Berufsmaturitätsarbeit 2019-2020

Hardware-Neuron

Tendai Rondof / Malik Schneider



Inhaltsverzeichnis

[II. Vorwort 4](#_Toc31498100)

[A. Tendai 4](#_Toc31498101)

[B. Malik 4](#_Toc31498102)

[C. Danksagungen 4](#_Toc31498103)

[III. Abstract 4](#_Toc31498104)

[IV. Einleitung 4](#_Toc31498105)

[V. Theorie 5](#_Toc31498106)

[A. Test Bedingungen 5](#_Toc31498107)

[1. Möglichkeiten 7](#_Toc31498108)

[2. Unmöglichkeiten 7](#_Toc31498109)

[B. Grundlegendes Model eines Neurons 8](#_Toc31498110)

[1. Biologisches Model 8](#_Toc31498111)

[2. Informatisches Model 9](#_Toc31498112)

[C. Unser Model 10](#_Toc31498113)

[D. Bedingungen/Einschränkungen 11](#_Toc31498114)

[E. Mathematischer Ansatz 11](#_Toc31498115)

[F. Aktivierung 12](#_Toc31498116)

[G. Fehlerberechnung 12](#_Toc31498117)

[H. Problematik von Theorie und Praxis 13](#_Toc31498118)

[I. Blockschaltbild 13](#_Toc31498119)

[VI. Praxis 14](#_Toc31498120)

[A. Ganzes Neuron 14](#_Toc31498121)

[B. Input 14](#_Toc31498122)

[1. DeltaApply 15](#_Toc31498123)

[C. Core 16](#_Toc31498124)

[D. Activation 16](#_Toc31498125)

[1. Vergleicher OPV 17](#_Toc31498126)

[E. Delta Calc 18](#_Toc31498127)

[VII. Gesamtschema von EAGLE 18](#_Toc31498128)

[VIII. Messungen 19](#_Toc31498129)

[A. Blockschaltbild des Versuchsaufbaus 19](#_Toc31498130)

[B. Programmierung des Mikrokontrollers 19](#_Toc31498131)

[IX. Fazit 19](#_Toc31498132)

[X. Formelverzeichnis 19](#_Toc31498133)

[XI. Abbildungsverzeichnis 19](#_Toc31498134)

[XII. 20](#_Toc31498135)

[XIII. Literaturverzeichnis 20](#_Toc31498136)

[XIV. Anhang 20](#_Toc31498137)

[A. Layout/Bestückungsplan von EAGLE 20](#_Toc31498138)

[B. Stückliste 20](#_Toc31498139)

# Vorwort

Seit einiger Zeit boomt das Interesse an künstlichen Intelligenzen und neue Ideen, wie sie umgesetzt werden können, sind hochgefragt. Wir erwarten nicht dass wir mit einem Neuron, dass aus einem elektronischen Schaltkreis besteht, das Rad neu erfinden, jedoch gelingt es uns vielleicht einen Weg zu beschreiten der bis jetzt noch niemand bedacht hat.

## Tendai

Ich habe dieses Thema ausgewählt, weil Künstliche Intelligenzen mich seit Jahren interessieren. Durch dieses Projekt kann ich den Wunsch ein Neuron auf Hardwarebasis zu erstellen, endlich erfüllen.

## Malik

Ich habe das Thema gewählt, weil darin zwei meiner grössten Interessensgebiete vereinigt sind: Die Elektronik und die Neurowissenschaft. Vor allem bei letzterem kann ich hiermit mein Wissen erweitern und vertiefen. In der Elektronik kann ich vor allem mein bereits bestehendes Wissen anwenden und trainieren, was meinem späteren Berufsleben ein wenig zugutekommen könnte.

## Danksagungen

Wir möchten unserem Abteilungsleiter, Oliver Schneider, in der MSW danken, welcher uns erlaubte an den Wochenenden in der MSW an unserer BMA zu arbeiten. Ohne seine Hilfe hätten wir unser Produkt sicher nicht rechtzeitig beenden können.

Daniel Büchler, Abteilungsleiter für die ersten beiden Lehrjahre der Elektroniker in der MSW hat uns freundlicherweise den Fräsbohrplotter zu Verfügung gestellt. Dieser Umstand hat unsere Arbeit ungeheuer beschleunigt.

