

Flipperkast en betaalmodule

6e jaar Elektriciteit-elektronica

Schooljaar 2020-2021

Robin Monseré

Begeleiders: Dhr. K. Hertens

Voorwoord

Voor onze geïntegreerde proef kregen we de opdracht om zelf een project te kiezen en dit te realiseren. Ik was dan ook best blij dat we zelf die keuze mochten maken. Ik ben voor een flipperkast gegaan, en er mijn eigen draai aan gegeven. Mijn focus ligt niet enkel op de flipperkast zelf maar ook de betaalmodule die ik er heb bijgemaakt. Deze werkt met RFID-tags. Wil je de flipperkast gebruiken? Zet wat Credits op je RFID-tag en je kan aan de slag! De gedachte hierachter is dat andere arcade games dan ook gespeeld worden met diezelfde badge. Om dit te realiseren heb ik gebruik gemaakt van meerdere Arduinos, solenoïdes, stuurschakelingen en andere elektronica zoals een I2C LCD, RFID-reader, etc.

Graag zou ik enkele mensen bedanken die me dit jaar geholpen hebben met mijn GIP, zonder hun hulp zou deze GIP niet tot stand zijn gekomen. Eerst en vooral wil ik meneer Hertens bedanken, die me doorheen het jaar tips en uitleg gaf. Daarnaast wil ik Gauthier Vanhove bedanken die met zijn 3D printer stukjes heeft gemaakt, Yoni Beheydt die me geholpen heeft met de constructie van de flipperkast en Lukas Oliver die me paint.net heeft leren kennen. Ook de rest van de klas die kwam testen en hun oprechte mening gaf.

Abstractum

Deze GIP was geen simpele opdracht. Ik heb er zeker veel van geleerd, niet enkel over de elektronica en de programmatie maar ook over planning, testen, 3D printen, prototyping, je werk bijhouden en documenteren. Voor mij is het zeker geslaagd als ik kijk naar de hoeveelheid kennis die ik hier uit gehaald heb.

Inhoudsopgave

[1 Inleiding 4](#_Toc65946769)

[2 Betaalmodule 5](#_Toc65946770)

[2.1 Werking 6](#_Toc65946771)

[2.2 Toetsenbord 7](#_Toc65946772)

[2.2.1 Aansluitingen 7](#_Toc65946773)

[2.2.2 Werking 7](#_Toc65946774)

[2.2.3 Programma 9](#_Toc65946775)

[2.3 RFID sensor module 12](#_Toc65946776)

[2.3.1 werking 12](#_Toc65946777)

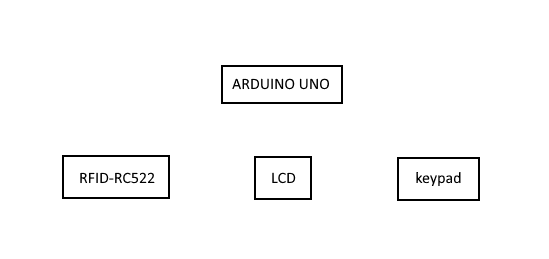
[2.3.2 Aansluitingen 13](#_Toc65946778)

[2.3.3 Programma 16](#_Toc65946779)

# Inleiding

Mijn GIP valt op te delen in 2 stukken, de flipperkast en de betaalmodule. Ik begin bij de betaalmodule, deze is volledig ge-3D-print. Hier wordt een Arduino Uno gebruikt, een LCD, keyboard en RFID-lezer.

Schematische voorstelling:



Figuur 1. Schematische voorstelling betaalmodule.

Daarna is het aan de flipperkast.

