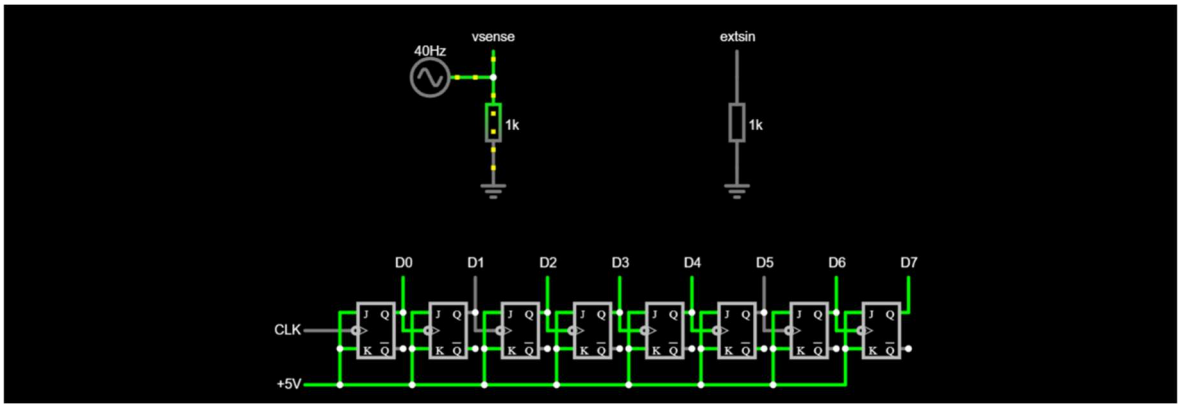

JavaScript: Grundlagenlabor im Browser - Schnittstelle zu Circuit.js

Studienarbeit

des Studienganges Elektrotechnik

an der Dualen Hochschule Baden-Württemberg Mannheim

von

Yi Hank Xie

Bearbeitungszeitraum	25.12.2021 – 04.04.2022
Matrikelnummer, Kurs	8986105, TEL 19 EE
Betreuer	Prof. Dr. Rüdiger Heintz

Erklärung

Ich versichere hiermit, dass ich meine Projektarbeit mit dem Thema: „Javascript: Grundlagenlabor im Browser - Schnittstelle zu Circuit.js“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Mannheim	04.04.2022	
Ort	*Datum*	*Yi Hank Xie*

Inhaltsverzeichnis

Begriffserklärung

Java Applet	Kleine Applikationen, welche außerhalb vom Browser aufgerufen wird
Tool	Werkzeuge, die dem Programmierer helfen Applikationen zu entwickeln
Plug-in	Computerprogramm, welches neue Funktionen zu einem ursprünglichen Programm hinzufügt, ohne das ursprüngliche Programm zu ändern
Compiler	Übersetzt eine bestimmte Programmiersprache in Maschinensprache oder einen code

Abbildungsverzeichnis

Codeverzeichnis

1 Aufgabenstellung

Die Studienarbeit setzt sich mit den Grundlagen von JavaScript und deren Verwendung mit CircuitJS auseinander. Das Ziel ist es ein Programm zu schreiben, welches als Schnittstelle zwischen fungiert. Dadurch solle es ermöglicht werden über eine Website auf die Simulation von CircuitJS einzugreifen. Es soll eine Methode entwickelt werden, die es dem Nutzer ermöglicht individuelle Elemente zu beobachten, sie anzeigen zu lassen und sie zu verändern.

Als Endprodukt wird eine Website programmiert, die eine Simulation von CircuitJS im Hintergrund laufen lässt und dabei die Parameter sowohl auslesen als auch anpassen kann

2 Einleitung

Die Verwendung von Computersimulationen fand bereits Mitte des 20ten Jahrhunderts ihren Ursprung. Damals wurde im Zuge des Manhattan- Projektes auf einem der ersten Computer verschiedene Designs für Atombomben getestet. Der Computer konnte die hydrodynamischen Gleichungen lösen, wodurch aufwendige und gefährliche Experimente vermieden werden konnten. [1] Heutzutage sind Simulationen kaum noch wegzudenken, da sie sowohl in der Forschung als auch im Alltag allgegenwärtig sind.

