PiiRi 4.0 – Anleitung

Inhaltsverzeichnis

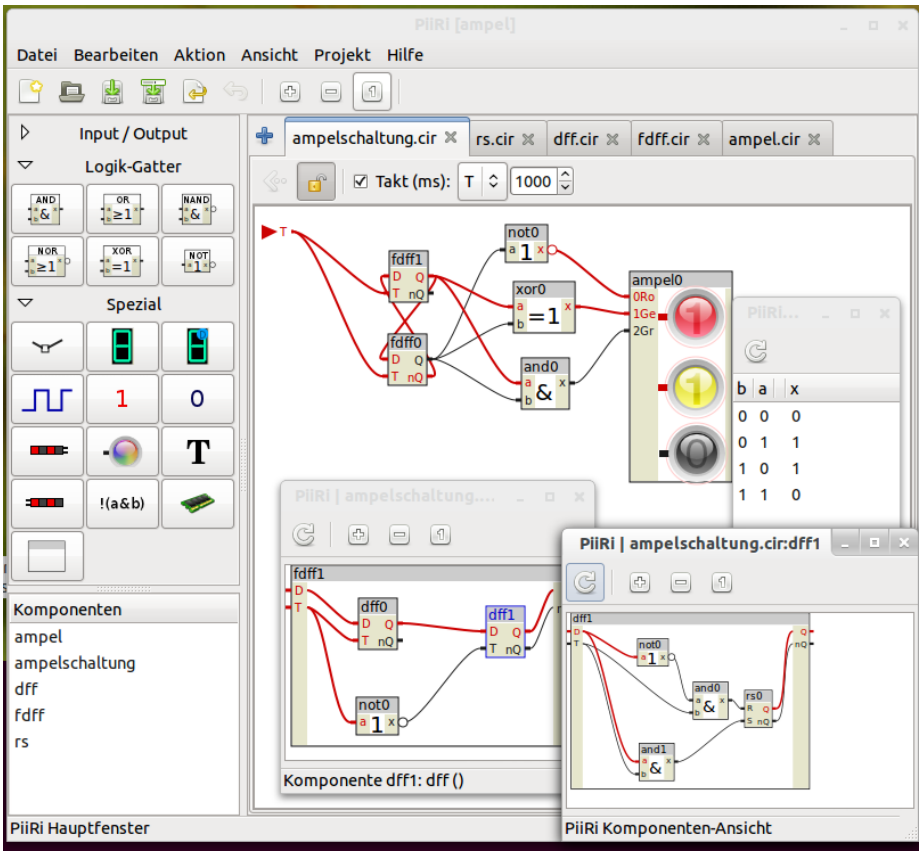
1 Einleitung

Mit **PiiRi** können logische Schaltungen gebaut und simuliert werden. Das Programm wird in erster Linie für den schulischen Gebrauch (Informatikunterricht) entwickelt.

Hinweis

PiiRi bietet *keine* physikalische Simulation und ist kein professionelles Tool für Wissenschaft oder Hardware-Entwicklung. Es ist vielmehr eine didaktische Software, deren Funktionsumfang und Implementierung sich alleine an diesem Zweck orientiert[1].

Neben den üblichen Logik-Gattern (UND, ODER, ...) gibt es auch viele spezielle Komponenten, wie eine Siebensegmentanzeige, LEDs, Taktgeber u.v.m. Jede Schaltung kann in anderen Schaltungen als Komponente – sozusagen als integrierter Baustein – verwendet werden. Während der Simulation kann man in die Schaltung hinein sehen, um die Vorgänge besser nachzuvollziehen.



Download und weiter Informationen auf

- https://launchpad.net/piiri
- http://beyer.xornet.de/interessen/computer/piiri/

Kontakt: piiri@xornet.de Stefan Beyer – 24. Feb. 2013

1 Zum Beispiel liefert das Programm an sich keine fertigen Komponenten für Flipflops oder Register. Diese können alle aus elementaren Komponenten zusammengebaut werden. Allerdings werden auch Komponenten-Sammlungen zur Verfügung gestellt.

