Integratie van Draadloze Communicatie Aspecten
Hogeschool Utrecht

Onderzoeksverslag

Auteur	*René de Kluis*
Student nummer	*1661627*
Datum	*28-03-2019*
Locatie	*Heidelberglaan 15, Utrecht*
Versie	*V2.0.0*
Status	*Oplevering 2*

Versiebeheer

De versie van dit document is genummerd door middel van drie getallen X, Y en Z. Hierbij staat de X voor de uitgave versie en Y voor de uitgebrachte conceptversie. Veranderingen van deze versies zullen opgenomen worden in de onderstaande tabel. De Z voor elke aanpassing dat aangebracht binnen de huidige conceptversie.

Versie	Beschrijving	Datum
Versie 1.0.0	Eerste Oplevering	06-01-2019
Versie 1.1.0	Verbeterd concept document	11-02-2019
Versie 2.0.0	Tweede Oplevering	28-03-2019

Voorwoord

Voor u ligt het onderzoeksverslag “Integratie van draadloze communicatie aspecten”. Het onderzoek voor dit verslag is uitgevoerd bij de Hogeschool Utrecht. Dit verslag is geschreven in het kader van mijn onderzoek semester dat deel uit maakt van het vierde jaar van de afstudeerrichting Technische Informatica van HBO-ICT. Het onderzoek is gedaan in opdracht van de Hogeschool Utrecht. Van september 2018 tot en met januari 2019 ben ik bezig geweest met het onderzoek en het schrijven van het verslag.

Samen met mijn begeleider, Wouter van Ooijen, heb ik een opdracht voor het onderzoek bedacht en na goedkeuring van het begeleidend docent, Huib Aldewereld, uitgevoerd.

Bij dezen wil ik graag mijn begeleider bedanken voor de begeleiding en zijn ondersteuning tijdens dit traject. Ook wil ik mijn vriendin bedanken voor het nalezen van het verslag op spellings- en grammatica fouten.

René de Kluis

IJsselstein, 28 maart 2019

Managementsamenvatting

Aangezien Internet of Things toepassingen steeds vaker voor komen in onze maatschappij, wil de Hogeschool Utrecht dit opnemen in het curriculum Technische Informatica van HBO-ICT. Alleen wordt bij deze opleiding gewerkt met de Arduino Due. Dit is een microcontroller waarmee studenten moeten programmeren. Deze microcontroller heeft geen geïntegreerde Wifi of Bluetooth modules, waardoor draadloze communicatie aspecten niet in de huidige cursussen van Technische Informatica voor komen.

De doelstelling van dit onderzoek is binnen het curriculum Technische Informatica van de Hogeschool Utrecht een nieuwe microcontroller te implementeren, waarbij een Wifi en/ of Bluetooth module geïntegreerd zit. Aangezien de studenten werken met libraries en tools van die door de Hogeschool Utrecht gemaakt zijn, zal de aansturing van de microcontroller hier ook in opgenomen dienen te worden. Voor het onderzoek is de volgende onderzoeksvraag opgesteld: *"Hoe kan een nieuwe microcontroller met geïntegreerde wifi en/of bluetooth in de huidige libraries en tools van de Hogeschool Utrecht opgenomen worden, zodat draadloze communicatie aspecten toegevoegd kunnen worden in de opgaven van Technische Informatica?"*

Het opnemen van een nieuwe microcontroller in de libraries en tools van de Hogeschool Utrecht kan met de volgende stappen. Eerst moet de aansturing van de microcontroller gerealiseerd worden in de BMPTK make tool. Deze tool, gemaakt door Wouter van Ooijen, is een op GNU make gebaseerde build tool, waarmee een programma voor verschillende microcontrollers gecompileerd en gebouwd kan worden. Daarna zal de Wifi en/ of Bluetooth library in HWLIB opgenomen moeten worden. HWLIB is een C++ library dat gemaakt is door Wouter van Ooijen en heeft verschillende functionaliteiten voor de aansturing van hardware en verschillende communicatieprotocollen. Als laatste zal de RTOS library van de Hogeschool Utrecht, dat gemaakt is door Wouter van Ooijen en Martin Wensink, opgenomen worden in het bouwproces, waardoor deze gebruikt kan worden op de microcontroller.

Na onderzoek bleek dat de esp32 van Espressif de beste keuze was om te gebruiken voor dit onderzoek. Om de aansturing van de esp32 te integreren in de BMPTK make tool is gekeken naar ESP-IDF. Dit is een raamwerk dat van Espressif wat de aansturing van de esp32 verzorgt en verschillende functionaliteiten bevat die op de esp32 gebruikt kunnen worden. In de ESP-IDF moest gekeken worden naar de verschillende componenten die benodigd waren voor de aansturing en welke vlaggen in de Makefiles van de BMPTK make tool opgenomen moesten worden.

Wanneer een Wifi en/ of Bluetooth library gerealiseerd is, kan deze toegevoegd worden aan de HWLIB library. Wanneer in de headerfiles en Makefiles van HWLIB aangegeven is wat de hoofdbestanden zijn van de gerealiseerde libraries kunnen deze in de code gebruikt worden.

In de aansturing van de esp32 vanuit ESP-IDF, wordt gebruik gemaakt van FreeRTOS. Hierdoor was het tijdens dit onderzoek niet mogelijk om de RTOS library van de Hogeschool Utrecht op te nemen in de aansturing. Wanneer de esp32 in de cursussen opgenomen wordt en gebruik gemaakt dient te worden van de RTOS van de Hogeschool Utrecht, zal een vervolg onderzoek nodig zijn.

Op basis van dit onderzoek wordt geadviseerd om de esp32 bij alle cursussen in te zetten. Dit zorgt voor uniformiteit binnen het curriculum.

Inhoudsopgave

1. Inleiding

In het vierde jaar van HBO-ICT hebben studenten de keuze om een onderzoeksemester te doen. Hierbij wordt een onderzoek gedaan naar een onderwerp dat door de student gekozen is. Dit verslag zal gaan over een onderzoek dat door René de Kluis uitgevoerd is bij Hogeschool Utrecht.

§1.1 Organisatorische Context

HBO-ICT bestaat uit de richtingen Software-Netwerk Engineering, Software-Information Engineering, Business-IT & Management en Technische informatica. Deze opleidingen zijn allemaal bachelor niveau. Dit onderzoek is gemaakt door een student van Technische informatica en het verslag is heeft deze studierichting ook als doelgroep.

Het onderzoek is voornamelijk uitgevoerd in het Turing Lab van de Hogeschool Utrecht. Dit is een werkplek met meerdere elektronische componenten en apparaten voor studenten die met technische elementen willen werken. Het bevind zich op de vierde verdieping van de Heidelberglaan 15 in Utrecht.

§1.1.1 Relatie tot andere projecten

Binnen de Hogeschool Utrecht wordt altijd gezocht naar vernieuwing en verrijking van de lesstof. Hierbij hoort ook het integreren van nieuwe microcontrollers in de lesstof. In dit onderzoek zal niet gewerkt worden met libraries en tools van de fabrikant van de microcontroller, maar met libraries en tools die door de Hogeschool Utrecht gemaakt zijn. Het zal hier gaan om de HWLIB en RTOS libraries en de BMPTK make tool. HWLIB, gemaakt door Wouter van Ooijen, is een C++ library waarmee, object georiënteerd, hardware aangestuurd kan worden. De RTOS (Real Time Operating System) library, gemaakt door Wouter van Ooijen en Marten Wensink, wordt gebruikt voor programma's waarbij de gebruiker op aangeduide tijdstippen bepaalde elementen wilt uitvoeren. Deze twee libraries worden in meerdere cursussen van Technische Informatica gebruikt. Het is de bedoeling dat deze libraries hardware abstractie bevatten, wat betekend dat de library voor elke mogelijke microcontroller gebruikt kan worden zonder aanpassingen. De BMPTK make tool, gemaakt door Wouter van Ooijen, is een op GNU make gebaseerde ontwikkel omgeving, waarmee een programma gemaakt kan worden voor microcontrollers. Deze tool zorgt dat de geschreven code op de hardware terecht komt en dat de microcontroller de code ook kan uitvoeren.

Bij de Hogeschool Utrecht streven ze naar herbruikbaarheid van deze libraries en tools. Bij het toevoegen van een nieuwe microcontroller zal hiernaar gekeken moeten worden. Dit onderzoek kan daardoor nieuwe inzichten geven of aanpassingen in de libraries of tools gemaakt dienen te worden, voor volledige herbruikbaarheid.

§1.1.2 Contact

Onderzoeker

Naam	*René de Kluis*
Student nummer	1661627
Mobiel	+31(0)6 429 405 74
E-Mail	rene.dekluis@student.hu.nl

Opdrachtgever

Naam	*Wouter van Ooijen*
Locatie	Heidelberglaan 15 – 4.060 3512 JE, UTRECHT
Telefoon	+31 (0)6 38150444
E-Mail	Wouter.vanooijen@hu.nl

§1.2 Probleemstelling

In dit deze paragraaf zal de aanleiding voor dit onderzoek besproken worden (de kwestie) en een onderzoeksvraag gevormd worden wat de rode draad zal vormen voor dit onderzoek. Aan het eind van dit document zal deze onderzoeksvraag beantwoord worden. Bij de kwestie zal het huidige probleem beschreven worden en de wensen van de opdrachtgever.

§1.2.1 Kwestie

Bij de afstudeerrichting Technische Informatica van HBO-ICT aan de Hogeschool Utrecht, wordt voor de meeste technische opdrachten de Arduino Due gebruikt. Dit is een programmeer bordje, waarmee onder andere sensoren uitgelezen en aangestuurd kunnen worden. De Arduino Due heeft geen geïntegreerde wifi of bluetooth modules, waardoor deze draadloze communicatie aspecten niet voorkomen in het curriculum van Technische Informatica.

Aangezien Internet of Things (IoT) toepassingen steeds meer in opkomst zijn, wil de Hogeschool Utrecht dit ook opnemen in de cursussen. Hiervoor willen zij een microcontroller gebruiken waarbij wifi en/of bluetooth geïntegreerd zit. Aangezien deze aspecten als toevoeging moeten dienen bij de huidige opgaven, zullen de huidige opgaven die voor de Arduino Due gegeven worden, met kleine aanpassingen ook op de nieuwe microcontroller moeten werken. Daarnaast wilt de Hogeschool Utrecht dat studenten op dezelfde manier kunnen programmeren voor verschillende microcontrollers, hiermee zal rekening gehouden moeten worden bij het toevoegen van de nieuwe microcontroller.

§1.2.2 Vraagstelling

Om Bluetooth en Wifi aspecten toe te voeren in de lesstof van Technische Informatica van de Hogeschool Utrecht, kan de volgende onderzoeksvraag opgesteld worden:

“Hoe kan een nieuwe microcontroller met geïntegreerde wifi en/of bluetooth in de huidige libraries en tools van de Hogeschool Utrecht opgenomen worden, zodat draadloze communicatie aspecten toegevoegd kunnen worden in de opgaven van Technische Informatica?”

Aangezien de gehele onderzoeksvraag complex is, is gekozen om deze op te delen in verschillende deelvragen. Deze deelvragen zullen uiteindelijk samen een antwoord kunnen geven op de onderzoeksvraag. Hieronder zullen de deelvragen uitgewerkt worden die uit de onderzoeksvraag gefilterd kunnen worden:

1. Welke microcontroller ondersteunen wifi en/of bluetooth?
2. Met welke microcontrollers, die wifi en/of bluetooth ondersteunen, kunnen de huidige opgaven van Technische Informatica gerealiseerd worden?
3. Welke aanpassingen zijn nodig om de gekozen microcontroller op te nemen in libraries en tools van de Hogeschool Utrecht?
4. Op welke manier kunnen wifi en/of bluetooth aspecten toegevoegd worden in de opgaven van Technische Informatica?
5. Hoe kan een wifi library opgenomen worden in HWLIB?
6. Hoe kan een bluetooth library opgenomen worden in HWLIB?

2. Theoretisch kader

In dit hoofdstuk zal de belangrijkste theoretische achtergrond van dit onderzoek besproken worden. Hierbij wordt in de eerste paragraaf duidelijkheid verschaft over de kenbegrippen die in de documenten centraal zullen staan en welke relaties de kernbegrippen met elkaar hebben.

§2.1 Kernbegrippen

In deze paragraaf zal verduidelijking worden gegeven over begrippen die in dit document centraal zullen staan. Als eest zal de herbruikbaarheid van libraries en tools van de Hogeschool Utrecht behandeld worden en daarna hardware abstractie waarmee de integratie van de gekozen microcontroller gerealiseerd zal worden.

§2.1.1 Herbruikbaarheid van libraries

Bij de herbruikbaarheid van de libraries van de Hogeschool Utrecht zal gekeken op welke manier de gekozen microcontroller geïntegreerd kan worden en welke eventuele problemen dit met zich mee brengt. Voor de studenten moet de aansturing van de gekozen microcontroller op dezelfde manier gebeuren, zoals dit nu kan met de Arduino Due. Daarvoor zal gekeken moeten worden naar modulariteit van de libraries en tools van de Hogeschool Utrecht, zodat deze gebruikt kunnen worden voor de gekozen microcontroller.

§2.1.2 Hardware abstractie

Bij hardware abstractie laat de programmeur de software geloven dat alle mogelijke hardware aanwezig is, maar in feite zal de software alleen de functionaliteiten uitvoeren voor de aangesloten hardware. Hierdoor maakt het voor het systeem niet uit als net een ander module of sensor aangesloten wordt op het systeem.
In de huidige libraries van de Hogeschool Utrecht voor de aansturing van hardware is hardware abstractie aanwezig. Het is namelijk mogelijk om bijvoorbeeld de Arduino Uno, Arduino Nano en Arduino Due op dezelfde manier aan te sturen. Aangezien de gekozen microcontroller hierbij toegevoegd dient te worden, zal hierdoor gekeken moeten worden naar de huidige hardware abstractie en of dit herbruikbaar is.

3. Keuze Microcontroller

Voor het uitkiezen van een microcontroller voor het onderzoek, zal gekeken moeten worden welke microcontrollers wifi en/ of bluetooth ondersteunen. In dit hoofdstuk zal daarom uitgezocht worden bij welke microcontrollers één of beide aspecten geïntegreerd zit.

Aangezien studenten met de nieuwe microcontroller moeten gaan werken, zijn een aantal filteringen gekozen bij het zoeken naar een geschikte microcontroller met een geïntegreerde wifi en/ of bluetooth module. Na overleg met de opdracht gever is besloten om microcontroller boven de €50,00 niet mee te nemen in de analyse. De rede hiervoor is dat de studenten de microcontroller zelf moeten aanschaffen en daarom de prijs niet te hoog mag liggen. Daarnaast zullen microcontrollers met zeer slechte of geen verkrijgbaarheid ook niet meegenomen worden, zodat studenten zonder te veel moeite de microcontroller kunnen aanschaffen. Na onderzoek zijn 11 microcontrollers gevonden die aan deze criteria voldeden. Deze microcontrollers zullen in de onderstaande tabel uitgewerkt worden, samen met de specificaties die van belang zijn voor het onderzoek. De zoektermen die gebruikt zijn voor het vinden van de microcontrollers en verdere specificaties van de microcontrollers zullen uitgewerkt worden in appendix B.

Naam Microcontroller	***Fabrikant***	***Processor***	***Klok snelheid (MHz)***	***ROM (KB)***	***RAM (KB)***	***Wifi***	***BT***	***Prijs[1]***
Bluno Nano	Arduino	Atmega328	16	1	32	Nee	Ja	€30,-
ESP32	Espressif	Xtensa LX6	240	448	520	Ja	Ja	€7,-
ESP8285	Espressif	Xtensa L106	26	*X*[2 3]	50	Ja	Nee	€2,-
ESP8266	Espressif	Xtensa L106	80	64[2]	32	Ja	Nee	€7,-
MT7681	MediaTek	Andes N9	80	*X*[3]	64	Ja	Nee	€4,-
nRF51822	Nordic	Cortex-M0	64	*X*[3]	32	Nee	Ja	€5,-
nRF52832	Nordic	Cortex-M4	64	*X*[3]	64	Nee	Ja	€6,-
NL6621	NuFront	Cortex-M3	40	96	192	Ja	Nee	€3,-
RTL8710	RealTek	Cortex-M3	125	1024	512	Ja	Nee	€5,-
RTL8195	RealTek	Cortex-M3	166	1024	512	Ja	Nee	€40,-
CC3200	Texas Instruments	Cortex-M4	80	10	256	Ja	Nee	€32,-

1 – De prijs is een gemiddelde van verschillende prijzen die op internet gevonden zijn.
2 – Geen programmeerbare ROM.
3 – Hoeveelheid ROM kon niet duidelijk gevonden worden.

Tabel 3.1 – Specificaties microcontrollers

De nieuwe microcontroller moet de Arduino Due vervangen in het curriculum Technische Informatica. Hierdoor is in dit onderzoek gekozen dat bij het kiezen van de microcontroller de klok snelheid, ROM grootte en RAM grootte minimaal moeten voldoen aan die van de Arduino Due. In de onderstaande tabel staan deze specificaties van de Arduino Due uitgewerkt, samen de onderzochte microcontrollers die aan deze eisen voldoen.

Naam	***Klok snelheid***	***ROM***	***RAM***
Arduino Due	84 MHz	16 KB	96 KB
ESP32	240 MHz	448 KB	520 KB
RTL8710	125 MHz	1024 KB	512 KB
RTL8195	166 MHz	1024 KB	512 KB

Tabel 3.2 – Potentiële microcontrollers

De vraagstelling van dit onderzoek stelt dat Wifi en/ of Bluetooth aspecten in de cursussen verwerkt dienen te worden. Om dit onderzoek toekomstbestendig te maken is daarom gekozen voor de esp32 van Espressif. Deze is namelijk de enige van de onderzochte microcontroller waarbij een Wifi en Bluetooth module geïntegreerd zit. De Hogeschool Utrecht kan hierbij dan kiezen welke van deze protocollen zij willen gebruiken binnen het curriculum. Bij de andere microcontrollers moet bij wisseling van het draadloze communicatie aspect ook de microcontroller wijzigen.

