Universität der Bundeswehr München

Fakultät für Elektrotechnik und Technische Informatik

Institut für Funkkommunikation

Prof. Dr. -Ing. Erwin Riederer

Bachelorarbeit

Zur Erlangung des akademischen Grades

Bachelor of Engineering (B. Eng.)

Repository für labAlive – Parametrisierter Aufruf von Simulationen

Name:	Maximilian Frank
Adresse:	Pennstraße 7
	81549 München
Geburtsdatum:	09.12.1991
Matrikelnummer:	1197303
Jahrgang:	2019
Beginn der Erstellung:	04.04.2022
Tag der Abgabe:	27.06.2022

Erklärung

gemäß Beschluss des Prüfungsausschusses für die Fachhochschulstudiengänge der UniBwM vom 25.03.2010

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, insbesondere keine anderen als die angegebenen Informationen.

Neubiberg, den

_____

Maximilian Frank

Erklärung

gemäß Beschluss des Prüfungsausschusses für die Fachhochschulstudiengänge der UniBwM vom 25.03.2010

Der Speicherung meiner [Bachelor- oder Master-] Arbeit zum Zweck der Plagiatsprüfung stimme ich zu. Ich versichere, dass die elektronische Version mit der gedruckten Version inhaltlich übereinstimmt.

Neubiberg, den

_____

Maximilian Frank

Inhaltverzeichnis

Abkürzungsverzeichnis

AM	Amplitudenmodulation
EM	Einseitenbandmodulation
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
JNLP	Java Network Launching Protocol
LSB	Lower Sideband
URL	Uniform Resource Locator

Abbildungsverzeichnis

1 Einleitung

1.1 Aufgabenstellung und Motivation

In der heutigen Zeit nehmen Digitalisierung, ortsunabhängige Kommunikation und Vernetzung bis in das kleinste System eine sehr große Rolle ein und sind kaum noch wegzudenken. Um die Entwicklung und Weiterentwicklung dafür zu gewährleisten, ist es in den letzten Jahren nötig geworden herkömmliche Simulationsverfahren in Laboren durch verschiedene Simulationssoftware zu ersetzen. Diese sind zum einen günstiger, nicht ortsungebunden und vor allem langlebiger als teure Laborgeräte. Nicht zu vergessen, dass durch die Einsparung dieser Geräte auch weniger CO2-Emmisonen entstehen, was heutzutage bei der Planung eine Rolle spielt. Zwar ist es nach wie vor nötig in der finalen Phase einer Entwicklung diese physisch zu simulieren, allerdings ist es auf dem Weg dort hin meist effizienter die Simulation digital zu simulieren.

Eine Plattform, eines solchen Online-Labors, ist LabAlive, welche die Möglichkeit bietet Simulationen für kommunikationstechnische Aufgabenstellungen rechnergestützt zu simulieren. Sie bietet dem Nutzeden die Möglichkeit Nachrichten zu übertragen, diese dafür zu modulieren sowie zu demodulieren und sich an jeder Stelle der Übertragungsstrecke das Signal mittels Messgeräts anzeigen zu lassen [1] . Während und vor der Simulation ist es dabei möglich verschieden Parameter zu ändern, Anzeigeelemente zu- und abzuschalten, diverse Benutzereinstellungen zu ändern und die zu übertragende Nachricht einzuspeisen. Außerdem ist es möglich das Schaltbild anzupassen und eigene Schaltungen zu entwickeln.

Diese Arbeit befasst sich damit, welche Möglichkeiten es gibt diese Simulationen zu erstellen sowie diese mit verschiedenen Parametern vorzuinitialisieren. Außerdem wird darin die Möglichkeit beschrieben, wie ein Schaltbild visuell mittels gewünschten Layouts verändert werden kann.

Das Wort Parameter entstammt aus der lateinischen Sprache, welches aus den beiden Wortteilen para (steht für „bei, neben, gegen“ und meter (steht für „[das] Maß“) zusammengesetzt wird. Für die Informatik bzw. das Programmieren bedeutet der Begriff, dass einem Programm ein Argument bereits in Vorhinein übergeben wird. [2]

Ziel dieser Arbeit ist es außerdem das Online-Experimente „Analog Demodulation“ auf der Plattform mit parametrisierten Aufrufen zu versehen und visuelle Anpassungen vorzunehmen sowie die Erstellung einer Onlinehilfe zur Erzeugung von Layouts.

1.2 Aufbau dieser Arbeit

In Kapitel 2 dieser Arbeit wird es zunächst darum gehen, welche Software verwendet wurde sowie deren Möglichkeiten. Dabei wird Kapitel 2.1 die Online-Plattform labAlive beschreiben und welche Möglichkeiten diese dem Anwender bietet. Außerdem wird darauf eingegangen, welchen Mehrwert sie für den Nutzer generiert. In den darauffolgenden Kapiteln 2.2 und 2.3 wird die für labAlive verwendete Entwicklungsumgebung Eclipse IDE und „Java Web Start“ beschrieben.

In Kapitel 3 folgt die theoretische Arbeit. Darin werden die verschiedenen Möglichkeiten zur Erstellung parametrisierte Aufrufe detailliert erklärt, es wird das neue Feature von LabAlive „my Apps“ sowie dessen Features erläutert und am Ende des Kapitels wird auf das Thema Layouts eingegangen und welche Varianten es für die Anwenderin/den Anwender von my Apps und MyLabAlive gibt ein Layout zu generieren.

Das Online-Experimente „Analog-Demodulation“, welches durch parametrisierte Aufrufe sowie mit diversen Änderungen ergänzt wurde folgt in Kapitel 4. Anschließend wird die erstellte Onlinehilfe zur Generierung eigener Layouts und deren Aufbau erläutert.

Als letztes folgt ein Fazit zu dieser Arbeit.

2 Verwendete Software

Dieses Kapitel beschreibt die verwendeten Softwares, welche für diese Arbeit genutzt wurden. Dabei wird auf die jeweilige Software, deren Versionen und den verwendeten Softwarekomponenten eingegangen.

2.1 LabAlive

LabAlive ist eine Online-Plattform, welche sich mit der Darstellung sowie der Simulation verschiedener kommunikationstechnischer Verfahren beschäftigt. Sie dient als eine Art virtuelles Labor, welches in der heutigen Zeit von großem Nutzen für Anwender insbesondere Studierende ist. Der Link, um diese Plattform zu erreichen lautet https://www.etti.unibw.de/labalive/.

Entwickler von labAlive ist Herr Prof. Dr.-Ing. Erwin Riederer vom Institut für Funkkommunikation an der Universität der Bundeswehr München. Bis zu dem heutigen Tag entwickelt er verschiedene Experimente, um die Plattform immer auf dem neusten Stand der Technik zu halten. Unterstützt wird er hierbei von seinen Mitarbeitern sowie Studenten, welche bei ihm studentische Arbeiten absolvieren können. Neben dem Vorteil, dass man von überall Zugriff auf diese Online-Plattform hat, sind weitere Vorteile, zum einen, dass der Nutzer die Simulationen so oft wie gewünscht aufrufen kann und somit nicht nur für einen begrenzten Zeitraum Zugriff darauf hat, zum anderen kann an teuren Laborgeräten gespart werden. Für Studenten bietet labAlive die Möglichkeit vorlesungsbegleitend Inhalte besser zu verstehen, mittels Praktika zu intensivieren und selbst Simulationen für die gewünschten Zwecke zu erstellen. Als Voraussetzung für das Nutzen der Simulationen wird lediglich ein Rechner benötigt, auf welchem „Java Runtime Environment“ und „IcedTea-Web“ installiert sind.

Dabei sind die Inhalte meist wie folgt aufgebaut. Die Nutzerinnen und Nutzer starten mit einem Tutorial. Darauf aufbauend kann im Anschluss mit dem neu erlangten Wissen ein Experiment sozusagen ein Versuch durchgeführt werden. Dadurch kann er nachvollziehen, ob er das Verfahren verstanden hat und auch anwenden kann. In manchen Fällen folgt darauf noch eine graphische Illustration der verschiedenen Möglichkeiten des Versuchs. Experimente bei denen es sich anbietet enden mit

einem Quiz um das erlangte Wissen abzuprüfen. Durch das Tutorial wird die Bedienung der Simulationen sehr gut erklärt. Zum Starten einer Simulation verwendet die Homepage einen Launch-Knopf. Durch Anklicken wird der Download der Simulation, in Form einer JNLP-Datei, gestartet. Nach dem Ausführen dieser Datei öffnet sich die gewünschte Simulation.

Das Repertoire von Versuchen ist äußerst groß, es reicht von allen möglichen Modulationsverfahren, über Methoden ein Zeitsignal in ein Spektrum zu transformieren bis hin zu komplexen Multiplex-Verfahren.

Durch das von Herr Prof. Dr.-Ing. Erwin Riederer zur Verfügung gestellte Skript „Simulation kommunikationstechnischer Systeme“ hat die Nutzerin/der Nutzer einen Überblick darüber, wie die Simulationen bedient und dessen Features verwendet werden, wie der Aufbau einer Simulation ist, wie die Entwicklung neuer Systeme vonstatten geht und wie neue Messgeräte erstellt werden können. Für das Erstellen neuer Simulationen ist die Eclipse Workspace für labAlive zu verwenden. In dieser Entwicklungsumgebung sind alle bis dato existierende Methoden und Klassen enthalten.

2.2 Eclipse

Das open-source Programm Eclipse dient zur Softwareentwicklung verschiedener Anwendungsbereiche. Neben dem Einsatz als integrierte Entwicklungsumgebung (IDE) für Java wird Eclipse heutzutage für eine Vielzahl anderer Programmiersprachen verwendet. Für diese Arbeit ist dabei die Programmiersprachen HTML zu benennen [3].

2.3 Java Web Start

Damit Anwendungen aus dem Internet heruntergeladen und gestartet werden können wird Java Web Start benötigt. Es dient unter anderem dazu, dass eine Anwendung per Mausklick direkt gestartet werden kann und die Anwenderin/der Anwender sich Installationen sowie Upgrades erspart, da Java Web Start sicher stellt immer die aktuelle Version der Anwendung auszuführen [4].

Im Falle von LabAlive werden auf diese Art und Weise Simulationen von Online-Experimenten heruntergeladen und gestartet. In Abbildung 1 sieht man einen Standardinhaltsbereich eines Online-Experiments. Mittels Mausklick auf den Launch-Button wird der Download der jeweiligen JNLP-Datei gestartet. Dabei wird die JNLP-Datei mit Java Web Start geöffnet und ausgeführt. Das JNLP-File enthält unter anderem Informationen zur Remote-Adresse für den Download der Java-Datei sowie die erste Klasse, die ausgeführt werden soll [5].

Abbildung 1: Standardinhaltsbereich eines Online-Experiments [6]

3 Theoretische Arbeit

3.1 Möglichkeiten parametrisierte Aufrufe zu realisieren

In diesem Kapitel werden die Möglichkeiten beschrieben einen parametrisierten Simulationsaufruf zu realisieren. Zum einen bietet die Onlineplattform labAlive die Möglichkeit mit myLabAlive eine eigene Simulation mittels Text zu erzeugen zum anderen gibt es die Möglichkeit schon vorhandene Simulationen zu verwenden und mit den gewünschten Parametern zu ergänzen. Als neues Feature gibt es des Weiteren die Möglichkeit mittels my Apps eigene Simulationen zu generieren und zu verwalten. Diese Verfahren werden in den folgenden Unterkapiteln genauer beschrieben.

3.1.1 MyLabAlive

MyLabAlive bietet die Möglichkeit eine eigene Simulation, für die eigenen Anforderungen und gewünschten Features, mittels Text zu erstellen. Die Nutzerin/der Nutzer hat dabei die Möglichkeit aus einem breiten Spektrum von Systemen, Messgeräten, Eingangssignalen und Verbindungsmöglichkeiten zu schöpfen.