# Abstract

# Einleitung

Das Ziel unserer BMA ist es, den Vorgang des Lernens mit greifbaren elektrischen Bauteilen zu simulieren. Dieser Aspekt allein hat schon eine gewisse Faszination an sich. Unter Machine-Learning versteht man etwas Komplexes, was sich auf Softwareebene abspielt. Für eine Künstliche Intelligenz sind riesige Mengen an Daten Nötig, um ihr etwas beizubringen. Aber wie so vieles, lässt sich auch eine Intelligenz, ob sie jetzt künstlich ist oder nicht, herunterbrechen auf ihre Elementarsten Teile. Beim Menschen sind es die Neuronen, bei der Maschine elektrische Schaltkreise. Unser Produkt ist ein wenig von beidem. Denn wir simulieren das menschliche Neuron mit einer elektrischen Schaltung.

Aufgrund unseres Lehrberufs, Elektroniker, setzten wir den Fokus auf die Funktionalität und nicht auf Biologische Richtigkeit. Das bedeutet, dass wir versuchten die Funktion elektronisch zu imitieren. Es fielen zwar viele der Biologischen Aspekte eines Neurons weg, jedoch bleibt der Kern des ganzen schlussendlich bestehen: das Lernen.

Um die Printplatte herzustellen, auf welcher wir unsere Schaltung aufgebaut haben, verwendeten wir den MSW-Internen Fräsbohrplotter, welcher uns die ersten Prototypen in Höchstgeschwindigkeit zur Verfügung stellen konnte.

Die Frage, die wir uns mit dieser BMA beantworten wollen, lautet:

Was ist eine Möglichkeit ein Neuron (Nervenzelle) mittels Elektronischen teilen zu realisieren und kann damit eine selbstlernende elektrische Schaltung (“Neuron”) erstellt werden?

Um diese Frage dreht sich die ganze Arbeit unserer BMA. Darum auch der Name Hardware-Neuron. Unser Endprodukt wird Ausschliesslich aus Elektronischer Hardware bestehen. Mit unserem Endprodukt werden wir uns den ersten Teil der Frage also selbst beantworten.

Zu Beginn des Hauptteils wird das Konzept zur Lösung des Problems in all ihren Details und mathematischen Formeln erklärt. Danach wird die Schaltung erklärt, die wir erstellt haben und nach den zuvor erläuterten Funktionen arbeitet. Darauf folgt das Design des Prints auf EAGLE und die Auflistung der Bauteile, welche auf den Print gehören.

# Theorie

Grundlage der Problematik

Es soll eine Schaltung erstellt werden, die Mathematischen Berechnungen eines Neurons imitiert. Dadurch kann eine lernende Schaltung entwickelt werden die dynamische bzw. lernende Schaltkreise ermöglichen. Zudem wird die Rechenzeit für ein Neuron reduziert.

Als lernen ist zu verstehen, dass die Schaltung berechnet, wie sie welche Eingänge verarbeiten soll. Das passiert indem der Nutzer am Eingang Daten anlegt und gleichzeitig einen Sollwert für den Ausgang definiert. Mit der Zeit lernt also die Schaltung, wie sie die Eingangsdaten verarbeiten soll und wie sie darauf reagieren soll.

## Test Bedingungen

Damit das Folgende auch gut verstanden werden kann eine kurze Erklärung zu den Fachbegriffen. In der Elektronik bzw. Digitaltechnik wird mit dem Binären Zahlensystem gearbeitet, diese Zahlen werden aus Einsen und Nullen gebildet.

Bsp.:

0001 entspricht im Dezimalsystem der Zahl 1.

0010 entspricht der Zahl 2.

0011 entspricht der Zahl 3.

0100 entspricht der Zahl 4.

Usw.

In der Praxis entspricht die 1 einer Spannung von 5 Volt (auch «High» genannt) und die 0 einer Spannung von 0V (auch «Low» genannt).

Um diese Informationen in der Elementarsten weise zu verarbeiten werden Drei Grundgatter verwendet:

AND:

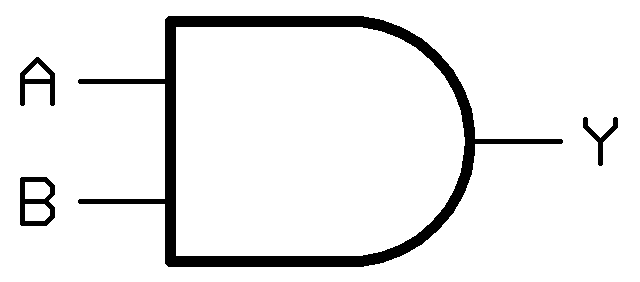


Abb 1: UND Gatter (Stefan506, Wikipedia, 2005)

Tabelle 1: Logiktabelle von einem UND Gatter

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Das AND-Gate verknüpft die Eingänge (A & B) so, dass der Ausgang (Y) nur einem «High» entspricht, wenn die A und B zur gleichen Zeit auch «High» sind. Jede andere Kombination führt zu einem «Low» am Ausgang.

