

Productrapport

Handleiding voor het schrijven van een productrapport



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

Auteurs

Hugo Arends

Datum

Augustus 2022

Versie

1.0

Revisies

Versie	Wanneer	Wie	Wat
0.1	Mei 2022	Hugo Arends	Eerste versie van het document.
0.2	Juni 2022	Hugo Arends	Feedback verwerkt van John van den Hooven en Peter Bijl.
0.3	Juli 2022	Hugo Arends	Feedback verwerkt van Jeroen Veen.
0.4	Aug. 2022	Hugo Arends	Samenvatting toegevoegd en tekstuele aanpassingen.
0.5	Aug. 2022	Hugo Arends	H6.2 Stroomverbruik toegevoegd en tekstuele aanpassingen.
0.6	Aug. 2022	Hugo Arends	Feedback verwerkt van Danielle van den Hoogen.
1.0	Aug. 2022	Hugo Arends	Eerste versie van het document, vertaald naar het Engels.

Voorwoord

*Dit document is een handleiding voor het schrijven van een productrapport. Alle pagina's met een blauwe achtergrond, zoals deze, beschrijven instructies voor het schrijven van de verschillende onderdelen van zo'n productrapport. Na ieder onderdeel volgt een voorbeeld in de vorm van een case study. Zo'n case study is duidelijk herkenbaar, doordat de achtergrond van die pagina's wit is. De case study is tevens de rode draad door deze handleiding en een voor mij boeiend onderwerp: een klok. De naam van dit fictieve case study project is **Kairos**.*

Ik vind het belangrijk om te benadrukken dat er voor een project niet één correct productrapport bestaat. Er zijn altijd meerdere manieren om hetzelfde systeem te beschrijven. Waar het om gaat is dat opdrachtgever(s) en opdrachtnemer(s) elkaar begrijpen gedurende de verschillende fasen van een project. Het productrapport is daarvoor een middel.

De belangrijkste doelen voor het schrijven van een productrapport zijn:

- *Product (re)produceren*

De productdocumentatie moet zodanig volledig zijn dat het product aan de hand van de documentatie kan worden gebouwd. Dit betekent dat de volledige elektronische schema's, software listings, aansluitschema's van kabels en connectoren, etc. aanwezig moeten zijn (in bijlagen).

- *Onderhoud mogelijk maken*

In de toekomst wijzigingen of uitbreidingen kunnen aanbrengen zonder dat het ontwerp opnieuw moet plaatsvinden. Hiervoor dienen dus keuzes met onderbouwing te worden vastgelegd om te voorkomen dat dezelfde afweging opnieuw moet worden gemaakt.

- *Kennisoverdracht aan collega engineers*

Het mogelijk maken dat de ontwikkeling wordt voortgezet door collega's. Het ontwerp overdragen aan een productieafdeling of aan een projectgroep die het product productierijp gaat maken. In ons geval moeten de collega-studenten zich op de hoogte kunnen stellen van hoe het totale product tot stand is gekomen om ook die dingen te leren waarbij ze zelf minder expliciet betrokken zijn geweest. Ze moeten zich m.b.v. het productrapport voorbereiden op het tentamen.

Deze handleiding is tot stand gekomen nadat ik 15 jaar projectgroepen, stagiaires en afstudeerders heb begeleid bij de opleiding Embedded Systems Engineering van de Hogeschool van Arnhem en Nijmegen. De informatie in dit document bouwt dan ook voort op het fantastische werk van mijn (gepensioneerde) collega's. Er waren voor mij twee aanleidingen om hun werk te herzien en aan te vullen.

Ten eerste was ik van mening dat de voorgaande productrapporthandleidingen een te abstract ontwerpniveau ambieerde. Keer op keer bleek het lastig voor studenten om een systeem op te delen in puur functionele blokken, waarbij geen onderscheid gemaakt wordt tussen hardware en software. In deze herziene handleiding wordt een systeembenadering gehanteerd, waarbij er vanuit deelsystemen direct een architectuur wordt ontworpen.

Ten tweede waren er geen handleidingen beschikbaar voor de laatste fasen van een project, namelijk de realisatie en het testen. Die heb ik in deze handleiding toegevoegd, inclusief voorbeelden.

*Hugo Arends
Arnhem 2022*

Samenvatting

*Beschrijving van de uitgangspunten, te bereiken doelen, wat wel en niet bereikt is, **bereikte resultaten**; de samenvatting moet een totaalindruk geven van de gehele opdracht en is maximaal 1 A4.*

Samenvatting - Case study Kairos

Het project Kairos is uitgevoerd voor het bedrijf *ESE clockworks*. Het doel van het project is om een werkend prototype te realiseren en documenteren waarmee aangetoond wordt hoe draadloze communicatie en een backup batterij toegevoegd zouden kunnen worden aan de klokken die het bedrijf produceert. De doelen worden gerealiseerd middels een prototype op basis van de FRDM-KL25Z ontwikkelkit.

Er is gestart met een vooronderzoek om vast te stellen hoe de FRDM-KL25Z ontwikkelkit geschikt gemaakt kan worden voor een backup batterij. Er moeten hiervoor extra componenten worden toegevoegd en verschillende aanpassingen aan de ontwikkelkit worden doorgevoerd. Daarnaast is er een vooronderzoek uitgevoerd naar de detectie van een hand die zich voor de klok bevindt. De keuze is gevallen op een TCRT5000 reflectieve optische sensor, met name vanwege de gunstige kosten en vormfactor.

De hoofdfunctie van het systeem is het weergeven van de tijd. In overleg met de opdrachtgever is een lijst van functionele- en technische specificaties opgesteld. Tevens is een user interface ontworpen, inclusief voorbeelden van de schermweergaven, om een compleet beeld te schetsen van hoe het systeem er uiteindelijk uit zou komen te zien.

Doormiddel van een architectuurschema is het systeem opgedeeld in subsystemen. Dit architectuurschema toont de deelsystemen en de interfaces. De hardware en software specificaties van die interfaces worden in detail besproken. De main applicatie is ontworpen volgens het cyclic executive with interrupt scheduler mechanisme.

De realisatie van de deelsystemen wordt in detail besproken aan de hand van de elektrische schema's en berekeningen voor spanning en stromen. Ook wordt het gerealiseerde PCB weergegeven en toegelicht. Ten behoeve van de softwareontwikkeling worden de instellingen van de Keil μ Vision IDE beschreven. Vervolgens wordt de softwarerealisatie van de switches, het OLED display en de main applicatie toegelicht aan de hand van code snippets en metingen met een logic analyzer.

Er is een acceptatietest uitgevoerd om het gerealiseerde prototype te testen aan de hand van de functionele specificaties. Hiertoe zijn vijf testscenario's gedocumenteerd. Drie van de testscenario's zijn geslaagd, één test scenario is deels geslaagd en één testscenario is niet geslaagd.

De hoofdoelen van het project Kairos waren het gebruik van een ARM microcontroller en de toevoeging van draadloze communicatie voor het synchroniseren van de tijd op afstand. Daarnaast is er inzicht verkregen in het energieverbruik van het FRDM-KL25Z board. Deze hoofdoelen zijn allemaal behaald en er is een werkend prototype opgeleverd. De ontworpen, gerealiseerd en geteste klok die tijdens het project Kairos tot stand is gekomen kan een groot succes worden genoemd. Mede dankzij dit project is het bedrijf *ESE clockworks* een stap dichterbij de realisatie van een nieuwe generatie digitale klokken.

Inhoudsopgave

Revisies	2
Voorwoord	3
Samenvatting.....	4
Samenvatting - Case study Kairos	5
Inhoudsopgave	6
1 Inleiding	9
1.1 Aanleiding.....	9
1.2 Doelstelling.....	9
1.3 Structuur van het rapport	9
1 Inleiding - Case study Kairos.....	10
2 Vooronderzoek	12
2 Vooronderzoek - Case study Kairos.....	13
2.1 FRDM-KL25Z geschikt maken voor backup batterij	13
2.2 Detecteren van een hand voor de klok	14
3 Functioneel ontwerp	16
3.1 Functionele specificaties	16
3.2 Technische specificaties	17
3.3 User interface	18
3 Functioneel ontwerp - Case study Kairos.....	19
3.1 Functionele specificaties	19
3.2 Technische specificaties	21
3.3 User interface	21
4 Technisch ontwerp	24
4.1 Architectuur.....	24
4.2 Interfaces.....	25
4.2.1 Voedingsspanning	25
4.2.2 Microcontroller – Sensor.....	25
4.2.3 Microcontroller – Actuator.....	25
4.2.4 Microcontroller – Communicatie – PC driver – App	26
4.3 Software	26
4 Technisch ontwerp - Case study Kairos.....	28
4.1 Architectuur.....	28

4.2	Interfaces	28
4.2.1	Voedingsspanning	28
4.2.2	Microcontroller – RGB LED	29
4.2.3	Microcontroller – Switches.....	29
4.2.4	Microcontroller – IR sensor	30
4.2.5	Microcontroller – RTC.....	30
4.2.6	Microcontroller – OLED display	31
4.2.7	Microcontroller – Bluetooth – Driver – App.....	31
4.3	Software	33
5	Realisatie	34
5.1	Hardware	34
5.2	Software	34
5	Realisatie - Case study Kairos	35
5.1	Hardware	35
5.1.1	Elektrisch schema	35
5.1.2	PCB ontwerp	36
5.2	Software	37
5.2.1	Keil μ Vision IDE.....	37
5.2.2	Switches.....	38
5.2.3	OLED display	39
5.2.4	Main.....	42
6	Testen	43
6	Testen - Case study Kairos.....	44
6.1	Acceptatietesten	44
6.2	Stroomverbruik	44
7	Conclusies en aanbevelingen	45
7	Conclusies en aanbevelingen - Case study Kairos	46
8	Verwijzingen	47
	Bijlage A.....	49
	Bijlage B	50
	Bijlage n	51
	Bijlage A – Elektrisch schema	52
	Bijlage B – Testscenario's	53
	Testscenario 1: Weergeven van de tijd	53
	Testscenario 2: Datum en tijd aanpassen	53

Testscenario 3: Detecteren van een hand voor de klok.....	55
Testscenario 4: TBD	56
Testscenario 5: TBD	56
Bijlage C – Metingen stroomverbruik.....	57
Meetopstelling	57
Geen aanpassingen	59
Aanpassingen voedingsspanning.....	60
SWD aanpassingen	62
Notities	64
Bijlage n	65

1 Inleiding

De inleiding is de toegang tot het rapport. Er wordt beschreven waarom en hoe het project uitgevoerd wordt. De inleiding wordt geschreven aan de hand van de richtlijnen zoals beschreven in het boek Schrijven voor Technici [1]. Dat betekent dat de inleiding hoofdstuk 1 is, geschreven is zonder (tussen)koppen en bestaat uit drie alineagroepen, waarin de aanleiding, de doelstelling en de structuur van het rapport worden beschreven.

1.1 Aanleiding

De aanleiding is geschreven vanuit het perspectief van de opdrachtgever. De eerste alinea beschrijft relevante achtergrondinformatie en geeft een informatieve situatieschets van de huidige situatie. De tweede alinea beschrijft het probleem of de wens van de opdrachtgever. De derde alinea beschrijft het belang/relevantie van het project.

1.2 Doelstelling

De doelstelling omvat een hoofdvraag (stellend of vragend) en het doel van het project. Tevens wordt beschreven op welke manier de hoofdvraag beantwoord zal worden, oftewel de werkwijze van het project. Tot slot worden de randvoorwaarden en uitgangspunten beschreven.

1.3 Structuur van het rapport

Beschrijving van de inhoud van het rapport per hoofdstuk, startend bij hoofdstuk 2. Na het lezen van de structuurbeschrijving is de rode draad van het rapport duidelijk.

1 Inleiding - Case study Kairos

Dit project wordt uitgevoerd in opdracht van het bedrijf *ESE clockworks*. Het bedrijf is gespecialiseerd in het maken van digitale klokken op basis van NXP KL2 series microcontrollers. Deze klokken hebben een OLED display dat de datum en tijd weergeeft en drukknoppen om die datum en tijd in te stellen. Middels dit project wil *ESE clockworks* voor hen twee nieuwe technologieën verkennen. Ten eerste overweegt het bedrijf om draadloze communicatie aan de klokken toe te voegen, zodat de tijd op afstand ingesteld kan worden in plaats van alléén via drukknoppen. Ten tweede overweegt het bedrijf om de klokken te voorzien van een backup batterij, zodat de tijd doortelt als de klok spanningsloos raakt.

Aan de hand van de kennis en inzichten die opgedaan worden door dit project kan *ESE clockworks* toekomstbestendige keuzes maken voor de nieuwste generatie klokken. Aangezien dit *het* juiste *tijdstip* is om deze nieuwe functionaliteit te ontwerpen en realiseren, heeft dit project de naam **Kairos**¹ gekregen.

Het doel van het project is het realiseren van een werkend prototype. De opgedane kennis, met name op het gebied van draadloze communicatie en een backup batterij, worden uitgebreid gedocumenteerd.

Het doel zal worden bewerkstelligd door een projectmatige aanpak. Voor de uitvoering van het project wordt een team van vijf embedded systems engineers samengesteld, waarbij één van hen de rol van projectleider vertegenwoordigd. Er wordt eerst een plan van aanpak opgesteld, waarin uitgebreid beschreven wordt wat er precies opgeleverd wordt en de manier waarop dat gedaan wordt. Dit plan van aanpak wordt besproken met de opdrachtgever en vormt een go/no-go-beslissingsmoment. Het totale project kent een doorlooptijd van zes maanden en wordt volgens het V-model uitgevoerd en gedocumenteerd. Er zal op regelmatige basis overleg plaatsvinden met de opdrachtgever om de voortgang te monitoren en het project bij te sturen.

ESE clockworks stelt nadrukkelijke de randvoorwaarde dat het prototype gerealiseerd wordt met een FRDM-KL25Z ontwikkelkit. Deze ontwikkelkit beschikt namelijk over uitstekende low power eigenschappen en de mogelijkheid om een backup batterij te gebruiken. Daarnaast tonen de huidige digitale klokken van *ESE clockworks* de datum en tijd op een 64x128 OLED scherm met een doorsnede van 0.96 inch. In dit project dient eenzelfde display te worden toegepast omwille van compatibiliteit. Het synchroniseren van de tijd moet draadloos gaan plaatsvinden. Welk medium daarvoor gebruikt zal worden (infrarood, bluetooth, wifi, etc.) wordt in een latere fase van het project bepaald. Dat betekent dat de klok in principe een ontvanger is, maar dat er ook een zender gerealiseerd moet worden. De focus van dit project is de echter de ontvanger. De zender is een PC of laptop waarop reeds bestaande applicaties gebruikt mogen worden om de tijd te verzenden. Welke applicaties dat zal zijn wordt in de ontwerpfase van dit project bepaald. Er hoeft dus geen PC applicatie te worden ontwikkeld. Aangezien het prototype ook voor promotionele doeleinden gebruikt gaat worden, heeft *ESE clockworks* de wens dat er een leuke gimmick gerealiseerd wordt: houdt iemand zijn hand voor de klok, dan moet de bedrijfsnaam op het display getoond worden.

Om de hoofdvraag van dit rapport te beantwoorden, beschrijft hoofdstuk 2 *Vooronderzoek* vragen en antwoorden die van belang zijn bij aanvang van het project. In hoofdstuk 3 worden de functies van het systeem beschreven. Met andere woorden, **wat** het systeem moet kunnen. In hoofdstuk 4 wordt het systeem op hoofdfuncties ontworpen. Hier wordt beschreven **hoe** het systeem gerealiseerd kan worden. De hardware interfaces worden gespecificeerd, alsmede de gekozen software architectuur. Vervolgens beschrijft hoofdstuk 5 in detail hoe de hardware en software van dit systeem gerealiseerd

¹ [https://nl.wikipedia.org/wiki/Kairos_\(mythologie\)](https://nl.wikipedia.org/wiki/Kairos_(mythologie))

zijn. De testen die uitgevoerd zijn met het prototype worden beschreven in hoofdstuk 6. En tot slot volgen in hoofdstuk 7 de conclusies en aanbevelingen.

2 Vooronderzoek

In een vooronderzoek worden vragen en antwoorden geformuleerd met betrekking tot onzekerheden om het project succesvol te kunnen realiseren. In het Plan van Aanpak is al een afweging gemaakt voor de tijd/kwaliteit verhouding van de vooronderzoeken. Als er nog te veel vraagtekens zijn en er nauwelijks tijd is voor een vooronderzoek, moet je je afvragen of je het project wel aanneemt. Een vooronderzoek kan betrekking hebben op iedere fase van het project. Tijdens het uitvoeren van het project zal er waar nodig verwezen worden naar de conclusies van dit vooronderzoek, om bijvoorbeeld keuzes te onderbouwen.

Vooronderzoeken kunnen volgens verschillende methoden uitgevoerd worden, zoals empirisch, deskresearch, raadplegen van een expert, literatuurstudies, rapid prototyping, enquêtes, etc. Als het vooronderzoek veel pagina's omvat, wordt aangeraden het onderzoek in de bijlagen te beschrijven en in dit hoofdstuk alleen de reden van het onderzoek en de conclusie te herhalen.

Hieronder worden enkele voorbeelden gegeven van algemene onderwerpen waar onderzoek naar gedaan zou kunnen worden.

- *Het meten van een signaal. Welke sensoren zijn daarvoor beschikbaar en voldoen aan de eisen van de opdracht.*
- *Experimenteren met door de opdrachtgever opgelegde hardware om gevoel te krijgen voor de toepasbaarheid in het gegeven project.*
- *De selectie van componenten, zoals de microcontroller.*
- *De selectie van een protocol voor datacommunicatie.*
- *Bepalen welke sensoren en/of actuatoren geschikt zijn voor de opdracht.*
- *Etc.*

Een hulpmiddel voor het maken van keuzes is de beslismatrix [2].

2 Vooronderzoek - Case study Kairos

Dit vooronderzoek beschrijft de geschiktheid van de FRDM-KL25Z ontwikkelkit voor dit project en manieren waarop een hand voor de klok gedetecteerd kan worden.

