Verantwoordings- en Onderzoeksverslag

Stage bij EVAbits

Friedel Schön

Opdrachtgever: EVAbits
Begeleider: Jan Stegenga en Erik Krallen, Kobus Bijker
Hogeschool: Hanze Groningen
Opleiding: HBO-ICT - Software Engineering

21–06–2024

Titel	Verantwoordingsverslag en Onderzoek Stage EVAbits
Plaats en Datum	Assen, 21–06–2024
Auteur	Friedel Schön
Studentnummer	445455
E-mail	f.schon@st.hanze.nl
Bedrijf	EVAbits
Locatie	Skagerak 26 9723JR Groningen
Bedrijfsbegeleider	Jan Stegenga Erik Kallen
Opleiding	Software Engineering, HBO-ICT
School	Hanze Groningen
Locatie	Zernikeplein 11 9747 AS Groningen
Docentbegeleider	Kobus Bijker Nienke van der Spek

Voorwoord

Voor u ligt het verantwoordingsverslag van mijn stage bij EVAbits, uitgevoerd gedurende semester 3.2 als onderdeel van de derdejaarsstage HBO-ICT aan de Hanze Groningen. Tijdens deze stage heb ik gewerkt aan de EVAjig, een innovatief product van EVAbits. EVAbits begon als een start-up in opdrachtprogrammering in Bedum en is inmiddels uitgegroeid tot een gevestigd bedrijf in Euvelgunne, Groningen. Sinds twee jaar werkt EVAbits aan de ontwikkeling van hun eigen product, de EVAjig. Dankzij mijn interesse en kennis in Embedded Programming heb ik EVAbits kunnen ondersteunen en een bijdrage kunnen leveren aan de EVAjig.

Dit verslag geeft een overzicht van mijn ervaringen en werkzaamheden gedurende de afgelopen twintig weken. Ik heb een leuke tijd gehad waarin ik mijn kennis heb kunnen verbreden en mezelf verder heb kunnen ontwikkelen. Mijn inzet en kennis werden bij EVAbits gewaardeerd, wat heeft geleid tot een werkcontract na afronding van mijn stage. Ik ben zeer dankbaar voor deze kans.

Graag wil ik mijn begeleider van de Hanze, Kobus Bijker, bedanken voor zijn ondersteuning en waardevolle feedback gedurende mijn stageperiode.

Assen, 21–06–2024

Friedel Schön

Samenvatting

Field-Programmable Gate Arrays (FPGA's) bieden op flexibele wijze een breed scala aan functies. Door de mogelijkheid om hardware opnieuw te configureren en gemakkelijk aan te passen, zijn FPGA's geschikt voor complexe en flexibele toepassingen. EVAbits, een klein bedrijf gevestigd in Euvelgunne, Groningen, is gespecialiseerd in Embedded Engineering en Architectural Design. Het bedrijf heeft sinds twee jaar een eigen product ontwikkeld: de EVAjig, een testkast voor het testen van printplaatassemblages (PCBA's).

De EVAjig maakt het mogelijk verschillende PCBA's te testen zonder aanpassingen. Momenteel is de EVAjig gebaseerd op een microcontroller met beperkte communicatiemogelijkheden, wat resulteert in een beperkt aantal te testen PCBA's. EVAbits streeft ernaar dit probleem op te lossen door een FPGA te gebruiken. Daarom is de opdracht gegeven om een proof-of-concept te ontwikkelen.

Een belangrijk aspect van dit project is het onderzoek naar het meest geschikte communicatieprotocol tussen de FPGA en de computer. De opdrachtgever vraagt specifiek naar de voor- en nadelen van verschillende communicatiemiddelen op het gebied van snelheid, betrouwbaarheid en kosten. Op basis van dit onderzoek zijn aanbevelingen geformuleerd voor toekomstige werkzaamheden.

Begrippenlijst

- **Architecture Design**: Het ontwerpen van de structuur en organisatie van hardware of software systemen.
- **Baud**: De eenheid van symbolen per seconde in communicatie, gebruikt om de snelheid van dataoverdracht in asynchrone protocollen zoals UART te specificeren.
- **Clock-signaal**: Een elektrisch signaal dat de timing en snelheid van communicatie in een protocol bepaalt.
- **(Data-)bus**: Meerdere peripherals zijn parallel aangesloten aan de datalijnen en communiceren zo onderling of met de master. De bus kan verbreed worden, zonder de complexiteit te moeten verhogen.
- **(Data-)lijn** of **Signaal**: Een bekabeling van chip naar chip.
- **Datastructuur**: Een specifieke manier om gegevens te organiseren, te beheren en op te slaan zodat ze efficiënt kunnen worden gebruikt en gecommuniceerd.
- **Half-Duplex**: Er kan maar in een richting gecommuniceerd worden tegelijkertijd.
- **Full-Duplex**: Er kan tegelijk in twee richtingen gecommuniceerd worden.
- **Echo** of **Ping-Pong**: Een programma, die op een *ping*-request een *pong*-response terug geeft. Op deze manier is de connectie en snelheid te testen.
- **Efinix Titanium**: Een type FPGA geproduceerd door Efinix, bekend om zijn snelheid en energiezuinigheid.
- **Embedded Programming**: Het schrijven van software die direct op de hardware van een apparaat draait, vaak gebruikt in speciale hardware- en software-integraties.
- **EVAjig**: Een testapparaat van EVAbits ontworpen om het programmeren, instellen en testen van printplaatassemblages (PCBA) te vergemakkelijken.
- **Field-Programmable Gate Array (FPGA)**: Een type geïntegreerd circuit dat door de gebruiker geconfigureerd kan worden om specifieke taken uit te voeren.
- **Google's Protobuffers (Protobuf)**: Een efficiënt en taalneutraal data-serialisatieformaat ontwikkeld door Google, gebruikt voor het uitwisselen van gestructureerde informatie tussen verschillende programma's en systemen.
- **Hardware Description Language (HDL)**: Een programmeertaal waarmee de configuraties en functies van de logische blokken binnen een FPGA worden beschreven.
- **Hot-swapping**: De mogelijkheid om componenten toe te voegen of te verwijderen uit een systeem zonder dat het systeem hoeft te worden uitgeschakeld of opnieuw opgestart.
- **Implementatie**: Het proces van het realiseren van een ontwerp of plan door middel van hardware of software.
- **Inter-Integrated-Circuit (I^2C)**: Een eenvoudige, goedkope bus-systeem protocol dat een master en een of meer slaves gebruikt voor communicatie, met twee lijnen: een voor data en een voor clock.
- **Master-Slave**: Een communicatieprincipe waarbij een master-apparaat controle heeft over de communicatie met één of meer slave-apparaten die reageren op de commando's van de master.