Auf der Homepage von MyLabAlive findet man eine detaillierte Beschreibung über die verschiedenen Möglichkeiten, welche MyLabAlive bietet. Da aktuell eine weitere studentische Arbeit MyLabAlive aufarbeitet und den Onlineauftritt überarbeitet wird in dieser Arbeit von einer tieferen Beschreibung abgesehen. [7]

Wie eine Simulationsbeschreibung in MyLabAlive aussieht, ist in Abbildung 1 zu sehen. Hier wird eine einfache Schaltung erzeugt in diesem Beispiel der Empfang und die Weiterverarbeitung eines Signals in einem Satelliten mit einem Transparent Repeater. Neben der eigentlichen Beschreibung der Simulation wird unter preferences die grafische Darstellung der Graphen verändert.

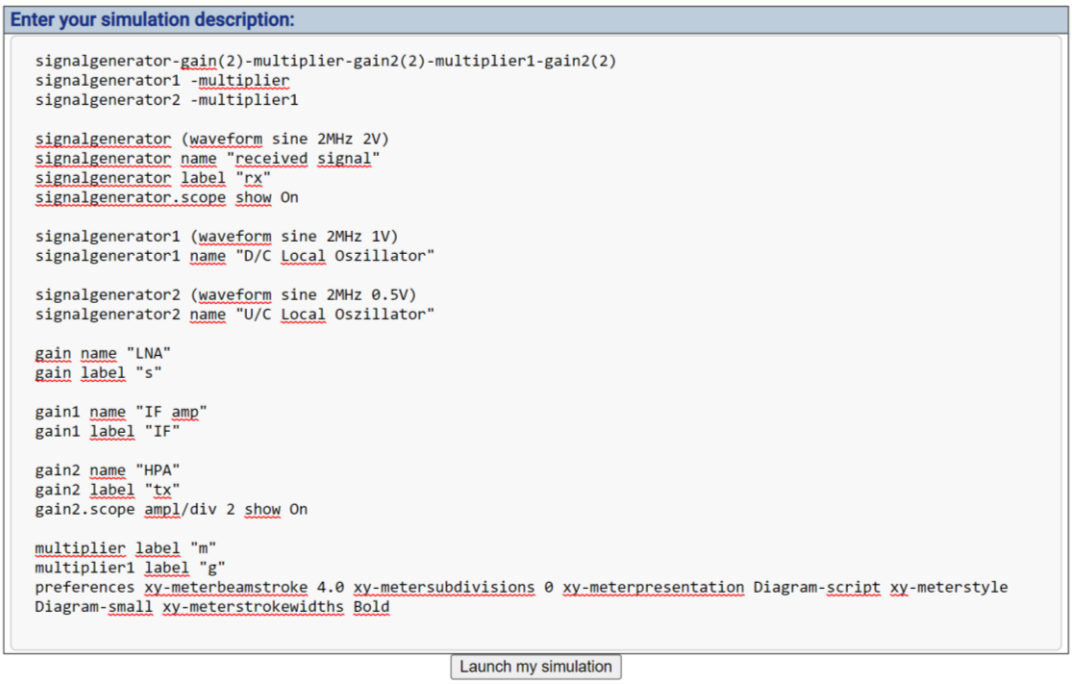


Abbildung 2: Simulationsbeschreibung eines Satelliten mit einem Transparent Repeater

Im Folgenden wird dieses Verfahren mit Hilfe des in Abbildung 2 gezeigten Beispiel erläutert.

Am Anfang einer Simulationserzeugung werden die Systeme, welche verwendet werden, verbunden und teilweise initialisiert. Dazu werden zeilenweise die Systemketten aufgeführt. Diese Verbindung der Systeme ist im Beispiel von Zeile 1-3 zu sehen. Als nächstes werden nacheinander die Systeme genauer beschrieben. Im Beispiel sind diese Systembeschreibungen von Zeile 5 – 27 zu sehen dabei sind die einzelnen Systembeschreibungen mit einer Leerzeile voneinander getrennt, dies dient zur Übersichtlichkeit. Neben den Parametern, mit welchen die einzelnen Systeme versehen werden, hat die Anwenderin/der Anwender mehrere Möglichkeiten die Systeme zu personalisieren und mögliche Messgeräte beim Start der Simulation bereits anzeigen zu lassen sowie deren Justierungen. Am Ende einer Simulationsbeschreibung in myLabAlive hat die Anwenderin/der Anwender außerdem die Möglichkeit die angezeigten Graphen graphisch zu verändern. Diese werden hinter „preferences“ angefügt, wichtig hierbei ist, dass keine Zeilenumbrüche dazwischen verwendet werden. Andernfalls werden alle folgenden Einstellungen der Graphen nicht mehr berücksichtigt. Diese „preferences“ sind im Beispiel von Zeile 28-29 eingefügt. In der folgender Abbildung 3 ist das Ergebnis der Simulationsbeschreibung von Abbildung 2 zu sehen.

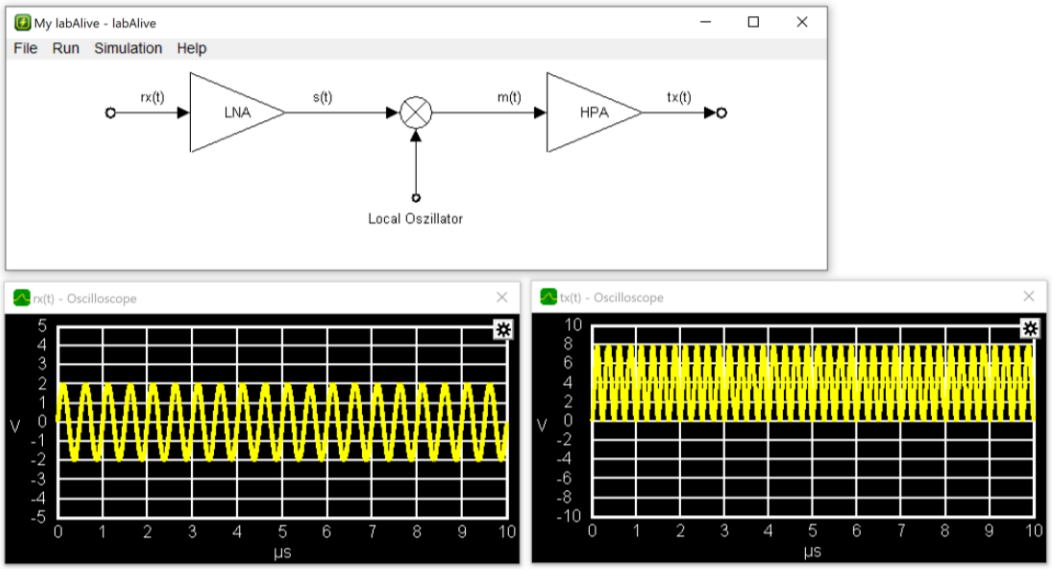


Abbildung 3: Simulation der vorangegangenen Simulationsbeschreibung mittels myLabAlive

Damit die Nutzerinnen und Nutzer einen Überblick über die möglichen Systeme, Signale, Messgeräte und mögliche Diagrammeinstellungen gewinnt ist in myLabAlive eine genau Beschreibung dazu vorhanden, welche anhand von Beispielen komplettiert wird.

3.1.2 Parametrisierter Simulationsaufruf mittels URL

In diesem Unterkapitel wird ein weiteres Verfahren beschrieben, wie ein parametrisierter Simulationsaufruf erzeugt werden kann. Dabei handelt es sich um die Erzeugung einer URL, welche neben der verwendeten Simulation auch die von der ursprünglichen Simulation abweichenden Parameter beinhaltet.

3.1.2.1 myLabAliveUserChangedParameters

Bevor die URL erzeugt werden kann, werden die veränderten Parameter in Textformat benötigt, um daraus eine URL zu erzeugen. Hierfür hat labAlive eine Funktion die diese Änderungen, die während eines Simulationsaufrufes getätigt wurden, in die Text-Datei myLabAliveUserChangedParameters abspeichert.

Die Speicherung der veränderten Parameter kann auf zwei Wegen erfolgen zum einen mit der Tastenkombination STRG+S zum anderen durch manuelles

Auswählen der Speicherung, welche in der Symbolleiste unter File „Save“ ausgewählt werden kann (siehe Abbildung 4).

Durch Speicherung wird der vergangene Inhalt der Datei mit den neuen Änderungen überschrieben. Dieses Dokument ist im Ordner myLabAlive abgelegt, welcher über den Pfad C:\Users\Username\labAlive\myLabAlive zu finden ist. Dabei werden allerdings nur die Veränderungen abgespeichert, die von der ursprünglichen Simulation abweichen.

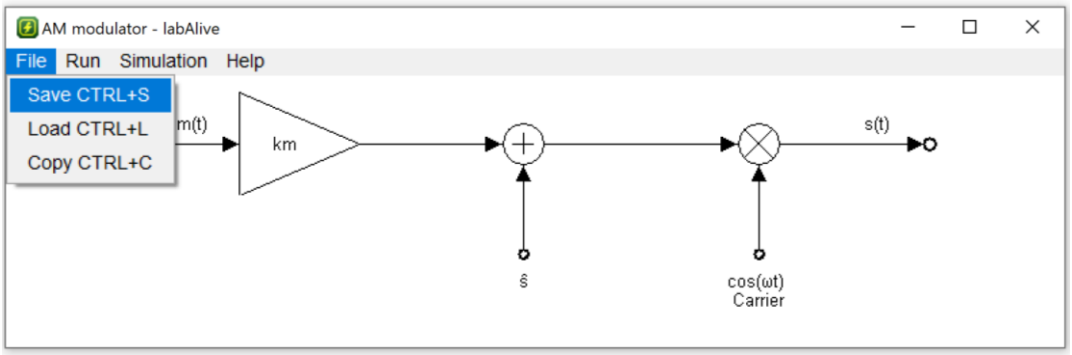


Abbildung 4: Online-Experiment zur AM

In Abbildung 4 ist das Online-Experiment zur Amplitudenmodulation (AM) zu sehen sowie die Möglichkeiten der Speicherung in die Textdatei, wie Diese beispielsweise befüllt aussehen kann ist in Abbildung 5 zu sehen [8].

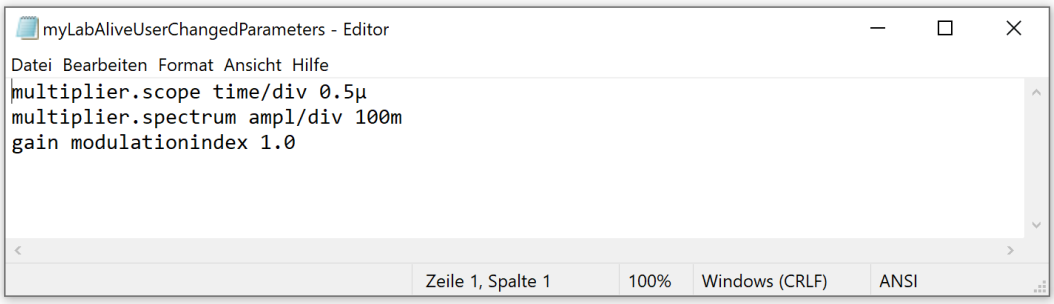


Abbildung 5: myLabAliveUserChangedParameters mit Änderungen bei AM

Für die Erstellung eines parametrisierten Aufrufs ist es bei diesem Verfahren nun notwendig die gewünschten Parameter während eines Simulationsaufrufs zu ändern, um diese Änderungen aus der Textdatei zu gewinnen. Alternativ wäre es möglich diese Änderungen per Hand selbst zu schreiben dazu wäre allerdings ein

genaues Wissen der Syntax in labAlive Voraussetzung und um ein Vielfaches zeitintensiver als dieses Verfahren.

