Arduino support voor Smart LEDs

Sarah Joseph
Departement Computerwetenschappen, Faculteit Wetenschappen
K.U. Leuven
sarah.joseph@student.kuleuven.be

Abstract

De toegankelijkheid voor het communiceren met het zichtbaar licht wordt onderzocht met een nadruk op snelheid en betrouwbaarheid. De paper heeft ten eerste onderzocht welke hardware componenten gebruikt kunnen worden voor Visible Light Communication (VLC). Vervolgens zijn er algoritmen geschreven geweest voor het structureren van de gegevens in pakketten en voor het oversampelen van deze pakketten bij ontvangst. [1] De VLC kanaal wordt gebruikt om gegevens te versturen naar een smart LED op afstand. Hiervoor wordt er bij ontvangst een In-circuit Serial Programming (ICSP) script gebruikt voor het communiceren met de bootloader van de smart LED. Vanuit deze serial interface kunnen er ten slotte gegevens in het application flash geheugen van de smart LED geplaatst worden. Het uiteindelijke resultaat is het maken van een optische poort dat vanuit de Arduino IDE gegevens uitwisselt met de bootloader van een smart LED met behulp van twee intermediaire microcontrollers.

1 Introductie

Optische communicatie methoden voor het uitwisselen van gegevens gebruiken vaak het zichtbaar spectrum van licht. Ook gegevensoverdracht met infrarood en ultraviolet licht zijn vormen van optische communicatie. Het gebruik van VLC als communicatievorm zou volgens meerdere bronnen radiocommunicatie kunnen evenaren in de toekomst. Dit is omdat VLC communicatie in eerste instantie economische voordelen biedt in vergelijking tot radiocommunicatie [1]. Conclusies over de mogelijke nadelige gezondheidseffecten van radiogolven op de mens bij het langetermijngebruik zijn ook nog niet gekend. De mogelijke gevaren van radiogolven zouden vervolgens het onderzoek in andere communicatievormen moeten aansporen. Dit is een vaststelling doordat er veel gebruik is van apparaten met radiogolven. Deze apparaten zijn overal aanwezig en kunnen in bepaalde gevallen voordeliger zijn met VLC communicatie [2].

1.1 Toepassingen van de VLC technologie

Er is al vastgesteld dat VLC een voordelige keuze kan worden voor optische communicatie in smart parkings en in ziekenhuizen [4]. Het gebruik van VLC is ook voordeliger in beveiligde communicatie. Het vorige wordt besloten doordat het verstuurde signaal niet te onderscheppen is buiten de ruimte waarin het verstuurd is geweest. Radiocommunicatie kan in tegenstelling tot VLC wel met een antenne worden opgevangen. VLC heeft vervolgens minder interferentieproblemen als andere methoden van communicatie. In een ziekenhuiskamer met meerdere communicerende apparaten is er een risico voor de mens als er interferentie is binnen de ruimte [3]. In een smart home kan de verlichting ook communiceren en worden geprogrammeerd met VLC. De luchtvaart communicatie en onderwater communicatie hebben ten slotte ook de voordelen nodig dat VLC biedt en die ontbreken in andere vormen van communicatie. Het risico op interferentie met noodzakelijke apparaten in de omgeving kunnen hiermee verhindert worden [1].

1.2 Bijdragen

De paper bouwt voort uit de bachelorproef van vorig jaar. Het werk van vorig jaar heeft ten eerste onderzoek verricht voor smart phones, LEDs, en camera's die kunnen communiceren met VLC. Vervolgens zijn er methoden voor het oversampelen en decoderen van gegevens onderzocht geweest. Er was ten slotte besloten dat VLC geschikt is als concurrent van radiocommunicatie. Het werk van dit jaar omtrent onderzoek over VLC protocols tussen LEDs en het praktisch onderzoek voor de critera van snelheid en betrouwbaarheid van LED-LED communicatie. Verder wordt er dit jaar een ICSP kanaal gemaakt, zodat het communiceren op afstand naar een smart LED mogelijk is.

[1] Alle software scripts beschreven in deze paper zijn samengezet in een GitHub repository: http://github.com/sarahjosephprojects/ArduinoSupportSmartLEDs.

