INF2C – groep 3

**Projectleden**

Tedjan Hulshof

Dennis Hartmann

Justin Bakker

Roy Heidotting

Melvin Bos

Tim Wennekes



**Hoe vindt de communicatie tussen de modules plaats?**

Inhoud

[Methode 3](#_Toc346191726)

[Hoe vindt de communicatie tussen de modules plaats? 4](#_Toc346191727)

[1. Checksum 4](#_Toc346191728)

[2. Reguliere expressie 5](#_Toc346191729)

[3. De toepassing 6](#_Toc346191730)

[Literatuurlijst 7](#_Toc346191731)

# Methode

Voor dit onderzoek is gebruik gemaakt van literatuuronderzoek. Verder zijn er praktijktests uitgevoerd naar het gebruik van een checksum.

# Hoe vindt de communicatie tussen de modules plaats?

Om de hoofdunit te voorzien van de vereiste informatie is het van belang dat er een betrouwbare manier van communicatie wordt opgezet tussen de sensorunit en de hoofdunit. De sensorunit zal binnen een bepaalde frequentie de waarde van zijn sensor en de ID van het apparaat moeten doorsturen naar de hoofdunit.

Om ervoor te zorgen dat de hoofdunit constant van juiste en actuele informatie te voorzien is het belangrijk dat er goed onderzocht wordt welke invloeden van buitenaf het signaal kunnen storen. Zo is het mogelijk dat sensoren tegelijk gaan zenden, waarbij de signalen elkaar verstoren.

Wanneer meerdere sensorunits tegelijk gaan zenden zal de ontvanger van de hoofdunit de signalen door elkaar gaan gooien. Dit komt omdat de ontvanger geen onderscheid maakt tussen wie wat stuurt. Het is daarom van belang dat de verstuurde signalen nogmaals worden gecontroleerd wanneer zij op de hoofdunit binnenkomen.

## 1. Checksum

Deze controle kan worden gedaan doormiddel van een checksum. Een checksum is een methode met een controlegetal om de ontvangen input te controleren. Deze methode wordt al toegepast bij bijvoorbeeld ISBN nummers.

Het signaal dat de sensoren uitzenden moet bestaan uit een unieke sleutel, die in het geheugen van de microcontroller staat, en de waarde die de microcontroller leest uit de sensor. Uit eindelijk zal dit er als volgt uitzien:

**A01:17.1**

Om dit te controleren is het mogelijk om een controlegetal mee te sturen met deze waarden. De hoofdunit kan dat controlegetal gebruiken om de ontvangen input te controleren op integriteit. Om het controlegetal te berekenen is er gekozen voor de volgende berekening. Alle getallen achter het gestuurde karakter zijn de bitwaarden in de ASCII-tabel.

“A” = 65

“0” = 48

“1” = 49

“:” = 58

“1” = 49

“7” = 55

“.” = 46

“1” = 49

Tot = 419

Het controlegetal wordt dan bepaald met de som 419 modulo 111, wat resulteert in 86. Het laatste getal wordt samen met een karakter als laatst gestuurd om het einde van het bericht aan te geven. Wanneer het bericht ontvangen is door de hoofdunit zal deze het omgekeerde doen van de sensorunit. Het bericht wordt uit elkaar getrokken en er wordt gecontroleerd of het controlegetal aan het eind van de boodschap nog klopt. Wanneer het bericht corrupt is zal het controlegetal niet meer kloppen.

Deze methode van controleren is vrij eenvoudig toe te passen, maar heeft ook een aantal nadelen. Wanneer de karakters slechts anders staan gerangschikt zal de checksum alsnog kloppen. Dit is te voorkomen door in de checksum bepaalde vermenigvuldigingen uit te voeren.

Dit zal er dan als volgt uit zien:

“A” = 65 \* 1 65

“0” = 48 \* 2 96

“1” = 49 \* 3 147

“:” = 58 \* 4 232

“1” = 49 \* 5 245

“7” = 55 \* 6 330

“.” = 46 \* 7 332

“1” = 49 \* 8 392

Totaal = 1839

Nu is het laatste getal 1839, na modulo 111 is dat 63. Nu moet de volgorde van de karakters behouden worden; afwijkingen in de verhoudingen geven uiteindelijk een verkeerd eindgetal.

## 2. Reguliere expressie

Een tweede manier om het bericht te controleren is te controleren met reguliere expressie. Met reguliere expressie kan het bericht worden gecontroleerd op patronen door bijvoorbeeld de microcontroller in de hoofdunit. Zo kan worden gezorgd dat wanneer berichten corrupt binnen komen, deze niet als relevant gezien worden omdat ze bijvoorbeeld tweemaal “:” bevatten of korter zijn dan verwacht.