Inzwischen werden Simulation beinahe überall verwendet, in der Medizin, der Produktentwicklung, der Astronomie, beim Vermitteln von theoretischen Inhalten. Selbst das Smartphone in der Tasche kann Computersimulationen laufen lassen, welche vor 30 Jahren noch einen Rechner in Schrankgröße benötigt hätte. Die Anwendungsgebiete sind nahezu Grenzenlos. Die Option einen realen Sachverhalt abzubilden, ermöglicht es jedem Nutzer selbstständig und schnell Versuche durchzuführen, welche sonst zu kosten- und zeitintensiv wären. Jedoch muss man beachten, dass eine Simulation niemals die exakte Realität abbildet. Es wird immer Abweichungen geben, da Simulationen auf, von Menschen erstellten, mathematischen Modellen basieren. [1] Nichtsdestotrotz haben Computersimulation einen enorm wichtigen Stellenwert in unserer Gesellschaft eingenommen, welcher nicht mehr wegzudenken ist. Die Weiterentwicklung von Simulationsprogrammen ist ein wichtiges Unterfangen, um es jedem zugänglich zu machen und das volle Potenzial zu nutzen. Der rasante Fortschritt und moderne Computer ermöglichen es heutzutage jedem ein Simulationsprogramm zu schreiben. Wie präzise und ausgereift es ist hängt vom jeweiligen Entwickler ab. Die Beschränkungen einer Simulation sind nämlich wie bereits erwähnt immer vorhanden, jedoch kann man so gut wie jede Größe, die man mathematisch modellieren kann, auch in eine Simulation einbauen.

Dies Studienarbeit hat den Zweck eine Schnittstelle zwischen dem Simulationsprogramm und einer Website zu schaffen. Dadurch soll das Simulationsprogramm zugänglicher und individuell gestaltbarer werden. Im folgenden Abschnitt wird auf das Simulationsprogramm CircuitJS eingegangen. Dabei wird die Entstehung, der aktuelle Stand des Programms und die Verwendung genauer erläutert.

3 Circuit

Das Programm „CircuitJS1“ wurde ursprünglich von Paul Falstad als ein Java Applet entwickelt. Später wurde es dann von Iain Sharp in einen Browser integriert. Es handelt sich hierbei um eine Schaltungssimulation, welche elektrotechnische Vorgänge darstellen kann. [2] [3]

3.1 Allgemeines

Das Programm ermöglicht es dem Nutzer unterschiedliche Schaltungen zu simulieren. Es ist ein wichtiges Tool für die Elektrotechnik und ermöglicht das Prüfen von theoretischen Werten unter der Berücksichtigung von verschiedenen Einflussfaktoren. Dabei gibt es eine große Auswahl von Bauteilen und bereits vorgefertigte Schaltungen. Durch die vielen Möglichkeiten hängt es vom Nutzer ab wie er das Programm verwenden möchte. Neben simplen, idealisierten Schaltungen können auch komplexe Schaltungen mit Ersatzimpedanzen dargestellt werden, indem beispielsweise Materialverluste oder Erdkapazitäten berücksichtigt werden. Ein weiterer großer Vorteil des Programms ist, dass man flexibel die Bauteile austauschen, und anpassen kann. Durch die vielen unterschiedlichen Einsatzmöglichkeiten und vorprogrammierten Schaltungen gibt es eine große Bandbreite von Nutzern weltweit. Das Programm ist kostenlos im Internet verfügbar und kann mithilfe von Tools runtergeladen und kompiliert werden.

