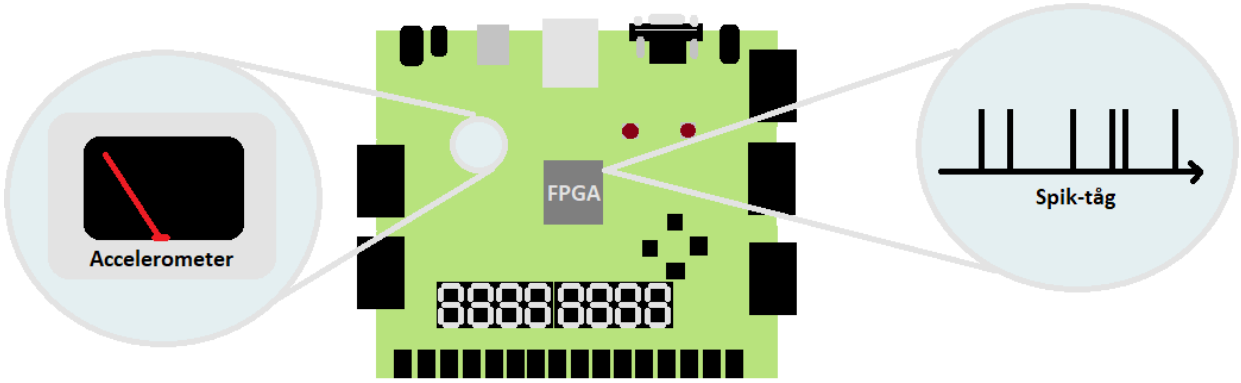
CHALMERS



Analog-till-Spiktåg-omvandlare

Spiktågs generering från en analog accelerations signal

Examensarbete inom Elektroteknik

PHILIP AXELSSON
CAROLINA WEBER

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg 2023
www.chalmers.se

Examensarbete 2023

Analog-till-Spiktåg-omvandlare

Spiktågs generering
från en analog accelerations signal

PHILIP AXELSSON
CAROLINA WEBER

CHALMERS

Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola
Göteborg 2023

Analog-till-Spiktåg-omvandlare
Spiktågs generering från en analog accelerations signal
PHILIP AXELSSON
CAROLINA WEBER

Handledare: Arne Linde, Institutionen för Data- och Informationsteknik
Handledare: Christo□er Weber, Wiretronic
Examinator: Lars Svensson, Institutionen för Data- och Informationsteknik

Examensarbete 2023
Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola
SE-412 96 Göteborg
Telefon +46 31 772 1000

Omslagsbild: Avbildning av testkortet med markerad accelerometer och fpga

Skriven i LaTeX
Göteborg 2023

Analog till spiktågs omvandlare
Spiktågs generering från en analog accelerations signal
PHILIP AXELSSON
CAROLINA WEBER
Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola

Sammanfattning

Projektet syftar till att modellera en omvandlare som genererar spiktåg baserat på accelerationsdata och efterliknar aktiviteten hos neuroner i människokroppen som detekterar acceleration. Målet var att skapa en energieffektiv och enkel modell för spiktågsgenerering som kan användas i samband med spikande neurala nätverk. Projektet omfattade flera centrala steg. Först genomfördes en grundlig litteraturgenomgång för att studera befintlig forskning inom området spiktågsgenerering och accelerationsdetektering. Baserat på den insamlade kunskapen och identifierade metoder valdes en lämplig metod för generering av spiktåg. Därefter utvecklades och implementerades omvandlaren. Det innefattade att utforma en modell för att omvandla accelerationsdata till spiktåg med hjälp av lämpliga inter-spike intervall. Omvandlaren är utformad för att vara anpassningsbar och möjliggör justering av inter-spike intervall (ISI) och tröskelnivåer för att kontrollera det genererade spiktågets firing-rate. Genom att ändra ISI-värden och tröskelnivåer kan man manipulera aktivitetsnivån hos spiktåget och säkerställa att det överensstämmer med önskat firing-rate. Slutligen presenteras resultaten och slutsatserna av projektet, inklusive en bedömning av omvandlarens förmåga att simulera neural aktivitet relaterad till acceleration. Projektets omfattning inkluderade även eventuella rekommendationer för vidareutveckling och tillämpningar av omvandlaren inom områden som neurala nätverk, robotik eller medicinsk teknik.

Nyckelord: Spiktåg, Inter-spike intervall, FPGA, SPI.

Förord

Denna rapport är skriven som ett Examensarbete för Högskoleingenjörs-programmet inom Elektroteknik på Chalmers Tekniska Högskola. Projektet är genomfört på institutionen för Data- och informationsteknik under våren 2023.

Vi skulle vilja rikta ett stort tack till vår handledare Arne Linde för hans vägledning och snabba återkopplingar. Vi vill även tacka vårt värdföretag Wiretronic som också givit värdefull insyn och vägledning under projektets gång. Slutligen vill vi även tacka examinatorn Lars Svensson för hans medverkan.

Philip Axelsson och Carolina Weber, Göteborg, Juni 2023

Akronymer

Nedan är listan över akronymer som har använts i denna avhandling listade i alfabetisk ordning:

AI	Artificial Intelligence
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuit
CS	Chip Select
FSM	Finite State Machine
ISI	Inter-Spike Interval
MOSI	Main Output Sub Input
MISO	Main Input Sub Output
SCK	System Clock
SDI	Subnode Digital Input
SDO	Subnode Digital Output
SNN	Spiking Neural Network
SPI	Serial Peripheral Interface
VHDL	(Very High-Speed Integrated Circuit) Hardware Description Language

Innehåll

Figurer

1
Inledning

Spikande neurala nätverk (SNN) har potentialen att vara mer energie ektiva än traditionella arti ciella neurala nätverk (ANN) på grund av deras händelsebaserade datahantering och beräkningar som endast utförs när det behövs. I SNN sker kommunikationen mellan neuroner genom diskreta händelser i form av aktionspotentialer (spikar), vilket innebär att beräkningar och dataöverföring sker vid speci ka tidpunkter när spikar inträ ar. Detta tillåter SNN att undvika onödiga beräkningar och minska den totala energiförbrukningen jämfört med ANN, där kontinuerliga matrisoperationer och beräkningar genomförs oavsett om det nns relevant data eller inte. SNN:s händelsebaserade tillvägagångssätt kan bidra till en mer e ektiv och sparsam användning av resurser i neurala nätverkstillämpningar [1].

För att kunna använda SNN behöver indata omvandlas till spiktåg innan den matas in i SNN. Detta innebär att kontinuerliga indata, till exempel sensoravläsningar eller signaler, måste översättas till sekvenser av spikar som representerar aktiviteten hos neuroner. Genom att omvandla indata till spiktåg kan SNN sedan bearbeta och analysera informationen på ett spikbaserat sätt för att utföra uppgifter som klassi cering, mönsterigenkänning eller beslutsfattande.

1.1 Bakgrund

De esta sensorer använder analoga mätvärden, såsom spänning eller strömnivåer. Mätvärdena ltrerar och samplas för att sedan omvandlas i en analog-till-digital-omvandlare. När mätvärdena omvandlats till ett digitalt format kan de användas i analys och datorberäkningar. Under denna omvandling utförs analyser och vidare datorberäkningar för varje mätsekvens, vilket förbrukar energi [2]. För att få ner energiförbrukningen kan beräkningar begränsas till att endast utföras då ett relevant mätvärde mäts, som till exempel när ett mätvärde ligger inom ett intervall eller når en tröskel. Beräkningarna kan även utföras på en optimerad analys och beräkningshårdvara som t.ex. en FPGA [2].

Analyser och vidare datorberäkningar kan i vissa fall vara baserade på maskininlärning så som ett Arti ciellt Neuralt Nätverk (ANN). Användandet av sådana beräkningar med mycket data kräver mycket minne och energi. Ett mer energie ektivt val av hantering av mätdata hade kunnat vara att istället utnyttja samma mekanism som nerver använder sig av i levande organismer. Man har upptäckt att nerver skickar i väg ett litet tåg av elektriska signaler som små spänningsspikar när man

reagerar på något [1]. När man t.ex. bränner sig skickas många spänningsspikar för att informera hjärnan om intensiteten av händelsen.

En neuromor sk metod tar inspiration från nervsystemet och hur det skickar signaler. Principen kan man nna i Spikande Neurala Nätverk (SNN) som använder sig av spikar istället för si ror som värden. Spikarna skickas i sekvenser och skapar tillsammans ett så kallat spiktåg. Intensiteten på en händelse avgör hur många spikar och hur tätt inpå varandra de förekommer. En intensiv impuls genererar ett tåg av spikar med hög frekvens, medan avsaknaden av en impuls genererar spikar med längre frekvens. Det betyder att när det är hög intensitet representeras med många spikar och låg intensitet med färre spikar per tidsenhet. Genom att läsa av tidsperioden mellan spikarna eller hur många spikar som inträ ar i ett spiktåg kan man läsa och skicka information [3].