2 Probleemstelling

2.1 Criteria van onderzoek

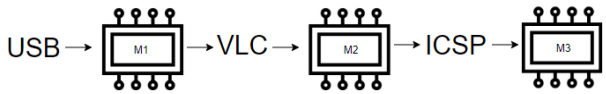
Er zijn aspecten van VLC communicatie tussen LEDs waarvoor er initieel geen directe oplossingen beschikbaar zijn. De LED-LED technologie is in de literatuur gedocumenteerd te werken aan snelheden van 100 tot 200 bits per seconde voor het communiceren van een paar bytes. De VLC technologie gebruikt vervolgens tussen de zender en ontvanger LEDs een korte afstand van maximaal enkele centimeter voor betrouwbare communicatie [6]. De Arduino bestanden die moeten worden verstuurd zijn standaard circa 10 000 bits. De snelheid van verzending moet worden bepaald, zodat het communiceren op een korte afstand kan verlopen met minimale fouten. De specifieke LED-LED afstand en model van LED moet ook bepaald worden zodat het verschil tussen de aan en uit toestand van de verzendende LED goed onderscheidbaar is voor de ontvanger. Er is ook de veranderende omgevingslicht dat een invloed heeft op de detectie van het signaal dat door de LED wordt verstuurt. Hiermee moet er ook rekening gehouden worden voor het verzenden van langere signalen [7].

2.2 Functionele benodigdheden

De asynchrone vorm van communicatie kan in eerste instantie worden geïmplementeerd voor VLC. Methoden voor synchronisatie zijn ook toegepast op VLC, maar er kan als alternatieve methode gekozen worden voor het opsporen en verwijderen van fouten en het behouden van de asynchrone communicatie. Er moet rekening gehouden worden met het percentage van data dat na overdracht gemiddeld correct wordt ontvangen. Dit kan met behulp van een checksum of een parity bit na de programmatie gegevens van een pakket [8]. De zender en ontvanger hebben ook een specifieke opstelling nodig in hun respectievelijke stroomkringen, waardoor de LEDs beiden taken van verzenden en ontvangen kunnen voltooien [4]. Vervolgens bestaat het arduino bestand dat wordt overgedragen uit hexadecimale karakters. Het verzendende bestand wordt met de Universal Serial Bus (USB) protocol gestuurd naar de eerste microcontroller. Deze eerste microcontroller beschikt dan over het verzendende programma en een array van hexadecimale tekens van programmatie gegevens. Bij de ontvangende microcontroller moet de VLC protocol de vertaling terug omkeren met oversampling en on-off keying. De on-off keying methode associeert de aan en uit toestand van de LED respectievelijk met een 1 en een 0. De binaire programmatie gegevens kunnen vervolgens terug worden omgezet in een hexadecimale array in de tweede microcontroller. In het VLC kanaal moet elke sequentie van programmatie gegevens worden omsloten met een detecteerbare hoofd sequentie of *preamble* en een *tail* sequentie. Hiermee kan het begin van deze programmatie gegevens worden gevonden en ook de mogelijke fouten in deze programmatie gegevens.

2.3 Programmatie op afstand

Het VLC protocol verstuurd de programmatie gegevens iteratief vanuit de verzendende microcontroller naar de ontvangende microcontroller in binaire vorm met on-off keying. De tweede microcontroller ontvangt de programmatie gegevens en functioneert als een buffer. De tweede microcontroller is nodig met het gebruik van asynchrone communicatie doordat er fouten moeten worden opgespoord voordat de programmagegevens kunnen worden doorgegeven via ICSP. Het programma moet vervolgens geschreven worden in het uitvoerbare geheugen van de derde microcontroller via de SPI interface van de bootloader. De tweede microcontroller kan daarmee een verbinding maken met de bootloader van de derde microcontroller via het ICSP protocol. Voor het communiceren met de bootloader moet deze initieel weten wat voor apparaat de tweede microcontroller is. Daarna kan de bootloader iteratief programmagegevens lezen vanuit de SPI interface. Deze worden vervolgens door de bootloader geschreven naar het uitvoerbaar geheugen. De tweede microcontroller zal daarvoor iteratief pakketten moeten doorgeven aan de bootloader. Hierdoor zal de Arduino IDE een optische poort ter beschikking hebben naar een LED op korte afstand door het programmeren van een verzendende microcontroller met een array van programmatie gegevens. De protocollen kunnen worden gezien in figuur 1.



