**Verslag/handleiding**

**Schooljaar: 2019 - 2020**

**Vak: Programmeertechnieken**

**Groepsleden: Lander Buysse, Laure Buysse**

**Inhoudstafel**

[Inleiding 3](#_Toc21287016)

[Gebruikersinterface 4](#_Toc21287017)

[Ontwerp 5](#_Toc21287018)

[Standaardtoestand 5](#_Toc21287019)

[TIIN (tijd instellen toestand) 5](#_Toc21287020)

[ALIN (alarm instellen toestand) 6](#_Toc21287021)

[MZIN 6](#_Toc21287022)

[Alarm 6](#_Toc21287023)

[Three Tier Architecture 7](#_Toc21287024)

[Planning 9](#_Toc21287025)

[Software 10](#_Toc21287026)

[Driver layer 10](#_Toc21287027)

[Middleware layer 10](#_Toc21287028)

[Top layer 10](#_Toc21287029)

[Besluit 11](#_Toc21287030)

# Inleiding

Het doel van dit project is het maken van een instelbare wekker. De wekker functioneert standaard als een klok. Een alarm kan ingesteld worden. Dit alarm kan ook aangezet of afgezet worden. Wanneer dit alarm afgaat, brandt een ledje en hoort men een muziekje. De gebruiker heeft de mogelijkheid om zijn muziekje zelf te kiezen uit enkele voorgeprogrammeerde keuzes. De mogelijkheid bestaat om van zomer- naar winteruur te switchen en omgekeerd met een druk op de knop.

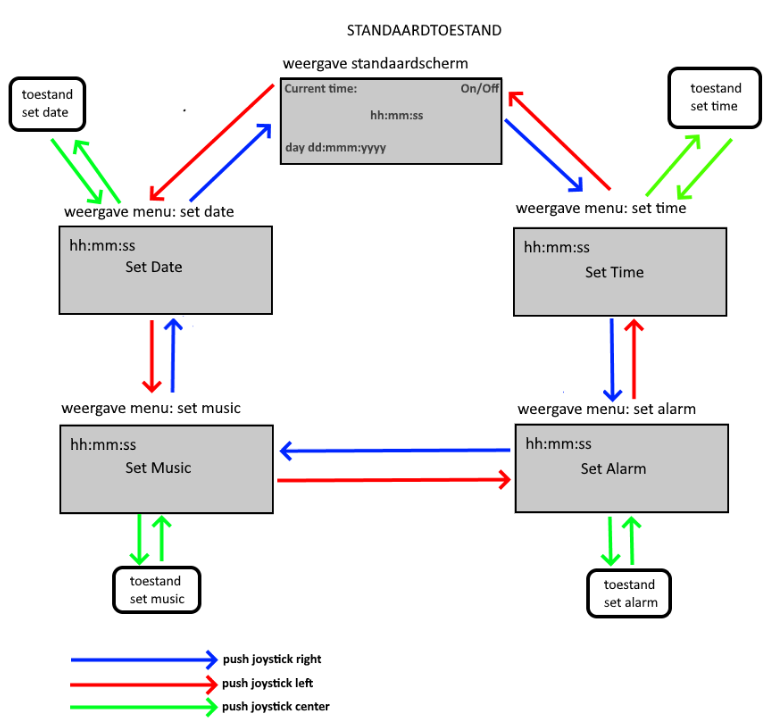
De programmatie van de wekker gebeurt in C met behulp van de MCUXpresso IDE. Het programma is opgebouwd volgens de *three tier architecture*.