E ektiviteten i ett SNN jämfört med ett traditionellt ANN ligger just i att energi endast förbrukas då en spik skickas genom nätet, medan nästan ingen energi förbrukas då inga spikar överförs mellan neuronerna [1]. På grund av att ett SNN är mer energie ektivt i jämförelse med ett traditionellt ANN har intresset ökat för att användas i s.k. 'edge computing', eller 'edge intelligence', applikationer. Applikationer där analyser sker vid mätpunkten, där det ofta inte är så lätt att få tillgång till energi, som t.ex. på en boj ute i Atlanten, eller en väderstation på en bergstopp. De ekologiska aspekter som rör projektet handlar i största mån om att minska energianvändningen och samtidigt ytta fram intelligent analys till fysiskt komplexa miljöer där tillgång på energi är begränsat [4]. Detta skulle inte bara innebära ekonomiska besparingar utan också besparingar av naturliga resurser och begränsa miljöpåverkan av dessa system. Dessvärre använder sig de esta sensorer av analoga värden som tidigare nämnts och därför behövs en analog till spike train omvandlare för att möjliggöra användandet av ett SNN i praktiken. i denna rapport undersöks möjligheten att generera ett spiktåg för inmatning till ett spikande neutalt näteverk på ett så energie ektivt sätt som möjligt. Detta med hjälp av en FPGA och en accelerometer.

Vid implementering av Arti ciell Intelligens uppkommer alltid särskilda etiska frågeställningar, som t.ex. kan konstgjord intelligens skada människor eller natur? Diskussionen om teknologiska framsteg och vetenskapliga upptäckter i samhället är det viktigt att inte förbise de baksidor och negativa följder som kan förekomma. Aldrig tidigare i mänsklighetens historia har tekniken utvecklats med en sådan anmärkningsvärd hastighet som nu. Det är ytterst viktigt att vara i framkant av denna utveckling och bromsa eller till och med avbryta om det behövs. Arti ciell intelligens uppfattas ännu utav många utanför forskarvärlden som något väldigt di ust vilket i sin tur gör det svårt att se dess många för-och nackdelar. Det räcker inte att endast använda sig av sina egna moralkompasser för att begränsa sig utan det krävs både lagar och restriktioner på internationell nivå för att balansera den snabba utvecklingen och anpassa den till dagens samhälle. Det är även viktigt att upprätthålla en kontinuerlig dialog med samhället och informera om de framsteg och de eventuella risker som kan förkomma. [5]

Huvudidén bakom SNN är att efterlikna biologiska processer och har stor potential att hjälpa till med integreringen av teknik med biologin som t.ex. proteser som är integrerade med kroppen och åstadkommer bättre liv för människor som behöver proteser. Det samhälleliga e ekterna av att använda denna sorts teknologi inom områden som proteser har potential till att berika handikappade människors vardag genom att underlätta och kanske till och med förbättra användandet av proteser [6].

1.2 Syfte

Syftet är att genom att mäta en människas rörelser med en accelerometer generera spiktåg som efterliknar de spiktåg som neuroner i människokroppen använder för att sända signaler. En enkel och energie ektiv omvandlare som kan användas för att generera spiktåg av analoga signaler från en accelerometer som sedan kan användas som indata till ett spikande neuralt nätverk.

1.3 Mål

Projektets mål är att utveckla en omvandlare för att konvertera analoga mätvärden från en accelerometer till spik-tåg genom användning av en FPGA (Field-Programmable Gate Array). Den genererade spik-tågsmodellen ska avspegla beteendet hos neuroner i människokroppen som detekterar acceleration.

En avgörande aspekt av målet är att förenkla modellen och tydligt demonstrera omvandlarens enkelhet och dess förmåga att generera spik-tåg baserat på accelerationsdata. Genom att betona och visa på omvandlarens användarvänlighet och praktiska tillämpbarhet strävar projektet efter att presentera en lättillgänglig och e ektiv lösning

1.4 Frågeställning

- Ett sätt att spara energi och skapa ett energie ektivt spikande neuralt nätverk är att anpassa och förenkla inmatningen. Finns det en enkel och energie ektiv metod som kan generera ett spiktåg utifrån analoga signaler?
- Är det möjligt att skapa ett sådant spiktåg med hjälp av en accelerometer och en FPGA?
- På vilket sätt kan omvandlaren anpassas för att generera spiktåg som efterliknar responsen hos neuroner i människokroppen som detekterar acceleration?

1.5 Avgränsningar

Projektet har vissa avgränsningar och krav som styr utformningen av omvandlaren. Pågrund av tidsbegränsingar utvecklas inget spikande neuralt nätverk i detta projekt. Dessutom är valet av accelerometer begränsat till en accelerometer som möjliggör enkel montering utan behov av lödning för att förenkla sammansättningen av

systemet.

Accelerationsdatan som används kommer inte ta hänsyn till vilken riktning som accelerationen utförs i. Accelerationens magnitud kommer endast tas i anspråk.

De genererade spik-tågen kommer endast vara en förenklad version av det verkliga beteendet hos neuroner. Neuroner har mycket detaljerat beteendemönster och en förenklad approximation av detta beteende kommer att användas för att efterlikna neuronerna.

2

Teknisk Bakgrund

I detta kapitel beskrivs den teori och tekniska bakgrund som ligger till grund för projektets utformande samt relevant information för att skapa förståelse för projektets innehåll och ändamål. Kapitlet är uppdelat under ett ertal underrubriker för att förenkla upplägget genom att separera teknisk bakgrund och fysiska komponenter till exempel. Vissa termer har inte översatts till svenska då en representativ översättning saknas.

2.1 FPGA

'Field Programmable gate array' är en typ av integrerad krets som kan kon gureras och programmeras för att utföra olika typer av digitala funktioner. Som namnet beskriver innehåller en FPGA en matris med programmerbara logiska block. Denna parallella arkitektur möjliggör processering av era uppgifter samtidigt. Istället för att bearbeta instruktioner sekventiellt som en traditionell processor gör, kan en FPGA bearbeta era instruktioner samtidigt genom att använda många parallella logiska block. Detta gör FPGAs till en kraftfull lösning för applikationer som kräver hög prestanda och snabb data bearbetning [7].

Möjligheten att kon gurera FPGAs till att bara använda de logiska block som behövs för en speci k applikation kan också minska energibehovet markant i jämförelse med traditionella processorer. E ektiviteten som erbjuds via högprestanda vid samtidigt lågt energibehov gör alltså FPGAs optimala för användning i komplicerade miljöer där energitillgången är begränsad [7].

Fördelen med FPGAs i förhållande till ASIC (Application Speci c Integrated Circuit) är kon gurerbarheten som en FPGA erbjuder. Produkten kan omkon gureras för att åtgärda eventuella buggar efter implementationen, och tiden det tar för en produkt att nå marknaden minskar kraftigt. Det beror på att det inte krävs att en speci k integrerad krets skapas, utan designen implementeras direkt på en FPGA.

För att programmera de logiska block som används i FPGAs används hårdvarubeskrivande språk så som till exempel VHDL eller Verilog [8] [9].

2.2 VHDL

VHDL står för '(Very High-Speed Integrated Circuit) Hardware Description Language' och är ett hårdvarubeskrivande språk som används för att beskriva digitala kretsar. Dessa kretsar kan sedan realiseras som grindmatriser. Grindmatrisernas parallella natur möjliggör programblock att köras parallellt med varandra vilket skiljer hårdvarubeskrivande språk från många andra programmeringsspråk där koden oftast exekveras sekvensiellt [8]. Genom att köra programblock parallellt exekveras alltså era olika delar av koden samtidigt.

2.3 SPI

SPI eller 'Serial Peripheral Interface' är ett seriekommunikationsprotokoll som används för att överföra data mellan olika digitala enheter som är anslutna till samma system. Det är ett synkront protokoll, vilket innebär att sändaren och mottagaren har en gemensam klocksignal som används för att bestämma när data ska skickas. Detta skapar ett e ektivt och snabbt sätt att överföra data men det kräver också att enheterna som ska kommunicera har samma klockfrekvens. [10]

SPI-protokollet används vanligtvis inom kommunikation mellan olika mikrokontroller eller andra inbyggda system där mycket snabb kommunikation mellan olika enheter är nödvändig. Spi klarar av mycket högre klockfrekvenser än liknande kommunikations protokoll så som I2C.

