1 Einleitung

Der *Logic Simulator*, im folgenden abgekürzt durch *LogicSim 3*, ist der Nachfolger von Andreas Tetzls *LogicSim 2*. *LogicSim 2* ist auf seiner Webseite `http://tetzl.de` veröffentlicht.

Neue Versionen von *LogicSim 3* sind auf der Webseite `http://sis.schule` veröffentlicht. Der Quelltext kann unter GitHub (Projektname: LogicSim3) heruntergeladen werden. Das Programm ist unter der GPL veröffentlicht und darf daher frei verändert werden, sofern der geänderte Quelltext wieder veröffentlicht wird. Eine kommerzielle closed-source Verwendung ist damit ausgeschlossen.

Die Entwicklung von *LogicSim 2* wurde seit 11 Jahren nicht mehr fortgeführt, mittlerweile ist die Applet-Technik abgekündigt und ist evtl. bald nicht mehr in den verfügbaren *Java Runtimes* enthalten. Der Autor der fortgeführten Software fehlten einige Funktionen, speziell eine Zoom-Funktion sowie Kopieren und Einfügen von Elementen. Auch konnten keine Beschriftungen zu Ein- und Ausgängen festgelegt werden. Dies war insbesondere bei Modulen problematisch. Auch das Rotieren von Elementen ist noch nicht möglich. Einige dieser Funktionen wurden bereits implementiert. Das Dateiformat wurde auf XML umgestellt, so können die Dateien auch händisch bearbeitet werden.

Wichtig für die neue Version war ein Einsatz in der Schule. Da *LogicSim 2* einfach zu bedienen ist und man direkt mit der Modellierung von Schaltkreisen beginnen kann, sollte der Einsatz in der Schule weiter verbessert werden. Die einzelnen »Gatter« sind nun in ein Verzeichnis ausgelagert worden und sind damit nicht mehr fest im *Java Archiv* (.jar-Datei) verankert. Dadurch können nun aus didaktischen Gründen einzelne Gatter entfernt werden, um z.B. nur die Grundgatter, Schalter und LEDs zur Verfügung zu stellen.

An der Erstellung der Software darf mitgearbeitet werden. Dazu ist das GitHub-Projekt vorhanden. Bitte nehmen Sie daher Kontakt über die GitHub-Projektseite auf.

Bitte verwenden Sie auch diesen Weg, falls Sie Fehler der Software melden möchten.

1.1 History

2020-04-03	Erstellung des Dokuments

2 Bedienung

2.1 Start

LogicSim 3 wird als JAR-Datei ausgeliefert. Hierbei handelt es sich um ein Java-Archiv, dass nur dann lauffähig ist, wenn auf dem Computer JAVA installiert ist. Der Start des Programms geschieht im Normalfall über einen Doppelklick. Sollte dies nicht funktionieren, so kann die Einrichtung des Doppelklicks i
Die Softwarebedienung erschließt sich in weiten Teilen von selbst. Besonderheiten:

- Bauteile werden in die Arbeitsfläche eingefügt, indem zunächst das Bauteil links in der Liste angeklickt wird. Durch einen Klick auf eine freie Stelle in der Arbeitsfläche wird das Teil mit dem oberen linken Punkt am Mauszeiger eingefügt.
- Neue Kabel werden eingefügt, indem stets *ein Ausgang* eines Gatters angeklickt wird. Das freie Kabelende ist immer am Mauszeiger. Jeder weitere Klick auf eine freie Stelle fügt in dem Kabel ein »Knick« ein. Ein Kabel wird fertiggestellt, in dem *ein Eingang* eines Bauteils angeklickt wird.
  Während der Bearbeitung des Kabels kann die Taste *Escape* gedrückt werden, um den zuletzt eingefügten Punkt zu löschen, falls versehentlich ein Punkt platziert wurde.
- Das Anklicken eines Bauteils oder eines Kabels führt zur Auswahl (*Selektion*) des Teils. Es kann dann per Pfeiltasten oder durch Drag&Drop mit der Maus verschoben werden. Bei Kabeln werden nur die inneren Punkte bewegt, die äußeren Punkte bleiben an angeschlossenen Bauteile. Werden Bauteilen bewegt, werden entsprechend nur die Kabelenden der angeschlossenen Kabel mitbewegt.
- Selektierte Bauteile oder Kabel können durch Druck auf die Backspace-Taste (auch Entf- oder Lösch-Taste) entfernt werden. Werden Bauteile gelöscht, so werden automatisch alle angeschlossenen Kabel mitgelöscht.
- Kabel müssen immer an zwei Enden angeschlossen sein, es gibt keine »Luftenden«.
- Das Mausrad wird nun unterstützt, Zoomen ist damit möglich, Zusätzlich dazu gibt es passende Schaltflächen zum Vergrößern, Verkleinern und auf den verwendeten Bereich Zoomen.