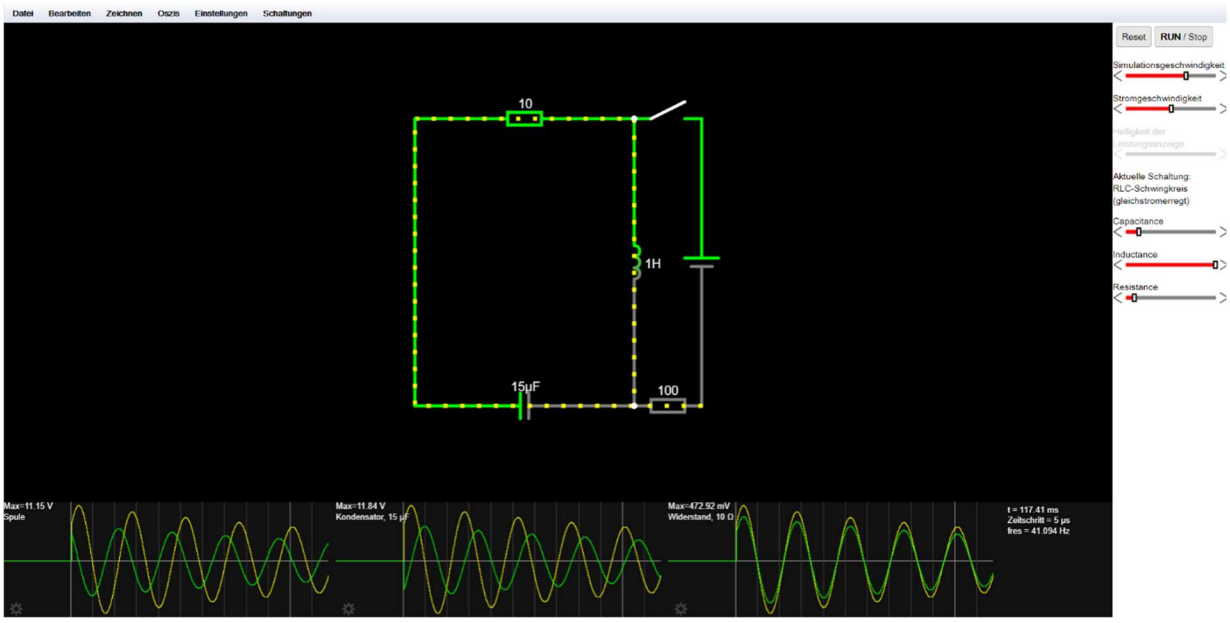

Abbildung 1: CircuitJS

3.2 Installation und Kompilierung

Um das Programm herunterzuladen und verfügbar zu machen, werden zwei Tools benötigt. Zunächst wird das open source Programmierwerkzeug „Eclipse IDE“ benötigt. Es handelt sich um eine integrierte Entwicklungsumgebung, welche hauptsächlich dafür verwendet wird, um Applikationen in Java zu erstellen. Eclipse war bis 2016 die beliebteste und ist inzwischen (Stand 27.03.2022) die zweitbeliebteste Entwicklungsumgebung für Java. [4] [5]

Durch verschiedene Erweiterungen können auch andere Entwicklungsaufgaben bewältigt werden. Um CircuitJS als Webversion zu verwenden, benötigen wir zusätzlich zur Entwicklungsumgebung eine Erweiterung namens „GWT plugin for Eclipse“. [6] Diese Erweiterung bzw. dieses Plug-in, wurde von Google LLC entwickelt. Es handelt sich um ein freies Webframework, also um ein nicht Nutzereingeschränktes Programmiergerüst. Die eigentliche Aufgabe ist es das Bauen einer Website zu vereinfachen. [7] Bei der Installation von CircuitJS dient es als ein Compiler, welcher die Sprache Java in JavaScript übersetzt. [8] Im Kapitel JavaScript wird näher drauf eingegangen.

Da die Installation des GWT Plug-ins nicht fehlerfrei funktioniert hat, sind die nächsten Schritte theoretisch geschildert. Nach dem die Daten kompiliert wurden, werden sie in einem Ordner abgespeichert. Es muss alles aus dem Ordner mit der Bezeichnung „war“ kopiert und auf den Webserver hochgeladen werden. Zusätzlich kann

man noch durch die Dateien „iframe.html“ und „shortrelay.php“ zusätzliche Anpassungen vornehmen. So ermöglicht es die erste Datei eine Werbung bzw. Anzeige zu schalten und die zweite sorgt dafür, dass man den Link für seine Seite verkürzen kann.