2 Programmoberfläche

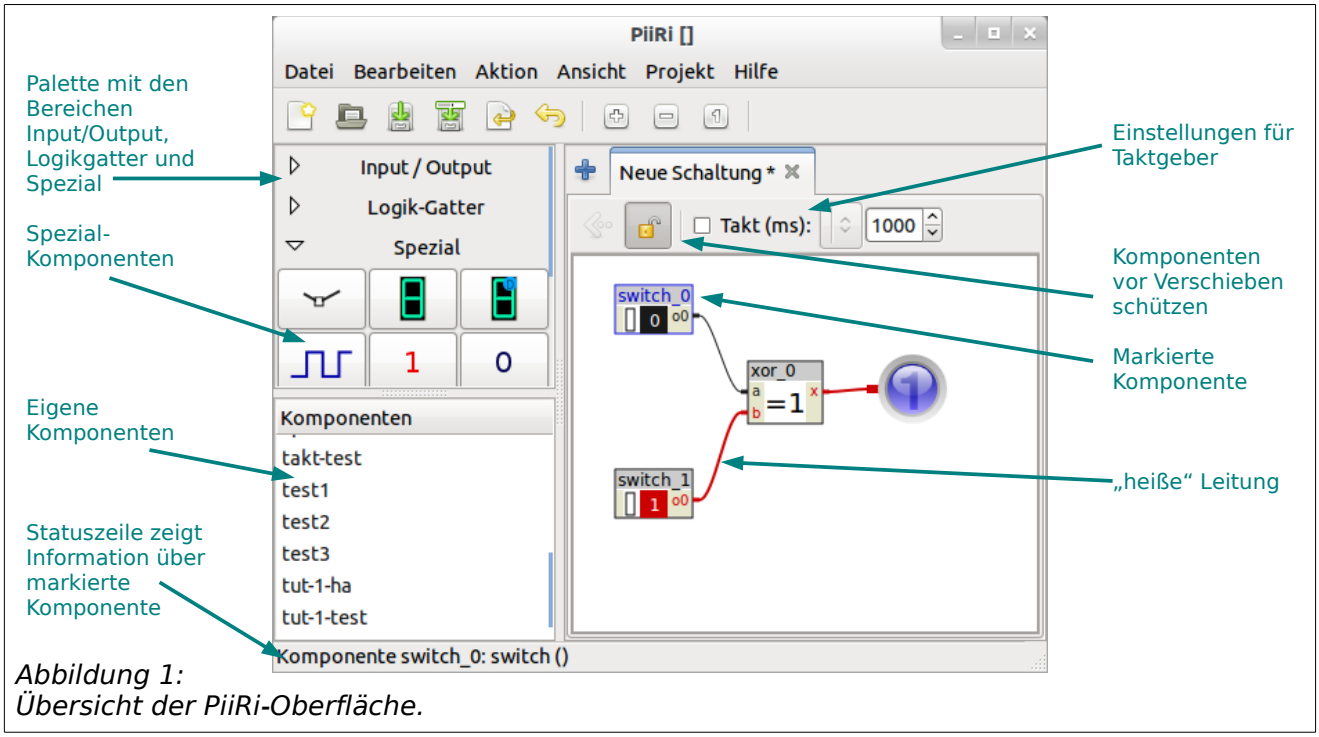


Abbildung 1:
Übersicht der PiiRi-Oberfläche.

Die Programmoberfläche ist im Wesentlichen in zwei Bereiche gegliedert:

- Links die **Palette** mit allen verfügbaren Komponenten.
- Rechts die zum Bearbeiten geöffneten **Schaltungen**, in mehreren Tabs (Reitern).

Das Erstellen, Öffnen und Bearbeiten von Schaltungen kann in zwei verschiedenen Modi erfolgen:

- **projektloser Modus**
  Die erstellten Schaltungen und Komponenten sind nicht einzelnen Projekten zugeordnet. Eignet sich gut zum Ausprobieren. Komponenten aus *Projekten* sind aber nicht ohne Weiteres verwendbar. Der projektlose Modus ist an den leeren Klammern „[]“ im Fenstertitel zu erkennen.
- **Projekt-Modus**
  Für jedes Projekt gibt es einen eigenen Ordner. Alle Komponenten, die in Schaltungen verwendet werden, müssen dort abgelegt sein. Dies eignet sich besonders für das Entwickeln komplexer Schaltungen, deren Komponenten aufeinander abgestimmt sind. Schaltungen *andere Projekte* oder des projektlosen Modus sind nicht ohne Weiteres verwendbar (siehe „Komponenten importieren“). Im Projekt-Modus ist im Fenstertitel der Projektname in „[Klammern]“ abzulesen.