OR:

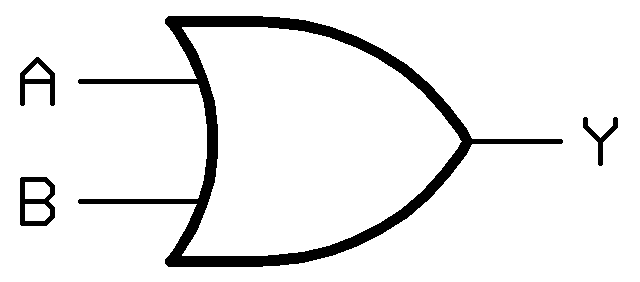


Abb 2: ODER Gatter (Stefan506, Wikipedia, 2005)

Tabelle 2: Logiktabelle von einem ODER Gatter

|  |  |  |
| --- | --- | --- |
| A | B | Y |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Das OR-Gate verknüpft die Eingänge (A & B) so, dass der Ausgang (Y) einem «High» entspricht, wenn mindestens einer der Beiden Eingänge «High» ist.

NOT:

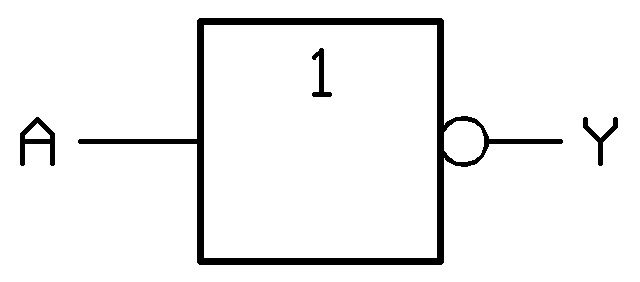


Abb 3: NICHT Gatter (Stefan506, Wikipedia, 2005)

Tabelle 3: Logiktabelle eines NICHT Gatter

|  |  |
| --- | --- |
| A | Y |
| 0 | 1 |
| 1 | 0 |

Das NOT-Gate besitzt nur einen Eingang (A). Das Gatter macht nichts anderes ausser dem Gegenteil von dem Eingang auszugeben.

### Möglichkeiten

Um zu überprüfen ob das Neuron auch lernt, soll es die Aufgabe von Logischen Gattern (AND, OR) übernehmen bzw. lernen. Um zu überprüfen, ob das Neuron in der Lage ist zu merken, wenn ein Eingang nicht benötigt wird, wird neben den notwendigen Eingängen auch noch ein überflüssiger Eingang benutzt, der Ignoriert werden soll. Wenn dies zuverlässig funktioniert, kann man davon ausgehen, dass das Neuron in der Lage ist bedingt komplexe Zusammenhänge zu approximieren und wichtige von nicht wichtiger Information zu unterscheiden.

### Unmöglichkeiten

Da das Neuron auf keinen Fall in der Lage ist Invertierungen (NOT-Gatter) zu imitieren, sind Kombinationen auf der Basis: «Wenn ein Eingang **nicht** eingeschaltet ist, dann…». Von vorherein ausgeschlossen und unmöglich für das Neuron zu erlernen.

Um es in Technischer Sprache zu sagen, das Neuron ist nicht in der Lage XOR, XNOR, NAND, NOR oder NOT Gatter zu erlernen, da nach dem Aufbau des Eingangs (Inputs[[1]](#footnote-1)) immer ein Stromzufluss herrschen muss.