2.1 FRDM-KL25Z geschikt maken voor backup batterij

Op de FRDM-KL25Z ontwikkelkit zijn voorbereidingen getroffen om een backup batterij te gebruiken. Hoewel de voorbereidingen helpen om relatief eenvoudig een backup batterij toe te passen, moeten er wel verschillende hardware aanpassingen gedaan worden. De user manual [3] beschrijft deze aanpassingen en de implicaties daarvan worden in dit vooronderzoek stap-voor-stap besproken.

1. Extra componenten

In de user manual [3] wordt in *Table 3. FRDM-KL25Z Power Supplies* aangegeven welke opties er zijn voor voeding van de ontwikkelkit. Een belangrijke opmerking heeft betrekking op het plaatsen van de batterijhouder:

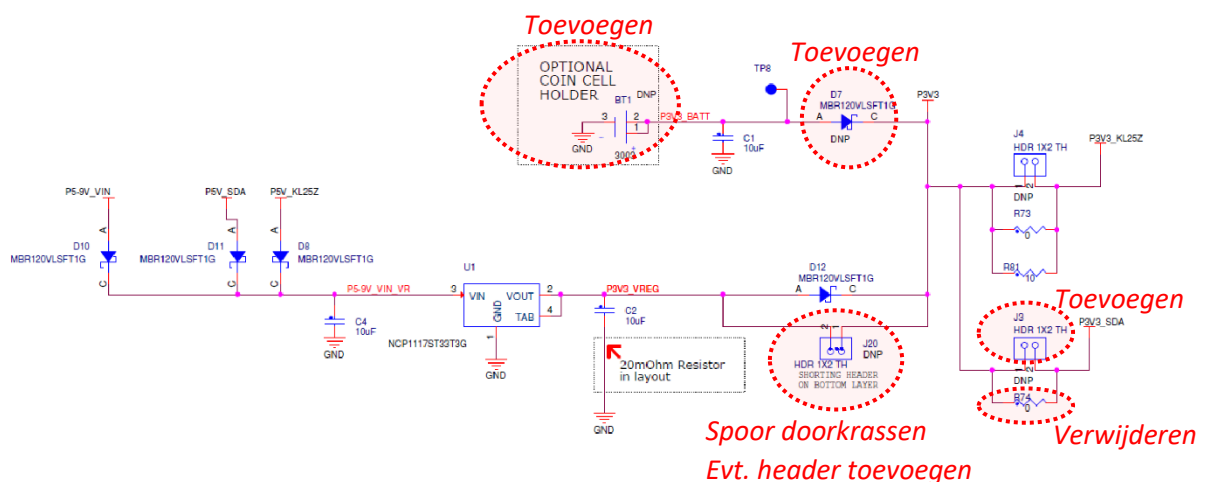
“If a coin cell battery is to be used add a small amount of solder to the coin cell ground pad before adding the battery holder. Also, it is recommended to populate D7 as a protection diode when using a coin cell battery.”

Een geschikte batterijhouder is te bestellen bij Mouser met bestelnummer [534-3003](#).

Door grote tekorten op de componentenmarkt is de voorgeschreven Shottky diode D7 niet verkrijgbaar. Een geschikt alternatief is op het moment van schrijven te bestellen bij Farnell met bestelnummer [2918857](#).

2. Aanpassingen board

Figure 3. Power Supply Schematic [3] toont het schema van het voedingscircuit. Dit schema is overgenomen in Figuur 1 en is van annotaties voorzien. De beschrijvingen van de annotaties gaan verder na Figuur 1.



Figuur 1. Geannoteerd voedingscircuit.

Zoals reeds bij punt 1 aangegeven moeten BT1 en D7 worden toegevoegd.

Shottky diode D12 zorgt er normaal gesproken voor dat er geen stroom van de batterij naar de U1 kan lopen. D12 is echter kortgesloten door J20 in de vorm van een printspoor (op de

bottom layer van de PCB). De reden hiervoor is dat er dan geen spanningsval over D12 is, omdat normaal gesproken er toch geen backup batterij is. Wordt de backup batterij wel gebruikt, dan moet J20 worden doorgekrast. Eventueel kan er een header op J20 worden geplaatst om later het doorgekraste spoor met een jumper ongedaan te maken.

Om zoveel mogelijk energie te besparen als het systeem door een batterij van spanning wordt voorzien, kan weerstand R74 worden verwijderd. Hiermee wordt voorkomen dat de SDA interface 3.3V voeding krijgt. Om toch de SDA interface te kunnen gebruiken tijdens het ontwikkelen (voor programmeren en debuggen), moet er een header worden geplaatst op J3.

Door deze aanpassingen te realiseren staat de backup batterij parallel aan de andere spanningsbronnen (P5-9V_VIN, P5V_SDA en P5V_KL25Z). Om te voorkomen dat de backup batterij te snel leeg wordt getrokken tijdens ontwikkelwerkzaamheden, kan de backup batterij tijdens die werkzaamheden het beste verwijderd worden. Om te voorkomen dat andere devices spanning krijgen die de backup batterij zouden kunnen leegtrekken, moet via bijvoorbeeld de ADC gecontroleerd worden of een andere spanningsbron aanwezig is. Zo nee, dan moeten er in software maatregelen worden genomen om de andere devices zo veel mogelijk uit te schakelen.

2.2 Detecteren van een hand voor de klok

Voor het detecteren van een hand voor de klok zijn experimenten gedaan met twee sensoren. Deze experimenten worden in detail beschreven in Bijlage X. De eerste sensor is de TCRT5000 [4]. Dit is een afstandssensor die gebruikmaakt van infrarood licht. De tweede sensor is de HC-SR04 [5]. Dit is een afstandssensor die gebruikmaakt van ultrasoon geluid.

De conclusie van het onderzoek is dat beide sensoren geschikt zijn voor het detecteren van een hand. De TCRT5000 heeft echter twee significante voordelen:

1. De vormfactor.
De HC-SR04 is groot t.o.v. 0.96 inch OLED display. De aandacht van een gebruiker zou dan te veel naar de sensor kunnen gaan en het complete systeem zou 'bulky' aanvoelen. De TCRT5000 past qua formaat uitstekend.
2. De prijs.
De TCRT5000 is ongeveer vijf keer goedkoper. Er moeten nog wel twee weerstanden bij aangeschaft worden, maar die zijn qua prijs te verwaarlozen.

Voor de volledigheid wordt in Tabel 1 de beslismatrix weergegeven die is vastgesteld op basis van de experimenten in Bijlage X. Ieder aspect wordt gescoord op een schaal van 1 tot 5.

Tabel 1: Beslismatrix voor het kiezen van een sensor om een hand te detecteren.

	Kosten	Vormfactor	Interface	Stroomverbruik	TOTAAL
Weegfactor	3	4	3	1	
HC-SR04	$1 \times 3 = 3$	$1 \times 4 = 4$	$3 \times 3 = 9$	$3 \times 1 = 3$	19
TCRT5000	$5 \times 3 = 15$	$5 \times 4 = 20$	$3 \times 3 = 9$	$4 \times 1 = 4$	48

Het product zal dus ontwikkeld worden met de TCRT5000 reflectieve optische sensor voor het detecteren van een hand voor de klok.

3 Functioneel ontwerp

*Het maken van een functioneel ontwerp heeft als doel het verkrijgen van een complete specificatie van het te ontwikkelen systeem in overleg met de klant/opdrachtgever. Je kijkt hierbij met de ogen van de klant. Een goed functioneel ontwerp is onmisbaar bij het ontwikkelen van een product. Het moet immers voldoen aan de eisen en wensen van de klant/opdrachtgever. Als deze niet tevreden is, is al het werk in feite voor niets geweest. Houd bij het schrijven van het functioneel ontwerp continue in gedachten dat er beschreven wordt **wat** het te ontwikkelen product moet doen. Niet **hoe** dat moet gebeuren.*

De uitvoerder van het project vertaalt het pakket van specificaties van de klant/opdrachtgever in een gedetailleerde functionele specificatie. Er wordt onderscheid gemaakt tussen zuiver functionele specificaties en technische specificaties. Het is de kunst om tijdens het opstellen van de specificaties genoeg detail aan te brengen, zodat opdrachtgever en opdrachtnemer elkaar begrijpen, maar ook niet te veel detail, want dat gaat ten koste van de doorlooptijd van het project. Daarbij blijkt het vaak onmogelijk om alle details exact te vangen. Het is dan raadzaam om af te spreken in iteraties te werken, waarbij de specificaties gaandeweg uitgebreid worden.

3.1 Functionele specificaties

In de functionele specificatie wordt nauwkeurig vastgelegd welke functies het te realiseren systeem allemaal moet kunnen uitvoeren. De basis hiervoor is gelegd in het hoofdstuk inleiding. Het vastleggen wordt gedaan aan de hand van de SMART criteria [6]. Idealiter zou elke specificatie derhalve aan de volgende criteria moeten voldoen:

Specifiek – Specificaties moeten specifiek zijn en niet generiek. Ze mogen niet openstaan voor verkeerde interpretaties wanneer ze door anderen worden gelezen.

Meetbaar – Specificaties moeten kwantificeerbaar zijn om de voltooiing ervan te verifiëren. Voorkom dat een specificatie niet volledig kan worden geverifieerd door het gebruik van niet-kwantitatieve termen (beste, optimaal, snelste) te vermijden.

Haalbaar – Zorg ervoor dat de specificatie realistisch kan worden bereikt, gegeven de bestaande omstandigheden en beschikbare middelen.

Relevant – Specificaties moeten relevant zijn voor een project. Hoewel alle specificaties belangrijk zijn, krijgen ze een prioriteit om in een vroeg stadium aan te geven welke specificaties voor het bedrijf de meeste toegevoegde waarde bieden. Prioriteiten worden toegekend volgens de MoSCoW methode [7]. Engineers zullen in eerste instantie proberen om alle Must-have, Should-have en Could-have-specificaties te realiseren, maar de Could-have en Should-have specificaties zullen als eerste worden los gelaten als de doorlooptijd van het project in gevaar komt. De Won't-have specificaties geven de mogelijkheid om het project af te bakenen.

Tijdgebonden – Elke specificatie moet tijdgebonden zijn of specificeren wanneer of hoe snel een eis moet worden ingevuld of uitgevoerd.

Gebruik een tabel om de functionele specificaties overzichtelijk en gegroepeerd weer te geven. Er worden maximaal drie niveaus aangebracht om een functionele specificatie steeds verder uit te diepen en daarmee SMARTer te maken. Geef iedere specificatie ook een identificatienummer, zodat er in de rest van het rapport makkelijk naar verwezen kan worden.

SMART functionele specificaties		
#	MoS CoW	Omschrijving
F1	M	SMART functionele specificatie van het hoogste niveau.
F1.1	M	SMART functionele sub specificatie met een specifiekere uitwerking.
F1.2	M	SMART functionele sub specificatie met een specifiekere uitwerking.
F2	S	SMART functionele specificatie van het hoogste niveau. Deze specificatie is dermate duidelijk beschreven dat er geen specifiekere uitwerking nodig is.
F3	M	SMART functionele specificatie van het hoogste niveau.
F3.1	M	SMART functionele sub specificatie met een specifiekere uitwerking.
F3.1.1	M	SMART functionele sub-sub specificatie met een specifiekere uitwerking.
F3.1.2	C	SMART functionele sub-sub specificatie met een specifiekere uitwerking.
F3.2	C	SMART functionele sub specificatie met een specifiekere uitwerking.
F3.2.1	C	SMART functionele sub-sub specificatie met een specifiekere uitwerking.
F3.2.2	C	Fun SMART functionele ctionele sub-sub specificatie met een specifiekere uitwerking.
F3.3	S	SMART functionele sub specificatie met een specifiekere uitwerking.
F3.3.1	S	SMART functionele sub-sub specificatie met een specifiekere uitwerking.
F3.3.2	S	SMART functionele sub-sub specificatie met een specifiekere uitwerking.
F4	M	SMART functionele specificatie van het hoogste niveau. Deze specificatie is dermate duidelijk beschreven dat er geen specifiekere uitwerking nodig is.

3.2 Technische specificaties

Bij de technische specificaties staan aanvullende specificaties die door de klant/opdrachtgever worden gesteld aan het product. Dit kan bijvoorbeeld zijn het type microcontroller of de programmeertaal waarmee het product moet worden gerealiseerd. Gebruik een tabel om de technische specificaties overzichtelijk en gegroepeerd weer te geven. Geef iedere specificatie ook een identificatienummer, zodat er in de rest van het rapport makkelijk naar verwezen kan worden.

SMART technische specificaties		
#	MoS CoW	Omschrijving
T1	M	SMART technische specificatie van het hoogste niveau.
T1.1	M	SMART technische sub specificatie met een specifiekere uitwerking.
T1.2	M	SMART technische sub specificatie met een specifiekere uitwerking.
T2	M	SMART technische specificatie van het hoogste niveau. Deze specificatie is dermate duidelijk beschreven dat er geen specifiekere uitwerking nodig is.
T3	C	SMART technische specificatie van het hoogste niveau. Deze specificatie is dermate duidelijk beschreven dat er geen specifiekere uitwerking nodig is.

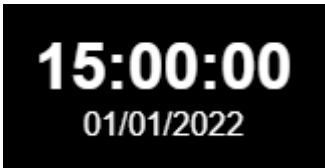


3.3 User interface



Geef een schets van het uiterlijk van het te ontwikkelen systeem. Hoe komt het systeem er voor de gebruiker uit te zien? Welke inputs zijn er en welke outputs? Wat verandert er aan de outputs als een bepaalde input verandert? Een schets van de user interface is een andere weergave van hetzelfde systeem zoals beschreven in de voorgaande paragrafen. Het is een communicatiemiddel tussen opdrachtgever en -nemer om de functionaliteit van het te realiseren systeem helder te krijgen.

3 Functioneel ontwerp - Case study Kairos

3.1 Functionele specificaties

De hoofdfunctie van het systeem is het weergegeven van de tijd. In overleg met de opdrachtgever is de volgende lijst van functionele specificaties opgesteld aan de hand van de SMART criteria [6]. De specificaties zijn in de tweede kolom voorzien van prioritering volgens de MoSCoW methode [7].

Functionele specificaties		
#	MoS CoW	Omschrijving
F1	M	De klok geeft de tijd weer.
F1.1	M	De tijd wordt weergegeven op een 64x128 OLED scherm.
F1.2	S	De tijd wordt op drie verschillende manier weergegeven. Lettertypes en andere weergave opties worden afgestemd met de opdrachtgever, zodra er een eerste prototype beschikbaar is.
F1.2.1	M	De klok toont de tijd groot en de datum klein. 
F1.2.2	M	De klok toont de datum groot en de tijd klein. 
F1.2.3	S	De klok toont de tijd analoog en de datum. 
F1.3	M	Met een switch (werknaam SW1) wordt de schermmodus zoals beschreven bij F1.2 geselecteerd. De switch gaat als het ware cyclisch door de verschillende schermmodi. Dus als de laatste weergaveoptie geselecteerd is, wordt de eerste weergaveoptie getoond.
F2	M	De tijd kan op de klok worden aangepast.
F2.1	M	Met een switch (werknaam SW2) wordt de setup geactiveerd. In de setup kunnen de datum en tijd worden ingesteld.

Functionele specificaties		
#	MoS CoW	Omschrijving
F2.2	M	<p>Zodra er op SW2 wordt gedrukt verschijnt de eerste setup optie. Op een instelscherm wordt één van de in te stellen getallen getoond. Met SW1 wordt de waarde met één opgehoogd. Als de waarde buiten het bereik komt, wordt de eerste waarde van het betreffende bereik getoond.</p> 
F2.3	M	Door op SW2 te drukken wordt naar het volgende setup optie gegaan.
F2.4	M	De setup opties worden in de volgende volgorde doorlopen: dag → maand → jaar → uren → minuten → seconden. Bij het verlaten van de laatste optie wordt teruggedaan naar de laatste weergavemodus.
F3	S	De klok toont de naam van het bedrijf.
F3.1	S	<p>Zodra een hand (of een ander object) zich binnen 10cm van de voorkant van de klok bevindt, wordt de naam van het bedrijf weergegeven.</p> 
F3.2	S	Zodra er zich geen object meer binnen 10cm van de voorkant van de klok bevindt, zal de naam van het bedrijf ongeveer 5 seconden in beeld blijven staan. Daarna wordt de schermmodus hervat zoals die was voordat de bedrijfsnaam werd getoond.
F4	M	De klok kan de tijd draadloos synchroniseren met behulp van een app.
F4.1	M	De app is een terminal applicatie die draait op een laptop waarmee de huidige datum en tijd verstuurd wordt. Als uitbreiding op dit project kan er in een later stadium een grafische applicatie worden ontwikkeld.
F4.2	C	De afstand tussen laptop en klok is minimaal 3 meter.
F4.3	W	Er wordt geen informatie van de klok naar de app gestuurd. Er moet in het ontwerp wel rekening mee worden gehouden dat dit in de toekomst mogelijk is.
F5	M	De klok houdt zelfstandig de tijd bij.
F5.1	M	De tijd wordt bijgehouden in seconden nauwkeurig.
F5.2	M	De nauwkeurigheid van bijgehouden tijd is plus of min een minuut per dag.

Functionele specificaties		
#	MoS CoW	Omschrijving
F6	M	De status van de tijd wordt weergegeven.
F6.1	M	Er knippert een LED iedere seconde (100ms aan, 900ms uit).
F6.2	S	De kleur van deze LED is afhankelijk van het volgende: <ul style="list-style-type: none"> De tijd is <i>niet</i> ingesteld door de gebruiker en <i>niet</i> gesynchroniseerd door de app, dan is de LED kleur rood. De tijd is <i>niet</i> ingesteld door de gebruiker en <i>wel</i> gesynchroniseerd door de app, dan is de LED kleur groen. De tijd is <i>wel</i> ingesteld door de gebruiker en <i>niet</i> gesynchroniseerd door de app, dan is de LED uit. De tijd is <i>wel</i> ingesteld door de gebruiker en <i>wel</i> gesynchroniseerd door de app, dan is de LED kleur afhankelijk van welke van de twee acties als laatste plaatsvond. De kleur is dan groen als de laatste actie synchronisatie door de app was en uit als de laatste actie ingesteld door de gebruiker was.