2.2 Einsatz in der Schule

Neu ist das Verzeichnis »gates« im Hauptverzeichnis des Programms. In ihm sind kompilierte Klassen, die ihrerseits Gatter und Bauteile für *LogicSim 3* bereitstellen. *LogicSim 3* wird dadurch erweiter- und beschränkbar. Durch Entfernen von Klassendateien aus diesem Verzeichnis sind die entsprechenden Bauteile im Programm nicht mehr verfügbar. So kann in einer Unterrichtsreihe in der Schule zunächst mit Grundkomponenten wie AND, OR, NOT sowie Schaltern und LEDs begonnen werden. Später können dann weitere Komponenten hinzugenommen werden, so dass Schülerinnen und Schülern bei den gewünschten Funktionen bleiben.

3 Module

Module sind dort interessant, wo eine Innensicht auf Bauteile gewünscht ist. Es lassen sich z.B. komplexere Bauteile wie Volladdierer oder Schieberegister aus einzelnen Gattern zusammensetzen. Dafür stehen pro Modul immer 16 Ein- und 16 Ausgänge zur Verfügung. Solche Schaltungen lassen sich dann abspeichern und als neue Bauteile als Block verwenden.

Neue Module werden über den Menüpunkt »Neues Modul erstellen« angelegt. Der Dateiname des Moduls entscheidet dann später über den Eintrag in der Liste der Bauteile links im Programm. Es dürfen zum Test an die Eingänge der `INPUTS` und an die Ausgänge der `OUTPUTS` weitere Bauteile angeschlossen werden und auch mit abgespeichert werden. Die Bauteile werden automatisch während der Verwendung als Block automatisch entfernt. Ein- und Ausgänge von Modulen können Beschriftungen tragen. Diese müssen derzeit noch per Hand eingetragen werden. Wie das geschieht, ist in Abschnitt **??** beschrieben.

4 Aufbau der digitalen Schaltungsdateien

4.1 Beschriftungen für Ein- und Ausgänge

5 Erweiterung

5.1 Verwendung mit Eclipse

Das Projekt wird derzeit mit Eclipse entwickelt, in der letzten Version. Als Java SDK kommt OpenJDK zum Einsatz.
Da LogicSim zur Entwicklungszeit und zum Start aus Eclipse heraus unbedingt die Verzeichnisse `circuits` und `modules` benötigt, ist darauf zu achten, dass diese Ordner zugreifbar sind. Da diese im Eclipse-Projekt-Hauptordner liegen, funktioniert das so.
In der JAR-Datei ist dies anders. Hier gibt es zusätzlich das Verzeichnis `gates` im Hauptordner, hier müssen die Klassen aus dem `gates`-Package enthalten sein. Der Vorteil liegt hier in einem einfachen Löschen und Hinzunehmen weiterer Gatter-Typen und Komponenten.
Die Zusammenstellung der Distribution bzw. des LogicSim-ZIP-Archivs erfolgt via ant-build-Skript. Abbildung **??** zeigt die Dateiansicht aus Eclipse heraus.

5.2 Übersetzung der Texte in andere Sprachen

Als Muster stehen die Dateien `de.txt` und `en.txt` zur Verfügung. Diese Dateien enthalten die neuen Sprachtexte, die von *LogicSim 3* verwendet werden. Vorhandene, andere Sprachdateien sind zwar übersetzt, nur sind die Sprachtextbezeichner noch nicht auf *LogicSim 3* umgestellt.