//uitleg three tier

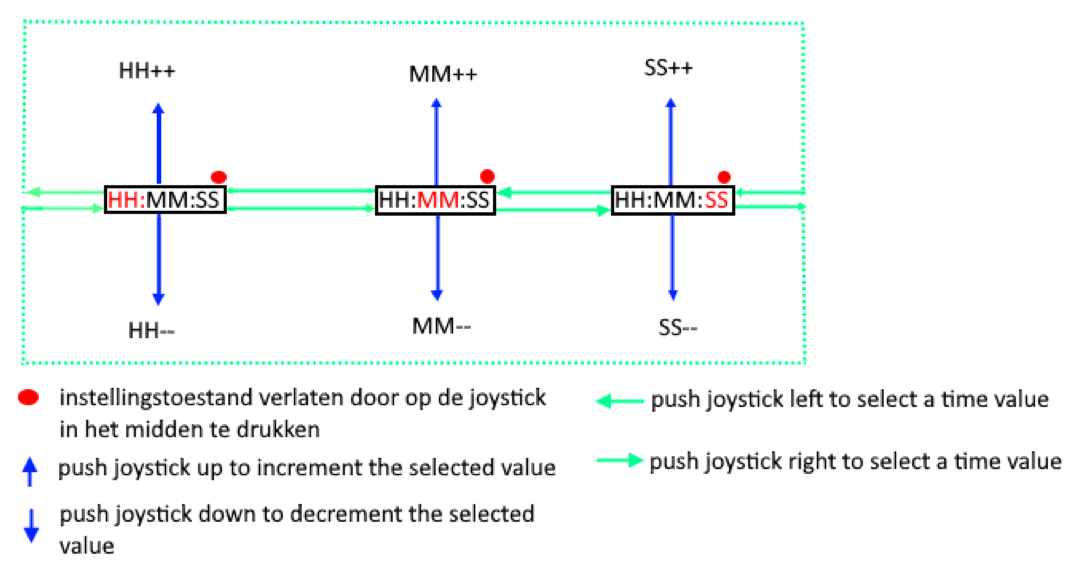
Te gebruiken middelen

* Mbed FRDM-K64F
* Applicaton shield voor Mbed FRDM-K64F
* MCU Expresso

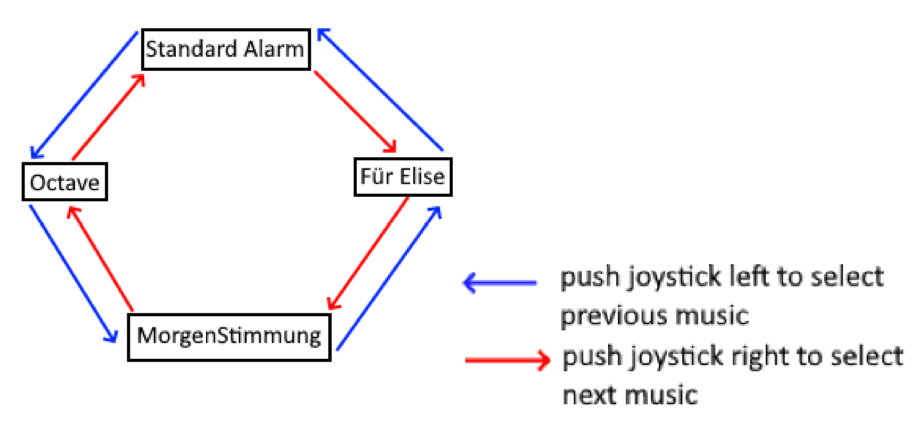
# Gebruikersinterface



Instellen tijd bij set time en set alarm:

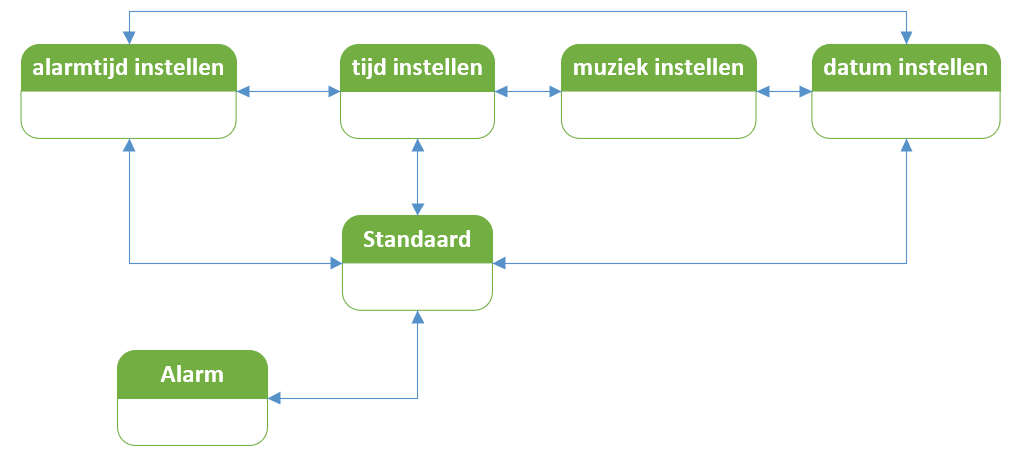


Schema instellen set music:



# Ontwerp

//uitleg over het statemachine en dat er één gekozen is



De wekker kan zich in vijf verschillende toestanden bevinden. Hieronder wordt elk van deze toestanden kort toegelicht. Bij het toelichten van de verschillende toestanden wordt gebruik gemaakt van volgende afkortingen:

1. TIIN= toestand waarbij de tijd wordt ingesteld
2. ALIN= toestand waarbij het alarm wordt ingesteld
3. MZIN= toestand waarbij het muziekje wordt geselecteerd

## Standaardtoestand

In de standaardtoestand wordt de huidige tijd weergeven op het lcd-schermpje. Met behulp van de joystick kan men een menu doorlopen. Op het LCD-scherm kan dan het volgende verschijnen: ‘muziek instellingen’, ‘tijd instellingen’, ‘alarm instellingen’ of de tijd. Door de joystick in te drukken wordt de keuze bevestigd en gaat men naar één van de andere toestanden (TIIN, MZIN of ALIN). Indien men zich op de tijd bevindt, zal het indrukken van de joystick niets doen aangezien men zich al in de standaardtoestand bevindt.

Indien het alarm aanstaat, kan ook naar de alarmtoestand worden overgegaan. Dit gebeurt wanneer de huidige tijd groter of gelijk is aan de tijd waarbij het alarm moet overgaan. Het alarm kan aangezet worden met behulp van een drukknop.

De tijd kan in deze toestand ook met behulp van een knop gemakkelijk van zomertijd naar wintertijd worden omgezet en omgekeerd. Een LED geeft aan in welke tijd (zomertijd of wintertijd) de klok zich bevindt.

## TIIN (tijd instellen toestand)

In de TIIN-toestand wordt op het LCD-scherm de tijd weergegeven volgens volgend format: HH:MM:SS. Deze kan daarop dan veranderd worden met behulp van een joystick. Wanneer de gewenste tijd is ingesteld, wordt de joystick opnieuw ingedrukt om de tijd te bevestigen en terug over te gaan naar de standaardtoestand.

## ALIN (alarm instellen toestand)

In de ALIN-toestand Op het LCD-scherm wordt de huidig ingestelde alarmtijd weergegeven. Met behulp van de joystick kan deze dan worden veranderd. De alarmtijd wordt weergegeven in hetzelfde format als de standaardtijd, namelijk : HH:MM:SS. Wanneer de gewenste alarmtijd is ingesteld, moet de joystick opnieuw ingedrukt worden om naar de standaardtoestand terug te keren.

## MZIN

In de MZIN-toestand wordt op het LCD-scherm weergegeven met welk deuntje de wekker staat ingesteld. Met de joystick kan overgegaan worden naar andere deuntjes. Wanneer het gewenste deuntje is gekozen, moet de joystick opnieuw ingedrukt worden om naar de standaardtoestand terug te keren.