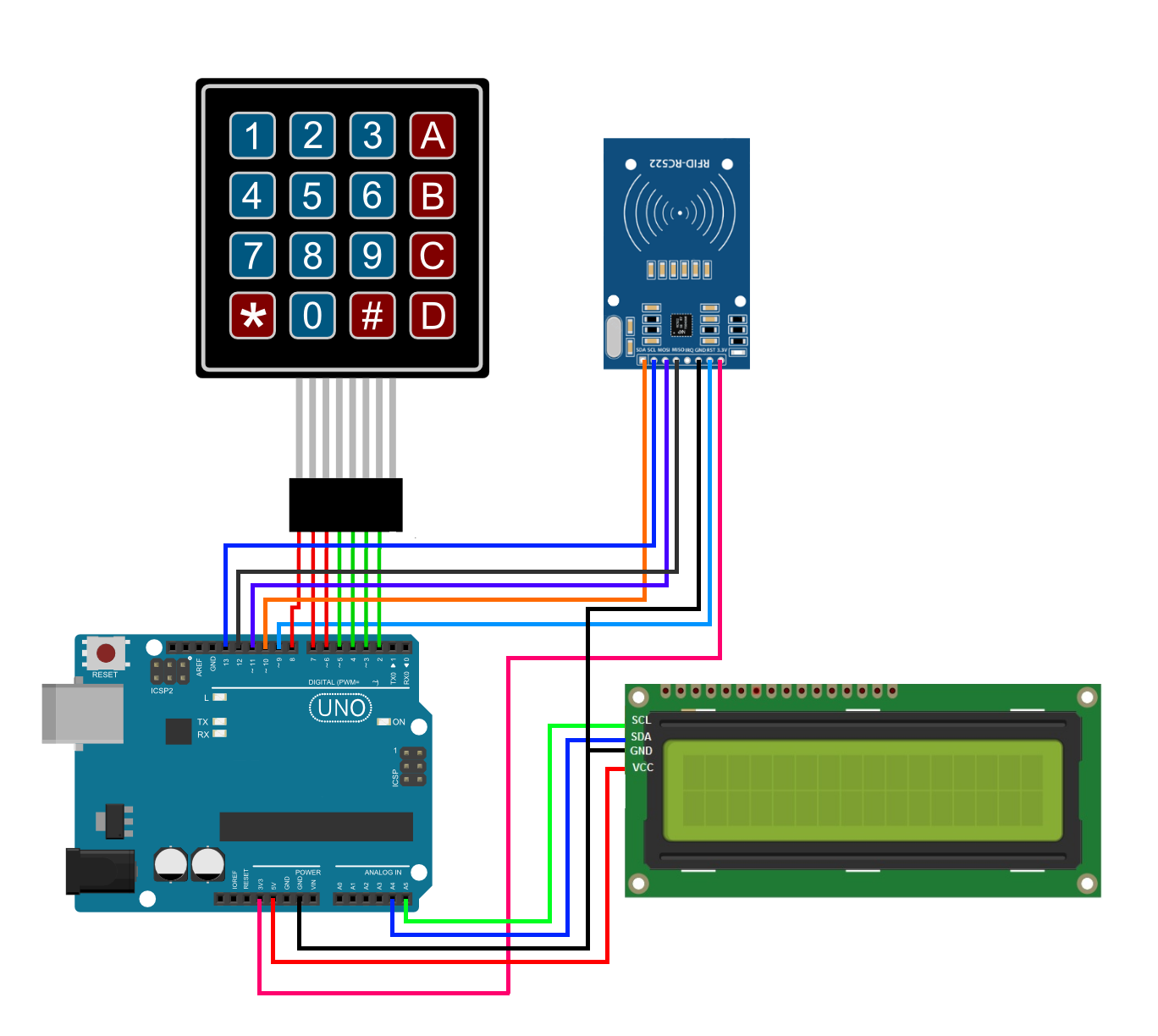
# 125KHz RFID Tag – 5pcsBetaalmodule

Figuur 2. RFID-badge.

De “betaalmodule” wordt gebruikt om “Credits” op een RFID-badge te zetten, dit kan je vergelijken met een bankkaart of de Prizma kaart waarmee je een maaltijd betaalt. Deze Credits heb je nodig om de flipperkast te kunnen gebruiken.

De badge wordt ingelezen via de RFID-RC552, en via het toetsenbord kan je navigeren op de LCD en ingeven hoeveel credits je op je badge wilt zetten.

De betaalmodule bestaat uit een Arduino Uno, een LCD, een membraan toetsenbord en RFID-lezer. De Arduino Uno had net genoeg digitale ingangen om alle componenten aan te sluiten.