4. Curriculum Technische Informatica

Voor het implementeren van draadloze communicatie aspecten in het Curriculum van Technische Informatica, is het van belang dat gekeken wordt naar de huidige cursussen die gegeven worden. In dit hoofdstuk zullen eerst de opgaven van Technische Informatica behandeld worden, waarbij gewerkt wordt met embedded systemen. Vervolgens zullen de implementatiemogelijkheden van draadloze communicatie aspecten bij deze cursussen besproken worden. Daarna zal behandeld worden welke impact het voor de cursussen heeft, wanneer de esp32 hierbij geïmplementeerd wordt. Als laatste zal een conclusie gegeven worden hoe de cursussen mogelijk aangepast kunnen worden, zodat de esp32 en de draadloze communicatie aspecten geïmplementeerd kunnen worden.

§4.1 Cursussen

In dit onderzoek wordt gekeken dat studenten met draadloze communicatie aspecten leren werken. Tijdens het curriculum Technische Informatica wordt bij een aantal cursussen de nadruk gelegd op het programmeren op embedded systemen. Voor relevantie van het onderzoek, zullen daarom alleen de volgende cursussen behandeld worden:

Jaar 1	**Jaar 2**	**Jaar 3**
TICT-V1IDP-15	TCTI-V2CPSE1-16	TCTI-VKATP-17
TICT-V1OOPC-15	TCTI-V2THDE-16	
TICT-V1IPASS-15	TCTI-V2MRB-14	
	TCTI-R2D2-17	

Volgend zullen deze cursussen verder uitgewerkt worden, zodat bepaald kan worden waar mogelijkheden zitten voor de implementatie van draadloze communicatie aspecten. De leerdoelen van de genoemde cursussen zullen verder uitgewerkt worden in appendix F.

§4.1.3 TICT-V1IDP-15

In het propedeusejaar wordt een interdisciplinaire (IDP) themaopdracht gegeven, waarbij studenten uit elke ICT richting (Software Netwerk Enginering, Software Information Enginering, Technische Informatica en Business IT en Management) samen moeten werken aan één opdracht. Bij deze themaopdracht kunnen de studenten kiezen uit drie verschillende opdrachten:

- Sportschool
    - Hierbij moeten studenten de sportschool automatiseren.
- Maeslandkering
    - Hierbij moeten studenten een werkend schaalmodel maken van de Maeslandkering.
- Domotica huis
    - Hierbij moeten studenten domotica toepassingen ontwikkelen voor in een huis.

De uitwerking van deze opdrachten worden gemaakt in Python op een Raspberry Pi. Hierbij worden de studenten beoordeeld op het ontwerp van de opdracht, de complexiteit van de bedachte uitwerking, samenwerking van het team en de creativiteit in de uitwerking.

§4.1.1 TICT-V1OOPC-15

Bij de cursus TICT-V1OOPC-15 maken studenten van Technische Informatica voor het eerst kennis met C++ programmeren. In deze cursus worden C++ taal constructies en patronen aangeleerd, hoe zij hun programmeercode moeten documenteren en UML klasse diagrammen moeten lezen en opstellen. Ook krijgen de studenten voor het eerst tijdens de studie te maken met een Arduino Due waarop zij kleine programma's moeten maken, zoals lampjes laten knipperen en een schuifregister[1] aansturen.

§4.1.2 TICT-IPASS-15

Aan het eind van het propedeusejaar wordt de themaopdracht IPASS gegeven. IPASS staat voor Individuele Propedeuse Assignment. Hierbij moet elke student zelf een opdracht bedenken, die hij/zij na goedkeuring van de docent, moet uitvoeren. Bij deze cursus worden de meest uiteenlopende opdrachten uitgewerkt. Van studenten die een spelconsole maken met de aansturing van PlayStation controllers, tot sloten die open gaan wanneer een specifiek deuntje gefloten wordt.

Deze cursus brengt daarnaast een expliciete voorwaarde met zich mee; wanneer deze cursus niet met een voldoende afgerond wordt, krijgt de student een negatief bindend studieadvies en mag hij/zij de rest van de studie niet voortzetten.

§4.1.4 TCTI-V2CPSE1-16

In het eerste blok van het tweede jaar wordt voorgebouwd op de C++ kennis die in het eerste jaar opgedaan is. Bij de cursus TCTI-V2CPSE1-16 krijgen de studenten onder andere te maken met het programmeren met een Real-Time Operating System (RTOS), nieuwe C++ taalconstructies en programmeer patronen. Daarnaast maken de studenten ook kennis met Assembler programmeren op een Cortex-M0 processor, Makefiles en leren zij hoe het bouwproces van een programma in zijn werk gaat.

[1] Een component waarmee het aantal GPIO pinnen van een microcontroller uitgebreid kan worden. Dit component werkt door middel van Binaire getallen die aangeven welke pin van het component stroom moet doorgeven.

§4.1.5 TCTI-V2THDE-16

Aan het eind van het eerste blok van het tweede jaar krijgen de studenten de themaopdracht 'Devices'. Hierbij moet een Lazergame systeem gemaakt worden. Hierbij moeten spelers op elkaar kunnen schieten om punten te verzamelen. Aan het eind van het spel geeft de speler aan de gamemaster door hoeveel punten zij die ronde verkregen hebben, waardoor een winnaar bepaald kan worden. Het systeem werkt door middel van infrarood signalen die het geweer uitzend, waardoor een speler geraakt kan worden en gelijk ziet wie hem neergeschoten heeft. Bij het communiceren naar de gamemaster worden infrarood commando's gestuurd die de hoeveelheid punten van de speler bevatten, waardoor de gamemaster een winnaar kan bepalen.

§4.1.7 TCTI-V2MRB-14

Bij de cursus Meten, Regelen & Besturen (MRB) leren de studenten meer over hardware componenten, de regeling van deze componenten en de manier waarop deze componenten verschillende aspecten kunnen meten. Bij deze cursus worden de volgende onderdelen behandeld:

- Sensoren
- Elektronische Componenten
- Filteren en Signaalprocessing
- Besturingscomponenten
- Motoren
- Voedingen

Naast de lessen dienen de studenten ook een opdracht te realiseren. Hierbij kunnen de studenten kiezen uit twee opdrachten.

Bij de eerste opdracht wordt een muziekinstrument gemaakt. Hierbij moet een pingpongbal door een ventilatorbuis bewegen. Wanneer de student zijn/ haar hand op de buis houd, moet de pingpongbal naar deze positie bewegen. Die positie genereerd een bepaalde toon, waardoor muziek gemaakt kan worden.

De tweede opdracht is het balanceren van een pingpongbal op een plaat. Deze plaat staat op drie servomotoren die aangestuurd kunnen worden. De student moet hierbij zorgen dat, door middel van het aansturen van de servomotoren, de pingpongbal altijd in het midden van de plaat blijft liggen.

§4.1.6 TCTI-R2D2-17

Bij de cursus Roving Robots en Distributed Devices (R2D2) moeten de studenten een winkel beginnen waarbij losse modules te koop zijn. Deze modules zijn kleine subsystemen, waar de studenten in scrumteams aan werken. Hierdoor zou een klant verschillende modules uit kunnen kiezen wat samen tot één systeem samengevoegd kan worden.

Bij deze cursus krijgen de studenten kennis met het scrum[2] werken en de taken die daarbij horen. Hierbij moeten zij sprints in teams uitvoeren met bijbehorende daily standup's[3] en aan het eind van

[2] Scrum is een Agile manier van werken, waarbij multidisciplinaire teams werken aan de realisatie van een product. Hierbij wordt gewerkt in korte sprints die maximaal 4 weken duren.
[3] Een dagelijkse bijeenkomst van de projectleden waarbij de werkzaamheden op elkaar af gestemd worden en een plan gemaakt wordt voor de komende 24 uur. (Agile Scrum Group, z.d.)

de sprint een evaluatie opstellen hoe de sprint verlopen is. Daarnaast wordt ook beoordeeld op het documenteren van de code en de modellen die erbij gemaakt worden. Het is namelijk bij deze cursus de bedoeling dat aan het eind van een sprint de module waaraan gewerkt is, doorgegeven wordt aan een volgend team dat ermee verder gaat.

§4.1.7 TCTI-VKATP-17

De laatste cursus waarbij een Arduino Due gebruikt wordt is TCTI-VKATP-17. Deze cursus bestaat uit een aantal onderdelen. Eerst leren studenten verschillende vormen van testen en het schrijven en implementeren van een testplan. Daarna maken studenten ook kennis met functioneel en aspect-georiënteerd programmeren. De eind opdracht bij deze cursus is het aansturen van een embedded systeem dat geautomatiseerd limonade kan mixen. Hierbij moeten de geleerde testen gedaan worden en de testplannen uitgevoerd worden. Daarna moeten de studenten de aansturing op verschillende manieren tot stand brengen. Dit moet bijvoorbeeld door een via een seriële verbinding, waarbij commando's real-time verstuurd worden en door een met een combinatie van de programmeertalen Python en C++.

§4.2 Implementatie van draadloze communicatie aspecten

In deze paragraaf zullen de implementatiemogelijkheden van draadloze communicatie aspecten in het curriculum van Technische Informatica beschreven worden.

Bij de cursus TICT-V1IDP-15 kan een keuze gemaakt worden uit drie verschillende opdrachten, zie §4.1.3. Bij deze opdrachten zijn meerdere mogelijkheden voor de implementatie van draadloze communicatie aspecten. Zo kan bij de sportschool-opdracht een toegangspoort ontwikkeld worden, waarbij leden van de sportschool een pas dienen te gebruiken om toegang te krijgen. De informatie van deze pas wordt dan over Wifi naar een computer gestuurd. Daarnaast kan bij de Maeslandkering-opdracht de waterstand over Wifi gecommuniceerd worden naar de centrale server. Bij de Domotica opdracht kunnen draadloze communicatie aspecten gebruikt worden voor communicatie tussen apparaten in het huis. Hierdoor maken de studenten kennis met het gebruik van draadloze communicatie aspecten voor hedendaagse toepassingen.

Bij de themaopdracht TICT-IPASS-15 bedenkt elke student zijn eigen opdracht, zie §4.1.2. Hierdoor zijn bij deze opdracht meerdere mogelijkheden om draadloze communicatie aspecten te gebruiken. Wanneer de student een IoT toepassing wil realiseren, kunnen hierbij draadloze communicatie aspecten gebruikt worden.

Bij de themaopdracht TCTI-V2THDE-16 wordt infrarood gebruikt voor de communicatie tussen spelers en naar de gamemaster, zie §4.1.5. Hierbij kunnen draadloze communicatie aspecten toegevoegd worden, zodat bijvoorbeeld de communicatie naar de gamemaster over Wifi verloopt i.p.v. infrarood. Ook kan de gamemaster door middel van draadloze communicatie aspecten naar alle spelers tegelijk commando's kunnen sturen, wanneer bijvoorbeeld het spel afgelopen is.

Bij de cursus TCTI-R2D2-17 kiezen studenten zelf wat voor module ze gaan ontwikkelen, zie §4.1.6. Hierdoor zijn, net zoals bij de cursus TICT-IPASS-15, veel mogelijkheden voor de implementatie van draadloze communicatie aspecten. Wanneer studenten bijvoorbeeld een module willen ontwikkelen

waarbij apparaten met elkaar communiceren of data door moeten sturen naar een server, kunnen draadloze communicatie aspecten gebruikt worden.

Als laatste kan bij de themaopdracht TCTI-VKATP-17 ook draadloze aspecten geïmplementeerd worden. Zoals eerder beschreven moeten studenten op verschillende manieren een geautomatiseerde limonade mixer aansturen. Hierbij kan gekozen dat een van de aansturingen over wifi of bluetooth moet gaan. Dan zou bijvoorbeeld de mixer vanaf een applicatie op je telefoon bedient kunnen worden.

§4.3 Impact bij implementatie esp32

Het implementeren van een nieuwe microcontroller in het curriculum kan gevolgen hebben voor de cursussen. In deze paragraaf zal beschreven worden wat voor impact de implementatie van de esp32 heeft op de huidige cursussen van het curriculum Technische Informatica.

Bij de cursussen TICT-V1IDP-15, TICT-V1OOPC-15, TICT-V1IPASS-15, TCTI-V2MRB-14, TCTI-R2D2-17 en TCTI-VKATP-17 zal het implementeren van de esp32 microcontroller met weinig tot geen problemen kunnen. Met de esp32 kan namelijk hetzelfde gedaan worden als met de Arduino Due die nu gebruikt wordt. Hierdoor zal de implementatie van deze microcontroller weinig tot geen impact hebben.

Het implementeren van de esp32 bij de cursus TCTI-V2CPSE1-16 zal veel impact hebben. De cursus heeft namelijk een aantal elementen in zich die veranderd dienen te worden wanneer deze microcontroller hierin geïntegreerd wordt. Zoals in §4.1.4 beschreven, krijgen de studenten bij deze cursus kennis over Assembler programmeren en het programmeren met een RTOS. De instructies die gebruikt kunnen worden bij Assembler is afhankelijk van de processor. De Assembler instructies van de esp32 verschillen is sommige aspecten van de Arduino Due, waardoor deze in de cursus ook veranderd moeten worden. In de tabel hiernaast staan een aantal voorbeelden van verschillen in instructies. In appendix D zijn alle Assembler instructies te vinden voor de esp32. Volgens Wouter van Ooijen zijn de ARM cortex instructies makkelijker voor studenten te begrijpen, ten opzichte van de instructies van Xtensa. Hierdoor kan gekozen worden om het assembler gedeelte te geven op een microcontroller met een ARM Cortex processor.

Arduino Due		Esp32
MOV	→	MOVI
B	→	CALL0
LDR	→	L16SI

Naast dat de Assembler mogelijke veranderingen door moet maken bij TCTI-V2CPSE1-16, bestaat ook de mogelijkheid dat het RTOS onderdeel moet veranderen. Bij deze cursus wordt namelijk een RTOS library gebruikt dat door de Hogeschool Utrecht gemaakt is (HU-RTOS), maar standaard zit FreeRTOS verweven in de aansturing van de esp32. Hierbij zijn vier verschillende mogelijkheden voor de implementatie van de esp32 in dit onderdeel:

1. Een interface van HU-RTOS maken, wat onderliggend FreeRTOS aanstuurt.
2. In dit onderdeel HU-RTOS vervangen met FreeRTOS.
3. FreeRTOS omschrijven, zodat deze hetzelfde werkt als HU-RTOS.
4. FreeRTOS uit de aansturing van de esp32 halen, zodat HU-RTOS geïmplementeerd kan worden.

De doelstelling van deze cursus is namelijk dat de studenten leren om met een RTOS te werken. Daardoor maakt is het voor de doelstelling niet van belang welke RTOS gebruikt wordt in de cursus. Concluderen kan gesteld worden dat wanneer de esp32 geïmplementeerd word, dit een grote impact zal hebben op de cursus TCTI-V2CPSE1-16.

Als laatste zal het implementeren van de esp32 bij de cursus TCTI-V2THDE-16 ook enige impact hebben. Bij deze themaopdracht wordt namelijk ook gebruik gemaakt van de RTOS library van de Hogeschool Utrecht. Hierbij geld hetzelfde als bij TCTI-CPSE1-16. Wanneer de esp32 bij deze cursus geïmplementeerd wordt, moet de keuze gemaakt op welke manier gebruik gemaakt zal worden van een RTOS.

§4.4 Conclusie

Concluderend zijn er meerdere mogelijkheden om draadloze communicatie aspecten te implementeren in de cursussen van Technische Informatica. In de onderstaande tabel zal de impact op de cursussen uitgewerkt worden.

Cursus	Impact Assembler	Impact RTOS	Draadloze communicatie mogelijkheden
TICT-V1IDP-15	Geen	Geen	Veel
TICT-V1OOPC-15	Geen	Geen	Geen
TICT-V1IPASS-15	Weinig	Weinig	Veel
TCTI-V2CPSE1-16	Hoog	Hoog	Weinig
TCTI-V2THDE-16	Geen	Hoog	Veel
TCTI-V2MRB-14	Geen	Geen	Geen
TCTI-R2D2-17	Weinig	Weinig	Veel
TCTI-VKATP-17	Geen	Geen	Veel

Tabel 4.4 – Impact op cursussen

Bij de cursussen TICT-IPASS-15 en TCTI-R2D2, moeten de studenten zelf een opdracht bedenken om te realiseren, dit maakt een vaste implementatie van draadloze communicatie aspecten in de cursus lastig. Wouter van Ooijen is docent bij de cursussen TICT-V1OOPC-15, TICT-IPASS-15, TCTI-V2CPSE1-16, TCTI-V2THDE-16 en TCTI-VKATP-17. Aangezien hij dicht bij de cursussen staat en de maker is van de libraries en tools, is met hem overlegd bij welke cursus het beste gestart kan worden met de implementatie van draadloze communicatie aspecten. Hij vond de cursus TICT-V2THDE-16 de beste optie hiervoor, aangezien deze themaopdracht veel mogelijkheden bied voor de implementatie van deze aspecten. Hierdoor wordt geadviseerd om te starten met de implementatie bij deze cursus.

5. Integratie Libraries en tools

In de Organisatorische Context is de Hardware Library (HWLIB), Real-Time Operating System (RTOS) en Bare Metal Programming Tool Kit (BMPTK) make tool besproken, zie §1.1.1. Dit hoofdstuk zal dieper op deze libraries en tools in gaan en zal het integreren van de van de esp32 in deze libraries en tools beschreven worden. In de eerste paragraaf zal ingegaan worden op de BMPTK make tool, daarna zal HWLIB behandeld worden en als laatste de RTOS library.

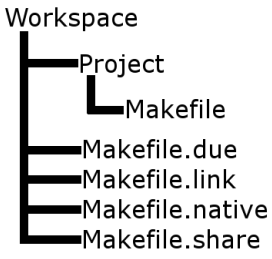
§5.1 BMPTK

Zoals eerder vermeld is de BMPTK make door gemaakt door Wouter van Ooijen voor de Hogeschool Utrecht. Met deze, op GNU make gebaseerde, tool kunnen programma's gecompileerd en gebouwd worden. Ook zorgt deze tool dat de geschreven code op de hardware terecht komt waardoor de microcontroller de code kan uitvoeren. BMPTK is gemaakt voor kleine microcontrollers die GCC, C, C++ of assembler gebruiken en kan voor Windows en Linux gebruikt worden. Aangezien deze tool binnen de curricula van Technische Informatica gebruikt wordt, zal de aansturing van de esp32 hierdoor in deze tool geïntegreerd moeten worden.