Figuur 1: Het USB kanaal wordt gebruikt voor het overdragen van het eerste programma naar microcontroller 1. Tussen microcontroller 1 en microcontroller 2 wordt het VLC protocol gebruikt. Vanuit microcontroller 2 dat functioneert als buffer wordt het ICSP protocol benut voor het programmeren van microcontroller 3 of de smart LED.

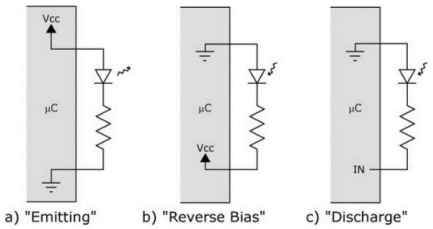
3 Voorgestelde Oplossingen

VLC gebruikt een *light emitting diode* of LED voor het verzenden en ontvangen van licht. Deze diode is een lamp van enkele millimeter dat onder meer twee pinnen bevat. De uiteinden van de diode worden de anode en de kathode genoemd [5]. De anode en kathode kunnen worden verbonden aan de pinnen van een microcontroller dat wordt geprogrammeerd door een computer.

3.1 Emitting en reverse bias toestand

Wanneer er stroom doorheen een LED gaat, zal er energie worden vrijgegeven als licht. De LED lampen hebben een dubbele functionaliteit en kunnen ook gebruikt worden als licht sensoren. Hiervoor moet de stroomkring een opstelling hebben in reverse bias. De fotonen die vervolgens invallen op

de diode worden gedetecteerd. Deze methoden zijn beschreven in figuur 2. In het eerste gedeelte geeft figuur 2a de emitting toestand weer. De stroom is in een voorwaartse richting en de LED wordt verlicht. Figuur 2b geeft de reverse bias toestand weer. Een condensator binnen de opstelling wordt opgeladen waardoor het systeem voorbereid is voor een meting. De meting wordt vervolgens verricht in de discharge toestand in figuur 2c. In deze laatste toestand wordt de tijdsduur gemeten voordat het invallend licht de input pin terug reset tot een threshold value. Dit gebeurt tijdens de ontlading van de condensator. De voorgestelde oplossing heeft een verzendende stroomkring met een LED in de emitting toestand en een ontvangende stroomkring met een LED in de reverse bias toestand [4]. De stroomkringen zijn beiden verbonden aan hun eigen microcontrollers die elektrisch worden aangedreven via USB.



Figuur 2: De licht emitting, reverse bias en ontladingstoestand kunnen voor de LED opstelling in de stroomkring benut worden voor het verzenden en ontvangst van lichtsignalen. Deze figuur is vanuit Dietz et al. Very Low-Cost Sensing (2003) [4].

3.2 Verzending in pakketten

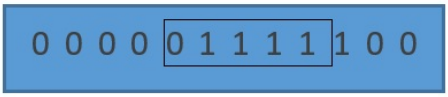
Met de bovenstaande ontwerpen voor de verzender en ontvanger kan de eerste microcontroller een signaal versturen dat gedecodeerd wordt met behulp van een threshold value. Deze waarde is het middelpunt tussen de gemiddelde waarde gedetecteerd voor de aan toestand en de uit toestand van het lampje [7]. De threshold bepaalt in de ontvangende LED vervolgens wat het verzonden signaal is [6]. Het bericht moet vervolgens ook gestructureerd worden voor detectie en verificatie in de ontvanger door middel van pakketten. De opeiningskeuze van het pakket is vervolgens beschreven in figuur 3. Elk pakket dat wordt ontvangen wordt gevonden door de preamble. De XOR checksum op het einde van de pakket verdeelt de sequentie van programmatie gegevens in twee stukken van 4 bits. Op elke index van de twee gevormde lijsten en dus voor die welbepaalde twee elementen voert de functie een XOR operatie. De resulterende 4 bits van de XOR checksum vormen de error detectie gedeelte om informatie in de data sequentie te verifiëren als een correct ontvangen signaal [8].

3.3 Oversampling en data-analyse

Het pakket met de preamble, data sequentie en checksum wordt in bits verstuurd over het VLC kanaal. Voor elke bit

Figuur 3: Het pakket heeft een preamble van 8 bits, een data sequentie van één byte en een XOR checksum van 4 bits.