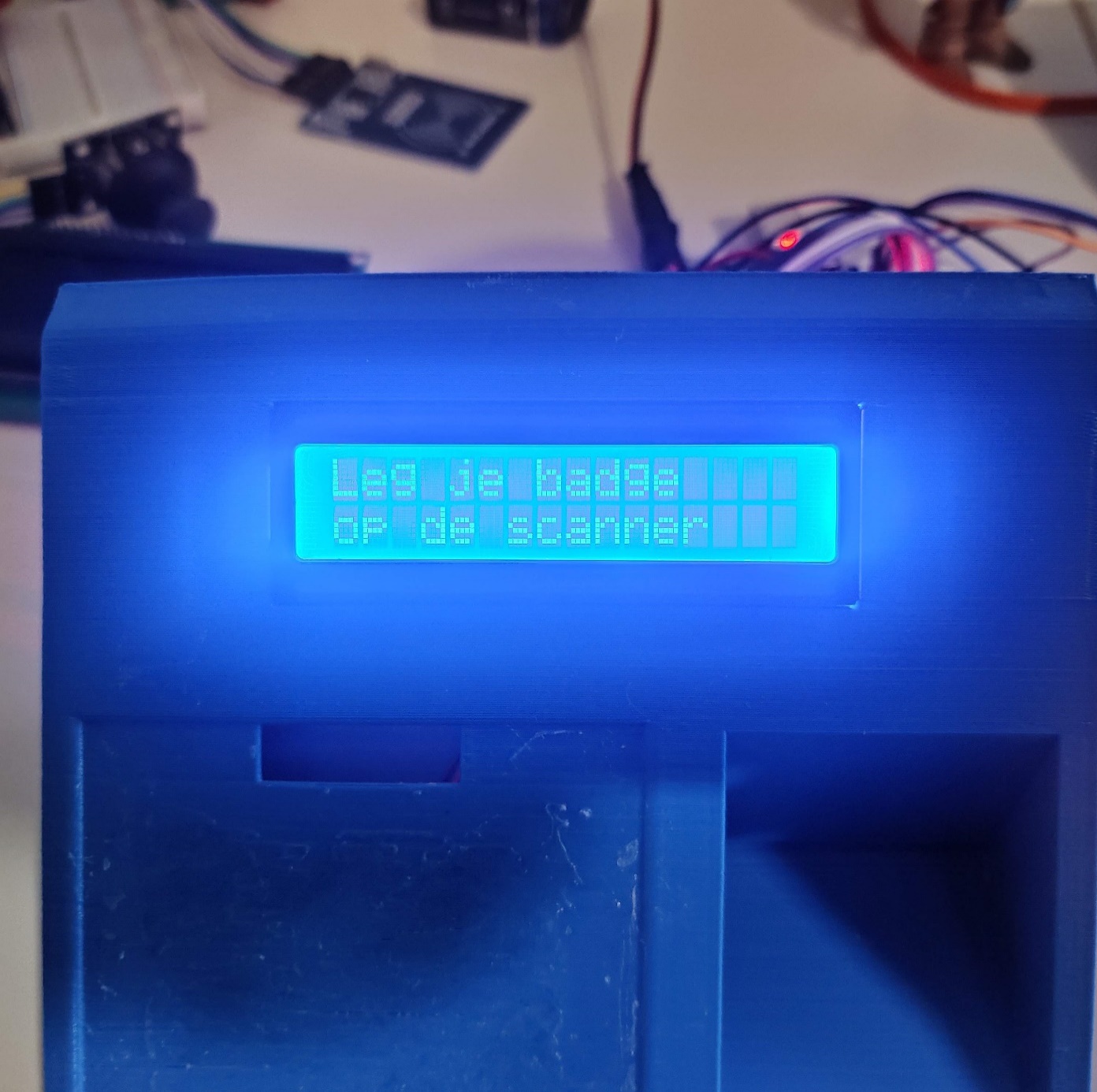


Figuur 3. Verbindingen voor de betaalmodule.

## Werking

De betaalmodule wacht tot er een badge gedetecteerd wordt, dan verschijnt er “leg je badge op de scanner” op de LCD, zoals in figuur 4. Als die een badge detecteert, zal er “welkom [naam badge]” op de LCD komen, dit komt omdat elke badge een specifieke UID heeft (vb: 5D 68 BD 02). Deze UID wordt ingelezen en vergeleken met UID’s die in het programma zitten. Zo weet het programma welke badge er op de scanner ligt. Het is de bedoeling dat de gebruiker de badge laat liggen tot alles klaar is. Daarna kan de gebruiker kiezen uit 2 dingen, Credits storten, of kijken wat het saldo is van de badge (zie figuur 5). Door op \* of # te drukken op het toetsenbord kom je in het corresponderend menu.

Figuur 4. LCD, keuzemenu.



Figuur 5. LCD, passieve toestand.

Druk je op \*, dan krijg je te zien hoeveel credits op de badge staan, door op # te duwen kom je terug in het keuzemenu. Als je kiest om credits te storten word je gevraagd hoeveel er moeten bijkomen. Door in te geven op het toetsenbord, met een maximum van 999, komt de waarde op het scherm. Je wordt gevraagd om te bevestigen door op \* te duwen. Als het programma de credits succesvol heeft gestort, verschijnt dit op het scherm. (figuur 6).



Figuur 6. LCD, storting.

Figuur 7. LCD, storting geslaagd.