3.1.2.2 URL-Encoder

Die im Unterkapitel zuvor gewonnen Änderungen, die in der Textdatei dokumentiert sind, müssen im nächsten Schritt zur Erstellung einer URL nun in das nötige Format überführt werden. Dazu ist es von Vorteil einen URL-Encoder zu verwenden. Alternativ ist es auch hier möglich dies selbst durchzuführen. In Abbildung 5 wurden die Änderungen, welche in der Textdatei vorgenommen wurden, nun in das URL-Format übersetzt [9].

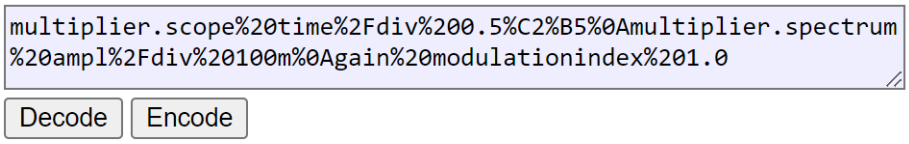


Abbildung 6: generiertes URL-Format mittels URL-Encoder

3.1.2.3 Erstellung der URL

Als letzter Schritt, zur Generierung eines parametrisierten Aufrufs, muss nun eine vollständige URL erzeugt werden. Dazu wird die ursprüngliche URL des Online-Experiments nun mit dem, in Unterkapitel 3.2.2, generierten String ergänzt. Dazu wird als Verbindung die Symbolfolge „?p=“ benötigt, welche auf darauffolgende Benutzeränderungen hinweist. Als Ergebnis dieser Zusammenführung erhält man in unserem Beispiel die URL: „https://www.etti.unibw.de/labalive/web-start/am.jnlp?p=multiplier.scope%20time%2Fdiv%200.5%C2%B5%0Amultiplier.spectrum%20ampl%2Fdiv%20100m%0Again%20modulationindex%201.0“

3.1.3 Parametrisierter Aufruf mit My Apps

Als letzte Möglichkeit, einen parametrisierten Aufruf zu erstellen, ist myApps zu benennen. Auf die genaue Beschreibung dieses neuen Features von LabAlive wird in dem folgenden Kapitel detailliert eingegangen.

Für die Erstellung eines parametrisierten Aufrufs bietet es dem Anwender die Möglichkeit die zuvor beschriebenen Verfahren zu nutzen und darüber hinaus.

Außerdem bietet dieses Verfahren den entscheidenden Vorteil, dass man schnell überprüfen kann ob und wie die veränderten Parameter funktionieren.

3.2 My Apps

Dieses Unterkapitel thematisiert das neue Feature von LabAlive. Die Idee dahinter ist es der Anwenderin und dem Anwender eine Möglichkeit zu verschaffen ein Profil zu erstellen, in welchem sie/er die erstellten Simulationen abspeichert, um diese direkt bei Bedarf erneut aufrufen zu können. Des Weiteren hat sie/er nun die Option ihre/seine Simulationen jederzeit zu editieren, in der eigenen Bibliothek zu verwalten und diese mit anderen Anwenderinnen und Anwendern zu teilen. Neben der Erstellung eigener Simulationen hat sie/er außerdem die Möglichkeit bestehende Experimente, welche man auf labAlive finden kann, für die persönlichen Präferenzen zu verändern und ebenfalls auf dem erstellten Profil abzuspeichern. Mit diesem neuen Feature ist die Möglichkeit, für die Nutzerin/ dem Nutzer einen parametrisierten Simulationsaufruf zu erstellen, um ein Vielfaches anwendungsfreundlicher als es bisher der Fall war. Im Folgenden wird dieses Feature genauer beleuchtet. [10]

3.2.1 Eigenes Profil erstellen und verwalten

Bevor die Anwenderin/ der Anwender mit dem Erstellen ihrer/seiner persönlichen Simulationssammlung beginnen kann ist es nötig ein Profil anzulegen. Dazu ist eine Registrierung auf labAlive notwendig. Da hier keine sicherheitsrelevanten Informationen oder Ähnliches anfallen wird auf der Homepage auf jegliche Art persönlicher Informationen verzichtet. Die einzige Option, welche der Anwender hat, ihr/sein Profil zu personalisieren, ist die Verwendung eines Benutzernamens. Durch Anklicken des Buttons „Register“ wird die Registrierung durchgeführt (siehe Abbildung 7).

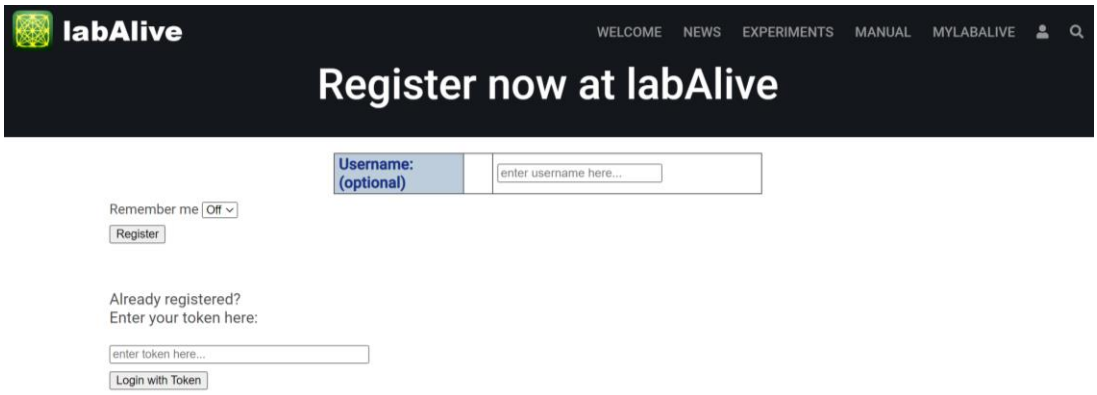


Abbildung 7: my Apps Registrierung

Nach erfolgreicher Registrierung bekommt die Anwenderin/ der Anwender einen persönlichen Token, mit welchem er jederzeit auf ihr/sein Profil zugreifen kann und diesen weiterleiten kann damit andere Userinnen und User Zugriff auf die Bibliothek erhält. Allerdings ist dies mit Vorsicht zu genießen da man mit diesem Token die gleichen Zugriffsrechte hat, wie man selbst. Dadurch besteht die Möglichkeit, dass in den falschen Händen die Simulationen verändert, unbrauchbar oder sogar gelöscht werden können, ohne vorausgehender Zustimmung.

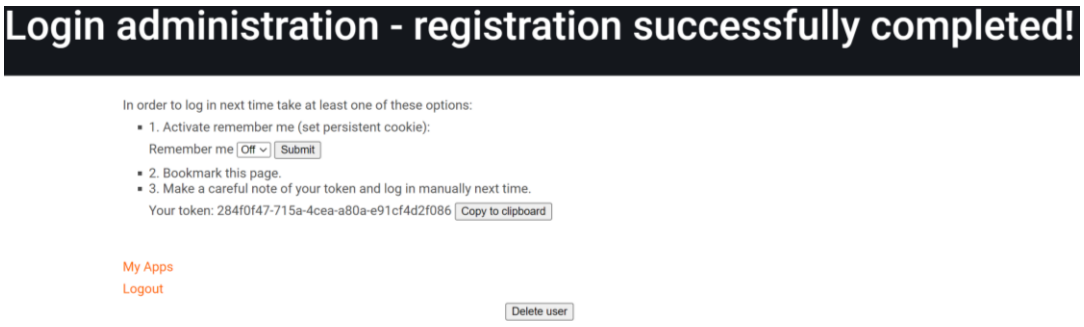


Abbildung 8: Login Administration nach erfolgreicher Registrierung

Für ein späteres erneutes Aufrufen des Profils gibt es drei mögliche Option zu wählen. Als Erstes, dass die Seite sich das angelegte Profil abspeichert, dazu setzt sie ein anhaltendes Cookie, um diese Option zu aktivieren gibt es unter dem 1. Punkt die Möglichkeit „remember me“ auszuwählen. Die zweite Option ist sich mittels Tokens anzumelden, diesen also abzuspeichern. Und als letztes die Möglichkeit die URL abzuspeichern, welche nach erfolgreicher Registrierung ersichtlich ist. Als Letztes besteht jederzeit die Möglichkeit durch „Delete user“, dass der Account gelöscht wird (siehe Abbildung 8).

3.2.2 Aufbau

In diesem Unterkapitel wird der Aufbau von „myApps“ erläutert und auf die diversen Features tiefer eingegangen. Das Profil ist zum derzeitigen Zeitpunkt in zwei Seiten unterteilt. Auf der ersten Seite namens „my Apps“ hat die Userin/ der User einen Überblick über ihre/seine bis dahin erstellten und abgespeicherten Simulationen. Dabei sind diese durch drei Kategorien beschrieben zum einen durch den Titel in der Beispielsimulation, welche in Abbildung 7 zu sehen ist, lautet dieser „Beispiel“. Zum anderen mit der Wiring Definition hier „signalgenerator-gain-sink“. Und als letzte Möglichkeit mittels Erstellungsdatum. Da sich im Laufe der Zeit eine große Zahl von Schaltungen ansammeln können kann dadurch eine Suche vereinfacht werden. Die Sortierung kann dabei durch anklicken der jeweiligen Spalten „Title“, „Wiring Definition“ und „Puplished Date“ verändert werden. Durch die Spalte „Launch“ kann entweder durch das Anklicken des Play-Buttons die Simulation direkt gestartet werden oder durch Anklicken von „Edit App“ die vorhandene Simulationsbeschreibung (siehe Abbildung 10) editiert werden.

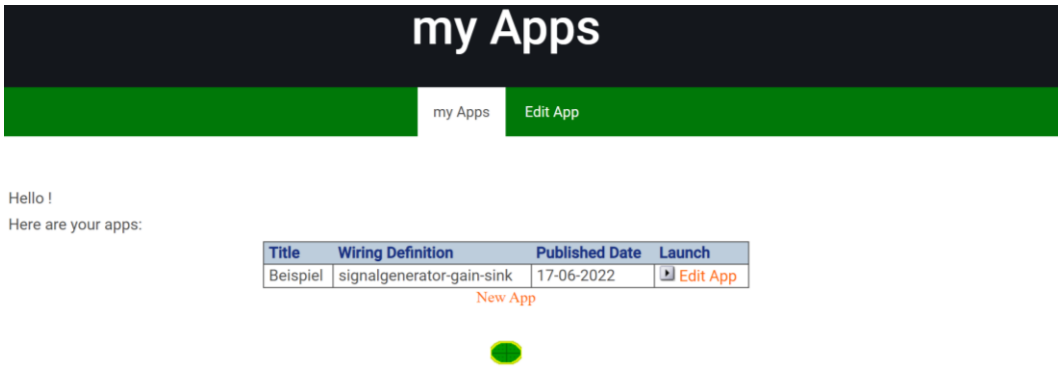


Abbildung 9: my Apps Seitenaufruf my Apps

Auf der zweiten Seite „Edit App“ kann sie/er eine neue Simulation anlegen oder ein bereits bestehendes Experiment editieren (siehe Abbildung 10). Im Folgenden wird die zweite Seite genauer beschrieben und auf alle Features eingegangen.