## Grundlegendes Model eines Neurons

### Biologisches Model

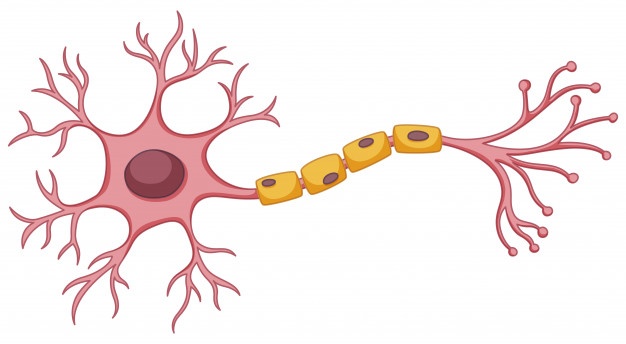


Abb 4: Biologisches Neuron

1)

4)

3)

2)

Das Neuron ist eine Zelle die Auf Signalübertragung spezialisiert ist. Sie kann Reize aufnehmen und ein Signal abgeben. In grosser Zahl und miteinander verbunden bilden sie das Nervensystem nahezu sämtlichen Säugetieren. Und ihre wohl wichtigste Funktion für uns Menschen ist, dass unser Zentrales Nervensystem (Gehirn) aus ihnen gebaut ist. Nach einer Schätzung etwa neunzig Milliarden[[2]](#footnote-2). Die Signalübertragung passiert über elektrische Signale, genau genommen über das weiterleiten von Ionensättigungen und nicht wie eine Typische Elektrische Schaltung über einen Elektronen Fluss.

In der Abb 4 wird ist ein solches Neuron Modellgetreu abgebildet. Die Dentriten (Abb 1.1) auch Postsynapse genannt nehmen die Reize von einem Verhergehenden Neuron oder einer Zelle auf. Der Zellkern/Zellkörper (Abb 4.2) ist das produktions und Speicher Zentrum der Zelle, dort wird die DNA aufbewahrt, der Stoffwechsel mit der Umgebung durchgefürt und Proteine produziert. Die für uns relevante Funktion dess Zellkern und Körpers, ist dass dort die Signale von den Dentriten Integrierend verarbeited werden. Über das Axon (Abb 4.3) wird das Integrierte Signal nach dem überschreiten eines threshholds in als ein Impuls übertragen und an die Endknöpfchen (Abb 4.4), auch präsynapse genannt, übergeben. Die Endknöpfchen leiten das Signal mittels Neurotransmitter[[3]](#footnote-3) an ein anderes Neuron weiter.

Dieser ganze durchlauf dauert einige Millisekunden.

Ein Bild, das Objekt enthält.

Automatisch generierte Beschreibung

Abb 5: Aktionspotenzial im Axon

Der Impuls der das Neuron am Axon besitz hat in etwa die Form von Abb 4. In der Abbildung wird auch ersichtlich wie das potenzial zuerst in einer E-Kurve vom Ruhepotential ansteigt und ab der Schwellspannung exponentiell bis zu einer Maximalspannung ansteigt, bevor die Spannung wieder zusammen bricht und das Neuron bis zum wiedererlangen des Ruhepotentials Nichts mer tun kann.

### Informatisches Model

Das Künstliche Neuron ist eine Struktur, welche dazu in der Lage ist, eine Funktion zu approximieren. Im Grunde hat diese Struktur im Minimum drei Abschnitte, die je zu einem Abschnitt im Biologischen Neuron korrespondieren. Mittlerweile wurden so viele Variationen der einzelnen Abschnitte entwickelt, die immer auf eine Anwendung optimiert sind, dass man sie hier nicht alle auflisten kann. Im Gegensatz zu dem Biologischen Neuron hat das Künstliche Neuron eine Laufzeit von meist unter einer µs, was über tausend Mal schneller ist.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung Einfach gesagt werden die Eingangssignale gewichtet, das bedeutet, mit einem Wert, der zuvor gelernt wurde, multipliziert, dann summiert und zum Schluss durch eine Aktivierungsfunktion ausgegeben. Diese Aktivierungsfunktion ist meistens nicht linear oder eine Schrittfunktion.

Wie beim Biologischen Neuron ist in der Informatik ein einzelnes Neuron ziemlich nutzlos, da sie ausser Simple aufgaben, wie die Funktion verschiedene Logische Operatoren, nichts können. Darum begann man diese Künstlichen Neuronen miteinander zu verknüpfen, wodurch Künstliche Neuronale Netzwerke entstanden sind. Mit der Hilfe dieser Netzwerke werden bis heute noch Künstliche Intelligenzen konstruiert die Aufgaben, wie Bilderkennung und Spracherkennung, übernehmen können.