Das Programm startet in der Regel im projektlosen Modus. Über das Projekt-Menü können neue Projekte erstellt und vorhandene geöffnet werden. Durch das Öffnen eines Projektes wird im Wesentlichen nur des Verzeichnis für die Schaltungskomponenten umgeschaltet – ansonsten kann genauso gearbeitet werden, wie im projektlosen Modus.

2.1 Prinzip: Schaltungen, Komponenten, Leitungen

Eine Schaltungen besteht aus Komponenten und evtl. Ein- und Ausgängen, die über Leitungen miteinander verbunden werden. Als Komponenten können fest vorgegebene Komponenten verwendet werden aber auch jede Schaltung kann selbst wieder als Komponente in einer anderen Schaltung verwendet werden.

2.2 Die Palette

Die Palette auf linken Seite ist in vier Bereiche gegliedert:

- **„Input / Output“** (Eingang / Ausgang)
  Klicken Sie auf „in“ bzw. „out“, werden zunächst automatisch durchnummerierte Ein- bzw. Ausgänge für die Schaltung erzeugt. Sie können im Textfeld den vorderen Teil des Namens angeben.
- **„Logik-Gatter“**
  In dieser Rubrik befinden sich die klassischen logischen Komponenten, mit denen ein logisches Schaltnetz aufgebaut werden kann. Diese Komponenten können einfach in die Schaltung gezogen werden (Drag&Drop). Weitere Details im Abschnitt *Komponenten* auf Seite 5.
- **„Spezial“**
  Hier findet man alle anderen vorgegebenen Komponenten. Sie können ebenfalls direkt in die Schaltung gezogen werden. Genaueres hierzu im Abschnitt *Komponenten* auf Seite 5.
- **Komponenten** (eigene)
  Eine Liste der (im Projekt) verfügbaren Schaltungen. Diese können ebenfalls als Komponente in die Schaltung gezogen werden.

2.3 Schaltungen aufbauen

Einmal eingefügt, können Ein- und Ausgänge jederzeit umbenannt werden. Seien Sie dabei aber vorsichtig, wenn die Schaltung bereits wo anders als Komponente verwendet wird!

Zum Testen der Schaltung können die Eingänge der Schaltung mit der Maus über die kleinen Pfeile an- und ausgeschaltet werden. Eine Möglichkeit, die Eingänge über die Tastatur zu steuern finden Sie in Abschnitt *Eingänge schalten* auf Seite 9. Mit der Maus können Ein- und Ausgänge markiert werden. Mit den Tasten ↑ und ↓ lässt man die Markierung wandern. Mit zusätzlicher Shift-Taste kann die Anordnung der Anschlüsse verändert werden.

Neue Anschlüsse haben standardmäßig *eine Leitung*, das heißt sie sind für die Übertragung eines Bits ausgelegt. Über EIGENSCHAFTEN bzw. F3 kann die Anzahl der Leitungen für einen Anschluss eingestellt werden: So können über einen Anschluss beispielsweise 8 Bit auf einmal übertragen werden. Das spart eine Menge Arbeit beim Verbinden der Komponenten.

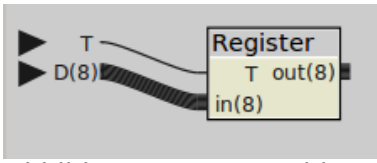


Abbildung 2: Der Anschluss T hat eine Leitung; D hat 8 Leitungen.

Verbindungen können mit der Maus gezogen werden. Hierbei gilt: eine

Verbindung wird von dort, wo das Signal her kommt nach dort, wo das Signal hingeht gezogen. In den meisten Fällen also von links nach rechts.