Protokollet kan vara antingen 3- eller 4-trådigt men i detta projekt används den lite vanligare 4-trådiga arkitekturen. Gränssnittet använder sig utav (MOSI) för att skicka data, (MISO) för att ta emot data, (SCK) för att styra klocksignalen och (CS) för att välja vilken enhet kommunikationen ska skickas mellan enligt gur 2.1 [10].

Figur 2.1: SPI kommunikations kon guering mellan huvudnod och subnod

SPI möjliggör kommunikation mellan den accelerometer som används i detta projekt och FPGA. Genom att välja en kombination av en 2-bits sel-ingång kan man manuellt välja vilket 8-bitars register som skall avläsas. Data hämtas på det valda

utgångsregistret 0x08, 0x09, 0x0A eller 0x0B. Där mätdata längs x-axeln på accelerometern läses av och kan väljas genom att ange '00' till sel-ingången. Mätdata från accelerometerns y-axel läses av på utgångsregister 0x09 och väljs manuellt genom att ange '01' på sel-ingången. Register 0x0A väljs manuellt med '10' på sel-ingången och mätadata läses av från accelerometerns z-axel. Det fjärde och sista utgångsregistret som går att välja på med sel-ingång '11' är status registret. Kontrollern skickar sedan vidare de adresser som mätdata skall läsas av och utför SPI transaktioner. För varje transaktion som skrivs eller hämtas 8-bits av data. Där den mest signi kanta biten är en teckenbit och resterande 7-bitar representerar ett accelerationvärde mellan 0 och 127.

2.4 Komponenter

Nedan presenteras de olika komponenter som används under projektgång.

2.4.1 ADXL362

ADXL362 är en extra energie ektiv accelerometer som drar mindre än 2A vid en 100 Hz data utmatningshastighet och 270 nA i rörelseutlöst väckningsläge. Den är utvecklad av Analog Devices, Inc. Accelerometern använder MEMS-teknik (Micro-Electro-Mechanical Systems) för att mäta acceleration och vibrationer i x-, y- och z-led. Den erbjuder två alternativa datautmatningsupplösningar: antingen 12 bitar eller 8 bitar som möjliggör för en e ektivare överföring av data genom enkelbyte-kommunikation, för applikationer där en lägre upplösning är tillräcklig [11].

Det nns också 3 olika mätområden tillgängliga på 2g 4g 8g där en upplösning på 1mg/LSB är tillgänglig för 2g-området. ADXL362 innehåller också många andra funktioner för e ektförminskning på system nivå [11].

2.4.2 Nexys A7

Nexys A7 är en utvecklingsplattform för inbyggda system och digital design som är utvecklad av Digilent, Inc. Företaget är specialiserade på att tillhandahålla hårdvaru- och mjukvaruverktyg för utbildning och forskning inom elektronik och dataveten-skap. Nexys A7 bygger på en Artix-7 FPGA från Xilinx som har en relativt hög prestanda, låg strömförbrukning och stora logikresurser. Det nns också en stor arsenal av olika portar som tex Ethernet, USB och VGA tillgängliga på platformen samt många olika sensorer, switchar och LED:s [12].

Figur 2.2: Nexys A7 utvecklingsplattform

2.5 Aktionspotential hos neuroner

Neuromor ska metoder tar inspiration från hjärnan och dess förmåga att skicka signaler. För att bidra med förståelse och kontext om hur ett system inspirerat av neuroners kommunikationsförmåga fungerar beskrivs i detta avsnitt hur aktionspotential hos neuroner i hjärnan fungerar på ett simpli erat sätt.

En neuron har ett viloläge och ett på-läge som inträ ar när tillräckligt med laddning når en tröskelnivå och elektriska pulser skickas iväg. Till en början när neuronen be nner sig i sitt viloläge är neuronen stabil. När sedan en sekvens med elektriska signaler skickas från andra neuroner i nervsystemet till neuronen i fråga uppstår en s.k. rumslig summering där era signaler kommer samtidigt och summeras i en knutpunkt till en och samma neuron. Tillsammans når signalerna neuronens tröskelnivå och en elektrisk signal skickas vidare till nästa neuron. Temporal summering kan också inträ a då signaler från en och samma neuron skickas i snabb följd efter varandra. Signalerna summeras då i knutpunkten och tillsammans når signalerna tröskelnivån och en aktionspotential kan utlösas. Signalen skickas sedan vidare till nästa neuron genom synapser och nästa neurons knutpunkt[13].

En aktionspotential är alltså en elektrisk puls som genereras när signalerna i synapsen nått över en tröskelnivå i neuroner. Aktionspotentialen aktiverar neuronen och en ny elektrisk signal skapas och skickas vidare till nästa neuron och så vidare. Detta bildar en kedjereaktion genom nervsystemet som gör att hjärnan kan reagera vid behov [14].

2.5.1 Proprioceptorer

i människokroppen finns det ca. 100 miljarder neuroner. De kategoriseras beroende på vad de reagerar på dvs. vad som gör att aktionspotential genereras. När t.ex. kroppen rör på sig är det en specifik sensorisk neuron som skapar den kedjereaktionen i nervsystemet som gör att en känsla av acceleration kan upplevas. Dessa neuroner finns i olika delar av människokroppen så som i muskler och leder och kallas för proprioceptorer. De känner av när kroppsdelar rör sig, spänningen i en muskel och de positioner kroppensdelar har. Proprioceptorerna reagerar när förändringar i muskelaktivitet inträffas och skickar signaler till hjärnan. Dessa neuroner är viktiga för att människan skall kunna uppfatta och anpassa sig efter de rörelser och acceleration som kroppen rör sig med. Informationen som proprioceptorerna skickar med sina elektriska pulser bidrar till att människan kan hålla sin balans, koordination och bilda sig en kroppsuppfattning. [15]

Varje neuron har en firing-rate, dvs. den frekvens med vilken en neuron avfyrar sina aktionspotential. Firing-rate går att mäta i antalet aktionspotential per sekund hos en neuron. Det är även möjligt att de nera neuroners aktivitet genom att relatera dess firing-rate till dess firing-rate intervall, som de nierar den högsta och lägsta firing-rate hos neuronen. För att överensstämma med tekniska sammanhang och uttrycksätt kommer spikar att ersätta termen aktionspotential framöver. Detta byte av terminologi syftar till att bättre överensstämma med den vanligare användningen av uttrycket inom det tekniska området.

Firing-rate-intervallet hos en proprioceptor kan variera en hel del, generellt ligger intervallet mellan några få spikar per sekund till 100 spikar per sekund [16].

2.6 Spiktåg

Tidsberoende signaler även kallade spikes eller spikar på svenska, kan användas för att emulera beteendet hos biologiska neuroners signaler. Detta beror på att spikar liknar de elektriska signaler som genereras av neuronerna i hjärnan. De är händelsebaserade avfyrning av aktionspotential när händelser inträffar. Ett spiktåg är alltså en sekvens av aktionspotential över tid och kan därför beskrivas som en stokastisk punktprocess. Det finns flera olika sätt att beskriva ett spiktåg, i form av antalet händelser som inträffar under en tidsperiod eller i form av tidsintervall mellan händelserna.

Enligt det först nämnda scenariot beskrivs spiktåg som en sekvens av spiktider, exempelvis t_1, t_2, ..., t_n (se figur 2.3 nedan), där n står för antalet spikar under en tidsperiod. Detta kan representeras med hjälp av frekvens, dvs. antalet spikar per tidsenhet. Detta kan beskriva intensiteten eller aktivitetsnivån i ett spiktåg. Det är även möjligt att representera spiktåg med hjälp av s.k. Spik-histogram. Där de grafiskt representeras av antalet spikar inom ett tidsintervall. Detta kan ge information om frekvensmönster och temporala strukturer i ett spiktåg.

Figur 2.3: spiktåg de nierat med spiktider

Enligt det sistnämnda scenariot beskrivs tiden mellan varje spik som inträ ar. Detta kan de nieras enligt ekvation 2.1.

$$T_i = t_{i+1} \quad t_i \tag{2.1}$$

där T_i står för tidsintervallet mellan tiden då en spik inträ ar, t_i , och tiden när nästa spik kommer att inträ a, t_{i+1} (se gur 2.4 nedan). Detta kallas för Inter-Spike Interval (ISI) och är den term som kommer användas i denna rapport för att beskriva ett spiktåg.

Figur 2.4: spiktåg de nierat med tidsintervall mellan spikarna

Det nns även många olika metoder för att generera ett spiktåg. Det är vanligt att använda sannolikhetsbaserade metoder så som en Poisson-process för att distribuera spikar över en tidsperiod. När denna metod används utgår man från att antalet händelser som sker inom ett tidsintervall följer en Poisson-fördelning [17].