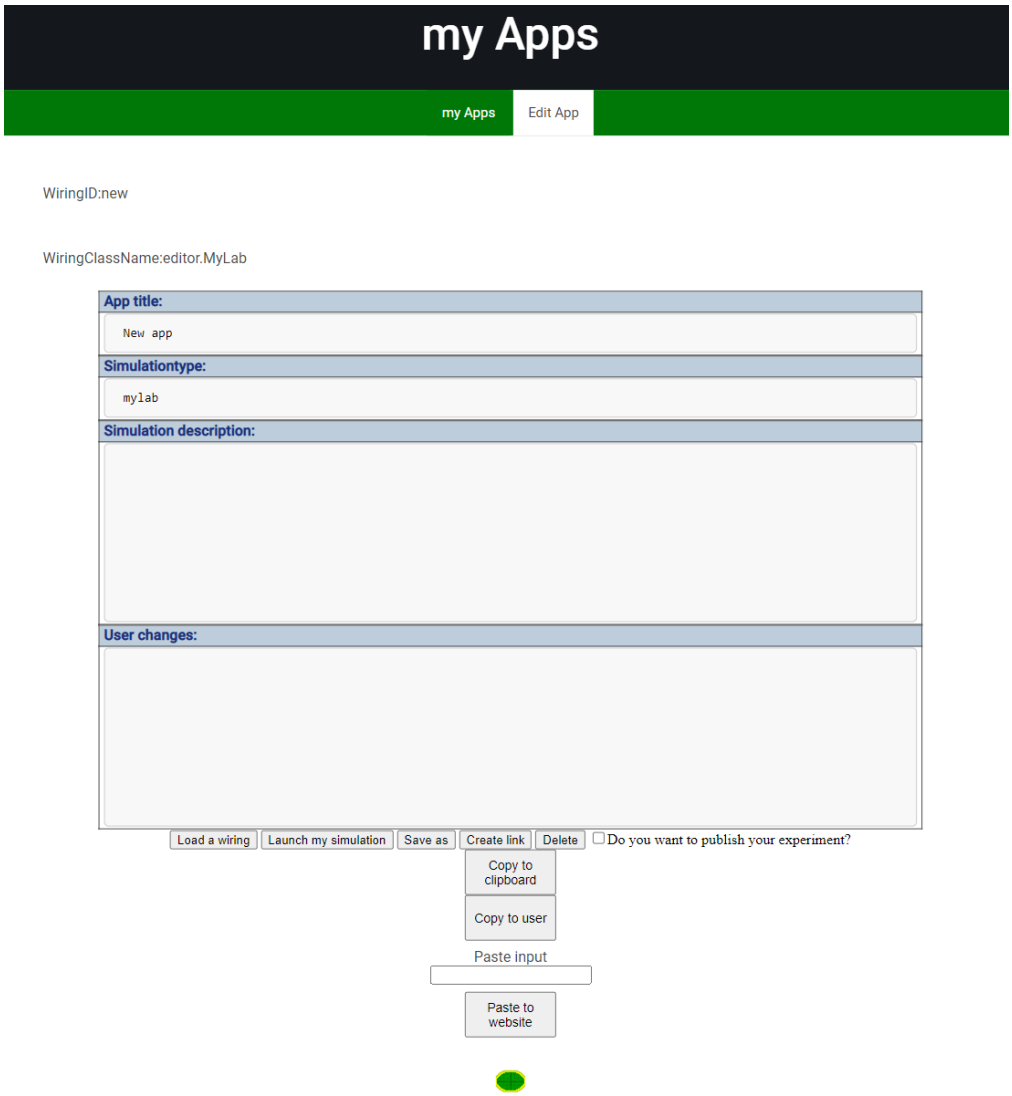


Abbildung 10: my Apps Seitenaufruf Edit App

Im ersten Feld „App title“ kann die Nutzerin/ der Nutzer die Simulation benennen. Das nächste Feld „Simulationstyp“ bietet die Möglichkeit entweder mit dem Begriff „mylab“ eine komplett neue Simulation zu erstellen, wie bei MyLabAlive oder ein bestehendes Experiment und dessen Simulation, wie bspw. im Falle des AM-Experiments, zu verwenden. Um ein solches Experiment zu verwenden ist es nötig den richtigen JNLP-Dateinamen zu verwenden. Diesen kann die Anwenderin/der Anwender auf der jeweiligen Seite des Experiments aus der URL entnehmen. Auf Abbildung 8 ist anhand des Beispiels der AM-Modulation die Stelle in der URL markiert, welche diese Information beinhaltet.

Abbildung 11: URL des Experiments der AM mit markierter Stelle der JNLP

Die eigentliche Simulation wird im Feld „Simulation description“ beschrieben. Darin wird die Schaltung zusammengesetzt, die Messgeräte beschrieben und die Systeme sowie Signale initialisiert und benannt. Im letzten Feld „User changes“ hat man des Weiteren die Möglichkeit die sogenannten „preferences“ also die gewünschte Darstellung der Diagramme zu ändern.

Des Weiteren ermöglicht es diese Seite eine Simulation zu laden, zu starten, abzuspeichern, einen URL dazu zu erstellen oder zu löschen. Auf die weiteren Features, welche hier außerdem möglich sind und die zur Anwenderfreundlichkeit der Seite enorm betragen, werden im Folgenden genauer erläutert.

3.2.3 Features

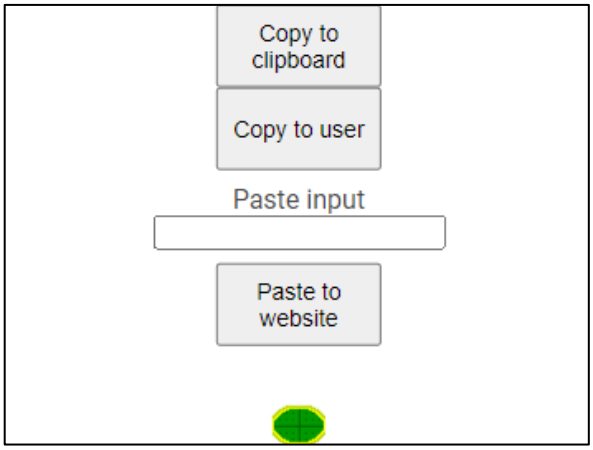


Abbildung 12: Features von my Apps

3.2.3.1 Copy to clipboard

Das erste Feature „Copy to Clipboard“ bietet die Möglichkeit die aktuelle Simulationsbeschreibung der App in die Zwischenablage dem sogenannten „Clipboard“ abzulegen. Ruft man nun beliebige zuvor gestartete Simulation auf, werden sofort die im Clipboard gespeicherte Simulation gestartet. Dabei ist es unbedeutend um welche Simulation es sich zuvor gehandelt hat. Dies erleichtert es der Anwenderin/dem Anwender nicht bei jeder Änderung eine Simulation herunterzuladen, sondern sie/er bedient sich der Zwischenablage in Verbindung einer bereits gestarteten Simulation. Insbesondere bei einer Fehlersuche ist dies von äußerstem Nutzen. Dieser Inhalt des Clipboards kann außerdem genutzt werden, um eine Simulationsbeschreibung von my Apps in Textform abzuspeichern oder so die Simulation an

Dritte weiterzusenden. Dieser Text kann dann in das Feature „paste input“, welches in diese Kapitel auch beschrieben wird, eingefügt werden. Wie ein solcher Text beispielsweise aussieht ist im Folgenden zu sehen.

```
*labAlive-app*
de.labAlive.wiring.editor.MyLab
id=42&t=Layout%203%20Splitter%20-%20Combined%20layout%20-%20Fehler%20beim%20Starten&s=mylab&w=sine%20%3E%20split%20%3E%20sink%0Asplit%20v%3E%20sink2&p=&pub=false&u=e741a868-7728-4e5b-b4e2-eec305708616&v=2
```

3.2.3.2 Copy user

Das nächste Feature ist der Button „copy user“ hierbei handelt es sich lediglich darum den Token des aktuell eingeloggten Users in der Zwischenablage zu speichern

3.2.3.3 Paste input/Paste to website

Das neue Feature, welche es der Anwenderin/dem Anwender während einer Simulation ermöglicht diesen aktuellen Simulationsaufruf in die Zwischenablage zu kopieren, kommt auch in my Apps zur Anwendung. Mittels Tastenkombination STRG+C oder manuelle Auswahl von Kopieren wird dies durchgeführt. Der kopierte Inhalt in der Zwischenablage kann im Anschluss in das Eingabefenster „Paste input“ eingefügt werden. Durch Klicken des Buttons „Paste to website“ wird nun die Simulation in my Apps mit dieser Simulationsbeschreibung befüllt.

3.2.3.4 Weitere Möglichkeiten

Des Weiteren ist es möglich die Änderungen, welche während der Simulationsausführung getätigt wurden, bei Speicherung mittels manueller Betätigung oder der Tastenkombination STRG+S automatisch in my Apps einzufügen. Dazu ist zu diesem Zeitpunkt die Überlegung das Speichern in die Text-Datei zu entfernen da sie mit dieser Erweiterung redundant geworden ist.

3.3 Layout

In diesem Kapitel wird beschrieben, welche Möglichkeiten der Nutzer hat selbst ein Layout in my Apps und myLabAlive zu erstellen. Bezüglich dem Thema Layouts, und wie diese generiert werden bzw. wie diese programmtechnisch realisiert wurden gingen dieser Bachelorarbeit mehrere Arbeiten voraus.

Zum einen die Arbeit „Layout Generator und GUI Erweiterungen für labAlive Blockdiagramme und Fenster“, welche den Layout Generator erst theoretisch beschreibt, als nächstes auf die Anordnung der Systeme eingeht und als letztes, wie der Layoutgenerator umgesetzt wurde. [10]

Die Arbeit „Blockdiagramm Layout von labAlive – Simulationen. Konzepte und Anwendungen“ ist in diesem Zusammenhang außerdem zu benennen. Darin geht es um die Layoutdefinitionen der FULL- und COMPRESSED-Methoden und der neu dazugekommenen COLLAPSED-Methode [11].

Diese Arbeit baut auf diesem Wissen auf und wird nicht jede Methode erneut beinhalten, sondern nur jene, welche für den Anwender von my Apps und myLabAlive relevant sind und wie diese erweitert wurden.

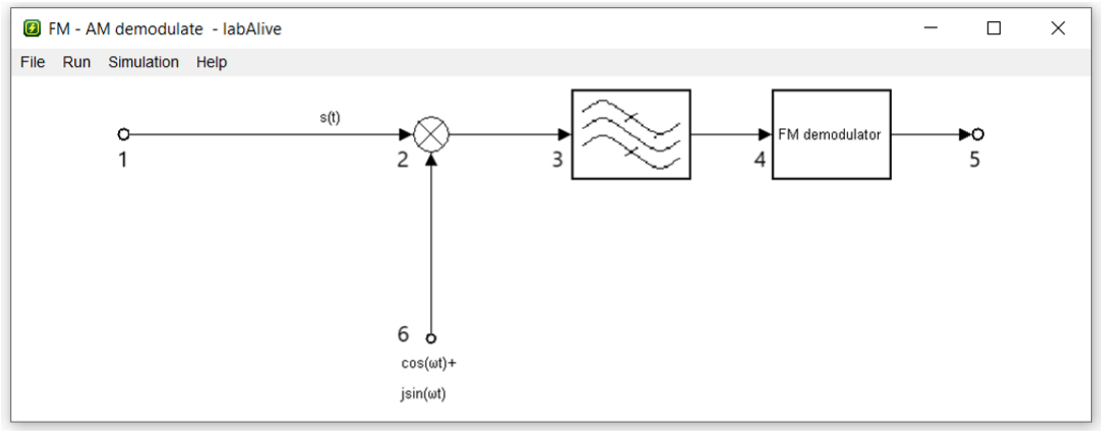
In diesem Kapitel wird zunächst darauf eingegangen was genau ein Layout ist und dessen Syntax. Als nächstes wird die FULL-Methode beschrieben, die für die Anwenderin/den Anwender in my Apps zur Verfügung steht und wie diese erweitert wurde, um benutzerfreundlicher für die Anwenderin/den Anwender zu sein. Als Letztes wird ein Ausblick Ideen und mögliche Ansätze für künftige Erweiterungen geben.

3.3.1 Beschreibung

Ein Layout beschreibt, wie ein Blockdiagramm und die darin enthaltenen Systeme positioniert und miteinander verbunden sind. Dabei gibt es wie bereits erwähnt verschiedene Methoden. Für den Nutzer von my Apps und myLabAlive ist dabei nur die FULL-Methode relevant. Den genauen Aufbau und deren Struktur wird in den Kapiteln 3.3.3 ff. erläutert. Davor wird in Kapitel 3.3.2 auf die Layoutdefinition eingegangen.