Abb 6: Chart of Neural Networks (Tch, 2017)

In Abb 6 werden alle bisher standardisierte Neuronale Netzwerke aufgelistet. Wobei die Punkte einem Neuron entsprechen und die Linien einer Verbindung zwischen zwei oder mehr Neuronen entsprechen.

An diesem Punkt kommt, eine für unsere BMA, wichtige Eigenschaft von Künstlichen Neuronen ins Spiel.

Neuronale Netze werden auf einem Computer simuliert, was bedeutet, dass der Signalverlauf durch das Neuronale Netzwerk Sequenziell und in der Taktung des Computers verläuft. Ein Computer kann immer nur eine Aufgabe zu einem Zeitpunkt erledigen und ist durch das trotz der schnelleren Laufzeit pro Neuron dem biologischen Neuron, welches alle Neuronen und Signale Simultan und in Echtzeit «Berechnen» kann, weit unterlegen.

## Unser Model

Im Idealfall soll unser Neuron so schnell wie das aus der Informatik funktionieren und dennoch in Echtzeit die Berechnung durchführen. Um die Schaltung einfach zu Skalieren[[4]](#footnote-4) und eine Abhängigkeit von Herstellern zu vermeiden, soll es nur aus einfachen elektronischen Komponenten, wie Transistoren, Widerstände, Dioden und Kondensatoren, bestehen. Das bedeute das keine Integrierte Schaltungen verwendet werden.

Somit ist klar, dass wir kein Rechenzentrum in Getakteter form haben können, sondern eine Schaltung benötigen die fliessend den Ausgang berechnen kann.

Bei dem Paper «Introduction to spiking neural networks» von (Ponulak & Kasinski, 2011) ist uns aufgefallen das es in der Informatik eine Art von Neuralem Netzwerk existiert, das darauf ausgelegt ist, ein Biologisches Neuron so gut wie möglich zu Imitieren. In der Informatik konnte sich diese Art von Neuralem Netzwerk bisher noch nicht gut durchsetzen, weil es mehr Rechenleistung als ein Netzwerk einer anderen Art benötigt und schwer zu trainieren ist.

Die Funktion dieser Art Neuron ist identisch zu der Grundstruktur vom Informatischen Model, jedoch ist der Ausgabewert nicht analog, sondern in einer digitalen Form, nach dem «Alles oder Nichts» Prinzip. Von diesem Prinzip hat diese art Netzwerk ihren Namen «Spiking Neural Network» was auf Deutsch «Gepulstes Neuronales Netzwerk» bedeutet.

Dieses Gepulste Neuronale Netzwerk benötigt vor allem mehr Rechenleistung, weil es die elektronischen Vorgänge in einem Neuron simuliert. Das ist bei einem Getaktetem Computer durchaus ein Problem, in der Elektronik jedoch nicht, da dort die Physik das berechnen der Schaltung übernimmt.

Der Nachteil ist, es ist zeitabhängig. Was bedeutet, dass zeitlich alles gut aufeinander abgestimmt sein muss, was das Trainieren unheimlich erschwert.

## Bedingungen/Einschränkungen

Es gibt gewisse Probleme bei der Übersetzung vom Model in die reale Welt. Zum einen ist die Praxis nie perfekt. Bauteile haben Toleranzen, Unendlichkeiten gibt es nicht und es ist nicht möglich Spannungen auf eine einfache Art und Weise zu multiplizieren. Hinzu kommt, dass die gewünschte Formel auch nicht in reiner Form vorkommen wird.

Um die Schaltung kleiner bzw. in einer sinnvollen Grösse zu behalten verzichten wir auf negative Eingangsströme, was die Diversität an lernbaren Funktionen, einschränkt aber keinen Einfluss auf die Funktion der Schaltung hat.

## Mathematischer Ansatz

|  |
| --- |
|  |

Formel 1: Berechnung der Kernkondensatorspannung

Wenn der Kern-Kondensator eine Gewisse Spannung erreicht wird ein «Spike» erzeugt, was am Ausgang ausgegeben wird. Dieser «Spike» wird durch eine Aktivierungsfunktion erzeugt.

## *Aktivierung*

|  |
| --- |
|  |

Formel 2: Funktion der Ausgabe

: Ausgabe der Schaltung

Sobald der Spike erzeugt wird, wird über den Discharge-Transitor auf 0V gezogen. Was zur Folge hat, dass wenn der Input dauerhaft auf 5V gestellt ist, sich ein «Spike Train» mit einer Festen Frequenz bildet.