Volgend zal in eerst de werking van BMPTK besproken en waar de mogelijkheden zitten voor het integreren van een nieuwe microcontroller. Vervolgens zal de werking van de ESP-IDF beschreven worden, aangezien dit platform origineel de aansturing van de esp32 verzorgt. Als laatste zal de integratie van de esp32 in BMPTK aan bot komen.

§5.1.1 Werking BMPTK

BMPTK werkt door middel van 'Makefiles'. Deze bestanden zitten op meerdere plaatsen verspreid om een programma te kunnen bouwen, zie figuur 6.2. In de BMPTK tool zelf zit een bestand 'Makefile.inc'. Dit bestand verzorgt het bouwproces van een 'Target', aangezien dit verschilt per microcontroller. De term Target staat voor de microcontroller waarvoor het programma gebouwd moet worden. De keuze van de Target en het zetten van andere essentiële waarden gebeurd in de Makefiles in de projectfolder en de workspace folder waar de verschillende projecten in zitten. Deze waarden bevatten bijvoorbeeld de naam van het project, welke bestanden meegenomen moeten worden bij het bouwproces en op welke seriële poort de microcontroller aangesloten zit, zie figuur 6.2.



Figuur 6.2 - Workspace Makefile structuur

```
# settings for esp32 projects
TARGET            ?= esp_devkit_v1
SERIAL_PORT       ?= COM5
CONSOLE_BAUDRATE  ?= 115200
```

Figuur 6.2 – Voorbeeld waarden in Makefile.due

Het bestand Makefile.inc in de bmptk folder is de kern van de BMPTK make tool. Dit bestand verzorgt namelijk het bouwen van een programma voor de verschillende Targets. In dit bestand wordt gekeken voor welke Target een programma gebouwd moet worden en zorgt voor de bijbehorende 'Flags' hiervoor. Deze Flags bestaan onder andere uit commando's die meegegeven dienen te worden aan de compiler of toolchain. Ook staan hierin extra bestanden en hun locaties die nodig zijn om de code te laten werken op de microcontroller. Zodra deze Flags correct gevult zijn, kan de code worden gecompileerd, gebouwd en daarna gedraaid worden op de microcontroller.

§5.1.2 Aansturing en integratie

Om de esp32 in BMPTK te kunnen integreren, zal eerst gekeken moeten worden naar de aansturing van deze microcontroller. Espressif, de maker van de esp32, heeft voor de aansturing van de microcontroller de Espressif IoT Development Framework (ESP-IDF) gemaakt. Aangezien de waarden die benodigd zijn voor de aansturing van de esp32 hierin zitten, zal dit raamwerk hieronder verder toegelicht worden. Daarna zal de daadwerkelijke integratie besproken worden.

§5.1.2.1 ESP-IDF

De Espressif IoT Development Framework (ESP-IDF) is de het raamwerk voor de aansturing van de esp32. De ESP-IDF bevat de benodigde onderdelen om de esp32 op verschillende manier aan te kunnen sturen. Zo zou het programma gebouwd kunnen worden met CMake[4], GNU Make of Python. Aangezien BMPTK op GNU Make gebaseerd is, zal in dit onderzoek voornamelijk naar dit onderdeel gekeken worden.

De ESP-IDF heeft drie verschillende onderdelen in zich:

- Components
- Make
- Tools

Hieronder zullen deze onderdelen verder toegelicht worden.

§5.1.2.1.1 Components

Alle functionaliteiten die gebruikt kunnen worden voor de esp32 worden componenten genoemd. Deze componenten zijn verschillende C-libraries die gebruikt kunnen worden bij het programmeren op de esp32. De belangrijkste basiscomponenten die gebruikt dienen te worden bij het bouwen van een applicatie voor de esp32 zijn:

- Bootloader
- Bootloader_support
- Esp32
- Freertos

Bootloader en bootloadersupport zijn twee componenten die samen werken om het programma op de een plaats te geven in het memory van de esp32 en zorgen daarmee ook dat de esp32 op kan starten en de applicatie kan vinden/ uitvoeren.

[4] Open-source, cross-platform bundel van tools, waarmee software gebouwd, getest en verpakt kan worden. (CMake, z.d.)

Het component esp32 bevast de core-functionaliteiten, die benodigd zijn voor het laten werken van het programma. Zo bevat deze de linkerscripts die de verschillende stukken programmeercode in kunnen delen in het memory, maar ook C-bestanden die zorgen voor het starten van de CPU en dat basisfunctionaliteiten gebruikt kunnen worden.

Freertos is een component dat ook standard gebruikt moet worden. Deze wordt gebruikt, aangezien de esp32 een dual core microprocessor is en met Freertos threads aangemaakt kunnen worden. Deze threads zorgen dat beide cores van de esp32 benut kunnen worden voor de programmeur.

Alle andere componenten/ functionaliteiten die gebruikt kunnen worden op de esp32 zullen verder uitgewerkt worden in appendix C.

§5.1.2.1.2 Make

Make bevat de Makefiles waarmee de benodigde bestanden verzameld, gecompileerd en gelinkt worden en het gehele programma gebouwd wordt.

Bij het verzamelen van de bestanden wordt gekeken naar de componenten die nodig zijn voor het draaien van de applicatie. Hierbij worden altijd de eerdergenoemde componenten meegenomen. Wanneer bijvoorbeeld het ethernet protocol ‘Modbus’ gebruikt wordt in een applicatie, zal dit component ook mee gecompileerd moeten worden.
Wanneer in beeld is gebracht welke componenten benodigd zijn voor het bouwen van de applicatie, wordt gekeken waar de source files (.c / .asm / .S/ .cpp/ etc.) en headerfiles (.h / .hpp) staan om deze om te zetten naar objecten.
Als dit allemaal zonder problemen verloopt kunnen de objecten daarna aan elkaar gelinkt worden door middel van de linker scripts die in de ESP-IDF zitten. Deze zorgen dat de gevormde objecten een plek in het geheugen krijgen. Wanneer voor alle objecten een plaats in het geheugen is toegekend kan dit geheel samengevoegd worden tot één applicatie dat op de esp32 geflashed kan worden.

§5.1.2.1.3 Tools

Tools bevat een groot aantal van onderdelen dat gebruikt kan worden voor de esp32, maar ook voor onderdelen in de ESP-IDF. Zo staan hier bestanden die het mogelijk maken om een applicatie met CMake te bouwen, default configuraties voor het bouwen van een applicatie aangemaakt kunnen worden, maar ook tools waarmee elementen van de esp32/ ESP-IDF getest kunnen worden of een programma op de esp32 geflashed kan worden.

Voor het flashen van een programma moet de esptool of idf tool aangeroepen worden. Hieraan worden configuraties meegegeven en op welke plaats de applicatie in het geheugen moet komen te staan. Wanneer het programma op de esp32 geflashed is, zal deze daarna direct uitgevoerd worden.

§5.2 HWLIB

De Hardware Library (HWLIB), gemaakt is door Wouter van Ooijen, is een C++ library waarmee, object georiënteerd, hardware aangestuurd kan worden. Volgend zal eerst de werking van deze library beschreven worden, daarna de stappen die ondernomen moeten worden voor het toevoegen van een Wifi en/ of Bluetooth library.

§5.2.1 Werking HWLIB

De HWLIB library heeft verschillende functionaliteiten om de aansturing van hardware te realiseren. Ook bevat deze library functionaliteiten waarmee bepaalde protocollen aangestuurd kunnen worden. De volgende functionaliteiten worden op dit moment door HWLIB ondersteund:

- Digitaal analoog conversie
- Aansturing van OLED scherm
- Grafische elementen voor OLED schermen
- I^2C aansturing
- SPI aansturing
- Aansturing voor shift-registers
- Aansturing voor matrix keypads
- Aansturing en uitlezen van GPIO pinnen

De library is zeer modulair, waardoor deze functionaliteiten bij een groot aantal microcontrollers gebruikt kan worden.

§5.2.2 Aansturing en integratie

Zoals hierboven beschreven bevat de HWLIB library verschillende functionaliteiten. Alleen met de functionaliteiten waarbij GPIO pinnen gebruikt worden, zal gezorgd moeten worden dat de juiste registers ingesteld worden voor de aansturing. Hiervoor heeft de HWLIB library voor elke microcontroller een eigen header file, waarbij deze benodigde waarden ingesteld worden. In het bestand ‘hwlib.hpp’ wordt gekeken welke microcontroller aangesloten is, en voor welke het de register waarden moet instellen. De informatie over welke microcontroller aangesloten zit, wordt in de Makefile gespecificeerd onder de waarde “Target”, zie §5.1.1.

Wanneer een Wifi en/ of bluetooth library gemaakt wordt, zal deze toegevoegd moeten worden in de library folder in HWLIB. De header file van de Wifi en/ of bluetooth library moet vervolgens toegevoegd worden in de header file “hwlib-all.hpp” en in het bestand “Makefile.inc”. Daarna zal deze gebruikt kunnen worden door de student.

§5.3 RTOS

Een Real-Time Operaring System (RTOS) is een besturingssysteem, waarbij een gebruiker op aangeduide tijdstippen bepaalde taken kan uitvoeren. De taken die uitgevoerd kunnen worden hebben ieder een eigen prioriteit, wat het systeem zal volgen bij het uitvoeren van de taken. Volgend zal eerst de werking van RTOS beschreven worden. Daarna zal de integratie van de RTOS library van de Hogeschool Utrecht in de aansturing van de esp32 beschreven worden.

§5.3.1 Werking RTOS

De RTOS library van de Hogeschool Utrecht is gemaakt door Wouter van Ooijen en Marten Wensink. Deze library een aantal functionaliteiten die gebruikt kunnen worden door de programmeur. Deze functionaliteiten zullen in de onderstaande tabel uitgewerkt worden, samen met een korte beschrijving.

Functionaliteit	**Beschrijving**
Mutex	Een Mutex is een taak dat aangemaakt kan worden binnen het besturingssysteem. Deze kan “tegelijk” uitgevoerd worden met andere taken.
Eventflag	Een Eventflag is een vlag die ingesteld kan worden. De code zal delen alleen uitvoeren, wanneer de bijbehorende vlag ingesteld wordt.
Channel	Een Channel is een lijst met data elementen. Een Channel kan alleen gelezen worden door de taak waarin deze is aangemaakt. Andere taken kunnen wel data elementen aan deze lijst toevoegen. De mogelijkheid om uit de Channel te lezen is geblokkeerd als de lijst van data elementen leeg is.
Timer	Een Timer is een tijdselement dat aangemaakt kan worden. Deze zal een bericht geven als de, door de programmeur ingestelde, tijd verstreken is.

Tabel 5.3.1 – RTOS functionaliteiten

Functionaliteit	Beschrijving
Clock	Een Clock is, net als de Timer, een tijdselement dat aangemaakt kan worden. Het verschil met de Timer is dat de Clock voor altijd draait en een Timer aangezet moet worden. Een Clock wordt gebruikt wanneer iets uitgevoerd dient te worden na een bepaalde tijd.
Pool	Een Pool lijkt in functionaliteit op een Channel, met het verschil dat wanneer een data element in de pool gezet wordt, het lezen van de pool geblokkeerd wordt. Dit geld ook als een data element gelezen wordt uit de pool.

Tabel 5.3.1 – RTOS functionaliteiten (vervolg)

§5.3.2 Aansturing en integratie

Tijdens de realisatie van dit onderzoek is niet gekeken naar de aansturing en integratie van de RTOS library van de Hogeschool Utrecht in de aansturing van de esp32. Echter zou deze RTOS library direct moeten werken, zodat deze opgenomen wordt bij het bouwproces en aangeroepen wordt in de code van het programma.
Zoals in §5.1.2.1.1 beschreven is, maakt FreeRTOS, wat een gratis RTOS is die veel gebruikt wordt door programmeurs, deel uit van de componenten van de ESP-IDF. Bij de aansturing van de esp32 zorgt verzorgt dit de aansturing van de verschillende processorkernen van de esp32. Ook verzorgt het extra functionaliteiten die diep in de aansturing van de microcontroller gebruikt worden. Tijdens dit onderzoek is het daarom niet gelukt de RTOS library van de Hogeschool Utrecht op te nemen in de aansturing van de esp32. Wanneer deze microcontroller in de cursussen geïmplementeerd wordt, waarbij een RTOS gebruikt moet worden, zal gekozen moeten worden uit de oplossingen die in §4.3 beschreven staan.

§5.4 Realisatie aansturing esp32

Nadat het onderzoek gedaan was naar de ESP-IDF konden de benodigde onderdelen in BMPTK gerealiseerd worden. Deze paragraaf zal de onderdelen bespreken die toegevoegd dienen te worden, samen met voorbeelden van de code die geschreven is. In de voorbeelden die gegeven worden zijn soms stukken code weg gelaten voor leesbaarheid van het verslag. De volledige code is te vinden op de GitHub repository van dit onderzoek[5].

§5.4.1 Makefile.inc

In het bestand Makefile.inc in de hoofdmap van BMPTK zijn een aantal onderdelen toegevoegd. Als eerste is een nieuwe target gemaakt voor de module die gebruikt is voor dit onderzoek. Daarna zijn verschillende functies gemaakt, waarbij de benodigde vlaggen gevuld worden en informatie uit de ESP-IDF gehaald wordt.

De eerste stap in de realisatie was het toevoegen van een nieuwe target. De module die gebruikt is bij dit onderzoek is de ESP32 DevKit v1, daarom heeft de target de naam “esp_devkit_v1” gekregen, zie voorbeeld 1. Op dit programmeerbordje zit de “esp_wroom_32” chip die een klok snelheid (XTAL) heeft van 240 MHz.

```
ifeq ($(TARGET),esp_devkit_v1)
  CHIP     ?= esp_wroom_32
  # Clock speed: 240 MHz
  XTAL     ?= 240000
endif
```

voorbeeld 1 - Target selectie

Daarna wordt in BMPTK gekeken voor welke chip de vlaggen gevuld dienen te worden. Om de correcte chip te vinden wordt een ‘ifeq’ functie gebruikt, zie voorbeeld 2. Hierbinnen wordt de “Xtensa_LX6” functie aangeroepen (Xtensa_LX6 is de processor van de esp32) en de ROM, RAM en STACK grootte gedefinieerd.

```
ifeq ($(CHIP),esp_wroom_32)
  $(eval $(Xtensa_LX6))
  ROM_SIZE      ?= 448k
  RAM_SIZE      ?= 520k
  STACK_SIZE    ?= 200000
  DEFINES       +=
endif
```

voorbeeld 2 – Chip selectie

Binnen de Xtensa_LX6 functie worden meerdere vlaggen gezet, zie voorbeeld 3. Eerst worden een aantal waarden gedefinieerd over de toolchain van de esp32, zoals de naam van de toolchain en de locatie hiervan. Vervolgens wordt de functie “Xtensa_LX” aangeroepen, de Xtensa LX6 maakt namelijk deel uit van de Xtensa LX serie. De basiswaarden zijn voor deze gelijk, waardoor dit zijn eigen functie gekregen heeft. Daarna worden een aantal vlaggen gevuld, waarbij onder andere het beginadres van de ROM en RAM gedefinieerd wordt. Als laatste worden hier bepaalde parameters gespecificeerd die meegegeven dienen te worden aan de compiler en commando’s die uitgevoerd dienen te worden voor het checken van de seriële poort of het runnen van het programma.

```
define Xtensa_LX6
  …….

  $(eval $(Xtensa_LX))

  ROM_START     ?= 0x40000000
  RAM_START     ?= 0x40070000
  PORT_CHECK    ?= $(CHECK_PORT) \
                          $(SERIAL_PORT)
  RESULTS           += $(BIN)
  …….

  RUN := …
  DEFINES       += -DDONT_USE_CMSIS_INIT
  DEFINES       += -DBMPTK_INCLUDE_CHIP
  RUN_TERMINAL  ?= …
endef
```

voorbeeld 3 – Xtensa_LX6 Define

[5] https://github.com/renedekluis/onderzoeksemester_18-19

De Xtensa_LX functie, zie codevoorbeeld 4, doet eigenlijk vrij weinig. Binnen deze functie wordt de Xtensa functie aangeroepen en er kunnen extra argumenten meegegeven worden voor het builden van het programma voor de Xtensa LX serie.

```
define Xtensa_LX
  $(eval $(Xtensa))
   ARCH_FLAGS :=
endef
```

voorbeeld 4 – Xtensa Define

Binnen de Xtensa functie, zie voorbeeld 5, worden de meest belangrijke vlaggen gevuld. Deze vlaggen verzorgen namelijk de basis aansturing van alle Xtensa processoren. Binnen deze functie wordt eerst de “Embedded” functie aangeroepen waarbij de basis functionaliteiten gezet worden voor elk embedded systeem. Vervolgens worden onder andere de benodigde source files, header files, linker script files en libraries gedefinieerd. Voor het vinden van de correcte source en header files zijn twee verschillende functies aangemaakt:

- “GenerateComponentInfo”
- “GenerateComponentSources”

```
define Xtensa
  $(eval $(Embedded))
……
  ESP_CORE        := $(BMPTK)/targets/esp/esp32
  INCLUDES     += -I$(PROJECT_PATH)/build/include
……
  $(foreach component, \
     $(COMPONENTS), \
     $(eval $(call GenerateComponentInfo,$(component))))
  $(foreach component, \
     $(SOURCE_COMPONENTS), \
     $(eval $(call GenerateComponentSources,$(component))))

  LINKERSCRIPT    := esp32_out.ld
…….
  LDFLAGS            += -L$(ESP_CORE)/lib/
…….

  OBJ          += bmptk_startup_esp32.o
  CPU          ?= XtensaLX6
…….
endef
```

voorbeeld 5 – Xtensa functie

In de functie GenerateComponentInfo, zie voorbeeld 6, wordt informatie vergaard over de header files van verschillende benodigde componenten uit de ESP-IDF. Deze functie wordt aangeroepen voor elk component dat benodigd is voor het bouwen van het programma. Vervolgens wordt gekeken in welke mappen van het component de headerfiles staan die benodigd zijn en wordt het volledige pad naar die map opgeslagen in de vlag “INCLUDES”. Daarna wordt gekeken of bepaalde compileer commando’s benodigd zijn en zo nodig toegevoegd aan de vlag “LDFLAGS”.

```
define GenerateComponentInfo
  COMPONENT_ADD_INCLUDEDIRS :=
  COMPONENT_SRCDIRS :=
   $(eval COMPONENT_PATH := …)
   COMPONENT_PATHS += $(COMPONENT_PATH)

   $(eval include $(COMPONENT_PATH)/component.mk)

   $(eval COMPONENT_INCLUDE_DIRS := )
   $(foreach dir, $(COMPONENT_ADD_INCLUDEDIRS), \
      $(eval COMPONENT_INCLUDE_DIRS += \
            $(subst /.,,$(COMPONENT_PATH)/$(dir)))\
      $(eval INCLUDES += \
            -I$(subst /.,,$(COMPONENT_PATH)/$(dir)))\
   )
   $(eval LDFLAGS += $(COMPONENT_CFLAGS))
   $(eval CLEAN += SOURCES)
endef
```

voorbeeld 6 – GenerateComponentInfo functie

De functie GenerateComponentSources, zie voorbeeld 7, zorgt voor de verzameling van informatie over de verschillende source files die benodigd zijn voor het component. Hierbij wordt ook weer gekeken naar de locatie van de mappen waar de source files in staan en wat het volledige pad hiervan is. Deze worden dan ook toegevoegd aan de “INCLUDES” vlag. Daarnaast worden de paden ook toegevoegd aan de vlag “SEARCH”. Deze vlag bevat namelijk de paden waar de compiler moet zoeken voor de locatie van verschillende source files. Als laatste worden de benodigde source files toegevoegd aan de vlag “SOURCES”, deze vlag bevat de namen van de verschillende source files die omgezet dienen te worden tot objecten.

```
define GenerateComponentSources
…….
  $(foreach dir, $(COMPONENT_ADD_INCLUDEDIRS), \
    $(eval COMPONENT_INCLUDE_DIRS += \
        $(subst /.,,$(COMPONENT_PATH)/$(dir))) \
    $(eval INCLUDES += \
        -I$(subst /.,,$(COMPONENT_PATH)/$(dir))) \
  )
  $(eval COMPONENT_SOURCE_DIRS :=)
  $(foreach dir, $(COMPONENT_SRCDIRS), \
    $(eval COMPONENT_SOURCE_DIRS += \
        $(subst /.,,$(COMPONENT_PATH)/$(dir)))\
    $(eval SEARCH += \
        -I$(subst /.,,$(COMPONENT_PATH)/$(dir)))\
  )
  $(foreach src, $(COMPONENT_SOURCES), \
    $(eval SOURCES += $(src))\
  )
endef
```

voorbeeld 7 – GenerateComponentSources *functie*

§5.4.2 Bmptk.mk

In het originele build systeem van de ESP-IDF wordt het bestand ‘project.mk’, dat in de make folder van het ESP-IDF staat, vanaf de Makefile in de workspace aangeroepen. Alleen heeft project.mk een aantal problemen. Zo kan het alleen op een Linux omgeving aangeroepen worden of vanuit een Linux gebaseerd command promt op Windows. Dit komt aangezien in project.mk Linux commando’s gebruikt worden voor bijvoorbeeld het aanmaken van mappen of kijken of bestanden al bestaan. Daarnaast verbied project.mk dat ‘:’ in padnamen voorkomen, aangezien dit problemen geeft met de verschillende targets die in de Makefiles aangeroepen worden. Om deze redenen is gekozen om dit bestand te herschrijven tot “bmptk.mk”. Hierbij zijn de Linux commando’s voor o.a. het aanmaken van mappen in de build folder van het project omgevormd tot een python script dat uitgevoerd wordt. Ook is in bmptk.mk het gebruikt van ‘:’ toegestaan in padnamen, waardoor het vanuit het Windows command promt uitgevoerd kan worden.

Naast deze essentiële omvormingen zijn ook nog twee extra functionaliteiten toegevoegd, voor het vinden van fouten in de code. Wanneer de target “see-defined-variables” als make commando aangeroepen wordt, worden alle gedefinieerde vlaggen in bmptk.mk uitgeprint op het scherm. En met het commando “see-var *<variabel naam>”* kan de waarde die in deze vlag zit bekeken worden. Hierdoor kunnen problemen zoals vlaggen die per ongeluk verkeerd gevuld worden snel gevonden worden door de programmeur.

6. Conclusie & aanbeveling

In dit hoofdstuk zullen de conclusie en aanbevelingen besproken worden. In de eerste paragraaf zal de conclusie van het onderzoek beschreven worden, daarna volgen enkele aanbevelingen die op basis van het onderzoek gedaan zijn.

§6.1 Conclusie

De doelstelling van dit onderzoek was om draadloze communicatie aspecten op te nemen binnen het curriculum Technische Informatica van HBO-ICT aan de Hogeschool Utrecht. Zij wilden deze aspecten in het curriculum opnemen, aangezien Internet of Things (IoT) toepassingen steeds meer in opkomst zijn. Daarom is aan het begin van dit document de volgende hoofdvraag opgesteld:

"Hoe kan een nieuwe microcontroller met geïntegreerde wifi en/of bluetooth in de huidige libraries en tools van de Hogeschool Utrecht opgenomen worden, zodat draadloze communicatie aspecten toegevoegd kunnen worden in de opgaven van Technische Informatica?"

Het opnemen van een nieuwe microcontroller kan gebeuren door de aansturing te integreren in de BMPTK make tool. Een Wifi en/ of Bluetooth library op te nemen in HWLIB en de RTOS library op te nemen in de aansturing van de esp32.

Voor het beantwoorden van deze onderzoeksvraag is onderzoek gedaan naar verschillende onderdelen. Eerst is onderzocht welke microcontroller, met geïntegreerde Wifi en/ of Bluetooth module, de Arduino Due kon vervangen. Hierbij werd de esp32 van Espressif gekozen, aangezien deze microcontroller als enige beide modules geïntegreerd heeft en voldeed aan de gestelde criteria.
Daarna is gekeken naar de verschillende cursussen van het curriculum Technische Informatica. Wanneer de Arduino Due vervangen wordt met de esp32 bleek dit een hoge impact te hebben op de cursus TCTI-V2CPSE1-16. Voor de rest van de cursussen zal de implementatie van de esp32 weinig impact ervaren. Daarnaast bleek de cursus TCTI-V2THDE-16 de beste cursus om te starten met de implementatie van draadloze communicatie aspecten. Deze cursus biedt meerdere mogelijkheden voor deze implementatie en zal weinig impact ervaren wanneer de Arduino Due met de esp32 vervangen wordt.
Als laatste is gekeken naar de libraries en tools van de Hogeschool Utrecht. Hieruit bleek dat het mogelijk is om de aansturing van de esp32 in BMPTK te integreren en een Wifi en/ of Bluetooth library in HWLIB op te nemen. Alleen het integreren van de RTOS library van de Hogeschool Utrecht in de esp32 bracht problemen met zich mee. Het bleek dat de basis aansturing van de esp32 gebruik maakt van FreeRTOS. Tijdens dit onderzoek is het niet gelukt om dit te kunnen vervangen met de RTOS library van de Hogeschool Utrecht. Wanneer de esp32 geïmplementeerd wordt in de cursussen met dit onderdeel, zal eerst een vervolgonderzoek gedaan moeten worden, welke RTOS library gebruikt wordt en wat eventueel daarvoor moet veranderen.

§6.2 Aanbeveling

In deze paragraaf worden aanbevelingen gegeven voor vervolg onderzoeken of veranderingen van onderdelen in het curriculum.

Ten eerste wordt geadviseerd om een vervolgonderzoek te starten naar de aansturing van de esp32, gericht op het RTOS onderdeel. Dit advies wordt gegeven, aangezien in dit onderzoek het niet is gelukt om de aansturing van de esp32 te realiseren zonder FreeRTOS. Hierdoor is het gebruik van de RTOS library van de Hogeschool Utrecht op de esp32 nog niet mogelijk. Wanneer de esp32 geïmplementeerd wordt bij de cursussen waarbij RTOS uitgelegd wordt, zal een keuze gemaakt moeten worden uit de volgende mogelijkheden:

1. Een interface van de RTOS library van de Hogeschool Utrecht maken, wat onderliggend FreeRTOS aanstuurt.
2. In dit onderdeel van de cursus de RTOS library van de Hogeschool Utrecht vervangen met FreeRTOS.
3. FreeRTOS omschrijven, zodat deze hetzelfde werkt als de RTOS library van de Hogeschool Utrecht.
4. FreeRTOS uit de aansturing van de esp32 halen, zodat de RTOS library van de Hogeschool Utrecht geïmplementeerd kan worden.

Daarnaast wordt huidig over de gehele duur van de opleiding de Arduino Due gebruikt. Om dit uniform te houden, kan geadviseerd worden om deze te vervangen met de esp32 op alle plaatsen waar de Arduino Due huidig gebruikt wordt. Dit zorgt ook dat studenten niet tijdens de duur van hun studie met verschillende microcontrollers hoeven te werken, die in hun functionaliteit net iets van elkaar verschillen. Echter wordt wel geadviseerd om de Arduino Due, of een andere microcontroller met een ARM Cortex processor te behouden in de cursus waarbij Assembler gegeven wordt. De Assembler instructies van Xtensa procesoren zijn namelijk lastiger, dan van de ARM Cortex processoren. Hierdoor het voor de student makkelijker om bij dit onderdeel de esp32 niet te gebruiken.

7. Evaluatie

In dit hoofdstuk zal het verloop van het onderzoek geëvalueerd worden. Hierbij zullen eerst de knelpunten beschreven worden, waar tegenaan gelopen werd. Daarna zullen de leerpunten van het onderzoek en afgesloten worden met een conclusie.

§7.1 Knelpunten

Tijdens de realisatie van dit onderzoek moest veel gewerkt worden met Makefiles. De taal die hierbij gebruikt wordt is in basis uitgelegd in de cursus TCTI-V2CPSE1-16. Alleen bevatte de code die bij deze cursus gegeven werd niet de complexiteit dat benodigd was bij dit onderzoek. In de Makefiles van de ESP-IDF werden veel taalconstructies gebruikt die niet gegeven worden bij de cursus, waardoor het lastig was om goed door te krijgen wat er precies gebeurde. Zo maakte zij bijvoorbeeld gebruik van 'Wildcard' functies of functies om tekst in variables aan te passen. Om deze elementen door te krijgen is veel onderzoek gedaan naar de verschillende functionaliteiten van GNU Make.

Daarnaast was het lastig om door te krijgen welke componenten uit de ESP-IDF meegenomen moesten worden tijdens het bouwproces van een applicatie. Dit heeft veel tijd van het onderzoek gekost, voordat de applicatie gebouwd kon worden. De rede waarom het Proof of Concept nog niet werkt is dat naast de applicatie van de student, nog twee interne applicaties gebouwd moeten worden uit de ESP-IDF en meegegeven dienen te worden bij de aansturing van de esp32. Om te zorgen dat deze onderdelen gebouwd konden worden was helaas te weinig tijd, waardoor het Proof of Concept mislukt is.

Als laatste kwamen tijdens de realisatie zeer vreemde fouten voor tijdens het integreren van de aansturing in de BMPTK make tool. Een voorbeeld hiervan is dat de source files correct omgezet werden naar objecten. Maar bij het linken van de objecten, zocht de linker naar objecten, waarbij een random letter verdwenen was. Zo werd bijvoorbeeld de source file "dport_access.c", het object "dport_access.o" gemaakt. Maar bij het linken werd gezocht naar "dport_aces.o", waardoor deze niet gevonden werd. Na dagen zoeken en navragen, bleek dat te veel paden naar sourcefiles in het bouw commando stond dat uitgevoerd werd. Wanneer deze namelijk verminderd werden, verdwenen de letter niet meer. Helaas hebben dit soort problemen voor veel vertraging in het onderzoek gezorgd. Andere problemen die voor kwamen tijdens de realisatie van het project staan verder uitgewerkt in appendix F.

§7.2 Leerpunten

Zoals eerder vermeld, is in dit onderzoek veel gewerkt met Makefiles. De taalconstructies die gebruikt zijn in de ESP-IDF waren lastig te begrijpen aangezien deze niet in de cursussen voor kwamen. Tijdens de realisatie van die project is daarom veel onderzoek gedaan naar functionaliteiten van GNU Make (GNU, z.d., p. Functions for String Substitution and Analysis) (GNU, z.d., p. The call Function) (GNU, z.d., p. The Function wildcard). Door dit onderzoek is veel kennis verkregen over Makefiles, wat het maken van projecten in de toekomst makkelijker zal maken.

Daarnaast was maken van de aansturing voor een microcontroller ook zeer interessant. Tijdens de cursussen gebruiken studenten namelijk alleen microcontrollers waarbij de aansturing al gerealiseerd is. Hierdoor is veel geleerd over de werking van de hardware van microcontrollers en welke stappen bij de realisatie van de aansturing ondernomen dienen te worden.

§7.3 Conclusie

Het is teleurstellend dat het Proof of Concept niet gerealiseerd kon worden binnen het gegeven tijdsbestek. Het liefst was aan het onderzoek een werkend product dat direct gebruikt toepasbaar was in de cursussen van Technische Informatica. Bij het bedenken van het onderzoek was de moeilijkheidsgraad onderschat. Mijn kennis over Makefiles was namelijk niet voldoende om het Proof of Concept binnen het gegeven tijdsbestek te realiseren. Wanneer ik hier meer kennis over had was de realisatie waarschijnlijk wel gelukt, aangezien tijdens de realisatie van het Proof of Concept veel tijd verloren ging aan het doorkrijgen van bepaalde functionaliteiten in Makefiles.

Maar zoals eerder in dit hoofdstuk vermeld, is tijdens de realisatie van dit onderzoek veel geleerd. Er is veel kennis verkregen over de werking van Makefiles en de functionaliteiten die gebruikt kunnen worden, waardoor hier makkelijker mee gewerkt zal kunnen worden in toekomstige projecten. Het project is helaas niet geslaagd, maar ik ben tevreden over de kennis die verkregen is door de realisatie van dit onderzoek.

8. Bibliografie

Agile Scrum Group. (z.d.). *De Daily standup meeting: uitleg en tips*. Opgehaald van Scrum Guide: https://scrumguide.nl/daily-standup-meeting/

Anthon. (2015, juni 12). *Invoke python script through make command*. Opgehaald van Unix & Linux: https://unix.stackexchange.com/questions/209173/invoke-python-script-through-make-command

Archlinux. (2018, November 26). *WPA supplicant*. Opgehaald van Archlinux: https://wiki.archlinux.org/index.php/WPA_supplicant

ARM MBED. (z.d.). *ARM MBED*. Opgehaald van ARM MBED: https://tls.mbed.org/

Botland. (z.d.). *Bluno Nano and BLE Bluetooth 4.0 - compatible with Arduino*. Opgehaald van Botland: https://botland.com.pl/en/plytki-zgodne-z-arduino-dfrobot/2998-bluno-nano-and-ble-bluetooth-40-compatible-with-arduino.html

Cee, Y. (sd). *ESP-WROOM-32 ESP32*. Your Cee. Opgehaald van https://nl.aliexpress.com/item/ESP-WROOM-32-ESP32-Bluetooth-and-WIFI-Dual-Core-CPU-with-Low-Power-Consumption-MCU/32793415575.html

Chan, E. (2018, Oktober 14). *FatFs - Generic FAT Filesystem Module*. Opgehaald van elm-chan: http://elm-chan.org/fsw/ff/00index_e.html

Cheshire, S., & Krochmal, M. (2013). *Internet Engineering Task Force (IETF).* Apple Inc.: Februari. Opgehaald van https://tools.ietf.org/html/rfc6762

CMake. (z.d.). *CMake*. Opgehaald van CMake: https://cmake.org/

Cooper, C. (1999, September 1). *Using Expat*. Opgehaald van XML: https://www.xml.com/pub/a/1999/09/expat/index.html

DealExtreme. (z.d.). *NL6621-Y1 Remote Control Serial Port naar Wi-Fi SDK Module voor Anduino*. Opgehaald van DealExtreme: https://www.dx.com/nl/p/nl6621-y1-remote-control-serial-port-to-wi-fi-module-with-sdk-for-anduino-diy-2063805?tc=EUR&ta=NL&gclid=CjwKCAiAs8XiBRAGEiwAFyQ-eiRcroPJNspsOGjilyNfXKL7VHNhb9nua00lCv-04f6rFGOTRJRz7xoCdc4QAvD_BwE#.XFF6E1xKjmE

Espressif Systems. (2017). *ESP8266 Technical Reference.* Espressif Systems. Opgehaald van https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf

Espressif Systems. (2018). *ESP8285 Datasheet.* Espressif Systems. Opgehaald van https://www.espressif.com/sites/default/files/documentation/0a-esp8285_datasheet_en.pdf

Espressif. (z.d.). *Non-volatile storage library*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/storage/nvs_flash.html

Espressif. (z.d.). *Partition Tables*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/partition-tables.html

Espressif. (z.d.). *Smart Config*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/network/esp_smartconfig.html?highlight=smartconfig

Espressif. (z.d.). *ULP coprocessor programming*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/ulp.html

GNU. (z.d.). *Function Call Syntax*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Syntax-of-Functions.html

GNU. (z.d.). *Functions for String Substitution and Analysis*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Text-Functions.html

GNU. (z.d.). *The call Function*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Call-Function.html

GNU. (z.d.). *The Function wildcard*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Wildcard-Function.html

Hogeschool Utrecht. (sd). *Turorials.* Opgehaald van Hogeschool Utrecht: https://www.bibliotheek.hu.nl/~/media/HU-BIBLIOTHEEK/Files/beoordelen.pdf?la=nl

Johnston, P. (2017, Mei 26). *JSMN: JSON parsing library*. Opgehaald van EMBEDDED ARTISTRY: https://embeddedartistry.com/blog/2017/3/28/jsmn-json-parser

Karl, M. J., LofgrenRobert, D., NormanGregory, B., & ThelinAnil, G. (1991). *Wear leveling techniques for flash EEPROM systems.* Washington D.C., Verenigde Staten: Western Digital Corp SanDisk Technologies LLC.