3.3.2 Layoutdefinition

In der Layoutdefinition legt die Anwenderin/der Anwender fest, wie die einzelnen Systeme einer Schaltung angeordnet sind. Zuvor wird in der LayoutId festgelegt, welche Systeme miteinander verbunden werden.



$$1 - 2 - 3 - 4 - 5, 6 - 2 \; ; \; 1 > 2 \; d \; 3 \; d \; 4 \; d \; 5, 6 \; > 2$$

Abbildung 13: Layoutdefinition mit vorangegangener LayoutId am Beispiel des Experiments der Analog-Demodulation-Challenge [6]

In Abbildung 8 ist eine solche Layoutdefinition mit vorangegangener LayoutId am Beispiel des Experiments der Analog-Demodulation-Challenge zu sehen. Dabei handelt es sich um den Teil vor dem Semikolon um die LayouId und dem darauf folgenden Teil um die Layoutdefinition. Für ein besseres Verständnis sind im Schaltbild die einzelnen Systeme nummeriert. Dabei stehen die Zahlenwerte für das jeweilige System. Die Symbole zwischen den einzelnen Zahlen in der Layoutdefinition stehen für die Richtungsanweisung. Durch ein Komma wird signalisiert, dass dieser Pfad zu Ende ist und ein neuer Pfad beginnt.

Definition der Richtungsanweisung:

Symbol	Bedeutung
^	2 Schritte nach oben
v	2 Schritte nach unten
>	2 Schritte nach rechts
<	2 Schritte nach links
w	1 Schritt nach oben
s	1 Schritt nach unten
d	1 Schritt nach rechts
a	1 Schritt nach links

Die Einzelschritte kommen nur in Sonderfälle zur Anwendung.

3.3.3 FULL-Methode

Die FULL-Methode beschreibt ein Layout, ohne Dieses zu komprimieren. Das bedeutet, dass in dieser Art von Methode alle Systeme, welche im Schaltbild sind, auch im Layout enthalten sein müssen. Dies macht es der Anwenderin/dem Anwender einfacher ein Layout zu erstellen.

Die beiden anderen Methoden COMPRESSED und COLLAPSED, welche es in LabAlive außerdem gibt, finden bei der Nutzerin/dem Nutzer keine Anwendung. Die folgenden Varianten, welche die Nutzerin/der Nutzer von my Apps und MyLabAlive basieren auf dieser FULL-Methode.

3.3.4 Varianten zur Layouterstellung in my Apps und MyLabAlive

Für die Anwenderinnen und Anwendern von my Apps und myLabAlive ist nur die Layoutdefinition relevant. Dafür hat sie/er zwei mögliche Verfahren die „Combined Method“ und die „Explicit Method“. Die Nutzerin/der Nutzer benötigen für die Erstellung eines Layouts in my Apps und MyLabAlive nur eine Layoutdefinition, die LayoutId wird nicht mehr benötigt. Serverseitig wird für die Erstellung des Layouts weiterhin mit der Layoutdefinition mit vorangegangener LayoutId gearbeitet,

dazu wird die erstellte Layoutdefinition in my Apps mittels den beiden Varianten in das ursprüngliche Layout umgewandelt.

3.3.4.1 Explicit Method

Diese Methode arbeitet mit dem bisherigen Prinzip die Systeme mit deren jeweiligen Nummerierungen anzusprechen. Das Layout wird nach der Verbindung der einzelnen Systeme beschrieben. In Abbildung 5 ist die Simulationsbeschreibung der in Abbildung 14 erstellten Simulation zu sehen und an welcher Stelle die Layoutdefinition beschrieben wird. Vor dieser Layoutdefinition werden die verschiedenen Systeme mittels „-“ verbunden, wichtig dabei ist es die einzelnen Wege der Schaltung per Zeilenumbruch voneinander zu trennen. Für ein besseres Verständnis, wie die einzelnen Systeme nummeriert werden, sind in Abbildung 14 die Systeme mit der jeweiligen Nummer versehen, mit der sie in der Layoutdefinition angesprochen werden.

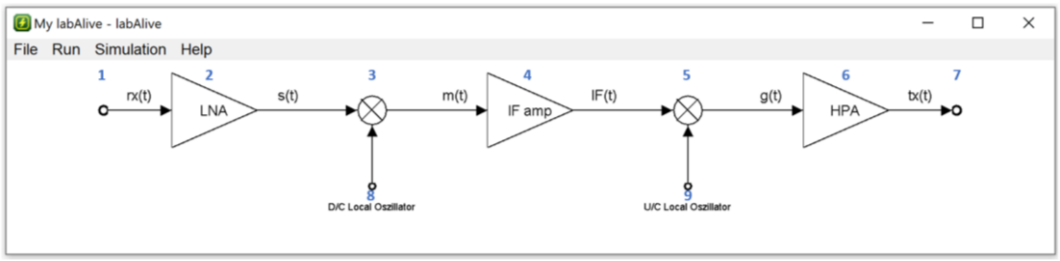


Abbildung 14: Simulationsaufruf mit eingefügter Nummerierung der Systeme

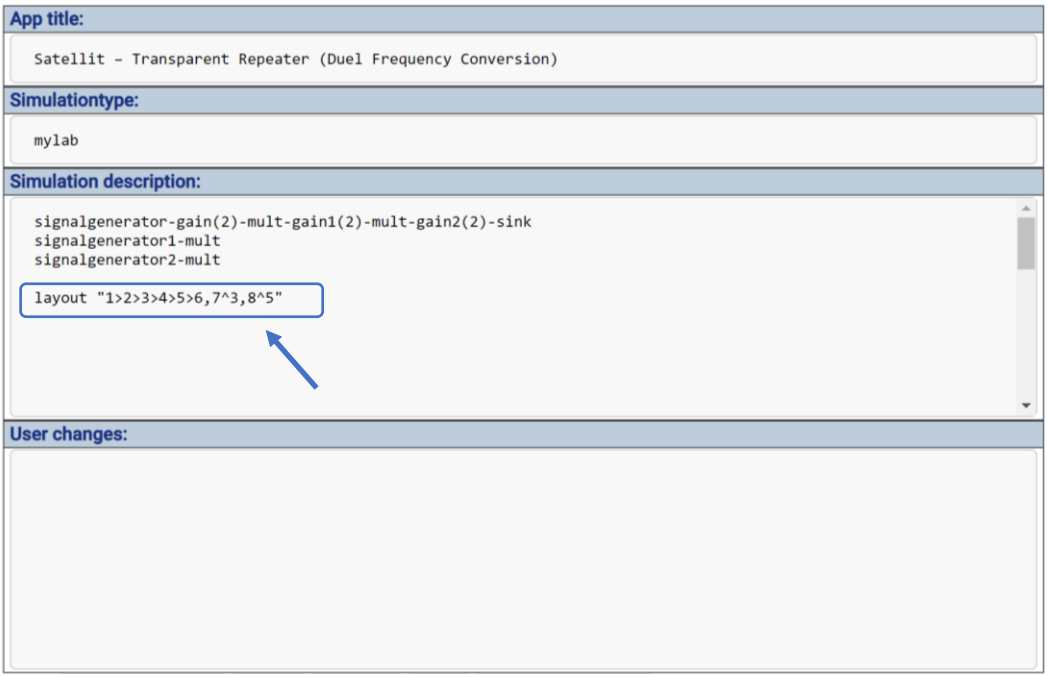


Abbildung 15: exemplarische Layoutdefinition mit alter Methode in my Apps
Verwendete Simulation: "Satellite – Transparent Repeater (Duel Frequency Conversion)"

3.3.4.2 Combined Method

Die Idee hinter dieser Methode ist es die Verbindung der einzelnen Systeme mit der Layoutdefinition zu vereinen. Dazu werden die „-“ zur Verbindung der Systeme mit der „Explicit Method“ durch die jeweiligen Richtungsanweisungen ersetzt. Dies erspart eine Nummerierung der Systeme sowie die separate Erstellung der Layoutdefinition. Wie diese Variante innerhalb einer Simulationsbeschreibung aussehen könnte, ist in Abbildung 17 hervorgehoben.

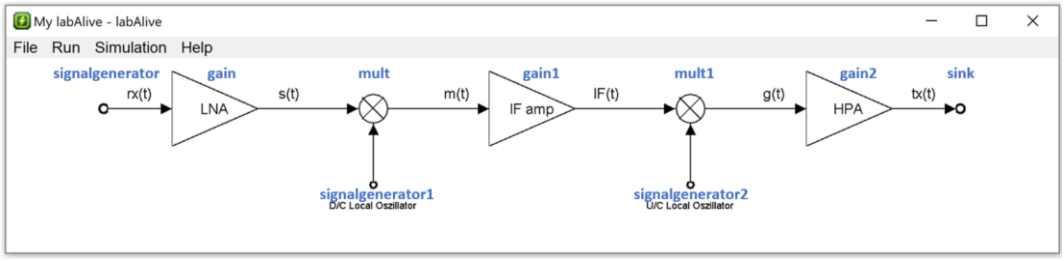


Abbildung 16: Simulationsaufruf mit eingefügter Systembezeichnung

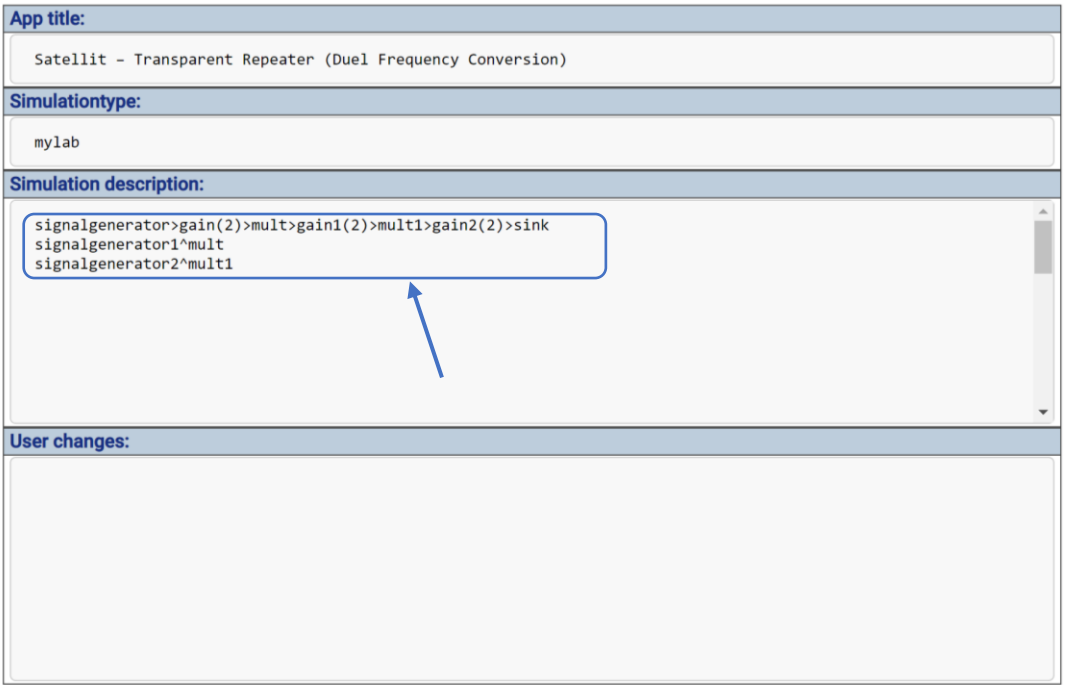


Abbildung 17: exemplarische Layoutdefinition mit neuer Methode in my Apps
Verwendete Simulation: "Satellit – Transparent Repeater (Duel Frequency Conversion)"

3.3.5 Syntax der Varianten

Welcher Syntax diese beide Verfahren unterliegen wird im Folgenden beschrieben. Die „Explicit Method“ sieht es vor die Layoutdefinition nach vorangegangener Initialisierung wie folgt zu definieren:

layout „*Layoutdefinition*“

Für das zuvor gezeigt Beispiel würde die Layoutdefinition lauten:

layout „1 > 2 > 3 > 4 > 5 > 6, 7 ^ 3,8 ^ 5“

Im Vergleich dazu ersetzen, im zweiten Verfahren, die Richtungsanweisungen das Verbindungssymbol „-“. Somit ist das Layout in der Initialisierung integriert und erspart der Nutzerin/dem Nutzer Zeilen in der Simulationsbeschreibung sowie das Umdenken, welche Nummerierung welchem System zuzuordnen ist.