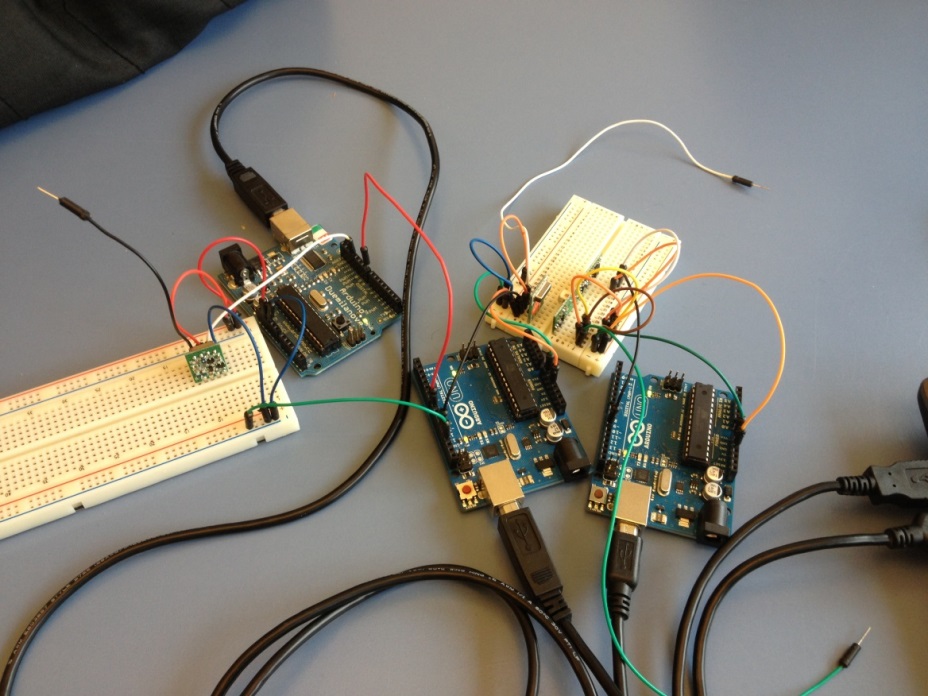
Een simpel voorbeeld van reguliere expressie:

Bericht: A01:17.1  
Controle: [Letter][Nummer][Nummer][:][Nummer][Nummer][.][Nummer]

De controle wordt uitgevoerd op karakterniveau. Hiermee wordt bedoeld dat de controle elk karakter langsgaat om te controleren of deze voldoet aan het voorgestelde patroon. Deze manier kijkt ook of het karakter op de juiste plaats in het bericht staat. Zo is het niet mogelijk om bijvoorbeeld “A011:7.1” te sturen. De controle ziet dan “:” op de verkeerde plek staan.

## 3. De toepassing

Om deze methode toe te passen is er een testopstelling gemaakt. Deze opstelling bestaat uit drie microcontrollers, een ontvangstmodule en twee zendmodules. Het is de bedoeling dat 2 zenders gaan interfereren en dat de ontvanger dit doormiddel van een checksum detecteert.

De zenders sturen 2 afwijkende berichten:

**Zender 1** : A01:17.0  
**Zender 2** : A02:17.1

Omdat de zenders dit bericht snel versturen, worden ze aan elkaar gekoppeld om op precies hetzelfde moment te zenden. Op deze manier worden ze expres geïnterfereerd. Als er word een bericht wordt verzonden stuurt de Arduino een signaal over naar de andere Arduino om ook te zenden.

Echter wanneer dit wordt gedaan ontvangt de zender niks, hier uit kun je opmaken dat er niet tegelijk een bericht verzonden kan worden. Echter mocht het gebeuren dat er 1ms of minder verschil tussen beide berichten zit kunnen de berichten niet worden gelezen.

Het checksum-programma is wel toegepast op met handmatig geschreven data. De (on)geldigheid hiervan werd telkens juist gedetecteerd.

# Literatuurlijst

Wikipedia. (sd). *Checksum*. Opgeroepen op 12 05, 2012, van Wikipedia: http://nl.wikipedia.org/wiki/Checksum

Wikipedia. (sd). *Reguliere Expressie*. Opgeroepen op 12 05, 2012, van Wikipedia: http://nl.wikipedia.org/wiki/Reguliere\_expressie