Es können mehrere Schaltungen zur selben Zeit in mehreren Tabs angezeigt und bearbeitet werden. Wenn die geöffneten Schaltungen voneinander abhängen, so können Sie bei Änderungen in einer Komponente betroffene Schaltungen über den Befehl ZURÜCKSETZEN neu laden (zuvor speichern!).

Im oberen Bereich des Tabs kann zum einen das Bewegen von Komponenten in der Schaltung deaktiviert, zum anderen ein Taktsignal auf einen beliebigen Eingang der Schaltung gelegt werden.

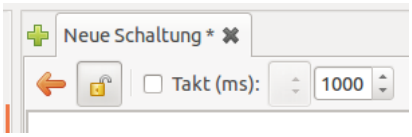


Abbildung 3: Steuerelemente eines Reiters.

3 Komponenten

Eine Übersicht über alle verfügbaren Komponenten ist ab Seite 6 in einer Tabelle zusammengefasst.

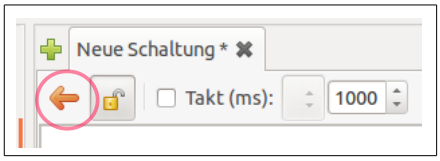
3.1 Details spezieller Komponenten

3.1.1 Eingebettete Schaltung

Wird eine *eingebettete Schaltung* erzeugt, so ist sie zunächst vollkommen leer: keine Anschlüsse, keine Schaltungskomponenten. Zum Bearbeiten markieren Sie die Komponente und drücken Sie F4 oder wählen Sie AKTION / KONTEXTMENÜ → KOMPONENTE BEARBEITEN.

Sie gelangen damit in einen speziellen Modus (*eingebetteter Modus*), in dem Sie den Inhalt der Komponente genauso bearbeiten können, wie eine normale Schaltung: Sie können Ein- und Ausgänge und beliebige weitere Komponenten hinzufügen und verbinden. Auch können Sie wiederum eingebettete Schaltungen verwenden. Denken Sie daran, dass Veränderungen an den Ein- und Ausgängen einer eingebetteten Schaltung sich möglicherweise auf die dort von außen angeschlossenen Verbindungen auswirkt.

Um einen eingebetteten Modus zu verlassen klicken Sie auf den Pfeil im oberen Bereich des Tabs (siehe Abbildung). Haben Sie verschachtelt mehrere eingebettete Modi geöffnet müssen Sie ggf. mehrmals auf diesen Pfeil klicken, um zur Hauptschaltung zurück zu kehren.



3.1.2 Adapter

Um Übergänge zwischen Mehradrigen und Einzelsignalen herzustellen gibt es die Adapter-Komponente. Eingestellt werden muss die Richtung und die Anzahl der zu verarbeitenden Signale.

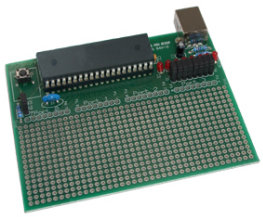
3.1.3 Funktions-Komponente*

!(a&b)

und:	&
oder:	|
nicht:	!
xor:	^

Über die Eigenschaften geben Sie einen logischen Term an und wie viele Eingänge die Komponente haben soll. Es werden dann automatisch Eingänge *x0*, *x1*, ... erzeugt. Ausgang ist immer *y*. Der Term kann aus den Variablen *x0*, *x1*, ..., logischen Operatoren (siehe Kasten rechts) und Klammern bestehen. Beispiel: `x0 | (!x0 & x1)`.

3.1.4 IO-Warrior-Komponente*



Diese Komponente bietet eine Schnittstelle zu einer über USB angeschlossenen *IO-Warrior-40*-Hardware (z.B. einem IOW40-Starterkit) von *Code Mercenaries*. Diese bietet 32 über USB ansteuerbare I/O-Pins: Einzelne Bits bzw. Signale können darüber eingelesen oder ausgegeben und in einer „realen“ Schaltung weiterverarbeitet werden.

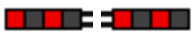
Mehr Infos: http://www.codemercs.com/io-warrior/?L=0.