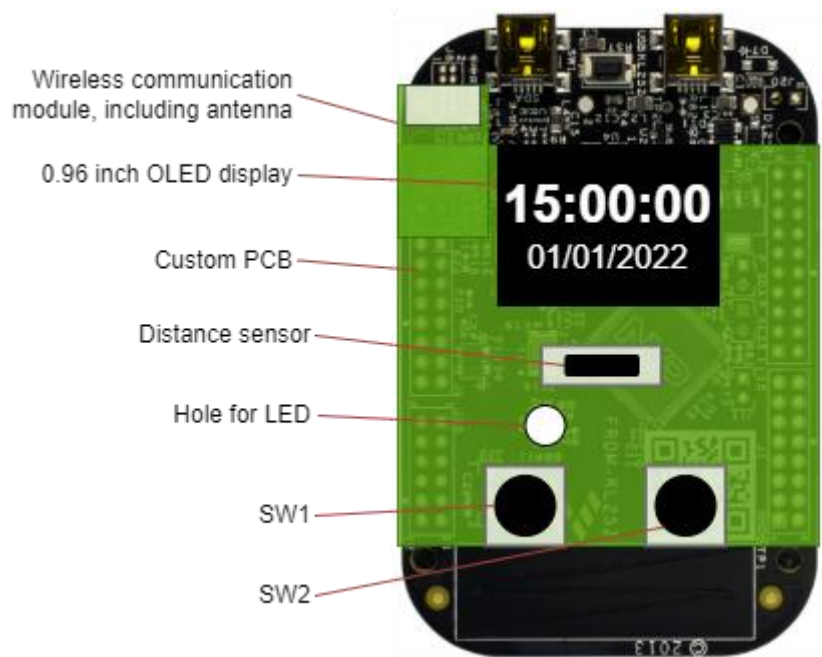
3.2 Technische specificaties

De volgende tabel toont de technische specificaties van het project.

SMART technische specificaties		
#	MoS CoW	Omschrijving
T1	M	Er wordt gebruik gemaakt van de FRDM-KL25Z ontwikkelkit.
T2	M	De hardware wordt gerealiseerd in de vorm van een shield voor op de FRDM-KL25Z ontwikkelkit. Er wordt geen behuizing gerealiseerd.
T3	M	Er wordt geprogrammeerd in de taal C volgens de C17 standaard.
T4	M	Er wordt een backup batterij gebruikt om de tijd actueel te houden als er geen andere voedingsbron is. Om energie te besparen mogen alle overige functies uitgezet worden, zoals het OLED scherm.

3.3 User interface

Om een idee te geven hoe de klok eruit komt te zien wordt een mockup weergave getoond in Figuur 2. In overleg met de opdrachtgever is de plaats van het OLED display vastgesteld. De switches komen onderaan op de PCB. De plaatsing van de overige componenten ligt niet vast en zal mede afhankelijk zijn van de uiteindelijke componentkeuze.



Figuur 2. Overzicht van de user interface.

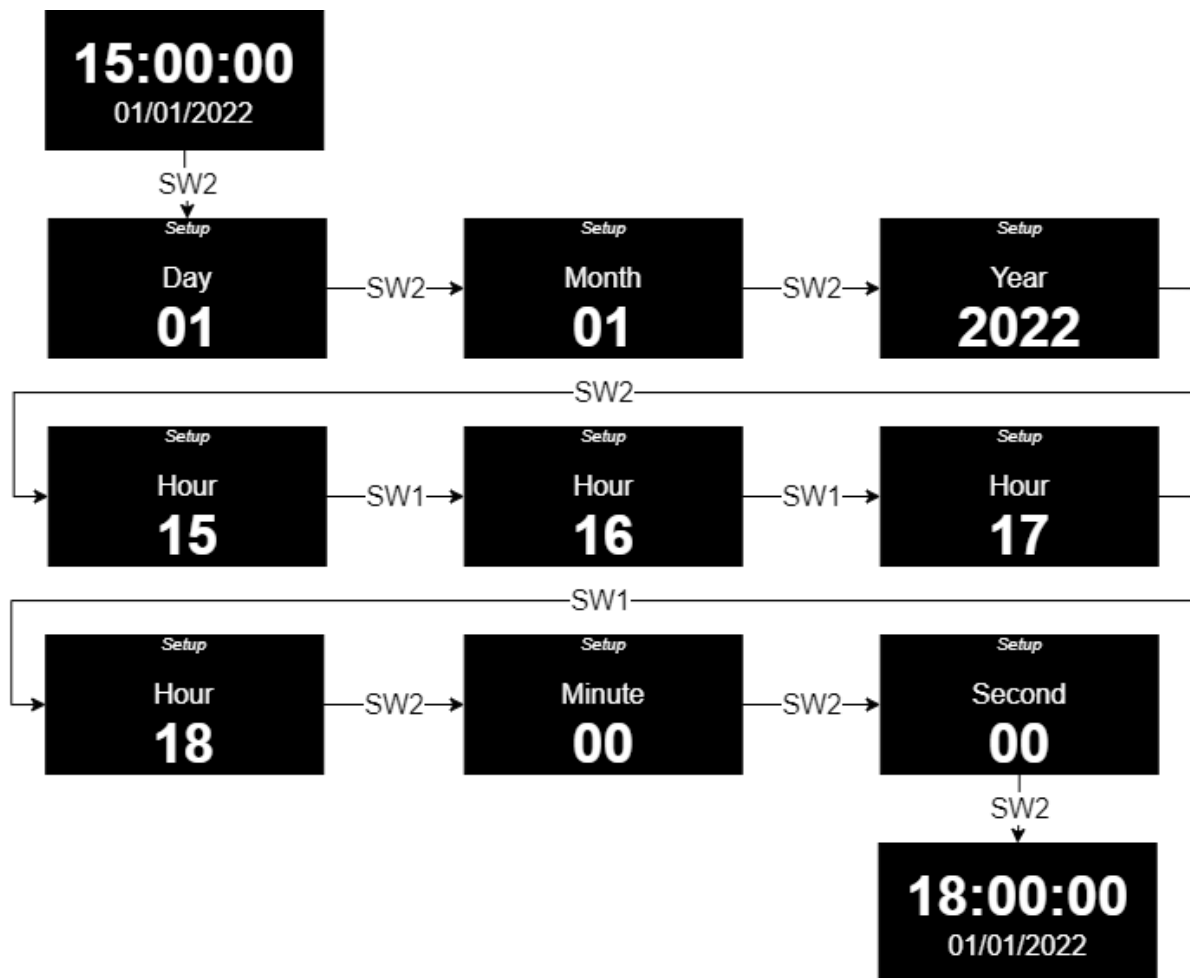
De afbeelding van de het FRDM-KL25Z board is overgenomen van <https://www.nxp.com/design/development-boards/freedom-development-boards/mcu-boards/freedom-development-platform-for-kinetis-kl14-kl15-kl24-kl25-mcus:FRDM-KL2>.

Figuur 3 toont wat er op het OLED display getoond zal worden als SW1 ingedrukt wordt.



Figuur 3. De verschillende tijdsweergaven.

Figuur 4 toont een voorbeeld van het instellen van de tijd. Door op SW2 te drukken, wordt de setup gestart en de tijd als het ware 'bevroren'. Met SW2 wordt in het voorbeeld vervolgens de verschillende setup opties geselecteerd. Aangekomen bij de uren, wordt er drie keer op SW1 gedrukt, om het aantal uren met drie te verhogen. Daarna wordt de setup afgesloten door drie keer op SW2 te drukken. De aangepaste tijd wordt weergegeven op het scherm en de seconden beginnen weer op te tellen.



Figuur 4. De setup opties en een voorbeeld van het aanpassen van de tijd.

4 Technisch ontwerp

In deze fase wordt nagedacht over de wijze van realisatie van de diverse functies. Het gaat hierbij om het **hoe** van het te ontwikkelen product. Het is niet de bedoeling dat er nieuwe specificaties worden beschreven. Dat is het wat en staat beschreven in het voorgaande hoofdstuk. Het technisch ontwerp heeft als doel het functioneel ontwerp te vertalen in een technische implementatie. Je kijkt nu met de ogen van de ontwerper.

4.1 Architectuur

Het systeem wordt eerst onderverdeeld in deelsystemen. Elk deelsysteem heeft een sterke interne samenhang en relatief weinig interactie met de overige deelsystemen. De samenhang tussen de deelsystemen wordt weergegeven in een architectuurschema. Er wordt een onderbouwde keuze gemaakt voor de interface tussen de deelsystemen op basis van de functionele specificaties.

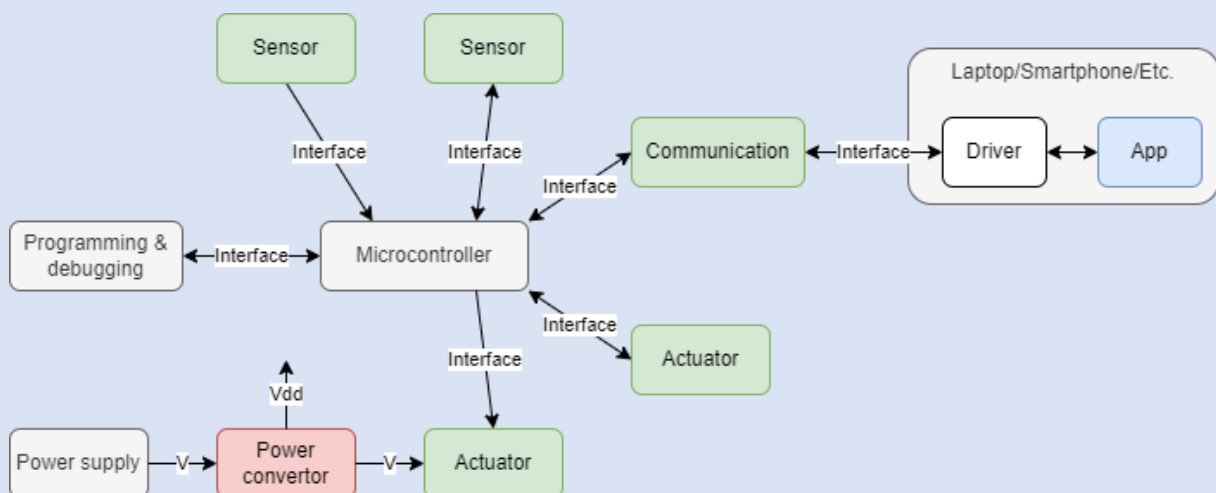
In Figuur 5 wordt een architectuur getoond die algemeen toepasbaar is voor embedded systemen. Het hart is de microcontroller die communiceert via verschillende interfaces met de deelsystemen. Deze interfaces moeten ondubbelzinnig worden vastgelegd in deze fase van het project.

Sensoren hebben een pijl richting de microcontroller, bijvoorbeeld bij een analoge meting, maar kunnen ook een dubbele pijl hebben, bijvoorbeeld bij een seriële bus zoals I2C. Dit laatste geldt ook voor actuators, maar actuators kunnen ook gerealiseerd worden met een enkele pijl richting de actuator. Denk bijvoorbeeld aan een PWM signaal.

Een veel toegepast subsysteem is communicatie met een ander apparaat, zoals een laptop, smartphone, etc. Die interface communiceert doorgaans ook in twee richtingen, maar dat hoeft niet.

Het is gebruikelijk om er rekening mee te houden dat de microcontroller van software updates moet kunnen worden voorzien. Daarom is er vaak sprake van een programmeer en/of debugging subsysteem. Er zijn verschillende interfaces waarmee dat mogelijk is, zoals SWD en JTAG.

Tot slot wordt er getoond hoe de voedingshuishouding wordt geregeld. Vanaf een spanningsbron wordt een spanningsomvormer gebruikt om de voedingspanning (Vdd) voor het embedded systeem te realiseren. Om het schema overzichtelijk te houden wordt Vdd niet naar alle subsystemen getekend. Het kan nodig zijn om meerdere spanningsniveaus in het embedded systeem beschikbaar te hebben, bijvoorbeeld voor het aansturen van motoren.



Figuur 5. Algemeen architectuurschema voor een embedded systeem.

4.2 Interfaces

Voor iedere interface wordt beschreven wat de elektrische- en/of de datacommunicatie-eigenschappen zijn. Soms worden deze keuzes gedicteerd door de (technische) eisen, maar vaak heb je hier als ontwerper ook keuzes te maken. Tevens wordt er voor iedere interface tussen de microcontroller en overige modules middels een UML sequencediagram een ontwerp voor de softwaredriver gemaakt.

4.2.1 Voedingsspanning

De voedingsspanning specificeert welke spanningen er in het systeem nodig zijn, welke spanningsbronnen er zijn, hoe die omgezet worden en welke maximale stroom er verwacht kan worden. Een specificatie wordt duidelijk herkenbaar geformuleerd:

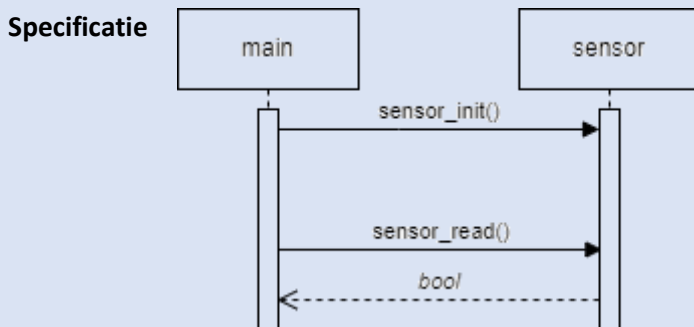
Specificatie Hier komt de tekst van de specificatie.

4.2.2 Microcontroller – Sensor

Voor sensoren geldt dat er beschreven wordt wat de sensor meet. Hierbij wordt waar mogelijk de keuze gekoppeld aan een specificatie. Het is belangrijk om zo volledig mogelijk te zijn, waarbij gedacht moet worden grootheden, eenheden, bereik, precisie, sample frequentie, etc. Een specificatie wordt duidelijk herkenbaar geformuleerd:

Specificatie Hier komt de tekst van de specificatie.

Er wordt tevens beschreven dat er een software driver wordt gerealiseerd. Een driver voor een sensor kent in ieder geval een functie om de driver te initialiseren en één of meerdere functies om waarden van de sensor te lezen. Optioneel kan er een functie voor het schrijven naar de sensor worden beschreven, bijvoorbeeld om de toestand van de sensor te lezen. Kies als prefix voor de namen van de functies de namen die ook in het architectuurschema zijn gebruikt, of een afkorting daarvan.



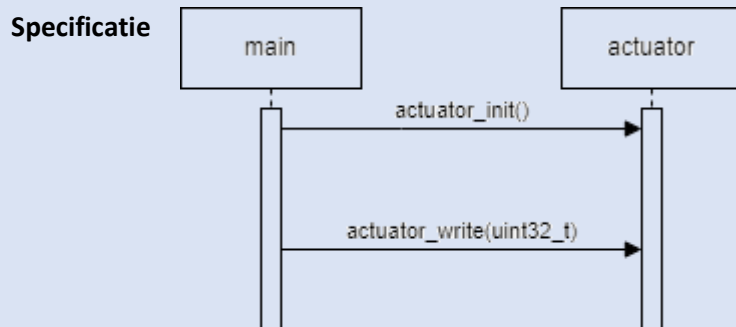
4.2.3 Microcontroller – Actuator

Voor actuatoren geldt dat het uitgangssignaal in zoveel mogelijk detail wordt beschreven. Hierbij wordt waar mogelijk de keuze gekoppeld aan een specificatie. Denk ook hier aan grootheden, eenheden, bereik, precisie, frequentie, etc. Een specificatie wordt duidelijk herkenbaar geformuleerd:

Specificatie Hier komt de tekst van de specificatie.

Er wordt tevens beschreven dat er een software driver wordt gerealiseerd. Een driver voor een actuator kent in ieder geval een functie om de driver te initialiseren en functies om waarden naar de

actuator te schrijven. Optioneel kan er een functie voor het lezen van de actuator worden beschreven, bijvoorbeeld om de toestand van een actuator te lezen. Kies als prefix voor de namen van de functies de namen die ook in het architectuurschema zijn gebruikt, of een afkorting daarvan.



4.2.4 Microcontroller – Communicatie – PC driver – App

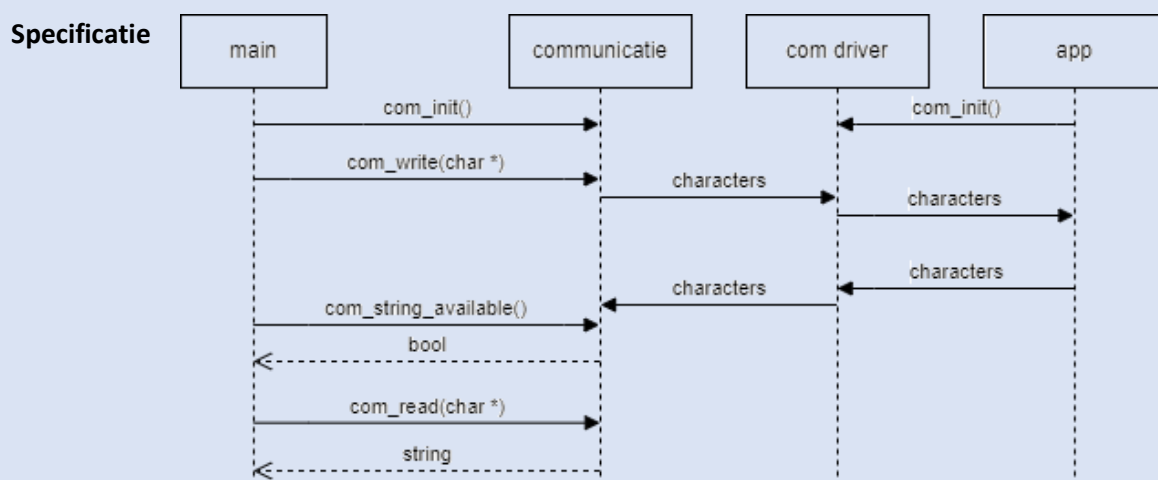
De specificatie van de communicatie met andere apparaten kent twee onderdelen: de interface(s) en het gegevensformaat.

Met betrekking tot de interface moet het volgende worden gespecificeerd:

- elektrisch – spanning, stroom, etc.
- protocol – RS232, I2C, parallel, etc.
- protocolinstellingen – zoals bitrate, etc.