- **Microcontroller**: Een klein computer op een chip met een processor, geheugen, en invoer-/uitvoerpoorten, gebruikt voor specifieke taken in elektronische systemen.
- **Node**: Een individueel apparaat of punt in een netwerk of bus-systeem dat deelneemt aan de communicatie.
- **Open Source**: Een programma is opgebouwd met geschreven code (de broncode). Ontwikkelaars kunnen ervoor kiezen deze broncode voor iedereen publiek toegankelijk te maken, iedereen kan de broncode lezen en verbeteren.
- **Peripheral**: Een controller (of computer) bestaat uit meerdere peripherals, dit zijn chips of elementen die een functie toevoegen. Voorbeeld: de processor op een microcontroller, de UART-naar-USB chip op een computer.
- **Printplaatassemblages (PCBA)**: Een geassembleerde printplaat waarop verschillende elektronische componenten zijn bevestigd.
- **RISC-V**: Een open standaard instructieset architectuur (ISA) gebaseerd op de principes van verminderde instructieset computing (RISC).
- **Real-Time Operating System (RTOS)**: Een besturingssysteem ontworpen om applicaties te draaien die binnen strikte tijdslimieten moeten reageren op gebeurtenissen, vaak gebruikt in embedded systemen.
- **Request-Response**: Een gebruiker stuurt een verzoek (request) naar een server, de server stuurt een antwoord (response) naar de gebruiker. Voorbeeld: De gebruiker vraagt `hanze.nl` op (request), de webserver stuurt de gegevens van de website (response).
- **Serial Peripheral Interface (SPI)**: Een flexibel bus-systeem protocol met drie signalen: clock, MOSI (Master Out, Slave In), en MISO (Master In, Slave Out), plus een optionele Chip-Select lijn.
- **Synchrone Communicatie**: Zender en ontvanger zijn gesynchroniseerd door een gezamenlijke clock. Er wordt alleen op de klokslag data verzonden of ontvangen.
- **Asynchrone Communicatie**: Zender en ontvanger zijn niet gesynchroniseerd door een gezamenlijke clock. Data kan onafhankelijk van elkaar verzonden worden.
- **Universal Asynchronous Receiver/Transmitter (UART)**: Een asynchroon protocol voor communicatie tussen twee nodes, waarbij twee datalijnen (RX en TX) gebruikt worden zonder clock, met een gedefinieerde baud-snelheid.
- **Versiebeheer**: Met versiebeheer worden veranderingen in de geschreven software opgeslagen in vorm van een versie, daardoor is het mogelijk naar oudere versies te kijken. Een groot voordeel is het efficiënte samenwerken.
- **Zephyr OS**: Een open-source real-time operating system ontwikkeld voor embedded apparaten, bekend om zijn lichtgewicht, schaalbare en veilige ontwerp.
- $\overline{\text{CS}}$: Een lijn boven een signaal-naam betekent, dat deze active-low is. Een lijn, die actief is als het laag is getrokken.

Inhoudsopgave

1 Hoofdstuk Introductie

De opdrachtgever EVAbits is een klein bedrijf gevestigd in Euvelgunne te Groningen. Het bedrijf is gespecialiseerd in het schrijven en onderhouden van software en firmware in de kader van Embedded Programming en Architecture Design. Het bedrijf is begonnen met opdrachtprogrammering, maar is ook sinds een aantal jaren bezig met hun eigen product: de EVAjig.

De EVAjig is een testapparaat dat speciaal is ontworpen om het proces van programmeren, instellen en testen van printplaatassemblages (PCBA) te vergemakkelijken. Dit apparaat is uitgerust met een dashboard en een online portaal. Een van de opvallende kenmerken van de EVAjig is de mogelijkheid om cassettes met PCBA's gemakkelijk te verwisselen. Dit betekent dat je met hetzelfde apparaat verschillende soorten PCB kunt ontwerpen en testen zonder veel tijd kwijt te zijn aan het aanpassen van de testopstelling. Daarnaast biedt de EVAjig een breed scala aan algemene functies, waardoor het een veelzijdig hulpmiddel is voor verschillende testbehoeften. Dit maakt het apparaat bijzonder geschikt voor bedrijven en technici die regelmatig met verschillende PCB ontwerpen werken.

Dit verantwoordingsverslag licht de opdracht toe en verantwoordt de gemaakte keuzes en de aanpak van de problemen. Het verslag begint met een toelichting op de EVAjig en het probleem wat de opdrachtgever ermee heeft. Vervolgens wordt het uitgevoerde onderzoek besproken in hoofdstuk vier, dit omvat een conclusie met aanbevelingen aan de opdrachtgever. In het vijfde hoofdstuk worden de HBO-I-competenties toegelicht en wordt beschreven hoe deze zijn bereikt. Er wordt afgesloten met 'Lessons Learned' met onderdelen die in de stage zijn geleerd of verbetert inclusief een korte zelfreflectie.

2 Hoofdstuk Opdrachtbeschrijving

In dit hoofdstuk wordt de stageopdracht uitgebreid beschreven en is opgedeeld in verschillende secties om een duidelijk overzicht te bieden. Eerst wordt in de sectie 'Probleem' het huidige probleem van de opdrachtgever toegelicht. Vervolgens wordt in de sectie 'Oplossing' de voorgestelde oplossing van EVAbits uitgelegd. De sectie 'Doel' behandelt de doelstellingen van de opdracht, inclusief de gemaakte afspraken en de scope van het project. Daarna worden in de sectie 'Stakeholders' de betrokken personen voorgesteld die een rol spelen in dit project. Tot slot worden in de sectie 'Beroepsproducten' de te ontwikkelen beroepsproducten gepresenteerd.

2.1 Probleem

De EVAjig is geïmplementeerd door middel van een microcontroller. Een microcontroller is een klein circuit dat als een complete computer op een chip fungeert. Het bevat alle essentiële componenten die een computer nodig heeft om te werken, zoals een processor (de 'hersenen' van de computer), geheugen (om gegevens op te slaan), en invoer-/uitvoerpoorten (om te communiceren met andere apparaten). Microcontrollers worden vaak gebruikt in elektronische apparaten en systemen om specifieke taken uit te voeren. Wat microcontrollers bijzonder maakt, is hun vermogen om geprogrammeerd te worden voor specifieke taken. Dit betekent dat je de microcontroller instructies kunt geven om bepaalde acties uit te voeren op basis van de informatie die het ontvangt van sensoren of andere invoerapparaten. Dit programmeerbare karakter maakt het flexibel en toepasbaar in een breed scala van toepassingen, van eenvoudige huishoudelijke apparaten tot complexe industriële machines.