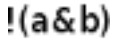
Diese Komponente ist derzeit noch in der Entwicklung; besonders unter Windows wurde sie noch nicht ausreichend getestet. Unterstützt wird derzeit nur der IO-Warrior 40. Die Komponente ist noch auf jeweils 8 Ein- und Ausgänge beschränkt: An den Ausgängen sind die Eingangssignale des Port 0 verfügbar. Die Eingänge werden an die Output-Pins des Port 3 geschrieben. Zukünftig soll hier – über die Eigenschaften einstellbar – mehr Flexibilität möglich sein und alle 32 I/O-Pins des Chips unterstützt werden.

* Diese Komponenten werden zunächst nicht in der Palette angezeigt. Sie müssen erst über die Einstellungen aktiviert werden.

3.2 Übersicht der Komponenten

Tabelle beginnt auf der nächsten Seite.

Name / Symbol	Beschreibung	Eingänge	Ausgänge	Eigenschaften
Logik-Gatter 	Die jeweilige logischen Operationen aus den Eingängen wird auf den Ausgang geschaltet. Wenn Sie wollen können Sie natürlich auch alle Gatter aus NAND- oder NOR-Gattern aufbauen.	a und b, bei „not“ nur a	x	**- Beschreibung** (zum Beispiel kurze Beschreibung, welche Aufgabe die Komponente in der Schaltung übernimmt) **- Weiterhin sichtbar** (die Komponente ist auch sichtbar, wenn die Schaltung in einer anderen Schaltung als Komponente verwendet wird)
Ecke 	Als Eckknoten für Leitungen und als Verteiler verwendbar.	Ein unsichtbarer Eingang	Ein unsichtbarer Ausgang	- Beschreibung - Weiterhin sichtbar
7-Segment-Anzeigen 	Eine typische 7-Segment-Anzeige. Einmal mit und einmal ohne 4-Bit-Decoder verfügbar. Die Version mit Decoder stellt eine 4-Bit-Zahl entsprechend als Hexadezimalziffer dar.	0-3 bzw. 0-7	-	- Beschreibung - Weiterhin sichtbar
Taktgeber 	Schaltet in einem bestimmten Zeitintervall den Ausgang abwechselnd ein und aus.	-	o, das Taktsignal	- Beschreibung - Weiterhin sichtbar **- Intervall** (in ms) **- Aktiv** (Taktgeber ist aktiv)
Eins / Null 	Liefert konstant eine logische Null oder eine logische Eins. Eine Null ist normalerweise nicht notwendig, weil ein offener Eingang eine Null darstellt.	-	o, für das konstante Signal	- Beschreibung - Weiterhin sichtbar
Bit-Schalter / Bit-Anzeige 	Eine flexible Reihe von Schaltern, bzw. Bit-Anzeigen. Die Anzahl der Felder kann eingestellt werden. Damit kann z.B. Kompakt ein Byte Dargestellt werden. In der horizontalen Darstellung wird die Anzeige um eine Hexadezimalzahl und eine Dezimalzahl ergänzt.	keine, bzw. i0, i1 … kann eingestellt werden.	Keine bzw. o0, o1 … kann eingestellt werden.	- Beschreibung - Weiterhin sichtbar **- Breite** (Anzahl der Ein- bzw. Ausgänge) **- Vertikale Anzeige** (Anordnung der Schalter bzw. Anzeigefelder)
LED 	Hiermit wird eine LED-Lampe nachgeahmt. Sie zeigt durch „leuchten“ an, ob das anliegende Signal „an“ ist.	i	-	- Beschreibung - Weiterhin sichtbar **- Farbe** (Farbe der „leuchtenden“ LED)
Text 	Mit dieser Komponente kann einfacher Text in der Schaltung platziert werden. Für die Schaltung selbst hat sie keine Bedeutung.	-	-	- Beschreibung - Weiterhin sichtbar **- Text** (anzuzeigender Text) **- Schriftgröße** ()
Eingebettete Schaltung 	Diese Komponente kann eine beliebige Schaltung enthalten. Der Inhalt wird aber nicht als externe Datei sondern mit der Schaltung zusammen gespeichert. Siehe „Details zu Komponenten“.	Bel. Hängt vom Inhalt der Schaltung ab.	Bel. Hängt vom Inhalt der Schaltung ab.	- Beschreibung - Weiterhin sichtbar