Daarnaast moet ondubbelzinnig vastgelegd worden hoe data tussen de microcontroller main en app wordt uitgewisseld, oftewel het gegevensformaat. Wordt er gebruik gemaakt van een bestaand gegevensformaat (zoals JSON, XML, CSV, etc.), of wordt er een zelfbedacht gegevensformaat geïmplementeerd? In het geval van dat laatste, dan moet dat gegevensformaat in deze paragraaf ondubbelzinnig gespecificeerd worden.

Tevens wordt er beschreven dat er een software driver wordt gerealiseerd. Een driver voor communicatie wordt wel voor een microcontroller gerealiseerd, maar niet voor de PC. Die laatste is namelijk doorgaans beschikbaar. Een communicatiedriver voor een microcontroller kent een initialisatiefunctie, een schrijffunctie en een leesfunctie. De parameters zijn afhankelijk van het gekozen gegevensformaat.



4.3 Software

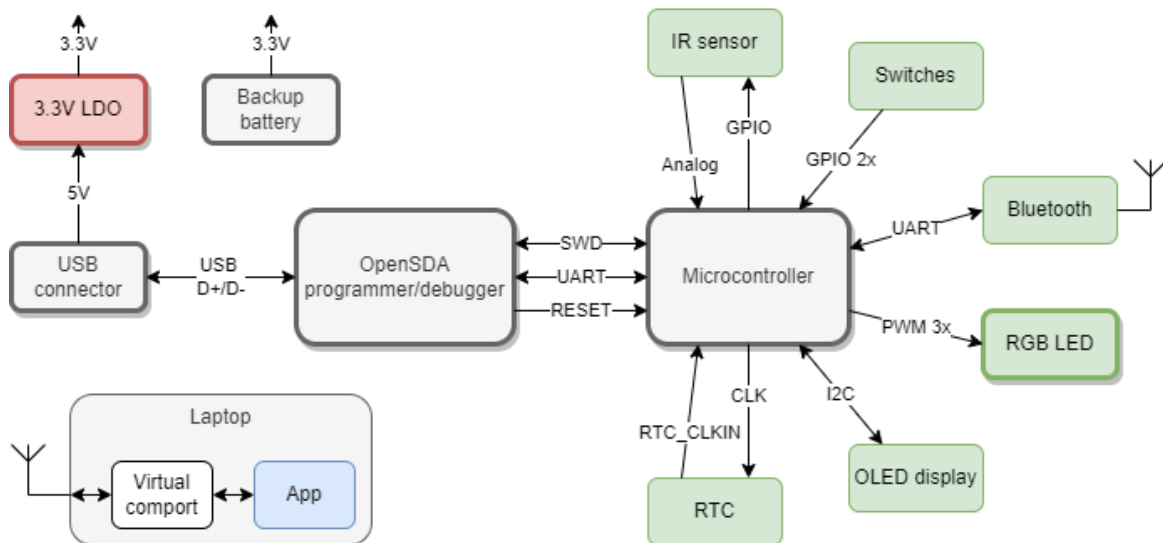
Van het hoofdprogramma worden een of meer ontwerpen van de software getoond en beschreven. Er zijn verschillende methoden om zo'n ontwerpen te beschrijven, zoals een flowchart,

toestandsdiagram, sequencediagram, klassendiagram, etc. Uit de beschrijvingen moet duidelijk blijken welke architectuur er gekozen is, bijvoorbeeld event driven, cyclic executive met interrupts, een RTOS, een toestandsmachine, of iets dergelijks.

4 Technisch ontwerp - Case study Kairos

4.1 Architectuur

Het architectuurschema voor de klok wordt getoond in Figuur 6. De tijd wordt lokaal bijgehouden door een realtime clock (RTC) module. Het OLED display toont die tijd. De tijd wordt via bluetooth draadloos gesynchroniseerd met de app. De RGB LED geeft met kleuren de synchronisatiestatus weer. Er wordt een RGB LED gebruikt voor het tonen van de synchronisatiestatus, maar de blauwe LED wordt niet gebruikt. Via twee switches kan de gebruiker de weergavemodus aanpassen. Om te detecteren of de gebruiker een hand voor de klok houdt wordt een infrarood sensor toegepast.



Figuur 6. Architectuurschema van de klok.

Volgens technische specificatie T1 moet er gebruik gemaakt worden van het FRDM-KL25Z ontwikkelkit. De functionaliteit van de blokken in Figuur 6 met een dikke rand zijn reeds aanwezig op deze ontwikkelkit, waaronder een MKL25Z128VLK4 microcontroller. De specificaties van het board worden beschreven in de user manual [3]. Alle informatie over de microcontroller is beschreven in de datasheet [8] en de reference manual [9].

Via de OpenSDA programmer/debugger kan de microcontroller van software updates worden voorzien. Hierop zal de CMSIS-DAP debug applicatie worden geprogrammeerd [10]. Deze applicatie implementeert naast een programmer/debugger ook een USB CDC interface waarmee serieel gecommuniceerd kan worden tussen een host en de microcontroller. Deze mogelijkheid zal gebruikt worden voor debug doeleinden tijdens de realisatie van dit project.

Om de voedingshuishouding zo eenvoudig mogelijk te houden moeten alle subsystemen op 3.3V werken. Die 3.3V wordt geleverd door een low dropout (LDO) spanningsregelaar en/of een backup batterij. Om het architectuurschema overzichtelijk te houden zijn de 3.3V pijlen naar alle subsystemen weggelaten.

In de volgende paragraaf volgt een gedetailleerde beschrijving van alle interfaces.

4.2 Interfaces

4.2.1 Voedingsspanning

Het systeem kent twee voedingsbronnen. De eerste is 5V via de USB connector. Een low dropout (LDO) spanningsregelaar verlaagt deze tot 3.3V.

Specificatie *De LDO moet een spanning van 5V verlagen tot 3.3V. De maximale stroom is 500mA.*

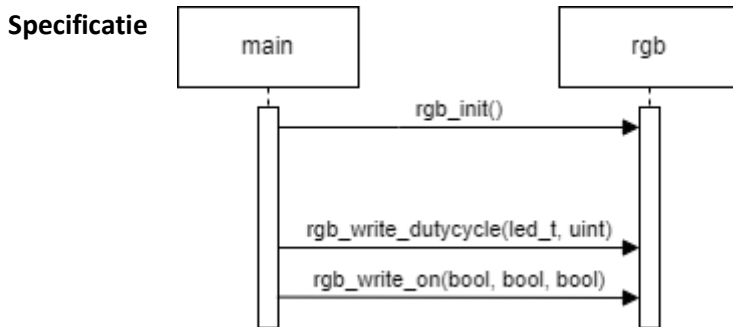
De tweede voedingsbron is de backup batterij. Deze batterij is rechtstreeks aangesloten op dezelfde 3.3V voedingsrail, dus zonder spanningsregelaar, zoals beschreven in het vooronderzoek in paragraaf 2.1.

4.2.2 Microcontroller – RGB LED

De RGB LED op het FRDM-KL25Z board is erg fel als deze met een digitale output pin wordt aangezet. Er worden daarom drie PWM signalen gegenereerd, zodat de LEDs gedimd kunnen worden.

Specificatie *Voor ieder PWM signaal geldt een frequentie van minimaal 100Hz en een instelbare duty cycle van 100 stappen.*

Er wordt een driver gerealiseerd om de hardware te configureren. Het ligt voor de hand om hier een timer voor te gebruiken. Per LED moet kunnen worden aangegeven wat de gewenste duty cycle is. Daarnaast wordt er een convenience functie gemaakt waarmee in één keer aangegeven kan worden van iedere LED of die aan of uit moet zijn. Hiervoor wordt een nader vast te stellen vaste duty cycle ingesteld.



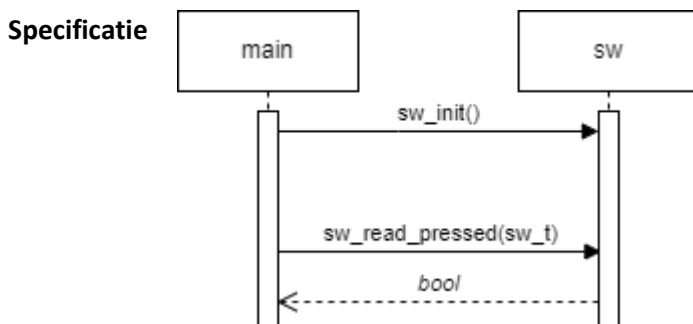
4.2.3 Microcontroller – Switches

De switches worden aangesloten op twee digitale input pinnen. Beide switches zijn laag actief.

Specificatie *Beide switches zijn laag actief:*

- *Niet ingedrukt, dan is de digitale input van de microcontroller logisch 1 (3.3V).*
- *Wel ingedrukt, dan is de digitale input van de microcontroller logisch 0 (GND).*

Er wordt een driver gerealiseerd om de hardware te configureren. Daarnaast wordt er een functie gerealiseerd waarbij de parameter aangeeft van welke switch de toestand gelezen dient te worden. De toestand is true als de switch ingedrukt is (logisch 0) en false als de switch niet ingedrukt is (logisch 1).



De driver leest de momentane waarde van de switch op basis van polling. Dat betekent dat als de switch tussen twee lees-operaties ingedrukt wordt, dan wordt dit niet door de driver onthouden. De main moet dus snel genoeg de switches pollen.

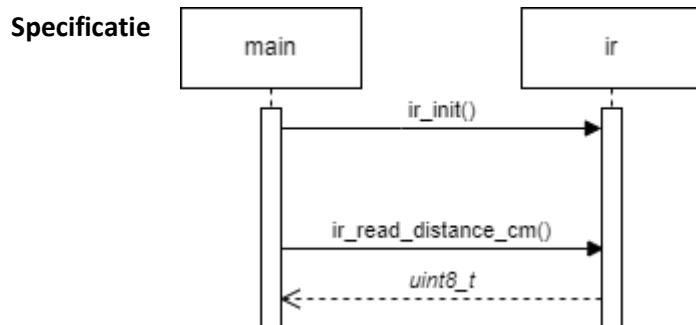
Specificatie *De toestand van beide switches wordt 10 keer per seconde gelezen.*

4.2.4 Microcontroller – IR sensor

Volgens functionele specificatie F3 moet de klok een hand kunnen detecteren die op verschillende afstanden van de klok wordt gehouden. Uit het vooronderzoek dat beschreven wordt in 2.2 Detecteren van een hand voor de klok blijkt dat de TCRT5000 reflectieve optische sensor van Vishay [4] hiervoor het meest geschikt is.

Specificatie *Er wordt 10 keer per seconde gemeten of er een hand gedetecteerd wordt en zo ja, wat de afstand is in cm.*

Er wordt een driver gerealiseerd waarmee de hardware wordt geconfigureerd en die de afstand in cm teruggeeft.



4.2.5 Microcontroller – RTC

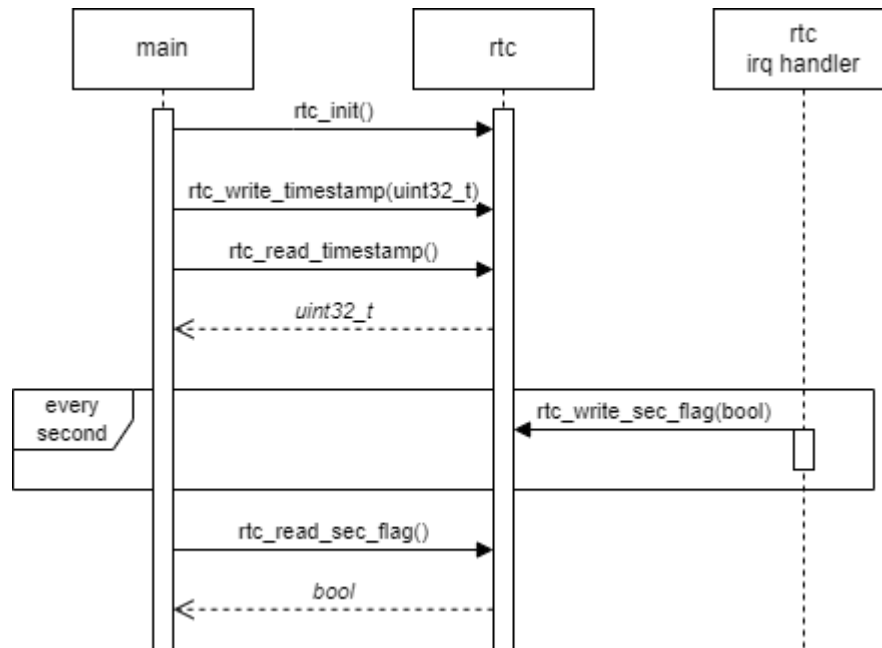
De geselecteerde microcontroller beschikt over een ingebouwde RTC module. Deze module moet extern van een 32.768 kHz kloksignaal worden voorzien. Het is mogelijk om dat kloksignaal met de microcontroller te genereren. Echter moet er dan wel een externe verbinding worden gemaakt tussen twee pinnen van de microcontroller, te weten PTC1 en PTC3. Details hiervan worden beschreven in een thread van de NXP Community [11].

Specificatie *Om de interne RTC te kunnen gebruiken, moeten PTC1 en PTC3 met elkaar worden verbonden.*

Er wordt een driver gerealiseerd om de hardware te configureren. De driver genereert iedere seconde een interrupt. Deze interrupt zet een globale flag die gebruikt wordt in main voor verschillende doeleinden, waaronder het updaten van het display.

Tevens geeft de driver de mogelijkheid om de unix timestamp te lezen en schrijven. De unix timestamp wordt opgeslagen als een 32-bit unsigned integer.

Specificatie



4.2.6 Microcontroller – OLED display

Volgens functionele specificatie F1.1 moet de datum en tijd weergegeven worden op een 128x64 OLED display. Het geselecteerde OLED display wordt aangestuurd met een I2C interface en bevat een SSD1306 single-chip OLED driver van Solomon Systech. De documentatie van deze SSD1306 [12] beschrijft hoe het OLED display aangestuurd wordt.

Tijdens de microcontrollerlessen op de HAN is er reeds met een dergelijk OLED display gewerkt en er is een driver beschikbaar. Er wordt om die reden in deze paragraaf geen ontwerp getoond. Er is geëxperimenteerd met deze driver, zie Bijlage X, en hieruit is gebleken dat de driver alle functies (en meer) implementeert om de functionele eisen te realiseren.

Een belangrijke eigenschap van deze driver is dat er wordt gewerkt op basis van een framebuffer. Dat betekent dat de te tonen data eerst lokaal in de microcontroller in een array wordt opgeslagen, voordat het via I2C naar het OLED display wordt verzonden. Dit verzenden gebeurt op basis van polling.

4.2.7 Microcontroller – Bluetooth – Driver – App

Volgens functionele specificatie F4 moet er draadloos gecommuniceerd worden met een app die op een laptop draait. Er is voor een seriële verbinding gekozen, waarbij de communicatie gerealiseerd wordt via bluetooth. Dat betekent dat de microcontroller via een bluetooth module communiceert en de laptop over bluetooth moet beschikken en de juiste drivers om een virtuele com poort te openen. De microcontroller communiceert met de bluetooth module middels een UART.

Specificatie *Er wordt serieel gecommuniceerd met 9600 bps, 8 databits, 1 stopbit en geen pariteit.*

Communicatie tussen microcontroller en app gebeurt middels strings. Een string bestaat uit meerdere karakters en wordt afgesloten met een LF karakter ('\n'), een CR karakter ('\r') of beiden. Het doel hiervan is synchronisatie. Dat betekent dus dat binnenkomende karakters gebufferd moeten worden en pas verwerkt zodra er CR/LF karakters gedetecteerd worden.

Specificatie *Het maximaal aantal te bufferen karakters is 80.*

Alle karakters in een string zijn leesbare ASCII tekens. Hierdoor is een bericht middels een terminal applicatie te verzenden en te ontvangen.

Specificatie *Alle communicatie tussen microcontroller en app gebeurt in frames met een variabele lengte. Een frame wordt afgesloten met een CR, LF of met beide karakters en daarna wordt de voorgaande data verwerkt.*

Volgens functionele specificatie F4.1 wordt de tijd door de app verstuurd naar de microcontroller. Hiervoor wordt een unix timestamp² gebruikt.

Specificatie *De tijd wordt van de app naar de microcontroller gestuurd in de vorm van de unix timestamp.*

Voorbeelden van geldige unix timestamp strings zijn:

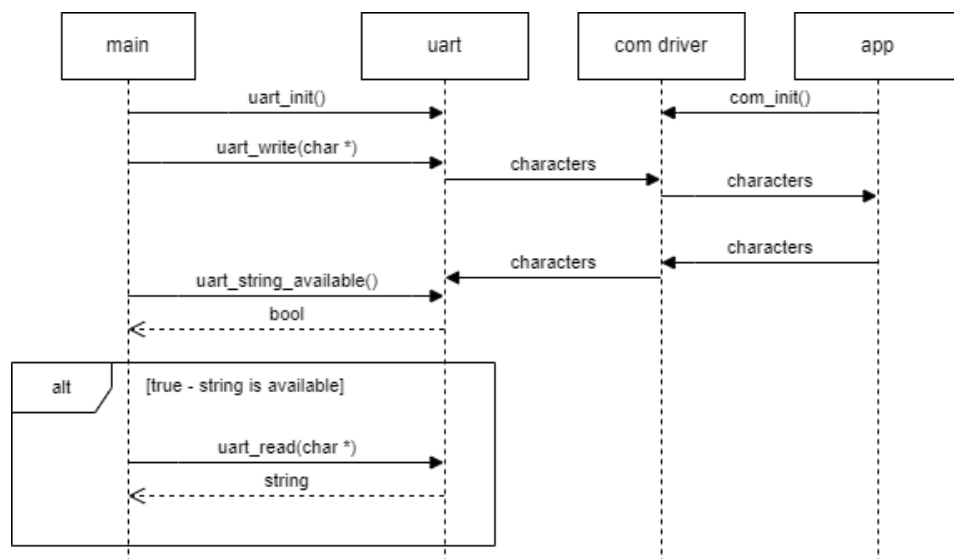
- "0\n"
- "10000\r\n"
- "1650033178\n\r"
- "1640991600\n"

Voor de microcontroller wordt er een driver gerealiseerd die de hardware configureert. Om aan functionele specificatie F4.3 tegemoet te komen, biedt de driver een functie om een string van karakters over te sturen. De driver voegt zelf geen CR/LF karakter toe aan een string, dat zal in main moeten gebeuren.