Een nadeel van de huidige implementatie met een microcontroller is de beperkte flexibiliteit in communicatie, die vaak via externe chips wordt afgehandeld. Communicatieprotocollen werken op een hoge snelheid, vaak in de miljoenen bytes per second. Een microcontroller is niet snel genoeg om deze communicatie in software af te handelen en laat dit over aan een externe chip. Op de microcontroller is de processor vast aan de communicatie chipjes gekoppeld, zonder de mogelijkheid iets aan deze constructie te veranderen. Meestal is dit geen probleem voor de applicatie van een microcontroller, maar voor de EVAjig is dit een beperkend criteria door de brede schaal aan PCBA's die moeten getest kunnen worden.

2.2 Oplossing

EVAbits heeft dit probleem herkent en heeft een opdracht begonnen om deze beperkingen te verhelpen. Het bedrijf heeft kennis in field-programmable gate arrays (FPGA). Een FPGA is een type circuit door de gebruiker kan worden geconfigureerd om specifieke taken uit te voeren. Deze configuratie gebeurt door middel van het programmeren van logische blokken en verbindingsroutes binnen de FPGA. Hierdoor kan een FPGA worden aangepast aan een breed scala van toepassingen, van eenvoudige logische functies tot complexe digitale systemen.

De kern van een FPGA bestaat uit duizenden tot miljoenen programmeerbare logische blokken en een netwerk van interconnecties. Deze logische blokken kunnen worden geconfigureerd om elementaire functies zoals AND, OR en NOT-operaties uit te voeren. Door deze elementaire blokken te combineren kunnen complexere circuits gevormd worden. De interconnecties zorgen ervoor dat deze logische blokken op verschillende manieren met elkaar kunnen worden verbonden, afhankelijk van de vereisten van de toepassing.

Een van de grootste voordelen van een FPGA is zijn flexibiliteit. Omdat het apparaat kan worden geherprogrammeerd, kunnen ontwerpers hun hardwareontwerpen aanpassen en optimaliseren zonder dat nieuwe fysieke chips nodig zijn. Daarnaast kunnen FPGA's meerdere taken parallel uitvoeren, wat

resulteert in hoge verwerkingssnelheden. Dit maakt ze ideaal voor gebruik in toepassingen zoals signaalverwerking, communicatiesystemen en complexe rekenintensieve taken.

In vergelijking met een microcontroller biedt een FPGA veel meer flexibiliteit. Waar een microcontroller vast gekoppeld is aan bepaalde communicatiechips, kan een FPGA deze functionaliteit emuleren door de hardware intern opnieuw te configureren. Dit betekent dat een FPGA verschillende soorten communicatieprotocollen kan nabootsen zonder de beperkingen van vooraf vastgestelde hardware. Hierdoor kan een FPGA meerdere functies en communicatieprotocollen tegelijkertijd afhandelen, wat ideaal is voor complexe en veelzijdige toepassingen. De configuraties binnen de FPGA worden beschreven in een Hardware Description Language (HDL), waarin signalen en primitieve logica's worden gedefinieerd.

Voor deze specifieke oplossing heeft de opdrachtgever gekozen voor de Efinix Titanium FPGA. Efinix staat bekend om hun snelle en energiezuinige FPGA's, wat hen een geschikte keuze maakt voor de vereisten van de EVAjig.

2.3 Doel

Het doel van de opdrachtgever is om binnen twintig weken een proof-of-concept te implementeren voor de EVAjig, waarbij de huidige microcontroller gebaseerde implementatie wordt vergeleken met een nieuwe FPGA-gebaseerde oplossing. Deze proof-of-concept moet aantonen dat de FPGA-oplossing verbetering biedt in flexibiliteit en communicatiecapaciteit vergeleken met de huidige implementatie.

2.4 Onderzoek

De opdrachtgever vroeg om een onderzoek te doen over de FPGA. De concrete onderzoeksvraag is "*Wat zijn voor- en nadelen van communicatiemiddelen tussen FPGA en computer ten opzichte van snelheid, betrouwbaarheid en kosten?*"

De huidige implementatie op de microcontroller, de software op de microcontroller, hoort bijna onveranderd op de FPGA-implementatie te kunnen draaien. Wel moet de software met de FPGA kunnen praten, dit vereist een vast communicatieprotocol tussen de software en de firmware architectuur. Er zijn vele communicatieprotocollen beschikbaar met elk hun eigen use-case. In dit verslag zou onderzocht worden, welk protocol het beste geschikt is voor communicatie tussen de firmware en de software ten opzichte van snelheid, flexibiliteit en kosten.

2.5 Stakeholders

Bij de opdracht zijn meerdere personen betrokken, deze personen zijn als volgt

Naam	Organisatie	Rol
Jan Stegenga	EVAbits	Stagebegeleiding
Erik Kallen	EVAbits	Stagebegeleiding
Kobus Bijker	Hanzehogeschool	Stagebegeleiding / Beoordelaar
Nienke van der Spek	Hanzehogeschool	Beoordelaar Communicatie

3 Hoofdstuk Projectaanpak

3.1 Beroepsproducten

Concrete beroepsproducten die van EVAbits werden gevraagd zijn:

3.1.1 Demonstratie Zephyr OS op Efinix' Sapphire Core

Voor de huidige implementatie op de microcontroller maakt de opdrachtgever gebruik van Zephyr OS. Zephyr OS is een Real-Time Operating System (RTOS), speciaal ontworpen om applicaties te draaien waarbij het cruciaal is om binnen strikte tijdslimieten te reageren op gebeurtenissen. Dit type besturingssysteem wordt veel toegepast in embedded systemen, waar betrouwbare en tijdige reacties op externe signalen van essentieel belang zijn.

Zephyr OS is een open-source RTOS dat is geoptimaliseerd voor gebruik op embedded apparaten. Het systeem is ontworpen om lichtgewicht, schaalbaar en veilig te zijn, waardoor het geschikt is voor een breed scala aan toepassingen, van simpele sensoren tot complexe industriële besturingssystemen.