## Fehlerberechnung

Damit das Neuron überhaupt etwas lernt, muss eine Art Feedback vom Nutzer oder von der Umgebung kommen. Dafür wird ein Rückgabewert () verwendet. Mit diesem Rückgabewert kann die Gewichtung angepasst werden, in dem verändert wird.

|  |
| --- |
|  |

Formel 3: Berechnung des Fehlers

: Kondensator am Gate des Transistors

: Spannung am Eingang

: Widerstand der die Ent-/Lade Geschwindigkeit des Kondensators angibt (Lern- Schrittgeschwindigkeit)

: Ziel Spike-Train, dass das Neuron erreichen soll

Da und zeitbegrenzte Pulse sind, wird der Kondensator nur ein klein wenig entladen/geladen was den Transistor am jeweiligen Eingang etwas mehr sperrt oder leitet. Dadurch können alle inputgewichte über die Zeit so eingestellt werden, dass dem Ausgang entspricht.

Dies ist natürlich nur bis zu einer gewissen Komplexität möglich. Vor allem da die Eingangsströme nicht negativ sein können.

## Problematik von Theorie und Praxis

Die Grösste Problematik ist, dass das Speichern von Analogen Spannungen nur sehr schlecht geht und sehr störungsanfällig ist. Man kann die Schaltung etwas Stabilisieren, in dem man den Print so gut wie möglich isoliert und jeglichen Energieverlust über benachbarten Leitungen zu vermeiden.

## Blockschaltbild

Output

Schaltung

Verbindungspunkte

- 3 Eingänge

- Sollwert für Output

- Rückrechnung Ein/Aus

Abb 7: Blockartige Übersicht der Schaltung

In der obigen Abbildung (Abb 7) wird die Minimalste Grundstruktur des gesamten dargestellt. Die gesamte Peripherie der Schaltung besteht aus sechs Verbindungspunkte. Drei Eingänge welche die Signale in die Schaltung führen. Einen Eingang für den Sollwert welcher als die Schaltung in der Trainings Phase benötigt, um zu lernen. Der Eingang für die Rückrechnung Ein/Aus ist dafür da, um der Schaltung zu sagen ob sie sich in der Trainings Phase oder im Betrieb befindet. Der Output ist die Ausgabe der Schaltung für einen Gegebenen Eingangszustand an den drei Eingänge. Die Ausgabe der Schaltung ist immer vorhanden, auch wenn sie sich in der Trainings Phase befindet.

# Praxis

## Ganzes Neuron



Abb 8: Detailierte Blockartige Übersicht der Schaltung

Der Eingang (Input) verarbeitet die Spannung an den Eingängen bis und summiert sie. Das Zentrum (Core) des Neurons speichert die Werte des Eingangs über die Zeit bis die Aktivierung (Activation) einsetzt und das Zentrum zurückgesetzt wird.

Der Eingang bei der Fehlerberechnung (DeltaCalcV2) entspricht dem Sollwert zu Trainingszeiten. Der Ausgang bei der Aktivierung wird zu jedem Zeitpunkt beim Training mit dem Sollwert mittels der Fehlerberechnung verarbeitet und der Eingang angepasst.

Das Neuron lernt somit über Zeit und der Eingang wird bei jedem Fehler um einen konstanten minimalen Schritt-Wert angepasst um ein Schwingen um den Idealwert zu vermeiden und um das Neuron robuster gegen einzelne Fehler in den Trainingsdaten zu machen.

## Input



Abb 9: Schaltung für die Eingänge

Die Eingangsspannungen bis werden über die, zum Teil leitenden, Transistoren in spezifische Ströme umgewandelt und am Ausgang summiert.

Die Spannungen an den Kondensatoren bestimmen wie stark die Transistoren durchsteuern. So kann der Widerstand der Transistoren kontrolliert werden. Die Spannung an den Kondensatoren wird über die Fehlerberechnung verändert.

Im mathematischen Sinne entspricht diese Teilschaltung dem Abschnitt

Aus der Formel für die Vorwärtsrechnung.

Der kommt von dem Transistor , der direkt an angeschlossen ist.