De driver implementeert een functie om te controleren of het CR/LF karakter is ontvangen. Zo ja, dan biedt de driver een functie om alle karakters die tot dan toe ontvangen zijn te lezen. Alle ontvangen CR/LF karakters wordt verwijderd en het string terminator karakter '\0' wordt aan het eind toegevoegd.

De app initialiseert de com driver. De karakters van de unix timestamp string worden door de app verstuurd. Het maakt niet uit of dat karakter-voor-karakter gebeurt, of dat alle karakters worden verzonden zodra er op bijvoorbeeld Enter wordt gedrukt. Alle binnenkomende karakters worden letterlijk weergegeven in de app.

Specificatie

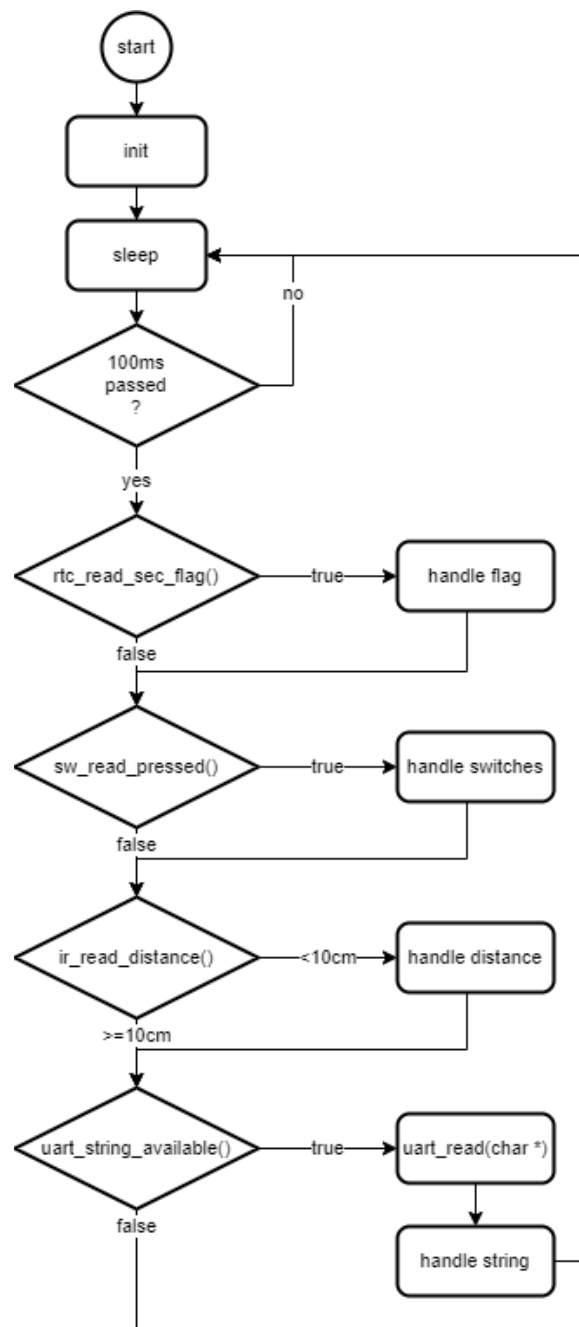


² <https://www.unixtimestamp.com/>

4.3 Software

Het main van de klok wordt middels een *cyclic executive with interrupt scheduler* geïmplementeerd. Deze architectuur zorgt voor een modulaire aanpak en de microcontroller kan in een low-power modus gezet worden als er geen code uitgevoerd hoeft te worden. Een pure *event-triggered* architectuur verdient niet de voorkeur, want er kunnen te veel peripherals tegelijk actief worden, waardoor events met een lage prioriteit te lang op verwerking zouden moeten wachten.

In de gekozen architectuur zal de main-loop 10 keer per seconde worden uitgevoerd. Bij iedere loop wordt er gecontroleerd wat de waarde van een sensor is en of er nieuwe data ontvangen is. Zo ja, dan zal in main de juiste actie worden uitgevoerd. Als alle acties uitgevoerd zijn en er is nog geen 100ms verstreken, dan zal de microcontroller in een low power modus worden gebracht. In Figuur 7 wordt middels een flowchart deze werkwijze getoond.



Figuur 7. Flowchart van de main-loop.

5 Realisatie

Details van de gerealiseerde hardware en software met bijbehorende toelichting en berekeningen (zoals voedingsstromen, waarden van componenten, etc.). Complete detailschema's van de hardware en listings van de software worden in de bijlagen opgenomen.

5.1 Hardware

Aan de hand van aansluitschema's wordt de gerealiseerde hardware toegelicht. Het werkt soms ook verhelderend om een afbeelding op te nemen van bijvoorbeeld een gerealiseerde PCB. Gebruik bij voorkeur afbeeldingen van een deel van het aansluitschema. Niet alles hoeft te worden toegelicht. Kies twee of drie van de meest relevante deelsystemen. Het complete aansluitschema moet terug te vinden zijn in de bijlagen.

5.2 Software

Aan de hand van code snippets wordt de gerealiseerde software toegelicht. Zorg ervoor dat de code goed leesbaar is middels syntax highlighting. Gebruik code snippets die niet langer zijn dan 20 regels en dat iedere regel code op één regel past. Niet alles hoeft te worden toegelicht. Kies twee of drie van de meest relevante deelsystemen. De volledige code wordt opgenomen als bijlage. Besteed ook aandacht aan de software ontwikkelomgeving. Vraag je hierbij af wat belangrijke informatie is voor een collega engineer die voor het eerst dezelfde ontwikkelomgeving gaat gebruiken.

5 Realisatie - Case study Kairos

5.1 Hardware

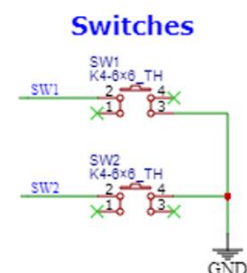
De hardware van de klok wordt beschreven aan de hand van het elektrisch schema en het PCB ontwerp.

5.1.1 Elektrisch schema

Het volledige elektrische schema van de klok is gegeven in Bijlage A. In deze paragraaf volgt een toelichting op de meest belangrijke deelschema's.

5.1.1.1 Switches

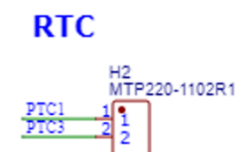
De signalen SW1 en SW2 zijn verbonden met de pinnen PTD3 en PTD5 van de microcontroller. Er zijn geen pullup weerstanden toegepast, omdat die intern in de microcontroller zitten. Ook is er geen anti-bouncer hardware gerealiseerd, omdat de software op basis van polling de logische waarde van de switches controleert.



Figuur 8. Hardware schema switches.

5.1.1.2 RTC

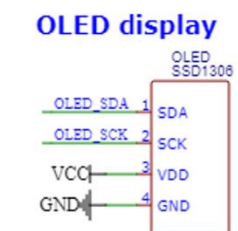
Voor het gebruik van de RTC module in de microcontroller moet er een externe verbinding worden gemaakt tussen PTC1 en PTC3. Deze verbinding wordt gerealiseerd middels een jumper, zodat de pinnen eventueel ook voor andere doeleinden gebruikt zouden kunnen worden. Strikt genomen is namelijk het leggen van een trace tussen deze twee pinnen voldoende.



Figuur 9. Hardware schema RTC.

5.1.1.3 OLED display

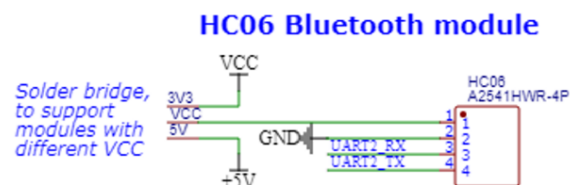
De 64x128 OLED module bevat een 4-pins male header. Op deze module bevinden zich reeds pullup weerstanden voor het I2C signaal, dus die worden niet ook nog eens toegevoegd aan dit ontwerp.



Figuur 10. Hardware schema OLED display.

5.1.1.4 Bluetooth module

Er zijn verschillende geschikte bluetooth modules, waaronder de HC05 en de HC06. In eerste instantie wordt er gekozen voor een HC06, maar als die slecht verkrijgbaar blijkt te zijn, dan is de HC05 pin compatible, op de voedingsspanning na. Via een solder bridge wordt daarom de mogelijkheid gerealiseerd om tijdens assemblage te bepalen welke voedingspanning er door de bluetooth module gebruikt wordt.



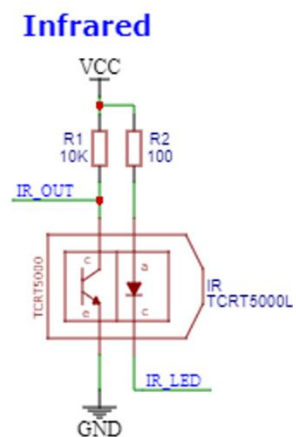
Figuur 11. Hardware schema bluetooth module.

5.1.1.5 IR sensor

De LED van de IR sensor wordt door de microcontroller aangezet met een laag actief signaal, genaamd IR_LED. De spanningsval over de LED is 1.25V, zoals beschreven in de TCRT5000 datasheet [4]. Bij een voedingspanning van 3.3V staat er dus $3.3V - 1.25V = 2.05V$ over weerstand R2. Door 100 Ohm te kiezen voor R2 wordt de stroom door de LED gelijk aan $\frac{2.05V}{100\Omega} = 20.5mA$. Dat is ruim binnen de marge van 60mA die in de datasheet wordt aangegeven als maximale forward stroom.

De collectorstroom mag volgens de datasheet maximaal 100mA bedragen. Met 10k Ohm en 3.3V wordt ook daar ruimschoots binnen gebleven.

Als er geen infraroodlicht gereflecteerd wordt, zal de transistor niet geleiden en de spanning op IR_OUT nagenoeg gelijk zijn aan VCC. Hoe meer de transistor in geleiding gaat ten gevolge van de reflectie door de infrarood LED, hoe lager de spanning op IR_OUT zal zijn. Deze uitgangsspanning is aangesloten op PTB0 (niet getoond in de figuur, wel in Bijlage A – Elektrisch schema), zodat deze met single ended kanaal 8 van ADC0 te lezen is.



Figuur 12. Hardware schema IR sensor.

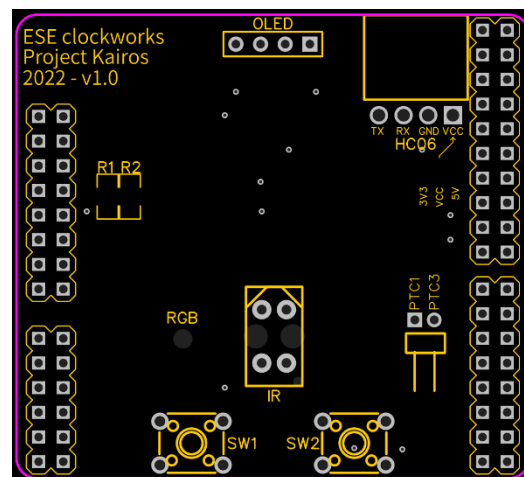
5.1.2 PCB ontwerp

De eerste fase van het PCB ontwerp is het plaatsen van de componenten. Dit wordt getoond in Figuur 13. Alle belangrijke componenten zijn geplaatst, zoals aangegeven door de opdrachtgever (zie ook Figuur 2), behalve de header voor de bluetooth module. Bij nader inzien bleek het handiger om die aan de rechterkant te plaatsen, omdat dan de USB aansluiting van de FRDM-KL25Z ontwikkelkit goed toegankelijk blijft.

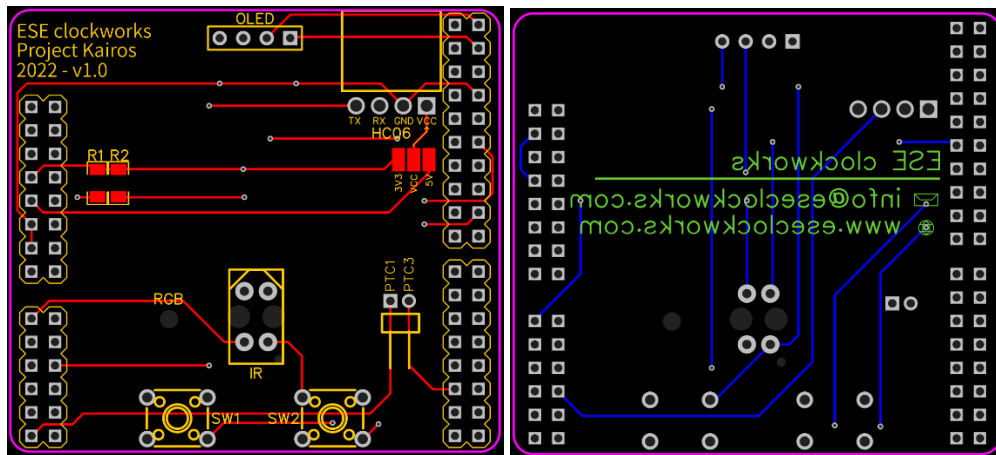
Vervolgens zijn de printsporen aangebracht.

Vanwege de relatieve eenvoud is voor een PCB

met twee lagen gekozen. Om het plaatsen van de printsporen te vereenvoudigen zijn de printsporen op de top-laag voornamelijk in horizontale richting en de printsporen op de bottom-laag in verticale richting. Het resultaat wordt getoond in Figuur 14.

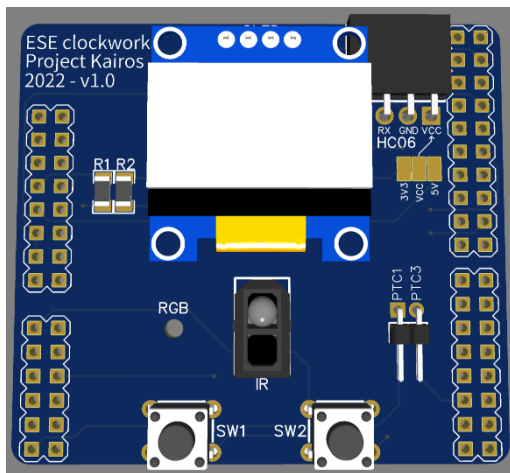


Figuur 13. Plaatsing van de componenten op de PCB.



Figuur 14. Top- (links) en bottomlaag (rechts) van de PCB.

De 3D view in Figuur 15 geeft een goed beeld van het te verwachten gemonteerde resultaat.



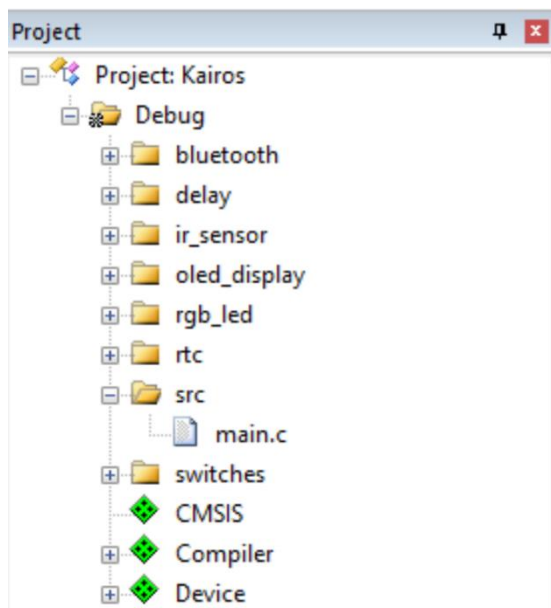
Figuur 15. 3D view van de PCB.

5.2 Software

De volledige listing van de gerealiseerde software is te vinden in Bijlage X. In deze paragraaf worden de belangrijkste software onderdelen toegelicht aan de hand van code snippets. Eerst worden de instellingen van de software ontwikkelomgeving besproken.

5.2.1 Keil μ Vision IDE

Voor het ontwikkelen van de software wordt de community edition van de Keil μ Vision IDE [13] gebruikt. Voor iedere sensor en actuator is er een aparte folder gemaakt. De projectstructuur wordt weergegeven in Figuur 16. Voor het configureren van dit project zijn de aanwijzingen uit application note 232 [14] gevolgd. Meer gedetailleerde instellingen specifiek voor dit project zijn terug te vinden in Bijlage X.



Figuur 16. Projectstructuur in Keil µVision IDE.