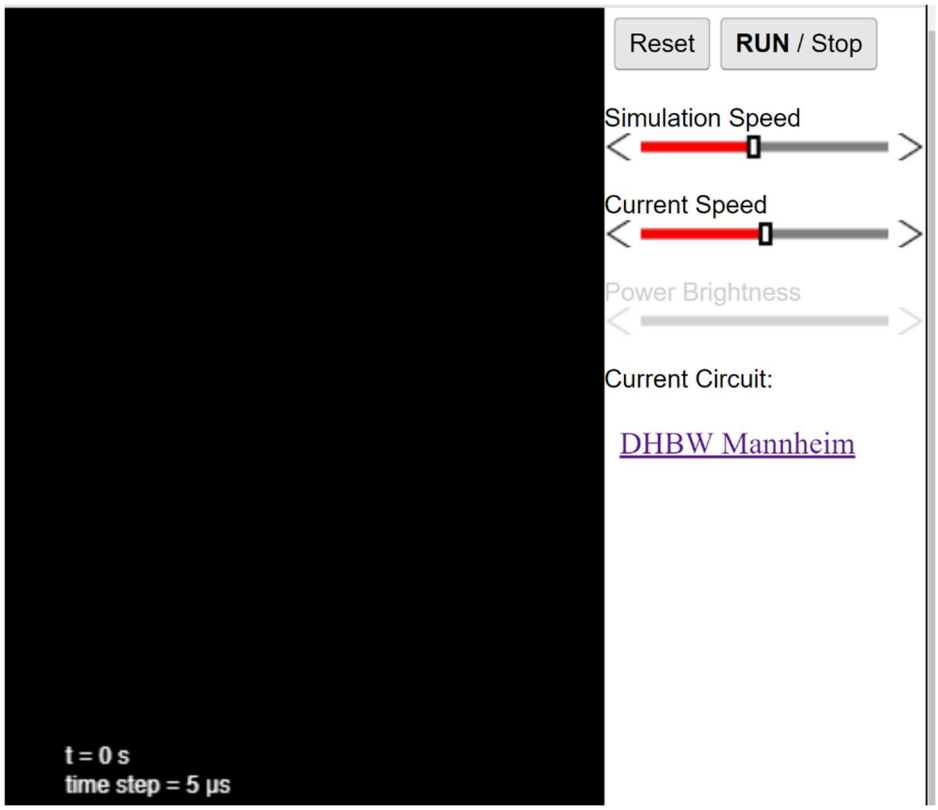

Abbildung 2: DHBW Mannheim Website als Anzeige

Im nächsten Kapitel wird auf die verwendete Programmiersprache JavaScript eingegangen. Diese ermöglicht die Interaktion zwischen Website und Simulation.

4 JavaScript

Laut einer Umfrage aus dem Jahr 2022 ist JavaScript die drittbeliebteste Programmiersprache der Welt. [9] Die Skriptsprache wurde 1995 von Netscape entwickelt und hatte den Zweck Benutzerinteraktionen mit Websites auszuwerten, und Inhalte dementsprechend zu verändern. Durch ihren Namen assoziieren fälschlicherweise viele Leute die Sprache mit Java. Allerdings wurden beide Sprachen komplett unabhängig voneinander entwickelt und basieren auf unterschiedlichen Grundkonzepten. Der ursprüngliche Name war LiveScript, jedoch erfreute sich Java zu der Zeit größter Beliebtheit, dies animierte Netscape dazu den Namen zu JavaScript zu ändern. [10] [4]

4.1 Allgemeines

JavaScript hat viele unterschiedliche Anwendungsgebiete. Die Interpreter-Sprache ist dafür bekannt, dynamische Manipulationen von Webseiten zu ermöglichen. [3] Die Sprache wird auch für Spiele, Anwendungsprogramme und in Microcontrollern eingesetzt. [4] Vor allem im Zusammenhang mit HTML wird JavaScript verwendet. Woran das liegt und welchen Vorteil das bringt wird im nächsten Abschnitt kurz erläutert.

4.2 HTML und JavaScript

„*HTML ist eine textbasierte Auszeichnungssprache zur Strukturierung elektronischer Dokumente…“* [11] Das bedeutet, dass die Sprache dazu verwendet wird, Texte und Daten auf Websites zu Gliedern und zu Formatieren. Die Abkürzung steht für „Hypertext Markup Language“. Websites, die auf dieser Sprache basieren bilden die Grundlage des World Wide Webs. [3] Der Nachteil der Programmiersprache ist, dass es keine flexiblen Änderungen der Inhalte zulässt. Dies bedeutet, dass man keine komplexere Nutzerinteraktionen verwenden kann. Neben Tasten, die einfache Befehle ausführen, kann keine Änderung von Elementen ausgeführt werden.

Um das zu ändern, verwenden die meisten Websites zusätzlich JavaScript. Hierdurch können HTML-Inhalte gelöscht, verschoben oder verändert werden. Es gibt unterschiedlichste Elemente, die man mithilfe von JavaScript beeinflussen kann. Für die Schnittstelle zwischen CircuitJS und der Website waren vor allem zwei von Bedeutung. „Input-Boxen“, um dem Nutzer aktive Anpassungen der Parameter zu ermöglichen und

ein Iframe, indem die Simulation läuft. Zunächst werden Iframes und Input-Boxen betrachtet.

4.3 Iframes und Input-Boxen

Prinzipiell sind Iframes (Inline Frames) ein HTML Dokument, welches in ein weiteres eingebettet sind. Beispielsweise könnte es sich wie bei Werbungen um Inhalte von einer anderen Website handeln, welche auf der Hauptwebsite angezeigt werden. [12]

Die Verwendung von Iframes ist daher bei einem Großteil der Websites vorhanden und wird daher auch von den meisten Webbrowsern unterstützt.

Inputboxen sind für Interaktionen zwischen Nutzer und Website zuständig. Die eingegebenen Werte können als Variablen festgehalten und verwendet werden. In vielen Fällen dienen sie zur Weitergabe von Zahlen oder einfachen Wörtern an das Programm.

