperatuursensor uiteindelijk aan de praat.

De belangrijkste lessen die ik heb geleerd, zijn dat ik mijn competenties niet moet onderschatten, mijn sterke punten als het gaat om werken in groepsverband moet benutten, sneller om hulp moet vragen en de software regelmatig op de hardware moet testen. Ik moet mijzelf niet te snel opzij zetten en minder kijken naar mijn tekortkomingen.

Al met al was dit project een waardevolle ervaring die mijn software vaardigheden en projectkennis heeft verbeterd. Het is belangrijk om te blijven leren en groeien, zelfs als je aanvankelijk niet tegen obstakels aan denkt te kunnen lopen...

## Zelfreflectie William

Reflectie op het TempusLudicus Project

Terugkijkend op dit project en de samenwerking ben ik erg positief over deze projectgroep. Er was een fijne samenwerking ondanks de wat beperkte contacturen.

In het begin kon ik met mijn al eerder opgedane kennis een mooie spurt maken en de andere projectleden op weg helpen met het gebruik van Git. Toen alles up en running was hebben we gebrainstormd over het project en een aantal keuzes gemaakt. Ook werden de taken verdeeld. Ik werd aangesteld om de git bij te houden en op te letten of alles soepel verliep.

Ik werd daarentegen ziek op het moment dat de git net in gebruik genomen werd, maar door de snelle schakeling van de rest, kreeg Kevin tijdelijk deze rol. Dit verliep goed.

Aan het project heb ik vooral de hardware driver voor de led strips geprogrammeerd. Dit is geïmplementeerd met de DMA en TIM peripheral. We hadden een probleem dat er een dubbelle hoeveelheid memory in gebruik werd genomen. Dit was niet optimaal. Hier heb ik ook een mooie oplossing voor gevonden, waardoor de specificaties die we hadden gekozen toch konden behalen.

Ik heb geleerd met dit project dat ondanks de lessen die ik met vorige projecten heb geleerd, namelijk eerder klaar zijn voordat de deadline er is. Dat 1 week van de voren alsnog erg krap is. Er is dus een mooie verbetering geweest. Maar hierin kon er nog iets verbeterd worden. En misschien ook sneller realiseren dat bepaalde features niet mogelijk zijn. En dus deze niet te ontwikkelen, zodat er tijd over is om eventuele fouten of bugs op te lossen.

Naast de led strip heb ik ook een hoop tijd gestoken in de architectuur van het programma. Nu is de main.c mooi overzichtelijk en is het in 1 oogopslag duidelijk wat er gebeurt. Ook heb ik hier en daar wat bugs verholpen en kleine verbeteringen toegepast.