Efinix is een fabrikant van FPGA's en biedt een reeks handige core-modules aan. De Sapphire Core, een processor logicamodule geoptimaliseerd voor hun FPGA, maakt het mogelijk om minder primitieve operaties binnen Zephyr OS uit te voeren. Deze core maakt gebruik van de RISC-V architectuur. Om verwarring met Zephyr OS te voorkomen, wordt deze core ook wel aangeduid als de RISC-V core.

Requirements

- Maak een programma, die een LED laat knipperen op Zephyr OS, die op de RISC-V Core kan draaien met gebruik van de Efinix Handleidingen. De aansturen werkt direct op de GPIO-bus van de RISC-V Core.

3.1.2 Demonstratie communicatie tussen de FPGA en een computer via EVAbus (RISC-V Core)

EVAbits heeft hun eigen datastructuur bedacht voor het effectieve aansturen van de EVAjig. Dit is geïmplementeerd met behulp van Google's Protobuffers (Protobuf), Protobuf is een efficiënt en taalneutraal data-serialisatieformaat ontwikkeld door Google, het wordt gebruikt voor het uitwisselen van gestructureerde informatie tussen verschillende programma's en systemen. Het maakt gebruik van een eenvoudig en uitgebreid berichtformaat om data in een compacte, snelle en makkelijk te lezen manier te coderen en decoderen.

Requirements

- Maak een basic echo-programma op Zephyr OS, die een dummy-response terug geeft aan de computer.

3.1.3 Demonstratie communicatie tussen de FPGA en RISC-V Core (VHDL)

Vervolgens moet er gecommuniceerd worden tussen de primitieve FPGA Core en de RISC-V Core om data zoals configuratie uit te wisselen. Dit moet van beide kanten ontwikkeld worden.

Requirements

- Maak een programma, dat een LED laat knipperen op Zephyr OS. Laat Zephyr OS communiceren met de FPGA-implementatie om de LED te laten knipperen.

3.1.4 Implementatie van een driver in Zephyr OS voor het aansturen van de FPGA

Zephyr OS is modulair ontwikkeld om een brede schaal aan microcontrollers te ondersteunen. Hiervoor zijn er verschillende drivers beschikbaar. De FPGA hoort zelf geconfigureerd te worden, hiervoor is geen driver beschikbaar en deze dient zelf ontwikkeld te worden. EVAbits wil graag dat er een driver voor Zephyr OS wordt geschreven om makkelijk de FPGA aan te sturen.

Requirement

- Met de standaard-interface van Zephyr OS moet een peripheral kunnen worden aangestuurd.
- De driver moet de conventions van Zephyr volgen.

3.1.5 Onderzoeksrapport communicatieprotocollen

De communicatie tussen de FPGA en de RISC-V Core kan op verschillende manieren geïmplementeerd worden. Hiervoor zijn vele protocollen beschikbaar en toepasbaar. De opdrachtgever wil graag een vergelijking hebben tussen deze protocollen om beter te kunnen kiezen welke voor de EVAjig het beste is.

3.1.5.1 Requirements

- Het onderzoeksrapport wordt onderdeel van het verantwoordingsverslag.
- Er moeten volgende protocollen vergeleken worden: SPI, I^2C, UART, APB

3.2 Planning

Er werd meegewerkt aan de Agile werkwijze van de opdrachtgever. EVAbits heeft aan begin van elke dag een ‘to do’-meeting, waarin wordt besproken wie welke taak gaat uitvoeren en of er problemen zijn. Aan het einde van de dag wordt een ‘recap’ gedaan, waarop terug wordt gekeken naar de dag.

3.2.1 Fases

Aan het begin van de stage werden vijf fases gedefinieerd:

Fase	Betekenis
Oriëntatie	Opdracht onderzoeken, een planning maken en oriënteren in het bedrijf
Ontwerp	Eerste ontwerpen voor de producten maken, details onderzoeken
Onderzoek	Communicatieprotocollen onderzoeken, werkwijze begrijpen
Ontwikkeling	Producten ontwikkelen
Afronding	Producten en de stage afronden, verslag afronden en eindpresentatie voorbereiden

3.2.2 Werkzaamheden

In de eerste weken werd kennis gemaakt met de opdracht en de elementen erom heen. Vele dingen waren onbekend en nooit behandeld op school, dus deze moesten onderzocht worden door opzoeken in het internet en de kennis binnen het team. Vooral aandacht eisend was het concept *Zephyr OS* en de tooling en ontwikkeling van de FPGA.

Week	Fase	Werkzaamheid
7	Oriëntatie	Stageplanning geschreven, Project gedefinieerd

Week	Fase	Werkzaamheid
8	Oriëntatie	Onderzoek Zephyr OS
9	Oriëntatie	Onderzoek FPGA
10	Oriëntatie	Onderzoek FPGA (Efinix Tooling)
11	Ontwerp	Combineren FPGA en Zephyr OS
12	Ontwerp	Communicatie FPGA en Computer (UART)
13	Ontwerp	Onderzoek huidige implementatie IO-module en EVAbus
14	Ontwerp	Communicatie FPGA en Zephyr OS (UART) - Knipperende LED
15	Ontwerp	Onderzoek Zephyr OS Internals (hoe werken drivers? hoe worden deze geschreven?)
16	Ontwerp	Schrijven UART driver voor Zephyr OS
17	Ontwerp	Implementatie EVAbus via UART
18	Onderzoek	Onderzoek Communicatieprotocollen
19	Onderzoek	Onderzoek Communicatieprotocollen
20	Ontwikkeling	Communicatie FPGA en Zephyr OS (APB)
21	Ontwikkeling	Communicatie FPGA en Zephyr OS (SPI)
22	Ontwikkeling	Communicatie FPGA en Zephyr OS (SPI)
	Afronding	Conceptverslag afmaken
23	Afronding	Communicatie FPGA en Zephyr OS (SPI)
24	Afronding	Afmaken Verantwoordingsverslag
25	Afronding	Afmaken Verantwoordingsverslag en Eindpresentatie
26	Afronding	Eindpresentatie

4 Hoofdstuk Onderzoek

Dit hoofdstuk gaat over de onderzoek: *Wat zijn voor- en nadelen van communicatiemiddelen tussen FPGA en computer ten opzichte van snelheid, betrouwbaarheid en kosten?*

Eerst worden verschillende protocollen toegelicht en vergeleken. Daaruit volgt een conclusie.