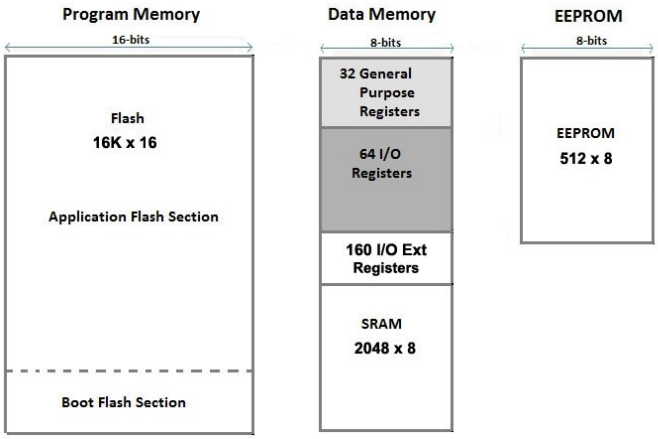
dat wordt verstuurd door de eerste microcontroller neemt de tweede microcontroller 5 samples. De methode van oversampling geeft accurater weer wat het verstuurde signaal was. De perioden van verzending en ontvangst van beide microcontrollers verschillen in kleine hoeveelheden. Door middel van oversampling zullen fouten met betrekking tot de asynchrone communicatie worden verminderd. Vervolgens wordt er over de ontvangen gegevens meermaals gedecodeerd. Hiervoor wordt de array met de ontvangen gegevens verschoven met één en twee indexen in de voorwaartse en in de achterwaartse richting. Indien de gevonden waarde eerst een 1 was, dan zal de kader met een gemiddelde waarde het dichste bij 1 gekozen worden uit de vijf kaders. Indien de gevonden waarde initieel 0 was, dan zal het gemiddelde dat het dichste bij 0 is gekozen worden uit de vijf kaders. De index zal vervolgens resetten tot de geselecteerde kader voor het verder decoderen van de array, zoals te zien in figuur 4.



Figuur 4: De huidige kader heeft een gemiddelde van 0.8 en de gedecodeerde waarde is 1, doordat het minder afstand heeft tot 0.8 dan de waarde 0. De kader wordt verschoven in de voorwaartse richting en achterwaartse richting. Bij een verschuiving van één index in de voorwaartse richting wordt het gemiddelde maximaal bij 1. Deze kader wordt dan geselecteerd en de index wordt met één opgehoogd voor het verder decoderen van de ontvangen gegevens.

3.4 Programmatie met ICSP interface

In de tweede microcontroller wordt er na het verzamelen van de programmatie gegevens een array opgevuld met elementen van twee hexadecimale symbolen. Deze byte array wordt gebruikt om de laatste microcontroller te programmeren met behulp van zijn bootloader gedeelte. De tweede microcontroller kan eveneens gezien worden de functie te benuttigen van een buffer voor het programmeren van de derde microcontroller. Het geheugen van elke microcontroller bevat de applicatie flash geheugen en de bootloader gedeelte van het geheugen. Dit kan worden gezien op figuur 5. De bootloader is een aparte en kleine gedeelte op het einde van de applicatie flash geheugen dat niet via de Arduino IDE kan worden geprogrammeerd [9]. De programmatie van de bootloader wordt uitgevoerd met ICSP. De geheugen van de bootloader kan worden ingesteld tussen 512 en 4096 bytes in de derde microcontroller. De applicatie flash geheugen dat kan gebruikt worden in de tweede microcontroller heeft 32 kilobytes ter beschikking. Voor deze redenen is de tweede microcontroller benut geweest als een buffer voor het verwerken van de ontvangen gegevens.