Libsodium. (2018, Augustus). *Introduction*. Opgehaald van Libsodium documentation: https://libsodium.gitbook.io/doc/

Linde, U. v. (2015, augustus). *Priotiteiten bepalen met MoSCoW*. Opgehaald van Xcess: https://www.xcess.nl/Blog/articleType/ArticleView/articleId/179/Prioriteiten-bepalen-met-MoSCoW

MediaTek Inc. (2015, januari 3). MediaTek MT7681 Datasheet. Opgehaald van https://docs.labs.mediatek.com/resource/linkit-connect-7681/en/documents

MQTT. (z.d.). *MQTT*. Opgehaald van MQTT: http://mqtt.org/

Nordic. (2014, september). nRF51 Series Reference Manual. Opgehaald van http://infocenter.nordicsemi.com/pdf/nRF51_RM_v3.0.pdf

Nordic. (2016, februari 17). nRF52832 - Product Specification v1.0. Opgehaald van https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.0.pdf

Purcell, A. (2009, juni 30). *Python Script Executed with Makefile*. Opgehaald van StackOverflow: https://stackoverflow.com/questions/1062436/python-script-executed-with-makefile

RealTek. (2016, oktober 6). Realtek Ameba RTL8195AM DEV01 User Manual. Opgehaald van https://cdn.sparkfun.com/assets/parts/1/2/2/3/6/UM0048_Realtek_Ameba_RTL8195AM_DEV_1v0_User_Manual_1v10_20161006.pdf

seeed. (z.d.). *RTL8710 WiFi Module*. Opgehaald van seeed: https://www.seeedstudio.com/RTL8710-WiFi-Module-p-2793.html

Shenzhen Boantong Technoloy Co., Ltd. (2016, mei 16). RTL00 WiFi Module. Opgehaald van http://aitendo3.sakura.ne.jp/aitendo_data/product_img/wireless/2.4G/RTL-00/RTL8710%20wifi%20module%20specification.pdf

SYSQA B.V. (2012). *ISO 25010: 2011.* Almere: SYSQA B.V.

Texas Instrumets. (2018, juni). CC3200 Technical Reference Manual. Opgehaald van http://www.ti.com/lit/ug/swru367d/swru367d.pdf

van Ooijen, W. (2018). *Studentenhandleiding Individual Propedeuse Assesssment.*

Vinschen, C., & Johnston, J. (z.d.). *Sourceware*. Opgehaald van Sourceware: http://www.sourceware.org/newlib/

Xtensa® Instruction Set Architecture. (2010). Santa Clara, Californië, Verenigde Staten: Tensilica, Inc. Opgeroepen op januari 4, 2019

9. Appendix

Appendix A – Plan van Aanpak

Integratie van draadloze communicatie aspecten Hogeschool Utrecht

Plan van Aanpak

Auteur	***René de Kluis***
Student nummer	*1661627*
Locatie	*Hogeschool Utrecht*
Datum	*18/10/2018*
Versie	*V1.0*
Status	*Eind Document*

Inhoudsopgave

1 Inleiding

In dit verslag zult u de aanpak voor het onderzoek naar de integratie van een nieuwe microcontroller in de bestaande libraries en tools van de Hogeschool Utrecht vinden en toevoeging van draadloze communicatie aspecten bij de opdrachten van de studierichting Technische Informatica.

Het onderzoek is onderdeel van het onderzoeksemester, dat in het vierde jaar van HBO-ICT aan de Hogeschool Utrecht gegeven wordt. Hierbij zal onderzoek gedaan worden naar microcontrollers waarbij wifi en/of bluetooth geïntegreerd zit. In dit document zal de probleemstelling voor het onderzoek naar voren komen, de aanpak hoe dit onderzoek uitgevoerd zal worden en de stappen die genomen zullen worden voor de realisatie van een Proof of Concept (PoC).

2 Organisatorische context

Het onderzoek zal uitgevoerd worden voor de afstudeerrichting Technische Informatie van de Hogeschool Utrecht. Hierbij zal de onderzoeker voornamelijk in het Turing lab van de Hogeschool Utrecht. Dit is een werkplek met meerdere elektronische componenten en apparaten voor studenten die met technische elementen willen werken. Het bevind zich op de vierde verdieping van de Heidelberglaan 15 in Utrecht.

§2.1 Relatie tot andere projecten

Binnen de Hogeschool Utrecht wordt altijd gezocht naar vernieuwing en verrijking van de lesstof. Hierbij hoort ook het integreren van nieuwe microcontrollers in de lesstof. In dit onderzoek zal niet gewerkt worden met libraries en tools van de fabrikant van de microcontroller, maar met libraries en tools die door de Hogeschool Utrecht gemaakt zijn. Het zal hier gaan om de HWLIB en RTOS libraries en de BMPTK make tool. HWLIB, gemaakt door Wouter van Ooijen, is een C++ library waarmee je, object georiënteerd, hardware kan aansturen. De RTOS (Real Time Operating System) library, gemaakt door Wouter van Ooijen en Marten Wensink, wordt gebruikt voor programma's waarbij de gebruiker op aangeduide tijdstippen bepaalde elementen wilt uitvoeren. Deze twee libraries worden in meerdere cursussen van Technische Informatica gebruikt. Het is de bedoeling dat deze libraries hardware abstractie bevatten, wat betekend dat de library voor elke mogelijke microcontroller gebruikt kan worden zonder aanpassingen. De BMPTK make tool, gemaakt door Wouter van Ooijen, is een, op make gebaseerde ontwikkel omgeving, waarmee een programma gemaakt kan worden voor microcontrollers. Deze tool zorgt dat de geschreven code op de hardware komt en dat deze de code ook kan uitvoeren.

Bij de Hogeschool Utrecht streven ze naar herbruikbaarheid van deze libraries en tools. Bij het toevoegen van een nieuwe microcontroller zal hiernaar gekeken moeten worden. Dit onderzoek kan daardoor nieuwe inzichten geven of aanpassingen in de libraries of tools gemaakt dienen te worden, voor volledige herbruikbaarheid.

§2.2 Contact

Onderzoeker

Naam	René de Kluis
Student nummer	1661627
Mobiel	+31(0)6 429 405 74
E-Mail	rene.dekluis@student.hu.nl

Opdrachtgever

Naam	Wouter van Ooijen
Locatie	Heidelberglaan 15 – 4.060 3512 JE, UTRECHT
Telefoon	+31 (0)6 38150444
E-Mail	Wouter.vanooijen@hu.nl

3 Probleemstelling

In dit hoofdstuk zal in de eerste paragraaf de aanleiding voor dit onderzoek besproken worden. Hierbij wordt het probleem beschreven en de wensen van de opdrachtgever. Daarna zullen de hoofd- en deelvragen opgesteld worden uit deze aanleiding. Deze vragen zullen de rode draad vormen voor het onderzoeksverslag.

§3.1 Kwestie

Bij de afstudeerrichting Technische Informatica van HBO-ICT aan de Hogeschool Utrecht, wordt voor de meeste technische opdrachten de Arduino Due gebruikt. Dit is een programmeer bordje, waarmee onder andere sensoren uitgelezen en aangestuurd kunnen worden. De Arduino Due heeft geen geïntegreerde wifi of bluetooth modules, waardoor deze aspecten niet voorkomen in de opdrachten die voor deze studierichting gegeven worden.

Aangezien Internet of Things (IoT) toepassingen steeds meer in opkomst zijn, wilt de Hogeschool Utrecht dit ook opnemen in de cursussen. Hiervoor willen zij een microcontroller gebruiken waarbij wifi en/of bluetooth geïntegreerd zit. Aangezien deze aspecten als toevoeging moeten dienen bij de huidige opgaven, zullen de huidige opgaven die voor de Arduino Due gegeven worden, ook op de nieuwe microcontroller moeten werken. Daarnaast wilt de Hogeschool Utrecht dat studenten op dezelfde manier kunnen programmeren voor verschillende microcontrollers, hiermee zal rekening gehouden moeten worden bij het toevoegen van de nieuwe microcontroller.

Ook zal in dit onderzoek gekeken worden naar de herbruikbaarheid van de libraries en tools die in §2.1 beschreven zijn. Als Proof of Concept (PoC) zal de integratie met de libraries en tools gerealiseerd worden voor de gekozen microcontroller. Na deze integratie zal gekeken worden of themaopdracht devices (een project wat in het eerste blok van jaar 2 gegeven wordt) gerealiseerd kan worden met de gekozen microcontroller.

§3.2 Vraagstelling

Om een nieuwe microcontroller te integreren in de huidige libraries en tools van de Hogeschool Utrecht kan de volgende hoofdvraag opgesteld worden:

"Hoe kan een nieuwe microcontroller met geïntegreerde wifi en/of bluetooth in de huidige libraries en tools van de Hogeschool Utrecht opgenomen worden, zodat draadloze communicatie aspecten toegevoegd kunnen worden in de opgaven van technische informatica?"

Aangezien de gehele onderzoeksvraag complex is, is gekozen om deze op te delen in verschillende deelvragen. Deze deelvragen zullen uiteindelijk samen een antwoord kunnen geven op de hoofdvraag. Hieronder zullen de deelvragen uitgewerkt worden die uit de hoofdvraag gefilterd kunnen worden.

1. Welke microcontroller ondersteunen wifi en/of bluetooth?
2. Met welke microcontrollers, die wifi en/of bluetooth ondersteunen, kunnen de huidige opgaven van Technische Informatica gerealiseerd worden?
3. Welke aanpassingen zijn nodig om de gekozen microcontroller op te nemen in libraries en tools van de Hogeschool Utrecht?
4. Op welke manier kunnen wifi en/of bluetooth aspecten toegevoegd worden in de opgaven van Technische Informatica?
5. Hoe kan een wifi library opgenomen worden in HWLIB?
6. Hoe kan een bluetooth library opgenomen worden in HWLIB?

4 Theoretisch kader

In dit hoofdstuk zal de belangrijkste theoretische achtergrond van dit onderzoek besproken worden. Hierbij wordt in de eerste paragraaf duidelijkheid verschaft over de kenbegrippen die in de documenten centraal zullen staan en welke relaties de kernbegrippen met elkaar hebben. Als laatste zullen ook de kwaliteitseisen besproken worden waar rekening mee gehouden zal worden bij de realisatie van het Proof of Concept (PoC).

§4.1 Kernbegrippen

In deze paragraaf zal verduidelijking worden gegeven over begrippen die in de documenten centraal zullen staan. Als eerst zal de herbruikbaarheid van de libraries en tools van de Hogeschool Utrecht behandeld worden en daarna hardware abstractie waarmee de integratie van de gekozen microcontroller gerealiseerd zal worden.

§4.1.1 Herbruikbaarheid van libraries

Bij de herbruikbaarheid van de libraries van de Hogeschool Utrecht zal gekeken op welke manier de gekozen microcontroller geïntegreerd kan worden en welke eventuele problemen dit met zich mee brengt. Voor de studenten moet de aansturing van de gekozen microcontroller op dezelfde manier gebeuren, zoals dit nu kan met de Arduino Due. Daarvoor zal gekeken moeten worden naar modulariteit van de libraries en tools van de Hogeschool Utrecht, zodat deze gebruikt kunnen worden voor de gekozen microcontroller.

§4.1.2 Hardware abstractie

Bij hardware abstractie laat de programmeur de software geloven dat alle mogelijke hardware aanwezig is, maar in feite zal de software alleen de functionaliteiten uitvoeren voor de aangesloten hardware. Hierdoor maakt het voor het systeem niet uit als net een ander module of sensor aangesloten wordt op het systeem.

In de huidige libraries van de Hogeschool Utrecht voor de aansturing van hardware is hardware abstractie aanwezig. Het is namelijk mogelijk om bijvoorbeeld de Aruino Uno, Arduino Nano en Arduino Due op dezelfde manier aan te sturen. In figuur 1 is de deel van de code te zien dat dit mogelijk maakt.

Aangezien de gekozen microcontroller hierbij toegevoegd dient te worden, zal hierdoor gekeken moeten worden naar de huidige hardware abstractie en of dit herbruikbaar is.

```
ifeq ($(TARGET),arduino_uno)
   CHIP         ?= atmega328
   XTAL         ?= 12000
endif

ifeq ($(TARGET),arduino_nano)
   CHIP         ?= atmega328
   XTAL         ?= 12000
endif

ifeq ($(TARGET),arduino_due)
   CHIP         ?= sam3x8e
   XTAL         ?= 12000
endif
```

Figuur 1- Hardware abstractie in BMPTK

§4.3 Kwaliteit

Zorgen voor een hoge kwaliteit van software is een hoge prioriteit voor een software ontwikkelaar. Daarom zal deze paragraaf gaan over de verschillende beoordelingscriteria waar rekening mee gehouden dient te worden bij het realiseren van onderzoek en het Proof of Concept (POC).

§4.3.1 ISO25010

Bij het ontwikkelen van het POC zullen enkele kwaliteitskenmerken van ISO25010 in acht genomen worden (SYSQA B.V., 2012). Volgend zullen de productkenmerken beschreven worden, die meegenomen worden bij de realisatie.

§4.3.1.1 Functionele geschiktheid

Bij functionele geschiktheid wordt gekeken naar de veronderstelde behoeften voor de gebruiker. Aangezien het product voor studenten is, zal hier rekening mee gehouden moeten worden.

§4.3.1.2 Prestatie-efficiëntie

Bij het maken van de wifi en bluetooth libraries, is het van belang dat het gebruik geen grote gevolgen heeft voor andere processen. Hierom zal bij de realisatie gelet worden op de snelheid van het product.

§4.3.1.3 Bruikbaarheid

Aangezien het product als lesstof bedoeld is, zal ook rekening gehouden moeten worden met de bruikbaarheid van het product. Studenten moeten leren om met de microcontroller om kunnen gaan, hierdoor zal de Leerbaarheid en bedienbaarheid een grote rol spelen bij de realisatie. Daarnaast stijgt het niveau programmeren van de studenten over de jaren, daardoor is het ook van belang om toegankelijkheid en het voorkomen van gebruikersfouten mee te nemen bij de ontwikkeling. Beginnende studenten zullen namelijk meer gebruikersfouten maken dan ervaren studenten; En de ervaren studenten zullen meer uit de microcontroller willen halen dan de beginnende.

§4.3.1.4 Onderhoudbaarheid

Onderhoudbaarheid zal ook een rol spelen bij de ontwikkeling, aangezien dit onderzoek door een huidige student gemaakt wordt, die in de komende jaren de studie waarschijnlijk zal verlaten. Als het gemaakte product door anderen onderhouden dient te worden, zal hierom rekening gehouden moeten worden met de modulariteit, herbruikbaarheid en wijzigbaarheid.

§4.3.1.5 Overdraagbaarheid

Net als bij onderhoudbaarheid, zal ook gezorgd moeten worden voor goede overdraagbaarheid van het product. Als uitbreidingen of aanpassingen in het product gemaakt dienen te worden, moet dit kunnen als de maker van het product niet meer aanwezig is. Hierdoor zal bij de realisatie gelet moeten worden op de aanpasbaarheid en vervangbaarheid. Ook zal installeerbaarheid meegenomen worden, zodat studenten het gemaakte product makkelijk kunnen installeren of verwijderen.

§4.3.2 Richtlijnen Hogeschool Utrecht

Naast de productkenmerken van ISO25010, zullen ook criteria in acht worden gehouden die aangeleerd zijn door de Hogeschool Utrecht. Zo zal bij het ontwikkelen van de software gekeken worden dat weinig codeduplicatie voorkomt. Dit betekend dat elementen in de code niet meermalig voor zullen komen, maar getracht wordt dit samen te voegen. Ook zal de software soft-coded gemaakt worden, zodat de modulariteit van het product groot zal zijn. Als laatste zal gekeken worden naar de kwaliteit van de Doxygen documentatie. Hierdoor zal de code ook leesbaar en begrijpelijk zijn als aanpassingen gemaakt dienen te worden of studenten ermee moeten werken. Om deze criteria te kunnen controleren zal feedback gevraagd worden aan het begeleidend docent of medestudenten.

6 Planning & Aanpak

Om het onderzoek tot een succesvol einde te kunnen brengen, is het van belang om een planning op te stellen en de te bedenken welke stappen ondernomen dienen te worden voor de aanpak van het onderzoek. In dit hoofdstuk zullen deze twee onderdelen besproken worden. In de eerste paragraaf zal de aanpak behandeld worden, daarna zal de planning opgesteld worden.

§6.1 Aanpak

Om een goed onderzoek te kunnen houden moet bepaald worden welke stappen genomen dienen te worden voor de realisatie. Om deze stappen te kunnen bepalen zal in de volgende paragraaf per deelvraag uitgewerkt worden welke stappen ondernomen zullen worden. Daarna zal ook de aanpak besproken worden voor de realisatie van het Proof of Concept.

§6.1.1 Aanpak per deelvraag

Volgend zal per deelvraag uitgewerkt worden welke stappen ondernomen dienen te worden om de deelvragen te kunnen beantwoorden.

In appendix A zal per deelvraag een analyse uitgewerkt staan wat voor soort deelvraag het is, op welke manier de dataverzameling zal plaatsvinden, op welke manier het de gevonden data geanalyseerd zal worden en in wat voor vorm het antwoord op de deelvraag verwacht kan worden.

Welke microcontroller ondersteunen wifi en/of bluetooth?

- Onderzoeken welke microcontroller wifi geïntegreerd hebben
- Onderzoeken welke microcontroller bluetooth geïntegreerd hebben
- Onderzoeken welke microcontroller relevant zijn voor studenten
    - Kosten van de microcontroller
    - Betrouwbaarheid dat de microcontroller nog lang in gebruik zal zijn

Met welke microcontrollers, die wifi en/of bluetooth ondersteunen, kunnen de huidige opgaven van Technische Informatica gerealiseerd worden?