2.6.1 Poisson-modell

I en Poisson-process avfyras spikar oberoende av varandra med en konstant genomsnittlig frekvens. Denna modell används ofta för att modellera slumpmässiga spikmönster. Tidsintervallet mellan avfyrade spikar (ISI) i ett Poisson 'spike trian' beror på en frekvens i antalet spikar per tidsenhet. Slumpmässiga ISI-värden genereras med exponentialfördelning, vilket gör att tidsintervallen varierar och är slumpmässigt korta eller långa. Även då ISI-värdet varierar kan ändå ett genomsnittligt ISI-värde beräknas utifrån fördelningens parametrar. Exponentialfördelningen har

en exponentiellt avtagande sannolikhet, dvs. att det är mer sannolikt att det genereras kortare tidsintervall mellan spikarna. För att generera nästa spik i spiktåget summeras det slumpmässiga ISI-värdet och den senaste spiktiden, se gur 2.5 nedan. Detta upprepas varje gång en nytt slumpmässig ISI-värde genererats och ett spiktåg byggs upp över en tidsaxel [17].

Figur 2.5: Generering av spiktåg med ISI

2.6.2 Intervall-Spik-kodning

Ett annat sätt att generera spiktåg är att använda sig av intervall-spik-kodning som till skillnad från Poisson modellen inte använder sig av slumpmässiga ISI-värden. Intervall-Spik kodning innebär att man använder tidsintervallen mellan avfyrade spikar (ISI) för att representera information. Metoden använder en förbestämd maximal och en förbestämd minimal ring-rate för att de niera ett intervall för ISI-värden. Detta intervall kallas ring-rate-intervall och kan används för att generera ett spiktåg genom att välja ett ISI-värde inom intervallet [18].

I vissa scenarion, när man strävar efter att efterlikna beteendet hos en speci k neuron, kan man dra inspiration från neuronets generella ring-rate-intervall för att generera ett liknande spiktåg. Genom att använda sig av intervall-spik kodning kan man utnyttja den maximala och minimala ring-raten hos neuronen för att generera ett spiktåg som ligger inom det förväntade aktivitetsområdet. En fördel med att använda intervall-spik kodning i dessa fall är att det är möjligt att anpassa den genererade aktiviteten till det förväntade beteendet hos neuronen. Detta kan vara särskilt användbart för modellering av aktiviteten hos biologiska neuroner eller för att säkerställa att det genererade spiktåget ligger inom speci ka gränser.

2.7 Spikande Neurala Nätverk

Ett spikande neuralt nätverk (SNN) är ett hjärninspirerat inlärningsramverk som kan användas för att åstadkomma en mer energie ektiv arti ciell intelligens (AI). En traditionell AI-metod för inlärning kräver ofta långa beräkningstider och därmed mycket energi. Där mätdata processas i stora matrisberäkningar som till och med kan vara erdimensionella dvs. tensorer som körs över GPU (Graphic Processing Unit). Beräkningarna sker även då en tensor är s.k. sparse , dvs. när det nns endast några få värden i en stor matris och resten är nollor. GPU-n laddar hela

tensorn eller matrisen i minnet och får köra igenom alla värden i matrisens celler. När ett ANN lär sig att känna igen en signal-typ, skickar man tusentals eller ännu er värden till nätet och ändrar variablerna i tensorerna till dess att nätet blir bra på att känna igen just det man vill. Metoden man använder i ANN är i huvudsak back och forward propagation samt gradient descent [19].

I jämförelse med ett traditionellt ANN har SNN potential att vara ett mer energieffektivt alternativ. Detta pga. att informaiton i from av diskreta spikar i tid endast överförs när det behövs, dvs. att med ett SNN kan den totala kommunikationsbelastningen minska och där med spara energi. I kontrast till att kommunikation kontinuerligt förekommer i ett traditionellt ANN. SNN är ett bra alternativ för att bearbeta händelsedriven data, där information endast uppdateras när en händelse inträ ar. Detta resulterar i att SNN reagerear på viktiga händelser och ignorerar irrelevant data [3]. Aktiviteten i ett SNN är oftast gles, vilket innebär att endast ett fåtal neuroner genererar spikar vid varje tidpunkt. Den glesa aktiviteten gör att färre beräkningar behöver utföras och färre signaler behöver överföras, vilket minskar energiförbrukningen. Till skillnad från ett ANN som i praktiken är en funktions approximation, dvs. att insignaler multipliceras med en variabel, summeras och körs in i en aktiveringsfunktion för att sedan skicka vidare ett nytt värde till nästa beräkningslager, sker aktiveringen av neuroner genom spikgenerering och överförning av spikar i ett SNN. I ett SNN används knutpunkter för att rikta och överföra spikar mellan neuroner. När en spik genereras i en neuron vid en knutpunkt, skickas den som en signal till den anslutna neuronern. Dessa neuroner tar emot spiken och kan reagera genom att generera egna spikar eller modi era sin aktivitet baserat på den inkommande signalen. Knutpunkterna i SNN möjliggör en mer diskret och selektiv kommunikation mellan neuroner, medan ANN förlitar sig på kontinuerliga beräkningar och överföring av signaler [19] [3].

Varje spik är en elektrisk potential som sedan klingar av efter spiken (se gur 2.6). När en serie av spikar, dvs. spiktåg används som input hinner inte alltid alla potentialer klinga av och därmed höjs den elektrisk potentialen i knutpunkten, som kommer till slut att nå en tröskelnivå likt mekanismen hos aktionspotenital hos neuroner i kroppen. Tröskelnivån är just det som ett SNN använder sig av för att träna [3].

Figur 2.6: Från spiktåg input genom ett förenklat SNN till output spiktåg

3

Metod

I detta kapitlet beskrivs de metoder som används för att nå resultatet som presenteras i kapitel 5. Projektet delades in i fem delmoment; förstudier, val av kretsar/-komponenter, utveckling av VHDL kod, veri ering och slutligen sammansättning av system.

3.1 Förstudier

Projektets utvecklingsområden innefattar era områden som är nya för utvecklarna, det kommer därför läggs mycket tid under de tidigare stadierna av projektet på att göra förstudier för att förstå dessa områden. Förstudierna innefattar undersökningar av existerande applikationer och upplägg av projektets omfång. Dessutom kommer det undersökas om det fanns förde nierade användargränsnitt att förhålla sig till under designfasen.

Under projektets gång utförs också förstudier då kunskaper inom områden som är nya för utvecklarna krävs, såsom SPI data överföring, omvandling av accelerations värde till avfyrningshastighet av spikar och accelerometer val.

3.2 Val av komponenter

För att välja komponenter bör först en studie göras av vad som nns tillgängligt på marknaden. Utifrån detta tas projektets krav i beaktning och resterande komponenters för- och nackdelar jämföras. Komponenternas kompatibilitet med varandra bör sedan tas i åtanke för ett slutligt val av komponenter.

3.3 Utveckling av omvandlare i VHDL

Projektet syftar till att utveckla en så energie ektiv och simpel omvandlare som möjligt. Olika metoder för att generera spik-tåg skall därför jämföras med hänsyn till projektets förutsättningar för att uppnå ett så optimalt resultat som möjligt. Efter att en metod för generering av spiktåg bestämts skall omvandlaren förenklas så mycket som möjligt.

3.3.1 Verktyg

Under utvecklingen av omvandlaren skall era olika mjukvaror användes för utveckling av olika delar av omvandlaren.

3.3.1.1 ModelSim

ModelSim är en programvara för simuleringen av digitala och analoga kretsar och användes för att validera och veri era dessa innan de implementerades på fysiska komponenter. Under utvecklingen av omvandlaren skall ModelSim användes i samband med Vivado för att på ett så e ektivt sätt som möjligt veri era omvandlarens funktionalitet utan att behöva syntesera hela designen samt för att validera e ektiviteten av omvandlaren i förhållande till tidigare versioner av koden [20].

3.3.1.2 Vivado

Vivado är en integrerad utvecklingsmiljö för design, implementation och veri kation av digitala kretsar på Xilinx-FPGA:er. Vivado innehåller också många olika verktyg för syntes, optimering av resursutnyttjande, systemintegration och validering. Implementationen av omvandlaren på hårdvaran och en stor del av utvecklingen skall göras i Vivado. Även Vivados inbyggda logikanalysator kommer användas under projektet för att analysera signaler inuti FPGA:n för felsökning samt för att analysera kommunikationen mellan accelerometern och FPGA:n via SPI-protokollet [21].