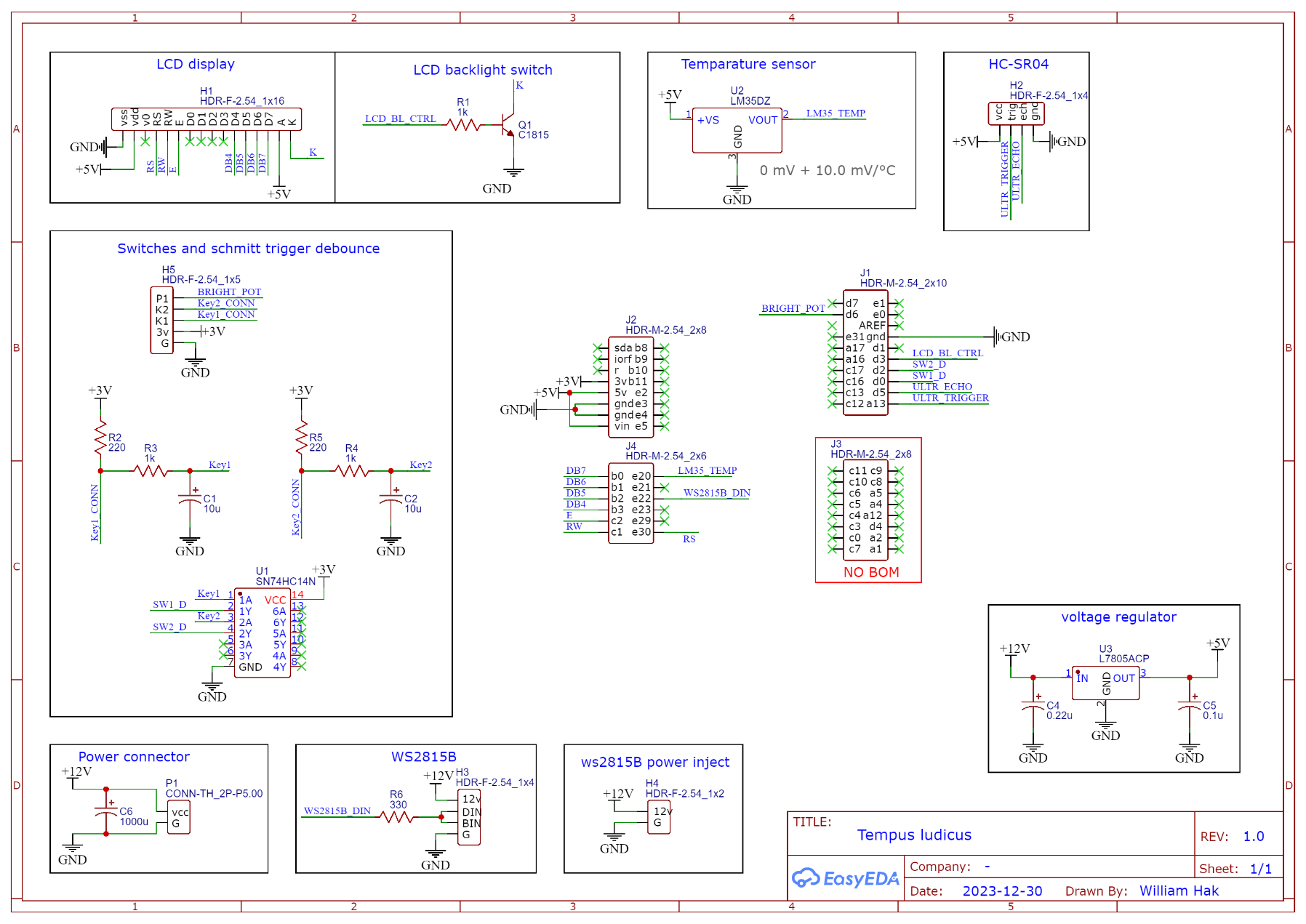
Naast dit alles heb ik ook de hardware schema’s en de pcb’s ontworpen. Dit heb ik nog niet vaak gedaan, en dus was dit een mooi moment om hier meer ervaring in op te doen. Ik heb met het ontwerpen in mijn achterhoofd de prototype toepassing gehad. Namelijk een single layer through hole proto pcb. Dus vandaar dat er soms wat gekke verbindingen zijn getekend. Ook zijn de semi transparante layers niet daadwerkelijke layers maar draadverbindingen op de PCB. Ik heb hier een hoop van geleerd en ben nu een stuk vloeiender met het ontwerpen van de hardware.