3.3.6 Ausblick und mögliche Erweiterungen

Während der Generierung dieser Varianten für die Anwenderin/den Anwender gab es diverse Denkansätze die Erzeugung von Layouts weiter zu optimieren. Ein Ansatz war dabei eine Automatisierung zu entwerfen, welche alle erstellten Layouts aller Profile von my Apps abgreift und diese auf der Datenbank von labAlive abspeichert. Da LabAlive hauptsächlich mit der Layout-Methode COMPRESSED arbeitet müssten hierfür diese abgewandelten FULL-Layouts als erstes in das Standard FULL-Layout überführt werden und im Anschluss in das COMPRESSED. Hierfür müsste zunächst ein Algorithmus herausfinden, welche der beiden Verfahren der Layouterzeugung bei myApps verwendet wurde.

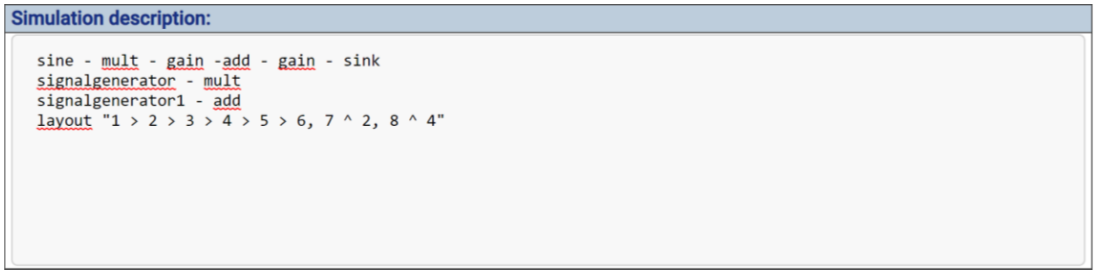


Abbildung 18: Beispiel einer Layoutdefinition mit der "Explicit Method"

Bei der ersten Variante wäre die nötige Nummerierung schon vorhanden. Da bei der COMPRESSED-Methode alle Systeme weggelassen werden, welche zwischen nur zwei Systemen sind, ist es möglich einen Algorithmus zu schreiben, der die Layoutdefinition nach Kommas durchsucht. Beim Detektieren eines Kommas wird jede Nummerierung in einem Array ablegt, welche nicht direkt am Anfang oder am Ende dieses Abschnitts der Layoutdefinition bis zum detektierten Komma zu finden ist (Abbildung 19).

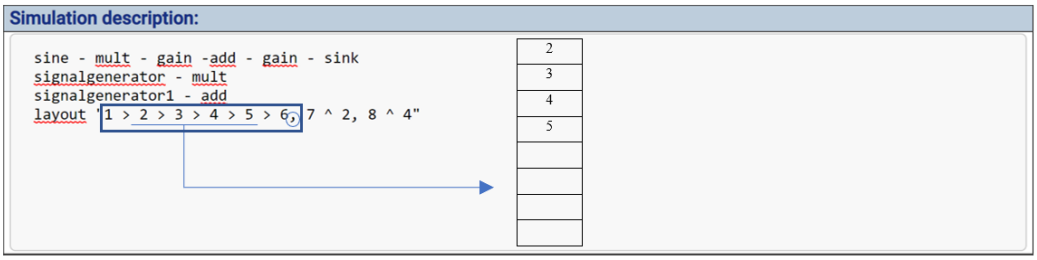


Abbildung 19: Befülltes Array nach detektieren des ersten Kommas

Danach wird angefangen, vom detektierten Komma ausgehend, das Nächste zu suchen und wieder jene Nummerierungen in das Array abzulegen, welche nicht am Anfang oder am Ende dieses Abschnitts der Layoutdefinition auftauchen. Sollte dabei eine Nummerierung am Anfang oder am Ende auftauchen, welche bereits im Array hinterlegt ist wird diese aus dem Array gelöscht.

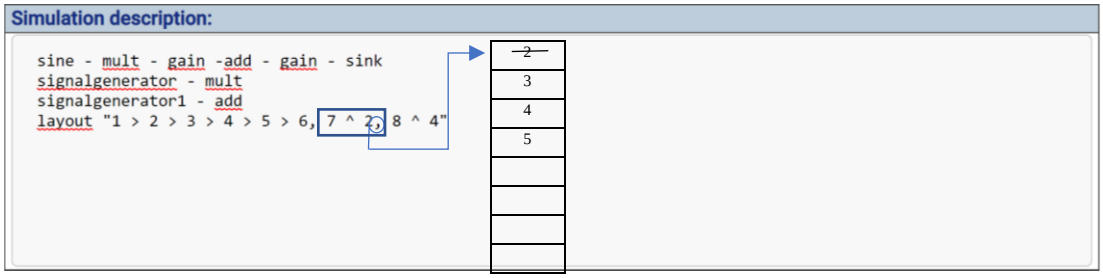


Abbildung 20: nächster detektierter Block mit dazugehöriger Änderung im Array

Dies wird so lange wiederholt, bis das Ende der Layoutdefinition erreicht ist. Im Anschluss werden alle Nummerierungen aus der Layoutdefinition, welche im Array enthalten sind, gelöscht sowie deren Richtungsanweisungen, die vor dieser Nummerierung auftauchen. Des Weiteren ist es im Anschluss notwendig die Nummerierungen dieser veränderten Layoutdefinition so anzupassen, dass diese erneut einer fortlaufenden Nummerierung unterliegen.

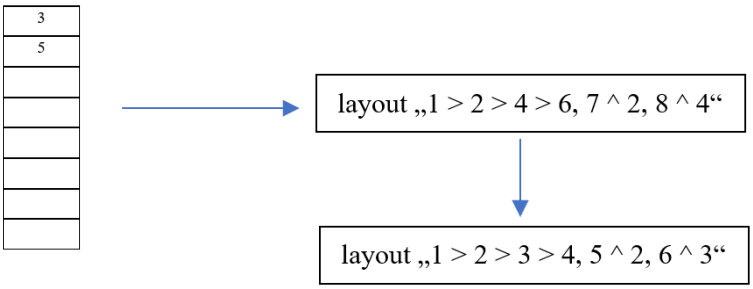


Abbildung 21:Ergebnis eines Algorithmus-Durchlaufs zur Erstellung eines COMPRESSED-Layouts

Diese Anpassung im System hätte jedoch Auswirkungen auf die Richtungsanweisungen, da bei der COMPRESSED- und der COLLAPSED-Methode davon ausgegangen wird, dass ausschließlich ein Doppelschritte nach rechts vor dem weggelassenen System vorhanden sein kann, also das Symbol „>“. Aus diesem Grund ist zu überdenken, ob die einfachen Schritte „w“ „a“ „s“ „d“ keinen Nutzen in Zukunft haben. Es gebe zwei Möglichkeiten diese zu ersetzen zum einen die

Richtungsanweisungen „^“ „<“ „v“ „>“ zu einfachen Schritten zu überführen oder nur noch Doppelschritte möglich zu machen. Beides hätte Auswirkungen auf diverse Klassen in der Workspace von LabAlive. Es müssten alle Klassen angepasst werden, welche mit den dann alten Richtungsanweisungen arbeiten.

Nach einer ersten Durchsicht wäre dieser Aufwand allerdings überschaubar und nicht mit großen Problemen verbunden.

4 Parametrisierte Aufrufe Analog-Demodulation

Das Online-Experiment Analog-Demodulation bietet es den Nutzerinnen und Nutzern die Möglichkeit sich mit den verschiedenen analogen Modulationsarten zu beüben. Dabei werden Signale, welche auf verschiedenen Frequenzbändern verschiedene Audiosignale beinhalten, auf die Simulation gegeben. Die Nutzerin/der Nutzer hat hierbei nun die Aufgabe herauszufinden auf welcher Frequenz, welche Modulationsart verwendet wurde. Hierfür hat er die Möglichkeit beim jeweiligen Demodulator sich das Audiosignal ausgeben zu lassen. Mittels Träger kann er nun das Signal auf der Frequenz verschieben und die Phase anzupassen. Wie dies genau funktioniert, wird in dem Experiment genau beschrieben.

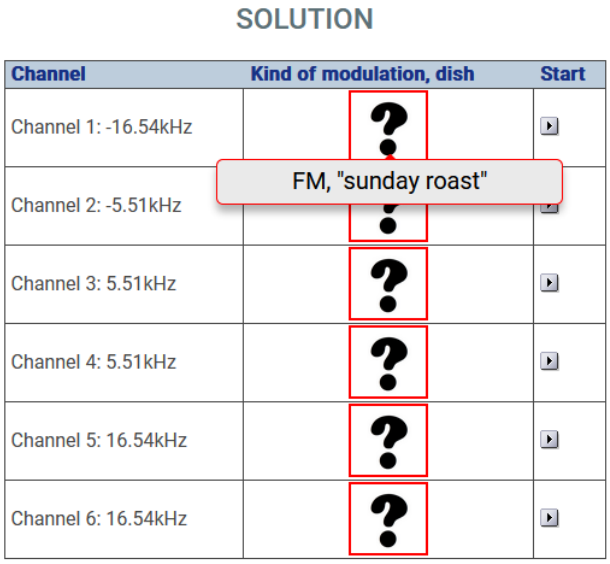


Abbildung 22:Lösungstabelle zur Analog-Demodulation

4.1 Erzeugung der parametrisierten Aufrufe

Die Seite wurde nun dahingehend ergänzt, sodass es nun die Möglichkeit gibt zu überprüfen, ob die Demodulation richtig ausgeführt wurde. Hierfür wurden die gesuchten Lösungen mit parametrisierten Aufrufen ergänzt, somit ist es jetzt möglich mittels eines Playbuttons die Simulation so zu starten, dass direkt nach Start das jeweilige gesuchte Audiosignal zu hören ist. Außerdem werden dazu die jeweiligen Graphen, welche das Audiosignal graphisch darstellen auch vorinitialisiert. Grafisch ist dieser Simulationsstartet mit einem Playbutton realisiert worden.

4.2 Erstellung der Seite mittels HTML

Der HTML-Code, welcher im Seitenaufruf ergänzt wurde, ist im Anhang zu sehen.

4.3 Aufgetretene Probleme

Während der Erstellung der parametrisierten Aufrufe sind mehrere Probleme/Fehler aufgetreten, welche behoben wurden. Zum einen war es anfangs nicht möglich die Trägerfrequenz sowie die dazugehörige Phase vor Simulationsstart einzustellen. Außerdem konnte bei der EM-Demodulation das untere Seitenband (LSB) zwar vorinitialisiert werden allerdings musste man es nach Simulationsstart erneut auswählen, bevor es den gewünschten Effekt hatte. Ein weiterer Bug, welcher zwar nicht direkt mit dem Thema parametrisierter Aufrufe zusammenhängt, allerdings während dessen Erstellung aufgefallen ist und behoben wurde, war dass die Speicherung der Simulation nur mittels Tastenkombination funktionierte und nicht zusätzlich mit der manuellen Auswahl der Speicherung.

5 Online-Hilfe zur Layoutgenerierung in my Apps

Das Thema Layout wurde in Kapitel 3.3 theoretisch erklärt und die verschiedenen Varianten, welche die Anwenderin/der Anwender zur Erstellung eines Layouts in my Apps hat, beschrieben. Damit sich die Nutzerin/der Nutzer von my Apps das nötige Wissen hierfür selbst aneignen kann wurde eine Online-Hilfe hierfür auf labAlive erstellt. Wie diese umgesetzt wurde, wird in diesem Kapitel beschrieben.