## Alarm

In de alarmtoestand, zal het alarm (de wekker) afgaan. Er zal er een ledje knipperen terwijl een geselecteerd muziekje speelt. Het LCD-scherm toont de alarmtijd. Deze toestand wordt bereikt wanneer de alarmtijd gelijk wordt aan de gewone tijd en kan enkel bereikt worden vanuit de standaardtoestand. Wanneer instellingen worden veranderd (muziek, alarm, tijd), kan het alarm niet afgaan. Deze toestand kan enkel verlaten worden wanneer het alarm wordt afgezet door middel van het indrukken van de joystick. Hierop wordt de wekker terug naar de standaardtoestand gezet (waarbij het alarm nu uitstaat).

//figuren menu standaard

//figuur menu in alarm instellingen (tijd instellingen is hetzelfde), eventueel ook muziek instellingen (is wel heel gelijkaardig)

# Three Tier Architecture

# Three Tier Architecture

De software voor dit project maakt gebruikt van de ‘three tier approach’. Dit omdat deze aanpak enkele grote voordelen biedt:

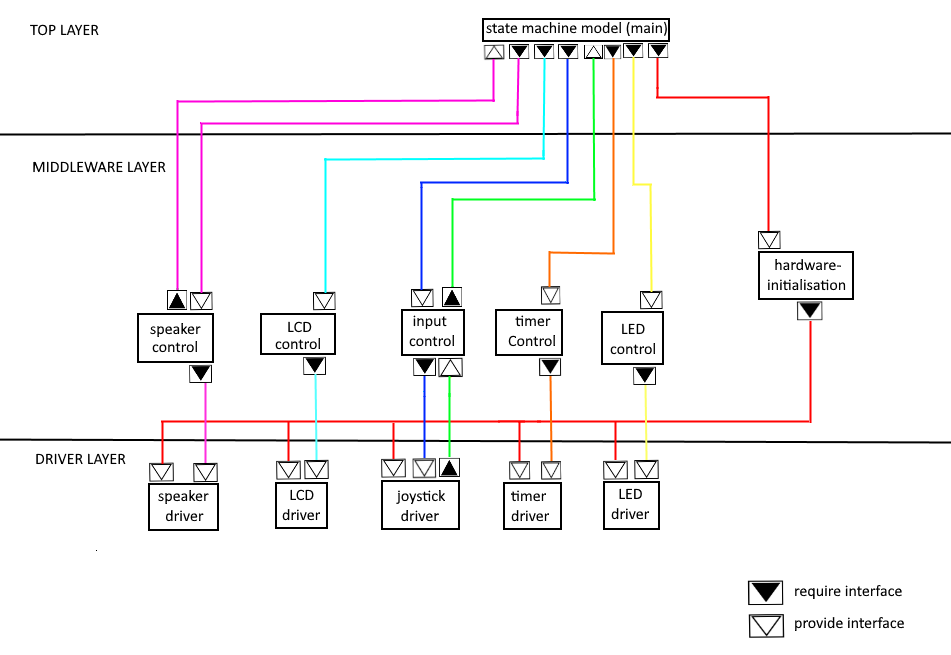
* De code is vrij simpel te onderhouden.
* De code is makkelijk te testen.
* Herbruikbaarheid van de code is zeer groot.

Three tier heeft zijn naam te danken aan de structuur waarin de code wordt opgebouwd. Er wordt namelijk gebruik gemaakt van drie lagen, ook wel tiers genoemd.

De bovenste laag in dit systeem noemt men de top layer. Deze laag is zich niet bewust van de hardware van het systeem of hoe de hardware geïmplementeerd is. Deze laag communiceert met de laag onder zich om zo een gemakkelijk te gebruiken systeem voor de gebruiker te bekomen. In dit project zal de top layer gebruik maken van een state machine model. De toestanden en werking hiervan zijn beschreven in de FRS.

De laag onder de top layer is de middleware layer. De middleware layer is zich, in tegenstelling tot de top layer, wel bewust van de hardware. Hoe de hardware geïmplementeerd is, interesseert deze laag zich echter niet. Afhankelijk van de hardware kan de middelste laag dienen als vertaler tussen de top layer en de driver layer alsook dienen om verdere berekeningen te doen die nodig zijn voor de top layer.