Name / Symbol	Beschreibung	Eingänge	Ausgänge	Eigenschaften
Funktions-Komponente* 	Über die Einstellungen kann ein logischer Term angegeben werden, mit dem der Ausgang aus den Eingangssignalen „berechnet“ wird. Siehe auch „Details zu Komponenten“.	x0, x1, ... Anzahl kann eingestellt werden.	y	- Beschreibung - Weiterhin sichtbar **- Eingangsbreite** (Anzahl Eingänge) **- Funktionsterm** (logischer Term zur Berechnung des Ausgangs)
RAM-Komponente* 	Diese Komponente simuliert einen einfachen Speicherbaustein. Eine solche Komponente kann selbstverständlich auch manuell zusammengebaut werden, allerdings ist das sehr Aufwändig und wäre dann nicht so flexibel konfigurierbar, wie es diese Komponente ist.	A0... Adresse in0... Eingangswert. S Schreiben bei negativer Taktflanke	out0... Hier liegt der Wert der Speicherzelle, die über den Adresseingang angesteuert wird	- Beschreibung - Weiterhin sichtbar **- Adressbits** (Anzahl der bits für die Adresse) **- Inhalt** (Inhalt des Speichers als Hexadezimalzahlen durch Leerzeichen getrennt) **- Inhalt-Bits** (Anzahl der Bits für die Speicherzellen) **- Zurückschreiben** (Änderungen im Speicher sollen dauerhaft gespeichert werden)
IO-Warrior-Komponente* 	Eine Schnittstelle mit dem IO-Warrior 40 von Code Mercendaries. Siehe auch „Details zu Komponenten“.	in0 – in7 An den IOW zu sendende Signale.	out0 – out7 Vom IOW empfangene Signale.	- Beschreibung - Weiterhin sichtbar **- Modus** (Es kann mehrere lesende Komponenten geben, aber nur eine schreibende.)

* Diese Komponenten werden zunächst nicht in der Palette angezeigt. Sie müssen erst über die Einstellungen aktiviert werden.

4 Weitere Funktionen

4.1.1 Komponente ansehen

Menü Aktion / Kontextmenü → Anzeigen Doppelklick auf die Komponente

Durch Doppelklick auf eine Komponente wird ein Anzeigefenster geöffnet, in dem man das Innenleben der Komponente „live“ verfolgen kann. Über das Anzeigefenster können Sie auch weiter in die Schaltung „hinabtauchen“.

Dies ist natürlich nur bei Komponenten möglich, die auch ein „echtes Innenleben“ haben; also Gatter und andere feste Komponenten können so nicht geöffnet werden.

4.1.2 Schaltung bearbeiten

Menü Aktion / Kontextmenü → Komponente bearbeiten F4

Die einer Komponente zugrunde liegende Schaltung kann in einem neuen Tab zum Bearbeiten geöffnet werden. Eine Ausnahme bilden hier die sogenannten „eingebetteten Schaltungen“, die im selben Tab zum Bearbeiten geöffnet werden, weil Sie in der aktuellen Schaltung eingebettet sind und nicht auf eine andere Datei verweisen (für Details siehe Abschnitt *Eingebettete Schaltung* auf Seite 5).

In der Liste der eigenen Komponenten (Palette) können die aktuell verfügbaren Schaltungen per Doppelklick zum Bearbeiten geöffnet werden. Hier können Sie

Schaltungen auch umbenennen oder ganz entfernen.

Beachten Sie aber beim Verändern von Schaltungen, dass evtl. andere Schaltungen diese verwenden und ggf. dann nicht mehr richtig funktionieren.

4.1.3 Mit Texteditor öffnen

Menü Aktion → Mit Texteditor öffnen

Über diese Funktion wird die XML-Datei der Schaltung mit einem bestimmten Programm geöffnet. Das zu verwendende Programm kann in den Programm-Einstellungen angegeben werden.

4.1.4 Wertetabelle

Menü Aktion → Wertetabelle

Eine Wertetabelle kann für Schaltungen mit einfachen Schaltnetzen angezeigt werden. Bei Komponenten mit Rückkopplungen macht eine Wertetabelle keinen Sinn.