5 Beispiel von Falstadt

Für die Umsetzung der Schnittstelle gibt es auf der Homepage von Falstad ein Beispiel. Dieses zeigt welche Anwendungsmöglichkeiten es gibt, wenn man über JavaScript auf den Simulator im Iframe eingreift. Es ist möglich die verwendeten Elemente und ihre Werte anzuzeigen. Außerdem kann man Werte über die Website ändern. [2]

5.1 Anzeigen von Elementen und Werten

Damit die Seite mit der Simulation interagieren kann, muss zuerst der Iframe über seine ID aufgerufen werden. Dies geschieht wie folgt.

```
var iframe = document.getElementById("circuitFrame");
```

Durch den Befehl „document.getElementById()“ kann man auf das jeweilige Element zugreifen. Danach können Werte je nach Notwendigkeit auf der Seite angezeigt werden. Durch Zugriff auf die ID der Quelle wird der zeitabhängige Spannungswert erfasst und angezeigt. Im Programm wird das über die Ausgabe einer Variable umgesetzt.

```
var vsense = sim.getNodeVoltage("vsense");
info.innerHTML += "<br>V(vsense) = " + round(vsense);
```

Anhand des Codes ist zu erkennen, dass die Quelle in einer Variable namens „vsense“ gespeichert und danach abgerufen wird. Dadurch können die Werte von jedem Bauelement einzeln abgespeichert und angezeigt werden. Mit dieser Basis lassen sich unterschiedlichste Anzeigen realisieren. Als weitere Möglichkeit wurde ein Schrittzähler, welcher durch einen Taktgeber betrieben wird, integriert. Durch eine Schleife wird im Binärzahlensystem gezählt und dann wird der Wert in Dezimalstellen ausgegeben.

Auch das automatische Erkennen von allen verwendeten Elementen, lässt sich durch Schleifen und Listen realisieren.

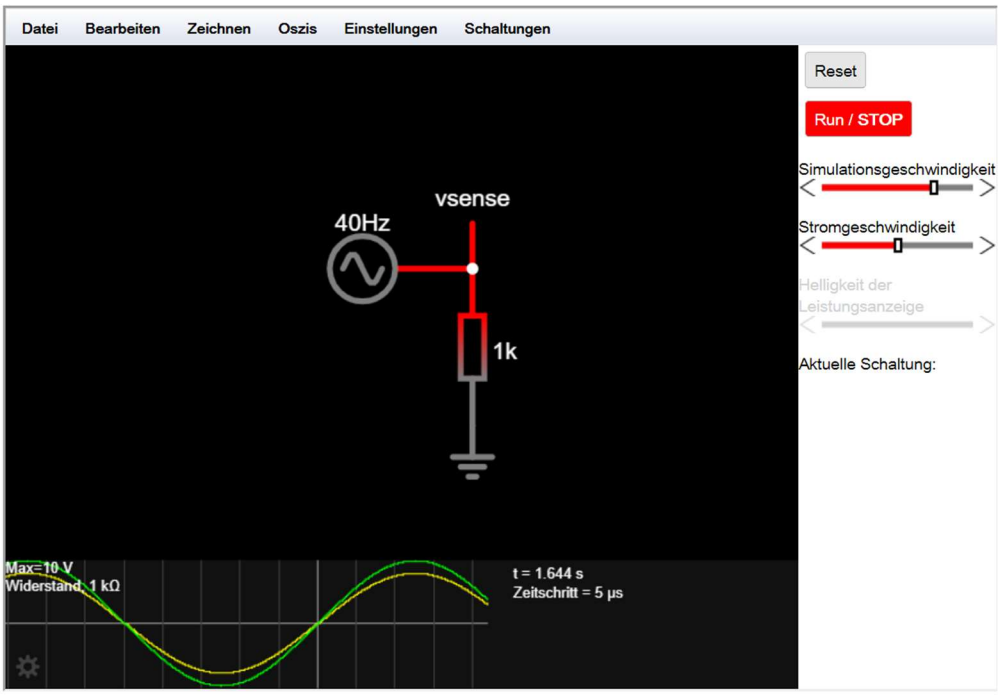

Abbildung 3: Anzeige von Spannungswerten durch JavaScript

5.2 Änderung von Werten

Wie bereits erwähnt ist neben der Anzeige auch die aktive Änderung von Werten möglich. In diesem Beispiel wurden durch Input-Boxen (s. Kapitel 4.3) vorab Werte festgelegt. Diese Werte legen die Frequenz und Amplitude einer Wechselspannungsquelle fest. Der Nutzer kann mit diese nach belieben anpassen, indem er die Werte in der Box verändert. Dies hat den Vorteil, dass man nicht immer auf die Bauteile klicken muss, um ihre Werte zu verändern. Des Weiteren können Werte dadurch individual benötigte Parameteränderungen vollziehen. So kann man Beispielsweise eine bestimmte Schrittfolgenänderung o. Ä einprogrammieren. Wie im vorherigen Beispiel werden dafür die IDs der Input-Boxen verwendet.

```
<input id="freq" value="10" />
<br>
extsin amplitude:
<input id="ampl" value="10" />
```

Der Wert wird dann in der ID abgespeichert. Um die Wechselspannung richtig abzubilden, wird der Spitzenwert mit einer Sinusfunktion multipliziert.

```
sim.setExtVoltage("extsin", ampl*Math.sin(freq*Math.PI*2*t));
```

Aufgrund der Formel wird die maximale Spannung der externen Spannungsquelle „extsin“ zum eingegebenen Amplitudenwert und alterniert während jeder Periode einmal.