- Onderzoeken welke functionaliteiten benodigd zijn voor de opdracht.
- Onderzoeken of er genoeg poorten aanwezig zijn op de gekozen microcontroller om de opgave te kunnen realiseren.
- Onderzoeken of de vereiste functionaliteiten op de gekozen microcontroller gemaakt kunnen worden

Welke aanpassingen zijn nodig om de gekozen microcontroller op te nemen in libraries en tools van de Hogeschool Utrecht?

Onderzoek naar BMPTK:

- Onderzoeken hoe BMPTK een programma bouwt.
- Onderzoeken hoe een programma voor de gekozen microcontroller gebouwd dient te worden.
- Onderzoeken hoe het bouwen van een programma voor de gekozen microcontroller door BMPTK kan gebeuren.
- Onderzoeken hoe de gekozen microcontroller in BMPTK opgenomen kan worden.

Onderzoek naar HWLIB:

- De functionaliteiten van HWLIB onderzoeken.
- Onderzoeken hoe HWLIB de hardware aanstuurt.
- Aansturing gekozen microcontroller onderzoeken.
- Onderzoeken hoe de gekozen microcontroller in HWLIB opgenomen kan worden.

Onderzoek naar RTOS:

- Functionaliteiten van RTOS onderzoeken.
- Onderzoeken of de gekozen microcontroller deze functionaliteiten kan ondersteunen.
- Onderzoeken hoe de RTOS functionaliteiten de gekozen microcontroller kunnen aansturen.

Op welke manier kunnen wifi en/of bluetooth aspecten toegevoegd worden in de opgaven van Technische Informatica?

- Onderzoeken welke functionaliteiten benodigd zijn voor de opdracht
- Analyseren welke functionaliteiten overgenomen zouden kunnen worden m.b.v. wifi of bluetooth
- Onderzoeken of opdrachten gemaakt kunnen worden met wifi, die aansluiten bij de leerdoelen
- Onderzoeken of opdrachten toegevoegd kunnen worden met bluetooth, die aansluiten bij de leerdoelen

Hoe kan een wifi library opgenomen worden in HWLIB?

- Aansturing van de verschillende Cores van de gekozen microcontroller onderzoeken.
- Snelheid van het aansturen van WiFi onderzoeken.
- Snelheid van het uitlezen van WiFi onderzoeken.
- Snelheid van het versturen van data over WiFi onderzoeken.
- Ontvangstsnelheid van dataoverdracht over WiFi onderzoeken.

Hoe kan een bluetooth library opgenomen worden in HWLIB?

- Aansturing van de verschillende Cores van de gekozen microcontroller onderzoeken.
- Snelheid van het aansturen van Bluetooth onderzoeken.
- Snelheid van het uitlezen van Bluetooth onderzoeken.
- Snelheid van het versturen van data over Bluetooth onderzoeken.
- Ontvangstsnelheid van dataoverdracht over Bluetooth onderzoeken

§6.2 MoSCoW analyse

Om de realisatie van het Proof of Concept (PoC) zo goed mogelijk te laten verlopen. Is het van belang van de eisen gedefinieerd worden. Dit zal in deze paragraaf behandeld worden. De eisen waaraan het PoC moet voldoen zullen aan de hand van een MoSCoW analyse worden uitgewerkt (Linde, 2015). Hierdoor kan een duidelijk beeld gevormd worden, welke taken vereist zijn voor het behalen van een werkend product (Must Have), welke taken eigenlijk vereist zijn, maar het product alsnog geslaagd is zonder (Should Have), welke taken in het product kunnen, maar niet direct noodzakelijk zijn (Could Have) en welke taken niet in het product zullen komen, maar misschien ideeën zijn voor in de toekomst (Won't Have).

BMPTK

Taak	Prioriteit
Het programma kan gebouwd worden	Must Have
Het programma kan serieel op de microcontroller geflashed worden	Must Have
Het BMPTK scherm wordt geopend als het programma op de microcontroller draait	Must Have
Het programma kan draadloos op de microcontroller geflashed worden	Could Have
Real time communicatie kan gebeuren via het BMPTK scherm	Could Have

Integratie met Opdrachten

Taak	Prioriteit
De benodigde poorten kunnen aangestuurd worden	Must Have
Het programma kan voldoen aan de originele eisen van de opdracht	Must Have
Assembler opdrachten kunnen gemaakt worden op de microcontroller	Should Have
'Free RTOS' kan samenwerken met HWLIB	Could Have
RTOS opdrachten kunnen gerealiseerd worden met 'Free RTOS'	Could Have

RTOS

Taak	Prioriteit
Channel kan gemaakt worden	Must Have
Channel kan aangestuurd worden	Must Have
Channel kan uitgelezen worden	Must Have
Clock kan aangemaakt worden	Must Have
Clock kan aangestuurd worden	Must Have
Clock kan uitgelezen worden	Must Have
Event kan aangemaakt worden	Must Have
Event kan aangestuurd worden	Must Have
Event kan uitgelezen worden	Must Have
Flag kan aangemaakt worden	Must Have
Flag kan aangestuurd worden	Must Have
Flag kan uitgelezen worden	Must Have
Mailbox kan aangemaakt worden	Must Have
Mailbox kan gevult worden	Must Have
Mailbox kan uitgelezen worden	Must Have
Mutex kan aangemaakt worden	Must Have
Mutex kan aangestuurd worden	Must Have
Pool kan aangemaakt worden	Must Have
Pool kan aangestuurd worden	Must Have
Pool kan uitgelezen worden	Must Have
Task kan aangemaakt worden	Must Have
Task kan aangestuurd worden	Must Have
Timer kan aangemaakt worden	Must Have
Timer kan aangestuurd worden	Must Have
Timer kan uitgelezen worden	Must Have
Waitable kan aangemaakt worden	Must Have
Waitable kan aangestuurd worden	Must Have
Co-routines kunnen gemaakt worden	Must Have
Co-routines kunnen gestart worden	Must Have

Wifi kan aangestuurd worden met RTOS	Must Have
Bluetooth kan aangestuurd worden	Must Have
Meerdere microcontrollers kunnen met elkaar communiceren	Could Have

HWLIB

Taak	Prioriteit
Pin_in kan aangemaakt worden	Must Have
Pin_in kan uitgelezen worden	Must Have
Pin_out kan aangemaakt worden	Must Have
Pin_out kan aangestuurd worden	Must Have
Pin_in_out kan aangemaakt worden	Must Have
Pin_in_out kan aangestuurd worden	Must Have
Pin_in_out kan uitgelezen worden	Must Have
Keypad kan aangemaakt worden	Must Have
Keypad kan uitgelezen worden	Must Have
Window kan aangemaakt worden	Must Have
Een cirkel kan op het window getekend worden	Must Have
Een vierkant kan op het window getekend worden	Must Have
Oled kan aangemaakt worden	Should Have
Oled kan aangestuurd worden	Should Have
ADC kan aangestuurd worden	Should Have
ADC kan uitgelezen worden	Should Have
DAC kan aangestuurd worden	Should Have
DAC kan uitgelezen worden	Should Have
RS232 verbinding kan gemaakt worden	Could Have
RS-232 verbinding kan aangestuurd worden	Could Have
RS-232 verbinding kan uitgelezen worden	Could Have

7 Ethische afwegingen

Bij de realisatie van elk onderzoek, zal rekening gehouden moeten worden met Ethische dilemma's. Dit houd in dat gekeken dient te worden wat voor impact de realisatie van het onderzoek kan hebben voor andere mensen. In dit hoofdstuk zullen deze vraagstukken behandeld worden en op welke manier hiermee omgegaan zal worden.

§7.1 Druk op student

Als een extra microcontroller opgenomen wordt in de lesstof, zal dit zorgen voor meer werkdruk bij de student. Deze zal namelijk ook moeten leren over deze microcontroller en hoe deze gebruikt dient te worden. Echter zal dit niet veel meer zijn dan de student nu moet doen, hierdoor zal dit vraagstuk geen impact hebben op het onderzoek.

§7.2 Extra kosten

Als de gekozen microcontroller in de lesstof opgenomen wordt, betekend dit dat de student hier ook naslagwerk voor dient te hebben. Dit zal waarschijnlijk resulteren in het kopen van extra boeken, wat kosten met zich mee draagt. Echter zullen deze kosten niet dusdanig hoog zijn dat dit waarschijnlijk een probleem zal vormen voor de studenten, waardoor dit vraagstuk geen impact zal hebben op het onderzoek.

Daarnaast wordt momenteel van de student verwacht dat hij/ zij de Arduino Due aanschaft voor de studie. Als de gekozen microcontroller wordt geïntegreerd, zal deze ook door de student aangeschaft moeten worden. Dit brengt extra kosten met zich mee, wat de student moet betalen. Hierdoor zal bij het bepalen van nieuwe de microcontroller, gelet worden op de kosten hiervan. Om het betaalbaar te houden voor studenten is, in overleg met Wouter van Ooijen, gekozen om het onderzoek te doen naar microcontrollers onder de €50,-.

8 Risico analyse

In dit hoofdstuk zal de risico analyse behandeld worden. Dit zal gaan over eventuele risico's die het onderzoek zouden kunnen belemmeren. Aangezien dit onderzoek door één persoon gemaakt zal worden, is geen sprake van grote invloeden van interne risico's. Hierdoor zal in dit hoofdstuk alleen de externe risico's behandeld worden.

§8.1 Externe Risico's

Externe Risico's zijn invloeden van buitenaf die het onderzoek zouden kunnen belemmeren. Voor het slagen van het onderzoek, is het van belang dat rekening gehouden wordt dat deze risico's zich kunnen voordoen. Deze risico's zullen, samen met de bijbehorende maatregelen, in tabel 2 behandeld worden.

Risico	**Uitleg**	**Maatregel**
Levering van de gekozen microcontroller duur lang	Om de aansturing van de gekozen microcontroller te kunnen maken, is het nodig om deze op tijd in bezit te hebben. Anders kan dit voor vertraging zorgen van het onderzoek.	Als de levering van de gekozen microcontroller te lang duurt, zal gekeken worden of andere mensen deze chip hebben en of deze geleend kan worden tot de bestelde microcontroller binnen is.
BMPTK kan niet samen werken met de toolchain van de gekozen microcontroller	Het kan zijn dat BMPTK niet samen kan werken met de toolchain van de gekozen microcontroller. Dit betekend dat het programma niet door BMPTK gebouwd kan worden en een extern make programma gebruikt dient te worden.	Mocht dit probleem zich voor doen, zal eerst met Wouter van Ooijen overleg plaats vinden om een oplossing hiervoor te vinden. Anders zal een omweg gemaakt worden om het alsnog te realiseren.

Tabel 2 – Externe risico's

Risico	Uitleg	Maatregel
RTOS kan niet gebruikt worden op de gekozen microcontroller	Als de RTOS van de Hogeschool Utrecht niet op de gekozen microcontroller gebruikt kan worden, kan die gevolgen hebben voor de realisatie van opdrachten.	Bij het voordoen van dit probleem zal gekeken worden naar het gebruik van een andere RTOS of een omweg gerealiseerd worden.
Functionaliteiten op de gekozen microcontroller kunnen niet door HWLIB aangestuurd worden	Het kan zijn dat sommige functionaliteiten op de gekozen microcontroller niet aangestuurd kunnen worden door HWLIB. Dit kan gevolgen hebben voor de realisatie van opdrachten.	Deze functionaliteiten zullen toegevoegd worden aan HWLIB.
Opdrachten van de Hogeschool Utrecht kunnen niet gerealiseerd worden op de gekozen microcontroller	Ook is een risico van dit onderzoek, dat de opdrachten die door de Hogeschool Utrecht gegeven worden, niet op de gekozen microcontroller te realiseren zijn. Dit heeft grote gevolgen voor de realisatie van dit onderzoek, aangezien de integratie van de microcontroller in de lesstof hierbij centraal staat.	Om dit risico te vermeiden zal gekeken worden naar de opdrachten en de functionaliteiten die benodigd worden. Mochten functionaliteiten niet gerealiseerd kunnen worden, zal overleg plaats vinden met de opdrachtgever.

Tabel 2 – Externe risico's (vervolg)

9 Communicatie

Goede communicatie is essentieel bij het maken van een onderzoek. Tijdens de onderzoeksperiode zijn er verschillende betrokkenen. Als iets verkeerd gaat of anders loopt als verwacht/ overeengekomen is, zal dit opgelost moeten worden.

Elke week zal minimaal één overleg plaats vinden van een half uur met het begeleidend docent, hierdoor blijft hij op de hoogte van de huidige status en het verloop van het onderzoek. In het geval dat het onderzoek veranderd dient te worden, zal hierbij eerst overleg plaats vinden met het begeleidend docent. Aangezien het begeleidend docent de huidige libraries van de Hogeschool Utrecht gemaakt heeft, zal met hem waarschijnlijk meer contact zijn dan boven beschreven is, aangezien hij daardoor waarschijnlijk het beste kan helpen als zich knelpunten voordoen tijdens het onderzoek.

10 Bibliografie

Agile Scrum Group. (z.d.). *De Daily standup meeting: uitleg en tips*. Opgehaald van Scrum Guide: https://scrumguide.nl/daily-standup-meeting/

Anthon. (2015, juni 12). *Invoke python script through make command*. Opgehaald van Unix & Linux: https://unix.stackexchange.com/questions/209173/invoke-python-script-through-make-command

Archlinux. (2018, November 26). *WPA supplicant*. Opgehaald van Archlinux: https://wiki.archlinux.org/index.php/WPA_supplicant

ARM MBED. (z.d.). *ARM MBED*. Opgehaald van ARM MBED: https://tls.mbed.org/

Botland. (z.d.). *Bluno Nano and BLE Bluetooth 4.0 - compatible with Arduino*. Opgehaald van Botland: https://botland.com.pl/en/plytki-zgodne-z-arduino-dfrobot/2998-bluno-nano-and-ble-bluetooth-40-compatible-with-arduino.html

Cee, Y. (sd). *ESP-WROOM-32 ESP32.* Your Cee. Opgehaald van https://nl.aliexpress.com/item/ESP-WROOM-32-ESP32-Bluetooth-and-WIFI-Dual-Core-CPU-with-Low-Power-Consumption-MCU/32793415575.html

Chan, E. (2018, Oktober 14). *FatFs - Generic FAT Filesystem Module*. Opgehaald van elm-chan: http://elm-chan.org/fsw/ff/00index_e.html

Cheshire, S., & Krochmal, M. (2013). *Internet Engineering Task Force (IETF).* Apple Inc.: Februari. Opgehaald van https://tools.ietf.org/html/rfc6762

CMake. (z.d.). *CMake*. Opgehaald van CMake: https://cmake.org/

Cooper, C. (1999, September 1). *Using Expat*. Opgehaald van XML: https://www.xml.com/pub/a/1999/09/expat/index.html

DealExtreme. (z.d.). *NL6621-Y1 Remote Control Serial Port naar Wi-Fi SDK Module voor Anduino*. Opgehaald van DealExtreme: https://www.dx.com/nl/p/nl6621-y1-remote-control-serial-port-to-wi-fi-module-with-sdk-for-anduino-diy-2063805?tc=EUR&ta=NL&gclid=CjwKCAiAs8XiBRAGEiwAFyQ-eiRcroPJNspsOGjilyNfXKL7VHNhb9nua00lCv-04f6rFGOTRJRz7xoCdc4QAvD_BwE#.XFF6E1xKjmE

Espressif Systems. (2017). *ESP8266 Technical Reference.* Espressif Systems. Opgehaald van https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf

Espressif Systems. (2018). *ESP8285 Datasheet.* Espressif Systems. Opgehaald van https://www.espressif.com/sites/default/files/documentation/0a-esp8285_datasheet_en.pdf

Espressif. (z.d.). *Non-volatile storage library*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/storage/nvs_flash.html

Espressif. (z.d.). *Partition Tables*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/partition-tables.html

Espressif. (z.d.). *Smart Config*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/network/esp_smartconfig.html?highlight=smartconfig

Espressif. (z.d.). *ULP coprocessor programming*. Opgehaald van ESP-IDF Programming Guide: https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/ulp.html

GNU. (z.d.). *Function Call Syntax*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Syntax-of-Functions.html

GNU. (z.d.). *Functions for String Substitution and Analysis*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Text-Functions.html

GNU. (z.d.). *The call Function*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Call-Function.html

GNU. (z.d.). *The Function wildcard*. Opgehaald van GNU: https://www.gnu.org/software/make/manual/html_node/Wildcard-Function.html

Hogeschool Utrecht. (sd). *Turorials.* Opgehaald van Hogeschool Utrecht: https://www.bibliotheek.hu.nl/~/media/HU-BIBLIOTHEEK/Files/beoordelen.pdf?la=nl

Johnston, P. (2017, Mei 26). *JSMN: JSON parsing library*. Opgehaald van EMBEDDED ARTISTRY: https://embeddedartistry.com/blog/2017/3/28/jsmn-json-parser

Karl, M. J., LofgrenRobert, D., NormanGregory, B., & ThelinAnil, G. (1991). *Wear leveling techniques for flash EEPROM systems.* Washington D.C., Verenigde Staten: Western Digital Corp SanDisk Technologies LLC.

Libsodium. (2018, Augustus). *Introduction*. Opgehaald van Libsodium documentation: https://libsodium.gitbook.io/doc/

Linde, U. v. (2015, augustus). *Priotiteiten bepalen met MoSCoW*. Opgehaald van Xcess: https://www.xcess.nl/Blog/articleType/ArticleView/articleId/179/Prioriteiten-bepalen-met-MoSCoW

MediaTek Inc. (2015, januari 3). MediaTek MT7681 Datasheet. Opgehaald van https://docs.labs.mediatek.com/resource/linkit-connect-7681/en/documents

MQTT. (z.d.). *MQTT*. Opgehaald van MQTT: http://mqtt.org/

Nordic. (2014, september). nRF51 Series Reference Manual. Opgehaald van http://infocenter.nordicsemi.com/pdf/nRF51_RM_v3.0.pdf

Nordic. (2016, februari 17). nRF52832 - Product Specification v1.0. Opgehaald van https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.0.pdf

Purcell, A. (2009, juni 30). *Python Script Executed with Makefile*. Opgehaald van StackOverflow: https://stackoverflow.com/questions/1062436/python-script-executed-with-makefile

RealTek. (2016, oktober 6). Realtek Ameba RTL8195AM DEV01 User Manual. Opgehaald van https://cdn.sparkfun.com/assets/parts/1/2/2/3/6/UM0048_Realtek_Ameba_RTL8195AM_DEV_1v0_User_Manual_1v10_20161006.pdf

seeed. (z.d.). *RTL8710 WiFi Module*. Opgehaald van seeed: https://www.seeedstudio.com/RTL8710-WiFi-Module-p-2793.html

Shenzhen Boantong Technoloy Co., Ltd. (2016, mei 16). RTL00 WiFi Module. Opgehaald van http://aitendo3.sakura.ne.jp/aitendo_data/product_img/wireless/2.4G/RTL-00/RTL8710%20wifi%20module%20specification.pdf

SYSQA B.V. (2012). *ISO 25010: 2011.* Almere: SYSQA B.V.