5.2.2 Switches

switches\switches.h

```
/// Defines the type for the keys
typedef enum
{
    SW1 = 0,
    SW2,
} sw_t;

// Function prototypes
void sw_init(void);
bool sw_read_pressed(const sw_t sw);
```

De twee switches worden gedefinieerd middels een enum. De parameter sw in de functie sw_read_pressed() wordt const gedeclareerd, omdat er geen reden is om die variabele aan te passen in de functie.

switches\switches.c

```
static PORT_Type * port_mapping[N_SWITCHES] = {PORTD, PORTD};
static GPIO_Type * gpio_mapping[N_SWITCHES] = {PTD, PTD};
static uint8_t pin_mapping[N_SWITCHES] = {3, 5};
```

Middels drie lookup tabellen wordt per switch aangegeven welke PORT peripheral, GPIO peripheral en pinnummer gebruikt worden.

switches\switches.c

```
/*!
 * \brief Initialises the switches on the shield
 *
 * This functions initializes the switches on the shield.
 */
void sw_init(void)
{
    // Enable clocks to PORT
```

```

SIM->SCGC5 |= SIM_SCGC5_PORTD_MASK;

// Configure all pins as follows:
// - MUX[2:0] = 001 : Alternative 1 (GPIO)
// - DSE = 0 : Low drive strength
// - PFE = 0 : Passive input filter is disabled
// - SRE = 0 : Fast slew rate is configured
// - PE = 1 : Internal pullup or pulldown resistor is enabled
// - PS = 1 : Internal pullup resistor is enabled
for(int i=0; i<N_SWITCHES; i++)
{
    port_mapping[i]->PCR[pin_mapping[i]] = 0b00100000011;
}

// Set port pins to inputs
for(int i=0; i<N_SWITCHES; i++)
{
    gpio_mapping[i]->PDDR &= ~(1<<pin_mapping[i]);
}
}

```

In de `sw_init()` functie wordt middels twee for-loops iedere pin geconfigureerd. Met de index `i` worden de juiste peripherals, registers en bitposities geselecteerd uit de lookup tabellen. Het commentaar beschrijft welke bits enabled, dan wel disabled worden.

switches\switches.c

```

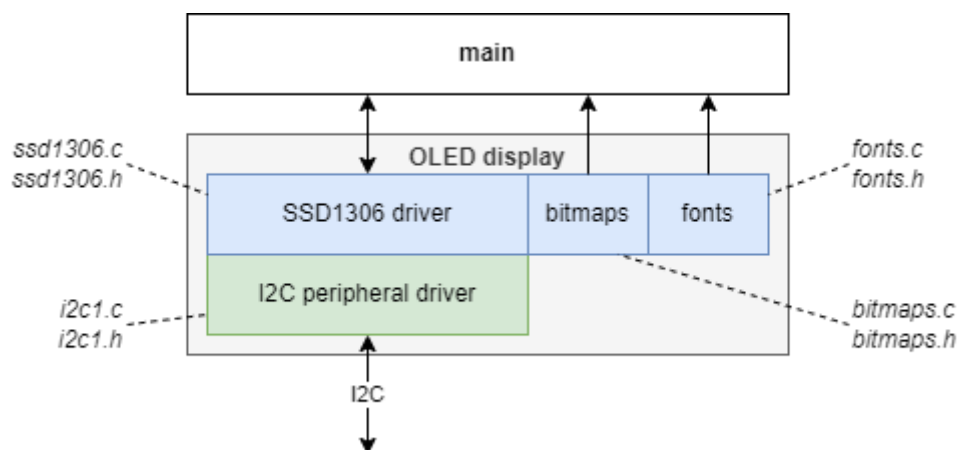
/*!
 * \brief Check if a switch is pressed
 *
 * This functions checks if a switch is pressed. The function simply checks the
 * value of the switch at the moment the function is called. It doesn't
 * remember if the switch has been pressed.
 *
 * \param[in] sw Switch that will be checked, must be of type ::sw_t
 *
 * \return True if the switch is pressed, false otherwise
 */
bool sw_read_pressed(const sw_t sw)
{
    // If the key is pressed, the bit at that position will read logic 0
    return ((gpio_mapping[sw]->PDIR & (1<<pin_mapping[sw])) == 0);
}

```

Op basis van de meegegeven parameter wordt gecontroleerd of die betreffende pin ingedrukt is. Een ingedrukte switch leest als logisch 0, dus wordt er na het bitwise and-en met het masker `(1<<pin_mapping[sw])`, logisch vergeleken met nul. Deze functie gebruikt wederom de lookup tabellen voor het selecteren van de corresponderende peripheral, register en bitpositie. Er wordt niet gecontroleerd of de parameter `sw` wel een waarde heeft binnen het geldige bereik (0 of 1). In zekere zin wordt dat door de compiler afgedwongen doordat het van het type `sw_t` is. Maar theoretisch gezien zou een waarde buiten het bereik van `sw_t` kunnen worden meegegeven. Er zal dan ongedefinieerd gedrag plaatsvinden.

5.2.3 OLED display

De software voor het OLED display is gelaagd gerealiseerd en kent meerdere files. Dit wordt weergegeven in Figuur 17. De blauwe onderdelen zijn zo gerealiseerd dat ze onafhankelijk van de interface zijn. In dit project wordt er gebruik gemaakt van de I2C interface, vandaar de I2C peripheral driver. Van alle onderdelen wordt in de volgende paragrafen de realisatie beschreven.



Figuur 17. Gelaagdheid en files van de software ten behoeve van de OLED display driver.

5.2.3.1 I2C peripheral driver

De I2C peripheral driver implementeert de hardware communicatie via I2C. Alle communicatie is gerealiseerd op basis van polling. Dat betekent dat er gewacht wordt totdat er een byte via de I2C interface verstuurd is. De driver implementeert drie functies, zoals te zien in de header file. Het verschil tussen de twee write functies is de waarde van het control byte die verstuurd wordt nadat het slave adres is verstuurd. Voor een command is dat namelijk 0x00 en voor data is dat 0x40 [12].

oled_display\i2c1.h

```

/*!
 * \brief Definition for the I2C timeout
 *
 * This timeout value is used in loops to wait for a bit to set/reset.
 * If the bit doesn't get set, the function returns.
 */
#define I2C_TIMEOUT (10000)

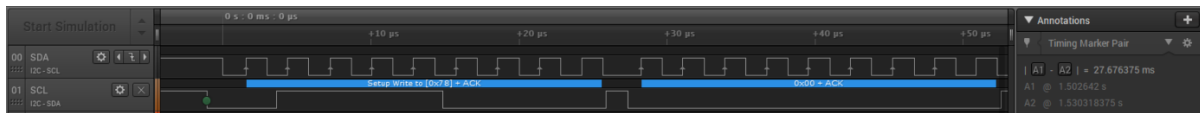
// Function prototypes
void i2c1_init(void);

bool i2c1_write_cmd(const uint8_t address,
                    const uint8_t cmd[],
                    const uint32_t n);
bool i2c1_write_data(const uint8_t address,
                     const uint8_t data[],
                     const uint32_t n);

```

Het pollen is gerealiseerd door in while-loop te wachten en niets te doen zolang het I2C_S_IICIF bit logisch 0 is. Als er een fout op mocht treden, zoals een verkeerd aangesloten display, dan zou dit betekenen dat de applicatie eindeloos in die while-loop blijft. Het I2C_S_IICIF bit blijft dan immers altijd logisch 0. Om een eindeloze lus te voorkomen is er een timeout gedefinieerd. Hoe lang deze timeout duurt is afhankelijk van de core klok. Middels een define is die instelbaar.

Het timing diagram in Figuur 18 toont de start van succesvolle communicatie met het OLED display. Het slave adres 0x78 wordt ge-acknowledged en daarna wordt 0x00 gestuurd om aan te geven dat de bytes die volgen geïnterpreteerd moeten worden als command bytes. Deze command bytes worden in Figuur 18 niet weergegeven.



Figuur 18. Timing diagram van I2C communicatie met het OLED display.

5.2.3.2 SSD1306 driver

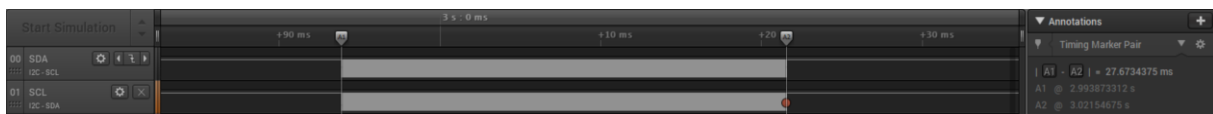
De SSD1306 driver is de interface voor de main applicatie. Er zijn verschillende functies beschikbaar, die de juiste commando's en data via de I2C peripheral driver naar het OLED display sturen. Enkele voorbeelden zijn de functie `ssd1306_setcontrast()` om het contrast in te stellen en `ssd1306_setpixel()` om een pixel op een (x,y) locatie aan of uit te zetten. De te sturen commando's en data zijn bepaald aan de hand van de datasheet van het OLED display [12].

De SSD1306 driver is geïmplementeerd middels een framebuffer waarin 8 pixels in een `uint8_t` worden opgeslagen. Deze is gedeclareerd in de file `ssd1306.c`. De define `SSD1306_SIZE` geeft het aantal pixels gedeeld door 8 en is voor een 128x64 gelijk aan 1024, oftewel 1 kilobyte.

oled_display\ssd1306.c

```
uint8_t ssd1306_framebuffer[SSD1306_SIZE];
```

Alle functies die data weergeven maken gebruik van deze framebuffer. Dat betekent dat bijvoorbeeld het aan of uitzetten van een pixel in de framebuffer gebeurt en niet op het OLED display. Om de framebuffer op het OLED display weer te geven moet de functie `ssd1306_update()` worden aangeroepen. Het timing diagram in Figuur 19 toont dat het ongeveer 27.7ms duurt om de complete framebuffer, dus alle 128x64 pixels te versturen.



Figuur 19. Timing diagram dat het versturen van de framebuffer toont.

Er is voor een framebuffer implementatie gekozen, omdat via I2C de pixels niet gelezen kunnen worden. Het is echter wel noodzakelijk om de waarden van de pixels te kennen als er display updates moeten plaatsvinden. Een bijkomend voordeel is dat het relatief snel is om die framebuffer te updaten, omdat er geen communicatie hoeft plaats te vinden. Hoewel het versturen van een complete framebuffer relatief lang duurt, is deze tijd wel altijd gelijk, namelijk ongeveer 30ms.

5.2.3.3 Fonts

De SSD1306 driver ondersteunt verschillende fonts. Deze fonts worden in font tabellen gedefinieerd in de file `fonts.c`. Het toevoegen van een nieuw font gaat aan de hand van de volgende stappen:

1. Kopieer een font versie $\geq 3.0.0$ van de volgende website:
<http://oleddisplay.squix.ch/>
2. Voeg de gegenereerde code toe aan de file `fonts.c`.
3. Verwijder `PROGMEM` van de declaratie.
4. Voeg een externe declaratie toe aan de file `fonts.h`.
5. Gebruik het font door een pointer mee te geven aan de functie `ssd1306_setfont()`.

5.2.3.4 Bitmaps

De SSD1306 driver ondersteunt het weergeven van bitmaps. Het toevoegen van een nieuwe bitmap gaat aan de hand van de volgende stappen:

1. Maak een 128x64 bitmap, bijvoorbeeld met de volgende tool:
http://en.radzio.dxp.pl/bitmap_converter/

2. Voeg de gegenereerde code toe aan de file `bitmaps.c`.
3. Voeg een externe declaratie toe aan de file `bitmaps.h`.
4. Teken de bitmap door een pointer mee te geven aan de functie `ssd1306_drawbitmap()`.

5.2.4 Main

De main applicatie initialiseert eerst alle software door de `init()` functies aan te roepen en een boodschap op de OLED display te tonen. Vervolgens wordt de eindeloze loop gestart en met de `__WFI()` instructie wordt de “Normal Wait – via WFI” chip power mode geactiveerd, zie reference manual [9] tabel 7.1. In deze power mode werken alle peripherals door en werkt de core in deep sleep mode. De NVIC blijft gevoelig voor interrupts, zodat de core uit de deep sleep mode komt als er een willekeurige interrupt optreedt, zoals een RTC interrupt.

Als de core uit deep sleep mode komt worden de vlaggen gepolled om te controleren welke actie er moet worden uitgevoerd. Het codevoorbeeld hieronder laat dat voor twee voorbeelden zien.

main.c

```

/*!
 * \brief Main application
 *
 * This main application of the Kairos project. It implements a cyclic
 * executive with interrupts scheduling mechanism.
 */
int main(void)
{
    // Initialize all modules and show message on OLED display

    while(1)
    {
        __WFI();
        // Core wakes up from interrupt, continue code execution

        // -----
        // RTC
        // -----
        if(rtc_read_sec_flag())
        {
            // The RTC has generated an interrupt indicating that a
            // second has passed

            // Read the current time
            uint32_t timestamp = rtc_read_timestamp()

            // Convert the timestamp and show datetime based on the mode
            // ...
        }

        // -----
        // SW1
        // -----
        if(sw_read_pressed(SW1))
        {
            // Switch 1 is pressed, handle the event
            // ...
        }

        // Etc.
    }
}

```

6 Testen

Ondubbelzinnige weergave hoe het systeem, de hardware en/of software getest is. Welke hardware en of software modules zijn getest, hoe zijn de functionele specificaties getest tijdens de acceptatietest? Welke testopstelling is gebruikt en wat zijn de uiteindelijke resultaten. Voldoen de testen aan de gestelde eisen? De resultaten worden voorzien van een duidelijk omschrijving welke eventuele problemen er nog zijn en hoe deze mogelijk zijn te verklaren. Zijn er eventuele 'work arounds' uitgevoerd tijdens het testen? De testen moeten zodanig omschreven zijn dat elke test door anderen te reproduceren is.

6 Testen - Case study Kairos

Dit hoofdstuk beschrijft de acceptatietesten en testen ten behoeve van het stroomverbruik die zijn uitgevoerd ten behoeve van het project Kairos.

6.1 Acceptatietesten

Er zijn in totaal vijf testscenario's uitgevoerd. Ieder testscenario heeft tot doel om de werking van een of meerdere functionele specificaties te verifiëren. Er zijn drie mogelijke uitkomsten, waarbij een functionele specificatie **wel** (✓), **deels** (🟡) of **niet** (✗) gerealiseerd is zoals gespecificeerd. In de tabel hieronder staat in één oogopslag het resultaat. In Bijlage B is voor ieder test scenario beschreven welke voorbereidingen er moeten worden getroffen, welke stappen uitgevoerd moeten worden en wat bij iedere stap de te verwachten uitvoer van het systeem is. Als de daadwerkelijke uitvoer gelijk is aan de verwachte uitvoer, dan is de test als correct gemarkeerd. Aan het eind van iedere test volgt eventueel een opmerking.

	Testscenario 1	Testscenario 2	Testscenario 3	Testscenario 4	Testscenario 5
F1	✓				
F2		✓			
F3			🟡		
F4				✗	
F5					✓
F6					✓

6.2 Stroomverbruik

Tijdens de ontwikkeling van het product is er niet specifiek rekening gehouden met het stroomverbruik in termen van eventuele hardware of software aanpassingen. De initiële focus lag op het werkend maken van de gevraagde functionele specificaties.

Wel zijn er metingen gedaan om te bepalen welke hardware aanpassingen aan de FRDM-KL25Z voor welke stroombesparing zorgen. Deze metingen worden in detail beschreven in Bijlage C – Metingen stroomverbruik. In Tabel 2 worden de resultaten samengevat.

Tabel 2. Samenvatting metingen stroomverbruik.

Nr.	Gemeten stroom [μA]	Bespaarde stroom [μA]	Aanpassing aan het FRDM-KL25Z board
1	8290	-	Geen.
2	5080	3210	Doorkrassen trace J20.
3	375	4705	Verwijder R74.
4	375	0	Doorkrassen trace J11.
5	38	337	Doorkrassen trace J14.

Door bovengenoemde aanpassingen te maken verbruikt het FRDM-KL25Z board 38μA. Daarbij moet gezegd worden dat er geen andere hardware actief is, zoals een LED of een OLED display. Aangezien een CR2032-batterij een capaciteit heeft van 225mAh, kan het systeem in theorie 225mAh / 38μA = 5921 uur werken. Dit is gelijk aan 246 dagen.

7 Conclusies en aanbevelingen

*Reflectie op de doelen van het project. Wat zijn de resultaten? Wat is wel en wat is niet gerealiseerd?
Wat kan er aan het product worden aangevuld, uitgebreid, verbeterd?*

7 Conclusies en aanbevelingen - Case study Kairos

De hoofdoelen van het project Kairos waren het gebruik van een ARM microcontroller en de toevoeging van draadloze communicatie voor het synchroniseren van de tijd op afstand. Daarnaast is er inzicht verkregen in het energieverbruik. Deze hoofdoelen zijn allemaal behaald en er is een werkend prototype opgeleverd.

De toegepaste microcontroller beschikt over veel meer peripherals dan noodzakelijk was voor dit project. Voor de realisatie van een product, moet de toegepaste microcontroller ten minste beschikken over 32kB FLASH geheugen, 8kB RAM geheugen, een timer, een UART, een ADC, een RTC, twee GPIO pinnen en low-power mogelijkheden. In de KL familie van NXP zijn hiervoor verschillende oplossingen te vinden, maar fabrikanten als STMicroelectronics en Atmel bieden soortgelijke oplossingen.

Draadloos communiceren is gerealiseerd middels een HC06 bluetooth module. Deze modules zijn, ondanks het heersende tekort aan chips, goed verkrijgbaar en niet duur. Voor de microcontroller is er een seriële (UART) verbinding en voor de meeste besturingssystemen is er een driver beschikbaar die middels een virtuele COM port toegang geeft tot de module. De gegevensoverdracht in dit project tussen app en microcontroller is zo eenvoudig mogelijk gehouden, omdat de focus op het embedded systeem lag. Als het project doorontwikkeld wordt, waarbij er een app ontwikkeld gaat worden, dan valt het aan te bevelen om naar een praktischer gegevensformaat over te gaan, JSON of CSV. Aan de andere kant heeft ASCII strings wel zijn voordelen, bijvoorbeeld omdat de verstuurde tekst leesbaar blijft.

Om inzicht te krijgen in het stroomverbruik zijn er verschillende hardware aanpassingen gedaan op het FRDM-KL25Z board. Binnen de gestelde duur van het project was er echter onvoldoende tijd om die aanpassingen te integreren in het project. In overleg met de opdrachtgever is daarom besloten de aanpassingen te documenteren (zie Bijlage C – Metingen stroomverbruik). Bij toekomstige ontwikkeling van digitale klokken valt het aan te bevelen om rekening te houden met deze bevindingen. Met name het uitschakelen van de SDA interface en de manier waarop de core in deep sleep mode wordt gezet, zijn daarbij van belang.

De ontworpen, gerealiseerd en geteste klok die tijdens het project Kairos tot stand is gekomen kan een groot succes worden genoemd. Mede dankzij dit project is het bedrijf *ESE clockworks* een stap dichterbij de realisatie van een nieuwe generatie digitale klokken.