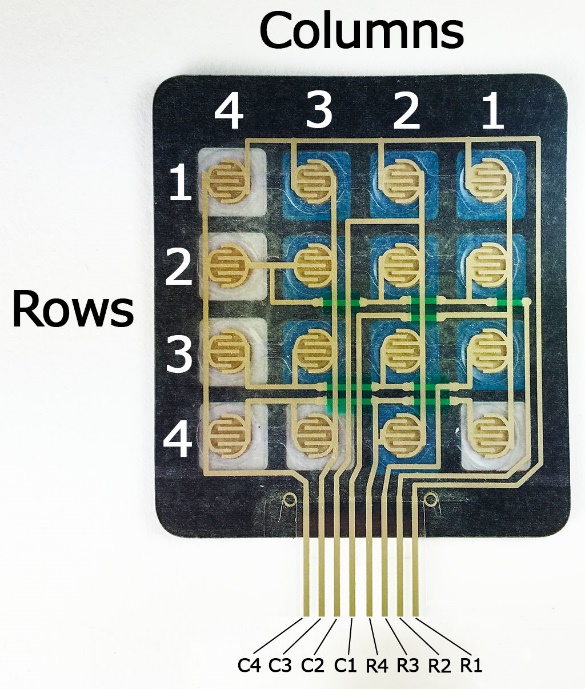
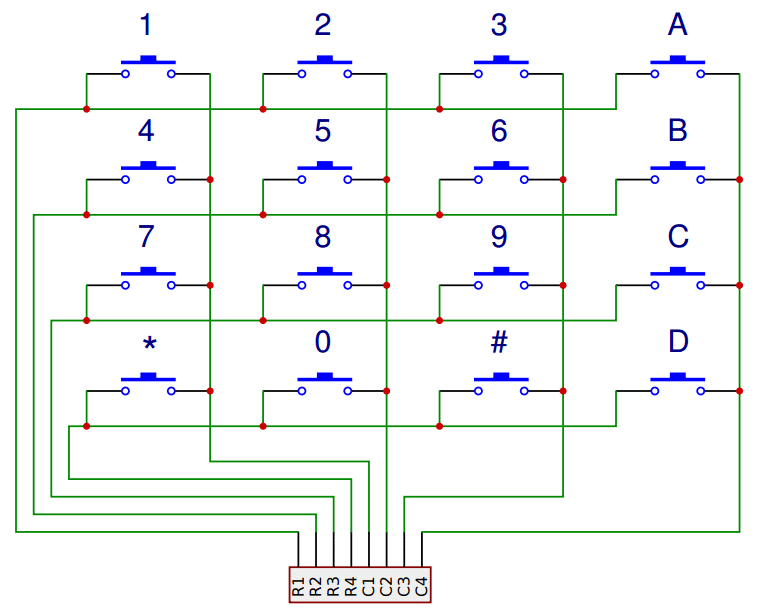
Als dit gedaan is, keert het programma terug naar het keuzemenu. Wanneer er plots een badge van de scanner wordt gehaald, dan zal de LCD terug naar het eerste scherm springen. Dan kan een andere persoon zijn badge op de scanner leggen.

## Toetsenbord

### Aansluitingen

Het aansluiten van dit membraantoetsenbord aan een Arduino is zeer simpel, er zijn 8 pinnen die moeten aangesloten zijn, deze kunnen rechtsreeks aan de digitale ingangen van de Arduino verbonden worden. Op het schema (figuur 3) kan je zien dat er echter maar 7 verbindingen worden gemaak, dit komt omdat de kolom met de knoppen A, B, C en D niet gebruikt worden. En die hoef ik dan ook niet aan te sluiten.

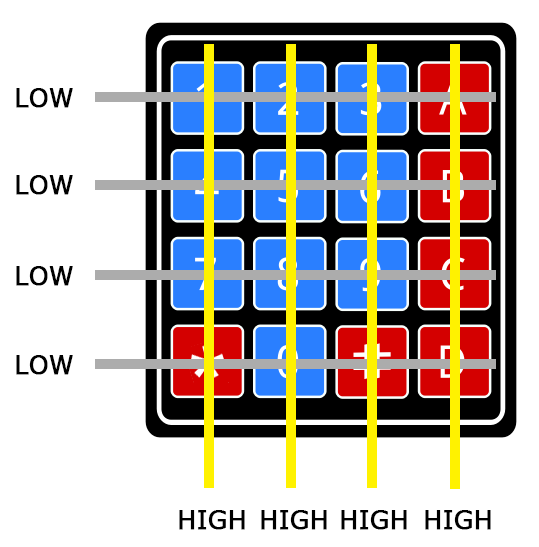
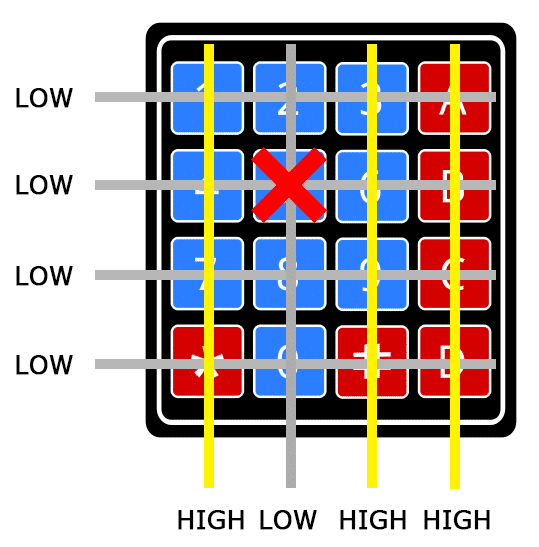
### Werking