Texas Instrumets. (2018, juni). CC3200 Technical Reference Manual. Opgehaald van http://www.ti.com/lit/ug/swru367d/swru367d.pdf

van Ooijen, W. (2018). *Studentenhandleiding Individual Propedeuse Assesssment.*

Vinschen, C., & Johnston, J. (z.d.). *Sourceware*. Opgehaald van Sourceware: http://www.sourceware.org/newlib/

Xtensa® Instruction Set Architecture. (2010). Santa Clara, Californië, Verenigde Staten: Tensilica, Inc. Opgeroepen op januari 4, 2019

11 Appendix

Appendix A – Deelvraag Analyse

In deze bijlage is de aanpak te vinden van de deelvragen. Deze aanpak zal bestaan uit het onderzoeken welk type de deelvraag heeft, welke methode gebruikt zal worden voor dataverzameling, hoe de verkregen data geanalyseerd zal worden en het verwachte resultaat op de deelvraag.

Welke microcontroller met geïntegreerde wifi en/of bluetooth kan het beste gekozen worden voor studenten?

Type	Vergelijkend Evaluerend
Methode dataverzameling	Literatuur onderzoek
Analyse Methode	Analyse bruikbaarheid van microcontroller voor studenten. Analyse betrouwbaarheid van de microcontroller.
Resultaat	Document met overzicht: Verschillende bruikbare microcontrollers met geïntegreerde wifi en/of bluetooth

Met welke microcontrollers, die wifi en/of bluetooth ondersteunen, kunnen de huidige opgaven van Technische Informatica gerealiseerd worden?

Type	Beschrijvend
Methode dataverzameling	Literatuur onderzoek Experimenteel onderzoek
Analyse Methode	Analyse bruikbaarheid microcontrollers voor de opdrachten.
Resultaat	Proof of Concept

Welke aanpassingen zijn nodig om de gekozen microcontroller op te nemen in libraries en tools van de Hogeschool Utrecht?

Type	Ontwerpend Verklarend
Methode dataverzameling	Experimenteel onderzoek Literatuur onderzoek
Analyse Methode	Analyse bruikbaarheid van de aansturing voor de gekozen microcontroller met de libraries en tools van de Hogeschool Utrecht.
Resultaat	Proof of Concept Document met uitleg om de gekozen microcontroller aan te sturen met de libraries en tools van de Hogeschool Utrecht.

Op welke manier kunnen wifi en/of bluetooth aspecten toegevoegd worden in de opgaven van Technische Informatica?

Type	Beschrijvend Definiërend Verklarend
Methode dataverzameling	Observatie Experimenteel onderzoek Literatuur onderzoek
Analyse Methode	Analyse bruikbaarheid van de gekozen microcontroller voor de huidige opdrachten van de Hogeschool Utrecht.
Resultaat	Document hoe wifi en/of bluetooth aspecten toegevoegd kunnen worden aan de opdracht en hoe de opdracht gerealiseerd kan worden met de gekozen microcontroller.

Hoe kan een wifi library opgenomen worden in HWLIB?

Type	Desk research Ontwerpend Verklarend
Methode dataverzameling	Experimenteel onderzoek Literatuur onderzoek
Analyse Methode	Analyse herbruikbaarheid van HWLIB. Analyse bruikbaarheid van wifi library in HWLIB.
Resultaat	Proof of Concept

Hoe kan een bluetooth library opgenomen worden in HWLIB?

Type	Desk research Ontwerpend Verklarend
Methode dataverzameling	Experimenteel onderzoek Literatuur onderzoek
Analyse Methode	Analyse herbruikbaarheid van HWLIB. Analyse bruikbaarheid van wifi library in HWLIB.
Resultaat	Proof of Concept

Appendix B – Methoden pad

In deze appendix zal beschreven worden welk pad gevolgd zal worden door de verschillende mogelijke methoden. Dit pad zal helpen bij de realisatie van het onderzoek. Eerst zal het pad gedefinieerd worden, waarna de verklaring volgt waarom deze stappen gekozen zijn en waarom precies in die volgorde.

Stakeholder Analyse (Field) → Expert interview (library) → Requirements Prioritization (Workshop) → Prototyping (Workshop) → Peer review (Showroom)

Eerst zal fieldwork gedaan worden door middel van een stakeholder analyse. Als de gekozen microcontroller de Arduino Due moet vervangen, zal met de docenten van Technische Informatica overlegd worden, wat zij van de module willen en wat volgens hen minimaal bij de realisatie opgenomen dient te worden.

Vervolgens zal met Wouter van Ooijen een expert interview gehouden worden, aangezien hij de huidige libraries gemaakt heeft die op de Hogeschool Utrecht gebruikt worden. Hierbij kunnen inzichten naar voren komen die nuttig zijn voor de ontwikkeling van de library voor de aansturing van de gekozen microcontroller.

Daarna zal in de workshop aan bod komen. Hier zal requirements prioritization gehouden worden. Om de library voor de gekozen microcontroller zo snel mogelijk bruikbaar te kunnen maken voor de stakeholder (Hogeschool Utrecht) is het van belang dat alle essentiële onderdelen aanwezig zijn. Daarna kan nog gekeken worden naar uitbreiding van de library, mocht dit nodig zijn.

Ook zal een Proof of Concept (PoC) gemaakt worden, wat de initiele veriant van de library zal vormen. Dit valt onder Prototyping uit de methode workshop.

Als laatste zullen peer reviews gehouden worden met Wouter van Ooijen, waarbij hij nuttige inzichten kan geven voor de verdere ontwikkeling van de library en advies geven voor aanpassing van gekozen implementaties.

Appendix C – Begrippenlijst

Arduino Due
Arduino Due is een microcontroller van het bedrijf Arduino. De microcontroller heeft een Cortex-M3 processor op het board en wordt in de lesstof van Technische Informatica gebruikt voor het aanleren van C++ en assembler programmeren op microcontrollers.

HWLIB
HWLIB is een C++ library van de Hogeschool Utrecht. Deze library kan gebruikt worden voor het aansturen van hardware.

RTOS
RTOS staat voor Real-Time Operating System. Dit is een besturingssysteem waarbij taken uitgevoerd kunnen worden op tijdstippen dat de gebruiker dit wilt.

BMPTK
Bare Metal Programming Tool Kit (BMPTK) is een op make gebaseerde ontwikkelomgeving dat door de Hogeschool Utrecht ontwikkeld is.

Appendix B – Microcontroller Specificaties

In deze bijlage zijn de specificaties van de mogelijke microcontrollers, waarbij wifi en/ of bluetooth geïntegreerd zit verder uitgewerkt. De microcontrollers staan per fabrikant uitgewerkt.

§B.1 – Zoektermen

Bij het onderzoek naar de verschillende microcontrollers waarbij Wifi en/ of Bluetooth geïntegreerd zit zijn meerdere zoektermen op Google gebruikt. Voor het zoeken naar microcontrollers met een geïntegreerde Bluetooth module zijn de volgende zoektermen gebruikt:

- Microcontroller AND “integrated bluetooth”
- Bluetooth microcontroller
- Microcontroller AND bluetooth

Voor het zoeken naar microcontrollers met een geïntegreerde Wifi module zijn de volgende zoektermen gebruikt:

- Microcontroller AND “integrated wifi”
- Wifi microcontroller
- Microcontroller AND wifi

§B.2 - Arduino

Naam	**Bluno Nano**
Fabriekant	Arduino
Processor	Atmega328
Frequentie	20 MHz
ROM	1 KB

RAM	32 KB
Wifi	Nee
Bluetooth	Ja Bluetooth 4.0
Prijs	€21,00 - €41,00
Verkrijgbaar bij AliExpress	Ja
Anders verkrijgbaar	Ja

§B.3 - Espressif

Naam	Esp32	Esp8285	ESP8266
Fabriekant	Espressif	Espressif	Espressif
Processor	Tensilica Xtensa LX6	Tensilica Xtensa L106	Tensilica Xtensa L106
Frequentie	80 – 240 MHz	26 – 52 MHz	80 MHz
ROM	448 KB	Geen programmeerbare ROM	64 KB (Niet programmeerbaar)
RAM	520 KB	50 KB	32 KB instruction RAM 32 KB instruction Cache RAM 80 KB user-data RAM 16 KB ETS system-data RAM
Wifi	Ja (2.4 – 2.5 GHz) IEEE802.11b/g/n	Ja	Ja
Bluetooth	Ja Bluetooth 4.2	Nee	Nee
Prijs	€5,00 - €11,00	€2,00	€2,00 - €12,00
Verkrijgbaar bij AliExpress	Ja	Ja	Ja
Anders verkrijgbaar	Ja	Ja	Ja

§B.4 - MediaTek

Naam	MT7681
Fabriekant	MediaTek
Processor	Andes N9
Frequentie	40 MHz
ROM	1 MB
RAM	48 KB
Wifi	Ja
Bluetooth	Nee
Prijs	€3,50
Verkrijgbaar bij AliExpress	Ja
Anders verkrijgbaar	Ja Mouser

§B.5 - Nordic

Naam	nRF51822	nRF52832
Fabriekant	Nordic	Nordic
Processor	ARM Cortex-M0	ARM Cortex-M4
Frequentie	150 - 275 MHz	64 MHz
ROM	256 KB	512 KB
RAM	32 KB	64KB
Wifi	Nee	Nee
Bluetooth	JA	Ja Bluetooth 4.0
Prijs	€30,00 - €34,00	€19,00 - €34,00
Verkrijgbaar bij AliExpress	Ja	Ja
Anders verkrijgbaar	Ja	Ja

§B.6 - NuFront

Naam	NL6621
Fabriekant	NuFront
Processor	Arm Cortex-M3
Frequentie	40 MHz
ROM	64 KB
RAM	192 KB
Wifi	Ja
Bluetooth	Nee
Prijs	€3,00
Verkrijgbaar bij AliExpress	Ja
Anders verkrijgbaar	Ja, Niet in Nederland

§B.7 - RealTek

Naam	RTL8710	RTL8195
Fabriekant	RealTek	RealTek
Processor	ARM Cortex-M3	ARM Cortex-M3
Frequentie		166MHz
ROM	1 MB	1MB
RAM	48 KB	512KB
Wifi	Ja 802.11b/g/n	Ja
Bluetooth	Nee	Nee
Prijs	€5,00	€40,00
Verkrijgbaar bij AliExpress	Ja	Ja
Anders verkrijgbaar	Ja Mouser	Ja Mouser

§B.8 - Texas Instruments

Naam	**CC3200**
Fabriekant	Texas Instruments
Processor	ARM Cortex-M4
Frequentie	80 MHz
ROM	1 MB
RAM	256 KB
Wifi	Ja 802.11b/g/n
Bluetooth	Nee
Prijs	€26,00 - €39,00
Verkrijgbaar bij AliExpress	Ja
Anders verkrijgbaar	Ja

Appendix C – ESP-IDF Componenten

In deze appendix zullen de verschillende componenten/ functionaliteiten uitgewerkt worden die de ESP-IDF bevat. Elk component is een C-library die gebruikt kan worden bij het maken van een programma voor de esp32.

Component	**Beschrijving**
app_trace	Library dat zorgt dat arbitraire data via een JTAG interface tussen de host en esp32 verstuurd kan worden tijdens het uitvoeren van het programma.
app_update	Library voor het updaten van apps. Deze library wordt o.a. gebruikt door de esp_https_ota library.
asio	Library voor Audio Stream Input/Output protocol.
aws_iot	Library gat apparaten die aan AWS services gekoppeld zijn, verbonden kunnen worden met andere apparaten.
bootloader	Library met basisinformatie die benodigd zijn voor het draaien van een programma op de esp32.
bootloader_support	Library met extra bestanden die benodigd zijn voor de bootloader.
bt	Bluetooth library.
coap	Library voor web-transfer protocol voor IoT applicaties.
console	Library dat een interactief console verzorgt over de seriële poort.
cxx	Library voor CXX functionaliteiten.
driver	Library dat ADC en SPI configuraties verzorgt.

esp-tls	Library voor het checken van authenticiteit van een server.
esp32	Core-elementen voor de aansturing van de esp32.
espcoredump	Library waardoor informatie van de esp32 opgeslagen worden tijdens het crashen van het programma.
esptool_py	Tool waarmee het programma op de esp32 geflashed kan worden.
esp_adc_cal	Library waarmee analoge waarden geconvergeerd kunnen worden naar digitale waarden.

esp_event	Library waarmee events aangemaakt kunnen worden op de esp32.
esp_https_ota	HTTPS library.
esp_https_server	HTTPS library.
esp_http_client	HTTP library.
esp_http_server	HTTP library.
esp_ringbuf	Library waarmee ring buffers aangemaakt kunnen worden.
ethernet	Library voor aansturing van ethernet.
expat	Library voor het lezen en schrijven van XML bestanden. (Cooper, 1999)
fatfs	Library dat FAT filesystemen ondersteund. Dit is een filesysteem dat ontwikkeld is voor kleine embedd systemen. (Chan, 2018)
freemodbus	Library wat hot Modbus protocol mogelijk maakt voor de esp32.
freertos	Library waardoor gebruik gemaakt kan worden van een RTOS.
heap	Library waardoor gewerkt kan worden met een Heap.
jsmn	Library voor verwerking van JSON bestanden. Dit is een lichtgewicht library, wat het bruikbaar maakt voor kleine embedded systemen. (Johnston, 2017)
json	Library voor verwerking van JSON bestanden.
libsodium	Library voor o.a. encyptie, decryptie en wachtwoord hashing. (Libsodium, 2018)
log	Library waarmee informatie gelogd kan worden en uitgeprint op het terminal.
lwip	Library voor het maken van een TCP/IP stack.
mbedtls	Library voor het gebruik van de embed library, waarmee SSL gebruikt kan worden. (ARM MBED, z.d.)
mdns	Library waarmee IP-addressen toegekent kunnen worden aan andere apparaten in een klein netwerk. (Cheshire & Krochmal, 2013)
micro-ecc	Library waarmee de Elliptic-curce Diffie-Hellman en Elliptic Curce Digital Signature Algorithm protocollen gebruikt kunnen worden.
mqtt	Library de gebruik van MQTT mogelijk maakt.
newlib	Library dat een bundel is van verschillende libraries voor embedded systemen. (Vinschen & Johnston, z.d.)
nghttp	Library om HTTP mogelijk te maken voor embedded systemen.

nvs_flash	Library waarmee variablen een waarde toegekent kunnen krijgen. Deze paren worden opgeslagen in flash. (Espressif, z.d.)
openssl	Library voor het gebruik van SSL.
partition_table	Library dat op de esp32 geflashed wordt. Als meerdere applicaties op de esp32 moeten draaien, bevat deze library informatie over de secties waar de applicaties in het geheugen opgeslagen staan. (Espressif, z.d.)
protobuf-c	Library voor protocol buffers.
protocomm	Library voor protocol communicatie.
pthread	Library voor het gebruik van pthread.
sdmmc	Library voor het gebruik van Secure Digital and MultiMediaCard's.
smartconfig_ack	Library waarmee informatie verkregen kan worden als SSID en wachtwoorden van een verbonden Acces Point. (Espressif, z.d.)
soc	Library voor het gebruik van System on a Chip.
spiffs	Library voor het gebruik van SPIFFS file systemen.
spi_flash	Library dat zorgt voor het leven, schrijven, verwijderen en mappen van secties in het memory.
tcpip_adapter	Library voor het gebruik van TCP/IP.
tcp_transport	Library voor het versturen van informatie over TCP.
ulp	Library voor Ultra Low Power coprocessor programmeren. (Espressif, z.d.)
unity	Library voor het gebruik van unity aspecten.
vfs	Library voor het creëren van een Viruteel FileSystem.
wear_levelling	Library waarmee flash memory opgeslagen kan worden. (Karl, LofgrenRobert, NormanGregory, & ThelinAnil, 1991)
wifi_provisioning	Library dat aansturing en het gebruik maken van Wifi mogelijk maakt.
wpa_supplicant	Library dat ethernetprotocollen mogelijk maakt voor point-to-point lan netwerken. (Archlinux, 2018)
xtensa-debug-module	Library dat debugging op Xtensa processoren mogelijk maakt.

Appendix D – Xtensa LX6 Assembler

In deze bijlage zullen de assembler instructies beschreven worden die gebruikt kunnen worden op de Xtensa LX6 processor in de esp32.

Volgens de handleiding *"Xtensa Instruction Set Architecture (ISA)"* (ISA, 2010), kunnen de onderstaande instructies gebruikt worden op de Xtensa LX6 processor.

§D.1 Load instructions

INSTRUCTION	DESCRIPTION
L8UI	8-bit unsigned load (8-bit offset)
L16SI	16-bit signed load (8-bit shifted offset)
L16UI	16-bit unsigned load (8-bit shifted offset)
L32I	32-bit load (8-bit shifted offset)
L32R	32-bit load PC-relative (16-bit negative word offset)

§D.2 Store instructions

INSTRUCTION	DESCRIPTION
S8I	8-bit store (8-bit offset)
S16I	16-bit store (8-bit shifted offset)
S32I	32-bit store (8-bit shifted offset)

§D.3 Memory ordening instructions

INSTRUCTION	DESCRIPTION
MEMW	Order memory accesses before with memory access after
EXTW	Order all external effects before with all external effects after

§D.4 Jump, Call instructions

INSTRUCTION	DESCRIPTION
CALL0	Call subroutine, PC-relative
CALLX0	Call subroutine, address in register
RET	Unconditional jump, PC-relative
J	Unconditional jump, address in register
JX	Subroutine return—jump to return address. Used to return from a routine called by CALL0/CALLX0.