3.3.1.3 Matlab

Matlab kommer också användes i en begränsad utsträckning för att analysera mätresultat samt att plotta grafer för att tydligare illustrera resultat i kapitel 5 [22].

3.4 Veri ering

Veri eringen av omvandlarens funktionalitet kommer testas i era steg under utvecklingen. Designen delas upp och varje del testas först försig och sedan tillsammans med er delar tills hela omvandlaren fungerade i simuleringar i ModelSim. Efter simuleringsstadiet kommer omvandlaren syntetiseras och veri eras i Vivado. Omvandlarens design kommer även veri eras separat på FPGA:n genom att importera ett accelerations värde genom switcharna på Nexys A7 och mata ut spiktåg på en av ljusdioderna.

3.5 Sammansättning

Slutligen sammanställs hela systemet genom att samplingen av accelerometerdata skickas via SPI-protokollet till FPGA:n. Där datan skall tas emot, skalas med en konstant till användbara värden och skall sedan omvandlas till ett spiktåg genom omvandlaren. Detta spiktåg skall sedan representeras visuellt genom en ljusdiod på Nexys.

4

Genomförande

I detta kapitel redovisas genomförandet av projektet. En av huvudtankarna för projektet var att konstruera omvandlaren på ett så enkelt och energie ektivt sätt som möjligt. Genom att t.ex. använda sig av en FPGA för att konvertera accelerationsvärden till spiktåg. Det innebar även att ta hänsyn till de krav och avgränsningar som fastställdes för projektet genom att forma omvandlaren på ett sätt som uppfyller dessa krav.

Under förstudierna undersöktes enkla sätt att skapa ett energie ektivt SNN. Det framkom att bearbetning av inmatningsdata kan minska energiförbrukningen för ett SNN. Genom att t.ex. reducera antalet spikar som överförs eller använda sig av e ektiva representationsformer kan spiktågen anpassas för att minska energiförbrukningen.

4.1 Jämförande av metoder

Det nns era metoder för att generera spiktåg. Två av metoderna som uppfyllde projektets avgränsningar valdes ut för en mer detaljerad jämförelse. Dessa metoder granskades för att bedöma deras lämplighet för att generera spiktåg baserat på accelerationsvärden.

Inter-spike intervall kodning även kallat ISI-kodning använder sig av information om önskade tidsintervall mellan spikarna i ett spiktåg. Genom att ha förde nerade inter-spike intervall (ISI-värden) kan kontrollerbara spiktåg genereras, dvs. att man använder sig av förbestämda ring-rate-intervall. Det är alltså möjligt att justera spik-tåget efter önskat ring-rate-intervall och kan där av kontrolleras. ISI-kodning kan därför vara användbart för att generera ett spiktåg som skall efterlikna speci ka neuroners beteende. Ett känt ring-rate intervall kan alltså användas för att forma spiktåget om så önskas.

Poisson-processen används ofta för att generera spikar i spik-tåg. Det är en stokastisk process där tidsintervallet mellan spikarna följer en exponentiell fördelning. Genom att använda en önskad frekvens för spiktåget kan man beräkna medelvärdet för den exponentiella fördelningen. Slumpmässiga ISI-värden generaras baserad på detta medelvärde. Detta ger poisson-processen en mer slumpmässig karaktär som kan vara användbart i system vars syfte är att likna naturlig neural aktivitet.

Kontroll över intervallen mellan spikarna i ett spiktåg ger möjligheten att forma spikmönstret på ett mer precist sätt. När förde nerade ISI-värden används undviker man beräkningar baserade på slumpmässiga värden och kan på så sätt minska den beräkningsmässiga belastningen på hårdvaran. ISI-kodning kan vara ett mer energie ektivt val eftersom den tillåter en mer kontrollerad generering av spikarna och onödigt aktivitet kan undvikas.

Poisson-processen kan ses som en mer komplex process att implementera i jämförelse med ISI-kodning, eftersom poisson-processen bygger på att generera slumpmässiga ISI-värden. Detta kräver en slumpgenerator och beräkningar av exponentiella fördelningar. Detta göra att det kan vara svårt att uppnå exakt önskad frekvensen för spiktåget, vilket kan innebära högre aktivitet och därav större energiförbrukning.

Metoden som valdes för detta projekt är ISI-metoden eftersom den erbjuder en enkel och kontrollerad generering av spiktåg. Genom att använda denna metod kan användandet av slumpgeneratorer och algoritmer för att skapa exponentiellt fördelade ISI-värden undvikas. Detta kan potentiellt leda till minksad energiförbrukning och förenkla processen för att generera spiktåg.

4.2 Val av komponenter

Efter att förstudien av marknaden och vilka komponenter som krävdes för projektet hade genomförts valdes komponenter i enlighet med dessa speci kationer. Det uppstod dock era förhinder efter detta valet eftersom många komponenter hade extremt långa leveranstider.

4.2.1 Accelerometer, ADXL362

Efter studien av tillgängliga accelerometrar på marknaden valdes ADXL362 på grund av dess kompatibilitet med projektets krav [11]. Dessutom är många analoga sensorer så små att specialutrustning krävs för att löda fast dem på plattformar för användning. Därför valdes denna sensor också eftersom den fanns tillgänglig förmonterad på plattformar för användning.

4.2.2 FPGA, Nexys A7

Nexys A7 användes till en början som en utvecklingsplattform för initiala tester av VHDL-koden, samt för att förstå användningen av SPI-överföringsprotokollet [12]. Men efter att leveranstiden påverkade inköpet av komponenter och med tanke på att Nexys-kortet redan innehöll en ADXL362-accelerometer, användes den för att implementera den slutgiltiga designen av spiktågsomvandlaren.

4.2.3 SPI

SPI protokollet som beskrivet i teori användes eftersom att accelerometern ADXL362 har ett inbyggt spi-interface för data överföringen.

En enkel kontroller skapad av Daniel Llamocca [23] används i detta system för att kommunicera med accelerometern via SPI med Artix-7 FPGA:n på Nexys A7 testkortet enligt gur 4.1.

Figur 4.1: Från spiktåg input genom ett förenklat SNN till output spiktåg

Kontrollen använder sig av ett nite state machine (FSM) för att utfärda kommandon för att kon gurera ADXL362.

4.3 Utveckling av omvandlare i VHDL

Omvandlandet av accelerationsvärde till spiktåg baseras på Intervall Spik kodning och använder ISI för att representera information i spiktåget. Förbestämda ring-rate intervall avgör med vilket ISI-värde spikarna kommer avfyras. Varje accelerationvärde som överförts via SPI från accelerometern till FPGA:n korresponderar med en individuell ring-rate. Höga accelerationsvärden genererar höga ring-rates och där med korta ISI. Låga accelerationsvärden genererar därmot låga ring-rate och spikar avfyras med längre ISI, se gur 4.2.

4.3.1 Spiktåg-generering

Spikarna i spiktåget genereras/avfyras när en räknare nått en tröskelnivå, Tröskelnivån står för antalet klockcykler som korresponderar med det maximala tidsintervallet som kan förekomma mellan spikarna i spiktåget, dvs. maximala ISI-värdet.

Genom att variera ISI-värdet kan information skickas i spiktåget. Variation i ISI-värdet kan åstadkommas genom att låta en räknare tilldelas ett värde, data, som kan de neras enligt ekvation 4.1.

$$Data(a) = ISI(a) \tag{4.1}$$

Där a står för accelerationsvärdet som överförs via SPI från accelerometern och trös-kelnivån är lika med det maximala ISI-värdet. Beroende på vad accelerationsvärdet är kommer data(a) att bli större eller mindre. Ju högre data blir desto mindre ISI-värde blir det. Syftet med detta är att representera intensitet eller höga mätvärden med korta tidsintervall mellan spikarna i spiktåget och långa tidsintervall för låg in-tensitet. I gur 4.2 är det möjligt att se skillnaden på små och stora ISI-värden som skapas av höga respektive låga accelerationsvärden. Dvs. att de genererar ett spik-tåg med korta tidsintervall, minimalt ISI, när accelerationsvärdet är högt och låga accelerationsvärden genererar långa tidsintervall mellan spikarna, maximala ISI.

Figur 4.2: Spiktåg med minimalt ISI för högt accelerationsvärde och maximalt ISI för lågt accelerationsvärde

Det maximala tidsintervallet för avfyrning av en spik dvs. maximala ISI-värdet fö-rekommer då det minimala accelerationsvärdet används för att generera en spik. Genom beräkning av det det minimala värdet data(a_{min}) som kan de neras som tröskelnivån subtraherat med det maximala ISI-värdet, dvs. ISI(a_{min}) enligt ekva-tion 4.2 är det möjligt att generera en spik med maximalt ISI. Detta illustreras i gur 4.3 nedan.

$$Data(a_{min}) = ISI(a_{min}) = 0 \tag{4.2}$$

där ISI(a_{min}) är lika med det maximala ISI-värdet för spiktåget. Det maximala ISI-värdet är lika med tröskelnivån då det minimala acceleration värdet används. Räknaren kommer alltså vara tvungen att räkna hela vägen från noll till tröskelnivån innan en spik avfyras. Detta gör att låga accelerationsvärden representeras med längre tidsintervall mellan avfyrade spikar och illustreras i gur 4.3 nedan.