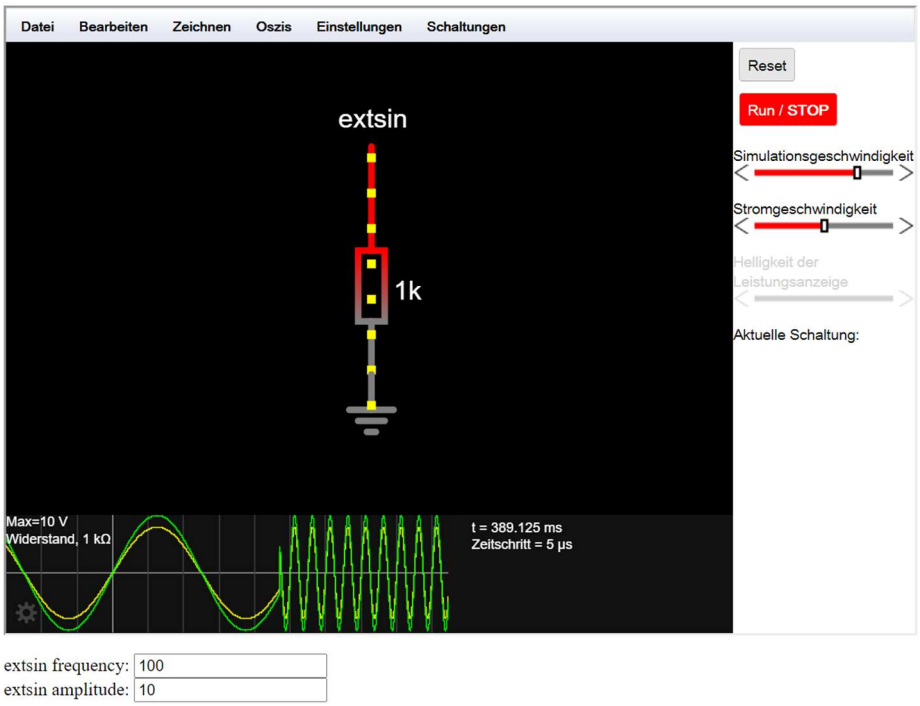

Abbildung 4: Anpassen von Spannungen über Input-Boxen

In der Abbildung wurde die Frequenz von 10 auf Hundert erhöht. Im künstlichen Oszilloskop kann man diese Veränderung erkennen.

6 Umsetzung

Die Realisierung des Projektes, welche mithilfe der Vorlage bzw. des Beispiels zu entwickeln gilt, benötigt eine Reihe von Kenntnissen und ein tieferes Verständnis für unterschiedliche Programmiersprachen. Als weitere Voraussetzung muss ein Web-Server zur Verfügung stehen über dem die Seite läuft. Die Realisierung des Projektes wurde in unterschiedlichen Schritten angegangen. Zunächst sollten die Werte eines Hoch- bzw. Tiefpassfilters ausgegeben und beeinflusst werden. Letztendlich scheiterte die Weiterentwicklung des Programmes an der Kompilierung des Programms.

Für die Umsetzung des Projektes, müssen zunächst grundlegende Kenntnisse in HTML vorliegen. Eine mögliche Aneignung dieser wird im folgenden Teil beschrieben.

6.1 HTML Basics

Das Grundgerüst wurde mithilfe von HTML konstruiert, der Code ist in 2 Segmente unterteilt. Der „Kopf“ bzw. „Head“ enthält Informationen, die nicht auf der Seite sichtbar sind. Darunter fallen Punkte wie Titel der Website oder stylische Aspekte.

```
<!DOCTYPE html>
<html>
<head>
<title>Studienarbeit Circuit</title>
<style>
h1 {color: rgb(28, 226, 44);}
body {background-color: rgb(0, 0, 0);}
p {color: rgb(28, 226, 44);}
a {text-decoration: none; color: black;}
</style>
</head>
```