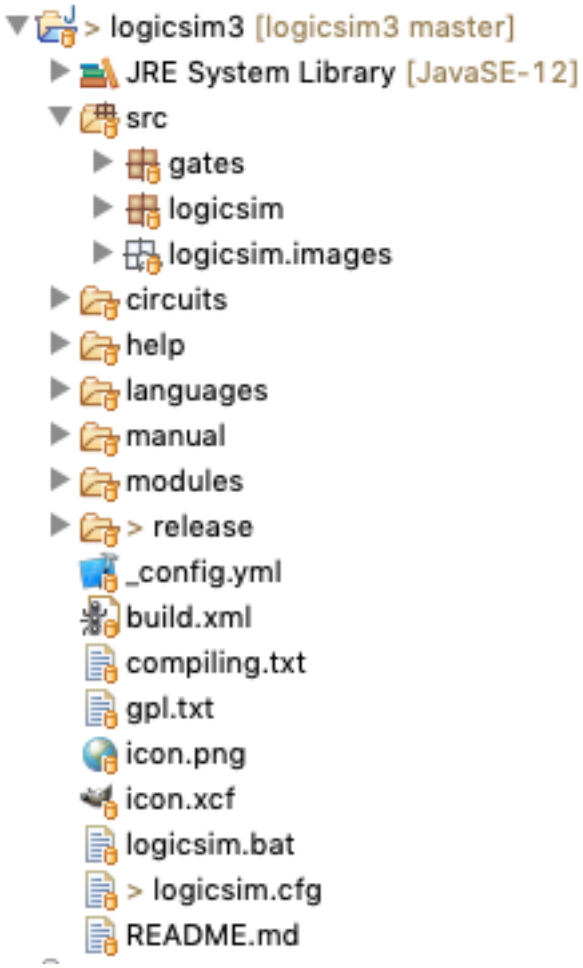

Abbildung 1: Struktur des Projekts in der Eclipse-Ansicht

5.3 Weiterentwicklung, Implementierung neuer Bauteile

Eine Erweiterung ist ohne Weiteres möglich. Dazu kann der Source-Tree von Github heruntergeladen werden[1]. Wird Eclipse verwendet, so kann direkt die Struktur aus Github übernommen werden. Das *src*-Verzeichnis enthält die Quelltexte zum Programm und den Bauteilen. Es sind keine externen Bibliotheken erforderlich.

Die Klasse `App.java` enthält die `main`-Methode für den Start der Software. Die Datei `build.xml` wird für das Generieren von Releases mittels *ant* verwendet.
Da der Aufbau der implementierten Logikgatter sehr einfach ist, können auch Schüler in Facharbeiten weitere, komplexere Gatter implementieren. Die Klasse `logicsim.Gate` stellt die API, gemeinsam mit der Oberklasse `logicsim.CircuitPart` bereit. Alle vorhandenen Gatter verwenden lediglich die definierten Methoden. Anhand der vorhandenen Gatterklassen kann das Vorgehen für weiterführende Bauteile abgeschaut werden. Auch Komponenten wie die 7-Segment-Anzeige sind Unterklassen von `logicsim.Gate`.

[1] s. im Internet: `https://github.com/codepiet/LogicSim3`

6 Nächste Schritte

Anhang

A Jar-Doppelklick unter Windows 10 einrichten

Damit der Doppelklick unter einem System funktioniert, müssen die JAR-Dateien einem Programm zugewiesen werden. Oft ist unter Windows 10 der Doppelklick auf Jar-Dateien nicht richtig eingestellt. Ein bloßes Einrichten von »Öffnen mit« führt zu keinem Ergebnis. Dies liegt daran, dass dem Programm »java.exe«, dass seinerseits die JAR-Datei lädt, ein weiterer Parameter (-jar) mitgegeben werden muss. Dies lässt sich gut über die Registry lösen.

1. Notieren, wo Java installiert ist, oft in `C:\Programme (x86)\Java\jreXXX\bin\java.exe`
2. »regedit« als Administrator ausführen.
3. Zum angegebenen Schlüssel in Abbildung ?? navigieren.
4. Den Schlüssel wie in Abbildung ?? angegeben ändern, dabei den richtigen Pfad von Schritt 1 übernehmen.