De knoppen op het toetsenbord zijn gerangschikt in rijen en kolommen. Onder elke knop zit een membraanschakelaar. Elke schakelaar in een rij is verbonden met de andere schakelaars in die rij, en elke schakelaar in een kolom is ook verbonden met elkaar (zie figuur 8 en figuur 9).

Figuur 8. Schema toetsenbord.

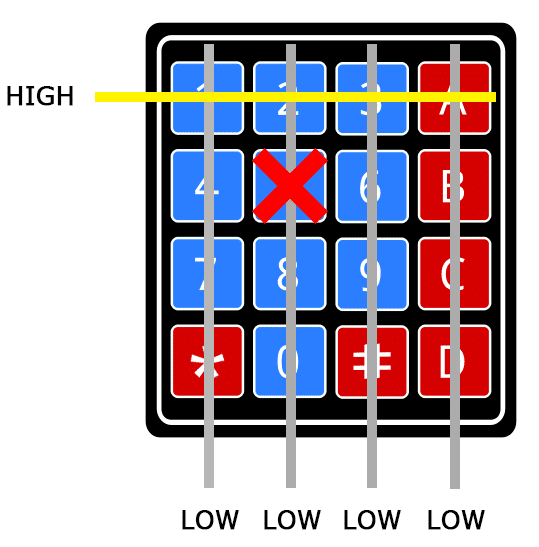
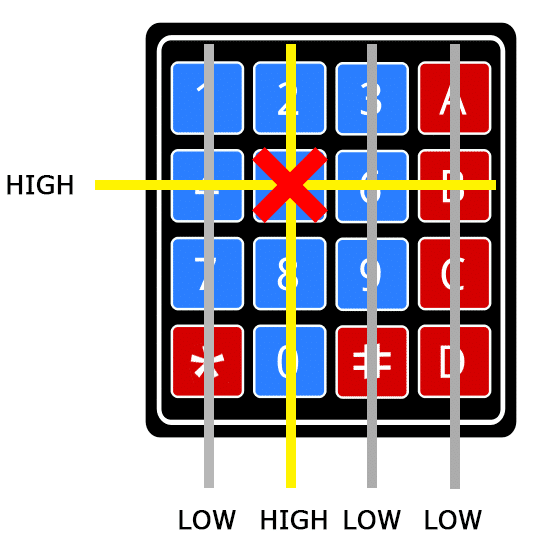
Figuur 9. Binnenkant toetsenbord.

Dit toetsenbord heeft 16 knoppen, en toch maar 8 aansluitingen. Het bepalen van welke knop er ingeduwd wordt, gebeurt in het programma. Dit gebeurt in 4 stappen. Ten eerste is elke kolom pin hoog, en elke rij pin laag. (Zie figuur 10). Wanneer er een knop wordt ingedrukt, wordt de kolom pin daarbij laag getrokken (zie figuur 11). De kolom van de ingedrukte knop is nu bepaald, de Arduino weet nu wel nog niet welke knop het precies is, het moet nu enkel de rij nog bepalen. Dit gebeurt door elke kolom pin nu laag te trekken en 1 voor 1 de rij pinnen kort hoog te maken. (Zie figuur 12) De Arduino leest de kolom pinnen in en wanneer die hoog komt (zie figuur 13) weet die nu ook in welke kolom de knop zit. Met die kennis beslist de Arduino welke knop je net ingedrukt hebt. Welk cijfer of letter waar staat op het toetsenbord, moet je in het programma declareren.



Figuur 10. “5” ingedrukt en kolom pin laag

Figuur 11. Geen knop ingedrukt.