1. Code: Beispiel eines <head> abschnittes

Im Körper sind die sichtbaren Kommentare, Knöpfe und Input-Boxen enthalten. Mithilfe von Knöpfen konnte die Anforderung einer auf Knopfdruck wechselnder Schaltung verwirklicht werden.

```
<button>
<a href="[circuitjs.html?startCircuit=Hochpassfil-
ter(Spule).txt]circuitjs.html?startCircuit=Hochpassfil-
ter(Spule).txt" target="iframe">Hochpassfilter
</a>
</button>
<button>
<a href="[circuitjs.html?startCircuit=Tiefpassfil-
ter(Spule).txt]circuitjs.html?startCircuit=Tiefpassfil-
ter(Spule).txt" target="iframe">Tiefpassfilter
</a>
</button>
```

2. Code: Knöpfe um zwischen Hoch und Tiefpassfilter zu wechseln

Beim Anklicken der Knöpfe wird das Iframe als Ziel ausgewählt. Daraufhin wird die Adresse, die das Iframe aufruft, geändert. Dadurch wird eine Website mit vorgefertigter Schaltung angezeigt. Die Funktion den Iframe zu „verstecken“ und im Hintergrund laufen zu lassen konnte so ebenfalls integriert werden.

```
<!--Hochpass- und Tiefpassfilter-->

<button onclick="document.getElementById('circuit-
frame').style.display='none'";hideshow()>Hide</button>

<button onclick="document.getElementById('circuit-
frame').style.display='block'";hideshow()>Show</button>
```

3. Code: Sichtbarkeit der Simulation auf der Seite

Es gibt auch die Möglichkeit den Iframe über die Eigenschaft „display“ zu verstecken jedoch wäre der Platz dann weiterhin für den Iframe reserviert und man hätte eine große Lücke auf der Seite. Nun da das Grundgerüst gebaut war kam die Problematik

auf, dass die Verbindung zum CircuitJS nicht aufgebaut werden konnte. Dies beruhte darauf, dass die Verbindung zu einem anderen HTML Dokument abgelehnt wurde, wenn dieses nicht auf einem Webserver läuft.

6.2 Xampp als Webserver Ersatz

Um dagegen Abhilfe zu schaffen, wird die Applikation „Xampp“ verwendet, um einen Privaten lokalen Server zu hosten. Dieser ermöglicht den Aufruf von CircuitJS innerhalb des Iframes. Jedoch funktioniert die Interaktion zwischen Website und Iframe nicht. Der Grund hierfür liegt darin, dass die zur Verfügung gestellte kompilierte Version von CircuitJS nicht die benötigten Bauteile enthält. Dies lässt darauf zurückschließen, dass die Version veraltet ist oder es zu Komplikationen bei der Kompilierung kam. Daher soll eine eigene kompilierte Version von CircuitJS den Fehler beheben.

6.3 Kompilieren

Wie in Kapitel 3.2 erklärt wird, muss man für die Kompilierung zwei Tools runterladen. Jedoch funktioniert die Integration des GWT-Plug-Ins, trotz wiederholtem Download, nicht. Daher konnte die eigene Kompilierung nicht durchgeführt werden. Durch andere Methoden wird versucht ohne das Plug-In eine Interaktion zwischen Website und der Simulation zu ermöglichen.

6.4 Weitere Ansätze

Weitere Ansätze, wie die Verwendung von Web-Crawlern oder die eigene Erweiterung der Menüleiste konnten die Schnittstelle nicht funktionstüchtig machen.

6.4.1 Web-Crawler

Ein Web-Crawler ist ein Programm das Seiten untersuchen kann. Außerdem werden links zu weiterführenden Websites ebenfalls durchsucht. Dies ermöglicht es ganze Websites mit ihren Anlagen und Downloads zu kopieren. Der Versuch die Beispielseite dadurch zu kopieren und die benötigten Dateien herunterzuladen funktionierte nicht. Dies könnte an den begrenzten Möglichkeiten der verwendeten Crawler liegen. Denn neben begrenzten Laufzeiten, gibt es auch Probleme beim Runterladen von mehreren Websites gleichzeitig. Ein weiteres Problem ist, dass das Crawlen von Websites die

JavaScript enthalten nicht Praktikabel ist. Da durch JavaScript der Inhalt einer Seite drastisch geändert wird, können die Daten nicht alle rauskopiert werden.

6.4.2 Einfügen in die Menüleiste

Ein anderer Versuch war es die Menüleiste von CircuitJS zu ändern und den Button einzufügen, jedoch hat dies eine bereits vorhandene These stützt. Das künstliche Einfügen der Schaltfläche hatte zufolge, dass die fehlende Quelle zwar abgebildet wurde, jedoch wurde die ID nicht gefunden. Dies bestätigt die Vermutung, dass die gegebenen kompilierten Dateien nicht die passenden Komponenten enthielten, um die Aufgabe zu erfüllen.