5.1 Strukturierung der Online-Hilfe

Die Online-Hilfe wurde explizit für die Nutzerinnen und Nutzer von my Apps und myLabAlive angefertigt. Dies ist dem begründet, da es nur hier der Anwenderin/dem Anwender möglich ist eigene Simulationen zu generieren. Demnach konnten in diesem Tutorial die beiden Layout-Methoden COMPRESSED und COLLAPSED vernachlässigt werden. Einzig und allein die FULL-Methode ist demnach interessant.

Als Einleitung für die Online-Hilfe wird der Nutzerin/dem Nutzer kurz und knapp erklärt was genau ein Layout ist und für was es genau benötigt wird.

Anschließend wird die erste von zwei möglichen Varianten die „Combined Method“ beschrieben. Zunächst wird der Aufbau der Methode beschrieben Nach dieser kurzen Beschreibung folgt eine Grafik, welche mit den jeweiligen Bezeichnungen der Systeme und den Positionierungsanweisungen ergänzt wurde. Als nächstes ist ein Eingabefeld mit der Simulationsbeschreibung für oben gezeigtes Beispiel eingefügt. In diesem Abschnitt sind außerdem die möglichen Positionierungsanweisungen und deren Bedeutung aufgelistet.

Die zweite mögliche Variante die „Explicit Method“ wird im nächsten Abschnitt erläutert. Bevor auf die genaue Layouterzeugung eingegangen wird, wird zunächst beschrieben, welche Änderungen diese Art von Layouterzeugung mit sich führen und wie diese durchgeführt werden. Da es sich hierbei um die Variante handelt, welche mit Nummerierung arbeitet und es wichtig ist, dass der Anwenderin/dem Anwender versteht welchen Regeln diese Nummerierung unterliegt, wurde ein GIF-Datei eingefügt, welche nach und nach die Systeme nummeriert sowie die

Positionierungsanweisungen zwischen den Systemen einfügt. Danach wird die Layoutdefinition, welche dieses Layout erzeugt, präsentiert.

Um das nun erlangte Wissen über die Erzeugung eines Layouts zu komplettieren, werden im Folgenden drei weitere Beispiele aufgeführt. Außerdem darin enthalten sind die verschiedenen Optionen, welche es gibt, einen Splitter zu integrieren.

Als letztes wird ist eine Übung zur Leistungskontrolle eingefügt, in welcher man eine gezeigte Schaltung und deren Systeme mit einer der zuvor beschriebenen Methoden erstellen soll. Hierzu ist das Eingabefeld schon mit den verwendeten Systemen befüllt allerdings nicht mit den richtigen Verbindungen und nicht sortiert. Auf den folgenden Seiten ist der Seitenaufruf abgebildet.

New experiment 1

Tutorial

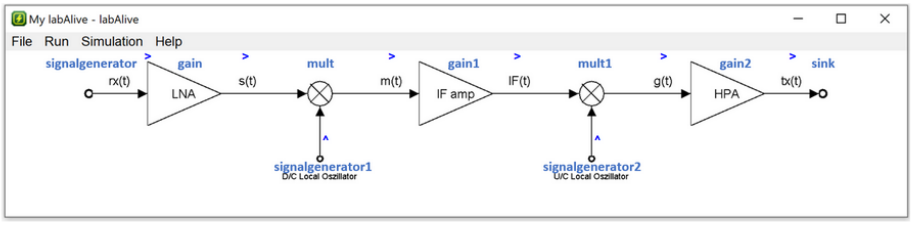
As labAlive App developer, I want to define a layout for my simulation, so that it starts showing a nice block diagram as primary GUI providing access to measure instruments and systems. In this tutorial you are going to learn two methods of how to create your own layout in my Apps. In general, a layout shows you how the components of a wiring are sufficient.

COMBINED METHOD

The idea of this method is to define the layout within the connection of the systems. The only thing you have to do is to connect the systems with the directional instructions you want to use. With these directional instructions you can position the systems in the wiring and they are defined as follows:

Symbol	Meaning
<	one step left
^	one step up
>	one step right
v	one step down

The following graphic shows you an example of how a layout of a wiring is built up with this method.



The connection of the systems combined with the layout would be:

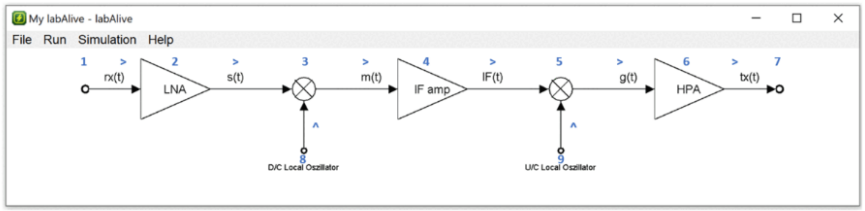
Enter your simulation description:

```
signalgnerator > gain > mult > gain1 > mult1 > gain2 > sink
signalgenerator1 ^ mult
signalgenerator2 ^ mult1
```

Launch my simulation

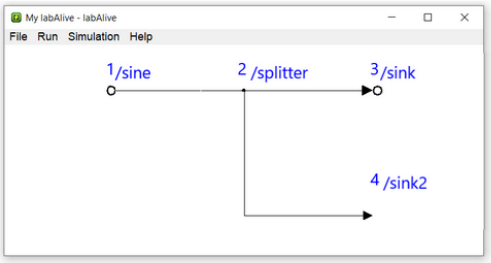
EXPLICIT METHOD

In contrast to the combined method in this method you have to connect the systems at first and after that the layout can be defined. To connect the systems, you have to replace the directional instructions with one single "-" between two systems. Also, you have to address the systems in the layout definition with the numbers of the systems. In the following graphic you can see an example of how a layout of a wiring is built up with this method.



The layout definition of this example would be: layout "1 > 2 > 3 > 4 > 5 > 6 > 7, 8 ^ 3, 9 ^ 5"

In the following you can see three examples. In this examples you can see three option of how to realize a splitter. After these examples you have the possibility to solve an exercise. At first you can see the wiring of the examples.



FIRST EXAMPLE

Enter your simulation description:

```
sine > split > sink
split v> sink2
```

Launch my simulation

In this example you can see the option to realize a splitter with the combined method

SECOND EXAMPLE

Enter your simulation description:

```
sine - split - sink
split - sink2
layout "1 > 2 > 3, 2 v> 4"
```

Launch my simulation

In this example you can see one of two options to realize a splitter with the explicit method.

THIRD EXAMPLE

Enter your simulation description:

```
sine - sink
sine - sink2
layout "1 > 2 > 3, 2 v> 4"
```

Launch my simulation

In this example you can see the other option to realize a splitter with the explicit method.

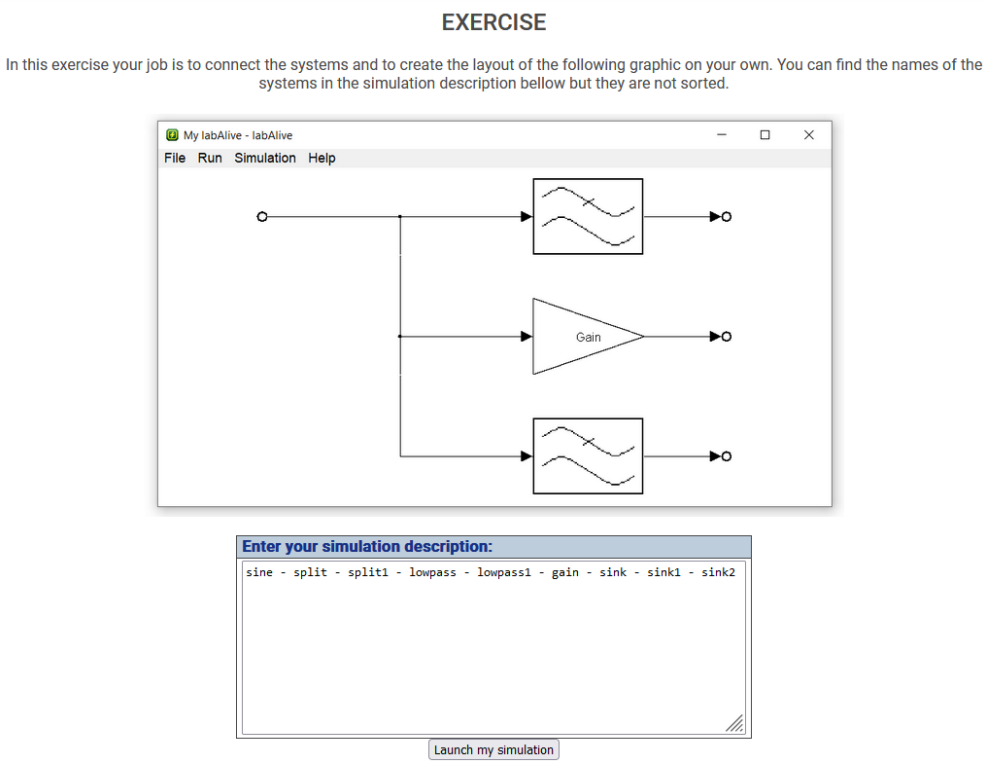


Abbildung 23: Layout Tutorial auf labAlive

5.2 Erstellung der Online-Hilfe mit HTML

Für die Darstellung auf der Homepage wurde die textbasierte Auszeichnungssprache HTML verwendet. Der Quellcode, welcher diesen Seitenaufruft erzeugt, ist im Anhang eingefügt.

6 Fazit

Ziel dieser Bachelorarbeit war es die verschiedenen Möglichkeiten, welche es für die Erstellung von parametrisierten Aufrufen gibt, aufzuarbeiten und für verschiedene Online-Experimente mit diesen zu ergänzen. Unteranderem wurde das Online-Experiment „Analog-Demodulation“ mit einer Lösungstabelle ergänzt, welche die vorinitialisierten Simulationsaufrufe beinhaltet. Diese Simulationsaufrufe wurden mit einem Playbutton versehen, mit dem man diesen Aufruf direkt starten kann. Dies steigert die Interaktivität des Experiments und für die Nutzerin/den Nutzer ist eine Möglichkeit geschaffen worden durch den voreingestellten Simulationsaufruf genau zu überprüfen, ob sie/er den Versuch richtig durchgeführt hat.

Das neu entwickelte Feature my Apps wurde in diesem Zusammenhang detailliert beschrieben, sodass es als eine Anleitung dafür fungieren kann.

Außerdem war es das Ziel die verschiedenen Varianten der Layoutdefinition, die in my Apps und MyLabAlive für die Anwenderinnen und Anwender zur Verfügung stehen, detailliert zu beschreiben und dafür eine Onlinehilfe zu erstellen. Mit der Onlinehilfe ist nun eine Möglichkeit geschaffen worden, dass man sich das Wissen dafür selbstständig aneignen kann. Mit der Hilfe von Mitstudenten wurde diese Onlinehilfe dahingehend erprobt, so dass davon auszugehen ist, dass jede Nutzerin/jeder Nutzer nun in der Lage ist ein Layout für ihre/seine Simulation zu erstellen.

Auf LabAlive sind die Ergänzung des Online-Experiments der „Analog-Demodulation“ und die Onlinehilfe für Nutzerinnen und Nutzer weltweit zugänglich.