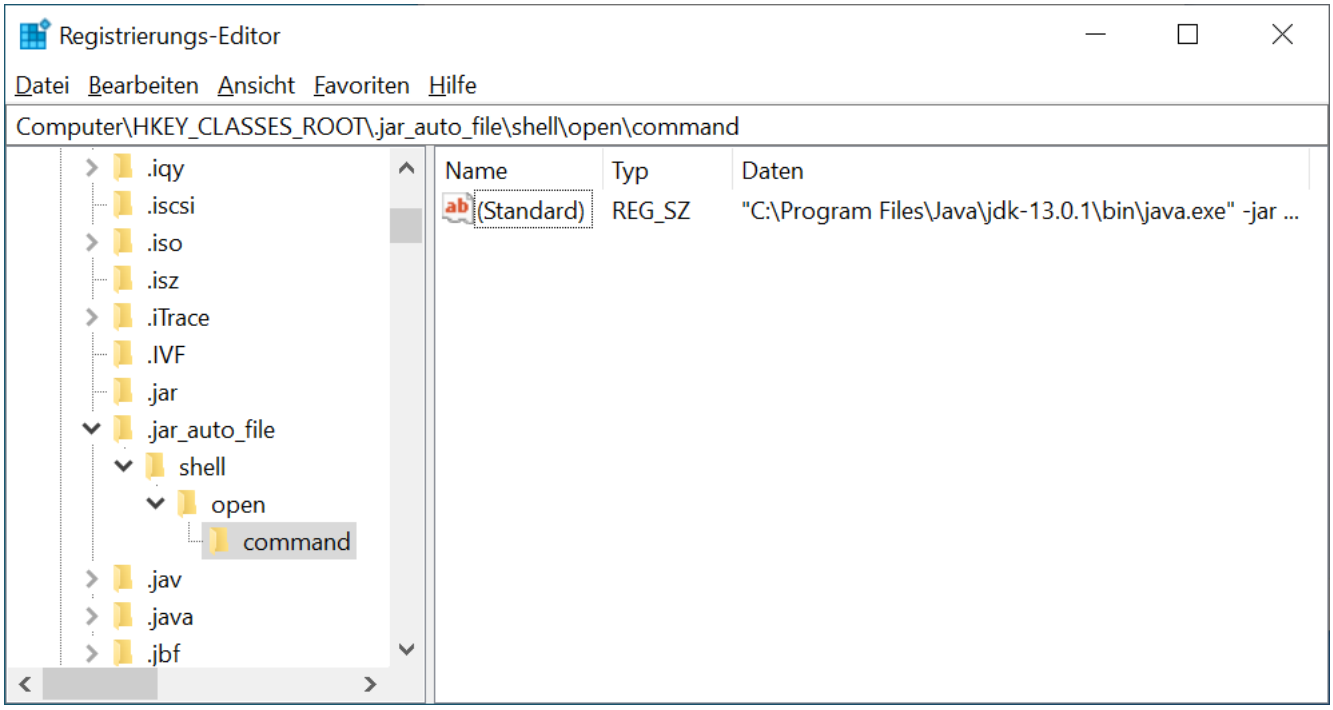

Abbildung 2: Regedit-Programm mit dargestelltem Pfad zum Schlüssel

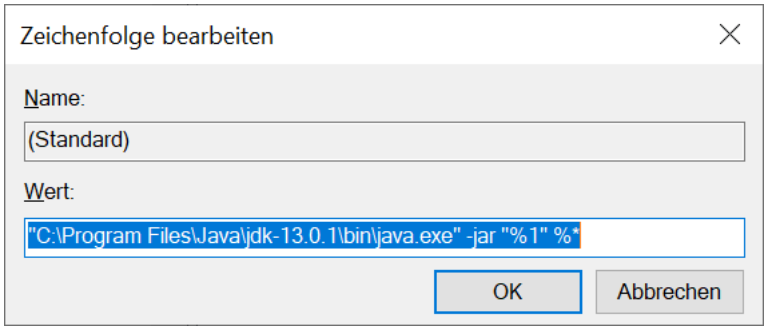

Abbildung 3: Bearbeiten des Schlüssels