Figur 4.3: Minimala accelerationsvärdet ger maximala ISI

För höga accelerations värden genereras däremot kortare tidsintervall. Ett maximalt accelerations värde ger maximala värdet data(a_{max}) som är lika med tröskelnivån subtraherat med det minimala ISI-värdet, dvs. ISI(a_{max}) enligt ekvation 4.3.

$$Data(a_{max}) = ISI(a_{max}) \quad (4.3)$$

Där ISI (a_{max}) är det minimala ISI-värdet för spiktåget och räknaren kommer alltså att endast behöva räkna den kortaste tidsintervallet för att nå tröskelnivån, se gur 4.4 nedan.

Figur 4.4: Maximala accelerationsvärde ger minimala ISI

Efter varje avfyrad spik tilldelas räknaren värdet av data(a). Räknaren räknar uppåt med ett steg för varje klockcykel tills dess att räknarvärdet blivit lika med tröskelnivån. En ny spik avfyras och ett nytt data(a) värde tilldelas räknaren och processen börjar om igen.

4.3.2 Konvertering av accelerationsvärdet

Accelerometervärdet skickas via SPI från accelerometern till FPGA:n för att användas som inmatning till omvandlaren. När mätdata från accelerometern skickas via SPI med 8-bitar till omvandlaren på FPGA:n är den mest signi kanta biten en teckenbit och resterande 7-bitar representerar accelerationsvärdet. Omvandlaren tar emot det 8-bitar långa accelerationsvärdet som inmatning och tolkar den som ett osignerat värde, a, dvs. ett värde mellan 0-127. Detta görs då teckenbiten endast har som syfte att bära information åt vilket håll accelerationen är riktad. i detta fall är denna speci ka information irrelevant eftersom information rörande magnituden av acceleration endast är av intresse.

Det är nödvändigt att skalera mätdata från accelerometern för att anpassa värdena till den skala som används i omvandlaren. Skalan i omvandlaren är uttryckt i antal klockcykler, där varje klockcykel motsvarar 10 ns på grund av användningen av en klockfrekvens på 100 MHz för detta system. För att utföra skaleringen av accelerationsvärdet används den räta linjens ekvationen. Det skalerade accelerationsvärdet placeras sedan i variabeln data(a), där 'a' representerar accelerationsvärdet före skaleringen. Därefter används data(a) för att skapa tidintervallet mellan avfyrade spikar

i ett spiktåg. Accelerations värdet multipliceras alltså med en konstant för att linjärt skalera accelerationsvärdet till data(a), enligt ekvation 4.4.

$$Data(a) = k \quad a \tag{4.4}$$

där a står för accelerationsvärdet mellan 0-127 och k står för konstanten.

Konstanten k varierar enligt ekvation 4.5, som beror på vilken ring-rate intervall man önskar att sitt spiktåg skall använda sig av. När ett ring-rate intervall är bestämt medförs ett ISI-intervall som påverkar variabeln data(a). Variabeln data(a) beror enligt ekvation 4.1 på tröskelnivån samt ISI-värdet för det accelerationsvärde som matas in i omvandlaren.

$$k = \frac{data(a_{max}) \quad data(a_{min})}{a_{max} \quad a_{min}} = \frac{(\quad ISI\ (a_{max})) \quad (\quad ISI\ (a_{min}))}{a_{max} \quad a_{min}} \tag{4.5}$$

Där det maximala accelerationsvärdet är 127 och minimala accelerationsvärdet är 0. Data(min a) är enligt ekvation 4.2 lika med noll, eftersom tröskelnivån är lika med ISI(min a). Ekvation 4.5 kan därför förenklas till följande ekvation 4.6.

$$k = \frac{data(a_{max})}{data(a_{max}) \quad data(a_{min})} = \frac{ISI\ (a_{max})}{a_{max} \quad a_{min}} \tag{4.6}$$

I detta fall används samma ring-rate-intervall som proprioceptorer har, dvs. några få spikar per sekund upp till 100 spikar per sekund. Detta speci ka ring-rate-intervall används för att efterlika beteendet hos proprioceptorer.

I VHDL-kod representeras alla ISI-värden med antal klockcykler som korresponderar med ISI-värdena i nanosekunder. En klockcykel är 10ns eftersom en klock frekvens på 100MHz används i detta system.

Den minimala ring-raten, $_{min}$, för detta system är 10 spikar per sekund och det maximala ISI-värdet blir där av 100 ms enligt ekvation 4.7.

$$ISI_{max} = ISI\ (a_{min}) = \frac{1}{min} = \frac{1}{10 spikar=s} = 100000000 ns = 10000000 klockcykler \tag{4.7}$$

Det maximala ring rate, $_{max}$, för detta system är 100 spikar på sekund och det minimala ISI-värdet blir därav 10 ms enligt ekvation 4.8. Det maximala ISI är som tidigare nämnt lika med tröskelnivån.

$$ISI_{min} = ISI_{a_{max}} = \frac{1}{max} = \frac{1}{100 spikar/sekund} = 10000000 ns = 1000000 klockcykler \quad (4.8)$$

ISI-värdet minskar linjärt mot accelerationsvärdet. På motsatt sätt ökar värdet data(a) mot accelerationsvärdet, se gur 4.5. Maximala accelerationsvärdet, 127 ger alltså data(a_{max}) enligt ekvation 4.3 och kan beräknas för detta speci ka fall enligt ekvation 4.9 .

$$data(a_{max}) = ISI(a_{max}) = 10000000 - 1000000 = 9000000 klockcykler \quad (4.9)$$

Lutingen av den räta linjen, dvs. konstanten k kan beräknas till 70866 i detta fall, enligt ekvation 4.10.

$$k = \frac{ISI(a_{max})}{a_{max} - a_{min}} = \frac{10000000 - 1000000}{127 - 0} = \frac{9000000}{127} = 70866 \quad (4.10)$$

Figur 4.5: ISI och data(a) i relation till accelerationsvärdet

4.3.3 Simulering

Simuleringar av omvandlaren gjordes i programmet ModelSim, där ett 8-bitars accelerationsvärde manuellt valdes. I gur 4.6 nedan används det maximala accelerationsvärdet, dvs. 127. I simuleringarna används en spik bredd på 1 ms för att efterlikna aktionspotential i faktiska neuroner.

Simuleringen redovisar antalet spikar som avfyras över 1 sekund när ett maximalt accelerationsvärde matas in till omvandlaren. Vilket medför att data(a) som styr storleken på ISI i förhållande till tröskelnivån beräknas till sitt största möjliga värde.

Det är möjligt att beräkna antalet förväntade spikar över en period med antagandet att accelerationsvärdet inte förändras under denna period dvs. att samma ring-rate används genom hela spiktåget och samma ISI repeteras om och om igen.
I ekvation 4.11 görs antagandet att minimala ISI-värdet används i form av klock-cykler som beräknades i ekvation 4.8. Genom att först addera ISI och spik bredden är det möjligt att beräkna antalet spikar, , under t.ex. 1 sekund, enligt ekvation 4.11.

$$(a_{max}) = \frac{\text{perioden}}{ISI\ (a_{max}) + \text{spikbredd}} = \frac{100000000}{1100000} = 90;9\text{spikar} \quad 100\text{spikar=sek} \quad (4.11)$$

Figur 4.6: Simulering av spiktåg genererat utifrån minimala accelerationsvärdet

När det minimala accelerationvärdet matas in i omvandlaren kommer spikar avfyras med minimala ring-rate vilket i detta fall är 10 spikar/sekund. Detta kan beräknas enligt ekvation 4.12, där antagandet är att spikarnas bredd endast är 1 ms.

$$(a_{min}) = \frac{\text{period}}{ISI\ (a_{min}) + \text{spikbredd}} = \frac{100000000}{10100000} = 9:9\text{spikar} \quad 10\text{spikar=sek} \quad (4.12)$$

i gur 4.7 nedan är det möjligt att se hur spikarna avfyras med jämt mellanrum efter att reset har satt data(a) till noll och räknaren räknat med maximala ISI och av-fyrat första spiken. Därefter avfyras spikar med ett minimalt ISI över en tidsperiod på 1 sekund med en konstant ring-rate då det minimala accelerationsvädet används.