Figuur 12. Rij pin trekt kolom pin hoog.

Figuur 13. Kolom pinnen laag, rij pinnen 1 voor 1 hoog.

### Programma

#### Bibliotheek

Het programmeren van een Arduino toetsenbord is zeer gemakkelijk. Door gebruik te maken van een bibliotheek hoef je zelf niet veel te programmeren maar kan je een functie oproepen die standaard in de bibliotheek zit. Voor vele sensoren en actuatoren kan je een bibliotheek vinden en gebruiken om het programma makkelijker te maken. Alle informatie over de bibliotheek kan je hier vinden: https://playground.arduino.cc/Code/Keypad/

#### Declareren

Je begint in het programma met de nodige variabelen te declareren, ook wordt er een matrix aangemaakt met de karakters van het toetsenbord. Hier worden de karakters A, B, C en D ook niet gedeclareerd, ik heb die toch niet nodig.

|  |
| --- |
| const int ROW\_NUM = 4;  const int COLUMN\_NUM = 3;  char keys[ROW\_NUM][COLUMN\_NUM] =  {   {'1', '2', '3'},   {'4', '5', '6'},   {'7', '8', '9'},   {'\*', '0', '#'}  };  byte pin\_rows[ROW\_NUM] = {8, 7, 6, 5}; // de rijen van het keyboard  byte pin\_column[COLUMN\_NUM] = {4, 3, 2}; // de kolommen van het keyboard  **Keypad** keypad = **Keypad**( makeKeymap(keys), pin\_rows, pin\_column, ROW\_NUM, COLUMN\_NUM ); |

Daarna maak je een “Keypad” object aan, dit gebeurt in rij 36 (figuur 14). Als je meerdere toetsenborden gebruikt voor je project dan maak je meerdere objecten aan en wijs je de correcte pinnen toe aan het juiste object. Voor het oproepen van een functie die in de bibliotheek zit, plaats je altijd het object ervoor. Zo weet de Arduino over welk toetsenbord je het hebt. Bij mijn GIP kan er geen verwarring zijn want ik gebruik maar 1.

#### keypad.getkey()

Om te weten welke toets er op een bepaald moment is ingedrukt, roep je de functie “keypad.getKey()” op, deze geeft een karakter terug. Wanneer er geen knop wordt ingedrukt op het moment, dan keert de “getKey” functie ‘NO\_KEY’terug. Dit karakter steek ik in een variabele die ik “keypressed”noem. Nu gaat het programma door 3‘if’en‘else if’ statements. Deze controleren of de variabele een ‘#’ of een‘\*’is, dan gaat het programma verder in de correcte functies. Als er echter op een andere knop gedrukt is, bv: 1 of 5, dan zal het programma in de laatste if else functie komen. Er komt een tekst op het scherm die de gebruiker laat weten dat je op‘#’of ‘\*’moet drukken. En dan gaat het programma verder.

|  |
| --- |
| char keypressed = keypad.getKey();     if (keypressed == '#')     {       KeuzeMenu = false;       STORT();     }     else if (keypressed == '\*') // \* voor naar balans menu te gaan.     {       KeuzeMenu = false;       BALANS();     }     else if (keypressed != NO\_KEY)     {       lcd.clear();       lcd.setCursor(0, 0);       lcd.print(F("Druk op # of \*"));       delay(1000);       lcd.clear();       lcd.print(F("#: Stort Credits"));       lcd.setCursor(0, 1);       lcd.print(F("\*: Bekijk balans"));     } |

#### keypad.waitForKey()

Soms is het nodig dat het programma moet wachten op een toets die ingedrukt wordt, dit kan gemakkelijk gedaan worden met de ‘waitForKey’functie. Dit stopt het volledige programma en wacht tot er een toets wordt ingedrukt. Als je ook wilt weten welke toets dit is dan ken je die toe aan een variabele. In het voorbeeld zit het programma in een while lus en telkens wanneer er op een toets wordt gedrukt dan wordt deze gecontroleerd, als het een ‘#’ is dan gaat het programma uit de while lus en verder door het programma.

|  |
| --- |
| char KeyPressed = ' ';     while (KeyPressed != '#')  {       KeyPressed = keypad.waitForKey();     }     KeuzeMenu = true;     KEUZEMENU(); |

FOTO

## RFID sensor module

Gebruik maken van een RFID kit is essentieel om te weten wie speelt op de flipperkast. Zo kan er een verdienmodel aangekoppeld worden. Ik heb voor de RC522 gekozen omdat deze zeer bekend is. Het is goedkoop, vaak minder dan 5 euro, het is betrouwbaar en heeft een laag verbruik.