4.1 Protocollen

In dit hoofdstuk worden verschillende protocollen toegelicht. Een protocol is een afspraak tussen apparaten hoe gecommuniceerd wordt. Een voorbeeld is een spreektaal, er is afgesproken hoe bepaalde dingen worden genoemd en dat er regels zijn in de taal. Als twee personen er niet over eens zijn, welke taal wordt gesproken of hoe zij de taal spreken, kunnen zij niet met elkaar communiceren. In communicatieprotocollen is gedefinieerd welke signalen (draadjes/lijnen) er zijn en hoe deze moeten worden aangestuurd; wanneer gaat stroom over welk signaal, hoeveel tijd zit er tussen etc.

Protocollen hebben meestal een clock-signaal, die de snelheid van communicatie bepaald. Bijvoorbeeld wanneer het signaal stijgend is (oftewel van een lage waarde naar een hoge waarde gaat) worden de data signalen op een bepaalde manier uitgelezen. Als het signaal van een hoge waarde naar een lage waarde gaat, is het een vallend signaal.

In de komenden beschrijvingen worden VDD en GND niet genoemd of meegeteld. VDD is de stroomvoering van desbetreffende chip en GND is de aarding van de chip.

Vele protocollen werken naar het bus- of master-slave principe. Bij een bus zijn meerdere apparaten, zoals chips en processoren, parallel aan de signalen aangesloten. Er is één master is gedefinieerd, die controle heeft over het gebeuren op de bus en één of meerdere slaves, die op aanvraag van de master kunnen lezen van de bus of schrijven op de bus. Een aangesloten chip zou voortaan *node* genoemd worden.

4.1.1 Inter-Integrated-Circuit Protocol

Inter-Integrated-Circuit (I^2C) is één van de meest makkelijke protocollen. I^2C is ontwikkeld in 1982 door Philips Semiconductor.[1(Chapter 1.1)] I^2C is een bus-systeem, die minstens één master en één slave eist. I^2C bestaat uit twee signalen, een voor de data die moet worden verstuurd en een voor de clock.[1(Chapter 2.1)]

Elke node heeft een adres waarop het wordt aangestuurd, er is alleen communicatie mogelijk tussen de master en een slave, het is niet mogelijk dat slaves onderling kunnen communiceren. Als standaard is 100kbit/s gedefinieerd als snelheid van communicatie, maar I^2C is compatibel tot een snelheid van 3,4Mbit/s genoemd *high speed mode*.[1(Chapter 1.2)]

Bedrading

Naam	Breedte	Functie
Serial Clock (SCL)	1	Dit is de clock voor de communicatie (master-driven)
Serial Data (SDA)	1	Over deze lijn worden command's of data gestuurd

Voordelen I^2C heeft een eenvoudige bedrading en is makkelijker en goedkoper te implementeren. Ook is hot-swapping met I^2C mogelijk, dat betekent dat slaves verwijderd of toegevoegd kunnen worden aan de bus zonder dat de bus uitgeschakeld of herstart moet worden.[2]

Nadelen Doordat er maar één datalijn is, is I^2C half-duplex. Half-duplex betekent, dat er of geschreven kan worden door de master of door een slave, er kan niet tegelijk gecommuniceerd worden.[1(Chapter 2.1)] Ook is I^2C relatief langzaam in vergelijking met bv. SPI.[1(Chapter 1.2),3]

4.1.2 Universal Asynchronous Receiver/Transmitter

Universal Asynchronous Receiver/Transmitter (UART) is een asynchroon, full-duplex protocol, dus er kan onafhankelijk tegelijk geschreven en gelezen worden. In tegenstelling tot de bovenstaande protocollen is UART geen bus-protocol en werkt alleen tussen twee nodes (point-to-point). UART heeft twee datalijnen: RX en TX, RX staat voor *Receive Text* en TX staat voor *Transmit Text* en heeft geen clock. Er is niet gedefinieerd, welke richting RX of TX data verstuurd door evenwaardige aard van nodes. Soms moet RX en TX gedraaid worden in de bedrading (één lijn van RX naar TX en één lijn van TX naar RX). Door beide nodes wordt een clock-snelheid gedefinieerd, dat wordt *Baud* genoemd. Er wordt simpelweg de data geschreven met de gedefinieerde snelheid inclusief een start- en stop-bit en optioneel een parity-bit voor fout-detectie.

Bedrading

Naam	Breedte	Functie
RX	1	Op deze lijn wordt data verzonden of ontvangen
TX	1	Op deze lijn wordt data verzonden of ontvangen

Voordelen Zoals bij I^2C heeft UART een eenvoudige bedrading, dus is het goedkoop te implementeren. Doordat het asynchroon is, zijn beide nodes evenwaardig en er moet niet één node als master gedefinieerd zijn.

Nadelen UART is niet geschikt voor meerdere nodes. Als er gecommuniceerd moet worden naar meerdere nodes, moeten er onderling UART-signals geplaatst worden. Door de asynchrone aard van UART, zou UART onstabiel worden bij hoge snelheden, dit verlaagd de maximale snelheid.

4.1.3 Serial Peripheral Interface

Het Serial Peripheral Interface (SPI) protocol is een bus-protocol dat, in vergelijking met zowel Inter-Integrated Circuit (I2C) als Universal Asynchronous Receiver/Transmitter (UART), complexer en flexibeler is. SPI maakt gebruik van drie hoofd-signalen: een clock, geleverd door de master, een MOSI-signaal (Master Out, Slave In) om data naar een slave te sturen, en een MISO-signaal (Master In, Slave Out) om data vanuit een slave naar de master te sturen. Daarnaast is er een optionele Chip-Select lijn om eenvoudig een specifieke node te selecteren.[4(Chapter 2.2)]

Bedrading

Naam	Breedte	Functie
Serial Clock (SCLK)	1	Dit is de clock voor de communicatie (master-driven)
Master-Out Slave-In (MOSI)	1	Data wordt verstuurd door de master naar de slaves
Master-In Slave-Out (MISO)	1	Data wordt verstuurd door een slave naar de master

Naam	Breedte	Functie
Chip Select ($\overline{CS}$)	n peripherals	Naar elke peripheral gaat een clip-select om te bepalen, wie geadresseerd is

Voordelen Een groot voordeel van SPI is de snelheid. SPI is niet gebonden aan een gedefinieerde snelheid en kan zo snel mogelijk communiceren. Ook is SPI full-duplex, dus is het zeer efficiënt bij een hoeveelheid data in beide richtingen. De implementatie is heel flexibel en geeft vrijheid voor verschillende toepassingen. Doordat er een gezamenlijke clock is, is SPI betrouwbaar bij hoge snelheden ten opzichte van UART.