Figur 4.7: Simulering av spiktåg genererat utifån maximala accelerationsvärdet

4.4 Sammansättning av system

Accelerometern var redan sammankopplad med FPGA:n på testkortet och med hjälp av SPI-överföringen som sköts av kontrollern [23] var det möjligt att överföra mätdata från accelerometern till omvandlaren på FPGA:n. Omvandlaren implementerades med Vivado på FPGA:n. Accelerationsvärdena tas emot av omvandlaren via kontrollern och matas ut ur omvandlaren i form av ett spiktåg.

Accelerationsvärdet representeras binärt med 8 LEDS på testkortet. Med en logikanalysator i Vivado var det möjligt att följa beräkningen av data(a) beroende av accelerationsvärdet.

Det är även möjligt att välja från vilken axel som SPI överföringen skall hämta mätdata från. Genom att används switch 0-1 på testkortet kan kombinationen '00', '01' och '10' väljas för att hämta mätdata från x-axeln, y-axeln respektive z-axeln. Spiktåget skickades sedan ut på en LED på testkortet som blinkar i den takt som omvandlaren genererar spikar med. När ring-rate-intervallet mellan 10 spikar/sekund och 100 spikar/sekund användes med en spik bredd på endast 1 ms var det omöjligt för ögat att se skillnaden på ISI i spiktåget.

För att spiktåget skall vara möjligt att se via en blinkande LED på testkortet behöver maximala och minimala ISI-värdet uppskalas. Denna uppskalning är möjlig att göra genom att öka tröskelnivån och därmed öka det maximala ISI-värdet. På grund av detta kommer konstantenk att beräknas till ett nytt värde i förhållande till den nya tröskelnivån.

5

Resultat

Efter att systemet sammansatts resulterade det i en fungerande spiktåg omvandlare som är anpassad för accelerometern ADXL362 med kommunikation över SPI-protokollet. Den slutgiltiga designen är anpassningsbar beroende på vilken ring-rate intervall som önskas hos spiktåget. Detta är möjligt på grund av Inter-Spik-Intervall-kodning som används för att generera spikarna i spiktåget. Genom att anpassa variabler som tröskelnivån-, och konstanten k i omvandlaren kan man anpassa omvandlaren efter önskat ring-rate-intervall. VHDL-koden för omvandlaren nns i Appendix 1.

Variablerna i koden i Appendix 1 är anpassad efter det önskade ring-rate-intervallet som används för att efterlikna beteendet hos proprioceptorer. Det maximala accelerationsvärdet, 127, genererar ett spiktåg med ett ISI på 10 ms och minimala accelerationsvärdet, 0, genererar ett spiktåg med ISI på 100ms. För att anpassa omvandlaren efter de ring-rate-intervall som önskas bör man ta hänsyn till klock-frekvensen för systemet som är 100MHz. Varje klockcykel tar 10ns och alla variabler som anges i koden anges med antalet klockcykler. Detta betyder att alla ISI som förde neras i koden först bör omvandlas till nanosekunder och sedan divideras med 10ns vilket resulterar i antalet klockcykler. Antalet klockcykler anges sedan binärt i koden.

Nedan i gur 5.1 visas 4 sekunder av accelerometervärden samplad med 100Hz från accelerometern. Den nedre grafen visar spiktåget som genereras av omvandlaren. Accelerationsvärdena som representeras i den övre grafen används som in-data till omvandlaren. Firing-rate intervallet för detta spiktåg ligger mellan 20 spikar/sekund till 200 spikar/sekund vilket är det dubbla värdet av implementationen på FPGA:n, detta gjordes för att anpassa spiktåget för att bättre kunna illustrera resultatet över en förhållandevis lång sampling.

Figur 5.1: Accelerometer data och korresponderande spiktåg

Omvandlaren använder sig av den data som matas in via SPI från accelerometern till FPGA:n. Datan som skickas via SPI uppdateras efter 24ms per sampling, pågrund av detta representeras inte varje förändring som accelerometern uppfattar. Spiktåget uppdateras alltså efter varje sampling som tar 24ms för SPI:n att skicka vidare till FPGA:n för att omvandlas till ISI-värde. Omvandlaren klarar, i kombination med SPI-överföring, av att uppdateras med en frekvens av 24 ms.

6
Diskussion

I detta projekt implementeras en linjär omvandling från accelerationsvärden till beräkning av ISI för generering av spiktåg. Omvandlaren är utformad för att vara anpassningsbar och möjliggör att firing-rate intervall kan ställas in enligt önskemål.

Konkret innebär detta att när accelerationsvärden ökar, kommer ISI att minska linjärt i proportion till ökningen av accelerationsvärdet. Detta möjliggör en direkt och förutsägbar relation mellan accelerationsnivån och genereringen av spiktåg. Genom att använda en linjär omvandling kan man enkelt anpassa spiktågens egenskaper baserat på accelerationsvärden, vilket ger ökad flexibilitet och kontroll över genererade spiktåg.

Dessvärre förekommer det ingen direkt korrelation mellan antalet aktionspotentialer som avfyras av neuroner i människokroppen och accelerationsnivån. Detta betyder att omvandlaren utvecklad i detta projekt är en förenklad version av verkligheten. Den använder en linjär omvandling för att generera spiktåg baserat på accelerationsvärden, vilket inte exakt återspeglar det komplexa beteendet hos neuroner i människokroppen[25]. Det är viktigt att vara medveten om att omvandlaren är en approximation och att den inte tar hänsyn till alla detaljer och varierande faktorer som kan påverka neuronernas avfyrningsmönster. Trots detta kan omvandlaren vara användbar för att simulera en övergripande respons på acceleration och skapa en förenklad modell för spiktågsgenerering.

På grund av leveranstider implementerades designen bara på Nexys A7 brädet med den inbyggda accelerometern vilket också leder till en mindre e□ektiv produkt. Om istället omvandlaren implementerats på en mer energie□ektiv FPGA med tillräcklig beräkningskraft och en separat ADXL362 hade en mer e□ektiv lösning skapats.

SPI-protokollet är inte heller en speciellt energie□ektiv lösning då dess avkodning och protokoll kod just nu tar en betydande del av FPGA:ns resurser. Det finns förmodligen mer e□ektiva lösningar för att till exempel öka dataöverförings hastigheten men på grund av att den valda accelerometern redan innehöll ett inbyggt SPI-interface användes detta.

7

Slutsats och förslag till vidareutveckling

Valet av Inter-Spike-Interval (ISI) kodning som metod för att generera spiktåg möjliggjorde en enkel linjär omvandling som utvecklats i detta projekt. Genom att använda Inter-Spike-Interval kodning istället för en Poisson-process underlättades implementationen av koden i VHDL på en FPGA. ISI-kodning innebär att spikar genereras med hjälp av förutbestämda tidsintervall mellan dem, vilket gör att det kan implementeras som en enklare och deterministisk process. Detta gör att konstruktionen av spiktågs genereringen blir mer direkt och mindre komplex, vilket förenklar implementationen av koden i VHDL.

För att efterlikna aktionspotentialen hos neuroner i människokroppen som detekterar acceleration, används en firing-rate intervall inspirerat av det generella firing-rate intervall som proprioceptorer använder för att skicka signaler. Genom att studera och analysera det typiska firing-rate intervallet hos dessa neuroner kan en förenklad referensskala etableras för att omvandla accelerationsvärden till motsvarande firing-rate intervall.

Denna anpassningsbara funktion gör det möjligt att skapa spiktåg med olika aktivitetsnivåer och tidsmönster beroende på specifika behov och krav. Detta ger användaren större flexibilitet och kontroll över genereringen av spiktåg. Det möjliggör anpassning till olika applikationer och experimentella scenarier där önskade firing-rate intervall kan variera. Därigenom kan omvandlaren vara ett användbart verktyg inom områden som neurala nätverk, biofysik och neurovetenskap där simulering av neural aktivitet är av intresse. Det är dock viktigt att notera att omvandlaren är en förenklad version av verkligheten och erbjuder en övergripande simulering av responsen på acceleration genom spiktågsgenerering.

För vidareutveckling för att förbättra energie□ektiviteten hade det varit möjligt att använda en FPGA med lägre energiförbrukning. Den befintliga FPGA:n valdes för att underlätta testning och implementation av koden under utvecklingsfasen eftersom FPGA:n var monterad på ett testkort. Detta beslut baserades på att göra utvecklingsprocessen smidigare, även om det innebar att energie□ektiviteten inte var optimal. Genom att använda den tillgängliga FPGA:n på Nexys A7 testkortet kunde projektet göra snabbare framsteg och implementationen av koden kunde ske mer e□ektivt.