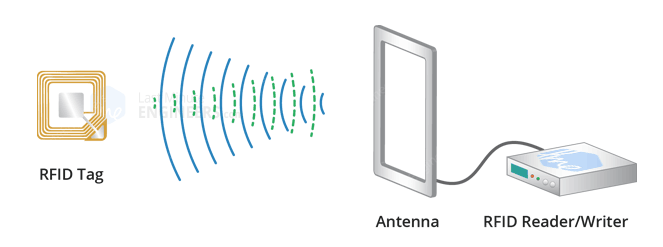
Er zijn 2 belangrijke onderdelen, de RC522, een sensor die een RFID chip kan uitlezen en naar schrijven. En de RFID chip, die zit meestal in een kaard of badge. (Zie figuur 14)

Figuur 14. Een volledige RFID-RC522 kit met kaard en badge.

Deze technologie wordt vaak toegepast in de praktijk. In grootte bedrijven wordt het gebruikt om te weten wanneer arbeiders beginnen aan hun shift of wanner ze naar huis gaan. Ook de Prizma eetkaart werkt met dit. Dit kan ook gebruikt worden om in een winkel automatisch af te rekenen.

RFID staat voor: **R**adio **F**requency **Id**entification.

### werking

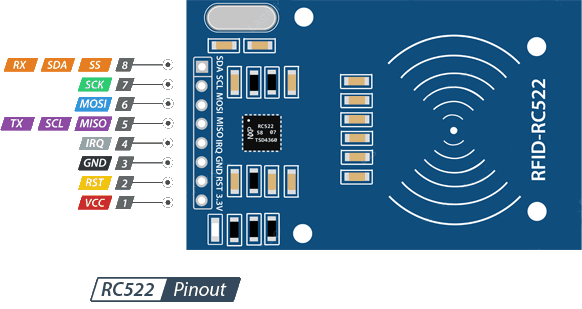
De RC522 module bestaat uit een radiofrequentiemodule en een antenne die een hoogfrequent elektromagnetisch veld genereert. De tag is meestal een passief apparaat, wat betekent dat er geen batterij in zit. In plaats daarvan bevat het een microchip die informatie opslaat en verwerkt, en een antenne om het ​​signaal te ontvangen en te verzenden.

Figuur 15. Principe werking van RFID.

Om de informatie die op een tag opgeslagen is te lezen, wordt deze in de buurt van de RC522 geplaatst (hoeft geen direct contact te hebben). De RC522 genereert een elektromagnetisch veld waardoor elektronen door de antenne van de tag bewegen en vervolgens de chip van stroom voorzien. De aangedreven chip in de tag reageert vervolgens door de opgeslagen informatie terug te sturen naar de lezer in de vorm van een ander radiosignaal. Dit wordt backscatter genoemd. De verandering in het elektromagnetisch veld wordt gedetecteerd en opgevangen door de lezer, die de gegevens vervolgens naar een computer of microcontroller stuurt. In dit geval een Arduino Uno, De communicatie tussen Arduino en de RC522 gebeurt via de SPI bus. Door opnieuw gebruik te maken van een bibliotheek kan dit vlot en simpel gedaan worden.

### Aansluitingen

De RC522 werkt op een spanning van 3.3V dus we beginnen met de VCC pin aan de 3.3V te verbinden en de GND pin aan de GND van de Arduino. De RST pin kan aangesloten worden op elke digitale ingang van de Arduino, hier is die op poort 9 aangesloten. De rest van de pinnen zijn voor de SPI communicatie, deze worden best verbonden met hardware SPI pinnen van de microcontroller. Deze zijn het snelst. Elk type Arduino bord heeft heeft andere pinnummers daarvoor. De Arduino Uno gebruikt pin 11 voor de MOSI (**M**aster **O**ut, **S**lave **I**n), pin 12 voor MISO (**M**aster **I**n, **S**lave **O**ut), pin 13 voor SCK (**S**erial **C**loc**k)** en pin 10 voor de CS (Chip **S**elect).