§D.5 Conditional Branch instructions

INSTRUCTION	DESCRIPTION
BALL	Branch if all of masked bits set
BNALL	Branch if not all of masked bits set
BANY	Branch if any of masked bits set
BNONE	Branch if none of masked bits set (All Clear)
BBC	Branch if bit clear
BBCI	Branch if bit clear immediate
BBS	Branch if bit set
BBSI	Branch if bit set immediate
BEQ	Branch if equal
BEQI	Branch if equal immediate
BEQZ	Branch if equal to zero
BNE	Branch if not equal
BNEI	Branch if not equal immediate
BNEZ	Branch if not equal to zero
BGE	Branch if greater than or equal
BGEI	Branch if greater than or equal immediate
BGEU	Branch if greater than or equal unsigned
BGEUI	Branch if greater than or equal unsigned immediate
BGEZ	Branch if greater than or equal to zero
BLT	Branch if less than
BLTI	Branch if less than immediate
BLTU	Branch if less than Unsigned
BLTUI	Branch if less than unsigned immediate
BLTZ	Branch if less than zero

§D.6 Move instructions

INSTRUCTION	DESCRIPTION
MOVI	Load register with 12-bit signed constant
MOVEQZ	Conditional move if zero
MOVGEZ	Conditional move if greater than or equal to zero
MOVLTZ	Conditional move if less than zero
MOVNEZ	Conditional move if non-zero

§D.7 Bitwise logical instructions

INSTRUCTION	DESCRIPTION	EXAMPLE
AND	Bitwise logical AND	AR[r] ← AR[s] and AR[t]
OR	Bitwise logical OR	AR[r] ← AR[s] or AR[t]
XOR	Bitwise logical exclusive OR	AR[r] ← AR[s] xor AR[t]

§D.8 Arithmetic instructions

INSTRUCTION	DESCRIPTION	EXAMPLE
ADDI	Add signed constant to register	AR[t] ← AR[s] + (imm87 24||imm8)
ADDMI	Add signed constant shifted by 8 to register	AR[t] ← AR[s] + (imm87 16||imm8||08)
ADD	Add two registers	AR[r] ← AR[s] + AR[t]
ADDX2	Add register to register shifted by 1	AR[r] ← (AR[s]30..0 || 0) + AR[t]
ADDX4	Add register to register shifted by 2	AR[r] ← (AR[s]29..0 || 02) + AR[t]
ADDX8	Add register to register shifted by 3	AR[r] ← (AR[s]28..0 || 03) + AR[t]
SUB	Subtract two registers	AR[r] ← AR[s] − AR[t]
SUBX2	Subtract register from register shifted by 1	AR[r] ← (AR[s]30..0 || 0) − AR[t]
SUBX4	Subtract register from register shifted by 2	AR[r] ← (AR[s]29..0 || 02) − AR[t]
SUBX8	Subtract register from register shifted by 3	AR[r] ← (AR[s]28..0 || 03) − AR[t]
NEG	Negate	AR[r] ← 0 − AR[t]
ABS	Absolute value	AR[r] ← if AR[s]31 then 0 − AR[s] else AR[s]

§D.9 Shift instructions

INSTRUCTION	DESCRIPTION
EXTUI	Extract unsigned field immediate Shifts right by 0..31 and ANDs with a mask of 1..16 ones The operation of this instruction when the number of mask bits exceeds the number of significant bits remaining after the shift is undefined and reserved for future use.
SRLI	Shift right logical immediate by 0..15 bit positions There is no SRLI for shifts ≥ 16; use EXTUI instead
SRAI	Shift right arithmetic immediate by 0..31 bit positions
SLLI	Shift left logical immediate by 1..31 bit positions (see page 525 for encoding of the immediate value).
SRC	Shift right combined (a funnel shift with shift amount from SAR) The two source registers are catenated, shifted, and the least significant 32 bits returned.
SLL	Shift left logical (Funnel shift AR[s] and 0 by shift amount from SAR)
SRL	Shift right logical (Funnel shift 0 and AR[s] by shift amount from SAR)
SRA	Shift right arithmetic (shift amount from SAR)
SSL	Set shift amount register (SAR) for shift left logical
SSR	Set shift amount register (SAR) for shift right logical This instruction differs from WSR to SAR in that only the five least significant bits of the register are used.
SSAI	Set shift amount register (SAR) immediate
SSA8B	Set shift amount register (SAR) for big-endian byte align The t field must be zero.
SSA8L	Set shift amount register (SAR) for little-endian byte align

§D.10 Processor control instructions

INSTRUCTION	DESCRIPTION
RSR	Read Special Register
WSR	Write Special Register
XSR	Exchange Special Register (combined RSR and WSR) Not present in T1030 and earlier processors
RUR	RUR reads 32 bits of TIE state into an address register.
WUR	WUR writes 32 bits to a TIE state register from an address register.
ISYNC	Instruction fetch synchronize: Waits for all previously fetched load, store, cache, and special register write instructions that affect instruction fetch to be performed before fetching the next instruction.
RSYNC	Instruction register synchronize: Waits for all previously fetched WSR and XSR instructions to be performed before interpreting the register fields of the next instruction. This operation is also performed as part of ISYNC
ESYNC	Register value synchronize: Waits for all previously fetched WSR and XSR instructions to be performed before the next instruction uses any register values. This operation is also performed as part of ISYNC and RSYNC.
DSYNC	Load/store synchronize: Waits for all previously fetched WSR and XSR instructions to be performed before interpreting the virtual address of the next load or store instruction. This operation is also performed as part of ISYNC, RSYNC, and ESYNC.
NOP	No operation

Appendix E – Makefile Errors

In deze appendix zullen de errors uitgewerkt worden die voor kwamen tijdens de realisatie van het Proof of Concept. Ook zal beschreven worden wat de oorzaak van de error is en wat gedaan is om deze op te lossen.

§E.1 Letters in objectnamen verdwenen

Tijdens de realisatie van het Proof of Concept deed zich bij de linker een vreemd probleem voor. Bij het zoeken naar de benodigde objecten verdwenen soms letters, waardoor de linker sommige objecten niet kon vinden. Zo zocht de linker bijvoorbeeld naar “dport_acess.o” in plaats van “dport_access.o”. De rede van dit probleem is dat waarschijnlijk de commando’s die uitgevoerd werden het maximale letteraantal overschreden. Deze commando’s bevatte namelijk zeer veel argumenten en paden naar verschillende bestanden. Om dit probleem te kunnen oplossen, zijn de benodigde componenten voor de basis aansturing van de esp32 samengevoegd tot één component “bmptk-core”. Hierdoor verminderde het aantal paden dat in het commando moest staan en verdwenen de letters niet meer.

§E.2 Multiple target pattern error

Tijdens de realisatie van het Proof of Concept was een van de eerste problemen waar tegenaan gestuit werd, de “Multiple target pattern error”. Dit probleem doet zich voor wanneer bij de dependensies van een target een ‘:’ voorkomt, zoals te zien in het onderstaande voorbeeld. De dubbelepunt geeft aan dat dit het einde is van de target die gespecificeerd wordt. Wanneer een dubbelepunt bij de dependensies komt de staan ziet GNU Make dit als het einde van een tweede (multiple) target. Hierbij is gekozen om dit op te lossen met een tussenstap. Binnen de target wordt een functie aangeroepen waarbij handmatig de dependensie gecontroleerd word.

```
Make: C:/user/project
        …
```

Code voorbeeld – dubbelepunt in target dependensie

§E.3 Missing separator

Het krijgen van een “Missing separator” error kan vaak veel onduidelijkheid geven. Meestal betekend dit dat ergens in de code een tab neergezet is, in plaats van vier spaties. Binnen defines moet namelijk gebruik gemaakt worden van spaties en binnen targets moeten tabs gebruikt worden. GNU Make is hier zeer gevoelig voor. Echter wordt deze error gegeven op de aanroep van de eerste functie. Hier kan zijn dat die functie een andere functie aan roept, die weer een andere functie aanroept, waar de fout in zit. Dit kan daardoor soms veel zoekwerk zijn als deze error gegeven wordt.

Appendix F – Cursus leerdoelen

Elke cursus die gegeven wordt heeft zijn eigen leerdoelen. Dit zijn onderwerpen die de student moet beheersen aan het eind van de betreffende cursus. In deze appendix zullen de leerdoelen uitgewerkt worden van de cursussen waarbij de nadruk op embedded programmeren ligt. De leerdoelen die beschreven worden zijn opgehaald uit de studentenhandleiding van de betreffende cursus.

Cursus	**Leerdoelen**
TICT-V1IDP-15	➢ Van een opdracht de context verkennen onder andere door het uitvoeren van literatuuronderzoek. ➢ Een probleem analyseren en in deelopdrachten opknippen. ➢ Op basis van beargumenteerde keuzes een oplossing uitwerken die in de context van de opdracht inzetbaar is. ➢ Testen uitvoeren om vast te stellen of de uitgewerkte oplossing aan de requirements voldoet. ➢ Relevante stakeholders identificeren en op grond hiervan een requirementsanalyse maken. ➢ Deelopdrachten planmatig en gestructureerd managen. ➢ Een ethisch aspect beschrijven en de voors of tegens kunnen beargumenteren. ➢ De samenwerking in het project tijdens teambijeenkomsten bespreken en afspraken hierover vastleggen. ➢ De oplossing zowel mondeling als schriftelijk presenteren aan de voor het product relevante doelgroep en daarbij laten zien of de gekozen oplossing in een brede context inzetbaar is. ➢ Onderbouwde feedback vragen, geven en ontvangen. ➢ Op basis van Belbin teamrollentest verwoorden welke rol hij/zij in het project wil bekleden. ➢ Op basis van bijvoorbeeld een kernkwaliteitenanalyse een eigen bijdragen in het project beschrijven met een STARR. ➢ Leerdoelen formuleren en het huidige niveau van de eigen professional skills beschrijven in een reflectieverslag.
TICT-V1OOPC-15	➢ Door middel van Git software code binnen halen en zelf geschreven code op een ordelijke manier op een eigen account publiceren. ➢ Door middel van een UML klasse diagram een decompositie maken van een te ontwikkelen applicatie met gebruik van associaties, aggregaties en overerving. ➢ Op basis van een UML klasse diagram of een andere vorm van specificatie C++ klassen schrijven met gebruik van private, public, inheritance en virtual. ➢ Een abstract data type met bijbehorende operatoren in C++ implementeren. ➢ De C++ taalconstructies defaults, const, reference, overloading, override, auto, en for(:) gebruiken. ➢ Een unit test schrijven en uitvoeren met gebruik van een unit test library. ➢ Een C++ klasse interface documenteren met behulp van Doxygen. ➢ De decorator en adapter patterns gebruiken.

	➢ Een cross-development systeem gebruiken om software te ontwikkelen voor een micro-controller. ➢ GPIO pinnen van een micro-controller gebruiken om eenvoudige inputs (schakelaars) en outputs (LEDs) te gebruiken. ➢ SPI en I2C gebruiken om peripherals te interfacen.
TICT-IPASS-15	➢ Een project ontwikkelen waarbij een of meerdere van de volgende onderwerpen in zit: o Digitale Techniek o Low-level data-communicatie o Electronica ➢ Algemene software Engineering ➢ Small-system-specifieke software engineering ➢ Een technisch probleem aanpakken op een kleine microcontroller ➢ Gebruik van een Operating System ➢ Zelf kiezen van een project met voldoende omvang ➢ Een Plan van Aanpak schrijven ➢ Gebruik maken van moderne en professionele technieken ➢ Een poster maken over het gerealiseerde project
TCTI-V2CPSE1-16	➢ Een make-based application building systeem gebruiken, begrijpen, en er kleine aanpassingen in aanbrengen. ➢ Een kort (tot ~ 20 regels) en eenvoudig (geen variabelen op de stack) stukje assembler schrijven in een specifieke assemblertaal. ➢ Vanuit assembler C++ code aan roepen, en vanuit C++ code assembler aan roepen, inclusief eenvoudge parameteroverdracht (in registers) ➢ Een kort (tot ~20 regels) stukje assembler doorgronden en verbeteringen in aanbrengen. ➢ De concurrency mechanismen begrijpen en gebruiken met behulp van een multithreading library. ➢ omgaan met in de cursus beschreven C++ en S.E. aspecten.
TCTI-V2THDE-16	➢ Op basis van een requirements architecture een solution architecture opstellen voor de software van een embedded systeem. ➢ De solution architecture omzetten naar een werkend embedded systeem. ➢ Code documenteren met gebruik van Doxygen. ➢ Gebruik maken van een versiebeheersysteem. ➢ Samenwerken met het team en de docenten. ➢ Verantwoordelijkheid tonen voor het functioneren van het team en de voortgang van het project. ➢ Een rapport schrijven. ➢ Onderzoek doen naar de implementatie van concurrency mechanismen m.b.v. de faciliteiten van een concreet operating system. ➢ Een planning van de taken en een werkverdeling maken.

TCTI-V2MRB-14	In deze cursus wordt kennis gemaakt met hardware en technieken om hardware te koppelen. Kennis wordt opgebouwd over de volgende onderwerpen: ➢ sensoren (karakteristieken, interfacing, fouten) ➢ elektronische componenten (basiscomponenten, OpAmps) ➢ filteren en signaalprocessing (analoog/digitaal, Fourier transformatie) ➢ besturingscomponenten (FPGA, PLC, DSP, GPU) ➢ motoren (typen motoren en eigenschappen) ➢ voedingen (spanning, stroom, vermogen, eigenschappen voeding)
TCTI-R2D2-17	➢ Zelfstandig nieuwe oplossingen (algoritmen) bedenken, ontwikkelen en testen rekening houdend met (soft real-time) performance en resource management binnen een gedistribueerd computer systeem. ➢ Een ontwerp voor een soft-/hardware oplossing onderbouwen en analyseren en deze vastleggen d.m.v. toepasselijke verslagleggingsmethodes. ➢ Actuatoren, sensoren, image processing technieken, computer vision algoritmen en AI-technieken inzetten als onderdeel van een gedistribueerd systeem. ➢ Afspraken tussen teams en professionals vastleggen en uitvoeren. ➢ Code beheren in een code versioning system gebruikmakend van een professionele workflow mechanisme inclusief branching en release management om zodanig het eigen werk te integreren met het werk van anderen. ➢ (Pro)Actief samenwerken met zowel zijn eigen als andere teams, mede door vroegtijdig acties te ondernemen bij problemen/uitlopende planning/gebrekkige samenwerking of andere problemen waardoor het project risico loopt. ➢ Een kritisch houding aannemen, door zowel het eigen werk(en) als het werk en de houding van teamleden op een methodische wijze te evalueren. ➢ Bij een gegeven opdracht en/of doelstelling relevante, precieze, consistente en functionele hoofd- en deelvragen afbakenen. ➢ Voor het beantwoorden van de deelvragen een passende en haalbare aanpak beschrijven, waarbij, indien van toepassing, wordt ingegaan op de procedure, participanten, data-verzameling (en meetinstrumenten) en data-analyse. ➢ Relevante (wetenschappelijke) bronnen vinden en evalueren op bruikbaarheid ten behoeve van het verkrijgen van inzicht in actuele visies/modellen/technologie binnen het desbetreffende domein. ➢ Een theoretisch kader te schrijven dat inzicht geeft in actuele visies/modellen/technologie binnen het desbetreffende domein. ➢ Een (taalkundig) verzorgd, gestructureerd onderzoeksrapport schrijven dat valide conclusies trekt op basis van de gevonden resultaten en een adequaat advies geeft. ➢ Een oordeel geven over het onderzoeksrapport van anderen. ➢ Een (cross-platform) ontwikkel- en testomgeving opzetten voor zowel soft- als hardware

TCTI-VKATP-17	➢ De verschillende vormen van testen (smoke, unit, integration, regression, system) uitleggen en toepassen. ➢ Een testplan schrijven en implementeren, inclusief de benodigde simulators, stubs en drivers. ➢ Programmeren volgens het Functional Programming paradigma, met name voor het schrijven van functiegedreven tests. ➢ Programmeren volgens het Aspect-Oriented Programming paradigma, voor het schrijven van object-overstijgende code. ➢ De concepten map, reduce, reflection en meta-programming uitleggen en toepassen. ➢ Een koppeling maken tussen Python en C++, met name voor het testen van embedded systeem software. ➢ Een embedded systeem ontwikkel/test traject opzetten waarbij mock-ups en simulaties geleidelijk worden vervangen door echte implementatie code.

Appendix G - Begrippen

Begrip	***Definitie***
App Trace	Application Level Tracing
ASIO	Audio Stream Input/Output
AWS IoT	Amazon Web Services Iot Platform
COAP	Contrained Application Protocol
Daily Standup	Een dagelijkse bijeenkomst van de projectleden waarbij de werkzaamheden op elkaar af gestemd worden en een plan gemaakt wordt voor de komende 24 uur.
Elliptic-curve Diffie-Hellman	Een encryptie protocol, waarbij beide partijen een publieke en privé elliptische kromme hebben. Met deze elliptische kommen kunnen berichten versleuteld worden.
Elliptic Curve Digital Signature Algorithm	Een algoritme voor de versleuteling van digitale handtekeningen dat gebruik maakt van elliptische krommen.
FATFS	Generic FAT Filesystem Module
LWIP	Lightweight TCP/IP stack
MDNS	Multicast DNS
MQTT	Een communicatie protocol voor kleine sensoren en mobile apparaten. Dit protocol wordt veel gebruikt bij Internet of Things toepassingen. (MQTT, z.d.)
NVS	Non-Volatile Storage
Schuifregister	Een component waarmee het aantal GPIO pinnen van een microcontroller uitgebreid kan worden. Dit component werkt door middel van Binaire getallen die aangeven welke pin van het component stroom moet doorgeven.
Scrum	Scrum is een Agile manier van werken, waarbij multidisciplinaire teams werken aan de realisatie van een product. Hierbij wordt gewerkt in korte sprints die maximaal 4 weken duren.
SDMMC	Secure Digital and MultiMediaCard


HOGESCHOOL
UTRECHT