7 Anhang

7.1 Quelltext für den Seitenaufruf der Online-Hilfe in HTML

```
<p style="text-align:center">As labAlive App developer, I want to define
a layout for my simulation, so that it starts showing a nice block dia-
gram as primary GUI providing access to measure instruments and systems.
In this tutorial you are going to learn two methods of how to create
your own layout in my Apps. In general, a layout shows you how the com-
ponents of a wiring are sufficient. </p>

<h1>Combined method</h1>

<p style="text-align:center">The idea of this method is to define the
layout within the connection of the systems. The only thing you have to
do is to connect the systems with the directional instructions you want
to use. With these directional instructions you can position the systems
in the wiring and they are defined as follows:</p>

<div class="table-wrapper">

 
    <table>
    <!--   align="center" border="1" frame="box">-->
    <tr>
      <th>Symbol</th>
      <th style="text-align:left"> Meaning</th>
    </tr>
    <tr>
            <td> < </td>
            <td style="text-align:left"> one step left</td>
      </tr>
      <tr>
            <td>^ </td>
            <td style="text-align:left"> one step up</td>
      </tr>
      <tr>
            <td>> </td>
            <td style="text-align:left"> one step right</td>
      </tr>
      <tr>
            <td>v </td>
            <td style="text-align:left"> one step down</td>
      </tr>
    </table>

<p style="text-align:center">The following graphic shows you an example
of how a layout of a wiring is built up with this method.</p>
```

```
<div align="center" class="figure-full"><img src="Layout_text.png"
style="width: 50vw; min-width: 140px;"></div>

<p></p>

<p style="text-align:center">The connection of the systems combined with
the layout would be: </p>

<p></p>

<form action="/labalive/webstart/mylab.jnlp" method="get">

      <table>

            <tr>

                  <th>Enter your simulation description:</th>

            </tr>

            <tr>

                  <td>

                  <textarea name="w" cols="70" rows="10">signalgnera-
tor>gain>mult>gain1>mult1>gain2>sink

signalgenerator1^mult

signalgenerator2^mult1</textarea>

                  </td>

            </tr>

      </table>

      <input type="submit" value="Launch my simulation"/>

</form>

</div>

<h1>Explicit method</h1>

<p style="text-align:center">In contrast to the combined method in this
method you have to connect the systems at first and after that the lay-
out can be defined. To connect the systems, you have to replace the di-
rectional instructions with one single "-" between two systems. Also,
you have to address the systems in the layout definition with the num-
bers of the systems. In the following graphic you can see an example of
how a layout of a wiring is built up with this method. </p>


<div align="center" class="figure-full"><img src="layout_numbering.gif"
style="width: 50vw; min-width: 140px;"></div>

<div class="caption"><p style="text-align: center">The layout-definition
of this example would be: 

layout "1 > 2 > 3 > 4 > 5 > 6 > 7, 8 ^ 3, 9 ^ 5"</p>

</div>

<p></p>

<p style="text-align:center">In the following you can see three exam-
ples. In these examples you can see three options of how to realize a
splitter.

After these examples you have the possibility to solve an exercise. At
first you can see the wiring of the examples. </p>

<h1></h1>
```

```
<div align="center" class="figure-full"><img src="example.png"
style="width: 25vw; min-width: 70px;"></div>
<p></p>
<p></p>
<h2>First Example</h2>
<form action="/labalive/webstart/mylab.jnlp" method="get">
	<table>
		<tr>
			<th>Enter your simulation description:</th>
		</tr>
		<tr>
			<td>
			<textarea name="w" cols="70" rows="10">sine > split
> sink
split v> sink2</textarea>
			</td>
		</tr>
	</table>
	<input type="submit" value="Launch my simulation"/>
</form>
<div class="caption"><p style="text-align: center">In this example you
can see the option to realize a splitter with the combined method</p>
</div>
<p></p>
<p></p>
<h2>Second Example</h2>
<form action="/labalive/webstart/mylab.jnlp" method="get">
	<table>
		<tr>
			<th>Enter your simulation description:</th>
		</tr>
		<tr>
			<td>
			<textarea name="w" cols="70" rows="10">sine - split
- sink
split - sink2
layout "1 > 2 > 3, 2 v> 4"</textarea>
			</td>
		</tr>
	</table>
	<input type="submit" value="Launch my simulation"/>
```

```
</form>
<div class="caption"><p style="text-align: center">In this example you can see one of two options to realize a splitter with the explicit method.</p>
</div>
<p></p>
<p></p>
<h2>Third Example</h2>
<form action="/labalive/webstart/mylab.jnlp" method="get">
	<table>
		<tr>
			<th>Enter your simulation description:</th>
		</tr>
		<tr>
			<td>
			<textarea name="w" cols="70" rows="10">sine - sink
sine - sink2
layout "1 > 2 > 3, 2 v> 4"</textarea>
			</td>
		</tr>
	</table>
	<input type="submit" value="Launch my simulation"/>
</form>
<div class="caption"><p style="text-align: center">In this example you can see the other option to realize a splitter with the explicit method.</p>
</div>
<h1>Exercise</h1>
<p style="text-align: center">In this exercise your job is to connect the systems and to create the layout of the following graphic on your own. You can find the names of the systems in the simulation description bellow but they are not sorted. </p>
<h1></h1>
<div align="center" class="figure-full"><img src="exercise_solution.png" style="width: 37.5vw; min-width: 105px;"></div>
<h1></h1>
<form action="/labalive/webstart/mylab.jnlp" method="get">
	<table>
		<tr>
			<th>Enter your simulation description:</th>
		</tr>
		<tr>
			<td>
```

```
                    <textarea name="w" cols="70" rows="10">sine - split
- split1 - lowpass - lowpass1 - gain - sink - sink1 - sink2
                    </textarea>
                    </td>
                </tr>
        </table>
        <input type="submit" value="Launch my simulation"/>
</form>
```

7.2 Ergänzter Quelltext für den Seitenaufruf der Analog-Demodulation in HTML

```
<h2>Solution</h2>
<div class="table-wrapper">
<table>
      <tr>
            <th>Channel</th>
            <th>Kind of modulation, dish</th>
            <th>Start</th>
      </tr>
      <tr>
            <td>Channel 1: -16.54kHz</td>
            <td style="text-align:center">
            <div class="formel">
            <div class="figure" align="center"><img src="question-
            mark.png" alt="" height="55"></div>
            <span class="formeltext">FM, "sunday roast"</span></div>
            </td>
            <td><a href="../../webstart/analogdemod.jnlp?p=fmcom-
            plexdemod.spectrum%20show%20On%0Afmcomplexdemod.playau-
            dio%20maxamplitude%20100m%20show%20On%0Areal2analyticsig-
            nal.signallogging%20show%20Off%0Acomplexsinegenera-
            tor%20frequency%2016.538k%0A"><img src="next.gif"
            alt="Start"></a></td>
    </tr>
    <tr>
            <td>Channel 2: -5.51kHz</td>
            <td style="text-align:center">
            <div class="formel">
            <div class="figure" align="center"><img src="question-
            mark.png" alt="" height="55"></div>
            <span class="formeltext">AM, "sticky toffee pudding"</span>
            </div>
            </td>
            <td><a href="../../webstart/analogdemod.jnlp?p=real2analyt-
            icsignal.signallogging%20show%20Off%0Atoreal1.spec-
            trum%20show%20On%0Atoreal1.playaudio%20maxampli-
            tude%200.2%20show%20On%0Acomplexsinegenerator%20fre-
            quency%205.5125k%0A"><img src="next.gif"
            alt="Start"></a></td>
    </tr>
    <tr>
            <td>Channel 3: 5.51kHz</td>
```

```
                <td style="text-align:center">
                <div class="formel">
                <div class="figure" align="center"><img src="question-
                mark.png" alt="" height="55"></div>
                <span class="formeltext">QAM, "chicken tikka masala"</span>
                </div></td>
                <td><a href="../../webstart/analogdemod.jnlp?p=toreal.spec-
                trum%20show%20On%0Atoreal.playaudio%20show%20On%0Areal2ana-
                lyticsignal.signallogging%20show%20Off%0Acomplexsinegenera-
                tor%20frequency%20-5.5125k%20phase%2057.0%0A%0A"><img
                src="next.gif" alt="Start"></a></td>
</tr>
<tr>
                <td>Channel 4: 5.51kHz</td>
                <td style="text-align:center">
                <div class="formel">
                <div class="figure" align="center"><img src="question-
                mark.png" alt="" height="55"></div>
                <span class="formeltext">QAM, "bangers and mash"</span>
                </div></td>
                <td><a href="../../webstart/analogdemod.jnlp?p=toreal.spec-
                trum%20show%20On%0Atoreal.playaudio%20show%20On%0Areal2ana-
                lyticsignal.spectrum%20show%20Off%0Areal2analyticsig-
                nal.signallogging%20show%20Off%0Acomplexsinegenera-
                tor%20frequency%20-5.5125k%20phase%20147.0%0A%0A"><img
                src="next.gif" alt="Start"></a></td>
    </tr>
    <tr>
                <td>Channel 5: 16.54kHz</td>
                <td style="text-align:center">
                <div class="formel">
                <div class="figure" align="center"><img src="question-^
                mark.png" alt="" height="55"></div>
                <span class="formeltext">SSB, LSB="Yorkshire pud-
ding"</span>
                </div></td>
                <td><a href="../../webstart/analogdemod.jnlp?p=real2analyt-
                icsignal.signallogging%20show%20Off%0Atoreal1.playau-
                dio%20maxamplitude%200.5%20show%20On%0Areal2analyticsig-
                nal%20sideband%20LowerSideBand%0Acomplexsinegenera-
                tor%20frequency%20-16.537k%0A"><img src="next.gif"
                alt="Start"></a></td>
    <tr>
                <td>Channel 6: 16.54kHz</td>
                <td style="text-align:center">
                <div class="formel">
```

```
                <div class="figure" align="center"><img src="question-
                mark.png" alt="" height="55"></div>
                <span class="formeltext">SSB, USB= "afternoon tea"</span>
                </div></td>
                <td><a href="../../webstart/analogdemod.jnlp?p=real2analyt-
                icsignal.signallogging%20show%20Off%0Atoreal1.playau-
                dio%20maxamplitude%200.5%20show%20On%0Areal2analyticsig-
                nal%20sideband%20LowerSideBand%0Acomplexsinegenera-
                tor%20frequency%20-16.537k%0A"><img src="next.gif"
                alt="Start"></a></td>
        </tr>
        </table>
</div>
```

8 Literaturverzeichnis

[1] P. D.-I. E. Riederer, „labAlive,“ 21 6 2022. [Online]. Available: https://www.etti.unibw.de/labalive/.

[2] „Wikipedia,“ [Online]. Available: https://de.wikipedia.org/wiki/Parameter. [Zugriff am 21 Juni 2022].

[3] „dev-insider,“ [Online]. Available: https://www.dev-insider.de/was-ist-eine-ide-a-600703/. [Zugriff am 21 Juni 2022].

[4] „Java,“ [Online]. Available: https://www.java.com/de/download/help/java_webstart_de.html. [Zugriff am 21 Juni 2022].

[5] „Minitool,“ [Online]. Available: https://de.minitool.com/datentraegerverwaltung/jnlp-datei.html. [Zugriff am 21 Juni 2022].

[6] „labAlive,“ [Online]. Available: https://www.etti.unibw.de/labalive/experiment/analog-demod/. [Zugriff am 22 Juni 2022].

[7] „labAlive,“ [Online]. Available: https://www.etti.unibw.de/labalive/my/. [Zugriff am 22 Juni 2022].

[8] P. D.-I. E. Riederer, „LabAlive,“ [Online]. Available: https://www.etti.unibw.de/labalive/experiment/am/. [Zugriff am 21 Juni 2022].

[9] „Meyerweb,“ [Online]. Available: https://meyerweb.com/eric/tools/dencoder/. [Zugriff am 21 Juni 2022].

[10] S. Niendorf, „LabAlive,“ [Online]. Available: https://www.etti.unibw.de/doc/thesis/Layout_Generator_und_GUI_Erweiterungen_f%C3%BCr_labAlive_Blockdiagramme_und_Fenster.pdf. [Zugriff am 21 Juni 2022].

[11] R. Wiesner, „LabAlive,“ [Online]. Available: https://www.etti.unibw.de/doc/thesis/Blockdiagramm_Layout_von_labAlive_Simulationen-Konzepte_und_Anwendungen.pdf. [Zugriff am 21 Juni 2022].