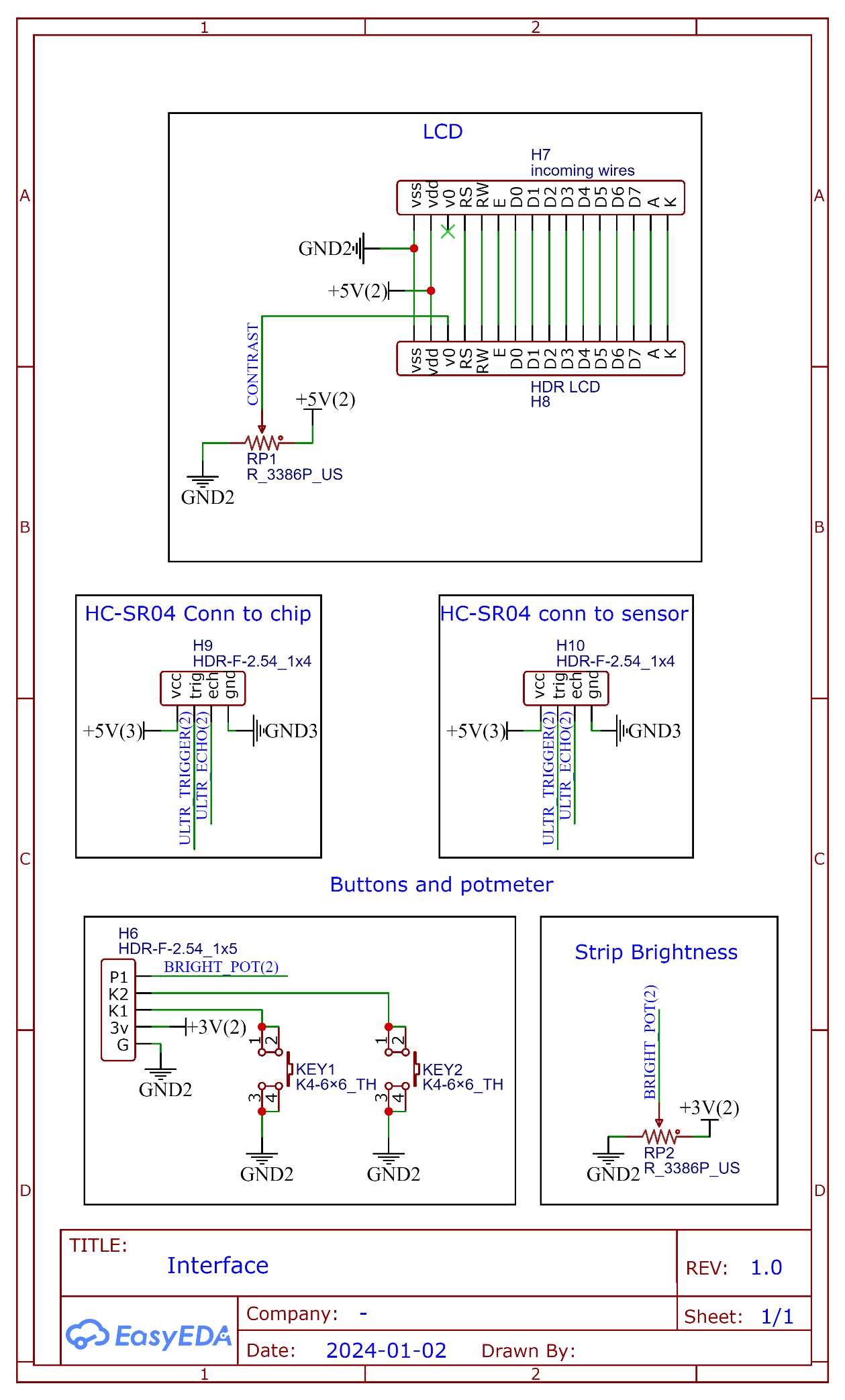
Roel had een cirkel hout geprepareerd, en ik heb deze cirkel hout voorzien van gaten, deze gaten dienen om de pcb’s vast te houden en draden door te lijden van achter het bord naar de voorkant. Hier is best wat tijd in gaan zitten om dit precies en netjes te maken. maar dit is goed gelukt.

Ook heb ik geleerd dat ik moet double checken of een pin op het prototype board wordt gebruikt ondanks dat hij naar buiten op de header wordt geleid. Er was namelijk een bug omdat ik de pinout had veranderd omdat dit beter uitkomt met de hardware. Was een temperatuur sensor verkeerde readings aan het geven. Uiteindelijk kwamen we er dus achter dat deze pin verbonden was met de blauwe on board led. Gelukkig was het daarna snel verholpen.