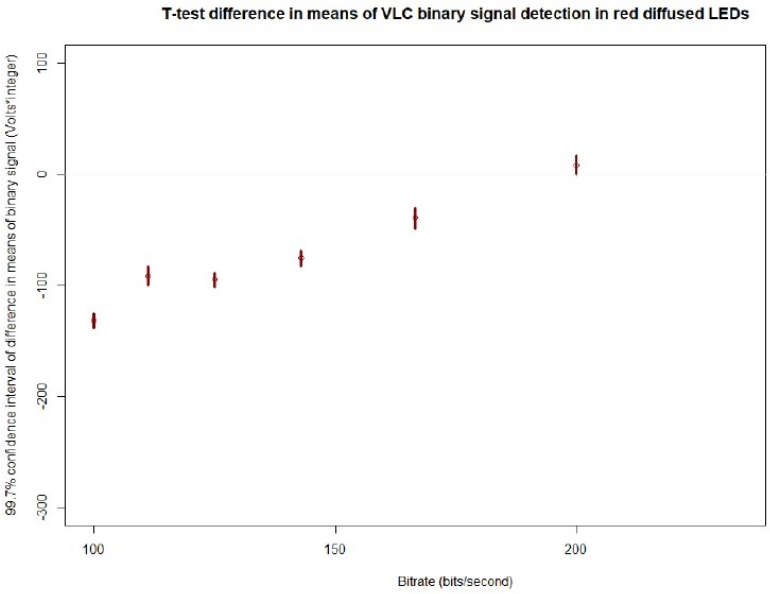
Figuur 5: De microcontroller heeft verschillende soorten van geheugen waarbij de flash geheugen bestaat uit de applicatie flash geheugen en de bootloader geheugen. Deze illustratie kan worden gevonden in de Arduino SPI documentatie [9].

Vervolgens kan de In-circuit Serial Programmer met zekerheid worden gebruikt om applicatie geheugen en eveneens het bootloader gedeelte van een aangesloten microcontroller te programmeren. Er wordt gekozen om de applicatie flash geheugen van de tweede microcontroller te programmeren. Hiermee kan doel van het programmeren van een Smart LED eveneens bereikt worden. De bootloader kan daarmee behouden worden om inkomende bytes van de tweede microcontroller te verplaatsen naar zijn eigen applicatie flash geheugen. Dit is nodig omdat de applicatie flash geheugen alleen kan bereikt worden via de bootloader tijdens de uitvoering van een programma. De bootloader op de derde microcontroller gebruikt vervolgens de Serial Peripheral Interface om de gegevens te ontvangen en op te slaan in het applicatie flash geheugen [9]. De serial communicatie vorm beduidt dat er één bit tegelijk wordt verstuurd in de interface gegevensstroom. Vervolgens kan met de SPI elke byte verzonden worden vanuit de tweede microcontroller waarna de bootloader van de derde microcontroller het ontvangen data schrijft naar het applicatie flash geheugen. Nadat het volledige programma is verzonden kan de uitvoering van het programma worden geactiveerd op de derde microcontroller.

De ICSP van de tweede naar de derde microcontroller wordt geïmplementeerd met behulp van de ArduinoISP. Dit bestand bevat de nodige methoden voor het programmeren van de derde microcontroller. Met behulp van de ArduinoISP kunnen de ontvangen gegevens na verficatie opgeslagen worden in een byte array in de tweede microcontroller. De byte array kan integreren binnen de programmatiecyclus van de ArduinoISP. De programmatie gebruikt ook hexadecimale data en wordt in bytes verstuurd. Na de reset methode is het apparaat klaar om gegevens te ontvangen via de SPI. Vervolgens wordt de derde microcontroller geinitialiseerd om instructies te accepteren en het kenteken van de tweede microcontroller op te slagen. Hierna worden de programmagegevens in gedeeltes van 133 bytes geschreven naar de applicatie flash geheugen en ten slotte wordt de programmatie geverifieerd voordat er een disconnect signaal wordt verstuurd. De programmatie wordt ten slotte geverifieerd door het lezen van de applicatie flash geheugen met AVR software.