De onderste laag is de driver layer. Deze laag is volledig afhankelijk van de hardware en dus het type microcontroller dat wordt gebruikt. In deze laag bevindt zich de implementatie van de drivers die de hardware nodig heeft. Algemene requests van de middleware layer worden hier vertaald voor de hardware.



Hieronder is te zien hoe de three tier approach in dit project toegepast. In dit driver layer ziet men alle drivers die nodig zijn om de nodige componenten aan te sturen:

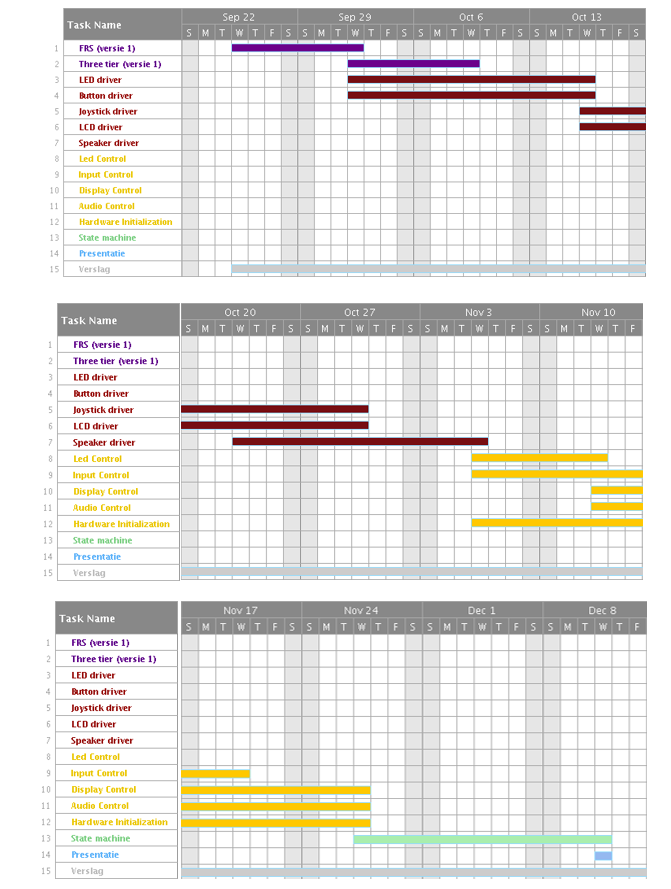
* Speaker driver
* Lcd driver
* Joystick driver
* Button driver
* LED driver

De laag hierboven (middelware layer) zal de specifieke hardware drivers verbinden met de state machine in de top layer. Hier worden ook extra berekeningen gedaan waar nodig.

# Planning

Op de figuur is de planning te zien van het project. Enkele belangrijke deadlines hierbij zijn:

* 02.10.2019 : indienen FRS (voorlopig versie)
* 09.10.2019 : indienen Three Tier Architecture
* 11.12.2019 : presentatie
* 18.12.2019 : indienen verslag



De taakverdeling ziet er als volgt uit:

|  |  |
| --- | --- |
| Taak | Persoon |
| FRS (versie 1) | Lander Buysse en Laure Buysse |
| Three tier (versie 1) | Laure Buysse |
| LED driver | Lander Buysse |
| Button driver | Laure Buysse |
| Joystick driver | Lander Buysse |
| LCD driver | Laure Buysse |
| Speaker driver | Lander Buysse |
| LED control | Lander Buysse |
| Input control | Lander Buysse en Laure Buysse |
| Display control | Laure Buysse |
| Audio control | Lander Buysse |
| Hardware initialisation | Lander Buysse en Laure Buysse |
| State machine | Lander Buysse (en Laure Buysse) |
| Verslag | Lander Buysse en Laure Buysse |

# Software

## Driver layer

//uitleg van alle drivers

## Middleware layer

//uitleg van alle middleware layres

## Top layer

//uitleg over top layer

# Besluit

//wat werd wel niet of niet uitgevoerd tov originele FRS

//wat is bijstesteld en uitgebreid

//obstakels en aanpak hiervan