Nadelen SPI ondersteunt full-duplex, waardoor de bedrading complexer is dan bij I^2C of UART. SPI vereist vier lijnen niet niet slechts twee zoals bij de net genoemde protocollen. Ook ondersteund SPI maar één master, die het clock-signaal maakt, daardoor is het moeilijk om verschillende peripherals aan te sluiten. I^2C ondersteund een bus-systeem, dus er kan een reeks nodes aangesloten worden zonder de complexiteit te moeten verhogen; SPI maakt gebruikt van de chip-select lijn, waardoor het niet mogelijk is een simpel bus te maken.

Ten opzichte van UART heeft SPI geen standaard manier om fouten te detecteren. Dit moet door de ontwikkelaar handmatig geïmplementeerd worden, wat de complexiteit kan verhogen.

4.1.4 Advanced Peripheral Bus

De Advanced Peripheral Bus (APB) is een onderdeel van de Advanced Microcontroller Bus Architecture (AMBA). AMBA is een specificatie voor communicatie tussen peripherals. APB is gedesignd voor lage datahoeveelheden en een simpel interface.[5(Chapter 1.1)] ABP is een synchrone, full-duplex communicatieprotocol.

Bedrading

Naam[5 table 2.1]	Breedte	Functie
PCLK	1	Dit is de clock voor de communicatie, kan apart gegenereed worden
$\overline{PRESETn}$	1	Hiermee kan elke peripheral gereset worden naar een bekende staat
PADDR	adres	Aansturing adresbus
PPROT	3	Beschermingstype, in hoeverre de data moet beschermd zijn
PNSE	1	Uitbreiding naar beschermingstype
PSELx	1	De aanvrager genereert een PSELx-signaal voor elke afhandelaar. PSELx geeft aan dat de afhandelaar is geselecteerd en dat een gegevensoverdracht vereist is
PENABLE	1	PENABLE geeft de tweede en volgende cycli van een communicatie aan
PWRITE	1	PWRITE geeft de schrijfrichting aan
PWDATA	data	Geeft de data aan, die wordt geschreven
PSTRB	data/8	PSTRB geeft aan welke bytelanes moeten worden bijgewerkt tijdens een schrijftransactie. Er is één schrijfstrobe voor elke 8 bits van de schrijfgegevensbus
PREADY	1	Gereed. PREADY wordt gebruikt om communicatie door de afhandelaar te verlengen.
PRDATA	data	In deze bus wordt data gelezen

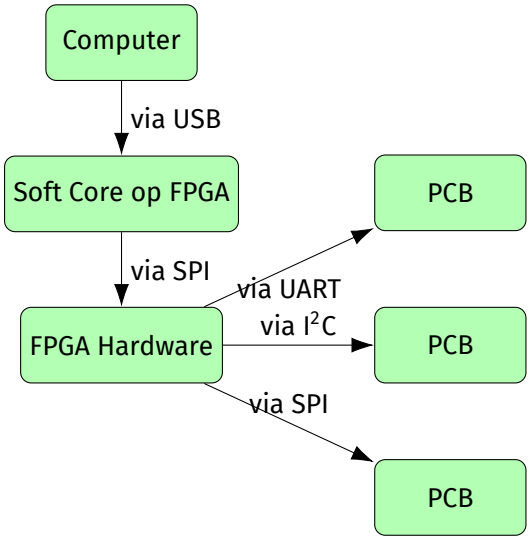
Naam[5 table 2.1]	Breedte	Functie
PSLVERR	0-1	Deze geeft aan, als er een fout is gevonden
PWAKEUP	1	Deze geeft aan, dat er iets wordt gedaan op de bus
PAUSER	u_request	Gebruikersaanvraagattribuut
PWUSER	u_data	Gebruikersschrijfgegevensattribuut
PRUSER	u_data	Gebruikersleesgegevensattribuut
PBUSER	u_response	Gebruikersreactieattribuut

Voordelen APB heeft meerdere voordelen ten opzichte van bovengenoemde protocollen. APB ondersteunt parallele communicatie, dus meerdere bits worden tegelijk verstuurd. De andere protocollen zijn puur serieel, dat betekent dat data bit-per-bit wordt verstuurd. Door de adres-architectuur is het eenvoudig om verschillende peripherals aan te spreken aan de bus, met SPI moet er een chip-select naar elke peripheral leggen om de zelfde uitkomst te behalen, en bij I^2C zou dit handmatig moeten worden geïmplementeerd.

Nadelen Ten opzichte van I^2C, SPI en UART is APB erg complex, doordat parallel data wordt verstuurd, moeten er vele lijnen worden aangesloten. Daardoor is APB meer geschikt voor een bus intern in bijvoorbeeld de FPGA dan voor communicatie onderling. En afsluitend is APB veel energie nodig, door de hoeveelheid lijnen met een lagere kloksnelheid en dus een lagere datasnelheid.

Een nadeel specifiek voor de Efinix FPGA is, dat de APB slecht gedocumenteerd staat en moeilijk is te implementeren. Er is ook niet veel informatie te vinden om een driver te ontwikkelen. Tijdens het onderzoek is het niet gelukt, APB te draaien op de FPGA van de opdrachtgever.

4.2 Toepassing op de EVAjig



In bovenstaande graaf wordt de communicatiestructuur binnen het systeem geïllustreerd. Het systeem omvat verschillende componenten die onderling communiceren om gegevens en commando's uit te wisselen. De componenten en communicatiepaden zijn als volgt:

- Computer: Dit fungeert als de bron van gegevens of commando's.
- Soft Core op FPGA: Dit is een verwerkingsunit die op een FPGA draait.

- FPGA Hardware: Fysieke hardwarecomponenten die worden gesimuleerd door de FPGA.

Verschillende PCB's worden getoond als bestemmingen voor gegevens of commando's vanuit de FPGA Hardware:

- PCB0: Ontvangt gegevens via het SPI (Serial Peripheral Interface) protocol.
- PCB1: Ontvangt gegevens via het I2C protocol.
- PCB2: Ontvangt gegevens via het UART protocol.

De communicatiepaden worden als volgt weergegeven:

- Computer naar Soft Core op FPGA: Gegevens of commando's worden via USB verzonden.
- Soft Core op FPGA naar FPGA Hardware: Communicatie vindt plaats via het SPI protocol.
- FPGA Hardware naar PCBs: De FPGA Hardware communiceert met de verschillende PCB's via hun respectievelijke communicatieprotocollen.

Deze diagram illustreert de complexe communicatiestructuur binnen het systeem, waarbij verschillende protocollen worden gebruikt om de juiste overdracht van gegevens te waarborgen.