4 Evaluatie

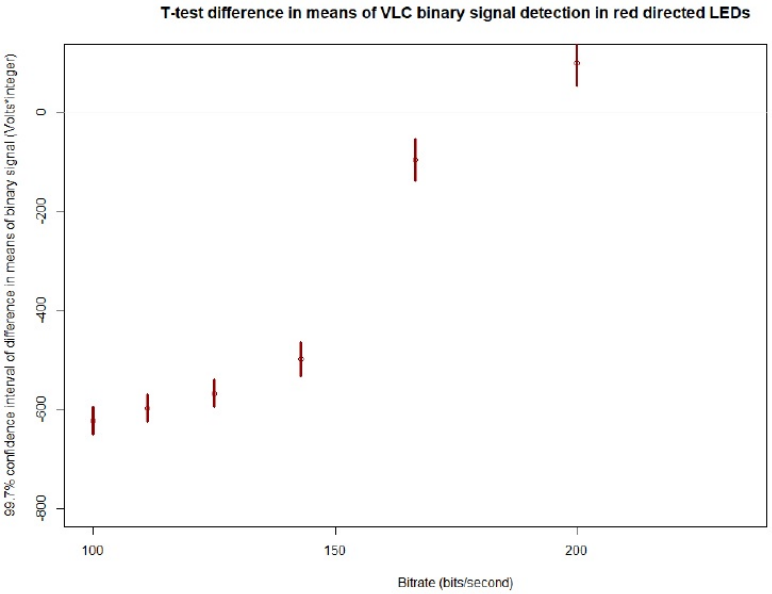
Het onderzoek werd gestart met de criteria van betrouwbaarheid in LED-LED communicatie. Hiervoor werd de communicatieafstand en type van LED onderzocht. Vervolgens zijn er experimenten uitgevoerd om de oorzaak van de gedetecteerde fouten te vinden bij ontvangst. De correctie van fouten zijn ten slotte met oversampling experimenten onderzocht geweest. Er is bevonden dat gerichte LEDs een hogere betrouwbaarheid hebben. De synchronisatie heeft vervolgens een grote invloed op de percentage van fouten. Het oversampelen en synchroniseren kunnen daarmee de hoeveelheid fouten sterk verminderen.



Figuur 6: De rode balk van de confidence interval geeft weer wat het verschil is tussen de ingelezen waarde voor de 0 bit en de 1 bit bij diffuse LEDs. Dit is geplot over de bitrate in seconden.

4.1 Diffuse en gerichte LEDs

De eerste hypothese is dat gerichte LEDs een betere optie vormen voor VLC in vergelijking tot diffuse LEDs. Diffuse LEDs zijn lichtbronnen die in alle richtingen schijnen in tegenstelling tot gerichte LEDs. De resultaten zijn weergegeven in figuur 6 voor diffuse LEDs en in figuur 7 voor gerichte LEDs. Voor het eerste experiment werd een wisselend signaal van 0 en 1 verstuurd. Hierna werd het ontvangen signaal per bit afwisselend toegevoegd aan twee aparte arrays voor het

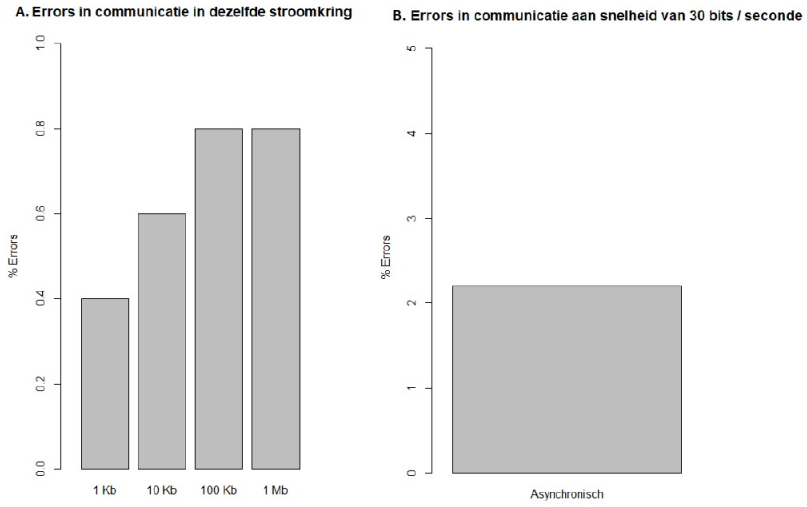


Figuur 7: De balk van de confidence interval geeft weer wat het interval van het verschil is in de gerichte LED tussen de ingelezen waarde voor de 0 bit en de 1 bit. Dit is geplot over de bitrate in seconden.

respectievelijk bewaren van het ontvangen 0 en 1 signaal. Een t-test werd vervolgens gebruikt om het verschil tussen de gemiddelden van de twee arrays te berekenen met een betrouwbaarheidsinterval. In verder detail is de hypothese dat er een groter verschil in het gemiddelde zou zijn bij gerichte LEDs, waardoor ze een betere optie vormen voor VLC. De bovenstaande figuur geeft weer dat er bij diffuse LEDs geen significant verschil meer is tussen de gedetecteerde waarde voor de 0 bit en de 1 bit wanneer de bitrate 200 bits per seconde bedraagt. Het berekende interval snijdt daar de x-as en bevat de nulwaarde. Hierdoor kan er besloten worden dat er vanaf deze snelheid geen asynchrone signalen meer gestuurd kunnen worden met een diffuse LED. Vervolgens werd hetzelfde experiment uitgevoerd met een gerichte LED. De resultaten zijn gegeven in figuur 8. Hier kan worden gezien dat er bij asynchrone communicatie nog steeds een significant verschil is bij 200 bits per seconde. Daarmee is er besloten geweest om te kiezen voor de gerichte LEDs voor het communiceren via VLC met de smart LED.