### DeltaApply



Abb 10: Schaltung für das Anpassen der Kondensatorspannung

Diese Teilschaltung ist für das Anpassen der Spannungen am Kondensator zuständig. Jeder Eingang besitzt eine solche Teilschaltung. Der Eingang Delta bestimmt ob der Core-Kondensator geladen oder entladen wird. Enable schaltet mit der Spannung am Transistor den Rückrechnungsvorgang ein oder aus. Der Eingang sagt der Schaltung ob der Eingang der Gesamtschaltung überhaupt aktiv war.

Diese Teilschaltung ist die Umsetzung des Formelabschnittes , wobei dem Enable Eingang, dem Eingang und dem Delta Eingang entspricht.

## Core



Abb 11: Schaltung des Kerns

Das Core besteht hauptsächlich aus einem Kondensator welcher über den Eingang aufgeladen wird. Für den Fall, dass keine Signale über die Eingänge bis kommen, sorgt der Widerstand dafür, dass sich der Kondensator langsam entlädt und nach einer gewissen Zeit der Kondensator wieder zurückgesetzt ist.

Die Mathematische Darstellung der Teilschaltung ist . Dabei entspricht dem Ausgang der Input-Teilschaltung.

## Activation



Abb 12: Aktivierungsschaltug

Die Aktivierung ist dazu da, um zu bestimmen ob ein Signal weiter bzw. ausgegeben werden soll. Wenn die Spannung am Core (Kondensator) über die 2.5V steigt, schaltet der Vergleicher von 0V auf 5V über eine sehr kurze Zeit. Durch diese rapide Änderung gibt es eine Spannungsspitze auf der anderen Seite des Kondensators. Diese Spannungsspitze wird über die zwei folgenden Transistoren verstärkt, damit das Ausgangssignal eine höhere Spannung erreicht und auch mehr Strom liefern kann.

Dieses Ausgangssignal wird über die Grüne Leiterbahn zurückgeführt und entlädt das Core (Kondensator) sehr schnell. Wodurch dieser wieder auf unter die 2.5V fällt und der Ausgang des Vergleichers wieder auf 0V schaltet. Danach beginnt der Vorgang von vorne.

Die Diode am Ausgang sorgt dafür, dass wenn der Ausgang des Vergleichers auf 0V ist, der Ausgang der Ganzen Schaltung einen «Tri-state»[[5]](#footnote-5) annimmt (Das bedeutet, dass kein Strom rückwärts in die Schaltung fliessen kann).

### Vergleicher OPV



Abb 13: Primitiver Operations Verstärker

Dies ist eine primitive Schaltung eines Komparators bzw. Vergleicher, welcher in der Elektronik eine Essenzielle Rolle spielt. Er vergleicht die Eingänge + & – und schaltet den Ausgang auf 5V, wenn die Spannung am + Eingang Grösser ist als die am – Eingang. Ist das umgekehrte der Fall wird der Ausgang auf 0V gesetzt.

## Delta Calc



Abb 14: Schaltung für die Fehlerberechnung

Die Delta-Calculation (Delta Calc) ist dafür zuständig, zu entscheiden, ob der Ausgang (Ist-Wert) mit dem Soll-Wert übereinstimmt. Der Ausgang «Delta» von diesem Abschnitt gibt an, ob die Kondensatoren an den Eingängen bis geladen oder entladen werden sollen. Der Ausgang «En» (Kurz für Enable) von diesem Abschnitt sagt ob dieses Laden oder Entladen der Kondensatoren überhabt durchgeführt werden soll oder nicht.

In Digital-technischer Sprache:

Der Ausgang «Delta» entspricht einem Und Gatter mit den Eingängen und . Der Ausgang «En» ist eine XNOR Verknüpfung von und . Sprich wen dann sollen keine Änderungen an den Kondensatoren vorgenommen werden.

# Gesamtschema von EAGLE[[6]](#footnote-6)

# Messungen

## Blockschaltbild des Versuchsaufbaus

Mikro-kontroller

Schaltung

3Eingänge & Sollwert Sollwert

Istwert

Abb 15: Blockartige darstellung der Messung

Der Mikrokontroller ist so programmiert, dass in einem 10kHz tackt hochgezählt wird und der momentan Wert, an den 3 Eingängen in binärer Form und den Sollwert mittels einer logischen Verknüpfung, an die Schaltung übertragen wird. Danach wird der Istwert der Schaltung vom Mikrokontroller eingelesen und abgespeichert. Pro Verknüpfung werden 100 Trainingsdurchläufe durchgeführt. Somit dauert ein Lerndurchgang ca. 70ms. Dies wird für die Verknüpfungen AND und OR durchgeführt und mehrmals nach einem Neustart der Schaltung wiederholt.