4.1.5 Impulsdiagramm

Menü Aktion → Impulsdiagramm

Für eine Schaltung kann ein Impulsdiagramm erstellt werden, indem man über einen bestimmten Zeitraum hinweg die Änderungen an den Ein- und Ausgängen aufzeichnen lässt. Das Diagramm kann als png-Grafik abgespeichert werden oder man speichert die Rohdaten als Textdatei. Auch ein experimenteller CSV-Export (Comma separated values) ist verfügbar.

4.1.6 Komponenten importieren

Menü Projekt → Komponenten importieren

Um in einem Projekt eine Schaltung eines anderen Projektes verwenden zu können, müssen Sie diese erst in das Projekt importieren. Dazu können Sie im Datei-Browser die zugehörigen .cir-Dateien entsprechend kopieren, oder Sie versuchen es mit dieser Funktion: Wenn Sie hier eine Schaltung (.cir-Datei) auswählen, so versucht das Programm alle zusätzlich benötigen (also die von der zu importierenden Schaltung als Komponenten verwendeten) Schaltungen zu ermitteln und ebenfalls zu importieren, damit alle Abhängigkeiten erfüllt sind. Diese Funktion hat experimentellen Charakter.

4.1.7 Eingänge schalten

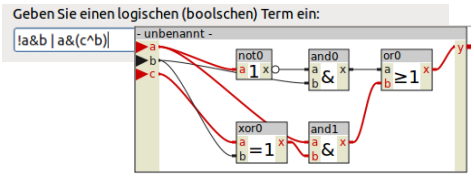
Über die kleinen Pfeile an den Eingängen können die Eingänge der Schaltung mit der Maus ein- und ausgeschaltet werden.

Auch über spezielle Tastaturkürzel können Eingänge geschaltet werden: Wird eine Buchstabentaste gedrückt, so wird der erste Eingang, dessen Namen mit diesem Buchstaben beginnt (egal ob GROSS oder klein) auf 1 gesetzt. Der Eingang bleibt solange auf 1, bis die Taste losgelassen wird. Mit dieser Funktion kann ein Eingang also wie mit einem Taster gesteuert werden.

Drückt man zusätzlich die Shift-Taste, so bleibt der Eingang eingeschaltet auch wenn man die Taste loslässt. Erst durch erneutes Drücken der Taste wird er dann wieder ausgeschaltet.

Durchnummerierte Eingänge (zum Beispiel x0, x1, x2, ...) können durch eine spezielle – zugegebenermaßen etwas umständliche – Tastenkombination gesteuert werden: <Ziffer>+<Buchstabe>, wobei die Ziffer auf der alphanumerischen Tastatur, nicht dem Ziffernblock gemeint ist. Beispiel: der Eingang „x5“ kann durch die Tastenkombination [5]+[x] geschaltet werden. Beachten Sie, dass die Ziffer zuerst gedrückt werden muss. Die Shift-Taste kann auch hier verwendet werden, um den Eingang eingeschaltet zu lassen.

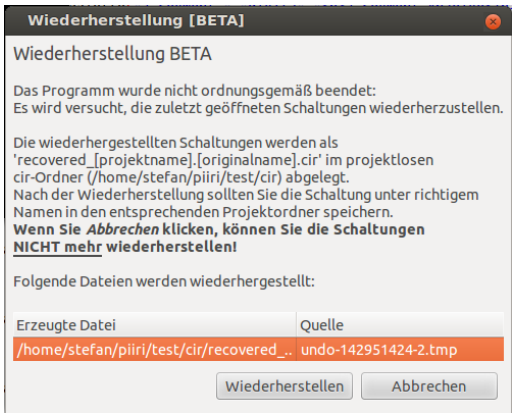
4.1.8 Schaltungs-Synthese



MENÜ AKTION → SCHALTUNS-SYNTHESE

Die Schalungs-Synthese kann aus einem logischen Term (wie bei einer Funktions-Komponente) eine konkrete Schaltung aus automatisch platziert und verschalteten Gattern erzeugen. Diese Funktion hat experimentellen Charakter.