4.2 Foutenanalyse

Het tweede experiment probeert te achterhalen als de fouten in de communicatie te wijten zijn aan de apparatuur of aan de asynchrone communicatie. De hypothese is dat de fouten te wijten zijn aan asynchrone communicatie. Voor experiment twee is er een alternatieve stroomkring gemaakt waarbij beiden de zender en de ontvanger in dezelfde stroomkring gezet zijn geweest. Hierdoor is de communicatie gesynchroniseerd vanaf het begin. In figuur 8 is asychrone communicatie met twee stroomkringen vergeleken met synchrone communicatie binnen eenzelfde stroomkring. Hieruit kan gezien worden dat er binnen dezelfde stroomkring minder gedetecteerde fouten zijn bij het verzenden en ontvangen van gegevens. Met synchronisatie kunnen ook grote snelheden bereikt worden met fouten onder 1 percent van de gegevens. Het versturen van een signaal en het ontvangen van datzelfde signaal verloopt hier direct achter elkaar in het programma. Bij het asynchronisch communiceren aan een lagere snelheid van 30 bits per seconden zijn er 2 percent aan fouten ontvangen. De gebruikte apparatuur is hetzelfde en daarmee kan vervolgens besloten worden dat het lichtsignaal aan hoge snelheden kan ontvangen worden met minimale fouten indien het verzenden en ontvangen helemaal gesynchroniseerd zijn. Bij het asynchronisch communiceren zijn er periodische fouten die moeten worden gedetecteerd.



Figuur 8: A. Percentage van fouten bij communicatie binnen dezelfde stroomkring aan hoge snelheden. B. Percentage van fouten bij communicatie tussen twee microcontrollers met een snelheid van 30 bits per seconde.

4.3 Oversampling

Het derde experiment test de invloeden van oversampling en synchronisatie op het percentage van fouten. De hypothese is vervolgens dat het oversampelen en synchroniseren zullen resulteren in minder fouten. In figuur 8 kan er gezien worden dat er aan 30 bits per seconden 2 percent fouten zijn in de data. Hierdoor zal er om de 50 bits een fout worden gedetecteerd. Daarmee kunnen gegevens van een grotere lengtes niet meer betrouwbaar verstuurd worden. Na het vijf maal oversampelen van het ontvangen signaal en het synchroniseren volgens paragraaf 3.3 zijn de fouten geminimaliseerd. Hierna waren er geen fouten meer gedetecteerd geweest aan een lage bitrate van 5 bits per seconden met een sample size van 50 pakketten. Deze trage communicatievorm zou daarmee betrouwbaar kunnen gebruikt worden voor het ontvangen van een lange array van bytes.

5 Verder werk

5.1 Foutenanalyse

De methoden voor het detecteren van fouten zouden kunnen worden uitgebreid. Aangezien er initieel bij asynchronische communicatie periodische fouten zijn, kunnen deze worden gecorrigeerd door middel van bit-stuffing. Hierna zou er kunnen gecommuniceerd worden aan hogere snelheden. Voor de bit-stuffing kan er een flag byte voor de preamble en achter de checksum worden toegevoegd. Indien er één bit is ontsnapt in één van de twee flags, dan kan er met bit-stuffing terug één bit ingezet worden na de tweede flag [15]. Hiermee zal er een additionele laag voor synchronisatie kunnen worden gemaakt in de tweede microcontroller.

5.2 Geheugenallocatie

Geheugenallocatie in de tweede microcontroller kan worden aangepast zodat er meer foutencorrectie functies kunnen worden aangemaakt. Er moet ook genoeg plaats zijn voor de ontvangen programmatie gegevens. Momenteel worden de programmatie gegevens die telkens weer worden gebruikt voor ICSP opgeslagen in het flash geheugen. De veranderlijke programmatie gegevens specifiek aan het verstuurde programma worden opgeslagen in SRAM. Bij het toevoegen van een foutendetectie functie kan er worden besloten welke gegevens van het programma in SRAM kunnen staan en welke onveranderlijke gegevens in het flash geheugen kunnen worden gezet. Gegevens die tijdens het uitzetten van de microcontroller moeten blijven onthouden worden kunnen toegewezen worden naar EEPROM.