Ytterligare vidareutveckling skulle kunna vara implementationen av ett Faltnings filter eller ett simpelt SNN för användning av det genererade spiktåget.

Litteraturförteckning

[1] J. V. Stone, Principles of Neural Information Theory: Computational Neuroscience and Metabolic E□ciency. USA: Sebtel Press, 2018. [Online]

[2] J. Fowers, G. Brown, P. Cooke, and G. Stitt, ”A Performance and Energy Comparison of FPGAs, GPUs, and Multicores for Sliding-Window Applications,” In Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Association for Computing Machinery, Monterey, CA, USA, 2012, pp. 47–56. ISBN 9781450311557.

[3] M. Fee (MIT OpenCourseWare), “8: Spike Trains - Intro to Neural Computation”, YouTube, Jun. 29, 2020. [Video] Tillgänglig: https://www.youtube.com/watch?v=osYGG7TKcz8&list=PLUl4u3cNGP61I4aI5T6OaFfRK2gihjiMm&index=8&ab_channel=MITOpenCourseWare

[4] R. Liu, F. Bixo, ”Analysering av Energie□ektiviteten för Träning av Spikande Neuronnät”, Kungliga Tekniska Högskolan, Stockholm, Sverige, TRITA-EECS-EX-2022:488, 2022. [Online] Tillgänglig: http://kth.diva-portal.org/smash/get/diva2:1702463/FULLTEXT01.pdf Hämtad: 2023-03-15

[5] Safe.ai. ”AI Risk.” 2023, [Online] Tillgänglig: https://www.safe.ai/ai-risk Hämtad: 2023-06-15

[6] C. Gea, N. Kasabov, Z. Liuc, J. Yang, ”A spiking neural network model for obstacle avoidance in simulated prosthetic vision”, Elsevier Inc., Mar. 2017, doi: 10.1016/j.ins.2017.03.006.

[7] T. Katakkar, ”Field Programmable Gate Arrays (FPGA),” EngineersGarage. [Online]. Tillgänglig: https://www.engineersgarage.com/field-programmable-gate-arrays-fpga/ Hämtad: 2023-04-13

[8] ”IEEE Standard for VHDL Language Reference Manual,” in IEEE Std 1076-2019, pp.1-673, 23 Dec. 2019, doi: 10.1109/IEEESTD.2019.8938196. [Online].

[9] "IEEE Standard Hardware Description Language Based on the Verilog(R) Hardware Description Language,"in IEEE Std 1364-1995, vol., no., pp. 1-688, 14 Oct. 1996. [Online]. Tillgänglig: https://doi.org/10.1109/IEEESTD.1996.81542. Hämtad: 2023-04-29

[10] P. Dhaker, ”Introduction to SPI Interface.” Analog Dialogue. [Online] Tillgänglig: https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html Hämtad: 2023-04-30

[11] Analog Devices. ”ADXL362: Low Power, 3-Axis MEMS Accelerometer.” [Online] Tillgänglig: https://www.analog.com/media/en/technical-documentation/data-sheets/adxl362.pdf Hämtad: 2023-04-12

[12] Digilent. ”Nexys A7 Reference Manual.” [Online] Tillgänglig: https://digilent.com/reference/programmable-logic/nexys-a7/reference-manual?redirect=1 Hämtad: 2023-04-04

[13] LibreTexts. ”How Neurons Communicate - Signal Summation.” [Online]. Tillgänglig: https://bio.libretexts.org/Bookshelves/Introductory_and_General_Biology/Book%3A_General_Biology_(Boundless)/35%3A_The_Nervous_System/35.07%3A_How_Neurons_Communicate_-_Signal_Summation. Hämtad: 2023-05-03

[14] University of Texas Health Science Center at Houston. ”Resting Potentials and Action Potentials” [Online]. Tillgänglig: https://nba.uth.tmc.edu/neuroscience/m/s1/chapter01.html. Hämtad: 2023-05-04.

[15] J.L. Taylor, ”Proprioception,” in Encyclopedia of Neuroscience, edited by L.R. Squire, Academic Press, 2009, pp. 1143-1149, ISBN 9780080450469, doi: 10.1016/B978-008045046-9.01907-0.

[16] R.A. DiCaprio, C.P. Billimoria, and B.C. Ludwar, ”Information Rate and Spike-Timing Precision of Proprioceptive A erents,” in Journal of Neurophysiology, vol. 98, no. 3, pp. 1706-1717, 2007. [Online]. Tillgänglig: https://doi.org/10.1152/jn.00176.2007 Hämtad: 2023-05-04

[17] D. S. B. Hwang, ”Introduction to the Poisson Process,” [Online]. Tillgänglig: https://www.cns.nyu.edu/~david/handouts/poisson.pdf. Hämtad: 2023-05-04

[18] E. Forno, V. Fra, R. Pignari, E. Macii, and G. Urgese, ”Spike encoding techniques for IoT time-varying signals benchmarked on a neuromorphic classification task,” in Frontiers in Neuroscience, vol. 16, 2022, Article 999029. [Online]. Tillgänglig: https://www.frontiersin.org/articles/10.3389/fnins.2022.999029. Hämtad: 2023-05-10. doi: 10.3389/fnins.2022.999029.

[19] S. Russell, P. Norvig, ”Artificial Neural Networks”, i Artificial Intelligence: A Modern Approach, 3 uppl., Harlow, England: Person Education Limited, 2016, kap. 18.7, ss.727-737.

[20] Siemens EDA. ModelSim."[Online]. Tillgänglig: https://eda.sw.siemens.com/en-US/ic/modelsim/. Hämtad: 2023-07-19.

[21] Xilinx. "Vivado Design Suite."[Online]. Tillgänglig: https://www.xilinx.com/products/design-tools/vivado.html. Hämtad: 2023-07-19.

[22] MathWorks. MATLAB."[Online]. Tillgänglig: https://se.mathworks.com/products/matlab.html. Hämtad: 2023-07-19.

[23] P. Llamocca, ”ADXL362 Accelerometer (SPI) - Basic Control” Oakland University, [Online]. Tillgänglig: https://www.secs.oakland.edu/~llamocca/Courses/W17_ECE378/Notes%20-%20Unit%205.pdf. Hämtad: 2023-04-01

[24] M. Pelgrom, Analog to Digital Conversion, 3rd ed. 2017. Springer International Publishing, 2017. [Online] Tillgänglig: https://link-springer-com.proxy.lib.chalmers.se/book/10.1007/978-3-319-44971-5 Hämtad: 2023-02-20

[25] University of Texas Health Science Center at Houston. ”Synaptic Transmission in the Central Nervous System” [Online]. Tillgänglig: https://nba.uth.tmc.edu/neuroscience/m/s1/chapter06.html Hämtad: 2023-07-19.

A
Appendix 1

A.1 Källkodslistning

```
begin
    process(clock)
    begin


    if rising_edge(clock) then
            if reset ='0' then
                counter <= (others=>'0');           --counter sätts till 0
                spike <= '0';
                spike_tid <= (others => '0');           --ingen spike
            else
            --(ignorerar msb dvs. teckenbiten)
             a <= a_input(6 downto 0);  --accelerationsvärde mellan 0-127
             --multiplicerar accelerations värdet med konsten k
             --för att få antalet klockcykler korresponerar med
             --det accelerationsvärde som matas in
             data<= std_logic_vector(unsigned(a) * unsigned(k));     --data(a)=a*k

--När spike = 0 -> counter börjar räckna upp till max
                if spike = '0' then
                --counter börjar räkna när spiketiden tagit slut
                  counter<= std_logic_vector(unsigned (counter) +1);
                  --räcknar upp till tröskelnivån vilket är i detta fall
                end if;
--När counter räcknat upp till tröskelnivån kommer ett nytt accelerationsvärde
--att avgöra avståndet till nästa spike
                if counter = threshold then
            --tar ett data(a) värde = tröskelnivån - antal klockcykler fram till nästa spik
                    counter <= std_logic_vector(unsigned(data));
                    -- en spik skapas, med bestämd bredd
                    spike_tid<=spike_bredd;
                end if;
--Så länge spike_tiden != 0 minskas spiketiden och spike = 1;
                if spike_tid /= spike_slut then
                    spike_tid <= std_logic_vector(unsigned(spike_tid) - 1);
                    spike<= '1';                            --en spik avfyras
        else
            spike<='0';

                end if;
        end if;
        end if;
   end process;
spike_train<=spike; --skriver spikar ut på spike train utgång
end Behavioral;
```

Figur A.1: Källkoden för spiktågs genereringen

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige
www.chalmers.se

CHALMERS