4.1.9 Wiederherstellung



Programmabstürze können leider nicht ausgeschlossen werden. Ein ungewolltes Programmende wird beim nächsten Start erkannt und der Wiederherstellungs-mechanismus wird gestartet. Die zuvor bearbeiteten Schaltung können aus automatisch angefertigten Sicherungsdateien wiederhergestellt werden.

5 Hinweise und Tipps

5.1.1 Schaltungen speichern

Verwenden Sie zum Speichern Ihrer Schaltungen nur den jeweiligen „cir“-Ordner des geöffneten Projektes bzw. den „cir“-Ordner des projektlosen Modus, falls Sie kein Projekt geöffnet haben. Andernfalls können die Schaltungen vom Programm nicht aufgefunden werden. Der richtige Ordner sollte im „Speichern“-Dialog i.d.R. direkt geöffnet sein.

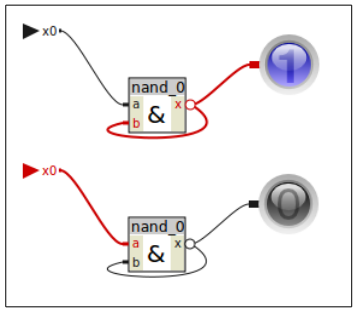
5.1.2 ⚠ Fehlermeldungen

„Zeitlimit überschritten!“

Diese Meldung erscheint, wenn eine Schaltung durch Rückkopplungen immer weiter simuliert werden muss, ohne dass sich ein stabiler Zustand einstellt. In einer echten Schaltung würde das einem schnellen Flackern entsprechen. In

der experimentellen *Step-Simulation* können solche Schalungen auch simuliert werden. Hier ist das Flackern dann zu sehen.

Beispiel: Diese Schaltung mit einem einfach rückgekoppelten NAND-Gatter hat einen stabilen Zustand, wenn x0 aus bleibt. Sobald x0 eingeschaltet wird, sind beide Eingänge des NAND-Gatters an und der Ausgang damit aus. Damit geht aber der Eingang b aus und der Ausgang wieder an usw...



6 Einstiegstutorial

6.1 Eine erste Schaltung

Öffne die Schaltung `tut-1-test.cir`. Die angezeigte Schaltung ist nur ein Plan als Bild und soll nun von dir gebaut werden.

a)

- Erstelle die linke Schaltung, indem du die Komponenten hinzufügst und sie wie abgebildet verbindest. (Die meisten der benötigten Komponenten findest du im Bereich „Spezial“.)
- Markiere die Komponente `xor0` und wähle aus dem Menü AKTION → WERTETABELLE, um eine Wertetabelle der Komponente anzuzeigen.
- Durch Anklicken der schwarzen Flächen auf den Schalter-Komponenten lassen sie sich an- und ausschalten.
- Probiere verschiedene Kombinationen und vergleiche mit der Wertetabelle.

b)

- Erstelle die rechte Schaltung. Die lange Schalter-Komponente bekommst du, wenn du bei einer Schalter-Komponente über die Eigenschaften „Breite“ auf 8 stellst. Achte darauf, dass du die richtige Segmentanzeige verwendest. Verbinde dann die Schalter mit den zwei Segmentanzeigen.
- Stelle die Schalter-Komponente über die Eigenschaften auf horizontal um und verschiebe die Komponenten so, dass man alles gut sehen kann.
- Teste. Vergleiche die in der Schalte-Komponente angezeigten Zahlen mit der Anzeige auf den beiden 7-Segment-Anzeigen.
- Positioniere die 7-Segment-Anzeigen so, dass die Hexadezimalzahl richtig angezeigt wird.

6.2 Einen Addierer bauen

Öffne die Schaltung `tut-1-ha.cir`.

a)

- Füge 2 Eingänge (a und b) und 2 Ausgänge (s und ü) hinzu.
- Füge die Komponenten hinzu und verbinde sie wie abgebildet.
- Diese Schaltung rechnet:
  $0_2 + 0_2 = 00_2 = 0$
  $1_2 + 0_2 = 01_2 = 1$
  $0_2 + 1_2 = 01_2 = 1$
  $1_2 + 1_2 = 10_2 = 2$ (die 1 ist dabei der Übertrag)
- Man nennt sie Halb-Addierer.
- …