Ik kijk positief terug op dit leuke project, en met de nieuwe kennis kunnen we naar het volgende hoofdstuk

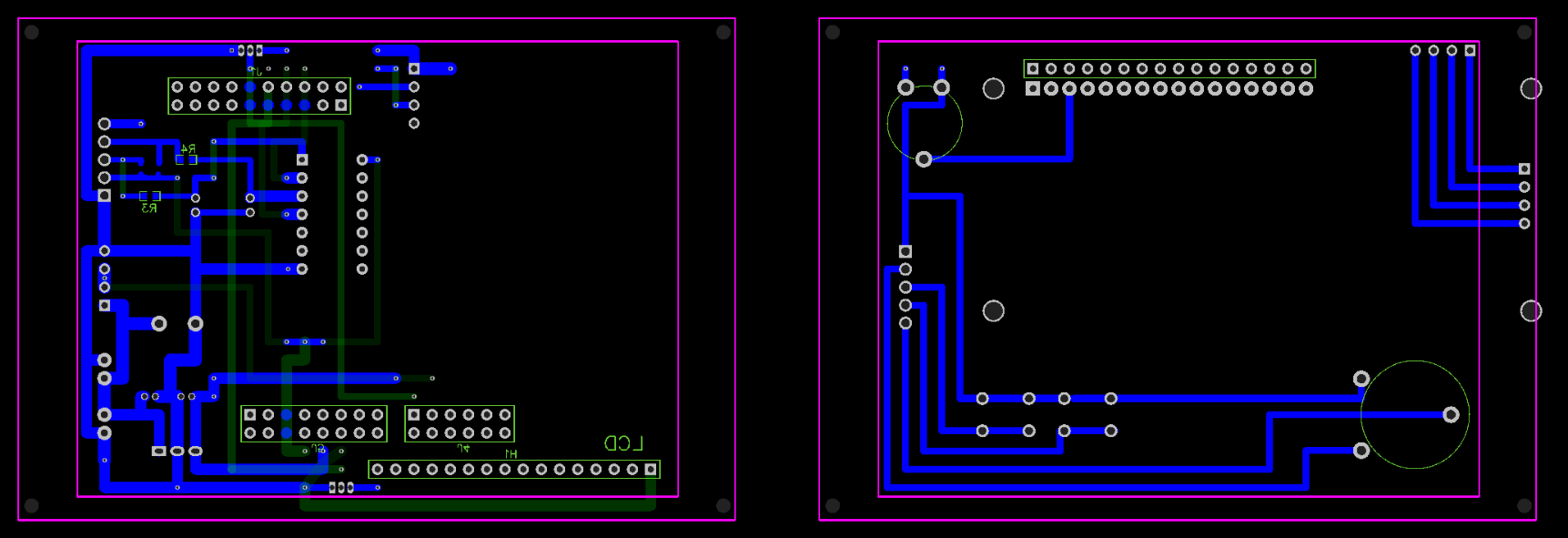
# Bijlage 5 – Hardware Schema’s





# Bijlage 6 – PCB-design

A screenshot of a computer

Description automatically generated

# Bijlage 7 – 3d View

A blue circuit board with many small holes

Description automatically generated A blue circuit board with white and black buttons and white dots

Description automatically generatedA blue circuit board with black and white objects

Description automatically generatedA computer chip with many small objects

Description automatically generated with medium confidence

A computer chip with a screen and buttons

Description automatically generatedA blue and green electronic device

Description automatically generated

1. Een UNIX timestamp is een getal dat het aantal seconden of milliseconden sinds middernacht 1 januari 1970 (GMT) uitdrukt. Indien de timestamp uit 10 cijfers bestaat betreft het seconden, bij 14 cijfers gaat het om milliseconden.

   De UNIX timestamp 1376510200 is bijvoorbeeld woensdag 14 augustus 2013, 19:56:40 GMT. Ons brein kan uiteraard niets met deze timestamp, om er iets zinnigs mee te doen moet de timestamp vertaald worden naar een datum/tijd format. [↑](#footnote-ref-2)