4.3 Vergelijking van SPI, I2C en UART

Hieronder is een tabel opgenomen waarin de drie communicatieprotocollen SPI, I2C en UART worden vergeleken op verschillende criteria:

Criteria	SPI (Serial Peripheral Interface)	I2C (Inter-Integrated Circuit)	UART (Universal Asynchronous Receiver-Transmitter)	APB (Advanced Peripheral Bus)
Protocol Type	Synchroon or Asynchroon	Synchroon	Asynchroon	Asynchroon, Parallel
Complexiteit	3 (MOSI, MISO, SCK) + optional CS	2 (SDA, SCL)	2 (TX, RX)	Hoog
Clock Signal	Ja (master-driven)	Ja (master-driven)	Nee	Ja (extern)
Data Snelheid	tot tientallen Mbps	tot 3.4 Mbps	max. ~1 Mbps voor standaard UART	tot enkele tientallen Mbps
Multi-Master	Nee	Ja	Nee	Nee
Multi-Slave	Ja (met aparte CS lijnen per slave)	Ja (adressering)	Nee (point-to-point)	Ja
Foutdetectie	Geen ingebouwde foutdetectie	Basis (ACK/NACK)	Basis (pariteitsbit)	Ja (optioneel)
Bus	Nee	Ja	Nee	Ja
Kosten	Matig (extra lijnen en complexiteit)	Laag (weinig lijnen, eenvoudiger)	Laag (weinig lijnen, eenvoudig)	Hoog (vele datalijnen)
Energiekosten	Hoog (door continue clock)	Laag tot matig (lage complexiteit, bus-systeem)	Laag (eenvoudig en lage snelheid)	Hoog (parallele communicatie)
Geschikt voor	Hoge snelheid, korte afstand, en meerdere slaves	Lange afstand, meerdere masters/slaves	Eenvoudige, lange afstand, en point-to-point verbindingen	Korte afstand, veel data, onderling peripherals

4.3.1 Conclusie

Tijdens de stage is uitgebreid onderzocht naar voor- en nadelen van verschillende communicatieprotocollen. Dit protocol zou dienen als communicatie tussen FPGA en de soft-core op de FPGA op de EVAjig. Onderzocht zijn de protocollen I^2C, UART en SPI, met elk verschillende kenmerken en beperkingen.

I^2C heeft een eenvoudige bedrading is is flexibel met de mogelijkheid om meerdere peripherals aan een bus aan te sluiten. Wel is I^2C beperkt door een lagere snelheid en half-duplex.

UART is ook eenvoudig te bedragen en flexibel te gebruiken, wel is UART ongeschikt voor het communiceren met meerdere peripherals. Het is een asynchroon protocol, wat het moeilijk maakt, data op een hoge snelheid te versturen.

SPI bied wel een hoge snelheid met full-duplex communicatie met de consequentie, dat de complexiteit relatief hoog is. Ook is SPI slecht voor bus-systemen, doordat er een chip-select lijn naar elke node moet gaan.

APB is wel het meest complexe protocol, wel is goed geschikt voor de FPGA ten opzichte van snelheid en flexibiliteit. In het onderzoek is het protocol geanalyseerd en geïmplementeerd, maar door gebrek aan documentatie is dit niet gelukt. In overleg met de opdrachtgever is een alternatieve gevonden.

Op basis van dit onderzoek is theoretisch bekeken, welk protocol het meest geschikt is voor de EVAjig. Gezien dat het gekozen protocol de bottleneck zou zijn, is snelheid een prioriteit. Ook is een bus-mogelijkheid niet vereist, er wordt slechts tussen twee nodes (de FPGA en de soft-core) gecommuniceerd. SPI blijkt het meest geschikte communicatieprotocol te zijn voor de EVAjig, het bied de snelheid het flexibiliteit. Doordat alles op de FPGA draait en door de Hardware Description Language wordt beschreven is complexiteit geen probleem.

4.3.2 Aanbevelingen

Op basis van de conclusies van dit onderzoek, zijn er volgende aanbevelingen opgesteld om de communicatie op de EVAjig te optimaliseren:

- **Integratie van SPI in de FPGA-architectuur**: Het wordt aanbevolen om de FPGA te ontwikkelen met het SPI-protocol, hier is het makkelijk om de bestaande hardware clock te gebruiken en dus de FPGA-kant de master te maken. Dit biedt maximale flexibiliteit in communicatie met verschillende apparaten en systemen.
- **Ontwikkeling van een SPI-driver voor Zephyr OS**: Efinix biedt geen SPI-driver aan voor Zephyr OS. Om de integratie van SPI eenvoudig te laten verlopen, moet er een aangepaste SPI-driver worden ontwikkeld voor Zephyr OS. Op basis van de geleverde driver van Efinix en de handleidingen kan de driver voor Zephyr OS worden ontwikkelt. Het is belangrijk, de driver goed te documenteren, zowel voor de gebruiker, maar ook voor de ontwikkelaar om het gebruik eenvoudig te maken.
- **Optimalisatie van communicatieprotocol**: Door de aard van de FPGA, kan er een hoge datasnelheid toegepast worden. Maak een test met meerderde kloksnelheden, om de meest geschikte snelheid te vinden. De snelheid bepaald ook te betrouwbaarheid van communicatie.
- **Optimalisatie door APB**: APB blijkt een goed geschikt protocol te zijn voor de FPGA, wel is er nog een gebrek aan documentatie in Efinix' handleidingen. Er wordt aanbevolen om dieper onderzoek naar de werkwijze te doen naar APB in plaats van SPI.

Door deze aanbevelingen kan EVAbits de functionaliteit van de EVAjig verbeteren, als er gebruik gemaakt wordt van een FPGA.

5 Hoofdstuk HBO-I competenties

Tijdens de stageperiode zijn er planningen gemaakt om de werkzaamheden gestructureerd aan te pakken. Dit hoofdstuk beschrijft hoe de HBO-I competenties zijn toegepast en ontwikkeld gedurende deze periode.

5.1 Onderzoeken

In de oriëntatie- en onderzoekfase werd uitgebreid onderzoek verricht naar de meest geschikte communicatieprotocollen voor de toepassing op de FPGA. Dit omvatte het vergelijken van de snelheid, betrouwbaarheid en kosten van verschillende protocollen zoals SPI, UART, I^2C en APB. Daarnaast werd onderzocht hoe de bestaande microcontroller-gebaseerde implementatie kon worden overgezet naar een FPGA-implementatie.