6 Conclusies

Het gebruik van VLC voor het uitwisselen van gegevens heeft veel toepassingen. VLC heeft als een unieke eigenschap dat het geen interferentieproblemen geeft met apparatuur in de omgeving. De opstelling is economisch voordelig en innovaties met VLC kunnen apparaten met radiocommunicatie vervangen. Er is ook het aspect van lange termijn gezondheidseffecten op de mens, waarbij VLC een veiligere optie biedt in vergelijking tot radiocommunicatie. VLC kan worden gebruikt voor het versturen van korte gegevens en werkt best op een korte afstand van 10 millimeter. De communicatie van korte gegevens is toegankelijk bij snelheden van circa 100 bits per seconde. Hiervoor worden gestructureerde pakketten gemaakt voor detectie en verificatie van de data sequenties. Het implementeren van oversampling en synchronisatie maakt VLC veel meer betrouwbaar voor lage snelheden van 5 bits per seconde. Hiermee kunnen langere gegevensstromen worden doorgestuurd. Voor het programmeren van een smart LED op afstand kan het ICSP protocol worden geïmplementeerd in een microcontroller dat functioneert als een buffer. Het VLC signaal wordt naar de buffer gestuurt en deze verwerkt het signaal waarna het via ICSP communiceert met een smart LED. De SPI bus en de bootloader van de smart LED worden gebruikt om data te verplaatsen naar de applicatie flash geheugen. VLC kan ten slotte worden uitgebreid met additionele algoritmen voor foutencorrectie en synchronisatie, zodat de communicatie kan verlopen aan hogere snelheden.

7 Bibliografie

1. The Future of VLC: Potential and Limitations, 2014. Prof. Maite Brandt-Pearce, University of Virginia Engineering.
2. Radio frequency radiation (RFR): the nature of exposure and carcinogenic potential. P. A. Valberg Cancer Causes Control. 1997 May; 8(3): 323–332. doi: 10.1023/A:1018449003394.
3. K Foster. "Radiofrequency Interference With Medical Devices". In: IEEE Eng Med. Bio. Mag. 17.3 (1998), p. 111–114.
4. Dietz, Paul Yerazunis, William Leigh, Darren. (2003). Very Low-Cost Sensing and Communication Using Bidirectional LEDs. 175-191. 10.1007/978-3-540-39653-614.
5. Schmid, S. et al. (2012) An LED-to-LED Visible Light Communication system with software-based synchronization [PDF presentation]. Available at: https://www.bu.edu/smartlighting/files/2012/10/Schmid.pdf (Accessed: 6 May 2019).
6. An LED-to-LED Visible Light Communication System with Software-Based Synchronization. ETH Zurich, Disney Research Zurich.
7. Adaptive visible light communication LED receiver, 2017. IEEE Sensors.
8. Schmidt (2019). Chapter 5: Error-Correcting Codes [online] Available at: http://people.cs.ksu.edu/~schmidt/115/ch5.html [Accessed 6 May 2019].
9. Arduino. (2019). ArduinoISP. [online] Available at: https://www.arduino.cc/en/Tutorial/ArduinoISP [Accessed 6 May 2019].
10. Arduino. (2019). Software Serial. [online] Available at: https://www.arduino.cc/en/Reference/SoftwareSerial [Accessed 6 May 2019].
11. Atmel AVR2054: Serial Bootloader User Guide. (2015). Firmware programming over USART/SPI/TWI and other serial interfaces. Atmel.
12. 8-bit Microcontroller with Low Power 2.4GHz Transceiver for ZigBee and IEEE 802.15.4. (2014). ATmega128RFA1. Atmel.
13. Logos Electromechanical, Zigduino, (2013), GitHub repository. https://github.com/logos-electromechanical/Zigduino
14. Arduino. (2019). Serial. [online] Available at: https://www.arduino.cc/reference/en/language/functions/communication/serial/ [Accessed 6 May 2019].
15. Tanenbaum, A.S., Wetherall, D. (2010). Computer networks, 5th Edition.