8 Verwijzingen

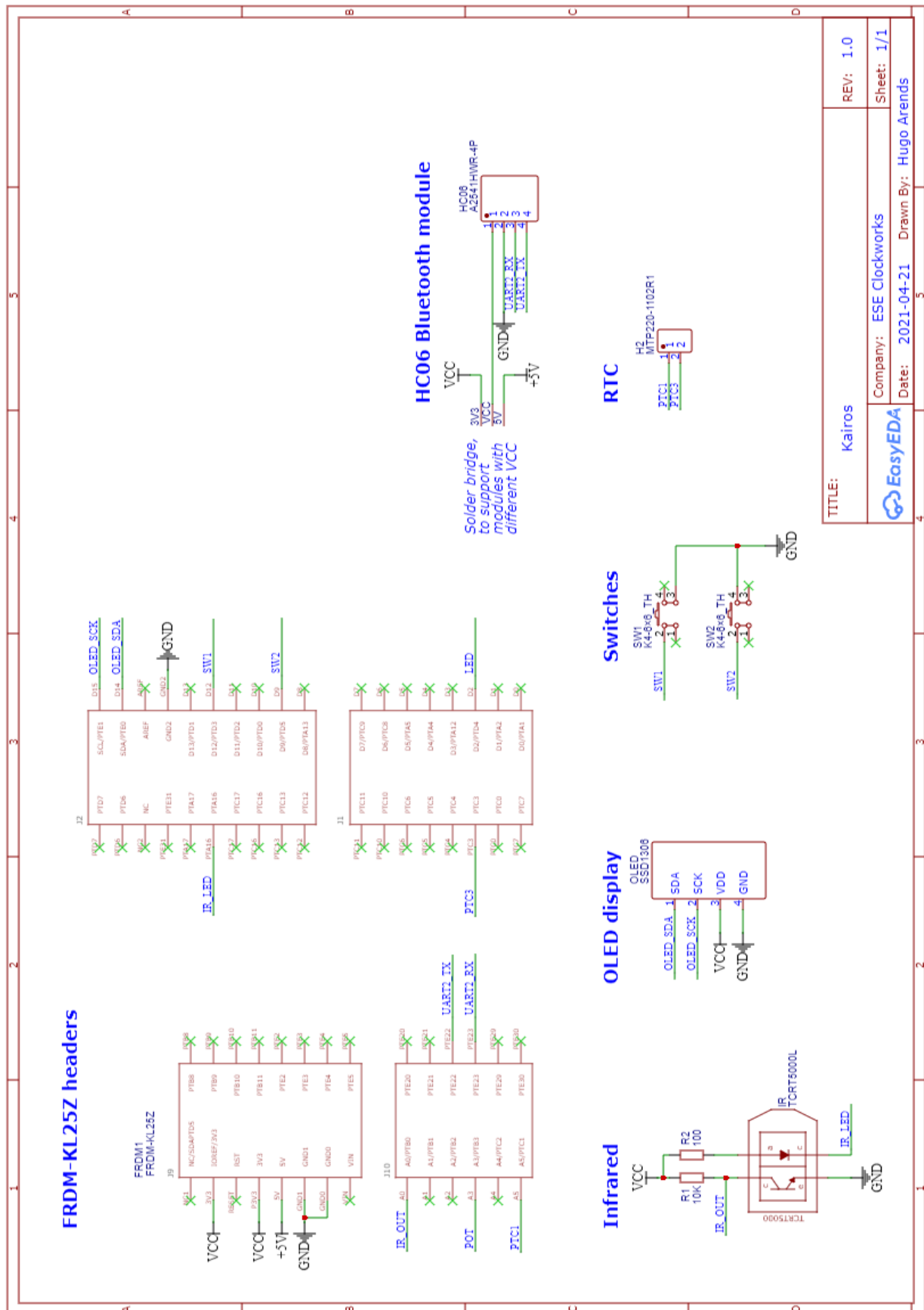
- [1] M. v. B. B. - v. Berckel, Schrijven voor technici, Noordhoff Uitgevers B.V., 2017.
- [2] Hmneverl, „De beslismatrix, het maken van keuzes,” Info.NU.nl, 18 11 2015. [Online]. Available: <https://mens-en-samenleving.infonu.nl/diversen/164525-de-beslismatrix-het-maken-van-keuzes.html>. [Geopend 06 07 2022].
- [3] Freescale Semiconductor, Inc., „FRDM-KL25Z User's Manual, Rev. 2.0,” 24 10 2013. [Online]. Available: <https://www.nxp.com/document/guide/get-started-with-the-frdm-kl25z:NGS-FRDM-KL25Z>.
- [4] Vishay Semiconductors, „TCRT5000(L), Reflective Optical Sensor with Transistor Output, Rev. 1.7,” 9 8 2017. [Online].
- [5] ELECFREAKS, 19 04 2022. [Online]. Available: <https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf>.
- [6] W. contributors, „SMART criteria,” Wikipedia, The Free Encyclopedia, 25 05 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=SMART_criteria&oldid=1089766780. [Geopend 07 07 2022].
- [7] W. contributors, „MoSCoW method,” Wikipedia, The Free Encyclopedia, 06 07 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=MoSCoW_method&oldid=1091822315. [Geopend 06 07 2022].
- [8] Freescale Semiconductor, Inc., „Kinetis KL25 Sub-Family, 48 MHz Cortex-M0+ Based Microcontroller with USB, Rev 5,” 08 2014. [Online].
- [9] Freescale Semiconductor, Inc., „KL25 Sub-Family Reference Manual, Rev. 3,” 9 2012. [Online].
- [10] NXP, „OpenSDA Serial and Debug Adapter,” 2022. [Online]. Available: <https://www.nxp.com/design/software/development-software/sensor-toolbox-sensor-development-ecosystem/opensda-serial-and-debug-adapter:OPENSDA?&tid=vanOpenSDA>.
- [11] Adrián Sánchez Cano, „Using RTC module on FRDM-KL25Z,” 5 3 2013. [Online]. Available: <https://community.nxp.com/docs/DOC-94734>.
- [12] Solomon Systech Limited, „SSD1306: Advanced Information,” 4 2008. [Online]. Available: <https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf>.
- [13] ARM, „µVision® IDE,” 26 04 2022. [Online]. Available: <https://www2.keil.com/mdk5/uvision/>.
- [14] ARM Developer, „KAN232 - MDK V5 Lab for Freescale Freedom KL25Z Board,” 26 04 2022. [Online]. Available: <https://developer.arm.com/documentation/kan232/latest/>.

- [15] NXP, „Kinetis® KL2x-72/96 MHz, USB Ultra-Low-Power Microcontrollers (MCUs) based on Arm® Cortex®-M0+ Core,” 19 04 2022. [Online]. Available: https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/kl-series-cortex-m0-plus/kinetis-kl2x-72-96-mhz-usb-ultra-low-power-microcontrollers-mcus-based-on-arm-cortex-m0-plus-core:KL2x?tab=Buy_Parametric_Tab#/.

Bijlage A

Bijlage B

Bijlage n



Bijlage B – Testscenario's

Deze bijlage beschrijft in detail de testscenario's ten behoeve van het uitvoeren van de acceptatietesten.

Testscenario 1: Weergeven van de tijd

Benodigheden:

- Klok
 - Hardware versie 1.0
 - Software versie 1.1
- Micro USB kabel

Geteste functionele eisen:

- F1

Test uitgevoerd:

- 07/07/2022 door H. Arends.

Stappen:

01	Sluit de klok aan op een spanningsbron met de micro USB kabel.	
	De volgende tijd wordt weergegeven: 00:00:00 01/01/1970 De tijd wordt groot weergegeven en de datum klein.	
	<i>Waargenomen zoals beschreven.</i>	✓
02	Druk eenmaal op SW1 en laat SW1 los.	
	De tijd wordt klein weergegeven en de datum groot. De tijd telt door.	
	<i>Waargenomen zoals beschreven.</i>	✓
03	Druk eenmaal op SW1 en laat SW1 los.	
	De tijd wordt analoog weergegeven op de linkerhelft van het OLED display. De datum wordt op de rechterhelft weergegeven, waarbij de maand voluit geschreven is, dus 'January'. De tijd telt door.	
	<i>Waargenomen zoals beschreven.</i>	✓
04	Druk eenmaal op SW1 en laat SW1 los.	
	De tijd wordt groot weergegeven en de datum klein. De tijd telt iedere seconde door.	
	<i>Waargenomen zoals beschreven.</i>	✓

Conclusie: ✓

Test geslaagd.

Testscenario 2: Datum en tijd aanpassen

Benodigheden:

- Klok
 - Hardware versie 1.0

- Software versie 1.1
- Micro USB kabel

Geteste functionele eisen:

- F2

Test uitgevoerd:

- 07/07/2022 door H. Arends.

Stappen:

01	Sluit de klok aan op een spanningsbron met de micro USB kabel.	
	De volgende tijd wordt weergegeven: 00:00:00 01/01/1970 De tijd wordt groot weergegeven en de datum klein.	
	<i>Waargenomen zoals beschreven.</i>	✓
02	Druk eenmaal op SW2 en laat SW2 los.	
	De tekst setup verschijnt in beeld. Instelbaar: dag Waarde: 01	
	<i>Waargenomen zoals beschreven.</i>	✓
03	Druk tien maal op SW1.	
	Instelbaar: dag Waarde: 11	
	<i>Waargenomen zoals beschreven.</i>	✓
04	Druk eenmaal op SW2 en laat SW2 los.	
	Instelbaar: maand Waarde: 01	
	<i>Waargenomen zoals beschreven.</i>	✓
05	Druk twintig maal op SW1.	
	Instelbaar: maand Waarde: 09	
	<i>Waargenomen zoals beschreven.</i>	✓
06	Druk eenmaal op SW2 en laat SW2 los.	
	Instelbaar: jaar Waarde: 1970	
	<i>Waargenomen zoals beschreven.</i>	✓
07	Druk eenmaal op SW2 en laat SW2 los.	
	Instelbaar: uren Waarde: 00	
	<i>Waargenomen zoals beschreven.</i>	✓
08	Druk drie maal op SW1.	
	Instelbaar: uren Waarde: 03	
	<i>Waargenomen zoals beschreven.</i>	✓
09	Druk eenmaal op SW2 en laat SW2 los.	

	Instelbaar: minuten Waarde: 00	
	<i>Waargenomen zoals beschreven.</i>	✓
10	Druk eenmaal op SW2 en laat SW2 los.	
	Instelbaar: seconden Waarde: afhankelijk van hoe snel er op SW2 is gedrukt aan het begin van de test	
	<i>Weergegeven aantal seconden is 04.</i>	✓
11	Druk eenmaal op SW2 en laat SW2 los.	
	De volgende tijd wordt weergegeven: 03:00:?? 11/09/1970 De tijd wordt groot weergegeven en de datum klein. De tijd telt iedere seconde door	
	<i>Weergegeven tijd is 03:00:04 11/09/1970</i>	✓

Conclusie: ✓

Test geslaagd.

Testscenario 3: Detecteren van een hand voor de klok

Benodigheden:

- Klok
 - Hardware versie 1.0
 - Software versie 1.1
- Micro USB kabel
- Liniaal

Geteste functionele eisen:

- F3

Test uitgevoerd:

- 07/07/2022 door H. Arends.

Stappen:

01	Sluit de klok aan op een spanningsbron met de micro USB kabel. Zorg ervoor dat er zich geen object binnen 10cm van de klok bevindt.	
	De volgende tijd wordt weergegeven: 00:00:00 01/01/1970 De tijd wordt groot weergegeven en de datum klein.	
	<i>Waargenomen zoals beschreven.</i>	✓
02	Houd je hand 15cm voor de klok.	
	Er is geen verandering waarneembaar. De tijd wordt nog steeds groot weergegeven en de datum klein.	
	<i>Waargenomen zoals beschreven.</i>	✓
03	Houd je hand 8cm voor de klok.	
	De klok toont de naam van het bedrijf.	
	<i>De tijd wordt nog steeds groot weergegeven en de datum klein. Pas vanaf 7,5cm wordt de hand juist gedetecteerd.</i>	🕒

04	Houd je hand 5cm voor de klok.	
	De klok toont de naam van het bedrijf.	
	<i>Waargenomen zoals beschreven.</i>	✓
05	Houd je hand 15cm voor de klok.	
	De klok toont nog 5 seconden de naam van het bedrijf. Na het verstrijken van die 5 seconden wordt de tijd groot weergegeven en de datum klein. De tijd tevens correct weergegeven, oftewel de verstreken tijd sinds stap 3.	
	<i>Waargenomen zoals beschreven.</i>	✓

Conclusie: 🟡

Hoewel de meeste stappen correct zijn, wordt bij stap drie een hand gedetecteerd vanaf 7,5cm. De test is dus grotendeels geslaagd.

Testscenario 4: TBD

Dit testscenario is (nog) niet beschreven.

Testscenario 5: TBD

Dit testscenario is (nog) niet beschreven.

Bijlage C – Metingen stroomverbruik

Tabel 3 vat de bevindingen samen. De gemeten stroom is de stroom wanneer de blauwe LED uit is. De gemeten stroom is cumulatief, wat betekent dat alle eerdere aanpassingen intact zijn gelaten. Door alle aanpassingen te maken, werkt het FRDM-KL25Z-bord op 38 μA .

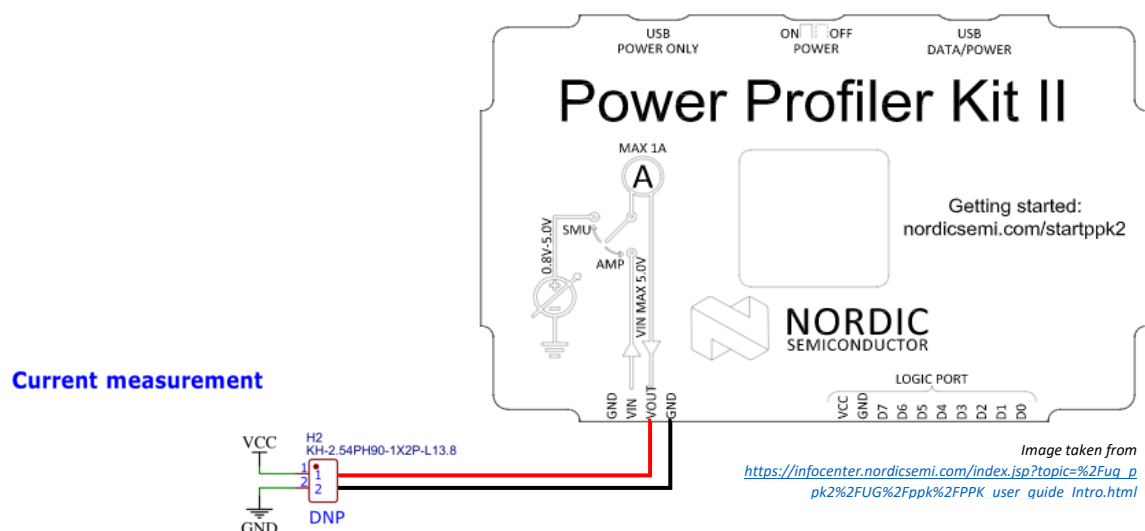
Tabel 3. Samenvatting metingen stroomverbruik.

Nr.	Gemeten stroom [μA]	Bespaarde stroom [μA]	Aanpassing aan het FRDM-KL25Z board
1	8290	-	Geen.
2	5080	3210	Doorkrassen trace J20.
3	375	4705	Verwijder R74.
4	375	0	Doorkrassen trace J11.
5	38	337	Doorkrassen trace J14.

Meetopstelling

Figuur 20 toont de meetopstelling. De [PPK2](#) is de enige spanningsbron (behalve de RTC-backup batterij). De PPK2 wordt gebruikt in Source-metermodus met de voedingsspanning ingesteld op 3000 mV.

VCC van header H2 is verbonden met P3V3 van het FRDM-KL25Z-ontwikkelbord.



Figuur 20. Meetopstelling met de PPK2 in Source meter mode.

De volgende code wordt gebruikt tijdens metingen. Deze code is gemaakt in Keil μVision V5.37.0.0 met optimalisatieniveau ingesteld op -O3 en met behulp van de device startup files die door Keil μVision zijn geleverd. Het project bevat een globale definitie `CLOCK_SETUP=3`, die de systeemklokken als volgt instelt:

- Chip externally clocked, ready for Very Low Power Run mode
- Multipurpose Clock Generator (MCG) in BLPE mode
- Reference clock source for MCG module: System oscillator reference clock
- Core clock = 4MHz
- Bus clock = 1MHz