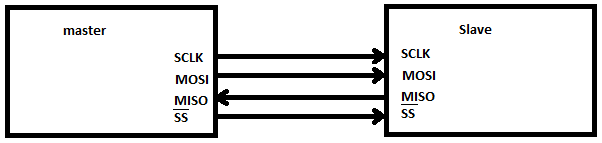


Figuur 16. Aansluitingen RC522

### SPI bus

De RC522 maakt gebruik van de SPI bus, dit is een welbekend en wereldwijd gebruikt communicatie protocol. De start van de communicatie gebeurt door de masterchip. Het is mogelijk om meerdere slaves te hebben, maar ze moeten dan ieder een aparte chipselect hebben. Er zijn altijd vier verbindingen voor communicatie nodig, namelijk:

* MISO: **M**aster **I**n **S**lave **O**ut, de data van de slave wordt hierover naar de master gestuurd.
* MOSI: **M**aster **O**ut **S**lave **I**n, op deze lijn wordt de data verzonden van master naar slave.
* SCLK: **S**eriële **C**loc**k**, wordt geleverd door de master.
* SS: **S**lave **S**elect, Deze lijn wordt actief laag aangestuurd. De lijn voor de geselecteerde slave zal laag zijn. Wanneer de communicatie met de slave gedaan is, zal de lijn een logische 1 krijgen.

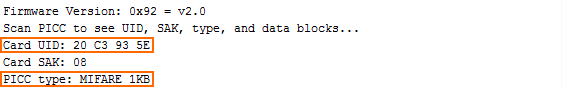


Figuur 17. Schematische voorstelling SPI bus.

### RFID badge

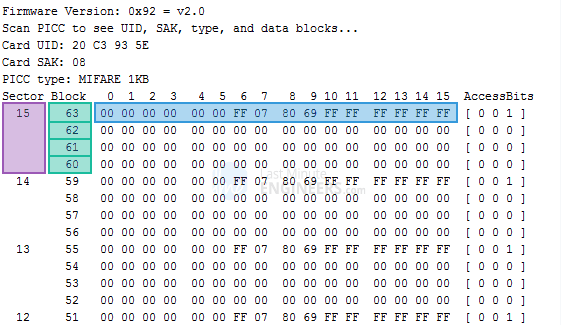
Ik heb beslist om een badge te gebruiken en geen kaart, de badge is wat kleiner en past op de RC522, de kaart heeft meer ruimte nodig.

In de bibliotheek die ik gebruik voor de RC522 zit een voorbeeld programma genaamd “Dumpinfo”. Dit programma is zeer handig, het schrijft niets van informatie naar de badge maar het toont alle nuttige informatie die in de badge opgeslagen is op het scherm van de seriële monitor. Dit is ook het merk, type, UID en hoeveel opslag er is. De badges die ik gebruik hebben 1KB aan data en de UID (**U**nique **Id**entifier) is “20 C3 93 5E”. Elke kaart heeft een andere UID, als deze door toeval toch hetzelfde is, of je wil die voor een andere reden veranderen, dan is dit ook mogelijk met een programma die in de bibliotheek zit.



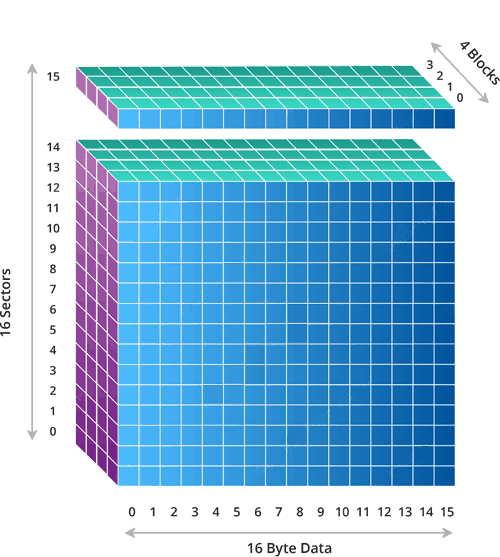
Figuur 18. Gegevens RFID badge.

#### Geheugen

De RFID badge heeft een geheugen van 1K, ook dit kunnen we perfect zien via het voorbeeld programma. Dit geheugen is georganiseerd in 16 sectoren (van 0 tot 15). Elke sector is verder verdeelt in 4 blokken, elke blok kan 16 bytes aan data opslaan.

Figuur 19. Geheugen RFID badge.

Dus als we het even uitrekenen, 16 bytes aan data per blok, en 4 blokken per sector, voor 16 sectoren: 16\*4\*16 = 1024 bytes = 1Kb. Dit geheugen kunnen we gebruiken om naar te schrijven en van te lezen, hier zetten we de hoeveelheid Credits op. Er moet enkel afgesproken worden op welke plaats we die data plaatsen. Je kan kiezen waar je dit doet, ik koos voor block 60.



Figuur 20. Schematische voorstelling 1Kb.

### Programma

De RC522 was niet gemakkelijk om te programmeren,

Bronnen:

<https://www.circuitbasics.com/how-to-set-up-a-keypad-on-an-arduino/>

<https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/>

<https://playground.arduino.cc/Code/Keypad/>