7 Ergebnis

Das Ergebnis der Studienarbeit ist, dass die Umsetzung der Schnittstelle, in der vorgegebenen Zeit nicht vollendet werden konnte. Bis auf die Interaktion zwischen der Website und der Simulation wurden alle Anforderungen erfüllt. Es wurden Schritte und Fehlschläge protokoliert, welche die Wiederaufnahme des Projektes effizienter und einfacher gestaltet. Das eigentliche Programmieren der Website kann größtenteils von der Beispielwebsite kopiert und verwendet werden. Die theoretische Umsetzung ist ebenfalls im Code enthalten.

Neben ID-Anpassungen wurden noch mehr Kommentare und Extraschaltflächen implementiert. Diese funktionieren in der Theorie genau wie in der Vorlage und benötigen nur noch die richtigen Dateien.

8 Fazit

Zusammengefasst lässt sich sagen, dass die Arbeit durch zukünftige Arbeiten oder als Projekt fortgesetzt werden kann. Durch den Faktor der fehlenden kompilierten Version, konnte sie nicht vollendet werden. Jedoch ist der Mehraufwand für künftige Studienarbeiten bzw. für Entwickler, die die Schnittstelle vollenden gering. Des Weiteren wurde die Aufgabenstellung durch die Fehlversuche präzisiert und die genauen Schwierigkeiten erkannt.

In der Theorie sollte eine aktuelle, kompilierte Version von CircuitJS reichen, um die Schnittstelle fertigzustellen. Sobald dies gelungen ist können die Beispiele von der Website kopiert und erweitert werden.

Literaturverzeichnis

[1] D. S. Lukas Fässler, 25 11 2021. [Online]. Available: https://doc.et.ethz.ch/latest/i_simulieren_TH.pdf. [Zugriff am 22 03 2022].

[2] P. Falstad, „falstad.com,“ [Online]. Available: http://www.falstad.com/circuit/. [Zugriff am 14 03 2022].

[3] „Wikipedia,“ [Online]. Available: https://de.wikipedia.org/wiki/Schaltungssimulation. [Zugriff am 22 03 2022].

[4] „Wikipedia,“ [Online]. Available: https://en.wikipedia.org/wiki/Eclipse_(software). [Zugriff am 24 03 2022].

[5] „eclipse,“ [Online]. Available: https://www.eclipse.org/org/. [Zugriff am 10 03 2022].

[6] „github.com,“ [Online]. Available: https://github.com/pfalstad/circuitjs1. [Zugriff am 01 03 2022].

[7] „evolve,“ [Online]. Available: https://evolve.ie/q-and-a/what-is-a-web-framework/. [Zugriff am 28 03 2022].

[8] „gwtproject,“ [Online]. Available: https://www.gwtproject.org/. [Zugriff am 29 03 2022].

[9] „statista,“ [Online]. Available: https://de.statista.com/statistik/daten/studie/678732/umfrage/beliebteste-programmiersprachen-weltweit-laut-pypl-index/. [Zugriff am 29 03 2022].

[10] „onlinesolutiongroup,“ [Online]. Available: https://www.onlinesolutionsgroup.de/blog/glossar/j/javascript/#:~:text=Die%20Skriptsprache%20wurde%20von%20den,erstmals%201995%20in%20Netscape%202.0.. [Zugriff am 11 02 2022].

[11] „Wikipedia,“ [Online]. Available: https://de.wikipedia.org/wiki/Hypertext_Markup_Language. [Zugriff am 21 03 2022].

[12] „Whatis,“ [Online]. Available: https://whatis.techtarget.com/definition/IFrame-Inline-Frame#:~:text=An%20IFrame%20(Inline%20Frame)%20is,advertisement%2C%20into%20a%20Web%20page.. [Zugriff am 24 03 2022].

[13] „w3schools.com,“ [Online]. Available: https://www.w3schools.com/js/. [Zugriff am 03 03 2022].

[14] „sitebulb,“ [Online]. Available: https://sitebulb.com/resources/guides/how-to-crawl-javascript-websites/. [Zugriff am 14 03 2022].

9 Anhang

Anhang A: **Code für die Studienarbeit**

Anhang B: **Beispielcode Falstadt**

Anhang C: **Gegebene kompilierte CircuitJS Version**

Anhang D: **ZIP-Datei um CircuitJS runterzuladen**

Anhang A:

Studienarbeit (1).html

Anhang B:

Circuit Simulator Applet.html

Anhang C:

circuitjs1-master.zip

Anhang D:

circuitjs1-master.zip