Verder wordt de lowpower timer (LPTMR) gebruikt voor het genereren van een interrupt na een time-out. Deze time-out wordt gedefinieerd in milliseconden met de functie `lptmr_set_timeout_ms()`. De LPTMR-functies worden niet gegeven in onderstaand codevoorbeeld.

```
int main(void)
{
    SIM->SCGC5 |= SIM_SCGC5_PORTD(1);
    PORTD->PCR[1] = PORT_PCR_MUX(1);
    GPIOD->PDDR |= (1<<1);

    // Setup power mode as follows:
    // - RUNM = 10: Very-Low-Power Run mode (VLPR)
    // - STOPM = 010: Very-Low-Power Stop (VLPS)
    SMC->PMCTRL = SMC_PMCTRL_RUNM(0b10) | SMC_PMCTRL_STOPM(0b010);

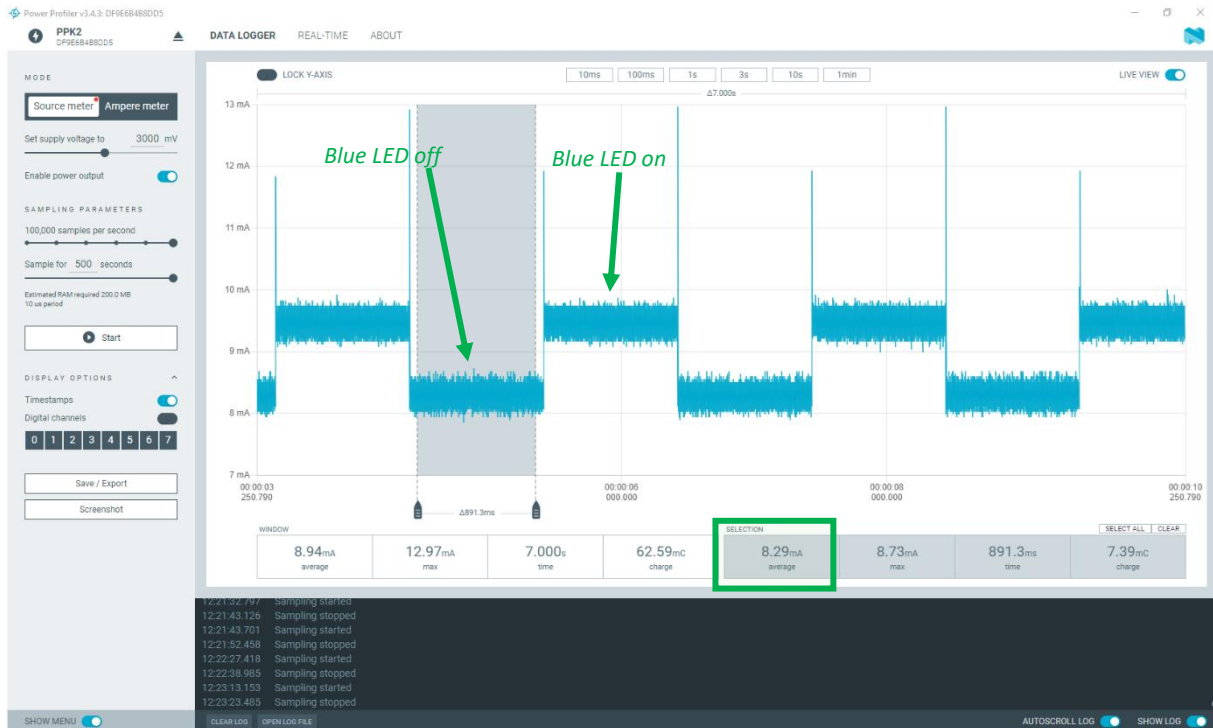
    // Configure the processor core to use deep sleep as its low power
    // mode
    SCB->SCR |= SCB_SCR_SLEEPDEEP_Msk;

    lptmr_init();
    lptmr_set_timeout_ms(1000);

    while(1)
    {
        __asm("wfi");

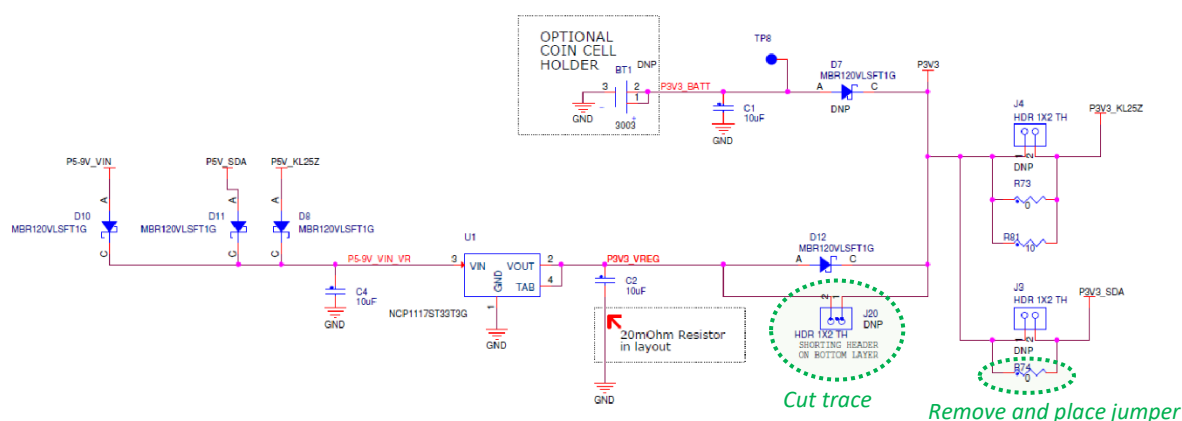
        if(lptmr_timeout_flag == true)
        {
            lptmr_timeout_flag = false;
            lptmr_set_timeout_ms(1000);
            GPIOD->PTOR = (1<<1);
        }
    }
}
```

Geen aanpassingen



Het geselecteerde grijze gebied toont de gemiddelde waarden wanneer de blauwe LED uit is.

Aanpassingen voedingsspanning

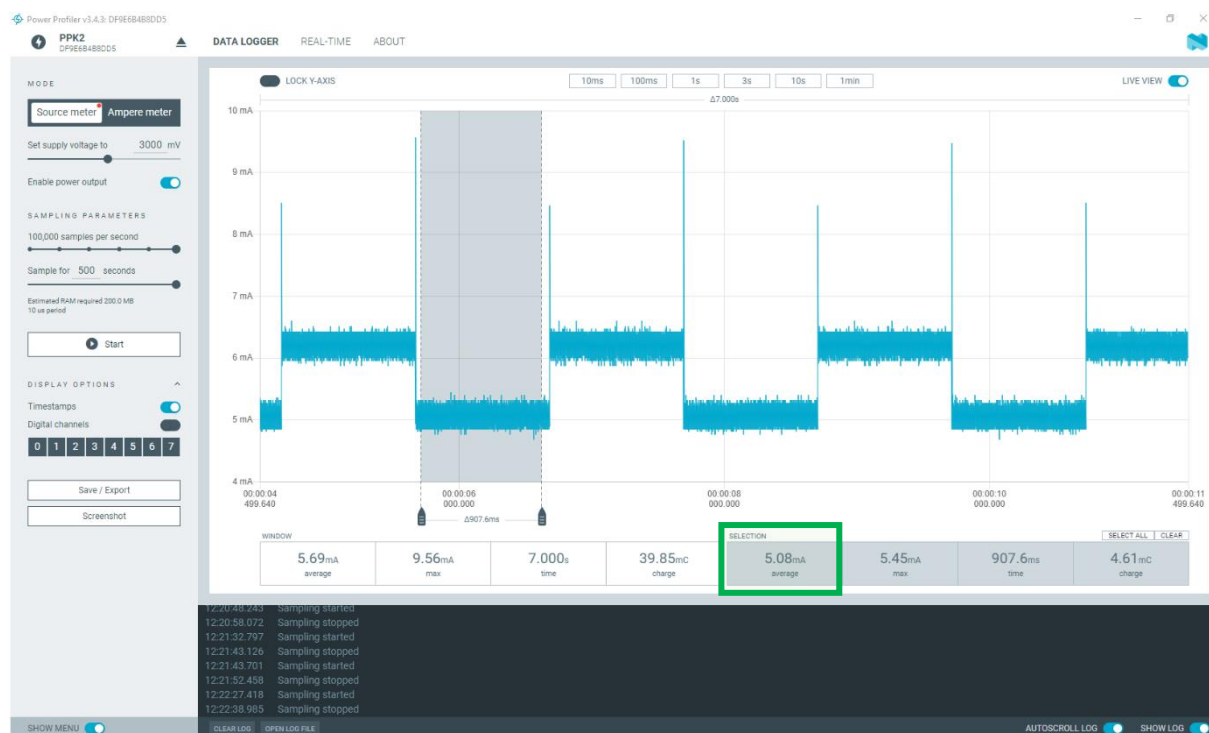


Figuur 21. Aanpassingen voedingsspanning.

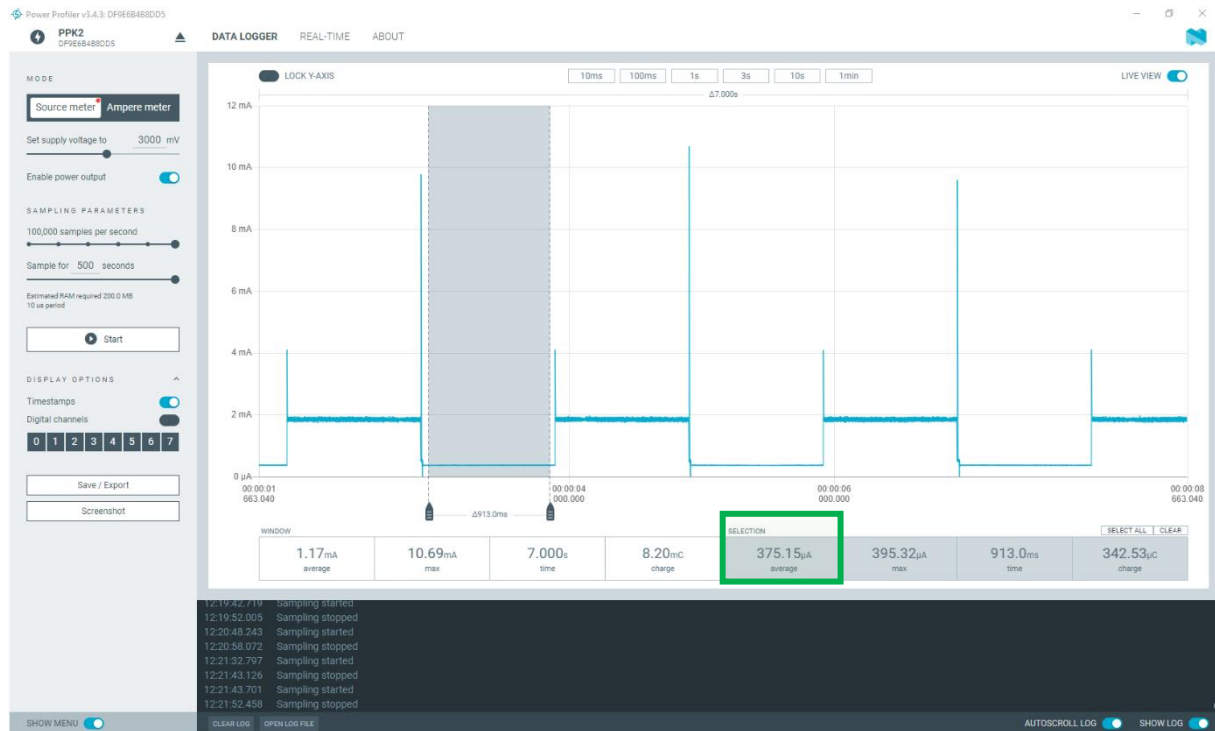
Figuur 21 toont de aanpassingen van de voeding. Het doorkrassen van trace J20 voorkomt dat de stroom van P3V3 terugvloeit naar U1. Optioneel kan een 2x1 header worden gesoldeerd om de voorwaartse spanning van diode D12 te omzeilen bij gebruik van een van de andere voedingsopties.

Het verwijderen van R74 isoleert de SDA van de P3V3-voeding. Op J3 werd een 2x1 header gesoldeerd om de SDA-interface in te schakelen bij het programmeren van de microcontroller.

Doorkrassen trace J20:

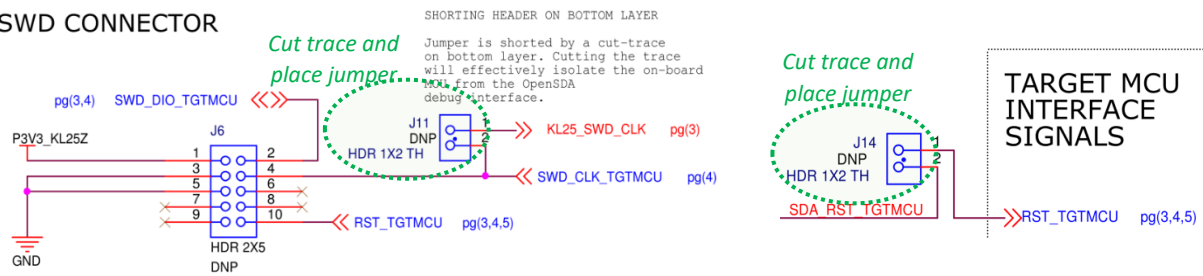


Verwijderen R74 (aanvullend):



SWD aanpassingen

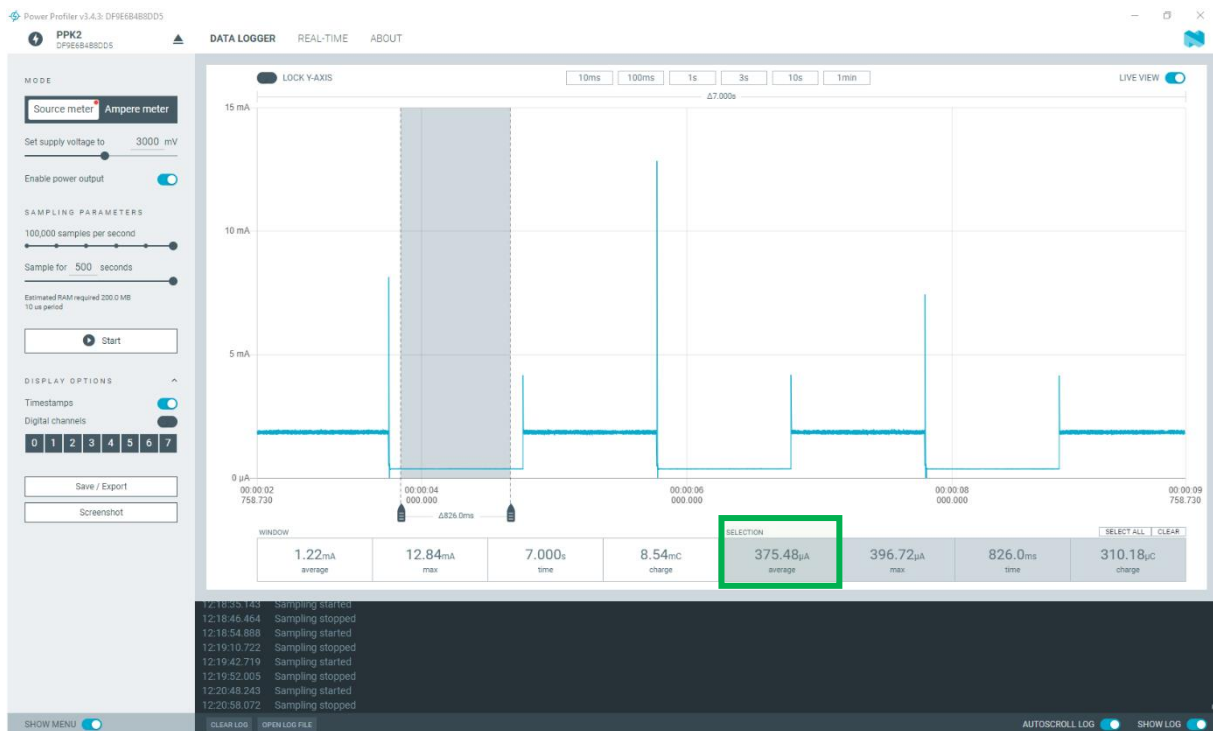
SWD CONNECTOR



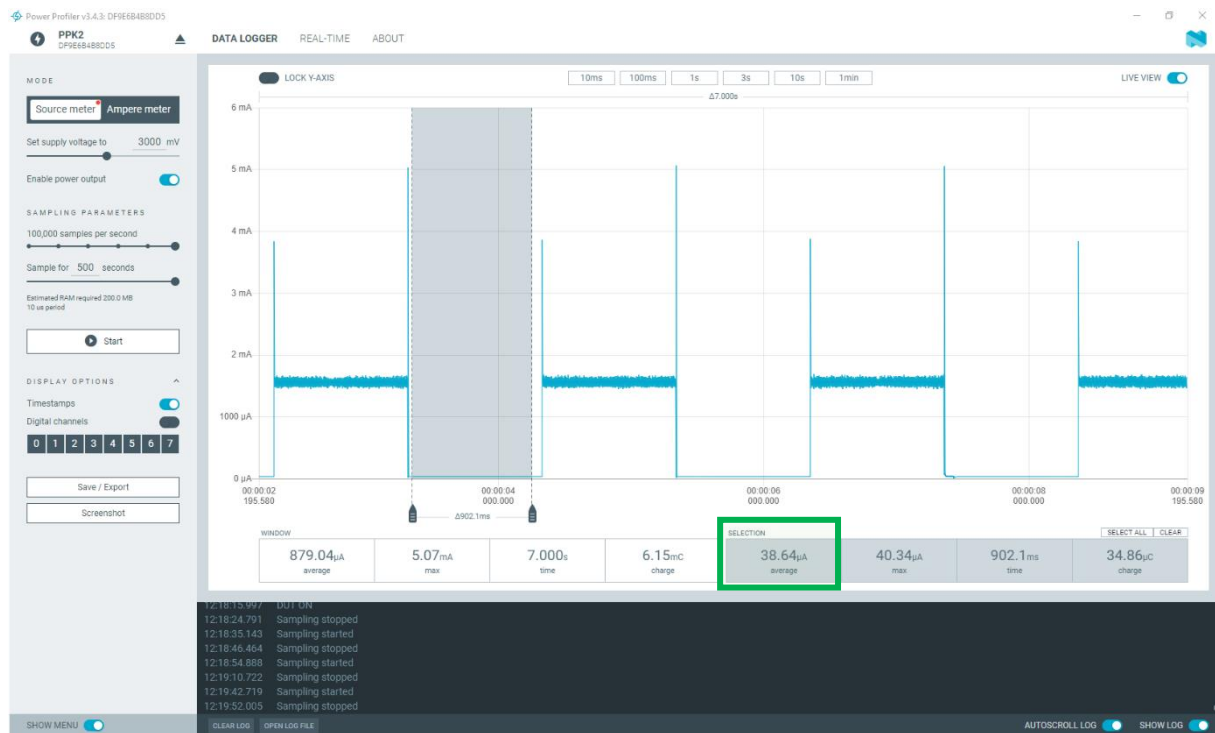
Figuur 22. SWD aanpassingen.

Figuur 22 toont de aanpassingen van de SWD-interface. Header J11 isoleert het SWD CLK-sigitaal. Dit bespaart echter slechts ongeveer $1\ \mu\text{A}$. Header J14 isoleert het SWD RST-sigitaal van de SDA-interface. Beide headers moeten worden kortgesloten om de microcontroller te kunnen programmeren.

Doorkrassen trace J11 (aanvullend):



Doorkrassen trace J14 (aanvullend):



Notities

- De FRDM-KL25Z-kit bevat een MMA8451Q inertial sensor. Na een power-on-reset (POR) bevindt deze sensor zich in de stand-bymodus, waarbij 1.8 μ A wordt gebruikt.
- Om de FRDM-KL25Z van stroom te voorzien vanaf de optionele knoopcelhouder (BT1), moet ook diode D7 worden gevuld. Twee geschikte onderdelen zijn:
 - Coin Cell Battery Holders TH COIN CELL BATTERY HOLDER 20mm
Mouser bestelnummer [534-3003](#)
 - Schottky Rectifier, 20 V, 1 A, Single, SOD-123FL, 2 Pins, 350 mV
Farnell bestelnummer [2918857](#)

Een CR2032-batterij heeft een capaciteit van 225mAh. In rust, dus als de blauwe LED uit is, gebruikt de FRDM-KL25Z ongeveer 38 μ A. Dit betekent dat het systeem in theorie 225mAh / 38 μ A = 5921 uur meegaat. Dit is gelijk aan 246 dagen.

Bijlage n