## Programmierung des Mikrokontrollers

# Fazit

# Formelverzeichnis

[Formel 1: Berechnung der Kernkondensatorspannung 10](#_Toc31497594)

[Formel 2: Funktion der Ausgabe 10](#_Toc31497595)

[Formel 3: Berechnung des Fehlers 11](#_Toc31497596)

# Abbildungsverzeichnis

[Abb 1: UND Gatter (Stefan506, Wikipedia, 2005) 6](#_Toc31498140)

[Abb 2: ODER Gatter (Stefan506, Wikipedia, 2005) 6](#_Toc31498141)

[Abb 3: NICHT Gatter (Stefan506, Wikipedia, 2005) 7](#_Toc31498142)

[Abb 4: Biologisches Neuron 8](https://studmswch-my.sharepoint.com/personal/tendai_rondof_stud_msw_ch/Documents/BMS/BMA/HardwareNeuron/Dokumentation/Harware%20Neuron%20Produkt%20Doku/schriftlicher%20Kommentar.docx#_Toc31498143)

[Abb 5: Aktionspotenzial im Axon 8](#_Toc31498144)

[Abb 6: Chart of Neural Networks (Tch, 2017) 9](https://studmswch-my.sharepoint.com/personal/tendai_rondof_stud_msw_ch/Documents/BMS/BMA/HardwareNeuron/Dokumentation/Harware%20Neuron%20Produkt%20Doku/schriftlicher%20Kommentar.docx#_Toc31498145)

[Abb 7: Blockartige Übersicht der Schaltung 13](#_Toc31498146)

[Abb 8: Detailierte Blockartige Übersicht der Schaltung 14](#_Toc31498147)

[Abb 9: Schaltung für die Eingänge 14](#_Toc31498148)

[Abb 10: Schaltung für das Anpassen der Kondensatorspannung 15](#_Toc31498149)

[Abb 11: Schaltung des Kerns 16](#_Toc31498150)

[Abb 12: Aktivierungsschaltug 16](#_Toc31498151)

[Abb 13: Primitiver Operations Verstärker 17](#_Toc31498152)

[Abb 14: Schaltung für die Fehlerberechnung 18](#_Toc31498153)

[Abb 15: Blockartige darstellung der Messung 19](https://studmswch-my.sharepoint.com/personal/tendai_rondof_stud_msw_ch/Documents/BMS/BMA/HardwareNeuron/Dokumentation/Harware%20Neuron%20Produkt%20Doku/schriftlicher%20Kommentar.docx#_Toc31498154)

# 

# Literaturverzeichnis

Ponulak, F., & Kasinski, A. (1. Januar 2011). Von https://www.ane.pl/cookie?page=linkout.php?pii=7146 abgerufen

Stefan506. (14. Dezember 2005). *Wikipedia.* Von https://de.m.wikipedia.org/wiki/Datei:Logic-gate-or-us.png abgerufen

Stefan506. (14. Dezember 2005). *Wikipedia.* Von https://de.wikipedia.org/wiki/Datei:Logic-gate-inv-iec.png abgerufen

Stefan506. (14. Dezember 2005). *Wikipedia.* Von https://de.wikipedia.org/wiki/Datei:Logic-gate-and-us.png abgerufen

Tch, A. (4. August 2017). *towardsdatascience.com*. Von https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464 abgerufen

Wikipedia. (3. Januar 2020). *Wikipedia/Neuron*. Von https://de.wikipedia.org/w/index.php?title=Nervenzelle&oldid=195474238 abgerufen

# Anhang

## Layout/Bestückungsplan von EAGLE

## Stückliste

1. Die Schaltung wird im Abschnitt Input erklärt [↑](#footnote-ref-1)
2. Diese Schätzung wurde aus Wikipedia entnommen [↑](#footnote-ref-2)
3. [↑](#footnote-ref-3)
4. Um auch eine Verwendung als Integrierte Schaltung zu ermöglichen [↑](#footnote-ref-4)
5. Tri-state ist eine Schaltung derer Ausgang so hochohmig ist, dass sie als nicht vorhanden zählt [↑](#footnote-ref-5)
6. EAGLE ist ein Layouteditor, der es ermöglicht Leiterplatten zu designen [↑](#footnote-ref-6)