5.2 Projectmatig Werken

Bij de opdrachtgever is een agile werkwijze gebruikt, waarbij iteratieve sessies hielpen om de status in de gaten te houden en aan te passen waar nodig. Aan het begin van de stage werd een duidelijke fasering gedefinieerd, die als leidraad diende voor het gehele project. Dit gestructureerde proces zorgde voor een overzichtelijke en efficiënte werkwijze, waarbij elke fase van het project zorgvuldig werd gepland en uitgevoerd. Bij het onderzoek zijn problemen opgekomen betreffend APB, deze problemen zijn projectmatig en in overleg met de opdrachtgever opgelost.

5.3 Analyseren

Een grondige analyse van de huidige implementatie van het systeem bracht diverse verbeterpunten aan het licht. Deze analyse omvatte het bekijken van de beperkingen en de mogelijkheden voor optimalisatie van de bestaande microcontroller-oplossing. De resultaten van deze analyse werden gebruikt om de nieuwe FPGA-implementatie te verbeteren, met als doel de functionaliteit en efficiëntie te verhogen. De protocollen werden geanalyseerd om het meest geschikte protocol te gebruiken.

5.4 Ontwerpen

Op basis van de uitslag uit de analysefase werden nieuwe implementaties ontworpen voor de FPGA. Het ontwerpen van deze nieuwe configuraties vereiste een diepgaand begrip van de FPGA-architectuur en de specificaties van de te gebruiken communicatieprotocollen. De ontwerpactiviteiten zorgden ervoor dat de nieuwe oplossing zowel flexibel als efficiënt was, met verbeterde communicatiecapaciteiten.

5.5 Realiseren

De nieuwe ontwerpen werden vervolgens geïmplementeerd en getest op de FPGA om de functionaliteit te verifiëren. Deze testfase was cruciaal om te bevestigen dat de theoretische ontwerpen in de praktijk werkten zoals gepland. Eventuele problemen die tijdens de tests naar voren kwamen, werden geanalyseerd en opgelost.

5.6 Adviseren

Op basis van de nieuwe implementatie werden aanbevelingen gedaan voor verdere optimalisaties en verbeterpunten. Deze aanbevelingen omvatten suggesties voor toekomstige verbeteringen en

uitbreidingen van de EVAjig. Door deze aanbevelingen kon EVAbits verder bouwen op de behaalde resultaten en de ontwikkeling van hun product continu verbeteren.

5.7 Schriftelijke Vaardigheden

Gedurende de stage werden diverse documentaties opgesteld, waaronder technische documenten en dit verantwoordingsverslag. Deze documenten dienden als waardevolle hulpmiddelen voor zowel de opdrachtgever als de Hanze. De geschreven stukken werden zorgvuldig samengesteld om de voortgang, bevindingen en aanbevelingen van het project helder en gedetailleerd weer te geven. Deze schriftelijke vaardigheden waren essentieel voor de communicatie en documentatie van het project.

6 Hoofdstuk Lessons Learned

Tijdens mijn stage bij EVAbits heb ik veel geleerd over verschillende technieken en professionele en persoonlijke ontwikkeling. Ik zou in dit hoofdstuk dieper op deze punten ingaan.

6.1 Technische Vaardigheden

- **Diepgaande kennis van FPGA's** Gedurende mijn stage heb ik uitgebreide ervaring opgedaan met FPGA's, met name de RISC-V Core van Efinix. Het ontwikkelen van schema's met gebruik van een Hardware Description Language was een uitdaging zowel het tooling om de FPGA heen.
- **Real-Time Operating Systems (RTOS)** Het werken met Zephyr OS heeft me waardevolle ervaring gegeven in het gebruik van een RTOS voor embedded systemen. Ik heb geleerd hoe ik effectieve drivers kan schrijven en hoe ik de voordelen van een RTOS kan benutten om efficiënte programma's te schrijven.
- **Communicatieprotocollen** De vergelijking van verschillende communicatieprotocollen (SPI, I^2C, UART) en de implementatie van de protocollen hebben mij veel over het innerlijk van computercommunicatie laten zien. Ik heb geleerd hoe ik de voor- en nadelen van elk protocol kan vergelijken om de beste oplossing voor een specifieke toepassing te kiezen.

6.2 Professionele Ontwikkeling

- **Problemen oplossen** In het onderzoek ben ik vast gelopen bij het implementeren van APB. Ik heb geleerd om daarmee om te gaan en naar alternatieven te zoeken om dit probleem te verhelpen.
- **Onderzoek en Analyse** Het onderzoeken en analyseren van de communicatieprotocollen heeft de vaardigheid versterkt. Ik heb geleerd een bestaande implementatie te begrijpen en te analyseren om deze theoretisch toe te passen op een andere casus.
- **Documentatie en Schriftelijke Vaardigheden** Het opstellen van documentatie en rapporten, zoals dit stageverslag, heeft mijn schriftelijke vaardigheden verbeterd. Ik heb geleerd hoe ik technische informatie duidelijk en gestructureerd kan presenteren, wat essentieel is voor het delen van kennis.

6.3 Reflectie en Toekomstige Verbeteringen

Het werken binnen de ICT is nieuw geweest en verschilt erg van eerder gedaan werk. Ik heb belangrijke contacten gemaakt en dat er een baan is aangeboden bevestigd mijn plek in de Embedded Engineering-sector.

Kortom, de stage bij EVAbits was een waardevolle leerervaring die me goed heeft voorbereid op mijn toekomstige carrière. Ik ben dankbaar voor de kansen die ik heb gekregen en kijk uit naar de uitdagingen en kansen die voor me liggen.

Literatuurlijst

1. Wu, J. (2022). *A basic guide to I^2C*. https://www.ti.com/lit/an/sbaa565/sbaa565.pdf; Texas Instruments.
2. Nguyen, D. (Bobby). (2020). *I2C solutions for hot swap applications*. https://www.ti.com/lit/an/scpa058a/scpa058a.pdf; Texas Instruments.
3. *What is the maximum SPI clock speed?* (2021). https://community.silabs.com/s/article/what-is-the-maximum-spi-clock-speed-x; Silicon Labs.
4. *KeyStone architecture serial peripheral interface (SPI) user guide*. (2012). https://www.ti.com/lit/ug/sprugp2a/sprugp2a.pdf; Texas Instruments.
5. *AMBA APB protocol specification*. (2023). https://documentation-service.arm.com/static/63fe2c1356ea36189d4e79f3; Arm Ltd.